CLOUDERA

EBOOK

The Agentic Reckoning

Why control is the only AI strategy that survives 2026

Table of Contents

A Field Report
The Agentic Reckoning 3

Section 01
The Reckoning 5

Section 02
The Cost Wall 8

Section 03
The Governance Wall 11

Section 04
The Orchestration Wall 14

Section 05
What Control Looks Like in Practice 17

Section 06
The Proof 20

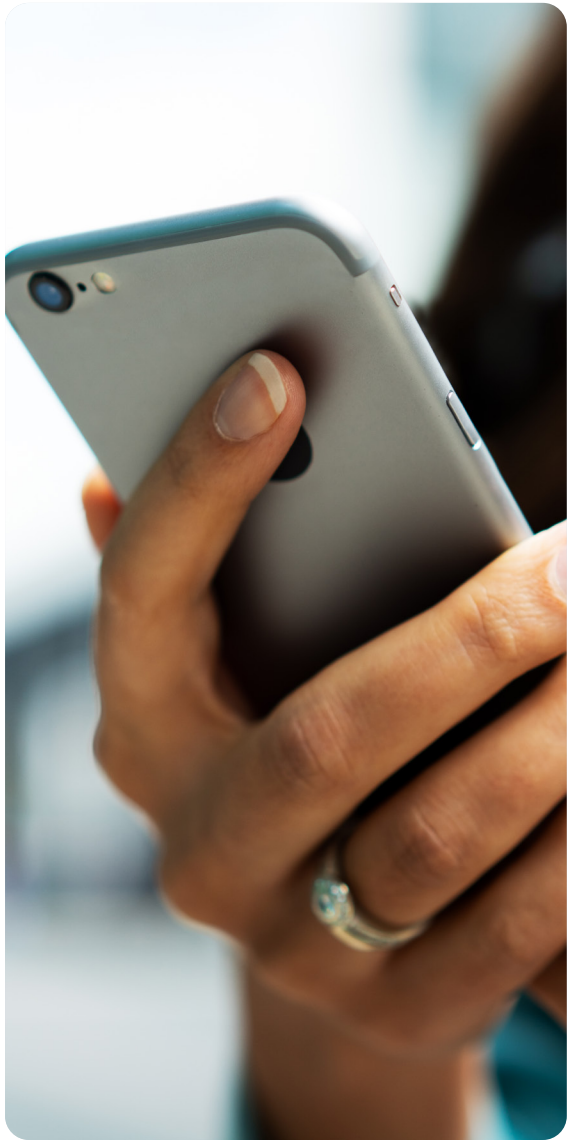
Section 07
The Operator's Checklist 24

CLOUDERA

A FIELD REPORT

The Agentic Reckoning

The Agentic Reckoning



Enterprises spent three years deploying AI agents as fast as they could build them. The bill has arrived, and it exposed a problem deeper than cost. We built systems we cannot see into: what they spend, what data they touch, whether their answers are true, and how many are running. Agentic AI hits three walls in production: cost, governance, and orchestration, and the same failure is underneath all three. You lost sight of your own system. What follows is a field report on getting the visibility back, the control that follows, and the architecture and measured numbers behind both.

SECTION 01

The Reckoning

The Reckoning

Three years ago the instructions were simple: deploy AI, put agents everywhere, and automate workflows one by one. The teams that moved fastest won the budget, and nobody asked what running at production scale would cost because at pilot scale, these projects cost almost nothing.

The bill has arrived.

$500 –2,000

per engineer, per month

That is what Uber's engineers were costing in agentic coding tokens when the company burned through its entire 2026 AI budget by April.[1] Weeks later Microsoft canceled most internal Claude Code licenses across the division that builds Windows, Microsoft 365, Outlook, and Teams, moving engineers onto its own tooling ahead of the fiscal year close.[2] Bryan Catanzaro, NVIDIA's VP of Applied Deep Learning Research, said it plainly: the cost of compute is far beyond the cost of the employees.[3]

[1]. Uber CTO Praveen Neppalli Naga, reported by The Information, April 2026. Per-engineer monthly API costs of $500 to $2,000 and the budget exhaustion are covered in Forbes, "Uber Burns Its 2026 AI Budget In Four Months On Claude Code," May 17, 2026: https://www.forbes.com/sites/janakirammsv/2026/05/17/uber-burns-its-2026-ai-budget-in-four-months-on-claude-code/ and Fortune, May 26, 2026: https://fortune.com/2026/05/26/uber-coo-ai-spending-tokens-claude-code/

[2]. Tom Warren, The Verge (Notepad newsletter), May 14, 2026. Microsoft's Experiences + Devices division was told to stop using Claude Code by June 30, 2026, the last day of Microsoft's fiscal year.

[3]. Bryan Catanzaro, VP of Applied Deep Learning Research, NVIDIA, to Axios, April 2026. Quoted in Fortune, April 28, 2026: https://fortune.com/2026/04/28/nvidia-executive-cost-of-ai-is-greater-than-cost-of-employees/

The most sophisticated AI operators on the planet could not model their own agentic economics until the invoice arrived. Retailers running on thinner margins will not survive the same approach. The reckoning is not that AI got expensive. It is that enterprises built systems they cannot see into. They cannot see what each agent costs to run, trace what data an agent touched, or prove it was allowed to access that data. They cannot tell whether an agent's confident answer is correct or hallucinated. They cannot map how many agents are running or which ones overlap. Three years of deploying agents everywhere produced a fleet of autonomous systems operating in the dark, and the dark is where cost, risk, and errors compound.

In the next 18 months, retailers will need to prove that they can control, secure, and govern AI.This challenge manifests as three walls every program hits when it moves from pilot to production: cost, governance, and orchestration Ultimately, these three walls reduce visibility into agentic workflows, which limits control over the system.

Three walls

Cost

Workloads run on the most expensive default

Governance

Agents act on data as the wrong identity

Orchestration

Crews multiply with no shared registry

Cost, governance, and orchestration each look like a separate failure. Each is a place where the enterprise loses sight of what its agents were doing.

In the next sections, we will explore these three walls in greater detail, and discuss how to overcome them.

SECTION 02

The Cost Wall

The Cost Wall

Start with the problem that already has invoices.

Agentic AI breaks budgets for a structural reason. A traditional software feature has a fixed cost per use. An agentic workflow does not. It calls a model, reasons over the result, calls a tool, reasons again, and potentially spawns a sub-agent. Every step burns tokens billed by the thousand. The same task costs wildly different amounts depending on how many reasoning loops it takes, which model runs it, and how often it fires. At pilot scale the cost is invisible. At production scale, with the whole organization running the workflow against a frontier API, the cost is the headline.

Uber's number is the canary in the coal mine, not an exception. They did not have a spending problem. They had a routing problem. Every workload, trivial and demanding alike, ran on the same expensive default, because no layer existed to decide otherwise.

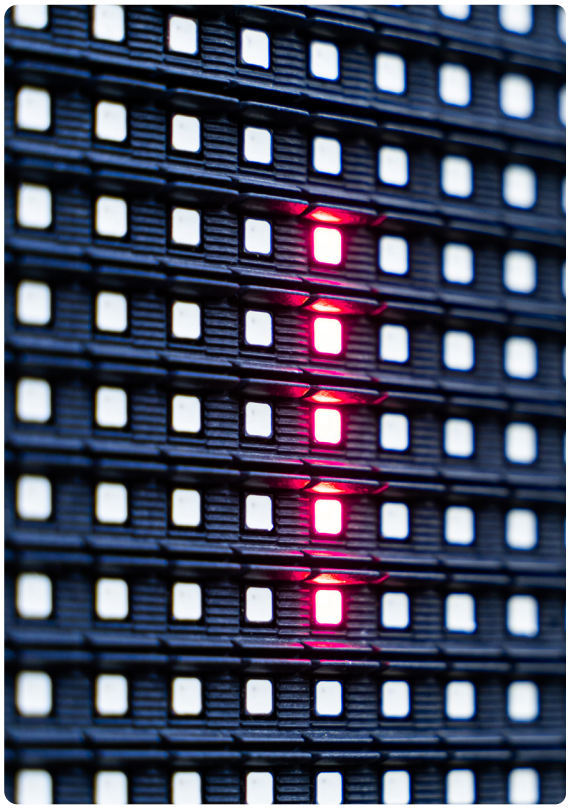
> **When you do not choose where a workload runs, you have chosen the most expensive option by omission.**

The solution is not to buy less AI. It is to engineer the portfolio. High-volume, repeatable, low-judgment work such as classifying a document, extracting fields, or scoring a record belongs on open-weight models running on compute the enterprise owns. That work should not be a per-token bill that scales with success. Frontier reasoning, the genuinely hard judgment calls, should stay on commercial APIs where premium rates buy premium capability. Most enterprises accidentally run the inverse. The cheap, repeatable work floods the expensive API because that is what the first pilot was wired to, and the expensive work is rare enough that no one notices it is subsidizing the waste.

The routing or orchestration layer that decides where each workload runs will be where organizations focus in 2026 and beyond. That is a sovereignty decision, not an optimization tweak. It decides whether your AI economics are yours or your vendor's.

You cannot engineer a portfolio you cannot measure. Routing requires knowing what each workload costs, and in a multi-agent system that number is not obvious. It lives in the execution traces: which agent fired, how many tokens it burned, and which tool it called.

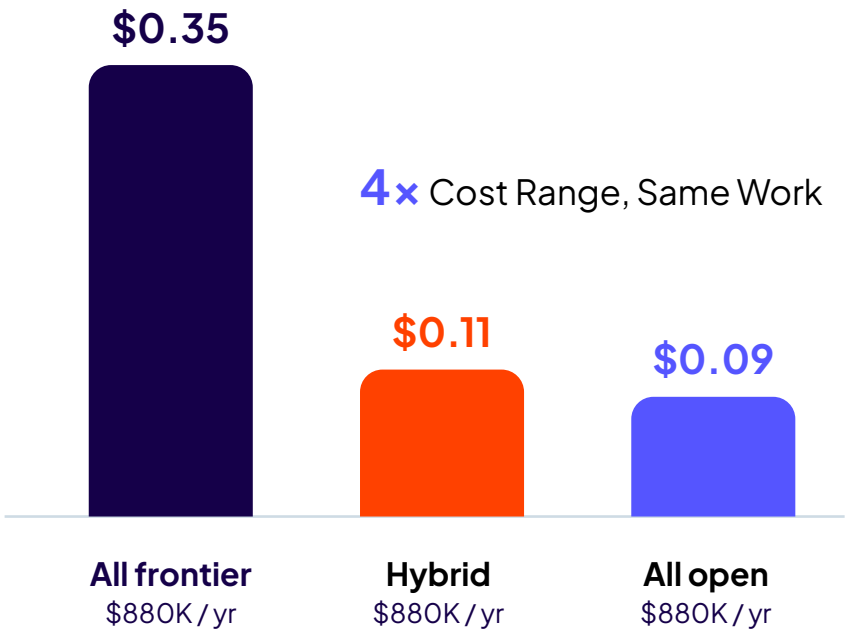
Without that telemetry, "move the cheap work to cheap compute" is a slogan you cannot act on, because you cannot tell the cheap work from the expensive work. Per-agent observability is the meter that makes the portfolio real.



To measure the costs per architecture, Cloudera built a five-agent product-innovation workflow in Cloudera Agent Studio, instrumented every agent, and ran the identical task under three configurations: every agent on a frontier commercial API, every agent on an open-weight, self-hosted model and a hybrid with four analytical agents on an open model andthe single creative agent on the frontier API.

The all-frontier run cost about $0.35. The hybrid cost about $0.11. The all-open run cost about $0.09. The token volume barely moved across the three, while the cost moved by a factor of four. The difference is the per-token rate, which is determined when an organization chooses where the workload runs.

Per-run cost for three configurations



A five-agent product-innovation workflow benchmark in Cloudera Agent Studio.

With ten thousand employees running this workflow class roughly two and a half million times a year, the all-frontier configuration costs about $880,000 annually and the hybrid costs about $278,000. It's the same workflow and the same output quality, but one architecture costs more than three times more.

Here is the part the routing skeptic needs to see. The frontier model on that one creative agent cost two and a half cents per run. The expensive model, placed exactly where creative judgment earns its price, added almost nothing to the bill. The cost lived in the analytical agents the open model ran, one of them especially. Running all five on frontier compute did not buy better answers on the four that did not need it. The customer pays a premium for work the open model handled fine.

The trace showed what the bill never would. A single agent drove the largest share of tokens in every configuration, as much as half, not because it reasoned more but because it made the most tool calls, and each call re-sent the growing transcript as new input. You cannot find that agent on a monthly invoice. You find it on a per-agent trace. Section 6 has the full breakdown of the benchmark.

The enterprises that build a routing layer will run agentic AI at a fraction of what the others pay.

SECTION 03

The Governance Wall

The Governance Wall

The cost wall has invoices.
The governance wall has incidents, and they stay quiet until they don't.

80%

of security teams report observing risky behavior from AI agents. Roughly two-thirds of organizations have no AI governance policy at all. Forrester expects an AI agent to cause a major enterprise breach in 2026, and the cause is mundane: governance built for human users was never extended to the non-human ones. **80% of organizations say their AI agents have already taken unintended actions, including reaching systems they were never authorized for**.[1] Among organizations breached in the past year, 68 percent had no policy to govern AI use or detect unsanctioned AI.[2] Forrester expects an agentic AI deployment to cause a publicly disclosed breach in 2026, and the cause is mundane: governance built for human users was never extended to the non-human ones.[3]

That is the whole problem in a sentence. An agent is mechanically a user. It authenticates, queries data, reads, writes, and calls tools. Every control that decides what a human can see was designed around a human identity. When an agent acts on your data, the real question is whose permissions the agent is operating under. That question is the one almost no one is asking, and it decides whether your governance framework holds up in the agentic era.

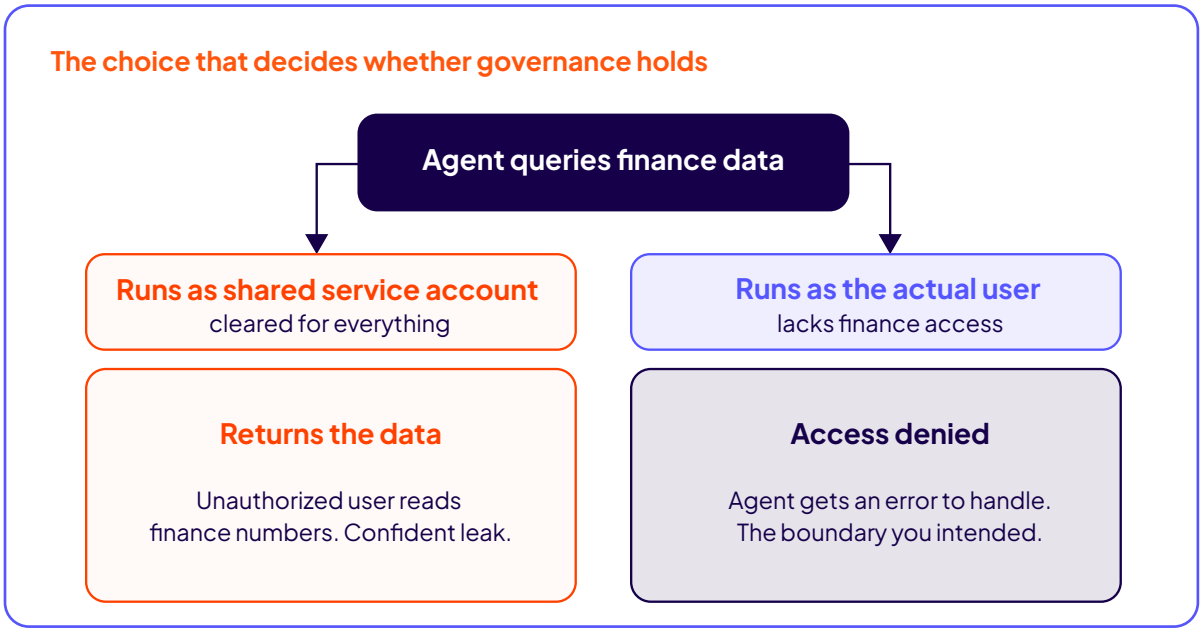
> **An agent is mechanically a user. The real question is whose permissions it operates under.**

1. SailPoint, AI Agents: The New Attack Surface, May 28, 2025. Global survey of 353 IT and security professionals conducted by Dimensional Research. 80% reported unintended agent actions, including accessing unauthorized systems (39%), sharing sensitive data (33%), and downloading sensitive content (32%). https://www.sailpoint.com/press-releases/sailpoint-ai-agent-adoption-report

2. IBM and Ponemon Institute, Cost of a Data Breach Report 2026, released July 29, 2026. Based on 602 organizations across 16 countries and 17 industries that experienced a breach between March 2025 and February 2026. 68% lacked governance to manage AI or detect shadow AI, up from 63% in 2025; 35% had no policy and 33% had one in development. Coverage: https://www.cybersecuritydive.com/news/data-breach-costs-ai-governance-ibm/826463/

3. Forrester, Predictions 2026: Cybersecurity And Risk. Senior analyst Paddy Harrington: an agentic AI deployment will cause a publicly disclosed breach and lead to employee dismissals. Coverage: https://www.infosecurity-magazine.com/news/forrester-agentic-ai-breach-2026/

Here is an example scenario. A workflow has three agents: one needs finance data, another needs marketing data, and the final one needs product data. Those datasets carry different access roles, and most employees are cleared for some of the data, but not all of it. When it comes to AI agents, most agentic design teams make access control choices without even realizing it.



Here is the distinction most vendors blur. The platform guarantees the denial, butit does not guarantee the workflow handles the denial well, and that part is yours to build. A denied agent can leave a silent gap, where the orchestrator quietly composes an answer from the data it did get and never flags the missing data. That is the worst case, because the model does not know it was denied unless you tell it. Or the denial can be caught and surfaced, so the output reads "this analysis excludes finance data you are not authorized for. Treat it as directional." It can alsohalt the workflow entirely when any required source is denied. Which one is right depends on the workflow. The platform cannot choose for you.

So access control is half of the governance wall, with two requirements: enforce against the right identity, and design the agent to treat a denial as a real signal, not a silent gap.

The other half has no enforcement engine at all. An agent can touch only authorized data and still hallucinate a number, invent a statistic, or ground its answer in the wrong half of the context. Access control asks whether the agent was allowed to see something. It says nothing about whether what the agent produced is true. That is output evaluation, a separate discipline, and it is the governance gap that almost no enterprise has closed. A confidently wrong answer the user was fully authorized to receive is still a governance failure. Access control was simply never built to catch it.

Two frameworks give this language, and your buyers may already use them. Forrester's AEGIS organizes agentic security across governance, identity, data, application, and threat domains, and its core idea, "least agency," is least privilege restated for non-human actors: agents get access to the narrowest permissions and toolsets they need. The OWASP Agentic AI Top 10 does the same on the risk side, cataloguing the failure modes this section describes in plain terms. If a leader in the room cites either, map it onto your architecture on the spot.

The governance wall has three jobs that often get lumped together: enforce what an agent can access against the right identity, record what it actually did, and evaluate whether its outputs can be trusted. Section 5 names which layer does which, including the one honest gap where the platform gives you a foundation and the rest is engineering.

SECTION 04

The Orchestration Wall

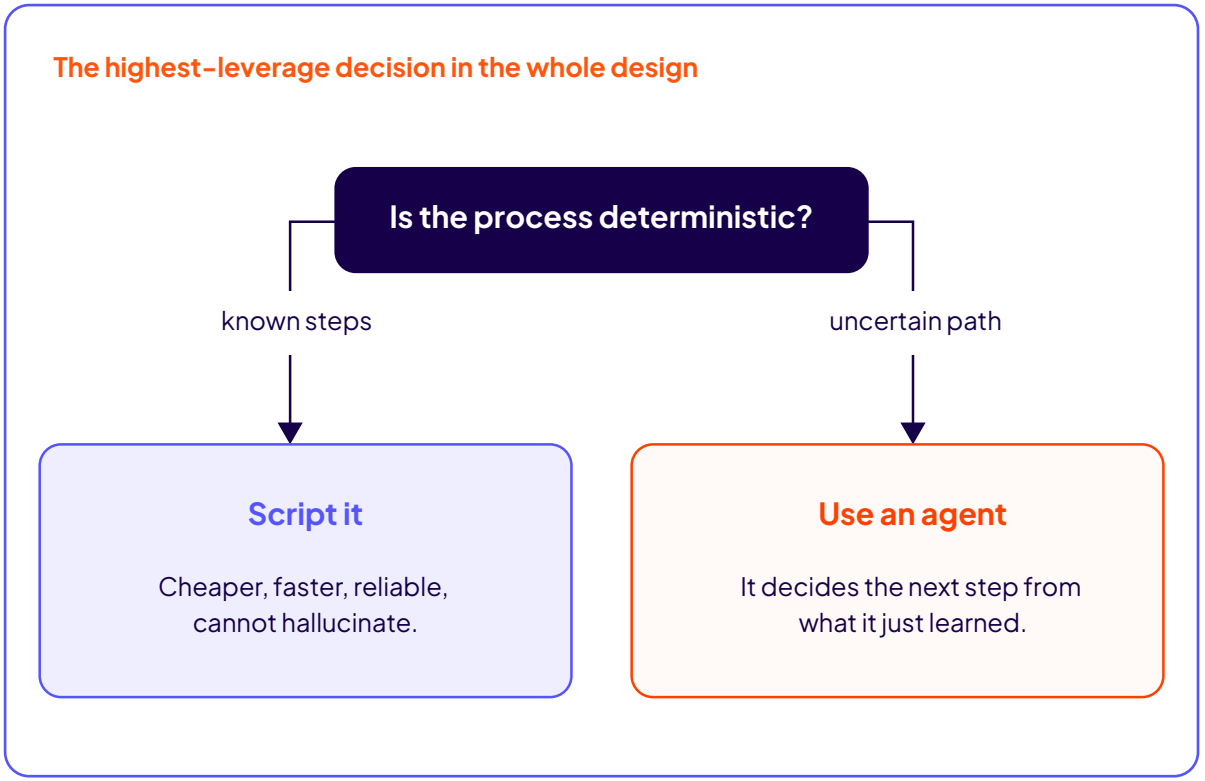
The Orchestration Wall

The third wall appears last, because it only shows up after the first agent works.

The sequence is predictable. One agent solves one problem. It works, so another team builds their own. A third builds two more. Within a year, the organization runs dozens of agents and crews built by different teams on different assumptions, with no shared registry of what exists. This is agent sprawl, the same pattern as shadow IT with one difference: shadow IT was inert until someone used it, while shadow agents act on their own. Forrester and Gartner both name the absence of central agent governance as a leading source of cost overruns and security gaps into late 2026.

Sprawl has three major impacts. It is duplicated spend, because five teams running overlapping crews on commercial APIs reproduce the Uber invoice inside your own walls. It is a duplicated risk, because every ungoverned crew is another confused deputy waiting to happen. And it is duplicated effort, because the same workflow gets built three times instead of being built once and then shared.

The discipline that prevents sprawl starts with a question asked before any agent is built: is this process deterministic or not?



CLOUDERA

Deterministic processes should be scripted, not made agentic. If the steps are known and the logic is fixed, a script is cheaper, faster, more reliable, and incapable of hallucinating. Wrapping that same process in a crew of reasoning agents adds cost and failure surface. Agents earn their cost on the other side of that line, where the path is genuinely uncertain and the system has to decide what to do next based on what it just learned.

When the process genuinely warrants agents, the next choice is whether to build a single capable agent with good tools or a coordinated crew. A single agent with a well-designed toolset handles more than most organizations expect and is far easier to observe, debug, and budget for. A crew is warranted when the work decomposes into specialized roles that genuinely benefit from separation, and even then it is often impractical and unnecessary. Beyond roughly six to eight specialists, coordination overhead and failure surface grow faster than the perceived benefits.

None of this is possible without seeing the topology. Organizations need visibility to identify duplicates, see which crews are running and retire redundancies, and inventory and govern fleets. Execution traces that are used to price workloads can be used to map orchestrations. They enable enterprises to see the fleet, decide what genuinely needs to be an agent, and stop paying to run the same workflow multiple times.

SECTION 05

What Control Looks Like in Practice

What Control Looks Like in Practice

Seeing your AI clearly takes five layers. Each answers one of the three walls. Bottom to top, here they are, with the Cloudera component that delivers each.

Routing
Decide where each workload runs
Agent Studio + AI Inference Service

Output Evaluation
Judge whether the answer can be trusted
Galileo

Behavioral Observability
See what agents did, and what it cost
PHOENIX

Access Governance
Control what agents reach
CLOUDERA SDX
Apache Ranger

Lineage
Know where the data came from
CLOUDERA Data Lineage

The data stays where it lives • Iceberg + NiFi, on infrastructure you control

Lineage

Know where the data came from.

Before you govern what an agent touches, you have to trace it. Cloudera Data Lineage maps lineage automatically across systems, so when an agent produces a number you can follow it to its source, and when a source changes you see everything downstream that breaks. One customer cut impact analysis from months to a single day, saving about a hundred research days a year. Another went from weeks to minutes, dramatically reducing labor-related costs associated with impact analysis.

Access governance

Control what agents reach.

This is Cloudera Unified Data Fabric, with Apache Ranger enforcing access at the row and column level consistently across cloud and on-premises environments. One bank used it to cut unauthorized access risk by over 90 percent with full audit traceability. Ranger enforces access controls against whatever identity the query runs as. The developer must still ensure that the agent runs as the actual user so Ranger evaluates the right person instead of a blanket service account. The platform makes per-identity enforcement possible. Building the agent to use it correctly is the work that separates a safe deployment from a confident leak.

Behavioral observability

See what agents did.

Cloudera Agent Studio records every step through Phoenix: which agent fired, which tool it called, how long it ran, and how many tokens it used, broken out per agent and per tool. This is the raw material for portfolio engineering. Token counts, multiplied by the price of whatever model each agent ran on, provides the per-agent cost, and that is the number that tells you which workloads to move to cheaper compute. Phoenix records. It does not judge. It will log a confident answer without knowing the answer is wrong.

Output evaluation

Judge whether the answer can be trusted.

This is the gap with no enforcement engine anywhere in the industry. An agent can touch only authorized data, run cheaply, leave a clean trace, and still invent a financial projection. Catching that inaccuracy means scoring the output for grounding. For this layer, Cloudera integrates Galileo, which measures whether an agent's output is supported by its inputs or fabricated. The most expensive failure in AI is not a wasted token. It is a confident wrong number an executive acts on.

Routing

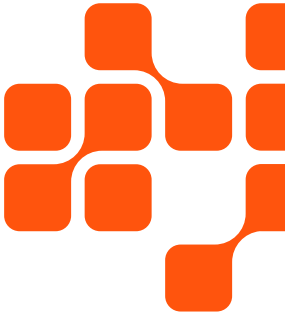
Decide where each workload runs.

Agent Studio orchestrates the agents and the Cloudera AI Inference Service serves the models, so the routing decision sits in infrastructure you control. Repeatable work runs on an open-weight model you host. Real reasoning runs on a frontier model.

You choose the infrastructure and the model on a per-workload basis, because you can finally see what each one costs.

Underneath all five, the data remains in place. Apache Iceberg as the table format and Apache NiFi as the movement layer mean an agent reasons over data in the environment you choose, not in whatever external API bills you by the token. The model comes to the data, not the other way around. Every layer above assumes you have not already handed your data to someone else's cloud to make the AI work.

These five layers provide the visibility and control needed for organizations to overcome the three walls and effectively and efficiently run AI.



SECTION 06

The Proof

The Proof

Here are three agentic builds, one for each wall, with defensible numbers.

Cost

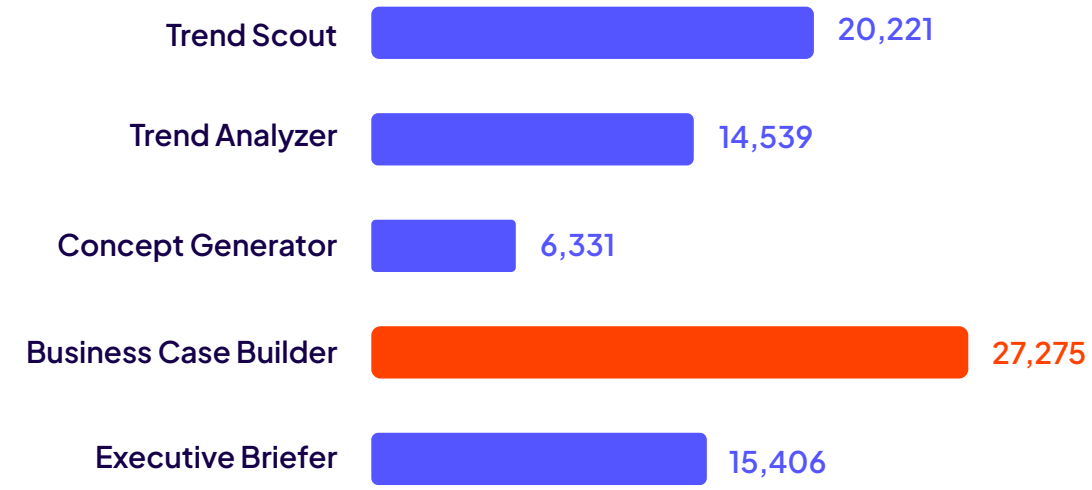
A 68 percent cut from routing on identical work.

We built a five-agent product innovation workflow in Cloudera Agent Studio: it scouts current trends, analyzes them against category data, generates concepts, builds the business cases, and writes the executive brief. We instrumented every agent and ran the identical task under three configurations: All five agents on a frontier commercial API, all five on an open-weight model we self-hosted host on our own Cloudera AI Inference endpoint, and a hybrid architecture with the four analytical agents on the open model and a single creative agent on the frontier API.

The all-frontier run cost about $0.35 per run. The hybrid cost about $0.11. The all-open run cost about $0.09. Scaled for a 10,000 employee organization running this one workflow class roughly two and a half million times a year, it would cost about $880,000, $278,000, and $223,000 respectively. The token volume across the three runs differed by less than a quarter, while the cost differed by a factor of four. The savings are in theper-token rate, which organizations can control by choosing where each agent runs.

Two findings from the trace are worth more than the headline. First, the workflow was overwhelmingly input, more than three-quarters of every run and as much as 86 percent. Almost nothing was related to the model generating new text. It was the same growing context re-sent on every tool call. Second, one agent, the business-case builder, drove the largest share of tokens in every configuration, climbing to half in the open and hybrid runs, because it made the most tool calls and each call re-billed the accumulated context as fresh input. Cost in an agentic workflow compounds with tool-loop depth, and it is overwhelmingly input. The only way to have that level of visibility is from per-agent traces.

Per-agent token use, all-frontier run



The Business Case Builder drove the largest single share.

The routing decision falls straight out of the trace. The frontier model on the single creative agent cost about two and a half cents per run. Placed where creative judgment might earn its price, the premium model added almost nothing to the bill. The cost lived in the analytical agents, the ones making repeated tool calls, and the open model ran them all fine.

Governance

A verdict grounded in real data, not assembled from a keyword.

The same build reproduced the governance failure from section 3. Asked to assess cannibalization risk for a proposed wearable, the business-case agent rated it a moderate risk because the words “Fitness Tracker” appeared in the affinity data. What the data actually said was narrow: a fitness tracker had once been bought alongside a resistance band, a 0.5 affinity score with no bearing on a premium wearable. The agent had real co-purchase data through a SQL tool. It still built a risk verdict on a string match rather than on what the data showed. Grounding the retrieval is necessary, but it is not sufficient. Without a layer that scores the output against the data it claims to rest on, a confident verdict assembled from a keyword reaches the executive with no flags.

A frontier configuration happened to not make that error in its run. That is not a defense. Behavior varied across models and across runs of the same model, and you cannot ship a system that depends on which model or run stayed honest. The control has to sit in the platform, not in the hope that the model behaves.

The corrected pattern is a separate build, a new-item evaluation platform. A supplier submits a product. The system returns a buy, decline, or modify verdict with a cannibalization estimate. The estimate is calculated from actual sales and affinity data through SQL, not generated by a model. The verdict comes from a fixed decision matrix, not a model’s opinion. When a near-identical protein bar is proposed into a category already holding 29 of them, with eight of them categorized as high-velocity, the system declines it and shows the existing sellers it would cannibalize. The number traces back to its source, which is the difference between a financial recommendation a category leader can act on and one they cannot. Category teams review new-item submissions over days and weeks. A grounded, governed workflow returns the verdict in seconds.

Orchestration

A bounded crew, and the instrumentation that catches it when it sprawls.

The same instrumentation that priced the cost experiment caught something the output never showed. In two of the three runs, the platform's loop detection fired. An agent had begun repeating itself, re-calling a tool it had already called, thrashing against a step that was deterministic and never needed a reasoning loop. The run still completed. The brief still looked finished. Only the trace showed the thrash. That is the orchestration wall in one image: a workflow can hand you a clean-looking answer while an agent loops underneath it, and you only know if the platform is watching.

The fix is to bound the orchestration so it cannot sprawl, then catch it when it does. The third build shows the bound. It is a store workforce scheduling workflow, not a sprawl of agents but a bounded sequential pipeline that forecasts demand, assigns staff, and checks costs,compliance, and quality. The deterministic parts, which include pulling labor data and checking compliance rules, are tool calls. Only the genuinely uncertain part, balancing competing constraints into a recommendation, is left to reasoning. It also handles the thing most demos skip. When an agent times out, the workflow does not crash. It falls back to a valid schedule and continues. A system that only works on the days nothing goes wrong is not a system. This one was built to fail safely, which is what governable orchestration actually looks like.

A note on the numbers. The cost figures come from real Phoenix traces of the workflow, measured per agent across three runs. Frontier pricing uses a published commercial rate of $2.50 per million input tokens and $15.00 per million output. The open comparison uses a public hosted rate of $0.50 input and $3.00 output, chosen deliberately so the percentages stay sourceable rather than resting on internal GPU economics, which only widen the gap at production utilization. The annual figures assume one workflow class run roughly two and a half million times a year at a ten-thousand-employee organization.

SECTION 07

The Operator’s Checklist

The Operator's Checklist

Eight questions to ask yourself about agentic systems running in your enterprise

Cost

Do you know what your AI costs to run?

- [ ] **1. Can you see token consumption per agent?**

  If you only have the aggregate, you cannot tell which workflow is expensive, and you cannot fix what you cannot isolate.

- [ ] **2. Do you decide where each workload runs, or does everything default to the same frontier model?**

  Repeatable, low-judgment work rarely needs the most expensive model. It is the most common waste and the easiest to fix once you can see it.

- [ ] **3. If your AI usage tripled next quarter, do you know what it would cost?**

  Repeatable, low-judgment work rarely needs the most expensive model. It is the most common waste and the easiest to fix once you can see it.

Governance

Can you control and trust what your agents do?

- [ ] **4. When an agent queries data, whose permissions is it using: the end user's or a shared service account?**

  On a shared service account, every user of that workflow inherits the access of the most privileged identity behind it, whether they are cleared for it or not.

- [ ] **5. Can you produce, on demand, a record of what data a given agent touched?**

  If your cannot, or if it takes weeks, then you have no working audit trail.

- [ ] **6. Is anything checking whether your agents' outputs are correct, or only whether they ran?**

  An agent can be perfectly governed on access and still invent the number an executive acts on.

Orchestration

Can you see and govern the whole fleet?

- [ ] **7. Does this process need agents at all?**

  If the steps are fixed and the logic is known, a script is cheaper, faster, and cannot hallucinate. Reserve agents for genuinely uncertain work.

- [ ] **8. When an agent in a workflow fails or times out, what happens to the rest?**

  If the workflow crashes, or you are not sure, the system works only on the days nothing goes wrong.

About Cloudera

Cloudera is the only hybrid data and AI platform company that large organizations trust to bring AI to their data anywhere it lives. Unlike other providers, Cloudera delivers a consistent cloud experience that converges public clouds, on-prem data centers, and the edge, leveraging a proven open-source foundation. As the pioneer in big data, Cloudera empowers businesses to apply AI and assert control over 100% of their data, in all forms, improving security, governance, and real-time and predictive insights. The world's largest brands across all industries rely on Cloudera to transform decision-making and ultimately boost bottom lines, safeguard against threats, and save lives.

To learn more, visit **Cloudera.com** and follow us on **LinkedIn** and **X**.

CLOUDERA

Cloudera, Inc. | 6220 America Center Dr, 5th Fl, San Jose, CA 95002 USA | cloudera.com

9619-001 Sept 16, 2026