HYPOTHESIS PAPER AND PRE-REGISTRATION DRAFT. GOVERNING AI MODEL SPEND.

Route on evidence, not on the prompt

A proposed design for keeping everyday AI work on cheaper models without blocking anyone from frontier intelligence, and a study protocol to test it against the right controls.

81%

lower cost per completed task on Sonnet than Opus in exp05, at equal completion

1.3%

extra saving a perfect prompt router could have added on top of that

0 / 24

times Copilot Auto upgraded the model when told a task was complex, in Michelin's audit

0

tasks in exp05 that only the frontier model could solve. The number this study must first change

Andrew Kaiserauer. September 2026. Version 0.2 (protocol revised; not yet registered, not yet tested).

Companion to "Cheapest per token is not cheapest per task" (exp05, v1.0).

Harness: `github.com/kornsour/model-routing`. Study id: **exp06**.

Evidence tags: **MEASURED** independent or controlled data. **VENDOR** self-reported. **UNTESTED** reasoned design. **ANALOGY** measured in another domain and carried over by argument.

What changed from v0.1

v0.1 stated the primary hypothesis against always-frontier. The companion study already shows always-mid-tier satisfies that bar with no escalation at all (100% vs 99% completion at 19% of the cost), so a study built on v0.1 could pass without the escalation ladder doing anything. v0.2 makes these changes:

1. **The control is the best fixed default, not the most expensive model.** Always-mid-tier is the primary control. Always-frontier is the ceiling.
2. **Headroom is a gated stage with a stopping rule**, not a setup step. If calibration finds no work the frontier model does and the mid-tier does not, the ladder has nothing to earn and the study reports that.
3. **Escalation quality is measured directly** as recall and precision against measured need, so a ladder that never escalates cannot pass by accident.
4. **The two decisions that sank the exp05 cascade are fixed before registration:** what counts as a verifier, and whether a handoff starts from a clean checkout.
5. **Every free parameter is frozen** (K, N, the advisor prompt, the classifier) on a calibration split that is never used for the confirmatory test.
6. **The margin is sized by a power calculation** and matches what the task count can resolve. The 2-point margin in v0.1 would need 650 to 2,000 tasks.
7. **Hypotheses that need humans (complaints, workarounds, clarifying questions) are moved to the field pilot** and the pilot's success criteria are tied to a measured baseline.
8. A section on where the exp05 harness and process fall short of best practice, and what exp06 changes.

Summary

Companies are overspending on AI because people pick the most capable model for everything. Setting a cheaper default helps until users switch back, and they do. Blocking the expensive model fails the minority of tasks that need it. Asking people to choose well does not work, because they lack the time and the information.

The instinctive fix is a router: a classifier that reads the prompt and picks the model. The evidence says this is the wrong place to decide. How hard a task is often cannot be seen in the prompt. Off-the-shelf routers send whole topics such as coding to the strongest model. In the one independent production audit found, a vendor router cost 6 to 18 times more than a small model that solved the same tasks.

The hypothesis. Decide the model on evidence gathered while the work happens, not on the prompt before it starts. Every session starts on a mid-tier model that users cannot change. Frontier intelligence is never blocked; it is reached by climbing an escalation ladder. The first rungs keep the same model and its cache (clarify, think harder, consult a frontier advisor). The upper rungs hand the task to a frontier session when there is evidence it is needed: failing tests, no progress, a

frustrated user, the advisor's recommendation, or an explicit request with a reason. A budget guard downgrades instead of blocking.

The prediction, stated against the right control. On work where the frontier model has real headroom over the mid-tier model, the ladder recovers most of that headroom at a small fraction of always-frontier's cost premium. On work with no headroom, the ladder costs no more than the plain mid-tier default, because it does not escalate. The companion study showed the second case. This study is designed to find the first, and to say so plainly if it cannot.

Three roles get separated. **The user decides the task. The working model decides how much intelligence it needs**, because it is the only party that sees the evidence (the code, the errors, the user's reactions). **The platform decides the budget** and enforces the starting point. Most current products mix these up.

1. The problem, precisely

The companion study measured 65 agentic coding tasks. A fresh session on Sonnet completed 100% of them at $0.099 per completed task. The same work on Opus completed 99% at $0.528. That is an 81% saving from the default alone. A perfect hindsight router, choosing the cheapest model that would succeed on each task, would have saved only another 1.3%. The money is in the default, and it leaks because people override it.

Anthropic's own cost documentation reaches the same diagnosis. Unexpectedly high Claude Code spend "usually traces back to long sessions that were never cleared or to Opus left as the default model"; "Sonnet handles most coding tasks well and costs less than Opus"[1].

Three approaches are commonly tried. Each fails for a known reason.

Approach	What happens	Evidence
Set a cheaper default	Works for most people, most of the time, but the users who care most about quality (often engineers) switch back. A default a user can change is only a suggestion.	ANALOGY Defaults move behavior strongly: 76% of new hires stayed at a 401(k) default rate, against 8% before auto-enrollment[2]. But programmers changed 40% to 80% of settings, against under 5% of general users[3].
Block the expensive model	Saves money and fails the tasks that need it. People route around it with personal accounts, or lose trust in the platform.	MEASURED When ChatGPT hid its model picker behind an automatic router, the backlash forced OpenAI to restore the picker within days[4].
Show people the cost	Helps a little. People do not bear the cost, and the effect fades.	ANALOGY Price display cut test ordering by 8.6% in one hospital trial[5] and had no significant effect in another[6]. Home energy reports cut use by about 2%[7].

That leaves automation: something the user cannot override, that still gets them the stronger model when the task truly needs it.

2. Why a prompt router is the wrong automation

A router that reads the first message and picks a model is the obvious answer, and the research is consistently discouraging about it for this job.

- **Difficulty is invisible up front.** SWE-Router (2026) argues formally that prompt-only routing on software tasks has an error floor, because "a similar issue can hide either a localized typo or a multi-module refactor"[8]. The companion study found the same: no task description predicted which tasks the cheap model would fail.
- **Routers route by topic, not difficulty.** A 2025 stress-test study found a standard BERT router sent every coding and math query to the strongest model, even when a weaker one would do[9]. For a company whose heavy users are engineers, that reproduces the "everyone on Opus" problem automatically.
- **Routers collapse to the expensive model.** Under generous budgets, existing routers sent nearly all queries to the strongest model, while an oracle needed it for about 20%[10]. Across 33 models and 10 routers, several recent methods, including commercial ones, did not reliably beat a simple baseline[11].
- **In production, the one independent audit found the router cost more.** MEASURED Michelin audited about 700 GitHub Copilot Auto requests over three days. On tasks a small model solved every time, Auto billed 6 to 18 times more. Auto "routes your first sentence and rarely reconsiders": the same seven questions cost 0.78 or 13.71 credits depending on which came first. Telling it a task was complex produced zero upgrades in 24 attempts. Michelin turned Auto off by default[12].
- **Routers can be gamed.** Short appended text reliably pushes commercial and open routers to the strongest model[13]. Internal users will find "this is critical, think very hard" within a week.
- **A vague prompt needs a question, not a bigger model.** On ambiguous coding tasks, asking clarifying questions raised GPT-4's pass rate from 70.96% to 80.80%[14]. A 2026 benchmark found ambiguity costs 7.8 to 19.8 points of pass rate, and a stronger model with extended thinking did slightly *worse* on ambiguous tasks[15].

A small classifier trained on a company's own outcomes still has a role, but as a light prior, not the gate. Section 4 describes where it fits.

3. The hypotheses

Stated so each can be tested, with the control named and the direction of the prediction fixed. Lab hypotheses (L) are tested in exp06. Field hypotheses (F) are tested in the pilot. H0 is a gate, not a hypothesis about the design.

Gate

> **H0 (headroom exists).** On a task set drawn from real handoffs and longer, multi-module work, the frontier model completes tasks that the mid-tier model does not: at least 10 tasks (or 10% of the registered set, whichever is larger) are *measured hard* in calibration (frontier passes at least 2 of 3 trials; mid-tier passes at most 1 of 3).

Stopping rule. If H0 fails after the task set has been extended once, exp06 does not proceed to the confirmatory run. The finding "no frontier-only work found in [description of the work sampled]" is published as the result, because it is the second time in a row and it means the mid-tier default is the whole answer for this class of work.

Primary

L1 (the ladder earns its cost over the best fixed default). On the *measured hard* stratum, the full ladder (`ladder`) completes at least 10 percentage points more tasks than always-mid-tier (`static_sonnet`), and its cost per completed task over the *whole* registered set is no more than 1.5× always-mid-tier's.

Both conditions are required. The first is a superiority test on the stratum where escalation can help. The second is a cost bound over the pooled set, so a ladder that wins the hard stratum by escalating everything does not pass.

L1-ceiling (reported alongside, no verdict role). Over the whole registered set, the ladder's completion is non-inferior to always-frontier (`static_opus`) within 10 points, and its cost per completed task is at most 30% of always-frontier's. This is the v0.1 claim. It is kept because it is what a buyer asks, and demoted because always-mid-tier already meets it.

Secondary (Holm-adjusted as a family; hypothesis-generating)

L2 (cache-preserving rungs do most of the work). Mid-tier with the advisor rung only (`sonnet_advisor`) recovers at least half of the hard-stratum completion gain that the full ladder recovers, and fewer than 15% of ladder sessions on the whole set reach a full frontier handoff (L3 or L4).

Paired with escalation quality, or it is meaningless: on tasks that always-mid-tier fails, the ladder escalates (advisor call, handoff, or both) in at least 70% of trials (**recall**). Of all ladder escalations, at least 50% are on tasks always-mid-tier fails in that trial or in the majority of its trials (**precision**). Recall below 50% means the working model under-escalates and L2 is not supported however low the handoff rate is.

L3 (evidence beats the prompt, on cost, not just accuracy). Explore-then-handoff with a clean-checkout restart (`explore_handoff_clean`) has a lower cost per completed task than a frozen prompt classifier on the brief alone (`C2_trained`), and non-inferior completion within 10 points. The classifier is trained on the calibration split and frozen before registration. Secondary: AUC for predicting *measured need* (mid-tier fails) from execution signals at turn K versus from the brief alone.

L4 (a wrong start anchors the next model). Handoff from a clean checkout (`explore_handoff_clean`) completes more hard-stratum tasks than handoff that inherits the mid-tier attempt's working tree (`explore_handoff_carry`), at no more than 1.25× the cost. exp05 saw this once, uncontrolled; here it is a registered arm.

L5 (clarifying reduces handoffs). On the *ambiguous* stratum (tasks with a deliberately underspecified brief and a scripted answer key for clarifying questions), the ladder with L0 enabled hands off to frontier in fewer sessions than the ladder with L0 disabled, at equal or better completion. Clarifying and escalating are not exclusive; the claim is that clarifying first removes the need for some escalations.

Field (pilot, Section 7.7)

> **F1 (fewer complaints than a wall).** An enforced mid-tier start with a visible, allowance-based "request frontier" path produces fewer help-desk tickets and fewer explicit-model-switch attempts per active user than a blocked frontier model, over the same period.
>
> **F2 (the request path stays rare).** Explicit frontier requests are used in fewer than 10% of sessions, and the share does not rise over the pilot period (no learned gaming).
>
> **F3 (spend falls against the measured baseline).** Spend per active user falls by at least the amount the shadow-mode log predicted, with completion proxies (merged pull requests, accepted outputs) not significantly lower than baseline.

Why this should work, in one line per mechanism

- **Enforced start:** captures the default effect, which is the largest and best-evidenced lever, without depending on user discipline.
- **Model-initiated escalation:** the working model has the evidence of difficulty that neither the user nor a prompt classifier has.
- **Cache-preserving rungs first:** in agent loops a model switch forfeits cached context, so the cheapest escalation is one that keeps the model. The companion study found cache reads and writes were most of each session's cost.
- **Handoffs at boundaries:** a new session starts with an empty cache anyway, so escalating there costs nothing extra in cache.
- **Allowance, not permission:** keeps a legitimate path open, so the controls feel like a policy rather than a wall.

4. The design

Figure 1. The escalation ladder. Every session starts at the top rung. Lower rungs cost more. Rungs marked *cache kept* keep the same model and its prompt cache; rungs marked *new cache* start a new frontier session with a written handoff. The budget guard applies to every rung.

Rung	Mechanism	Trigger	Cache
Start	Mid-tier model, medium effort. Enforced; the user cannot change the starting model.	Every session	kept
L0 Clarify	Ask one to three questions before starting	Request is ambiguous (the model judges, optionally helped by a small classifier)	kept
L1 Effort	Same model, higher reasoning effort	Model's own judgment, or a classifier prior on the request	kept
L2 Advisor	Frontier model reads the transcript and returns guidance; the working model continues	Model decides: before committing to an approach, on a recurring error, before declaring done. **Plus a forced check** before the model declares a task done (Section 4.2)	kept
L3 Handoff	New frontier session, clean checkout, with a written brief of the work so far and the failure log	Verifier still failing after K attempts; no progress in N turns; user rephrases or says "that's wrong"; advisor recommends it	new
L4 Request	User asks for the frontier model with a one-line reason; counted against a monthly allowance; logged	Explicit	new
Budget guard	Past a per-person allowance, frontier requests downgrade to the default model with a notice instead of being blocked	Every rung	

4.1 Enforcement point

Enforcement has to sit where the user cannot change it. In priority order:

- **Managed client settings.** For example, Claude Code's managed `model` setting sets the starting model on every launch, overriding the user's saved choice[16]. Gap: the advisor needs the frontier model on the allowlist, so `/model` still works. Closing that needs a gateway rule. UNTESTED
- **An API gateway** in front of every API-based tool. The gateway pins each session to its starting model, using a session id header Claude Code sends on every request[17], and rewrites direct requests for frontier models into the escalation path instead of rejecting them.
- **Admin console defaults** for chat products, which are weaker (Section 6).

Routing is decided per session and kept fixed within it. Changing model mid-session breaks the prompt cache, and per-turn routing anchors badly anyway (Michelin found the first sentence decided the whole session). L3 is not a mid-session switch: it is a new session, which is where the cache is empty regardless.

4.2 The rungs and their evidence

Rung	Evidence status
L0 Clarify	**MEASURED** in coding benchmarks[14][15], with simulated users. Untested inside an agent loop.
L1 Effort	**VENDOR** Anthropic reports large token savings at lower effort[18]. No head-to-head against cross-model routing on an agentic benchmark (companion, §6.2).
L2 Advisor	**VENDOR** Sonnet with an Opus advisor: +2.7 points on SWE-bench Multilingual and 11.9% lower cost per task[19]. The advisor does not invalidate the main model's prompt cache in Claude Code[20] and is a server-side tool in the Claude API[21].
L3 Handoff	**MEASURED** explore-then-escalate beat prompt-only routing in SWE-Router[8]. The clean-checkout rule and the user-signal triggers are **UNTESTED**; exp05 saw the anchoring failure once.
L4 Request	**UNTESTED**

The forced advisor check. v0.1 named "the model may not escalate when it should" as the main risk and left it to the model's judgment. v0.2 adds one mandatory advisor call before the working model declares a task done, on every ladder session. It costs one frontier read of the transcript per session and gives the recall measurement a floor. Whether the forced check is worth its cost is answered by comparing `ladder` with `ladder_noforce` (Section 7.2).

The verifier. exp05's cascade never escalated because its checker (visible tests green, something in scope changed) accepted partially correct work. Execution signals are only as good as the check behind them. exp06's verifier for the L3 trigger is: the task's visible tests, **plus** a generated test pass (the working model writes tests for the brief's stated behavior before it starts, and those tests must pass), **plus** a scope check. The hidden tests remain the grader and are never visible to any rung. An arm with the hidden tests as the verifier (`ladder_ideal`) is the non-deployable upper bound, as `D_ideal` was in exp05.

Frozen parameters. K (verifier failures before handoff), N (turns without a changed file or a new test passing), the advisor system prompt, the L0 ambiguity prompt, and the clarifying-question cap are chosen on the calibration split and written into the `[preregistration]` table. They are not tuned on confirmatory data. If a pre-registered grid is run (for example K in {2, 3}), every cell is reported.

4.3 Where a trained classifier still helps

A small model trained on the company's own logged outcomes (which requests needed escalation and which did not) is worth building, in two limited roles. It sets the **starting effort level**. It **pre-approves the frontier model for planning-type requests**, such as a request that opens with an architecture question: the planner/executor split, where a frontier model plans and a cheaper one executes, has shown quality gains in Aider and ships as `opusplan` in Claude Code[22]. It never blocks escalation, and it runs in shadow mode first. Commercial tools already exist for training such a router on your own evaluation data[23], and research routers that use conversation history and user context outperform prompt-only ones[24].

In exp06 the classifier is the comparison arm for L3 (`C2_trained`), trained on the calibration split's briefs and measured labels, frozen, and hashed into the registration.

4.4 Budget guard

Past a per-person allowance, requests for the frontier model are downgraded to the default model with a notice, rather than blocked. This behavior already exists in some gateways: LiteLLM's `budget_fallbacks` reroutes to a fallback model when a key's budget is exceeded "instead of returning a budget_exceeded error"[25], and Cloudflare AI Gateway can fall back to a cheaper model inside a dynamic route[26]. Every chat product surveyed blocks instead.

4.5 Transparency

Every response shows a small routing receipt, for example "Sonnet · Opus consulted twice". The GPT-5 experience shows that silent routing destroys trust[4]. A receipt makes escalation visible as a feature: the user can see that the stronger model was brought in when it mattered. In the pilot, the receipt is also the data source for F2.

5. What it should cost

The economics depend on one number: how often a session climbs to a full frontier handoff (L3 or L4). v0.1 modeled this with workload-average costs. That model is biased in the ladder's favor, and it is replaced here.

Why the v0.1 model was optimistic. It priced a failed mid-tier attempt at the average mid-tier session ($0.109 with advisor overhead) and a frontier run at the average frontier session ($0.528). But the sessions that escalate are the hard ones. In exp05's calibration, hard-batch tasks cost $0.09 to $0.15 on Sonnet and $0.63 to $0.89 on Opus, against $0.07 and $0.26 on the easy batch. Haiku's failures burned the whole 40-turn budget. Both inputs to the escalated-session cost are low, so the 79% break-even in v0.1 is an upper bound, not an estimate.

The v0.2 model, to be filled from calibration. Let s be the share of sessions that reach a handoff, a the advisor overhead per session (measured, not assumed at 10%), $c\_m(hard)$ and $c\_f(hard)$ the measured mid-tier and frontier costs on the hard stratum, and $c\_m(easy)$ the mid-tier cost on the rest:

$$\text{cost per session} \approx (1 - s) \cdot (c\_m(easy) + a) + s \cdot (c\_m(hard) + a + c\_f(hard))$$

The break-even handoff rate against always-frontier is reported from calibration data before registration, with the stratum costs stated. Figure 2 of v0.1 (linear, break-even at 79%) is withdrawn until then. The qualitative conclusion stands in either model: the main risk is not an escalation rate that is too high; it is a model that under-escalates and quietly delivers worse work. That is why L2's recall measurement is registered next to L1.

6. What can be built today, and what is missing

Surface	Available now	Gap
Claude Code	Managed model (the starting model on every launch); availableModels allowlist; advisorModel for the L2 rung; CLAUDE_CODE_SUBAGENT_MODEL; opusplan; effort caps; gateway support with session-id and request-class headers[16][17][20]	The advisor needs the frontier model on the allowlist, so users can still switch to it with /model. Closing that requires a gateway rule that rewrites direct frontier requests at session start. **UNTESTED**
Claude.ai Enterprise	Organization or role default model; a beta "always start with default" setting; model restrictions by role; per-member spend caps with a request-credits flow[27][28]	Users can still switch model within a chat. No advisor in chat, no soft downgrade
ChatGPT Enterprise	Admin default model and reasoning level; role-based model access; spend limits with approval requests[29]	Same gaps; routing inside ChatGPT is not configurable by the customer
GitHub Copilot	Per-model enable or disable policies; per-user budgets; Auto model selection[30]	No admin default model found; budgets block rather than downgrade; Auto audited poorly[12]
API gateways (LiteLLM, Cloudflare, Portkey, Vercel)	Session pinning, model rewrite rules, per-user budgets; soft downgrade in LiteLLM and Cloudflare[25][26]	Escalation signals must be built by the operator

What vendors would need to build for this to work in chat products as well as coding tools:

1. **Model-initiated escalation in chat apps.** The advisor pattern exposed in Claude.ai and ChatGPT, so a mid-tier chat can call on a frontier model mid-conversation.
2. **An enforced start that still escalates.** An admin setting meaning "users start on X and cannot switch, but the model may escalate", rather than today's choice between an overridable default and a hard restriction.
3. **Allowance-based requests.** A per-person frontier allowance, with a reason field, that downgrades softly when used up.
4. **Routing receipts and outcome logs.** Per-response records of which models were used and why, exportable so companies can train their own priors and audit escalation.

7. Study protocol (exp06)

The protocol extends the exp05 harness (`src/model_routing/dispatch/`), reuses its deterministic grading, task-clustered statistics, randomized block ordering, provenance hashing and `[preregistration]` mechanism, and changes what Section 9 says needs changing.

7.1 Stage 0: task set and calibration (gated by H0)

Sources, in this order of priority.

1. **Real handoffs.** The 562 task chips harvested from the author's own Claude Code transcripts (`make harvest-chips`). Each is converted into a fixture task with hidden tests only where a deterministic grader can be written; the brief is kept at its real length (median

about 500 tokens, versus about 270 for exp05's synthetic briefs). Selection is by a seeded random draw from the harvested set, not by the author's judgment of difficulty.

2. **Long, coupled work.** Multi-module refactors and migrations in the three fixture repos, with briefs of 1,000 to 2,500 tokens matching real subagent dispatches. Written before any model is run on them.
3. **An ambiguous stratum** for L5: briefs with one deliberately missing requirement, plus a scripted answer key so a simulated user can answer clarifying questions deterministically. The simulated user is a fixed script keyed on question intent, not an LLM.
4. **A non-coding stratum** (analysis and document tasks) is included only where an answer key allows deterministic grading. Rubric grading by an LLM judge is not used; the companion study's "no LLM judge" rule stands.

What is not allowed. Adding tasks *because* the mid-tier model failed them in a trial run. That selects on the outcome. Tasks are written or drawn first, then calibrated. If calibration shows too little headroom, the fix is to draw more tasks from the sources above (one extension allowed), not to hand-pick failures.

Calibration. Every task on `static_haiku`, `static_sonnet`, `static_opus`, **3 trials** (exp05 ran 2 for budget; 3 is required here because the labels decide the stratum). Labels are measured: *easy* (mid-tier passes 3/3), *medium* (mid-tier 1 to 2 of 3), *hard* (mid-tier ≤ 1/3 and frontier ≥ 2/3), *unsolved* (frontier ≤ 1/3; excluded before registration with the reason recorded).

Split. The calibrated set is divided by a seeded draw, stratified by label, into a **tuning split (30%)** and a **confirmatory split (70%)**. K, N, the advisor and L0 prompts, and `C2_trained` are fitted on the tuning split only. The confirmatory split's hash is what the registration freezes. The tuning split is never scored in the confirmatory report.

Gate H0 is evaluated on the confirmatory split.

7.2 Arms

Arm	What happens	Role
static_sonnet	Fresh session on the mid-tier model, medium effort	**Primary control** (best fixed default)
static_opus	Fresh session on the frontier model, default effort	Ceiling
ladder	Full ladder, L0 to L3, with the forced advisor check before done. (L4 has no lab analogue.)	**Treatment**
ladder_noforce	Full ladder without the forced advisor check	Prices the forced check
ladder_noL0	Full ladder with clarifying disabled	L5 comparison (ambiguous stratum only)
sonnet_advisor	Mid-tier with the L2 rung only; no handoff	L2
sonnet_effort	Mid-tier with the L1 rung only (effort raised on the model's own judgment)	L1 evidence
explore_handoff_clean	Mid-tier for up to K verifier failures or N stalled turns, then frontier from a clean checkout with a brief and the failure log	L3, L4
explore_handoff_carry	Same, but the frontier session inherits the working tree	L4
C2_trained	Frozen classifier on the brief alone picks the model; fresh session on the pick	L3 comparison
ladder_ideal	Ladder with hidden tests as the verifier	Non-deployable upper bound
oracle	Computed: cheapest static arm that passed, per task (majority of trials)	Bounds any router

Budget priority, fixed before the run. If spend forces arms to be dropped, they are dropped from the bottom of this list: `ladder_ideal`, `sonnet_effort`, `ladder_noL0`, `explore_handoff_carry`, `ladder_noforce`, `C2_trained`, `sonnet_advisor`. The four arms `static_sonnet`, `static_opus`, `ladder`, `explore_handoff_clean` are required; dropping any of them makes the run exploratory.

Oracle definition. Per task using the majority of trials, not per task-and-trial. exp05's per-trial oracle is the most generous possible bound; the per-task version is the one a deployable policy could in principle match.

7.3 Design controls

- **Paired, randomized block.** Every task runs under every arm for 3 trials. Within each (task, trial) block the arms run in a seeded random order. Sessions are sequential.
- **Turn cap.** 40 turns for every arm, and the report states how many cells hit it per arm. A sensitivity analysis re-scores with capped cells excluded, so a cap that binds unevenly (it bound only on Haiku in exp05) does not silently decide the cost result.
- **Blinding.** No arm reads a task's label or stratum. The working model's prompts are identical across `ladder*` arms except for the rungs enabled. Graders are deterministic. The analyst does not see arm labels in the per-task pass matrix until the registered statistics have been computed (the report tool computes them; the author reads the summary).

- **Provenance.** Harness git SHA, config hash, confirmatory-split task hash, tuning-split hash, classifier weights hash, seed, CLI versions, resolved model ids for every session, and the frozen prompts, all in `meta.json`.
- **Intention to treat.** Every planned cell counts once. Sessions ending in the turn cap, wall-clock timeout, per-session budget cap or a 4xx are graded as-is. Provider outages and account usage limits are set aside and redone (this was a mid-run deviation in exp05; here it is registered up front).
- **No re-runs, no drops after the fact.** A defective task is excluded from every arm, recorded, and the run becomes exploratory.
- **Single operator, single vendor.** Stated as a limitation. A Codex replication track is a separate run with its own registration.

7.4 Measurement

- **Primary metric:** cost per completed task, list price, summed over worker, advisor, router and escalation sessions. Parent-seeding is excluded as in exp05.
- **Completion:** hidden tests pass, visible tests still pass, nothing outside `allowed_paths` changed. Deterministic.
- **Escalation events**, logged per session with the turn number and the trigger: advisor call (voluntary or forced), L0 question asked, effort raised, handoff (with K or N reached), session declared done.
- **Escalation need** (the label for recall and precision): the task's `static_sonnet` outcome in the same trial, and its majority outcome across trials. Both are reported.
- **Recall** = escalations on needed trials ÷ needed trials. **Precision** = escalations on needed trials ÷ all escalations. Computed for "any escalation" and for "handoff" separately.
- **Secondary:** handoff rate, advisor calls per session, cache hit rate, turns per session, tokens re-read, verifier false-accept rate (verifier green but hidden tests fail), clarifying questions asked on the ambiguous stratum.

7.5 Statistics and decision rules

Computed by `model_routing.dispatch.report.summarize`, extended for the new comparisons.

- **Intervals.** 95% task-clustered bootstrap, 2,000 draws, all trials of a task resampled together.
- **L1, condition 1 (hard stratum, superiority).** Paired completion difference `ladder − static_sonnet`. Supported if the lower bound of the interval exceeds +10 points. Not supported if the upper bound is below +10. Otherwise inconclusive.
- **L1, condition 2 (whole set, cost bound).** Ratio `cpt(ladder) / cpt(static_sonnet)`. Supported if the upper bound of the interval is below 1.5. Not supported if the lower bound exceeds 1.5.
- **L1 verdict** = supported only if both conditions are supported.
- **L1-ceiling.** Non-inferiority within 10 points and cost ratio interval entirely below 0.30, reported without a verdict role.

- **L2 to L5.** Same interval rules on their stated comparisons. `p_saving`, `p_pass` (two-sided task-clustered permutation) and `p_noninf` reported; Holm adjustment across L2 to L5 as a family. Recall and precision reported with bootstrap intervals; the L2 verdict requires recall ≥ 0.70 (lower bound ≥ 0.50).
- **Sensitivity analyses, pre-specified:** capped cells excluded; advisor calls billed at the 5-minute cache rate; the per-trial oracle alongside the per-task one.
- **Power.** Paired McNemar approximation, $n \approx (z_\alpha + z_\beta)^2 \cdot p_{disc} / \delta^2$, $\alpha = 0.05$ two-sided, power 0.80, where `p_disc` is the share of paired task-trials on which the two arms disagree.

Claim	δ	discordance 17%	25%	33%
Superiority, +10 pts (L1 cond. 1)	0.10	133 paired (44 tasks)	196 (65 tasks)	259 (86 tasks)
Superiority, +5 pts	0.05	533 (178 tasks)	784 (261 tasks)	1,035 (345 tasks)
Non-inferiority, 10 pts (L1-ceiling, L3)	0.10	133 (44 tasks)	196 (65 tasks)	259 (86 tasks)
Non-inferiority, 2 pts (v0.1's margin)	0.02	3,332 (1,111 tasks)	4,900 (1,633 tasks)	6,468 (2,156 tasks)

The rows are paired task-trials, with tasks at 3 trials in parentheses. Two conclusions fixed the protocol: v0.1's 2-point margin is out of reach at any budget this study will have, and the +10-point superiority claim on the hard stratum needs **45 to 90 hard tasks** in the confirmatory split depending on discordance. That sets the task-set target: at least 65 measured-hard tasks in the confirmatory split (about 95 before the split), plus the easy and medium tasks needed for the cost bound, plus the ambiguous stratum. The report's power note recomputes from observed discordance in calibration before registration; if the hard count is short, the set is extended once (Section 7.1) before the run.

7.6 Registration checklist

Registration (`make dispatch-preregister ... WRITE=1`) happens only when every item is true, and each is recorded with its date:

1. Calibration run at 3 trials on every static arm, on every task.
2. H0 met on the confirmatory split; hard-task count meets the power target at the observed discordance.
3. Tuning split and confirmatory split hashed; K, N, prompts and `C2_trained` frozen and hashed.
4. Verifier's false-accept rate measured on the tuning split and reported.
5. Smoke run on real models of every arm, on at least two hard tasks, showing each escalation path fires at least once (the exp05 cascade was registered before its escalation path had ever fired on a real failure).
6. Spend estimate within budget for the required arms; budget priority list recorded.
7. Decision rules, margins, sensitivity analyses and the arm list written into the `[preregistration]` table, committed, and the commit SHA recorded here.

Anything changed after registration goes under **Deviations** with the date and reason, and the run is exploratory.

7.7 Field pilot

The lab cannot test F1 to F3. The pilot is designed so its numbers can be interpreted.

- **Baseline first (2 weeks).** Measure current spend per active user, model mix, explicit switch rate, and completion proxies for every participating team, before anything changes. If most teams already run mostly on the mid-tier model, the achievable saving is small and the pilot's success criterion is set from the baseline, not from a fixed 50%.
- **Shadow mode (2 to 4 weeks).** Log what the ladder would have done while users work as today. This yields the predicted saving used in F3, outcome labels for the classifier prior, and an estimate of how often L3 triggers fire on real work.
- **Stepped-wedge rollout (8 to 12 weeks)** rather than a single A/B split. Teams are randomized to a start week; every team is eventually treated, and each team serves as its own control. With a handful of teams this recovers far more power than a parallel A/B and avoids the "which two teams are comparable" problem.
- **Outcomes.** Spend per active user; merged pull requests and accepted outputs per active user; explicit frontier requests per session (from the routing receipt); help-desk tickets tagged model-access; a two-question satisfaction pulse. Personal-account workarounds cannot be observed directly and are not claimed; a question on the pulse asks about them.
- **Detectable effects, stated before the pilot.** With T teams and a stepped wedge, the report states the minimum detectable spend change and satisfaction change at the observed between-team variance. A 0.3-point satisfaction difference on a 5-point scale is not resolvable with fewer than roughly 200 respondents per period; the pilot either enrolls that many or reports satisfaction descriptively.
- **Success criteria (F3).** Spend per active user falls by at least 70% of the shadow-mode prediction; completion proxies within 10% of baseline; F2's request share under 10% and not rising.

8. Ways this could be wrong

- **Models may not escalate when they should.** If the mid-tier model is overconfident, it will not call the advisor and will deliver mediocre work. The forced advisor check and the recall measurement exist so this shows up as a number rather than a suspicion. If recall is low even with the forced check, the ladder's premise (the working model can see that it needs help) is wrong.
- **Advisor costs may add up.** Each advisor call rereads the full transcript at frontier prices, uncached[20]. On long sessions, frequent consultations could cost more than a handoff. `ladder_noforce` and the 5-minute-cache sensitivity analysis size this.
- **Headroom may not exist for this class of work.** exp05 found none. If exp06's calibration finds none either after extending the set, that is the result, and the practical advice is the companion study's: set the default to the mid-tier model and stop.

- **The verifier may still false-accept.** Generated tests can be wrong in the same direction as the code. The false-accept rate is measured and `ladder_ideal` shows the ceiling a perfect verifier would reach.
- **Some work needs the frontier model all the way through.** Long, tightly coupled refactors may pay for a failed mid-tier attempt and then the handoff. A classifier prior for such requests may be worth its errors; `C2_trained` measures that.
- **Non-coding work has weaker signals.** Without tests, the ladder relies on the model's judgment and on user reactions, which are unvalidated triggers. exp06's non-coding stratum is small and deterministically graded; the general case is a limitation.
- **The simulated user is a script.** L5's result says clarifying questions help when the answer exists and is given cleanly. Real users answer late, partially, or wrongly.
- **Single author, single vendor, list prices.** The tasks are written by the person who designed the policies. The real-handoff stratum limits that, but does not remove it. A second vendor track and, ideally, a task set contributed by someone else would.
- **Vendor products move fast.** Several controls cited here shipped in 2026. Plan names, defaults and prices will change; the design should depend on the pattern, not a specific setting.

9. Where the exp05 setup falls short of best practice, and what exp06 changes

The companion study was pre-registered, deterministically graded and reported with its exploratory results labelled. These are the places where its process would not survive a careful reviewer, recorded here so exp06 does not repeat them.

exp05 practice	Problem	exp06 rule
The headroom gate (“cheapest model passes 50 to 70%”) was not met (83%) and the run was registered anyway, with a subgroup analysis as the remedy	A gate that can be waived is not a gate. The registration also chose the most favorable option of three after seeing calibration data	H0 is a stopping rule with one allowed extension, evaluated on the confirmatory split, before any arm other than the static baselines is run
Calibration at 2 trials, planned at 3, for budget	Labels that decide strata were measured with less precision than the outcomes they stratify	3 calibration trials are a registration requirement
Thresholds, prompts and the cascade rule (escalate_on_error) were decided after smoke runs on the same task set used for the confirmatory run	Tuning and testing on the same tasks	Tuning split (30%) and confirmatory split (70%), hashed separately
The comparison that decided the paper (always-Sonnet vs the router) was exploratory	The registered primary was the easy comparison against the most expensive model	The primary control is the best fixed default
The cascade’s checker was known to be weak, and the escalation path had never fired on a real failure when the run was registered	An arm that cannot escalate does not test escalation	Verifier defined and its false-accept rate measured before registration; every escalation path must fire in a smoke run on a real failure
A mid-run deviation (redo outages) changed the inclusion rule, with a sound reason, during stage 1	Reasonable, but a deviation is a deviation	Outage handling is registered up front
“Secondary policies may be dropped for budget; not a deviation” with no order stated	Which arms survive a budget squeeze should not be a choice made after seeing early results	Budget priority list fixed before the run
The 40-turn cap produced 36 errored cells, all on one model, and the cap’s effect on that model’s cost is not separated	The cap is part of the treatment for the model it binds on	Capped-cell count per arm reported; sensitivity analysis excluding them is pre-specified
The oracle is per task and trial	The most generous bound; a deployable policy cannot match it	Per-task majority oracle as the headline; per-trial alongside
Synthetic tasks written by the author, with briefs about half the length of real ones	Author bias toward tasks the policies handle; briefs unlike production	Real-handoff stratum drawn at random from the harvested chips; long-brief stratum; sources fixed before calibration
One power calculation, based on the pilot’s discordance, with a 10-point margin chosen to fit the budget	Fine as far as it goes, but the margin should be stated as a limitation of resolution, not only as a business judgment	Power table published in the protocol; the margin is stated as the smallest effect the budget can resolve, and smaller effects are explicitly out of scope
Analyst reads per-cell outcomes as they land	Ordinary for a single-author study, but it invites looking	The report tool computes registered statistics before the per-task matrix is read

None of this changes the companion study’s conclusions, which rest on a 5× cost gap and 100% pass rates that no reasonable reanalysis moves. It changes what exp06 has to do to be believed on a closer call.

Evidence base

Evidence was gathered on 25 September 2026 from primary sources where possible; vendor claims are tagged. Several 2026 papers were read from abstracts or HTML summaries.

1. Anthropic. Claude Code: manage costs effectively. code.claude.com/docs/en/costs
2. Madrian and Shea. The Power of Suggestion: Inertia in 401(k) Participation. NBER w7682; slides eml.berkeley.edu/symposia/sage02/slides/madrian.pdf
3. Spool. Do users change their settings? UIE, 2011. archive.uie.com/brainsparks/2011/09/14/do-users-change-their-settings
4. TechCrunch. ChatGPT's model picker is back, and it's complicated. 12 August 2025. techcrunch.com/2025/08/12/chatgpts-model-picker-is-back-and-its-complicated; Altman on reasoning usage via simonwillison.net/2025/Aug/10/sam-altman
5. Feldman et al. Impact of providing fee data on laboratory test ordering. JAMA Internal Medicine, 2013. jamanetwork.com/journals/jamainternalmedicine/fullarticle/1678807
6. Sedrak et al. Effect of a price transparency intervention (PRICE trial). JAMA Internal Medicine, 2017. jamanetwork.com/journals/jamainternalmedicine/fullarticle/2619519
7. Allcott. Opower home energy reports evaluation. povertyactionlab.org/evaluation/opower-evaluating-impact-home-energy-reports-energy-conservation-united-states
8. Son et al. SWE-Router: Routing in Multi-turn Agentic Software Engineering Tasks. June 2026. arxiv.org/abs/2607.00053
9. How Robust Are Router-LLMs? 2025. arxiv.org/abs/2504.07113
10. When Routing Collapses. 2026. arxiv.org/abs/2602.03478
11. LLMRouterBench. 2026. arxiv.org/abs/2601.07206
12. Masseboeuf. Why we turned off Copilot Auto mode by default. Michelin, 18 September 2026. blogit.michelin.io/copilot-auto-mode-off-by-default
13. Shafran et al. Rerouting LLM Routers. 2025. arxiv.org/abs/2501.01818
14. Mu et al. ClarifyGPT. 2023. arxiv.org/abs/2310.10996
15. ClarifyCodeBench. 2026. arxiv.org/abs/2607.00711
16. Anthropic. Claude Code settings and managed settings. code.claude.com/docs/en/settings; code.claude.com/docs/en/managed-settings
17. Anthropic. Claude Code LLM gateway and gateway protocol. code.claude.com/docs/en/llm-gateway; code.claude.com/docs/en/llm-gateway-protocol
18. Anthropic. Introducing Claude Opus 4.5 (effort results). anthropic.com/news/claude-opus-4-5
19. Anthropic. The advisor strategy. 9 April 2026. claude.com/blog/the-advisor-strategy
20. Anthropic. Escalate hard decisions with the advisor tool. code.claude.com/docs/en/advisor
21. Anthropic. Advisor tool (Claude API). platform.claude.com/docs/en/agents-and-tools/tool-use/advisor-tool
22. Aider. Separating code reasoning and editing. aider.chat/2024/09/26/architect.html; Anthropic model configuration (opusplan). code.claude.com/docs/en/model-config

23. Not Diamond. Router training quickstart. docs.notdiamond.ai/docs/router-training-quickstart
24. GMTRouter. 2025. arxiv.org/abs/2511.08590
25. LiteLLM. Budget fallbacks. docs.litellm.ai/docs/proxy/budget_fallbacks
26. Cloudflare. AI Gateway spend limits and dynamic routing. developers.cloudflare.com/ai-gateway/features/spend-limits; developers.cloudflare.com/ai-gateway/features/dynamic-routing
27. Anthropic. Set a default model for your organization. support.claude.com/en/articles/15330088
28. Anthropic. Manage model access for your organization; manage usage credits. support.claude.com/en/articles/15694740; support.claude.com/en/articles/12005970
29. OpenAI. Managing workspace settings and usage limits in ChatGPT Enterprise. help.openai.com/en/articles/8411955; help.openai.com/en/articles/20001001
30. GitHub. Manage availability of models; budgets for usage-based billing; auto model selection. docs.github.com/en/copilot
31. Kaiserauer. Cheapest per token is not cheapest per task. exp05, v1.0, September 2026. github.com/kornsour/model-routing, `docs/experiments/preregistration-exp05.md`, `docs/experiments/findings/2026-09-23-exp05-calibration.md`

Appendix A. Frozen parameters (to be filled at registration)

Parameter	Value	Chosen on	Hash or commit
K (verifier failures before handoff)		tuning split	
N (turns without progress before handoff)		tuning split	
Advisor system prompt		tuning split	
L0 ambiguity prompt and question cap		tuning split	
C2_trained weights		tuning split	
Verifier definition and measured false-accept rate		tuning split	
Confirmatory split task hash			
Tuning split task hash			
Harness commit			
Registration commit and timestamp			

Appendix B. Deviations

None yet. Anything changed after registration is listed here with the date and the reason, and the affected run is reported as exploratory.