Execution Latency in Multi-Agent Orchestration: A Skynet Operational Case Study

Exzil Calanza

Independent Researcher, Iloilo City, Philippines

Independent research preprint. IEEE-style formatting only; this work has not been accepted by, published by, certified by, or peer reviewed by IEEE.

ABSTRACT This case study analyzes an operational Skynet session managing 21 work items through persistent TODO state, atomic request ledgers, skill routing, browser-profile guards, and anti-drift review. The source run shows how persistent control state can make a complex multi-agent workflow recoverable and how execution, verification, and proof latency become visible system constraints once task intent is externalized. The evidence is a single operational run rather than a controlled benchmark, so the paper treats the result as a bounded case study rather than a general performance claim.

INDEX TERMS ¢â‚¬â€□ multi-agent orchestration; task decomposition; execution latency; anti-drift control; intent preservation; verification loops

I. INTRODUCTION

STUDY TYPE: Case study. **RESEARCH QUESTION:** How do persistent task state, skill routing, and multi-model coordination preserve intent and bound execution latency in multi-step agent workflows?

This preprint formalizes the existing public evidence record as a case study. No new experiment, dataset, measurement, or validation is claimed beyond the source article and its cited evidence.

The important shift is not that one model became omniscient. The shift is that a system can preserve intent across many agents, providers, tools, browser surfaces, proofs, and corrections. When that works, the limiting factor starts to look less like intelligence and more like how fast the system can think, verify, and execute.

II. THE CLAIM

This is a reflection on a real Skynet run, not a claim of general artificial intelligence. In this session, the operator gave a flooded request: create a Skynet breakthrough reflection, centralize skills, enforce a permanent TODO protocol, connect memory to TODOs, use Claude as anti-drift advisor, use Gemini only under accountability, audit a timelapse video, fix a weak platform post for scientific readers, and prepare for platform and LinkedIn publication through visual CDP proof.

The operational finding is the control pattern. Skynet does not depend on one model remembering everything in a fragile chat window. It turns the request into persistent TODO state, an atomic request ledger, skill routing, memory synchronization, source verification, video audit, CDP proof, and Claude review. That makes the work recoverable.

III. WHY THE BRAIN METAPHOR FITS, WITH LIMITS

The useful metaphor is not "one AI brain." It is closer to left and right hemispheres plus temporary task-specific regions. Codex handled implementation, file edits, CLI validation, and local audits. Claude acted as an independent anti-drift reviewer. Gemini remains reserved for specific high-complexity research or generation tasks where its output can be tamed by source checks and visual proof. Browser/CDP lanes become sensory-motor channels rather than memory.

The metaphor has limits. AI providers are not biological hemispheres, and the system is not conscious. The technical point is specialization plus coordination: different lanes produce, critique, verify, or render. Skynet's job is to keep them from drifting away from the user's actual request.

IV. WHAT MADE THE RUN NON-DRIFTING

TABLE 1

Control layer	Artifact	Why it matters
TODO continuity	`data/todos.json`	The active work survives interruptions and prevents false stopping.

Atomic request ledger	`data/skynet_atomic_request_ledger.json`	Twenty-one user requests were decomposed into proof-bearing work items.
Memory bridge	`tools/skynet_todo_memory.py`	TODO and ledger digests are synced into persistent orchestrator memory.
Skill routing	`data/skynet_skill_registry.json`	Skynet skills are selected from a registry, not remembered from stale context.
Advisor review	Claude anti-drift verdicts	Claude first returned `APPROVE_WITH_FIXES`, then `APPROVE` after corrections.
Visual/browser proof	SOCIALS CDP guard on port 9226	Live platform/social work stays tied to the visible SOCIALS browser profile.
Video evidence	`E:Youtube525525.mp4` audit	The timelapse was verified as a 1920x1080, 30 fps, 2:28 MP4 before LinkedIn use.

V. EXTERNAL CONTEXT: AGENTS NEED PROTOCOLS, NOT JUST MODELS

The direction is consistent with the broader AI-agent ecosystem. MCP describes itself as an open standard that lets AI applications connect to external systems such as files, databases, tools, and workflows [1]. Google's A2A announcement frames multi-agent systems as an interoperability problem across vendors, tools, and enterprise applications, with task management, collaboration, and artifacts built into the protocol vocabulary [2]. AutoGen showed earlier that multi-agent conversation frameworks can combine LLMs, tools, code, and human input for complex applications [3].

Those sources point to the same operational lesson: the next leap is less about a single model answering in isolation and more about reliable coordination. A system must know which lane should act, what evidence is required, and when a result is good enough to advance.

VI. THE BOTTLENECK WAS WALL-CLOCK EXECUTION

TABLE 2

Operation	Observed time	Result
Skill registry audit	~1.6s	Pass
TODO-memory sync and assert	~1.9s	Pass
Source liveness and title verification	~5.2s	3/3 pass
Timelapse video audit	~4.0s	Pass
Claude anti-drift review, first pass	~162.8s	APPROVE_WITH_FIXES
Claude anti-drift review, follow-up	~55.8s	APPROVE
SOCIALS CDP guard	~1.3s	Pass

VII. THE PRACTICAL FINDING

Once the right checks exist, intelligence is not the only scarce resource. The slow path is moving work through real surfaces: waiting for model review, verifying sources, auditing video frames, guarding a browser profile, and reconciling TODO/memory state after changes. In this run, execution latency and proof latency constrained workflow completion time and recovery cadence.

That changes how AI systems should be designed. A smarter model helps, but a slower or driftier execution loop still fails the user. A durable orchestration layer can make a slightly messy multi-model workflow more reliable than a single brilliant response that cannot remember what remains to be done.

VIII. WHAT SKYNET PROVED IN THIS RUN

- Flooded requests can be atomized: the session was decomposed into 21 tracked work items instead of one vague objective.
- Memory can be made accountable: the TODO-memory bridge stores digests so future agents can detect drift.
- Advisors can be useful without taking over: Claude caught real issues in status sync, source hygiene, and scientific framing.
- Visual proof matters: browser and social work are gated behind a visible SOCIALS CDP profile, not invisible assumptions.
- Video can be treated as evidence: the timelapse is audited before it becomes a LinkedIn proof asset.

IX. LIMITATIONS

This is a single operational run. It does not prove that Skynet is generally optimal, autonomous, or immune to mistakes. One race condition did appear: a parallel assert checked memory before a TODO close and sync had fully settled. The system caught the mismatch and a sequential sync/assert cleared it. That is the correct failure mode: expose drift, then repair it with proof.

The benchmark also reports observed local timings, not a peer-reviewed model benchmark. It should be read as evidence for an engineering thesis: in real agentic workflows, orchestration speed, verification speed, and recovery speed become first-class performance variables. The Claude anti-drift advisor resolved to `claude-opus-4-6` in this session's direct CLI routing, which is reported here as observed provider behavior rather than hidden or normalized.

X. START SIGNAL, NOT FINISH LINE

This is the start signal: AI providers can stop being isolated chat boxes and start acting like coordinated specialists inside one accountable operating loop. The system can form new working parts when the task demands it: a TODO lane, memory lane, advisor lane, visual proof lane, video audit lane, and publishing lane.

The bar now is not only how smart the model is. The bar is how fast the system can think, verify, execute, correct itself, and keep going without losing the user's original intent.

XII. LIMITATIONS AND CLAIM BOUNDARY

This is a single-run observation with local timing. The source records a race condition during a parallel state assertion and does not establish peer-reviewed or cross-environment performance benchmarks.

Claim boundary - Use technical case-study framing for the speed claim rather than presenting a universal breakthrough or general performance law.

XIII. REFERENCES AND SOURCE RECORD

- [1] Model Context Protocol documentation: What is MCP?
- [2] Google Developers Blog: Announcing the Agent2Agent Protocol
- [3] AutoGen: Enabling Next-Gen LLM Applications via Multi-Agent Conversation

AUTHOR BIOGRAPHY

EXZIL CALANZA is an independent researcher and software builder based in Iloilo City, Philippines. His work focuses on browser and desktop automation, reproducible evaluation, agent reliability, workflow verification, and AI-assisted systems engineering.