You are not a model. Don’t price per token.

Charging per token is becoming the default for AI companies. For most AI applications, it is a mistake.

TUGCE ERTEN AND SARAH WANG

AUG 27, 2026

America | Tech | Opinion | Culture | Charts

Token pricing began in the right place: the model layer. When OpenAI launched its API in 2020, charging for the computation a model consumed was a sensible way to meter raw inference. However, ChatGPT’s debut two years later helped spark a wave of applications built on that infrastructure that do much more. These new products combine proprietary data, tools, orchestration, integrations, and workflow logic to complete work on a customer’s behalf.

When an application prices that work in tokens, it imports the model provider's cost structure into the relationship with the customer and anchors the product's value to a unit whose cost keeps falling. Based on our work, it's often a mistake to carry the model layer's pricing logic into the application layer.

Instead, we believe companies should price at the highest layer of value that they can reliably measure, attribute, and defend.

- If you sell model access, price tokens.
- If you turn models into useful work, price the recognizable value unit, often through credits.
- If you deliver a clear and attributable business result, price the outcome.

If You Sell Value, Don't Price Per Token

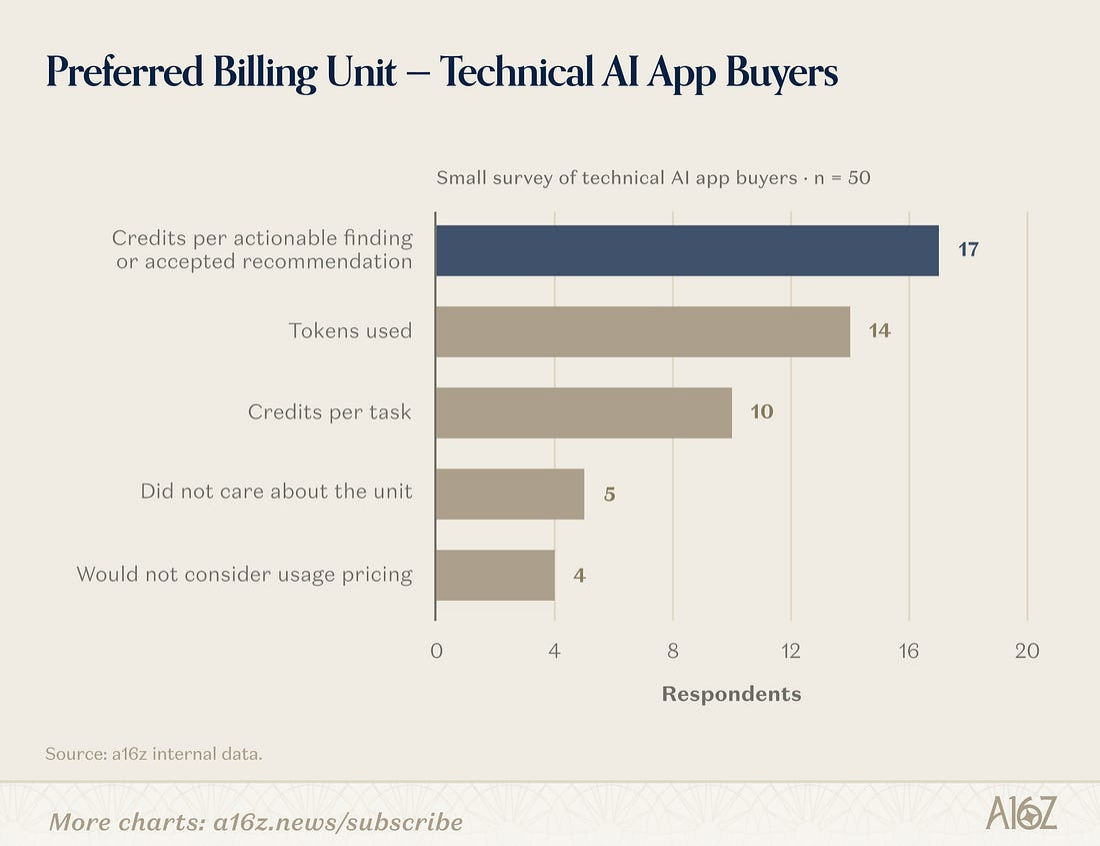
Layer	What the customer buys	Best pricing unit
Model	Raw inference	Tokens
Application	Useful work	Recognizable value unit, often packaged in credits
Outcome	Measurable business result	Outcome itself

Note: For illustrative purposes only.

A16Z

Getting this wrong is difficult to undo. A token-based price trains customers to compare an application with raw compute. It gives away the value of data, workflow, and orchestration; exposes customers to technical complexity they cannot forecast; and can lock the vendor into weak margins. Credits do not solve that problem on their own if they are merely cost-plus tokens; they still meter infrastructure as opposed to value.

Pricing around the recognizable work does the opposite. It makes value legible, spend forecastable, and product improvements economically valuable to both sides. In our survey of 50 technical AI buyers, 27 preferred credits tied to recognizable work, while only 14 preferred tokens.



1. Price the Layer You Sell

The two ends of the stack are relatively straightforward. Model providers sell inference, which can be metered in tokens. Some applications deliver outcomes that are observable and attributable enough to price directly.

The middle is harder. That's where most AI applications live.

For example, an account-research agent is not selling searches and model calls. It is selling a completed account brief. A coding agent is selling an implemented change. A data platform might sell a completed query, pipeline, or agent run. The application's job is to abstract the complexity beneath that unit of work.

Because every category packages value differently, there is no universal AI application metric. Voice AI may begin with minutes, then move toward conversations resolved. Copilots may begin with seats, then add usage-based pricing as agentic work creates greater variation in cost and value.

The right question is not, "What is the AI pricing metric?" It is: "**What unit of value does the customer already understand, and can we measure it consistently?**"

2. Customers Want Legibility, Not Tokens

Customers will still ask about tokens. Usually, they are asking for one of two things: comparability or control.

First, buyers want a common benchmark. Tokens appear to let them compare a specialized application with a general-purpose model API or an internal build. But the comparison is usually false precision. Each application combines different models, data, tools, and levels of automation. A token flowing through one product does not produce the same work as a token flowing through another.

Second, buyers want to allocate spend. Finance and IT teams need to trace AI spend to a department, project, client, or invoice, often across a growing portfolio of applications. Requiring them to forecast tokens separately for every product recreates the complexity those products are meant to hide.

Both needs are reasonable. Nonetheless, token pricing often creates more friction than clarity.

A support leader can estimate how many conversations the company handles. It is much harder to predict context length, retrieval volume, retries, reasoning time, or output tokens. What should be a straightforward ROI calculation becomes a separate compute-forecasting exercise for every AI application the company deploys.

The better answer is to expose enough underlying usage detail to build trust without turning that usage into the commercial unit. Show customers what work was completed, where capacity went, and why certain tasks consumed more than others. Give finance and IT the reporting they need for budgets and chargebacks.

Transparency does not require the billing meter and the underlying cost meter to be the same.

In highly competitive or technically sophisticated markets, companies may still need to offer token pass-through, particularly for unusually expensive or volatile model usage. But this should be an explicit component of a hybrid model, not the default expression of the product's value.

3. Credits Should Map Work to Value, Not Hide Tokens

For the broad middle of the AI stack, credits can be an effective way to package variable work. But a credit is a currency, not a value metric. *What matters is what the credit buys*.

A weak credit system converts token counts into an opaque internal currency. It hides the meter without improving it.

A strong credit system maps to work the customer recognizes. It might use a few intuitive effort bands: a small bug fix costs less than a multi-file feature; summarizing one contract clause costs less than reviewing the full agreement; enriching one record costs less than running a multi-step account research workflow.

The best credit systems do three things:

- **Abstract infrastructure complexity.** Customers buy the work, not the ingredients.
- **Explain relative effort.** Simple work consumes a little; standard work consumes a predictable amount; complex work consumes more.
- **Create commercial flexibility.** One pool can cover several workloads, agents, or automations while procurement manages one contract.

The test is comprehension. Buyers usually know their workload before they know their compute load. If customers cannot understand the credit system in a few sentences, it is too complicated.

4. Credits Can Protect Margins

Credits do more than just make usage understandable. Designed well, they also protect your gross margins.

In traditional SaaS, an additional user often adds little marginal cost. In AI applications, every user can generate inference, retrieval, search, tool calls, third-party data, media generation, retries, and failed runs. Fast growth can hide a weak business if each new dollar of revenue is quickly paid back to model, cloud, or data providers.

The pricing system needs to separate two decisions:

1. **What is the work worth?** Customer value, willingness to pay, and competition determine the price of the credit pool
2. **What does the work cost to deliver?** Relative cost and complexity determine how many credits each task consumes

This separation enables the vendor to protect margin while preserving the margin upside from model routing, caching, prompt optimization,

better infrastructure, and a more diverse (and changing!) mix of proprietary and open-source models. In fact, if underlying model costs fall, the application may even be able to retain part of the benefit because customers are paying for work, not just reimbursing the company for compute.

Clay offers a helpful example and has often been ahead of the curve in how they think about pricing. In a 2026 pricing memo, the company wrote that it had mispriced credits in its Pro segment back in 2022 and operated that segment at a loss for years. Its new model separates Data Credits for third-party data from Actions for orchestration work. Clay keeps fixed pricing for models with predictable costs while passing through the actual token cost of more volatile and expensive reasoning models without a markup. In other words, Clay uses different meters for different layers: Actions for platform value and token pass-through for unpredictable model costs.[1]

5. Move to Outcomes When Value Is Clear

Credits are most useful when the product performs valuable work but the final business result is not cleanly attributable. Once the outcome becomes observable, attributable, and valuable enough to support a stable price, the meter should move again.

Under those conditions, price the outcome: a resolved support conversation, a qualified lead, a booked meeting, a processed claim, or a recovered dollar.

If the results are not attributable enough, price the unit of work through credits.

If the customer is buying raw model access, price tokens.

Many products will use hybrids: seats for access, credits for variable work, token pass-through for unusually expensive or unpredictable model calls, and outcome fees where attribution is clean. Multiple

meters are not the problem, but avoid using the wrong meter for the wrong layer.

Make Value Visible

Token pricing pulls the customer conversation toward a cost curve that keeps falling. This is a poor anchor for a product whose usefulness, reliability, and role in the workflow should keep rising.

The better path is to price at the highest layer of value you can reliably measure, attribute, and defend. Translate variable work into understandable units. Use credits to package those units when flexibility matters. Move toward outcomes as soon as customers can recognize and trust them.

---

1 https://www.clay.com/blog/clay-pricing-memo-internal