The Age of Agentic AI

Mastering Claude Code & Cowork: From Passive Chatbots to Digital FTEs
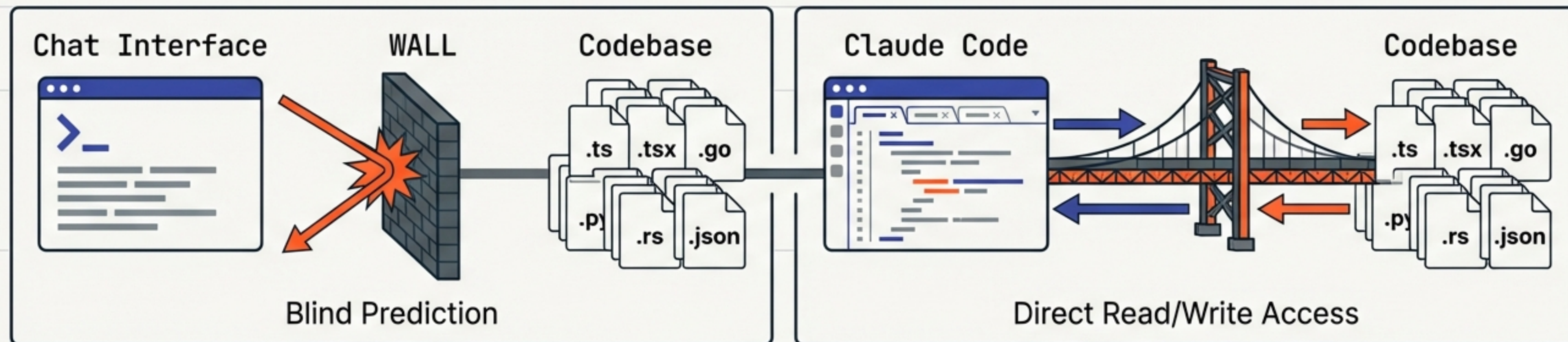

Passive AI (2023)

Agentic AI (2025+)

System Architecture

File System — JetBrains Hono

Browser — JetBrains Mene

Agentic Brain

Terminal — JetBrains Hono

```
> execute_script.sh
process_data()
  preess_data()
  ...
}
>
```

API — JetBrains Mene

Token

Response

Action/Hooks

MCP — JetBrains Hono

Model Context Protocol

Context Management

Memory Retrieval

Toel Use Request

Feedback Loop
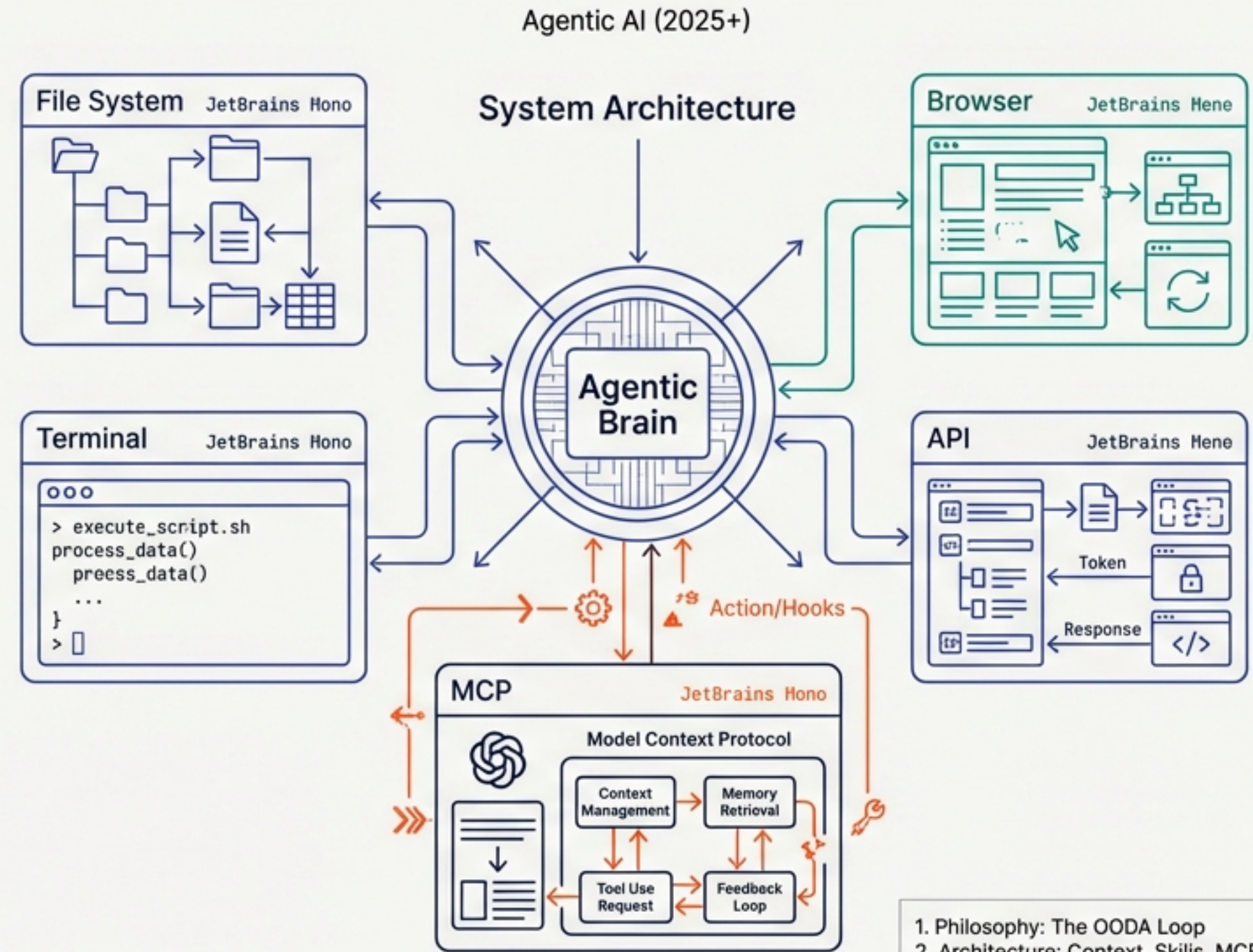
1. Philosophy: The OODA Loop
2. Architecture: Context, Skilis, MCP
3. Orchestration: Subagents & Hooks
4. Business: The Digital FTE

NotebookLM

The Product Overhang

Capabilities Waiting for Access

Chat Interface — WALL — Codebase

.ts .tsx .go .py .rs .json

Blind Prediction

Claude Code — Codebase

.ts .tsx .go .py .rs .json

Direct Read/Write Access

90%
of Claude Code codebase written by Claude Code itself.

20% → 80%
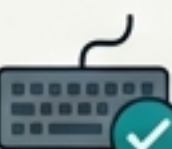Internal daily usage jump at Anthropic (Nov '24 - May '25).
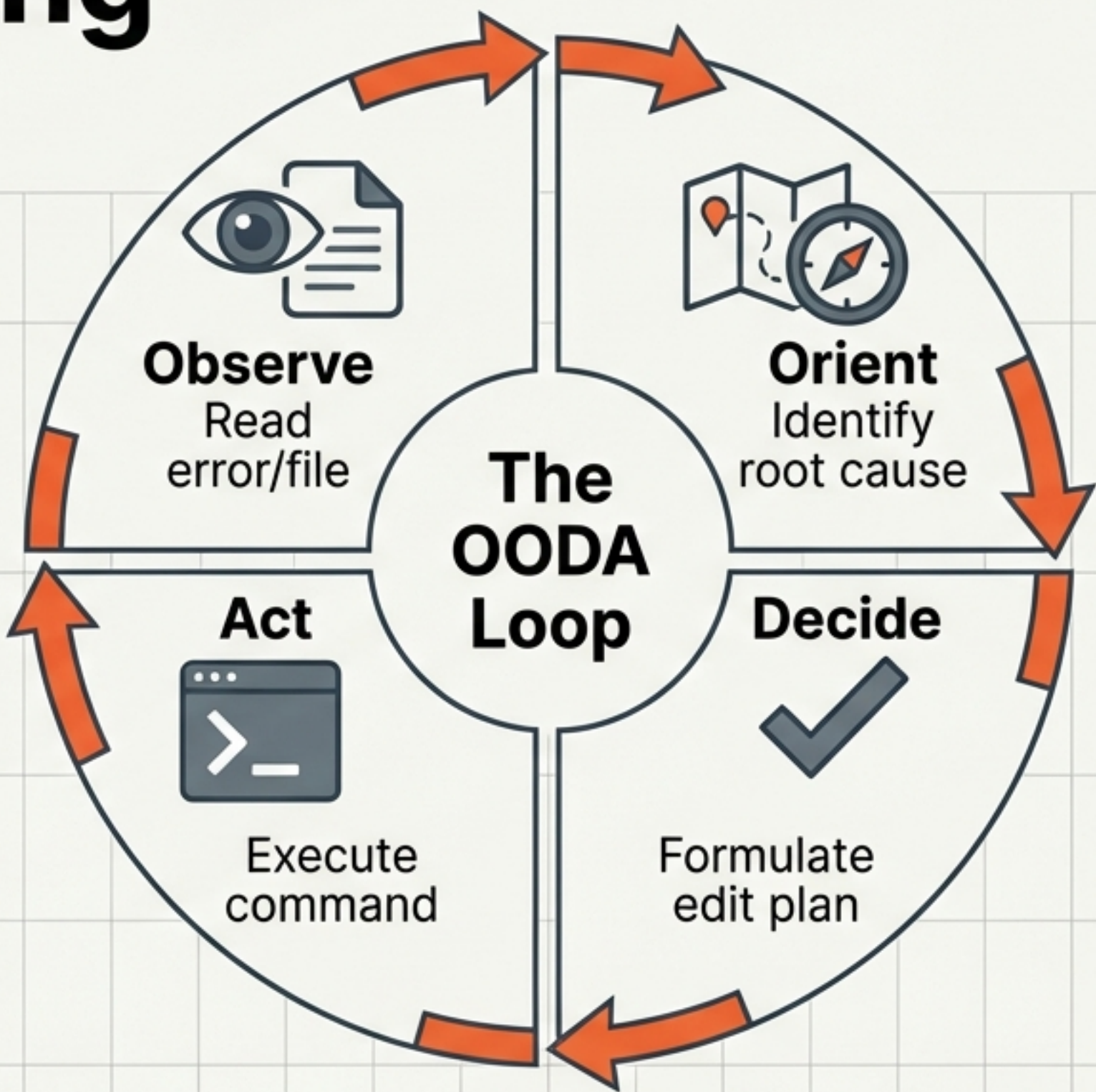
5 PRs/Day
Average engineer throughput (vs. typical 1-2).

AI didn't need to get smarter; it needed access. When Boris Cherny gave Claude filesystem access, it naturally began to explore, follow imports, and understand project structure without instruction.

Prediction vs. Reasoning
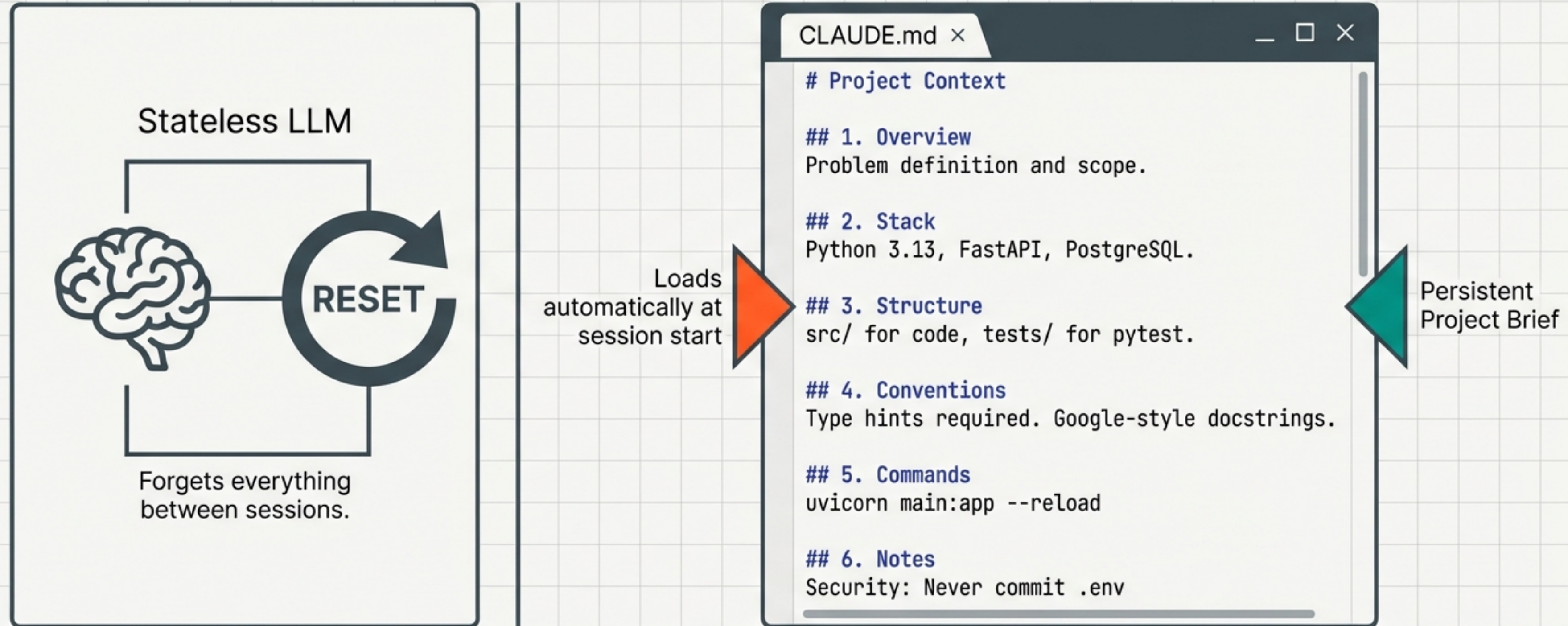
The OODA Loop Architecture

Passive AI (Chat)	Agentic AI (Claude Code)
• **Role:** Consultant on phone	• **Role:** Pair Programmer
• **Visibility:** Blind to screen	• **Visibility:** Sees filesystem
• **Action:** Predicts next word	• **Action:** Reasons & Acts
• **Flow:** Single-shot response	• **Flow:** Iterative Loop
• **Friction:** High (Copy-Paste)	• **Friction:** Low (Direct Edit)

Observe
Read error/file

Orient
Identify root cause

The OODA Loop

Act
Execute command

Decide
Formulate edit plan

"Friction Removal > Feature Addition"

Solving Statelessness with CLAUDE.md

The Filesystem as External Memory

Stateless LLM

RESET

Forgets everything between sessions.

Loads automatically at session start

CLAUDE.md ×

```
# Project Context

## 1. Overview
Problem definition and scope.

## 2. Stack
Python 3.13, FastAPI, PostgreSQL.

## 3. Structure
src/ for code, tests/ for pytest.

## 4. Conventions
Type hints required. Google-style docstrings.

## 5. Commands
uvicorn main:app --reload

## 6. Notes
Security: Never commit .env
```

Persistent Project Brief

Standardization: Use `AGENTS.md` for universal compatibility, `CLAUDE.md` for specialized features.

NotebookLM

Skills: Encoding Expertise

Constraining Non-Determinism with Procedures

```
root/
├── .claude/
│   └── skills/
│       ├── internal-comms/
│       │   └── SKILL.md
│       └── legal-review/
│           └── SKILL.md
```
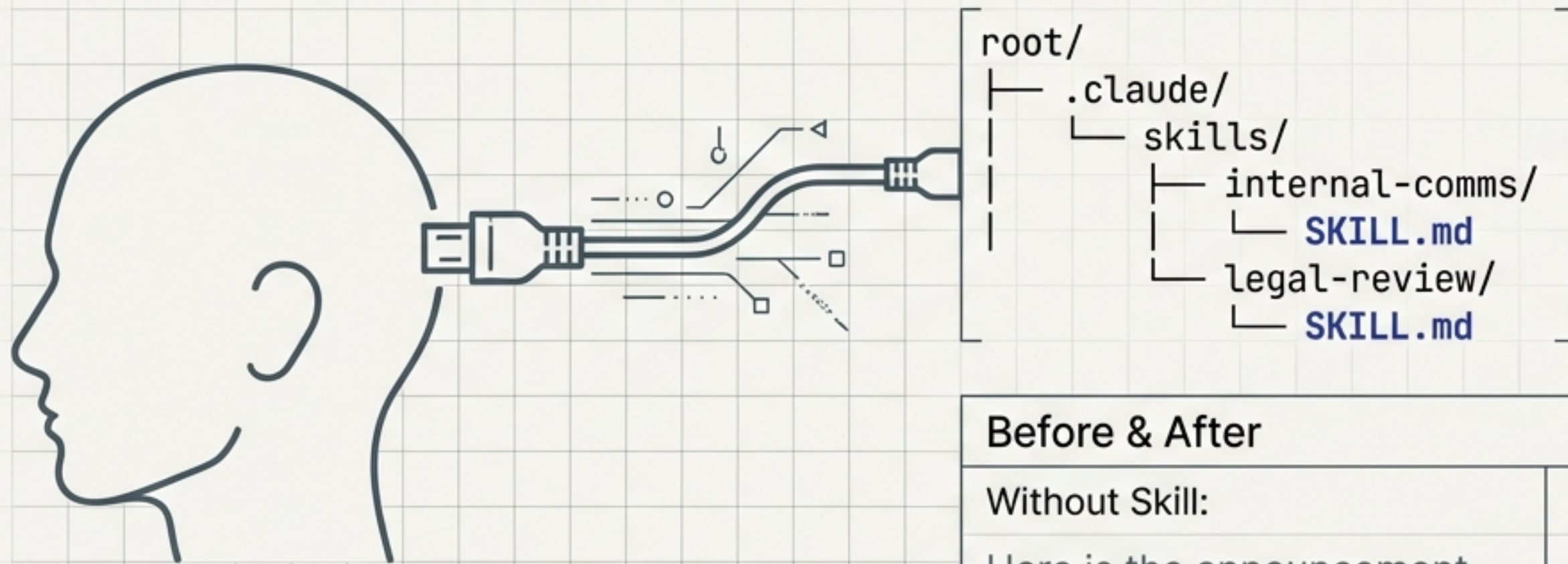
Before & After

Without Skill:	With `internal-comms` Skill:
Here is the announcement. Meeting at 5pm.	Hey team! 🚀 Quick reminder for our sync at 5pm. See you there? 👇

Mechanism: AI output is random (non-deterministic). Skills constrain this drift by enforcing your specific boundaries, tone, and procedures defined in **SKILL.md**.

Extending Reach with MCP

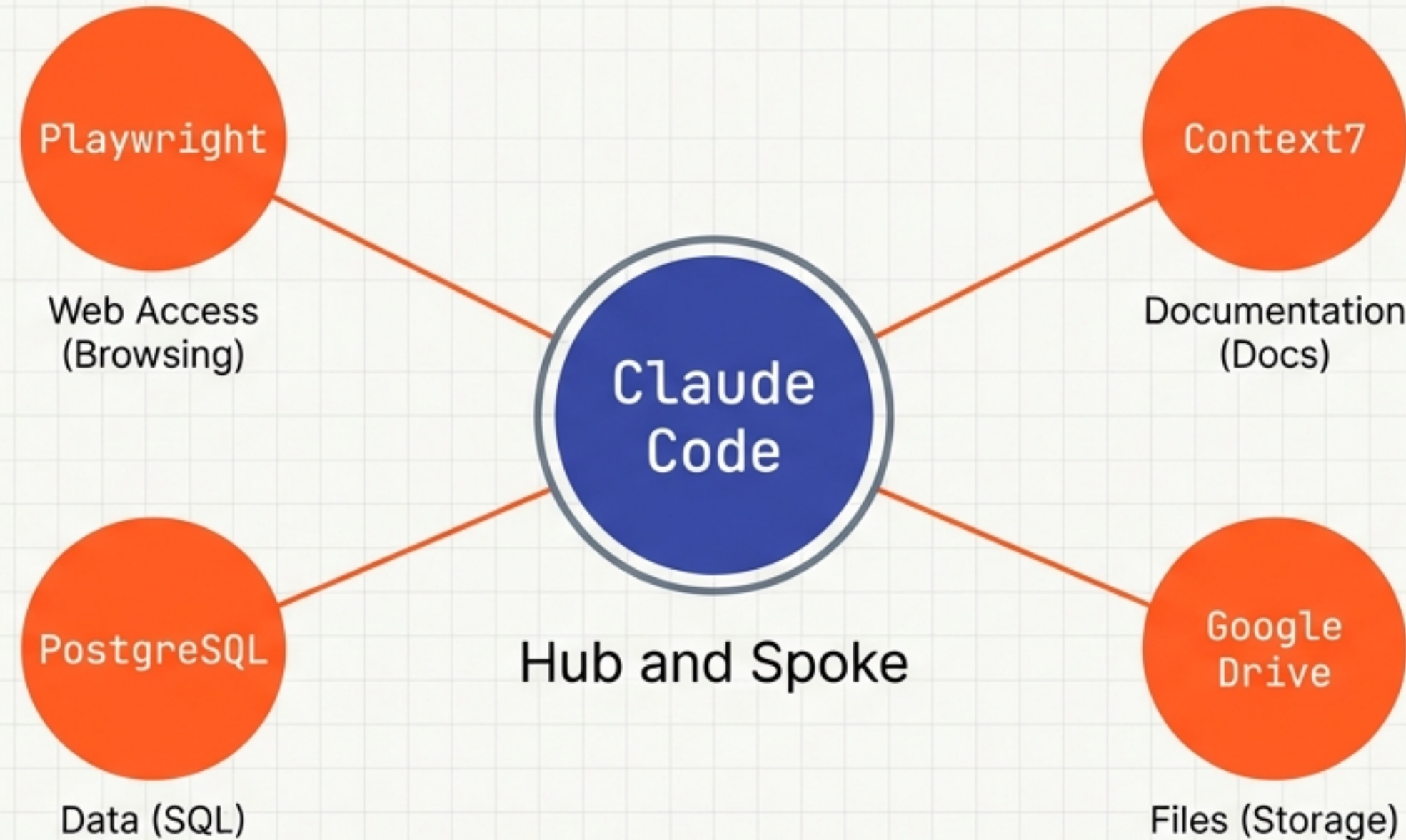Model Context Protocol: The "Phone Book" for Tools

Playwright

Web Access
(Browsing)

Context7

Documentation
(Docs)

**Claude
Code**

Hub and Spoke

Safety:

Access controlled
via allow-lists.
Secrets stored in
environment
variables, never
plain text.

PostgreSQL

Data (SQL)

**Google
Drive**

Files (Storage)

**Context
(CLAUDE.md)** + **Procedures
(Skills)** + **Access
(MCP)** = **Autonomous
Agent**

NotebookLM

Orchestrating a Team of Subagents

Context Isolation & Parallel Execution

Main Agent (Controller)
Delegates Tasks

Explore Agent (Haiku)
Inter Medium
Scans files, maps structure.

Plan Agent (Sonnet)
Inter Medium
Creates implementation strategy.

Builder Agent
Inter Medium
Executes code changes.
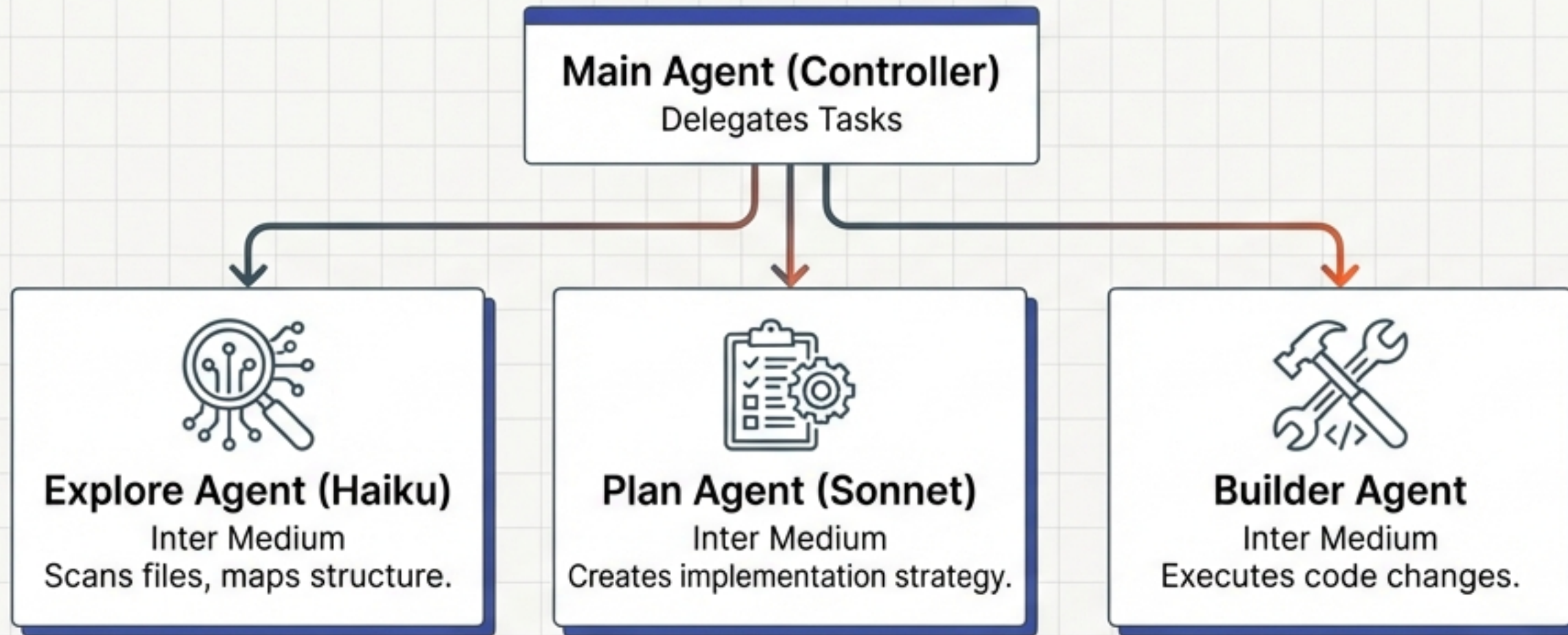
Parallel Orchestration

Explore

Plan

Builder
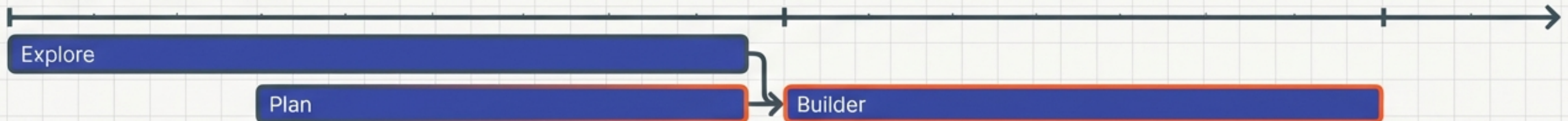
Don't ask one AI to do everything. Use specialized subagents with isolated context windows to prevent hallucination and clutter.

NotebookLM

Hooks: Event-Driven Automation
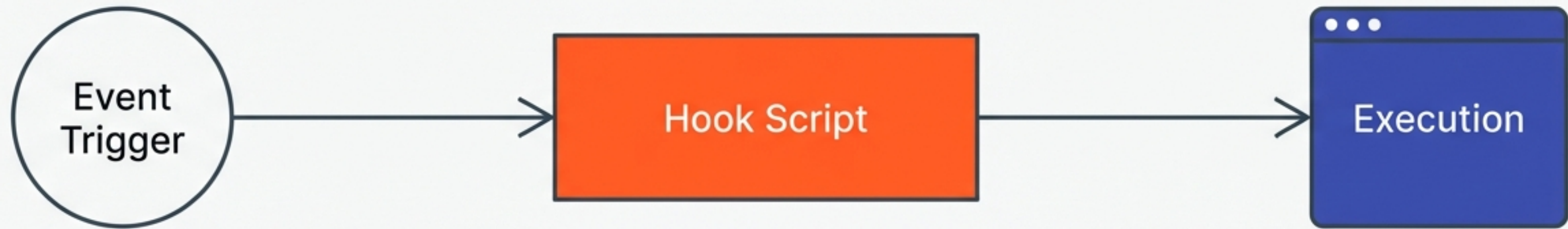
Programming the Agent's Environment

PreToolUse (Safety)
Trigger: Agent tries to edit `.env`
Action: Block operation & Warn user

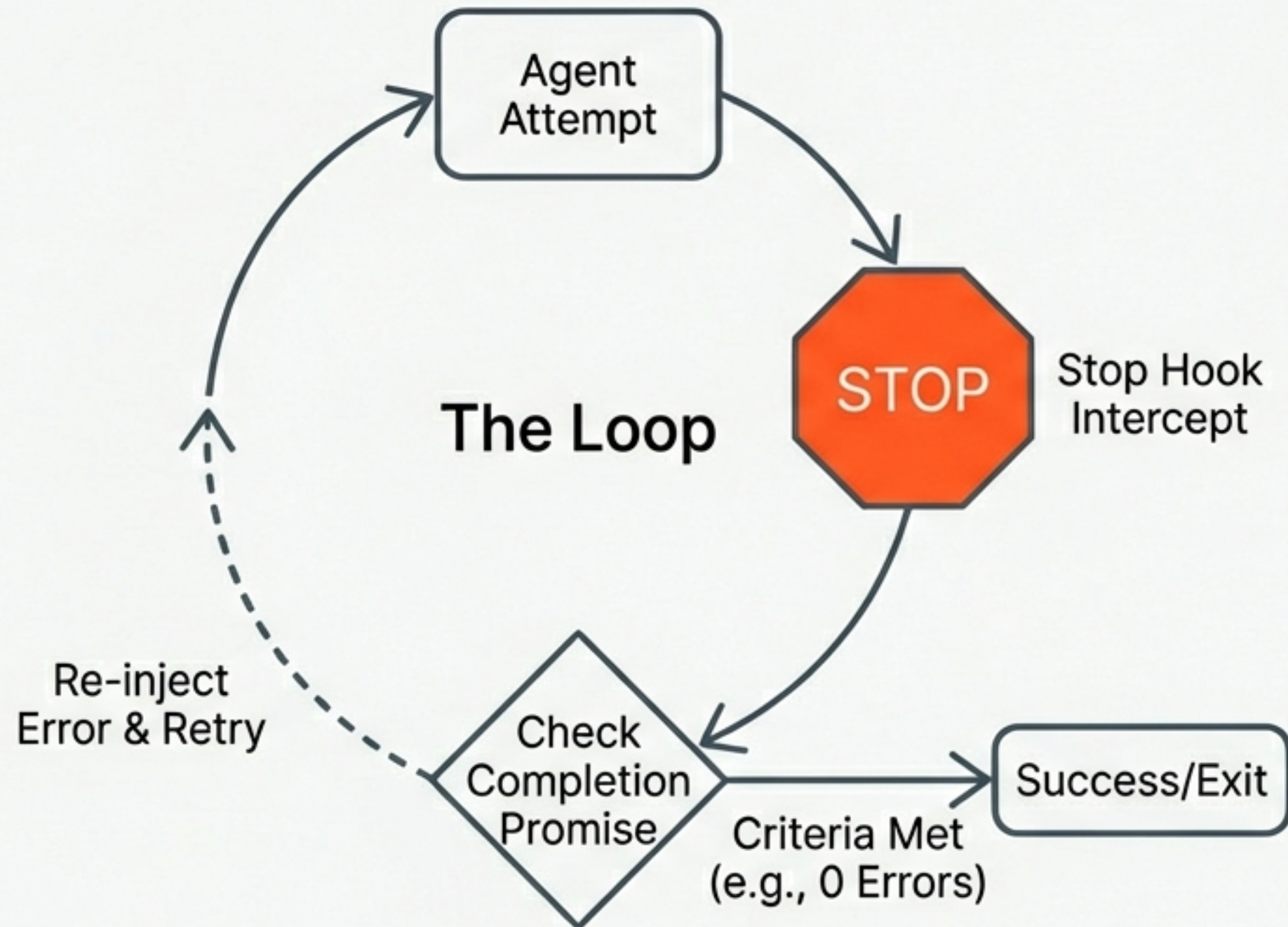| PostToolUse (Quality) |
| --- |
| Trigger: File saved |
| Action: Run `prettier` auto-format |

SessionStart (Context)
Trigger: Launching Claude
Action: Inject specific project logs

"Turn 'Please format this' into a hard system rule."

The Ralph Wiggum Loop
Self-Healing Workflows & Iteration

The Loop diagram:
- **Agent Attempt** → **STOP** (Stop Hook Intercept) → **Check Completion Promise** (diamond) → **Success/Exit** (Criteria Met, e.g., 0 Errors)
- Re-inject Error & Retry (dashed arrow back to Agent Attempt)
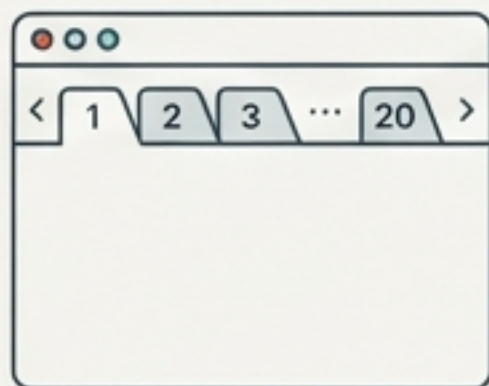
Solving Iteration Fatigue.
- **Manual:** Run -> Error -> Copy/Paste -> Fix.
- **Autonomous:** Agent tries, fails, learns, and retries until success.

Criteria for Use: Objective, Verifiable, Deterministic (e.g., Linting, Tests, Upgrades). JetBrains Mono
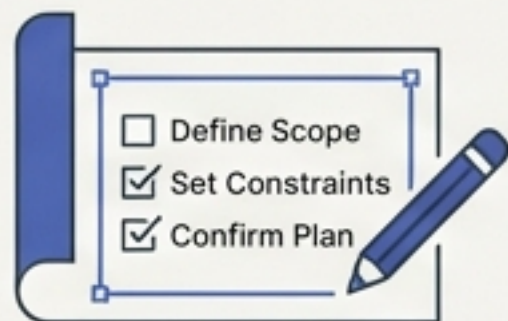
The Creator's Workflow

Orchestration in Practice (Boris Cherny)

Parallel Sessions

Treat Claude as capacity to schedule. Run 15-20 tabs simultaneously.

Plan Mode First

☐ Define Scope
☑ Set Constraints
☑ Confirm Plan

Align on the plan before execution. A wrong fast answer costs more than a right slow one.

Verification Loops

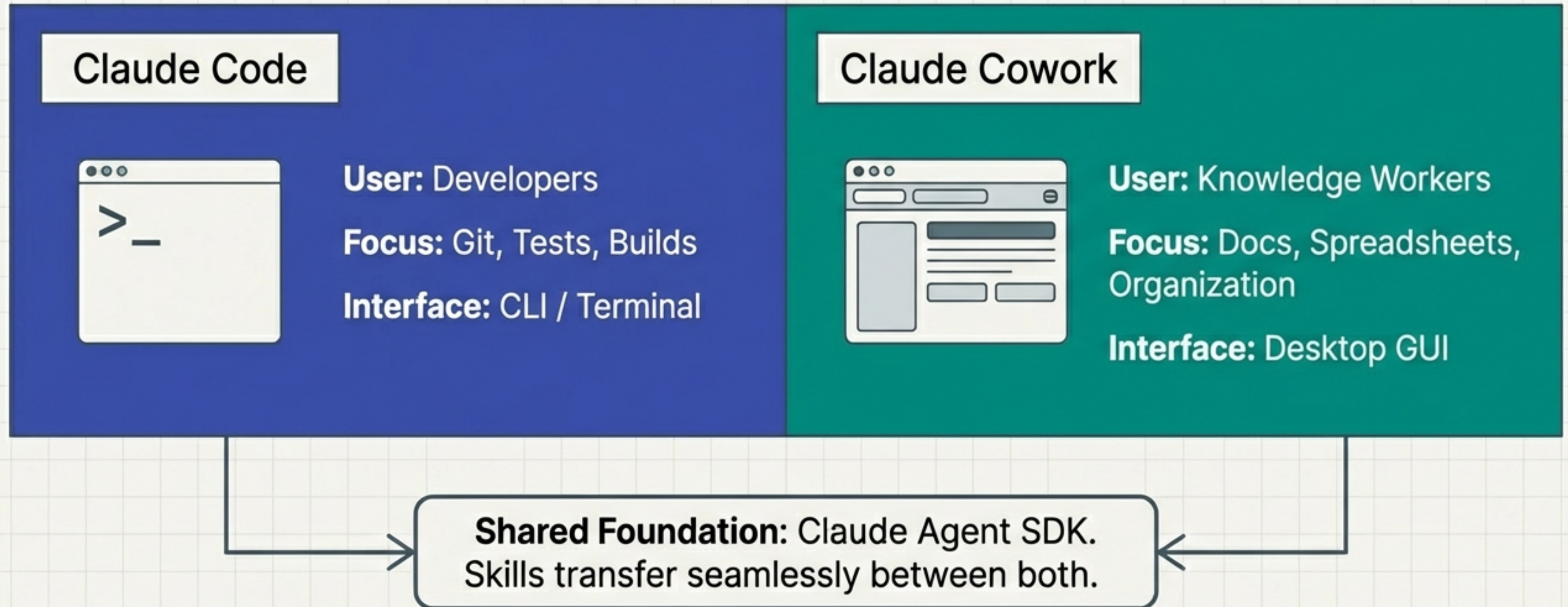Trust but Verify. Use hooks and browser tools to force self-checking.

Living Context

CLAUDE.md

Every mistake updates `CLAUDE.md`. Build institutional memory.

The Second Pivot: Claude Cowork

Agentic Power for Knowledge Workers

Claude Code

User: Developers

Focus: Git, Tests, Builds

Interface: CLI / Terminal

Claude Cowork

User: Knowledge Workers

Focus: Docs, Spreadsheets, Organization

Interface: Desktop GUI

Shared Foundation: Claude Agent SDK. Skills transfer seamlessly between both.

NotebookLM

Cowork in Action

From Chat to Execution

Organization

Prompt: "Clean up Downloads folder."

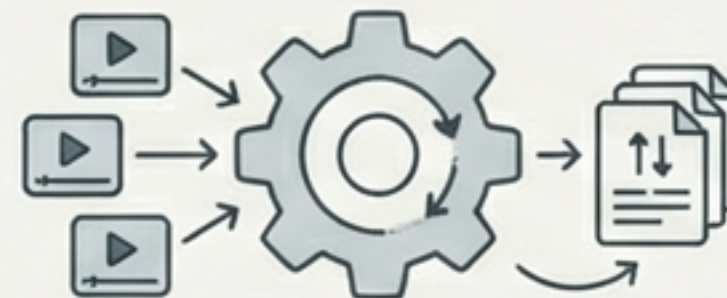**Action:** Organizes by file type, deletes old installers.

Synthesis

Prompt: "Read 20 PDFs and extract themes."

Action: Multi-document analysis and summary generation.

Batch Processing

Prompt: "Convert & compress 50 videos."

Action: Batch file operations.
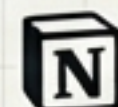
Connectors: MCP for Everyone.

Zero-code integrations for everyday tools.

Google Drive   slack   Notion   GitHub

NotebookLM

The Office Suite

Built-in Capabilities for Standard Formats

DOCX

- Tracked Changes
- Formatting Preservation
- Header/Footer Management

XLSX

FORMULA

- Formula Awareness
- Data Analysis
- Chart Generation

PPTX

- Slide Creation from Outline
- Theme Application
- Layout Management

PDF

- OCR Extraction
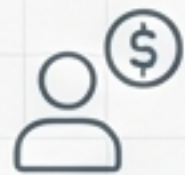- Structure Analysis
- Table Identification

Native manipulations, not text approximations.

The Business Model: Digital FTEs
Selling Outcomes, Not Time
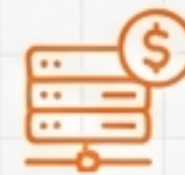
Economics Comparison

Human Employee			Digital FTE		
$5k/mo	40 hrs/week	Slow Ramp	$500/mo	168 hrs/week	Instant Scale

4 Revenue Models

Subscription	Success Fee	Licensing	Marketplace
Hosted Agent (e.g., Contract Reviewer).	Pay-per-result (e.g., $5/lead).	Sell the SKILL.md (Enterprise).	Volume sales (App Stores).

Concept: Moving from 'Tool User' to 'IP Creator'.

The Agent Factory

The Future of Work

$$\frac{\text{Context (CLAUDE.md)} + \text{Procedures (Skills)} + \text{Access (MCP)}}{} = \textbf{Digital FTE}$$

Manual Operator → Assisted User → Agentic Orchestrator → Agent Factory Owner

Stop building agents from scratch. Build Skills instead.