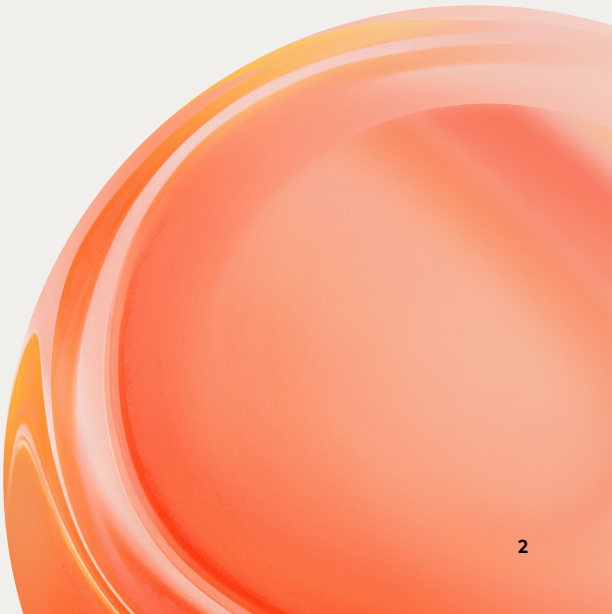
WHITE PAPER
SEPTEMBER 2026

The new economics of AI

How to build, budget and price AI solutions in the age of large language models

Contents



Executive summary

Large language model systems do more than introduce token costs: they turn intelligence itself into a variable economic input — consumed to deliver an outcome and, through use, converted into new intelligence, hence value.

The economics of an AI product are therefore determined not simply by usage or token volume, but by how much intelligence each customer outcome requires and how effectively the intelligence created through use is retained and reused. The central challenge is not simply to minimize tokens, but the selective allocation of intelligence: delivering each outcome with only the complexity it justifies, while ensuring that the resulting knowledge compounds for the organization and the customers.

Today, deploying intelligence and generating value from it are no longer the same thing. The gap between the two is an economics problem, and most organizations are not yet equipped to solve it. Unlike traditional software, where adding an extra user costs almost nothing, LLM costs are variable by design — they grow with usage, compound with complexity and erode margin whenever an experience is poorly designed or a technology choice is made without understanding its financial consequences. Gartner research reported that 70% of enterprise agentic implementations will fail to return the expected value.[1] McKinsey's 2025 global AI survey found that only 39% of organizations report any enterprise-level EBIT impact from AI.[2] These are not merely technology failures. They are economics failures. In the LLM era, poorly designed AI products do not merely create bad products; they create structurally unprofitable ones. It follows that product design is no longer only a customer experience discipline: It is an economics discipline.

This paper is organized into three sections:

SECTION 1: THE PROBLEM SPACE

- Introduction — why AI requires a new economic model
- The shift from software economics to AI economics
- The death of the fixed margin

SECTION 2: AI ECONOMICS

- Three cost pillars: adopt, build, provide
- Product and experience economics
- Architecture economics
- Commercial and revenue economics
- Capturing AI value — from deployment to P&L impact
- Illustrative example

SECTION 3: CONCLUSIONS

- Implications for organizations
- Recommendations for leadership

The paper argues that durable competitive advantage in AI will rarely come from access to a model alone. It will come from delivering specialized customer outcomes with greater effectiveness and lower cost than competitors: choosing where advanced intelligence adds value, designing the right architecture, pricing it sustainably, and capturing measurable business results.

Authors:
Neil Taylor, SVP, AI & Data Strategy
Mara Pometti, VP, AI & Data Strategy

Caveats: In this paper, "AI" primarily refers to LLM-based systems with varying levels of reasoning, autonomy, and orchestration. Deterministic technologies—including rules engines and traditional machine-learning models —have different cost structures and are not subject to the same token-based, reasoning-dependent economics described here.

All figures and workloads are illustrative and do not represent the costs, architecture, customers or commercial plans of any specific organization. API prices are based on publicly available provider pricing at the time of writing and may change.

1. With AI Agents, you need a new way to calculate cost and value, Gartner, January 2026
2. McKinsey, The state of AI in 2025, November 2025

01

The problem space

Introduction — why AI requires a new economic model

In 2024, a major European fintech publicly announced its AI assistant was performing the work of hundreds of customer service agents — before reversing course and resuming human hiring within a year. In 2026, a major city government paid more than $500,000 to deploy an AI agent to help businesses navigate public services — it produced consistently wrong answers and was shut down within months. Between those two headlines sits the most important lesson in AI economics: Deploying intelligence and generating value from it are not the same thing.

The gap is not only a technology problem. It is an economics problem most organizations have not equipped themselves to solve. Organizations approve or stop AI initiatives without understanding production costs, price AI-enabled products using logic inherited from SaaS, and measure ROI using frameworks designed for a world where software's marginal cost was effectively zero.

Leaders should ask five questions when dealing with LLM economics:

- How much does it cost us to consume AI for internal use cases for employees?
- How much does it cost us to build and deploy AI solutions for our customers?
- How should you price and sell both the products you build and services you deliver (e.g., consulting) with AI?
- How should you build these solutions in a way that is cost-efficient?
- What's the best trade-off you should strike without technology infatuations?

This paper addresses them through three connected economic lenses:

01 • The cost structure

LLM-based systems carry variable cost per interaction. Scale improves economics only when usage, architecture, and operating cost are actively managed.

02 • The product decisions

Product design, architecture, and pricing are economically coupled. Product choices determine the intelligence required; architecture determines its cost; and pricing determines whether that cost is recovered.

03 • The value capture

What you design determines both your cost floor and your revenue ceiling. The agentic complexity required to deliver an outcome is embedded in the design itself.

Strategic questions organized by archetypes

Costs

Adopt

- What is the real cost per employee of actually using the tools?
- Are we getting a productivity return on our seat licenses?

Build and deploy

- What is the full cost of keeping our AI stack running — proprietary model, third-party APIs, and the engineering required to sustain both?
- Where should workloads sit — our AI factory, external LLM providers, or both?
- If hybrid, how do we decide what runs where?

Provide to customers

- How much does every single customer interaction cost us?

Product design

Experience decisions

- What specific problem are we solving for the user?
- How "smart" (complex) does the AI actually need to be to solve it?

Architecture decisions

- How do we build this so it stays cheap to run as we grow?
- Are we using the most cost-effective model for the task?

Commercial decisions

- How do we price this to make sure we don't lose money as customers use it more?

Value capture

Customer outcomes and revenue

- How do we prove this AI is making us more money (revenue ceiling)?
- How do we prove it is making us more efficient (cost floor)?

The shift from software economics to AI economics

The frameworks, cost structures and revenue models we used to justify technology investments are no longer sufficient for AI agents — and that is because of the intrinsic nature of how these systems are built and how they work. Traditional technology was licensed software. You paid a fixed price. You could predict your costs and build a business case. AI agents are disrupting this model entirely. By design, agentic systems are highly unpredictable in terms of cost. They rely on inference — what the industry calls the act of a model processing input and producing output. Inference is the entire cost engine of AI.

Yet, before understanding the new AI cost structures, it helps to understand what is being paid for when an LLM processes a request and what the new economic units are:

When using LLMs, you or your customer pay every time the system thinks. When LLMs think, they process tokens — the smallest fragments of text a model reads or generates. A word is roughly 1.3 tokens. A sentence is twenty to thirty. A multi-turn conversation carries the entire history, reprocessed at every exchange. Tokens are not a unit of storage. They are a unit of active computation, billed in real time at rates set by whoever provides the model.

When a user sends a message to an LLM, the model does not search a database. It processes the entire input, calculates probability distributions across its vocabulary, and generates a response one token at a time. This process — where the LLM computes each token from scratch — is called inference. Inference requires GPUs or TPUs, specialized hardware that can run millions of mathematical operations in parallel. Standard processors cannot do this efficiently; this is why AI infrastructure is expensive and why the choice between cloud or owned hardware (AI factory) is a meaningful financial decision. While the user sees a conversation, the computer sees a series of discrete API calls, each one triggering inference.

The cost of reading inputs versus generating outputs (inference) reflects a different amount of computation at each stage. This produces distinct token types with different cost implications:

Token type	What it is	Cost implication
Input tokens	The full prompt sent to the model — system instructions, conversation history, retrieved context, and the user's query.	$2.50–$5.00 per million tokens (frontier models, 2026 pricing)
Output tokens	The generated response. Generated sequentially, one token at a time.	3–5x more expensive than input tokens. Dominates interaction costs.
Cached tokens	Tokens returned from a stored computation of an identical or near-identical prompt.	As low as 10% of standard input price. The most underused cost lever in production AI.
Reasoning tokens	Internal "thought tokens" a model generates before producing its answer. The user never sees them.	Even a 20-word answer may require 2,000 internal reasoning tokens — charged at output rates.

Unlike a search engine that indexes content once and retrieves it cheaply, an LLM performs a fresh computation on every request. The larger the model, the more parameters participate. The longer the input, the more computation required. The more complex the task — multi-step reasoning, tool use, reflection — the more times that computation repeats.

> ***More computation equals more cost. Scale does not reduce this the way software companies are used to. The computation does not get amortized across more users. It compounds with them.***

From this it becomes clear how tokens shape the new economy around AI. **Tokens are the unit of account for LLMs**, and therefore of every AI agent powered by them. Inference — and particularly output and reasoning — is where costs quietly compound, especially as you add multi-step flows to an agent's process.

Data centers are now AI factories with one job — generating tokens that we then reconstitute into music, words, videos, research, chemicals, or proteins. The product is not hardware. The product is not software. The product is tokens. With that lens, everything clicks:

- **AI data centers** = intelligence factories
- **Tokens** = economic unit
- **New cost and revenue structures** depend on tokens as the economic unit of intelligence

The implication for enterprise finance is direct. Every AI interaction is a manufacturing run. The cost of that run is not determined by the license fee on a pricing page — it is determined by the architecture of the system running it: how many tokens it consumes, what type of tokens, at what rate, across how many steps. AI is now an industrial production system, tokens are the output, and the entire enterprise stack needs to be redesigned around that fact.

The death of the fixed margin

We are moving from storing and retrieving information to generating intelligence in real time. AI breaks the SaaS economic model.

The software economics that most finance and commercial teams use as their reference frame are built on a single assumption that AI breaks: the near-zero marginal cost of an additional user. That is the SaaS mechanism — fixed infrastructure is built once and its cost distributed across an expanding user base. As users grow, fixed cost is diluted and margin climbs, reaching 80%-90% in high-performing SaaS businesses.

AI breaks this model. In AI, user #10,001 costs you real money: each additional user adds real inference cost. Each additional interaction adds compute. The marginal cost is not zero — it is determined by token volume, orchestration complexity, and infrastructure tier. Andreessen Horowitz identified the pattern early, finding gross margins among AI companies clustering at 50%–60% against a 60%–80% plus benchmark for comparable software businesses.[3]

Six years on, the pattern holds. ICONIQ's 2026 survey of software companies building AI products puts average gross margin at 45% in 2025 and a projected 53% in 2026 — improving, yet still structurally below software. What matters is the reason for the improvement. ICONIQ attributes it to inference cost management and model routing, not to the problem resolving itself. Margins in this band do not recover on their own. They move where the cost structure is actively managed.

Agentic systems intensify this, because they multiply computation per outcome rather than per user. Gartner estimates that agentic models consume between 5 and 30 times more tokens per task than a standard chatbot[4]. This is not a technical curiosity. It is an important financial consideration[5].

3. Martin Casado and Matt Bornstein, The new business of AI (and how it's different from traditional software), Andreessen Horowitz, February 2020.

4. Gartner predicts that by 2030, performing inference on an LLM with 1 trillion parameters will cost GenAI providers over 90% less than in 2025, Apr 2026

5. Gartner predicts that by 2030, performing inference on an LLM with 1 trillion parameters will cost GenAI providers over 90% less than in 2025, Apr 2026

Diverging Cost Structures: SaaS vs AI-Native App Economics

Illustrative — modeled figures, not empirical benchmarks

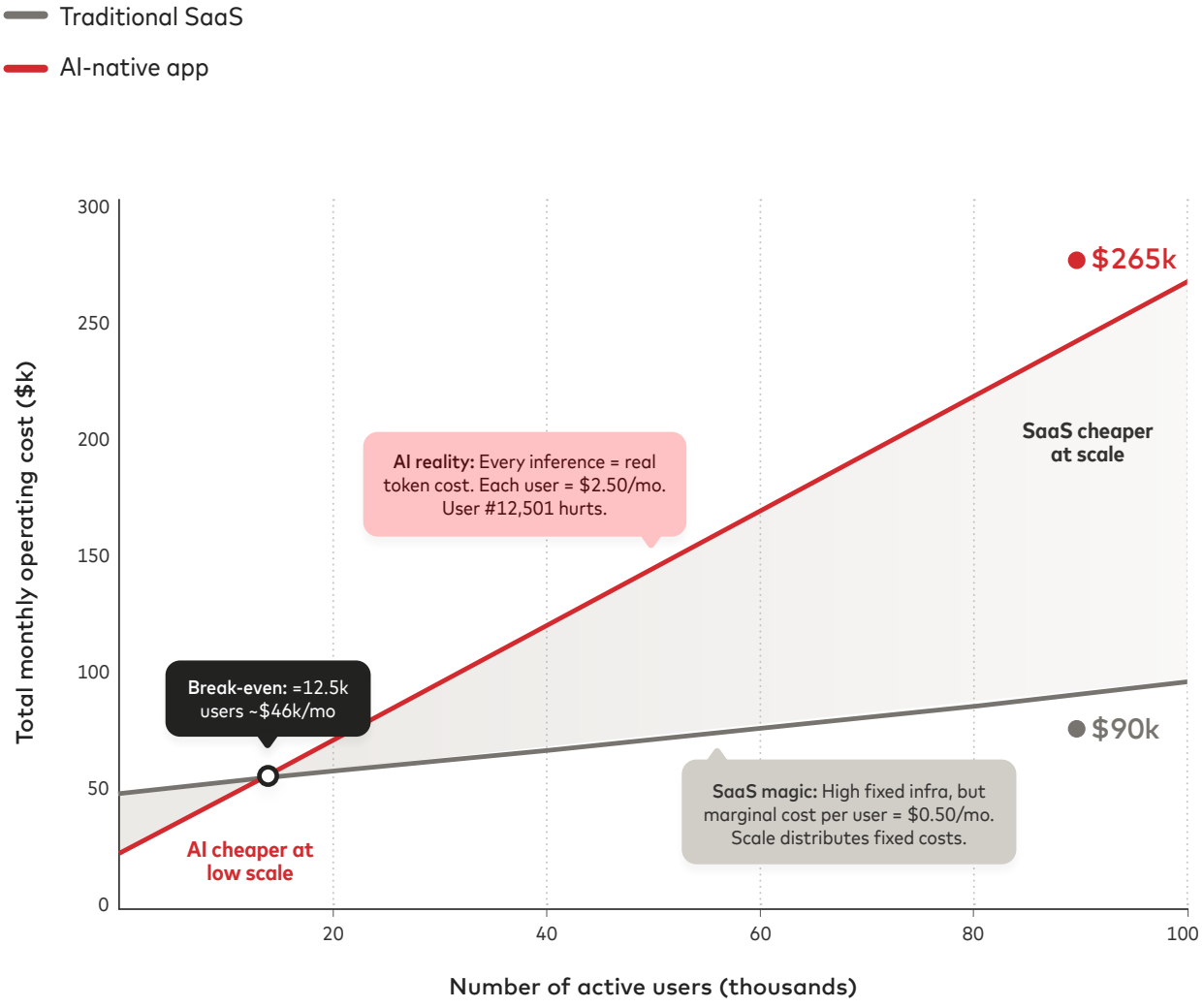


Exhibit 1. Illustrative. Diverging cost structures: SaaS vs AI-native app economics. Traditional SaaS assumes $40k fixed + $0.50/user/month. AI-native assumes $15k fixed + $2.50/user/month. Actual costs will vary significantly depending on model choice, usage patterns, and infrastructure decisions.

Source: Illustrative model.

The variability of AI costs has rapidly raised concern across the industry — even in world-class engineering teams. At one major social media platform, employees consumed 60 trillion tokens in a single month, and the company moved quickly to limit usage. Recently, there are examples of companies that exhausted their entire AI budget for the year in just four months — a pattern repeated at major players in enterprise software and cloud data platforms.

To summarize, the main differences between LLM-based systems and SaaS software are:

1. Costs are variable rather than fixed
2. Infrastructure and compute scale with usage and complexity
3. AI products require continuous optimization

	Traditional SaaS	LLM-based systems
Cost structure	Fixed infrastructure	Variable per interaction
Marginal cost per user	Near zero	Scales with usage and complexity
Gross margin	80-90%	~50–60% or less (and requires active management to maintain, it could be lower)
Budget predictability	High	Low without FinOps discipline
Key optimization lever	Mostly once at build time	Continuous — every interaction

Why the traditional product development model changes with AI

Traditional SaaS products can often move sequentially from prototype to pilot, MVP, production and extension. Features are added over time based on user demand and sales opportunity, while the marginal cost of serving additional usage remains relatively low. Architecture and operating economics can therefore be refined as the product matures.

AI products behave differently. Each feature can introduce additional model calls, context, memory, tools, agent handoffs, validation loops and retries. When new capabilities are layered onto an existing agentic system, they do not simply add functionality; they can compound the cost and complexity of every interaction. Decisions made during prototyping may therefore become embedded in the production architecture and become expensive to reverse later. Equally, simple features expected in production-grade products, such as an "undo" function or multiuser collaboration, can add disproportionate costs to an AI product if they were not part of the original design.

Example:

An AI assistant may begin as a simple cash-flow alert tool. Later, the team adds explanations, recommendations, scenario modelling, and automatic actions. Each addition can require more context, extra model calls, new tools, validation steps, and retries. What began as one low-cost interaction can become a chain of six or seven underlying calls. By the time the product reaches production, the original prototype architecture may be too expensive to scale — yet difficult to redesign because every new feature now depends on it.

This does not mean the full production solution must be built upfront. It means that the intended production experience and economics must be considered from the beginning. Early prototypes should test not only whether the experience works, but also how the system behaves: which tasks require AI, which model tier is sufficient, how much context is needed, where orchestration adds value, and what each workflow will cost at scale.

The AI product-development lifecycle therefore becomes:

Define
Specify the customer outcome, intended production experience, value metric, and acceptable cost-to-serve before development begins.

Prototype
Test the minimum agentic experience and identify the models, data, tools, and orchestration it requires.

Pilot
Validate customer adoption, system behaviors, cost-to-serve and whether the outcome can be measured and attributed.

Deploy
Scale only once the architecture, operating controls, pricing model, and target margin are viable.

Monitor
Track cost-to-serve, model behaviors, quality, usage, margin and outcome performance so drift or unexpected cost growth is identified early.

Extend
Add capabilities selectively, where the incremental customer value justifies the additional complexity and operating cost.

The key difference from traditional SaaS is that production economics cannot be deferred until launch. They must be defined upfront, tested in the prototype, validated in the pilot, monitored in production, and used as a condition for scale and extension. The next section examines the four economic decisions that shape those economics at every stage: product design determines what intelligence is required; architecture determines its cost; pricing determines whether that cost is recovered; and value capture determines whether the outcome reaches the P&L.

02

AI economics

Three cost pillars: adopt, build, provide

Organizations fail when they manage AI as a single budget line. AI costs do not sit in one place on the P&L. They accumulate across three distinct perspectives — each with different drivers and implications for how the organization should budget and price. Understanding the difference is the first step to managing them. What are the three costs every enterprise must track?

The cost of adopting AI

What you pay to give employees access to AI tools for their internal workflows. The metric that matters is cost per activated user, not cost per seat. PwC's Global Workforce Hopes and Fears Survey 2025, covering nearly 50,000 workers across 48 economies, found that 14% use generative AI daily and that daily users are far more likely to report productivity gains than infrequent users. Value concentrates sharply in the smallest group. The economic consequence is that a license base and an activated base are different populations, and only one of them is earning the fee. Any organization can calculate its own exposure: Divide total license spend by daily active users rather than seats issued. Where the two diverge, the effective cost per productive user is a multiple of the headline price. Access does not create value. Activation and adoption do.

The cost of building and deploying AI solutions

This is what an organization pays to create and sustain AI-powered products and workflows. External APIs require little upfront capital and charge by consumption. Owned or reserved infrastructure requires greater commitment but may reduce marginal cost for large, stable workloads. Compliance, data residency, operational resilience and security requirements may constrain where workloads can run, particularly in regulated sectors; these constraints should be assessed against the applicable legal and supervisory requirements rather than assumed from regulation alone.

The cost of providing AI to customers

What you pay every time a customer uses an AI product you have built. Unlike SaaS, where serving an additional user costs near zero, every AI answer carries a direct compute cost in real time. Each customer interaction burns inference tokens and triggers API calls. These are real-time costs of goods sold that your pricing model must recover to sustain margin.

Based on these cost pillars, the traditional enterprise technology stack that was designed for a world where compute was a fixed cost and serving an additional user was essentially free is now changing. AI inverts the old IT and infrastructure model. The stack is no longer infrastructure to be provisioned — it is a production system to be managed, where every layer carries a cost that scales with usage. Two categories of cost now operate simultaneously:

- **Operational costs (OpEx)** are incurred every time the system runs: model inference, agent orchestration, data retrieval, quality evaluation, and monitoring. These scale directly with usage.
- **Capital costs (CapEx)** are incurred when building the system: data pipelines, model fine-tuning , agent architecture, integrations, and infrastructure. These are largely upfront and amortized over time. These can also, depending on accounting rules, be categorized as research & development

Most enterprise AI budgets account for the capital build. Few fully account for the ongoing operational costs, which is where the majority of total cost accrues over a deployed system's life. As AI workloads scale, variable costs associated with compute, storage, inference, monitoring, and model maintenance can become a significant source of expense. If these costs are not incorporated into pricing and commercial models, they can erode margins over time. A cost base that is not fully understood or accounted for cannot be effectively priced against, and unpriced variable costs ultimately manifest as margin compression.

For enterprises building LLM solutions, the question is not which to choose — it is how to balance both. AI system costs fall into two categories that operate on different financial models and must be planned for separately.

Product and experience economics

Most organizations make their technology decisions before their experience decisions. That sequencing is what creates the economics problem.

The experience an AI product or feature delivers is not separate from its cost structure — it is the cause of it. The agentic complexity required to deliver a given outcome: how many agents, how many tool calls, what retrieval architecture, what guardrail levels, how much context the system maintains, is embedded in the experience design. A system designed to reactively answer queries uses a fundamentally different — and cheaper — architecture than one designed to proactively detect a problem and initiate action without being asked. The difference is not a configuration choice. It is a design choice that manifests directly in infrastructure cost.

Designing a good AI product or feature is not a UX investment that happens to improve economics. It is the primary economic decision — the one that determines whether the token bill is a cost of value creation or a cost of waste.

Product and feature design are the only levers that directly affect both sides of the AI margin: The value created for the customer and the cost of delivering it.

The most consequential economic mistakes happen when the level of product complexity does not match the value of the use case. Organizations may over-engineer a simple task with multiple agents, deep reasoning, and expensive orchestration when a lightweight workflow would produce the same outcome. They may also under-engineer a complex or high-stakes use case, limiting its reliability and customer value.

These decisions should therefore be made at the product-design stage, before the architecture and infrastructure are committed.

For example, some features create meaningful value at relatively low cost. Cash-flow insights, anomaly alerts, and spend nudges can often rely on structured data and lightweight models. They are inexpensive to operate, but can materially improve adoption, engagement, and retention.

Other capabilities are costly but strategically valuable. Scenario planning, FX simulations and competitive ecosystem modeling may require advanced reasoning, multiple data sources and more complex orchestration. Their higher delivery cost can be justified when they create differentiated outcomes that customers are willing to pay for.

A third category includes essential but undifferentiated functions, such as transaction categorization, basic reporting and frequently asked questions. These capabilities may be necessary to complete the experience, but they should be delivered as efficiently as possible.

The greatest economic risk comes from features that are expensive to operate but create little measurable value: over-engineered experiences, complex reasoning without a clear customer outcome, or agents that customers rarely use. In these cases, costs increase while impact remains flat.

The discipline of AI product design is therefore to allocate complexity selectively: invest heavily where it creates differentiated or measurable value, keep foundational capabilities inexpensive, and remove features whose cost consistently exceeds their contribution. Applying the same model, reasoning depth, and orchestration architecture to every use case removes this discipline. It increases costs without improving outcomes and can result in the most expensive features delivering the least value.

Architecture economics

AI architecture is often treated as a technical implementation choice, but it determines the unit economics of every interaction. Decisions about model selection, reasoning depth, context, memory, orchestration, and infrastructure define how much each customer outcome costs to deliver. In an age of compute scarcity and variable inference costs, competitive advantage belongs to organizations that treat architecture as a cost discipline — designing systems to deliver the required outcome with the minimum viable complexity.

Most engineering teams, however, treat architecture decisions as something to address after deployment. In AI, that is often too late. Many of the decisions that determine cost, performance, and customer experience are made at design time, once the product, its use cases, and the outcomes to be delivered are defined. These decisions are not merely technical; they are financial ones.

Providers are improving inference economics rapidly, and it is tempting to read this as a problem that resolves itself. Gartner's forecast is that inference on a trillion-parameter model will cost providers over 90% less in 2030 than in 2025[6]. Its conclusion for enterprises is the opposite of reassuring: those declines will not be fully passed through to customers, and because token consumption rises faster than unit costs fall, total inference spend is expected to increase rather than decrease.

The mechanism is visible in provider behavior. The engineers of the major frontier labs had found ways to optimize more than half the cost of running existing models, cutting the GPUs required to serve logged-out the frontier model's traffic to a couple of hundred. The model did not change. The way it was served did.

The same logic operates one layer up, where enterprises sit. One of the major frontier labs reports that prompt caching can reduce cost by up to 90% and latency by up to 85% for long, repeated prompts — but only for systems architected so that prompts are cacheable. The saving is available to anyone; it is captured only by those who design for it.

Efficiency gains accrue first to the party that makes them. They reach the next party in the chain only through a deliberate pricing decision, and often only in part. An enterprise waiting for provider price declines is waiting on someone else's commercial timetable. The gains it engineers into its own architecture arrive immediately and in full.

Architectural choices can reduce the cost of an AI solution without weakening the customer outcome. Systems can route classification, summarization, and extraction to smaller models while reserving frontier models for work that genuinely requires complex reasoning. Organizations that operate their own models may also use techniques such as quantization and distillation to reduce computational requirements. Application teams can limit token consumption by retrieving only the context relevant to a request rather than repeatedly processing full histories.

Other techniques reduce the work performed on each request. Prompt caching reuses previously computed prompt prefixes, context pruning removes irrelevant information, and batch processing moves non-urgent tasks to lower-cost execution. Speculative decoding can improve serving efficiency in model-hosting environments by using a smaller draft model whose outputs are verified by a larger model.

The specific techniques will continue to evolve. The underlying economic principle is more durable: architecture should allocate expensive computation selectively, using the lowest-cost configuration capable of delivering the required level of quality, reliability, and customer value.

The broader principle is that inference should not be treated as a fixed pipeline. Efficient systems dynamically determine which model, context, memory and processing method each request requires. The objective is not to minimize cost indiscriminately, but to spend computational resources only where they improve the outcome. Some platform providers offer generic routing features that route prompts to the most appropriate grade model. However, enterprise wide routing does not remove the need for product-level discipline. A central platform may direct requests to the most appropriate model, but it cannot compensate for a product that uses AI unnecessarily, carries excessive context, invokes too many tools or agents, or applies complex reasoning by default. Routing optimizes the model selected for a request; product and architecture design determine whether that request — and its full chain of computation — should exist in the first place.

6. Gartner predicts that by 2030, performing inference on an LLM with 1 trillion parameters will cost GenAI providers over 90% less than in 2025, Apr 2026

A second architectural decision sits above request-level optimization: which models the organization uses and where those workloads run.

- Proprietary frontier models accessed through cloud API providers offer immediate access to state-of-the-art capability with minimal upfront investment. They are well suited to experimentation, rapidly evolving use cases, and complex reasoning, but their costs scale with usage.
- Open-weight models (often referred to as "open-source" models), deployed on dedicated or owned infrastructure, offer greater control over data, architecture, and operating cost. When quantized, distilled, or specialized for a domain, they can serve large volumes of repeatable tasks at substantially lower marginal cost, although they require investment in infrastructure and engineering.
- Organization-built or heavily customized models represent the highest level of investment. They require significant expertise and training costs but may be justified where organizations need proprietary capabilities, highly specialized performance, regulatory control, or long-term cost advantages at very large scale.

Hybrid architectures increasingly combine all three approaches, selecting the source and cost of intelligence according to the outcome required: frontier proprietary models for exceptional reasoning, lower-cost open-weight models for predictable high-volume specialized workloads, and internal models where strategic differentiation, control, or regulatory requirements justify ownership. The objective is not to maximize model capability across the stack, but to allocate the most expensive intelligence only to the use cases where it creates commensurate value.

Together, the architectural decisions and the infrastructure placement decision form the complete cost structure of an AI solution.

The economic question is therefore not whether cloud or owned compute is universally better. It is when a workload becomes sufficiently stable and large for the cost advantage of greater infrastructure control to outweigh the flexibility of external APIs. Organizations that postpone this decision may discover that a successful product has also created an unnecessarily expensive operating model.

Enterprise scenario	Cloud API	Hybrid	On-premises
Employee coding and productivity tools	Best fit. Pay per use, no capital commitment, easy to scale	Not typically needed	Rarely justified
Internal AI solutions (custom workflows)	Right for early stages and variable demand	Right as usage grows and patterns stabilize	Consider when volume is high and data sovereignty is critical
Customer-facing AI products (agents, agentic workflows)	Start here. Validates economics before committing capital	Optimizes cost as volume becomes predictable	Often the most cost-effective choice at scale. Evaluate breakeven

Commercial and revenue economics

The commercial decision — how to price and sell AI-powered products — is where the cost, product, and architecture economics meet the market. Most organizations are making this decision with tools inherited from SaaS, and the result is pricing that systematically fails to recover the real cost of delivery. AI pricing must do more than support sales. It must ensure that revenue grows in line with the cost and value of delivery.

Four pricing models are emerging:

Pricing model	Margin implication	When it works
Per seat **What you charge for:** Access, regardless of usage	Dangerous for AI: costs are variable, but revenue is fixed. As usage grows, cost grows — revenue does not.	Only viable if usage is predictable and capped.
Per-use **What you charge for:** Consumption — API calls, tokens processed	Aligns with cost structure, but creates customer unpredictability. Does not tie revenue to business outcomes.	Internal tooling, developer platforms.
Per-workflow **What you charge for:** Completed actions — resolved cases, processed documents	Closest alignment between cost (what the workflow consumes) and revenue (what it delivers).	B2B automation, document processing, back-office AI.
Per outcome **What you charge for:** Defined business results — fraud prevented, revenue generated, cost saved	Maximum alignment. As underlying model costs fall, margin expands. Requires robust outcome measurement.	High-value enterprise AI where outcomes are attributable.

Why per-seat pricing is risky for AI

Per-seat pricing assumes that the cost of serving each user is broadly similar. That is generally true for traditional software, where the marginal cost of additional usage is low. It is not true for AI. A casual user making simple requests may cost very little to serve, while a power user running long, multi-step workflows may consume significantly more models, context, tools, and compute. Under flat pricing, both generate the same revenue despite having very different costs. As adoption and usage grow, the most engaged customers can therefore become the least profitable.

This does not mean per-seat pricing is impossible. It means it usually requires usage limits, tiered access, credits, or other controls that prevent consumption from becoming disconnected from revenue.

In an agentic world, the value exchange shifts from access and usage to outcomes: customers should pay for what the value agents actually create, not for a seat license.

How the market is adapting

One example of this model in practice: Emerging companies specialized in building AI agents for customer service are already pricing around completed outcomes — resolved cases, prevented cancellations or successful upsells — rather than model calls. The customer pays for the result. The alignment with customer value is strong, and the constraint is equally clear: It works only where the outcome can be measured reliably and attributed cleanly to the agent rather than to the employees, systems and customer behavior surrounding it. That constraint is narrowing rather than disqualifying. The shift from selling software to selling outcomes is the structural direction of travel in AI business models, because it follows the change in what is being delivered. Where an agent completes work rather than assisting a user, the seat has stopped describing the thing the customer is buying, and pricing that still measures access is measuring the wrong variable.

Some AI-native developer tools have moved toward credit-based consumption models in which users draw down an allowance based on actual token usage. More demanding tasks consume more credits, allowing revenue to rise with the underlying cost of serving the user.

The broader lesson is that AI pricing cannot be separated from product and architecture decisions. Price on the most proximate outcome you can cleanly attribute, not the most ultimate outcome you wish you could claim. A $500/month AI product that reliably prevents a $5,000 liquidity crisis has a 10× value ratio. That is the pricing anchor — not the cost of inference, not the price of a competitor's SaaS license. The pricing model must reflect what drives cost, what the customer values, and how reliably the outcome can be measured. Otherwise, greater adoption may increase usage without improving profitability.

Capturing AI value — from deployment to P&L impact

Pricing AI correctly is necessary, but it is insufficient. Deploying AI does not automatically create financial value. The final economic challenge is ensuring that improvements in speed, quality or decision-making translate into measurable business results.

This is difficult because AI often creates value indirectly. It may make hundreds of tasks faster across multiple teams without producing an immediate or visible change in revenue or cost. The technology can perform successfully while the investment still fails the organization's ROI test because the resulting value is dispersed, poorly attributed, or never converted into a financial outcome.

Productivity creates capacity, not automatic savings

One of the most common measurement errors is to treat time saved as money saved. When AI allows employees to complete work faster, the organization does not automatically reduce its costs. The time released may simply be absorbed by additional meetings, email or other low-value activity. Consider a consulting team that uses AI to complete in two days work that previously took two weeks. The firm has not saved the remaining eight days of salary. It has created additional capacity. That capacity becomes valuable only when it is deliberately converted into something the business can recognize, such as:

- Serving more customers;
- Increasing delivery volume;
- Improving the quality of the work;
- Reducing time to market; or
- Avoiding future hiring or external expenditure.

The value therefore lies not in the time released, but in what the organization does with it.

The attribution problem

Attribution becomes more difficult as AI systems take on broader workflows. An agent may help resolve a complaint, qualify a sales opportunity, or process an application, but the final outcome may also depend on employees, customer behavior, existing systems, market conditions, and operational processes. The further the claimed benefit sits from the AI's direct action, the harder it becomes to prove that the AI caused it.

This matters because finance teams are unlikely to recognize savings or revenue gains that cannot be supported by credible evidence. Broad claims such as "improved productivity," "better customer experience," or "higher revenue potential" may describe genuine benefits, but they are too diffuse to support investment decisions or appear clearly on the P&L.

Value capture must be designed before deployment

The solution is not to deploy first and attempt to calculate value afterwards. Each initiative should begin with a specific business outcome that can be baselined, measured, and linked credibly to the AI intervention.

The strongest measures usually sit close to the action performed by the system. A customer-service agent may not be able to claim responsibility for overall customer lifetime value, but it can credibly influence resolution rates, handling time, escalation rates, and service costs. A sales agent may not own total revenue, but it can be measured against qualified leads, conversion between defined stages, or time required to prepare a proposal.

The implementation should then be designed around that outcome, including:

- the baseline before deployment;
- the operational metric the AI is expected to change;
- the financial value associated with that change;
- the owner responsible for converting the improvement into business value; and
- the mechanism through which the result reaches revenue, cost, capacity, or risk.

Without this discipline, organizations risk measuring activity rather than impact: prompts submitted, users activated, hours nominally saved, or tasks completed. These measures show that the system is being used, but not whether the business is economically better off.

Converting operational gains into financial value

AI value typically reaches the P&L through a limited number of pathways:

- **Revenue growth:** increasing conversion, retention, transaction volume, cross-sell, or speed to market.
- **Cost reduction:** reducing external expenditure, manual processing, rework, service demand, or required staffing.
- **Capacity creation:** enabling the same workforce to process more work or avoid incremental hiring.
- **Risk reduction:** preventing fraud, errors, losses, compliance failures, or operational disruption.

The organization must make the conversion pathway explicit. For example, reducing the time required to process an application creates capacity. It becomes financial value only if the business increases application volume, redeploys employees, avoids additional hiring, or reduces outsourcing costs.

The distinction is critical: an operational improvement is not yet a captured financial benefit.

Measure what the AI can credibly influence

The right metric must be economically meaningful but close enough to the AI's action for attribution to remain credible. Metrics that are too narrow may prove that the system worked but fail to demonstrate material value. Metrics that are too broad, such as total revenue or company-wide productivity, are influenced by too many variables to attribute confidently.

The objective is to identify the nearest measurable outcome that connects the system's performance to a financial consequence. Value capture therefore depends on three linked questions:

- What measurable outcome did the AI change?
- How does that outcome create financial value?
- What action ensures that value is realized rather than absorbed?

AI does not create P&L impact merely by performing work faster or better. It creates impact when an operational improvement is deliberately converted into revenue, cost reduction, productive capacity, or avoided risk.

AI value capture compounds in a way SaaS never did. In SaaS, value erodes as competitors replicate features. In AI, every interaction deepens the institutional knowledge embedded in the system — knowledge that belongs to the deployment, not the model provider.

Illustrative example

Consider an AI Virtual CFO serving **1,000 small businesses** and processing approximately **10,000 requests per day, or 300,000 per month**. The service provides routine cash-flow alerts, financial analysis, financing support, and complex simulations. To the customer, these capabilities appear within one product. Economically, however, they are very different workloads.

The central question is not simply how many customers the product serves, but how much intelligence each request requires. Treating every interaction as a complex agentic workflow creates a very different cost base from routing each task to the lowest-cost method capable of delivering the required outcome.

For illustration, monthly demand is assumed to consist of:

- 70% routine tasks: transaction explanations, categorization, alerts and summaries;
- 20% workflow-level tasks: cash-flow analysis, supplier planning and financing preparation;
- 10% complex tasks: liquidity stress tests, expansion decisions and financing simulations.

Product design determines where cost is created

The first decision is not which model to use. It is which customer needs justify greater intelligence, personalization, autonomy, and workflow complexity. Routine capabilities may be used frequently and create strong engagement while remaining inexpensive to provide. Complex simulations may create more value per interaction, but require more data, reasoning, orchestration, and oversight.

The product should therefore avoid applying the same level of intelligence to every interaction. Routine guidance should remain lightweight and inexpensive; advanced reasoning should be reserved for decisions where it materially changes the customer outcome.

Illustrative

Virtual CFO use case	Value to customer	Cost to deliver	Margin implication
Transaction categorization and explanations	Low	Low	Necessary foundation, but limited differentiation. Margin depends on delivering it at minimal cost.
Cash-flow alerts and anomaly detection	High	Low	Attractive margin: frequent, visible value delivered through lightweight models and structured data.
Financing preparation and comparison	High	Medium	Strong margin potential if deeper analysis supports better customer decisions and is priced as an advanced workflow.
Liquidity stress testing and expansion simulation	High	High	Differentiated but expensive. Margin depends on premium pricing or outcome-linked fees that reflect the value of the decision.

Architecture: allocate compute according to the task

Architecture converts the product hierarchy into unit economics. A cost-aware system first asks whether an LLM is needed at all. Rules, deterministic workflows, traditional machine learning, or small language models may handle threshold alerts, reconciliation checks, categorization, and structured calculations more cheaply and reliably. Mid-tier LLMs can support more ambiguous analysis, while frontier models and multi-agent orchestration should be reserved for complex, high-value decisions.

Current API pricing shows why model selection matters. As of mid-2026, input costs across leading providers range from roughly $1 to $5 per million tokens for standard models, with output costs running three to six times higher. Providers typically offer lower rates for cached inputs and 50% discounts for eligible batch workloads. These prices are the starting point, not the whole cost.

In an agentic system, cost compounds beyond the first model call. Additional context increases input-token consumption; memory retrieval adds more information to the prompt; every agent hand-off creates another call; tool results must be interpreted; and failed attempts or validation loops trigger retries. Guardrails, which in production systems are critical, also add costs through the same mechanisms, sometimes more than doubling the cost of the underlying interaction. What appears to the customer as one interaction may therefore become several underlying calls, each carrying its own inference, context, memory, tool, and orchestration cost.

Cost-unaware versus cost-aware design

The examples below compare two ways of delivering the same AI product to the same customers at the same level of demand.

Cost-unaware design

Treats model capability and agentic complexity as defaults. It uses powerful models, long context windows, memory retrieval, multiple agents, and tool calls across a broad range of tasks, without first defining whether each element is necessary for the customer outcome to deliver.

Cost-aware design

Starts with the opposite question: what is the lowest-cost combination of rules, models, context, tools, and orchestration that can deliver each task at the required level of quality, reliability, and value, and collectively achieve the expected customer outcome.

The difference is therefore not simply which model is selected. It is whether cost is treated as a design constraint from the outset. The same demand can produce very different operating economics depending on how selectively intelligence and complexity are applied.

Total Monthly AI Cost: Cost-Unaware vs Cost-Aware design

Illustrative example based on the fictional AI Virtual CFO use case described above
Same 1,000 customers · Same 300,000 monthly requests

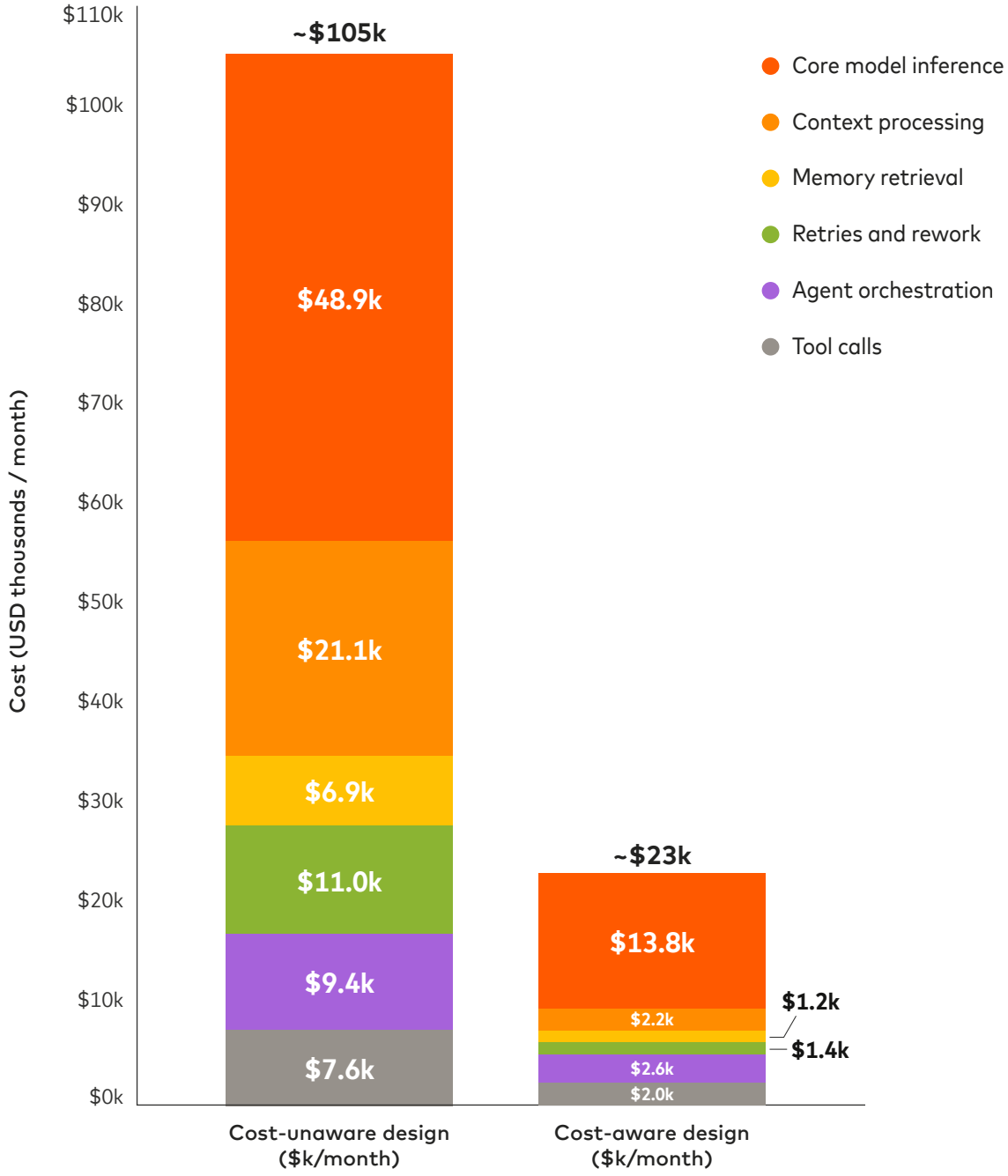


Exhibit 2. Illustrative. Total monthly AI cost including memory retrieval: approximately $105k for the cost-unaware design versus $23k for the cost-aware design. This is a 78% reduction, or about $82k per month. The cost-unaware design is approximately 4.5 times as expensive.

Source: Illustrative model. Inference anchored to published OpenAI and Anthropic API pricing, July 2026.

What drives the cost difference

For the same 1,000 customers and 300,000 monthly requests, the cost-unaware design reaches approximately $105,000 per month, compared with $23,000 for the cost-aware design.

The difference comes from four design choices:

	Cost-unaware design	Cost-aware design
1 **Routine tasks**	Frontier model, long context, and agentic workflow by default	Rules, deterministic workflows, traditional ML, or a small/ open-weight model; use frontier LLMs only where language understanding is required
2 **Workflow-level analysis**	Frontier model with full orchestration for every request	Smaller proprietary or open-weight models supported by structured data and workflow logic; escalate only when additional reasoning creates value
3 **Complex simulations**	Frontier model and multi-agent orchestration	Frontier model, deeper reasoning, and selective orchestration where value and risk justify the cost
4 **Context and memory**	Full customer history and memory repeatedly processed	Retrieve only relevant context, cache stable instructions, and use structured data wherever possible

These choices reduce:

- Core inference from **$48.9k to $13.8k**
- Context and memory costs from **$28.0k to $3.4k**
- Retries and rework from **$11.0k to $1.4k**
- Agent and tool orchestration from **$17.0k to $4.6k**

The result is a reduction of approximately $82,000 per month, or 78%, without changing the customer base, demand or portfolio of capabilities.

The following chart shows where that reduction comes from. The largest saving, for example, comes from not using premium intelligence where it is unnecessary. Because routine demand represents 70% of requests, routing it to rules, workflows, traditional models, or smaller LLMs has a disproportionate effect on inference cost. Further savings come from controlling the components that compound that cost: pruning context, structuring memory, caching stable instructions, reducing agent hand-offs and preventing avoidable retries.

Total Monthly AI Cost: Cost-Unaware vs Cost-Aware design

Same 1,000 customers · Same 300,000 monthly requests

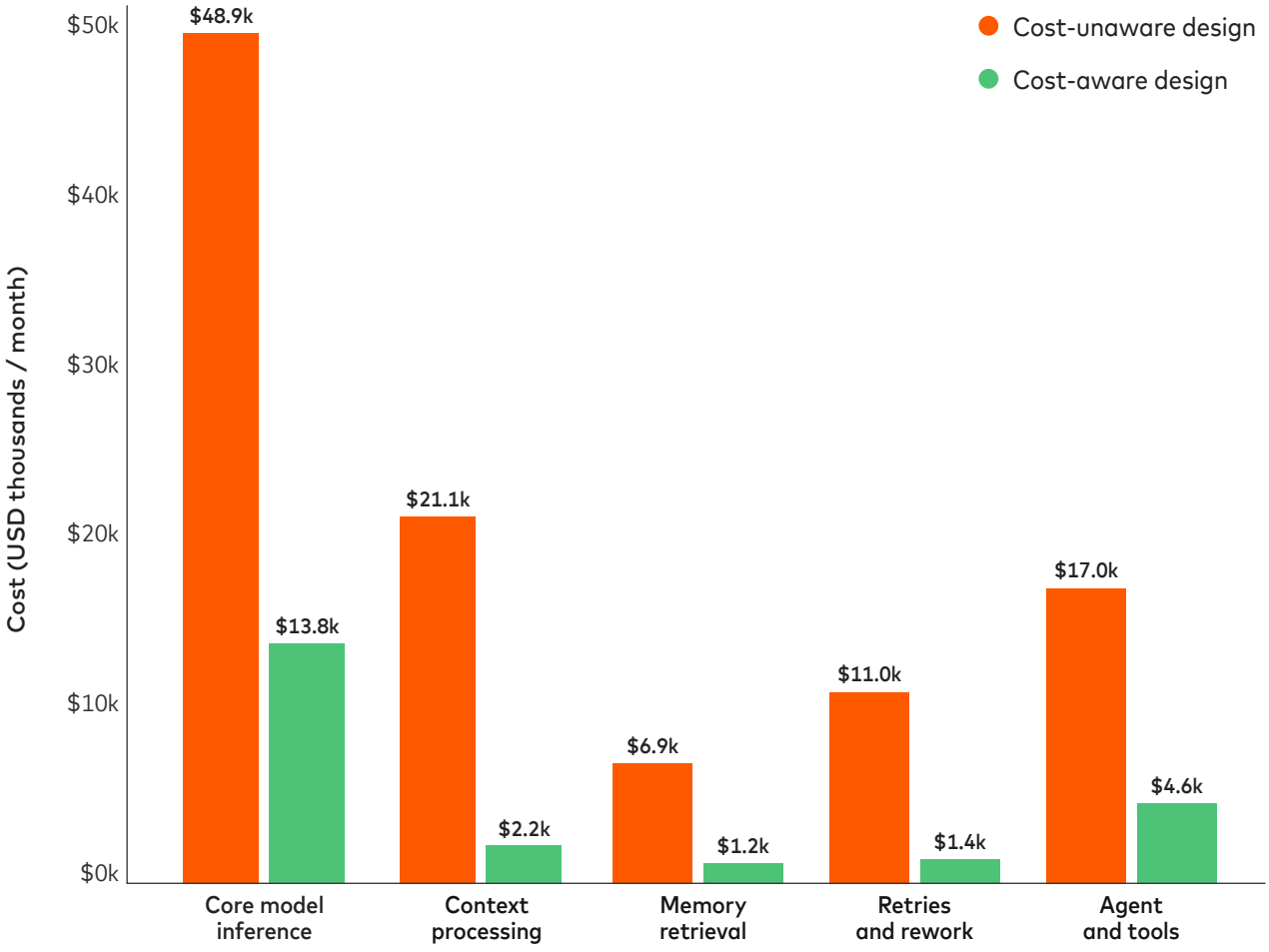


Exhibit 3. Illustrative. Component-by-component comparison of five primary cost categories. The corrected totals are $104.9k versus $23.2k, matching Exhibit 2. Actual costs will depend on token volumes, reasoning depth, tool pricing, negotiated enterprise rates, external data fees, and customer behavior.

These figures are illustrative rather than a forecast. Core inference is anchored to published OpenAI and Anthropic API pricing; the remaining categories use explicit workload assumptions to show how context, memory, retries, tools, and orchestration can compound spend. Actual costs will depend on token volumes, reasoning depth, tool pricing, negotiated enterprise rates, external data fees, and customer behavior.

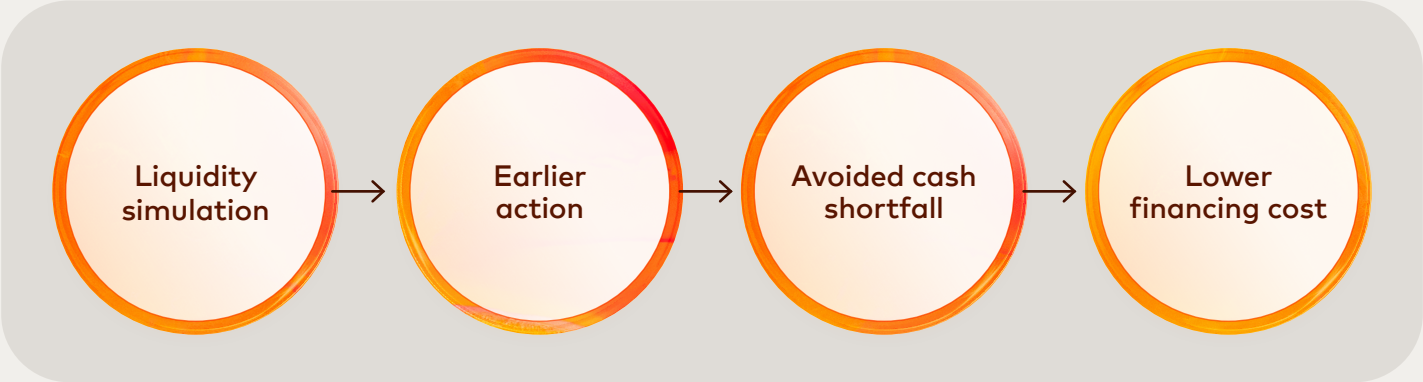
Commercial model: align revenue with usage and complexity

The product and architecture decisions determine which commercial models are economically viable. A subscription can cover routine services whose cost is predictable. Credits can bridge fixed subscription revenue and variable AI consumption: alerts and summaries may use few or no credits, while financing assessments, liquidity forecasts, and simulations consume more. This preserves simplicity for the customer without forcing light users to subsidize power users.

As complexity grows, pricing can move closer to usage or completed workflows. A customer may pay separately for a financing assessment, a liquidity stress test, or a scenario-planning session. The price should not be derived mechanically from token cost — the customer is buying a useful financial workflow — but the underlying cost establishes the minimum level required to protect margin. For the highest-value engagements, fees can move toward measurable outcomes such as reduced liquidity risk, lower financing costs, or improved working-capital performance.

Value capture: connect the workflow to a financial result

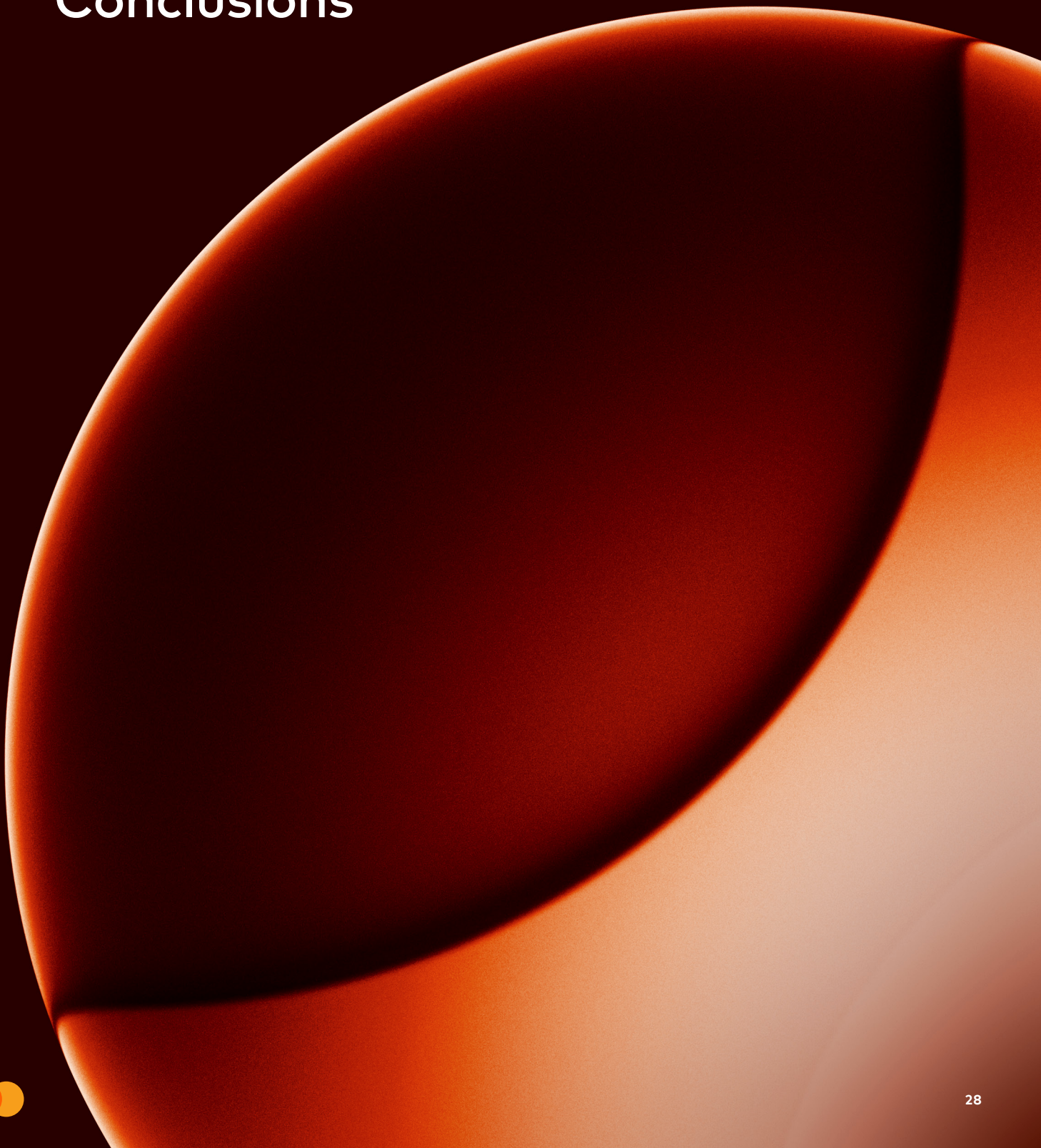
The final question is what the Virtual CFO actually delivers for the customer. A liquidity forecast has no economic value simply because it is accurate. Value is captured when it enables the business to act earlier, avoid a cash shortfall, reduce emergency borrowing, or improve the timing of supplier payments.



That chain connects the full economic model. Product design determines where advanced intelligence is justified. Architecture determines how much it costs to deliver. Context, memory, orchestration, and retries determine how that cost compounds. Pricing determines whether the variable cost is recovered. Value capture determines whether the customer receives—and pays for—a measurable financial result.

03

Conclusions



Implications for organizations

AI must be governed on lifetime economics, not build cost
Agentic products create variable and persistent operating costs that can grow with adoption.

Value must be measurable before deployment
Every initiative needs a clear business outcome that can be attributed, tracked and translated into P&L impact.

Product design is now a margin decision
Customer outcomes, feature complexity, architecture and pricing must be designed together from the outset.

Shared AI infrastructure is an economic advantage
Common FinOps, evaluation, safety, and telemetry capabilities reduce the cost and risk of every future product.

Unit economics must determine what scales
High-value, attributable use cases should receive compute and investment; high-cost, low-value initiatives should be stopped early.

Recommendations for leadership

01 **Establish a standard AI economics baseline**
Define cost per interaction, workflow, and outcome and model cost-to-serve by use case, segment and heavy-user behavior.

02 **Adopt a clear decision sequence**
Start with the customer outcome and target margin, design the minimum viable agentic experience, then select the model, orchestration and infrastructure.

03 **Make AI FinOps mandatory**
Build in budgets, rate limits, fallback models, caching, context controls and graceful degradation before production.

04 **Standardize architecture optimization**
Use model routing, structured outputs, context pruning and deliberate batch-versus-real-time processing across the portfolio.

05 **Align pricing with variable cost and value**
Use subscriptions for predictable usage, credits or per-workflow pricing for variable demand and outcome-based pricing only where attribution is robust

06 **Require one measurable outcome per initiative**
Set the baseline, owner, measurement window and financial conversion path before deployment.

07 **Embed the framework in governance**
Make economics a formal gate for funding and scale, assign named cross-functional owners and review cost-to-serve, margin and outcome performance regularly.