

You rent a frontier mind for every decision. Most were never a frontier problem.

Agentic systems run on thousands of small, bounded decisions a day, and most enterprises send every one to a frontier model. A \$92 training run and a new class of decision model show that this is a choice, not a necessity — and that the decision layer belongs to the enterprise.

SUMMARY OF THE ARGUMENT

Most of what an enterprise AI system does is not reasoning. It is a long run of small, bounded decisions: which queue a request belongs to, how urgent it is, whether a submission is complete, whether the next action is safe to take. In most stacks today, every one of those decisions is a call to a frontier model — billed per token, hosted elsewhere, and changed on someone else's schedule.

Two developments in 2025 and 2026 make that habit optional. Andrej Karpathy's public work showed that a working chat model can be trained end to end for about ninety-two dollars on a single node, which turns *could we own a model?* from a research question into a line item. And TypeSafe's Jev showed that a model can be built to return only typed decisions with calibrated confidence, never free text.

This paper argues that every governed system should have a **decision layer**: one interface per decision, with several minds behind it. Rules decide first. A small model the enterprise owns handles the routine decisions inside its own network. A System One model is an optional adapter where data residency allows. The frontier model is kept for language, judgement and the cases the cheaper minds are unsure about, and a person sees only the genuine forks.

The result is a cost that stays flat as volume grows, decisions that behave the same way twice, and data that stays where the regulator wants it. The paper sets out the source of the argument, the three kinds of mind, how they fit together, what it takes in practice, the economics, where it works and where it does not, and a phased path to adoption.

IN THIS PAPER

- 1 The source, and what it proves
- 2 The decision economy
- 3 Three minds for decisions
- 4 How they come together
- 5 What it actually takes
- 6 The economics
- 7 Where it works
- 8 Where it does not, and the risks
- 9 What to look for in a delivery model
- 10 A phased path to adoption

1 The source, and what it proves

This paper starts from *Small Language Model Engineering 2026*, an independent study note that synthesises Andrej Karpathy's open-source repositories, discussion threads and public posts. The note is not written, reviewed or endorsed by Karpathy, and neither is this paper. The figures below are as Karpathy published them in his nanochat discussions, reached through that note. We credit both and claim neither.

The central result is **nanochat**, released in October 2025: one readable repository that takes raw text all the way to a chat model through four named stages — pretraining, midtraining, supervised finetuning and optional reinforcement learning. A January 2026 miniseries extended it into a ladder of model sizes, each set by a single dial, with a published cost for each rung.

Run	Size	Time on 8×H100	Cost	Benchmark (CORE)
Depth 17	GPT-3 Small grade	~26 minutes	~\$10	0.148
Depth 20, the speedrun	~561M parameters	3 h 51 m	\$92.40	0.222
Depth 32	Above GPT-2	~33 hours	~\$1,000	0.31
Depth 181, extrapolated	GPT-3 175B grade	~1,869 days	~\$1.07M	0.427

Three lessons carry over directly to enterprise systems.

- **Ownership has a price, and it is small at the bottom.** A model that beats the 2019 state of the art costs about a thousand dollars to train. General capability is what becomes expensive; narrow jobs sit on the cheap rungs of the ladder.
- **Small models are poor generalists.** The \$92 model scores close to chance on a broad knowledge test and solves roughly one grade-school maths problem in twenty. Nobody should mistake it for an assistant.
- **Method beats model.** One dial instead of a hyperparameter sweep, staged training where each stage has a named job, and measurement against something you did not train yourself. Those disciplines transfer even where the model itself does not.

The question “could we just train one?” has stopped being rhetorical. It is now a number you can put next to the invoice you already pay.

2 The decision economy

An agent loop is mostly decisions, not thought. Before an AI worker drafts a single reply it has already asked a series of short questions: what is this, how urgent is it, which tool applies, is the input complete, is the next action safe, was the last output acceptable. Each answer is short, and each is drawn from a fixed set of options.

Those decisions dominate volume. A single customer request triggers several of them, and in most stacks a frontier model answers every one. The bill grows with traffic, the latency stacks up step by step, and two runs on the same input can disagree with each other — which is exactly what an auditor does not want to hear about a decision that moved money or touched a customer.

The dividing line

What can move off the frontier model follows a simple rule.

- **If the answer is a function of the input, a small model can learn it.** Routing, triage, classification, completeness checks, risk gating, format validation.
- **If the answer needs knowledge that is not in the input, it cannot.** Explaining a regulation, drafting a letter, weighing a credit case, planning a piece of work.

The second kind stays with the frontier model — or, better, with a model that looks the facts up rather than remembering them. That is Karpathy’s own picture of where small models are heading: a compact “cognitive core” that sacrifices encyclopaedic recall for capability and consults the world for facts. Most of the first kind was never a frontier problem at all.

3 Three minds for decisions

Three kinds of model can now make a routine decision. They differ less in what they can do than in what the enterprise ends up holding.

	Frontier model	System One model (Jev)	Owned small model
What it returns	Text, which the system must parse	A typed choice, score or yes/no, with probabilities	Whatever it is trained for: a label from a fixed set
Confidence	Self-reported, usually overconfident	Trained to be calibrated	Must be calibrated on the enterprise’s own held-out data
Speed	Seconds per decision	70–500 ms, as published by the vendor	Milliseconds on local hardware
Cost shape	Per token; grows with traffic	Per input token; very low early-access pricing	Flat: the server, whatever the volume
Where it runs	Vendor cloud	Vendor cloud, US-hosted at launch	The enterprise’s network or an in-region host
Changes under you	Yes, on the vendor’s schedule	Yes, by vendor version	No: the same weights until you retrain
Best at	Language, judgement, open questions	Instant atomic judgements on data it may be sent	High-volume decisions on the enterprise’s own traffic

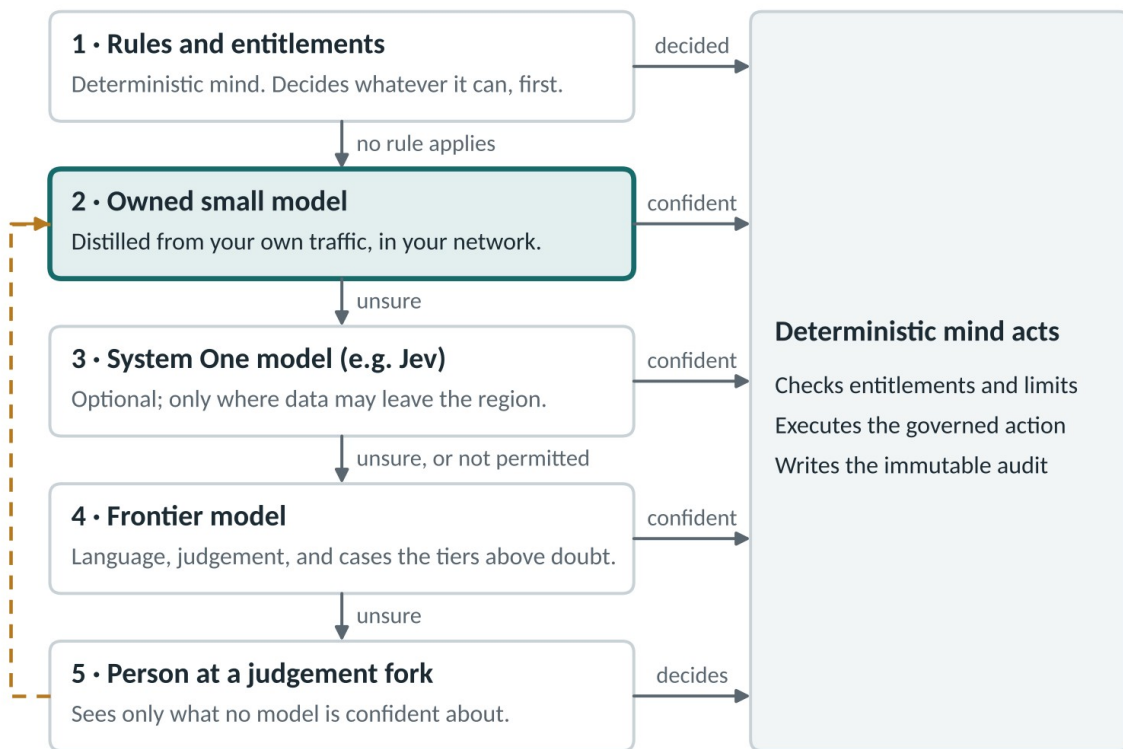
Jev, launched in early access by TypeSafe AI in September 2026, is the first public “System One” model, named for Kahneman’s fast, intuitive mode of thought. It gives up text generation entirely: the system defines the answer space and the model returns a probability distribution over it. That is why it cannot invent an option, though it can still choose the wrong one. Its real contribution is the interface: typed answers, honest confidence, and decisions decomposed into atomic questions recombined in deterministic code.

The **owned small model** is what Karpathy’s work makes affordable. It has none of Jev’s built-in calibration, and it must be trained, served and watched. But it has the one property neither rented option can offer: it never leaves the enterprise’s control.

4 How they come together

The three minds are not rivals; they are rungs. Each decision starts at the cheapest mind that might answer it and climbs only when that mind is unsure. Above them all sits a deterministic layer that has the final say: whatever tier answers, rules check the answer, execute it and record it.

FIGURE 1 • THE DECISION CASCADE



Dashed line: every escalated case becomes training data for the owned model.

A decision moves to a costlier mind only when the cheaper one is unsure. Whichever tier answers, the deterministic mind checks entitlements, executes the action and writes the audit.

Three properties of the cascade matter to a regulated buyer.

- **One contract, swappable minds.** Each decision is defined once: its input, its allowed answers, its confidence threshold and its escalation rule. The minds behind it are adapters. Changing a model does not change the system.
- **Residency is a routing rule.** For data that may not leave the region, the System One tier is simply skipped and the owned model hands straight to the frontier tier, which can itself be an in-region deployment.
- **The system improves where it is uncertain.** Every case escalated past the owned model becomes training data for its next release. The frontier model's share of decisions falls as the owned model learns, and the person at the top sees genuine forks only.

This sits naturally inside an architecture that already separates a deterministic mind from a governed language mind, as earlier papers in this series have described. The decision layer is not a new product category. It is a socket that a well-governed system already has room for.

Rules decide first and act last. Everything in between is a question of which mind is cheapest and still confident.

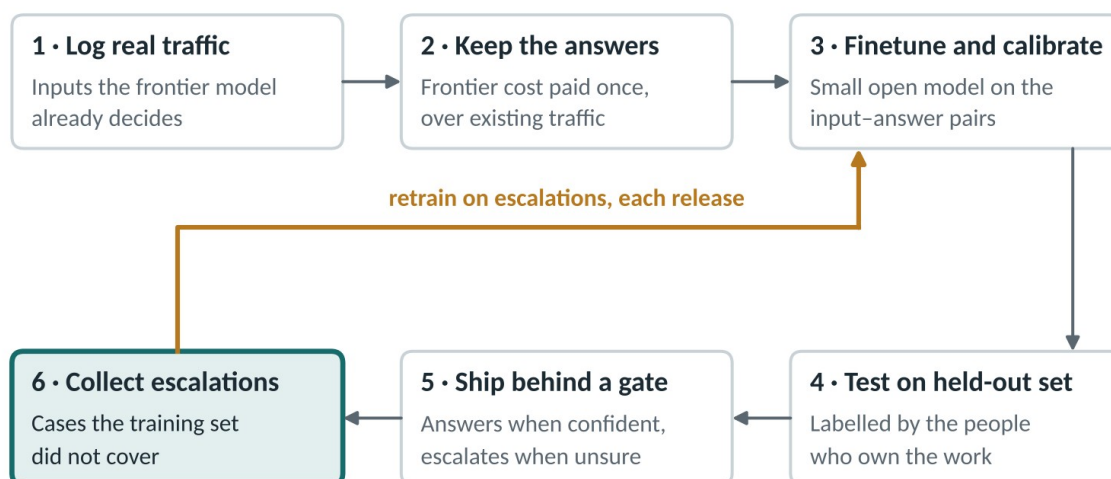
5 What it actually takes

Almost nobody should train from scratch. The study note sets out four routes to a small model that does one job, in rising order of effort. The right one is the cheapest that clears the bar.

Route	When it fits	Effort
Prompt an open small model off the shelf	The task is generic	Hours
Finetune that model on labelled examples	A few thousand labels exist	Days; tens of dollars
Distil from the frontier model on real traffic	The decision already runs in production	One to two weeks; hundreds of dollars
Train from scratch, nanochat-style	The data looks like nothing on the shelf	Weeks; \$100–\$1,000 of compute

For an enterprise already running AI, **distillation is the right default**, because the training data already exists in its logs.

FIGURE 2 • THE DISTILLATION LOOP



The frontier model is paid once, for the teacher pass. After that, the escalation log does the teaching: the owned model improves in production, where a prompted model stays where it started.

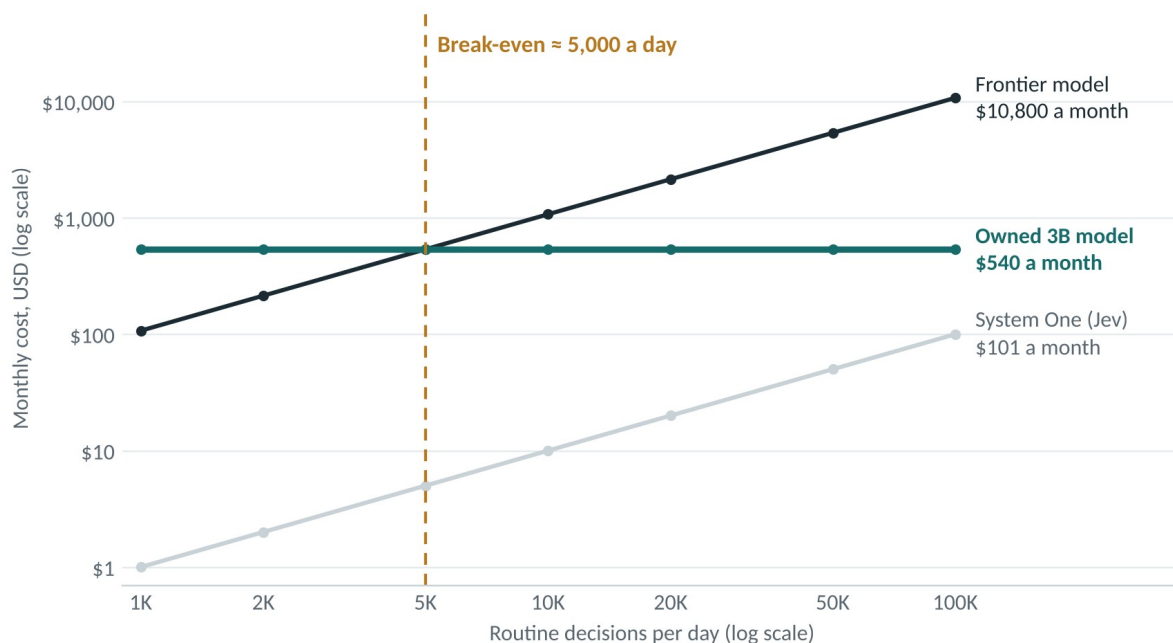
Two engineering facts decide whether it works

- **Calibration is not free.** A finetuned small model's raw scores are not honest probabilities. Constrain it to a fixed label set, then calibrate its scores against a held-out set before setting the gate threshold. Jev is trained for this; an owned model has to be tuned for it.
- **Size decides where it runs.** Weight memory is roughly parameters multiplied by bytes per parameter. A three-billion-parameter model needs about 6 GB at 16-bit precision and about 1.5 GB at 4-bit, which fits a single modest GPU in the enterprise's own data centre. The accuracy lost to compression must be measured on the enterprise's own task, not on a public benchmark.

6 The economics

At volume, the frontier bill is the one that scales; the other two barely move. Figure 3 works through a single routine classification decision of about 800 input and 20 output tokens.

FIGURE 3 • MONTHLY COST OF ONE ROUTINE DECISION, BY DAILY VOLUME



Worked example from list prices, not a measurement. Frontier at \$4 / \$20 per million input / output tokens; System One at \$0.042 per million input tokens (TypeSafe early-access price); owned 3B model on one rented GPU at about \$0.75 an hour. Engineering time, utilisation and the cost of a wrong answer are excluded.

The frontier and GPU figures follow the study note's worked example; the System One line is our own arithmetic from the vendor's published price. Owning also carries a one-off of a few hundred dollars for the teacher pass. The honest reading has three parts.

- **On price alone, renting a System One model wins.** Owning a model is not the cheapest way to make a routine decision.
- **Owning wins on everything price leaves out.** No data leaves the network, nothing reprices or depreciates, and the decision behaves the same way next quarter as it does today. For regulated buyers in the Gulf, that usually decides the matter.
- **Below the break-even — about 5,000 decisions a day in this example — keep renting.** The engineering effort to own a model costs more than the bill it saves.

The threshold question is not “is the small model as good?” It is “at what volume does the recurring bill exceed the one-off cost of owning?”

7 Where it works

The best candidates share four traits: high volume, a fixed answer set, an answer drawn from the input alone, and existing traffic to learn from. Across the kinds of system enterprises are now putting into production, these fit well.

Workload	The decision	Escalates when
Agent action gating	Is this step safe, does it need approval, or is it blocked?	Confidence is low, or the action writes to a system of record
Request routing across a suite of systems	Which agent, queue or module owns this?	The request spans domains
Order and intake triage	Intent, urgency and completeness	Information is missing or the customer is unusual
Fund-flow classification	Which programme category; is it anomalous?	A transaction falls outside its pattern
Lending and leasing document intake	Document type; complete or not	A document is unreadable or contradicts another
Regulatory query routing	Which jurisdiction and domain applies?	The question crosses domains or has no clear home
Support and operations queues	Category and priority	The ticket is a complaint, or high-value

Agent action gating is the clearest first win: it is the most frequent decision in any agentic system, its labels are stable, and every escalation is already a human checkpoint. Classification of financial flows that carry no personal data is the cleanest pilot for an owned model, because it removes the residency question from the first experiment.

8 Where it does not, and the risks

Small models fail confidently when they are asked to know things. Anything that depends on world knowledge, multi-step reasoning or careful writing stays with the frontier model: drafting, underwriting judgements, compliance answers, specifications and planning.

The risks of a decision layer are known ones, and each has a known control.

Risk	Control
Measuring on impressions. A small model that looks right on ten hand-picked examples fails on the long tail.	Build the held-out set before any training; ship nothing that has not been measured against it.

Risk	Control
False confidence. An uncalibrated gate lets wrong answers through with a high score.	Calibrate, then set the threshold from the cost of a wrong answer, not from a default.
Drift. Inputs change, and a model trained once degrades quietly.	Treat the escalation log as the early warning and the retraining set.
Vendor and residency exposure. Jev is early-access, US-hosted and benchmarked by its own vendor.	Use it only where data may leave the region, until that changes.
Operating overhead. Serving, monitoring, rollback and retraining are real work.	Treat the owned model as a capability to run, not a file to download.
Optimising one number. Automated tuning loops find ways to move a metric that nobody intended.	Keep a person in charge of what is being measured.

9 What to look for in a delivery model

An enterprise does not need to build a decision layer from first principles. It needs whoever builds its systems to have made room for one. Five tests separate a system that can host a decision layer from one that will have to be rebuilt to get it.

1. **Does the deterministic layer decide first and act last?** If a language model sits in the decision path with the final word, no amount of tiering will make the system governable.
2. **Is each decision defined as a contract?** Input, allowed answers, threshold and escalation rule, written down once, with the model behind it treated as an adapter.
3. **Does the system run with the language layer down?** Every model-touched workflow should have a rules-only fallback. An owned small model makes that fallback far more capable; it should not be the first thing that ever sat there.
4. **Who owns the model trained on your traffic?** A distilled model is built from the enterprise's own data. It should be the enterprise's asset, in its own network, under the same terms as the source code.
5. **Does the commercial model reward a cheaper decision?** If the deliverer is paid by usage of a rented model, it has no reason to move decisions off it. If it is paid on outcome or hands over the source, it does.

10 A phased path to adoption

Start with one decision, not a platform. Each phase has a gate, and most decisions stop well before the last phase.

Phase	Work	Gate to the next phase
1 · Find the decision	Pick the decision the system makes most often. Log real inputs and the frontier model's answers.	A steady stream of logged input-answer pairs

Phase	Work	Gate to the next phase
2 · Build the yardstick	The people who own the work label a held-out sample by hand and agree the accuracy that would justify switching.	A written ship threshold
3 · Try the shelf	Prompt an open small model — and a System One model where residency allows — against the yardstick.	If either clears it, ship behind the confidence gate
4 · Distil	Otherwise, finetune a small open model on the logged pairs and calibrate it.	The threshold is met on the held-out set
5 · Place it	Compress the model, measure the damage on the same set, and size hardware in-network or in-region.	Accuracy and latency hold after compression
6 · Run the flywheel	Ship behind the gate, send unsure cases up the cascade, retrain on them each release.	Then pick the next decision

None of this makes a small model the equal of a frontier one, and nothing in Karpathy's work claims it does. What it removes is the mystery. Having trained or distilled one model for one decision, an enterprise will know which of the thousand calls in its system were never a frontier problem at all — and it will own the answer to every one of them.

The reason to own the small model is not that it is better than the frontier one. It is that, having done it once, you know which decisions never needed renting.

About this series

This is the ninth paper in a series on enterprise AI as an operating system rather than a tool. Paper One established where the value is. Paper Two set out the method for re-deriving a process from its outcome. Paper Three described delivery, and Paper Four turned the method into a strategy. Paper Five applied it to the accounting and advisory practice; Paper Six separated the commercial model from the delivery model that makes an outcome contractable. Paper Seven described the governed workforce that does the routine work, and Paper Eight addressed who builds and who carries the risk. This paper turns to the smallest unit inside all of them: the individual decision, and which mind should make it.

Every paper is readable at xamun.ai/whitepapers. This paper is Xamun's own position. It is not affiliated with, reviewed by or endorsed by Andrej Karpathy, Eureka Labs, OpenAI or TypeSafe AI. Model figures are reproduced as published by their authors; cost figures are worked examples, not measurements.

Sources and references

- *Small Language Model Engineering 2026* — independent study note synthesising Karpathy's public material; source for the four routes, the distillation loop, the memory arithmetic and the frontier and GPU worked example.
- A. Karpathy, "Introducing nanochat: The best ChatGPT that \$100 can buy," [karpathy/nanochat Discussion #1](#), October 2025 — the four-stage pipeline and the \$92.40 speedrun.
- A. Karpathy, "nanochat miniseries v1," [karpathy/nanochat Discussion #420](#), January 2026 — the depth ladder and its extrapolations.
- A. Karpathy, "\$1000 tier nanochat run," [karpathy/nanochat Discussion #8](#), October 2025 — the depth-32 run.
- A. Karpathy, public post on the LLM "cognitive core," 2025 — the small, tool-using model that looks facts up.
- A. Karpathy, autoresearch repository and public post, March 2026 — the automated experiment loop.
- TypeSafe AI, "Introducing System One Models and Jev," September 2026, and product documentation (docs.typesafe.ai) — Jev's answer types, calibration and pricing.
- DataCamp, "System One Models: Jev," 2026; P. Pillitteri, "TypeSafe launches Jev," 16 September 2026 — independent coverage of the launch.
- V. Sanh et al., "DistilBERT, a distilled version of BERT," 2019 — the reference point for how much a distilled student retains.
- Xamun Position Papers No. 1 to No. 8 (August–October 2026), xamun.ai/whitepapers.