// WHITE PAPER · PUBLISHED 16 SEPTEMBER 2026

AI Spend: Three Budgets Pretending to Be One

What part of your AI bill is a project, and what part is a utility

Part 8 of 10 in *The Money Language Series*. Each piece stands on its own. The others: Part 1 · Part 2 · Part 3 · Part 4 · Part 5 · Part 6 · Part 7 · Part 9 · Part 10.

Jason Newell · MAX Research Collective · Report WPR-BIZ-009 · Published 16 September 2026
Licensed under the Creative Commons Attribution 4.0 International License (CC BY 4.0), https://creativecommons.org/licenses/by/4.0/
Original: https://www.maxresearchcollective.com/articles/ai-spend.html
Cite as: Newell, J. (2026). AI Spend: Three Budgets Pretending to Be One. MAX Research Collective, WPR-BIZ-009. https://www.maxresearchcollective.com/articles/ai-spend.html

Ask a technology leader for their AI budget and you usually get one number. Ask what is inside it and you get four categories, two of which nobody is forecasting and one of which is quietly eating gross margin.

The most useful thing I have done on this subject is refuse to let AI be one line. It is three lines. They behave differently from each other, and the accounting, the forecasting and the governance are distinct for each.

The three budgets

	TOOLING	BUILDING	SERVING
What it is	Seats, subscriptions, developer assistants	Implementation, integration, re-architecture	Inference on behalf of customers, and GPU capacity
Cost behaviour	Per seat, semi-fixed	A one-time project	Variable, scaling with usage
Where it lands	Operating expense, research or administration	Expensed or capitalized	Cost of revenue
Book treatment	Expensed as incurred	Potentially capitalizable under ASC 350-40	Expensed as consumed
Tax treatment	Deductible	Domestic work deductible now under Section 174A	Deductible, and owned GPUs take bonus depreciation
Forecasting	Easy, it tracks headcount	Medium, it is a scoped project	Hard, it tracks behaviour
Who owns it	Engineering leadership	The project sponsor	Product and finance together
Touches gross margin?	No	No	Yes, directly

One line of that table carries the argument. Serving is the budget that moves gross margin. It is usually the one with no owner.

Budget 1, tooling

Coding assistants, chat subscriptions and meeting tools sit here. The spend is per seat, predictable, easy to approve and easy to sprawl, and the mistakes are small individually.

Seat sprawl is the first one. Licences get assigned at onboarding and never reclaimed, which is the same problem as any subscription software and takes the same fix: quarterly reclamation against actual sign-in data.

Overlapping tools are the second. Three assistants do one job because three teams each chose one, and consolidation buys back real money plus more context-switching time than the money is worth.

The third is API keys bought on personal cards and reimbursed through expenses, which is the shadow IT of this cycle: unbudgeted, ungoverned, and frequently the route by which company data reaches a service nobody approved.

None of this is strategically interesting. It is hygiene, the numbers are small against the third budget, and the risk is that it absorbs the governance attention the third budget needs.

Budget 2, building

This one deserves to be run as a capital project and usually gets run as a science experiment.

COMPONENT	SHARE OF EFFORT	CAPITALIZABLE?
Data pipeline and preparation	Large	Often, once the approach is settled
Retrieval and indexing infrastructure	Medium	Often
Evaluation harness and test sets	Medium	Often
Agent orchestration and tool integration	Medium	Often
Guardrails, safety and policy enforcement	Medium	Often
Model selection experiments and spikes	Small	No, this is where the uncertainty lives
Prompt iteration after launch	Ongoing	No, this is maintenance

ASU 2025-06 removed the project-stage model and replaced it with an authorization-plus-probability test carrying an explicit brake for significant development uncertainty, and that line happens to fall in a place that suits AI work. The exploratory stretch — where you are still finding out whether a model can reach the accuracy the product needs — is genuinely uncertain, so it gets expensed. Once feasibility is proven, funding is committed and the remaining question is execution, the build can be capitalized from that date forward.

That maps onto how a good AI project already runs: a bounded evaluation phase ending in a kill decision, then a build. A kill decision buys two things at once: project governance, and a capitalization story. An organization without one is missing both, and the gap has a single cause.

The artifact that carries both is one page. "We evaluated three approaches, approach B reached 91 percent on our evaluation set, we are committing $1.4 million to build it, the sponsor is X and the target is the third quarter." That document is the

project charter and the capitalization evidence at the same time, and writing it takes an afternoon.

Budget 3, serving, and the gross margin problem

Here is the part that changed the economics of software and that most organizations are still catching up with.

In classic software, the marginal cost of serving one more user of one more feature rounds to zero. That rounding is why the category carries gross margins of 75 to 85 percent; the whole valuation model rests on it.

Inference refuses to round. Every invocation costs money, and the cost scales three ways at once: with usage, with context length, and with how capable a model somebody chose. That produces a failure mode the industry has not had before: a feature customers love can dilute gross margin at the exact moment it succeeds.

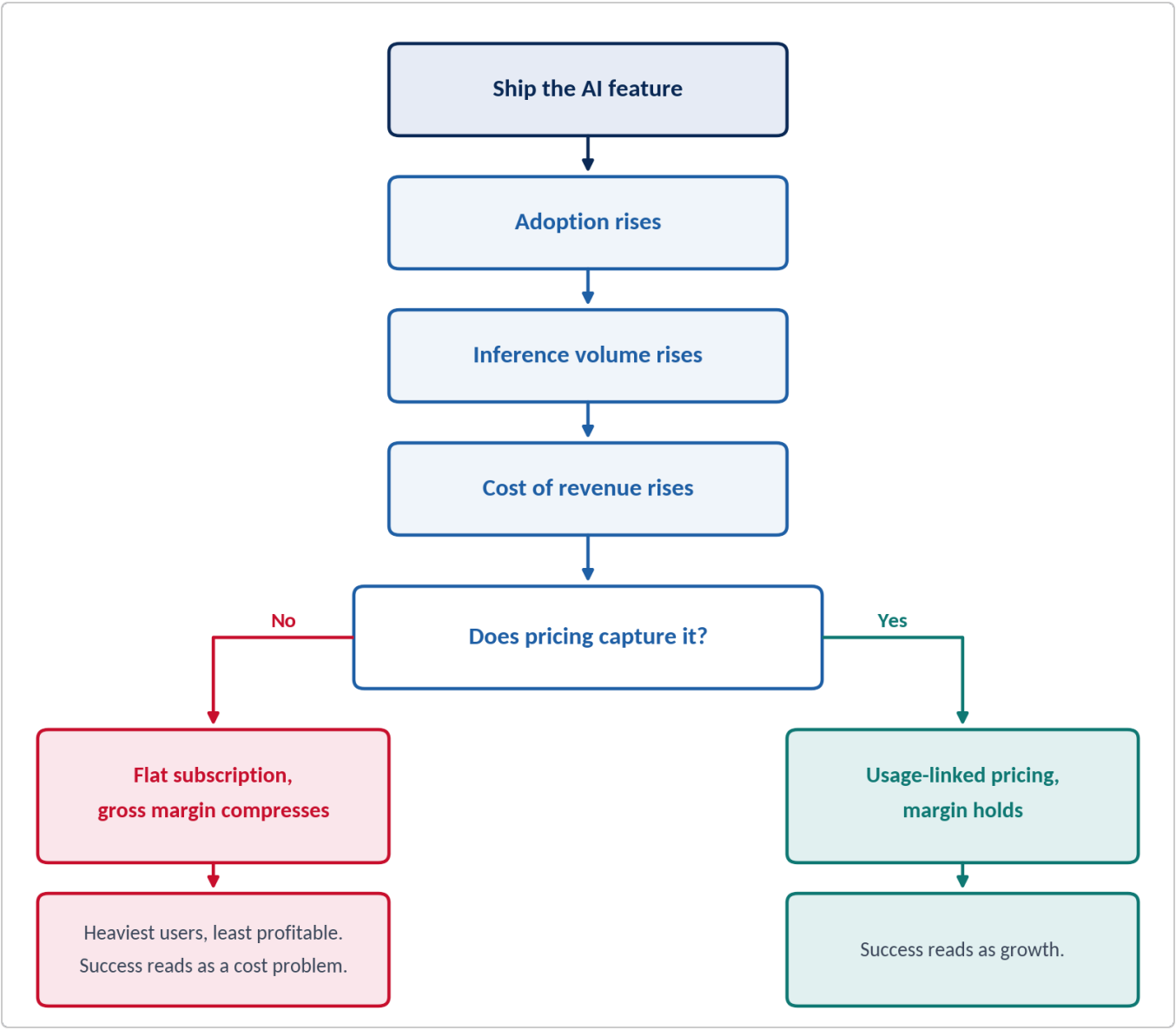


Figure 1. One chain leads to two endings. Everything above the gate is engineering and everything below it is pricing, which is why a launch that clears every technical bar can still arrive at the left-hand ending.

The questions that follow are pricing questions, and engineering has to supply the numbers behind them. What does the median customer cost to serve, per month, in inference? What does the customer at the ninety-fifth percentile cost? At what level of usage does a customer stop being profitable? Does this feature belong in the base tier or in a paid add-on?

Most companies cannot answer the first two, because inference spend is not attributed per customer. That instrumentation has become mandatory. It is the same

discipline as tagging cloud resources so cost of revenue can be split from research spend, which Part 2 covers, applied to a cost that varies far more.

LEVER	EFFECT	THE TRADE
Model tiering, small models for simple tasks	Large	Routing logic, and an evaluation per tier
Caching common responses	Large where it applies	Staleness and invalidation
Prompt and context compression	Moderate	Quality risk if trimmed too far
Batching non-interactive work	Moderate	Latency, so it suits asynchronous cases only
Fine-tuning a smaller model	Large at volume	Upfront cost and ongoing maintenance
Usage-linked pricing	Solves it commercially	Sales friction, and harder forecasting for the customer
Rate limits and quotas by tier	Direct	A cost paid in customer experience

The first lever returns the most and gets used the least. Routing a simple request to a smaller model is unglamorous engineering with a direct and measurable effect on gross margin. It competes for attention against work that demos better, and it loses.

Owning GPUs

For most companies the answer is rent. Sustained utilization is the number that changes it.

SUSTAINED UTILIZATION	USUALLY BEST
Under 30 percent	On-demand cloud
30 to 60 percent	Reserved or committed cloud capacity
Over 60 to 70 percent, stable, with 18 months of visibility	Owning, in colocation

Cross that line and you are back in Part 4 territory: useful-life estimates, bonus depreciation, power and cooling as a fixed cost per rack unit, and the real risk that a chip's economics degrade before the chip does. The tax side is favourable right now, with 100 percent bonus depreciation permanent for qualified property acquired and placed in service after 19 January 2025, so a GPU purchase can be deducted in full in year one for tax while depreciating over five years for book.

One thing stays off the table: buying accelerators against a demand forecast built on a feature that launched last quarter. Utilization has to be demonstrated.

Why AI made cloud forecasting worse

Flexera's 2026 State of the Cloud Report put self-estimated wasted infrastructure and platform spend at 29 percent — the first increase in five years — and attributed the reversal specifically to AI workloads making rightsizing and forecasting harder. In the same survey 81 percent reported using generative AI, up from 72 percent the year before and 47 percent the year before that, and 53 percent named security and compliance as their top challenge for cloud AI work.

That is a signal worth sitting with. Cloud cost discipline had improved for five straight years, and one workload category reversed it. If your FinOps numbers got worse this year, the practice is probably fine. The problem got harder faster than the practice got better.

What I actually do

1. Split the budget into three named lines with three different owners. Nothing else in this article works until that exists.
2. Instrument inference cost per customer and per feature before the second AI feature ships, because retrofitting attribution across a live estate is painful and nobody volunteers for it.
3. Put an evaluation gate with a real kill decision in front of every AI build. Good governance and clean capitalization turn out to be the same artifact.
4. Write the authorization document: date, sponsor, scope, amount, expected completion. That page is the capitalization trigger under the new standard.
5. Review model routing every quarter. The cheapest adequate model changes every few months and almost nobody goes back to check.
6. Treat gross margin impact as a launch criterion, alongside latency and error rate. A feature that moves margin past an agreed threshold brings pricing into the launch.
7. Buy GPUs on demonstrated utilization (demonstrated meaning measured, over months, on the workload that will actually run there). A forecast is a forecast, whatever the slide calls it.
8. Assume the forecast will be wrong, and commit to the confident floor, which is Part 7's rule applied to the least predictable workload anyone runs.

Newell, J. (2026). MAX Research Collective, WPR-BIZ-009. CC BY 4.0

Glossary

TERM	DEFINITION
Agentic workload	A system where models take multi-step actions using tools
Context window	How much text a model considers at once, where more costs more
Cost of revenue	What it costs to deliver what customers bought
Evaluation set	A test suite measuring model output against known-good answers
Fine-tuning	Adapting a smaller model to your task to cut serving cost
Gross margin	Revenue minus cost of revenue, as a percentage
Guardrails	Controls on what a model may accept or produce
Inference	Running a trained model to produce output, which is the per-use cost
Model routing, or tiering	Sending simpler requests to cheaper models
Retrieval	Fetching relevant data to give a model context at query time
Significant development uncertainty	The ASU 2025-06 concept that blocks capitalization while feasibility is still in doubt
Token	The unit models are billed in, roughly a word fragment

References

- Flexera, 2026 State of the Cloud Report, on waste, generative AI adoption and AI governance
- FASB, ASU 2025-06, Subtopic 350-40, issued 18 September 2025
- IRC Section 174A and Section 168(k), as amended by the One Big Beautiful Bill Act, 2025

- theCUBE Research, "Resetting GPU Depreciation", on utilization and the value cascade
- DORA, 2025 State of AI-assisted Software Development, for adoption context

Best next read

Part 9, AI and Headcount. The spend is budgeted. Part 9 takes the harder question: whether the productivity evidence supports planning a smaller team, and what it actually shows.

The full series

PART	TITLE
1	CapEx, OpEx, and EBITDA: The Same Dollar, Two Different Stories
2	Reading the Financials: The Primer for Everyone Who Never Had To Care
3	Capitalize or Expense: Your Jira Board Is Now Audit Evidence
4	Depreciation and Useful Life: The Chip Outlives Its Economics
5	Placement Economics: The Right Answer, Applied Everywhere
6	Lift-and-Shift or Re-Architect: The Two That Modernized Later
7	Vendor Economics: The Email in November
8	AI Spend: Three Budgets Pretending to Be One, which is this one
9	AI and Headcount
10	Outsourcing After Section 174A

This series is an introduction for technology leaders. It is accounting and tax context rather than accounting or tax advice. Treatment is fact-specific. Model it

with your CFO and auditor before you commit.

CAREER & LEADERSHIP WPR-BIZ-009

// ABOUT THE AUTHOR

Jason Newell leads organizations and people in technology, and is a researcher, deep technologist and builder — from a CRAY to agentic AI. He founded MAX Research Collective to build AI for regulated industries that runs where its owner has control, with healthcare as its strongest focus. About Jason → Connect on LinkedIn →