Hoist, on LinkedIn

The LinkedIn post announcing Hoist, mirrored here: what the proxy does, the live turn savings on τ^2-bench retail and airline, and where it fits next to LLMCompiler and TraceCompiler.

october 15, 2026 · Notes

Much of an agent's cost is LLM turns that decide the next call. Many of those decisions are already settled by what the last tool returned: one result names an id, the next call reads it.

Hoist is an MCP proxy that makes those reads for the agent, in the same tool result. It learns by counting, from recorded tool calls, which reads follow which and where their arguments come from. It only reads, and it asks no model.

Live on τ^2-bench retail and airline, Sonnet 5 took ∼20.5% fewer LLM turns and Haiku 4.5 ∼22.4% fewer. How much it saves depends on the domain: there is room where one result points at the next read, and little where each request spells out what to look up.

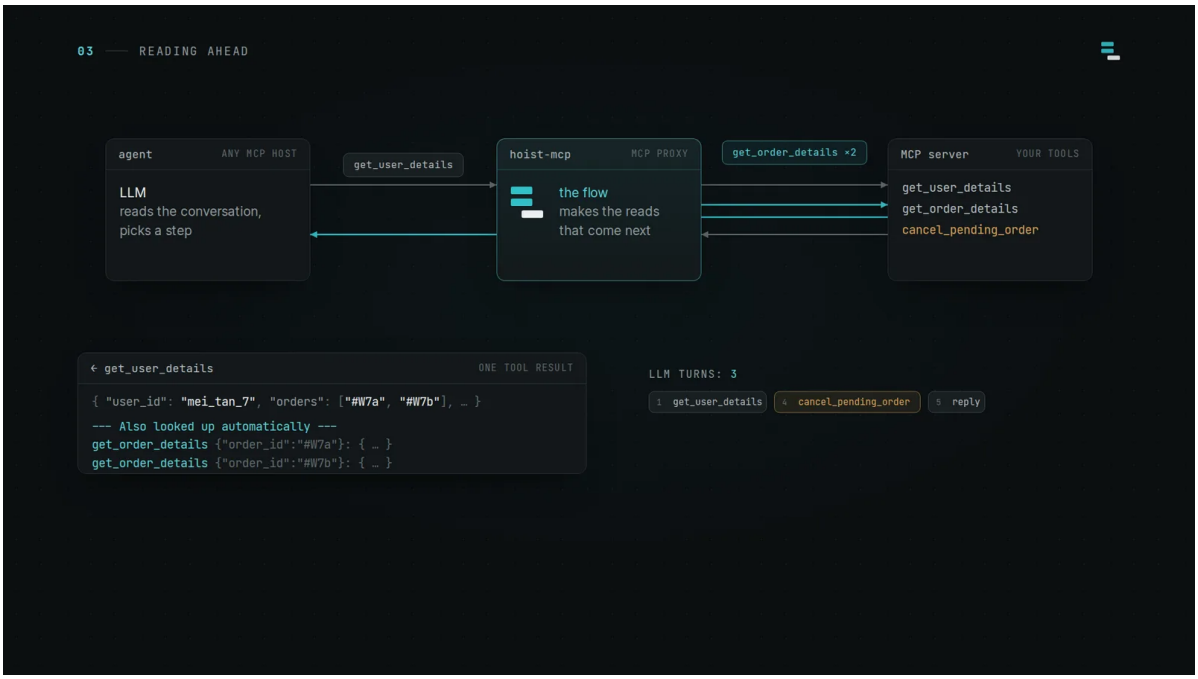
It's my take, from a probabilistic-programming angle, on what projects like LLMCompiler and TraceCompiler aim at. I'll post about the underlying fugue ecosystem (hand-rolled PPL in Rust) at some point soon too. Interested to hear thoughts from anyone compiling workflows from traces or building on a PPL.

#MCP #LLMAgents #ProbabilisticProgramming #OpenSource

Posted on LinkedIn · the repository · the longer write-up · the paper

https://alexnodeland.com/timeline/261015_hoist-on-linkedin/

© 2026 all rights reserved, alex nodeland



the explainer video from the post (3:04). it plays on linkedin.

© 2026 all rights reserved, alex nodeland