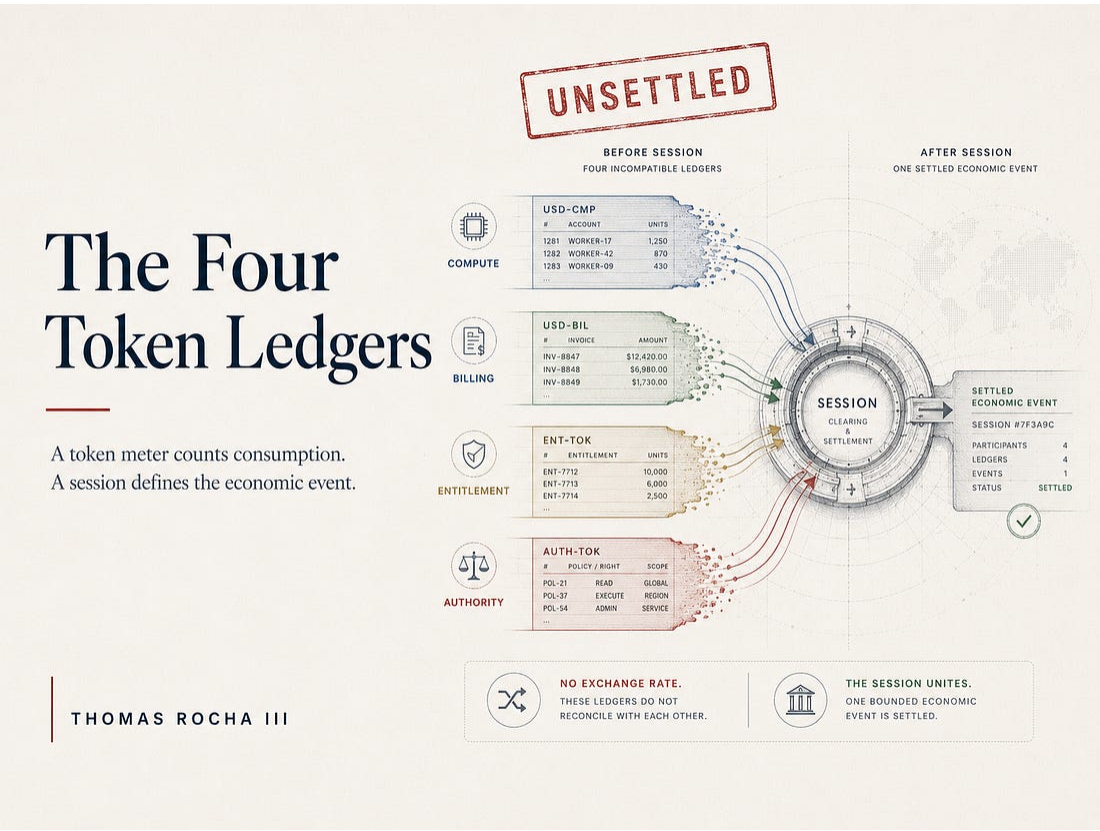
The Four Token Ledgers

A token meter counts consumption. A session defines the economic event.

THOMAS ROCHA III
MAY 22, 2026



In the spring of 2026, two of the world's largest engineering organizations admitted, in different ways, that they had lost control of their AI spending.

In April, Business Insider and The Information surfaced that Uber had already exhausted its 2026 Claude Code budget within the first months of the year. Uber’s leadership acknowledged the overrun. CEO Dara

Khosrowshahi said roughly 10% of code changes were produced by autonomous agents under human review. The cost surface was agent-driven, the tools were doing what they were marketed to do, and the budgeting assumptions did not survive contact with the consumption pattern.

In May, The Verge reported that Microsoft's Experiences and Devices division, covering Windows, Microsoft 365, Outlook, Teams, and Surface, is winding down most Claude Code usage by the end of June and steering developers toward GitHub Copilot CLI. Microsoft framed the move as platform convergence around a tool the company can shape directly with GitHub. The reporting around the decision also pointed to operating-expense pressure aligned with the June 30 fiscal-year close, and to the difficulty of forecasting token-based consumption at scale.

Two events. Two unrelated enterprises. Different industries, different stacks, different decisions in response. The same architectural cause underneath.

The press has framed this as a subscription-versus-utility problem. As a predictability problem. As a procurement problem. The rise of the chief financial officer in AI buying decisions. All of those framings are true. None of them is the story.

The story is that token billing has been treated as a cost model when it is actually a meter without a transaction. The visible cost is the token consumed. The invisible cost is everything that had to fire before consumption began and everything that continues firing after the budget has burned through. The Uber overrun and the Microsoft pullback are what it looks like when an enterprise tries to control the cost of something that has never been bound as an economic event.

A note before the cost direction debate

This essay is not about whether tokens are getting more expensive or less expensive. That question is malformed until something else is settled first.

A reasonable reader will arrive at the Uber and Microsoft reporting having absorbed several true statements from the broader coverage. Inference unit costs have fallen sharply over the past eighteen months. Sam Altman has put numbers on it. A16z has put numbers on it. NVIDIA's Blackwell deployments have put numbers on it. Gartner projects further reductions through 2030. The hardware is cheaper, the models are more efficient, and the per-call price is collapsing.

All of that is true. None of it tells you whether the cost of agentic work is going up, down, or sideways, because the unit *cost per token* is not yet attached to anything an enterprise can budget against.

A token is doing at least four different jobs simultaneously, and *cheaper* or *more expensive* lands differently on each. The model-unit token (the computational quantity the GPU processes) has been getting cheaper. The billing-unit token (the line item on the invoice) has been getting cheaper per unit and more numerous per task. The entitlement-unit token (the quota the plan allocates) is drawn down faster as agents do more per workflow. The authority-event token (the participation act admitted into a live interaction) is not currently priced by anyone, so its cost shows up later in different ledgers under different names.

When the critics of enterprise AI cost stories point out that inference is becoming radically cheaper, they are correct, and they are also describing only the first ledger. When enterprise CFOs report that AI budgets are being exhausted earlier than planned, they are correct and are primarily describing the second and third. When something goes wrong in an agentic workflow, and the compliance review reconstructs what happened, the cost shows up in the fourth, often months after the work was done.

The deflation argument and the budget-overrun argument are not in conflict. They are describing different ledgers at different time horizons. The reason they appear to contradict each other is that the field uses a single word to describe four economic events, and those events are moving in different directions at the same time.

Until the token is attached to a bounded economic unit (one that says: this work began here, was authorized here, drew entitlement here, ran compute here, was billed here, and ended here), there is no cost basis. There is only meter reading.

The deeper problem is that AI economics has no settlement standard. A token functions today the way a floating commodity quote functions in the absence of a reserve asset and a clearing unit. Everyone says *cost per token*, but no one has established what the token settles against. Compute is denominated in floating-point operations. Billing is denominated in dollars per million units. Entitlement is denominated in plan quotas. Authority is not denominated at all. The four denominations do not convert against each other, and there is no clearinghouse where they jointly resolve. The industry has been operating as if these were equivalent units. They are not. They are different currencies with no exchange rate.

This essay is about what binds the four ledgers into a single transaction and what supplies the missing clearinghouse.

Token is four different costs

The same overloaded word that hid four distinct architectural functions in the prior piece now hides four distinct cost centers here. Pulling the word apart is again the first move, because cost control begins with knowing what cost was actually incurred.

Token as model unit. The computational quantity the model consumes or produces. This is what the GPU absorbs. The cost is compute: electricity, cooling, depreciation on the silicon, capacity opportunity

cost. When the system retries a failed call, runs an extra summarization pass, regenerates output that was previously rejected, or maintains a longer context window than the task requires, it is spending compute resources. The compute cost is real and is paid by whoever owns the inference infrastructure, which may not be the same actor billed for consumption.

Token as billing unit. The line item on the invoice. This is what the vendor counts and what the buyer pays. The cost is metered spend: dollars per million tokens, multiplied by usage, attached to a billing account. When the meter advances, the billing unit accrues regardless of whether the underlying work was authorized, useful, or repeated. Microsoft's reported exposure was at least partly visible in this ledger. Uber's budget overrun was more directly visible there. The meter does not care whether the inference was retried by the agent after the first attempt failed validation, or whether two agents triggered the same query because they were unaware of each other.

Token as entitlement unit. The quota the plan allocates. This is what the carrier, the enterprise procurement team, or the family-plan owner cares about. The cost is allocation drawdown: a Max 20x plan with a notional usage allowance, an enterprise seat with a token cap, and a family-plan pool with shared consumption. When entitlement is drawn, it is drawn from a pool sized based on an assumption about who and what would consume it. If a junior developer runs a multi-agent workflow that consumes more entitlement in an afternoon than a senior architect uses in a month, the allocation has not changed. The drawdown has. The plan is still being honored. The plan was just not designed for the consumer who is honoring it.

Token as authority event. The participation act admitted into a live interaction. This is what nobody is currently counting. The cost is governance load: the policy reconciliation that runs before the inference, the audit binding that runs during, the compliance review that runs after, the forensic reconstruction that runs when something goes

wrong, the liability exposure that accumulates across all of it. When the authority chain remains open longer than it should, every later retry, tool call, model invocation, and delegated agent is operating under a permission that nobody has re-evaluated. The cost shows up later, in different ledgers, and is usually never attributed back to the original lack of session termination.

Four ledgers. One word. The vendor counts the billing ledger. The model provider counts the compute ledger. The procurement team counts the entitlement ledger. Nobody is counting the authority ledger, because nobody has a place to count it.

Cost accounting versus cost control

Microsoft's response to its forecasting and platform-control problem was to switch to a seat-based internal default. The Uber response, judging by public reporting, will likely rhyme with: cap the spend, narrow the surface, restrict the agents, or find a tool whose unit economics are easier to predict. CloudZero, Apptio, Anodot, and the broader category of FinOps vendors are positioning themselves into this opening with forecasting and alerting tools for token spend. CFOs are now asking the questions CTOs were asking last year.

All of this activity is cost accounting. None of it is cost control.

The distinction matters, and it is the load-bearing economic argument of this essay.

Cost accounting tells you where the money went. It produces invoices, dashboards, forecasts, alerts, reports. It tells the enterprise that the AI budget was exhausted in four months instead of twelve. It does not change what happens next. It does not stop the agent from retrying a failed call. It does not deny the entitlement drawdown that is about to push the team over its cap. It does not terminate the session that is about to spawn a multi-agent cascade. It produces the receipt.

Cost control requires three things that cost accounting does not provide.

The first is classification. You cannot optimize what you cannot identify. A token meter that aggregates all four ledger types into a single number shows the dollar amount. It does not tell you whether the spend was driven by legitimate compute, billing leakage, entitlement abuse, or authority decay. The four cost centers respond to different interventions. Treating them as one number means every intervention is a blunt instrument. The enterprise either cuts everything or accepts the spend it does not understand.

The second is influence over the cause. You cannot control what you cannot influence before it happens. A forecast that the budget will exceed its cap by Tuesday is information. A control surface that denies the next retry, downgrades the next model selection, expires the next authority grant, or terminates the next session is influence. The first is observation. The second is governance. The Uber and Microsoft events are the public form of enterprises that have the first and lack the second.

The third is a unit small enough to act on. Cost accounting reports against the month, the quarter, the fiscal year. Cost control acts against the next inference. The unit that operates at inference-time is not the budget. It is the session, and the session is what the architecture currently does not provide.

A token meter is not a cost-control system. It is a receipt.

The economic premise

The four ledgers only become governable if a session binds them together.

Without a session, the four ledgers describe one live interaction from four incompatible positions. The model counts compute. The vendor

counts billing. The plan counts entitlement. The compliance system tries to reconstruct authority after the fact. Each is locally accurate. None is jointly meaningful.

A session is what closes the gap. The session is the bounded unit that can say: this inference was admitted here, billed here, allocated here, authorized here, and terminated here. With that boundary in place, the four ledgers describe the same economic event from four complementary positions, and the system can act on the event before, during, and after its execution. Without the boundary, the ledgers drift, and the enterprise reconciles after the meter has run.

This is the economic case for session governance, and it is the one the press is not yet making. The Microsoft and Uber events are being read as pricing-model failures. They are pricing-model failures, but only because the pricing model is operating in the absence of a bounded transaction. Token billing without a session is invoicing without a contract. You can count what crossed the meter. You cannot define what was bought, by whom, under what authority, for what purpose.

When the session is the governor, the four ledgers cohere. Compute spending is bounded by the session's compute scope. Billing is bound by the session's accounting scope. Entitlement is drawn against the session's authorization scope. Authority is held by the session's policy scope. When the session terminates, all four close at once. The compute stops, the billing closes, the entitlement is restored, and the authority chain dies. That is the economic shape of a governed transaction, and it is the shape that current architectures do not produce.

Agents are why this is breaking now

The Uber and Microsoft events did not happen because token prices moved in either direction. They occurred because the priced unit was not the unit consumed.

A model invocation is bounded by a prompt and a response. The cost of a model invocation is roughly predictable: the input length, the output length, the model selected, the rate card. Enterprises have been pricing model use for two years. The unit economics are well understood at the per-call level.

An agent invocation is bounded by a task, and a task can require an unbounded number of model calls, tool calls, retries, memory reads, memory writes, and delegations to other agents before the task is considered complete. The cost of an agent invocation is not roughly predictable. It is a sequence of cost-bearing acts, each of which appears locally rational, with no bound on the sequence's total cost beyond whatever the agent eventually decides is done.

That is the cost surface that broke Uber's budget and pressured Microsoft's. Not the price per token. The unboundedness of task-level consumption that token pricing was never designed to govern. Khosrowshahi's reported figure of roughly ten percent of code changes produced by autonomous agents is the visible part of an agentic deployment whose invisible part was a year of budget consumed in a fraction of a year.

Multi-agent systems compound the problem. Agent A delegates to Agent B, which calls a third-party model, which routes to a tool, which writes to a memory that Agent C later acts on. Each step appears locally valid. None of the steps was authorized as part of the original task in any way the cost ledgers can recognize. The compute fires, the meter advances, the entitlement drains, and the authority chain extends across actors who never explicitly consented to be on the chain. The bill arrives at the end of the month, attributed to no one in particular, with no way to identify which step in which cascade should have been refused.

A model spends tokens to answer. An agent spends authority to act, and the authority spends across all four ledgers at once. The carriers

selling AI tokens, the cloud providers selling API access, and the enterprises buying both are all about to discover that the unit they have been counting is not the unit they need to govern.

Why FinOps does not close the gap

The vendor category that has emerged in response to enterprise AI cost surprise is being called FinOps, and the named players (CloudZero, Apptio, Anodot, and a growing list of newer entrants) are building forecasting, alerting, and attribution tools. The tools are real. The work is competent. The market need is being filled.

The architectural problem is that forecasting and attribution are downstream functions. They observe what has already happened or is about to happen. They cannot refuse a retry. They cannot revoke an authority grant. They cannot terminate a session. They cannot deny a tool call that the agent has already decided to make. They tell the CFO how fast the budget is bleeding. They do not bound what the system is allowed to spend, against what authority, for what purpose.

The deeper problem is that the FinOps category is trying to build accounting infrastructure for an economy that has not yet defined its unit of account. A FinOps tool can sum the meter readings across compute, billing, and entitlement. It cannot convert them to a single denomination because there is no settlement standard against which the conversions would resolve. The dashboards aggregate four different currencies into a column labeled in dollars and report the total. The total is technically accurate. It is also not telling the enterprise what was bought.

The FinOps tools sit on top of the four ledgers and try to make sense of them as observed phenomena. The architectural answer sits underneath the four ledgers and binds them into a single transaction the system can act on. Those are different layers of the stack, and the second is not what is currently being built outside of patent disclosures and a small set of research efforts.

This is not a criticism of the FinOps vendors. They are building what their customers will pay for in the current market. What their customers actually need is one layer down, and the market has not yet recognized that the layer is missing. The Microsoft and Uber events are the visible form of the recognition arriving.

The session is the economic container

The closing argument is structural.

Cost control requires three properties that current architectures do not jointly provide: classification of which ledger the cost belongs to, influence over the cause before it incurs additional cost, and a unit small enough to act on at inference time. The session is the architectural object that supplies all three. Compute is bounded by the session's compute scope. Billing is bound by the session's accounting scope. Entitlement is bounded by the session's authorization scope. Authority is held by the session's policy scope. The four ledgers cohere because the session is what binds them together.

Without that binding, the enterprise has receipts. With it, the enterprise has cost control.

The deeper claim is the one this essay opened with and is closing on. *Cost per token* is not a well-formed measurement. It is a meter reading that becomes a cost basis only when the token is attached to a bounded economic event. Unit-cost deflation at the model ledger does not establish cost deflation at the session ledger. Unit-cost inflation at the entitlement ledger does not establish inflation at the compute ledger. The four ledgers can move in different directions at the same time, and the field has been arguing about which direction the cost is moving while operating under a unit that does not specify which cost is being discussed.

Until the session defines the economic event, the cost-direction debate cannot be settled. It cannot even be coherently framed.

The token is a metered unit of movement. The session is the clearinghouse.

Without the clearinghouse, there is no final settlement. There are only accumulated meter readings in four currencies that do not convert against each other.

A token meter counts consumption.

A session defines the economic event.

The model does not govern the session.

The session governs the model, and the session is what makes the cost of the model expressible in the first place.

That is the economic corollary to the architectural argument, and it is the part that the procurement teams, the FinOps vendors, and the CFOs reading the Uber and Microsoft reporting are about to need.