emma

INSIGHT BRIEF

CLOUD COST · SAAS

Where Does the Cloud Money Go?

Turning cloud spend from a black box into a number you can actually defend

FOR CIO / CTO · VPS OF ENGINEERING · HEAD OF CLOUD/PLATFORM

You probably already suspect you're overpaying for cloud. You're likely right. But that's not your real problem.

Public cloud spending hit roughly $723 billion in 2025 (Gartner) and, at current growth rates, is on track to approach $880 billion in 2026. For many software companies, cloud is now the largest cost line after payroll. And a stubborn share of it is wasted: Flexera's 2026 State of the Cloud puts estimated waste at about **29%**, while **85%** of organizations now call managing cloud spend their single biggest cloud challenge, and budgets overshoot reported by about 17% of organizations this year.

The deeper issue isn't the size of the bill, though — it's that almost nobody can explain it:

Most teams can't explain their own cloud bill

Share of organizations — and the waste that follows when the bill is a black box

Sources: CloudZero, 2025 State of Cloud Cost Intelligence (visibly & unit-cost tracking); Flexera, 2026 State of the Cloud (waste)

Read that as a diagnosis. Most teams don't have a spending problem — they have a visibility problem. Overspend is the symptom; the black box is the disease. And you can't fix what you can't see.

> **From the field**
>
> Across 400+ recent cloud buying conversations we analyzed, cost came up in roughly **three out of four.** But the most common pain wasn't "it's too expensive." It was a version of: "we only find out what we spent when the bill arrives." The anxiety is rarely about the number itself — it's about not being able to see or steer it.

Why your cloud bill is a black box

Three structural forces make modern cloud spend genuinely hard to read. Naming them is the first step to fixing them.

Tagging breaks down at exactly the scale you need it

The default plan is “we’ll tag everything.” In practice, tag-dependent approaches leave 15–30% of spend unallocated in most environments.

Shared services, multi-tenant platforms, and Kubernetes clusters can’t be cleanly tagged at all — one cluster serves many teams, products, and customers at once. The moment a resource is untagged or inconsistently tagged, its cost vanishes into an “unallocated” pool nobody owns.

The estate is fragmented — often by accident

Hybrid is now the norm (about 73% of organizations), and multi-cloud keeps rising — frequently driven by mergers and team silos rather than deliberate strategy (Flexera).

Layer on SaaS tools, data platforms, and AI APIs, each with its own bill, format, and billing cadence, and your true “cloud” cost is scattered across consoles that don’t reconcile with one another. No single view means no single answer.

The bill arrives weeks after the decision that caused it

A monthly invoice lands long after the deploy, the misconfigured autoscaler, or the dev environment left running over the weekend. AI and Kubernetes workloads can spike in minutes.

By the time the cost shows up on a statement, the money is already spent and the context is cold. Detecting spend at month-end is, structurally, always too late.

The one-sentence version:

your bill tells you what you spent — not who spent it, what drove it, or whether it was worth it.

The reframe: from “what did it cost?” to “was it worth it?”

The teams that get cloud cost under control stop asking “how do we cut the bill?” and start asking “what does a unit of our business cost to run?” That means unit economics: cost per customer, per tenant, per feature, per transaction, per AI inference. A raw bill can’t answer whether spend was justified. A cost-per-unit tied to revenue can.

Why this isn’t academic — a concrete example: a feature that costs **$50,000/month** but drives $2M in revenue is not a cost-cutting target.

A feature that costs **$40,000/month** and is barely used is — even though its absolute spend is lower.

Total spend is the wrong lens. Cost relative to value is the right one, and it routinely flips which “expensive” things you should actually leave alone.

The cheaper feature is the one to cut

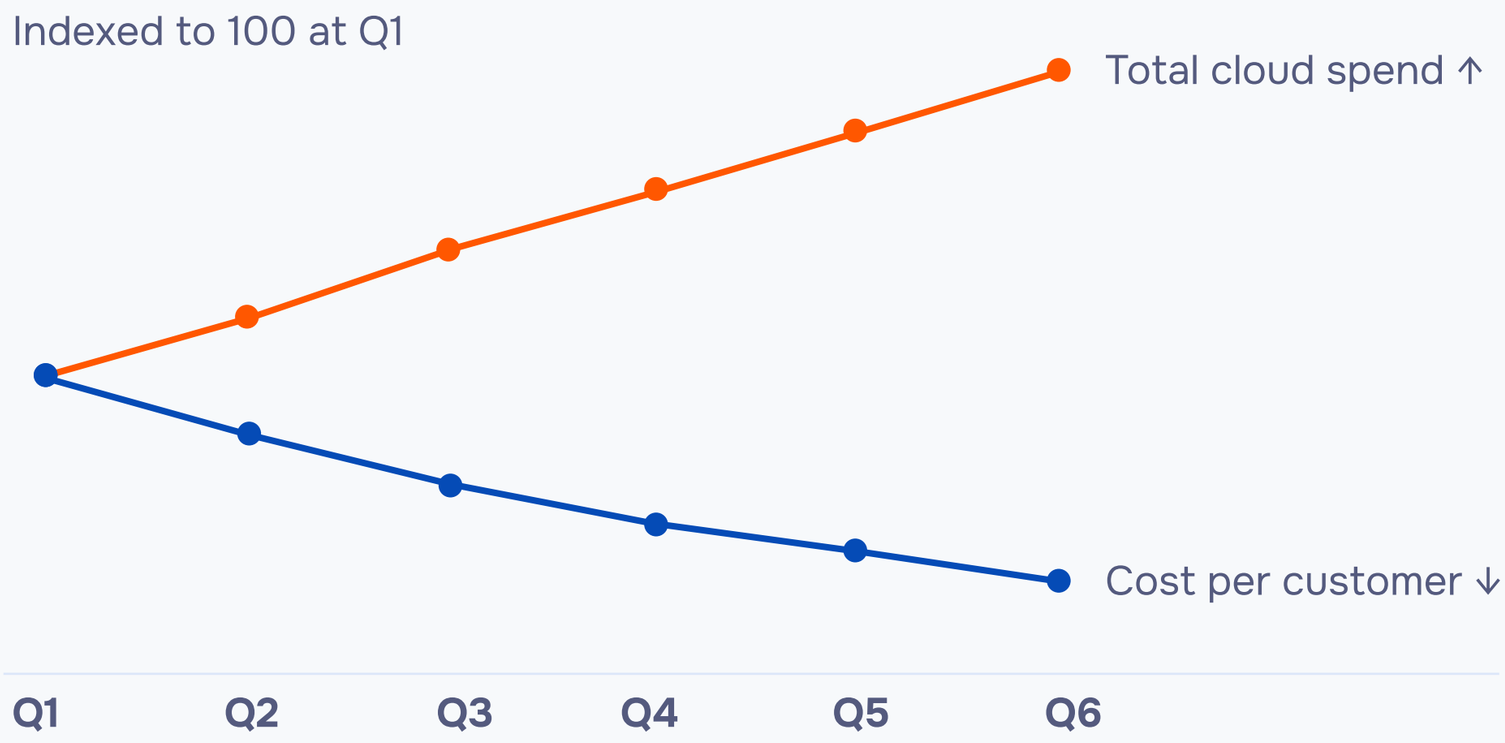
Total spend is the wrong lens — weigh each cost against the value it creates

Illustrative examle. The lower cost feture can be the one destroying margin

For a SaaS business this lands directly on the P&L: cloud is cost of goods sold. Your cost to serve a customer shapes gross margin, pricing, which segments are worth chasing, and the efficiency story investors underwrite. A useful executive metric is your Cloud Efficiency Rate — cloud cost as a share of revenue. The counterintuitive truth it reveals: your cloud bill can grow while your efficiency improves, as long as unit costs fall as you scale. That single reframe moves the board conversation from “stop spending” to “improve the ratio.”

Your cloud bill can grow while your efficiency improves

When unit cost fails as you scale, a rising bill is healthy growth — manage the ratio, not the total



Illustrative. Cloud Efficiency Rate = cloud cost as a share of revenue.

Where are you on the ladder?

Cloud cost maturity is a climb from a single number to a defensible ratio. The uncomfortable finding across the industry: most organizations are stuck on the bottom rungs — they've bought dashboards and achieved some visibility, but they don't act on it. Visibility nobody acts on is just decoration.

STAGE	WHAT IT LOOKS LIKE	THE QUESTION YOU CAN FINALLY ANSWER
Blind	One big monthly number. Surprises arrive by invoice.	"What did we spend?"
See	Spend broken down by service and account in native consoles.	"Which service cost what?"
Attribute	Spend mapped to team, product and customer — including shared and untagged resources.	"Who and what drove it?"
Own	Every cost has an owner; showback in engineers' workflow; anomalies caught in real time.	"Who's accountable — and did we catch it in time?"
Operate	Unit economics tied to revenue; cost is a guardrail in the SDLC, not a cleanup job.	"Was it worth it?"

Most SaaS teams plateau at **See**. The leap that changes everything is **See → Attribute → Own** — the point where cost stops being finance's monthly mystery and becomes an engineering signal.

A 10-question diagnostic for your next staff meeting

Read these aloud with your engineering and finance leads. If you can't confidently answer seven of them, the gap is visibility, not budget.

01 WHAT DOES IT COST US TO SERVE ONE CUSTOMER — AND IS THAT RISING FASTER THAN REVENUE PER CUSTOMER?

02 WHICH CUSTOMERS OR TENANTS ARE UNPROFITABLE TO SERVE RIGHT NOW?

03 WHAT DID LAST NIGHT'S DEPLOY COST US?

04 WHAT SHARE OF OUR BILL IS UNALLOCATED — NOT TIED TO ANY TEAM, PRODUCT, OR CUSTOMER?

05 WHAT ARE OUR TOP FIVE COST DRIVERS THIS MONTH, AND WHO OWNS EACH ONE?

06 WHAT DOES OUR MOST-LOVED FEATURE COST TO RUN — AND WHAT DOES OUR LEAST-USED ONE COST?

07 WHAT IS OUR CLOUD COST AS A PERCENTAGE OF REVENUE, AND WHICH WAY IS IT TRENDING?

08 WHAT DID OUR AI / ML FEATURES COST LAST MONTH, BROKEN DOWN BY MODEL OR FEATURE?

09 HOW LONG AFTER A SPIKE DO WE FIND OUT — MINUTES, DAYS, OR AT MONTH-END?

10 IF WE HAD TO CUT 20% TOMORROW, DO WE KNOW EXACTLY WHERE — WITHOUT RISKING RELIABILITY?

The playbook: six moves that actually move the number

Skip the table stakes (“right-size your instances,” “buy a dashboard”). These are the moves that separate teams who control cost from teams who just watch it.

MOVE 1

Track the distribution of cost-per-customer, not the average

“Measure cost per customer” is good advice and the blended average is nearly useless — it hides the actual problem. What changes decisions is the **distribution**: rank every customer or tenant by cost-to-serve and lay it against what they pay.

You will almost always find two things — a few **“underwater whales”** (big logos on flat or heavily discounted contracts quietly burning gross margin) and a free/trial tier that costs more to serve than your paying base. That one ranked list reframes pricing, renewal terms, and which segments to chase far more than any total-spend chart. Pick one unit metric to start — but make it the one tied to a P&L decision you’re about to make: a renewal, a new tier, a pricing change.

MOVE 2

Meter AI per feature and per request — it breaks the old playbook

This is the 2026 trap. AI spend is **supplementary** (it adds cost, it doesn’t replace), it’s usage-metered, it can spike in minutes, and it usually hides inside compute and managed-service line items instead of a clean “AI” line — so it’s invisible exactly when it’s growing fastest.

The non-obvious risk for SaaS: AI features are often **priced flat but cost variable**, so your heaviest users can carry negative marginal margin. Meter inference cost per feature and per request, then put it next to whether that feature is monetized and how power users consume it. If cost-per-active-user on an AI feature is climbing toward its price, you have a pricing problem, not a cost problem.

MOVE 3

Give each team a cost number they own like an SLO — don’t ask them to “care”

“Use reserved instances and savings plans” is table stakes (and fewer than half of teams even do it). The non-obvious part is how much to commit. A growth-stage SaaS that commits to 100% of current usage will over-buy right before it re-architects, moves to ARM/Graviton, or shifts regions for data-residency — and gets locked into the wrong thing.

The discipline: commit only to your **usage floor** (the trough you never drop below), cover ~70–80% rather than 100%, and layer flexible savings plans over reserved instances so a re-platforming doesn’t strand the commitment. Commitments are a bet on your baseload — size the bet to what you’re actually certain of.

MOVE 4

Commit to your trough, not your average

“Use reserved instances and savings plans” is table stakes (and fewer than half of teams even do it). The non-obvious part is how much to commit. A growth-stage SaaS that commits to 100% of current usage will over-buy right before it re-architects, moves to ARM/Graviton, or shifts regions for data-residency — and gets locked into the wrong thing.

The discipline: commit only to your **usage floor** (the trough you never drop below), cover ~70–80% rather than 100%, and layer flexible savings plans over reserved instances so a re-platforming doesn’t strand the commitment. Commitments are a bet on your baseload — size the bet to what you’re actually certain of.

MOVE 5

Put a unit-cost target in the design review, before the code exists

“Treat cost as a non-functional requirement” is true but toothless as usually stated.
Make it concrete: add one line to your architecture/design-review template, next to latency and availability — a new service must **declare its target unit cost (per request, per tenant, per inference) before it ships.**
Then forecast that unit cost **at 10×**: if cost-per-customer rises as you scale instead of falling, you’ve found an architecture problem while it’s still cheap to fix — not after it’s baked into COGS.

That is the difference between governance as prevention and governance as monthly cleanup, and it’s why governance is the fastest-rising priority in the field: cleanup doesn’t scale.

Seven anti-patterns to avoid

Optimizing before you can attribute

you right-size the visible half of the bill and miss the shared-infra half that’s actually growing.

Trusting the blended cost-per-customer average

it hides the underwater whales and the free tier eating your margin. Always read the distribution.

Treating AI like just another cloud line

it’s metered, supplementary, and buried inside compute — budget and meter it on its own or it erodes margin silently.

Committing to 100% of usage

you lock into the wrong instances right before a re-architecture. Commit to the trough, not the peak.

The showback dashboard with no owner and no budget

visibility without a number someone is accountable for changes no behavior.

Across-the-board percentage cuts

they starve profitable workloads and spare the wasteful ones. Cut by cost-to-value, not by cost.

Comparing multi-cloud bills without normalizing them

AWS, Azure and GCP line items aren’t comparable until you map them to a common schema (the FOCUS standard). Until then, “are we cheaper on X?” is unanswerable.

Who needs to be in the room

Cloud cost is not a finance project handed down for cleanup. The spend is created by architecture decisions, it shows up as COGS, and it’s priced by product. Solving it is a three-legged stool:

Engineering & Platform leaders

own the architecture choices that create cost — and need cost signal in their workflow to act on it.

The technical executive (CTO / CIO)

owns the trade-off between velocity, reliability, and margin — and owns the number to the board.

Finance & Product

own gross margin, COGS, and pricing — which cost-per-customer directly informs.

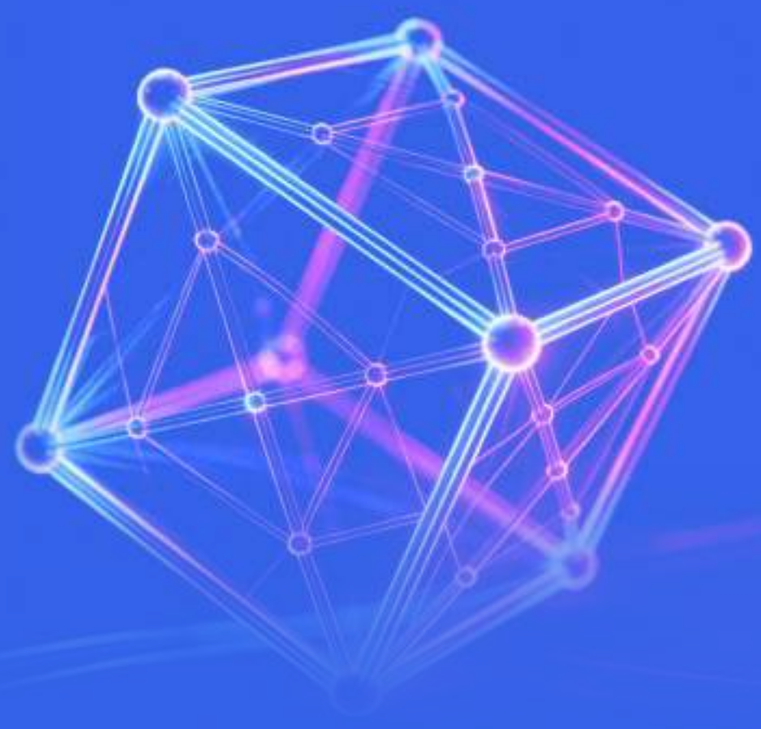
The thing that aligns them is a single shared number everyone trusts: your **cost per customer and Cloud Efficiency Rate**, trended over time. When engineering, finance, and product argue from the same number, the conversation shifts from blame to trade-offs — which is where good decisions get made.

Start Monday: a 30 / 60 / 90 plan

01 **First 30 days** — see clearly. Consolidate every cloud, SaaS and AI bill into one view. Find your top five cost drivers and your unallocated percentage. Choose one unit metric tied to a decision you’re about to make.

02 **By 60 days** — attribute and own. Allocate spend to teams, products and customers, including shared clusters by consumption. Stand up showback and one real-time anomaly alert. Rank customers by cost-to-serve; kill idle headroom; commit to the trough.

03 **By 90 days** — operate. Publish cost per customer and your efficiency ratio to engineering, finance and product. Add a unit-cost target to your design-review template. Set a quarterly target tied to the ratio, not the absolute bill.



The one number to put on the wall

Stop staring at the total. Put your unit metric — cost per customer — and its trend against revenue per customer where the team can see it. That single ratio turns “are we overpaying?” from a recurring anxiety into a question you can answer, trend, and defend. It’s also the move that turns cloud cost from a finance fire drill into an engineering habit — which is the only version of cost control that lasts.

This brief was put together by the team at emma. If a second set of eyes on your own estate would help — a quick read on your top cost drivers, your unallocated spend, and your cost-to-serve across clouds — that’s a conversation we’re glad to have. No deck required.

Sources

- Gartner, worldwide public cloud end-user spending (2025 actual / 2026 forecast), via CloudZero.
- Flexera, 2026 State of the Cloud Report (waste ~29%, 85% cite managing spend as top challenge, budgets +17%, hybrid 73%, commitment-discount adoption under half).
- FinOps Foundation, State of FinOps 2025–2026 (waste reduction the #1 priority; “empower engineers” the prior #1; full allocation & forecasting; “getting to unit economics” rising; governance the rising future priority; centralized-enablement model; FinOps reports into CTO/CIO ~78%).
- CloudZero, 2025 State of Cloud Cost Intelligence and cloud cost guides (only ~30% can identify where budget goes; 43% track unit cost; 15–30% unallocated under tag-dependent approaches; unit-economics reframing; the $50k vs $40k feature example; ~20–30% typical reduction).
- FinOps Foundation FOCUS (Open Cost & Usage Specification) for cross-provider bill normalization.

emma proprietary analysis of 400+ recorded cloud buying conversations (theme frequency; recurring customer language).