RJL SOLUTIONS

OPERATORS' CORNER / DATA, AI AND OPERATIONS

Rewiring Data Management with AI

Approved intent, bounded agents and less repeated coordination, from request to retirement.

CDO · DATA ENGINEERING · STEWARDSHIP

Richard White
RJL Solutions · rjlsol.com

VERSION 1 · PUBLISHED EDITION
OCTOBER 2026

01 / EXECUTIVE SUMMARY

Remove the repeated translation, not just the wait.

FOR DATA, TECHNOLOGY AND BUSINESS LEADERS

Most data work isn't hard because the code is hard. It's slow because someone translates business intent again at every handoff. The owner explains the need. An analyst finds sources, an engineer interprets fields, a steward reconstructs the definition and a reviewer asks for evidence. When a source or a definition changes, much of that chain runs again.

AI can make each handoff faster and leave every one of them in place. The bigger opportunity is to connect the steps around one approved record of intent. Then automation proposes the pipeline, tests, policy bindings, documentation and release evidence together, and the business owner sees only the decisions that need judgment.

What we recommend

- Capture intent once. Approved definitions, examples and contracts become the inputs that drive implementation. They stop being documents written after it.
- Give agents explicit authority, and make them earn it. They gather evidence and propose versioned changes. They execute only narrow actions with independent authorization.
- Run data as a service. Measure freshness, correctness and recovery from the consumer's side, and keep certification tied to current evidence.
- Prove total effort fell before claiming money. Count the work that moved to reviewers and platform teams, and keep released capacity apart from planned cash benefit.

THE TEST An approved business intent reaches a supported data service with less repeated coordination. Agents stay within delegated authority. And the team can detect failing evidence and recover. That's a measurable way to change data management.

---

02 / THE PROBLEM

Faster handoffs keep the handoffs.

Think of a recurring request for a claim-cost or customer-exposure measure. Each person in the chain holds part of the meaning, and each restates it in a different artifact: a ticket, a mapping sheet, SQL, a glossary entry, a review note. This is an illustrative workflow to investigate. I'm not claiming it describes every organization.

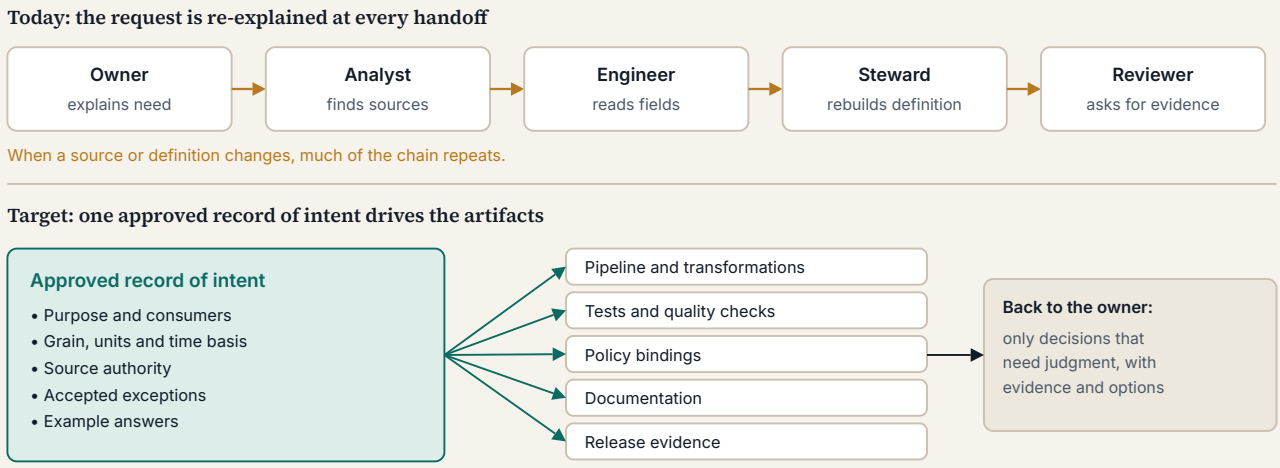


Exhibit: Remove the repeated translation, not just the time each handoff takes.

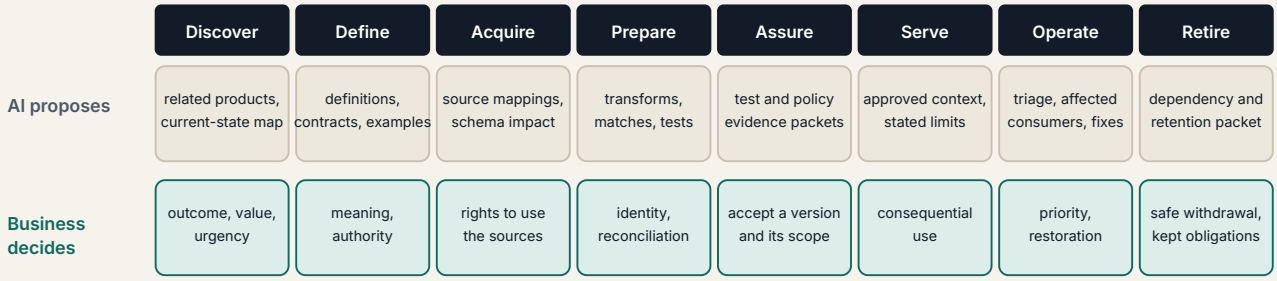
Put a conversational assistant on each step and you may speed each one up while keeping all five. The design that changes the economics connects the steps instead. The owner approves the purpose, grain, time basis, source authority, acceptable exceptions and example answers. Automation uses that record to produce related artifacts, checks them against each other and routes unresolved decisions to the right person.

This can replace routine documentation requests with generated proposals, repeated source discovery with maintained dependency evidence, and unstructured incident handoffs with a packet of observed failures and affected consumers. It can also expose requests that should reuse an existing product. Treat these as opportunities to prove on a bounded workflow. They aren't savings anyone has achieved.

DO NOT HIDE THE WORK Remove an approval only when its decision right and control purpose are covered somewhere else. Moving the same review into an invisible agent queue isn't automation.

AI proposes. The business keeps its decisions.

The opportunity spans the whole data lifecycle, from discovering what exists to retiring what's no longer needed. At every stage there's work AI can assist or propose, and a decision only the business can make. Name both. Otherwise automation turns into an unapproved interpretation that looks authoritative.



Each activity needs a defined interface, a named decision owner and a test of whether the automation helps.
These are automation opportunities, not new certification states: an agent cannot certify by updating metadata.

Exhibit: Across the lifecycle, AI proposes and the business keeps the decisions only it can make.

Connect the stages through identifiers and versions

A change to meaning needs a different response from a change to physical structure. A copied document, a generated index or a derived product needs a declared relationship to its source and a tested treatment of changes. Keep the product types distinct, as Data Products in Practice does. Operational feeds need latency, ordering and failure behavior. Analytical products need refresh, completeness, period represented and finality. Knowledge products need source versions, and propagation of revisions, deletions and permissions. One generic freshness badge can't describe all three.

RJL's Certification Standard uses sandbox, managed sandbox, certified, degraded, deprecated and retired. Automation may prepare the evidence for a move between these states. The authorized workflow makes the transition.

04 / BUSINESS PARTICIPATION

Fewer forms, better decisions.

Spend business participation on the decisions only the business can make. An owner should receive a proposed interpretation with its evidence, alternatives, affected consumers and a clear request for judgment. Asking that person to maintain several copies of the same definition defeats the purpose of automation.

DECISION	EVIDENCE TO PUT IN FRONT OF THE OWNER
Approve meaning	Proposed grain, units, time basis, source precedence, examples and boundary cases
Resolve a conflict	Competing definitions, consequences for each consumer, and who has decision authority
Accept a material change	Version difference, changed outputs, relevant tests, restrictions and residual issues
Decide an exception	Impact, compensating controls, permitted scope, expiry and retest condition

Start from conversation and existing work

Start with an interview and representative cases. An intake agent can draft a contract from the discussion, compare it with approved definitions and list the decisions still missing. Accepted meaning goes into the controlled record and gets reused in implementation and evaluation. Unresolved meaning stays visibly unresolved.

Existing work can bootstrap meaning. Snowflake's Semantic View Autopilot generates semantic context from descriptions, example queries, metadata and query history. Its Power BI ingestion became generally available on 18 August 2026, though report-level measures and time-intelligence functions aren't fully supported yet [19]. An existing query still needs scrutiny before its interpretation becomes approved meaning.

Master data works the same way. Research presented at COLING 2025 shows large language models can rank and compare candidate entity matches [21]. That's evidence for an approach. It doesn't validate any particular enterprise's master data. Test false merges and missed matches on representative records, and keep approved identity and survivorship rules. A shared party identifier can connect two records while the domains still define customer differently. The agent should surface that difference, not normalize it away.

05 / CONTRACT-DRIVEN DELIVERY

An approved contract should drive the build.

A machine-readable contract connects meaning and intended use to the artifacts that implement them. The Open Data Contract Standard (ODCS) gives producer-consumer agreements a common structure: schema, quality rules, roles and service commitments [16]. Adopting the format creates no behavior by itself. The platform has to turn each declaration into a service and a check.

```
apiVersion: v3.2.0          # ODCS v3; illustrative, abbreviated
kind: DataContract
id: urn:datacontract:claims:claim-context
name: claim-context
version: 2.1.0
status: active
description:
  purpose: Claim status and context for examiners; excludes adjudication
schema:
  - name: claim_status
    properties:
      - name: claim_id
        logicalType: string
        primaryKey: true
        required: true
      - name: status_code
        logicalType: string
        quality:
          - type: sql
            description: only statuses listed in this contract
            query: SELECT COUNT(*) FROM claim_status WHERE status_code NOT IN (...)
            mustBe: 0
      - name: status_effective_at   # valid time, not load time
        logicalType: timestamp
slaProperties:
  - property: latency
    value: 15
    unit: m
```

Vendors now show this pattern working. A Snowflake developer guide, updated 6 July 2026, uses an ODCS contract with its Cortex Code agent to generate dbt models and tests, masking policies, data metric functions and monitoring. After a contract change, it regenerates only the affected artifacts [18]. It starts from agreed requirements and synthetic data and includes review checkpoints. Treat it as a worked demonstration. It's no evidence that an enterprise product is production-ready in a sprint. Databricks documents Genie Code generating and running code and building pipelines, governed by the user's Unity Catalog permissions [20]. Permission to run code doesn't establish that the code is correct.

THE CHANGE PROPOSAL

One package per change: the revised contract, the candidate implementation, tests, policy bindings, documentation and expected consumer effects. Run it against a representative environment and inspect the resulting data. The job status alone tells you little. Look for late events, duplicate keys, changing reference data and joins that alter grain.

Compatibility checks are only as good as the contract.

For a source change, compare the observed schema and behavior with the accepted contract. An agent can propose a mapping or migration and name the consumers at risk. A change that keeps field names but alters units, timing or source authority still needs a meaning review. An automated compatibility check sees only what the contract declares and the instrumentation records.

OpenLineage's dataset, job and run model gives you an interoperable basis for collecting execution lineage [17]. Keep observed relationships, manually supplied edges and inferred dependencies distinct. When a source changes, a lineage graph tells you which products, tests and consumers to review. It can't vouch for relationships nobody captured. Databricks, for example, records externally supplied lineage in Unity Catalog but not in its lineage system tables, so a connector that reads only those tables will miss it [12].

CHANGE TYPE	WHAT AUTOMATION CAN DO	WHAT STILL NEEDS A PERSON
Physical: new column, renamed field	Propose mapping, regenerate tests, identify consumers	Engineer confirms behavior on representative data
Behavioral: late or duplicate events	Detect, quantify, propose handling and replay	Owner accepts the consumer impact
Semantic: units, timing, authority	Flag the mismatch with evidence and options	Owner decides meaning; consumers notified
Policy: new restriction or purpose	Propose policy binding and denial tests	Control owner approves; never automatic

Engineering stays responsible for proving that the generated artifacts express the contract and can be operated. Reuse the approved deployment path and existing platforms before adding another service to run.

06 / QUALITY

Match the check to the use.

Keep three things apart: the rule being tested, a statistical signal and the business judgment that data is fit for a purpose. Snowflake's data metric functions measure attributes, and expectations decide whether a measurement passes [7]. Databricks anomaly detection learns freshness from commit history and completeness from historical row counts; it's in Public Preview [8]. Neither establishes every business fact.

CHECK	USE IT FOR	WATCH FOR
Deterministic rule	Allowed codes, uniqueness, referential integrity, reconciliation to a source	Only as complete as the rules written
Anomaly signal	Directing investigation when volume or timing departs from history	A legitimate change looks unusual; a persistent error can look normal
AI-suggested check	Proposing rules and summarizing evidence	Review its assumptions before deploying
Answer evaluation	Whether a response is supported by its cited source	Clean inputs do not guarantee a correct answer

AI adds checks closer to consumption

A retrieval-augmented generation (RAG) system needs relevant, current source content and permission-aware retrieval. Microsoft documents Azure AI Search security filters as generally available and native document permissions as preview. It also states that permission changes reach results only after they're synchronized to the index [9]. So a source access change needs a tested propagation path to the retrieval layer.

The test boundary should include source versions, effective dates, chunk completeness, retrieval relevance and whether each answer is supported by its citation. Keep data-quality validation separate from answer evaluation. For training and fine-tuning, assess provenance and representativeness and check for evaluation contamination. NIST's Generative AI Profile supports these lifecycle practices; the exact tests have to fit the use [2].

07 / OPERATIONS

Measure what the consumer experiences.

Site reliability engineering applies to data services as well as APIs. Google's guidance on pipelines covers freshness, correctness, end-to-end measurement, canarying and recovery [1]. A green scheduler can sit alongside stale or corrupted consumer data. And restarting a failed job may still leave outputs that need repair or reprocessing.

A service-level indicator (SLI) measures an outcome. A service-level objective (SLO) sets its target. An error budget is the shortfall you tolerate over a stated period, and it should drive operating decisions [10]. The commitments below are illustrative. Targets have to fit the use.

OUTCOME	EXAMPLE COMMITMENT	OPERATING RESPONSE
Freshness	Over 28 days, 99% of eligible records reach the serving dataset within 15 minutes of source event time	Alert on fast budget burn; investigate delayed stages
Correctness	Daily financial totals reconcile to the designated source within an approved tolerance	Hold publication when reconciliation fails
Recovery	Replay a failed partition without duplicate outputs	Exercise the runbook; verify repaired consumer data

Define eligible records with reconciled source counts or a producer manifest. State what each timestamp means. Count late arrivals that miss the deadline as failures. An error budget is never permission to breach a security or control requirement. Measure detection, containment and verified recovery separately.

AGENTS IN OPERATIONS

An agent can assemble incident evidence, link affected partitions and consumers and recommend a runbook step. An automated repair needs authorization, and a check of its effect on consumers, before it counts as recovery.

08 / AGENT STEWARDSHIP

Give the agent a boundary, then earn the next one.

Calling every automated check an agent hides a design choice. Anthropic distinguishes workflows, which follow predefined paths, from agents, which choose their own process and tools [3]. For a fixed validation rule, a predictable workflow is usually the right design.

Assisted stewardship is already concrete. Snowflake documents Cortex Code governance skills for investigating access, classifying sensitive data, analyzing quality failures and working with masking and row access policies, each limited by the user's privileges [4]. DataHub's MCP server exposes discovery, ownership, glossary, lineage and quality context to AI tools. Its mutation tools stay off unless explicitly enabled, and its proposal tools submit changes for review instead of applying them [5]. These are documented capabilities. They don't prove autonomous stewardship is reliable.

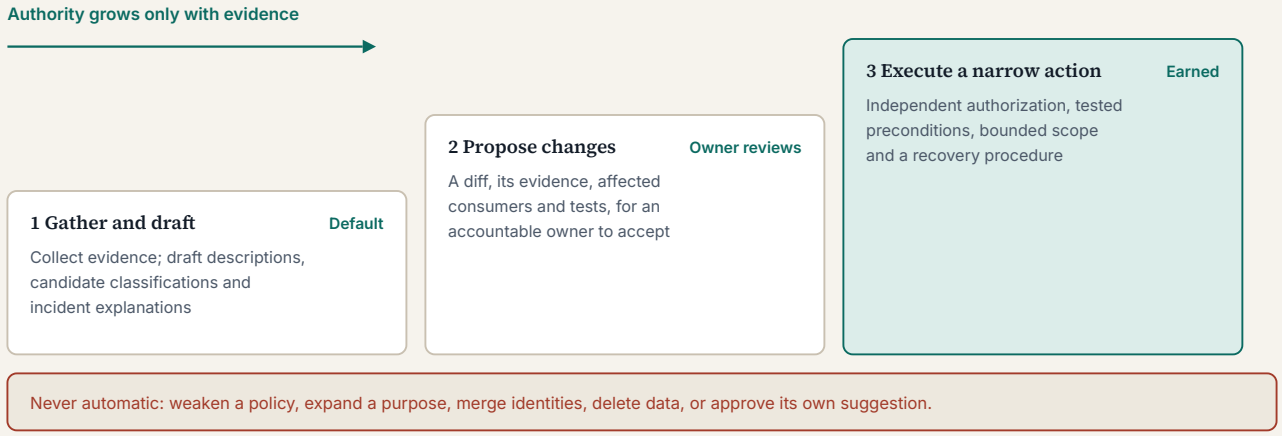


Exhibit: Agent authority is earned one step at a time, and some actions are never automatic.

Retrieved documents, metadata descriptions and tool responses can contain hostile instructions. OWASP names excessive functionality, permissions and autonomy as the causes of excessive agency, and recommends enforcing authorization in downstream systems [6]. Give the agent its own identity and a declared owner. When it acts for a person, bind tool use to the delegated scope and record the identity context. Test the real read and write paths, denied operations included, before extending authority.

Evaluate the agent on the hard cases.

An agent might propose that a column holds customer identifiers, citing the schema, observed patterns and source documentation. The steward approves the classification and intended use. A deterministic service applies the approved policy. The model's suggestion never becomes its own authorization.

MEASURE	WHAT IT TELLS YOU	EXAMPLE THRESHOLD TO AGREE
Unsupported claims	Proposals not backed by retrieved evidence	Zero accepted into the record
Harmful proposed changes	Proposals that would weaken access or alter meaning silently	All caught before approval
Missed escalations	Decisions the agent made that belonged to a person	Zero on the test set
Reviewer corrections	Edits needed before a proposal is accepted	Falling over successive releases
Cost per accepted task	Model, evaluation and review cost divided by accepted proposals	Below the manual baseline

Build the test set from representative cases: conflicting definitions, stale metadata, misleading descriptions and attempted instruction injection. Capture the evidence retrieved, tool calls, actual changes and approvals, without exposing secrets or unnecessary personal data. These are proposed operating measures. They aren't published benchmarks.

THE HUMAN ROLE GETS HARDER The steward now has to judge whether the evidence is adequate, and whether a fluent proposal is valid for the business. Faster documentation is worth little if it makes an unapproved interpretation look authoritative.

09 / CATALOGS

Start native. Prove the gap before you buy.

Two meanings of catalog need separating. An operational catalog resolves data objects and takes part in access and execution. An enterprise metadata catalog supports discovery, business vocabulary, ownership and context across systems. One product can play both roles. An Iceberg catalog and a business glossary still solve different problems.

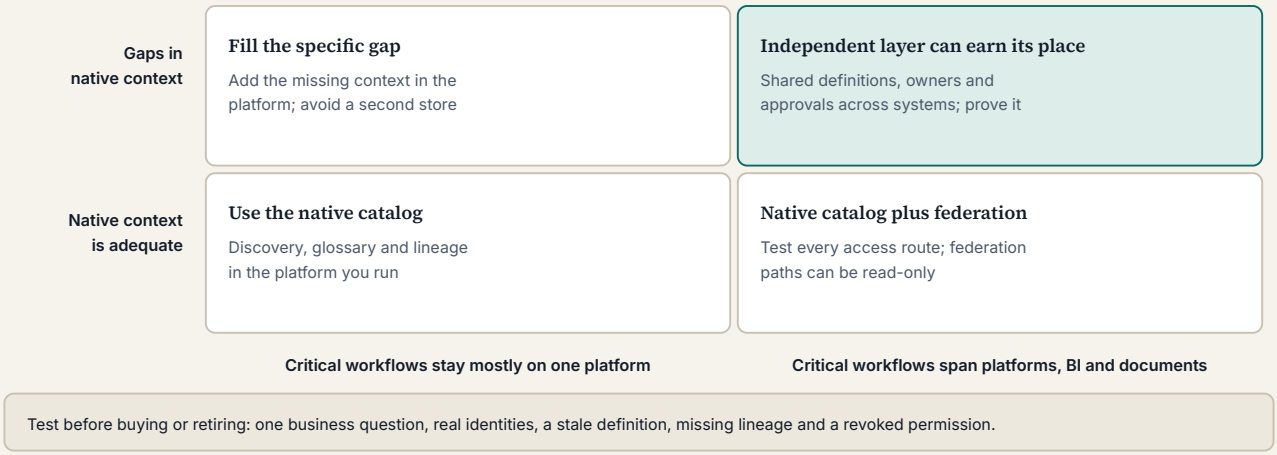


Exhibit: An independent catalog earns its place where workflows span systems and native context falls short.

Native reach is growing. Databricks foreign catalogs give read-only query access to external systems through Unity Catalog [11]. Snowflake Horizon Catalog serves Iceberg tables to external engines over the Iceberg REST interface [13], and its Iceberg Scan Plan API, in Public Preview since 10 September 2026, applies Snowflake row access and masking policies when supported engines such as Spark, Trino and PyIceberg read through that path. It requires Enterprise Edition and can add compute, temporary storage and transfer charges [15]. Snowflake Open Catalog, the managed Apache Polaris service, no longer accepts first-time accounts; new customers are directed to Horizon [14].

The case for an independent layer is strongest where critical workflows span platforms, operational databases, BI and document stores, and the native stack doesn’t supply shared definitions, ownership, approvals or incident context. Consolidating everything in one platform can make migration and enterprise-wide discovery harder. A second metadata store creates synchronization and conflict work. Weigh both against real workflows, including staff effort and exportable metadata. A longer connector list proves nothing.

One claims request, end to end.

This is an illustrative target workflow. It isn't a client deployment or an achieved result. It extends the claims examples used across RJL's architecture papers. The scope is a claim-context product and assistant for examiners, and it excludes adjudication and payments.

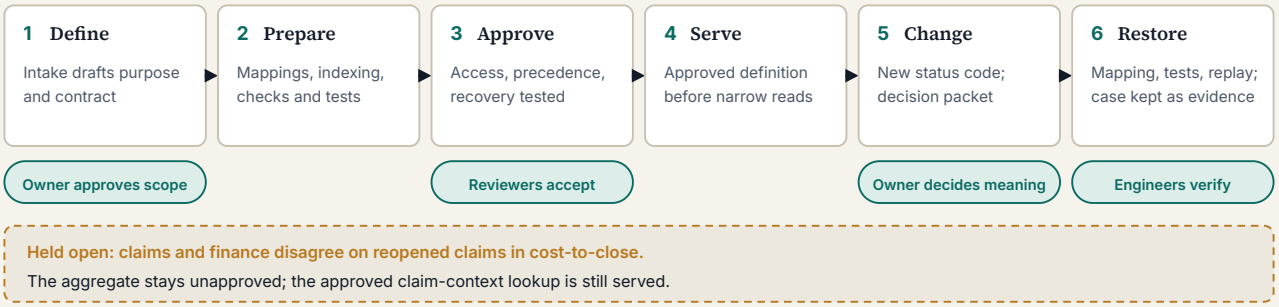


Exhibit: One claims request, six steps, with the people who decide at each gate. Illustrative.

STEP	WHAT HAPPENS
1 Define	An intake agent drafts purpose and contract from approved sources and the owner's explanation. It separates claim status (a product) from cost-to-close (an aggregate definition). The owner approves the lookup's scope and the excluded decisions.
2 Prepare	Automation proposes source mappings, API behavior, document indexing, freshness checks and tests. Review cases include reopened claims, late payments, repeated identifiers across tenants and procedures with different effective dates.
3 Approve	Release checks cover permitted and denied access, source precedence, reconciliation, consumer behavior and recovery. Reviewers accept one implementation and policy version for the stated use.
4 Serve	The assistant retrieves the approved definition and current product condition before using narrow read tools. Serving services enforce entitlements. Answers separate observed facts from an unresolved party match and keep source references.
5 Change	A source sends a status code the contract doesn't list. Monitoring records it. An investigation agent links affected partitions, consumers, recent source changes and candidate meanings. The owner gets one decision packet instead of several tickets.
6 Restore	After approval, an agent proposes the mapping and new tests. Engineers verify the replay and serving output. The failure case becomes regression evidence.

If nobody can interpret the new status reliably, the consumer sees the defined restricted or unavailable response. The agent must never invent a safe-looking mapping to finish the task.

Count total effort, then separate capacity from cash.

The illustrative numbers below follow the claim-context product through one year of accepted changes. They show the method. They aren't a measured result. Replace every value with your own baseline.

PER ACCEPTED CHANGE (40 A YEAR)	TODAY	TARGET	LOADED RATE
Business owner hours	6.0	2.5	$120
Engineer hours	16.0	8.0	$110
Steward and reviewer hours	5.0	4.0	$95
Cost of effort per change	$2,955	$1,560	
Elapsed time, request to accepted release	25 business days	9 business days	

ANNUAL VIEW	VALUE	WHAT IT MEANS
Effort cost today	$118,200	40 changes at $2,955
Effort cost in the target	$62,400	40 changes at $1,560
Capacity released	$55,800	500 hours: 140 owner, 320 engineer, 40 reviewer. Not cash.
New run cost	$18,000	Model use $4,000; evaluation and regression upkeep $9,000; platform share $5,000
Capacity net of run cost	$37,800	Still capacity, not cash
Planned cash benefit	$35,200	Only the 320 engineer hours, if the owner reduces contractor hours by that amount
Planned net benefit	$17,200	Planned cash benefit minus new run cost

Reviewer time barely falls. Review gets more demanding, not less. At one product, the planned net benefit would take about three and a half years to recover a $60,000 build, before counting transition costs. The case rests on reuse. If the same tooling serves six similar products with the same per-product results, planned net benefit comes to about $103,200 a year against that shared build. The shorter elapsed time matters to consumers. It isn't cash.

USE AI TOKEN ECONOMICS

The same distinctions apply as in RJL's AI Token Economics guide. Capacity isn't cash. A cash claim needs a named owner and a way to realize it. And the benefit-cost ratio isn't ROI. Neither model is an NPV or IRR case.

Separate what you can deploy from what is still research.

The art of the possible is a data service that keeps comparing business intent, implementation, observed data, policy and consumer behavior. It finds inconsistencies, prepares a candidate correction and supplies the evidence for a release decision. That's a target architecture assembled from capabilities at different levels of maturity.

	What can reasonably be claimed	How to use it
Documented product capability	Assistance, semantic generation, tracing	Verify status; test it in your estate
Worked demonstration	A contract can drive related artifacts in the example	Reproduce it; test your own assumptions
Emerging research	Agents improved systems in studied settings	Bounded experiment, held-out acceptance tests

A fully autonomous data estate is an aspiration, not a property of any tool cited in this guide.

Exhibit: Match the claim to the evidence behind it.

One frontier is improvement from operational evidence. Databricks' MLflow guidance connects traces and expert feedback to evaluation datasets, fixes and production monitoring [22]. Scorers and model judges need evaluation too, and their labels aren't business truth. Research is moving faster. A paper submitted on 8 October 2026 introduces CoCoEvolve, which optimizes data agents, LLM-powered pipelines and data engineering workflows using historical evidence and live probes [24]. Its results are benchmark-specific. It isn't a generally available service that safely improves an enterprise unattended.

The design implication is still worth pursuing. An improvement agent could draft changes to prompts, transformation code, metadata mappings or tool contracts after recurring failures. A separate evaluation and release authority would decide whether to accept them. Keep held-out acceptance cases apart from the examples used to optimize, and never let the agent rewrite its own success criteria.

Two more experiments are worth running. One agent compares required controls with their observed implementation and prepares evidence of gaps. Another detects duplicated products and proposes consolidation. Neither gets automatic authority to weaken a policy, widen a purpose, merge identities or delete data.

Automation is worth what it keeps supported.

A release establishes a version and scope that consumers may rely on. Later evidence can withdraw that reliance. A review date, a source change, a policy failure or a material quality breach should trigger the response the contract defines. An unconditional badge left in the catalog hides it.

LINK IN THE EVIDENCE CHAIN	RECORD
Approved definition	Version, owner, approval date and scope
Sources and documents	Versions or snapshots used, with effective dates
Transformation and indexing runs	Run identifiers, code version, inputs and outcomes
Quality and policy results	Checks run, results, policy version and binding coverage
Release decision	Who accepted which combination, for what use
AI execution	Inputs, retrieved evidence and actions, where technically available; not a promise of the model's internal reasoning

Cost is an operating signal too. An optimization agent can propose changes to query plans, schedules, storage tiers, model routing or context size. Before accepting, test the effect on freshness, correctness, permissions and total service cost. A cheaper model call can create more review work. A rarely used product can still serve a critical quarter-end process.

Retire with the same discipline as release

NIST's AI RMF Playbook asks for processes to decommission and phase out AI systems safely, with suggested actions covering dependencies and continuity [23]. Applied to data products, an agent can prepare a retirement packet; the owner decides whether the product can be withdrawn. Engineers verify interface removal and access withdrawal, and authorized teams handle retained records and legal holds. Indexes, caches, exports, derived products, model artifacts and backups may each need separate action. The same goes for agents. Retire identities, credentials, tools and schedules through controlled procedures. Treat an absence of recent calls as evidence to review. It doesn't authorize deletion.

Prove one lifecycle before scaling the model.

Start with one consequential workflow whose burden you can measure and whose business owner will take part. Use the existing platform and shared controls. The sequence follows RJL's delivery method. It adds no separate transformation program and promises no timeline.

RJL PHASE	WHAT TO PROVE
Frame	Baseline the burden; name decisions and owners; agree product scope and example answers; check source rights and existing services
Prove	Run assisted intake and a contract-driven candidate in a controlled environment; test meaning, permissions, failures, review effort and cost
Build	Make generation, validation, promotion, monitoring, recovery and consumer onboarding repeatable
Run and transfer	Measure accepted outcomes and total effort; hand over decisions, runbooks, evaluations and ownership to the team

Put a difficult definition, a source change, a failed quality check, a permission revocation and a recovery drill in the pilot. Agree stop conditions before it starts: unsupported interpretation, not enough enforcement evidence, an unacceptable review burden, or a cost profile the outcome doesn't support.

Report request-to-accepted-release time, owner hours per accepted change, reviewer corrections, escaped defects, time consumers spend with impaired data and verified recovery. Compare like-for-like scope and demand. Report technical savings apart from work absorbed by reviewers and shared teams. Set targets only after the baseline.

FOR A LEAN FIRM

One owner, one engineer and one reviewer can run this on the platform you already have. Use native contracts, quality checks and catalog features. Keep the decision record in the repository next to the code. Skip the independent catalog until the test in section 09 shows a gap. Don't skip the stop conditions, the revocation test or the recovery drill.

Templates to copy into your first workflow.

Change proposal packet

- Revised contract with a version difference and the reason for change
- Candidate implementation, generated tests and policy bindings
- Results on representative data: late, duplicate and boundary cases
- Consumers affected, expected effect and notification plan
- Decisions requested, each with an owner and a deadline
- Rollback or compensation step, and how it was tested

Retirement packet

- Usage over the review window and the critical periodic uses
- Lineage to downstream products, indexes, caches, exports and models
- Consumer confirmations and migration evidence
- Entitlements to withdraw and credentials to revoke
- Retention requirements and legal holds, with the team that owns them

Pilot stop conditions, agreed before the start

CONDITION	STOP OR RESHAPE WHEN
Interpretation	The agent’s proposals need the owner to rewrite meaning more often than to approve it
Enforcement	A denied caller can read through any serving path, or revocation lag exceeds the agreed window
Review burden	Reviewer hours per change rise above the baseline after two releases
Cost	Run cost per accepted change exceeds the effort it removes

16 / DECISION

What a decision needs.

Before funding a broader program, a leadership team should see evidence on five points from the first workflow.

QUESTION	EVIDENCE THAT ANSWERS IT
Did repeated coordination fall?	Owner, engineer and reviewer hours per accepted change against the baseline, like for like
Is meaning still owned?	Every accepted definition has an owner, a version and a recorded approval; disputes are visible, not resolved by default
Did agents stay within authority?	Denied operations tested; no unapproved changes; escalations caught on the test set
Can the team recover?	A revocation and a recovery drill passed, with consumer data verified after repair
Is the money real?	Capacity and planned cash benefit reported separately, with a named owner for each cash claim

Read alongside

- The Data and AI Operating Model: ownership, decision rights, funding and shared delivery.
- Data Products in Practice and the Certification Standard: the product, its contract, and acceptance of a version.
- Context Before Query and Agent Onboarding: governed context and agent obligations.
- Patterns and Product Fit: platform-specific controls and their release status.
- AI Token Economics: run cost, capacity and planned benefit by workflow.

DISCLOSURE The author has worked closely with several vendors named here, including Cyera, ServiceNow, Snowflake, Databricks, DataHub and Microsoft, as a customer and practitioner. No vendor reviewed or paid for this paper. Products are named as examples of the patterns, not endorsements.

ARCHITECTURE UPDATE / GOVERNED DELIVERY

Turn approved intent into tested behavior.

Connect stewardship, data engineering and software delivery around the same versioned contract.

STAGE	INVOICE EXAMPLE AND ACCOUNTABLE DECISION
Approve intent	Finance defines what an invoice hold means, the permitted explanation and which changes require a reviewer. Domain owners approve mappings and exceptions.
Generate a change proposal	A bounded agent proposes schema mappings, quality tests, policy bindings, API code and documentation in a reviewable change. It cannot grant itself rights or release its own change.
Test the contract	CI checks meaning, source reconciliation, allowed and denied access, stale context, cross-tenant isolation and command idempotency. Catalog registration alone does not satisfy these tests.
Release and reconcile	Authorized reviewers approve the version. Deployment records policy and tool versions, then checks that deployed behavior matches the approved contract.
Operate and improve	SRE-style stewardship agents detect drift, assemble evidence and propose repairs within bounded authority. Humans resolve meaning disputes and material policy changes.

For the synthetic invoice example in Data Products in Practice, a late payment changes the current balance from 500 to 300. A stewardship agent can assemble the affected versions, lineage, tests and consumer notices. The authorized owner decides whether a closed-period restatement is appropriate; data replay must not replay an external business command.

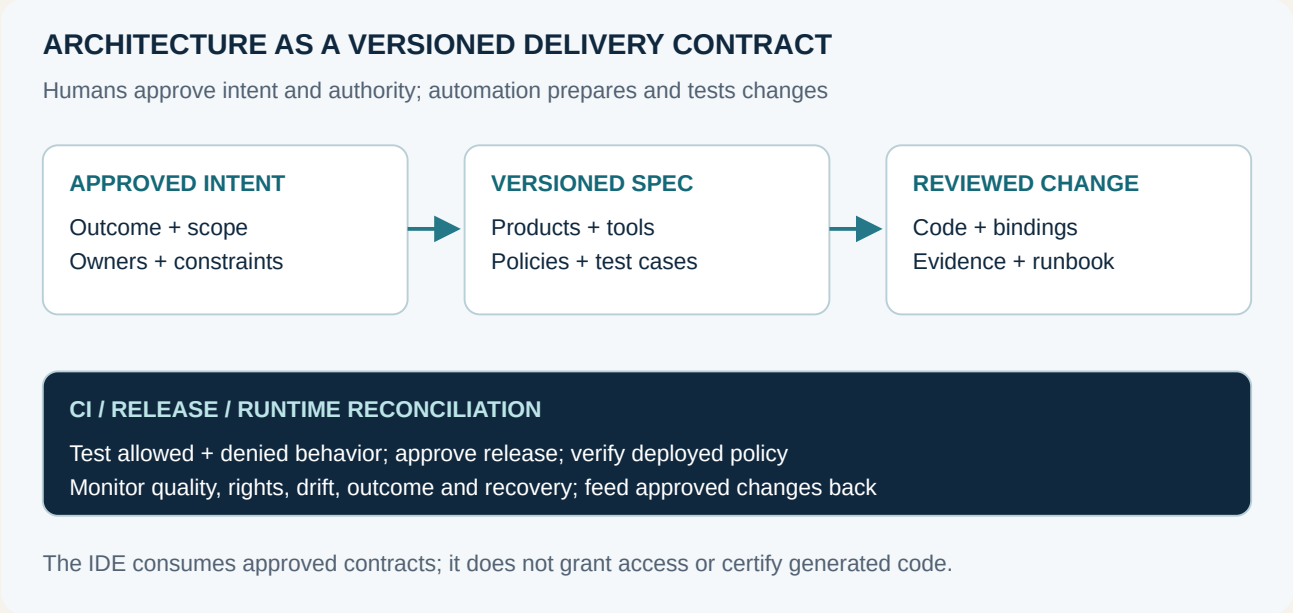
One enterprise vocabulary is not a prerequisite. Finance and customer servicing may use different definitions of outstanding balance. Publish each definition with its purpose, version and reconciliation. Automate checks against the approved mappings rather than forcing an artificial consensus.

Data and software delivery share version control, CI/CD and operational discipline. Data adds continuing obligations for source rights, historical corrections, semantic compatibility and consumer reconciliation. A code rollback may not undo a data change or an external transaction.

Design references checked 29 October 2026: Data mesh principles · Document-level access control · RAG security. These additions are RJL implementation recommendations, not a vendor certification or a compliance determination.

Give engineers and agents the same approved starting point.

A diagram explains the design. Versioned contracts, templates and tests make its boundaries reproducible.



RJL delivery design. An agent-prepared change remains subject to accountable review.

Store the approved business intent, product and tool schemas, semantic mappings, policy references, quality commitments, evaluation cases and runbooks in controlled repositories. Give each release a manifest of compatible versions. Stable interfaces need explicit deprecation and migration rules; they are not promises that contracts never change.

An IDE such as Cursor can retrieve approved specifications and propose implementation changes. Supply only the metadata and sample data allowed for that development environment. Keep production secrets and unnecessary sensitive records outside prompts, local indexes and development logs. Treat external repository and retrieved text as untrusted input.

CI checks the proposed change, including denial and isolation tests. Deployment promotes approved artifacts; runtime reconciliation detects drift between intended and actual policy. Stewardship agents can open repair proposals or perform explicitly preauthorized low-impact repairs. Material meaning, access and retention changes require the designated human authority.

Design references checked 29 October 2026: Data mesh principles · Document-level access control · RAG security. These additions are RJL implementation recommendations, not a vendor certification or a compliance determination.

Checked 14 October 2026.

Numbers in the text refer to this list. Product documentation establishes documented behavior and prerequisites, not comparative superiority or business results. Release status changes often; confirm the applicable cloud, edition and version before relying on a feature. Recommendations, thresholds and the worked example are RJL's proposed operating model and illustrative values, not client results. Prepared by Richard White. Version 1, October 2026.

1. Google, The Site Reliability Workbook, ch. 13: Data Processing Pipelines. https://sre.google/workbook/data-processing/
2. NIST, AI 600-1, Generative Artificial Intelligence Profile, July 2024. https://nvlpubs.nist.gov/nistpubs/ai/NIST.AI.600-1.pdf
3. Anthropic, Building Effective AI Agents, December 2024. https://www.anthropic.com/engineering/building-effective-agents
4. Snowflake, Data governance skills for Cortex Code. https://docs.snowflake.com/en/user-guide/governance-skills
5. DataHub, MCP server feature guide: read, mutation and proposal tools. https://docs.datahub.com/docs/features/feature-guides/mcp
6. OWASP, LLM06:2025 Excessive Agency. https://genai.owasp.org/llmrisk/llm062025-excessive-agency/
7. Snowflake, Introduction to data quality and data metric functions. https://docs.snowflake.com/en/user-guide/data-quality-intro
8. Databricks, Anomaly detection (Public Preview): freshness and completeness. https://docs.databricks.com/aws/en/data-quality-monitoring/anomaly-detection/
9. Microsoft, Document-level access control in Azure AI Search: GA security filters, preview permissions, synchronization. https://learn.microsoft.com/en-us/azure/search/search-document-level-access-overview
10. Google, The Site Reliability Workbook, ch. 2: Implementing SLOs. https://sre.google/workbook/implementing-slos/
11. Databricks, Manage and work with foreign catalogs (read-only queries). https://docs.databricks.com/aws/query-federation/foreign-catalogs
12. Databricks, External lineage: not recorded in lineage system tables. https://docs.databricks.com/aws/data-governance/unity-catalog/external-lineage
13. Snowflake, Govern Iceberg tables for external engines with Horizon Catalog. https://www.snowflake.com/en/developers/guides/govern-iceberg-tables-for-external-engines-with-snowflake-horizon/
14. Snowflake, Open Catalog overview: no first-time accounts; new customers directed to Horizon. https://docs.snowflake.com/en/user-guide/opencatalog/overview
15. Snowflake, Enforce data protection policies on Iceberg tables using the Scan Plan API (Public Preview, 10 Sep 2026). https://docs.snowflake.com/en/user-guide/tables-iceberg-enforce-access-policies-scan-plan-api
16. Bitol, Open Data Contract Standard. https://bitol-io.github.io/open-data-contract-standard/
17. OpenLineage, documentation: dataset, job and run model. https://openlineage.io/docs/
18. Snowflake, Accelerate Data Product Delivery with CoCo, updated 6 July 2026 (worked tutorial, synthetic data). https://www.snowflake.com/en/developers/guides/accelerate-data-product-delivery/
19. Snowflake, Power BI ingestion for Semantic View Autopilot, generally available 18 August 2026. https://docs.snowflake.com/en/release-notes/2026/other/2026-08-18-semantic-views-power-bi-ingestion-ga
20. Databricks, Genie Code. https://docs.databricks.com/aws/en/genie-code/
21. Wang et al., Match, Compare, or Select? An Investigation of Large Language Models for Entity Matching, COLING 2025. https://aclanthology.org/2025.coling-main.8
22. Databricks, Agent observability and quality with MLflow. https://docs.databricks.com/aws/en/mlflow3/genai/agent-observability-and-quality
23. NIST, AI RMF Playbook, GOVERN 1.7: decommissioning (voluntary guidance). https://airc.nist.gov/airmf-resources/playbook
24. Yoo et al., Meta-Reflective Evolution Effectively Improves Production AI Systems, arXiv:2610.12794, 8 October 2026 (research preprint). https://arxiv.org/abs/2610.12794

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.

PRODUCTS **Data Products in Practice**
Boundaries, contracts, quality and certification.

ECONOMICS **AI Token Economics**
Run cost, capacity and planned benefit by workflow.

ENGINEERING **Data Is Not Just Code**
Context, evidence and disciplined data delivery.

RJL SOLUTIONS

WEB rjlsol.com

EMAIL rwhite@rjlsol.com

Examples and figures are illustrative and describe no client engagement or client result. Vendor capabilities and release status were checked on 14 October 2026.