



# Building Enterprise AI Agents

*Five things I would tell myself in 2022*

Anbang Xu · Director of AI · NVIDIA

Enterprise AI & Automation · September 2026

# 45 submissions told me something uncomfortable

*“Almost nobody submitted work on making an agent work. Nearly everyone submitted work on proving it works, and keeping it working.”*

— reviewing the KDD 2026 workshop on enterprise AI agents

That is not a complaint. It is a status report. Getting an agent to do the task stopped being the hard part around 2024. Everything expensive now sits downstream: evaluating behaviour you cannot fully specify, governing actions you cannot fully predict, and staying correct while the models underneath you change every six weeks.

17

production agents

30K+

employees served

12

publications

10

GTC posters and talks

*Everything that follows comes from systems that shipped, papers that went through review, and failures that got written down.*

# Four eras, and the five things they taught me

FOUR ERAS · each one ended when a different bottleneck appeared

|                                                                                                         |                                                                                     |                                                                                                          |                                                                                                            |
|---------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------|
| <p>2022–23</p> <p><b>Chatbots</b></p> <p><i>Retrieval quality</i><br/>→ <i>pipeline engineering</i></p> | <p>2024</p> <p><b>RAG pipelines</b></p> <p><i>Answering</i><br/>→ <i>acting</i></p> | <p>2025</p> <p><b>Agents + flywheels</b></p> <p><i>Per-agent effort</i><br/>→ <i>shared platform</i></p> | <p>2026</p> <p><b>Skills + workers</b></p> <p><i>Owning the harness</i><br/>→ <i>owning governance</i></p> |
|---------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------|

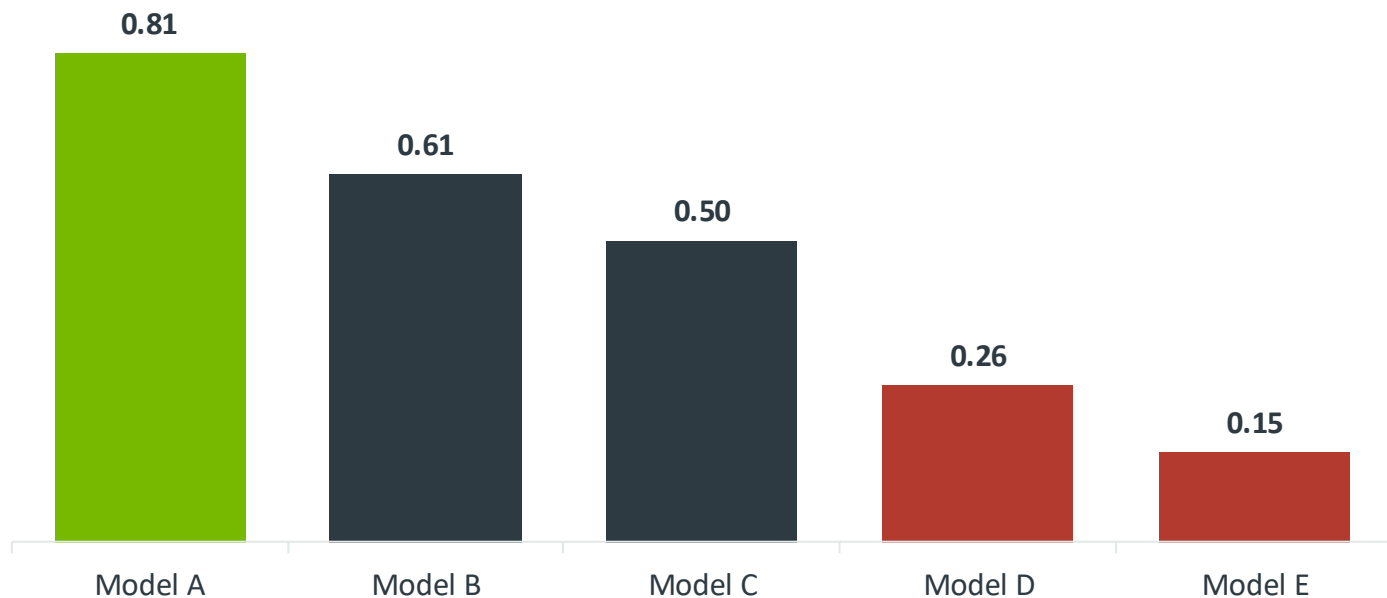
FIVE THINGS I WOULD TELL MYSELF IN 2022

|   |                                                |                                                                                 |
|---|------------------------------------------------|---------------------------------------------------------------------------------|
| 1 | <b>The bottleneck was never the model.</b>     | Fifteen control points · enterprise data · governance code · prompt portability |
| 2 | <b>Agentic is an architecture, not a goal.</b> | Three deletions that improved both axes · the four questions we ask at intake   |
| 3 | <b>Deployed is not done.</b>                   | Where the failures actually are · the MAPE loop · 10× smaller at equal accuracy |
| 4 | <b>Own the control plane, not the harness.</b> | The five-layer cake · skills as the unit · memory as a trust surface            |
| 5 | <b>Evaluation is the frontier.</b>             | Feedback is scarce · your judge is wrong · you need a world, not a prompt set   |

Every era began by discovering that the previous era's hard problem was no longer the hard problem. The rest of the talk is these five, in order, with the evidence.

# Swap the model, change nothing else, watch quality collapse

*One text-to-SQL system. One prompt set, tuned for the first model. Five models. Execution accuracy.*



*Model A is the one the prompts were tuned on.*

**0.81 → 0.15**

Same task. Same data. Same prompts. Only the model changed.

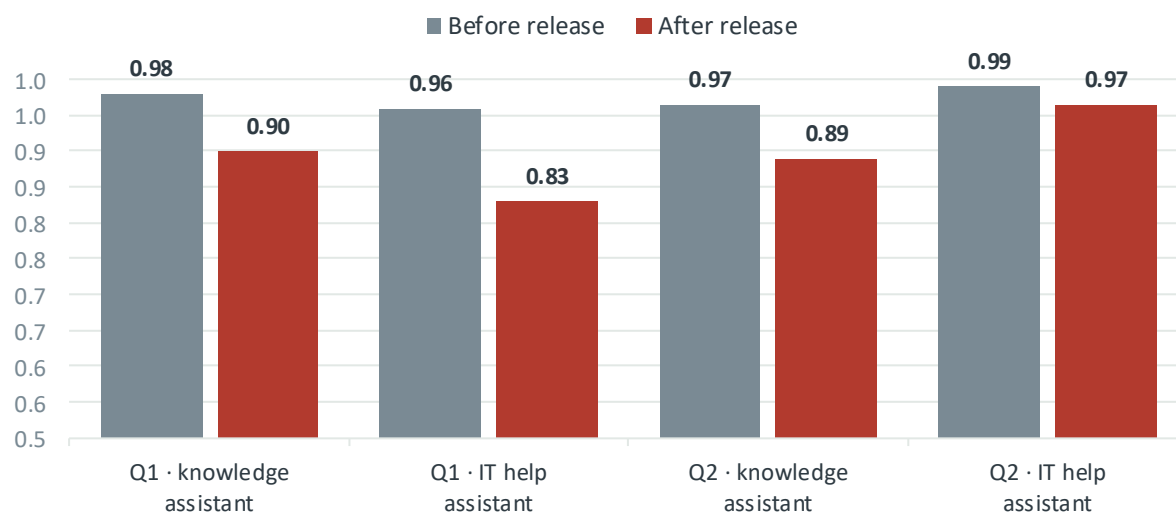
Prompts are model-specific assets with a shelf life measured in months, and new models arrive weekly.

If your quality lives in prompts, your quality is not portable. Push it into skills, tools, deterministic post-processing, and evaluation — the things that survive a model swap.

# The same weights are not the same system

*Same query, same prompt, same weights. The only thing that changed was the serving configuration.*

Answer consistency for an identical query, before and after a release



*Cosine similarity between answers to the same question. We deployed; consistency fell.*

## Same weights, different system

Two endpoints, the same 70B open-weights model, one query run 100 times: 25 correct answers on one, 100 on the other.  $\chi^2 = 55.23$ ,  $p < 0.001$ .

On a second 70B model, one inference engine produced two distinct outputs across 100 identical calls — a 92 / 8 split — where two other engines produced one.  $\chi^2 = 16.41$ .

The fix was a runtime flag, not a prompt. Batch size changes numerics, and numerics change token probabilities.

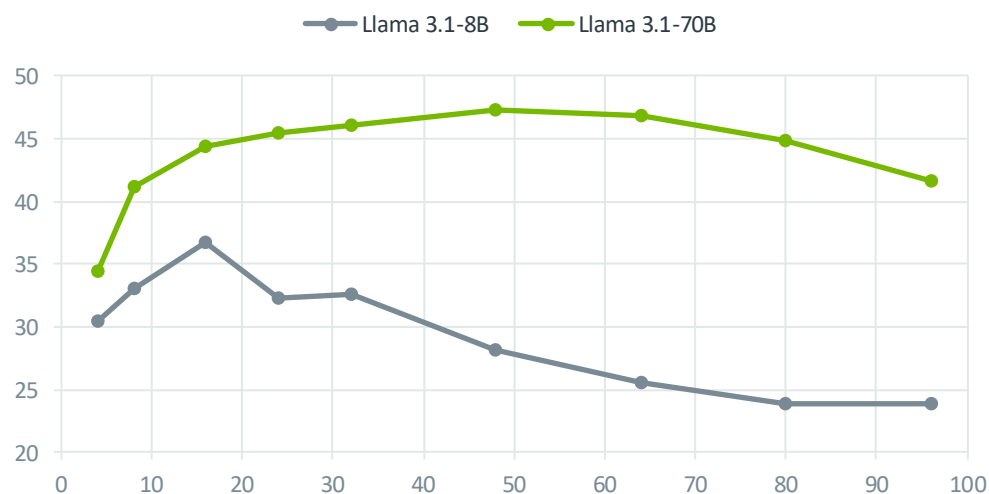
## Independently replicated at roughly 100× scale, five months later

Pape, Evertz and Schönherr surveyed 200 inference engines and found that backend choice alone shifts benchmark scores by up to 16.6 percentage points — holding weights, decoding parameters and hardware constant. Across 35,000 ML publications, the inference stack is almost never reported.

# More context is not always better

*InfinityBench EN.QA, Llama 3.1 as the generator. Answer quality against retrieved context length.*

## EN.QA · F1 score vs context (K)

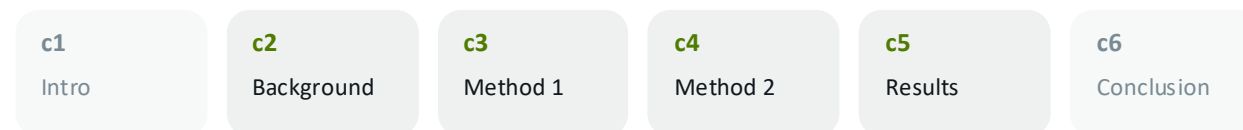


Peak 47.25 F1 at 48K. Same model, no retrieval at 117K: 34.26.

### An inverted U

The peak moves with the model — 8B at 16K, 70B at 48K. Sweep it, and re-sweep after an upgrade.

## The concrete example · one document, six chunks, top-4 retrieved



Top 4 by similarity score: c4, c2, c5, c3



### Vanilla RAG — score-descending

**c4 → c2 → c5 → c3**

The model reads Method step 2 before step 1.



### Order-preserve — same four, sorted by chunk index

**c2 → c3 → c4 → c5**

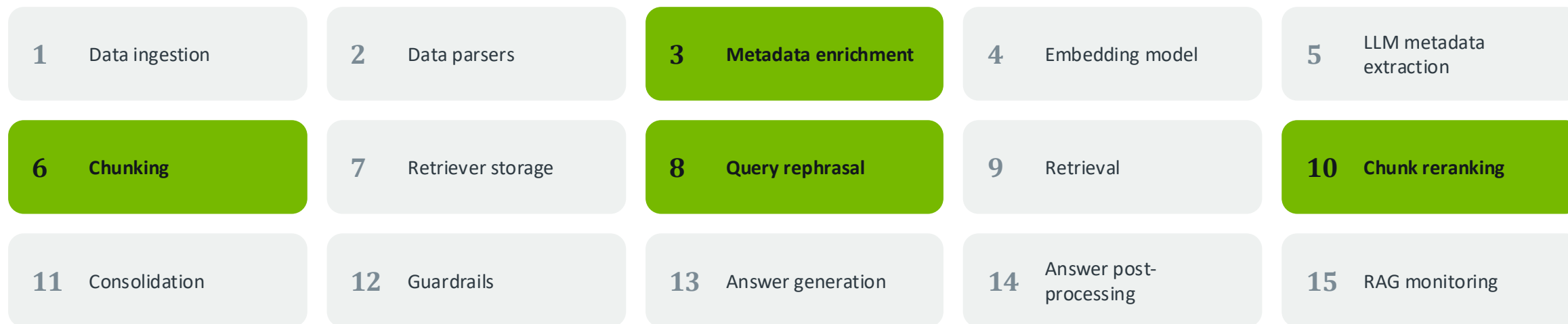
Background → Method 1 → Method 2 → Results.

**38.40 → 44.43**

F1 at 128 chunks, retrieved set held constant.  
One sort, six points.

# Fifteen control points — and the four that decide your accuracy

*Findings based on three production assistants.*



**Not one of the four is the model. That is the whole lesson, in one slide.**

Answer quality depends on retrieval relevance, and retrieval relevance depends on those four. Everything else is table stakes. Two production rules came out of this. Vector databases are bad at entities, because dense embeddings compress away exactly the rare tokens — people names, part numbers, internal acronyms — that enterprise queries hinge on; run hybrid lexical and vector search. And generate citations from the chunk source, never from the model: a hallucinated citation is worse than none, because it looks verified.

# We opened the agent loop and counted the lines

# 13,423

lines in the production agent loop of our employee assistant

## Most of it is not AI.

Permission checks. Consent prompts. Audit emission. Retry and rollback. Redaction. Tool allow-listing. Rate limiting. Identity propagation. Graceful degradation so the user is never blocked.

### Governance code exceeds AI code by roughly an order of magnitude.

- Budget for it. The demo is a weekend; the guards are the quarter.
- It is where durable differentiation lives — prompts are copyable, this is not.
- Every team building from scratch converges on the same thirteen thousand lines. Build it once, centrally.
- And it is why the agent runtime matters less than it appears: the runtime is the cheap part.

# Three times we deleted the agent — and it got better on both axes

1

## Text-to-SQL over supply chain data

Rejected an agentic architecture. Preprocessing, mined few-shot examples, deterministic SQL post-processing, plain-language explanation.

**11.8% → 100%**

2

## Long-document financial assistant

Dropped query decomposition entirely; moved to whole-document long-context retrieval.

**+7.7 pts -46% latency**

3

## Enterprise knowledge assistant

Collapsed a seven-expert mixture-of-experts into one planning agent over thin retrieval tools.

**fewer calls, lower latency**

!

## But the symmetric failure is real — a pure rules engine could not absorb natural-language rules

The pattern that works: the model reasons over rules; deterministic code executes the calculation.

# Four questions we ask before building an agent

*Run at project intake. It has killed more agent proposals than it has approved.*

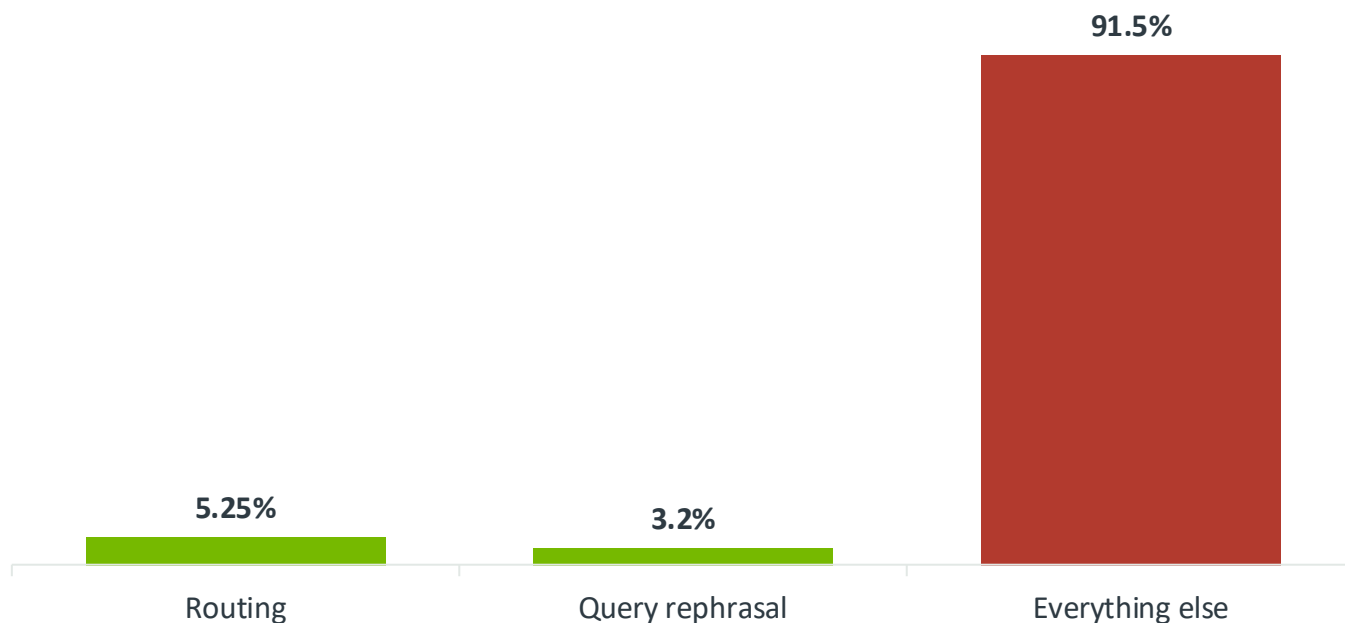
| Question                                                                               | If no / low                                                             | If yes / high        |
|----------------------------------------------------------------------------------------|-------------------------------------------------------------------------|----------------------|
| Is the task genuinely complex — planning across steps you cannot enumerate in advance? | Build a workflow. Faster, cheaper, testable.                            | Agent                |
| Is it valuable enough to absorb the per-invocation cost?                               | Workflow. Sub-cent tasks do not survive an agent loop.                  | Agent                |
| Are all parts doable with the tools and data you have?                                 | Reduce scope. A partly-capable agent erodes trust faster than no agent. | Agent                |
| What is the cost of an error — and how fast would anyone notice?                       | Read-only, or human-in-the-loop confirmation.                           | Autonomous execution |

**AI workflows are still right for a large share of enterprise use cases.**  
**“Agentic” is an architecture, not a goal.**

And when you do build one: autonomy is granted, not assumed.  
Observe → suggest → confirm → execute. Human-in-the-loop is the adoption mechanism, not a limitation.

# Attribution beats intuition — by an order of magnitude

*Three months of production feedback from a 30,000-user assistant, classified by failure stage.*



## We fixed the 8.45%.

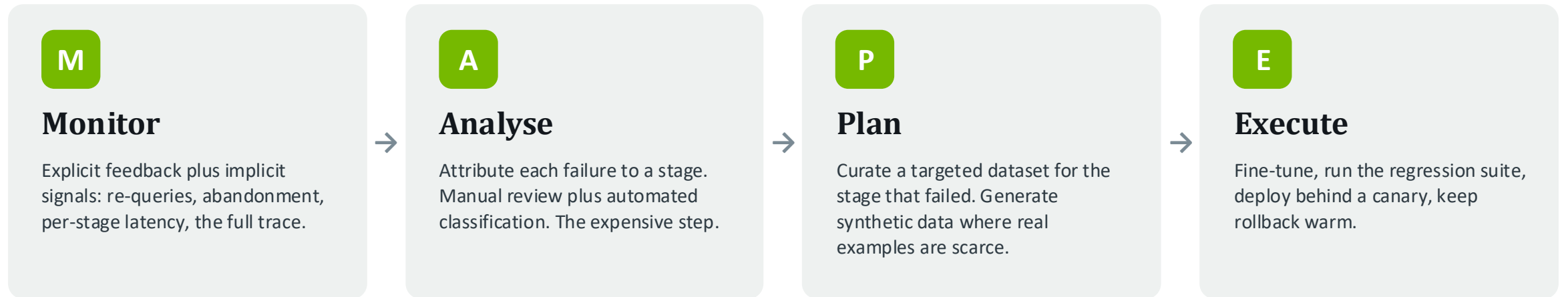
Deliberately — they were the tractable targets, and the results were genuinely good.

But the honest reading is that the overwhelming majority of what users experience as failure lives in retrieval, ranking and generation, where attribution is much harder.

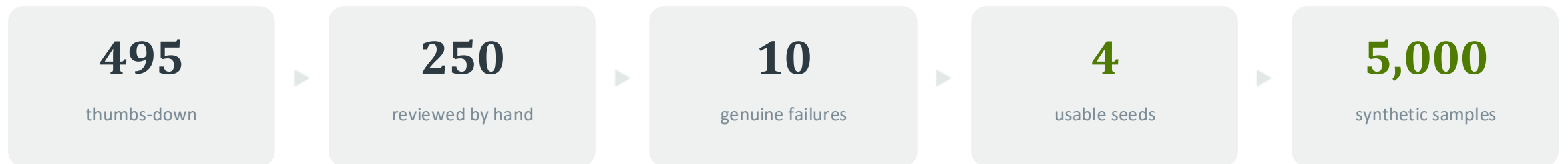
Skip attribution and optimise on instinct, and you will ship a real improvement to eight percent of your failures that no user will notice.

# The flywheel is a control loop, not a metaphor

*MAPE — monitor, analyse, plan, execute — borrowed from twenty years of self-adaptive systems research.*

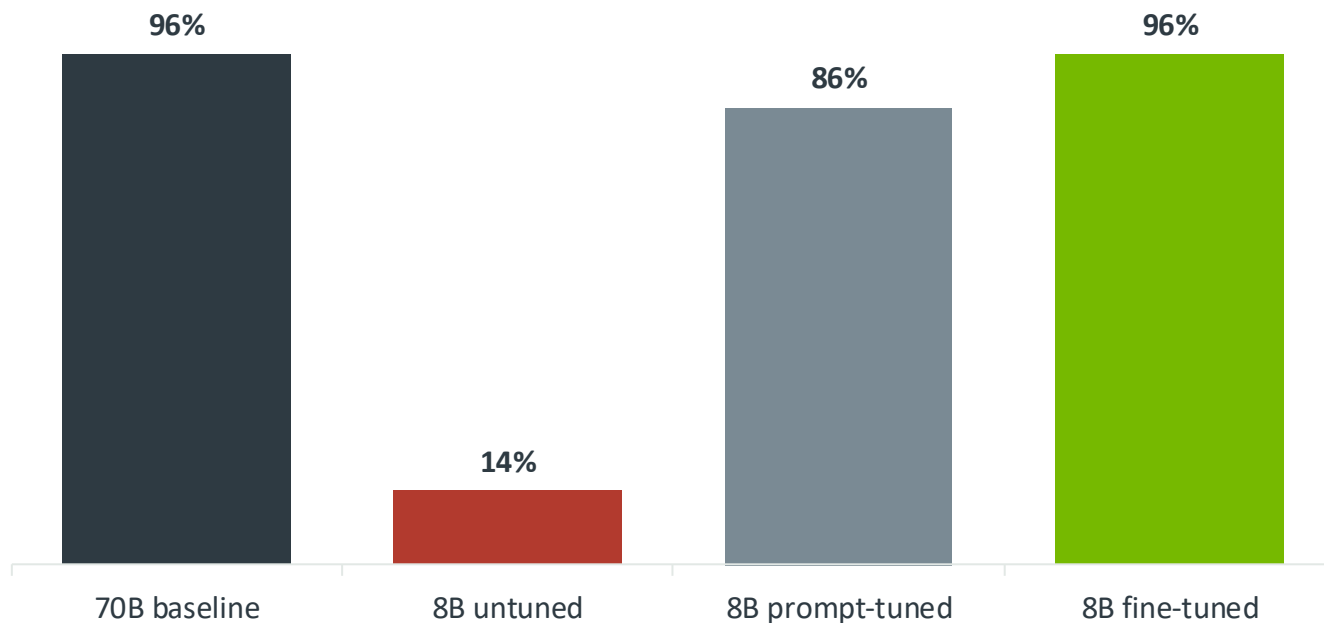


**What the Plan step actually cost — the funnel, at real scale**



# Ten times smaller, same accuracy, seventy percent faster

*Expert routing. The untuned small model is the number that matters most here.*



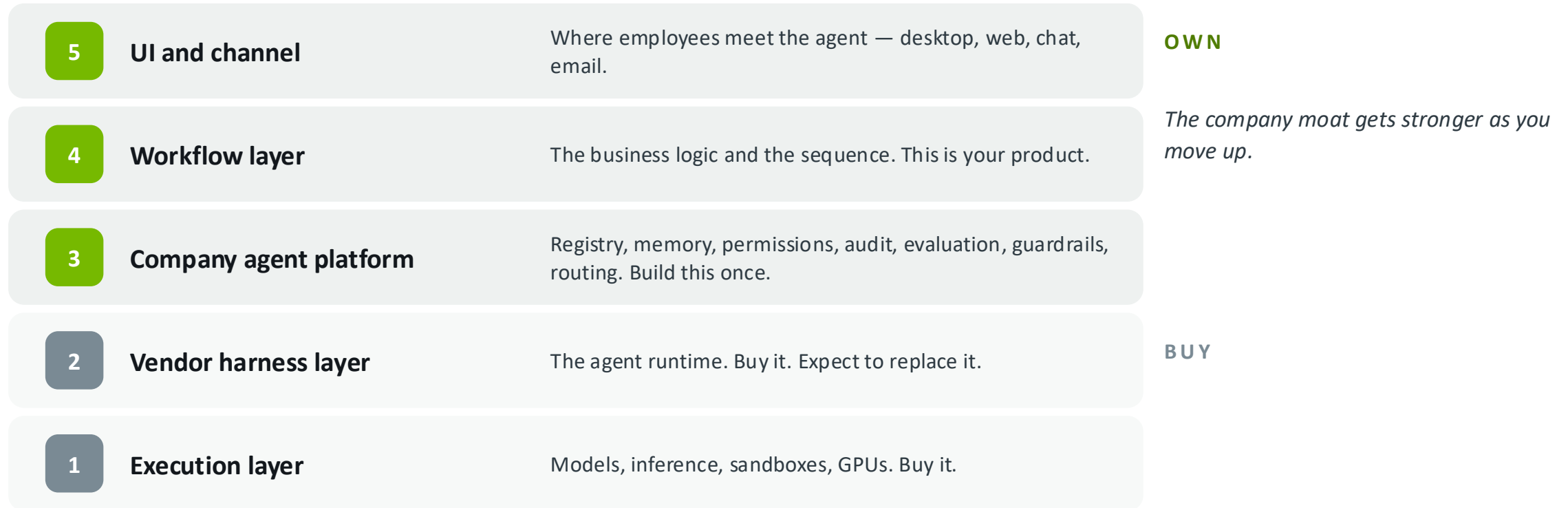
**0.26s → 0.08s**

latency at identical accuracy

The 14% is the point. Out of the box the small model could not do the task at all. Prompt engineering got it to 86% — close enough to be tempting, far enough to be unshippable. Fine-tuning on a curated set built around thirty-two real production failures closed the last ten points.

Same pattern on rephrasal: 73.8% → 77.5%, with a 40% latency cut.

# Enterprise AI is a five-layer cake — your moat is at the top



**Don't rebuild every sandbox. Build the enterprise control plane that makes every vendor replaceable.**

# Your feedback is scarcer, and your judge is wronger, than you think

**30,000**

employees served

**800**

active users per week

**1,224**

feedback signals in 3 months

**495**

of them negative

## What that does to your method

- Feedback is negatively biased by construction. Your labelled set is not a sample of your traffic — it is a sample of your worst traffic.
- A thumbs-down tells you something broke, not what. Instrument implicit signals: re-queries, abandonment, copy events.
- Privacy rules may forbid storing the pairs you need. Plan redaction before you plan analysis.

## And the judge that saves you the reading

**140**

routings flagged  
by the judge

**32**

were actually  
incorrect

*We still use the judge every day. We just never let it write ground truth.*

# You cannot evaluate a personalised agent with a prompt set. You need a world.

## Why prompt sets fail

An agent's decisions depend on persistent identity and evolving state. Tasks belong to someone. Ownership changes. Priorities compete. Context accumulates across mail, chat, calendar and project history. Without a stateful model of the employee, a benchmark has no notion of my task, my responsibility, my stakeholders.

## Synthetic, but stateful

Real employee data is privacy-sensitive and cannot be replayed. So the substrate has to be synthetic — but synthetic messages alone are only observations. They carry no durable identity, ownership, task state, or cross-channel history. The world is the artifact, not the prompts.

**100**

synthetic employee  
worlds

**1,550**

ground-truth  
priorities

**4,513**

signal messages

**5,834**

noise and trap  
messages

*More than half the benchmark is distractors — because that is what a real inbox is. Public agent benchmarks are almost entirely signal, and almost none evaluate noise rejection at all.*

# From 17 agents to 7.5 million

*“Those 75,000 employees will be working with 7.5 million agents.” — Jensen Huang, GTC 2026*

**1 The bottleneck was never the model**

7.5 million agents cannot be configured by hand. They have to be generated from context that already exists.

**2 Agentic is an architecture, not a goal**

Every unnecessary model call gets multiplied by millions. Restraint stops being taste and becomes economics.

**3 Deployed is not done**

Nobody can hand-maintain 7.5 million agents. Either they improve themselves, or they decay.

**4 Own the control plane, not the harness**

Thirteen thousand lines of governance, times 7.5 million, is not a number. You build it once or not at all.

**5 Evaluation is the frontier**

You cannot human-review 7.5 million agents. This is the one we have not solved.

## That is the part we have to build.