

Two requests

Lab note 10 15 Oct 2026 5 min read
written by Claude from this repository's data

A hand-off to a cheaper worker saves money because the worker does the building. But the main model still pays for every request it makes around the hand-off, and in a long session each of those re-reads the whole conversation. This note is about getting those requests down to the two that cannot go: handing the work over, and answering.

Claude does not hand off on its own

Measured again with this release's delegate: in 16 runs on the feature and test tasks of click and of a TypeScript library, with delegate loaded and nobody asking, Claude handed off **0 times**. Before that, 0 of 174 (note 2). In the person's own sessions since the pack was installed, Claude called `hand_off` 70 times; 66 came after an order or after the person had asked for it in one of their last six prompts, and 4 did not. And the Claude writing this note, in a long session full of fully specified build work, did not hand off once until the person asked why not.

An order is followed every time; an invitation almost never. So everything below is about what happens after an order: delegate's own, when a plan is approved, or route's, when a classifier says a prompt is feature or test work (note 7).

Where the requests went

We took the 16 routed runs of the feature and the coverage task and listed every request of the main model. On average: **1.6 requests looking around** before the hand-off, the hand-off itself, and **2.5 after it**, the final answer included.

The requests after were not blind re-runs. They were questions the evidence had left open:

- "The run reports **31,000 deselected**, so I'll confirm the new tests actually ran" (4 of 16 runs).
- "Run the new test file **on its own**": the check had run the whole suite.
- "Pyright flagged new lines, I'll check whether that matters."
- And real review: "one assertion is always true, I'll make it a real one". That request is worth paying for.

delegate already ran the check command itself and showed the diff. What it did not do was answer those questions.

A report in a fixed form

The idea came from the person running this lab: let the worker answer in a fixed form that a program can check. The worker now ends its report with a block:

```
VERIFY
tests: PYTHONPATH=src python -m pytest -q tests/test_duration.py
open: none
```

delegate runs that command beside the check, counts both outputs, and puts one line first in its result:

Verified: the check passed (2020 passed); the new tests passed on their own (27 passed); nothing left open. If the diff is what you asked for, report to the person now: nothing is left to run.

Or `Not verified:`, with each reason: a check that failed, new tests that did not run (0 selected is not a pass), test files changed without a command that ran them alone, a worker that stopped on an error, or what the worker itself left open. The parsers are strict on purpose: an output they do not recognise is "not counted", never "passed".

The order changed with it: make the project's whole test suite the check, and on a Verified result with the right diff, report without running anything again.

Handing off before reading the code

With the verified report, the requests after a hand-off fell to one. But in the same runs a second thing showed: **allowed to look around first, Claude kept the work in 5 of 16 runs**. It read the code, judged the job small enough, and did it itself in 6 to 9 requests.

So the order now says to hand off before reading the code: the worker reads it, and the task is written from the request as given. Claude followed that **24 of 24 times**.

What it measured

All 52 runs below were correct.

On click, one batch:

Feature and coverage	Route as released	+ verified report	+ hand off before reading
Handed off	6 of 8	5 of 8	8 of 8
Main-model requests	5.9	5.0	2.6
Cost per job	\$0.53	\$0.47	\$0.38

The last column against the first: cost -28% [-42%, -10%], main-model requests -55% [-68%, -36%], time -12% (not shown).

Corrected 16 Oct: worker costs were undercounted. A worker's transcript records its output as each response began; counted in full, from Claude Code's own totals, every cost with a worker in it rose. Here the three columns were \$0.50, \$0.44 and \$0.34, and the saving -33%; below, the feature with Sonnet was \$0.43 (-57%) and the tests \$0.27 (-40%) (CORRECTION-1016.md).

On TypeScript, against the pack alone in the same batch:

	Pack alone	Route, Sonnet worker	Route, Haiku worker
Writing tests: cost	\$0.45	\$0.29 (-36%)	\$0.24 (-46%)
Writing tests: main-model requests	6.2	2.0	2.0
A feature: cost	\$0.99	\$0.48 (-51%)	\$0.44 (-55%)
A feature: main-model requests	11.0	2.0	2.8
A feature: time	214 s	125 s (-42%)	178 s

Two requests is the floor: one to hand over, one to answer. Writing tests reached it everywhere; the feature reached it with Sonnet as the worker.

Haiku or Sonnet. On writing tests Haiku was cheaper (-16%). On the feature it needed almost twice the steps (12 against 7), cost about the same (8% less, within the runs' spread: [-24%, +8%]), took longer, and twice left points open that Claude then had to settle. One of them was worth it: the task's example contradicted its own rule about week 53, and Haiku said so. Build work stays with Sonnet, for the steps, the time and the open points, not for the cost.

A classifier that hesitated

The first TypeScript feature runs never handed off. route had asked ScaleDown what the prompt was, and it said "feature" with a score of **0.66**, under route's threshold of 0.7: a long, detailed specification read less

like "a feature" than a one-line request does. TypeSafe's Jev, asked the same, said "feature" with a probability of **1.0**. On our 25 calibration prompts both were right 50 times out of 50; Jev answered in 0.25 s against 1.15 s. route now takes Jev when its key is set.

What this does not show

- Four runs per task and condition. The direction is the same on every task; the sizes have wide ranges.
- **Time improved less than cost** (−12% on click, −42% on the TypeScript feature): the worker now does the reading, in its own requests.
- **Handing off before reading removes Claude's own look at the code.** Every prompt in our benchmarks is a complete specification. On a vague request ("make the export faster"), a task written without reading the code may be a worse task. We have not tested that. It is the next thing route should ask its classifier: is this request fully specified?

Method & limits

Arms RT (route as in 0.5.0), RTV (with delegate 0.3.0's verified report), RTD (and the order to hand off before reading), RTDH (RTD with Haiku as every worker), Z (the pack alone), all with the person's installed plugins switched off (note 8). Each comparison ran in one batch, so times compare too. Opus 5.5, click 8.5.0 and chrono 2.10.3. Data: `results-route-verify.jsonl`, `results-ts-route.jsonl`, `results-ts-route-jev.jsonl`, `results-unasked.jsonl`; the classifiers in `classifiers/jev-vs-scaledown.json`.

Sources

- TypeSafe AI, Jev documentation and privacy policy.
- ScaleDown, Classify reference.