RJL SOLUTIONS

PRACTITIONER STANDARD / DATA PRODUCTS

Data Product Certification Standard

A practical contract for meaning, access, quality and operation.

MEANING · ACCESS · QUALITY · OWNERSHIP

Richard White
RJL Solutions · rjlsol.com

OCTOBER 2026

Certify a stated product, and a stated scope.

A certified data product has a named owner, an approved meaning and interface, enforced access, tested service commitments and evidence that supports continued reliance. Certification records a decision about a version and a scope. A catalog entry, a reachable table or a completed manifest doesn't produce it automatically.

This standard is for domain data product owners, stewards, platform engineers, security teams and release reviewers. It proposes an internal certification pattern for the connected enterprise architecture. It's a reference standard. It isn't external accreditation, a vendor schema or a claim of regulatory conformity. Adopt it and version it through your own governance process.

STATE	PERMITTED RELIANCE AND REQUIRED DECISION
`sandbox`	Approved experimental use within defined data and audience limits
`managed_sandbox`	Named reviewers use real governed data, with controls and support agreed for the trial
`certified`	Approved consumers rely on the version within its accepted contract
`degraded`	A recorded breach or uncertainty restricts use through enforced conditions
`deprecated`	Replacement and migration deadline given; agreed service continues
`retired`	Interfaces and access are removed, and retained records are handled under their policies

A Proposed intake records owner and purpose. Typical path: `sandbox`, `managed_sandbox`, `certified`. A breach moves a certified product to `degraded`, and it returns only after retesting. A named replacement moves it to `deprecated`, then `retired` once consumers migrate; any state may be retired.

An agent, tool or dashboard still needs its own contract and acceptance. A source certified for monthly reporting may be wrong for a live operational answer. Approve the consumer's use of the dependency, not just the dependency. A consumer that changes certified data and shares the result has published a new product, and that product needs its own certification.

Keep declarations and decisions separate

Store `data-product.yaml` beside the implementation code. Validate it against a versioned schema, resolve its references, and publish the accepted version in the catalog. The product owner proposes a lifecycle state; the release workflow controls promotion and records the decision.

> **A contributor must not obtain certification by editing the word `certified`.**

WHAT A DECISION MUST RESOLVE TO	Record the exact manifest and implementation version in a protected release decision, with references that resolve to immutable, integrity-checked versions. A moving link cannot show what was approved.

02 / THE MANIFEST

Publish the product manifest.

References keep the manifest compact. Each one has to identify a controlled, versioned artifact, and owner aliases have to resolve to accountable people and funded teams. Digests are shortened here.

DATA-PRODUCT.YAML — ILLUSTRATIVE STRUCTURE

```yaml
schema_version: rjl.data-product.v1
id: claims.claim-status
version: 0.1.0
domain: claims
lifecycle: managed_sandbox

owners:
  product: claims-data-owner
  steward: claims-data-steward
  service: claims-platform-operations
  support_ref: ops/support-v1.yaml
scope:
  purpose: claim_context_lookup
  consumers: [claims-context-assistant]
  excluded: [adjudication, payments, model_training]
sources_ref: lineage/authoritative-sources-v1.yaml
interfaces:
  - id: claim-status-read
    kind: read_api
    contract_ref: contracts/claim-status-v1.yaml
    compatibility_ref: contracts/migration-v1.yaml
semantics_ref: semantics/claim-status-v1.yaml
policy_ref: policy/claims-context-v1.yaml
quality_ref: quality/claim-status-checks-v1.yaml
service:
  slo_ref: ops/claim-status-slos-v1.yaml
  recovery_ref: ops/recovery-v1.yaml
  runbook_ref: ops/claim-status-runbook-v1.yaml
evidence:
  lineage_ref: lineage/claim-status-v1.json
  test_pack_ref: tests/claim-status-v1.yaml
  results_ref: evidence/trial-run-001.json@sha256:9f2c…
  deployment_ref: releases/trial-001.json
  audit_ref: evidence/access-and-retention-v1.yaml
  decision_ref: decisions/managed-sandbox-001.json@a41e7c2

decision:
  decided_at: 2026-09-14
  review_by: 2026-12-14
cost:
  workload_ref: cost/claim-status-workload-v1.yaml
  limits_ref: cost/claim-status-limits-v1.yaml
  escalation_owner: claims-data-owner
```

Source and lineage references separate authority from technical ancestry. The policy reference governs enforcement. The results and decision references are pinned by digest or commit, so they can’t silently follow the latest run.

03 / THE FIELDS

Define each field as an enforceable contract.

Reject unknown fields, wrong types, duplicate keys, missing declarations and invalid states. Require non-empty identifiers, references, consumer and interface lists, and explicit exclusions even when they're empty. Resolve references before promotion.

FIELD OR GROUP	WHY IT EXISTS, AND WHAT ITS CONTRACT MUST CONTAIN
`schema_version` `id, version, domain`	Parser version, unique stable identity, contract version, and the accountable domain
`lifecycle`	One of the six states; the certification workflow reconciles it to the release decision
`owners`	Product and steward authority, operating team, support hours and escalation
`scope`	Approved purpose and consumers, plus exclusions; wider use needs approval and an impact review
`sources_ref`	Authority for each fact, effective time, refresh path, and conflict precedence
`interfaces`	Named kind, schema, access path, errors, masking, and compatibility or migration rules
`semantics_ref`	Grain, keys, time, units, definitions, allowed joins, metrics, and relevant ontology mappings
`policy_ref`	Classification, isolation, purpose, entitlement, export, residency, retention, applicable consent
`quality_ref`	Freshness, completeness, validity and reconciliation checks, with populations, limits, owners and failure handling
`service`	SLO measures and windows, recovery objectives, runbook, alerts, restriction, and recovery tests
`evidence` `decision`	Lineage, test pack, results, deployment, audit location and retention, exact-release approval, and decision and `review_by` dates
`cost`	Expected load, cost attribution, quotas and budgets, enforcement behavior, and escalation owner

Use explicit units. Distinguish event time, source-update time and ingestion time. Freshness targets name the clock and the breach response. Completeness measures name populations and exclusions. Joins keep the approved grain and expose unresolved matches.

FOR THE CERTIFIED STATE — `decision_ref` must authorize the exact product, implementation, environment, users and scope. Other states require their corresponding records. The referenced contracts form part of the certification boundary.

04 / PROMOTION

Require evidence before promotion.

Run contract, policy, quality, lineage and service checks alongside a review of meaning and intended use. Record the criteria, population, results, versions, environment, evaluator and date. This standard requires the checks below. Implementations decide their test mechanics.

RELEASE CHECK	EVIDENCE AND DECISION HOLDER
Meaning and source	Steward and domain owner approve definitions, grain, authoritative facts, reconciliation, and the difficult boundary cases
Interface compatibility	Product owner and engineering validate schema, error behavior, consumer impact, and migration
Access and purpose	Security and policy owners verify allowed and denied routes, revocation, isolation, masking, caches, and export
Quality and service	Product and service owners accept measured freshness and quality, load limits, monitoring, and recovery evidence
Trace and cost	Engineering demonstrates source-to-output lineage, request evidence, retained versions, attribution, and limits
Ownership and release	Authorized reviewers approve scope, support, residual issues, the decision record, and catalog publication

QUALITY/CLAIM-STATUS-CHECKS-V1.YAML — ONE WORKED CHECK (ILLUSTRATIVE THRESHOLDS)

```yaml
id: claim-status-freshness
measure: check_time - source_watermark  # heartbeat, not last event
schedule: every 1 minute
threshold:
  max_lag_minutes: 15
  percentile: 95
  window: 1h
report_also: event_time_lag  # end-to-end latency
on_breach: [restrict_interface, notify_consumers]
```

Test the cases that reveal consequential failure

Test duplicate claim identifiers across tenants, contradictory statuses, reopened claims, late events, missing assignments and stale caches. Verify the time basis and the permitted fields. Revoke a caller and test enforcement across the API, tool, index and cache paths.

Test outages, schema changes, invalid joins and quality breaches. Verify the agreed degraded or unavailable response, and the alert. Before certification, a declared target needs a working measurement and a working response.

Record the certification decision.

Keep the decision identifier, the approvers and their authority, the time, the product and manifest version, the implementation and policy versions, the environment, the interfaces, the approved consumers and purpose, the evidence references, the conditions, and the next review trigger.

Publish those conditions with the catalog entry, and protect the decision from ordinary contributor edits.

Reject promotion when access, isolation, required evidence, source authority or a required acceptance condition fails. Where policy allows a bounded exception, record the risk authority, the compensating control, the restriction, the expiry and the retest condition.

> **The exception cannot silently widen purpose or user entitlement.**

The product owner stays accountable throughout. Specialists decide within their mandates. The authorized workflow promotes the version and records who can suspend it.

Maintain the right to rely on the product

Certification continues only while the product meets its accepted conditions, and a `certified` product lapses to `degraded` when `review_by` passes without a recorded renewal; a `managed_sandbox` trial returns to `sandbox`. Monitor quality, service, authorization, dependency changes and actual consumer use. Record breaches and restrictions against the product identifier and the release.

Validate the declaration itself

Check each manifest against JSON Schema 2020-12 with `additionalProperties: false`, and load YAML with a parser that rejects duplicate keys, because common loaders such as PyYAML silently keep the last one.

THE BADGE IS NOT THE EVIDENCE — An unchanged catalog badge does not override contrary operating evidence.

---

05 / CONTINUING OWNERSHIP

Assign the work that sustains the contract.

The domain funds the product owner and the steward who maintain source meaning and consumer fit. The service team owns alerts, incidents, recovery and access operations. Shared platform teams supply repeatable validation, identity propagation, evidence storage and catalog publication. Governance maintains the adopted standard and the delegated approval rules.

Give these teams capacity, and a review cadence set by the product's consequence and how fast it changes. The owner has to be able to identify consumers, investigate a request, restrict use, approve a correction and coordinate a migration. Demonstrate those tasks before accepting operational transfer.

TRIGGER	REQUIRED ACTION UNDER THIS STANDARD
Meaning, schema or source change	Assess consumer and control impact, version the contract, and repeat the relevant approval and tests
Access or purpose expansion	Obtain the new policy and business approval, and test the expanded routes before use
Quality, service or evidence breach	Restrict the affected interface or scope, notify consumers, correct, and record the restoration decision
Owner or support change	Confirm accountable capacity and transfer records before continued reliance
Deprecation or retirement	Publish dates and replacements, migrate consumers, revoke access, and apply retention policies

Version meaning as well as structure

A new meaning, unit, grain, authority rule or time basis can break a consumer even when the field names are unchanged. Under this convention, incompatible contract changes require a new major version and a migration plan, and `certified` requires version 1.0.0 or later, because 0.x versions promise no compatibility. Compatible additions require an impact review; corrections require retesting. State the coexistence and retirement rules explicitly.

Before adopting the product, consumers should be able to see the approved version, purpose, restrictions, service conditions and support route. Agents and tools have to keep the product version in their evidence, and handle degraded or retired states according to their own contracts.

For large banks, the standard supports BCBS 239 principles on accuracy and integrity, completeness, timeliness and adaptability, and the lineage supervisors use to test them. RJL can help adapt this standard to one domain, and establish the contract, the validation path, the certification authority and the continuing operating model. Request a working session at **rjlsol.com/contact** or write to **rwhite@rjlsol.com**.

PRODUCT CONSTRUCTION / PROMOTION EVIDENCE

Processing stage and assurance state are different.

A gold representation can be uncertified. A supported product earns an assurance state through evidence for a stated version and use.

GATE	EVIDENCE AND CONSEQUENCE
Controlled intake	Source authority, rights, schema checks, provenance, replay/deduplication and retention. Failed intake is isolated; it does not gain publication rights.
Validated transformation	Key and type rules, quarantine record, late-correction handling and source reconciliation. Record any material unresolved exception.
Supported publication	Approved contract, grain, permitted uses, interface/version, positive and negative access tests, consumer acceptance and named run ownership.
Consumer-specific assurance	Record the certification approver, scope, tests and conditions. Exploratory analysis and final reporting need not share the same assurance scope.
Continuing operation	Monitor commitments, rights and dependencies. A material failure can require degraded status, a publication hold or recertification.

For synthetic Invoice Lines v1, retain the exact source snapshot or controlled references, rule and schema versions, reconciliation results, denied cross-entity access test, cutoff and approval. For derived Receivables Exposure, add join/cardinality tests and input-version compatibility. Upstream certification alone does not certify a new composition.

A late payment, changed definition, revoked source right or broken serving control can trigger reassessment. Keep the lifecycle and assurance states already defined in this standard; bronze/silver/gold is not a replacement state machine. See Data Products in Practice for the complete construction example.

Design references reviewed 29 October 2026: Databricks: medallion architecture; Designing data products; Google SRE: service objectives. The worked records, controls and targets are illustrative RJL recommendations.

ABOUT

Richard White

I was global firm chief data officer at McKinsey & Company, owning data, AI and platforms end to end, and before that chief data officer at New York Life, where I built the chief data office and reported data risk to risk committees at multiple levels. Earlier I led data at the Federal Reserve Bank of New York and enterprise data governance at Citigroup. I founded RJL Solutions in 2024. My book, *The Day After the Demo*, is free at rjlsol.com/book.

RJL Solutions connects strategy, architecture, engineering, governance and operation through Frame, Prove, Build, and Run & Transfer. Each phase is separately scoped and priced in a written proposal before you commit. The first sixty-minute working session is free.

TECHNICAL **Agent Onboarding: Treating an Agent Like a Data Product**
How a consumer of these contracts is itself certified, in 13 pages.

GOVERNANCE **AI Governance Evidence**
What reviewers need to inspect, and how to build it into delivery.

ARCHITECTURE **The Four-Part System**
Where these products sit in the wider architecture, in 16 pages.

RJL SOLUTIONS

WEB rjlsol.com

EMAIL rwhite@rjlsol.com

This is an RJL Solutions reference standard. It isn't an external accreditation, a vendor schema or a claim of regulatory conformity. The claims example and the manifest structure are illustrative. Adopt and version the standard through your own governance process.