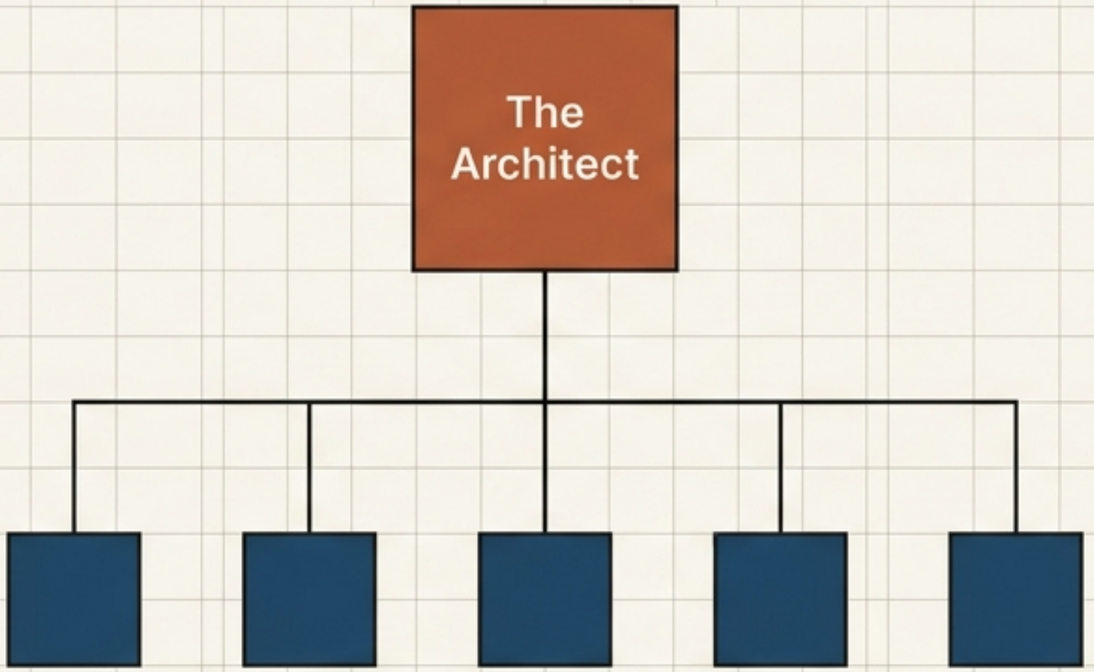
The Super Individual

Why the AI economy belongs to systematic thinkers and architects.



A blueprint for shifting from execution to orchestration.

46%

The percentage of workforce cut by Block.

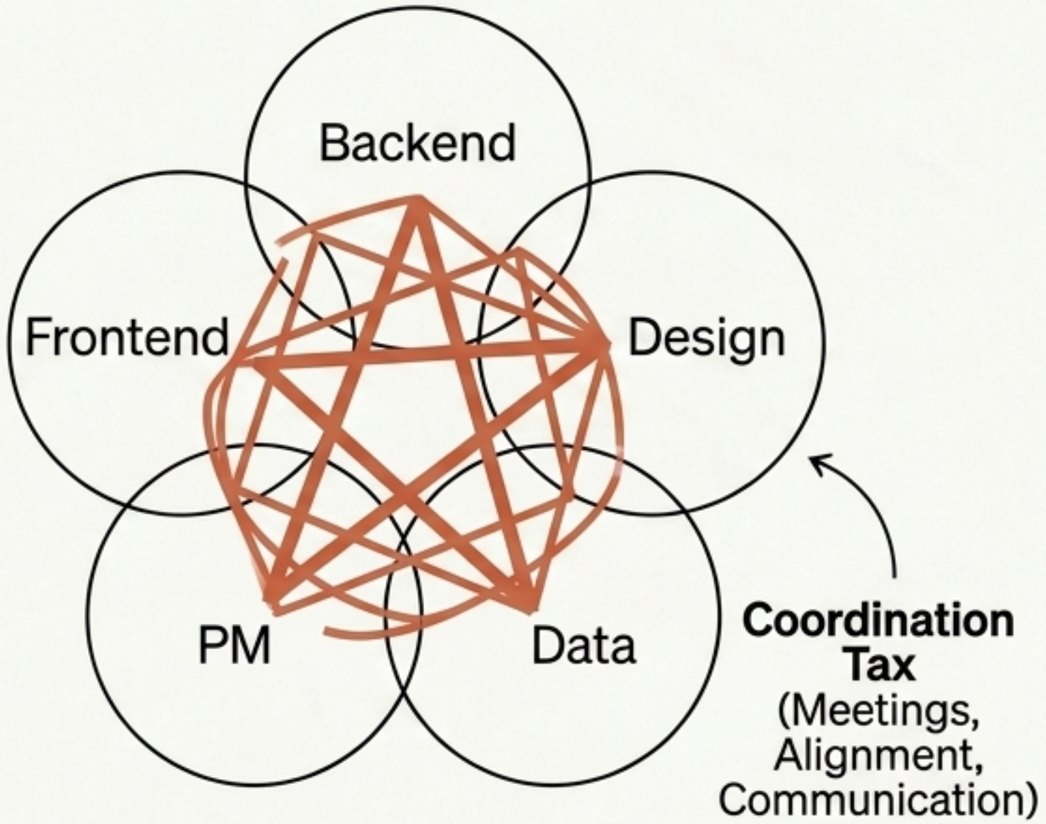
> The savings are going into AI. The headcount isn't coming back.
>
> – Jack Dorsey

AI replaces headcount. Not hypothetically. Now. The question is no longer "Will AI take my job?" but "Who thrives in this new architecture?"

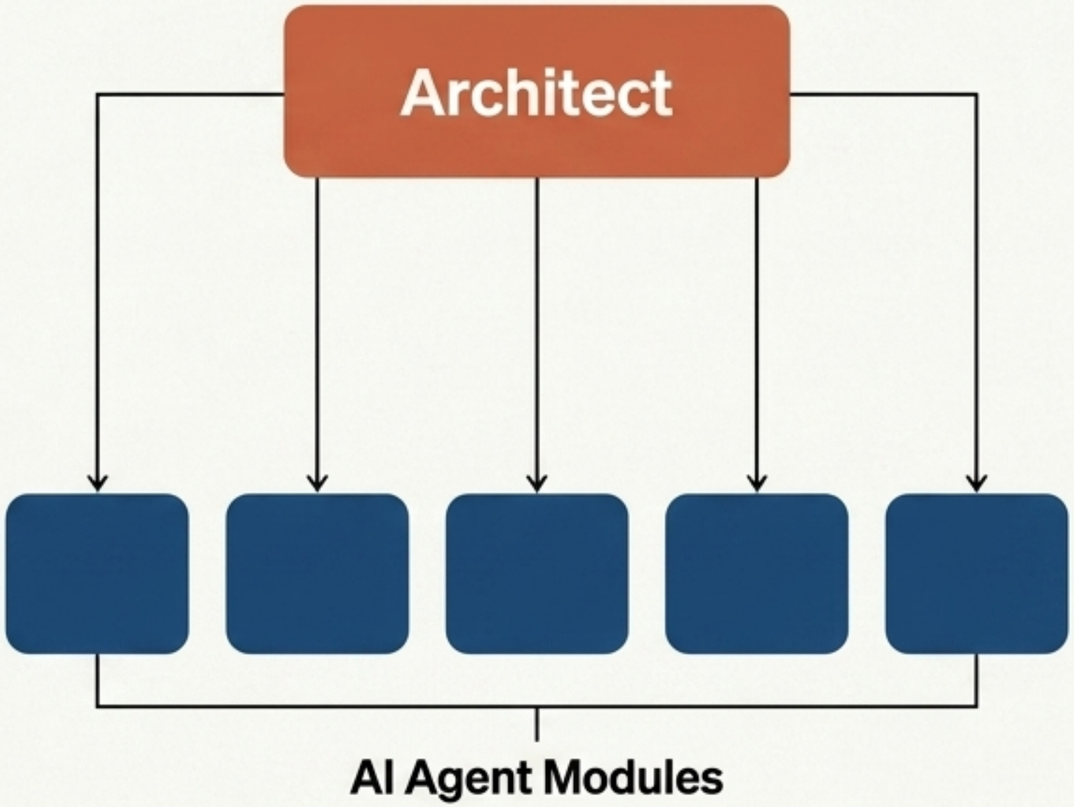
The Team Structure Evolution

Over 30% of founders are now solo. The constraint of the 5-person MVP has evaporated.

The Legacy MVP



The Super Individual MVP

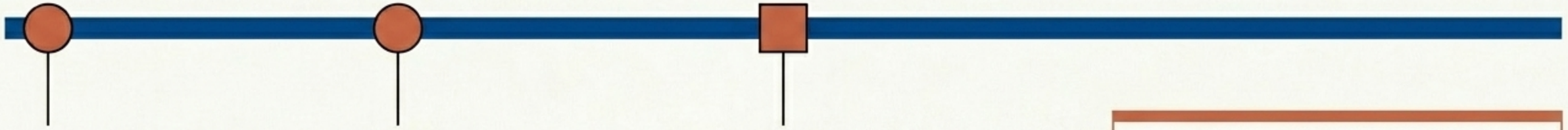


The Velocity Timeline

Synthesizing speed: The timeline of an orchestrator.

Legacy Development

Super Individual Velocity



Mio v1 — Empty repo to working voice companion in 4 days.

Mio v2 — Complete product pivot engineered by one person.

PanPanMao — 10 shipped apps built solo with Claude Code in 29 days.

One-person companies will compete directly with teams of 10-20. And they will win on speed.

The Bottleneck Shift

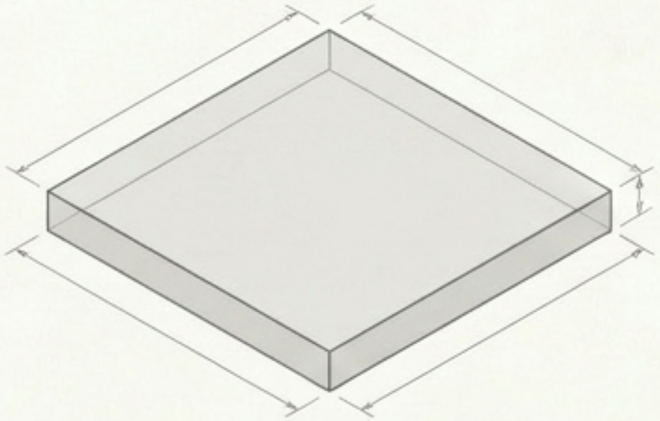
	The Old Era	The Centaur Era
Scarce Resource	Capital & Headcount	Human Taste & Judgment
Core Bottleneck	Execution Power	Problem Definition
Primary Meta-Skill	Specialization	Problem Decomposition
Definition of “Done”	Code Compiles	Edge Cases Solved

Execution is now infinite and cheap. The bottleneck is knowing what to execute.

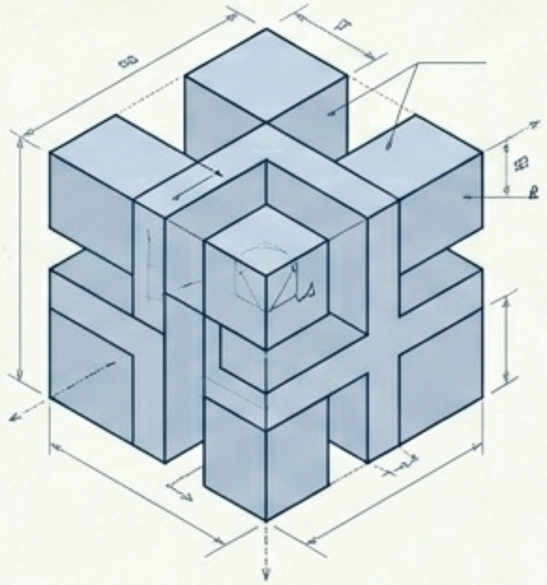
The Adoption Spectrum

Models adapt to you. Instruction syntax is a depreciating asset.

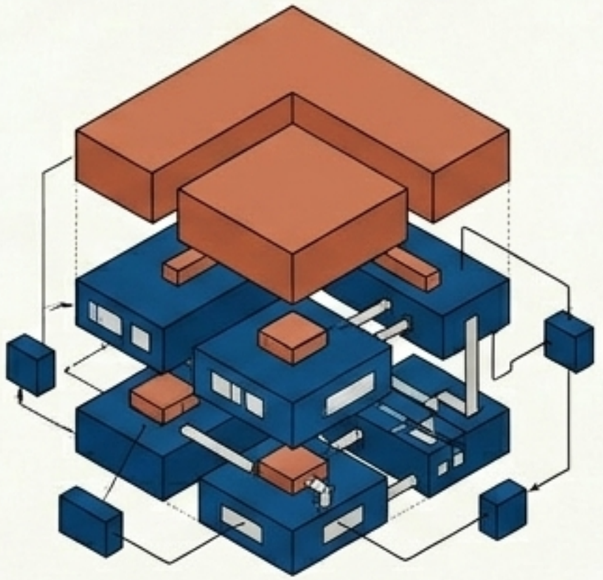
The AI Commentator



The Prompt Engineer



The System Architect



The AI Commentator

Can deliver a 45-minute talk on the AI revolution. Has never built anything with it.

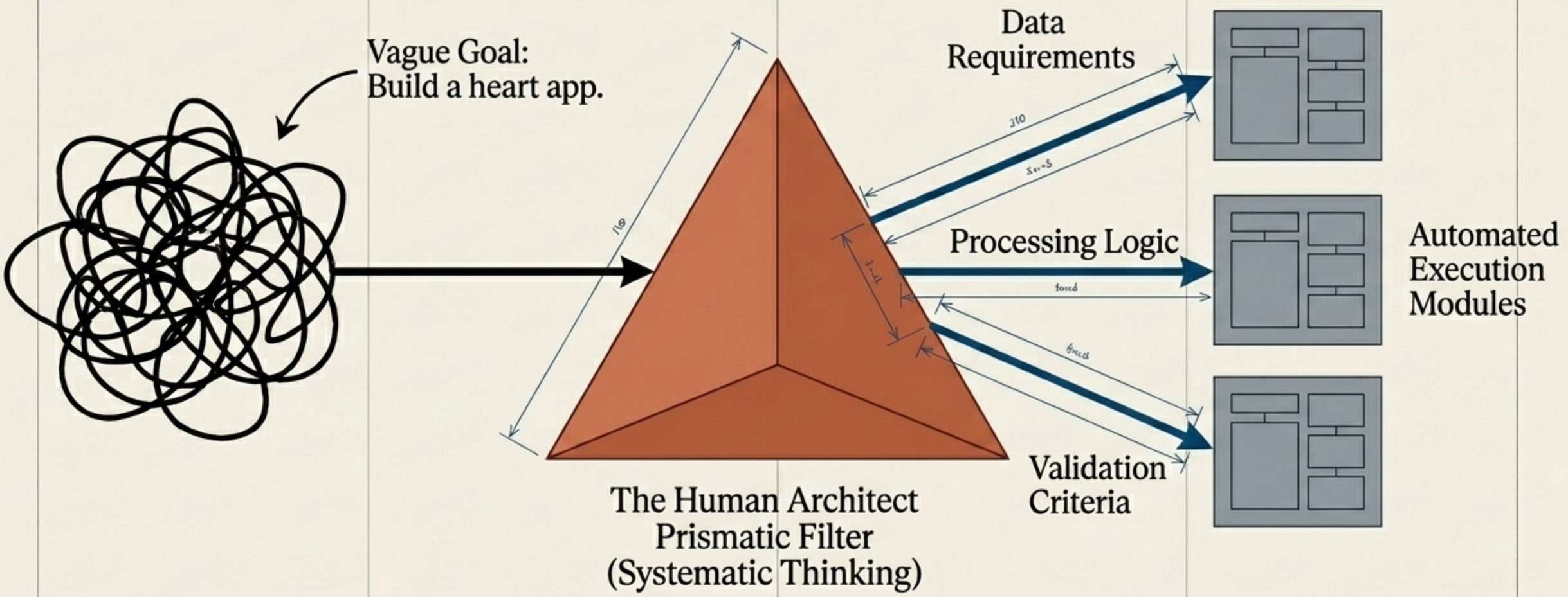
The Prompt Engineer

Focuses on crafting perfect, rigid instructions. Blind spot: Over-optimizes for syntax rather than problem structure.

The System Architect

Employs first principles. Defines critical paths. Orchestrates agents. Knows the game is problem definition.

The Decomposition Engine



The AI handles the subproblems. The human defines what the subproblems are.

What changed is the execution layer. What remains is the systematic thinking.

Writing Unit Tests (Execution)

```
assert(x == y);
expect(output).to.equal(true);
assert(process.status === 'OK');
expect(result.data.length).to.be(10);
assert(x > 0);
expect(output).not.to.be(null);
assert(x == y);
expect(output).to.equal(true);
assert(process.status === 'OK');
expect(result.data.length).to.be(10);
assert(x > 0);
expect(output).not.to.be(null);
assert(x == y);
expect(output).to.equal(true);
assert(process.status === 'OK');
expect(result.data.length).to.be(10);
assert(x > 0);
expect(output).not.to.be(null);
```

Defining System Behavior (Orchestration)

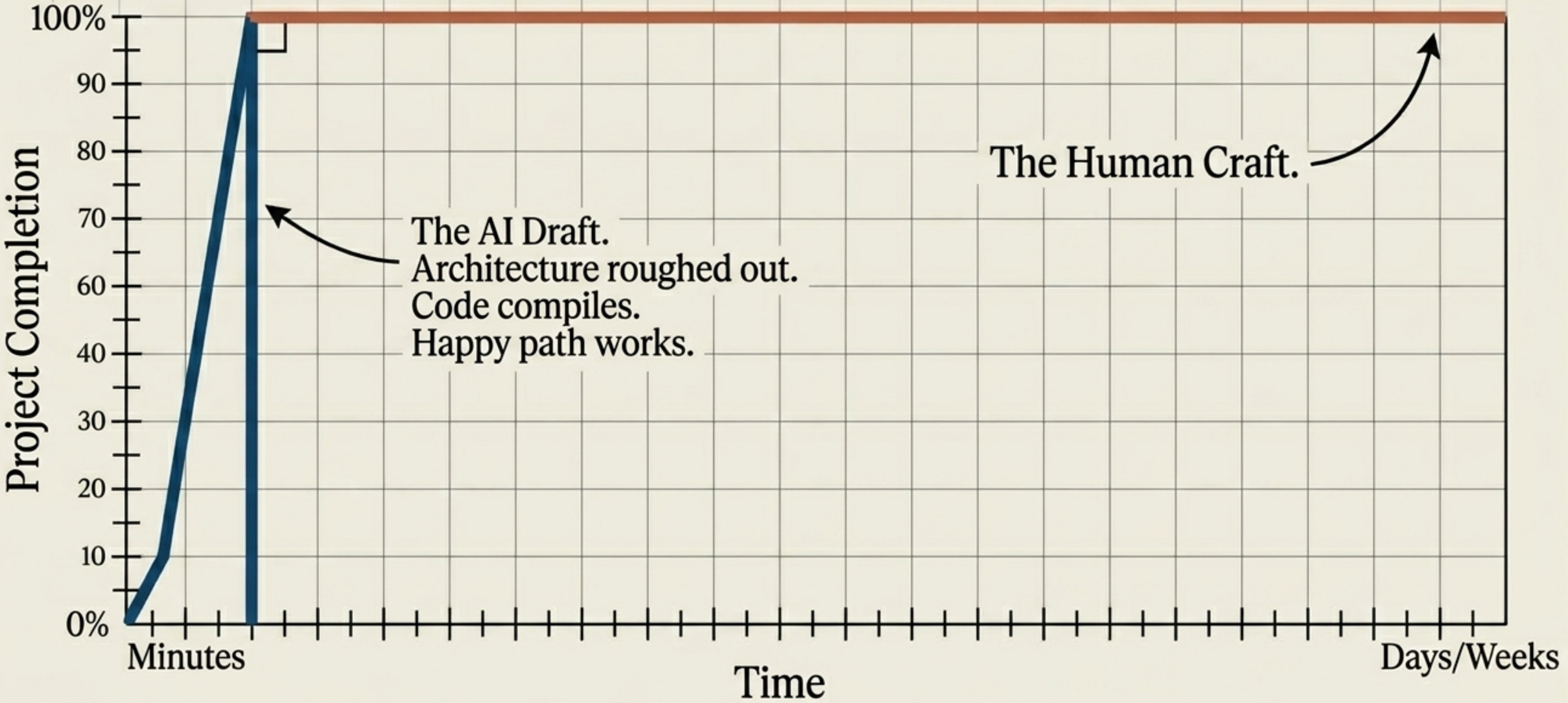
- Inputs
- Expected Outputs
- Invariants
- Failure Modes

↓

Instantly Generated Test Block

The 99/1 Paradox

Why orchestration feels simultaneously magical and agonizing.



Surviving the Flatline

"AI completes 99% of the work in 1% of the time, but you spend 99% of your time solving that last 1% of detail." — David Shell

The Edge Case

The specific, unanticipated input parameter that causes a cascading system crash.

The UX Flow

The interface technically functions, but feels inherently overwhelming to a new human user.

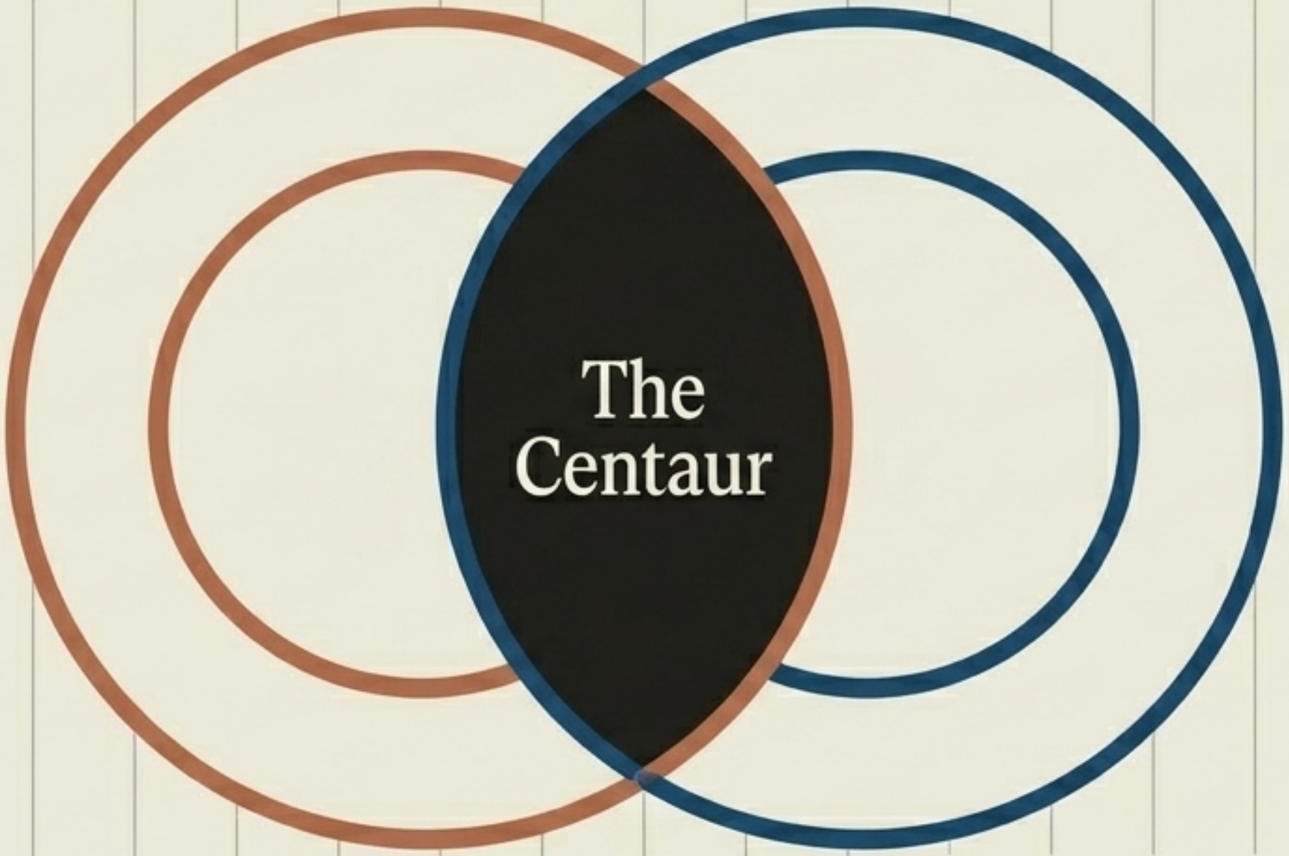
The Deployment

Code operates perfectly in a local environment but fails in production due to an obscure variable.

The Tone

Generated copy that is grammatically flawless but feels subtly inauthentic to the brand voice.

The Centaur Era



Dario Amodei defines the most effective unit not as a human or an AI, but a human fused with AI. Half human judgment, half machine execution.

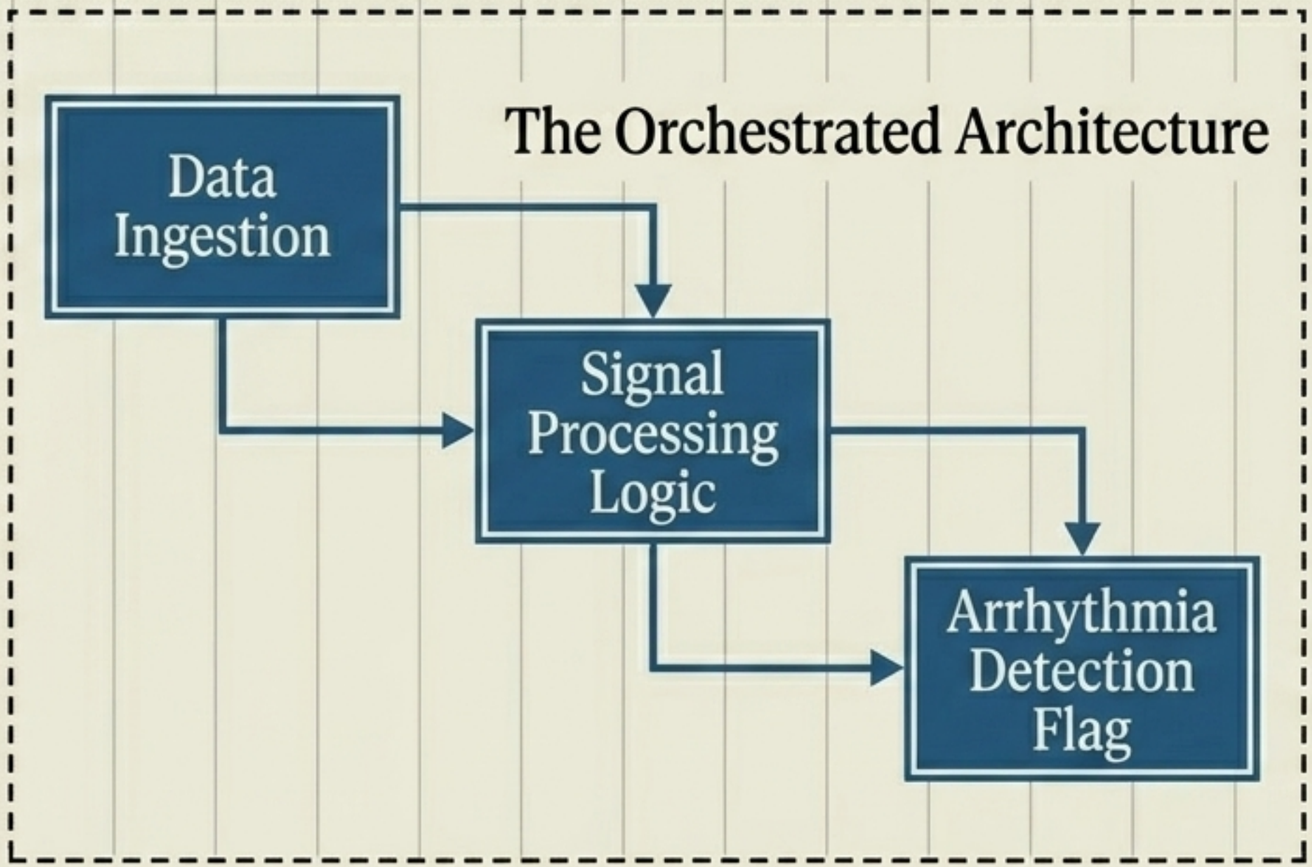
The Networking Centaur

Analyzing 1,000 conference profiles.
Manual time: 10+ hours. Centaur time: 2 minutes.
The system doesn't replace networking judgment; it extends human taste across an impossible dataset.

Domain Orchestration > Domain Expertise

The Orchestrator

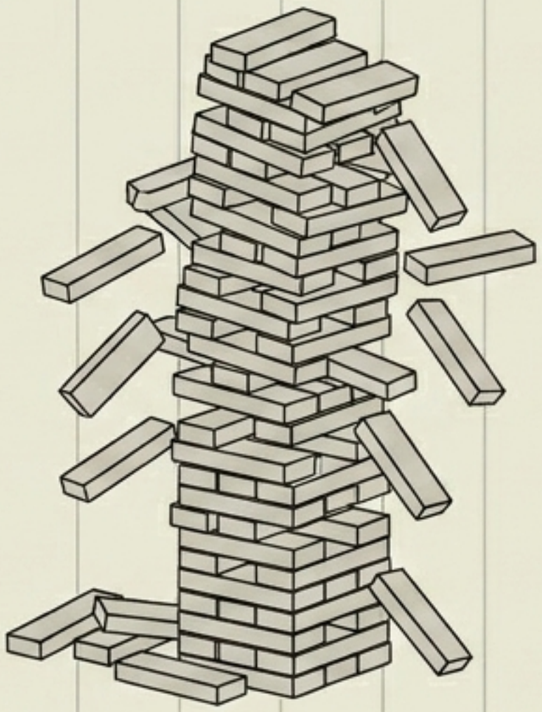
Critical Attributes	Irrelevant Background
☑ 18 years old	~~CS Student~~
☑ Domain curiosity	~~Medical Device~~ Engineer
☑ Systematic thinking	



They didn’t ask a chatbot to “make a heart app.”
They orchestrated the subproblems.

The Volume Trap vs. The Precision Target

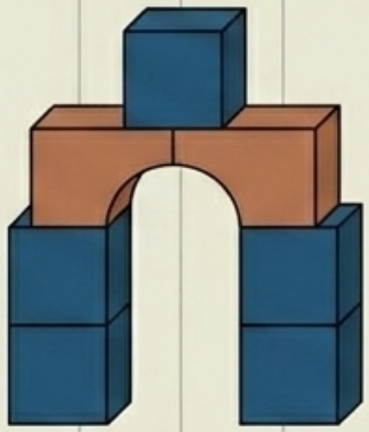
The Trap



Token Consumption

5,000 lines of generated slop that solves the wrong problem quickly.

The Target

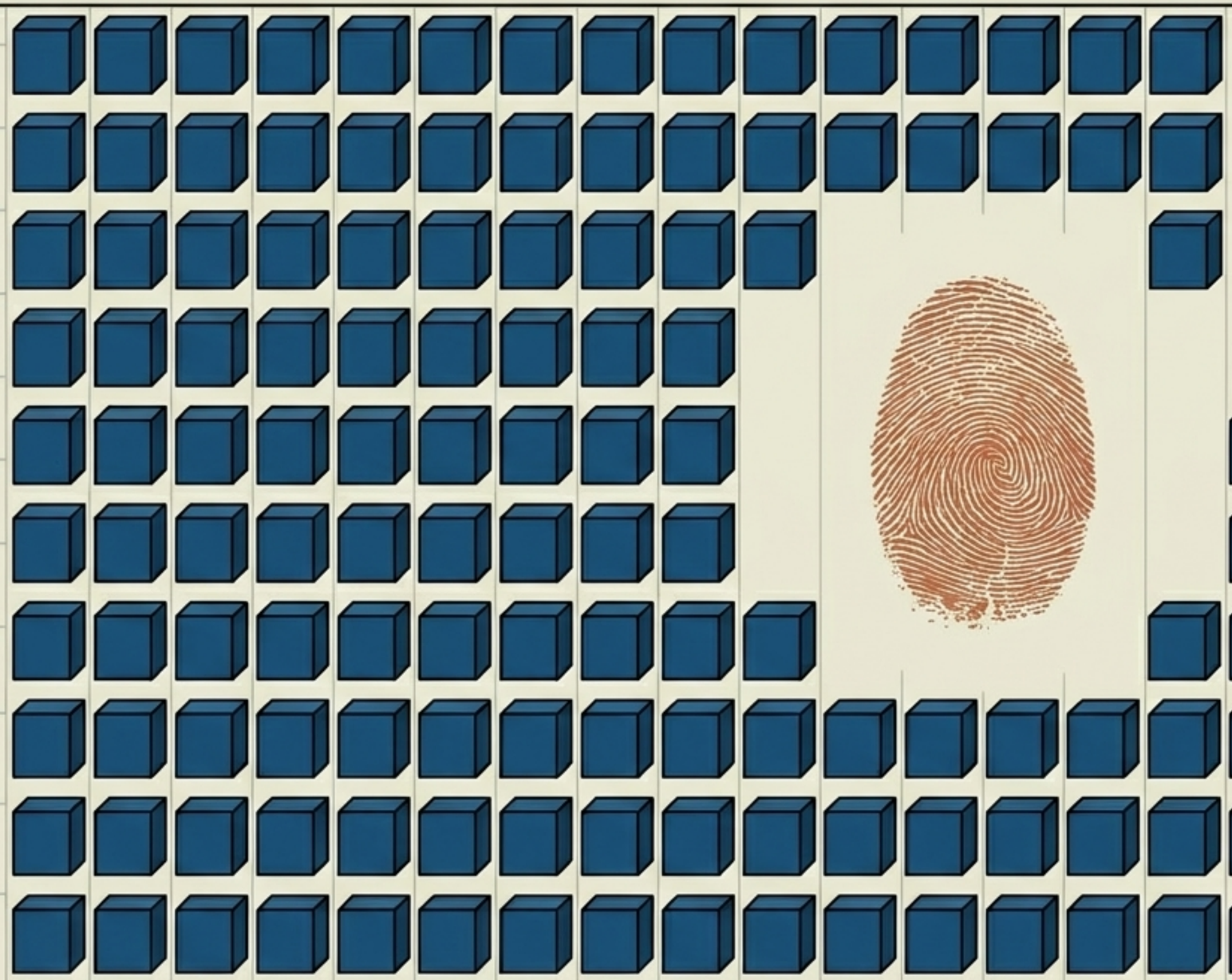


Problem Architecture

50 lines of carefully architected code that solves the exact right problem.

Measuring developer productivity by token consumption completely misses the point. The differentiator is the quality of direction.

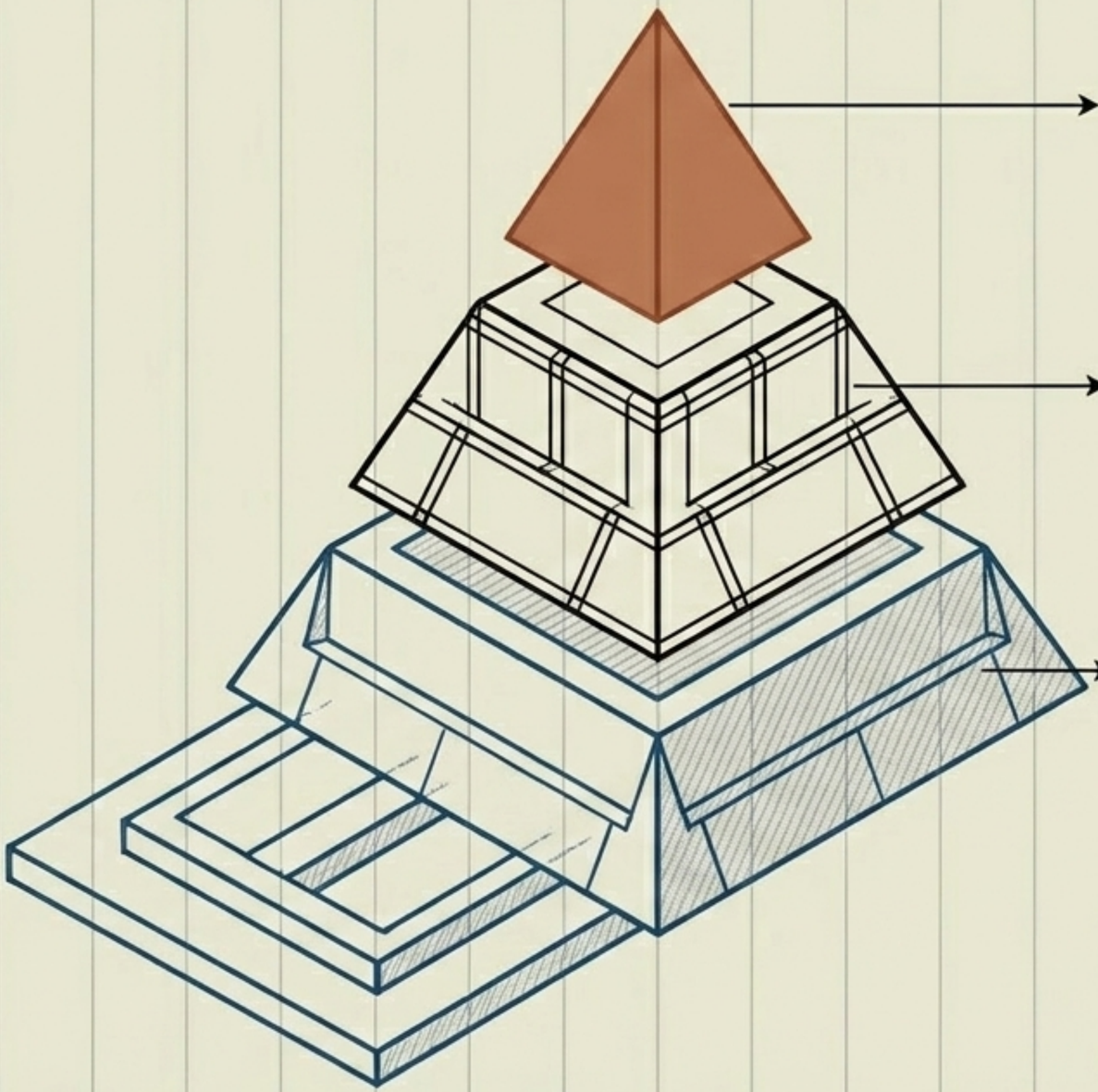
The Last Moat: Aesthetic Judgment



When AI can generate unlimited everything, the scarce resource shifts from production to curation.

- Someone must choose the right friction.
- Someone must evaluate the subtle emphasis.
- Judgment cannot be reduced to a training objective.

The Super Individual Stack



Aesthetic Judgment

Taste, curation, empathy, direction.
The irreplaceable human Moat.

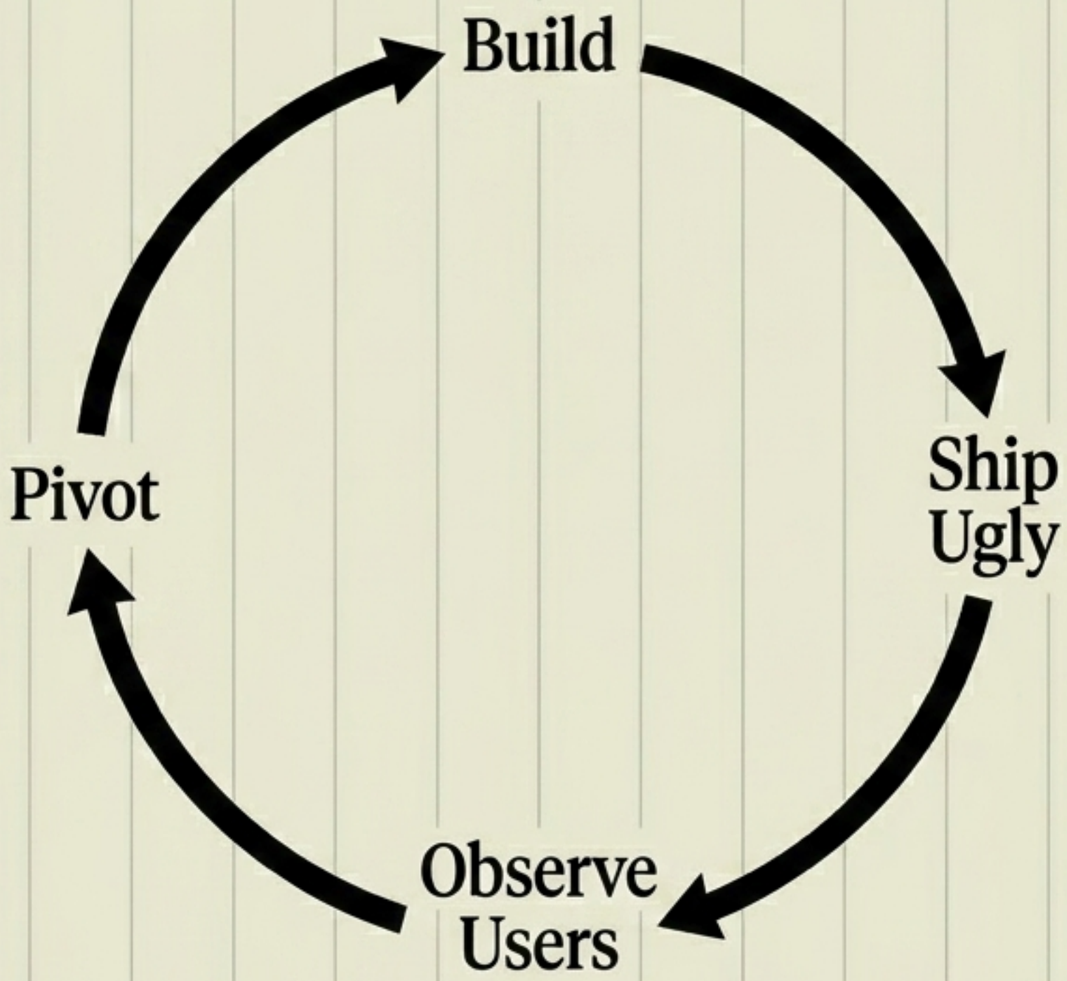
Problem Decomposition

First principles, defining critical paths, orchestrating agents.
The Engine.

AI Execution

Generating code, processing data, drafting copy.
Infinite, cheap, fast.
The Commodity.

Playbook I: Ship Ugly



Case Study: Mio

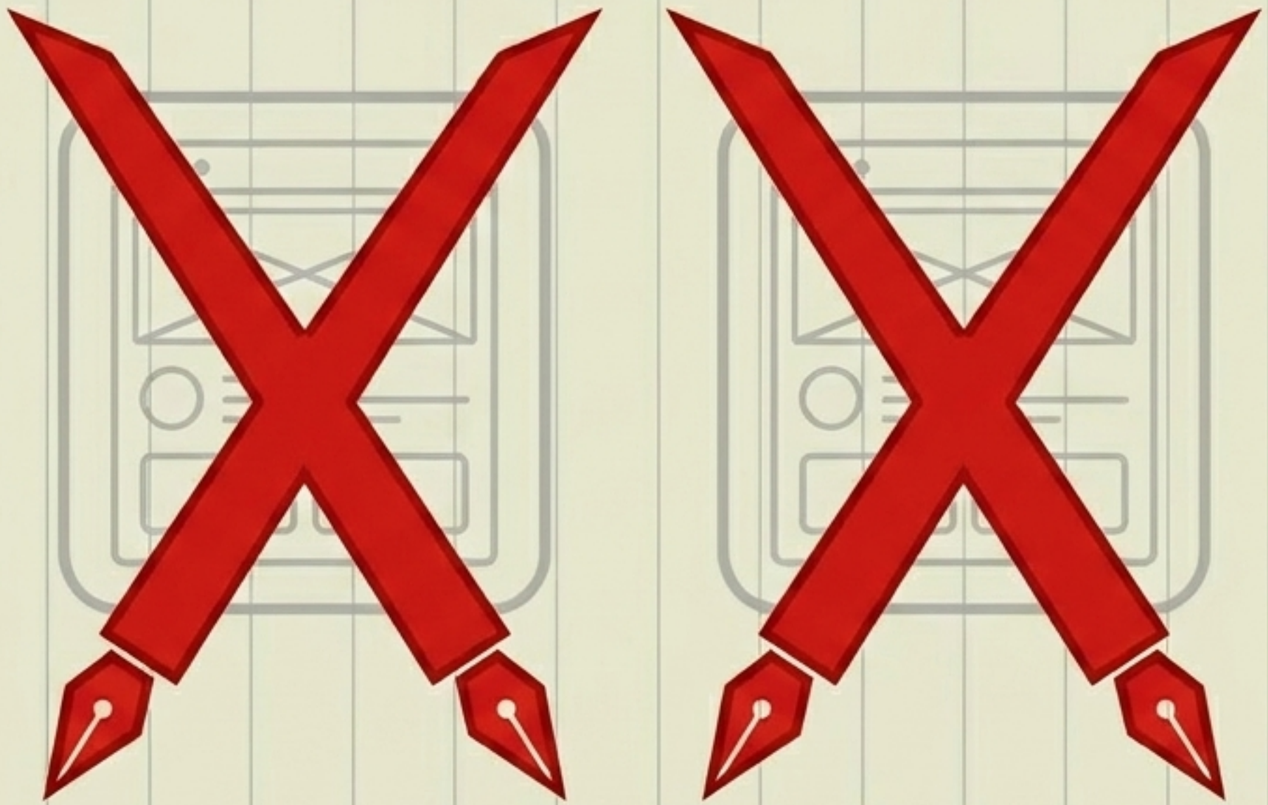
Mio v1 was rough and disposable. But watching real users interact with it revealed a human pattern the AI couldn't see.

This observation led to the complete pivot of v2.

The first version is a research instrument, not a product.

Playbook II: Kill Your Darlings Fast

AI lowers the cost of execution to near-zero.
You should be building three versions of an idea and throwing away two.
Avoid the temptation to polish what isn't working.



The PanPanMao Sprint: Apps without immediate traction were abandoned within days.

Playbook III: Decompose, Don’t Prompt

[WHAT_DOES_DONE_LOOK_LIKE.md]

1. Core Objective: _____

_____

2. Critical Path Constraints: _____

_____

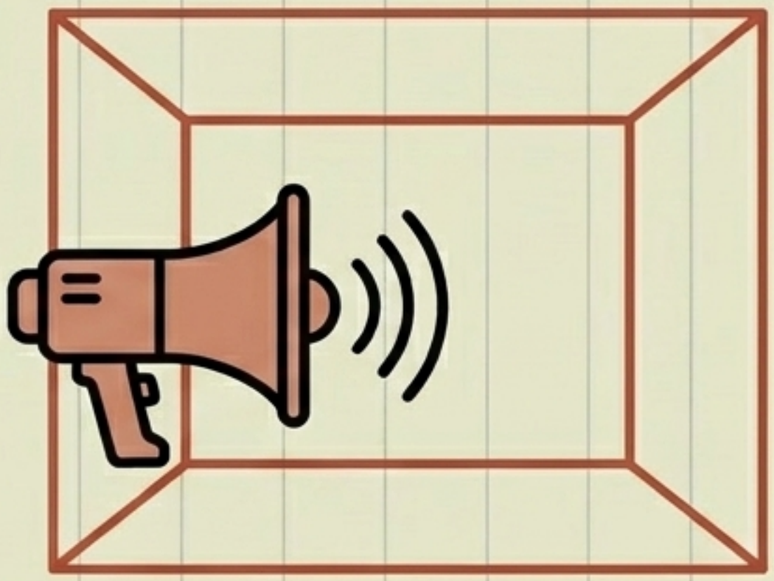
3. Success Validation: _____

The meta-skill that compounds is breaking problems down before writing a single prompt. Keep a running doc defining ‘done’. Skipping this fundamentally guarantees wasted days going in circles.

The Anti-Playbook: Traps to Avoid

Benchmark Shopping

Spending weeks agonizing over whether a model is 3% better on a leaderboard. Compute is massively subsidized right now. The math is temporarily absurd. Just build.



The Commentator Trap

Understanding AI conceptually is vastly different from orchestrating it. You only understand parallel agent tasks by forcing them to actually build something.

Systematic thinking and taste are not downloadable.

They are forged by making things and watching them break.

The tools are ready. The differentiator is you.