

Theo's TypeScript-to-Rust port: Opus 5.5 succeeded by porting Microsoft's Go version against hard targets

Coding Agents Alpha Tracker

2026-10-11

Theo's TypeScript-to-Rust port: Opus 5.5 succeeded by porting Microsoft's Go version against hard targets

By Coding Agents Alpha Tracker • October 11, 2026

Theo's TypeScript compiler rewrite in Rust, Huntley's golden-oracle ports and Berman's game recreation all run the same kind of agent loop: a reference implementation to check against and a measurable goal. Also covered: coordinator-and-thread setups, steering instead of queueing, and new data on whether agent teams pay off.

TS Rust: pick the right reference, set a measurable goal, then work on the environment

Theo released TS Rust, a rewrite of the TypeScript compiler, type checker and LSP in Rust, published as a real npm package. He says agents worked on it for five months. About \$400k of Codex tokens got nowhere, then about \$20k of Opus 5.5 finished it in two weeks. He says he hasn't read a single line of the code [1]. His benchmarks put it at 12.5x faster than TypeScript v6 and about 2x faster than Microsoft's Go-based TSGO. He notes that Bun's `bun check` Rust rewrite is faster still [1].

Three lessons are reusable:

- **Start from the easiest source to port, not the original.** Opus didn't translate the JavaScript. It ported Microsoft's Go version line by line, including a port of Go's standard library and goroutine runtime. Theo says GPT-5.6 Sol and Astra struggled with the direct JS-to-Rust route [1].

- **Make the end state numeric.** The first build was 2–3x slower than Go. Before the change, the agent kept declaring success after reaching full compatibility plus one performance win. The fix was a `/goal` of “minimum 2x faster than the go build. Continue iterating until we get there,” and it ran for two weeks [1].
- **Your job is the environment.** His prompts were short: read `AGENTS.md` and the accountability/audit files, keep multiple workflows and subagents running in parallel, and “I trust your judgment fully” [1]. The real work happened elsewhere. He fixed OS-level out-of-memory crashes that were knocking his effort setting down to medium, and he gave the agent SSH access to two spare VPSs when it ran short of compute. Separate audit threads tracked blockers and recurring failure modes so agents stopped stepping on each other [1].

On cost, he estimates about \$24k at full API prices, or roughly 925–983% of one account’s weekly limit spread across six Claude subscriptions. With a free reset, he puts the effective cost at about \$500 [1].

Golden oracles and verifiable loops

Geoffrey Huntley writes up a conversation with **Justin Cormack** that applies the same idea to infrastructure. Cormack had an agent write `mkfs.xfs` in Rust with byte-for-byte identical output and every flag supported. It reverse-engineered the on-disk formats one at a time, with tests across block sizes, in a few hours. The recipe: treat the original tool as the oracle, generate output with both at different sizes, diff them, port the tests, and automate it [2]. Huntley also used a loop to port Go’s Stripe library to OCaml for a unikernel [2]. When Cormack had an agent prototype an OS, it put the tests in Nix flakes. NixOS machine tests can spin up a fleet to check network rules against the app [2].

On language choice, Huntley says OCaml’s `.mli` interface files are “really efficient context” for agents and that its compiles are fast. Compile time limits a loop’s throughput: four agents building Cormack’s roughly million-line Rust S3 clone compete for disk and CPU, so “you end up spending more on fast machines than on tokens” [2].

Matthew Berman is running the same loop on Super Mario World for the Mod Retro. GPT-6.1 Sol writes code, compares the result with what the game should look like, and iterates. It has been going for more than four days, and he expects a few more [3]. He points to `/goal` and `/loop` in Claude Code, Codex and Cursor as the way to set this up [3].

Coordinators that delegate

Riley Brown’s starter prompt for Claude Code Projects asks the agent to rank work by importance, scan your email, Slack and other connections for possible threads and automations, and run an “Organizer” thread that keeps an editable

to-do list [4]. He keeps the coordinator in the cloud on Opus 5.5 so it is always available. Some threads run locally on a MacBook and two always-on Mac minis, and routine threads use Haiku or Sonnet [4]. His website project has grown to more than 35 threads. He talks to the coordinator from his phone, and it routes each request to the right named thread [5]. Grok bot’s “primary bot” works the same way: asked to polish dark mode and open a PR, it passed the task to a dev agent [5]. SpaceX says Grok bot will now send hard tasks to other backend models, including Opus 5.5 [5].

Kent C. Dodds merged 104 PRs in one day and published the GitHub search so others can judge the work [6]. By bedtime he was at 110, with six agents still running, and he hadn’t opened his computer that day [7]. He credits Grok bot as a major part of the workflow [8]. His biggest friction now is GitHub notification email [9]. **Huntley** says he recreated Gitpod in under 10 days and plans an open-source, on-prem release [10]. He now provisions workloads on it through Grok bot from his phone [11]. His disclosures say SpaceXSI comps his Grok bot usage [12].

Steer, don’t queue

Theo argues that modern models can absorb a mid-task “steer” message without getting distracted, so he doesn’t queue messages [13]. T3 Code lets you change the default [14]. Kent says Grok bot doesn’t support queueing at all, and it copes when he interrupts with unrelated tasks. He guesses that it delegates everything to subagents [15]. Codex steering is now immediate. Brown notes that steers used to wait for the next tool call, which could waste tokens first [5].

Does fan-out pay?

- **Vals AI** ran GPT-6 Sol and Opus 5.5 on Vibe Code Bench, both alone and as teams. Teams cost 1.8–5.1x more. Only Sol at medium effort improved significantly, by 7.3 points, delegating in parallel along architectural lines. Opus ran sequential waves of about 6.8 subagents with no significant gain [16].
- **LangChain** says sending each Open SWE task to the cheapest model that can handle it cut median cost per task by 64% [16].
- **Claude Managed Agents** dynamic workflows are in public beta (`multiagent_20261001`). They fan out to as many as 1,000 agents per run, and Anthropic advises starting with scoped tasks because token use can be high [16].
- Claude Max and Team now include monthly API credits: \$100 for Max 5x, \$200 for Max 20x, and up to \$500 pooled for Team. They work on any model, in your own code or third-party harnesses [17].

Sources

1. You can rewrite typescript in rust for \$420k
2. unikernels were hard. key word: were.
3. Everything has been solved
4. X post by @rileybrown
5. Claude's New Updates Are Insane... But Grok Bot Is EVEN Bigger
6. X post by @kentcdodds
7. X post by @kentcdodds
8. X post by @kentcdodds
9. X post by @kentcdodds
10. X post by @GeoffreyHuntley
11. X post by @GeoffreyHuntley
12. X post by @GeoffreyHuntley
13. X post by @theo
14. X post by @theo
15. X post by @kentcdodds
16. [AINews] TypeSafe/Jev at >\$100M ARR, \$7.5B valuation 3 weeks after launch
17. X post by @ClaudeDevs