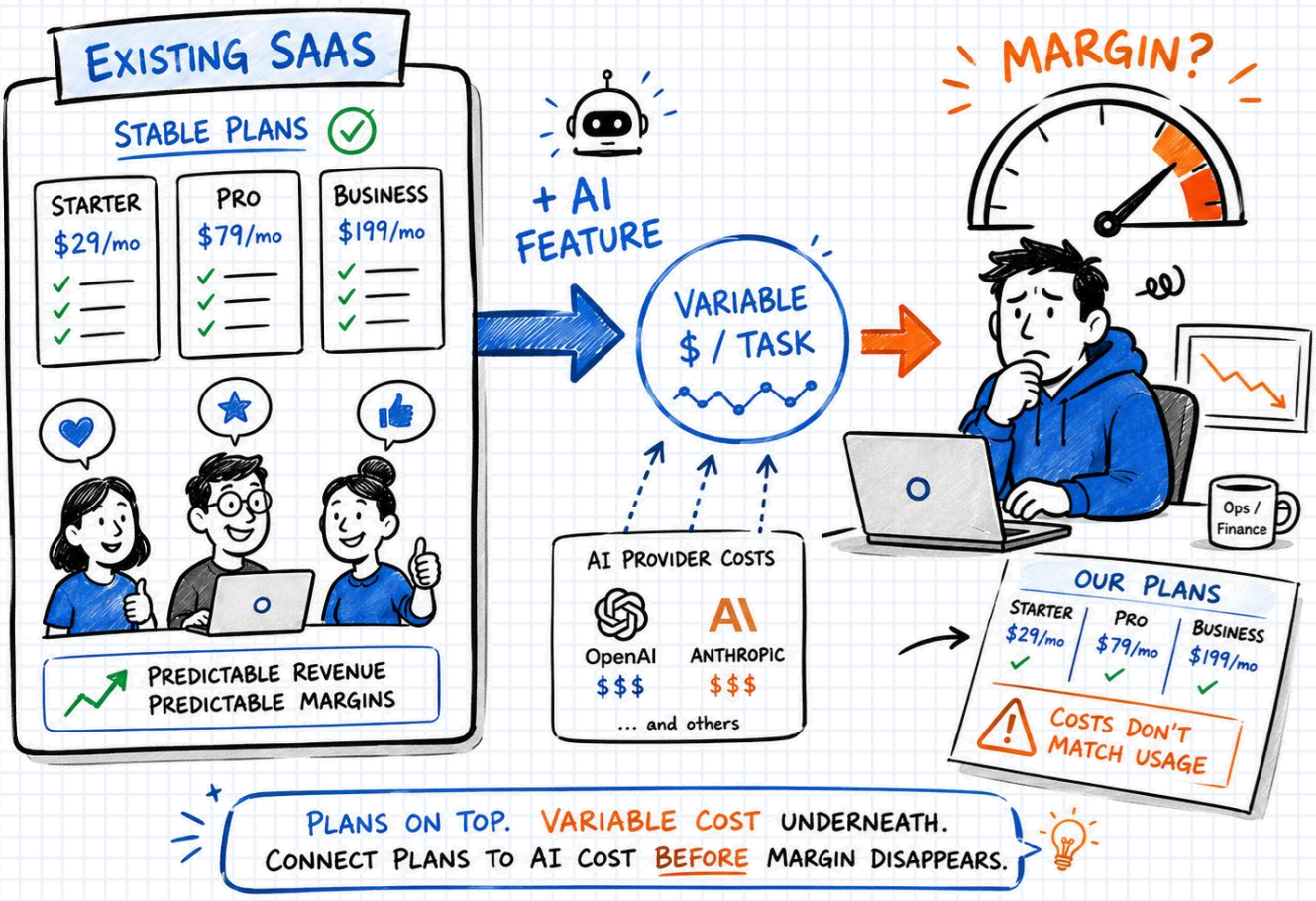
USAGETAP FIELD GUIDE 02

Adding AI Without Breaking Your SaaS Economics

10 small decisions that maintain margins when adding variable-cost AI features

For product, engineering, finance, and commercial teams adding LLM or agentic features to an existing application.



Keep the product moving. Make the new variable cost deliberate.

This is not a finance transformation program and it is not an argument against bundling AI into an existing plan. It is a short sequence of field checks for teams whose product economics changed the moment an AI feature reached real customers.

The spine: cost per useful task

A customer asks your application to do something useful. The implementation may use one model call or fifty. Measure the complete task first, then connect it to account revenue, included usage, guardrails, optimization, and pricing.

Existing products have extra constraints

- You already have plans, contracts, entitlements, renewal cycles, and margin expectations.
- Customers may assume a new AI feature is simply part of the subscription they already bought.
- One account can consume far more AI than another while paying the same seat price.
- Pricing changes can create customer friction, so instrumentation should arrive before the correction.
- The objective is not to meter everything customers can see. Meter enough underneath to protect a simple commercial experience.

A useful test

If you can answer all ten field questions with account-level numbers, you can decide what to include, what to limit, what to optimize, and when a usage or outcome-based component is actually justified.

Rough, explainable economics beat precise numbers that nobody can connect to a customer or task.

Mark the moment your unit economics changed

What did this feature add to the cost of one useful customer task?

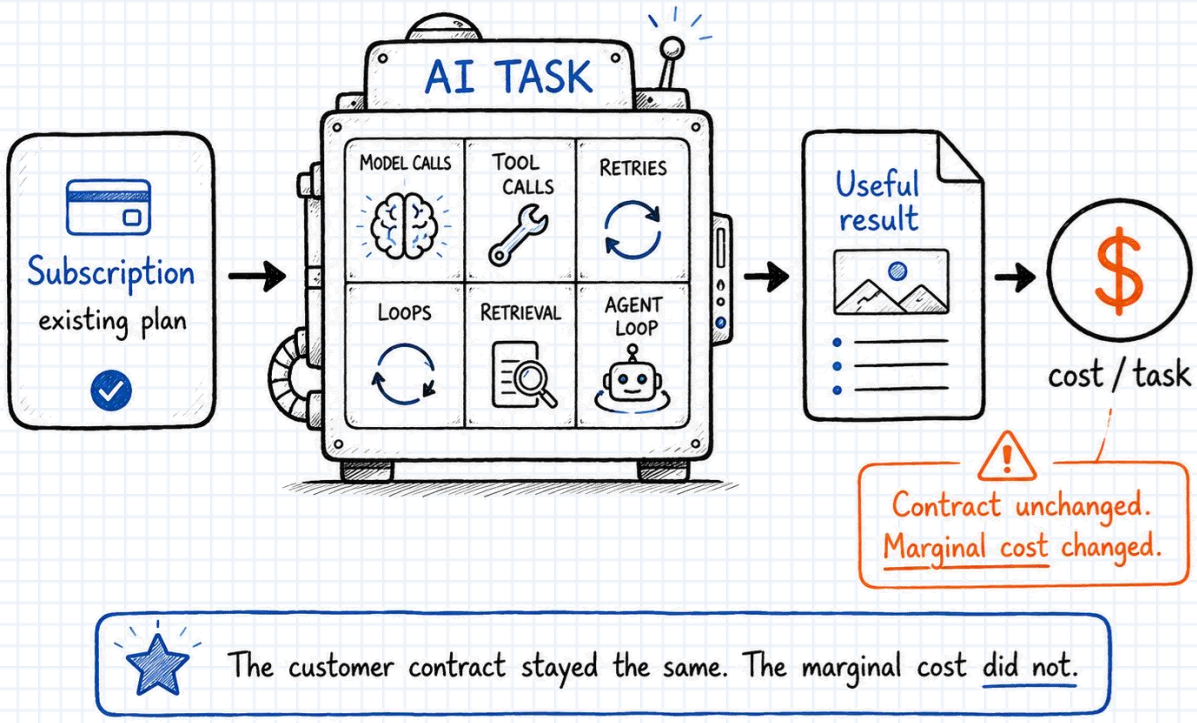
Your existing product already has a commercial model. Seats, subscriptions, transactions, storage, or another unit may have predictable economics. Adding an AI feature introduces a new cost that can rise every time the customer asks the product to do more.

Do not start with tokens. Pick one useful AI-enabled task and trace it end to end: model calls, embeddings, retrieval, tools, retries, vision, external inference, and failed attempts that are part of normal behavior. The result is the variable AI cost of completing that task.

This number becomes the bridge between engineering behavior and the plan the customer already pays for.

FIELD RULE

Cost per useful task = every variable AI cost required to complete one customer outcome inside the existing product.



FIELD 01 / 10 - PUT IT TO WORK

5-MINUTE FIELD EXERCISE: Trace one real AI feature task

1. Choose one customer-visible AI task that is already shipped or close to release.
2. Trace it from user action to completed outcome.
3. Add model, tool, retrieval, retry, and other direct variable AI costs.
4. Repeat with 5-10 production-like examples.
5. Write down the typical cost and the costly failure path.

Task: _____

Typical cost per completed task: $_____

Costly failure / retry path: $_____

Before you move on: you can name one AI task and state what it costs to complete, not just what one model call costs.

DO NOW / NOT YET

DO NOW	NOT YET
[ ] Measure the whole task.	Allocate salaries and fixed cloud overhead.
[ ] Use production-like inputs and realistic context.	Optimize every call before you know which tasks matter.
[ ] Distinguish cost per attempt from cost per successful task.	Use provider token totals as the customer-facing unit.

FIELD NOTES

- If the feature sometimes fails, cost per successful task is often the more useful commercial number.
- Keep the task name stable even if providers or models change.

Build the AI feature P&L

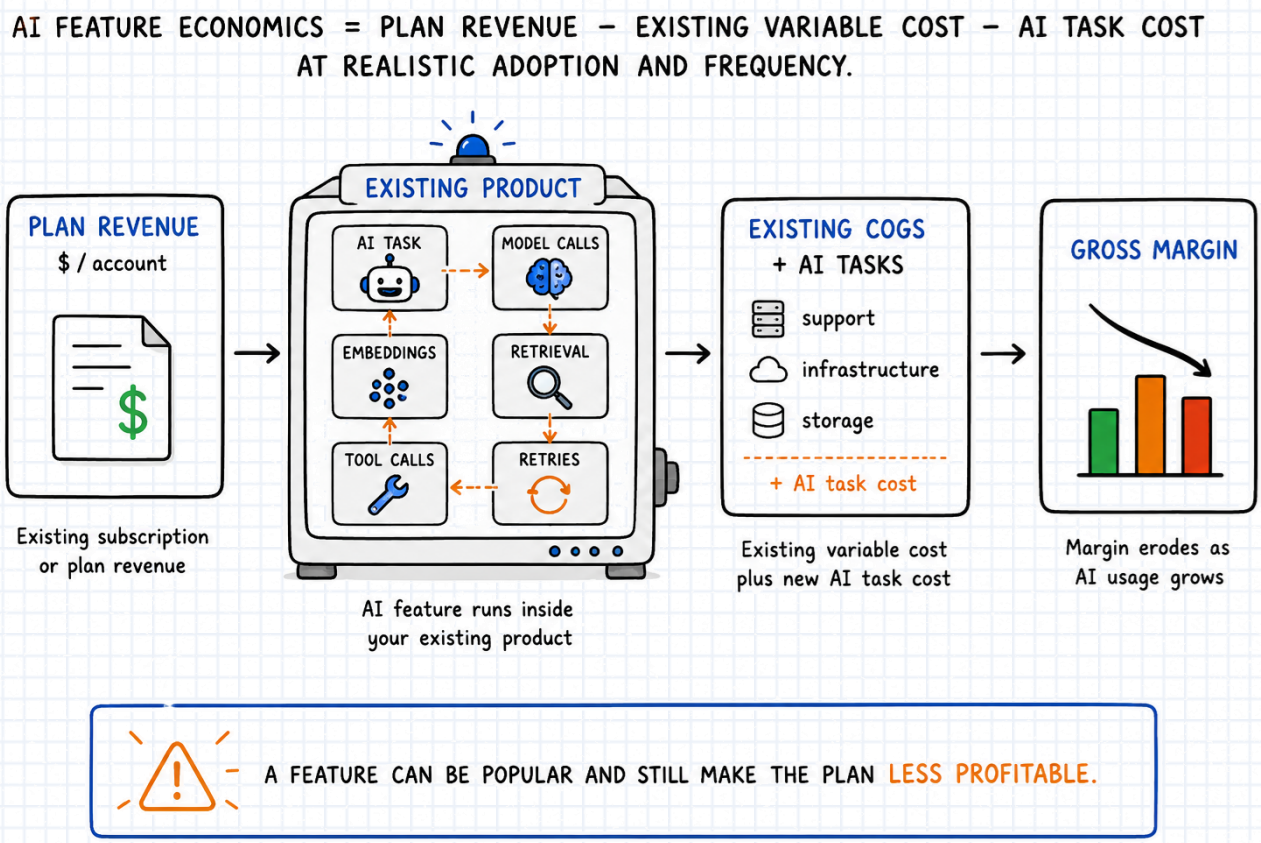
Is the feature improving the product while quietly eroding the plan margin?

A provider bill can be small in absolute dollars and still be material to a plan with tight margins. Existing businesses need the AI cost expressed against the revenue and gross margin of the account or plan that consumes it.

Start with a simple contribution view: account or plan revenue, existing variable service cost, new AI task cost, and resulting margin. Then model adoption. A feature used by 10% of seats has different economics from one that becomes the default workflow.

Run an ordinary month and a high-adoption month. You are looking for the point where a popular feature changes the economics of a plan.

FIELD RULE



FIELD 02 / 10 - PUT IT TO WORK

5-MINUTE FIELD EXERCISE: Run the before-and-after margin check

1. Pick one existing plan and a representative account on it.
2. Record monthly revenue and current variable service cost.
3. Add expected AI tasks per account multiplied by cost per task.
4. Run normal, 3x-adoption, and heavy-use scenarios.
5. Circle the first scenario that makes the plan margin uncomfortable.

Plan / account revenue: $_____

AI cost at normal adoption: $_____

AI cost at 3x adoption: $_____

Margin floor we do not want to cross: _____%

Before you move on: you know how much AI adoption the existing plan can absorb before commercial action is needed.

DO NOW / NOT YET

DO NOW	NOT YET
[ ] Express AI cost as a percentage of plan revenue or gross profit.	Wait for the provider invoice to become large.
[ ] Model adoption and task frequency separately.	Assume AI cost is “R&D” after the feature is customer-facing.
[ ] Show the scenario to product and finance in the same units.	Use one blended company average for every plan.

FIELD NOTES

- The first purpose of the model is to reveal sensitivity, not to forecast perfectly.
- A plan with high gross margin can intentionally subsidize AI; make the subsidy explicit.

IN THE WILD Notion is a useful public example of why “included AI” still needs resource controls. In May 2025 it announced all-in-one Business and Enterprise plans with unlimited Notion AI; current help documents apply six-hour and monthly allowances to certain AI features. [6]

Attribute AI usage to customers, plans, and tasks

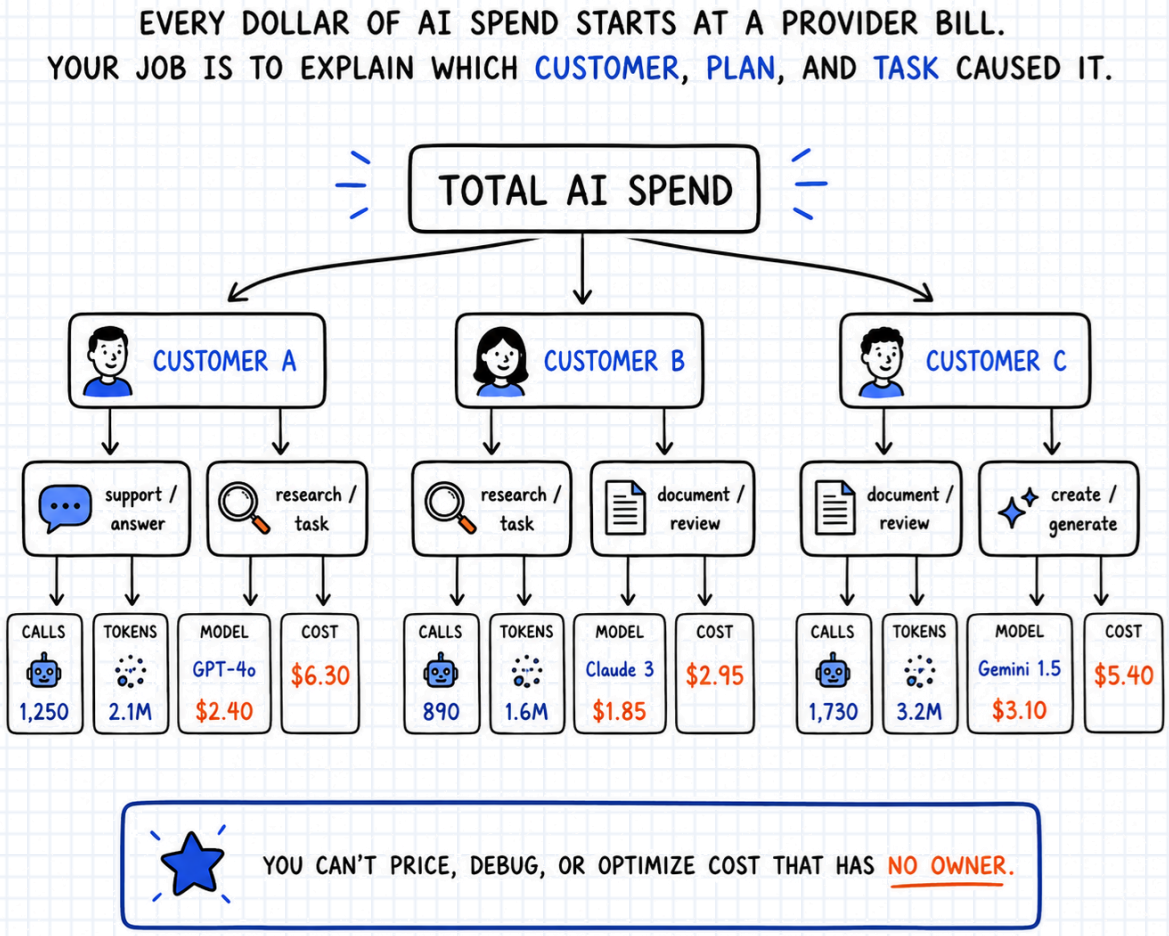
Can you explain yesterday's AI spend in product language?

Provider dashboards answer "what did we spend?" They usually cannot answer "which customer, which plan, which feature, and what useful task caused it?" Those are the questions needed for margin decisions.

Attach a small metadata envelope to each AI event or completed task: account, plan, feature/task, provider/model, usage, cost, outcome, and timestamp. Add user-level attribution only when it is useful and appropriate.

You do not have to retain prompt content to get commercial visibility. Metadata is enough to identify cost concentration, adoption, and unexpected workflows.

FIELD RULE



FIELD 03 / 10 - PUT IT TO WORK

5-MINUTE FIELD EXERCISE: Create the minimum economics event

1. List the customer/account and plan identifiers you already have.
2. Choose one stable task or feature identifier.
3. Record provider/model, input/output usage, direct cost, outcome, and timestamp.
4. Run one report grouped by account and one grouped by task.
5. Explain the top three cost drivers without opening the provider dashboard.

Account key: _____

Plan key: _____

Task key: _____

Top AI cost-driving account this week: _____

Before you move on: you can answer "which customer, which plan, which task, how much?" for meaningful AI spend.

DO NOW / NOT YET

DO NOW	NOT YET
[ ] Use identifiers already present in your product.	Store every prompt "for observability."
[ ] Separate production, trial, and internal traffic.	Build a new data warehouse before an event log works.
[ ] Keep task names stable across model changes.	Treat provider/model as a substitute for product attribution.

FIELD NOTES

- Attribution is the foundation for limits, support explanations, forecasting, and usage-based billing.
- Metadata-first instrumentation also reduces privacy and retention pressure.

Find the heavy tail and hidden cross-subsidy

Which customers are using far more AI than their plan economics can support?

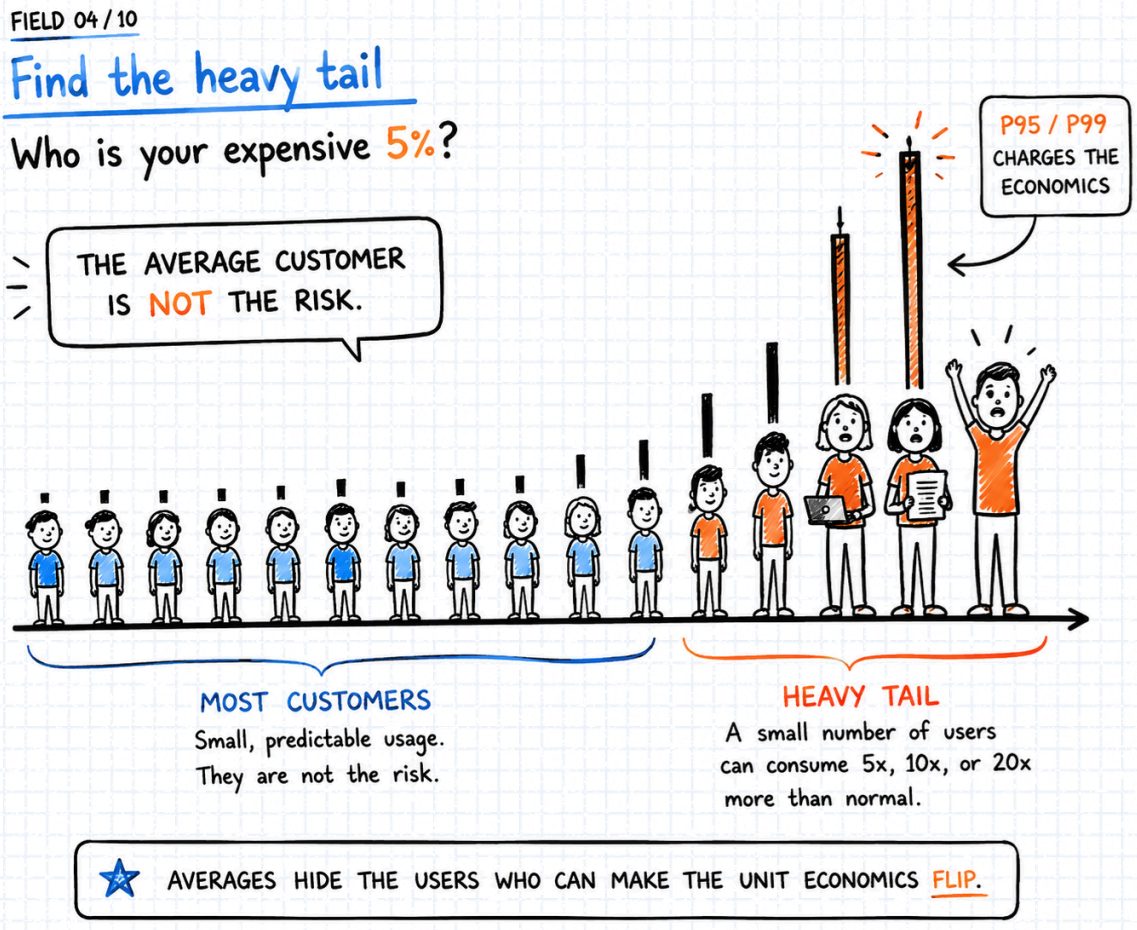
AI usage rarely follows the neat average implied by a seat plan. A small number of accounts may upload larger documents, run deeper research, trigger more agent work, or simply discover the feature first.

Rank accounts by AI cost and compare that cost with plan revenue. Look at the median, P95/P99, and the most expensive accounts. The goal is not to punish power users; it is to know whether the commercial model makes their success economically sustainable.

Cross-subsidy can be a deliberate acquisition or retention choice. It becomes dangerous when nobody can see it.

FIELD RULE

Know the typical account, the expensive tail, and AI cost as a share of each account's revenue or gross profit.



FIELD 04 / 10 - PUT IT TO WORK

5-MINUTE FIELD EXERCISE: Run an account-level tail check

1. Rank accounts by total AI cost for the last useful period.
2. Record median, P95, P99, and maximum AI cost.
3. For the top five accounts, divide AI cost by account revenue.
4. Identify the feature or task driving each expensive account.
5. Decide whether the pattern is healthy power usage, abuse, a product bug, or a packaging mismatch.

Median AI cost / account: $_____

P95 AI cost / account: $_____

Highest AI cost as % of that account revenue: _____%

Task driving the tail: _____

Before you move on: you know which accounts are being cross-subsidized and why.

DO NOW / NOT YET

DO NOW	NOT YET
[ ] Inspect revenue and usage together.	Set every account limit from the global average.
[ ] Talk to high-usage customers before assuming the usage is bad.	Design the whole price book around one outlier.
[ ] Separate valuable power use from loops, retries, or misuse.	Assume high usage automatically means low margin.

FIELD NOTES

- A high-usage customer can be your best customer if the plan captures enough of the value.
- Tail behavior is often the earliest evidence that an included allowance or usage add-on is needed.

IN THE WILD GitHub's Copilot packaging shows how measurement can evolve as premium AI usage broadens. Premium-request allowances and overage controls were enforced in 2025; in June 2026 GitHub moved new billing to AI Credits whose cost depends on model and token usage. [4]

Choose the commercial unit customers can understand

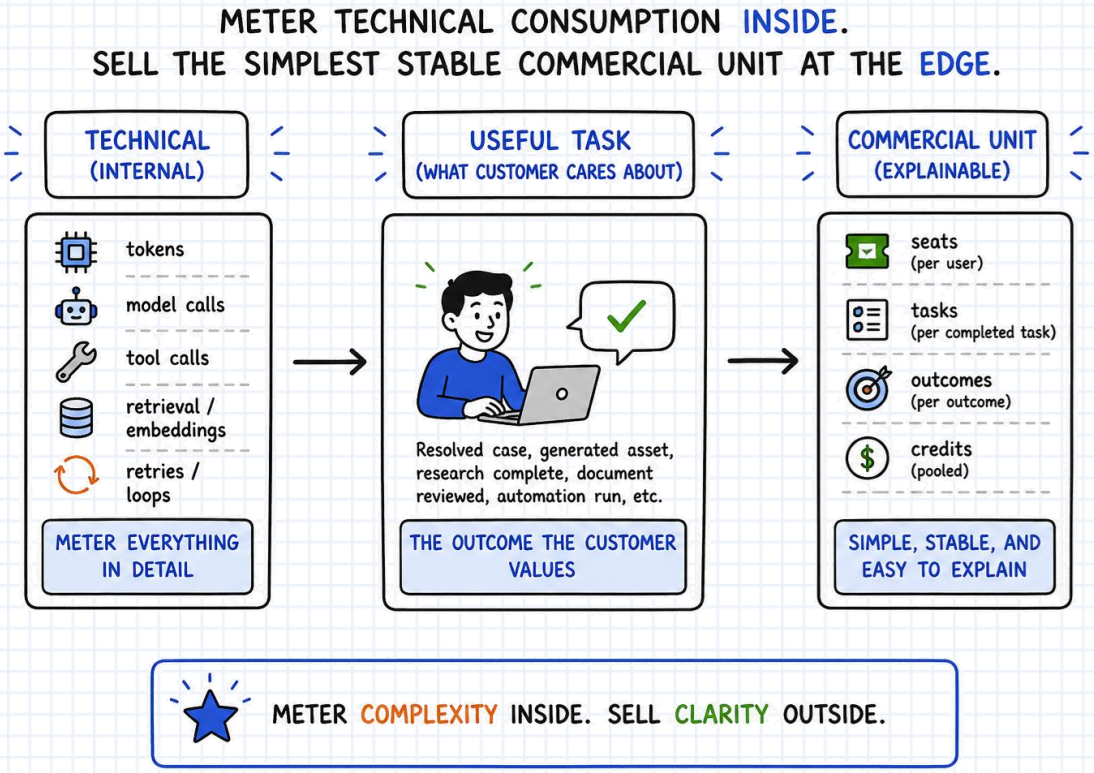
What are you actually selling: seats, tasks, outcomes, credits, or a hybrid?

Tokens, context windows, tool calls, and model multipliers are implementation details. Customers usually care about the useful thing the product does: a resolved case, generated asset, completed research task, reviewed document, or automation run.

Choose a commercial unit close enough to customer value that it is explainable, but stable enough to meter consistently. Seats may remain the base subscription while AI uses an included allowance, per-task charge, outcome charge, or pooled credits.

Do not expose technical complexity unless the buyer genuinely benefits from it. Meter technical units underneath; translate them into a simple commercial unit at the boundary.

FIELD RULE



FIELD 05 / 10 - PUT IT TO WORK

5-MINUTE FIELD EXERCISE: Translate technical usage into a value unit

1. Write the customer outcome for one AI feature in plain language.
2. List the technical units that create it: calls, tokens, steps, models, tools.
3. Choose one candidate commercial unit: task, outcome, credit, included access, or hybrid.
4. Test whether a customer could predict what increases their bill or consumes their allowance.
5. Check whether the unit still works if you switch models or providers.

Customer value event: _____

Technical units underneath: _____

Candidate commercial unit: _____

Before you move on: you can explain the AI usage unit without mentioning model tokens unless the customer asks.

DO NOW / NOT YET

DO NOW	NOT YET
[ ] Keep the customer-facing unit stable across provider changes. [ ] Use outcome pricing only when the outcome can be measured consistently. [ ] Preserve a simple base plan where that helps buying.	Publish a rate card for every provider token type. Copy a competitor's unit without your own workflow economics. Call something "unlimited" before defining fair-use behavior.

FIELD NOTES

- A credit can be useful when tasks have meaningfully different costs, but it adds abstraction.
- Outcome pricing aligns with value but requires a clear, auditable definition of success

IN THE WILD Intercom's current Fin AI Agent pricing starts at $0.99 per outcome and charges at most one outcome per conversation. Zendesk now measures AI-agent billing in automated resolutions, replacing older bot MAU / Answer Bot pricing. Both anchor the commercial unit to a completed customer-service outcome. [1][2]

Design included usage before you design overage

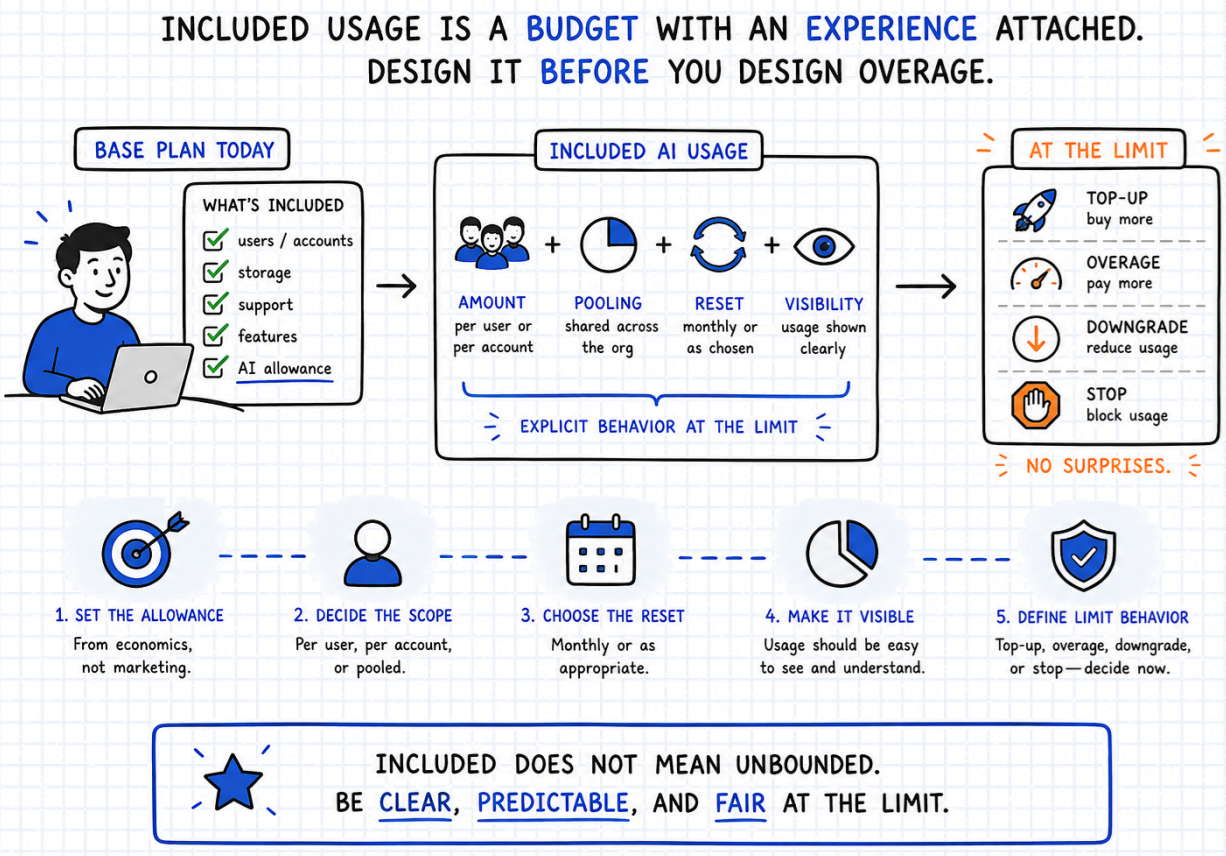
How much AI should the existing plan include, and what happens after that?

For an established product, included usage is often the least disruptive path. It keeps the base plan familiar while placing a deliberate budget around variable AI consumption.

Set the allowance from economics, not from a round marketing number. Decide whether it is per user, per account, or pooled across the organization; whether it resets monthly; whether unused allowance rolls over; and whether overage is automatic, opt-in, prepaid, or blocked.

The most important design choice is the exhausted-allowance experience. Surprise charges and surprise lockouts are both avoidable.

FIELD RULE



FIELD 06 / 10 - PUT IT TO WORK

5-MINUTE FIELD EXERCISE: Write the allowance contract

1. Choose an AI cost budget that the plan can absorb at target margin.
2. Convert it into the customer-facing unit from Field 05.
3. Choose per-seat, per-account, or pooled allocation.
4. Write reset, rollover, top-up/overage, and limit behavior.
5. Run the heavy-tail account from Field 04 through the allowance.

AI budget included in plan: $_____

Customer-facing allowance: _____

Pool: per seat / per account / organization

At the limit we will: _____

Before you move on: you can state exactly what "AI included" means operationally and commercially.

DO NOW / NOT YET

DO NOW	NOT YET
[ ] Show allowance consumption before the limit is reached.	Use unlimited as a substitute for an allowance policy.
[ ] Give admins a budget or overage control when appropriate.	Make overage possible without visibility.
[ ] Test the exhausted-allowance experience with a real workflow.	Hide the reset or rollover rules in legal text.

FIELD NOTES

- Pooled allowances often fit enterprise accounts better than rigid per-seat caps.
- A prepaid usage pack can improve predictability when customers dislike open-ended overage.

IN THE WILD Adobe uses monthly generative credits across Creative Cloud and Firefly, with premium features consuming credits and add-on packs for more capacity. Atlassian's Rovo uses pooled monthly credits tied to plan and seat counts, with complexity-based consumption. [5][7]

Put guardrails before the invoice

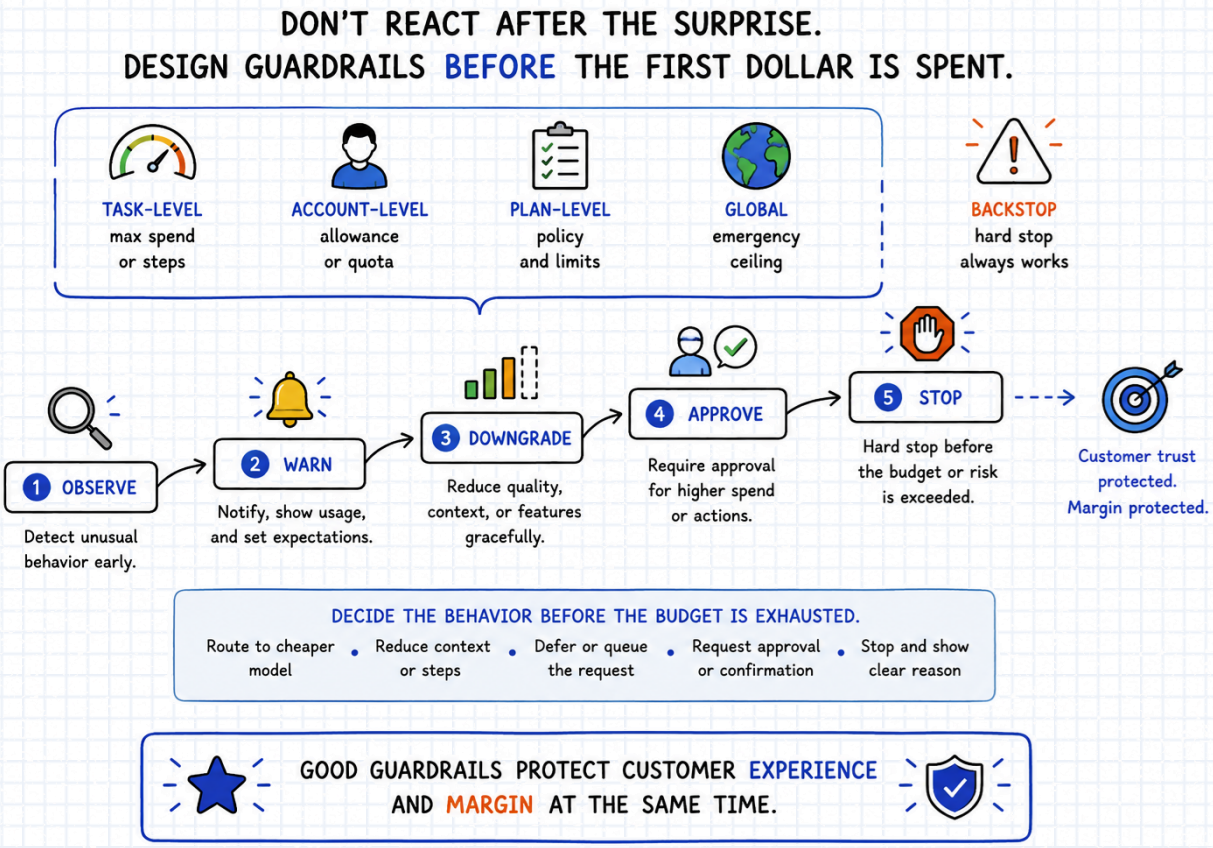
What will the product do before one customer, task, or agent creates an ugly surprise?

A financial control is also a product behavior. When a customer approaches a budget, the application can warn, route to a cheaper model, reduce context, queue work, request approval, or stop. Those choices should be made before the first incident.

Use more than one boundary: task-level maximum spend or steps, account-level allowance, plan-level policy, and a global emergency ceiling. The closer the guardrail is to the runaway workflow, the more graceful the response can be.

Provider budgets are useful backstops, but they are too coarse to replace product-level controls.

FIELD RULE



FIELD 07 / 10 - PUT IT TO WORK

5-MINUTE FIELD EXERCISE: Write the guardrail ladder

1. Pick the most expensive customer-facing AI task.
2. Set a task cost or step ceiling and an account-period budget.
3. Define warning, downgrade/throttle, approval, and stop behavior.
4. Add one product-wide emergency ceiling.
5. Force the limit in a test environment and observe the user experience.

Task budget: $_____ or _____ steps

Account budget / allowance: _____

Global emergency ceiling: $_____

Graceful fallback: _____

Before you move on: the team knows exactly what happens before spend becomes unbounded.

DO NOW / NOT YET

DO NOW	NOT YET
[ ] Make limits visible and explainable.	Rely on a human noticing a provider invoice.
[ ] Prefer graceful degradation for paid production workflows.	Assume provider rate limits are a cost-control strategy.
[ ] Log the reason when a task is stopped or downgraded.	Hide repeated agent loops behind automatic retries.

FIELD NOTES

- A hard stop may be fine for an experiment and unacceptable for a critical enterprise workflow.
- Budget exhaustion is useful telemetry: it can reveal product defects, pricing mismatch, or genuine customer demand.

Audit the context before you buy more inference

How much of every AI request is context the task does not actually need?

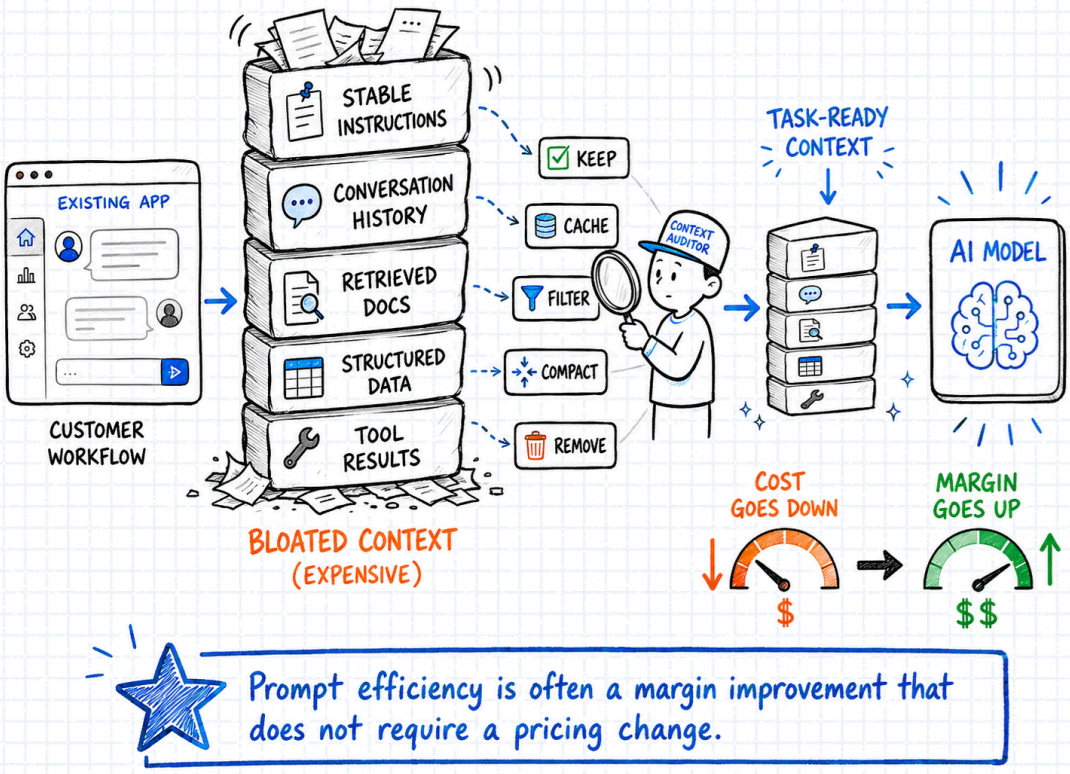
Existing applications accumulate context. Stable system instructions grow, customer history is replayed, retrieval returns too much, structured data is verbose, and tool results are copied back in full. Input can become the largest part of task cost and latency.

Audit the request as layers. For each layer ask whether it must be sent every time, whether it can be cached, filtered, summarized, compacted, fetched later, or removed. Protect IDs, numbers, constraints, code, and other content that cannot safely change.

Change one layer at a time and evaluate the completed task. The objective is not the smallest prompt; it is the smallest context that preserves the outcome.

FIELD RULE

Optimize context at the task level: keep what changes the outcome, cache or compact what repeats, and remove what does neither.



FIELD 08 / 10 - PUT IT TO WORK

5-MINUTE FIELD EXERCISE: Run a one-task context audit

1. Take one representative task and list every context layer sent to the model.
2. Estimate which layers dominate input size.
3. Mark each layer: keep, cache, filter, summarize, compact, fetch later, or remove.
4. Change the safest high-volume layer first.
5. Retest the same task examples for quality, latency, and cost.

Largest context layer: _____

First safe reduction: _____

Task cost before / after: $_____ / $_____

Input size before / after: _____ / _____

Before you move on: you have reduced one meaningful context layer without losing the task outcome.

DO NOW / NOT YET

DO NOW	NOT YET
[ ] Preserve stable prompt prefixes when provider caching rewards them. [ ] Filter retrieval to the smallest useful evidence set. [ ] Measure savings and quality on complete tasks.	Rewrite the entire system prompt in one experiment. Assume all tokens are equally disposable. Use lossy compression without an evaluation and fallback path.

FIELD NOTES

- Structured data, HTML, transcripts, tool results, and long histories are common places to look first.
- Prompt audits are product hygiene: repeat them as workflows and agents accumulate context.

Right-size the model and the agent workflow

Are you paying premium-model economics for work that does not need them?

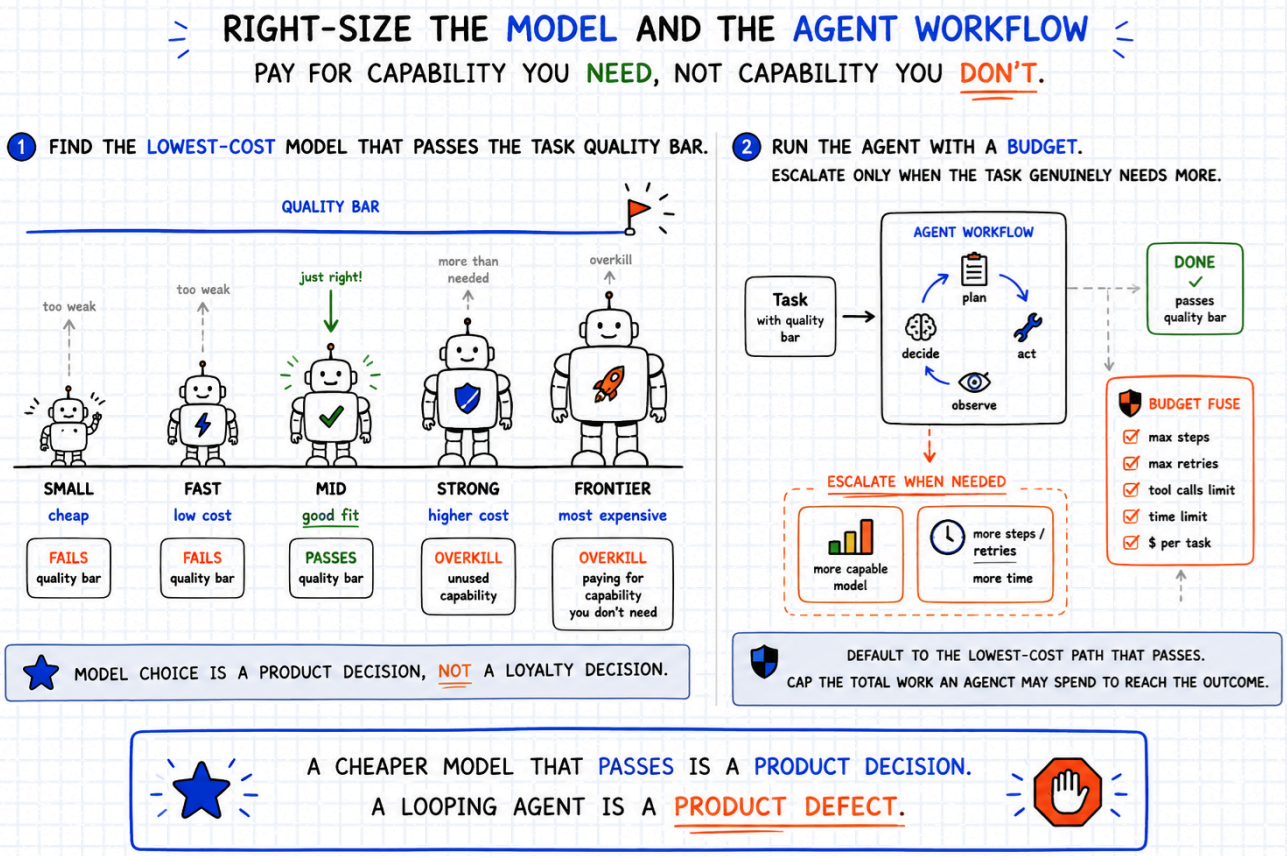
Model choice should be a task decision, not a company-wide preference. Define the quality bar for a representative task, then find the lowest-cost model that reliably clears it. Escalate only when the task genuinely needs more capability.

Agentic workflows need an additional budget: maximum steps, retries, tool calls, wall-clock time, and dollars per completed task. A single user request can fan out into many inference calls without the user seeing that multiplier.

Also test caching, batch execution, and asynchronous paths where latency is not part of the value proposition.

FIELD RULE

Default to the lowest-cost path that passes the task quality bar; cap the total work an agent may spend to reach the outcome.



FIELD 09 / 10 - PUT IT TO WORK

5-MINUTE FIELD EXERCISE: Benchmark and budget one workflow

1. Collect 10-30 representative examples of one AI task.
2. Define the quality threshold before comparing models.
3. Run the current model and at least one lower-cost alternative.
4. For an agent task, force one tool failure and inspect retries/steps.
5. Set the default model, escalation rule, and maximum agent budget.

Quality bar: _____

Lowest-cost passing model: _____

Task cost before / after: $_____ / $_____

Agent max steps / retries: _____ / _____

Before you move on: you can explain why the default model and agent budget are appropriate for this task.

DO NOW / NOT YET

DO NOW	NOT YET
[ ] Evaluate difficult examples, not only demos. [ ] Keep a fallback or escalation route. [ ] Treat repeated loops as both a cost and quality signal.	Route solely from public model benchmarks. Use a bigger model to compensate for bloated context. Cap only tokens per individual call in an agent workflow.

FIELD NOTES

- Model prices and capabilities change quickly; the durable asset is your task-level evaluation set.
- For non-interactive work, batching and queueing can be as valuable as model switching.

Turn pricing and limits into a feedback loop

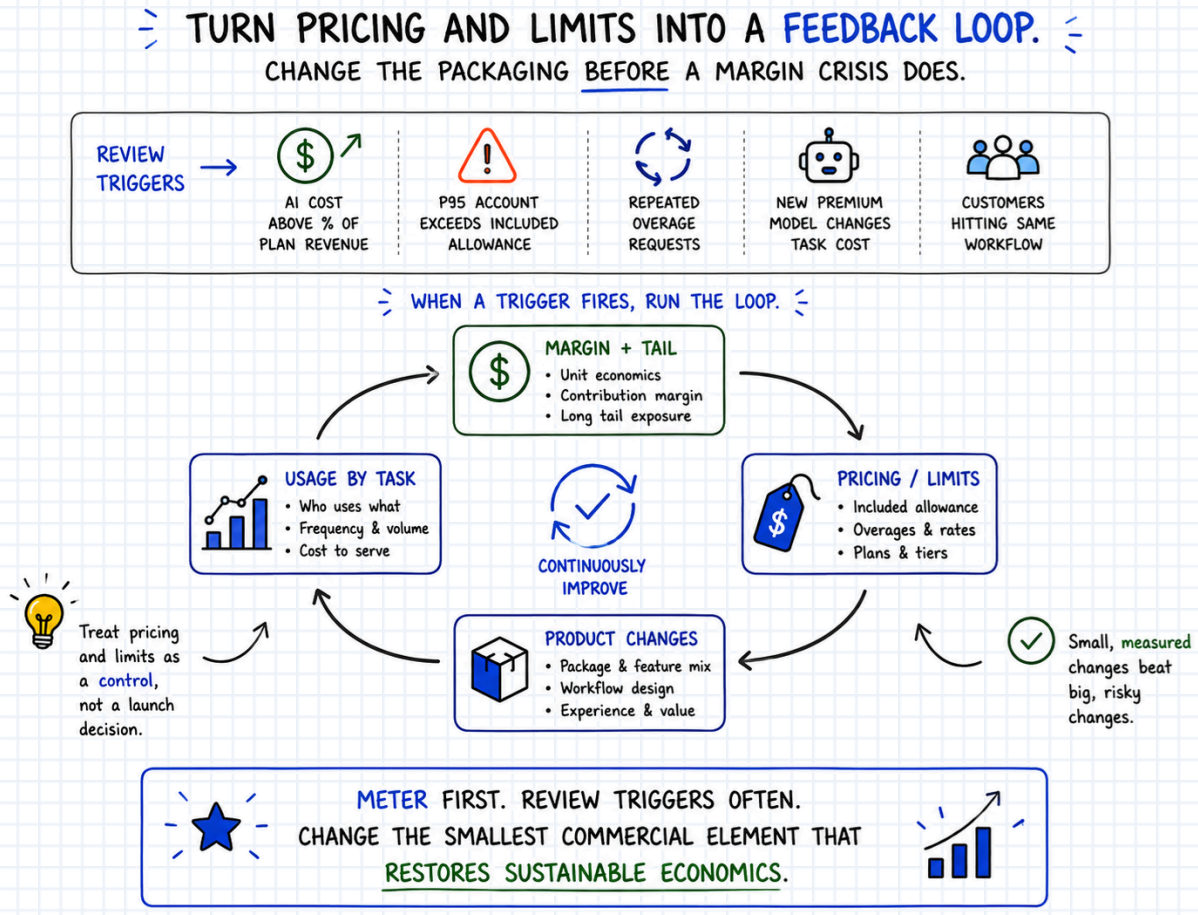
What evidence will make you change the packaging - before a margin crisis does?

The first AI price is rarely the final AI price. Adoption, models, agent behavior, customer value, and provider economics all move. Treat pricing and limits as a controlled feedback loop rather than a one-time launch decision.

Choose review triggers now: AI cost above a percentage of plan revenue, P95 account exceeding the included allowance, repeated overage requests, a new premium model changing task cost, or customers repeatedly hitting the same valuable workflow.

When the trigger fires, you have options short of a wholesale price change: change the included allowance, add a usage pack, introduce a credit pool, separate a premium workflow, or move a measurable task to outcome pricing.

FIELD RULE



FIELD 10 / 10 - PUT IT TO WORK

5-MINUTE FIELD EXERCISE: Write the pricing review trigger

1. Choose 2-3 economics signals that should trigger review.
2. Pick an owner and a monthly or quarterly review cadence.
3. List the smallest packaging changes available before a full repricing.
4. Define how grandfathering, notice, and admin controls would work for existing customers.
5. Write the first trigger in one sentence that product, finance, and engineering all understand.

Pricing / allowance trigger: _____

Review owner: _____

Review cadence: _____

Smallest likely next change: _____

Before you move on: the team knows what evidence causes a packaging review and what options exist before a disruptive repricing.

DO NOW / NOT YET

DO NOW	NOT YET
[ ] Meter before monetizing.	Wait for renewal season to discover unprofitable usage.
[ ] Give existing customers notice and predictability.	Copy another company's credit system without a value map.
[ ] Review economics when provider/model mix changes materially.	Treat the first AI launch price as permanent.

FIELD NOTES

- The best pricing change is often smaller than expected: an included allowance plus visibility can be enough.
- Commercial feedback should include product quality: a costly task that customers love may deserve expansion, not suppression.

IN THE WILD Salesforce kept its $2-per-conversation Agentforce option and added Flex Credits in 2025, priced around actions ($0.10 per action at the published rate), explicitly to align spend more closely with work performed. GitHub likewise changed its AI billing unit as the product mix evolved. [3][4]

Your AI feature economics readiness card

This is not a compliance score. It is a quick check that an established product can grow AI usage without making cost, margin, and customer limits invisible.

[ ] 1. I can state the variable cost of at least one useful AI task.

[ ] 2. I know how AI adoption changes the gross margin of an existing plan.

[ ] 3. AI usage can be attributed to account, plan, and task.

[ ] 4. I know the median and heavy-tail AI cost per account and its share of revenue.

[ ] 5. I have chosen a customer-facing value unit, even if it is only used for internal planning today.

[ ] 6. “Included AI” has an explicit allowance, pool, reset, and limit behavior.

[ ] 7. Task, account, and global guardrails have graceful product behaviors.

[ ] 8. I have audited one high-volume context path and measured before/after task cost.

[ ] 9. The default model passes a task-level quality bar and agent workflows have step/cost budgets.

[ ] 10. I have written the evidence that will trigger a pricing or allowance review.

A simple next 30 days

When	Focus	Outcome
Week 1	Measure	Cost per task, feature P&L, account/plan/task attribution.
Week 2	Bound	Heavy-tail review, included allowance, guardrail ladder.
Week 3	Improve	Context audit, model benchmark, agent budget.
Week 4	Decide	Commercial unit, review trigger, rollout/notice plan.

The goal is not to turn product teams into finance teams. It is to keep AI adoption fast while making the economics visible and intentional.

A NOTE FROM USAGETAP

Use the guide even if you never use our product.

This field guide is meant to be useful on its own. Every exercise can be done with provider dashboards, logs, a spreadsheet, and a few product decisions.

UsageTap exists because the manual version becomes harder as customer and AI workflows multiply. It is built to help teams connect customer usage to the economics of AI features without forcing the commercial model to expose tokens or provider details.

The operating idea is simple:

- Optimize wasted context and model spend.
- Meter consumption where the product understands the customer and task.
- Forecast exposure before a plan or account becomes a surprise.
- Monetize usage only when the value and economics justify it.

Field Guide 01 is for startups and accelerator cohorts preparing to launch. This companion guide is for established software companies adding AI features to products that already have customers, plans, entitlements, and margin expectations.

See UsageTap.com for more information

What established software companies are teaching us in public

The companies below are not presented as failures. Their public pricing and packaging changes show a recurring reality: as AI moves from demo to everyday workflow, the commercial unit, allowance, and controls tend to become more explicit. Snapshot: September 1, 2026.

Company	Public pattern	Why it matters
Intercom [1]	Outcome pricing	Fin is priced from $0.99 per outcome, with at most one outcome charged per conversation. Lesson: a measurable value event can be cleaner than exposing compute.
Zendesk [2]	Automated resolutions	AI-agent pricing now uses automated resolutions, replacing older bot MAU / Answer Bot models. Lesson: the meter can move toward completed work.
Salesforce [3]	Conversation + action credits	Agentforce supports $2 per conversation plus Flex Credits where a published action consumes 20 credits ($0.10). Lesson: one unit may not fit every agent workflow.
GitHub [4]	Requests -> AI credits	Copilot enforced premium-request allowances in 2025 and moved new billing in 2026 to AI Credits based on model and tokens. Lesson: the meter can evolve with workload shape.
Adobe [5]	Included generative credits	Creative Cloud / Firefly use monthly credits and add-on capacity for premium AI. Lesson: included capacity can preserve a simple subscription while bounding expensive work.
Notion / Atlassian [6][7]	Included AI + allowances	Notion now applies allowances to some AI features; Atlassian uses pooled Rovo credits. Lesson: included AI can still have explicit resource budgets.

The recurring pattern is not “usage pricing everywhere.” It is visibility first, a bounded included experience, and a unit that can evolve as the AI feature becomes real.

Official sources for the public case notes

Pricing and packaging change frequently. The case notes above are snapshots of official company materials available on September 1, 2026. They support the described public packaging patterns; they are not claims that any company made a mistake.

[1] Intercom - Fin AI Agent outcomes

Current outcome definitions and prices, including $0.99 resolution/procedure handoff/disqualification and one outcome per conversation.

https://www.intercom.com/help/en/articles/8205718-fin-ai-agent-outcomes

[2] Zendesk - Moving to automated resolutions

Zendesk states AI-agent pricing is now measured in automated resolutions, replacing former monthly active users pricing for bots and Answer Bot resolutions.

https://support.zendesk.com/hc/en-us/articles/6931689272090-Moving-to-automated-resolutions-from-existing-bot-pricing-plans

[3] Salesforce - Agentforce flexible pricing

Salesforce introduced Flex Credits in May 2025 alongside conversation pricing: $500 per 100,000 credits, 20 credits ($0.10) per action, and $2 per conversation.

https://www.salesforce.com/news/press-releases/2025/05/15/agentforce-flexible-pricing-news/

[4] GitHub - Copilot billing changes

GitHub documented premium-request allowances in 2025 and says that as of June 1, 2026 new billing uses AI Credits where interaction cost depends on model and tokens.

https://docs.github.com/en/copilot/reference/copilot-billing/request-based-billing-legacy/what-changed-with-billing

[5] Adobe - Generative credits FAQ

Adobe documents plan-specific monthly generative credits, premium vs standard feature treatment, and additional credit plans.

https://helpx.adobe.com/creative-cloud/apps/generative-ai/generative-credits-faq.html

[6] Notion - AI plan and usage allowance

Notion announced all-in-one Business/Enterprise AI pricing in May 2025; current help documents describe six-hour/monthly usage allowances for certain AI features.

https://www.notion.com/releases/2025-05-13

https://www.notion.com/help/manage-your-usage-allowance-for-notion-ai

[7] Atlassian - Rovo usage allowance

Atlassian documents pooled Rovo credits by subscription/seat, monthly resets, and complexity-based consumption for advanced AI features.

https://support.atlassian.com/rovo/docs/rovo-usage-limits/

For a downloadable field guide, timestamping examples matters: keep the lessons durable and re-check the public numbers before reprinting.