

Why Fully Agentic Development is **Unsolvable**

Principal-Agent Problems with Dynamic, Unobservable, and Fractal User Intent

Rex W. Douglass, PhD

October 2026

rexdouglass.com · rexdouglass@locallyoptimallabs.com

Developers Want Fully Agentic Development

State an open-ended problem; agents solve it, start to finish, without you.

“Towards self-driving codebases” Cursor, 2026

“... work in parallel on a shared codebase without active human intervention” Anthropic, 2026

“... a software product with 0 lines of manually-written code” OpenAI, 2026

“Human attention is a constant bottleneck ... people are ‘bottlenecking the agents.’” Anthropic, 2026

Unfortunately, social science has known that perfect delegation is impossible, for many reasons, including:

- **User Intent is Fractal** The user doesn't want just one thing, they want many competing things with increasingly specific tradeoffs that aren't fully specifiable a priori.
- **User Intent is Latent** Even if the user had a perfectly specified intent, it's impossible to fully communicate it without actually building the entire project to completion by hand as the blueprint.
- **User Intent is Dynamic** Even if user intent was fully specified and perfectly communicated, it would change the second the user learned something new during the development. Constructed preferences: people often don't hold a real preference until they actually face the choice.
- **Agent Performance is Unverifiable** The more the user needs to delegate to an expert, mechanically the less able they are to check their work.

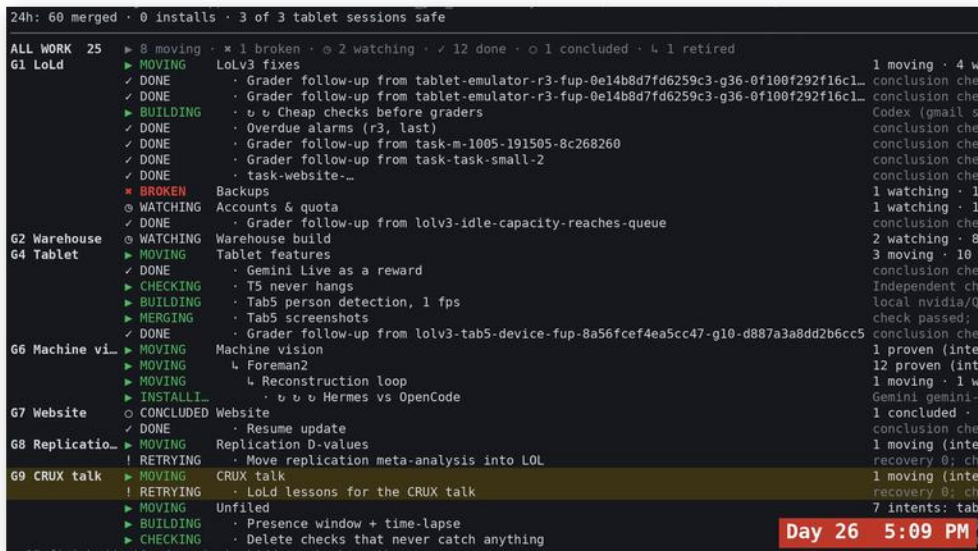
The success and failure of agentic systems can be understood by how they tackle each of these four problems.

In Practice, All Agentic Systems Are Really Optimizing 3 Trade-Offs

- 1 The human's time, against how close the result gets to what they want.
- 2 The computational cost of each attempt, against its chance of success.
- 3 How far to go beyond the user's expertise and ability to verify.

This talk will illustrate each using my own home Jarvis-like agent orchestrator with a case study of one of my more fun projects.

Home Jarvis: one day on its dashboard (video)



The screenshot shows a terminal window with a dark background. At the top, it displays system status: '24h: 60 merged · 0 installs · 3 of 3 tablet sessions safe'. Below this, there's a section titled 'ALL WORK 25' followed by a list of tasks and their progress. Tasks include 'G1 LoId', 'G2 Warehouse', 'G4 Tablet', 'G6 Machine vi...', 'G7 Website', 'G8 Replicatio...', and 'G9 CRUX talk'. Each task has a list of sub-tasks with status indicators like 'MOVING', 'DONE', 'BROKEN', 'WATCHING', 'BUILDING', 'CHECKING', 'MERGING', 'INSTALLING', 'RETRYING', and 'UNFILED'. The bottom right corner of the terminal shows 'Day 26 5:09 PM'.

How it works

Me
at a terminal, or on my phone

A Head
Claude Opus 5.5; turns what I say into intents

A State Machine
intent → code → PR → merge → deliver

An Optimizer and Resource Allocator
14 seats: Claude, Codex, Gemini, Grok, local GPU

A Memory
sourced research the agents and I can query

39 days of home projects

Messages from me	~2,400
Tokens	~100 B
Model runs	~23,000
Intents	~2,300
Repositories	8
Merged pull requests	1,794
Lines of code added	1.3 M

Outline of the Talk (~25 min)

1 Case Study: A Tablet for Helping a Child Focus

- My Intent: Teach My Son How to Focus
- What I Built: An Automated and Personalized Classroom
- How to Measure Focus? How to Measure Learning?
- Evidence of Improvement

2 Four Problems

- Problem 1: User Intent is Fractal
- Problem 2: User Intent is Latent
- Problem 3: User Intent is Dynamic
- Problem 4: Agent Performance is Unverifiable

3 Three Trade-Offs

- Trade-Off 1: The User's Time vs. What the User Wants
- Trade-Off 2: Cost per Attempt vs. Chance of Success
- Trade-Off 3: How Far Beyond the User's Expertise to Go

4 What I Built, and Why

- My Agentic Development Compromise: A Home Jarvis
- Why a Head
- Why a State Machine
- Why an Optimizer and Resource Allocator
- Why a Memory

5 An Implied Research Agenda

- Efficiently and Correctly Soliciting Intent
- Measuring the Fractal and Dynamic Level of Intent
- Measuring the Cost of Verifiability

My Intent: Teach My Son How to Focus

Build a system that solicits focus, on realistic assignments/settings, measures performance, and scientifically tests interventions.

Realistic assignments

- Handwriting
- Match the school's letterforms
- Sight words tracked
- Worksheet variety
- Read aloud
- Tomorrow's lessons ready
- Write left to right
- Content live, veto
- Am I done? score sheet
- Worksheets
- Play buttons for long text
- More real math
- Count, don't guess
- Paper opens after parent override
- Sheet elements overlap
- Plan veto in console
- Tap the score to see the list
- Journal every finished sheet
- Name page letter guides

Soliciting focus

- Desk display companion
- Gemini button opens Gemini
- Desk display live progress
- Desk display timer
- Tablet drives desk display
- Volume cap and no caption popup
- Gemini can't reach outsiders
- Gemini Live as a reward
- Desk display page as drawn
- One page on the desk display
- Desk display never hangs

Measuring performance

- Console views
- Recorded desk display frames for detector ground truth
- Small tablet data
- Desk display screenshots
- Presence detect
- Desk display person detection, 1 fps
- Fix the broken export
- Preview templates and audio
- Export that scales
- Watch-me preview on tracing
- Presence window + time-lapse
- Save both cameras as training data
- Desk display person detector research
- Dial in stroke sensitivity
- Console 3 columns
- Console open

Hardware

- We own the tablet
- Wifi ownership
- Ready before he is
- Hear the tablet's audio
- One owner for the tablet
- Tablet stays locked USB-side-up
- One working backup a day, no redundant copies

Testing interventions

- Synthetic learner

Building and shipping it

- One tested build, installed once
- Emulator-first, one big update
- Tablet engineering review
- Release train
- Sim: 3 sheets finished, a desk display face per sheet
- Emulator lane
- Emulator lane boots reliably

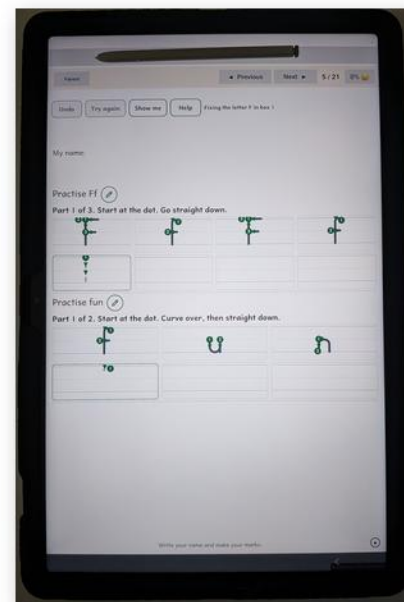
Everything else

- Worksheet speed
- Day follower
- Tablet lock
- Lesson planner
- No break button
- Capture setting

What I Built: An Automated and Personalized Classroom



Worksheets: 20 new ones every day (handwriting, phonics, sight words, math). Shown with a made-up learner and synthetic ink.



Tablet: Samsung Galaxy Tab S10 FE+ and S Pen



Desk display: M5Stack Tab5 (ESP32-P4, camera). Time, a face for each finished sheet, focus and away time

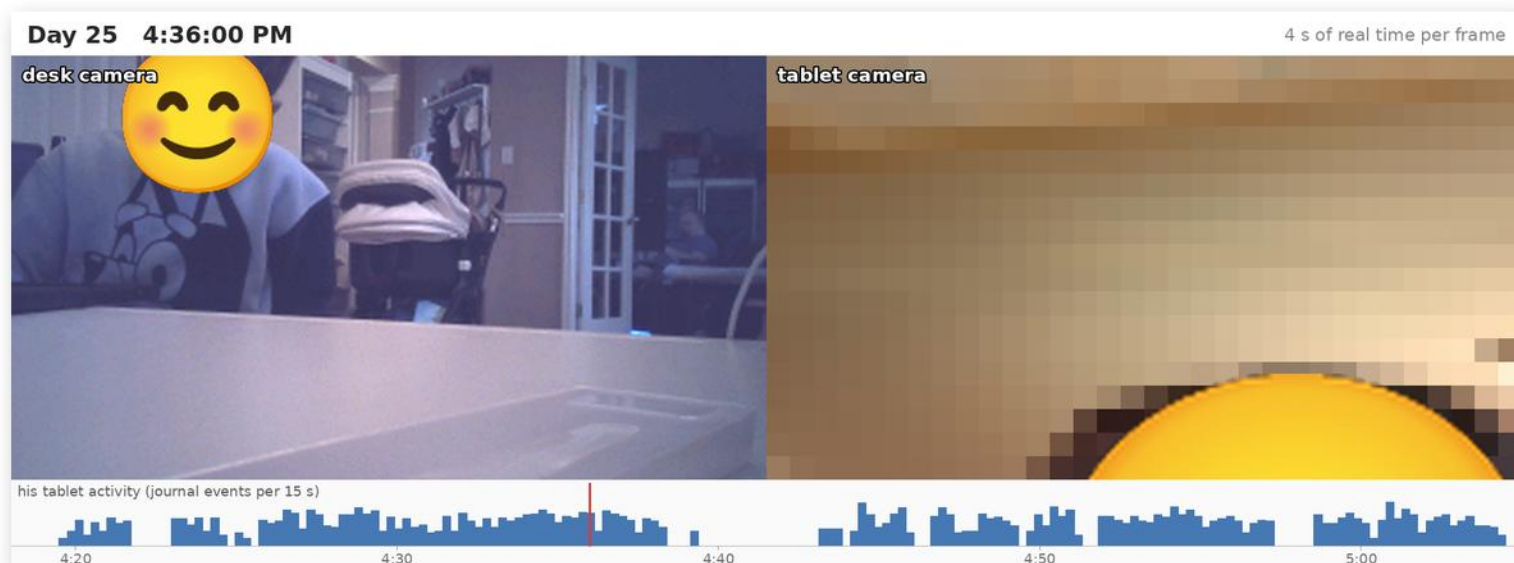


Parent console: every stroke saved and replayable



Auto-generation: templates make each day's sheets from his level

How to Measure Focus? How to Measure Learning?



Two cameras and the tablet's own journal on one clock (a whole session as a time-lapse, his face covered). When he leaves, the bars stop.

Trace addition equations Day 26 +1:44 4x speed

1. Trace $2 + 1 = 3$

$2 + 1 = 3$ $2 + 1 = 3$

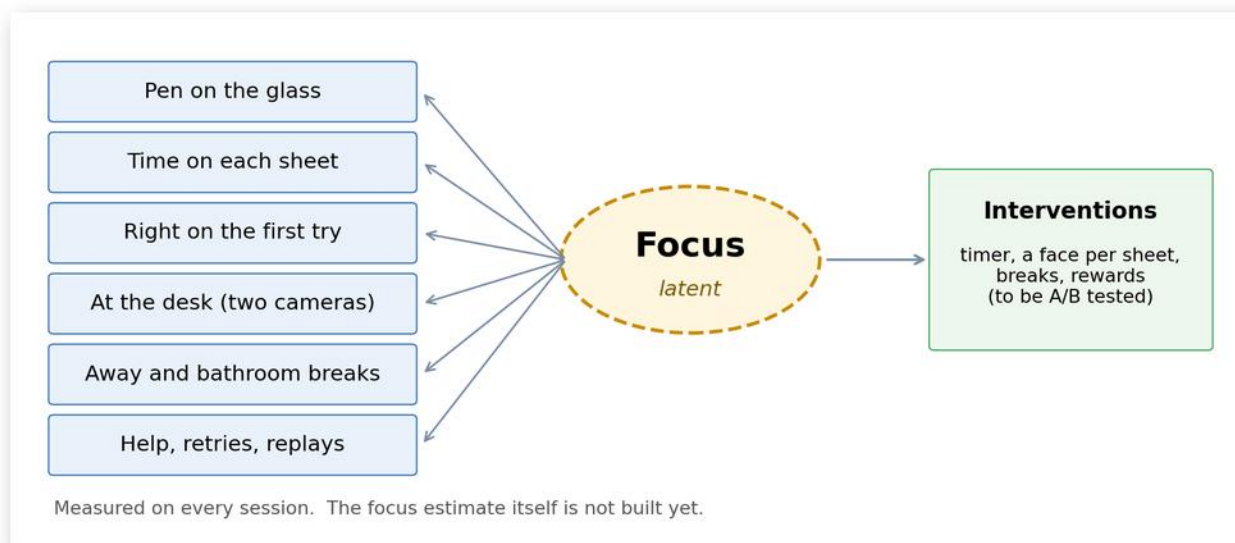
2. Trace $4 + 1 = 5$

$4 + 1 = 5$ $4 + 1 = 5$

3. Trace $1 + 1 = 2$

$1 + 1 = 2$ $1 + 1 = 2$

A worksheet filled in, stroke by stroke, at the tablet's own timing (4x speed)



Performance, handwriting & attention

Date: Day 26 Open

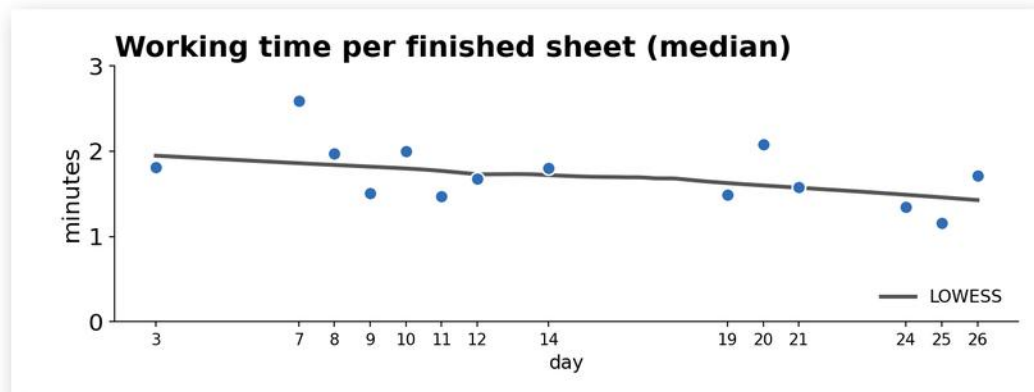
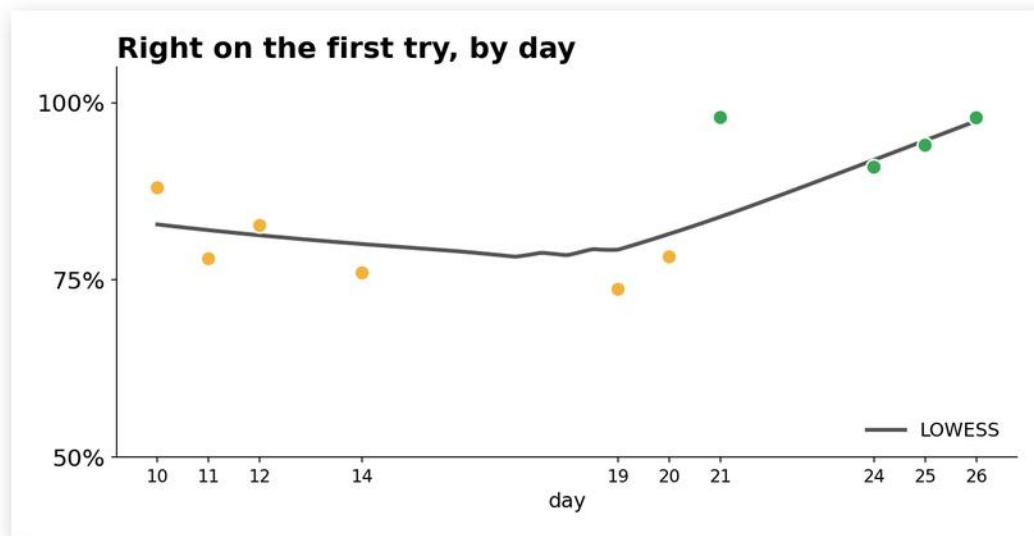
Day 26

Counting and number taps - all saved days

Skill / day	Right on first Check	Picked 2+ numbers	Median time to Check	Counted along after a miss
Trace addition equations	38 of 44	7 of 44	17.8 sec	—
Day 26	6 of 6	0 of 4	40.3 sec	—
Day 25	6 of 6	1 of 6	18.8 sec	—
Day 21	6 of 6	2 of 6	17.9 sec	—
Day 20	5 of 6	1 of 6	25.4 sec	—
Day 19	3 of 4	0 of 4	30.7 sec	—
Day 14	5 of 6	2 of 6	11.8 sec	—
Day 12	6 of 6	0 of 6	7.9 sec	—
One more, one less	52 of 54	4 of 54	8.4 sec	—
Day 26	6 of 6	0 of 6	7.4 sec	—
Day 25	6 of 6	0 of 6	8 sec	—
Day 24	6 of 6	2 of 6	8.3 sec	—
Day 21	6 of 6	0 of 6	7.3 sec	—
Day 20	6 of 6	1 of 6	8.8 sec	—
Day 19	5 of 6	0 of 6	9.8 sec	—
Day 12	5 of 6	1 of 6	10.6 sec	—
Trace subtraction equations	22 of 33	4 of 33	40 sec	—
Day 26	6 of 6	3 of 6	44.4 sec	—

Every answer and stroke saved and timed, per skill, per day

Evidence of Improvement



Writing, the first of each that day

	4	8	3
Day 7			3
Day 8	1	2	3
Day 9	1	8	
Day 11	1	2	3
Day 14	2	8	
Day 19	1	8	3
Day 20	1	8	3
Day 21	1	2	3
Day 24	4		3
Day 25	4	8	3
Day 26	4	8	3

Problem 1: User Intent is Fractal

In any continuous system, between any two details there are more smaller details.

1 Length depends on the ruler

$$L(r) = Fr^{1-D}$$

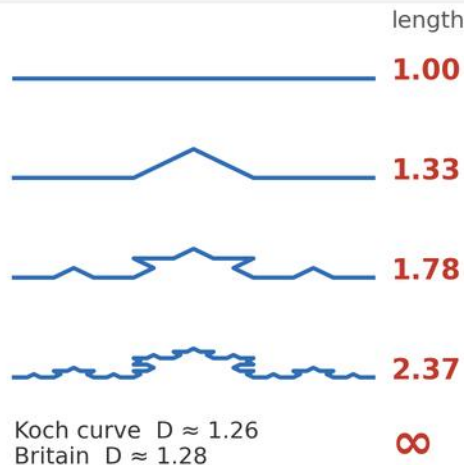
L measured length

r ruler length

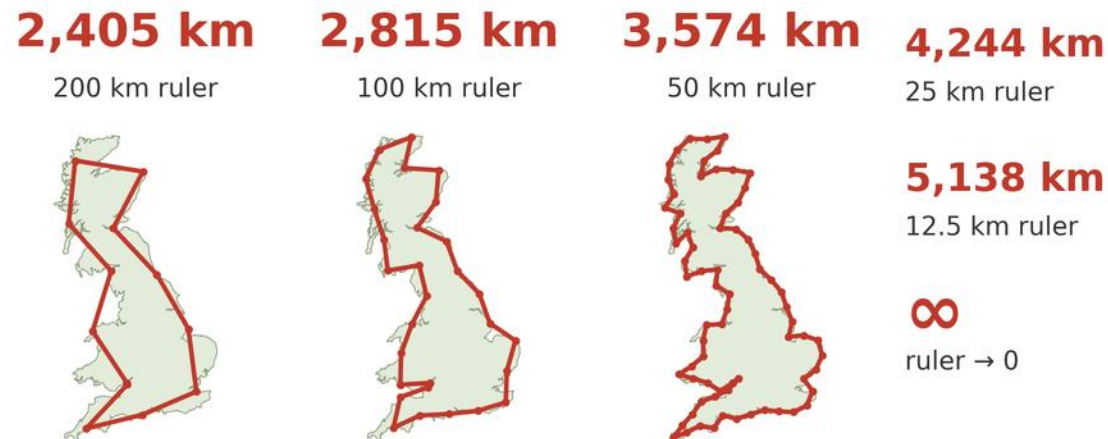
D fractal dimension

$D = 1$: smooth, L settles

$D > 1$: $L \rightarrow \infty$ as $r \rightarrow 0$



2 Coastline of Britain Problem: shorter ruler, longer coast



3 History has a coastline too

Cuban Missile Crisis Timeline of Events

14	discovers nuclear missiles in	starts crisis	705
17	mobilizes troops for invasion of		13
18	meets to deny presence of missiles		21
22	blockades		400
	demands withdrawal		7
	raises nuclear alert		5
	threatens military attack		10
23	OAS statement of support for		7
	meets to deny offensive intention		8
24	respects blockade of		23
	raises nuclear alert		14
25	confronts at the		30
26	offers remove missiles for no invasion		60
27	nuclear missiles in operational		5
	accidentally violates airspace		25
	interdicts submarine		16
	offers withdraw of nuclear missiles in for		115
	missiles in		32
	promises to not invade		119
28	U-2 shot down over		647
	withdraws nuclear missiles in trade for		
	promise to not invade ends crisis		

"We need a yardstick for history."

Douglass et al. 2022

Between any two events in an international crisis, we could always find other important events.

4 So does our intent tree

1x the whole project



3x one branch



5x one intent: the intents it spawned, and theirs

One tested build, installed once

- Test values left in live state
 - Status projection retains a retired producer
 - One-time migration hidden behind an unrelated daily/build trigger
 - A migrated outcome wait also gates unfinished release verification
 - Repeated linear policy lookup inside per-intent gates
- Cache grows with every historical clone
 - A successful background job can escape result collection
 - Successful background results can bypass collection
 - Successful background results can bypass collection
- Refresh throttle applies only after success
 - A synthetic current-release backoff record with due=10**400 and failures=...
- Shared progress file overwritten by one intent
- Shared progress file replaced by one intent



Problem 2: User Intent is Latent

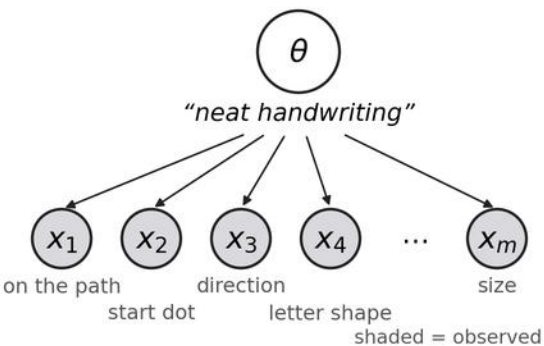
What you want is never observed directly, and it has more dimensions than anything you say.

1 Intent is a latent variable

θ
what you want: latent

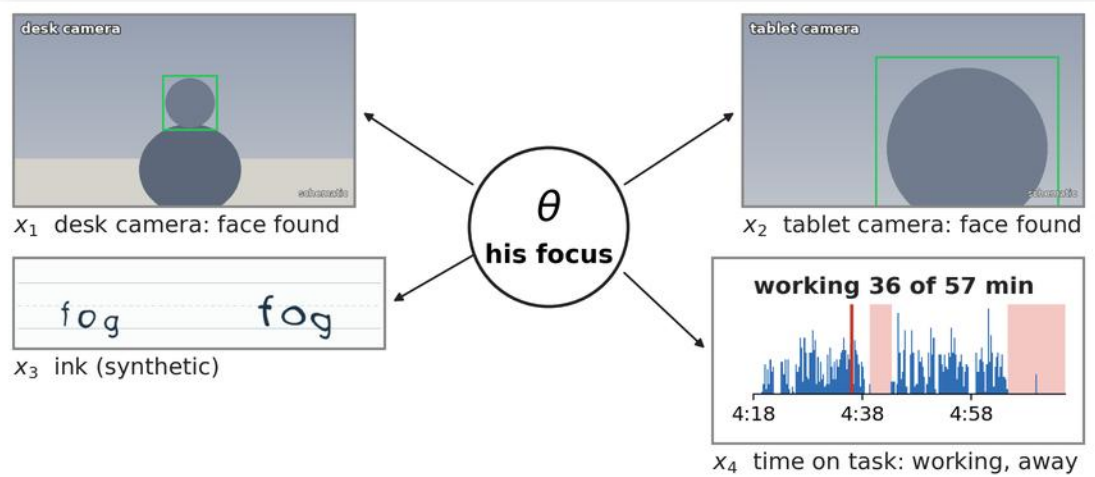
$x_j = f_j(\theta) + \epsilon_j$
the instrument: what gets measured

$j = 1, \dots, m$
which x , which f : unspecified

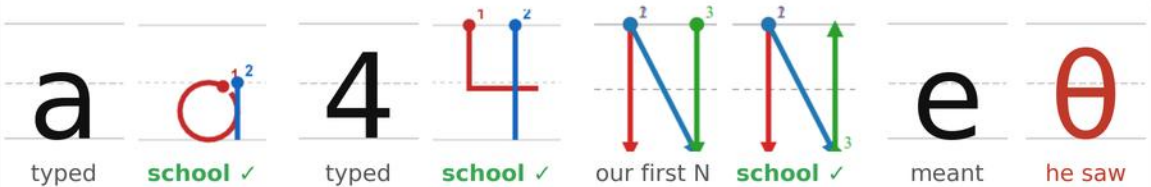


The words name θ . They don't say how to measure it.
The agent has to invent the instrument.

3 No right answer: “childhood focus”



2 No right answer: “neat handwriting”



Every pair is neat handwriting. Only one is what his teacher wanted.
“So he's drawing thetas and getting dinged each time.” (the user, Day 14)

So how is an LLM supposed to guess correctly?

4 And those are the easy ones

Handwriting and focus are the easy ones: we can at least sketch a measurement design. These came up too, and we can't:

When does the tablet become the distraction?

The tool that measures his focus also competes for it.

Which breaks are legitimate?

A bathroom trip and an escape look the same in the log.

Is he learning, or gaming the check?

Tapping every option until it turns green scores as correct.

Behavior and ethics, not penmanship

Tapping the right answer in a story is not the same as not hitting.

Was it him, us, or a bug?

A failed sheet can be any of the three.

Problem 3: User Intent is Dynamic

You don't really know what you want until you start building it, the building is the learning experience.

1 θ is not fixed: building moves it

θ_t what you want at time t : latent

b_t what gets built, and seen

$$\theta_{t+1} \sim p(\theta \mid b_1, \dots, b_t)$$

Seeing the build moves θ .

"The hardest single part of building a software system is deciding precisely what to build."

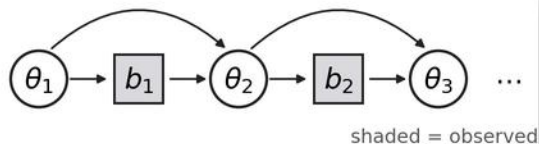
Brooks 1986

"The information needed to understand the problem depends upon one's idea for solving it."

Rittel & Webber 1973

"preferences are often constructed—not merely revealed—in the process of elicitation"

Slovic 1995



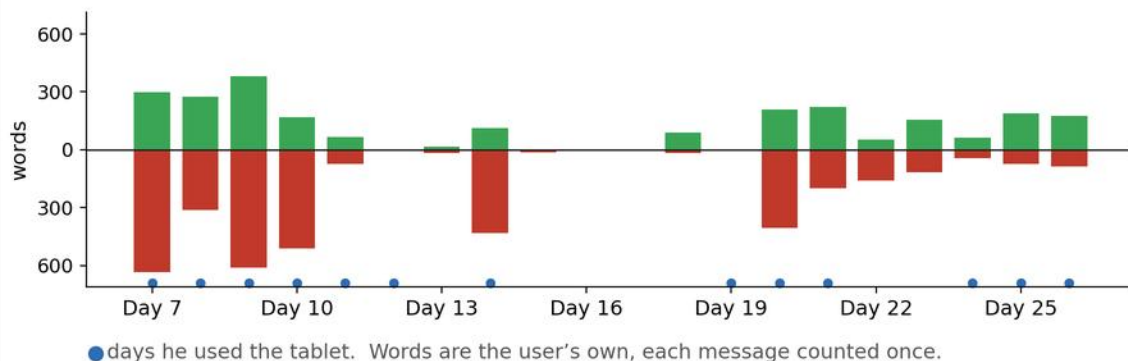
3 Learned watching him use it

Day 9	direction	"he's tracing/writing lines sometimes right to left or randomly"
Day 9	guessing	"he's clicking every option until he gets the green response"
Day 14	the e	"he's drawing thetas and getting dinged each time"
Day 14	finishing	"he rarely finishes everything to exactly 100%"
Day 14	rewards	"he likes talking to gemini through its little android live button"
Day 25	strokes	"some of the letters are throwing him"
Day 26	leaving	"See how much he bounces around and leaves."

2 More words changed the intents than started them

2,471 words started 56 intents

3,718 words in 110 later messages changed them



4 Learned trying to build it

Day 9	a parent UI	"can I fire up the local UI, see the boy's assignments, see templates we use to make those assignments"
Day 9	templates	"instead of having LLMs hand draft every sheet every day. Which is wasteful and hard to review/improve."
Day 26	sign-off	"Some of the early morality lessons were inappropriate LLM slop so I had to start signing off on them"
Day 21	his school	"She want us to change the worksheets to match closer to that font, especially 4 and lower case a"
Day 25	a simulator	"you should be able to just fill out worksheets and see the [desk display] simulation changing"
Day 24	hardware	"it needs to only kiosk with up on the same side as the usb port and not allow rotation"

No plan survives contact with the enemy.

after Helmuth von Moltke

Problem 4: Agent Performance is Unverifiable

Agentic work can only be trusted as far as the user can check it.

1 Delegation is a principal-agent problem

You hand a decision to an agent that knows more than you and acts on its own objective, then see only a noisy signal of what it did.

you want $a^* = \operatorname{argmax}_a U(a; \theta)$

it does $a = \operatorname{argmax}_a U_A(a; \theta_A)$

you see only $s = v(a) + \varepsilon$

Here, you are the principal and the agentic system is the agent.

“full observation of actions is either impossible or prohibitively costly”

— Holmström 1979

Also: Ross 1973; Jensen & Meckling 1976; Akerlof 1970

3 It can't be trusted to test itself

The agents wrote 6,546 tests for their own code.

What 4,258 of them did over 2,450 runs:

2,124 never failed

1,670 failed, caught nothing

464 caught a real bug

Many meaningless tests:

- checks that a line of code exists, not that it runs
- checks that a word appears in the source
- checks the emulator, not the real tablet

2 You can't check the code yourself

27 days to build.



45 working days just to read.



179,200

lines of code

9

languages

22

libraries and toolchains

4 Its preferences, if any, aren't yours

Its wants were trained in by its lab, not by you.

U_A = whatever RLHF, constitutions and safety rules taught it to want

Ouyang et al. 2022; Bai et al. 2022

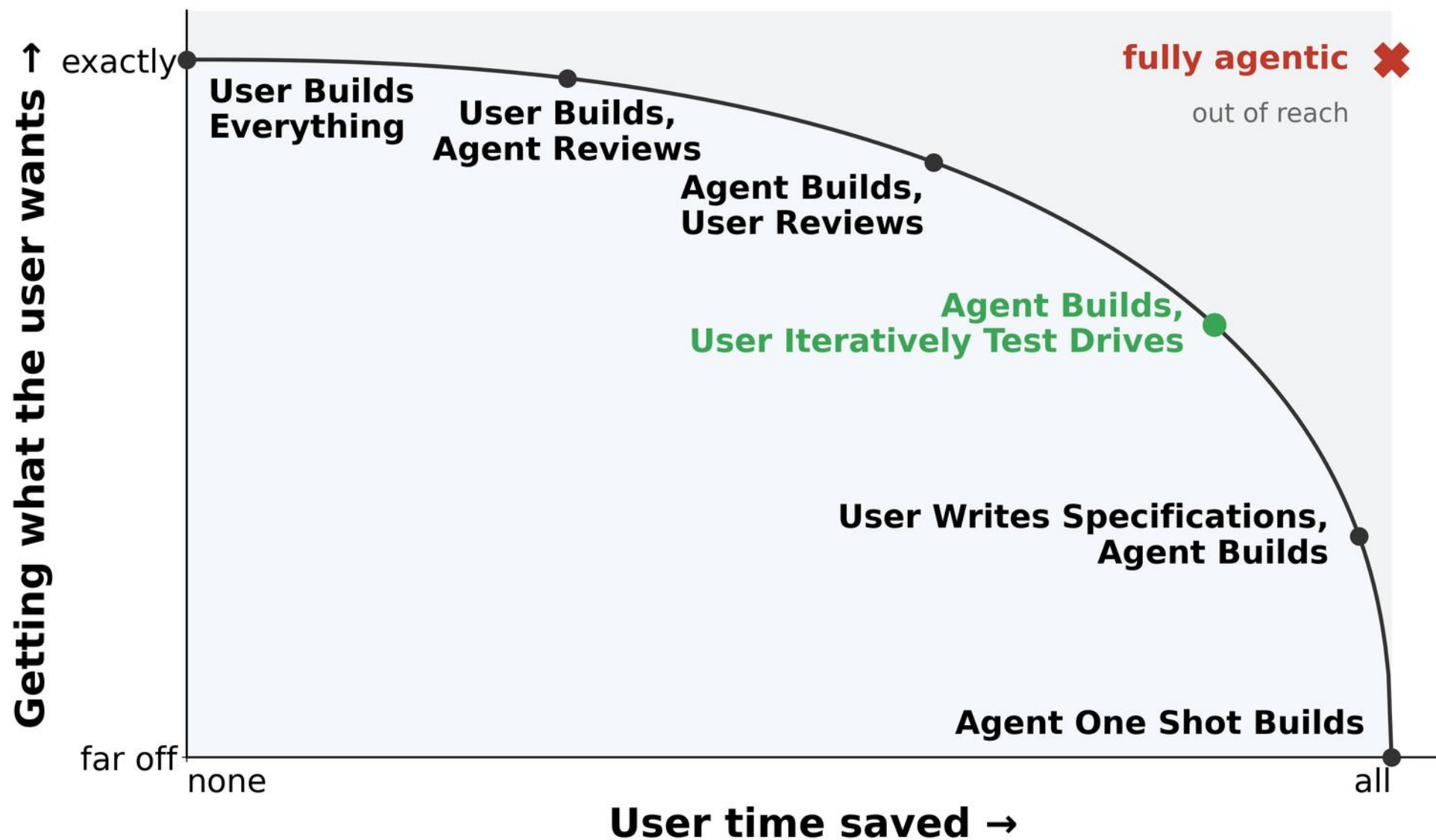
We wanted: his worksheets ready every day

It kept doing instead:

- re-backed up the tablet before each of ten daily bug fixes
- made me use each feature myself before it would call it working
- stopped work whenever GitHub was unreachable

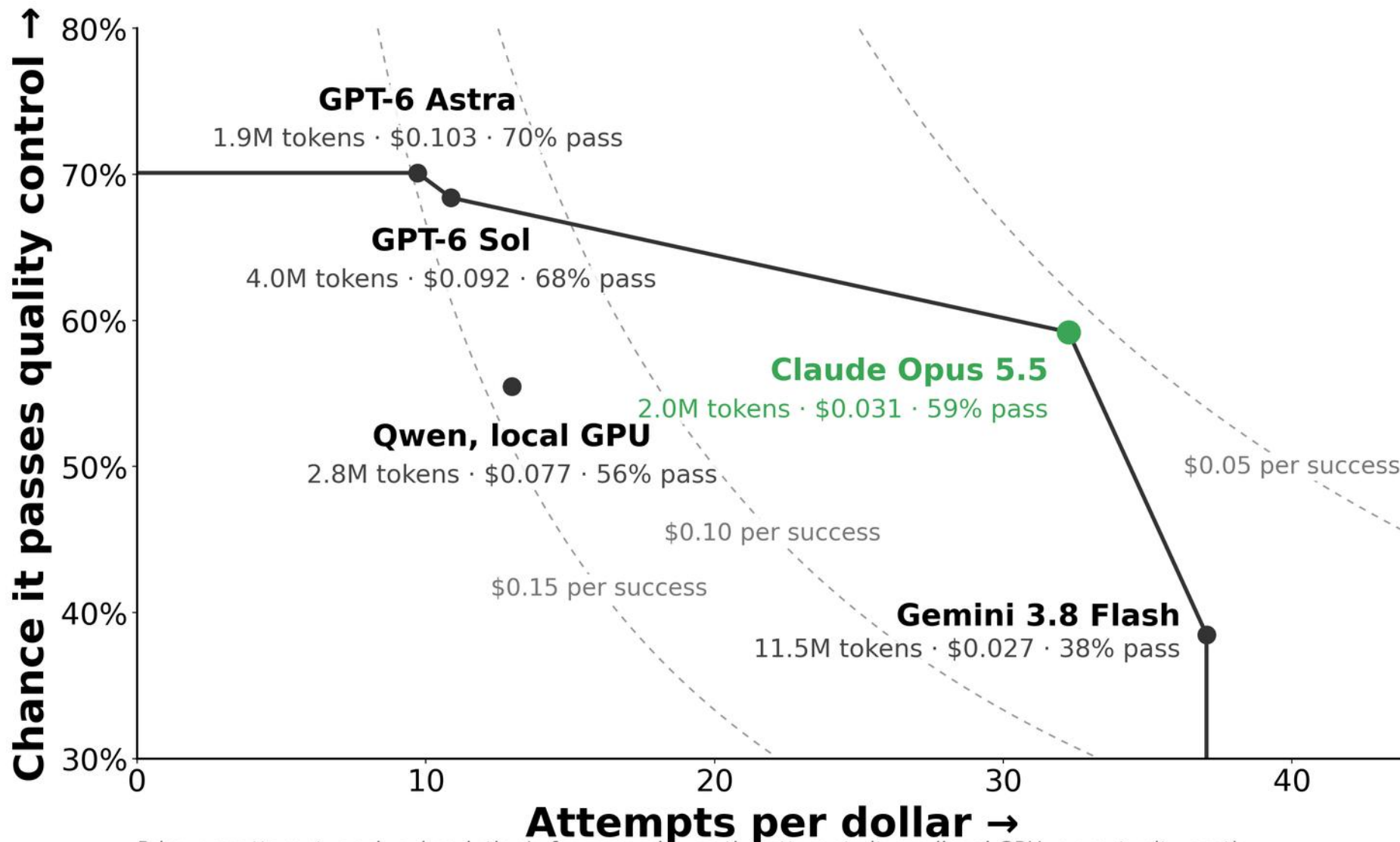
Trade-Off 1: The User's Time vs. What the User Wants

Not whether, but how much: the user trades time saved against how close the result gets.



Trade-Off 2: Cost per Attempt vs. Chance of Success

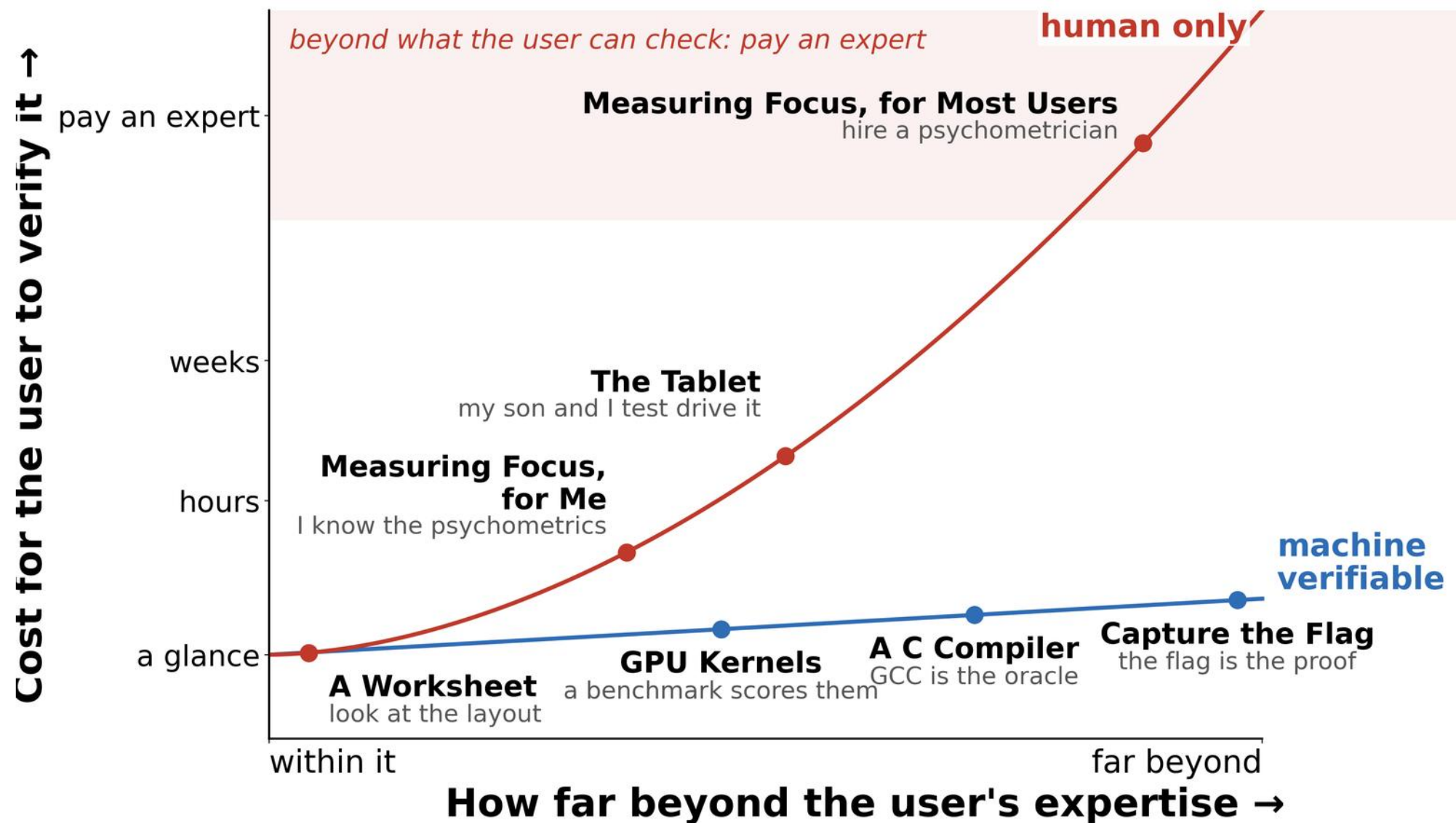
Delegating to an agent isn't free: less guidance saves user time but costs more search.



Price per attempt: each subscription's fee spread over the attempts it ran (local GPU: opportunity cost).
Tokens: median per attempt. The system's own records, Days 7-26.

Trade-Off 3: How Far Beyond the User's Expertise to Go

Some intents are cheap for the user to verify; others cost weeks of their time, or an expert they must hire.



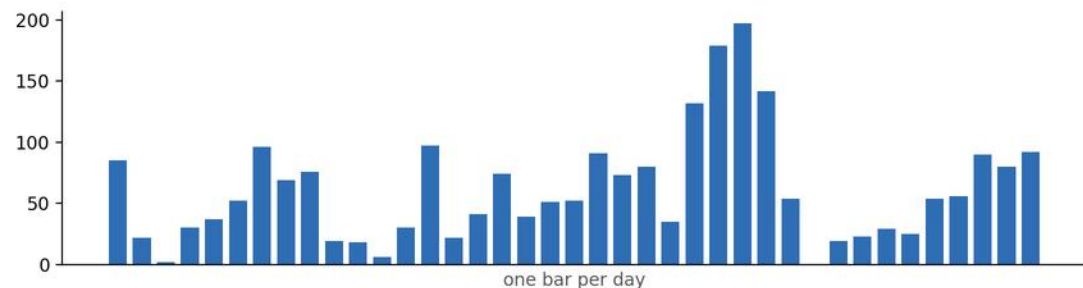
My Agentic Development Compromise: A Home Jarvis

One person's agentic development system, run from home.

1 A Head

Claude Opus 5.5 in a terminal; I drive it from the desk or my phone. It saves every message, splits it into asks, and files them as intents.

~2,400 messages from me, 39 days



3 An Optimizer and Resource Allocator

Every 30 seconds, for each job, it picks:

$\min_a \$ (a) / P(\text{done right} | a)$, a = provider, model, effort

subject to

- each provider's 5-hour and weekly quota, before it resets
- how many jobs each can run at once
- the workstation's CPU and memory

8

providers

40

models

5

effort levels

2

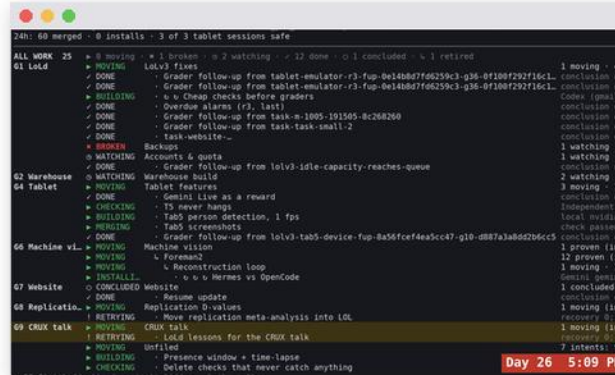
quota windows
each

~23,000

model runs

Quota is perishable: a window that resets unused is capacity lost.

2 A State Machine



(the dashboard, 2 PM to 10:30 PM today: video)

intent → code → pull request →
merge → deliver

~2,300 intents
1,794 merged pull requests

4 A Memory

Document store

15,111 sources captured byte for byte, each with its hash and usage rights.

Knowledge graph

Every claim stays tied to a verbatim quote in its source; ~280 database schemas.

Fermi estimates

Break a number nobody publishes into sourced pieces, then update it.

"How many people used the internet in 2025?"

Fuzzy list enumeration

Find the members of an open-ended list and estimate how many are still missing.

visual applications: 92 found in 8 rounds

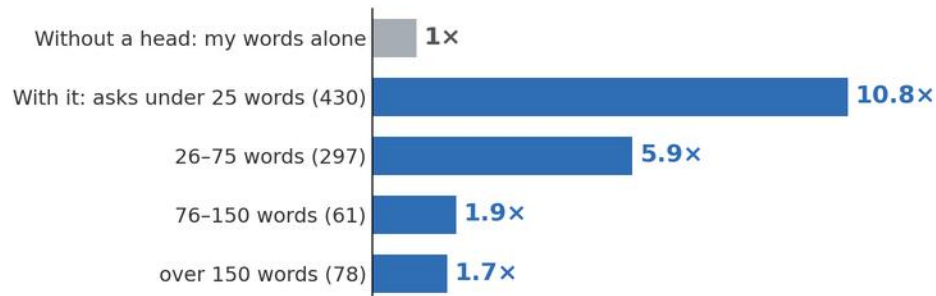
Plus Bayesian models over the evidence. Agents and I query all of it over MCP.

Why a Head

Each quadrant is one of the four problems: what it is like without a head, and with one.

1 User Intent is Fractal

I talk at my level; the head writes the detail below. The shorter I spoke, the more it filled in.

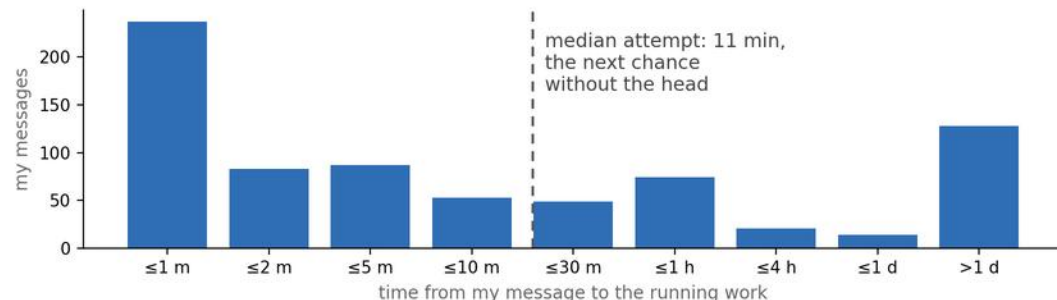


A short ask became about eleven times as many words of detail, all of it inferred for me.

866 intents, Days 6-26: my verbatim words 45,596; words written for me 217,734 (4.8x). Median per intent.

3 User Intent is Dynamic

When I change my mind, the head puts my words into the work that is already running.



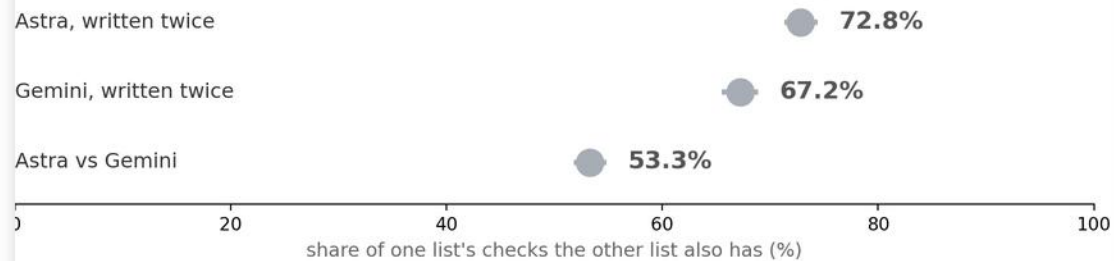
A change of mind reaches running work in a median 3.3 minutes, not at the end of an attempt.

746 routings of my messages, Days 10-27. Attempt length from 8,251 worker attempts, Days 7-26.

2 User Intent is Latent

Without one written checklist, each worker guesses what done means; the head writes one and keeps it.

With a head: one list, written once, read by every worker and grader



Without one written list, two models reading my words agree on only about half of what done means.

Observables written independently for 434 intents, Day 24; overlap judged by a third model; 95% intervals.

4 Agent Performance is Unverifiable

The head checks the real device instead of trusting the machine's own "done".

Without the head: the machine's check (emulator)

3 of 3 recorded

With the head: read on the real tablet

0 of 28 recorded

The machine's check said every sheet was recorded; the tablet had recorded none of his 28.

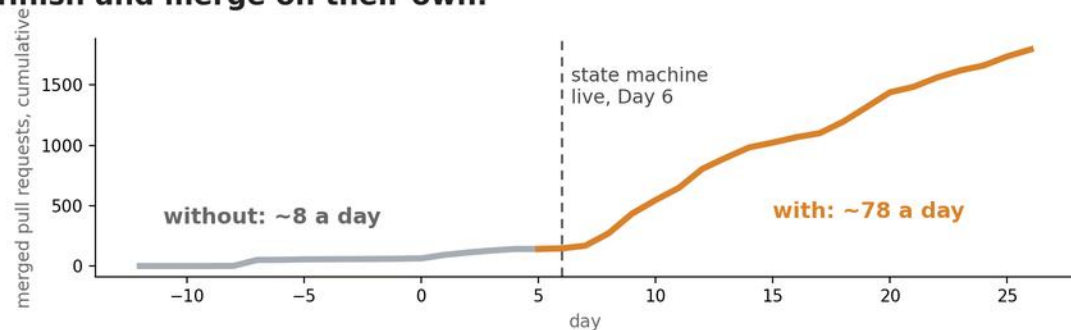
Days 25-26: emulator check (3 of 3 sheets) against the tablet's own record (0 of 28).

Why a State Machine

Each quadrant is one of the four problems: what it is like without a state machine, and with one.

1 User Intent is Fractal

The state machine cuts the work into pieces that can each finish and merge on their own.

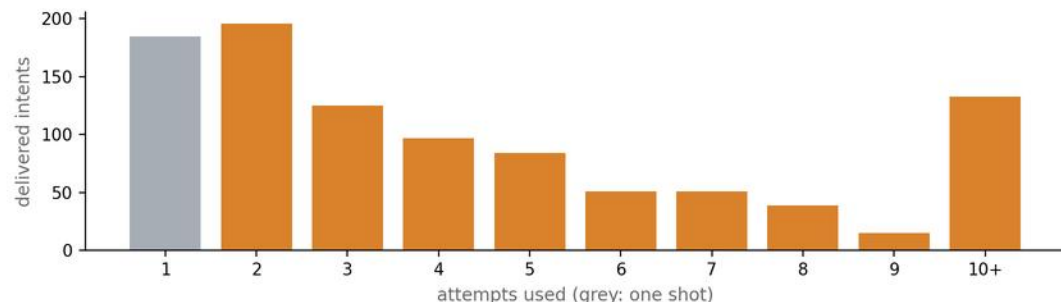


Once every piece had its own state, merged work went from about 8 to about 78 pull requests a day.

Merged pull requests, all 8 repositories, Day -12 to Day 26 (1,794). The head went live the same day; models improved too.

3 User Intent is Dynamic

Short loops of build, look and correct let each piece move with what I learn.



One shot would have delivered 19% of what the loops delivered; 14% needed ten tries or more.

976 delivered intents, Days 6-26, by number of attempts used.

2 User Intent is Latent

Each piece stores my words verbatim, so no attempt starts from a summary of what I said.

Without a state machine: my words in one long conversation

summarized 23 times

With it: my words stored with the piece

never summarized

One long conversation loses my words to summaries; a stored piece keeps them for every attempt.

This head session: 23 summaries so far. Workers: 369 summaries in 8,252 attempts, each restarting from the stored words.

4 Agent Performance is Unverifiable

Workers may not merge their own work: every “done” goes to a grader first.

Without the gate: the worker says it is done

4,971 claims

With it: the grader agrees

2,156 (43%)

More than half of the workers' “done” claims were sent back before they could merge.

4,971 graded worker “ready” exits, Days 7-26. A grade is a guess too: two graders agree 58-73% (164 changes).

Why an Optimizer and Resource Allocator

Every 30 s, for each job, pick $a^* = \operatorname{argmin}_a \$(a) / P(\text{met} \mid a)$ among 82 priced seat $\times$ model $\times$ effort options, within every limit in 1.

1 User Intent is Fractal

Detail costs labor; the allocator turns every subscription into labor at once.

Without it: the busiest single provider (OpenAI)



With it: every provider at once (OpenAI, Gemini, Claude, local GPUs)



Using every provider at once gave about 2.6 times the labor of the best one alone.

Model runs a day, Days 7–26 (19,116 runs); Gemini alone 269, Claude alone 238.

3 User Intent is Dynamic

It re-plans every 30 seconds, on forecasts of how long each attempt will take.

Without it: one fixed guess (2 hours)



With it: learned forecasts



Learned forecasts land within a factor of two 25 times as often as a fixed guess, still only half the time.

Worker attempts, Days 7–26: fixed 2-hour guess 1,963; learned forecast 6,064 (median actual/forecast 1.01).

2 User Intent is Latent

Which model will satisfy me is hidden until it tries, so after a failure the allocator can move the work.

Without it: retry on the same model (1,752)



With it: moved to a different model (1,107)



After a failed attempt, a different model passed about a third more often than the same one again.

Consecutive graded attempts on the same intent after a fail, Days 7–26. Observational, not randomized.

4 Agent Performance is Unverifiable

Checking costs labor too, and the allocator budgets for it before a merge, not after.

Without a check: repairing after a bad merge



With it: one check before the merge



One check before a merge costs about a tenth of repairing a bad merge afterwards.

Means: 5,985 checks, Days 7–26; 176 complete repairs after a bad merge (grader study, Day 24).

Why a Memory

Each quadrant is one of the four problems: what it is like without a memory, and with one.

1 User Intent is Fractal

The memory keeps every round of an open-ended search, so it can count what it has and estimate what is missing.

Without a memory: one search pass

11 found

With it: 8 rounds, kept and de-duplicated

92 found

Kept across rounds, the search found eight times more and can bound what is still missing.

Visual applications, Day 12: 11 found in round 1; 92 after 8 kept rounds; total estimated 66–11,396.

3 User Intent is Dynamic

Sources are captured once and kept, so a change of mind can reuse evidence instead of searching again.

Without it: public web search

9 of 12 found

With it: the memory

0 of 12 found

Keeping evidence works; finding it again does not yet: plain web search beat the memory.

12 lookup questions, Day 17. A small test; web search stayed the default.

2 User Intent is Latent

It measures numbers nobody publishes by breaking them into sourced pieces, each with a range.

A calibrated 50% range would hold the truth

15 of 30

The memory's 50% ranges held it

29 of 30

Its 50% ranges held the truth 29 times in 30: honest, but far too wide to call calibrated.

30 held-out economic series, Day 10. The 90% ranges held it 30 of 30 (27 expected). A bare model guess has no range to check.

4 Agent Performance is Unverifiable

Every claim keeps a verbatim quote from a saved source, so checking it is reading the quote.

Without a memory: quotes confirmed word for word

10 of 30

With it: a quote is kept only if it matches the saved copy (45,030 passages)

Without a saved source, two thirds of the agents' quotes could not be confirmed.

30 quotes cited by agents with no saved source, Day 8: 10 word for word, 1 close, 11 not on the page, 8 page would not load.

An Implied Research Agenda

What this case study says we need to learn next.

1

Efficiently and Correctly Soliciting Intent

User Intent is Latent

Two models reading the same request agreed on only about half of what done means; I spent more words changing what I wanted than stating it.

What is the fewest questions that pins down what the user wants?

2

Measuring the Fractal and Dynamic Level of Intent

User Intent is Fractal and Dynamic

Tablet-app pieces took a median of ten tries to land, and one rule spawned 334 follow-up pieces in two and a half days.

Can we tell, before building, how deep an intent goes and how fast it will move?

3

Measuring the Cost of Verifiability

Agent Performance is Unverifiable

A check before merge cost a tenth of a repair after one, yet two checks agree barely better than chance, and some intents only a person can judge.

What does checking an intent cost, and who has to pay it: a machine, the user, or a hired expert?

Thank you

Rex W. Douglass, PhD

rexdouglass.com

rexdouglass@locallyoptimallabs.com

Agents can shrink the human's part, never remove it.

Fully Agentic Development is Unsolvable

Extra

References

Mandelbrot, B. (1967). How long is the coast of Britain? Statistical self-similarity and fractional dimension. *Science* 156(3775).

Douglass, R. W., et al. (2022). Introducing the ICB dataset: very high recall and precision event extraction from narratives about international crises. arXiv:2202.07081.

Brooks, F. P. (1986). No silver bullet: essence and accidents of software engineering. *Proc. IFIP*.

Rittel, H. W. J., & Webber, M. M. (1973). Dilemmas in a general theory of planning. *Policy Sciences* 4.

Slovic, P. (1995). The construction of preference. *American Psychologist* 50(5).

Moltke, H. von (1871). Über Strategie.

Jensen, M. C., & Meckling, W. H. (1976). Theory of the firm. *Journal of Financial Economics* 3(4).

Holmström, B. (1979). Moral hazard and observability. *Bell Journal of Economics* 10(1).

Ouyang, L., et al. (2022). Training language models to follow instructions with human feedback. *NeurIPS*.

Bai, Y., et al. (2022). Constitutional AI: harmlessness from AI feedback. arXiv:2212.08073.

Cohen, J. (2006). Best Kept Secrets of Peer Code Review. SmartBear Software.

Lopopolo, R. (2026). Harness engineering: leveraging Codex in an agent-first world. OpenAI.

Anthropic (2026). How employment is changing at Anthropic. Research report, Oct 16.

Anthropic (2025). How we built our multi-agent research system. Engineering blog.

Anthropic (2025). Agentic misalignment: how LLMs could be insider threats.

Carlini, N. (2026). Building a C compiler with a team of parallel Claudes. Anthropic engineering blog.

Yan, W. (2025). Don't build multi-agents. Cognition blog.

Yan, W. (2026). Multi-agents: what's actually working. Cognition blog.

Lin, W. (2026). Scaling long-running autonomous coding; Towards self-driving codebases. Cursor blog.

Willison, S. (2026). Wilson Lin on FastRender: a browser built by thousands of parallel agents.

Cemri, M., et al. (2025). Why do multi-agent LLM systems fail? *NeurIPS Datasets and Benchmarks*.

METR & Redwood Research (2026). OpenAI-Hugging Face incident investigation; OpenAI (2026), The Hugging Face incident and the road ahead.

OpenAI (2026). Misalignment report: using a funded test cryptocurrency wallet without authorization.

Alibaba (2025). Let it flow: ROME, arXiv:2512.24873, §3.1.4 (GPU capacity repurposed for crypto mining).

Shambaugh, S. (2026). An AI agent published a hit piece on me.

ITU (2025). Facts and Figures 2025: Internet use.

Ross, S. A. (1973). The economic theory of agency: the principal's problem. *American Economic Review* 63(2).

Akerlof, G. A. (1970). The market for "lemons": quality uncertainty and the market mechanism. *QJE* 84(3).

Related Work

The human stays in the loop

“not as a swarm of autonomous actors, but as a coordinated system that scales human taste”

Yan, W. (2026). Multi-agents: what’s actually working. Cognition blog.

“Humans steer. Agents execute.” People still set priorities, write acceptance criteria and validate outcomes.

Lopopolo, R. (2026). Harness engineering: leveraging Codex in an agent-first world. OpenAI.

User Intent is Dynamic

Frontier models degrade sharply once intent is revealed, revised or redirected mid-task.

Tack, J., Laban, P., & Neville, J. (2026). LLMs get lost in evolving user intent. Microsoft Research, arXiv:2607.20734.

A router scores a user’s correction as the model failing. Sometimes it is the user learning what they want.

Cursor (2026). How Cursor Router chooses the right model for the task. Cursor blog.

Agent Performance is Unverifiable

An LLM judge agreed with a scripted check 93% of the time, and both encoded the same wrong rule.

Braintrust (2026). Behavior scoring vs output scoring for coding agents.

The human’s time

Developers expected AI to cut task time 24%; it added 19%.

Becker, J., et al. (2025). Measuring the impact of early-2025 AI on experienced open-source developer productivity. METR, arXiv:2507.09089.

A year on, many would not work without AI, and time per task was unreliable once they ran several agents at once.

METR (2026). We are changing our developer productivity experiment design.

Dollars per success

Routing 7% of calls to Opus cut cost 74% for 6 points of accuracy, and did not measurably beat the cheap model alone.

NVIDIA (2026). NeMo Switchyard; LangChain (2026). How many of your agent’s calls actually need a frontier model?

A planner, generator and evaluator built whole apps in 6 hours for \$200; one agent alone took 20 minutes and \$9.

Rajasekaran, P. (2026). Harness design for long-running application development. Anthropic.