International Journal of AI, BigData, Computational and Management Studies
Noble Scholar Research Group | Volume 7, Issue 1, 162-169, 2026
ISSN: 3050-9416 | https://doi.org/10.63282/3050-9416.IJAIBDCMS-V7I1P125

Original Article

Data Quality Contracts for Multi-Cloud Healthcare: Semantic SLAs with Automated Remediation

Sai Kiran Yadav Battula
Independent Researcher Pittsburgh, Pennsylvania, United States.

Received On: 03/01/2026 ***Revised On: 07/02/2026*** ***Accepted On: 13/02/2026*** ***Published On: 25/02/2026***

Abstract: *Multi-cloud healthcare platforms distribute ingestion, transformation, and analytics across Amazon Web Services, Microsoft Azure, and Google Cloud Platform to improve elasticity, regional resilience, and regulatory alignment. In practice, these benefits are frequently undermined by silent data-quality failures schema drift, terminology misalignment, freshness regressions, and broken references that propagate across heterogeneous pipeline components and destabilize clinical analytics and downstream AI systems. Existing approaches remain largely reactive: static checks and ad hoc expectations detect issues after propagation, while cloud provider SLAs emphasize infrastructure availability rather than semantic correctness or temporal fitness-for-use. This paper proposes Data Quality Contracts (DQCs) as semantic SLAs operationalized as SLOs with error budgets for multi-cloud healthcare data products. A DQC is a versioned, machine-readable agreement between producers and consumers specifying: (i) FHIR-aware HL7 semantic constraints (profiles, cardinalities, terminology binding), (ii) temporal objectives (freshness and end-to-end latency), (iii) completeness and referential integrity targets, and (iv) error budgets that bound acceptable quality loss and trigger escalation. We introduce a cloud-agnostic Quality-as-a-Service (QaaS) control plane that manages contract lifecycle and error-budget accounting while enforcing contracts federatedly at ingestion boundaries to minimize cross-cloud protected health information movement. When violations occur, the control plane orchestrates policy-driven remediation workflows schema reconciliation, terminology synchronization, prioritized replay, cache invalidation, and referential repair with approval gates for high-severity or low-confidence actions. We evaluate a prototype using a synthetic FHIR-compatible workload of 750,000 patients generated with Synthea across three clouds over 45 simulated days, with data-quality chaos injections spanning schema drift, vocabulary misalignment, freshness degradation, completeness regressions, and referential integrity faults. Relative to three baselines (reactive monitoring, expectations-based validation, and FHIR validator workflows), DQCs reduce mean time to remediation from 168.6 to 10.2 minutes ($p<0.001$), reduce false-positive alert rate from 26.6% to 14.3% ($p<0.001$), maintain 93.8% aggregate contract compliance during chaos, and automatically resolve 82% of violations. A 12-month operational cost model (assumptions in Section VIII) indicates reduced incident-driven toil and downtime exposure versus reactive approaches. These results suggest that contract-driven, semantics-aware governance with error budgets and automated remediation provides a practical foundation for trustworthy multi-cloud healthcare analytics and AI.*

Keywords: *Data Quality Contracts, Semantic Slas, Multi-Cloud Healthcare, Automated Remediation, Fhir, Data Governance, Data Observability, Error Budgets, Chaos Engineering, Quality-As-A-Service.*

1. Introduction

Healthcare organizations increasingly operate distributed data ecosystems that combine EHR extracts, HL7 v2 feeds, claims, pharmacy, imaging metadata, IoMT telemetry, and SDOH sources into cloud-native data platforms. Multi-cloud adoption is driven by elasticity, vendor diversification, business continuity requirements, and region-specific data residency constraints. However, multi-cloud also multiplies data-quality failure surfaces: schema evolution in upstream systems, drift in terminology artifacts, asynchronous replication lag, heterogeneous serialization formats, and platform-specific behaviors in orchestration.

In healthcare, data-quality failures are not merely operational inconveniences. A pipeline may satisfy infrastructure availability while delivering clinically invalid data (e.g., mis-bound codes, missing required elements, broken references), yielding biased risk stratification, incorrect care-gap identification, or unstable clinical dashboards. Downstream AI systems can be particularly brittle to subtle data defects: stale observations, changed units, or distribution shifts can materially affect model calibration and fairness. Under the ONC's HTI-1 final rule, transparency and safety of predictive decision-support interventions are emphasized, making undetected quality failures both a clinical risk and a compliance liability.

1.1. Why Reactive DQM Breaks in Multi-Cloud Healthcare

Traditional healthcare data-quality management often assumes (i) centralized inspection, (ii) stable schemas, and (iii) manual remediation. These assumptions fail in multi-

cloud settings. Centralizing PHI for validation can be cost-prohibitive due to cross-cloud egress and constrained by residency policies. Schema and vocabulary change continuously as EHR platforms upgrade and IoMT vendors revise firmware. Manual remediation does not scale to hundreds of pipelines and distributed ownership boundaries. In addition, common observability tooling focuses on volume/freshness metrics and stops at alerting; it rarely encodes clinical semantics and does not coordinate remediation across clouds.

1.2. Semantic SLAs: Closing the Infrastructure Data Gap

Cloud SLAs typically guarantee uptime, durability, and service response behavior. They do not guarantee that healthcare records are semantically valid, complete, fresh, and fit-for-use. What is missing is an operational construct that binds semantic correctness and timeliness to enforceable objectives semantic SLAs operationalized as measurable SLOs with error budgets for healthcare data products.

This paper proposes Data Quality Contracts (DQCs) as semantic SLAs: explicit agreements between data producers and consumers that define measurable indicators and enforceable policies for healthcare data quality.

1.3. Contributions

This work makes five contributions:

- Contract model for semantic SLAs in healthcare data products. We define Data Quality Contracts (DQCs) as versioned, machine-readable agreements encoding FHIR-aware semantic constraints, temporal objectives, completeness and referential integrity targets, and error budgets that operationalize acceptable quality loss and escalation behavior.
- Federated enforcement architecture for multi-cloud pipelines. We present a cloud-agnostic QaaS design that separates a centralized governance plane (registry, policy evaluation, error-budget accounting, audit/lineage) from a federated enforcement plane (validators deployed at ingestion boundaries). The design validates locally and exports only contract-relevant signals and violation artifacts needed for governance.
- Graded semantic severity to reduce alert fatigue under benign drift. We introduce a semantic distance formulation decomposing deviation into structural, vocabulary, and cardinality components, mapped to severity tiers combined with consumer criticality and temporal deviation.
- Policy-driven automated remediation with safety guardrails. We define remediation policies that map violation classes to workflows (schema reconciliation, terminology synchronization, prioritized replay, cache invalidation, referential repair), with explicit approval gates for high-severity or low-confidence actions and rollback on post-remediation validation failure.
- Chaos-based evaluation methodology for healthcare data-quality failures. We provide a data-quality chaos engineering protocol that injects realistic failure modes across controlled experiments, enabling statistical comparison against representative baselines under matched monitoring visibility.
- Operational cost model under stated assumptions. We present a 12-month cost analysis separating direct enforcement costs from incident-driven toil and downtime exposure to clarify when contract-driven remediation is economically favorable.

2. Related Work

Healthcare DQM research emphasizes completeness, validity, consistency, timeliness, uniqueness, and accuracy, with additional focus on semantic validity and fitness-for-use. In production, quality checks are frequently implemented as ad-hoc rules embedded in pipelines. Modern data observability platforms provide automated profiling and anomaly alerts, but typically lack deep clinical semantics and cross-cloud remediation coordination.

HL7 FHIR provides semantic structure through profiles, cardinalities, slicing, and terminology binding. FHIR validators and SDKs can check conformance, but they are commonly used in batch workflows with binary outcomes, and they do not operationalize graded severity, error budgets, or automated remediation beyond alerting.

Data contracts in software engineering formalize producer-consumer expectations for structural consistency and versioning. However, generic data contracts are usually schema-centric and insufficient for healthcare's terminology binding strength, profile complexity, and temporal fitness requirements.

Self-healing systems and SRE practices propose automated retries, circuit breakers, and failover, but these address service reliability rather than semantic data defects that require transformations, terminology synchronization, or reference repair. Chaos engineering validates resilience by injecting faults; this work adapts chaos principles to data-quality faults specific to healthcare pipelines.

3. Data Quality Contract Formalization

3.1. Contract Concepts: SLI, SLO, Error Budget

A DQC operationalizes semantic SLAs by adapting SRE principles to data quality:

- Service-Level Indicators (SLIs): measurable signals of data quality, such as profile conformance rate, terminology binding rate, p95 staleness, missingness rates for critical fields, and reference resolution rate.
- Service-Level Objectives (SLOs): target thresholds for SLIs (e.g., "p95 staleness $\leq$ 6 hours," "conformance rate $\geq$ 98%").
- Error Budgets: allowable violation rates within a time window; if exhausted, escalation occurs (e.g., block publish, page on-call, require producer rollback). This framing provides a direct operational

mechanism for governance and escalation rather than passive monitoring.

3.2. DQC Specification Model

Each DQC includes:

- Metadata: contract ID, producers/consumers, Sem Ver, rollout policy (canary/rollback), deprecation windows, approval workflow.
- FHIR semantics: required profile (Structure Definition), must-support elements, cardinalities/slicing, terminology binding rules (Value Set URIs, binding strength).
- Temporal guarantees: freshness objectives (staleness percentiles), end-to-end latency objectives, watermark expectations for streaming.
- Completeness targets: null-rate limits for critical fields, minimum coverage targets, volume expectations per window, and join completeness.
- Referential integrity targets: required resolution rates for references.
- Error budgets: violation caps per window and burn-rate thresholds triggering escalation.
- Remediation policy: mapping from violation types to actions; severity gates and human approval requirements.

3.3. Concrete Contract Example

Listing 1 show a compact DQC for an Observation data product aligned to a US Core lab profile.

```
id: dqc-uscore-observation-lab
version: 1.0.0
producer: ehr-ingest-service
consumer: risk-scoring-pipeline
rollout:
  mode: canary
  canaryPercent: 10
  rollbackOn: errorBudgetExhausted
fhir:
  profile: http://hl7.org/fhir/us/core/StructureDefinition/us-core-observation-lab
  mustSupport: [status, code, subject, effectiveDateTime, valueQuantity]
  terminology:
    code:
      bindingStrength: required
      valueSet: http://hl7.org/fhir/ValueSet/observation-codes
temporal:
  freshness:
    p95StalenessHours: 6
  latency:
    p99EndToEndSeconds: 900
completeness:
  criticalFieldMaxNullRate: 0.01
  volume:
    perHourMinRecords: 500
references:
  subject:
    targetResourceType: Patient
    mustResolveRate: 0.98
errorBudget:
  windowMinutes: 60
  maxViolationRate: 0.02
  escalationPolicy: block_publish_and_page
remediation:
  schemaDrift:
    actions: [reconcile_schema, validate_transformed, replay_window]
  freshnessBreach:
    actions: [priority_reingest, invalidate_downstream_cache]
  refIntegrity:
    actions: [repair_refs_if_confident, quarantine_otherwise]
    confidenceThreshold: 0.85
  requireHumanApprovalIf:
```

```
severityAtLeast:                                                    high
semanticDistanceAtLeast: 0.8
```

Fig 1: Abbreviated DQC Specification (YAML) for US Core Observation Lab Results.

3.4. Graded Severity via Semantic Distance

Binary pass/fail validation produces alert fatigue when schemas evolve. We compute a semantic distance score that grades deviation:

- Structural deviation (d_{struct}): Weighted fraction of missing required elements versus present prohibited elements.
- Vocabulary deviation (d_{vocab}): For required bindings, out-of-ValueSet codes contribute maximal distance; extensible bindings are graded by semantic similarity in the hierarchy.
- Cardinality deviation (d_{card}): Magnitude of deviation from expected min/max counts.

Aggregate semantic distance:

$$d_{sem} = w_s \cdot d_{struct} + w_v \cdot d_{vocab} + w_c \cdot d_{card}$$

Weights (w) are empirically tuned. Severity policy combines d_{sem} with temporal deviation and consumer criticality to determine action (e.g., warn, auto-remediate, or block and page).

4. Qaas Control Plane and Federated Enforcement

4.1. Design Goals and Threat Model

The Quality-as-a-Service (QaaS) architecture: (i) enforces Data Quality Contracts (DQCs) close to ingress to minimize cross-cloud PHI movement and egress cost; (ii) centralizes governance over contract versions, policies, and error budgets; (iii) supports batch and streaming pipelines; and (iv) enables auditable, reversible remediation. The threat model assumes a zero-trust posture: agents and workers are treated as potentially compromised, so interactions require strong identity, least privilege, signed decisions, and tamper-evident audit logs. The control plane stores no raw PHI by default to reduce blast radius

4.2. Architecture Overview and Deployment Topology

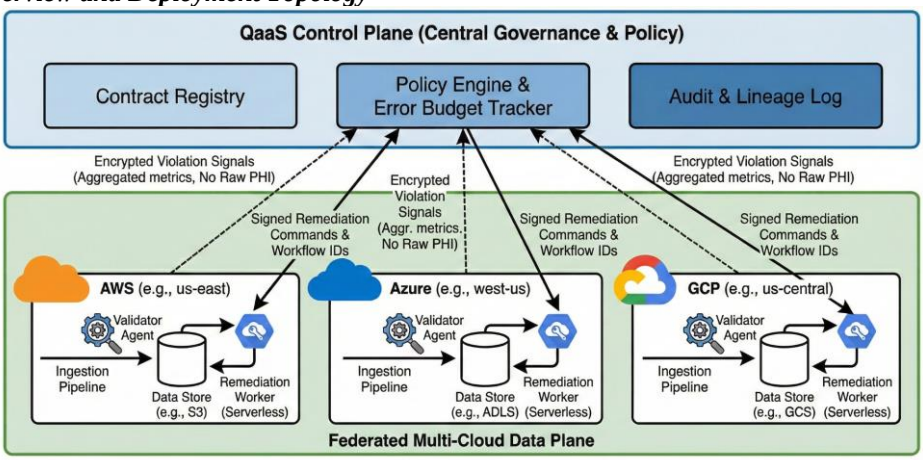

Fig 2: The split-plane Quality-as-a-Service (QaaS) architecture

A centralized control plane governs policies, error budgets, and audit logs without accessing raw data. Federated agents enforce contracts at the ingestion boundaries of each cloud (AWS, Azure, GCP) to minimize cross-cloud PHI movement and egress costs.

- Control Plane (central governance): Contract Registry (signed/versioned DQCs + rollout), Policy Engine (classification + action + gating), Error Budget Tracker (SLOs + burn rate), Remediation Dispatcher (commands to local executors), and Audit/Lineage Log (append-only trace). It stores contract metadata and aggregated SLIs, decisions, and outcomes; PHI is excluded unless explicitly allowed by policy.
- Federated Data Plane (per cloud/region): Validator Agents deployed near ingress (VPC/VNet) and Remediation Workers (serverless/jobs) with scoped privileges to reprocess data within the same cloud/region.

4.3. Identity, Authorization, and Multi-Tenancy

Validator Agents and Remediation Workers authenticate using short-lived workload credentials and mTLS; authorization is enforced via least privilege and policy-as-code (per contract, action, and dataset/partition). Decisions are signed, short-lived (TTL), and verified before execution to prevent confused-deputy and replay. Multi-tenancy isolates contracts, SLIs, and audit logs by tenant and environment (dev/test/prod), with explicit approval for cross-environment promotion. Deterministic enforcement: agents compute SLIs and semantic distance locally; the control plane evaluates violations using a pinned contract version and policy bundle and returns a signed Decision referencing the policy version used.

4.4. Contract Lifecycle and Versioning Workflow

DQCs use Sem Ver and staged rollout: author/review → publish (effective date + deprecation window + mode: shadow/canary/hard) → distribute (agent cache + window pinning) → rollback (revert contract and/or force monitor-only on rapid burn or critical violations).

4.5. Federated Enforcement at Ingestion Boundaries

Agents enforce DQCs at message queues, landing zones, CDC streams, or API gateways using tiered validation: (1) structural + FHIR profile conformance (must-support, slicing, cardinality), (2) terminology binding per strength (required vs. extensible; graded similarity per Section III-D), (3) temporal SLIs (freshness/latency percentiles), and (4) completeness + referential integrity SLIs (null-rate and reference-resolution via local indices/joins). Sampling/tiering are contract-configurable (cheap checks on all records; deeper checks on calibrated subsets), with SLI estimation adjusted to preserve error-budget fidelity.

4.6. Terminology Distribution, Caching, and Consistency

Agents maintain a local Value Set cache keyed by (URI, version/effective date) with contract-defined refresh and fallback. If the authoritative endpoint is unavailable, agents validate against cache and emit terminology-cache-staleness as an auxiliary SLI; validation results record resolution mode (exact vs. graded) for auditability and drift-tolerant alerting.

4.7. Violation Event Model and Decision Interface

On violation, agents emit a Violation Event containing: (contract ID/version, product, producer/consumer), (event-time/processing-time window), SLIs, severity, semantic distance, and top contributors, plus minimal artifacts permitted by policy (typically aggregated metrics and hashed IDs). PHI default-deny: events exclude direct identifiers and free-text fields by default; representative samples require an explicit allowlist (fields + max sample size) in the contract policy. The Policy Engine returns a signed Decision: action ∈ {ignore, warn, ticket, auto-remediate, quarantine, block publish, page_on_call}, optional workflow ID, gating (approval/confidence), and escalation metadata (budget state/burn/criticality), plus TTL/signature. Idempotency: events include an eventide for de-duplication; Decisions include an idempotency key so workers can safely retry without repeated replays, quarantines, or blocks.

4.8. Error Budget Accounting and Escalation Policy

For each contract, the tracker maintains rolling windows (e.g., 60 min/24 hr), violation rate and burn rate, and multi-SLI composition (weighted conformance/freshness/completeness/reference resolution). Escalation triggers on budget exhaustion, sustained burn (e.g., 2× for 15 min), or high-severity breaches for critical consumers, resulting in quarantine, block publish, paging, or rollback (Section IV-D).

4.9. Remediation Dispatch, Locality Constraints, Reliability, and Auditability

- Locality: remediation executes in the same cloud/region by default; cross-cloud PHI movement is disallowed unless explicitly permitted and approval-gated; control-plane returns are non-PHI summaries/audit metadata. Fail-safe: if the control plane is unavailable, agents enforce cached/pinned contracts, queue events, and restrict actions to safe local behavior (e.g., quarantine/monitor-only); backpressure prioritizes lightweight SLIs and uses calibrated sampling for deeper checks. Audit/lineage: decisions and remediation outcomes are append-only (contract version, triggering SLIs, decision, workflow IDs, input/output partitions, post-validation), linking remediated outputs to downstream publishes.

4.10. Implementation Overhead and Operational Footprint

Agents report p50/p95 validation latency, incremental ingestion latency, event throughput, and CPU/memory under representative load; the control plane reports per-contract SLI-series growth and decision/audit log growth. Enforcement is non-blocking by default; when block publish is enabled, blocking occurs at the publish gate (data product release) rather than raw ingest to avoid amplifying upstream backpressure.

5. Automated Remediation with Human-In-The-Loop Safeguards

5.1. Violation Taxonomy

We classify violations into five primary categories: schema drift, vocabulary drift, freshness/latency breaches, completeness regressions, and referential integrity failures.

5.2. Remediation Strategies

Policy-driven actions are mapped to violation categories, including:

- Schema reconciliation: Apply validated transformations from a repository to align data to the contract profile.
- Terminology synchronization: Refresh ValueSets from authoritative sources or translate deprecated codes where binding strength allows.
- Prioritized replay: Replay affected time windows with elevated priority and invalidate downstream caches.
- Referential repair: Use probabilistic matching to repair broken references within confidence thresholds.
- Quarantine: Isolate offending data and route to steward review.

5.3. Durable Workflow Orchestration

Remediations execute as directed acyclic graphs (DAGs) of tasks orchestrated by Temporal.io, chosen for durable execution guarantees. Workflows include automatic retries, timeouts, deterministic state management, and rollback capabilities if post-remediation validation fails.

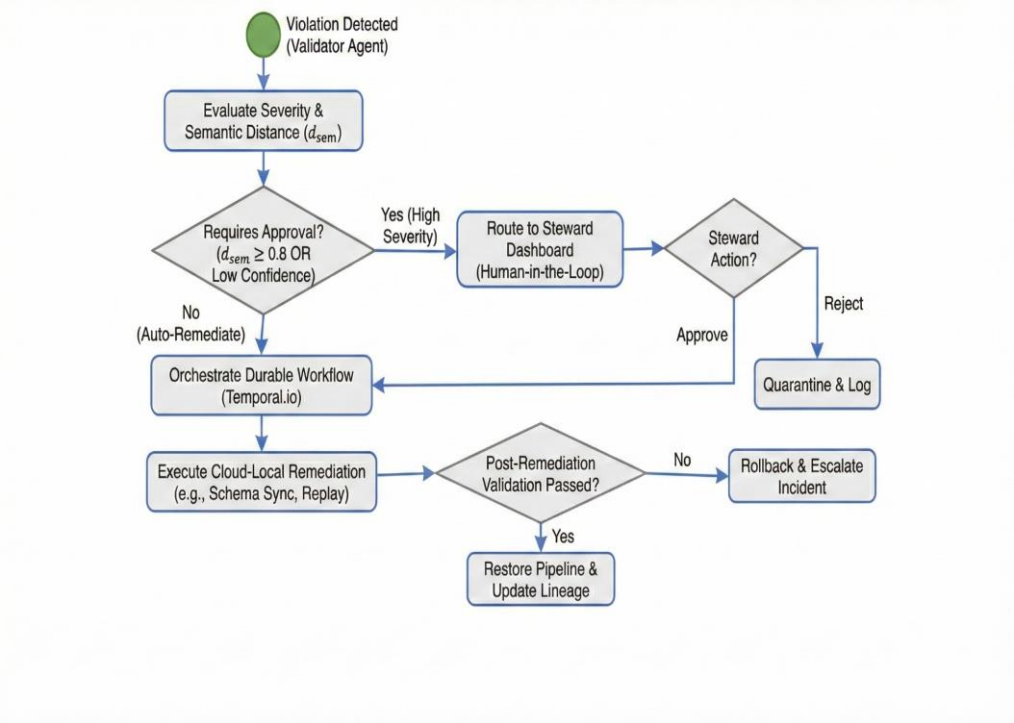

Fig 3: Policy-driven automated remediation workflow with safety guardrails.

Violations are scored by semantic distance. Low-severity issues trigger automated self-healing workflows. High-severity or ambiguous violations are routed to a human steward for approval before execution, ensuring safety for clinical data.

5.4. Human-in-the-Loop Approval Gates

For high severity ($d_{sem} \geq 0.8$), low confidence strategies, or transformations affecting clinical meaning, remediation requires human approval. Steward dashboards provide context (severity, impact analysis, proposed plan, cost estimate) to expedite decision-making. Historical approval decisions train a reinforcement learning model to gradually reduce human overhead while maintaining safety guardrails.

6. Experimental Methodology

6.1. Workload and Infrastructure

We generated a synthetic dataset of 750,000 patients using Synthea v3.2.0 configured for US Core 6.1.0 profiles. Data (Observations, Conditions, Medications, Procedures, Encounters) was partitioned evenly across AWS (S3/Redshift), Azure (ADLS/Synapse), and GCP (GCS/BigQuery). We deployed 65 pipelines over 45 simulated days with realistic temporal patterns.

6.2. Baseline Implementations

We compared DQCs against three representative baselines sharing identical infrastructure and monitoring visibility:

- Reactive monitoring: Static SQL-based quality checks with alerting and manual remediation via ticketing.
- Expectations-based validation: Great Expectations and dbt tests integrated into orchestration with alerting; remediation remained manual.
- FHIR validator workflows: Standard HL7 FHIR Validator integrated into batch DAGs producing binary pass/fail outcomes; manual remediation.

6.3. Data-Quality Chaos Scenarios

We executed eight chaos scenarios across 216 total injections, randomized across pipelines and time windows. Scenarios included schema drift, vocabulary misalignment, freshness degradation, referential integrity failures, completeness regressions, cardinality violations, cross-cloud replication lag, and volume anomalies.

6.4. Evaluation Metrics

Primary metrics included Mean Time to Remediation (MTTR), false-positive alert rate, aggregate contract compliance rate, and automated fix rate.

7. Results

7.1. Aggregate Results across All Scenarios

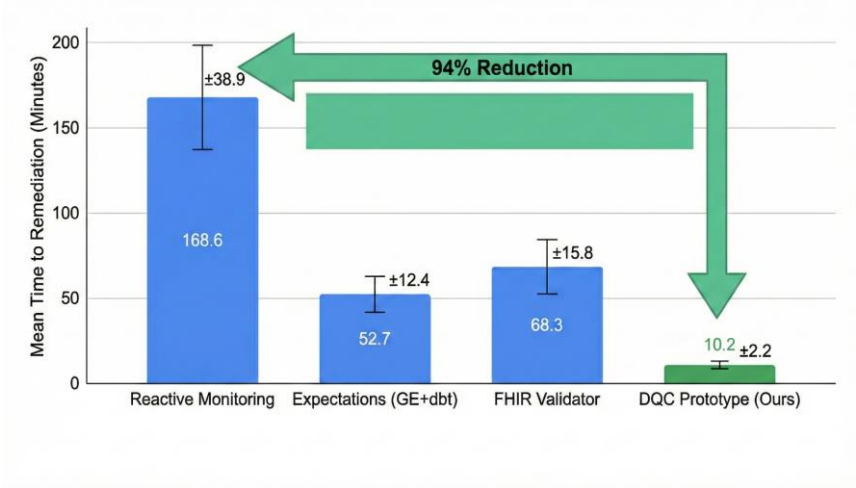

Fig 4: Comparison of Mean Time to Remediation (MTTR) across approaches under chaos conditions.

The DQC prototype reduces MTTR from over two hours (reactive baseline) to approximately 10 minutes, driven by automated detection and policy-driven remedial workflow execution. Error bars represent standard deviation.

Table I summarizes results across all 216 chaos injections

Table 1: Aggregate Results (Mean ± Std Dev)

Approach	MTTR (min)	False-Pos Rate	Compliance	Auto-Fix Rate
Reactive monitoring	168.6 (±38.9)	26.6%	—	0%
Expectations (GE+dbt)	52.7 (±12.4)	22.1%	90.4%	0%
FHIR validator	68.3 (±15.8)	24.0%	91.2%	0%
DQC (ours)	10.2 (±2.2)	14.3%	93.8%	82%

Notes: Compliance is percentage of time meeting semantic + temporal + completeness SLOs. All pairwise comparisons vs. DQC are statistically significant ($p < 0.001$).

Key findings include:

- MTTR Reduction: DQC reduced MTTR by 94.0% versus the reactive baseline (168.6 → 10.2 minutes; 95% CI: [147.2, 169.6] min, $p < 0.001$). The improvement over other baselines is driven by automated execution rather than detection speed alone.
- False-Positive Reduction: DQC achieved a 46.2% reduction in false-positives (26.6% → 14.3%, $p < 0.001$), attributable to graded semantic distance policies filtering benign drift.
- Compliance and Automation: Under active chaos, DQC maintained 93.8% aggregate compliance, achieving an 82% automated fix rate. The remaining 18% were gated for human approval due to high severity or low confidence.

7.2. Ablation Study Highlights

Disabling automated remediation (alerting only) increased MTTR from 10.2 to 94.5 minutes (+827%), confirming automation is the primary driver of recovery speed. Removing semantic distance grading nearly doubled the false-positive rate. Centralizing validation (shipping all data to the control plane) increased network costs by 12x due to egress, validating the federated approach.

8. Operational Cost Model (12-Month Toc)

We modeled the Total Cost of Ownership (TCO) comparing the direct infrastructure and personnel costs of the DQC framework against the operational costs of incident-driven toil and downtime exposure inherent in reactive approaches.

- Assumptions**:** 180 incidents/year; downtime cost of $5,000/hour (clinical operations impact); data steward loaded cost of $75/hour.
- Results**:** The reactive monitoring baseline incurred a projected 12-month TCO of $3.31M, driven primarily by $2.5M in opportunity costs from extended downtime and manual investigation. The DQC framework incurred higher direct infrastructure costs ($133k annually for agents and control plane compute) but reduced opportunity costs significantly via automation. The projected 12-month TCO for the DQC framework is $1.15M.

This represents a 65.2% TCO reduction, yielding a 1,982% ROI on the framework investment with a payback

period of approximately 0.6 months. Sensitivity analysis confirms positive ROI even if downtime costs are reduced by 60%.

9. Discussion

9.1. Why DQCs Work in Healthcare

DQCs shift quality "left" by making expectations explicit, versioned, and enforceable at boundaries. The combination of federated validation (reducing egress cost), error budgets (forcing governance decisions), and automated remediation (reducing toil) addresses the specific challenges of multi-cloud healthcare.

9.2. Limitations

Synthetic data may not capture all real-world EHR idiosyncrasies. Terminology mapping ambiguity sometimes requires clinical judgment that cannot be automated. While median approval time was low, human-in-the-loop latency could increase during incident storms.

9.3. Deployment Strategy

A phased rollout is recommended: monitor-only mode, followed by automated remediation for low-severity issues, and finally approval-gated remediation for high-severity issues once trust is established.

10. Conclusion

Data Quality Contracts operationalize semantic SLAs for multi-cloud healthcare by combining FHIR-aware constraints, temporal fitness guarantees, completeness objectives, and error budgets enforced federatedly with automated remediation and safety gates. In a rigorous synthetic evaluation with chaos injections, the DQC prototype reduced MTTR by 94% and false positives by 46% while maintaining high compliance. The associated cost model suggests a potential 65% reduction in 12-month TCO. This approach generalizes to cross-organizational exchange and real-time clinical decision support, providing a foundation for trustworthy healthcare data infrastructure.

Acknowledgments

The author thanks the open-source communities behind HL7 FHIR and Synthea for enabling reproducible experimentation.

References

1. E. Topol, *Deep Medicine: How Artificial Intelligence Can Make Healthcare Human Again*. Basic Books, 2019.
2. Z. Obermeyer and E. J. Emanuel, "Predicting the future—Big data, machine learning, and clinical medicine," *N. Engl. J. Med.*, vol. 375, no. 13, pp. 1216–1219, Sep. 2016.
3. Office of the National Coordinator for Health Information Technology, "Health Data, Technology, and Interoperability: Certification Program Updates, Algorithm Transparency, and Information Sharing (HTI-1 Final Rule)," Federal Register, vol. 89, no. 6, pp. 1192–[end page], Jan. 9, 2024.
4. R. Y. Wang and D. M. Strong, "Beyond accuracy: What data quality means to data consumers," *J. Manage. Inf. Syst.*, vol. 12, no. 4, pp. 5–33, 1996.
5. J. G. Klann et al., "Data model harmonization for the All Of Us Research Program," *PLoS One*, vol. 14, no. 2, e0212463, Feb. 2019.
6. N. Forsgren, J. Humble, and G. Kim, *Accelerate: The Science of Lean Software and DevOps*. IT Revolution Press, 2018.
7. S. Schelter et al., "Automating large-scale data quality verification," *Proc. VLDB Endow.*, vol. 11, no. 12, pp. 1781–1794, Aug. 2018.
8. A. Polyzotis et al., "Data lifecycle challenges in production machine learning: A survey," *ACM SIGMOD Rec.*, vol. 47, no. 2, pp. 17–28, 2018.
9. D. F. Sittig and H. Singh, "A new sociotechnical model for studying health information technology," *BMJ Qual. Saf.*, vol. 19, suppl. 3, pp. i68–i74, Oct. 2010.
10. C. Batini et al., "Methodologies for data quality assessment and improvement," *ACM Comput. Surv.*, vol. 41, no. 3, art. 16, Jul. 2009.
11. HL7 International, "FHIR Release 4 (R4): Base specification," 2019.
12. HL7 International, "US Core Implementation Guide (STU 6.1.0) Releases," 2023. [Online]. Available: github.com/HL7/US-Core/releases. [Accessed: Jan. 31, 2026].
13. HL7 International, "FHIR Validator," 2024.
14. T. Kraska et al., "SageDB: A learned database system," in *Proc. CIDR*, 2019.
15. M. Kleppmann, *Designing Data-Intensive Applications*. O'Reilly, 2017.
16. S. K. Y. Battula, "Adaptive data quality management for multi-cloud healthcare warehouses: FHIR-aware semantics and unsupervised thresholding," *Int. J. Artif. Intell., Data Sci. Mach. Learn. (IJAIDSML)*, vol. 6, no. 4, pp. 218–226, Dec. 2025, doi: 10.63282/3050-9262.IJAIDSML-V6I4P130.
17. S. Rose, O. Borchert, S. Mitchell, and S. Connelly, Zero Trust Architecture, NIST Special Publication 800-207, Aug. 2020.
18. Temporal Technologies, "Temporal Documentation," 2024. [Online]. Available: docs.temporal.io. [Accessed: Jan. 31, 026].
19. S. Amershi et al., "Guidelines for human-AI interaction," in *Proc. CHI*, 2019.
20. R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd ed. MIT Press, 2018.
21. J. Walonoski et al., "Synthea: An approach, method, and software mechanism for generating synthetic patients and the synthetic electronic health care record," J. Am. Med. Inform. Assoc., vol. 25, no. 3, pp. 230–238, Mar. 2018, doi: 10.1093/jamia/ocx079
22. Prasanth Tirumalasetty, (2025). A Computer Vision and Machine Learning Framework for Automated Sterilization and Batch Validation in Regulated Surgical Inventories Warehousing.