

The AI gateway

The IT leader's guide to the context and control layer behind production AI.

September 2026

CDATA

Foreword

Most organizations running AI today are over budget, under-governed, and building on architectures that were never designed for production AI at scale. The gap is the layer between users, agents, systems and models: how traffic is routed, how data is reached, and who is allowed to do what.

This guide works through the AI gateway from first principles: why it emerged, what it controls, where routing alone falls short, and how to measure whether it's working once it's live. You'll walk through eight short parts covering LLM gateways, MCP gateways, and the governance, context, and connectivity layers that most gateways overlook today.

Read it in order or jump to the part you need. Either way, our aim is to give data and IT leaders a clear, vendor-honest map of the category before you commit budget to it.

– The CData Connect AI team

Contents

The series, at a glance

01	Why enterprise AI is over budget and under-delivering Three structural gaps, routing, data access, and governance, behind most of the waste.	05
02	What is an AI gateway, and why it matters now The centralized control layer between AI apps and the models, tools, and data they use.	07
03	What an MCP gateway does that an API gateway can't Routing intent and enforcing tool policy, not just HTTP requests.	09
04	What is an LLM gateway, and where it fits in the stack Model selection, cost optimization, and failover, but not the data problem.	11
05	Why a gateway without context just moves the problem Controlling model traffic without governing data relocates the failure downstream.	13
06	What to look for in a context layer What separates real data governance from a pass-through proxy.	15
07	Use cases an enterprise AI gateway makes possible From natural-language queries over enterprise data to AI-powered BI without SQL.	17
08	How to measure AI gateway success after deployment ROI tracking, audit logging, and cost-per-outcome metrics that prove it works.	19
—	Glossary – terms used in this series	21

By the Numbers

Why this matters, in six figures

93%

of enterprises are already over their AI budgets.

9 in 10

firms report no productivity effect from AI, the ones using it average 1.5 hours a week.

21.7%

now cite direct financial impact as their primary AI ROI metric, nearly double.

97.6%

token reduction from schema-aware retrieval on a governed multi-source query.

98.5%

grounded accuracy across 378 real-world prompts in CData's MCP benchmark.

<1 in 5

organizations track well-defined KPIs for gen AI, the practice most tied to impact.

The Problem

Why enterprise AI is over budget and under-delivering

93% of enterprises are already over their AI budgets. Three structural gaps, routing, data access, and governance, account for most of the waste.

Enterprise AI was supposed to follow a predictable path: pilots prove value, a measured rollout follows, and ROI justifies the spend. What most organizations found instead is that production AI is expensive, hard to forecast, and largely ungoverned. McKinsey's May 2026 Enterprise AI FinOps survey found that 93% of organizations are already exceeding their AI budgets, with spend rising nearly fourfold as deployments scale from isolated use cases to enterprise-wide.

Part of the difficulty is that AI pricing behaves unlike any prior software category. Seat licenses are predictable; token consumption is not, usage can vary by as much as 30x running the same task across different workflows and prompts. Spend is also fragmented: McKinsey found 20–30% of AI spend routinely unaccounted for, scattered across cloud providers, model vendors, platforms, and business units with no consolidated view.

Three structural gaps drive the overruns

Routing. Without an intelligent routing layer, every request lands on the same frontier model, whether it's complex reasoning or a simple lookup. Research on complexity routing (RouteLLM, ICLR 2025) held ~95% of frontier-model performance while sending only 14–26% of requests to the expensive model, a 75–85% cost reduction on routed workloads. Most teams capture none of it because that layer was never built.

Data access. When agents fetch enterprise data without a schema-aware layer, they get everything, full schemas, unfiltered records, surrounding metadata, and the model pays to process all of it. CData's own benchmarks found a 97.6% reduction in token consumption on governed multi-source queries compared with unfiltered access.

Governance. Only 20–25% of organizations have mature AI FinOps practices, so cost overruns become visible only after the invoice arrives, by definition, after the optimization window has closed. Without per-request attribution tagged to team, use case, and workflow, there's no way to trace waste back to its source.

A production problem, not a pilot problem

Most enterprises have reached a specific moment: AI is live, agents are running, and the first full procurement cycle has closed. The accountability problem has a concrete shape. Which team generated last month's overage? Which agent workflow looped 500 times unnecessarily? Which fetch returned 50,000 rows when the use case needed 12? Attribution infrastructure is the prerequisite for optimization work, not an optional reporting layer to add once things settle.

About a third of organizations McKinsey surveyed have already achieved 20–30% cost reductions through active optimization, and what separates them is architecture, not discipline: intelligent routing, schema-aware data access, cost attribution, and budget enforcement, treated as infrastructure rather than four sequential projects. For a company spending \$5 million a year on AI, that's \$1–1.5 million in recoverable spend.

Key takeaway

The problem isn't the models. It's an architecture that was never designed for production AI at scale.

Read the full guide – cdata.com/blog/enterprise-ai-cost-control

The Category

What is an AI gateway, and why it matters now

An AI gateway is the centralized control layer that sits between your AI applications and the models, tools, and data they use.

Enterprise AI has become a distributed-systems problem. A single organization now runs copilots, autonomous agents, and AI features spanning several model providers, each reaching into a different set of internal tools and data, and most of that traffic flows with no shared point of control. The result is familiar: costs no one can fully attribute, security gaps that surface one integration at a time, and no consistent record of what the AI did. The AI gateway is the architectural response to that gap.

Instead of each application integrating directly with each provider and each data source, applications integrate once with the gateway, and the gateway handles routing, security, and governance uniformly on the other side. That single point of contact turns a sprawl of point-to-point connections into something an organization can reason about.

What a production gateway controls

A production AI gateway governs a defined set of control domains: authentication and access control, request routing and model selection, rate limiting and cost enforcement, data access and permission filtering, prompt inspection and guardrails, and observability and audit logging. The payoff is that these policies live in one place, teams write authentication, budgets, and audit trails once, and every AI interaction inherits them.

An API gateway isn't a substitute. AI traffic is billed by the token, streams back incrementally, carries prompt-injection risk, and is agent-to-tool rather than client-to-endpoint, none of which an API gateway was built to handle. The category is still early, Gartner projects that 70% of engineering teams building multi-model applications will use AI gateways by 2028, up from roughly 25% in 2025.

When does an enterprise need one?

Most organizations cross into gateway territory when more than one of these is true at the same time:

- 1 AI applications run across more than one LLM provider, and the organization needs unified cost tracking and automatic failover.
- 2 Agents can reach enterprise data, but no centralized mechanism enforces what each user or agent may see.
- 3 AI spend runs without per-team budget enforcement, and no alert fires before a runaway workflow exhausts the budget.
- 4 A compliance obligation requires an audit trail of every AI query where none exists.
- 5 Several teams independently integrate the same enterprise data sources, duplicating work with no shared governance.

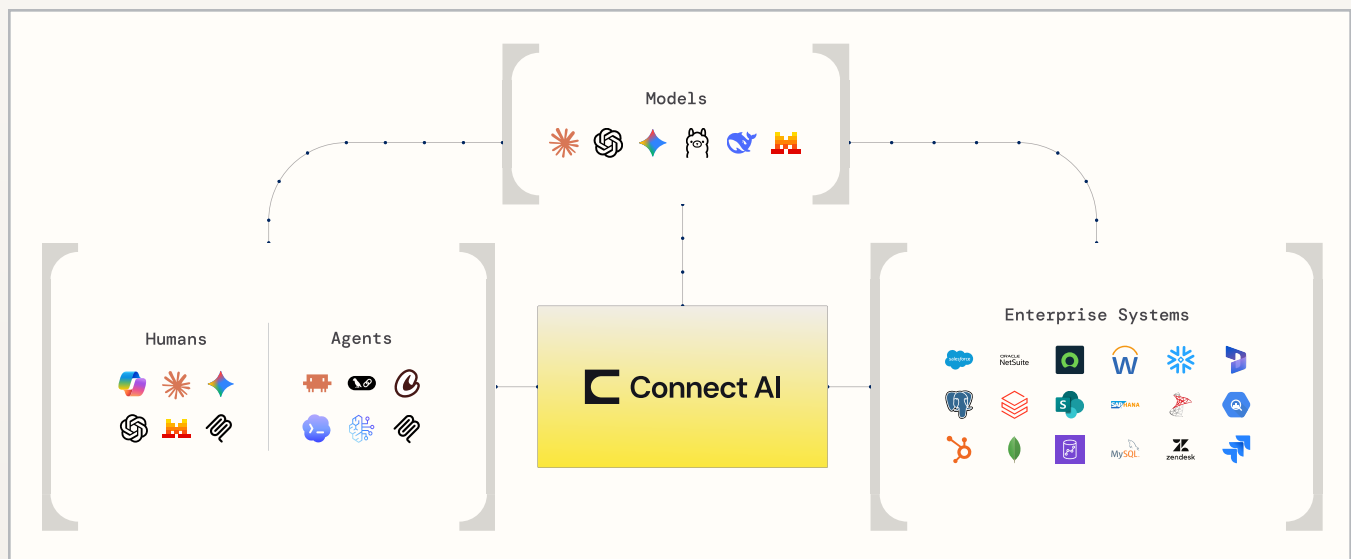


Figure 1. The AI gateway as a single control layer between AI apps and agents and the models, tools, and data they use.

Read the full guide – cdata.com/blog/what-is-an-ai-gateway

MCP vs API

What an MCP gateway does that an API gateway can't

An API gateway routes HTTP requests. An MCP gateway routes intent, enforces tool policies, and governs what your agents are allowed to do.

Enterprises are giving AI agents access to internal systems faster than they are deciding how that access should be governed. Teams stand up Model Context Protocol (MCP) servers so agents can reach source control, databases, ticketing, and internal APIs, and each server grants an agent the ability to run commands, modify records, and trigger workflows. Without a central point of control, the blast radius of a single misconfigured server has no natural limit.

An MCP gateway is a reverse proxy purpose-built for MCP traffic. It sits between agents and the servers they call, authenticating and governing every tool invocation before it reaches a backend system. An API gateway, mature as it is, can't stand in for one: it governs endpoint-level access for HTTP services, while an MCP gateway governs agent-to-tool semantics.

Three things it governs that an API gateway can't

Tool discovery. An agent receives a list of available tools and decides which to invoke, so deciding which tools each agent may even see becomes a governance decision in its own right. A gateway exposes only the tools appropriate to a role and context, which also keeps tool definitions from bloating the model's context window.

Identity propagation. In production an agent acts on behalf of a user. The gateway carries that user's identity into every tool call, so backend systems enforce per-user permissions rather than granting the agent its own standing access.

Human-in-the-loop and audit. Sensitive actions, database writes, workflow triggers, deletions, can be paused mid-workflow for approval, a control classic API gateways have no equivalent for. And every tool call is logged with its identity and result, the record compliance teams require and debugging depends on.

Gateway vs. server, and where authentication starts

The distinction from an MCP server is a matter of layer. A server exposes tools, it connects an agent to GitHub, Postgres, or Slack and executes the calls. The gateway is the control plane above those servers, deciding which agent can see which tools, under which identity, and with what constraints. Authentication is where that difference first becomes concrete: the 2025 MCP specification standardized on OAuth 2.1, and a gateway acting as the OAuth resource server validates tokens and enforces scopes before any tool runs.

API gateway

Routes and secures HTTP requests. Knows endpoints, not intent.

MCP gateway

Routes intent, enforces tool policy, and governs what agents may do.

Read the full guide – cdata.com/blog/mcp-gateway-vs-api-gateway

What is an LLM gateway, and where it fits in the stack

The LLM gateway is the model-routing layer inside a broader AI gateway. It handles model selection, cost optimization, and failover, but it doesn't solve the data problem.

In early 2026, Uber reportedly burned through its entire annual AI budget in four months after encouraging engineers to use AI coding tools as heavily as possible, every request metered by the token, subject to no ceiling and no attribution, until the year's budget was gone in April. It's the public version of a story playing out inside many organizations.

An LLM gateway is the infrastructure layer that brings model traffic under control. It sits between your applications and the model providers they call, exposing a single API while centralizing multi-model routing, cost tracking, semantic caching, observability, and failover. Integrate once with the gateway, and switching a model or adding a fallback becomes a configuration change rather than a rewrite.

Where it fits, and where it stops

Model routing is where most of the savings come from: examine each request and send it to the model best suited to the task, reserving frontier pricing for the work that needs it. Semantic caching compounds the effect by returning a stored answer when a new prompt is close enough in meaning to a recent one. Alongside cost control, the gateway earns its place on reliability, observability on every call and automatic failover when a provider degrades.

Picture a production deployment as three layers: the application layer where features live, the model layer governed by the LLM gateway, and the data layer where enterprise data lives. Most teams reason carefully about the first two and underbuild the third, and that third layer is where a large share of stalled AI pilots trace back to. The LLM gateway routes traffic to models efficiently; it does not decide how a model gets authorized access to current data. That problem lives at the data layer.

Four functions define a proper LLM gateway

It handles routing and failover, deciding which model serves each request and where traffic goes when a provider degrades. It handles cost tracking and enforcement, attributing every token to a team or project and capping spend before it runs away. It provides observability, logging routing decisions, latency, and token usage for each call. And it applies security guardrails, screening requests before they reach a provider. The deeper benefit is decoupling: when an application talks to one stable interface rather than a provider's SDK, the choice of model becomes a runtime decision instead of an architectural commitment.

For chatbots and code assistants calling model APIs directly, an LLM gateway is the right primary control layer; for agents reaching enterprise tools and data, it's necessary but not sufficient.

Key Takeaway

Model routing is one layer of the gateway, not the whole of it. Efficient routing to bad data is still a bad answer.

Read the full guide – cdata.com/blog/what-is-an-llm-gateway

Why a gateway without context just moves the problem

A routing gateway that controls model traffic without governing data access just moves the failure point downstream.

Assume you already have a gateway: model traffic routed, costs controlled, maybe MCP governance for tool calls. The traffic layer is handled, and yet the AI still returns answers that are subtly wrong, adoption plateaus, and the security review keeps asking a question the gateway can't answer: how does the model know it's allowed to see this data? Routing gateways govern how requests are handled. They do not govern what the AI knows.

A well-routed wrong answer is still a wrong answer. Gallagher's 2026 survey of 1,200+ businesses found AI errors and hallucinations top the list of adoption threats, cited by 57% of respondents, and most are data-context failures, not model failures, which no amount of routing tuning can fix.

The three kinds of enterprise context

Live company data. Operational records queried in real time, not copied in advance. A nightly index is a snapshot that has already moved on, for anything time-sensitive, that lag produces confident wrong answers.

Semantic definitions. The business meaning behind the fields. An AI that queries Salesforce correctly but doesn't know what "Amount" refers to will misinterpret data it retrieved correctly.

Tacit knowledge. The implicit vocabulary and reasoning shortcuts a team uses to read its own data, the hardest layer to formalize, and where the category is still heading. Correctness lives at the intersection of all three.

What routing does well, and can't touch

A routing gateway centralizes model selection, enforces rate limits and cost controls, propagates identity, and logs traffic. These are real problems, solved well. But a routing gateway sits between your applications and your models; it never touches the enterprise systems that hold the data the AI is asked about. Where the model gets its figures, how current they are, and whether they're filtered to what this user may see are not routing questions.

Without context

- ✗ Guesses the table, metric, and join
- ✗ Re-explores the schema every request
- ✗ Burns tokens, returns answers you can't trust

With context

- ✓ Knows the definition, the source, the join
- ✓ One precise call, no exploration
- ✓ Accurate answers, a fraction of the cost

Figure 2. Routing alone moves the failure downstream; a governed context layer resolves it before the model runs.

Read the full guide – cdata.com/blog/ai-gateway-context-layer

Evaluation

What to look for in a context layer

Not all context layers are equal. Four things separate real data governance from a pass-through proxy.

Some products labeled "context layer" govern nothing, they pass requests through untouched. A context layer is distinct from a RAG pipeline (which retrieves documents from a periodic index by similarity), a BI semantic layer (which describes data for reporting but doesn't serve it to a model), and a data catalog (which inventories without querying). Its job is specific: provide access to the right live systems, correctly interpreted and correctly scoped, at the moment a model needs it.

Whether you build or buy, evaluate any context layer on four criteria, and on governance depth, not connector count alone.

Criterion 1

Source coverage

Reaches the systems your workflows actually touch, cloud, on-premises, and custom fields, not just what a pilot happened to hit.

Criterion 2

Schema intelligence

Understands relationships and business meaning, and resolves questions that span systems, not just isolated connectors.

Criterion 3

Permission enforcement

Every query runs under the requesting user's own credentials, enforced at the data layer, with per-query audit logs.

Criterion 4

Measurable accuracy

Grounded-accuracy and token-efficiency numbers on a workload like yours. A team that can't produce them hasn't measured what matters.

Where RAG reaches its limit

RAG is the right default for unstructured, content-heavy work, knowledge bases, documentation, long-text summarization. It reaches its limit on operational data in three ways: index lag, where a snapshot trails live systems; permission blindness, where retrieval ranks by similarity rather than authorization; and structured mismatch, where joining Salesforce accounts to NetSuite invoices takes query logic, not similarity search. For most enterprise deployments the conclusion is not either-or but both.

The criteria in depth

Permission enforcement. Every query should run under the requesting user's own credentials, not a shared service account every agent inherits, applied at the data layer, because application-layer enforcement operates after data has already been retrieved. Deployments that get it wrong either over-scope and surface records the user was never cleared to see, or lock access down so hard the AI can't answer real questions, the diminished-deployment outcome 81.6% of organizations reported in EMA research. Per-query audit logs tied to user identity are not optional in regulated industries, yet Deloitte's 2026 State of AI found only 21% have a mature governance model for agentic AI.

Measurable accuracy. Context-layer failures look exactly like model failures unless there is observability at the data-access layer. In CData's own testing, schema-aware retrieval cut token usage by 97.6% on a governed multi-source query spanning Salesforce, Snowflake, and ServiceNow, and a separate benchmark measured 98.5% grounded accuracy across 378 real-world prompts, against 65–75% for other providers. The closing move in any evaluation is to request the same evidence on a workload that resembles yours.

Read the full guide – cdata.com/blog/context-layer-for-ai

What It Enables

Use cases an enterprise AI gateway makes possible

From natural-language queries over enterprise data to AI-powered BI without SQL, the use cases a properly architected gateway opens up.

The most valuable enterprise AI applications rarely just generate text. They answer a real business question or take an action by reading live data from the systems your teams run, and return something current and correct. That's exactly where adoption stalls, the distance between a convincing demo and a value-add feature is almost always a data-connectivity and context gap, not a limit on the model's intelligence.

With routing and context both in place, capabilities that were impractical to govern become routine to ship. Every pattern below rests on the same foundation: per-user permission enforcement, full query audit logging, and schema-aware access, applied before any record reaches the model.

Self-service BI	Plain-English questions over a live warehouse, a figure, chart, or summary without SQL or schema knowledge.
CRM & ERP copilots	Current pipeline from Salesforce and actuals from SAP or NetSuite in one exchange, a join problem before it's an AI problem.
Cross-system search	One entry point for answers scattered across Jira, ServiceNow, HRIS, and wikis, with document-level permissions honored.
Write & actions	Open a ticket, advance an opportunity, correct an order, acting on the live system, logged and revocable.
Regulated industries	Healthcare, finance, and government access where governance is a precondition, with an audit trail that satisfies review.

Table 1. Five use cases a properly architected gateway makes deployable, and what each one returns.

What makes each one deployable

Self-service BI. Connected to a live warehouse, Snowflake, BigQuery, Redshift, a natural-language interface accepts a plain-English question, resolves it into the right query, and returns a figure, chart, or summary without the requester knowing SQL. What makes it deployable rather than risky is the data layer between the interface and the warehouse: it decides which tables and columns the user may see, records every query, and returns only the fields a question needs.

CRM & ERP copilots. The value appears when a copilot reads live Salesforce or HubSpot data and live SAP, NetSuite, or Workday data in the same exchange, which accounts flagged at risk in the CRM also carry open invoices in the ERP. Those answers live in two or more systems never designed to be queried together, so it's a join problem before it's an AI problem, and the copilot has to respect existing sharing rules at the data layer.

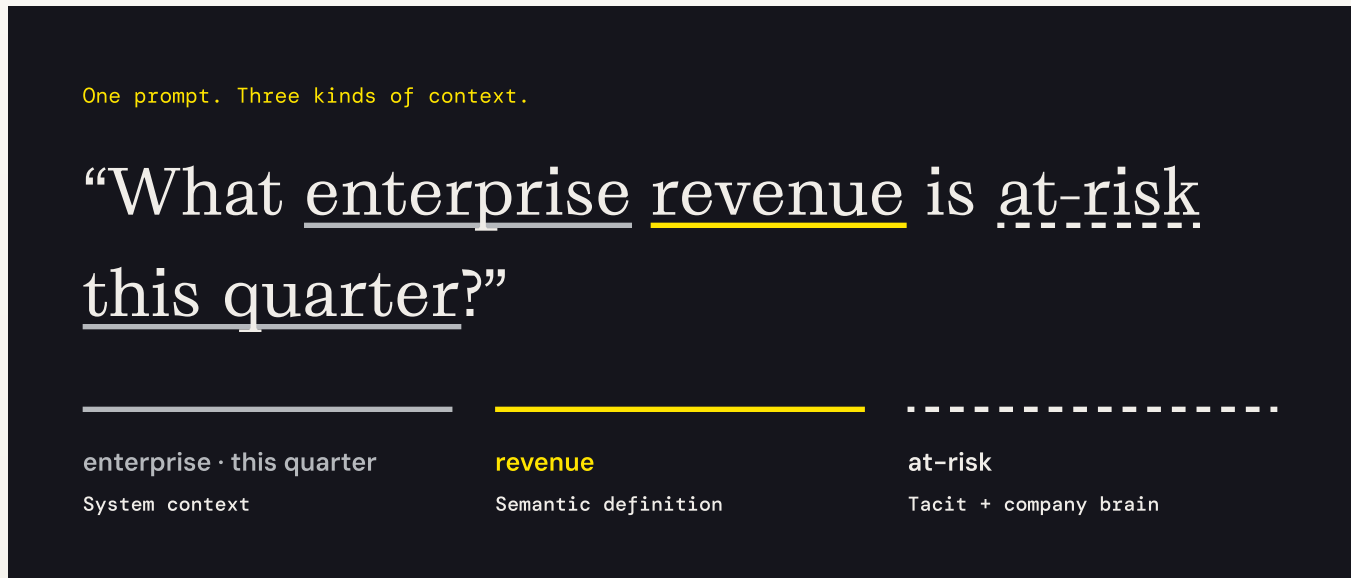


Figure 3. A single plain-language question carries several distinct kinds of context the gateway has to resolve.

Read the full guide – cdata.com/blog/ai-gateway-use-cases

Measurement

How to measure AI gateway success after deployment

ROI tracking, audit logging, cost-per-outcome metrics: how to know your gateway is working, and what success looks like in production.

By the time a gateway reaches production, the question changes to a plainer, harder one from the people who funded it: is it working? Adoption metrics, logins, usage volume, answer the wrong version of it. Success is whether decisions got faster, costs came down, and risk stayed contained.

Most organizations deploy without capturing baselines first, and without a pre-deployment reading of cost, latency, and governance coverage there is nothing to measure improvement against. Structured measurement separates the problem into three layers, each answering a different question for a different audience.

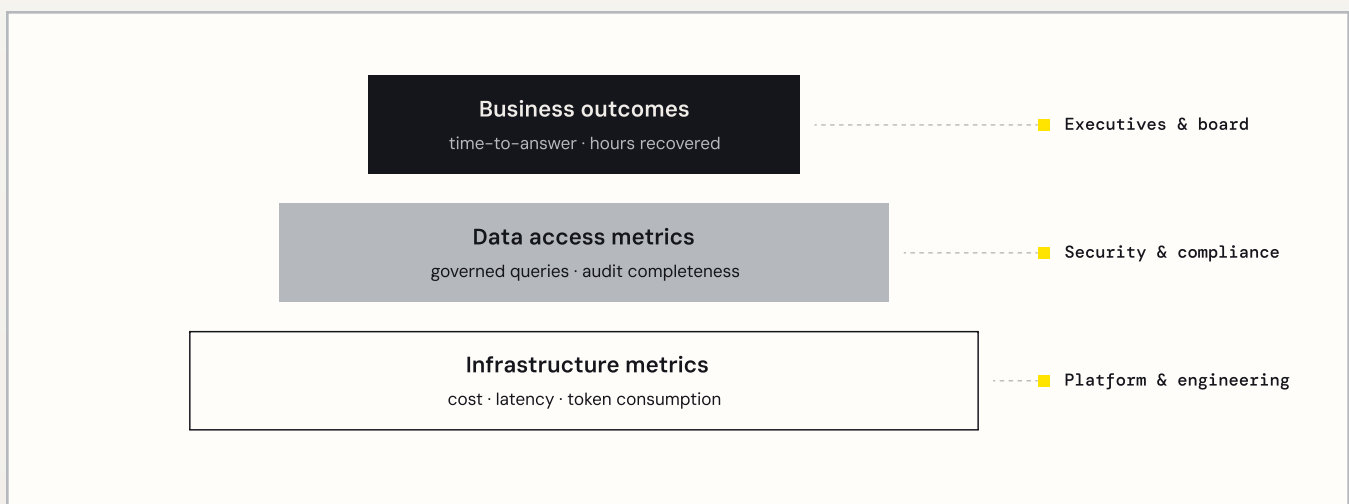


Figure 4. Three layers of measurement, each answering to a different audience.

Reading each layer

Infrastructure metrics describe the operating cost structure: request volume and latency by model, team, and use case; token consumption; cost per request; cache hit rate; provider availability. Cost per request by use case shows whether routing is working or defaulting expensive traffic to a frontier model. Fed into alerts, these metrics catch runaway cost and outages before users report them.

Data-access metrics answer governance questions: query volume per source, permission-denial rate, audit-log completeness, and schema-exposure rate. The audit log is the most underused asset here, treated as compliance overhead when it's also the diagnostic record that tells you whether a wrong answer was a model error, a data-access error, or a permission error.

Business-outcome metrics are the ones executives ask about, and they reward discipline set before deployment. McKinsey found that tracking well-defined gen-AI KPIs is the practice most correlated with bottom-line impact, yet fewer than one in five organizations do it. Define the categories early: time-to-answer measured before and after, analyst hours recovered, cost per resolved inquiry, revenue influenced where it can be attributed honestly. Futurum's 1H 2026 survey of 830 IT decision-makers found direct financial impact nearly doubled to 21.7% as the primary ROI metric.

The same metrics build the business case before deployment: the baseline cost of today's data-access approach, the projected reduction from centralizing it, and the token-efficiency gain from schema-aware access. In CData's own testing, schema-aware retrieval cut token usage by 97.6% on a governed multi-source query, a useful anchor when you model your own range.

The Tvhroughline

The gateway proves its worth the way any infrastructure does, by producing the evidence that it works. Organizations that build measurement in from day one can answer the board's question with data rather than intuition.

Read the full guide – cdata.com/blog/measure-ai-gateway-success

Glossary

Terms used in this series

AI gateway

The centralized control layer between AI applications and the models, tools, and data they use.

API gateway

Routes and secures HTTP requests between clients and services, endpoints, not intent.

MCP gateway

Routes agent intent, enforces tool policy, and governs the actions agents are allowed to take.

LLM gateway

The model-routing layer inside an AI gateway: model selection, cost optimization, and failover.

Context layer

The data-access layer that governs what an agent can see and returns it schema-aware and permission-filtered.

Schema-aware retrieval

Returning only the fields and rows a query needs instead of a full schema dump.

Token efficiency

The ratio of useful data returned to tokens consumed, a proxy for operational cost.

Audit-log completeness

The share of queries captured with full user identity and data attribution, effectively binary.

Self-service rate

Data questions resolved through the gateway without a human analyst or a custom query.

Time-to-answer

Elapsed time from a business question to a governed, data-backed answer, measured before and after.

Next Steps

Your enterprise data, finally AI-ready

CData Connect AI gives your assistants and agents live, governed access to hundreds of enterprise systems, so they reason over your actual business data, not just what they were trained on.

- **Request early access** • <https://www.cdata.com/ai/gateway-early-access/>
- **Learn about Connect AI** • [CData.com/ai](https://www.cdata.com/ai)

CData is the data layer that makes AI work in production, live connectivity and replication across hundreds of critical enterprise sources, semantic context, and built-in governance. Powering AI for Databricks, Microsoft, Google, Palantir, and 10,000+ customers worldwide.