RESEARCH REPORT

# The Simplest Thing That Works: RAG, Fine-Tuning, or AI Agents?

*Teams keep asking "which AI technology should we use?" The better question is "which problem do we actually have?" A plain-language guide to choosing well, and to not paying for complexity you don't need.*

**SDTC Digital** · What We Think

September 2026 · Approx. 5,000 words

PART 1

# Content Metadata

| | |
|---|---|
| **Content type** | Research Report |
| **Title** | The Simplest Thing That Works: RAG, Fine-Tuning, or AI Agents? |
| **Slug** | rag-fine-tuning-ai-agents-architecture |
| **Meta title** | RAG vs Fine-Tuning vs AI Agents: A Research Report \| SDTC Digital |
| **Meta description** | RAG, fine-tuning, and AI agents solve three different problems. Our research report gives leaders a plain-language way to choose, and to avoid paying for complexity they don't need. |
| **Excerpt / card summary** | Teams keep asking "which AI technology should we use?" when the better question is "which problem do we actually have?" This report explains RAG, fine-tuning, and AI agents in plain language, what each really costs, and how to choose the simplest architecture that works. |
| **Hero image direction** | An editorial photograph of a well-organised workbench with three distinct tools laid out cleanly, shot from above in muted tones. The report's core idea is choosing the right tool for the job rather than the most impressive one. No robots, circuit imagery, or glowing effects. |
| **Card image direction** | A simple flat graphic of three doors of increasing size, in SDTC's brand palette, suggesting three paths and a choice. Clean and legible at thumbnail size beside the "Research Report" category label. |
| **Primary audience** | Founders, CTOs, CIOs, heads of engineering, product leaders, and SaaS operators facing build decisions about AI features and platforms. |
| **Search intent** | Informational and evaluative: "RAG vs fine-tuning," "RAG vs fine-tuning vs agents," "when to use AI agents," "enterprise AI architecture," "how to choose AI architecture." |
| **Suggested related articles** | 1) The Last Mile: Why AI Projects Fail on the Way to Production (Research Report). 2) The Adoption Gap: Why Enterprise AI Works in Some Companies and Stalls in Most (Research Report). 3) A Guide to preparing enterprise data for AI. 4) A Perspective on governing agentic AI. 5) A Field Note on measuring AI system quality in production. |

PART 2 · REPORT BODY

# The Simplest Thing That Works: RAG, Fine-Tuning, or AI Agents?

## Executive Summary

Somewhere in your organisation, a team is probably debating a question that sounds technical but is really a business decision: should we use RAG, fine-tune a model, or build an AI agent?

Stripped of jargon, the three options are simple. RAG, short for retrieval-augmented generation, lets an AI look things up in your company's own documents before it answers, like an open-book exam. Fine-tuning sends the model on a training course, permanently teaching it your domain's language, style, and rules. AI agents give the model hands as well as a voice: the ability to plan a task, use your systems, and carry work through to completion.

> **CENTRAL FINDING**
>
> Most organisations get this decision wrong not by picking the wrong technology, but by starting with technology at all. The pattern that separates well-run AI programmes is diagnosis before architecture: **wrong knowledge, use RAG; wrong behaviour, fine-tune; wrong outcome, build an agent.**

The market data shows where practice has settled. Menlo Ventures' enterprise research found 51% of enterprise AI deployments using RAG in production, against just 9% relying primarily on fine-tuning, because looking information up is cheaper, faster to ship, and easier to govern than retraining a model. Yet the same research record shows that combined approaches outperform either alone: a UC Berkeley, Microsoft, and Meta research collaboration (RAFT) demonstrated it formally, and one domain benchmark found fine-tuning lifted accuracy from 75% to 81% while the hybrid reached 86%.

The deeper finding is about discipline, not technology. Each step up the ladder, from a simple chatbot to RAG to fine-tuning to autonomous agents, adds capability and adds cost, complexity, and risk in roughly equal measure. Gartner attributes the abandonment of 60% of AI projects through 2026 to data that was never ready, which sits beneath all three approaches. The organisations that win choose the least complex architecture that solves the actual problem, build the data foundation first, and add autonomy only where the value clearly outweighs the risk. This report explains how to make that choice well.

## The Context

Two years ago, the architecture question barely existed for most companies. Teams bolted a chatbot onto a general-purpose model and called it an AI strategy. That era is over, for a practical reason: general models do not know your business. They have never read your policies, your contracts, your product documentation, or your customer history. Ask them about any of it and they will often answer anyway, confidently and wrongly, a failure known as hallucination, which happens because these models are prediction machines: when they cannot find an answer, they generate the most plausible-sounding one instead.

Three ways of closing that gap have matured into the standard toolkit.

**RAG** emerged from a 2020 research paper by Meta AI and has become the workhorse of enterprise AI. IBM's explanation uses a kitchen analogy that is hard to improve on: a general model is an amateur home cook, and RAG hands that cook a cookbook for your specific cuisine. The model stays the same; it simply gets to consult your trusted documents, in real time, before answering, and can cite which page the answer came from.

**Fine-tuning** is the cooking course instead of the cookbook. The model itself is retrained on your examples until your domain's terminology, formats, and habits are baked into its behaviour. Techniques with names like LoRA have made this far cheaper than it once was, but it remains a genuine training project: it needs curated example data, specialist skills, and repeat effort every time your knowledge changes.

**Agents** are the newest layer and the current centre of industry attention. An agent doesn't just answer a question; it pursues a goal: planning steps, calling other software, checking its own results, and escalating to a human when unsure. That shift, from saying to doing, is why agents excite leaders, and why they raise governance questions the other two approaches never did.

The market has voted with its deployments. Menlo Ventures' 2024 enterprise survey found 51% of production AI deployments using RAG and only 9% leaning primarily on fine-tuning. Meanwhile the industry conversation has moved from "which model?" to "which architecture?", and for good reason: models are replaced every few months, while architecture endures. As one CIO-focused analysis in our sources put it, the models you are obsessing over today will be obsolete within a year; your architecture will not.

## The Core Problem

The core problem is that organisations frame this as a technology choice when it is actually a diagnosis.

Walk into most AI architecture debates and the opening question is some version of "should we build agents?" or "should we fine-tune?" These questions cannot be answered as asked, because each technology solves a different failure. The practitioner rule quoted earlier deserves restating because it converts the debate into a diagnosis:

- If the system's answers are **out of date or missing your company's facts**, you have a knowledge problem. That is what RAG solves.
- If the system **knows enough but speaks or formats things wrongly**, misusing your terminology, breaking your output structures, missing your tone, you have a behaviour problem. That is what fine-tuning solves.
- If a correct answer **isn't actually the deliverable**, because something must get done, a ticket resolved, a record updated, a process completed, you have an outcome problem. That is what agents solve.

Misdiagnosis is expensive in predictable ways. The most common error in the source material is teams reaching for fine-tuning when their real complaint is stale knowledge, which RAG would fix at a fraction of the effort, and which fine-tuning cannot fix at all, because a fine-tuned model's knowledge is frozen at the moment of training and starts going stale immediately. The second most common error is jumping to agents because they are the exciting option, when the use case needed a reliable answer, not an autonomous actor, and the project inherits all of the governance burden of autonomy with none of the payoff.

The problem compounds because these three are not actually rivals. Production systems routinely combine them: a model fine-tuned for your domain's behaviour, fed current facts through RAG, orchestrated by an agent where action is required. Treating them as competing bets, pick one, back it, defend it, produces architectures shaped by internal politics rather than by the problem.

And beneath all three sits the layer that actually decides most outcomes. Gartner's February 2025 research attributes the projected abandonment of 60% of AI projects through 2026 to a single cause: data that was not ready for AI. RAG retrieving from a messy, contradictory document store returns messy, contradictory answers. Fine-tuning on poor examples bakes the errors in permanently. Agents acting on bad data act badly, at speed. The architecture debate is, too often, a debate about the upper floors of a building whose foundation has not been inspected.

## What Leaders Often Get Wrong

**Asking "which one?" instead of "what's broken?"** The framing error described above is the root of most others. The right opening question, as one architect in our sources puts it, is: what business problem are we solving, what level of intelligence does it need, and what is the simplest architecture that delivers the outcome? Sophistication is not a goal; it is a cost.

**Treating fine-tuning as a way to teach the model facts.** Fine-tuning changes how a model behaves, not what it knows about current events in your business. Retraining a model to fix a factual gap is, in the words of one analysis, a waste of capital: the fact will change again, and each change demands another training cycle, with data preparation, evaluation, and redeployment. Facts belong in a retrieval layer that can be updated in minutes. Fine-tuning is for the things that change slowly: terminology, reasoning style, output structure.

**Underestimating what RAG really involves.** RAG demos are famously easy; RAG in production is a data engineering discipline. Documents must be cleaned, split into sensible pieces (called chunking), converted into a searchable mathematical form (embeddings, stored in a vector database, which is a database that finds information by meaning rather than by keyword), kept fresh as sources change, and, critically, retrieved with the same access permissions as the original systems. A retrieval layer that ignores permissions will cheerfully surface documents to people who were never entitled to see them. The "RAG trap" named in our sources is assuming the technology compensates for poor source data. It amplifies it.

**Ignoring the economics until the bill arrives.** The three approaches have fundamentally different cost shapes, and leaders regularly discover this in the CFO's office. RAG is pay-as-you-go: cheap to start, but every single query carries the cost of the retrieved context, so the bill grows in a straight line with usage. Fine-tuning is an upfront investment: expensive to do once, then cheap per query, and it can let a smaller, faster model do the work of a big one. Agents are the most expensive of all per task, a single request can trigger dozens or hundreds of internal reasoning steps, each costing money, but they are judged on a different metric: cost per completed task versus the human labour they replace. Choosing an architecture without modelling these curves at your realistic volume is how "affordable" pilots become unaffordable products.

**Granting autonomy the use case never asked for.** One of the clearest frameworks in our research describes six levels of system maturity, from a simple chatbot, to RAG, to a single agent, to multi-agent systems, to autonomous workflows, to fully governed enterprise agents. Its point is not that higher is better. Every level up adds coordination overhead, latency, failure modes, and governance burden. The discipline the framework urges is appropriate autonomy with proportional control: add tools, memory, and independence only when the value clearly outweighs the risk. Not every AI feature needs an agent. Not every agent needs a team of agents.

**Forgetting that someone has to watch all of this.** A traditional application either works or visibly breaks. AI systems fail quietly: retrieval quality drifts, a model update changes behaviour, an agent starts looping, costs creep. Leaders budget for building and under-budget for observing, evaluating, and improving, which is where the long-term quality of these systems is actually determined.

## The Evidence and Patterns

### Where practice has settled, and why

The adoption data tells a clear story. Menlo Ventures found 51% of enterprise AI deployments using RAG in production versus 9% relying primarily on fine-tuning. The reasons recur across every source: RAG needs no training data, ships in weeks rather than months, updates instantly when documents change, and, decisively for regulated industries, can show its sources. Because the model's answer is grounded in retrieved documents, the system can cite exactly where a claim came from, an audit trail that a fine-tuned model, synthesising from its internal parameters, cannot easily provide. Keeping

sensitive knowledge in a governed store, rather than baked into model weights where it cannot be selectively deleted, also fits how data protection law actually works.

Real deployments show both the power and the honest difficulty. Notion's Q&A assistant is effectively a large-scale RAG system over customers' workspaces; the hard engineering problem was not retrieval but enforcing user permissions during retrieval. LinkedIn combined RAG with knowledge graphs for its support system and reported a 77.6% improvement in retrieval accuracy and a 28.6% reduction in median issue resolution time. The lesson in both: the differentiating work is data and access discipline, not the AI itself.

## Where fine-tuning earns its cost

Fine-tuning's 9% share does not mean it is losing; it means it is specialised. The evidence shows it winning in four situations. When output must be exactly right in structure, machine-readable formats, code, clinical documentation, fine-tuned models excel: a fine-tuned coding model (Cosine) and a fine-tuned text-to-SQL model (Distyl) have topped their respective public benchmarks. When volume is very high and knowledge is stable, the economics flip: one cost analysis in our sources models RAG's per-query context overhead at roughly $8,750 a month at 10 million queries and $87,500 at 100 million (illustrative modelling; verify against your own pricing), at which point a one-off training investment looks cheap. When speed matters, fine-tuned models skip the retrieval steps entirely. And when deep domain reasoning matters more than fresh facts, training beats retrieving: in a specialised agriculture benchmark, fine-tuning lifted accuracy from 75% to 81%.

## The strongest pattern: layers, not rivals

That same agriculture benchmark contains the report's most important number: the hybrid system, fine-tuning plus retrieval, reached 86%, beating both. This is not an isolated result. The RAFT research collaboration between UC Berkeley, Microsoft, and Meta showed formally that models explicitly trained to use retrieved context, including learning to ignore irrelevant retrieved documents, outperform either approach alone. The pattern has a memorable practitioner name: **fine-tune for form, RAG for knowledge.**

Production systems bear it out. Harvey, the legal AI company, reportedly fine-tuned on 10 billion tokens of case law to master legal reasoning and style, and still uses RAG to bring in current cases and updates (reported figures; verify before publication). Healthcare deployments fine-tune for clinical terminology and documentation standards, then retrieve the latest research and patient records. The architecture question, at maturity, stops being either/or.

## Agents: the frontier, with guardrails still being built

Agents show the widest gap between excitement and production readiness. The capability is real: an agent can plan, call software tools, check its own work, and complete multi-step processes end to end, which is why the sources describe the shift as moving from systems that provide a service to systems that complete work. But the same sources are unusually candid about failure modes: agents can wander into loops, burning money on repeated reasoning steps; their logic paths are harder to predict and

audit; and a single user request triggering 50 to 500 internal steps makes cost governance a first-class design problem. Gartner's related prediction, cited in our earlier research, that 40% of agentic AI projects will be cancelled by 2027, mostly on governance grounds, matches this picture. The emerging consensus design pattern wraps agents in guardrails: constrained tools, spending limits, audit logs, and human approval gates for consequential actions.

### The layer beneath all of it

Every source, independently, arrives at the same foundation. Gartner's 60%-abandonment figure for projects without AI-ready data has already been cited. The architecture-focused sources say it structurally: generative AI is a layered system, data at the bottom, then the model, then retrieval, then agents, then applications, and weaknesses at lower layers cannot be fixed at higher ones. Teams routinely build top-down, starting with the visible chatbot, and discover the foundation cannot hold it. Building bottom-up is slower to demo and much faster to finish.

## Strategic Implications

**Architecture, not model choice, is where durable advantage lives.** Models leapfrog each other every few months; committing your business logic to any one of them is renting a depreciating asset. The strategic asset is the architecture around the model: your data pipelines, retrieval layer, evaluation sets, and governance. The practical corollary from our sources: keep the model swappable behind an abstraction from day one. Model-agnosticism is cheap to design in at the beginning and expensive to retrofit once business logic is welded to one vendor's interface.

**The economics should drive sequencing, and they favour starting simple.** RAG's pay-as-you-go shape makes it the rational first move for almost every use case: weeks to pilot, no training data required, easy to change. Fine-tuning's investment shape makes it a second move, made from evidence: once a RAG system is live, its logs show precisely where the model underperforms and which query types dominate volume, which is exactly the information needed to decide whether training is worth it and what to train on. This phased path, retrieve first, observe, then train where volume and stability justify it, appears independently in multiple sources and is the closest thing this field has to a consensus playbook.

**For SaaS leaders, the architecture choice is a product decision.** Customers experience these choices directly: RAG-grounded features can show sources, which builds trust; fine-tuned features respond faster and in exactly the right format; agentic features complete work but demand visible guardrails to be trusted with it. The winning pattern in the industry examples, retail assistants that are fine-tuned for brand voice and RAG-grounded in live inventory, platforms where agents execute user intent using retrieval for context, is composition in service of a specific customer experience, not allegiance to one technique.

**Agents change the business case, and the risk profile, together.** An answer-giving system is measured in cost per query; an agent is measured in cost per completed task against the labour it

replaces, which is why agent ROI can dwarf the other approaches when the workflow genuinely suits autonomy. But autonomy that touches customers, money, or records carries operational and reputational risk that information retrieval never did. The strategic posture the evidence supports: pursue agents for well-bounded, high-volume workflows with clear success criteria, and treat unbounded autonomy as a research project, not a roadmap item.

**Budget for the watching, not just the building.** Whichever architecture you choose, the ongoing work, evaluating quality, monitoring drift and cost, refreshing indexes or retraining, is where systems live or die. Organisations that treat AI as a capability with an operating budget, rather than a project with an end date, are the ones whose second and tenth deployments compound.

## Technical Implications

**Every serious deployment converges on the same layered anatomy.** Across the source material, production systems repeat one shape: a security and access layer in front; an orchestration layer that routes requests; a knowledge and retrieval layer where RAG lives; the model layer itself, increasingly several models of different sizes; a tool and agent layer where action happens; guardrails that validate outputs before they reach users or systems; and observability underneath it all. Teams do not need every layer on day one. They need to know which layers their use case requires, and to leave the seams in place for the rest.

**Retrieval quality is an engineering discipline with known levers.** Basic RAG, embed the documents, fetch the closest matches, is a starting point, not an end state. The evidenced upgrade path: hybrid retrieval that combines meaning-based search with old-fashioned keyword matching (better for exact terms like product codes and clause numbers); reranking retrieved passages before use; checking retrieved context and re-querying when it is weak (so-called corrective RAG) for high-accuracy domains; and agentic retrieval, where the system decides what to look up and iterates, for genuinely complex questions. Each step adds cost and latency; the same simplest-thing-that-works rule applies inside RAG as above it.

**Fine-tune with modern efficiency, and with evaluation discipline.** Parameter-efficient techniques such as LoRA, which train a small add-on rather than the whole model, have brought fine-tuning within reach of ordinary teams and single-GPU budgets. The binding constraint is not compute but data and rigour: several hundred high-quality examples at minimum, a held-out test set defined before training, and quarterly re-evaluation thereafter, because fine-tuned models silently go stale as the domain moves. If you combine tuning with retrieval, note RAFT's lesson: models must be trained to use retrieved context, including when to distrust it, or the combination underperforms.

**Route work to the cheapest model that can do it.** Sending every request to the largest model is the most common cost mistake in the record. The emerging pattern is a gateway in front of multiple models, small ones for routine requests, large ones for hard reasoning, specialised ones for code or vision, with automatic failover when a provider stumbles. This one layer addresses cost, latency, vendor

lock-in, and resilience simultaneously, and it is what makes model-agnosticism real rather than aspirational.

**Agents need engineered boundaries, not behavioural hopes.** The reliable agent deployments in the sources share a design: an explicit list of tools the agent may use and nothing more; spending and step limits that halt loops; logged traces of every decision and tool call; validation before any consequential action; and human approval gates where stakes are high. Autonomy is granted in degrees, and it is granted by architecture, not by prompt.

**Observability must follow the request, end to end.** An AI request now crosses gateway, retrieval, model, tools, and guardrails, and a failure anywhere looks to the user like "the AI was wrong." Production systems need tracing that reconstructs each request's full journey, which documents were retrieved, which model ran, what it cost, what the agent did, connected to quality evaluation so teams can tell not just what happened but whether it was good. "Did the API return 200?" is no longer the question; "was the answer grounded, useful, and worth its cost?" is.

## Decision Framework

The framework that follows condenses the evidence into three steps a leadership team can run in a single working session.

### Step 1: Diagnose the failure, in one sentence.

Complete this sentence honestly: "The system we have (or the model we tested) fails because..."

- "...it doesn't know our information, or its answers are out of date." → **Knowledge problem. Start with RAG.**
- "...it knows enough, but the style, structure, or terminology of its output is wrong." → **Behaviour problem. Consider fine-tuning.**
- "...an answer isn't enough; work has to actually get done across systems." → **Outcome problem. Consider an agent.**

Most teams discover their problem is the first kind, which is why most production deployments are RAG. If your diagnosis is the second or third, ask once more whether better retrieval or better prompting would suffice; the honest answer is often yes.

### Step 2: Check the three quantities that change the answer.

- **How fast does the knowledge change?** Daily or weekly: retrieval is strongly favoured; retraining cannot keep up. Stable for months: fine-tuning becomes viable and eventually attractive.
- **How many queries, really?** At modest volume, RAG's per-query overhead is trivial and its flexibility wins. At very high volume, tens of millions of queries a month and beyond, the per-query cost of injected context compounds, and fine-tuning's one-off investment starts to pay for itself.

Model this with your own numbers before deciding; the crossover is real but its location depends on your pricing and traffic.

- **What can your team actually operate?** RAG demands data engineering: pipelines, indexes, freshness, permissions. Fine-tuning demands machine-learning discipline: labelled data, evaluation, retraining cycles. Agents demand both, plus governance. An architecture your team cannot operate will fail regardless of its merits on paper; many production failures blamed on model quality are infrastructure immaturity wearing a disguise.

### Step 3: Climb the autonomy ladder only as far as the problem requires.

Think of the options as rungs: plain model → RAG → fine-tuned (or both) → single agent → multiple agents → autonomous workflow. Two rules govern the climb. Take the lowest rung that solves the diagnosed problem, because every rung up adds cost, latency, failure modes, and governance burden. And earn each rung with evidence from the one below: deploy retrieval, learn from its logs where behaviour falls short, fine-tune if volume and stability justify it, and add agency only for workflows where a completed task, not an answer, is the deliverable and where guardrails can be genuinely enforced.

One further check applies before any of this: **is the data ready?** If the honest answer is no, the architecture debate is premature. Fix the foundation; it is the layer Gartner's abandonment statistic lives in.

## Risks and Failure Modes

**The perpetual retraining treadmill.** Fine-tuning chosen for fast-moving knowledge condemns the team to endless cycles of data preparation, training, evaluation, and redeployment, always behind reality. This is the most common architectural misdiagnosis in the record, and it is fully preventable at decision time.

**The confident nonsense problem.** A fine-tuned model asked about things outside its domain does not say "I don't know"; it generates plausible answers with full confidence. RAG reduces this risk by grounding answers in retrieved evidence, but only if retrieval is good: fed a weak or contradictory document store, the model synthesises noise into fluent, wrong output. Both failure modes are invisible in a demo and corrosive in production.

**Permissions leakage through retrieval.** A RAG layer that indexes everything and retrieves for everyone becomes a machine for surfacing documents to the wrong people. Access controls must travel with the data into the retrieval layer, the hardest and least glamorous part of the Notion-style deployments in our sources, and the part that decides whether the system is an asset or an incident.

**Agent drift and runaway cost.** An agent without step limits, spending caps, and constrained tools can loop, rack up hundreds of billed reasoning steps on a single request, or take an action nobody intended. The mitigation is architectural: sandboxes, budgets, audit trails, and human gates for consequential actions, designed in from the start, not bolted on after the first surprise.

**The invisible bill.** RAG's per-query overhead, agents' multiplied reasoning steps, and premium-model defaults each grow quietly with adoption. Programmes have been cancelled not because value was absent but because cost visibility was: the first itemised bill arrived as a shock. Cost monitoring per use case, and routing work to the smallest capable model, belong in the launch checklist, not the post-mortem.

**Vendor lock-in by a thousand prompts.** Every prompt, retrieval format, and workaround written against one provider's quirks is a small weld to that vendor. The models will change; if your evaluation sets and abstractions don't let you swap providers and re-test in days, you have surrendered your negotiating position and your upgrade path.

**Stale excellence.** A fine-tuned model that was excellent at launch degrades invisibly as the domain drifts away from its training snapshot; a retrieval index whose ingestion quietly breaks serves last quarter's truth indefinitely. Both demand scheduled re-evaluation against refreshed benchmarks, the unglamorous habit that separates systems that stay good from systems that were once good.

## Recommendations

**1. Run the diagnosis before any architecture debate.** One sentence: is the failure knowledge, behaviour, or outcome? Write it down and make every proposal answer to it. This single habit prevents the majority of expensive misdirections in the evidence base.

**2. Default to RAG, deliberately.** For most use cases it is the right first move: fastest to value, easiest to change, auditable by design. Treat that default as a decision, not a drift: scope the data engineering honestly, chunking, freshness, and above all permission-aware retrieval, because they are the actual project.

**3. Let production logs, not enthusiasm, make the fine-tuning case.** After a quarter of live RAG traffic, review where the model underperforms and what dominates volume. Fine-tune when three things line up: stable knowledge, high volume or strict format and latency needs, and several hundred quality examples you can actually produce. Then evaluate quarterly, forever.

**4. Add agency in degrees, with the guardrails built first.** Start with one agent, a constrained tool list, step and spend limits, full tracing, and human approval on consequential actions. Expand autonomy only as evidence accumulates that the agent earns it. If a workflow cannot be given clear success criteria and boundaries, it is not ready for an agent.

**5. Put a routing layer in front of your models from the start.** Route routine work to small, cheap models and reserve the largest for what genuinely needs them; keep a failover path. This is the highest-leverage cost and resilience decision available, and it makes switching providers an evaluation exercise rather than a rebuild.

**6. Fund the data foundation as the first-class workstream it is.** Clean sources, ownership, freshness, and access control beneath whichever architecture you choose. The 60%-abandonment

statistic is not about models; it is about this layer.

**7. Build evaluation and observability alongside the feature, not after it.** An owned evaluation set for quality, end-to-end tracing for behaviour, and per-use-case cost dashboards. You cannot improve, or defend, a system you cannot see.

**8. Keep the exits open.** Abstract the model provider, version your prompts and datasets, and rehearse a model swap once a year. Architecture is what remains valuable when the model underneath it changes, which it will.

## How SDTC Digital Thinks About This

SDTC Digital builds AI-powered products and platforms for a living, which means we sit inside this decision weekly, and we have earned our scepticism of every default answer, including our own.

Our position matches where the evidence landed. We treat RAG as the sensible first architecture for most enterprise problems, and we treat it with respect: in our experience the demo is the easy tenth of the work, and permission-aware retrieval over clean, fresh data is the hard nine-tenths that decides whether the system deserves production. We reach for fine-tuning when live traffic proves the case, when a stable domain, real volume, or strict output requirements make the investment arithmetic honest. And we are enthusiastic about agents in exactly one form: bounded ones, with engineered guardrails, full traces, and human gates where actions have consequences.

Above all, we hold the view that gives this report its title. The most sophisticated architecture is not an achievement; it is a cost that must justify itself. The achievement is the simplest system that solves the diagnosed problem, on a data foundation that can carry it, with the seams left in place to grow when the evidence says grow. That is what production-grade means to us, and it is the standard we build to.

## Conclusion

The industry has spent two years asking "RAG or fine-tuning or agents?" as if it were a contest with a winner. The evidence says it is a diagnosis with a sequence. The three approaches solve three different failures, knowledge, behaviour, outcome; they compose rather than compete; and the organisations getting real value choose the least complexity that works, prove each step with evidence from the last, and pour their scarce discipline into the layer none of the debate mentions: the data underneath.

The architecture question, answered well, is not really a question about AI. It is the oldest engineering question there is, what does the problem actually need?, asked freshly of powerful new tools. The teams that keep asking it that way will still have working systems when this year's favourite model is a footnote.

**Choose the simplest thing that works. Then make it excellent.**

*Sources referenced in this report include IBM's technical explainer on RAG and fine-tuning; Databricks' enterprise guidance on RAG, fine-tuning, and hybrid adoption; Actian's production cost analysis of RAG versus fine-tuning, citing Menlo Ventures' 2024 State of Generative AI in the Enterprise (51% RAG vs 9% fine-tuning in production) and the RAFT research from UC Berkeley, Microsoft, and Meta; Gartner's February 2025 research on AI-ready data (60% projected project abandonment through 2026); reported production examples including Notion's Q&A assistant, LinkedIn's RAG and knowledge-graph support system (+77.6% retrieval accuracy, –28.6% median resolution time), Harvey AI, Cosine (SWE-bench), and Distyl (BIRD-SQL); and practitioner architecture frameworks published by ITTech Pulse, GeekyAnts, Impressico, and several enterprise AI architects. Cost figures are illustrative modelling from the cited analyses and should be recalculated against current pricing; vendor-reported benchmark results and company-specific figures (including Harvey's training corpus) should be verified before publication.*