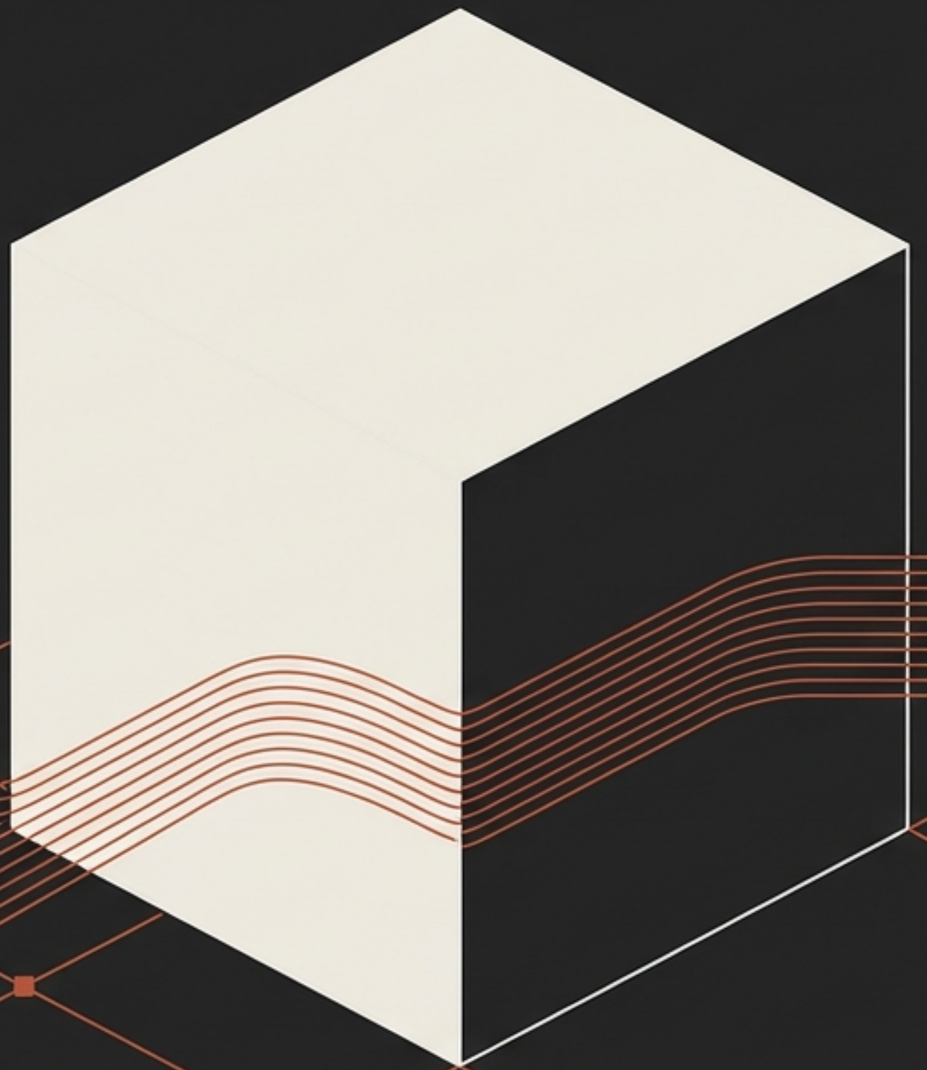
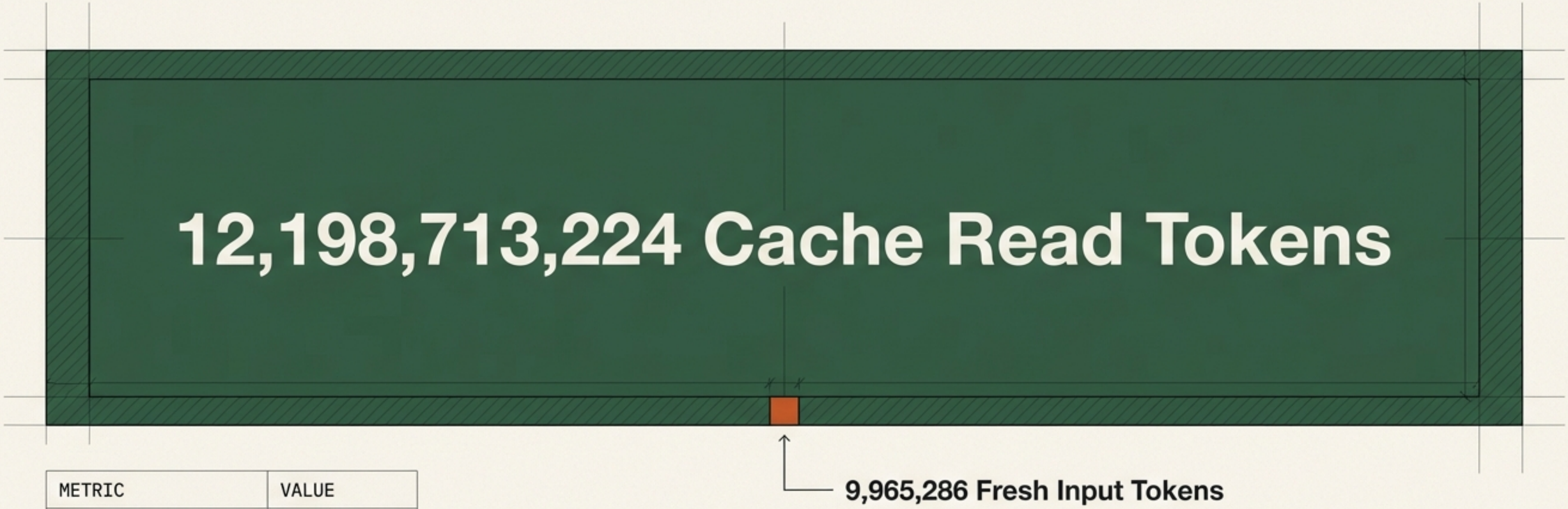
The Prompt Cache as Permanent Infrastructure

Lessons from 3,007 Claude Code Sessions.



Twelve billion tokens served from cache across 55 days

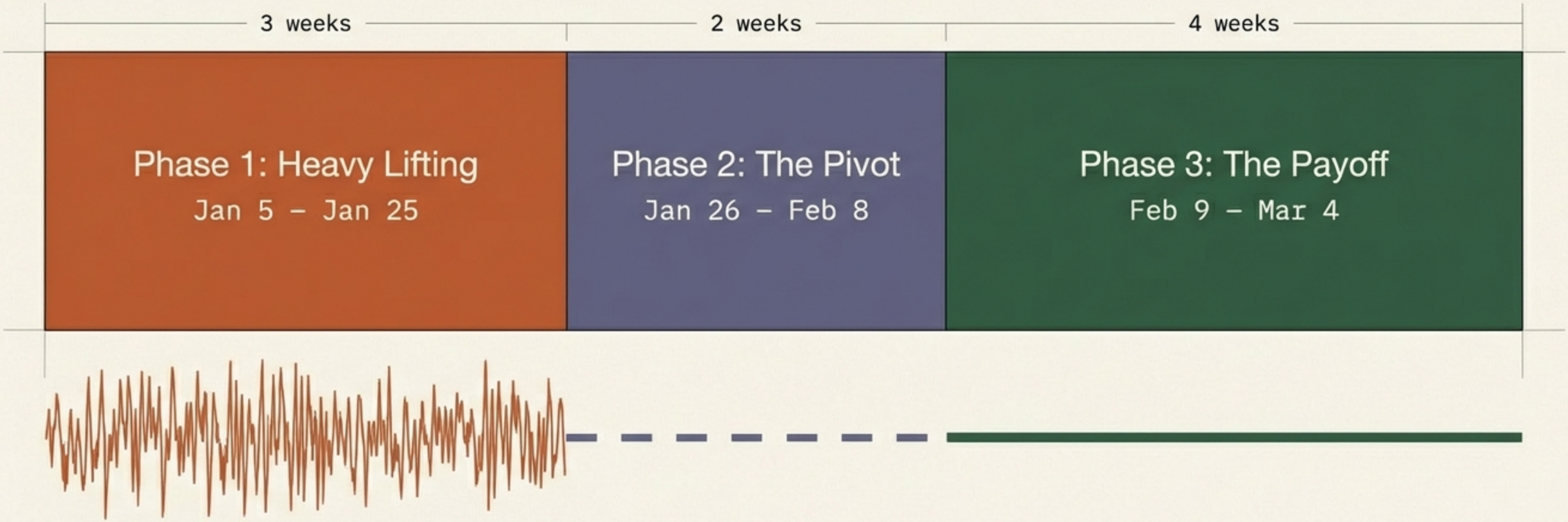
For every fresh token sent to the model, 1,224 were served directly from cache.



METRIC	VALUE
Sessions	3,007
Output tokens	13,279,048
Non-cached total	23,244,334
Grand total tokens	12.8B

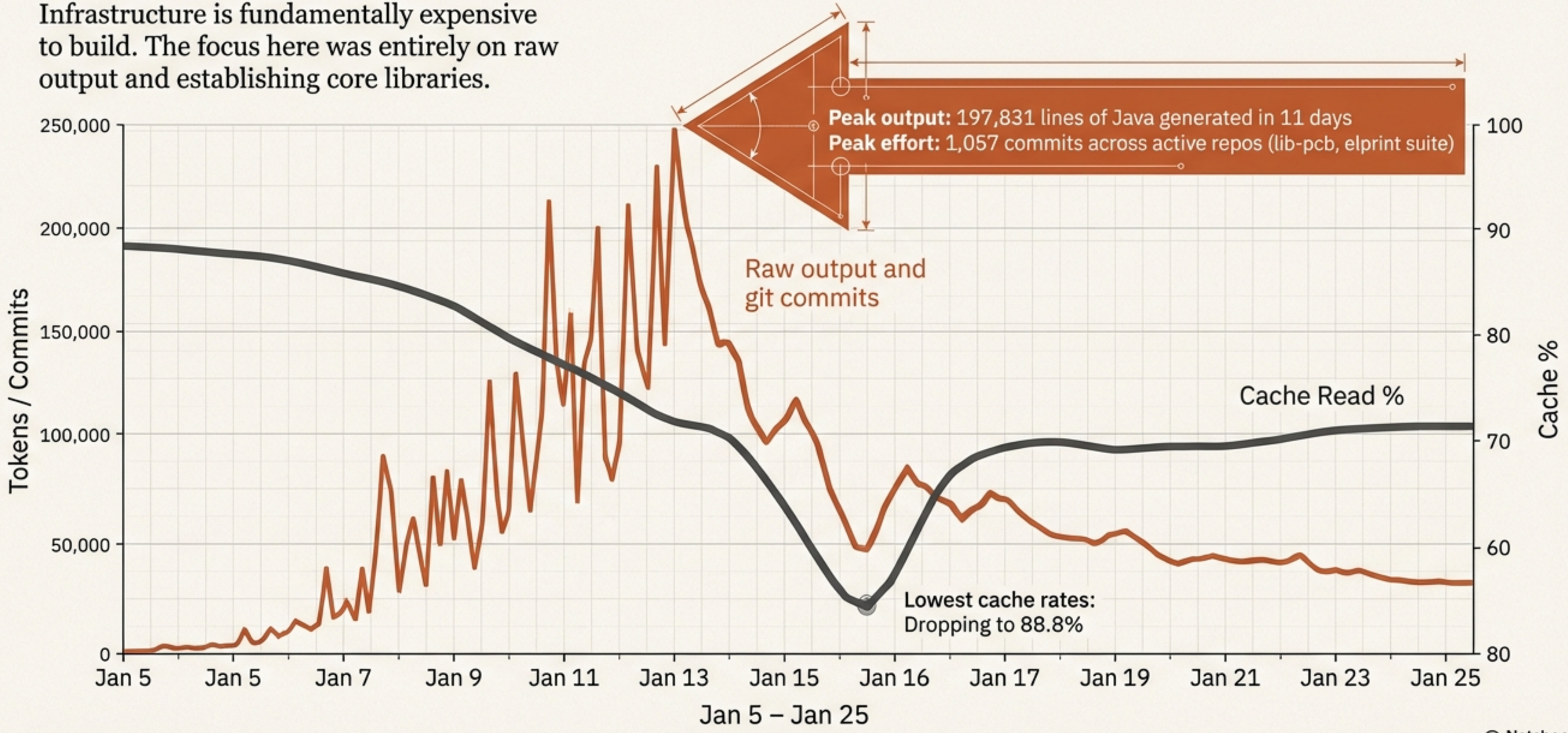
Infrastructure maturity happens in three distinct phases

Tracking Git commits, token usage, and cache hit rates over 55 days reveals the exact lifecycle of AI-augmented development.



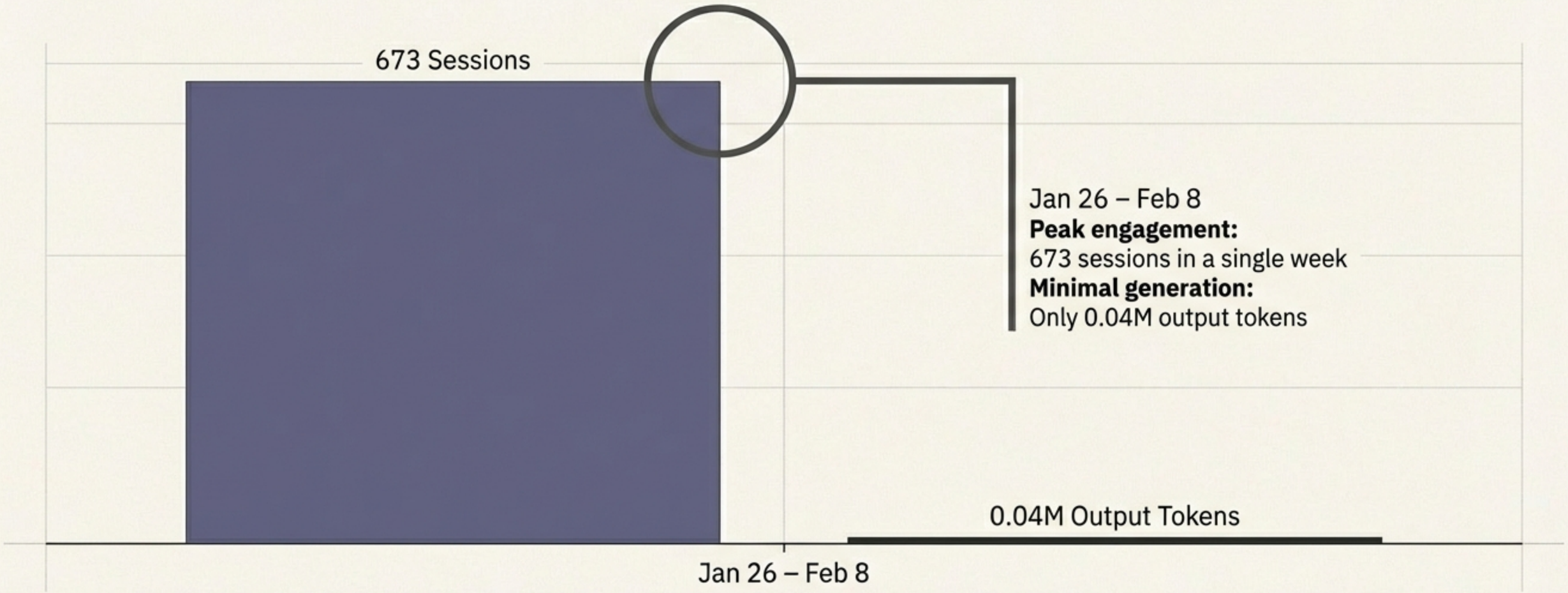
Phase 1 requires heavy raw generation to build the underlying architecture to fram chop in Helvetica Neue

Infrastructure is fundamentally expensive to build. The focus here was entirely on raw output and establishing core libraries.



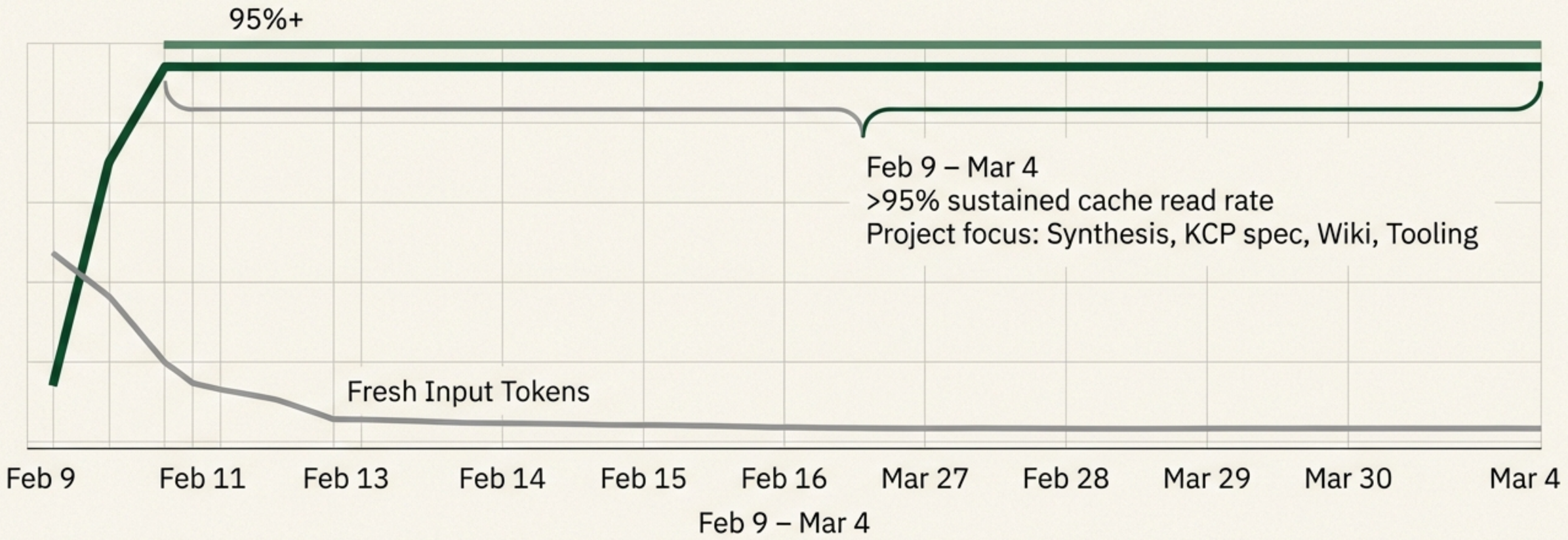
Phase 2 shifts workflow from raw generation to deep comprehension

Short-lived sessions dominated this period. The model behavior pivoted to codebase search, quick lookups, and exploration as foundational libraries wrapped up.



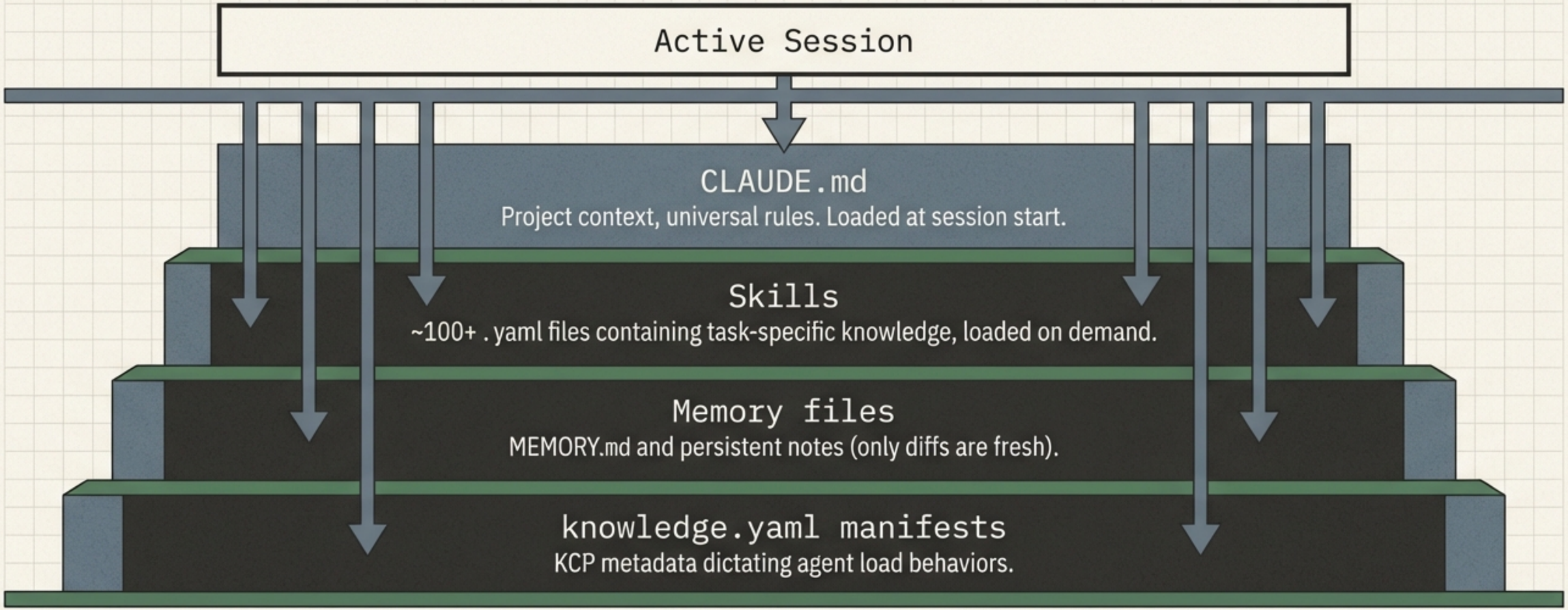
Phase 3 achieves sustained efficiency through persistent knowledge

The infrastructure begins paying for itself. Fresh input drops significantly as stable context handles the cognitive load of high-level synthesis and documentation.



A stable cache requires a dedicated knowledge layer

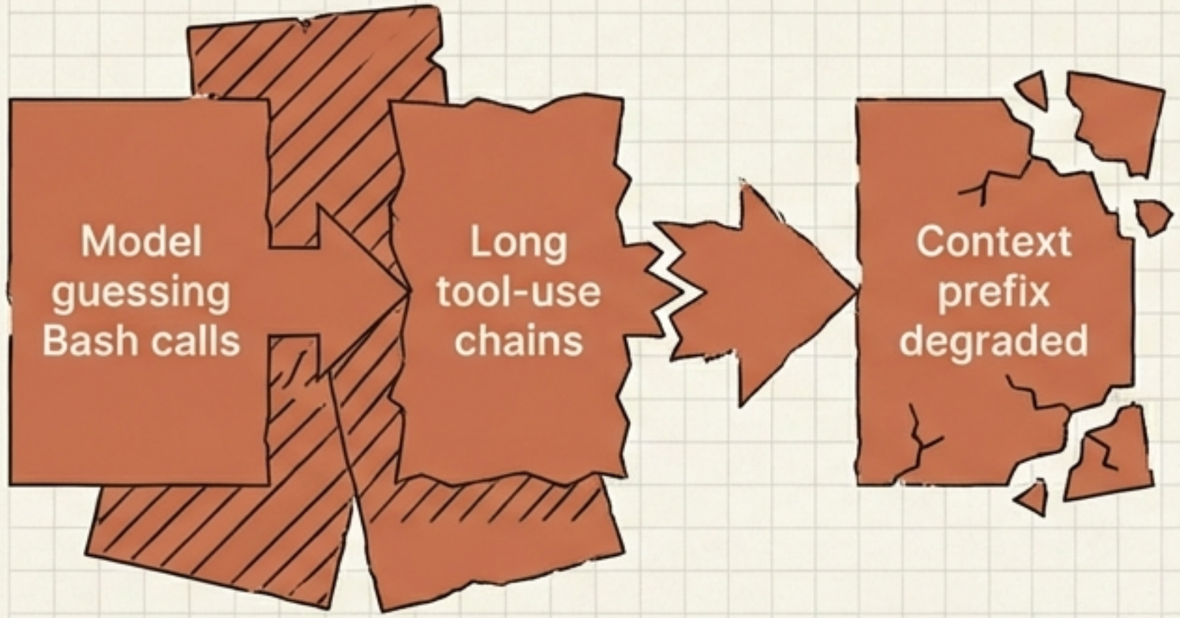
Anthropic's prompt caching relies on a stable context prefix. The larger and more consistent this prefix remains between sessions, the higher the hit rate.



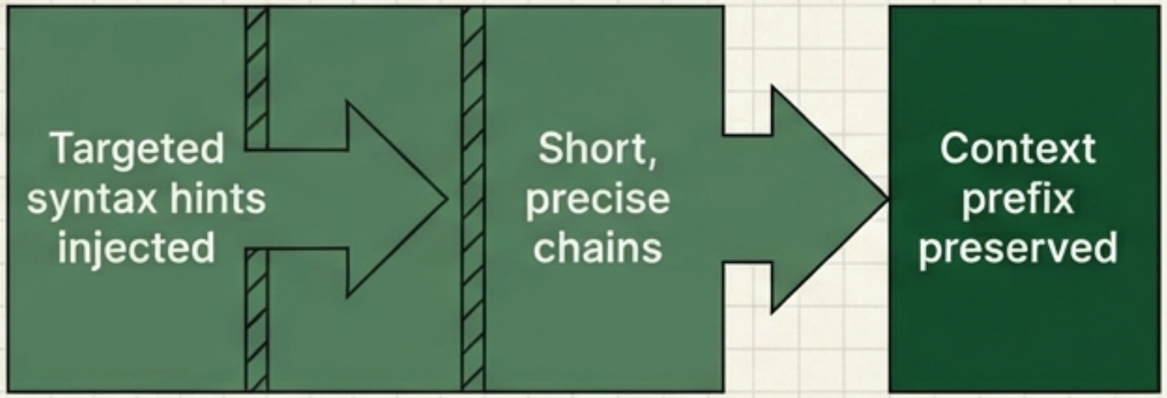
Targeted syntax injection actively shrinks the context window

Preventing the model from guessing syntax reduces tool-use chains. Less fresh-token noise preserves the stable context prefix, directly driving the 96%+ cache rates seen in February.

Before: High Fresh-Token Noise



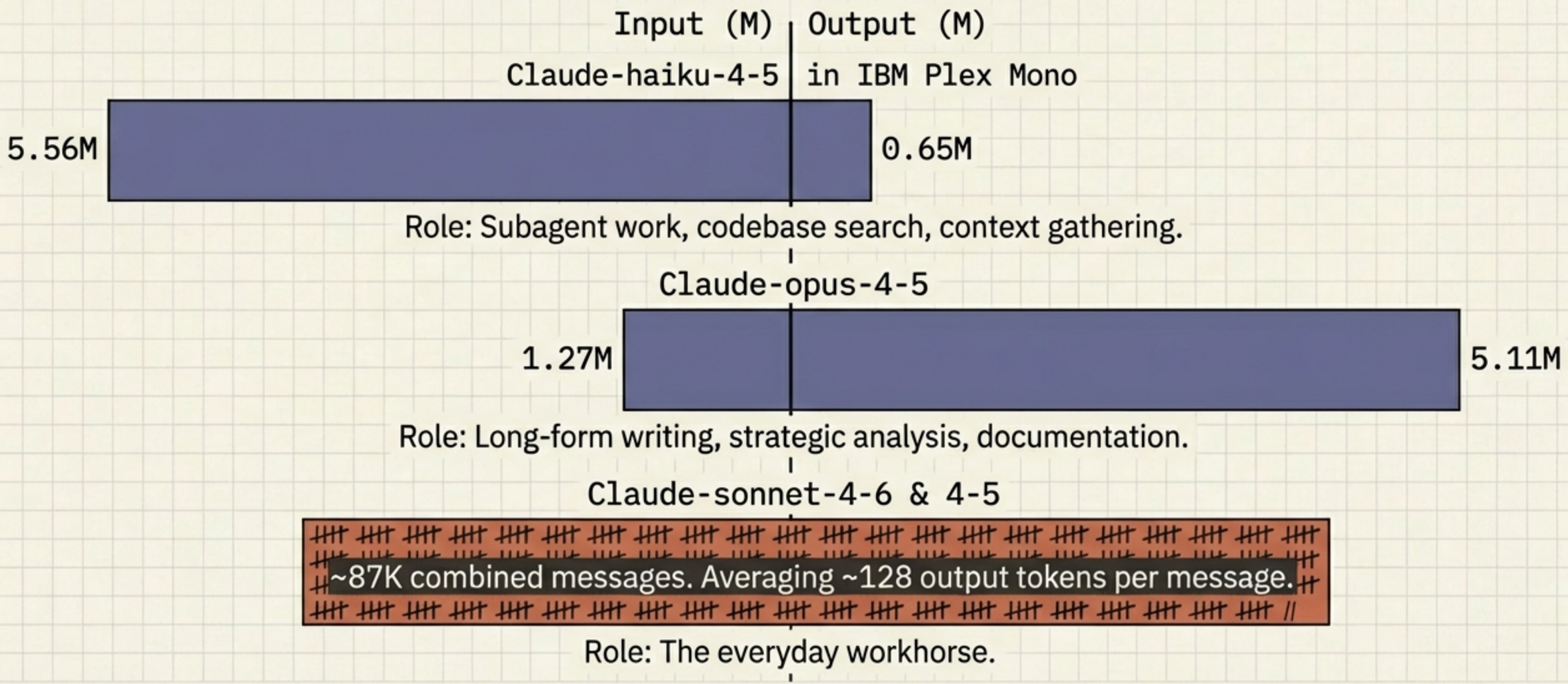
After: The kcp-commands effect



Result: Saves 33% of the context window per session

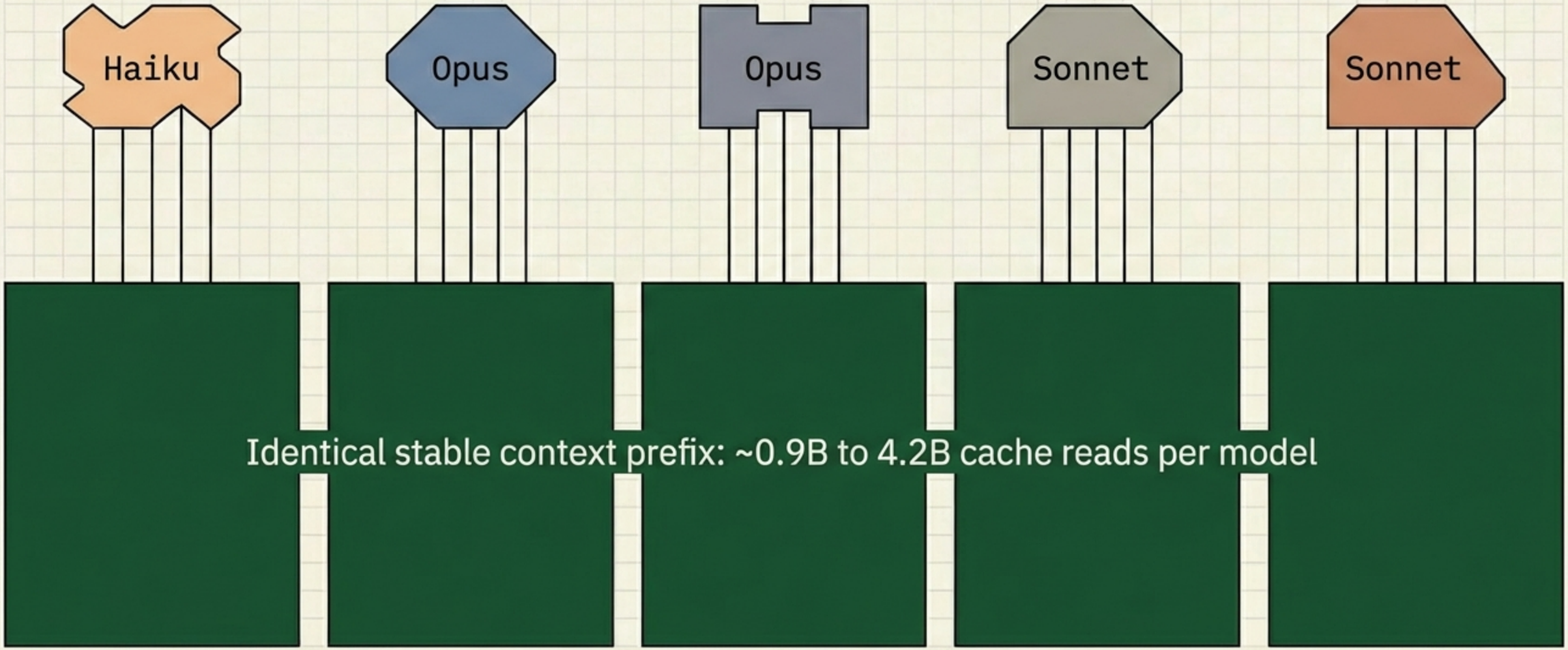
Specialized models handle distinct tasks within the same workflow

Five model generations were utilized across the 3,007 sessions, strictly separated by cognitive cost and output requirements.



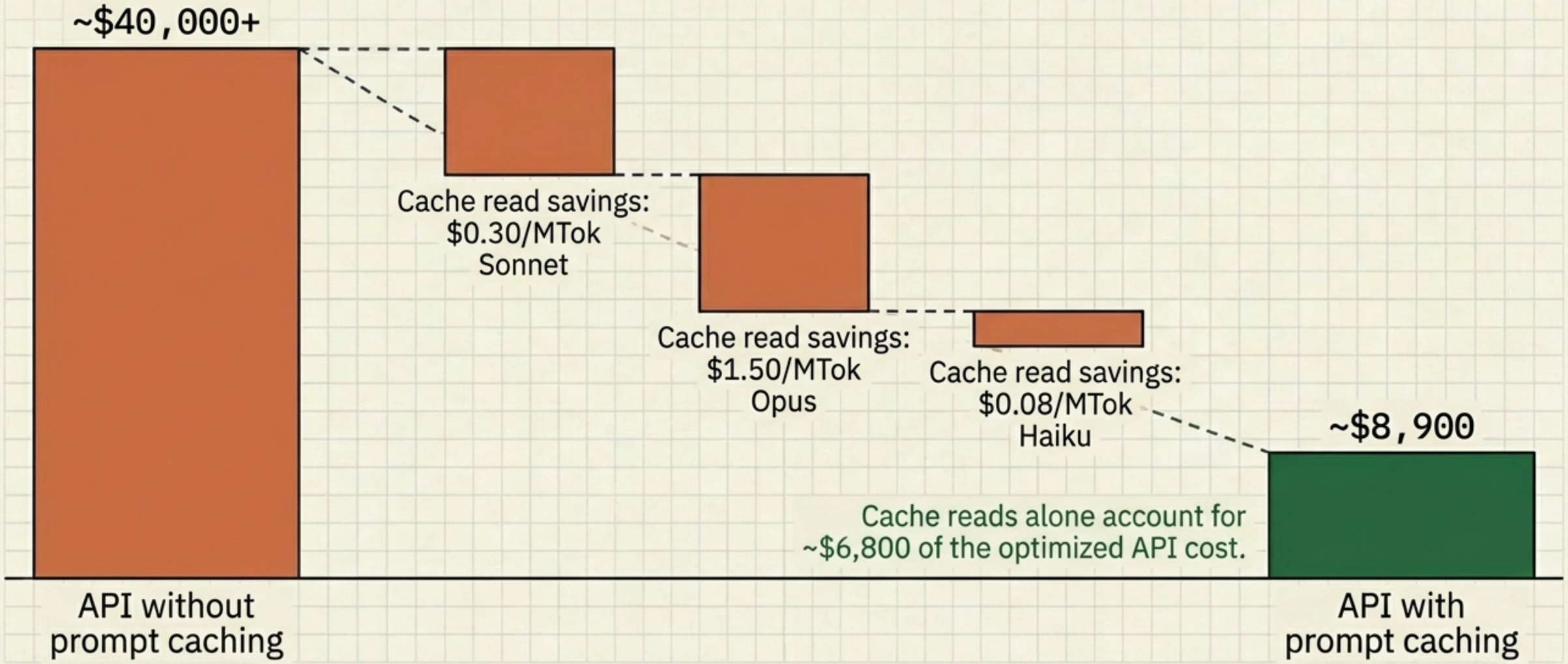
A unified context prefix benefits every model equally

The knowledge layer is entirely model-agnostic. The investment in structuring context pays dividends regardless of which model is actively generating the output.



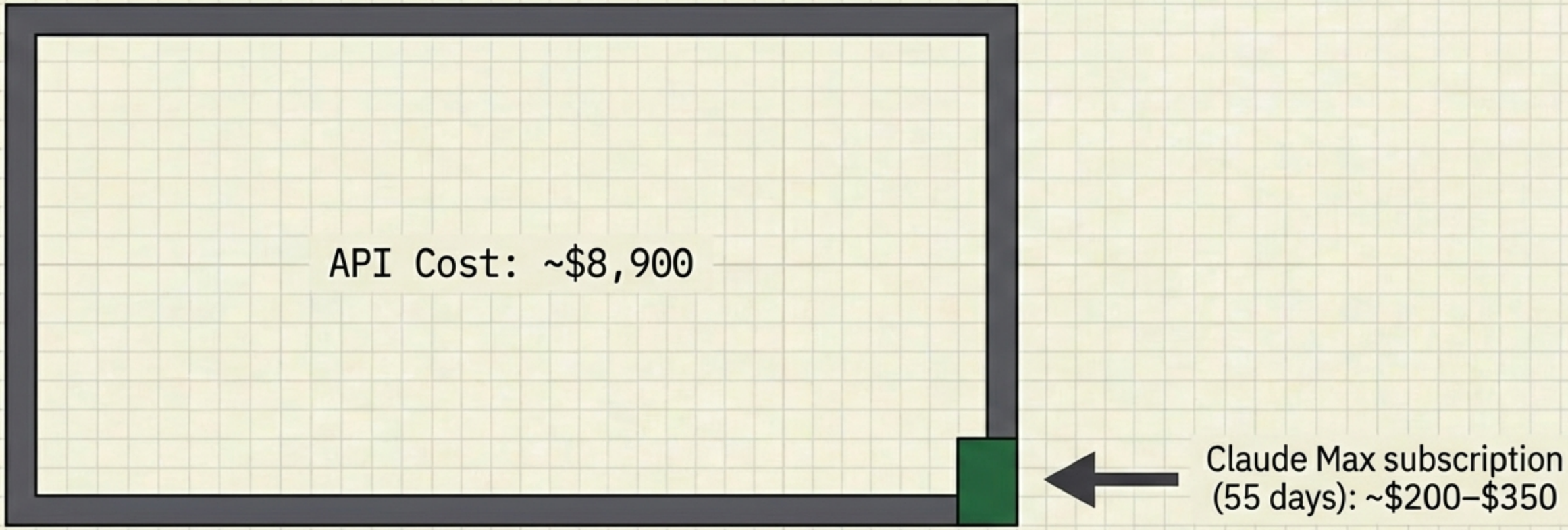
Disposable context fundamentally breaks the economic model

The caching infrastructure provides a 4.5x pure cost reduction on the API. Processing 12.2 billion tokens without it is financially ruinous.



Flat-rate subscriptions make high-frequency agentic workflows viable

The subscription model fundamentally alters the economics. It absorbs the high-frequency friction of autonomous agents, allowing the underlying knowledge infrastructure to compound without per-token penalties.



Actual output achieved: 2,412 commits across 15+ repositories over 55 days.

Treat your prompt context as infrastructure, not scaffolding.

Every hour spent building a well-structured CLAUDE.md, a precise skills library, or a knowledge manifest reduces the fresh token cost of every future session. Write it once. Structure it well. Let the cache do the rest.