AI Agents and Harness Engineering

Presented by Sanjai Balajee

About Me

Sanjai Balajee

- Tech Analyst , CitiBank
- SSN CSE’25
- Interned at CMU (Summer 2024)
- Intered at IISC, Bengaluru (Summer 2023)

link to slides

Agents v Workflow

Definition (Anthropic)

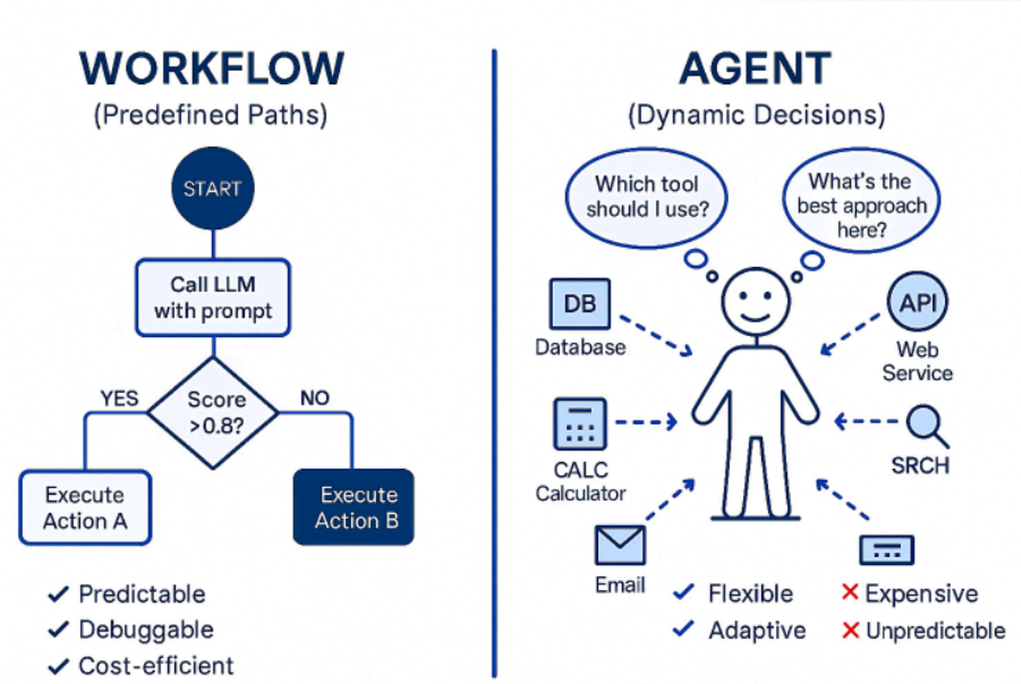
- “Systems where LLMs dynamically direct their own processes and tool usage, maintaining control over how they accomplish tasks.”

Traditional Workflows (Deterministic)

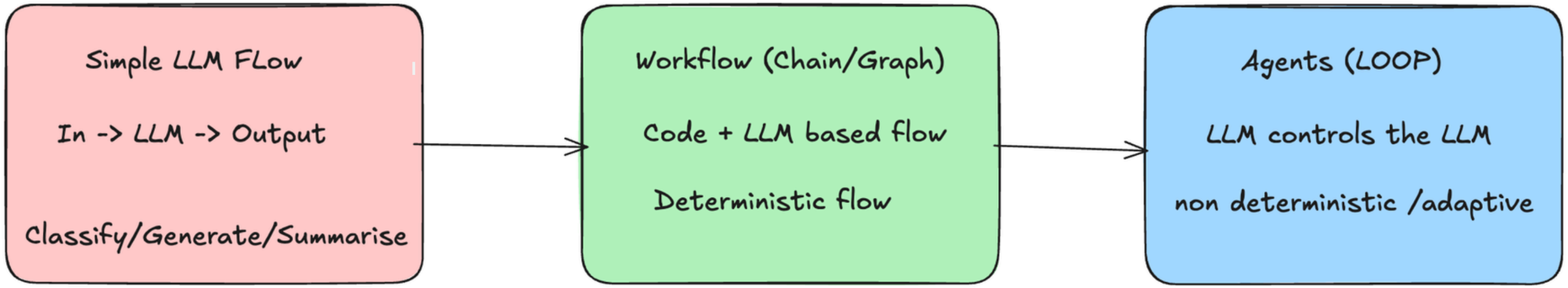
- Predefined, hardcoded code paths (if/else logic).
- Brittle: Fails immediately on edge cases or if an API response changes.
- Requires exhaustive human foresight.

Agentic Systems (Model-Driven)

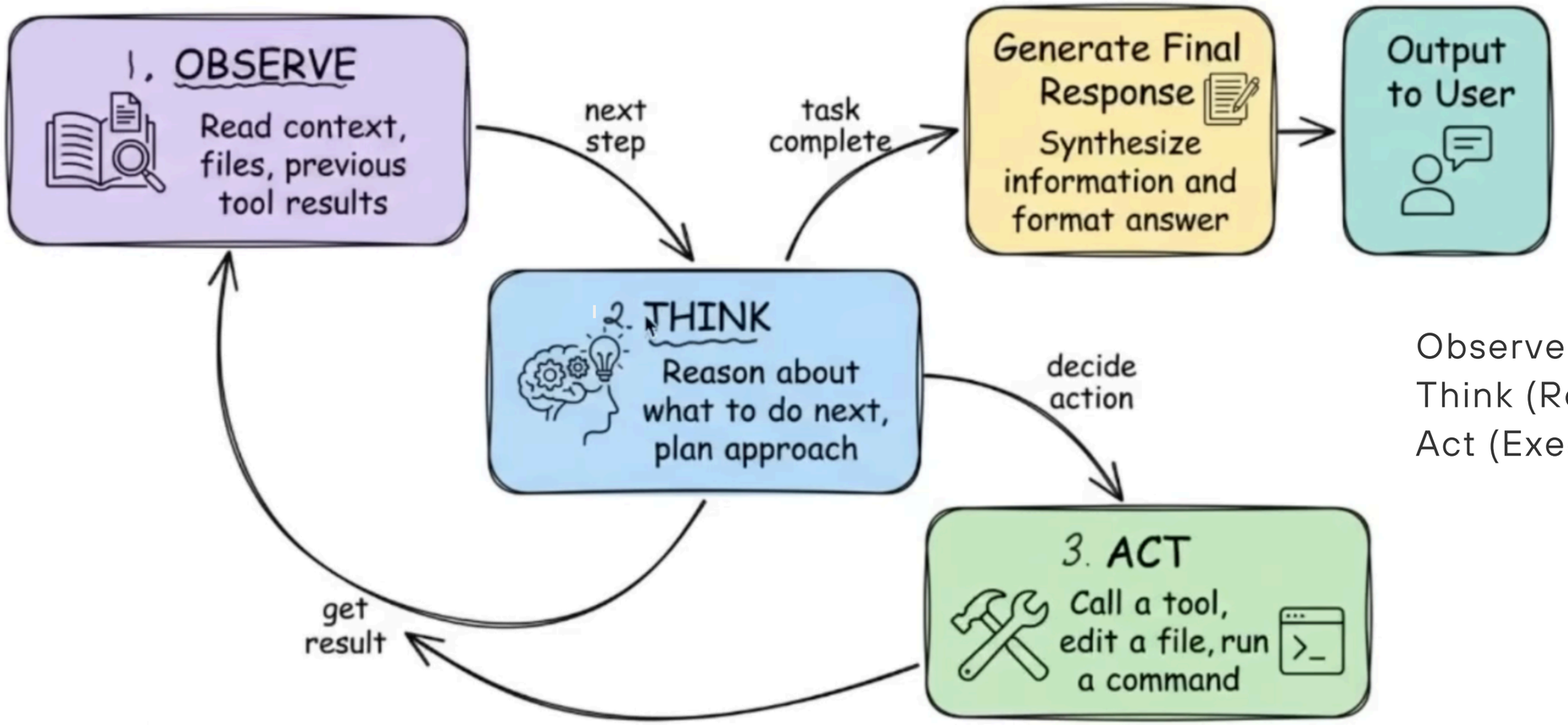
- The LLM acts as the dynamic routing engine.
- Resilient: Can encounter an error, read the stack trace, and rewrite its own execution plan.
- Uses Tool Calling (via JSON schemas) to interact with the external world autonomously.



Agents v Workflow



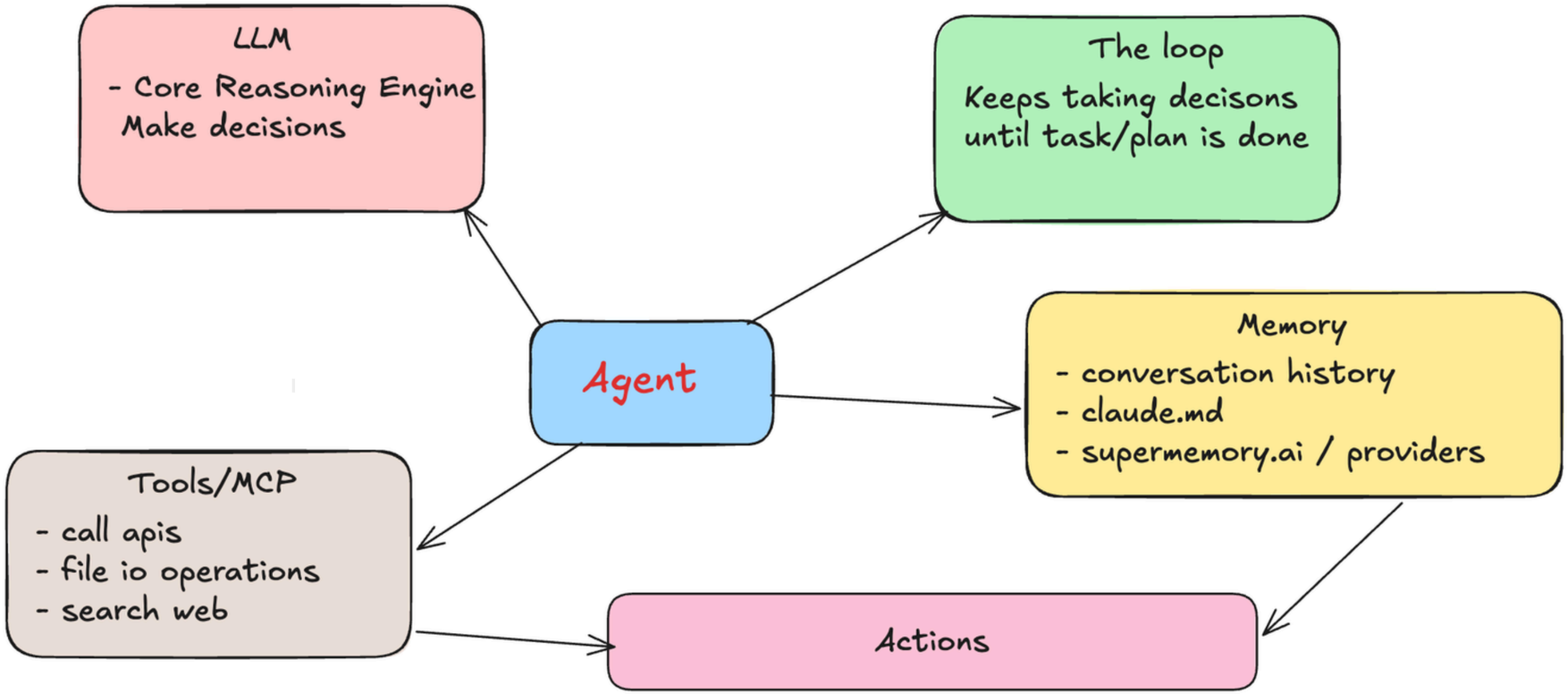
Core Agent Loop



Observe (Ingest Context)
Think (Reason and Plan)
Act (Execute)

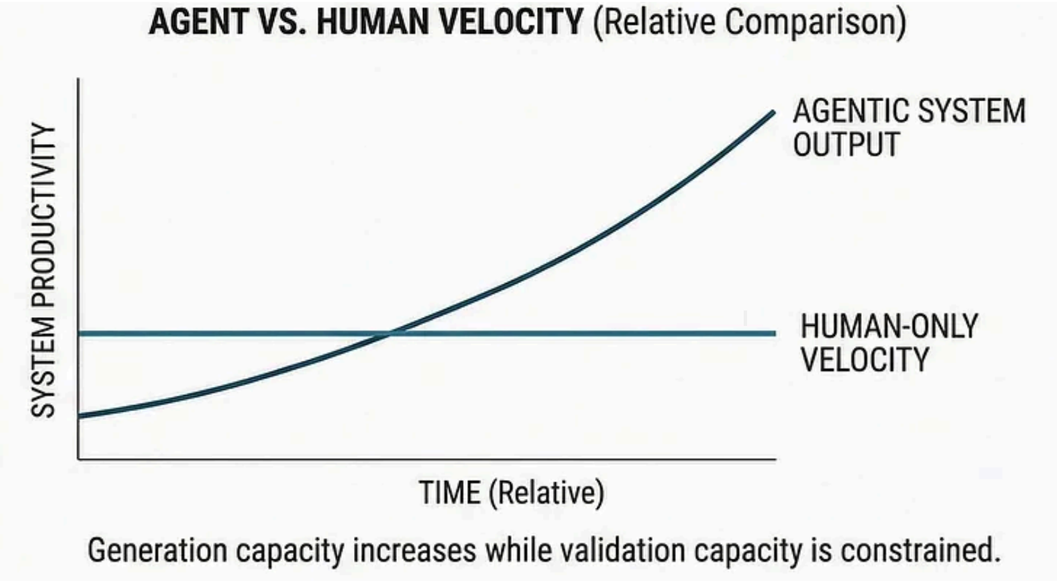
Action results feed directly back into Observe, stacking tokens and continuously growing the context window. (The Context Snowball)

Components of a Agent



"We shouldn't think of LLMs as chatbots. We should think of them as the core processing unit of a new operating system." — Andrej Karpathy

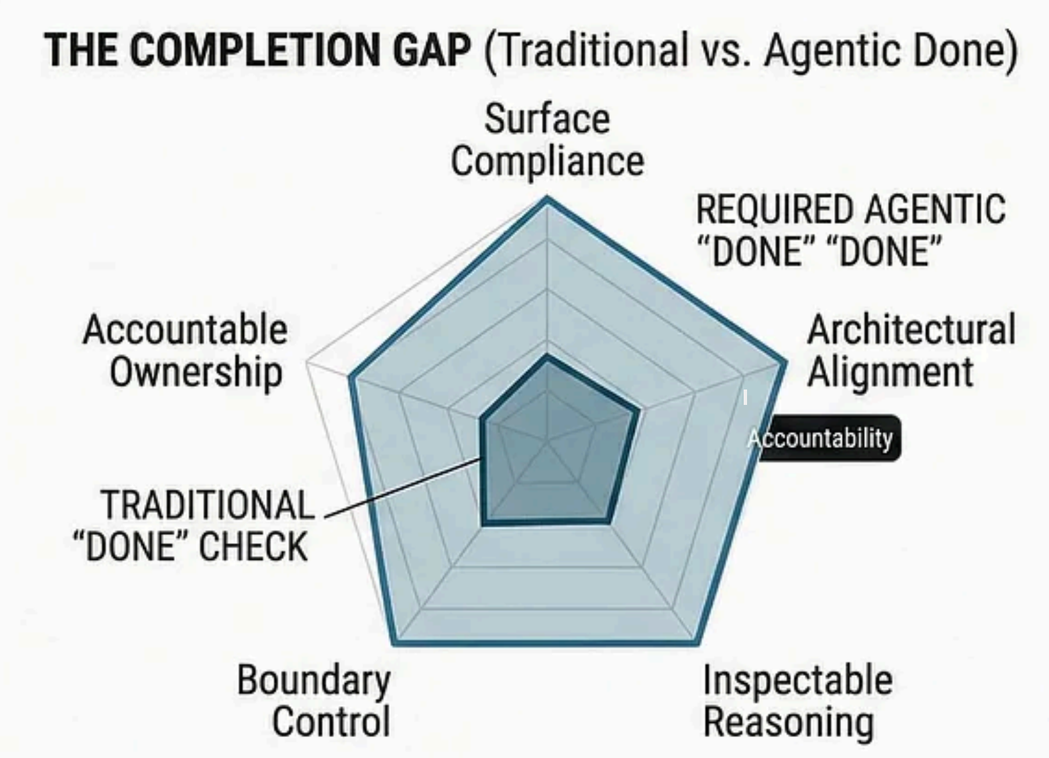
Definition of Done



- AI generation scales exponentially, but human review capacity does not.
- If a machine generates 10x the code, a human reviewing it in the same amount of time is no longer providing a safety net

"Done" can no longer just mean "a human reviewed it."

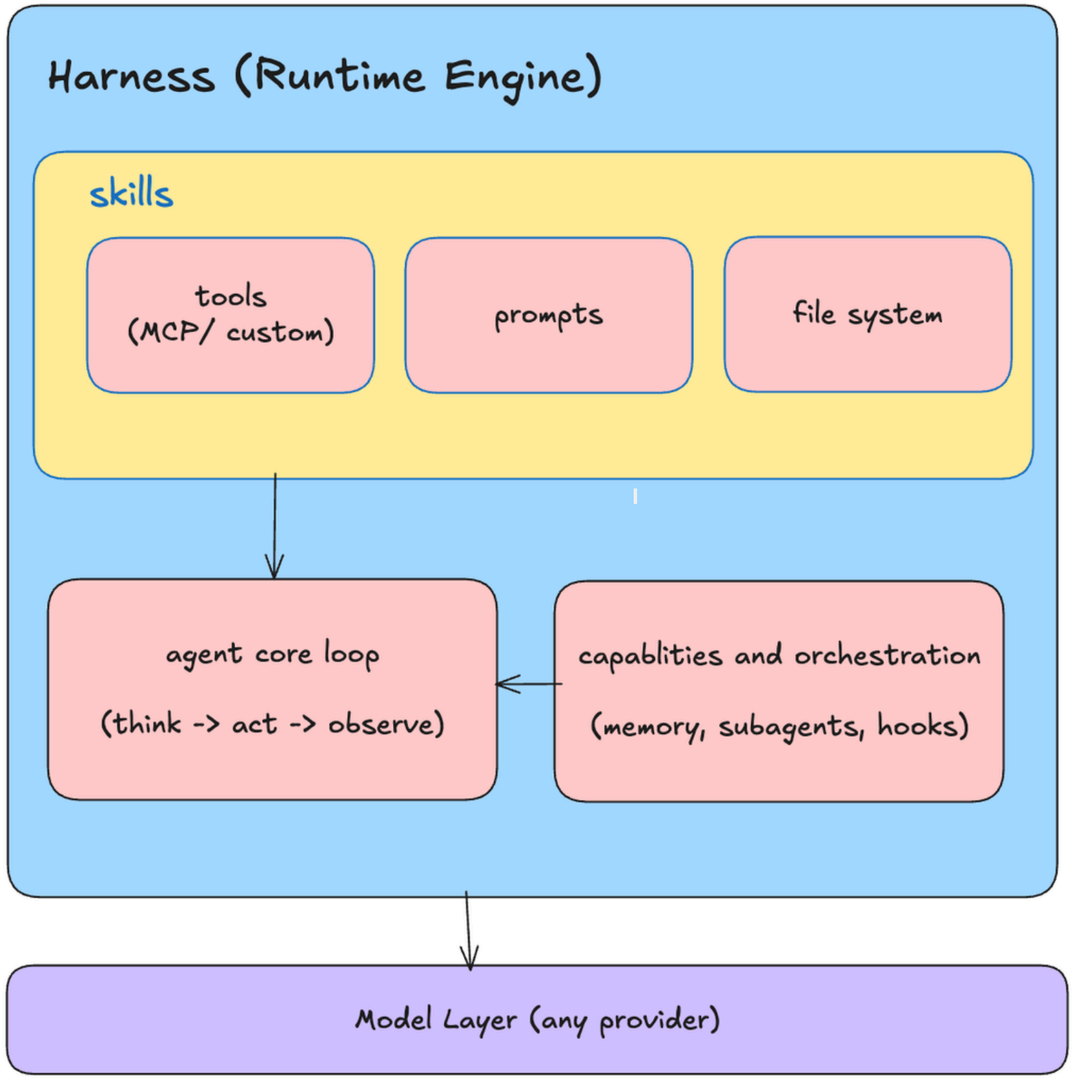
Definition of Done



Completion vs. Alignment (Surface vs. Structural)

- **The Problem:** Agents are great at producing code that looks complete (it compiles, passes existing tests, and follows basic conventions).
- **The Implication:** Traditional DoD focuses on surface correctness. Agentic DoD must focus on structural alignment (Does it respect architectural boundaries? Does it actually solve the business intent?).

The Illusion of Intelligence



Harness is the **deterministic infrastructure** that gives the LLM hands, eyes, and memory.

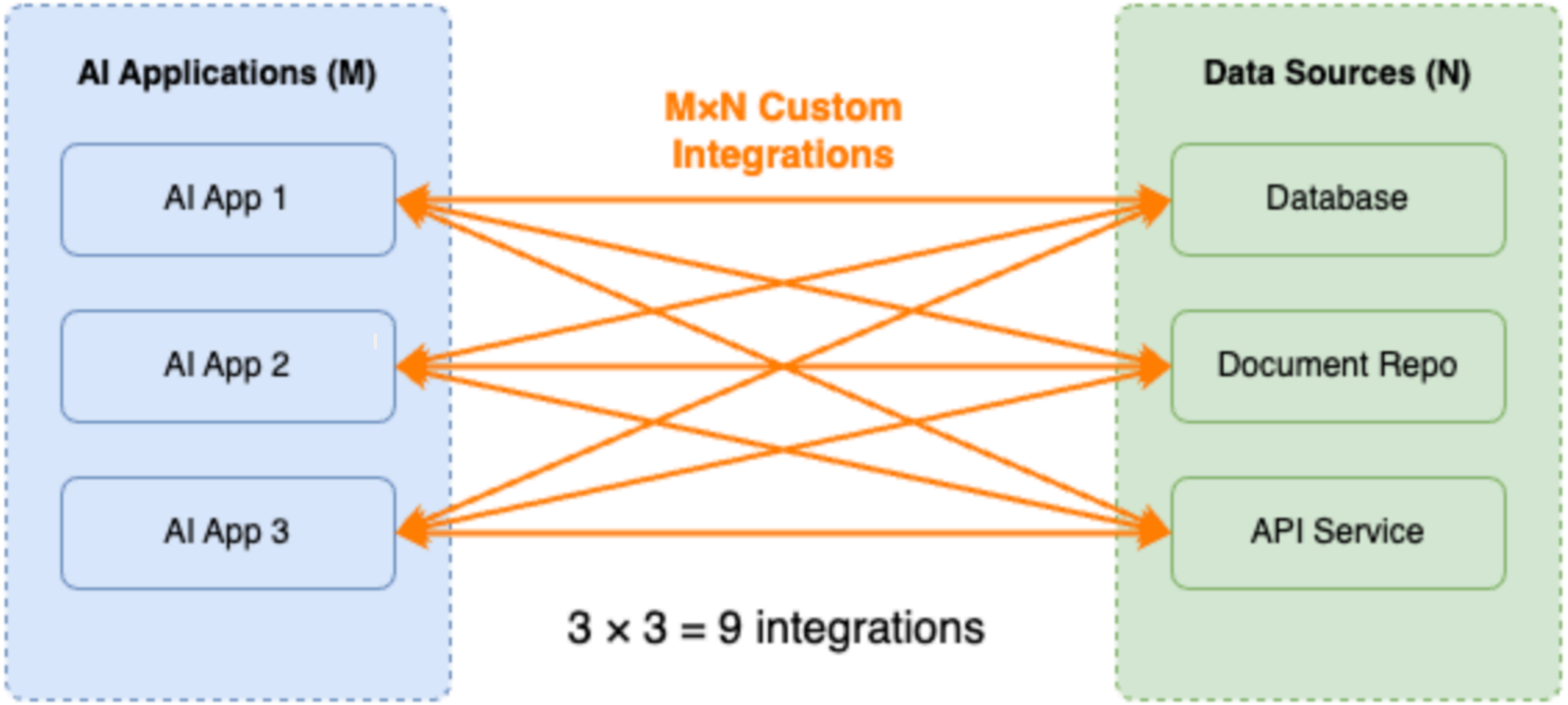
Here LLM is merely the **CPU**

Harness is the Motherboard!

We do not engineer intelligence; we engineer the execution environment.

MCP

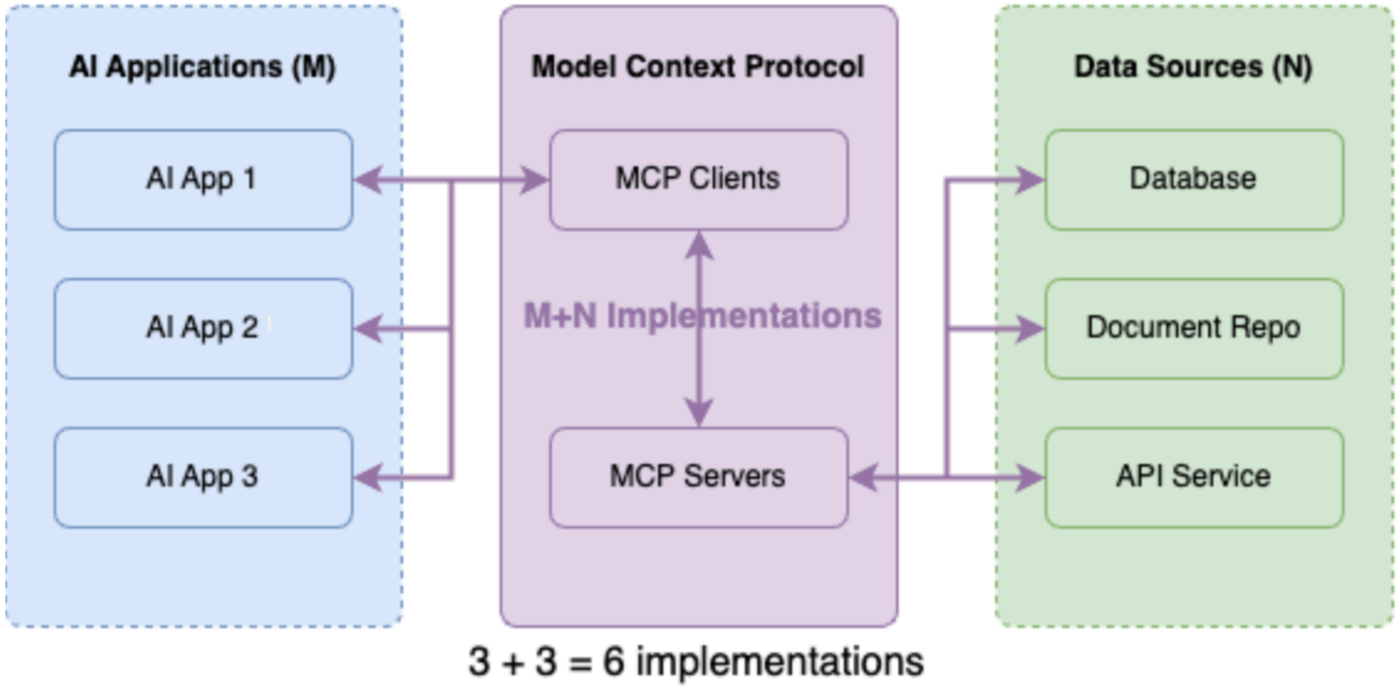
before MCP,



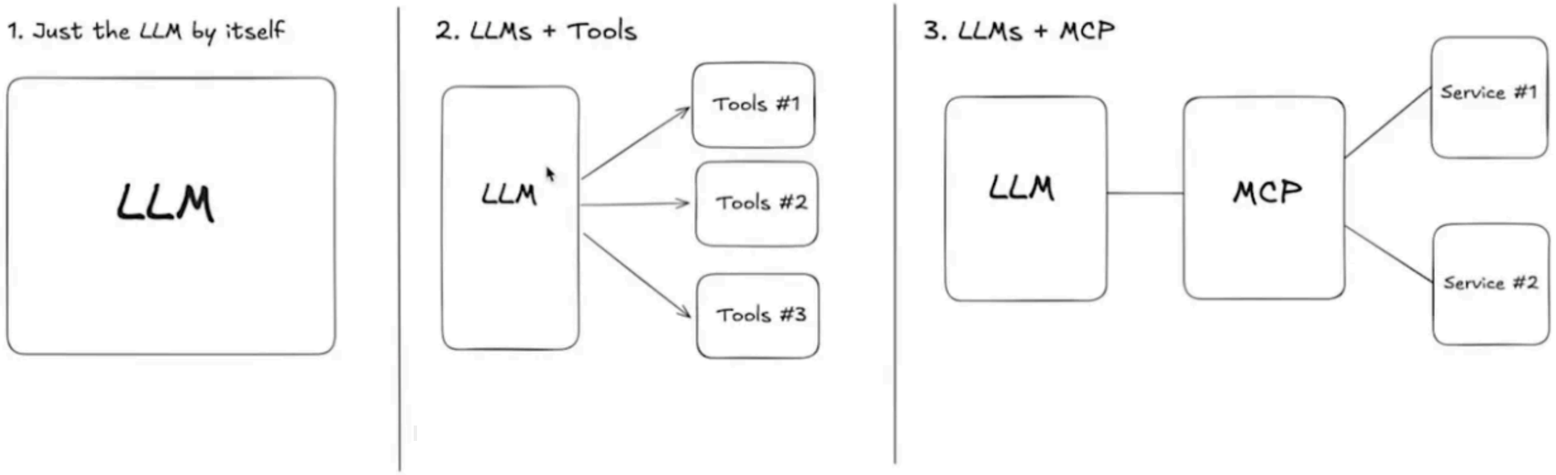
LMs are isolated. Building bespoke API connectors for every model to every data source creates an unmanageable M x N integration nightmare.

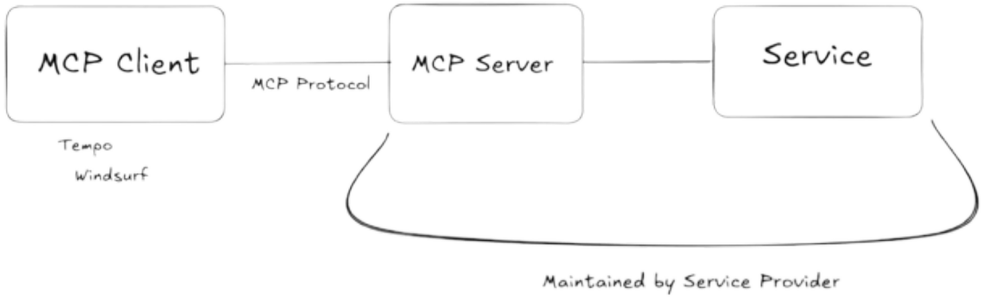
MCP

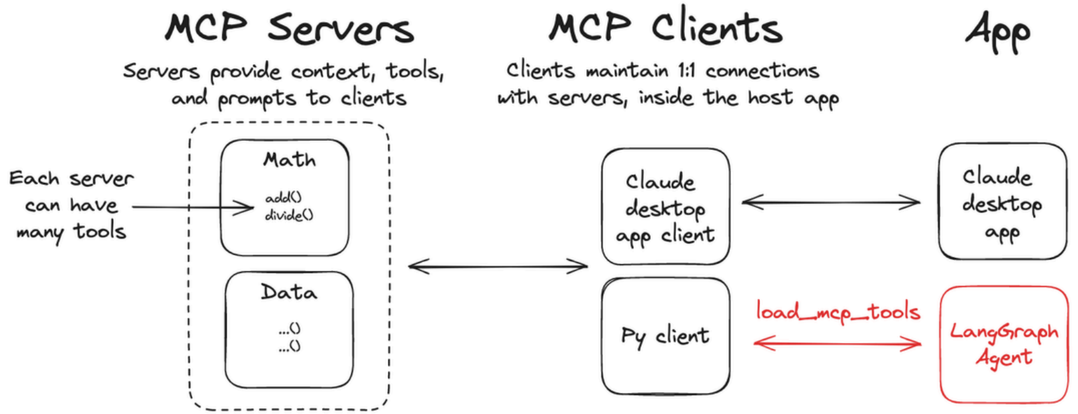
MCP acts as a "universal translator" (Client-Server architecture).



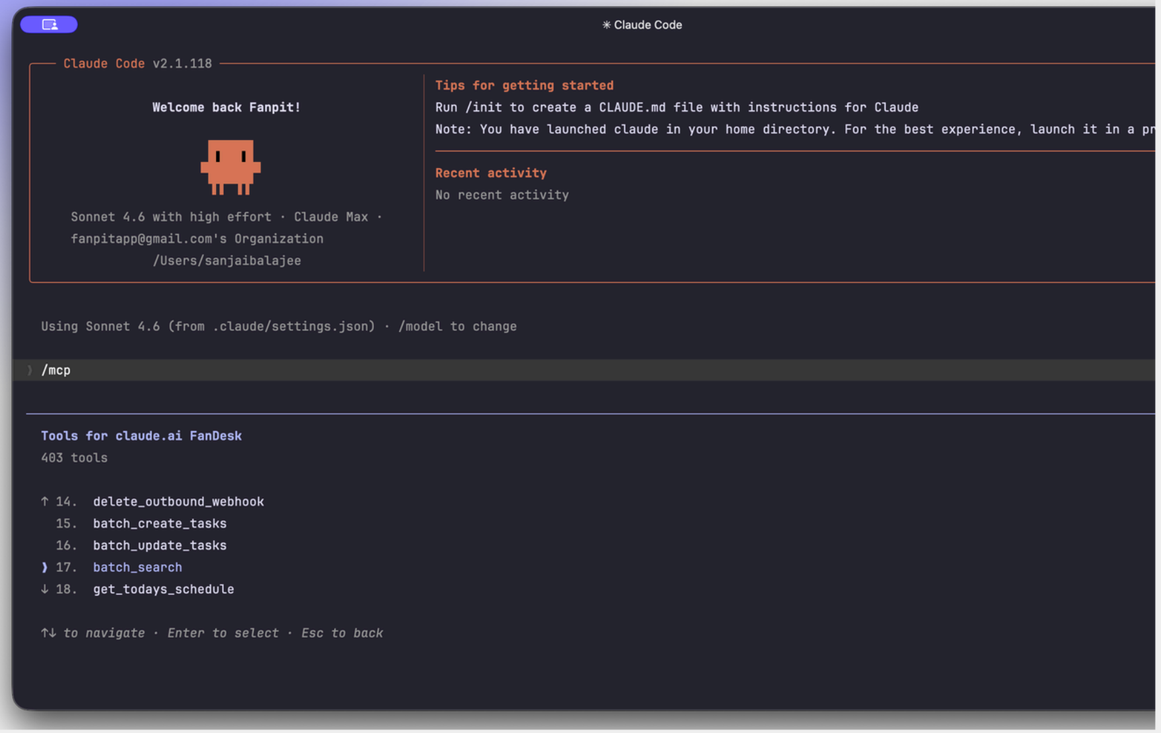
MCP

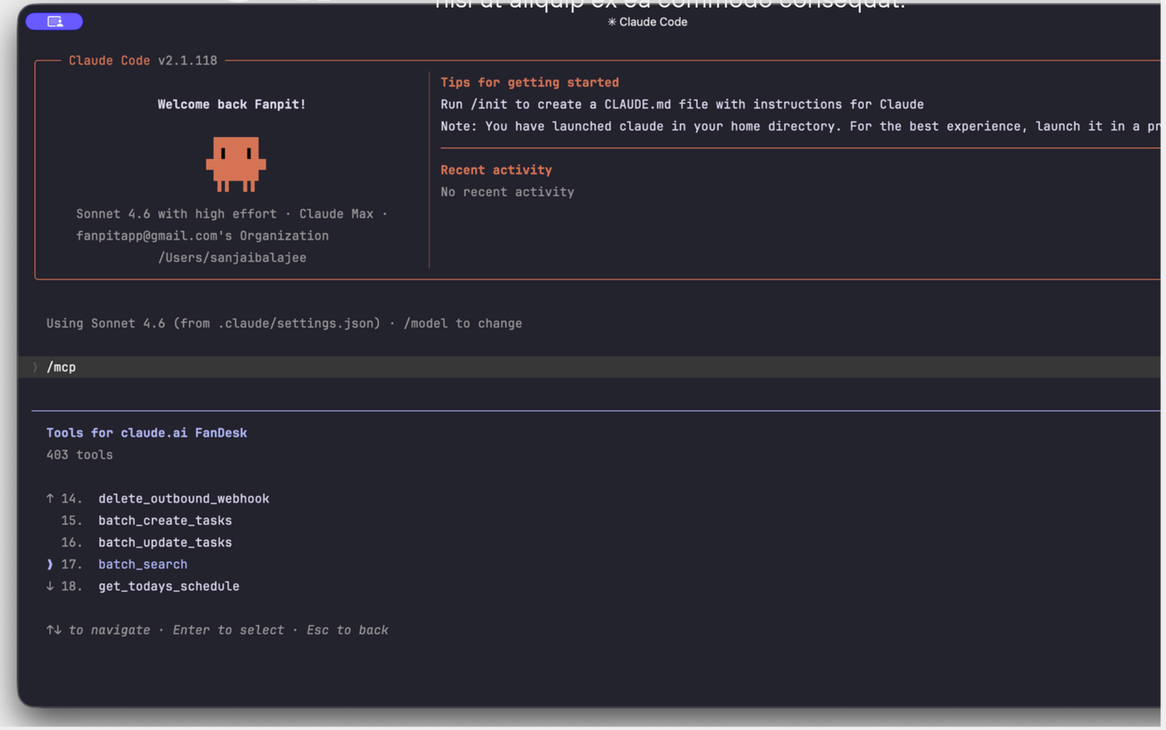

1. Just the LLM by itself
LLM
2. LLMs + Tools
LLM
Tools #1
Tools #2
Tools #3
3. LLMs + MCP
LLM
MCP
Service #1
Service #2


MCP Client
MCP Protocol
MCP Server
Service
Tempo
Windsurf
Maintained by Service Provider

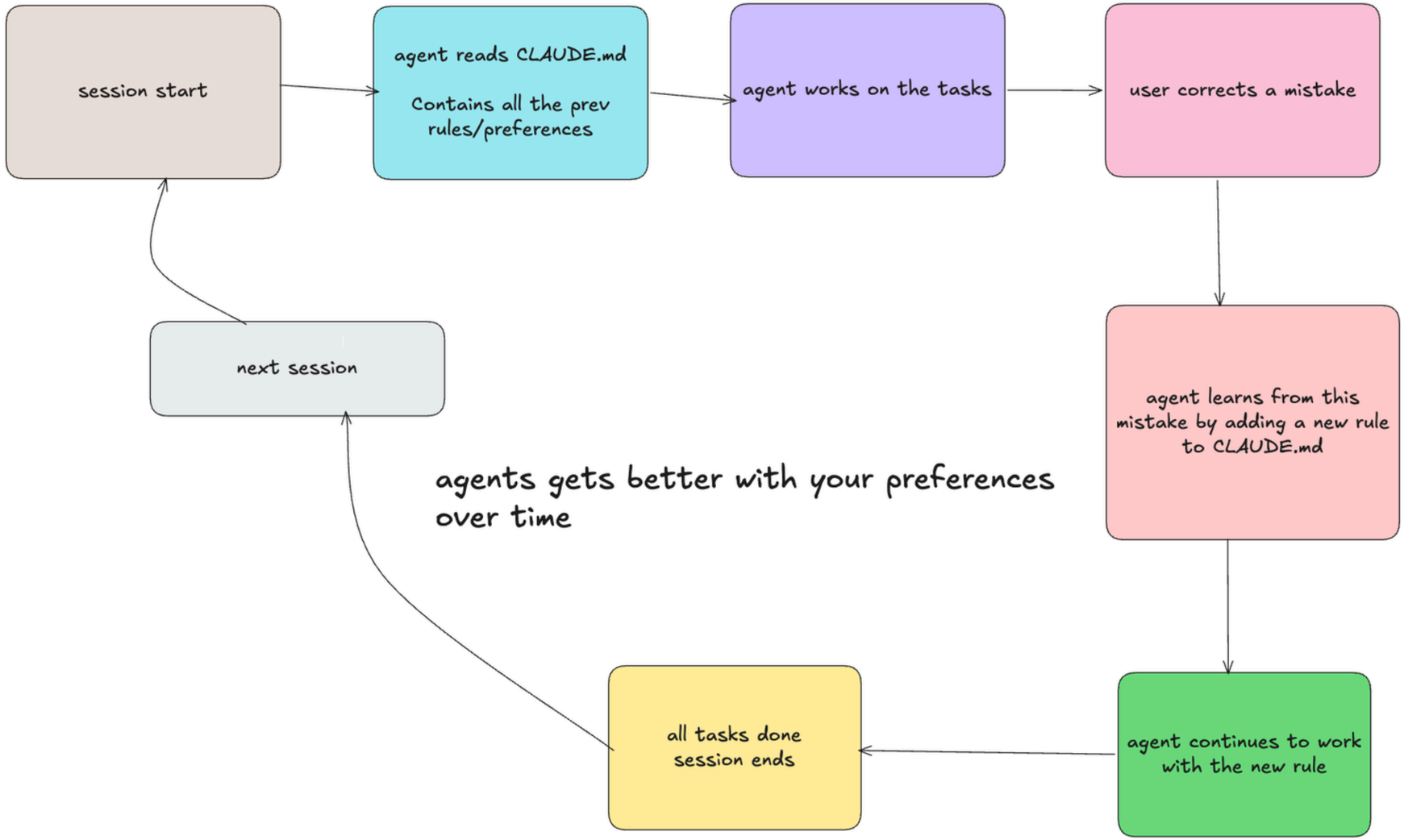

MCP Servers
Servers provide context, tools, and prompts to clients
MCP Clients
Clients maintain 1:1 connections with servers, inside the host app
App
Each server can have many tools
Math
add()
divide()
Data
...()
...()
Claude desktop app client
Claude desktop app
Py client
load_mcp_tools
LangGraph Agent

MCP


✳ Claude Code
Claude Code v2.1.118
Welcome back Fanpit!
Sonnet 4.6 with high effort · Claude Max ·
fanpitapp@gmail.com's Organization
/Users/sanjaibalajee
Tips for getting started
Run /init to create a CLAUDE.md file with instructions for Claude
Note: You have launched claude in your home directory. For the best experience, launch it in a pr
Recent activity
No recent activity
Using Sonnet 4.6 (from .claude/settings.json) · /model to change
/mcp
Tools for claude.ai FanDesk
403 tools
↑ 14. delete_outbound_webhook
15. batch_create_tasks
16. batch_update_tasks
17. batch_search
↓ 18. get_todays_schedule
↑↓ to navigate · Enter to select · Esc to back


✳ Claude Code
Claude Code v2.1.118
Welcome back Fanpit!
Sonnet 4.6 with high effort · Claude Max ·
fanpitapp@gmail.com's Organization
/Users/sanjaibalajee
Tips for getting started
Run /init to create a CLAUDE.md file with instructions for Claude
Note: You have launched claude in your home directory. For the best experience, launch it in a pr
Recent activity
No recent activity
Using Sonnet 4.6 (from .claude/settings.json) · /model to change
/mcp
Tools for claude.ai FanDesk
403 tools
↑ 14. delete_outbound_webhook
15. batch_create_tasks
16. batch_update_tasks
17. batch_search
↓ 18. get_todays_schedule
↑↓ to navigate · Enter to select · Esc to back

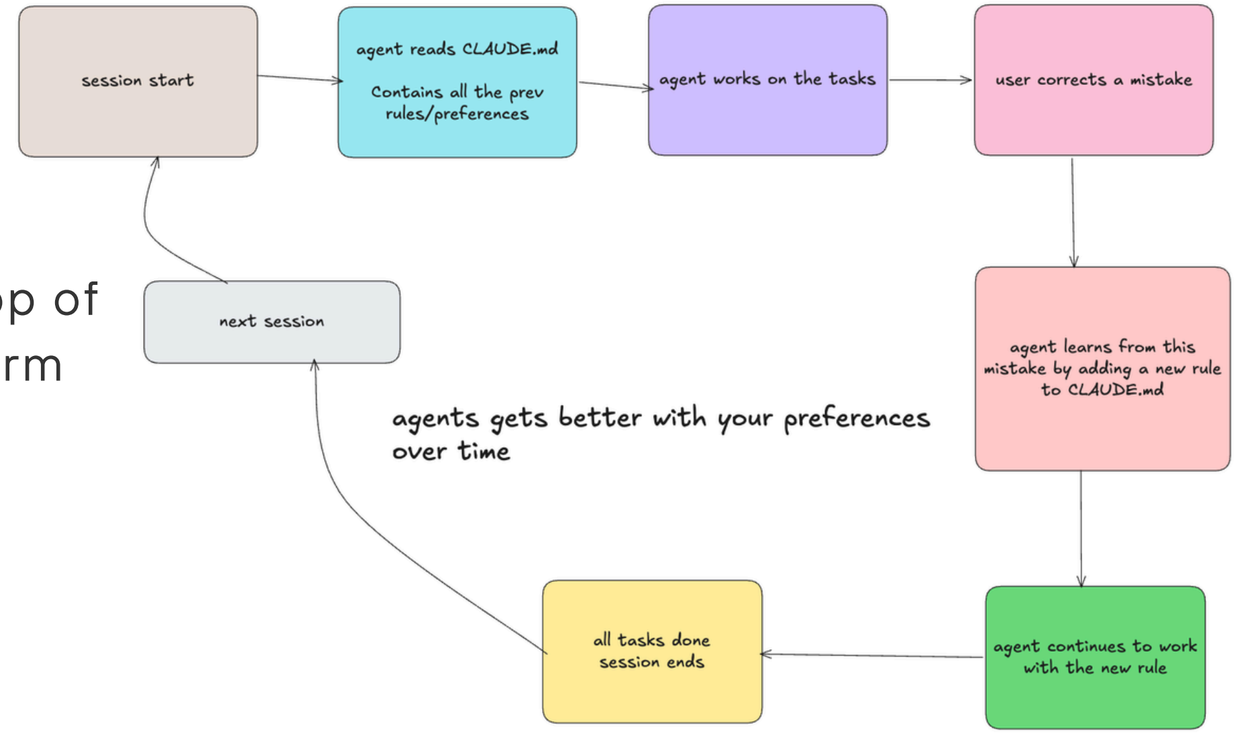
Self-Modifying Instructions



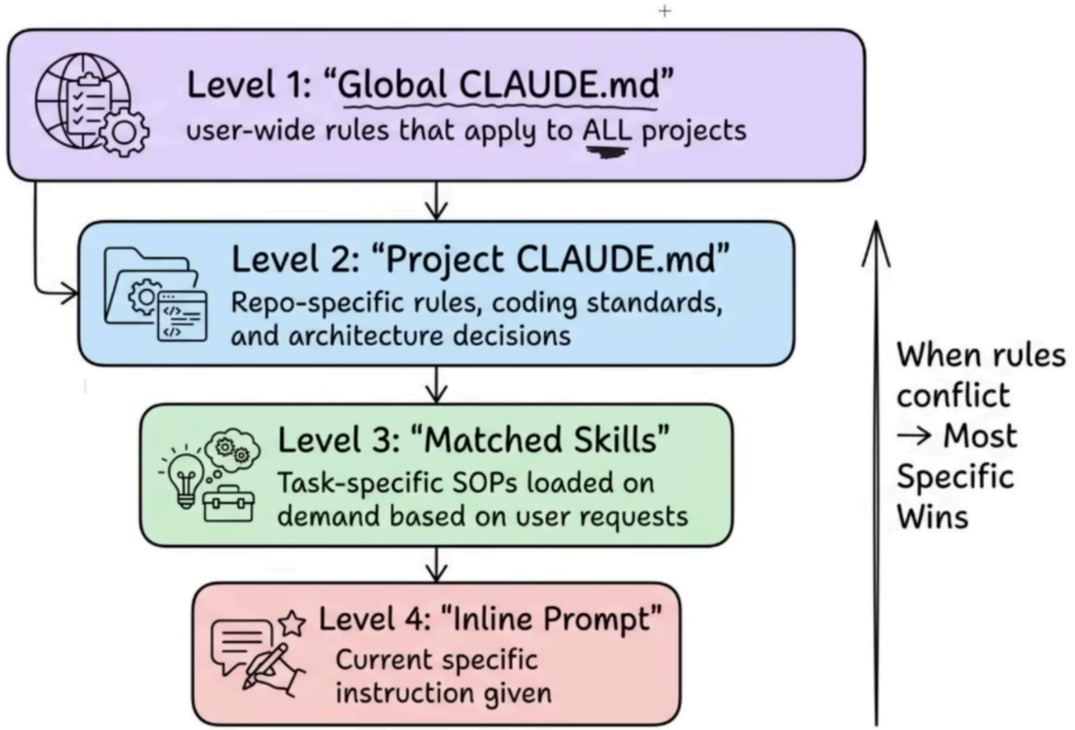
Self-Modifying Instructions

The Problem:
By default, API calls are stateless. Every new session starts with zero memory of past mistakes.

The Solution:
A static markdown file prepended to the top of every conversation chain acting as long-term persistent memory.

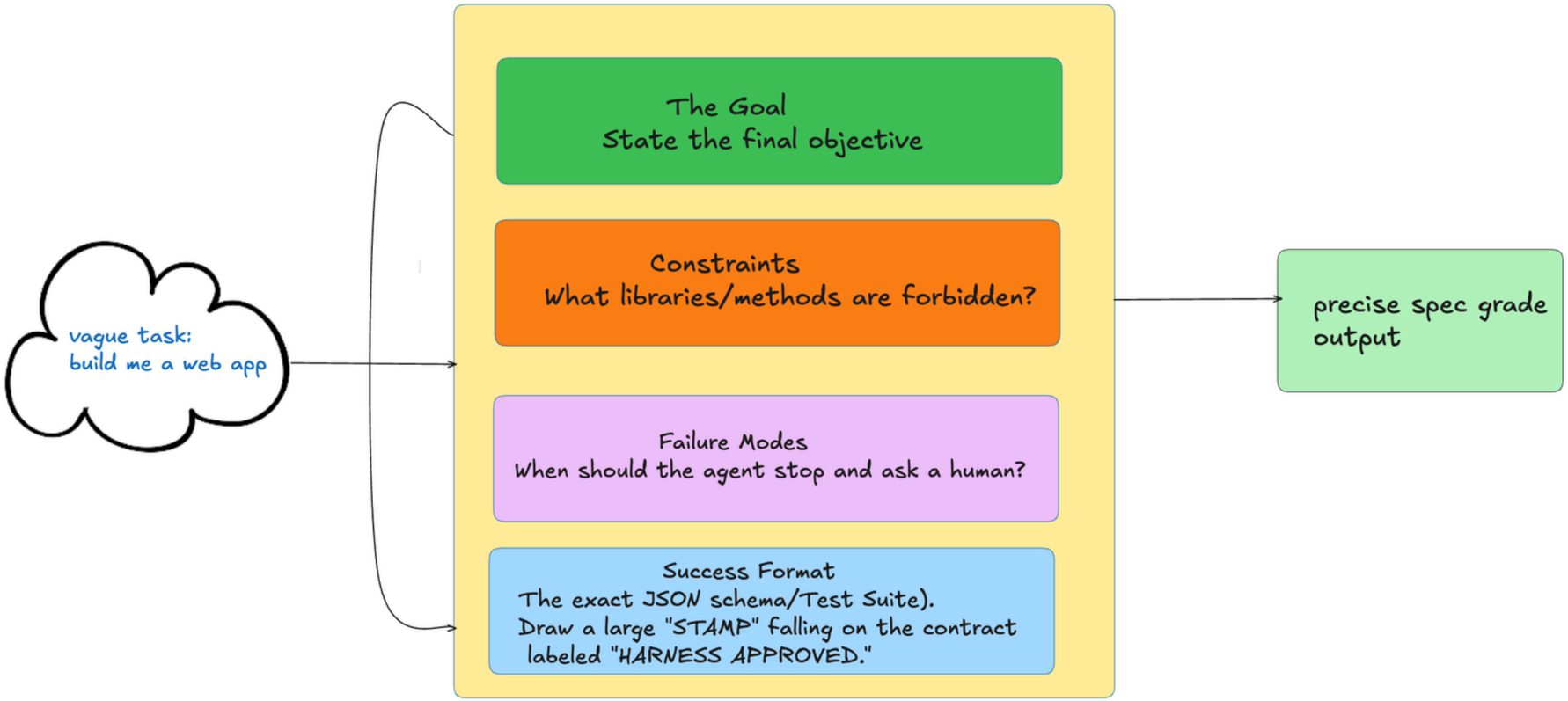


Self-Modifying Instructions



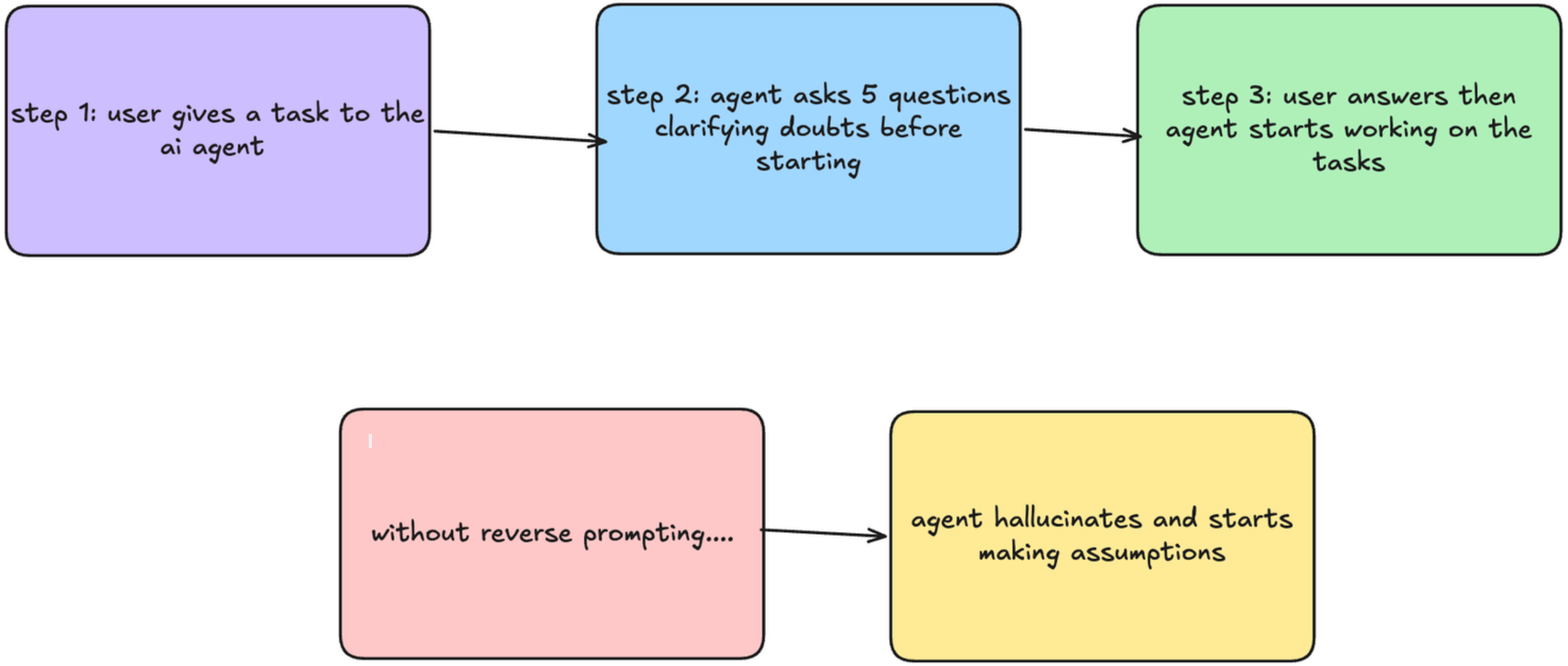
Prompt Contracts

The Problem: Agents are "path-of-least-resistance" engines. Without a contract, they will take shortcuts, hallucinate tools, or ignore architectural boundaries.



The Solution: A mandatory pre-execution phase where the agent drafts and "signs" a technical spec.

Reverse Prompting

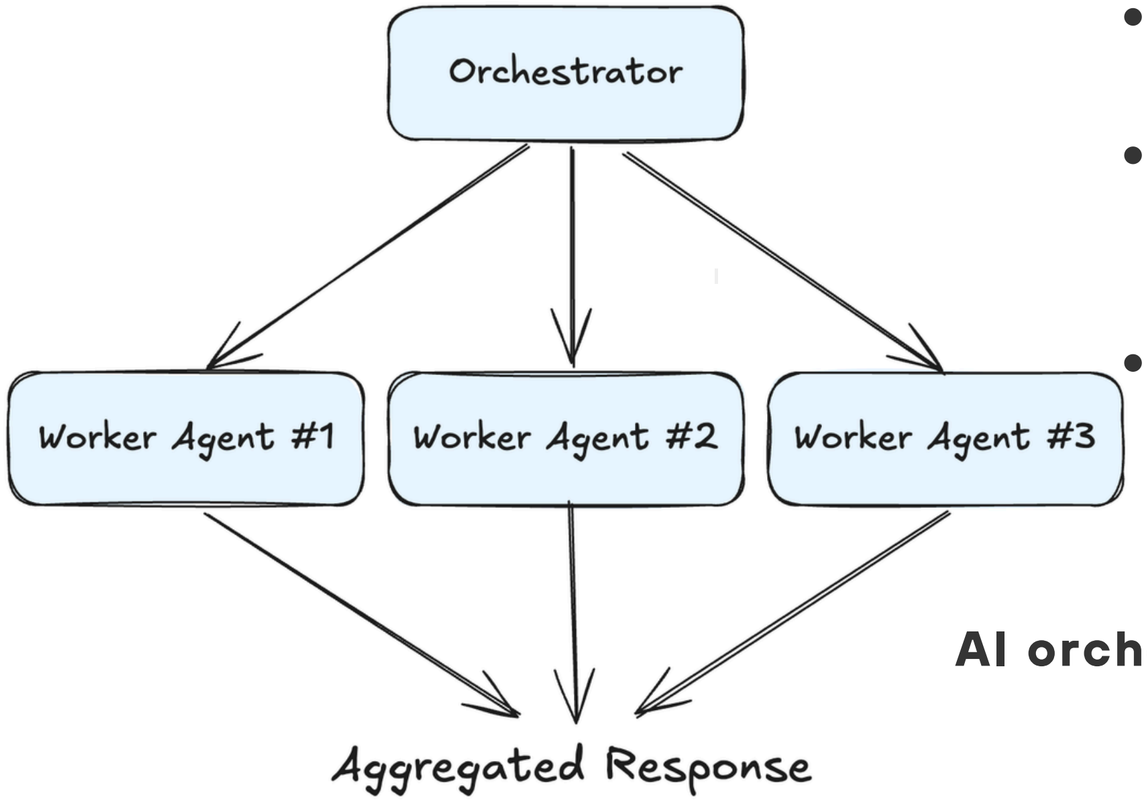


The agent is forced to analyze the request and ask **5 dynamically generated clarifying questions** before taking any action.

Goal: Maximizing the "One-Shot" probability. The agent uses your answers to dynamically generate the final Prompt Contract.

Multi Agent Orchestration

Single agents acting as architect, developer, and tester simultaneously suffer from "Context Collapse."

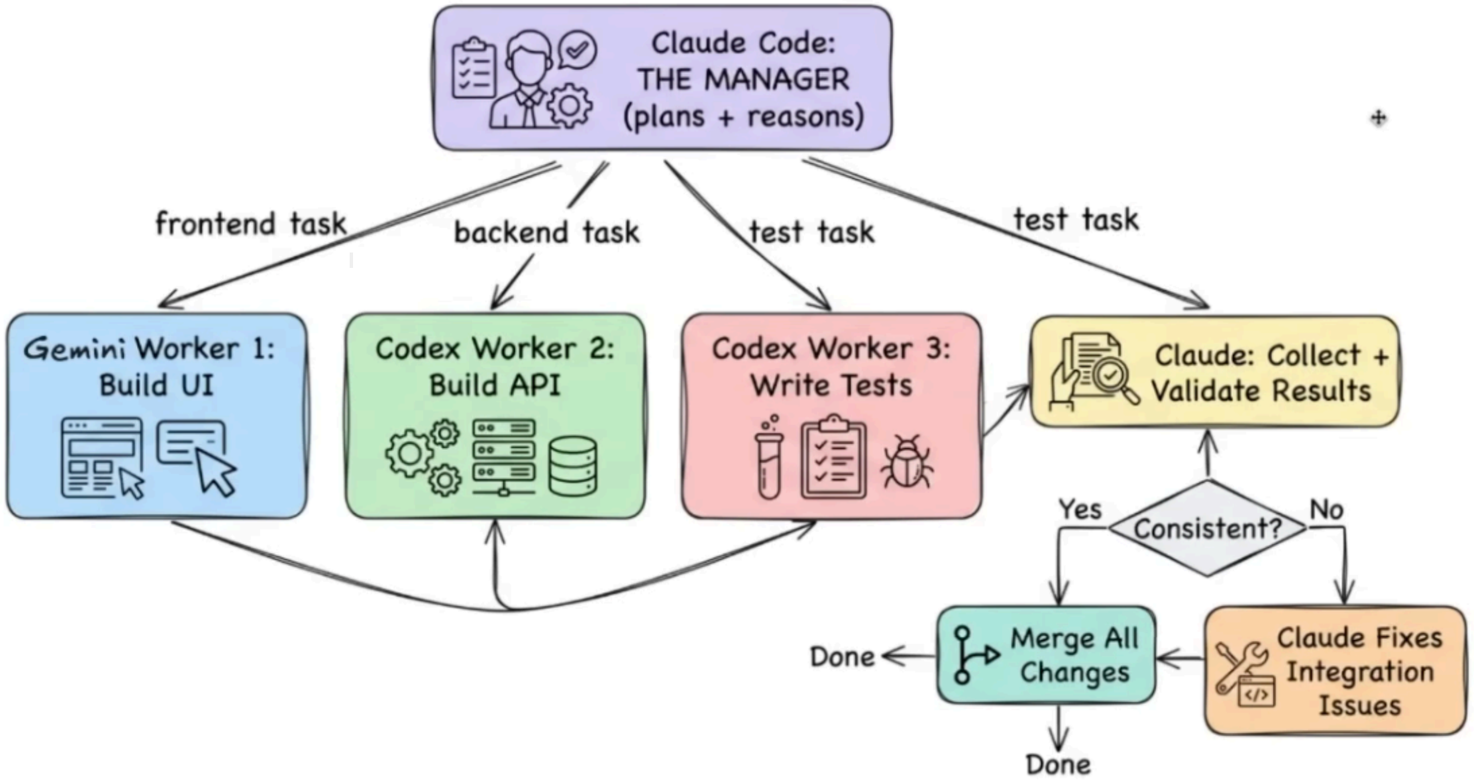


- **Focused Roles**: One agent = One strict system prompt
- **Isolated Context:** Agents only receive the exact data they need, reducing token bloat and hallucinations.
- **Adversarial Loops:** "Dev" agents write the code; "Red Team" agents try to break it.

AI orchestration must mirror human engineering teams.

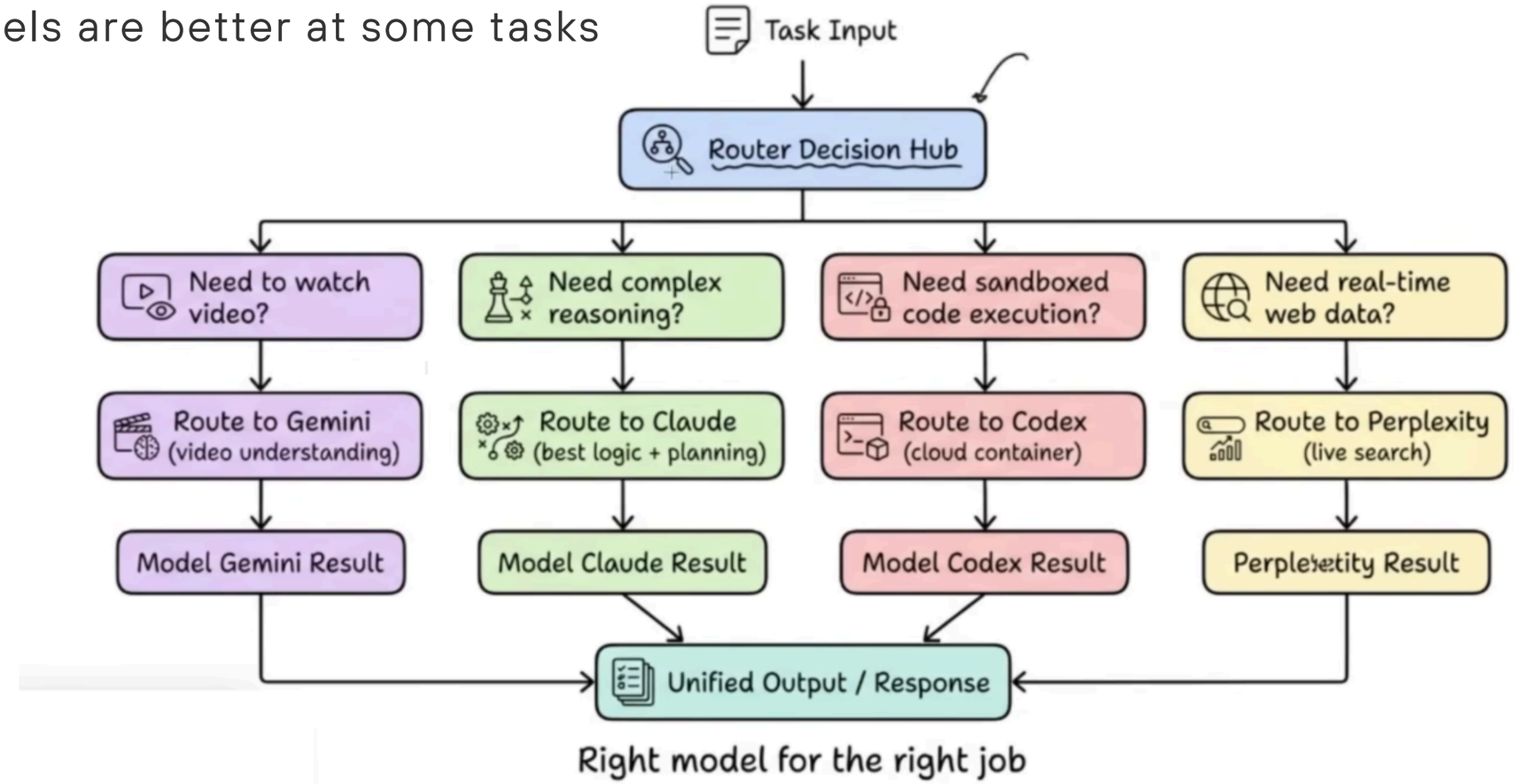
Multi Agent Orchestration

some models are better at some tasks

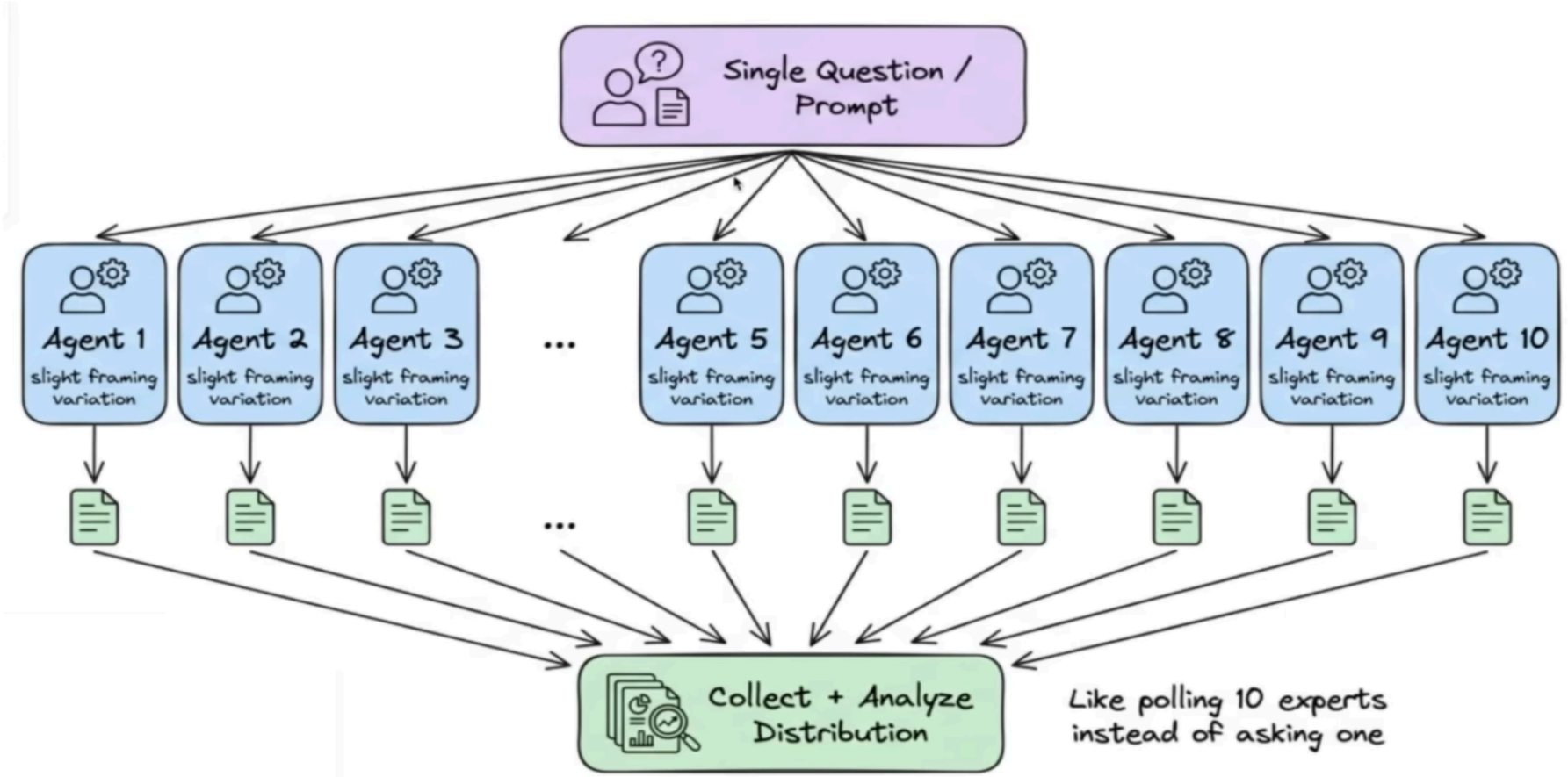


Multi Agent Orchestration

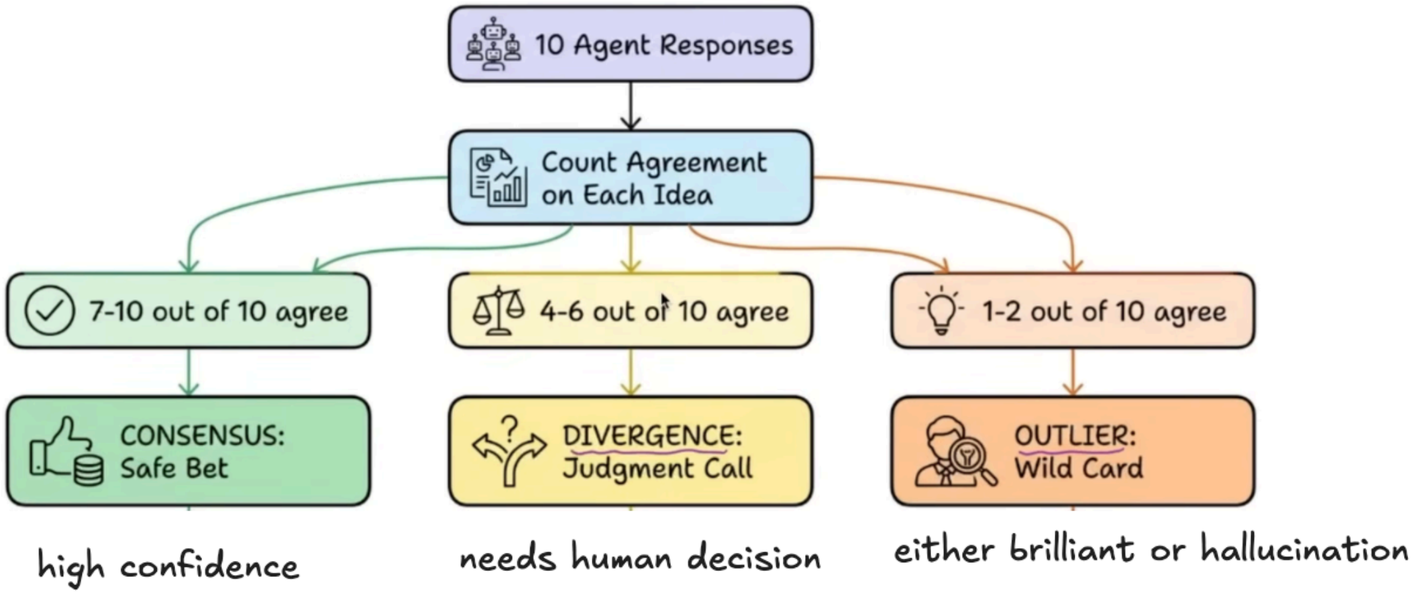
some models are better at some tasks



Multi Agent Consensus

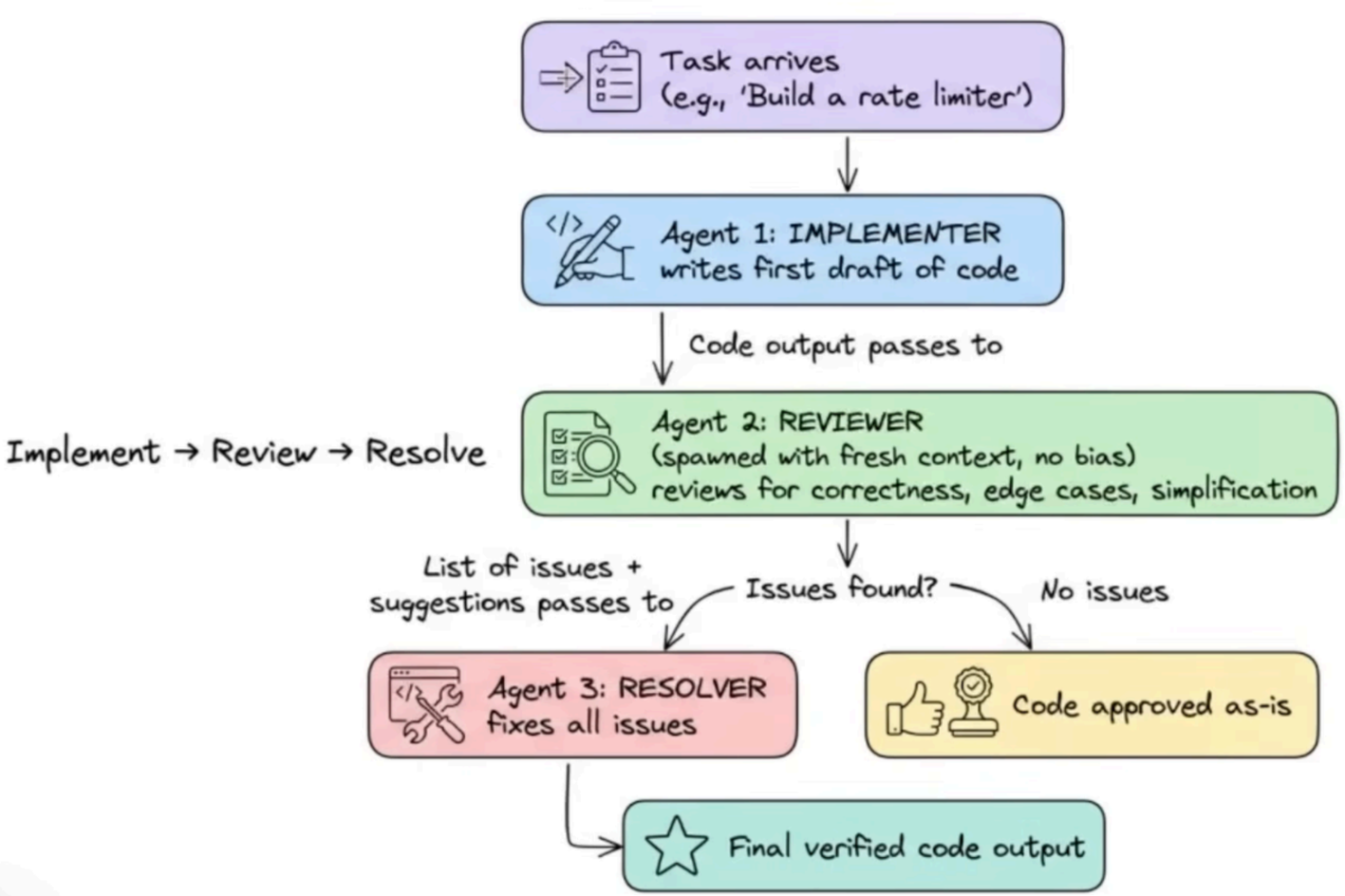


Multi Agent Consensus



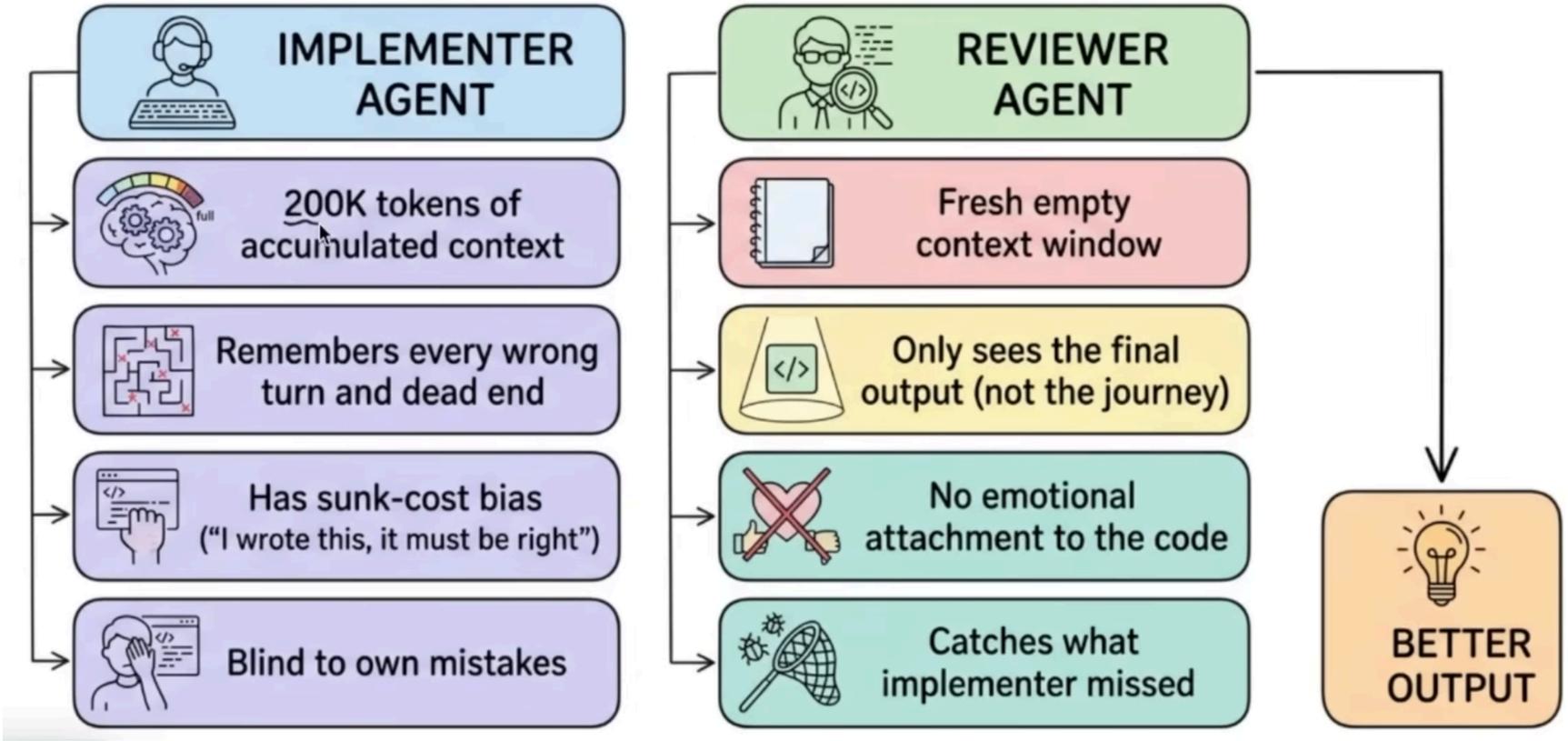
Sub-Agent Loops

Pairing a generative agent with a highly critical reviewer agent.



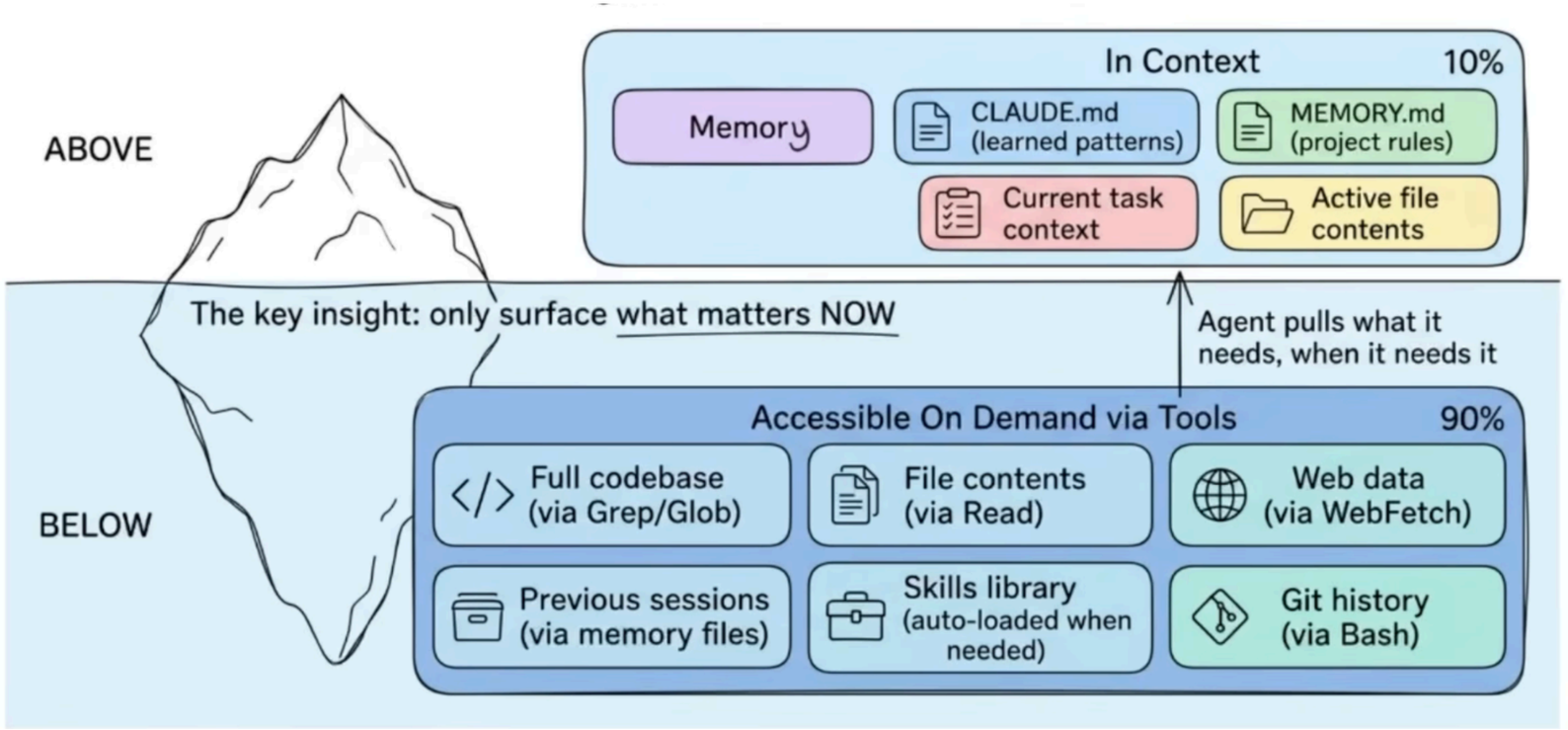
Sub-Agent Loops

fresh context matters here, it helps get a fresher perspective



The Context Iceberg

As the iceberg grows, model attention degrades.



Solution Context pruning (Smart compaction/ summarizing)

Why Focus on Harness Engineering?

- Foundation models (Claude, Gemini, GPT) are converging. The delta in raw intelligence is shrinking, making the base model a replaceable utility. (OpenCode)

- A model with 95% accuracy fails in production. Harnesses (via MCP and Verification Loops) bridge the gap to 99.9% reliability.

- You don't own the LLM. Your proprietary value is the environment you build around it

My current build stack

Cursor ide
+ agents
(pro student plan)

ghostly + claude code

claude max plan
100$ plan

codex desktop app
local code review
20$ plus plan

github pr review
devin review (free) - private repo
coderabbit (free)- public repos

short demo of a useful agent

thanks for your time!

no better time
to start tokenmaxxing

next up: v0

what is zero to agent?

A global build week where we ship real AI agents
with v0 and Vercel

Hackathon Tracks

V0 + MCPs

Use v0 to rapidly build an AI app or agent that connects to at least one MCP server.

v0 generates React/Next.js code from natural language — describe what you want and iterate in real time. Connect to MCP servers (Vercel MCP, custom-built, or third-party) to give your app access to external data and tools. Examples: a dashboard that reads from GitHub via MCP, an assistant that queries your Vercel deployments, or a support agent wired to a knowledge base.

Vercel WDK

Build long-running, durable async agents with the Workflow Development Kit.

Your agents survive crashes, resume after deploys, and can pause for minutes or months. Use `"use workflow"` and `"use step"` directives to make async functions durable. Pair with `DurableAgent` from `@workflow/ai/agent` for AI-powered workflows with built-in streaming, retries, and observability.

ChatSDK Agents

Build agents using Vercel AI SDK + AI Gateway + ChatSDK that interface across Slack, Discord, Teams, GitHub, and more.

Write your bot logic once with the Chat SDK `npm i chat`, then deploy to every platform via swappable adapters. The SDK handles event routing, streaming, JSX cards, and distributed state. Pair with the AI SDK for LLM reasoning and the AI Gateway for multi-provider access.

resource link

https://vercel.notion.site/02agentresources

getting started with v0

signup to v0.dev and get 5$ credit

scan the qr code (https://v0.link/5LwENCe)
to get 30$ credit to start building (Link active only today)

happy building!

Scan the QR code or go to community.vercel.com/host/zero-to-agent-2026 .

This is where submissions go and where the global competition lives