International Journal of Research Publications in Engineering, Technology and Management (IJRPETM)

|www.ijrpetm.com | ISSN: 2454-7875 | editor@ijrpetm.com |A Bimonthly, Peer Reviewed & Scholarly Journal|

||Volume 8, Issue 3, May-June 2025||

DOI:10.15662/IJRPETM.2025.0803015

An Ingestion-Gated Lakehouse Architecture for Multi-CRO SDTM Harmonization in Clinical Trial Data Management

Sai Krishna Puli

Data Architecture & AI Integration, Georgia, USA

ABSTRACT: Sponsors of global clinical programs typically receive Study Data Tabulation Model (SDTM) datasets from several Contract Research Organizations (CROs), each with its own Electronic Data Capture configuration, mapping conventions, and delivery cadence, and nonconformance is commonly found late, through downstream review. This paper proposes an ingestion-gated lakehouse architecture for multi-CRO SDTM harmonization. Data custody stays in CRO-local environments; a conformance gate at each CRO boundary evaluates every delivery against rules, schemas, and controlled-terminology versions pinned for the study and served by a sponsor metadata control plane; and only accepted or flagged domains enter a bronze, silver, and gold lakehouse, where subject identity and terminology are harmonized. The design separates storage services from the conformance engine, classifies validation failures into classes with defined dispositions, treats ADaM derivation and traceability as a distinct layer, and specifies rule governance, subject-identity resolution, failure handling, regulatory and validation controls, and non-functional requirements. The central mechanism is earlier detection: moving the discovery of conformance issues from downstream review to the point of delivery, where correction is cheaper and remains with the originating CRO. To make the proposal evaluable, the paper defines seven architecture metrics and a reproducible, seeded simulation of the detection mechanism (14 synthetic studies, 5 synthetic CROs, 1,247 domain deliveries), extended by a parameter sensitivity analysis and a break-even structure. The simulation demonstrates the mechanism under disclosed assumptions and is not a prediction of performance. Its outcome depends jointly on the gate's catch rate and on how much resolution cost an early catch avoids; only the unit resolution cost range (USD 28-225) is drawn from the literature, and the break-even analysis uses placeholder costs and an avoided-downstream-cost term that the paper cannot quantify. No implementation in a production clinical environment is reported, and most of the proposed metrics, the identity-resolution design, and the regulatory validation controls are specified but neither implemented nor measured. The contribution is the architecture, its mechanism, and a framework for evaluating it in practice.

KEYWORDS: Data lakehouse, CDISC SDTM, ADaM, Define-XML, conformance validation, metadata control plane, subject identity resolution, clinical trial data management, data architecture, simulation study.

I. INTRODUCTION

Anyone who has sat in a database-lock war room knows the pattern: a CRO's domain delivery looks fine on the surface, passes the CRO's own QC, and still breaks a downstream program three weeks later because a controlled-terminology version shifted or a supplemental qualifier was renamed. Sponsors conducting multinational clinical trials commonly contract three to eight Contract Research Organizations (CROs) per program, each responsible for data management at a defined set of sites. Each CRO typically operates its own Electronic Data Capture (EDC) system configuration and internal SDTM mapping conventions, so that SDTM-compliant deliverables from different CROs on the same program are compliant with the CDISC Study Data Tabulation Model specification in structure but may still diverge in controlled-terminology version usage, supplemental-qualifier conventions, and file delivery cadence. Reconciling these heterogeneous deliveries into a single, harmonized, analysis-ready dataset at the sponsor side is, in prevailing industry practice, performed substantially through manual data management effort after each delivery — effort that scales roughly linearly with the number of CRO relationships and rarely gets easier as a program grows.

Independent of the clinical trials domain, the data engineering literature has converged on the “lakehouse” architectural pattern — a data platform combining the low-cost, open-format storage of a data lake with the transactional guarantees and query performance historically associated with a data warehouse. Open table formats such as Delta Lake and Apache Iceberg (Apache Software Foundation, n.d.) provide the ACID-transactional, schema-evolution-aware storage layer that makes this pattern practical over commodity cloud object storage. Separately, clinical data managers report

|www.ijrpetm.com | ISSN: 2454-7875 | editor@ijrpetm.com |A Bimonthly, Peer Reviewed & Scholarly Journal|

||Volume 8, Issue 3, May-June 2025||

DOI:10.15662/IJRPETM.2025.0803015

that conformance and data-quality issues in multi-organization deliveries are commonly found through downstream review and resolved through query and reconciliation cycles, whose cost this paper models only with illustrative, author-chosen inputs.

This paper proposes an architecture, described in Section 4, that applies the lakehouse pattern — specifically, a medallion (bronze/silver/gold) layering with an automated conformance-checking gate — to the problem of multi-CRO SDTM and ADaM harmonization. The two are treated as distinct problems (Sections 4.4 and 4.5): SDTM harmonization concerns structure, terminology, and identity, whereas ADaM harmonization concerns derivations, analysis populations, and traceability. Only the SDTM conformance gate is evaluated by simulation; the ADaM layer is specified architecturally and is not evaluated. The literature search conducted for this paper (Section 2) did not identify a published architecture combining these elements for multi-CRO clinical data harmonization. The review was targeted and not systematic and does not cover unpublished vendor or in-house implementations, so this is a statement about the literature reviewed and not a claim of priority. To illustrate the architecture's intended quantitative behavior, Section 5 describes a synthetic dataset and a simulation model with author-chosen, illustrative cost inputs; Section 6 reports the simulation results; Section 7 examines the proposal's limitations.

II. LITERATURE REVIEW

This section positions the proposal against seven bodies of work: the clinical data standards it builds on, automated conformance validation, data quality frameworks, EDC-to-SDTM production, lakehouse architecture, data contracts and federated data governance, and clinical data integration. Standards bodies' pages, project documentation, regulatory notices, and recent papers are cited, and non-peer-reviewed sources are identified as such. Table 1 summarizes the positioning. The search was not exhaustive.

2.1 Clinical Data Standards

The architecture is organized around the CDISC foundational standards. The Study Data Tabulation Model (SDTM) defines the structure of tabulated study data (CDISC, n.d.-f), and the Analysis Data Model (ADaM) defines analysis-ready datasets derived from it (CDISC, n.d.-a). Define-XML is the machine-readable metadata describing the structure, origin, and terminology bindings of submitted datasets (CDISC, n.d.-e), and CDISC Controlled Terminology supplies the standard code lists, developed with the National Cancer Institute's Enterprise Vocabulary Services and released in dated versions (CDISC, n.d.-b). The dated versioning of terminology is what allows the architecture to pin a terminology version per study (Section 4.2). Dataset-JSON is a newer exchange standard for tabular data in JSON, designed to address the limitations of legacy formats and able to reference a Define-XML document for fuller metadata (CDISC, n.d.-d; version 1.1 dated 5 December 2024). In April 2025 the U.S. Food and Drug Administration published a request for comments on whether to accept Dataset-JSON version 1.1 for regulatory submissions, with long-term potential to replace the SAS version 5 transport format (Food and Drug Administration, 2025); the notice states no adoption date. For this paper the relevance is practical: a gate must accept whichever exchange format a CRO delivers, and the architecture handles this by parsing deliveries in an adapter ahead of the gate, so that the validation classes of Table 6 apply to the logical dataset and not to a file format (Section 4.3).

2.2 Automated Conformance Validation

Rule-based validation of CDISC datasets is established practice. The OpenCDISC validation framework, distributed through Pinnacle 21 (n.d.), is a rule-based validator for CDISC-formatted data. CDISC's CORE initiative publishes an open-source rules engine intended for validating clinical trial data against data standards, with the conformance rules expressed in a machine-executable form (CDISC, n.d.-c). Conference literature reports specification-driven SDTM generation in production (Zhang et al., 2026) and validation of AI-generated SDTM mappings against CORE using synthetic data (Belligoli et al., 2026). These tools and rule sets address what a dataset must satisfy and how to execute those checks. They do not address, and were not designed to address, the questions on which this paper concentrates: at which point in a multi-party delivery flow the check runs, how rules are versioned, promoted, and pinned per study across many studies and CROs, what disposition follows each class of failure, and what happens when the validation machinery itself fails. The proposed architecture therefore treats a rules engine such as CORE as a candidate implementation of the conformance engine of Section 4.2.2 and does not compete with it; whether CORE's rule set, execution model, and performance suit ingestion-time use in this architecture is untested here.

|www.ijrpetm.com | ISSN: 2454-7875 | editor@ijrpetm.com |A Bimonthly, Peer Reviewed & Scholarly Journal|

||Volume 8, Issue 3, May-June 2025||

DOI:10.15662/IJRPETM.2025.0803015

2.3 Data Quality Frameworks

Data-quality assessment frameworks for health data commonly organize checks into conformance, completeness, and plausibility, with conformance subdivided into value, relational, and computational conformance. The gate described here is a verification-context control that operates mainly on conformance (value conformance for terminology and data types, relational conformance for keys and cross-domain relationships); completeness appears only through delivery manifests (Section 4.7, scenario 9), and plausibility is not addressed. Declarative, metadata-driven validation, in which constraint checks are expressed as rules and not as bespoke code, is the style the rules proposed here follow; general data-validation tooling carries no knowledge of clinical standards and does not concern multi-organization clinical deliveries.

2.4 EDC-to-SDTM Production

A separate literature addresses producing SDTM from source data. Zhang et al. (2026) report a specification-driven SDTM engine in production, and Belligoli et al. (2026) validate AI-generated mappings with CORE. This work concerns generating SDTM at a single producer. The present architecture begins where it ends: it assumes several independent producers, each free to use any such tool inside its own environment (Section 4.1), and concerns what the sponsor does on receipt.

2.5 Lakehouse, Open Table Formats, and Medallion Layering

The lakehouse pattern combines open direct-access data formats with transactional metadata layers to provide data-warehouse-grade reliability without maintaining separate lake and warehouse systems. Delta Lake and Apache Iceberg are open table formats that provide ACID transactions over cloud object stores, with snapshot-based versioning and schema evolution (Apache Software Foundation, n.d., for Iceberg; the citation is to project documentation). The medallion layering of bronze, silver, and gold tables, in which data quality and structure improve progressively from raw to curated, is documented as a design pattern in Databricks (n.d.) documentation and not in the peer-reviewed literature located. These sources establish the storage and layering substrate of the proposal. They say nothing about clinical conformance semantics, and Section 4.2.2 explains why a table format's schema enforcement is not a substitute for a conformance engine.

2.6 Data Contracts, Schema Registries, and Federated Governance

Three bodies of practice inform the metadata control plane. The Open Data Contract Standard (Bitol, n.d.) defines a vendor-neutral YAML format for contracts between data producers and consumers covering schema, quality rules, ownership, and service levels, which corresponds to the Dataset Contracts component of Section 4.2. Schema registries, as documented for Confluent Schema Registry (Confluent, n.d.), version each schema and check compatibility before accepting a new version, which corresponds to the Schema Versions component. In both cases the contract or schema is whatever the producer and consumer agree; in this architecture the semantics are fixed in advance by an external standard and a regulator, and terminology is controlled, so these tools supply form and mechanics, not content.

Data mesh (Dehghani, 2020) proposes four principles: domain-oriented ownership, data as a product, a self-serve data platform, and federated computational governance, in which a federation sets global rules that the platform enforces automatically. The proposal shares the structure of the first, second, and fourth principles: CRO custody corresponds to domain ownership, a contract-bound delivery to a data product, and the metadata control plane with boundary gates to computational governance. It differs in three respects. The domains are external contracted organizations, not internal teams. Accountability to regulators stays with the sponsor and cannot be delegated. And there is no self-serve platform inside the CROs; the sponsor supplies a versioned gate package instead. Dehghani (2020) also notes that identifying an entity that crosses domain boundaries is a global concern, which parallels the treatment of subject identity in Section 4.4.1. Because the pattern is described mainly in practitioner writing, it is used here as an organizing analogy and not as a validated reference architecture.

2.7 Clinical Data Integration, Warehousing, and Query Burden

This paper makes no literature-based claim about the share of queries that change data, the cost of resolving a query, or the challenges of multi-site harmonization. Broader clinical data platforms such as i2b2 and the OHDSI community's open tools target observational or clinical-record data under a common data model chosen for analysis, whereas the present problem is regulated, submission-oriented, and fixed to CDISC structures. Vendor and trade publications describe lakehouse platforms for clinical data and CRO data sharing; in the searches conducted for this review no peer-reviewed publication was found that evaluates such a platform for multi-CRO SDTM conformance, and vendor material is not used here as evidence.

|www.ijrpetm.com | ISSN: 2454-7875 | editor@ijrpetm.com |A Bimonthly, Peer Reviewed & Scholarly Journal|

||Volume 8, Issue 3, May-June 2025||

DOI:10.15662/IJRPETM.2025.0803015

Cost inputs used later in the paper are therefore author-chosen illustrative values and not literature-derived (Sections 5.3 and 6.3).

2.8 Positioning Within the Literature Reviewed

Table 1 sets the reviewed work against what it leaves open for the problem of this paper. The standards, validation engines, quality vocabularies, storage layers, contract formats, and governance patterns each supply a part of the solution. Within the literature reviewed, we did not identify a published architecture combining them for multi-CRO SDTM reconciliation, with conformance enforced at the CRO boundary under governed, version-pinned rules, defined failure dispositions, identity resolution, and failure handling, together with a framework for evaluating it. The review was targeted and not systematic, and unpublished vendor and in-house systems may already combine these elements. This paper therefore does not claim priority for the combination, and none of the individual components is presented as novel.

Table.1. Positioning of the proposal against related work

Area	Representative sources	What they address	What it leaves open for this problem
Clinical data standards	SDTM, ADaM, Define-XML, Controlled Terminology, Dataset-JSON (CDISC, n.d.-a, n.d.-b, n.d.-d, n.d.-e, n.d.-f)	What conformant datasets, metadata, terminology, and exchange files look like	Where in a multi-CRO flow conformance is checked, against which pinned versions, and what happens on failure
Conformance validation engines	OpenCDISC framework (Pinnacle 21, n.d.); CDISC Rules Engine (CDISC, n.d.-c); validation of AI-generated mappings (Belligoli et al., 2026)	Executing standards-derived rules against datasets	Rule governance across studies and CROs, placement at the CRO boundary, failure dispositions, and the cost of late detection
Data quality frameworks	Conformance, completeness, and plausibility checks; declarative data tests (no source cited)	A vocabulary for conformance, completeness, and plausibility; constraint checking on large data	Clinical-standards-aware rule libraries and their governance in a regulated sponsor-CRO relationship
EDC-to-SDTM production	Metadata-driven and learned mapping (Zhang et al., 2026; Belligoli et al., 2026)	Generating SDTM from source data at one producer	Receiving, validating, and harmonizing SDTM from several independent producers
Lakehouse, table formats, medallion layering	Apache Software Foundation (n.d.); Databricks (n.d.)	Transactional open storage, versioning, and staged refinement	Clinical conformance semantics; the table format is a storage service, not a validator (Section 4.2.2)
Data contracts and schema registries	Open Data Contract Standard (Bitol, n.d.); Schema Registry (Confluent, n.d.)	Machine-readable producer-consumer agreements and versioned schemas with compatibility checks	Contracts and schemas whose semantics are defined by an external standard (SDTM) and a regulator, with controlled terminology
Data mesh and federated governance	Dehghani (2020)	Domain ownership, data as a product, federated computational governance	Domains that are external contracted organizations, with sponsor accountability fixed by regulation
Clinical data integration and warehousing	No peer-reviewed multi-CRO platform evaluation located	Multi-site, EHR, and observational data integration; query burden in trial data management	Multi-CRO sponsor deliverables in CDISC form, validated at ingestion

|www.ijrpetm.com | ISSN: 2454-7875 | editor@ijrpetm.com |A Bimonthly, Peer Reviewed & Scholarly Journal|

||Volume 8, Issue 3, May-June 2025||

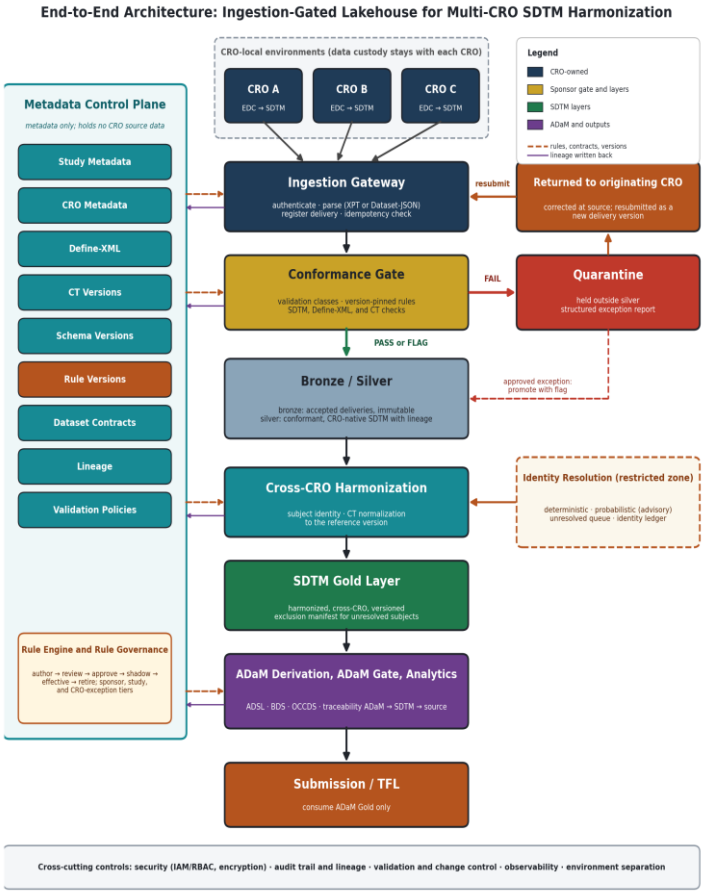
DOI:10.15662/IJRPETM.2025.0803015

III. PROBLEM STATEMENT

Manual, downstream reconciliation of multi-CRO SDTM deliveries has two structural properties this paper treats as the target for architectural intervention. First, conformance issues (schema drift, controlled-terminology version mismatch, naming-convention divergence) are typically discovered only when a downstream conformance check or statistical programming step fails, rather than at the point of ingestion, which delays remediation and concentrates rework near database lock, when schedule pressure is highest. Second, reconciliation effort scales with the number of concurrent CRO relationships and studies, and manual reconciliation does not benefit from economies of scale across studies in the way an automated, reusable conformance-rules engine could. The problem this paper addresses is whether an ingestion-time, automated conformance gate — rather than downstream detection — can be expected, under a stated and transparent simulation model with author-chosen illustrative cost inputs, to reduce both the delay between issue occurrence and issue detection, and the total resolution cost associated with those issues.

IV. PROPOSED METHODOLOGY (SYSTEM ARCHITECTURE)

The architecture described in this section is a design proposal. No implementation or production deployment is reported. The architecture is federated in a specific sense: each CRO retains its own local data environment, and a conformance gate at each CRO boundary checks every delivery against contracts and rules published by a sponsor-side federation and metadata plane; only conformant (or flagged) domains cross into the sponsor's harmonized lakehouse. Figure 1 shows the end-to-end flow. Deliveries from CRO-local environments enter an ingestion gateway and the conformance gate; passed or flagged domains land in bronze and silver, while quarantined domains are returned to the originating CRO for correction and resubmission (an approved exception can promote a quarantined domain with a flag); harmonization produces the gold layer, and the ADaM layer feeds submission and tables, figures, and listings. The metadata control plane supplies pinned contracts, rules, and versions to each stage. The qualities the design must exhibit are stated as non-functional requirements in Section 4.9.



|www.ijrpetm.com | ISSN: 2454-7875 | editor@ijrpetm.com |A Bimonthly, Peer Reviewed & Scholarly Journal|

||Volume 8, Issue 3, May-June 2025||

DOI:10.15662/IJRPETM.2025.0803015

Table 2 restates the architecture of Figure 1 as a reference architecture: twelve layers, the responsibility of each, and the technology capability it requires, with the section that specifies it. The technology column names capabilities, not products. Any product or service providing the capability could realize a layer, and the table does not recommend or evaluate particular technologies. Two layers are both concerned with validation but address different objects: the Validation layer checks the conformance of data, and the Compliance layer validates the system itself as a computerized system. The canonical model of the Harmonization layer is not a new data model; it is the sponsor's reference SDTM version, with the canonical subject identifier of Section 4.4.1. The table is a planning template. No layer has been implemented or tested.

Table.2.Reference architecture: layers, responsibilities, and required technology capabilities

Layer	Responsibility	Key technology capability	Section
Source	CRO delivery	SFTP or API	4.1
Ingestion	File and API acquisition	Landing service	4.3
Bronze	Immutable source	Open table format	4.2.2, 4.4
Validation	Conformance	Rules engine	4.3
Quarantine	Exception management	Exception store	4.3, 4.7
Silver	Conformant SDTM	Curated tables	4.4
Harmonization	Cross-CRO alignment	Canonical model	4.4, 4.4.1
Gold	Analysis-ready data	Curated analytical layer	4.4, 4.5
Metadata	Governance	Metadata registry	4.2
Observability	Monitoring	Metrics and logging	4.9
Security	Access control	IAM and RBAC	4.8
Compliance	GxP controls	Computer system validation and change management	4.8

4.1 CRO-Local Data Environments

Each CRO continues to operate its own EDC configuration and SDTM build inside its own environment; the architecture does not require CROs to move source or working data into a shared platform. This is the property that makes the design federated rather than merely centralized: data custody, local mapping conventions, and operational tooling remain with the originating CRO, and the sponsor interacts with each environment only through a defined contract and a delivery boundary.

4.2 Sponsor Federation and Metadata Control Plane

The sponsor operates a metadata control plane that holds every shared, versioned artifact the CROs are held to. It has nine components (Table 3). Figure 2 shows its core artifacts (Define-XML, controlled terminology, schemas, rules), the versioned bundle that pins them per study, the lineage record, and the stages that resolve artifacts by version: ingestion, validation, harmonization, and audit. Figure 1 shows where the plane sits in the overall topology. Rules and contracts flow outward from the plane to each CRO-boundary gate (dashed arrows in Figure 1), while gated deliveries flow directly to the lakehouse and only their lineage records are written back to the plane. The plane is deliberately metadata-only: it coordinates and governs, but does not hold CRO source data. The decomposition is a design proposal; it is not drawn from a reported implementation.

International Journal of Research Publications in Engineering, Technology and Management (IJRPETM)

|www.ijrpetm.com | ISSN: 2454-7875 | editor@ijrpetm.com |A Bimonthly, Peer Reviewed & Scholarly Journal|

||Volume 8, Issue 3, May-June 2025||

DOI:10.15662/IJRPETM.2025.0803015

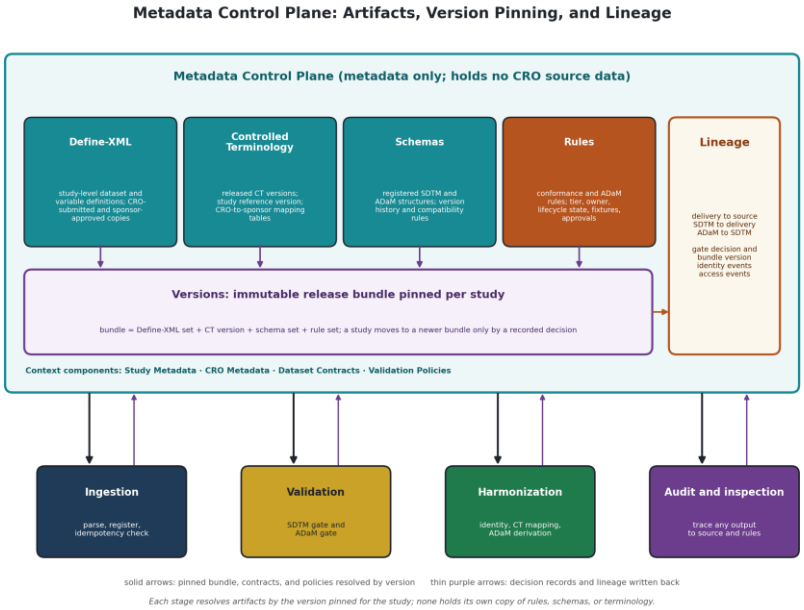


Table.3. Components of the proposed metadata control plane.

Component	Holds	Stages served
Study Metadata	Study identifiers, protocol version, phase, reference CT version, and the pinned bundle of rule, schema, and CT versions for the study	Ingestion; Validation; Harmonization
CRO Metadata	CROs, contracted scope, delivery obligations and schedule, active exception records	Ingestion; Validation
Define-XML	Study-level Define-XML (CDISC, n.d.-e) as submitted by each CRO and as approved by the sponsor	Validation; Harmonization
CT Versions	Released controlled-terminology versions, the sponsor reference version per study, and CRO-to-sponsor mapping tables	Ingestion; Validation; Harmonization
Schema Versions	Registered SDTM and ADaM dataset and variable structures, with version history and compatibility rules	Ingestion; Validation
Rule Versions	Conformance and ADaM rules with identifiers, versions, tiers, lifecycle state, owners, and test evidence (Section 4.2.1); released rule sets as immutable tagged bundles	Validation
Dataset Contracts	Expected datasets per study and delivery, their required structure and timing, and delivery status (accepted, flagged, quarantined)	Ingestion; Validation
Lineage	Delivery-to-source, SDTM-to-delivery, and ADaM-to-SDTM lineage, plus the rule-set version behind every gate decision	Validation; Harmonization; audit
Validation Policies	Default disposition per validation class (Table 6), flag thresholds, escalation paths, and what each gate must record	Validation

4.2.1 Rule Governance

A rule library is only as trustworthy as the process that changes it, and the gates are only as consistent as the rule versions they run. Table 4 states the proposed answer to each governance question a quality reviewer would be expected to ask. These are proposed policies, not reported practice. Which role titles and approval thresholds are appropriate, and how electronic-record and validation expectations apply, would be determined by the sponsor's own quality system; this paper does not assert regulatory compliance.

|www.ijrpetm.com | ISSN: 2454-7875 | editor@ijrpetm.com |A Bimonthly, Peer Reviewed & Scholarly Journal|

||Volume 8, Issue 3, May-June 2025||

DOI:10.15662/IJRPETM.2025.0803015

Table.4. Proposed rule-governance model

Question	Proposed design
Who owns rules?	Each rule has a named owner recorded in the Rule Versions component. Sponsor-tier rules are owned by the sponsor's data standards function; study-tier rules by the study data management lead; CRO exceptions are owned by the sponsor study data manager, never by the CRO that requested them.
How are rules versioned?	Every rule has an immutable identifier and a version; a published rule set is an immutable, tagged bundle in the Rule Versions component. A study pins one rule-set version and moves to a later one only by a recorded decision. Every gate decision records the rule-set version that produced it.
How are rules promoted?	Lifecycle states: draft, tested, approved, shadow, effective, deprecated, retired. A new or changed rule first runs in shadow mode, where findings are recorded but nothing is quarantined, and becomes effective only after the shadow findings are reviewed.
How are rules tested?	Each rule carries fixtures of known-pass and known-fail datasets. Before approval, the candidate rule set is replayed against archived bronze deliveries (using table time-travel) to report which past decisions would change, so impact is known before enforcement.
Can rules be study-specific?	Yes, through a study overlay on the sponsor rule set. An overlay may add rules or tighten severity. It may not weaken a sponsor-tier hard-fail rule; relaxing one requires an exception.
Can CRO-specific exceptions exist?	Yes, as explicit exception records scoped to a CRO, study, domain, and variable, with justification, approver, compensating check, and expiry date. Exceptions are visible on the delivery and in the Lineage component; no silent exceptions exist, and an expired exception reverts to the underlying rule.
How are rules retired?	A rule is deprecated with advance notice and retired only when no active study pins a rule set that contains it. Retired rules are never deleted, so historical gate decisions remain reproducible.
Who approves a rule change?	A change request is authored, independently reviewed, and approved by three different people (segregation of duties). The approval, reviewer, and test evidence are stored with the rule version. Sponsor-tier changes require the data standards owner and quality assurance; study-tier changes require the study data management lead and a reviewer.

Listing 1. Illustrative rule and exception records for the Rule Versions component (Section 4.2.1). Proposal-level pseudocode, not a reported implementation.

```
proposal-pseudocode -- rule_versions.yaml
# Illustrative rule record and CRO exception record (proposal-level; not implemented)
rule: DM_CT_VERSION_MATCH
  version: 1.3.0
  tier: SPONSOR            # SPONSOR | STUDY_OVERLAY
  owner: data_standards
  state: EFFECTIVE         # DRAFT|TESTED|APPROVED|SHADOW|EFFECTIVE|DEPRECATED|RETIRED
  fixtures: [dm_ct_pass_01, dm_ct_fail_01]
  approval: {author: A, reviewer: B, approver: C}   # three distinct identities
  supersedes: 1.2.0

exception: EXC-0042
  rule: DM_CT_VERSION_MATCH
  scope: {cro: CRO_B, study: STUDY_X, domain: DM, variable: ETHNIC}
  justification: "CT release lag; mapping table registered"
  compensating_check: ct_mapping_table_applied
  approved_by: sponsor_study_data_manager
  expires: 2027-03-31      # reverts to rule after expiry
```

|www.ijrpetm.com | ISSN: 2454-7875 | editor@ijrpetm.com |A Bimonthly, Peer Reviewed & Scholarly Journal|

||Volume 8, Issue 3, May-June 2025||

DOI:10.15662/IJRPETM.2025.0803015

4.2.2 Separation of Storage Services and the Conformance Engine

Two different kinds of component appear in the architecture, and they solve different problems. An open table format (Apache Software Foundation, n.d.) is a storage service: it governs how tables are written, versioned, and read. An SDTM conformance engine is a validation service: it decides whether the content of a delivery is acceptable under the CDISC standards and the study's own definitions. Schema evolution in a table format is not SDTM conformance validation. A table format can tell that an incoming write is incompatible with the table's declared schema, or can be told to accept a new column; it has no knowledge of whether that column is permitted in the domain, whether a value belongs to the controlled terminology version the study references, or whether the dataset agrees with its Define-XML. Conversely, a write that satisfies the table schema can still violate SDTM, and a write that the format would reject on schema grounds is only one of the failure types the conformance engine must classify (Table 6). Table 5 assigns responsibilities to each service.

Table.5.Proposed separation of responsibilities between the open table format and the SDTM conformance engine

Concern	Open table format (storage service)	SDTM conformance engine (validation service)
Primary question	Is this write consistent, atomic, and recoverable?	Is this delivery acceptable under SDTM and the study definition?
Schema	Schema evolution and enforcement against the table's declared schema	Variables and structure permitted by SDTM and by the registered study schema (Schema Versions)
Transactions and storage	Atomic commits, concurrent-write isolation, storage consistency	Not a concern of the engine
Versioning and snapshots	Table snapshots and time-travel read of earlier table states	Rule-set, CT, and schema version pinned for the study; the snapshot used is recorded in Lineage
Rules and metadata	None	SDTM and domain rules, variable rules, Define-XML validation, controlled terminology, cross-domain rules, study overlays
Disposition	Commit or fail a write	Reject, quarantine, flag, or defer (Table 6)
Where it runs	Inside the lakehouse, at write	At the CRO-boundary gate, before landing; and after harmonization for cross-CRO checks

The two services are composed in a fixed order. The conformance engine runs first and makes the disposition decision; only a delivery that is accepted or flagged is committed to bronze, where the table format makes the write atomic and creates a snapshot whose identifier the engine records in the Lineage component alongside the rule-set version used. Automatic schema evolution is not relied on to admit new structure into silver or gold; a structural change is accepted only after the change has been registered in Schema Versions through the governance process of Section 4.2.1. The table format then contributes what it is suited to, which is reproducibility: because earlier snapshots remain readable, a past gate decision can be replayed against exactly the data it saw. This division is a design choice of the proposed architecture and has not been tested in a reported implementation.

4.3 Boundary Conformance Gates

A gate sits at each CRO boundary. Before a domain leaves the CRO environment, it is checked against the rules published by the metadata plane, evaluated against the study's Define-XML metadata (CDISC, n.d.-e). Because the same rule-set version is applied at every gate, conformance is evaluated identically across CROs. Rather than treating all failures alike, every rule is assigned to one of eleven validation classes, and each class has a default disposition held in the Validation Policies component of the control plane (Table 6). Four dispositions are used. Reject returns the whole delivery without landing it, because the delivery cannot be interpreted (for example, a required variable is absent). Quarantine holds the affected domain outside silver and returns a structured, machine-readable exception report to the originating CRO, so that correction happens where the data is owned. Flag promotes the domain with an annotation that travels with it into lineage. Downstream means the check is not evaluated at the SDTM gate but at a later stage. Defaults can be tightened by a study overlay, and relaxed only through an approved exception (Section 4.2.1). Deliveries may arrive as SAS transport files or, where adopted, as Dataset-JSON (Section 2.1); an adapter ahead

|www.ijrpetm.com | ISSN: 2454-7875 | editor@ijrpetm.com |A Bimonthly, Peer Reviewed & Scholarly Journal|

||Volume 8, Issue 3, May-June 2025||

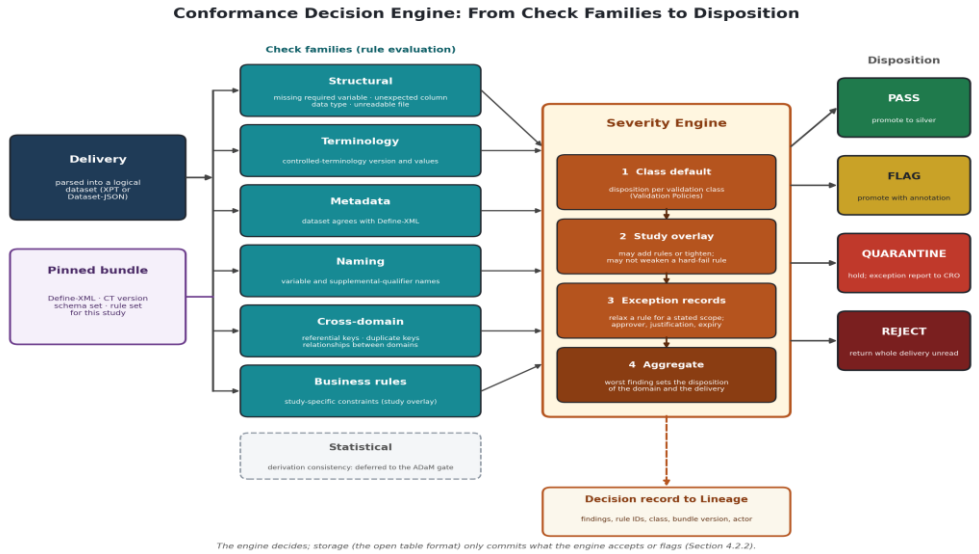
DOI:10.15662/IJRPETM.2025.0803015

of the gate parses either into a logical dataset, so that the classes apply regardless of file format, and an unreadable or malformed file fails the structural class.

Table.6. Proposed validation classification model with default dispositions. The defaults are design proposals and are configurable per study.

Validation class	Example	Default action	Evaluated at
Structural	Missing required variable	Reject	CRO-boundary gate
Schema	Unexpected column	Quarantine	CRO-boundary gate
Data type	Character value in a numeric variable	Quarantine	CRO-boundary gate
Controlled terminology	Wrong CT version	Quarantine	CRO-boundary gate
Metadata	Dataset does not match Define-XML	Quarantine	CRO-boundary gate
Naming	Invalid variable name	Flag	CRO-boundary gate
Referential	Invalid STUDYID	Quarantine	CRO-boundary gate
Record-level	Duplicate key	Flag or Reject (per policy)	CRO-boundary gate
Cross-domain	Invalid relationship between domains	Quarantine	CRO-boundary gate; cross-CRO harmonization
Business rule	Study-specific constraint	Configurable	Study overlay at the gate
Statistical	ADaM derivation inconsistency	Downstream	ADaM gate (Section 4.5)

Figure 3 shows how a delivery is evaluated inside the gate. Rules are grouped into six check families, which gather the eleven classes of Table 6 as follows: structural (structural, schema, and data type), terminology, metadata, naming, cross-domain (referential, record-level, and cross-domain), and business rules; statistical checks are deferred to the ADaM gate. Each family produces findings, and a severity engine converts them into a disposition in four steps: the class default is read from the Validation Policies component, the study overlay may tighten it, approved exception records may relax it for a stated scope and period, and the worst finding sets the disposition of the domain and of the delivery. The outcome is PASS, FLAG, QUARANTINE, or REJECT, and a decision record that names the findings, the rules, and the bundle version is written to Lineage.



International Journal of Research Publications in Engineering, Technology and Management (IJRPETM)

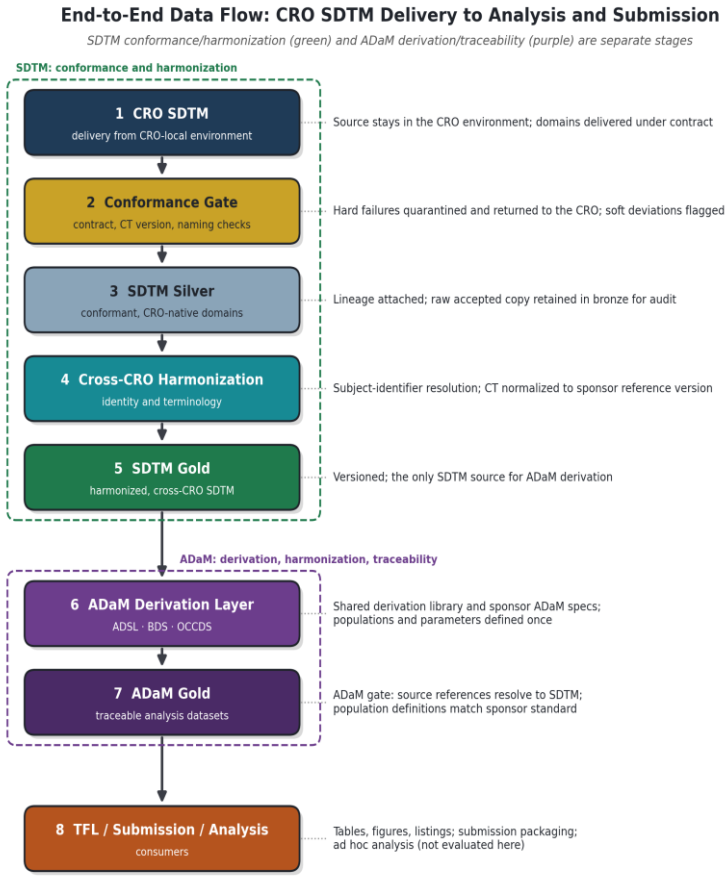
|www.ijrpetm.com | ISSN: 2454-7875 | editor@ijrpetm.com |A Bimonthly, Peer Reviewed & Scholarly Journal|

||Volume 8, Issue 3, May-June 2025||

DOI:10.15662/IJRPETM.2025.0803015

4.4 End-to-End Data Flow: From CRO SDTM to Analysis and Submission

Figure 4 traces a delivery through eight stages, in two zones that are deliberately kept apart: SDTM conformance and harmonization (stages 1-5) and ADaM derivation, harmonization, and traceability (stages 6-7), followed by the downstream consumers (stage 8). The stages are as follows. (1) CRO SDTM: the CRO delivers SDTM domains from its local environment under the contract described in Section 4.3. (2) Conformance Gate: the delivery is checked against the registered rules; hard failures are quarantined and returned to the CRO, and soft deviations are promoted with a flag. (3) SDTM Silver: accepted, CRO-native conformant domains are held with lineage metadata drawn from the metadata plane; the unmodified accepted copy is retained in bronze, stored in an ACID-transactional open table format (Apache Software Foundation, n.d.) whose snapshots support audit reproducibility; the table format provides storage guarantees, while conformance decisions are made by the engine of Section 4.2.2. (4) Cross-CRO Harmonization: subject identifiers are resolved across CROs through the identity-resolution service of Section 4.4.1 (relevant for platform and basket trials that source sites from several CROs) and controlled terminology is normalized to the sponsor's reference CT version. (5) SDTM Gold: harmonized, cross-CRO, cross-study SDTM datasets are published as versioned tables, and this is the only SDTM source from which ADaM is derived.



(6) ADaM Derivation Layer: ADSL, BDS, and OCCDS datasets (CDISC, n.d.-a) are produced from SDTM Gold using sponsor ADaM specifications and a shared, version-controlled derivation library, so that analysis populations and parameters are defined once rather than separately by each CRO's statistical programmers. (7) ADaM Gold: before publication, an ADaM gate verifies that every source reference resolves to SDTM data at the recorded version and that population definitions match the sponsor standard; the published datasets carry traceability from ADaM to SDTM to source. (8) TFL / Submission / Analysis: tables, figures, listings, submission packaging, and ad hoc analyses consume ADaM Gold only. Stages 4 and 6 through 8 are architecturally described but are not addressed by the simulation in Section 5, which models only conformance-issue detection timing and resolution cost at stage 2 and does not measure identity-resolution accuracy, derivation correctness, or downstream effects - a scope limitation discussed further in Section 7.

International Journal of Research Publications in Engineering, Technology and Management (IJRPETM)

|www.ijrpetm.com | ISSN: 2454-7875 | editor@ijrpetm.com |A Bimonthly, Peer Reviewed & Scholarly Journal|

||Volume 8, Issue 3, May-June 2025||

DOI:10.15662/IJRPETM.2025.0803015

4.4.1 Subject Identity Resolution

Subject identity resolution is a component of the architecture in its own right, because every cross-CRO join in gold and every ADaM population depends on it. It is described here architecturally. The simulation does not evaluate its accuracy, and the design below is a proposal that has not been implemented. Identity is handled in three levels of identifier (Table 7), resolved by an identity-resolution service with three tiers (Figure 5), and kept in a restricted zone so that the rest of the lakehouse sees only pseudonymous canonical identifiers.

Subject Identity Resolution in the Cross-CRO Harmonization Stage

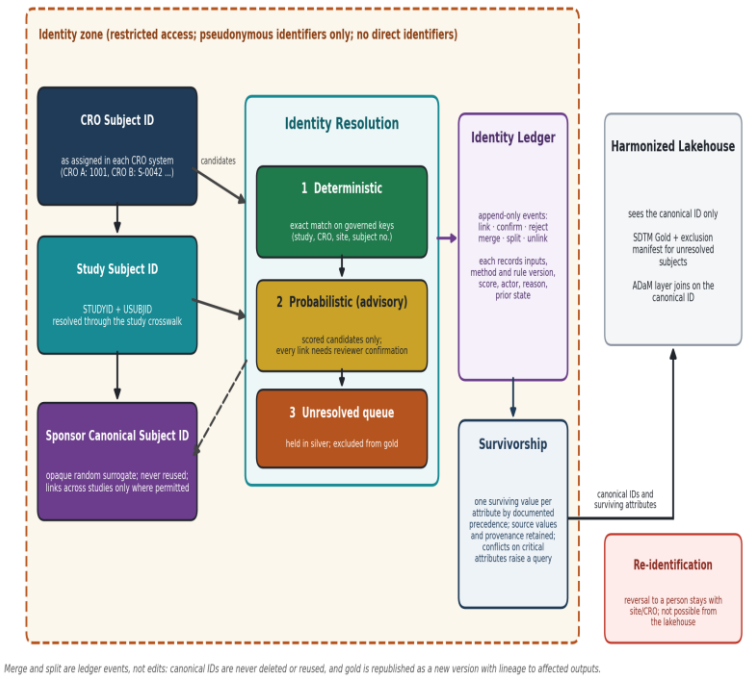


Table.7. Identifier hierarchy used in subject identity resolution.

Identifier level	Assigned by	Scope and uniqueness	Used for
CRO Subject ID	The CRO's own system	Unique only within that CRO's study; formats differ by CRO; the same string may denote different people at different CROs	Source key; never joined on its own
Study Subject ID	Sponsor-governed crosswalk (STUDYID with USUBJID)	Unique within the study; one per person per study	Joining SDTM domains across CROs within a study
Sponsor Canonical Subject ID	Identity service	Opaque random surrogate; unique across the sponsor's holdings; never reused or deleted	Gold SDTM and ADaM joins; links across studies only where permitted

Deterministic matching is the primary tier. A CRO Subject ID is mapped to a Study Subject ID through a crosswalk built from the study's enrolment and randomization records, which the sponsor is assumed to hold as its system of record (identifiers only, for example from randomization), and the match is exact on governed keys (study, CRO, site, and the CRO's subject number). The matching rules are versioned in the rule component and are applied identically to every CRO. The crosswalk key always includes the CRO and the study, so a bare identifier string is never used for a join. Most subjects are expected to resolve at this tier, although the proportion has not been measured.

|www.ijrpetm.com | ISSN: 2454-7875 | editor@ijrpetm.com |A Bimonthly, Peer Reviewed & Scholarly Journal|

||Volume 8, Issue 3, May-June 2025||

DOI:10.15662/IJRPETM.2025.0803015

Probabilistic matching is the second tier and is advisory only. Where no deterministic match exists, candidate links may be scored using attributes available in SDTM, such as site, sex, age, and key dates, with blocking to limit the comparison space and a probabilistic record-linkage model to weigh agreement and disagreement. In this design a score never creates a link by itself. Every probabilistic candidate is presented to a named reviewer, and the link exists only after confirmation. The probabilistic tier thus reduces reviewer effort without moving the decision out of human hands. Subjects with no confirmed match go to the third tier, an unresolved queue. They remain in silver, are excluded from gold, and appear in an exclusion manifest that gold publishes with each version, so that ADaM derivation and analysts can see which subjects are absent and why. Queue age and escalation follow the Validation Policies component.
Collision handling covers four cases. (1) Identifier collision: the same string is used by two CROs for different people. This is harmless because the crosswalk key includes the CRO. (2) Competing claims: two source identifiers map to one canonical identifier but their attributes conflict, for example in sex or birth year. Both links are held in conflict and neither is used until a reviewer decides. (3) Multiple identifiers for one person within a study, as after re-screening or replacement: linked only on confirmation, with the earlier identifier kept as an alias. (4) Identifier reuse: a canonical identifier is never reassigned, even after a split or a retirement.

Merges and splits are events, not edits. A merge declares that two canonical identifiers denote one person: one survives, and the other becomes a retired alias that resolves to the survivor. A split declares that one canonical identifier wrongly joined two people: new identifiers are issued and the affected source records are reassigned individually. Splits are the harder operation because every downstream record must be re-attributed, so they require approval by two reviewers, as rule changes do (Section 4.2.1). Both events are applied bitemporally, so the state of identity at any past date can be reconstructed. Because a merge or split changes the population behind published datasets, gold is republished as a new version, and the Lineage component lists the ADaM datasets and outputs built on the earlier one, using the same mechanism as a late correction (Section 4.7, scenario 10).

The audit trail is an append-only identity ledger within the Lineage component. Each event records its type (link, confirm, reject, merge, split, or unlink), the source identifiers involved, the method and rule version or score that produced it, the actor (the service or a named reviewer), the stated reason, the time, and the prior state. Nothing in the ledger is overwritten, so any canonical identifier can be traced back to the source identifiers and decisions behind it.

Survivorship governs which value a linked subject carries in gold when sources disagree on an attribute. A documented precedence per attribute is held in the Validation Policies component, with the sponsor's enrolment system ahead of CRO-derived values. Source values and their provenance are always retained. For critical attributes such as sex, date of birth or age, and treatment arm, a disagreement is not resolved by precedence but raised to the CRO as a query, since silently choosing a value would hide a data problem. Disagreements in formatting or terminology are normalized under the controlled-terminology mapping.

Privacy is addressed by separating identity from analysis. The identity service and its crosswalk sit in a restricted zone with role-based access limited to named identity stewards and every access logged. The lakehouse and the ADaM layer see only the canonical identifier. No direct identifiers (such as names or full birth dates) are required to flow into the identity step, which works from study-level pseudonymous identifiers and a minimal set of attributes. Canonical identifiers are random surrogates and not hashes of guessable inputs, so they cannot be regenerated from data. The design follows from the federated structure of Section 4.1: reversal from a study identifier to a person remains with the site and CRO that hold the source, and there is no path from the lakehouse to a named individual. Several re-identification controls follow. Linkage across studies is disabled by default and enabled per study pair only where protocols and consents permit it. The quasi-identifiers used by the probabilistic tier are limited and reviewed for the re-identification risk they add. Extracts that leave the lakehouse apply small-cell and rare-attribute suppression as defined by the sponsor's privacy assessment. Any break-glass reversal, for example for safety follow-up, goes through the CRO's own channel and not through the lakehouse. Which data-protection requirements apply (for example, GDPR or HIPAA, depending on jurisdiction) and whether this design satisfies them is for the sponsor's privacy and legal review; this paper makes no compliance claim.

4.5 SDTM Harmonization versus ADaM Derivation, Harmonization, and Traceability

SDTM and ADaM harmonization are not the same problem and the architecture treats them separately. SDTM datasets are tabulations of collected data: harmonizing them means making domains, variables, controlled terminology, supplemental qualifiers, identifiers, relationships between records, and mapping conventions consistent across CROs. ADaM datasets are analysis-ready datasets built from SDTM: harmonizing them also means making derivations,

|www.ijrpetm.com | ISSN: 2454-7875 | editor@ijrpetm.com |A Bimonthly, Peer Reviewed & Scholarly Journal|

||Volume 8, Issue 3, May-June 2025||

DOI:10.15662/IJRPETM.2025.0803015

analysis variables, analysis populations, parameterization, and derived endpoints consistent, and keeping each derived value traceable to the SDTM data it came from (CDISC, n.d.-a). ADaM therefore adds statistical-programming dependencies - on the statistical analysis plan, the analysis population definitions, and the derivation code itself - that have no SDTM counterpart. Table 8 summarizes the distinction as the architecture applies it.

Table.8. How the architecture distinguishes SDTM harmonization from ADaM derivation, harmonization, and traceability.

Dimension	SDTM conformance / harmonization	ADaM derivation / harmonization / traceability
Unit of work	Domains and variables (e.g., DM, AE, LB, VS)	Analysis datasets: ADSL (subject level), BDS-type (e.g., ADLB, ADVS), occurrence-type (e.g., ADAE)
Main sources of cross-CRO divergence	CT version, supplemental-qualifier conventions, identifiers, mapping conventions	Derivation logic, analysis-population flags, parameter naming and coding, derived-endpoint definitions
What the gate checks	Schema, CT version, required variables, naming rules against Define-XML	Structure against ADaM specifications; population-flag and parameter definitions against the sponsor standard; resolvability of every source reference
Traceability requirement	Lineage from delivery to source system	Each analysis variable traceable to SDTM variables, and onward to source
Governing artifact in metadata plane	Schema Versions, Rule Versions, CT Versions, Validation Policies	ADaM specification (Schema Versions), shared derivation library (Rule Versions), and Lineage
Principal dependencies	CRO mapping practice; sponsor CT policy	Statistical analysis plan; statistical programming; validated derivation code
Evaluated in this paper	**Gate detection timing and cost (simulation, Sections 5-6)**	**Not evaluated; architectural specification only**

The ADaM layer sits above the gold SDTM layer in the harmonized lakehouse (Figures 1 and 3) and is governed by two metadata-plane artifacts: sponsor-defined ADaM specifications (dataset structure, parameter and population definitions) and a shared, version-controlled derivation library. Two supply modes are possible and are not mutually exclusive. In the first, CROs deliver ADaM datasets built in their own environments; a boundary gate checks structure against the sponsor specification, compares population flags and parameter coding with the sponsor standard, and verifies that every source reference resolves to SDTM data at the version used. In the second, the sponsor derives ADaM datasets from harmonized gold SDTM using the shared derivation library, which removes cross-CRO

|www.ijrpetm.com | ISSN: 2454-7875 | editor@ijrpetm.com |A Bimonthly, Peer Reviewed & Scholarly Journal|

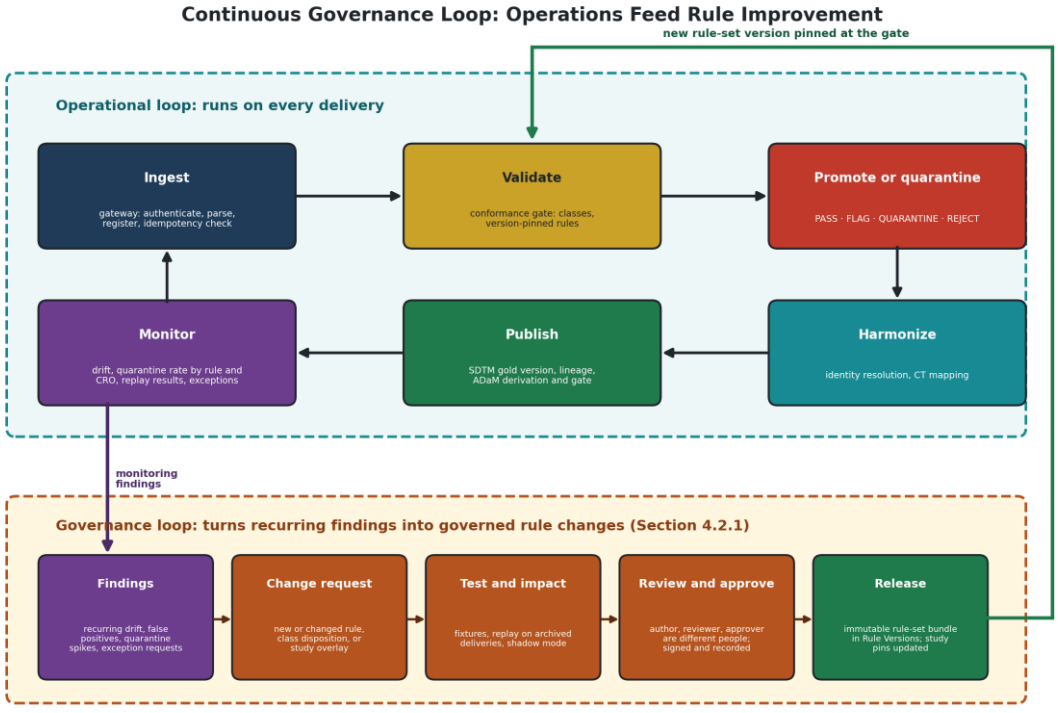
||Volume 8, Issue 3, May-June 2025||

DOI:10.15662/IJRPETM.2025.0803015

divergence in derivation logic by construction but moves programming and validation responsibility to the sponsor. This paper does not recommend one mode over the other, because no evidence was gathered to compare them. In either mode the lineage catalog records, for each analysis variable, the derivation version and the SDTM variables and dataset versions it used, so that a change in a harmonized SDTM dataset can be traced forward to the analysis datasets it affects.

4.6 Operational Lifecycle and Continuous Governance

The architecture in Sections 4.1-4.5 is not a one-time build; it is intended to run as two coupled loops across the life of a multi-CRO program. Figure 4 depicted the mechanism at the level of a single delivery; Figure 6 depicts the recurring cycles it is intended to support. The operational loop runs on every delivery: ingest, validate, promote or quarantine, harmonize, publish, and monitor. The governance loop runs when monitoring produces findings worth acting on: findings become change requests, are tested by fixtures, replay, and shadow mode, are reviewed and approved by separate people, and are released as a new immutable rule-set bundle that studies then pin at the gate (Section 4.2.1). Monitoring findings thus feed the conformance-rules library through a controlled path, and the cycle does not reset to a blank rule set each time a new CRO or study is onboarded.



The loop is the mechanism by which detection coverage is meant to improve over a program's life rather than stay fixed at the rules authored at go-live.

This framing matters for a practical reason: a conformance-rules library that only ever grows by reacting to individual quarantine events will plateau in coverage. The monitor-and-feed-back step is where recurring, cross-study drift patterns (e.g., a specific CRO's recurring supplemental-qualifier convention) are promoted from one-off exceptions into standing rules, which is the mechanism by which the architecture's detection coverage is intended to improve over a program's lifetime rather than staying fixed at whatever rules were authored at go-live.

4.7 Failure Modes and Architectural Response

An architecture is better judged by how it behaves when something goes wrong than by its normal path. Table 9 walks through ten failure scenarios: four that originate in CRO practice, two in the architecture's own components, one in cross-CRO harmonization, and three in delivery logistics. Each is stated with the point at which it would be detected, the proposed response, and the risk that remains. The walk-throughs are analytical: none has been run against an implementation or modeled in the simulation of Section 5, and the responses follow from the components and policies defined in Sections 4.1-4.6.

|www.ijrpetm.com | ISSN: 2454-7875 | editor@ijrpetm.com |A Bimonthly, Peer Reviewed & Scholarly Journal|

||Volume 8, Issue 3, May-June 2025||

DOI:10.15662/IJRPETM.2025.0803015

Table.9.Failure scenarios, proposed detection and response, and residual risk. Design analysis, not tested

#	Scenario	Detection	Response and containment	Residual risk
1	CRO submits incorrect Define-XML	Define-XML component validates the submission (well-formedness, agreement with registered schema versions) before it is used as a reference	Submission quarantined; dependent deliveries held as pending metadata, not evaluated against unapproved metadata	Well-formed but semantically wrong definitions pass; depends on sponsor review
2	Correct structure, incorrect terminology	Controlled-terminology class check against the declared CT version	Domain quarantined; exception report to the CRO	Valid but wrong terms (a legal value that is clinically wrong) are not detectable
3	Mapping convention changed without notification	Drift monitors: schema fingerprint, value-set and record-count comparison with the study's prior deliveries	Delivery flagged and held for review when drift exceeds the policy threshold; change request raised with the CRO	Semantic changes that leave structure and distributions intact can go undetected
4	Rule engine contains an incorrect rule	Quarantine-rate monitoring per rule across CROs (false positives); replay and audit (false negatives)	Study pins reverted to the prior rule-set version; affected decisions identified by version and re-evaluated by replay	False negatives persist until replay or downstream discovery
5	Controlled-terminology component unavailable	Gate health check on the component	Fail closed: deliveries wait in landing; gates continue on the locally cached, pinned (immutable) CT version	Delay; new CT versions cannot be adopted during the outage
6	Two CROs use different legitimate CT versions	Declared version tested against the study's permitted versions, not against a single version	Each delivery validated on its declared version in silver; normalized to the reference version at harmonization using registered mapping tables	Unmappable or many-to-one terms; mapping-table errors
7	Subject identity cannot be resolved	Cross-CRO harmonization identity step	Unresolved subjects held in silver with an identity-exception queue; gold published for resolved subjects with an explicit manifest of exclusions; no automatic merge on uncertain matches	False merges if review is poor; exclusions bias analyses if unreviewed
8	Duplicate delivery arrives	Delivery key plus content hash in the Dataset Contracts component	Identical content: acknowledged, logged, no change. Same key, different content: held as a conflict for sponsor decision	Re-exports that differ in bytes but not in meaning are treated as new versions
9	Partial domain delivery arrives	Completeness check against the contract's manifest (expected datasets, record counts, checksums)	Delivery held as incomplete in landing; last complete version stays current; no partial promotion	Without a manifest, truncation is detectable only by record-count drift
10	Correction arrives after gold publication	Delivery key matches a published version	Passes the same gate; new silver and gold versions published as new snapshots; prior versions remain readable; lineage identifies ADaM and TFL outputs built on the superseded version	Outputs already reported or submitted need a controlled process beyond the architecture

|www.ijrpetm.com | ISSN: 2454-7875 | editor@ijrpetm.com |A Bimonthly, Peer Reviewed & Scholarly Journal|

||Volume 8, Issue 3, May-June 2025||

DOI:10.15662/IJRPETM.2025.0803015

Five design principles recur across the scenarios. First, the architecture fails closed: when a check cannot be run, as when the controlled-terminology component is unreachable (scenario 5), the delivery waits and is never passed with the check skipped. Second, nothing is deleted; bronze is append-only, so quarantine, conflict, and supersession all preserve the original delivery. Third, every decision records the rule-set, schema, and CT versions that produced it, which is what makes an incorrect rule (scenario 4) or a late correction (scenario 10) recoverable by replay and impact analysis rather than by manual reconstruction. Fourth, ingestion is idempotent, keyed by delivery identity and content hash (scenario 8). Fifth, ambiguity goes to a person: identity resolution and mapping conflicts are queued for review and not settled by heuristics.

Scenarios 1 and 2 test the boundary between structure and meaning. A CRO-submitted Define-XML is itself a delivery and is validated before it becomes the reference against which datasets are checked; the sponsor-approved copy, not the CRO's raw submission, is what gates use. Terminology checks (scenario 2) catch values outside the declared CT version but cannot judge whether a legal value is correct for the subject. The gate therefore establishes conformance, not data correctness, a limitation that applies to all classes in Table 6.

Scenario 3 is the hardest to contain. A change in a mapping convention that the CRO does not announce is visible only if it alters structure, value sets, or volumes. The proposed response combines a contractual obligation to notify (recorded in the CRO component) with drift monitors that compare each delivery to the study's earlier deliveries and hold it for review when the difference exceeds the policy threshold. A change that preserves structure and distributions, for example a different but plausible derivation rule, would pass, and periodic reconciliation to source would be needed to find it.

Scenarios 4 and 5 concern the architecture's own components. An incorrect rule can fail in two ways. If it rejects valid data, quarantine rates for that rule rise across several CROs at once, which is a recognizable signature; the response is to repin studies to the previous rule-set version. If it admits invalid data, the failure is silent, and because gate decisions are tied to rule-set versions, affected deliveries can be identified and re-evaluated once the fault is found. The shadow-mode and fixture requirements of Section 4.2.1 are intended to make such rules rarer but cannot eliminate them. For an unavailable terminology component, the immutability of released CT versions is what makes a local cache safe: a pinned version cannot change underneath the gate, so validation can continue in degraded mode, whereas adopting a new version must wait for the component to return.

Scenarios 6 and 7 concern multi-CRO harmonization. Two CROs working to different legitimate CT versions do not represent an error, so the conformance rule tests membership in the study's permitted versions rather than equality with a single version. Each delivery is validated on the version it declares, and normalization to the sponsor reference version happens at the harmonization stage using registered mapping tables; terms that cannot be mapped are held there, not at the gate. When a subject's identity cannot be resolved, gold is published for the resolved subjects together with a manifest of the unresolved ones, so exclusions are visible to ADaM derivation and to analysts. Uncertain matches are queued for review and never merged automatically, because a false merge combines two people's data and is harder to detect and reverse than an unresolved subject.

Scenarios 8 to 10 concern delivery logistics. An identical redelivery is acknowledged and logged; same key with different content is a conflict for the sponsor to decide and never silently overwrites the earlier delivery. A partial domain is detected against the contract manifest and held in landing. A late correction is a new version that passes the same gate, produces new silver and gold snapshots, and leaves earlier versions readable; Lineage lists the ADaM datasets and outputs built on the superseded gold version. Whether a correction after database lock is permitted is a study decision outside the architecture.

4.8 Regulatory and Validation Architecture

The architecture would produce records that support regulatory submissions, so it has to be governed and validated as a system as well as designed as a pipeline. This subsection states the controls the proposal assumes. It is organized around the expectations that regulators and quality auditors apply to electronic systems in clinical research: that systems be validated to ensure accuracy, reliability, and the ability to discern invalid or altered records, that access be limited to authorized individuals, and that secure, time-stamped audit trails record operator actions without obscuring earlier entries and be retained as long as the records they concern (21 C.F.R. pt. 11, 2026, § 11.10), and that computerised systems used in trials be fit for purpose, with relevant metadata and audit trails managed accordingly (International Council for Harmonisation, 2025). These sources are used here to organize the controls. They are not a

|www.ijrpetm.com | ISSN: 2454-7875 | editor@ijrpetm.com |A Bimonthly, Peer Reviewed & Scholarly Journal|

||Volume 8, Issue 3, May-June 2025||

DOI:10.15662/IJRPETM.2025.0803015

compliance claim: whether a given implementation satisfies them, and which of them apply to a particular study, would be determined by the sponsor's quality system.

Figure 7 gives the structure. Governance sits above two branches that both act on the production lakehouse and control plane. The security branch (identity and access management, encryption, and audit logging) governs who can do what and records it. The validation branch (rule and transformation validation, change control, and release approval) governs what is allowed to run. Environment separation and resilience controls apply to the whole. Table 10 states each control area and where it is realized.

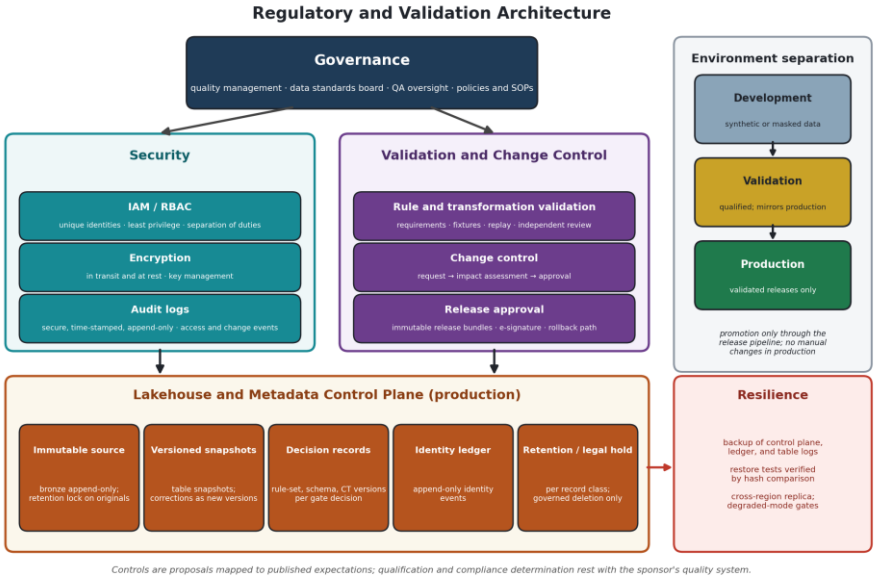


Table.10. Regulatory and validation controls of the proposed architecture. Design proposals, not an implemented or qualified system.

Control area	Proposed architectural control	Realized in
Audit trail	Secure, system-generated, time-stamped, append-only records of gate decisions, rule changes, exceptions, identity events, and access; changes never obscure earlier entries; retained as long as the records they describe	Lineage; identity ledger; security logs
Immutable source data	Bronze is append-only and holds every delivery as received, including quarantined ones; retention lock on originals; corrections arrive as new versions	Bronze layer; table snapshots
Electronic records	Gate decisions, approvals, and exceptions are managed as electronic records; approvals carry the signer, time, and meaning and are bound to the record, using the sponsor's existing signature and identity service	Rule Versions; Lineage; sponsor quality system
Role-based access control	Named roles with least privilege (for example rule author, reviewer, release approver, identity steward, CRO delivery user, analyst, platform administrator); a CRO role sees only its own boundary; periodic access review	IAM service; all components
Separation of duties	Author, reviewer, and approver of a rule are different identities; identity splits need two reviewers; an exception approver is not its requester; platform administrators cannot approve releases	Rule governance (Section 4.2.1); identity service
Validation of rules	Risk-based: each rule has a stated requirement, known-pass and known-fail fixtures, replay against archived deliveries, and recorded results before it can become effective	Rule Versions; validation environment
Validation of transformations	Harmonization mappings, CT mapping tables, and ADaM derivations follow specification, implementation, independent	Harmonization and ADaM layers

|www.ijrpetm.com | ISSN: 2454-7875 | editor@ijrpetm.com |A Bimonthly, Peer Reviewed & Scholarly Journal|

||Volume 8, Issue 3, May-June 2025||

DOI:10.15662/IJRPETM.2025.0803015

	verification, and regression against expected outputs before release	
Controlled change management	Change request, impact assessment by replay, approval, release in the validation environment, then promotion; configuration and infrastructure changes follow the same path; emergency changes get retrospective review	Release pipeline; Governance
Rule-version traceability	Every gate decision records the rule-set, schema, and CT versions in force, and every study records the bundle it pins	Version and Lineage records
Reproducibility	Pinned versions, deterministic jobs, and readable earlier snapshots allow a past decision to be re-run; periodic reproduction tests compare hashes of re-run outputs with stored ones	Table snapshots; replay service
Data retention	Retention schedule per record class set by sponsor policy and applicable regulation; legal hold; deletion only through a governed process that is itself recorded	Retention policy; storage controls
Backup and recovery	Backups of the control plane, identity ledger, and table transaction logs held separately from the primary store (snapshots are not backups); restores tested and verified by hash comparison; recovery objectives set higher for the control plane and ledger than for derived layers	Backup service
Disaster recovery	Cross-region replication of bronze and the control plane; documented failover; gates continue in degraded mode on cached pinned versions (Section 4.7, scenario 5); lineage replay checks consistency after recovery	Replica; runbooks
Environment separation	Development, validation, and production are separate; development uses synthetic or masked data; the validation environment mirrors production; gate packages for CRO environments come from the same release	Release pipeline; accounts and networks

Validation is risk-based. Validating a rule means more than testing the engine that runs it. The engine, as software, is validated as a system once per version, while each rule's content is validated through its requirement, fixtures, and replay results, and the evidence is stored with the rule version. Validation effort is scaled to the impact of the rule: a hard-failing rule that can quarantine a delivery warrants more evidence than a naming flag. Environment separation supports this, because rules and transformations are qualified in an environment that mirrors production before promotion, and no production data is used in development.

Resilience controls follow from the asymmetry of the layers. Derived layers can be rebuilt from bronze and the control plane, but bronze, the control plane, and the identity ledger cannot be rebuilt from anything else, so they receive the strictest backup, replication, and restore-testing requirements. Table snapshots support reproducibility but do not replace backups, since they live in the same store. A restore is considered successful only when re-computed hashes match those recorded before the failure. After a regional recovery, lineage replay is used to confirm that the layers are consistent with one another. The recovery objectives themselves are not specified here because they depend on operational and contractual factors outside this proposal.

4.9 Non-Functional Requirements

A pipeline design becomes an architecture only when it is stated against the qualities it must exhibit. Table 11 lists twelve non-functional requirements, the architectural mechanism that is meant to satisfy each, and how each would be verified. The requirements are written in parameterized form. Symbols such as N, M, the latency bounds, and the recovery objectives are to be set by the adopting sponsor from its portfolio, risk, and contractual position; this paper does not propose values for them because there is no basis in the evidence for doing so. The requirements are design targets. No implementation exists, and none of them has been measured.

|www.ijrpetm.com | ISSN: 2454-7875 | editor@ijrpetm.com |A Bimonthly, Peer Reviewed & Scholarly Journal|

||Volume 8, Issue 3, May-June 2025||

DOI:10.15662/IJRPETM.2025.0803015

Table.11. Non-functional requirements of the proposed architecture. Requirements and mechanisms are design proposals; parameter values are to be set by the adopting sponsor.

Requirement	Statement	Architectural mechanism	Verification
Scalability	Support N CROs and M studies concurrently, and a stated deliveries-per-day load, without redesign; N, M, and load come from the sponsor's portfolio plan	Gates run per CRO boundary and share no state with one another; storage is partitioned by CRO, study, and delivery date; rules are evaluated in parallel by domain; gates read pinned versions from a cached copy of the control plane	Load test at twice the planned N, M, and daily load; throughput and latency against deliveries per day
Availability	A defined SLA for the gate service and for the control plane, expressed as monthly availability targets set by the sponsor	Gates continue on cached pinned versions during a control-plane outage (Section 4.7, scenario 5); a gate failure holds deliveries instead of losing or passing them; the control plane is replicated across zones	Availability measured by synthetic probes; scheduled outage drills
Performance	Maximum ingestion-to-validation latency: the 95th percentile time from receipt of a delivery to its disposition (accept, flag, quarantine, or reject) must not exceed a stated bound for deliveries up to a stated size; a second bound applies to gold publication	Validation at the boundary before landing; parallel evaluation per domain; incremental harmonization of accepted domains only	Latency distribution per delivery size class
Security	Encryption in transit and at rest; role-based access with least privilege; CRO isolation; identity data held in a restricted zone	IAM and key management (Section 4.8); identity zone (Section 4.4.1); each CRO role confined to its own boundary	Periodic access review; security testing under the sponsor's policy
Auditability	Full lineage and an immutable audit trail: every gate decision, rule change, exception, identity event, and access is recorded and retained	Append-only bronze, decision records with rule/schema/CT versions, identity ledger, secure logs (Sections 4.4.1 and 4.8)	Sampled trace-back from an ADaM variable to the source delivery must succeed in every audit test
Recoverability	Recovery point and recovery time objectives set per component class: control plane and identity ledger, bronze, and derived layers (rebuildable from the first two)	Separate backups; cross-region replication; restore tests verified by hash comparison (Section 4.8)	Scheduled restore and failover exercises against the stated RPO and RTO
Data quality	Validation coverage: the share of in-scope SDTM domains and variables covered by at least one effective rule in each validation class (Table 6), with coverage and quarantine-rate targets set per study	Rule Versions and Dataset Contracts compared with the study's Define-XML variable list; class-based dispositions; monitoring of quarantine rates by rule and CRO	Coverage report per study and release; false-positive and quarantine-rate trends

|www.ijrpetm.com | ISSN: 2454-7875 | editor@ijrpetm.com |A Bimonthly, Peer Reviewed & Scholarly Journal|

||Volume 8, Issue 3, May-June 2025||

DOI:10.15662/IJRPETM.2025.0803015

Extensibility	Onboard a new CRO, study, domain, validation class, or standards version by configuration and governed rule changes, without changing platform code	CRO Metadata, Dataset Contracts, and a versioned gate package deployed to the CRO environment; rule tiers and study overlays (Section 4.2.1)	Onboarding of a new CRO using a conformance test delivery; elapsed onboarding time
Maintainability	A versioned rule engine and versioned rules, mappings, and derivations that can be changed, tested, and rolled back independently	Rules, mappings, and derivations as version-controlled configuration or code; immutable release bundles; automated regression suites; upgrades through the release pipeline	Regression suite pass rate per release; time to roll back a release
Compliance	Computer system validation and GxP-consistent controls, determined by the sponsor's quality system	Validation, change control, electronic records, and environment separation (Section 4.8)	Validation evidence per release; audit and inspection readiness review
Observability	Metrics, logs, and traces sufficient to see the state of every delivery and the health of every component	Metrics for dispositions by class, quarantine rate by rule and CRO, latency, queue ages, and replay results; structured logs; traces keyed by delivery identifier from gate to ADaM; alerts on drift and rule-level quarantine spikes (Section 4.7, scenarios 3 and 4)	Dashboards and alert tests; trace completeness check for a sample of deliveries
Reproducibility	Deterministic dataset versions: the same inputs under the same pinned versions yield identical outputs	Pinned rule, schema, CT, and mapping versions; readable snapshots; deterministic jobs; random choices such as canonical subject identifiers are drawn once and stored in the identity ledger so re-runs reuse them	Periodic re-run of a past delivery with a hash comparison against the stored output

Several requirements pull against each other, and the design resolves them explicitly. Refusing to pass a delivery whose checks cannot run (the fail-closed principle of Section 4.7) lowers apparent availability, so the gate keeps running on cached, pinned versions and accepts delay only when a genuinely new version is required. Broader validation coverage lengthens validation time, so coverage is governed per class and study. Immutability and retention may conflict with erasure obligations in some jurisdictions; holding only pseudonymous identifiers (Section 4.4.1) narrows but does not remove the conflict, which is a matter for the sponsor's privacy and legal review. The metrics of Section 6.1 (Table 14) would monitor several of these requirements in operation.

Two of the requirements connect to results elsewhere in this paper. The scalability requirement is the axis along which the break-even analysis of Section 6.3 varies, in studies and deliveries per year, but the simulation of Section 5 covers only 14 studies and 5 CROs, and no scalability or performance test has been run. The data-quality requirement is the one the simulation addresses, and only in part: it models detection timing and resolution cost for ten SDTM domains and does not measure validation coverage.

V. SIMULATION DESIGN AND SYNTHETIC DATASET

This section describes a fully disclosed, reproducible simulation study conducted to illustrate the conformance gate's intended mechanism. No part of this section describes a real system, a real CRO, a real study, or measured field data; all figures reported here and in Section 6 are outputs of the generative model and cost procedure described below, and the complete generation code and output dataset are provided as supplementary files accompanying this manuscript.

|www.ijrpetm.com | ISSN: 2454-7875 | editor@ijrpetm.com |A Bimonthly, Peer Reviewed & Scholarly Journal|

||Volume 8, Issue 3, May-June 2025||

DOI:10.15662/IJRPETM.2025.0803015

5.1 Purpose and Scope

The simulation's purpose is narrow: to illustrate, under a transparent and fully stated set of modeling assumptions, the mechanism by which an ingestion-time conformance gate is expected to change the timing and resolution cost of conformance issues relative to downstream, manual detection. The simulation does not model harmonization accuracy, subject-identity resolution, query or retrieval performance, or implementation cost, and should not be read as evidence of the magnitude of benefit a real deployment would achieve.

5.2 Data-Generating Model

A synthetic population of 5 CROs and 14 studies was generated over a 24-month horizon, with each study active for a randomly assigned 8–24 month window and served by one primary CRO. In each active study-month, a delivery of 3–9 SDTM domains (drawn from a 10-domain list: DM, AE, LB, VS, EX, CM, MH, DS, EG, SU) is simulated. For each delivered domain, a conformance issue occurs independently with probability 0.14 (an illustrative parameter, not fitted to a reported real-world defect rate, since no published schema-drift incidence rate specific to multi-CRO SDTM delivery was located during the literature review). Under the simulated baseline (Scenario A), an issue is discovered after a delay drawn from a lognormal distribution (median 9 days) and treated as equivalent in resolution effort to one query-cycle event. Under the simulated proposed architecture (Scenario B), the same underlying issue occurrence is reused, and each issue is independently caught at ingestion with probability 0.80 (illustrative); caught issues incur near-zero discovery delay and a reduced resolution effort, reflecting a structured, machine-generated exception report rather than a full manual query cycle.

5.3 Illustrative Cost Inputs

The simulation does not model federation as such: it treats the gate as an ingestion-time check and does not represent the CRO-side deployment effort, metadata-plane governance, or network and access controls that a federated topology introduces (Section 7). Resolution cost per issue is drawn from an author-chosen illustrative range of USD 28–225. This range is an assumption and is not taken from any cited source; it is intended only to set the order of magnitude of a cross-functional resolution effort and has not been validated against any organization's data. For issues caught at ingestion under the proposed architecture, a discount of 80% relative to the published low-end query cost is applied, reflecting the reduced cross-functional back-and-forth expected for a structured exception report; this discount factor is an author assumption with no independent published citation and is reported as such.

Table.12. Composition of the simulation study and illustrative cost basis (simulation output; cost inputs are author assumptions).

Simulation property	Value
Simulated domain-delivery records	1,247
Simulated CROs	5
Simulated studies	14
Simulated conformance issues generated	169 (13.55% of deliveries)
Random seed (reproducible)	20260816
Unit-cost input	Author-chosen illustrative range, USD 28-225 per issue (uncited)

|www.ijrpetm.com | ISSN: 2454-7875 | editor@ijrpetm.com |A Bimonthly, Peer Reviewed & Scholarly Journal|

||Volume 8, Issue 3, May-June 2025||

DOI:10.15662/IJRPETM.2025.0803015

VI. SIMULATION RESULTS AND DISCUSSION

All values in this section are outputs of the simulation described in Section 5, computed directly from the generated synthetic dataset by descriptive aggregation over the 1,247 simulated records, with the dollar figures obtained by applying the author-chosen illustrative cost range to the simulated issue counts as described in Section 5.3.

USD 28-225 **ASSUMED UNIT COST** Author assumption, uncited	**-82.9%** **SIM. DISCOVERY DELAY** Simulation output	**82.3%** **SIM. ISSUES CAUGHT AT INGESTION** Simulation output	**-65.8%** **SIM. RESOLUTION COST** Simulation output

Table.13. Simulated outcomes, Scenario A vs. Scenario B, with illustrative resolution cost inputs.

Metric	Scenario A (baseline)	Scenario B (proposed)	Change
Simulated conformance issues, total	169	169 (same issue population)	—
Issues caught at ingestion	0 (not modeled)	139 (82.3% of issues)	—
Mean discovery delay (days)	**10.37**	**1.77**	**−82.9%**
Total resolution cost, low bound	**USD 4,732**	**USD 1,618**	**−65.8%**
Total resolution cost, high bound	USD 38,025	USD 13,005	**−65.8%**
Total simulated DCR-equivalent count	171	24	**−86.0%**

The cost reduction percentage is identical at the low and high assumed cost bounds (−65.8%) because the per-unit cost cancels out of the ratio between scenarios under the model's linear cost structure (Section 5.3); this is a property of the calibration method, not an empirical finding. Its magnitude follows directly from the assumed 80% catch probability and 80% cost discount, which Section 6.2 therefore varies across a grid. The simulation shows only that the proposed mechanism (earlier detection, lower per-issue resolution cost, on an assumed unit-cost basis) is internally consistent and computable, not that a deployment will produce a reduction of this magnitude.

> ▸ **PRACTICAL IMPLICATION**
> The simulation's most transferable finding is not the specific percentage reduction but the mechanism: an ingestion-time gate converts a class of issue that currently surfaces near database lock — when schedule pressure and rework cost are both highest — into one that surfaces same-day, when it is cheapest and least disruptive to fix. Data management leads evaluating this architecture should weigh that timing shift on its own merits, independent of the illustrative dollar figures in Table 13.

6.1 Architecture Metrics

The four headline measures above (discovery delay, issues caught, resolution cost, and DCR-equivalent count) describe the simulated process. A reviewer evaluating the architecture itself needs measures of how the architecture performs as a system. Seven are defined in Table 14, each with its formula, the unit, the source in the architecture from which it would be computed, and whether the present simulation produces it. The false positive rate as defined here is the proportion of quarantines that were wrong, which some texts call the false discovery rate; it is not the conventional rate of false positives among all valid deliveries, and the definition is stated so that the two are not confused.

|www.ijrpetm.com | ISSN: 2454-7875 | editor@ijrpetm.com |A Bimonthly, Peer Reviewed & Scholarly Journal|

||Volume 8, Issue 3, May-June 2025||

DOI:10.15662/IJRPETM.2025.0803015

Table.14. Architecture metrics: definitions, data sources, and simulation status.

Metric	Definition	Unit	Measured from	Simulation status
Detection Coverage (DC)	Issues detected by the gate / total detectable issues	ratio	Gate decision records; issues later found downstream and traced to the delivery	Simulated, but driven by an input parameter
False Positive Rate (FPR)	Incorrectly quarantined deliveries / total quarantined deliveries	ratio	Quarantine records and their reviewed outcome (confirmed issue or released)	Not simulated
Mean Time to Detect (MTTD)	Mean elapsed time from receipt of a delivery to detection of an issue in it	days	Delivery receipt time and detection time in Lineage	Simulated
Mean Time to Resolve (MTTR)	Mean elapsed time from detection to acceptance of a corrected delivery	days	Exception report time and acceptance time of the superseding version	Not simulated; effort hours reported as a proxy
CRO Onboarding Time (Tonboard)	Working days from registration in CRO Metadata to the first conformance-test delivery accepted without hard failures	days	CRO Metadata and Dataset Contracts event times, per CRO	Not simulated
Rule Reuse Ratio (RR)	Reusable rules / total rules, where reusable means a sponsor-tier rule applied unchanged in at least two studies	ratio	Rule Versions and the study pins in Study Metadata	Not simulated
Validation Coverage (VC)	Applicable requirements automatically validated / total applicable requirements	ratio	Rule Versions and Dataset Contracts compared with the study's Define-XML and sponsor specifications	Not simulated

Only three of the seven can be obtained from the simulation, and they differ in how much they show. Mean Time to Detect follows from the delay distributions of Section 5.2 and is the same quantity as the mean discovery delay reported in Table 13. Detection Coverage is computed from the simulated catch flags, but the simulation draws those flags from an assumed catch probability of 80%, so the realized value is a statement about sampling variation around that input and not a result about the gate; it is shown only to complete the set. The simulation records resolution effort in hours and not elapsed time to resolve, so the hours per issue are reported as a proxy for Mean Time to Resolve and should not be read as an elapsed time. In the base run, Detection Coverage was 82.2% for Scenario B (80.1%, 5th-95th percentile 75.0-85.1%, over 500 replications); Mean Time to Detect was 10.37 days for A and 1.77 days for B (500-replication means 10.20 and 2.14); and resolution effort was 2.60 hours per issue for A and 0.79 for B (500-replication means 2.60 and 0.84). These are simulation outputs under author-chosen parameters, not measurements.

The remaining four metrics (False Positive Rate, CRO Onboarding Time, Rule Reuse Ratio, and Validation Coverage) require an operating system and are not addressed by the simulation; they bear most directly on whether the architecture is practical, and each should be reported with its underlying counts, per study and per CRO.

6.2 Sensitivity Analysis: Operating Envelope

The base-case result rests on three author-chosen parameters, so the simulation was repeated over a grid: ingestion catch rate (50%, 60%, 70%, 80%, 90%, 95%), baseline issue rate per domain delivery (5%, 10%, 15%, 20%, 30%), and the resolution-cost discount applied to issues caught at ingestion (20%, 40%, 60%, 80%), giving 120 combinations. Each combination was run for 500 replications using the same population structure as Section 5.2 (14 studies, 3-9 of ten domains per delivery, on average 1,153 domain deliveries per replication) with seed 20260816 and common random numbers across combinations. Cost reduction is defined as in Section 5.3: baseline cost is the unit cost for every issue, and under the gate a caught issue costs the unit cost reduced by the discount while an uncaught issue costs the full unit

|www.ijrpetm.com | ISSN: 2454-7875 | editor@ijrpetm.com |A Bimonthly, Peer Reviewed & Scholarly Journal|

||Volume 8, Issue 3, May-June 2025||

DOI:10.15662/IJRPETM.2025.0803015

cost. Figure 8 and Table 15 report the results; the red box marks the grid cell nearest the base case (80% catch, 15% issue rate against the 14% used in Section 5, 80% discount). All values are simulation outputs.

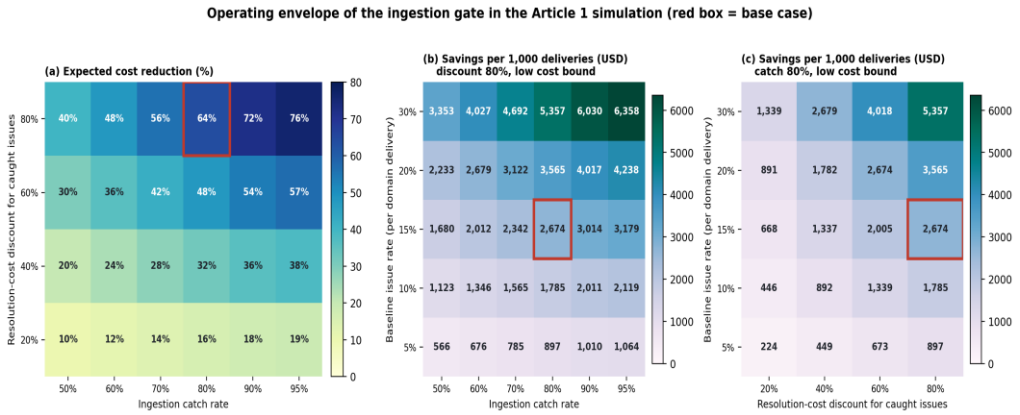


Table.15. Mean simulated cost reduction (%) by ingestion catch rate and caught-issue resolution-cost discount (issue rate 15%; mean of 500 replications). Values are simulation outputs.

Caught-issue discount \ Catch rate	50%	60%	70%	80%	90%	95%
20%	10.0%	12.0%	14.0%	16.0%	18.0%	19.0%
40%	20.1%	24.0%	28.0%	32.0%	36.0%	38.0%
60%	30.1%	36.1%	42.0%	47.9%	54.0%	57.0%
80%	40.1%	48.1%	56.0%	63.9%	72.0%	76.0%

Three results follow. First, because the model's cost structure is linear, the expected percentage reduction equals the product of catch rate and discount (for example 0.80 x 0.80 = 64%, against a simulated mean of 63.9%); it does not depend on the issue rate or on the unit cost. The heat map in panel (a) is therefore a closed-form surface that the Monte Carlo runs confirm, not an independent discovery. Across the grid the reduction ranges from 10% (50% catch, 20% discount) to 76% (95% catch, 80% discount); it reaches 50% or more in 6 of the 24 catch-by-discount combinations (catch of 70% or more with an 80% discount, or catch of 90% or more with a 60% discount), and stays below 25% in 8 of them, all of which have a discount of 40% or less. Within this model the discount matters as much as the catch rate: a gate that catches 95% of issues but lowers their resolution cost by only 20% yields about 19%, less than a gate that catches 50% at an 80% discount (40%).

Second, the issue rate governs the absolute benefit and the variability of the result, not the percentage. At the low published unit cost (USD 28), expected savings at 80% catch and an 80% discount rise from about USD 897 per 1,000 domain deliveries at a 5% issue rate to about USD 5,357 at 30% (panels b and c); at the high published unit cost (USD 225) the figures are roughly eight times larger. At the same time, the 5th-95th percentile interval of the simulated reduction at 80% catch and 80% discount narrows from 57-71% at a 5% issue rate to 61-67% at 30%, because fewer issues make the realized catch fraction less stable over a program of this size.

Third, the envelope is conditional because the model contains no cost for operating the gate: every grid cell shows a positive reduction, which is not evidence that the architecture pays for itself (Section 6.3 adds placeholder costs), and neither axis is validated by the grid.

6.3 Break-Even Analysis

Section 6.2 shows the savings a gate can generate but not what the architecture costs. This section sets annual cost against annual avoided cost to find the program size at which the two meet. It is an illustrative scenario analysis, not an economic evaluation. No input is drawn from the literature; the unit resolution cost of USD 28-225 is itself an author-chosen assumption. Every cost input in Table 16 is an author-chosen placeholder, set to span roughly an order of

|www.ijrpetm.com | ISSN: 2454-7875 | editor@ijrpetm.com |A Bimonthly, Peer Reviewed & Scholarly Journal|

||Volume 8, Issue 3, May-June 2025||

DOI:10.15662/IJRPETM.2025.0803015

magnitude and not taken from any vendor, sponsor, or published source, and the avoided-cost components other than query resolution have no empirical support in this paper. The model is therefore best used as a template into which an organization inserts its own figures.

Annual cost is the sum of six items: platform operation, rule development (amortized over three years), rule maintenance, cloud and storage (a fixed part plus a part per domain delivery), validation (amortized initial validation plus annual revalidation), and CRO onboarding (amortized per CRO). Annual avoided cost per caught issue is the unit cost C times the sum of two terms: the documented caught-issue discount used in Section 5.3 (80%) and a multiple M that stands for all other avoided cost, namely rework, delay, and database-lock remediation. M = 0 means that only the documented query-resolution saving is counted. Deliveries are scaled from the simulation: its mean of six domain deliveries per study-month gives 72 domain deliveries per study-year, and the base-case issue rate (14%) and catch rate (80%) of Section 5 are retained.

Table.16. Illustrative annual-cost scenarios for the proposed architecture. All values are author placeholders, not data.

Cost item (illustrative placeholder, USD)	Lean	Mid	Enterprise
Platform operation (annual)	120k	300k	600k
Rule development (one-time; amortized over 3 years)	80k	200k	400k
Rule maintenance (share of development cost per year)	15%	20%	25%
Cloud and storage (fixed per year + per domain delivery)	12k + USD 0.5	36k + USD 1.0	100k + USD 2.0
Validation (one-time; amortized) and annual revalidation	60k; 20%/yr	150k; 25%/yr	300k; 30%/yr
CRO onboarding (one-time per CRO; amortized)	15k	30k	50k
Total annual cost at 50 studies and 5 CROs (USD)	**229,467**	**583,767**	**1,213,867**

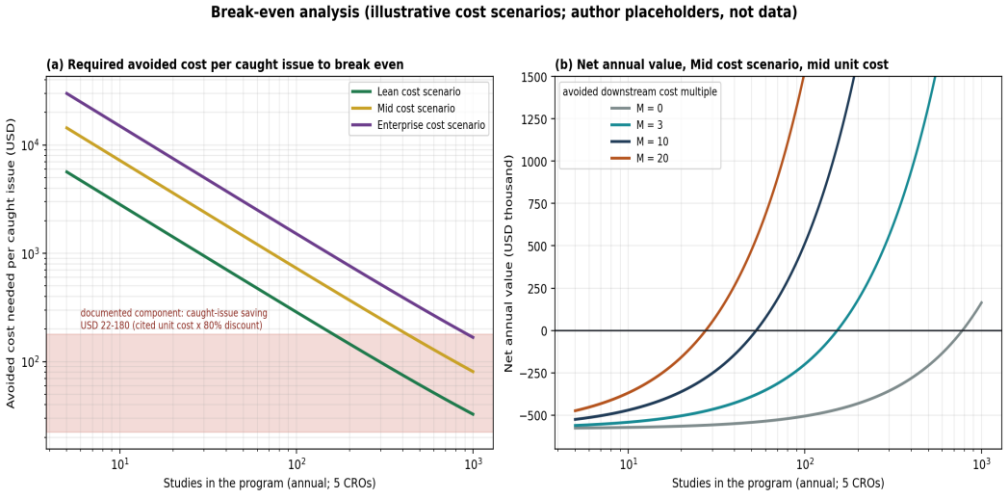


Table 17 and Figure 9 give the results. With only the documented component counted (M = 0) and the mid-range unit cost (USD 126.50), the gate does not pay for itself until about 292 studies per year in the Lean scenario, 780 in the Mid scenario, and 1,795 in the Enterprise scenario, which correspond to roughly 21,013, 56,139, and 129,271 domain deliveries a year. These program sizes are well above the 14 studies of the simulation. The result is sensitive to the unit cost: at the cited upper bound (USD 225) the Lean scenario breaks even at about 161 studies, and at the lower bound (USD 28) at about 1,574.

|www.ijrpetm.com | ISSN: 2454-7875 | editor@ijrpetm.com |A Bimonthly, Peer Reviewed & Scholarly Journal|

||Volume 8, Issue 3, May-June 2025||

DOI:10.15662/IJRPETM.2025.0803015

Table.17. Break-even number of studies per year (5 CROs) at the mid-range unit cost, with the range across the cited unit-cost bounds (USD 225 to USD 28) in brackets. M is the avoided downstream cost per caught issue as a multiple of unit cost. Values are model outputs under placeholder inputs.

Scenario	M = 0	M = 3	M = 10	M = 20
Lean	292 [161-1,574]	59 [33-277]	21 [12-95]	11 [6-49]
Mid	780 [421-5,341]	153 [85-738]	53 [30-245]	27 [15-125]
Enterprise	1,795 [923-32,939]	323 [179-1,690]	111 [62-526]	57 [32-265]

Break-even moves quickly once avoided downstream cost is counted. If rework, delay, and database-lock remediation together avoid ten times the unit cost per caught issue (M = 10), break-even falls to about 21, 53, and 111 studies per year in the three scenarios. Whether M is of that order is the central empirical question, and this paper has no data to answer it. Inverting the calculation gives the avoided downstream multiple M that a program of 14, 50, or 200 studies would need for the gate to break even (Lean, Mid, and Enterprise scenarios: about 15, 40, and 84 at 14 studies; 4, 11, and 23 at 50; 0.4, 2.1, and 5.3 at 200). At the simulation's scale of 14 studies, the Mid scenario needs a multiple of about 40 times the unit cost (roughly USD 5,148 per caught issue), whereas at 200 studies it needs about 2.1 times.

Three points follow. First, scale in studies and deliveries drives the result; the number of CROs hardly does. Raising the CRO count from 3 to 10 shifts the Mid-scenario break-even at M = 0 by about 12%, because in this model CROs enter only through onboarding cost, while benefits scale with deliveries. The model does not capture benefits that may grow with CRO heterogeneity, such as avoided cross-CRO inconsistency, and so understates any such effect. Second, the fixed components (platform, rule development, validation) dominate cost at small scale, which is why the break-even is a question of portfolio size rather than of gate performance alone. Third, the answer to the question of how many studies, CROs, or deliveries justify the architecture is conditional: under these placeholders it is justified on documented query savings alone only for programs of several hundred studies, and for smaller programs only if avoided delay, rework, and database-lock cost per caught issue is of the order of ten times the unit cost or more. These statements are properties of the model and its placeholder inputs and are not findings about real programs.

VII. LIMITATIONS AND THREATS TO VALIDITY

Simulation parameters are partially unvalidated. The issue-occurrence rate (14%) and ingestion catch rate (80%) are author-chosen illustrative parameters, not fitted to a real CRO delivery dataset; the resolution cost range (USD 28-225) is likewise an uncited author assumption. A sensitivity analysis (Section 6.2) varies catch rate, issue rate, and the caught-issue cost discount over a grid, but the grid bounds are themselves author-chosen, and the gate's own operating cost enters only through the placeholder scenarios of Section 6.3.

Most architecture metrics are unmeasured. Of the seven defined in Section 6.1, the simulation yields only Mean Time to Detect, a Detection Coverage that is set by an input parameter, and a resolution-effort proxy for Mean Time to Resolve. False Positive Rate, CRO Onboarding Time, Rule Reuse Ratio, and Validation Coverage have no values in this paper, and the simulation does not model false positives at all, so its cost figures exclude the cost of incorrectly quarantined deliveries.

The non-functional requirements of Section 4.9 are stated, not demonstrated. No requirement has a proposed numeric target, and none has been tested; scalability, availability, latency, and recoverability in particular would have to be established in a prototype under load and failure conditions.

The failure-scenario analysis in Section 4.7 is a design walk-through. It has not been exercised against an implementation, and the detection and response claims, including the drift thresholds of scenario 3 and the degraded-mode behavior of scenario 5, are proposals whose effectiveness is unmeasured.

|www.ijrpetm.com | ISSN: 2454-7875 | editor@ijrpetm.com |A Bimonthly, Peer Reviewed & Scholarly Journal|

||Volume 8, Issue 3, May-June 2025||

DOI:10.15662/IJRPETM.2025.0803015

The break-even analysis rests on placeholder costs. None of the cost inputs in Table 16 and none of the avoided-cost multiples in Table 17 is drawn from data. In particular, the avoided cost of rework, delay, and database-lock remediation is represented by a single multiple that the paper cannot support, and the break-even program sizes reported in Section 6.3 should not be quoted as evidence that the architecture is, or is not, economically justified. They show how the answer depends on scale and on that unmeasured multiple.

Simulation scope is narrow. The model covers only conformance-issue detection timing and resolution cost. It does not model subject-identity resolution across CROs (stage 4 in Section 4.4; architecture in Section 4.4.1), controlled-terminology harmonization accuracy, retrieval/query performance of the underlying open table format, or the implementation and maintenance cost of the conformance-rules library itself — the last of which is a real, non-trivial cost this simulation does not offset against the reported savings.

The simulation covers SDTM only; the ADaM layer is unevaluated. The simulated deliveries cover ten SDTM domains (DM, AE, LB, VS, EX, CM, MH, DS, EG, SU). The ADaM architecture in Section 4.5 is a specification: it has no simulation, no implemented rules, and no evidence that the proposed traceability and population checks catch real divergence. ADaM harmonization also depends on statistical-programming practice and analysis-plan definitions that vary by sponsor and study and are not modeled. The reported results should not be read as covering ADaM, or SDTM domains beyond the ten simulated, nor as evidence for any particular validation-class disposition.

Federation introduces coordination costs that are not evaluated. Running a gate inside each CRO environment requires CROs to deploy and maintain the sponsor's rules, keep rule versions synchronized across gates, and accept sponsor-defined contracts; the metadata control plane itself becomes a governed asset; Section 4.2.1 proposes a change-control model, but its availability, access model, and operating cost are not specified here, and neither the governance model nor rule-change workload is evaluated by the simulation, which treats the rule set as fixed. Whether CROs would accept in-environment gates, or whether a sponsor-hosted gate would be needed in practice, is an open question this paper does not resolve.

No implemented system. This paper does not report a built or deployed system. Feasibility, engineering effort, and integration cost with real CRO delivery workflows are not assessed.
Literature review scope. The review in Section 2 is targeted rather than systematic; the statement that we did not identify a published architecture combining these elements applies only to the literature reviewed, which was not systematically searched, and does not rule out published work that was missed or unpublished proprietary implementations.

Regulatory validation is specified only as architecture. Section 4.8 states the controls the proposal assumes, but no validation plan, qualification protocol, or validation evidence exists, no implementation has been qualified, and the mapping of controls to regulatory expectations is illustrative. A deployment handling data destined for regulatory submission would require a computer system validation (CSV) plan and execution under the sponsor's quality system.

Total cost of ownership is not netted against the reported savings. Section 6's cost reduction is gross, not net: it does not offset the engineering effort to build and maintain the conformance-rules library (Section 4.6), the platform infrastructure cost of the lakehouse layer itself, or the change-management effort of onboarding each CRO onto a contract-style delivery process. A defensible business case would need a real implementation's actual build and run cost, which this paper does not estimate.

VIII. CONCLUSION AND FUTURE SCOPE

This paper reviewed published literature on lakehouse architecture (Apache Software Foundation, n.d.; Databricks, n.d.), and proposed a federated lakehouse architecture with an automated, ingestion-time conformance gate for multi-CRO SDTM harmonization. To illustrate the proposed gate's intended mechanism, a fully documented, reproducible simulation study was constructed, with resolution cost inputs chosen by the author as illustrative assumptions; all quantitative results in Sections 5–6 are simulation outputs and are not claims about real-world performance. Future work required before any real-world efficacy claim could be made includes calibration of the sensitivity grid's parameters, collection of actual platform, rule-governance, validation, and onboarding costs and measurement of avoided downstream cost, calibration against real (de-identified, compliance-cleared) CRO delivery data where

|www.ijrpetm.com | ISSN: 2454-7875 | editor@ijrpetm.com |A Bimonthly, Peer Reviewed & Scholarly Journal|

||Volume 8, Issue 3, May-June 2025||

DOI:10.15662/IJRPETM.2025.0803015

available, a prototype implementation of the conformance gate against real or realistic SDTM deliverables, and a formal computer-system-validation plan, with executed qualification evidence, for GxP deployment.

REFERENCES

1. Apache Software Foundation. (n.d.). Apache Iceberg — The open table format for analytic datasets. Retrieved August 16, 2026, from https://iceberg.apache.org/
2. Belligoli, P., Weberpals, C., & Flores Lopez, Y. (2026). Closing the loop: Validating AI-generated SDTM mappings using CDISC CORE and synthetic data [Conference paper]. PharmaSUG 2026. https://pharmasug.org/proceedings/2026/SS/PharmaSUG-2026-SS-261.pdf
3. Bitol. (n.d.). Open Data Contract Standard [Specification]. GitHub. Retrieved October 6, 2026, from https://github.com/bitol-io/open-data-contract-standard
4. Clinical Data Interchange Standards Consortium. (n.d.-a). Analysis Data Model (ADaM). Retrieved October 6, 2026, from https://www.cdisc.org/standards/foundational/adam
5. Clinical Data Interchange Standards Consortium. (n.d.-b). CDISC Controlled Terminology. Retrieved October 6, 2026, from https://www.cdisc.org/standards/terminology/controlled-terminology
6. Clinical Data Interchange Standards Consortium. (n.d.-c). CDISC Rules Engine (CORE) [Computer software]. GitHub. Retrieved October 6, 2026, from https://github.com/cdisc-org/cdisc-rules-engine
7. Clinical Data Interchange Standards Consortium. (n.d.-d). Dataset-JSON. Retrieved October 6, 2026, from https://www.cdisc.org/standards/data-exchange/dataset-json
8. Clinical Data Interchange Standards Consortium. (n.d.-e). Define-XML. Retrieved August 16, 2026, from https://www.cdisc.org/standards/data-exchange/define-xml
9. Clinical Data Interchange Standards Consortium. (n.d.-f). Study Data Tabulation Model (SDTM). Retrieved October 6, 2026, from https://www.cdisc.org/standards/foundational/sdtm
10. Confluent. (n.d.). Schema evolution and compatibility for Schema Registry on Confluent Platform. Retrieved October 6, 2026, from https://docs.confluent.io/platform/current/schema-registry/fundamentals/schema-evolution.html
11. Databricks. (n.d.). What is the medallion lakehouse architecture? Retrieved October 6, 2026, from https://docs.databricks.com/lakehouse/medallion.html
12. Dehghani, Z. (2020, December 3). Data mesh principles and logical architecture. MartinFowler.com. https://martinfowler.com/articles/data-mesh-principles.html
13. Electronic records; electronic signatures, 21 C.F.R. pt. 11 (2026). https://www.ecfr.gov/current/title-21/chapter-I/subchapter-A/part-11
14. Food and Drug Administration. (2025, April 9). Electronic study data submission and data standards for Clinical Data Interchange Standards Consortium Dataset-JavaScript Object Notation (Docket No. FDA-2025-N-0129). Federal Register, 90(67). https://www.govinfo.gov/content/pkg/FR-2025-04-09/html/2025-06051.htm
15. International Council for Harmonisation of Technical Requirements for Pharmaceuticals for Human Use. (2025). ICH harmonised guideline: Guideline for good clinical practice E6(R3) (Final version, adopted 6 January 2025). https://database.ich.org/sites/default/files/ICH_E6%28R3%29_Step4_FinalGuideline_2025_0106.pdf
16. Pinnacle 21. (n.d.). OpenCDISC validation framework. Retrieved October 6, 2026, from https://pinnacle21.com/projects/validator/opencdisc-validation-framework
17. Zhang, L. X., Lanoue, J., & Nielsen, U. (2026). From specification to SDTM at speed: Deploying the SDTM engine in production [Conference paper]. PharmaSUG 2026. https://pharmasug.org/proceedings/2026/DS/PharmaSUG-2026-DS-440.pdf