CHAPTER 7 / 4 MINUTE BRIEF

AI Services and Token Capacity

Read the full chapter → / Read this brief online →

Before any theory: the four ways to buy tokens

There is more than one way to buy the same token, and the differences matter to a capacity plan.

The most direct path is the provider's own API, usually priced by consumption: so much per million input tokens, so much per million output tokens, with separate treatment for caching and other features. Payment may be prepaid, invoiced, or governed by a contract; a usage-based price does not establish when the money changes hands.[1][2]

The second path is through a platform. Microsoft Foundry, Google's Gemini Enterprise Agent Platform (formerly Vertex AI) and OpenRouter provide model access under their own deployment, routing and billing arrangements.[3][4][5] Before comparing rates for a

[1] OpenRouter, FAQ, https://openrouter.ai/docs/faq, and "API Credit & Rate Limits," https://openrouter.ai/docs/api_reference/limits, checked October 4, 2026; Stripe, "Stripe agrees to acquire OpenRouter to help businesses optimize token routing and usage," August 19, 2026, https://stripe.com/newsroom/news/stripe-agrees-to-acquire-openrouter. Product terms and transaction announcement; no transaction price is inferred.

[2] OpenAI, "GPT-6 Astra: A new generation of intelligence," September 3, 2026, https://openai.com/index/gpt-6-astra/. Launch Standard rates $10/$50 per million input/output tokens; Fast mode twice the Standard price and advertised at up to twice the speed. On OpenRouter, which reports weekly route traffic, Astra was among the most-used routes through early October 2026 and processed about 98.6 billion tokens on October 9, 2026, moving down the popularity list.

[3] Microsoft, "Understanding deployment types in Microsoft Foundry Models," https://learn.microsoft.com/en-us/azure/foundry/foundry-models/concepts/deployment-types, checked October 3, 2026. Deployment categories, supported processing boundaries and model-specific availability; managed-compute deployments are a separate option.

[4] Google Cloud, "Use Provisioned Throughput," https://docs.cloud.google.com/gemini-enterprise-agent-platform/models/provisioned-throughput/use-provisioned-throughput, checked October 4, 2026. Default spillover, the dedicated-only 429, enforcement windows and spillover hidden by one-minute averages. Renamed from Vertex AI on April 22, 2026, per the release notes: https://docs.cloud.google.com/gemini-enterprise-agent-platform/release-notes.

[5] OpenRouter, FAQ, https://openrouter.ai/docs/faq, and "API Credit & Rate Limits," https://openrouter.ai/docs/api_reference/limits, checked October 4, 2026; Stripe, "Stripe agrees to acquire OpenRouter to help businesses optimize token routing and usage," August 19, 2026,

named model, establish which version and features each offering supports, who supplies the inference, where processing may occur and which agreement pays for it. An existing cloud agreement can make this path operationally convenient; the workload still needs its own comparison.

The third path is a negotiated agreement. Large consumers may negotiate discounts, usage minimums, and model-specific terms. Negotiation does not guarantee a discount or held supply, and the price of the same work can depend on which agreement it flows through.

The fourth path is provisioned capacity, because a workload can hit a throughput limit while its monthly spending is modest. Providers sell arrangements that reserve a defined level of processing capability, subject to the model, region, workload and service terms, and the charge continues while that capacity is idle. A spending commitment alone reserves no capacity.[67]

Explore this section in the full chapter →

---

https://stripe.com/newsroom/news/stripe-agrees-to-acquire-openrouter. Product terms and transaction announcement; no transaction price is inferred.

[6]Microsoft, "Provisioned throughput billing and cost management," https://learn.microsoft.com/en-us/azure/foundry/openai/concepts/provisioned-throughput-billing, checked October 4, 2026. Deployed throughput incurs hourly charges whether or not it is used; a reservation is a discount on those charges and does not guarantee capacity.

[7]Google Cloud, "Use Provisioned Throughput," https://docs.cloud.google.com/gemini-enterprise-agent-platform/models/provisioned-throughput/use-provisioned-throughput, checked October 4, 2026. Default spillover, the dedicated-only 429, enforcement windows and spillover hidden by one-minute averages. Renamed from Vertex AI on April 22, 2026, per the release notes: https://docs.cloud.google.com/gemini-enterprise-agent-platform/release-notes.

Token capacity is a different kind of capacity

What limits an AI service is a combination of three things that no single graph shows:

- **Scarcity.** Rate limits cap requests, tokens or concurrency within a window, and their scope varies by model and account. On shared service a limit is permission to send rather than capacity held for the sender, so a request within it can still be refused when the provider is busy. Queuing and retries are application decisions with latency and cost consequences.[8]
- **Pricing.** Per-token charges, subscriptions and provisioned capacity respond differently to demand. Compare the full cost of the service, including unused commitments, before drawing a conclusion from its token rate.
- **Budgets.** A prepaid balance can run out; a commitment can charge after demand disappears. Alerts, hard caps and advisory limits behave differently, so the plan needs to say which one is in force and what happens when it is reached.

Explore this section in the full chapter →

[8] OpenAI, "Rate limits," https://developers.openai.com/api/docs/guides/rate-limits, and "Error codes," https://developers.openai.com/api/docs/guides/error-codes; Google Cloud, "Standard PayGo," https://docs.cloud.google.com/gemini-enterprise-agent-platform/models/deploy/standard-paygo, and "Priority PayGo," https://docs.cloud.google.com/gemini-enterprise-agent-platform/models/deploy/priority-paygo; Microsoft, "Enable priority processing for Microsoft Foundry Models," https://learn.microsoft.com/en-us/azure/foundry/openai/concepts/priority-processing; AWS, "Service tiers for optimizing performance and cost," https://docs.aws.amazon.com/bedrock/latest/userguide/service-tiers-inference.html; checked October 3, 2026. Documented ramp, retry, spend-limit and downgrade behavior; shared pool sizes are not published.

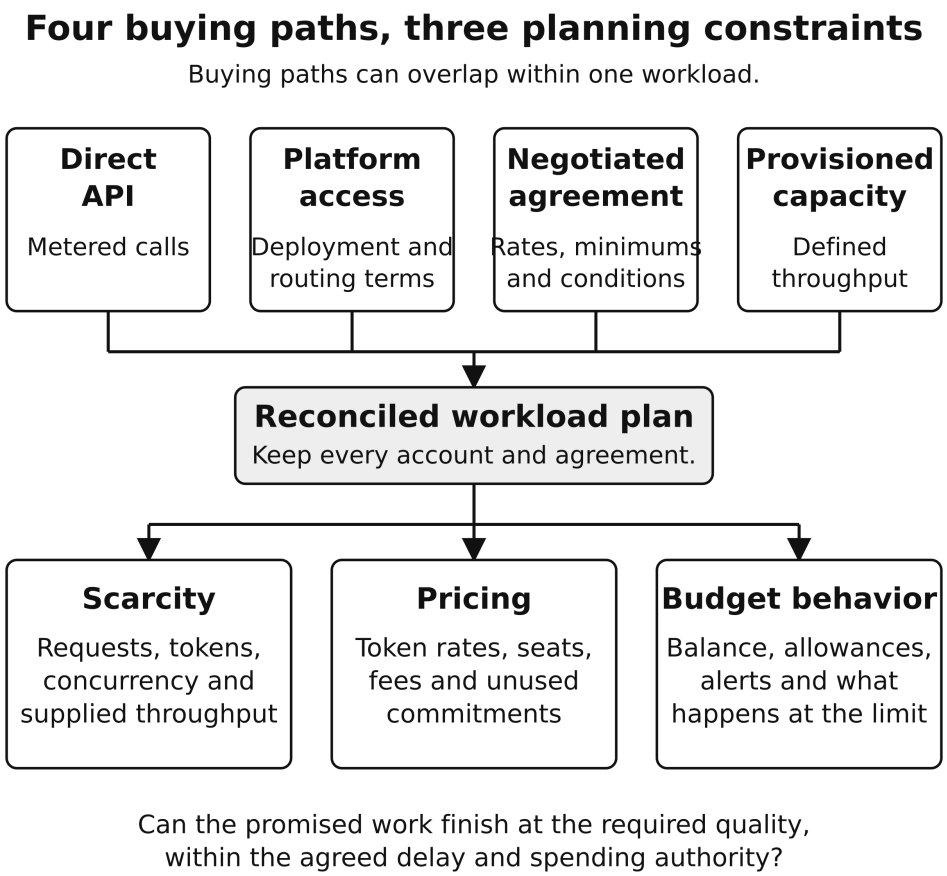


Diagram 7.1. Four buying paths feed one workload plan. Scarcity, pricing and budget behavior remain separate checks even when the organization buys through several paths.

Explore this section in the full chapter →

The model product list and how to read it

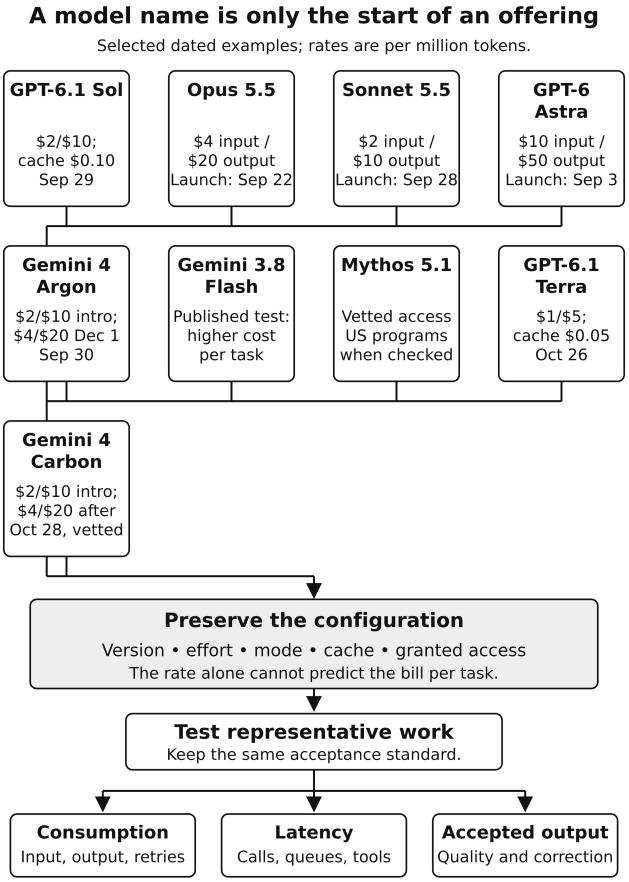


Diagram 7.2. A dated model offering feeds a configuration record, then a test on representative work. The selected examples illustrate different prices, consumption and access conditions; they are not a performance ranking.

Explore this section in the full chapter →

Sizing supply for the busy interval

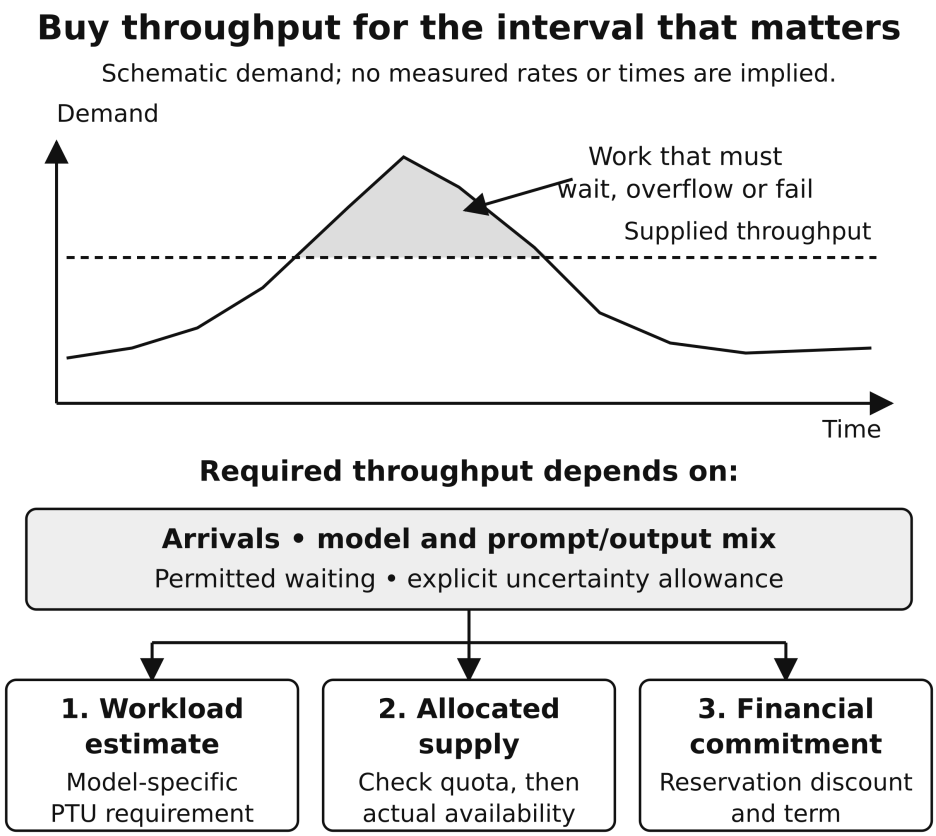


Diagram 7.3. A schematic busy interval exceeds supplied throughput. The amount to buy depends on arrivals, the model and prompt/output mix, permitted waiting and uncertainty; the workload estimate, allocated supply and financial commitment remain separate records.

Explore this section in the full chapter →

Measure the efficiency of the models, not just the spend

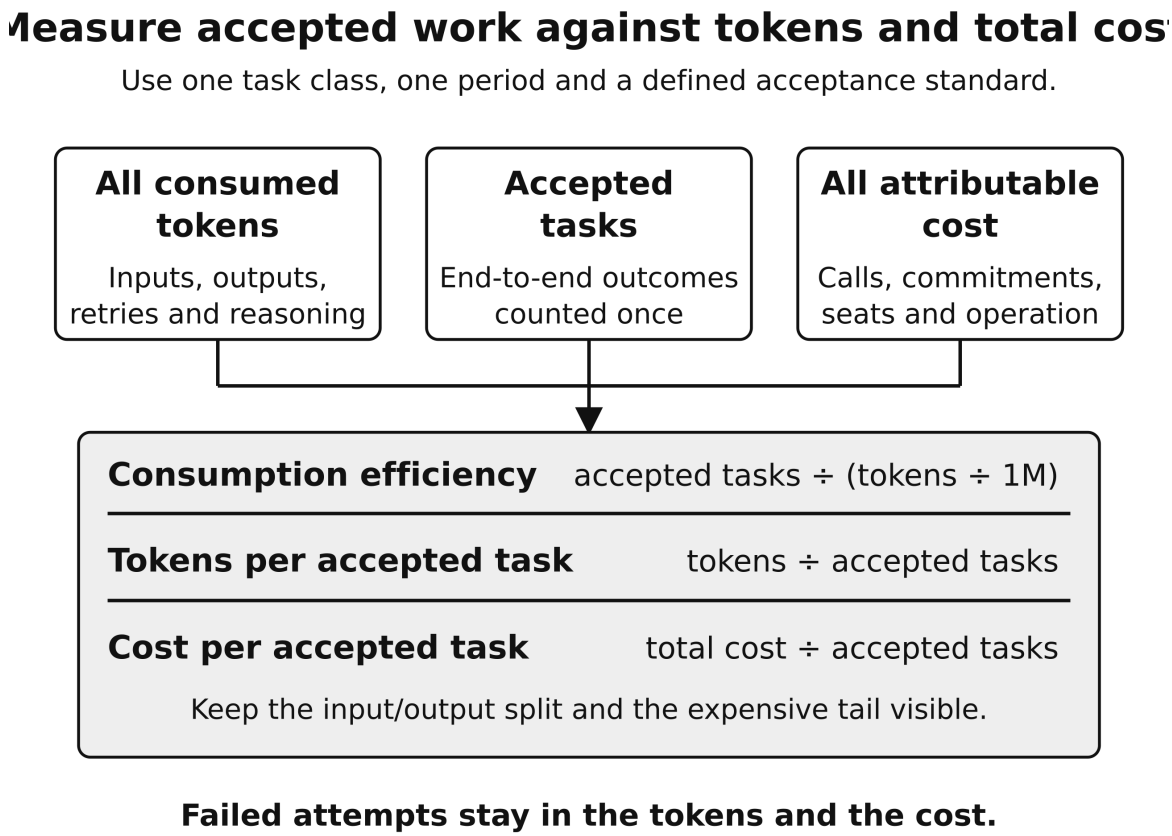


Diagram 7.4. Three measures share the same task class, time window and acceptance standard. Accepted tasks per million tokens and tokens per accepted task are reciprocals, read as efficiency and as intensity; total cost per accepted task adds price and fixed charges. Failed attempts remain in consumption and cost.

Explore this section in the full chapter →

Consumption efficiency counts accepted tasks per million tokens:

$$\frac{\text{accepted tasks}}{\text{tokens consumed}/1{,}000{,}000}.$$

Record it per model and per task class, measured monthly.

Accepted ties the count to the business's outcome standard; preserve whether the result needed correction and what that correction cost. *Per task class* keeps unlike work from being averaged together. Model-level comparisons are useful when a model performs the complete task. For a workflow spanning several models, report the end-to-end outcome once and use step-level costs and acceptance checks to explain each model's contribution. Crediting every model with the same outcome and then summing would count the work repeatedly.

Explore this section in the full chapter →

Recommended practices

Size supply for the work that must finish on time. Use measured demand over the relevant interval, the permitted delay and an explicit uncertainty allowance. Decide which work may queue, which may use overflow and which needs reserved throughput, and test any reservation on price as well as supply.

Run the efficiency measurement every month, with an explanation. Report accepted tasks and token consumption by model and task class, alongside total cost and retries. Investigate changes rather than assuming they represent improvement or waste.

Explore this section in the full chapter →

Keep the budget policy usable across models and providers. State who may spend, what is counted and what happens at the limit separately from the feature implementing it. A provider change may require different controls: a balance, gateway quota, spend cap and commitment make different promises. Re-test their behavior rather than transferring the old configuration by name.

Explore this section in the full chapter →