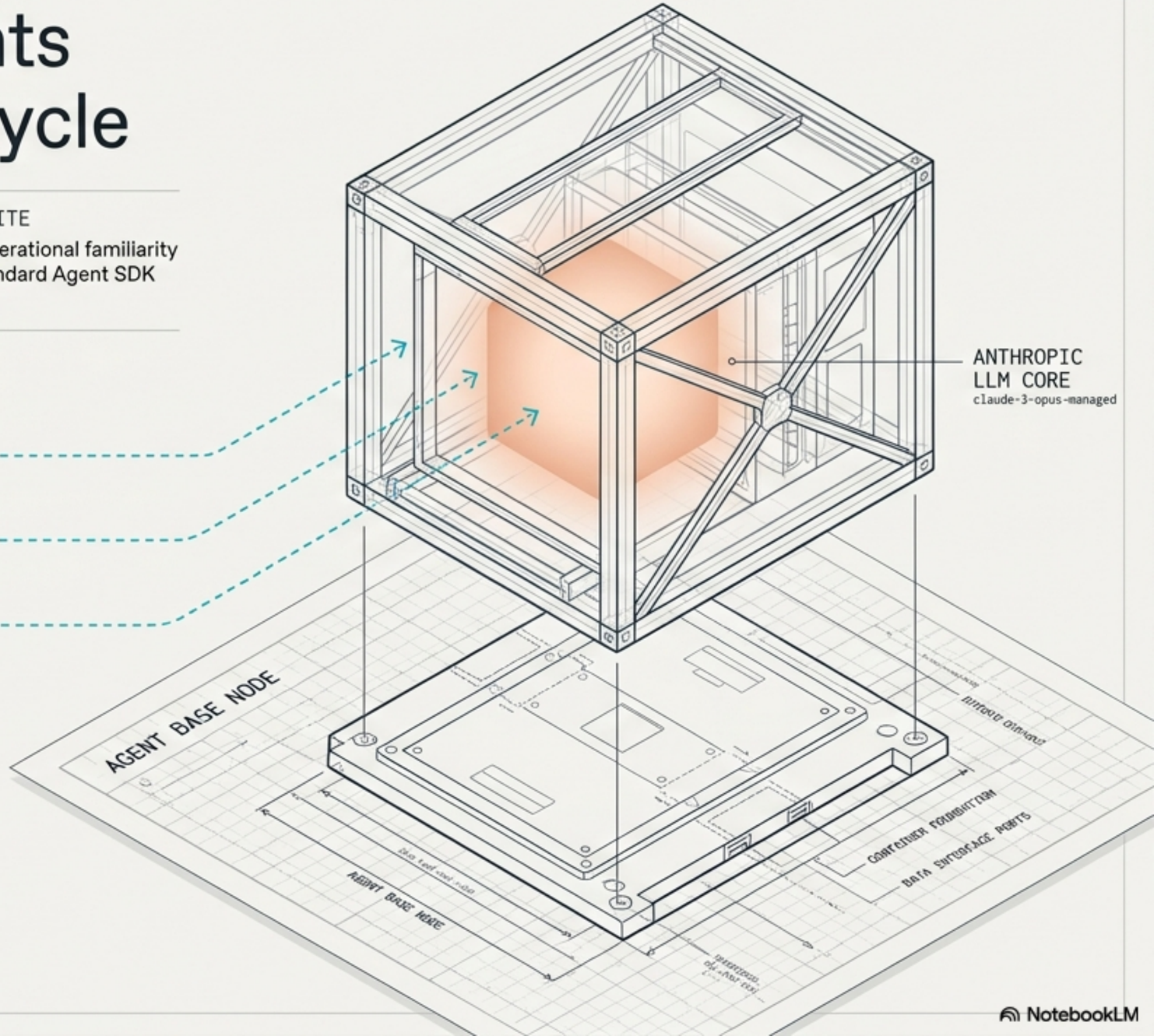
Claude Managed Agents Architecture and Lifecycle

RELEASE CONTEXT	GLOBAL REQUIREMENT	PREREQUISITE
Public Beta (Launched April 8, 2026)	All API requests mandate the `managed-agents-2026-04-01` beta header.	Assumes operational familiarity with the standard Agent SDK (Chapter 1).

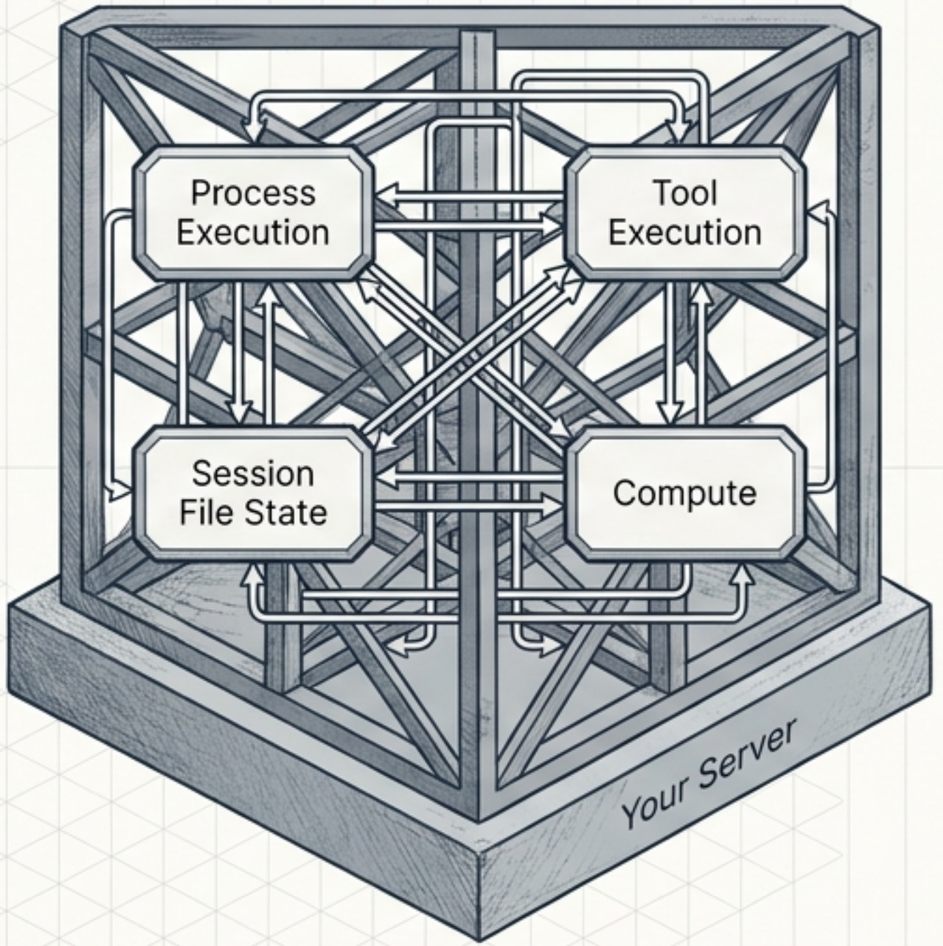
INBOUND: STREAMING DATA / EVENTS
(api/stream/v1/event_stream)

INBOUND: USER INPUTS / DYNAMIC CONTEXT
(api/stream/v1/user_input)

INBOUND: SYSTEM SIGNALS / CONFIGURATION UPDATES
(api/stream/v1/system_signal)

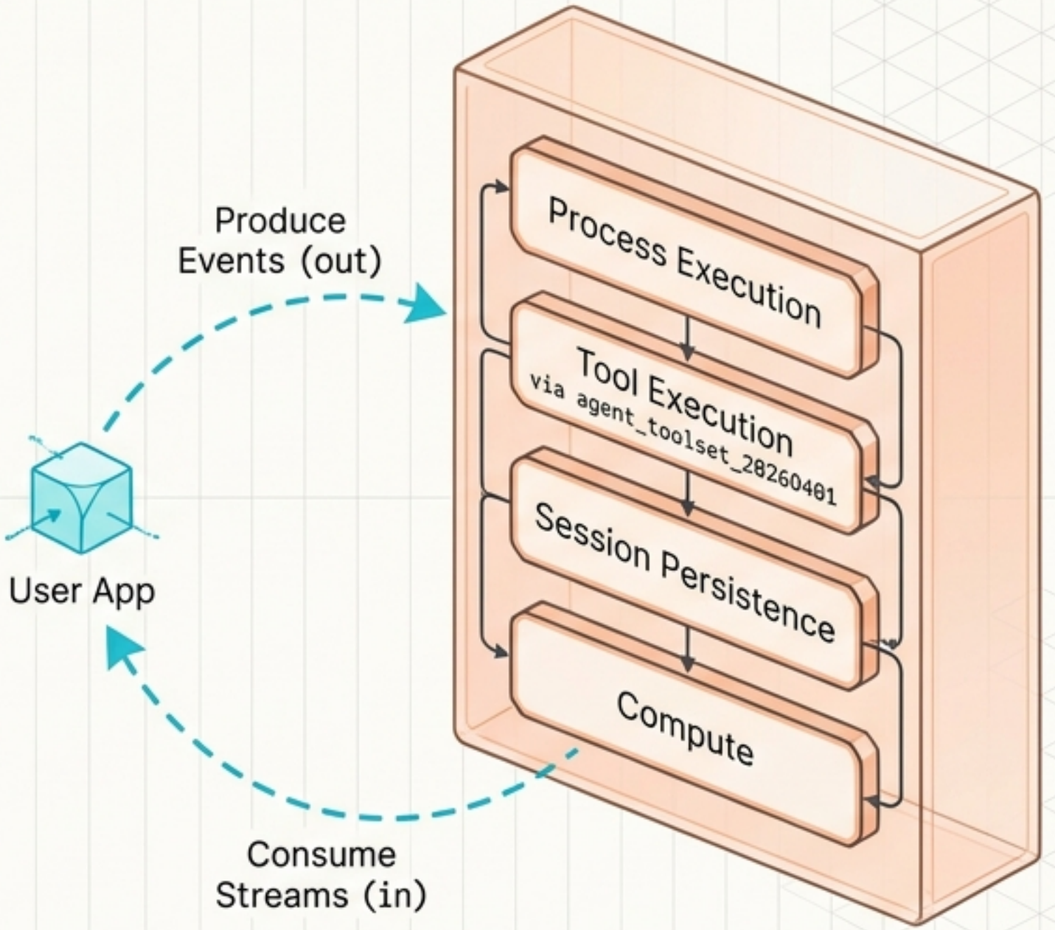


Agent SDK (Your Infra)



You manage the execution loop.
Your application acts as the executor.

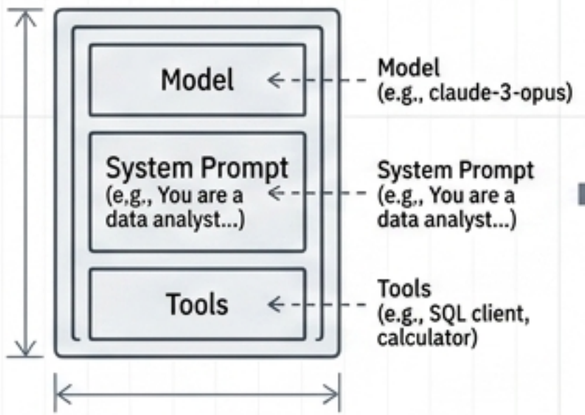
Managed Agents (Anthropic Infra)



Anthropic handles the sandbox. Your application transitions to an event producer and consumer.

The functional primitives of a managed deployment

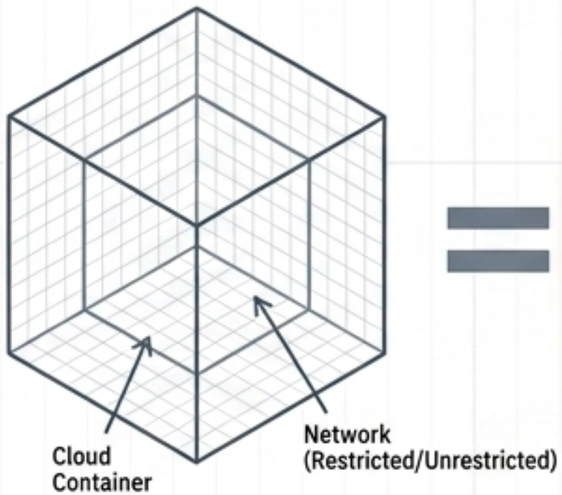
1 Saved Configuration (Model, System Prompt, Tools). A static blueprint, similar to a Docker image.



AGENT

Saved Configuration (Model, System Prompt, Tools). A static blueprint, similar to a Docker image.

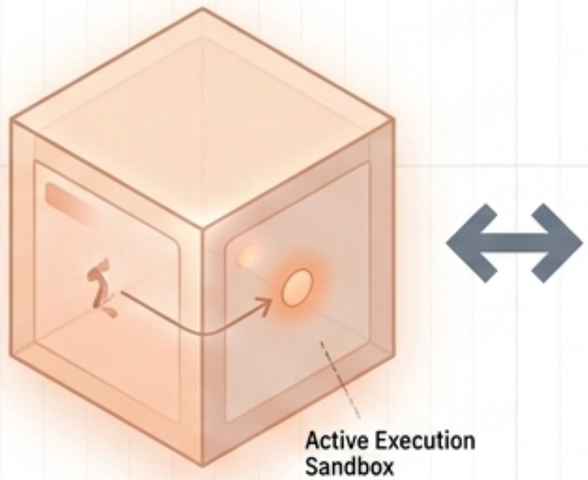
2 Cloud Container Template. Defines type: cloud with restricted or unrestricted networking.



ENVIRONMENT

Cloud Container Template. Defines type: cloud with restricted or unrestricted networking.

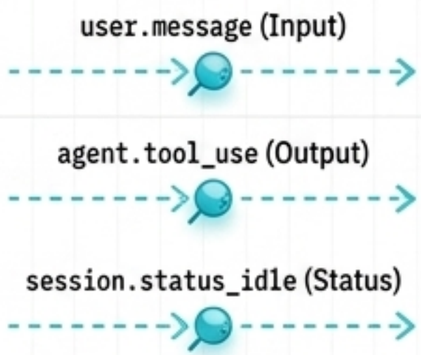
3 Running Instance. 1 Session = 1 Task. Active execution sandbox.



SESSION

Running Instance. 1 Session = 1 Task. Active execution sandbox.

4 Server-Sent Events (SSE) I/O. Key streams: `user.message`, `agent.tool_use`, and `session.status_idle`.

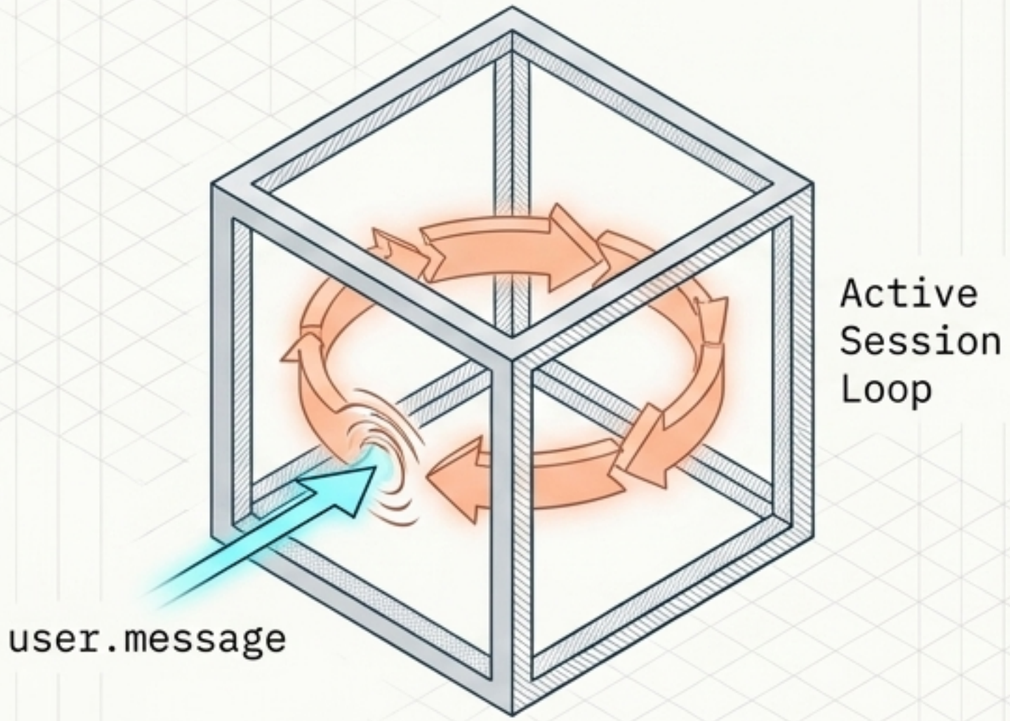


EVENTS

Server-Sent Events (SSE) I/O. Key streams: user.message, agent.tool_use, and session.status_idle.

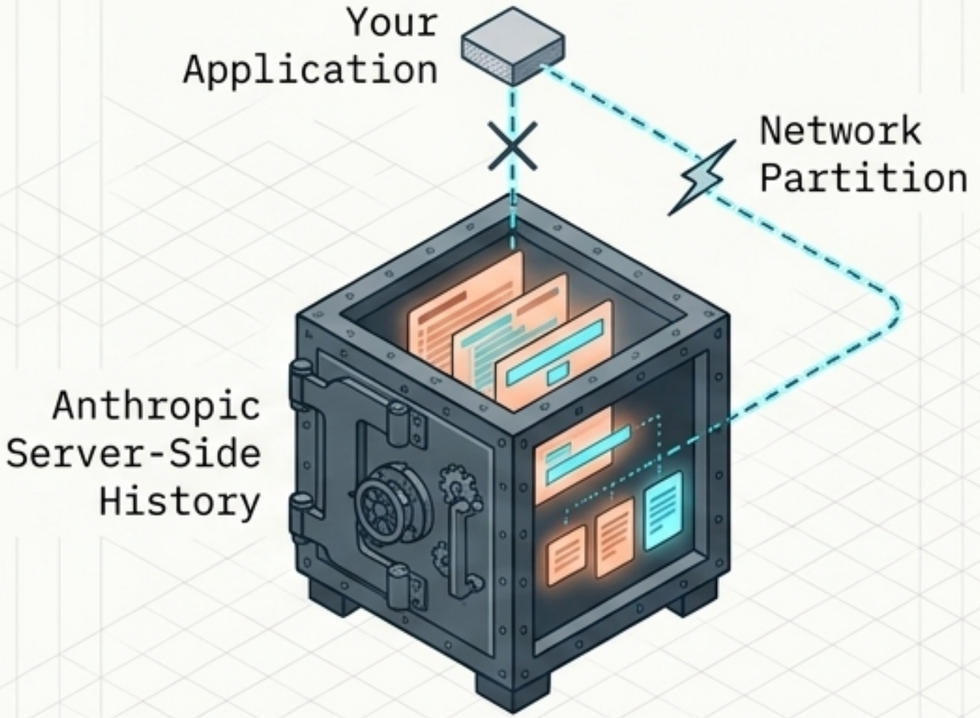
Real-time steering and server-side state persistence

DYNAMIC STEERING



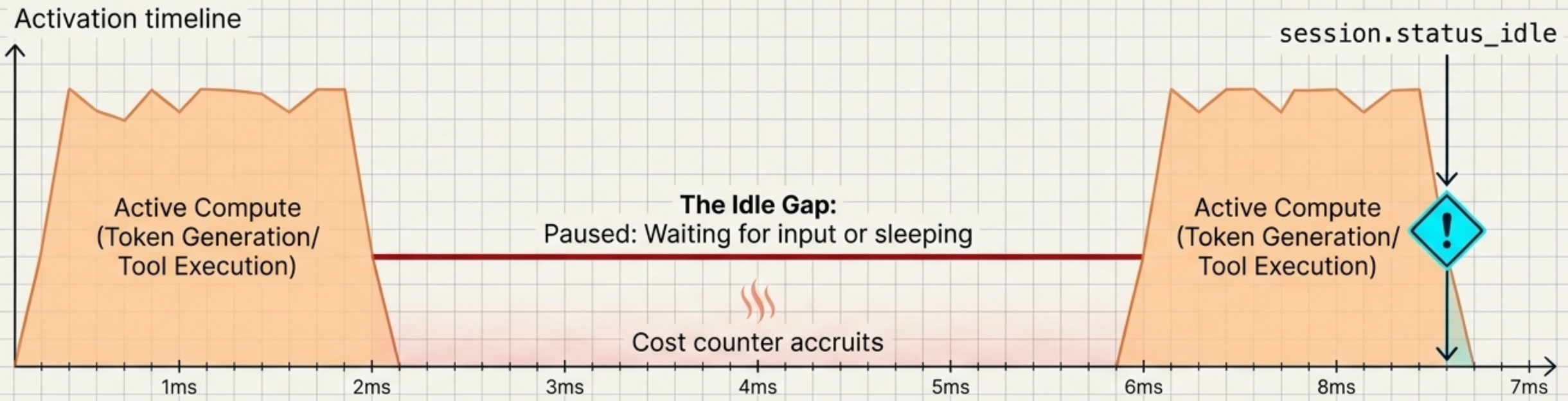
Change direction without restarting or stopping the generator. Unlike the Agent SDK, new instructions can be injected while the remote agent works.

STATE PERSISTENCE



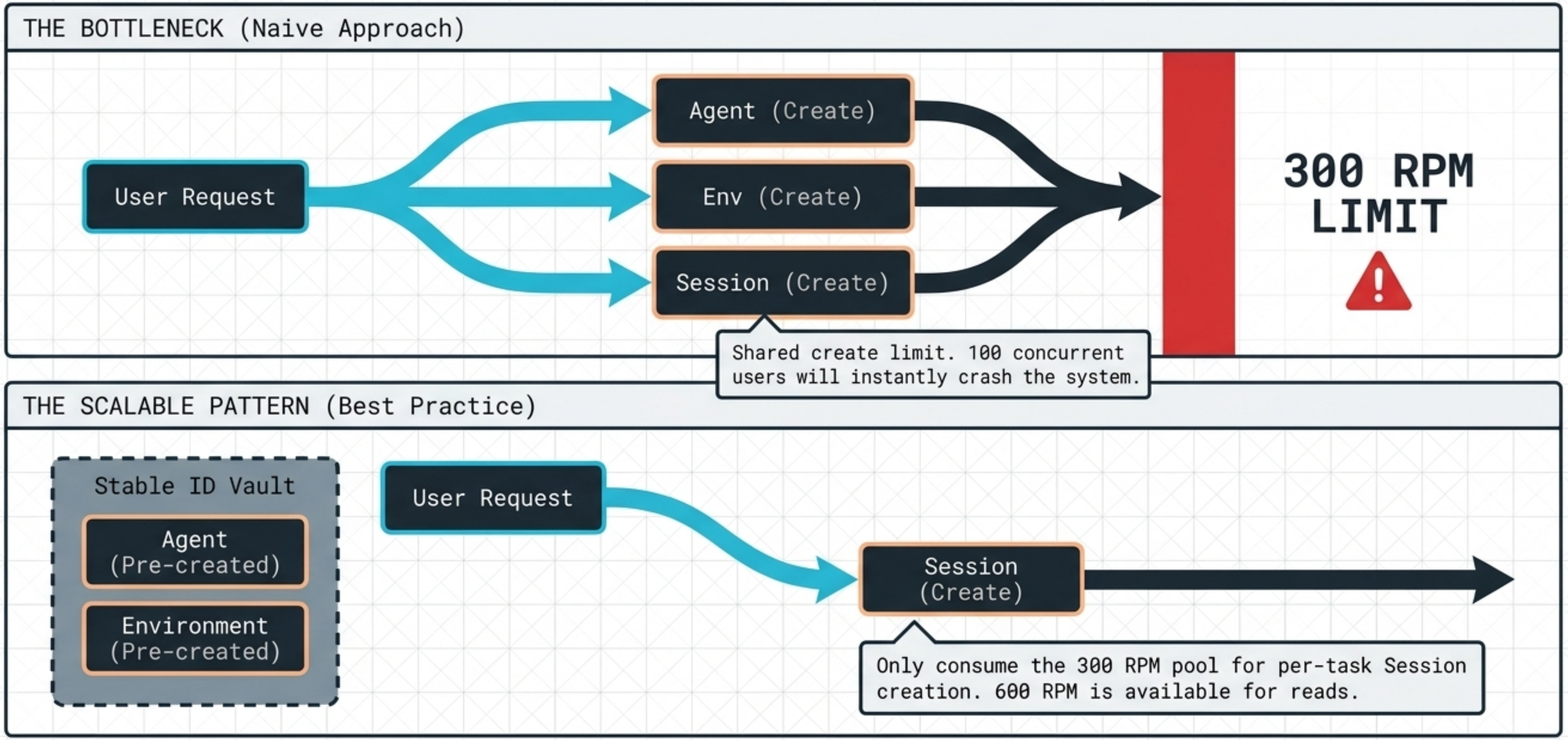
The source of truth lives in Anthropic's infrastructure. Replaces local JSONL files. Allows retrospective replay, survives local disconnects, and lets multiple processes query the same session simultaneously.

The lifecycle cost trap: Idle time still accrues runtime exposure



THE TRAP	THE FIX
Pricing requires accounting for Managed Agents runtime + standard token costs. Sessions waiting for messages or sleeping between tool calls silently accrue runtime exposure.	Your stream loop must detect session.status_idle and explicitly close the session (status='completed') to prevent runaway costs. Never use Managed Agents for 30-minute polling loops.

Designing around the 300 RPM infrastructure bottleneck



The architectural decision matrix: Managed Agents vs Agent SDK

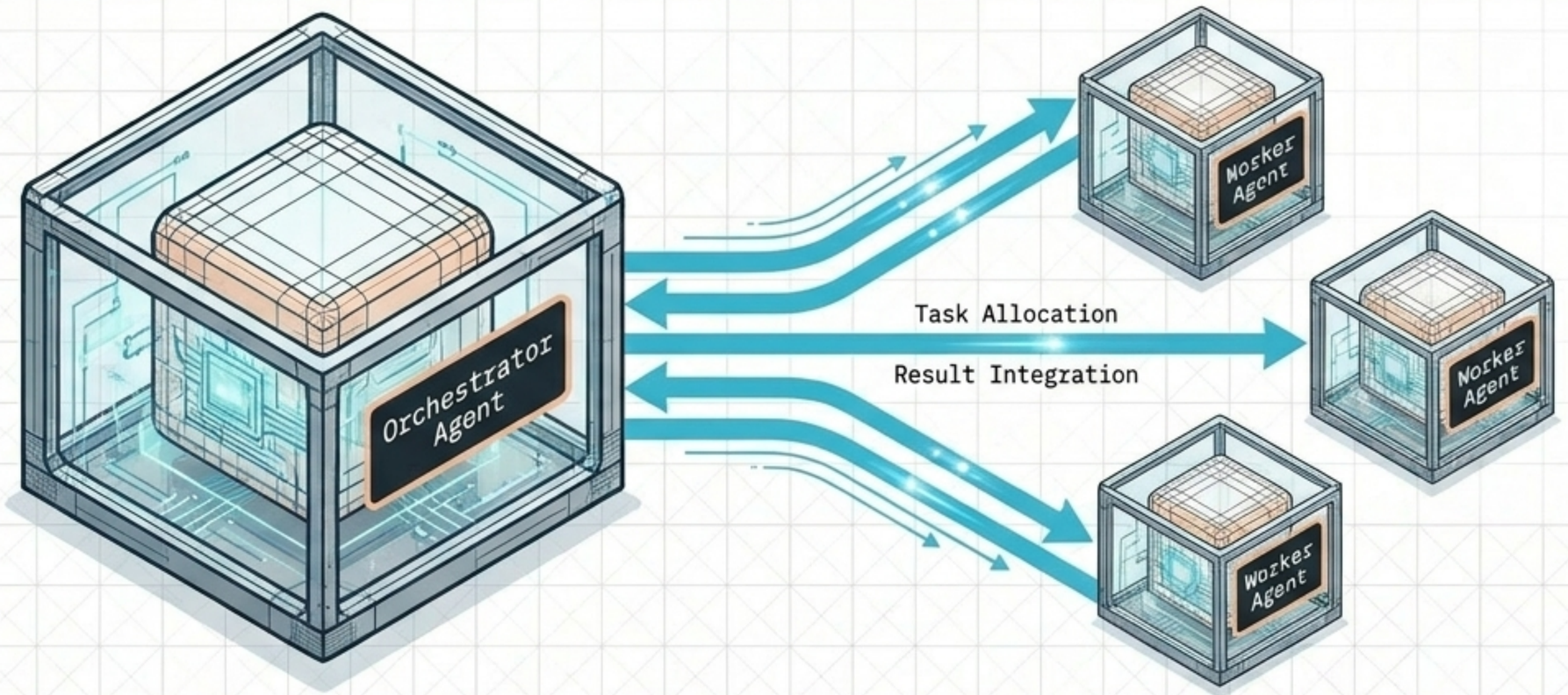
Workload Scenario	Recommended Path	Architectural Rationale
Long-running cloud tasks (>5 min) & Async workflows	✅ Managed Agents	Anthropic handles the robust cloud sandbox and infrastructure.
High-concurrency end-user serving	✅ Managed Agents	Automatic scaling without managing local execution loops.
Short, stateless tasks (e.g., 15-30s GitHub webhooks)	● Agent SDK	Serverless functions (Lambda/Cloud Run) are cheaper and simpler.
Operating on local files or custom in-process Python	● Agent SDK	Requires direct, absolute local filesystem and process control.
Local prototyping without cloud budget	● Agent SDK	Zero runtime overhead; skips cloud billing cycles.

Note: The canonical path (Prototype SDK → Production Managed) only applies if your production workload is genuinely long-running and asynchronous.

The Horizon: Multi-agent orchestration

STATUS: RESEARCH PREVIEW
Requires separate access approval.

Hub-and-Spoke



CAPABILITY	NEXT STEPS
Achieves true parallelism—multiple agents running simultaneously in hosted containers, rather than sequentially in-process.	Proceed to Chapter 3: The Model Context Protocol (MCP) to turn general Claude into a specialized internal system operator by securely connecting external tools.