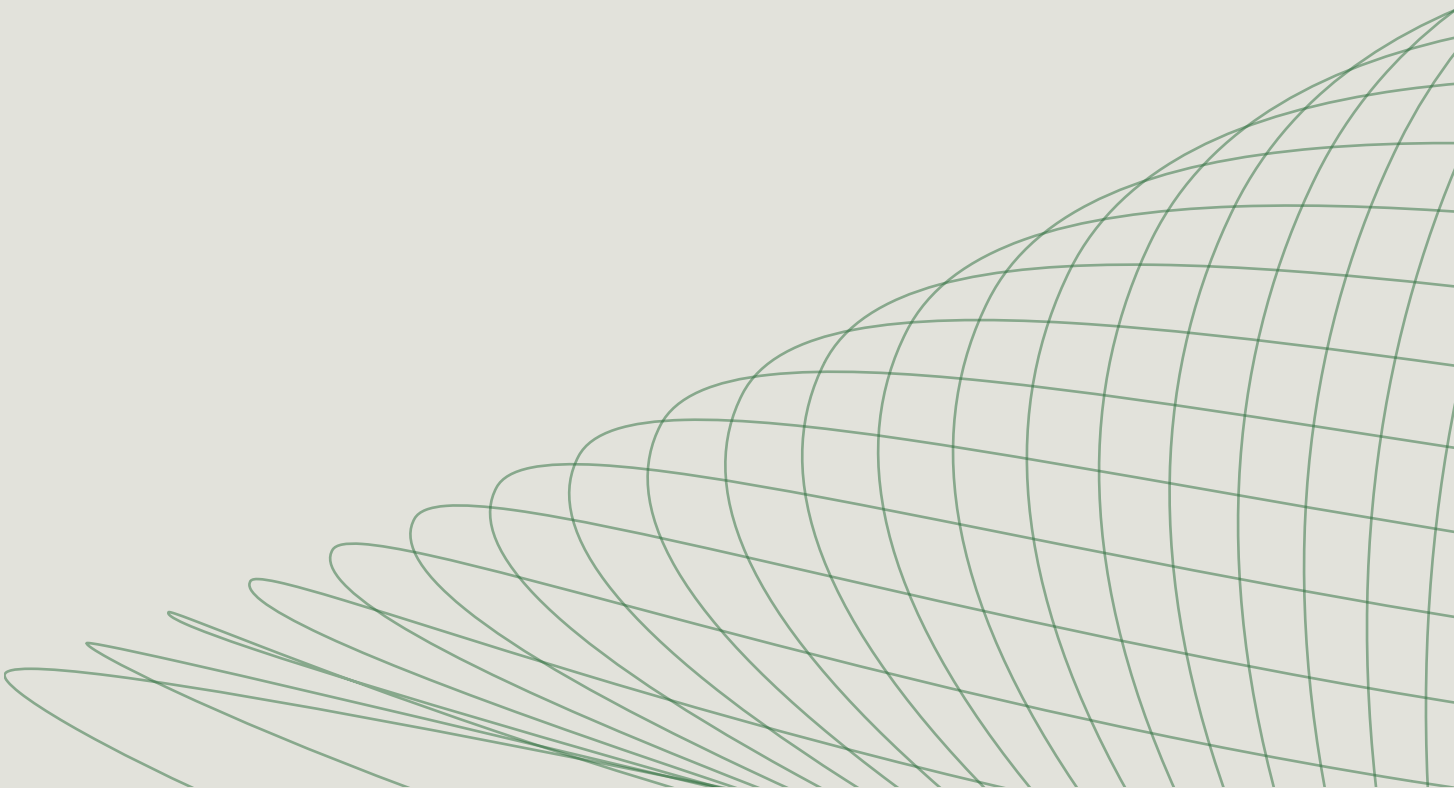
STACKLOK

How to thrive in a multi-model world

Enterprise AI value lives outside of the model, in the context, tools, policies, and knowledge around it. It's imperative that you own it.



Contents

An AI agent that does real work needs two things: a model, and the system an enterprise builds around that model. The model assembles no context, calls no tools, enforces no policy, and remembers nothing between sessions. The system does all of that. It is where the engineering effort goes, where institutional knowledge accumulates, and where the durable value of an enterprise's AI investment lives.

Enterprises rent the model and build the system. But right now, most enterprises are building that system inside someone else's walls, with little to no control.

This paper makes four arguments:

1. Frontier labs and hyperscalers have structured their agent stacks as walled gardens, not through malice but through good product design.
2. The economics of enterprise AI have shifted in ways that make those walls far more expensive than they looked eighteen months ago.
3. The components that actually create value, the context, tools, policies, memory, and evals, are exactly the components at risk of being trapped behind those walls.
4. The answer is an open agent platform: one where every part of that system is owned by the enterprise and portable across models, clouds, and vendors.

The dangers of walled gardens

The frontier labs and hyperscalers are not running a scheme. They are doing what well-run companies do: shipping coherent, vertically integrated products. OpenAI and Anthropic have built genuinely good agent stacks. The hyperscalers have wrapped those stacks in managed platforms that are easy to adopt and demo. That quality makes the walls hard to see.

Savvy leaders are watching where artifacts accumulate. For example:

- Agent definitions get written in one provider's format
- Tools and skills are packaged in a way that only one runtime loads
- Memory and conversation cannot be exported with fidelity

- Evaluation suites and traces live in one vendor's console
- Guardrails are expressed in a policy language with a single implementation
- Orchestration features are available on a first-party endpoint and nowhere else

No single item on the list above is a locked door. But if one or more of the artifacts sounds familiar, you face a growing swell of switching costs.

Independent analysts have started naming the mechanism. Kai Waehner, who advises Global 2000 enterprises on AI architecture, observes that when agents run on a vendor's proprietary orchestration layer, "lock-in compounds at every layer of the stack." Model choice, workflow automation, data, governance, and staff expertise become entangled with a single vendor, and each dependency multiplies the others.

Enterprises feel the weight before they can name it. In a 2026 AvePoint analysis of AI vendor dependency, 47 percent of enterprise executives said losing their primary AI vendor entirely would disrupt a key business function. When Builder.ai, a Microsoft-backed platform once valued at 1.3 billion dollars, collapsed in 2025, customers found themselves unable to access systems and data they had built their operations around.

The pattern is familiar to anyone who lived through the early cloud era, when enterprise workloads accumulated behind proprietary orchestration APIs, Kubernetes emerged to move the substrate into the commons. The same accumulation is happening one layer up. The difference is what gets trapped this time: not infrastructure configuration, but the enterprise's own operational knowledge.

Trends reshaping AI initiatives

Three changes in the AI market make this lock-in more expensive than it looked a year ago.

From tokenmaxxing to tokenomics

The early phase of enterprise AI rewarded whoever could consume the most tokens fastest, in *hopes* of maximizing productivity. That phase is over, and the numbers explain why. Deloitte analysis cited in Forbes found that per-token inference costs have dropped roughly 280-fold over two years, yet enterprise AI bills keep rising because consumption is growing faster than prices are falling. The FinOps Foundation's State of FinOps 2026 report, covering 1,192 organizations and 83 billion dollars in cloud spend, found that AI workloads now account for 18 percent of cloud spend at AI-forward enterprises, up from 4 percent in 2023.

Meanwhile the return side of the ledger is under scrutiny. PwC's 29th Global CEO Survey of 4,454 chief executives found that 56 percent report AI has produced neither increased revenue nor decreased costs.

Engineering leaders now carry token budgets the way they carry cloud budgets, and the winning operational pattern is routing each workload to the cheapest model that clears the quality bar. A summarization job does not need the same model as a code migration.

Tokenomics only works if switching models is cheap. Inside a walled garden, switching is the one thing that is never cheap.

From single-model to multi-model

Model portfolios are already the norm, whatever the incumbent stacks would prefer. In a16z's 2026 enterprise survey, 81 percent of respondents reported orchestrating three or more model families in production, up from 68 percent a year earlier. Menlo Ventures' State of Generative AI in the Enterprise, based on 495 US enterprise AI decision-makers, found 37 percent of enterprises running five or more models.

The pattern behind those numbers is deliberate. Tech-forward enterprises reserve frontier labs for high-stakes reasoning, route volume work to cheaper or smaller models, and keep a fallback ready for outages, price changes, and deprecations. Model choice is becoming a routine operating decision, made monthly or weekly rather than once per procurement cycle. The system around the model has to survive those decisions.

From frontier-only to open-weight options

Open-weight models such as DeepSeek, Qwen, and Kimi now rival proprietary systems on coding and much of everyday reasoning, and a Linux Foundation survey of more than 700 technology leaders across 41 countries found 63 percent already using open models somewhere in their stack.

Capability is no longer the constraint. What holds enterprises back from exercising model choice is everything around the model: governance, evaluation, and the effort of rebuilding a working system on a different foundation. An open kernel with a proprietary userland is not an open operating system. Open weights with a closed agent stack is the same trade. Downloading the model solves nothing if the system that makes the model useful cannot move with it.

All three shifts point in the same direction: the value of flexibility at the model layer is rising, and the walled-garden architecture is precisely what prevents enterprises from capturing it.

What is at stake

Gartner predicts that over 40 percent of agentic AI projects will be canceled by the end of 2027, citing escalating costs, unclear business value, and inadequate risk controls. Anushree Verma, Senior Director Analyst at Gartner, describes most current projects as "driven by hype" and "often misapplied."

Why do agent projects fail? Model capability is not on Gartner's list. The top factors are (1) cost, (2) value measurement, and (3) risk control. All of these are properties of the system around the model. Enterprises are not failing because the models cannot do the work; they are failing because the system that would make the work governable, measurable, and affordable was either never built or was built inside a stack that hides its costs and constraints.

Define your agent stack

The word "agent" hides the parts that matter. The system an enterprise builds around a model includes, at minimum:

- **The harness.** The runtime that plans, invokes tools, retries and repairs failures, and hands results to the next step. This is where most agent engineering effort actually goes, and where a vendor's proprietary formats bind deepest.
- **Tools.** Standards-based interfaces to enterprise data and actions, increasingly via MCP servers. A tool inventory encodes years of decisions about which systems agents may touch and how.
- **Skills.** Instructions and packaged behaviors an agent can load: how to file a ticket, how to structure a compliance memo, and how your organization defines done.
- **Context.** Documents, playbooks, conversation histories, and the pipelines that assemble them. Context engineering is where domain knowledge becomes machine-usable.
- **Memory and state.** What agents learn across sessions and carry through a task in flight. A memory store that exports only as a transcript dump is not owned; it is hostage.
- **Evals.** The test suites and results that define what "correct" means for your business. Evals are how buyers exercise choice (and ensures vendors don't grade their own homework).
- **Policies and audit.** Who may call what, under which constraints, with records a security team can inspect. Regulations such as the EU AI Act require exactly the kind of inspection that black-box stacks resist.

Claims adjudication, clinical documentation, fraud investigation, chip verification: the domain knowledge for these lives in enterprises, not in labs. Every one of the above listed components encodes that knowledge, which means every one of them is an asset, and every one of them is currently at risk of being expressed in a format only one vendor can run.

The seams are starting to open. Finish the job.

The industry has already proven that neutral seams work at this layer. The Model Context Protocol, released as an open standard in November 2024, was rapidly adopted by OpenAI, Google, Microsoft and more, and its SDKs now see roughly 100 million monthly downloads. In December 2025, the protocol was donated to the Agentic AI Foundation under the Linux Foundation, placing tool connectivity under vendor-neutral, community governance. The public server ecosystem has passed 10,000 active servers.

MCP demonstrates the pattern: publish the seam, govern it neutrally, and an entire market forms around it. The protocol's own maintainers report that enterprises deploying it now run into a predictable set of production needs, including audit trails, SSO-integrated authentication, and configuration portability. Those needs mark the frontier of what must open next.

Because one open seam is not an open system. Tool connectivity is standardizing. Agent and skill definitions, memory and state export, evaluation formats, and policy languages remain, for the most part, proprietary to each stack. An enterprise can move its tools today and still lose its agents, its accumulated context, its evaluation history, and its policy posture in a migration. The commons needs to extend across every component listed above, with conformance that is tested rather than asserted.

The case for an open agent platform

An open agent platform is the architecture that follows from these facts: open-source components, specified interfaces at the seams, and every artifact in the system stored in a form that any runtime can execute. The argument for it comes down to three kinds of control.

Control of cost

When agent definitions, tools, and evals are portable, model selection becomes a policy decision instead of a migration project. Enterprises can run a frontier lab where quality demands it and an open-weight or small model where volume demands it, and adjust that routing as prices move. Given that inference now dominates operational AI budgets, and that FinOps practitioners rank AI cost management as their most sought-after skill for the coming year, the ability to reroute workloads is not an optimization. It is the primary cost lever an enterprise holds.

Portability also changes every negotiation. A vendor facing a customer with a credible exit will price differently than a vendor facing a customer with two years of unexportable state. Tokenomics requires portability to be a strategy.

Control of data

A portable system can be self-hosted. Context, memory, credentials, session state, and audit logs stay inside the enterprise's own trust boundary, on infrastructure its security team already governs, subject to network policies it already enforces. Open-source components can be audited rather than trusted; a claim about where data flows can be verified by reading the code and inspecting the cluster rather than by accepting a compliance PDF.

For regulated or security-conscious organizations, that is the difference between a deployable architecture and a compliance finding. It also answers the sovereignty question that 2026's export-control disputes pushed to the center of vendor selection: a system you host on your own infrastructure, in your own jurisdiction, cannot be switched off from someone else's.

Control of destiny

The AI market is turbulent. Pricing changes overnight, models are deprecated on short notice, startups are acquired or collapse, and today's benchmark leader is next quarter's also-ran. An enterprise whose system is portable can absorb all of that: swap the model, keep the harness, tools, memory, evals, and policies intact. An enterprise whose system is fused to one stack inherits every one of its vendor's strategic decisions, and pays the price.

There is precedent for openness winning this argument. A decade ago, workload orchestration could have remained a set of proprietary vendor products. Kubernetes moved it into the commons instead, and the result was a larger market, not a smaller one, including for the vendors who bet on openness. The substrate became shared and the competition moved up the stack to where it benefitted their customers most. The same choice is being made right now at the agent layer, and the stakes are higher, because this layer holds the enterprise's knowledge rather than its infrastructure.

The argument for vertically-integrated stacks

Vertical integration is producing excellent systems right now. Offerings from the frontier labs and hyperscalers have increasingly improved in performance. Tight coupling between a model and its harness lets a lab co-design them and ship capability faster. Premature standardization is a real failure mode, as it can calcify layers that should keep moving.

The answer is sequencing. Standardize where the shape of the problem is understood, and let the rest race. Tool invocation is already standardized and thriving. Agent definitions, state export, evaluation formats, and policy expression are understood well enough to specify now. Nobody is proposing to freeze reasoning architectures or force labs to open their weights. The ask is narrower: the joints between the pieces should be public, specified, and testable, and the first-party path should be one path among several rather than the only executable one.

In reality, *some* will go all in on one particular model, vendor, or system and be successful. It's up to your organizational values, relationships, and risk appetite to decide if all in makes sense. Your team has to commit to those risks.

The five-part test to protect against lock-in

The practical version of this argument fits in a procurement requirement. Before committing to any agent platform, ask five questions and require demonstrations, not assurances:

1. ### Export the agents

   Can your agent and skill definitions be produced in a format another runtime can execute? Ask to see it run elsewhere.

2. ### Export the memory

   Can accumulated context and state be extracted with fidelity, or only as a transcript dump?

3. ### Export the evals

   Can your evaluation suites and historical results move with you, so a replacement model can be judged against the same bar?

4. ### Export the policies

   Can authorization rules and audit records be expressed in a form a second system can enforce, and can your auditors inspect them without vendor mediation?

5. ### Price the exit

   Have the vendor walk through, in writing, what a migration off their stack would require. The length of the silence is the height of the wall.

Then actually run the export, before signature, while you still have leverage. If a platform cannot pass these tests, you are moving in as a tenant, and the rent will be set after you have unpacked.

Enterprises still hold the advantage, because the substrate of the agent era is being poured right now and no vendor's position is locked. The system around the model is where your value will live for the next decade. Own it, keep it portable, and make every vendor compete for the right to run it.

Sources

- Gartner, "Gartner Predicts Over 40% of Agentic AI Projects Will Be Canceled by End of 2027," June 25, 2025
- Gartner, 2026 Hype Cycle for Agentic AI (deployment and intent figures as reported by IHL Services, June 2026)
- FinOps Foundation, State of FinOps 2026 (as reported by CIO.com, April 2026)
- PwC, 29th Global CEO Survey (as reported by CIO.com, April 2026)
- Deloitte inference cost analysis (as cited in Forbes Technology Council, July 2026)
- a16z, "AI Adoption by the Numbers," Kimberly Tan, 2026
- Menlo Ventures, "2025: The State of Generative AI in the Enterprise," December 2025
- Linux Foundation, "The Economic and Workforce Impacts of Open Source AI," 2026
- Kai Waehner, "Enterprise Agentic AI Landscape" and "Trusted Agentic AI Landscape," April and August 2026
- AvePoint, "Avoid AI Vendor Lock-In: A Multi-Model AI Strategy," 2026
- WorkOS, "Everything Your Team Needs to Know About MCP in 2026," March 2026
- Model Context Protocol Blog, "The 2026 MCP Roadmap," March 2026
- Reuters and industry reporting on the Builder.ai collapse, 2025