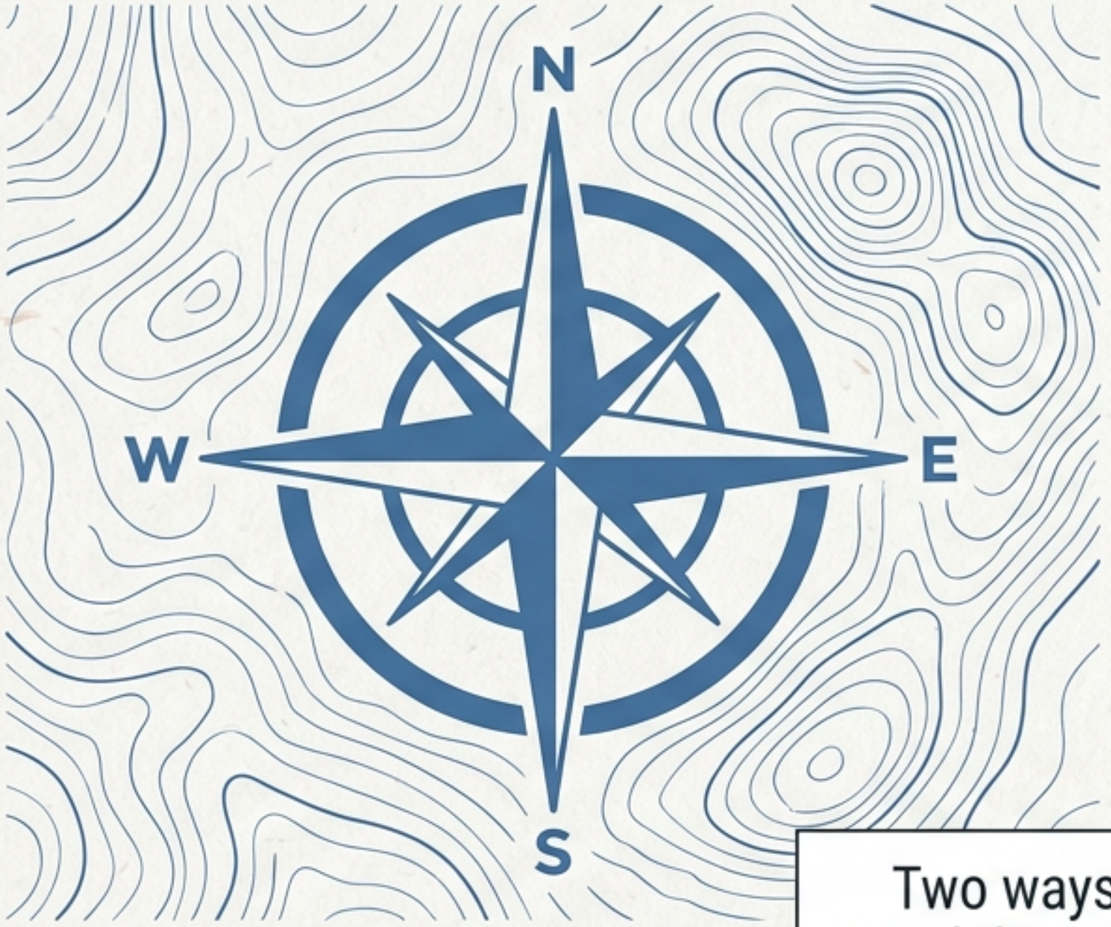
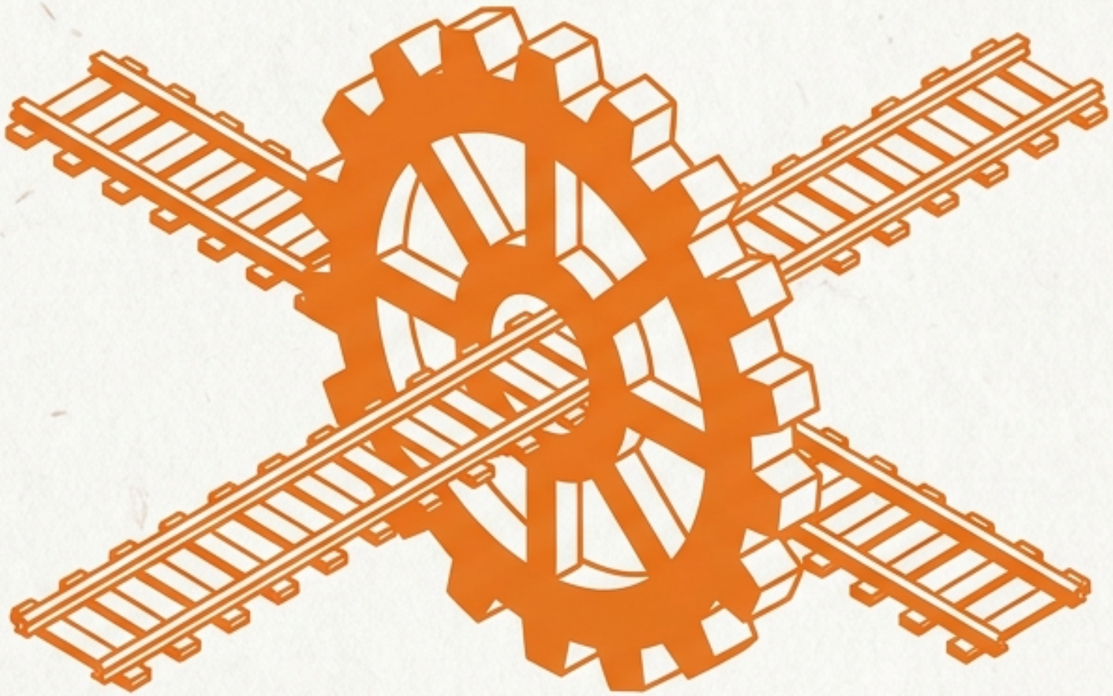
MCP: THE SCOUT



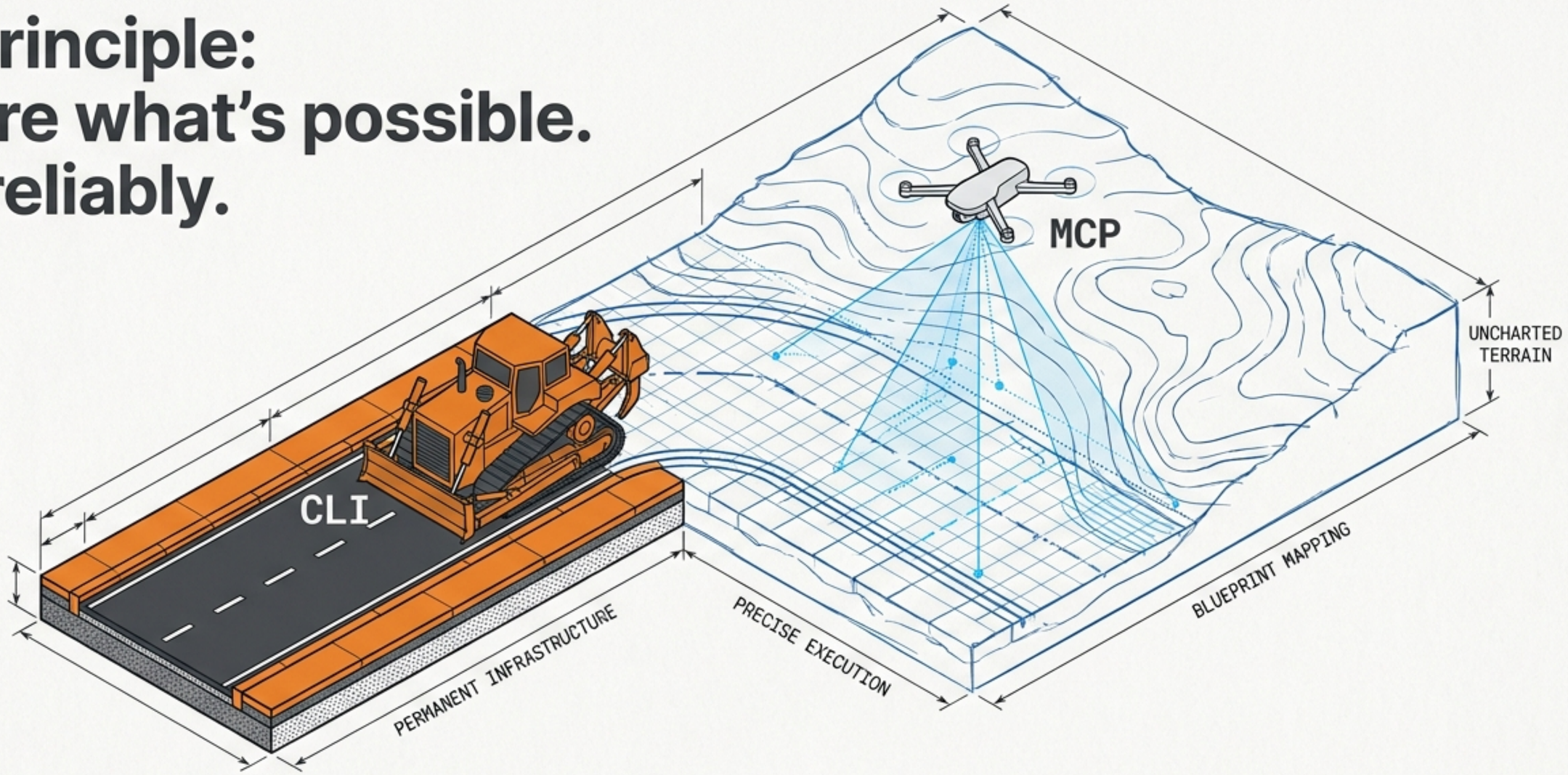
CLI: THE WORKHORSE



Two ways to give an agent capabilities—and one job each.

Based on the architectural model by Bjorn Hansen, 2026.

The Principle: Explore what's possible. Do it reliably.



MCP scouts:

Built to map the unknown and discover tool-shaped tasks.

CLI drives:

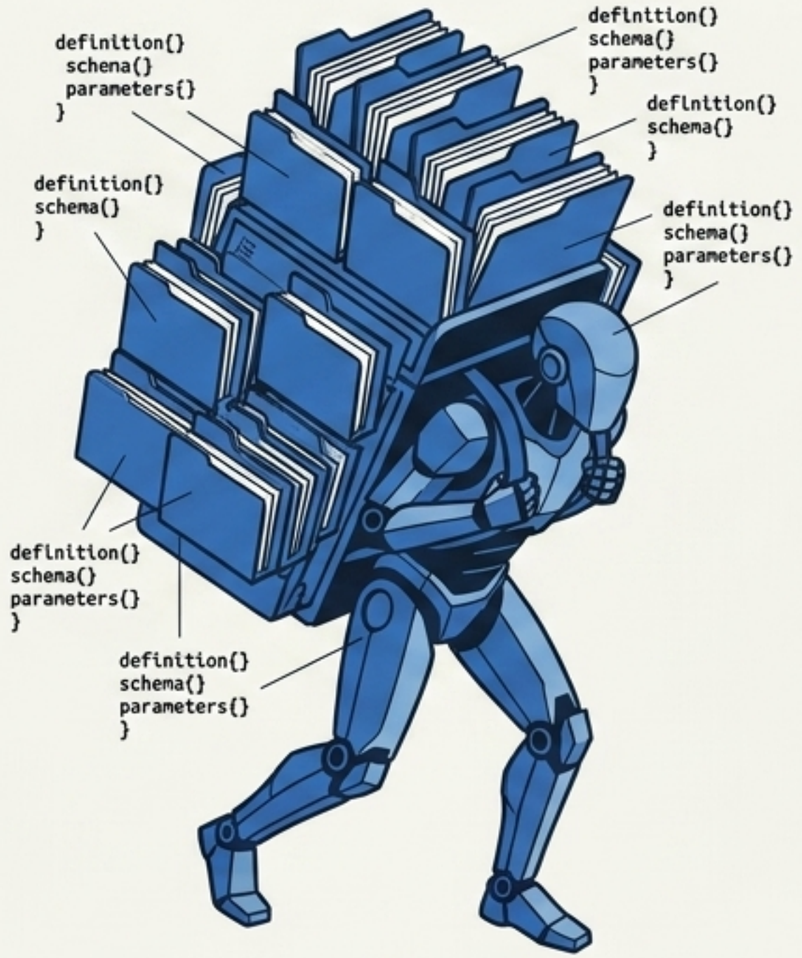
Built to execute repeated tasks with industrial efficiency.

The Rule:

A proven MCP run graduates into a CLI command.

MCP: The Scout

Explore what's possible, when no CLI exists.



Best For:
Discovery, one-off probes, and unhardened tasks. Self-describing and discoverable.

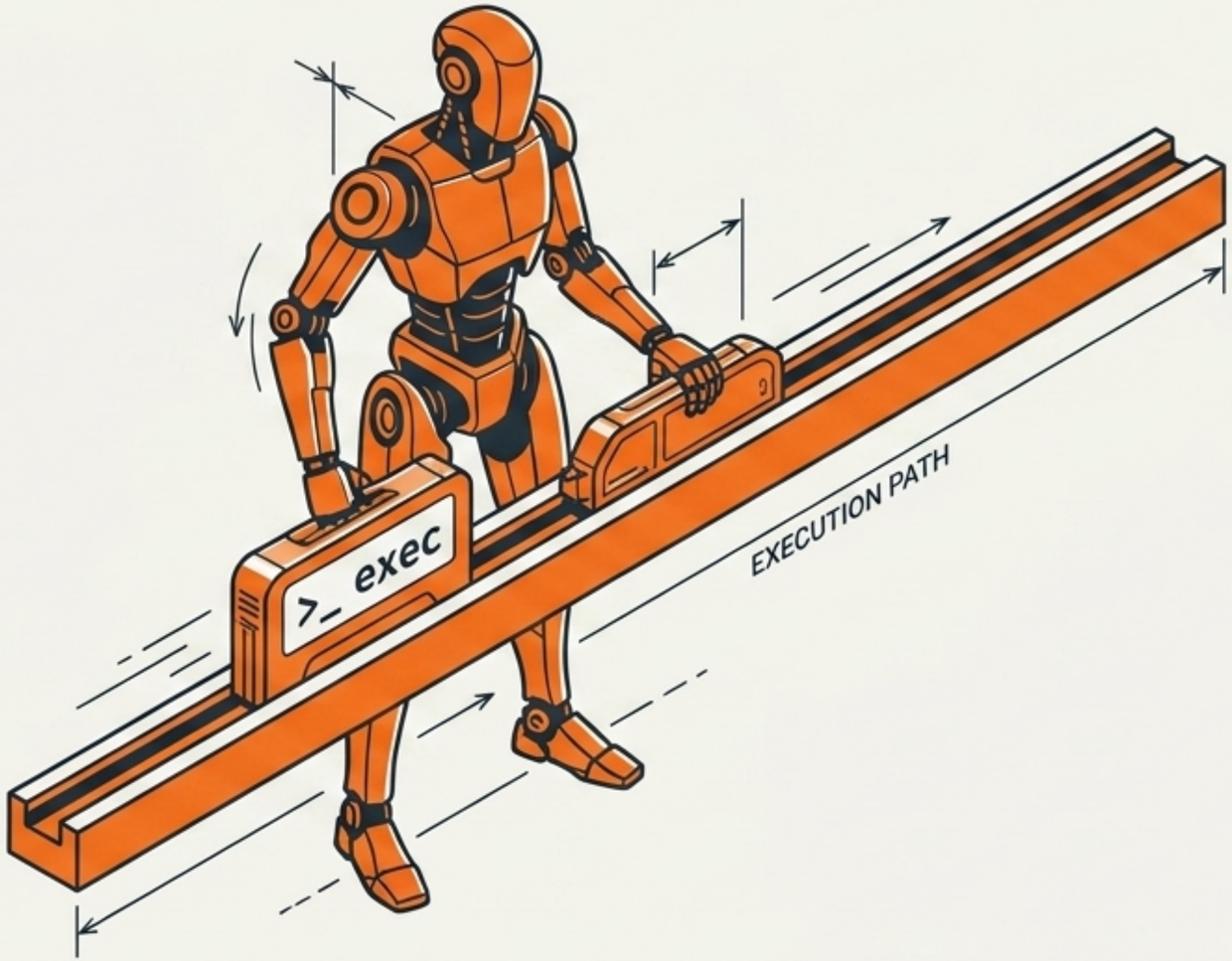
Context Cost [HEAVY]:
Every tool definition loads into the agent's context upfront.

Reliability [LOWER]:
Larger surface area leads to higher variance.

> **The Catch:** Real value, but the wrong workhorse. Do not make it your primary driver.

CLI: The Workhorse

Do it reliably—every time, at low cost.



Best For:
Primary driver, production environments, and any repeatable run.

Context Cost [LIGHT]:
One flexible command, not a pile of tool definitions.

Reliability [HIGH]:
Deterministic, testable, and fully hardened.

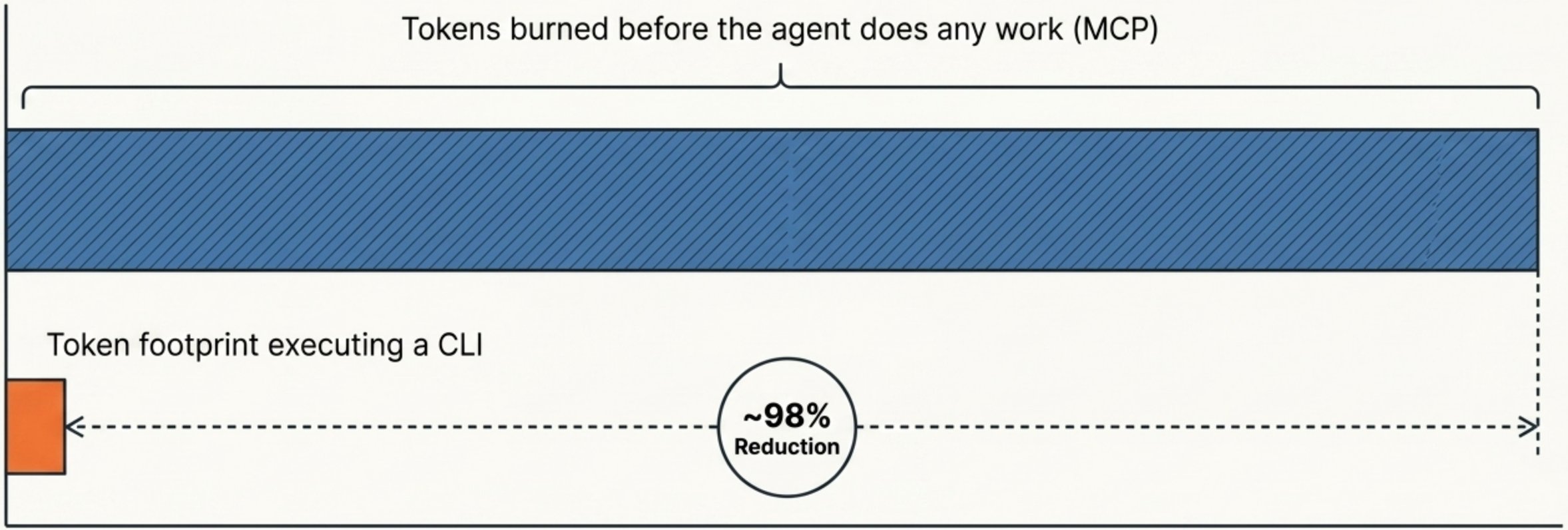
> **The Catch:** You have to know the specific command—but finding commands is what the agent is for.

The Diagnostic Matrix

	MCP (The Scout)	CLI (The Workhorse)
Primary Function:	Discovery & Probing	Production Execution
Context Cost:	Heavy (Pre-loaded definitions)	Light (Single command invocation)
Reliability Profile:	Variable (High surface area)	Deterministic (Hardened)
Target User:	Exploring Agents	Executing Agents (and humans)

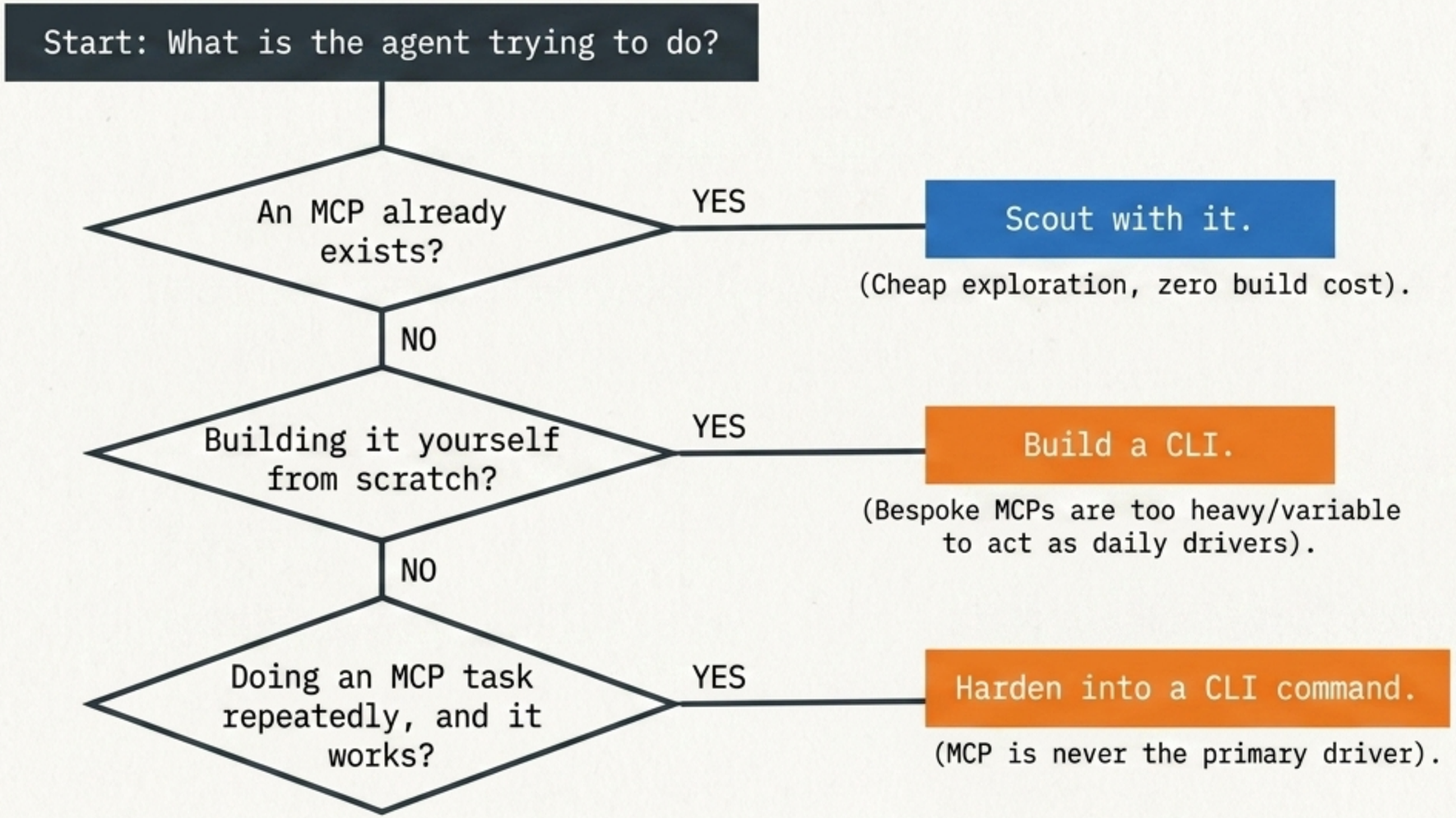
The Context Cost: Why CLI Drives

Anthropic's engineering team discovered that loading vast MCP tool definitions aggressively burns context before execution even begins. Shifting agents to code execution (calling CLIs/scripts directly) slashes token usage dramatically.



Community reports confirm massive token reduction using code execution over raw tool calls.

Which one, when?



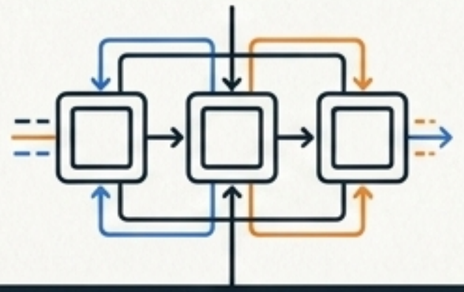
The Lineage of CLI-for-Agents

1. Early Pattern (Obsidian)

Ecosystem exposed via Local **REST API + MCP.**

The community pioneered '**code mode**', proving code/CLI execution beats raw tool calls by cutting tokens 40–60%.

2. Carried Forward (Peter Steinberger)



2. Carried Forward
(Peter Steinberger)

CLI-first agentic engineering.

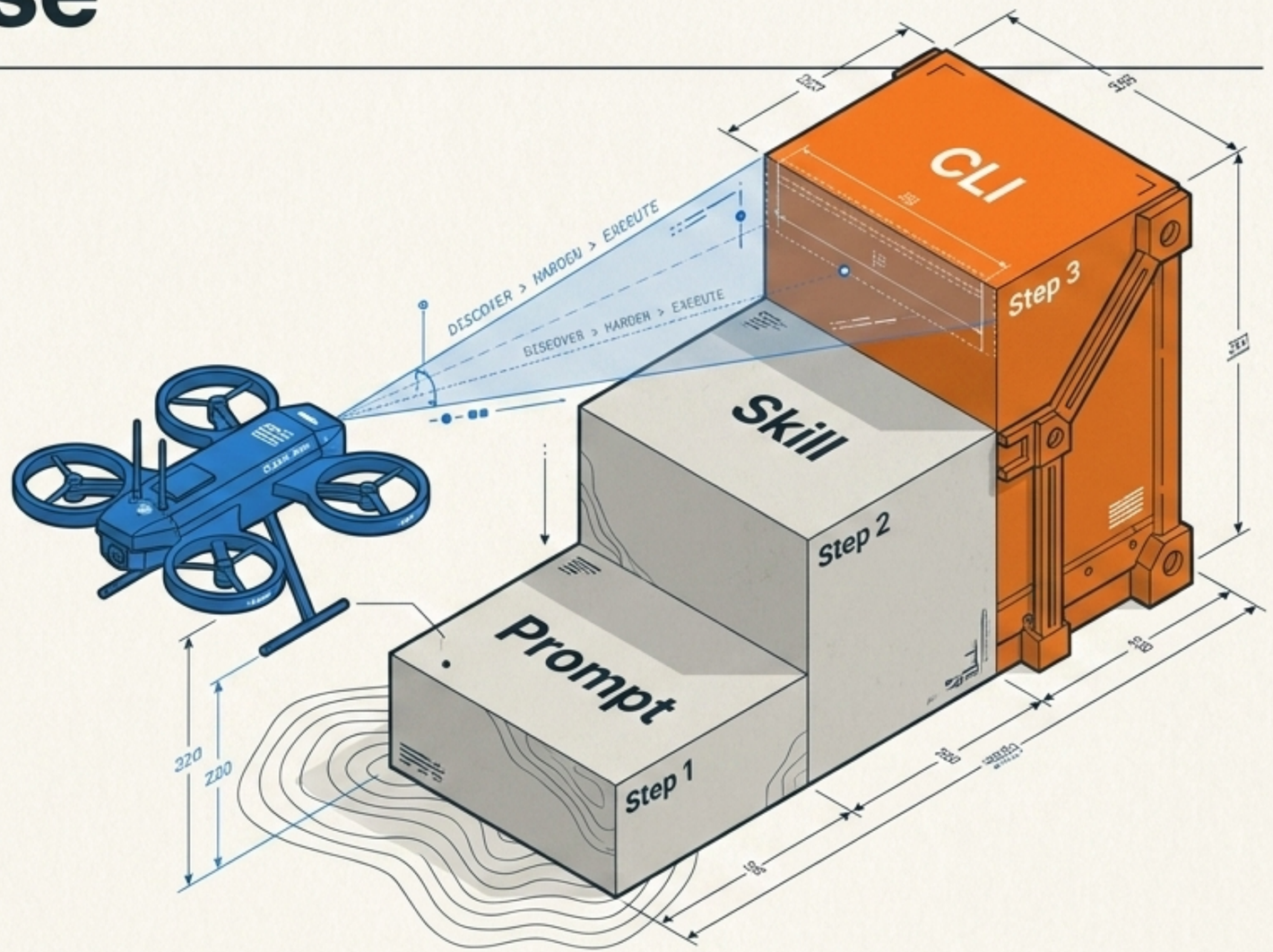
Demonstrated moving completely to a codex CLI as a daily driver, successfully running 3–8 agents in parallel.

3. Mainstream (Google)

Shipped a Workspace CLI (gws) alongside a **Gemini CLI extension**, giving agents direct terminal access to Gmail, Docs, and Sheets.

System Architecture: The Build Staircase

- MCP is not a rung on the engineering staircase.
- It is a scout that flies alongside the early, experimental altitudes (Prompt/Skill).
- You use the scout to discover what is possible in the fog, and you harden those findings onto the solid CLI rung.



The Ultimate Rule

Scout with MCP. Drive with CLI.

MCP is never the primary driver. Harden what you repeat.

Sources: Anthropic Engineering | Simon Willison | Firecrawl | mcpservers.org | VentureBeat