MARKET RESEARCH BRIEFING

Enterprise Software After AI

Build, Buy, or Let AI Build It?

How agentic AI is rewriting the economics of enterprise software — and why a 30% cut in the cost of producing software is a 10% cut in the cost of owning it.

An evidence briefing for CEOs, CIOs and CTOs, CFOs, COOs, chief digital and AI officers, business-unit leaders, enterprise-architecture and transformation leads, and investors in enterprise software. All figures are stated in US dollars with the year of the underlying data. Evidence is current as of September 2026.

1. Executive Summary

Two things happened to enterprise software at once, and most strategy documents conflate them. Generative and agentic AI made software dramatically cheaper and faster to *produce*. Separately, and more consequentially, AI is changing the economics of *owning, buying, consuming and operating* it. This briefing tests whether the first change is large enough to overturn the second. On the evidence available in September 2026 it is not, for most of the estate — but it clearly is for workflows serving large user populations and for platforms serving several adjacent processes, and the consumption layer is shifting underneath both.

What is already happening

- **Building is being chosen more often, but succeeding no more often.** 32% of organizations report deciding against buying off-the-shelf software and building it instead with coding agents; among AI high performers the figure is 48% (McKinsey, 2026, survey, n=1,719). Against this, the largest published comparison of AI deployment outcomes finds internally built tools reaching production about 33% of the time versus roughly 67% for externally sourced ones (MIT NANDA, 2025, survey). A decision to build is not a working system.

- **Coding is faster; delivery is not proportionally better.** A within-engineer study of 16,223 Microsoft engineers over 43 weeks found 40.5% more pull requests completed in high-usage weeks than in zero-usage weeks (95% CI 38.9–42.1%), at constant coding time (academic, 2026). Independent survey evidence finds AI adoption positively related to delivery throughput and *negatively* related to delivery stability (DORA, 2025, survey, n≈5,000). In the model in Section 6, a 30% cut in build effort plus 20% agentic leverage on the operating lifecycle converts to a **10.4% five-year TCO advantage over a traditional build** — real, and a third of the headline.
- **Where it is costed end to end, AI-assisted development does pay — conditionally.** The one study that models money rather than effort puts first-year return at 39% and three-year return at 727% for a 500-engineer organization, with payback at about eight months, and reproduces the greenfield-versus-legacy split independently: 35–40% on simple new work, about 10% or less on complex legacy code (DORA, 2026, analyst research). The return accrues to organizations that already have quality internal platforms, disciplined version control and AI-accessible data — which is the same condition every other finding here turns on.
- **The enterprise-level payoff is still missing.** 37% of organizations attribute any EBIT impact at all to AI, and only 6% report an impact of 5% or more — both essentially flat year over year — while 80% of the same respondents report improved *personal* productivity (McKinsey, 2026, survey). Individual productivity and enterprise productivity are not the same variable.
- **The strongest case for building is not cost per application; it is scale, portfolio leverage and control.** Seats are the decisive variable and they cut against buying: subscription scales with users and build cost does not, so past roughly 430 seats a build is cheaper, and at 4,200 seats it is cheaper by around $11m per application at list price. The larger effect is compounding: a second application built on the same internal platform inherits identity, the data model, the integration fabric, CI/CD, observability and the agent layer, so it costs $1.18m against $2.05m for the next commercial product — and each further SaaS vendor makes the estate *more* expensive to integrate, not less. Cumulative cost crosses over at barely more than one application. Buying is priced per application; building is priced per platform, and only the second application onward reveals the difference. Add direct control of model routing, data residency and audit. Every one of these requires a capability the enterprise must already have — and model and API spend is only 2.5% of five-year TCO, so control of it is an option on a future cost, not a present saving (*StratoFactory analysis*).
- **Software spending is still rising, but retention is cracking.** Worldwide software spend is forecast at $1,468bn in 2026, up 15.5% on 2025's $1,271bn (Gartner, 2026, analyst forecast), while 36% of purchased SaaS licenses go unused (Zylo, 2026, observed data). Median gross revenue retention across private B2B software companies fell to 84% in 2026 from 88%, and the 75th percentile from 95% to 91% (Benchmarkit via The SaaS CFO, 2026, observed data) — consistent with seat rationalization rather than logo loss.

What is emerging

- **Pricing is being re-based, not abandoned.** IDC forecasts that pure seat-based pricing will be obsolete by 2028, forcing about 70% of vendors to refactor around consumption, outcomes or capability (analyst forecast, 2025). Meanwhile Microsoft passed 30 million paid Copilot seats in FY2026 and Salesforce reported Agentforce annual recurring revenue above $1.5bn, up more than 240% (vendor-reported filings, 2026). Agent revenue is real and growing fast from a small base; seat revenue has not yet fallen. Gartner separately puts $234bn — about 20% of enterprise application SaaS spend by 2030 — as exposed to "agentic arbitrage" while forecasting agentic AI at roughly 30% of enterprise application software revenue by 2035 (analyst forecasts, 2026 and 2025): redistribution, not contraction.

What remains speculative

- Whether agents become the primary consumers of enterprise software at scale — only 17% of organizations have deployed any, though over 60% expect to within two years, and more than 40% of agentic projects are forecast canceled by end-2027 (Gartner, 2026 and 2025) — and whether the cost of owning AI-built software falls. The security pass rate of AI-generated code has been stuck at 56% across more than 100 models (Veracode, 2026, benchmark), while duplication is at record highs and refactoring is down about 70% since 2023 (GitClear, 2026, observed data).

Strategic implications

First, separate the three cost questions. Cheap software creation, cheap software ownership and cheap business outcomes are three different economics, linked far more weakly than vendor narratives imply. **Second, the unit of the decision is the estate, not the application.** Priced one application at a time, buying wins below roughly 430 seats and loses above it; priced across four adjacent workflows on a shared internal platform, the crossover falls to barely more than one application — but only where component reuse is actually realized, and the observed trend in AI-written code runs against it. **Third, the highest-return move in 2026–2028 is orchestration**: deploying agents across the existing estate, which requires no replacement decision and monetizes integration already paid for. **Fourth, the constraint is no longer engineering capacity but verification capacity** — review, testing, security, evaluation, audit — which is human and scarce. **Fifth, apply two tests before the cost model**: is this software a source of advantage or the price of parity, and does its value sit in code or in a maintained domain corpus? Cheap code is not cheap content, and buy-to-learn is still the cheapest way to find out whether a workflow is worth owning at all. **Sixth, procurement should stop asking "build or buy?" and start asking "which layer should we own?"** — and price the control that ownership brings over inference cost, data residency and audit as an option, not as a saving.

Forward-looking predictions

Five falsifiable calls, each with the indicator that would confirm or refute it, are set out in Section 10: seat pricing survives 2028 as a floor with consumption riders above it; the enterprise EBIT-impact figure stays below 50% through 2027; a large enterprise publicly reverses a build-instead-of-buy decision on maintenance grounds; model and API consumption becomes a separately disclosed budget line; and displacement appears as seat compression inside retained vendors rather than vendor replacement.

2. The Old Equation: Why Enterprises Buy

The build-versus-buy rule that governed enterprise IT for three decades was not a preference. It was arithmetic. A vendor amortizes one development effort across thousands of customers; an enterprise amortizes it across one. Where a process was standardized the vendor's unit economics were unassailable and the enterprise bought; where it was genuinely differentiating the enterprise built, and accepted the cost.

What made that arithmetic lopsided was never the cost of writing code. It was everything attached to it: an implementation methodology, a certified integration catalog, a security program, a compliance posture maintained across jurisdictions, 24/7 support, a documented upgrade path, a partner ecosystem, and domain logic representing tens of thousands of customer-years of edge cases. None of that is code. All of it is cost, and all of it is shared.

The scale that arithmetic produced is worth stating plainly. Worldwide software spending reached $1,271bn in 2025 and is forecast at $1,468bn in 2026, growing 15.5% — faster than IT services at 5.3% and faster than the overall IT market's 14.2% (Gartner, 2026, analyst forecast). Only data-center systems grow faster, at 62.5%, and that spend is AI infrastructure, not application software. Software is not a shrinking category.

At the level of the individual enterprise, the same market looks less disciplined. Median SaaS spend per employee is $9,455 and 36% of purchased licenses go unused; the same benchmark finds large enterprises adding applications far faster than they retire them, most SaaS spend controlled by business units rather than IT, and unmanaged expensed purchasing growing at triple-digit rates (Zylo, 2026, observed data). Buying is efficient at the level of the vendor's cost structure and wasteful at the level of the buyer's portfolio. Both statements have always been true simultaneously.

The internal alternative carried costs that were easy to underestimate at approval and impossible to escape afterwards. Panorama's 2026 study of 170 enterprise system implementations found a median duration of nine months, with more than a quarter over budget and about a quarter over schedule (survey, 2026). Technical debt compounds it: CIOs at large firms have put debt at 20–40% of the value of their entire technology estate, with 10–20% of new-product budget diverted to servicing it (McKinsey, 2020, survey — dated, used only as a pre-AI baseline).

The vendor proposition, stated on its strongest terms, is therefore this: *you are not buying software, you are buying the amortization of every non-code cost of software across a customer base you could never assemble yourself.* Any argument that AI overturns the build/buy equilibrium has to show that AI attacks that amortization, not merely the cost of typing code. Section 3 examines what AI actually changed, and Section 4 examines what it did not.

3. The New Equation: What AI Actually Changed in Software Production

The production side of the equation has genuinely moved, and the best evidence for it is also the most carefully constructed. A within-engineer dose-response study of 16,223 software engineers at Microsoft's Cloud and AI organization, observed over 43 weeks from February to December 2025, found that engineers completed approximately 40.5% more pull requests in their highest AI-usage weeks than in their zero-usage weeks, with a 95% confidence interval of 38.9% to 42.1%, holding coding time and browser time constant (academic, 2026). The gradient was monotonic with diminishing returns — 21.0% at low usage, 39.4% at moderate, 40.5% at high — and survived seven falsification tests, including a placebo treatment using a non-coding assistant, which showed a flat gradient of ±3%.

Two qualifications matter more than the headline. The authors state explicitly that the estimate is an *efficiency* effect at constant coding time, not a total productivity effect, and they place the study on the associational rung of causal inference, noting that only a randomized experiment could close the gap. And the unit of measurement is the pull request — an activity count, not a value measure.

The randomized evidence, and its own revision

The best-known randomized result points the other way. METR's 2025 trial gave 16 experienced open-source maintainers 246 real issues from repositories they knew intimately, and found them **19% slower** with AI available, having predicted they would be 24% faster and having believed afterwards that they had been 20% faster (academic RCT, 2025). That finding has been quoted more widely than any other in this literature.

It should now be quoted with its correction. In February 2026 METR published a redesign of the experiment, reporting preliminary figures of −18% for original participants (CI −38% to +9%) and −4% for new developers (CI −15% to +9%), and disclosing a severe selection problem: between 30% and 50% of developers withheld tasks they expected AI to complete quickly, and others declined to participate at all rather than work without AI. METR's own current position is that AI had *likely* accelerated developer productivity by early 2026, and that its data provide only weak evidence about the magnitude (academic, 2026). The headline "AI makes developers slower" is no longer the position of the organization that produced it.

How wide the range actually is

No single number represents this literature, and quoting one is the most common error in the debate. A 2026 review of the agentic software-development literature places the spread at **13.6% to 55.8% time savings across controlled studies**, with Microsoft field data at 12.9–21.8% more pull requests per week and the Accenture study at 7.5–8.7% (academic review, 2026). That Microsoft range and the 40.5% above are the same company measured two ways — average weekly output against high-usage weeks compared with the same engineer's zero-usage weeks — and the gap between them is a good measure of how much a specification choice can move a headline. The same review records the capability shift underneath it: issue-resolution on the SWE-bench Verified benchmark rose from 1.96% in October 2023 to about 78% by April 2026. Its own conclusion is not that productivity is solved but that human review has become the bottleneck — with agents "producing ten plausible patches per hour", tooling for high-throughput review is the missing piece, and it flags an explicit technical-debt hypothesis that AI-assisted programming may reduce experienced-developer productivity by inflating maintenance burden.

The largest independent measurement is the most useful, because it decomposes the range instead of averaging it. Stanford's software-engineering productivity program scores commits from roughly 100,000 developers across hundreds of companies using a model trained on expert-panel ratings rather than counting commits. Its own summary of the period to 2025 is a **modest average lift, below the hype, real but uneven**, with an inflection from around December 2025. The decomposition reported from that dataset is the part that matters for a business case. The gain is largest on **greenfield, low-complexity work — of the order of 30–35%** — and smallest on **complex changes to existing codebases, where it falls to single digits and some teams record net decreases**; it is materially weaker in less common programming languages; and across a set of matched teams the **median gain was about 10%**. One case in the dataset recorded a 14% rise in pull requests with no measurable rise in effective output, the increase absorbed by rework. A companion 2026 paper is titled *AI Writes Faster Than Humans Can Review* (academic, 2025–2026; breakdown via secondary reporting). At the optimistic end, McKinsey describes agentic delivery producing "threefold to fivefold improvements in productivity, with a 60 percent reduction in team size" — a consultancy claim with no disclosed method, and one whose own article concedes that freed capacity is typically reinvested in road maps rather than banked (analyst, 2026). The dispersion has a pattern, and it is the same one Stanford decomposes: METR measured expert developers on codebases they had written, where the marginal value of a suggestion is lowest, and the Microsoft study a large heterogeneous population on unfamiliar and routine work, where it is highest. **AI's production gain is real, concentrates in new, simple and unfamiliar work, and shrinks toward zero on complex changes to mature codebases.** The 30% build-effort reduction used in Section 6 is therefore a *greenfield, mid-complexity* figure — defensible for the new application modeled there, and roughly double what the same evidence supports for extending an existing system.

The one study that costed it end to end

Everything above measures effort, not money. One study crosses that line, and it is the most directly relevant piece of evidence in this briefing. DORA's May 2026 analysis models the return on AI-assisted development for a 500-person engineering organization and reports a **first-year return of 39% — $11.6m of value against $8.4m of investment, with payback at about eight months — and a three-year average return of 727%** (analyst research, 2026). Those are large numbers and they are not vendor marketing; they come from the same program whose delivery metrics this section has already used against the optimists.

Three things in it matter as much as the headline. First, it independently reproduces the greenfield-brownfield split: **35–40% on simple greenfield tasks and about 10% or less on complex legacy code** — arrived at separately from the Stanford dataset and agreeing with it closely. Second, it prices the instability that Section 4 describes qualitatively: a change-failure rate rising from 5% to 6% costs the modeled organization **$344,000**, an explicit debit against the gains. Third, it identifies a **J-curve** — a genuine productivity dip during rollout, driven by learning, review overhead and process adaptation, which it argues should be budgeted for as a tuition cost rather than treated as failure. Its conclusion is not that the tools pay; it is that "the greatest returns come not from the tools themselves but from a strategic focus on the underlying organizational system", with returns realized through reinvested capacity rather than headcount reduction, which it explicitly discourages.

This is the strongest evidence in the briefing that AI-assisted development pays, and it should be weighted accordingly. It is also conditional in exactly the way the rest of the evidence is: the return accrues to organizations that already have quality internal platforms, disciplined version control and AI-accessible data. The same finding read from the other end says an organization without those foundations should not expect 39%.

What the practitioners report

Survey evidence is directionally consistent and quantitatively softer. Stack Overflow's survey of 49,000+ developers found 80% using AI tools, trust in output accuracy down to 29% from 40%, and 66% spending *more* time fixing code that is "almost right, but not quite" (2025): adoption and confidence moving in opposite directions. Vendor-published measurement is more favorable and should be weighted accordingly — GitHub's study with Accenture reported 8.69% more pull requests per developer and a 15% higher merge rate, sample size undisclosed (**vendor-reported**, 2024).

The critical distinction is between the developer and the delivery system, and DORA's 2025 study of nearly 5,000 technology professionals is the cleanest statement of it available. 90% use AI at work and over 80% report higher productivity, while 30% have little or no trust in AI-generated code. AI adoption showed a **positive** relationship with software-delivery throughput and product performance — a reversal from 2024 — and a **negative** relationship with delivery stability. DORA's interpretation is that AI amplifies whatever the organization already is: where automated testing, version control and fast feedback exist, higher change volume converts into delivery; where they do not, it converts into instability.

This is why benchmark gains have not shown up as enterprise savings. 56% of organizations report cost decreases in software engineering (Stanford HAI, 2026, aggregated survey) — a directional self-report with no stated magnitude — while only 37% attribute any EBIT impact to AI at all (McKinsey, 2026, survey). Gartner reads the same period as AI "fueling the demand for more software engineers, not fewer", even while forecasting that 60% of organizations adopt four- to five-person engineering teams by 2029, from 15% in 2026 (analyst forecast, 2026). Faster code has so far produced more software, not less spending.

4. The Productivity Paradox: Cheap Code, Expensive Software

The proposition this section tests is precise: **AI lowers the marginal cost of creating software faster than it lowers the total cost of owning it.** Ownership costs are the ones AI does not remove, and several of them AI appears to increase.

Verification does not scale with generation

The most direct evidence is security. Veracode's July 2026 benchmark, covering more than 100 models across four testing snapshots, found a **security pass rate of 56% — virtually unchanged year on year** — against a syntax pass rate near 100%. Models now write code that compiles almost every time and code that is secure barely half the time. The gap is wider between languages than between models: Python passes at 63% and Java at 30%, while the spread across model families is narrow, and neither model size nor coding specialization improves the result. Reasoning models do better than non-reasoning ones, at 56% against 51%, which the report reads as extended reasoning acting as a form of internal review. Generation capability improved substantially over the period; security did not follow it.

Maintainability shows the same shape. GitClear's analysis of 623 million code changes from 2023 to 2026 found block duplication up 81% on 2023 and at the highest level in its record, while refactoring activity fell by about 70%. Five further structural signals it tracks — copy-paste within commits, cross-file reuse, moved code, error-masking constructs and short-horizon churn — all moved the same way over the same period; the underlying figures are in the published study (observed data, 2026). The study is correlational: it observes repositories over a period in which AI adoption rose, and does not isolate AI as the cause. But its direction is corroborated by the practitioner surveys — 66% of developers spending more time fixing near-miss output is the same phenomenon measured from the other end (Stack Overflow, 2025, survey).

The costs that do not move

Several ownership costs are untouched by cheaper code generation. Integration cost is a function of the number and quality of interfaces, not of how the code behind them was written. Compliance cost is a function of jurisdiction: high-risk obligations under the EU AI Act attach from 2 August 2026, requiring deployers to assign human oversight and retain automatically generated logs for six months, with penalties up to €15m or 3% of global turnover — though parts may be deferred to 2027–2028 (Holland & Knight analysis of Regulation (EU) 2024/1689, April 2026). Incident cost is rising: the global average breach costs $4.99m, AI-enabled breaches about $6m, one in four malicious breaches is now AI-enabled, and compromised APIs, applications and plug-ins account for 27% of AI-related breach causes (IBM, 2026, observed study).

Then there are the costs AI creates. Model and API consumption is a new operating line whose unit price is collapsing and whose total is not. Inference prices fell between 9× and 900× per year depending on task between 2023 and 2025; GPT-4-level performance on one benchmark fell from $37.50 to $0.12 per million tokens in 21 months (Epoch AI, 2025, observed data). Yet Gartner forecasts that **AI coding costs will overtake the average developer's salary by 2028**, because agentic consumption grows faster than unit prices fall (analyst forecast, 2026). Enterprises already feel it: 78% of IT leaders report unexpected charges from consumption or AI pricing, and a majority have cut projects because of unplanned SaaS cost increases (Zylo, 2026, survey).

Agent supervision is a further new line. Organizations now manage an average of 109 machine identities for every human identity and expect AI-agent identities to grow 85% over twelve months against 56% for human identities, while most cannot state what those agents may access or when their permissions are revoked (Palo Alto Networks, 2026, survey). Every autonomous actor added to an estate adds an identity to govern, a log to retain, an evaluation to run and a failure mode to own.

The arithmetic of the paradox

The picture is coherent. Code generation — the line AI attacks most directly — is a minority of an enterprise application's lifetime cost. Verification, integration, compliance, maintenance, support, change management and incident exposure are the majority. AI moves them unevenly: it leaves integration, compliance and incident exposure essentially flat, adds two entirely new lines in inference and agent supervision, and does cut labor on maintenance and testing — which, as Section 6 shows, is where most of the realizable saving actually sits, and also where the quality evidence says the work will grow. That is why a 30–40% gain in coding efficiency does not produce a 30–40% cut in the cost of ownership, and why 37% of enterprises report zero EBIT impact while 80% of their people report being more productive. **The binding constraint has moved from the capacity to write software to the capacity to verify it** — and verification capacity is human, scarce, and growing linearly.

5. Build, Buy, Blend, Orchestrate

The four-option framing replaces a binary that no longer describes what enterprises actually do. **BUILD** means owning the application end to end. **BUY** means licensing a commercial product and configuring it. **BLEND** means licensing commercial infrastructure and owning the proprietary workflow, data model, business logic and agents that sit on it. **ORCHESTRATE** means changing nothing about the application estate and deploying agents across it — automating the work rather than replacing the software that work runs through.

The fourth option is the one the current debate keeps missing, and the one the evidence most supports. It requires no migration, no re-platforming and no vendor exit; it monetizes integration investment already made; its cost scales with usage rather than headcount or scope; and it is most consistent with the finding that AI's measured wins concentrate in back-office automation rather than in customer-facing systems (MIT NANDA, 2025, survey).

Dimension	BUILD	BUY	BLEND	ORCHESTRATE
Upfront cost	High — 4,800 eng-hrs traditional; ~3,360 AI-assisted	Moderate — implementation ≈0.85× year-1 subscription	Highest — subscription plus extension build	Lowest — no replacement, no migration
Time to value	9–18 months	3–9 months	6–12 months	6–12 weeks per process
Customization	Unlimited	Bounded by configuration	Unlimited above the platform	None to the application; full to the workflow
Differentiation	Maximum, if the process is differentiating	None — same product as competitors	High, in the layer you own	High, and invisible to competitors
Source of advantage	The process and data, not the code — building alone differentiates nothing	Parity, efficiently bought; advantage must come from elsewhere	Advantage above a parity substrate	Advantage in how work is executed, not in what is licensed

Dimension	BUILD	BUY	BLEND	ORCHESTRATE
Where the value sits	Suits code-heavy value: forms, queues, workflows, dashboards	Essential where value is a maintained corpus — legal, tax, payroll, trade compliance	Vendor corpus underneath, your logic on top	Neither — it monetizes what is already there
Convergence risk	None	High if the vendor ships the agent and every competitor runs the same process	Low — you own the agent and the process	Low
Security posture	Owned; 44% of AI-generated code fails security tests (Veracode, 2026)	Vendor-certified; shared responsibility	Split, and therefore the hardest to assure	Agent identity and permission risk (109 machine IDs per human, 2026)
Compliance burden	Fully owned, per jurisdiction	Vendor-maintained	Vendor for platform; owned for extensions	Deployer obligations under the EU AI Act from Aug 2026
Integration effort	Highest — every interface is yours	Moderate — catalogd connectors	High — two directions to maintain	Reuses integration already paid for
Maintenance	18–18.5% of delivered labor per year, less 20% agentic leverage on hours (modeled)	Absorbed in subscription	Both, on different clocks	Agent evaluation and drift, not code maintenance
Scalability of cost	Flat in users; steep in scope	Linear in seats	Linear in seats plus flat in scope	Linear in tasks and tokens
Vendor dependence	None on application; high on models	High	Moderate — platform substitutable in principle	Moderate — model and platform dependence
IP and data control	Full	Limited; extraction rights must be negotiated	Full above the platform	Full — data never leaves the estate
AI capability	Whatever you build and evaluate	Whatever the vendor ships	Vendor floor plus your own agents	Your own agents across every application
Five-year TCO (modeled, 400 users)	$1.96m AI-assisted / $2.18m traditional	$1.87m	$2.70m	Not modeled — incremental to the estate
Portfolio leverage (reuse across applications)	High — marginal app ~$1.18m vs $1.96m standalone, if reuse is enforced	None — each vendor is its own stack; marginal app ~$2.05m and rising	High above the platform; between the other two from the second application onward	Very high — one agent layer serves the whole estate
Inference cost control (FinOps)	Full — routing, model choice, caching, context budgets	None — vendor's model at vendor's markup and policy	Full for own agents; none for embedded vendor AI	Full — you own every token the agents spend
Data residency, tenancy, audit control	Full; no subprocessor chain	Vendor-defined; subprocessor chain to inventory	Split; the hardest boundary to evidence	Full — data stays in the estate

Dimension	BUILD	BUY	BLEND	ORCHESTRATE
Regulatory role (EU AI Act, high-risk)	Provider — conformity assessment, documentation, registration	Deployer — oversight and six-month log retention	Provider for extensions; deployer for the platform	Deployer, plus provider for any agent you build
Reversibility	Low — sunk build, owned maintenance	Moderate if integration was kept shallow	Moderate — platform substitutable in principle	High — agents can be switched off
Organizational capability required	Highest — internal builds reach production ~33% of the time (MIT NANDA, 2025)	Lowest	High	Moderate, but governance-intensive

Figure 1. Build, Buy, Blend, Orchestrate compared across twenty-two decision dimensions. Thresholds are StratoFactory analysis, calibrated against the TCO model in Section 6; they are decision heuristics, not measured market values.

Two tests that precede the four options

Cost is the third question, not the first. Two prior tests decide which of the four options is even eligible, and both are more discriminating than the TCO model.

The first is the advantage test: is this software a source of advantage, or the price of parity? The objection to buying is real and usually stated as: you cannot differentiate on a product your competitors also license. That is true, and for most of the estate beside the point. Packaged software was never sold as a source of advantage; it was sold as the cheapest route to *parity* in capabilities that are necessary but not distinguishing. The question is not whether the software differentiates — almost none of it does — but whether the *process running through it* is one where being different is worth paying for. The record says enterprises judge that badly. In one survey of 1,719 organizations, nearly nine in ten report regular use of AI in at least one business function and 37% attribute any EBIT impact to it; within the same respondents, 73% of high performers had fundamentally redesigned their workflows against 25% of everyone else (McKinsey, 2026, survey). **Outcome variance on identical technology is enormous, which is evidence that the technology is not where the variance comes from.** An application is not differentiated because it was built internally — a bespoke expense tool is a worse commodity, not an advantage. The honest use of this test is subtractive: it should disqualify most candidates for building, not license them.

The second is the code-or-corpus test: where does the product's value actually sit? Section 4 established that AI does not make software cheap to own. This test establishes something separate: **AI does not create domain content.** Where a product's value is mostly code — a form, a queue, a workflow, a dashboard, a report — an agent can now reproduce a workable version of it, and the build/buy line has genuinely moved. Where the value is mostly a maintained corpus, the line has not moved at all, and may have moved toward buying. Contract and legal technology, tax engines, payroll and statutory reporting, trade compliance and sanctions screening, clinical coding, e-invoicing mandates: what the enterprise is paying for is a clause and template library, jurisdictional rules maintained against changing law, filing and authority integrations, and thousands of customer-years of edge cases — in a domain where the enterprise employs a dozen specialists and the vendor employs hundreds. An agent can write the application. It cannot supply the corpus, maintain it against next quarter's statute, or carry the liability for getting it wrong. **Cheap code is not cheap content**, and non-core specialist domains are precisely where buying is strongest, not weakest.

There is one caveat that cuts the other way, and it should be priced at signature rather than discovered at renewal. The deeper the domain, the stronger the lock-in: a legal or tax platform ends up holding the corpus — your contracts, your positions, your history — in its own schema, and becomes a system of record you did not intend to buy. Buy for domain depth, and negotiate data extraction hardest exactly where the domain is deepest.

Where each option becomes attractive

Given those two tests, the cost thresholds fall out quickly, and Figure 1 carries the full comparison. **BUY** where the process is standardized, the regulatory or certification burden is high, the application is a system of record, or the value sits in a corpus — and note that on cost alone buying wins at *small* seat counts, below roughly 430 users in the Section 6 model, not large ones, because subscription scales with seats and build cost does not. **BUILD** where the workflow is genuinely differentiating, the enterprise holds data no vendor can access, several adjacent workflows can share one platform, the seat count is large, control of inference cost or data residency is part of the requirement, and the organization can verify what it builds; the binding condition is the last one, which is what the 33%-versus-67% production-deployment gap measures (MIT NANDA, 2025, survey). **BLEND** where the advantage test passes but the corpus test fails: commercial substrate, proprietary workflow, agents and data. **ORCHESTRATE** wherever the estate is already integrated and the work passing through it is repetitive, rule-bounded and high-volume — the only option requiring no replacement decision, and therefore the only one that is cheap to reverse. Its risks are governance risks rather than capital risks: identity sprawl, permission inheritance, log retention and evaluation, all cost lines rather than write-offs.

Sequencing beats a single decision. The four options are usually presented as a choice made once, at the point of demand. They are better used as a sequence. Buying is the cheapest way to discover whether a workflow is worth owning at all: for the price of a subscription, and in one or two renewal cycles, a commercial product answers questions a business case cannot — how many people really use it, which fifth of the functionality carries the value, where the process genuinely differs from the industry pattern, and whether the demand persists. Building first commits capital before any of that is known. This explains the 32% build-instead-of-buy figure (McKinsey, 2026, survey) better than "AI made building cheap": enterprises know which workflows are worth owning because they rented them first. The option holds its value only if it is bought deliberately — shallow integration, contracted data-extraction rights, a defined exit — because the switching costs that protect vendors also trap buyers who integrated deeply into a product they meant to leave. Buy to learn; build to scale; and decide which you are doing before signing.

The organizational threshold

One dimension in Figure 1 dominates the rest and is systematically underweighted in business cases. Every option except BUY requires a capability most enterprises do not have and cannot buy quickly: the ability to review, test, secure, evaluate and operate software a machine wrote. An organization that lacks it should treat BUILD as unavailable whatever the cost model says — the practical meaning of the 33% internal-build deployment rate.

6. The Economics: An Original TCO Model

This section models one representative decision under four options. It is a **constructed scenario, not a case study**, and every input is either sourced or declared as a StratoFactory estimate. The scenario is a mid-complexity operational workflow application — order exception handling and supplier collaboration — serving **400 business users** over a **five-year life** in a US cost base, in 2026 US dollars.

Assumptions (all inputs visible). Fully loaded engineer cost **$196,000 per year** — BLS median software-developer wage of $135,980 (May 2025) uplifted 1.44× for employer taxes, benefits, tooling, facilities and management overhead (*StratoFactory estimate*); 1,700 productive hours per engineer-year. Traditional build effort **4,800 engineer-hours**, plus 1,400 hours of QA, 900 hours of integration and 500 hours of security work (*StratoFactory analysis*). AI-assisted build-effort reduction **30%** — a *greenfield* figure at the optimistic end for mid-complexity work, since the measured greenfield range is 30–35% on low-complexity tasks and 10–15% on high-complexity ones, and 5–20% on brownfield work (*StratoFactory estimate*, deliberately generous to the build case). At 20% the AI-assisted build is 5.0% cheaper than a traditional one rather than 10.4%; at 12%, mid-way through the brownfield range, it is 0.8% cheaper and $295,000 worse than buying. This single assumption decides whether extending an existing system should be built at all. Verification uplift on AI-assisted output **+16%** of remaining engineering hours, derived from the 66% of developers reporting more time fixing near-miss output and the 44% security-failure rate (*StratoFactory estimate*). Annual maintenance **18% of delivered labor for traditional builds, 18.5% for AI-assisted**, the uplift derived from the observed collapse in refactoring and rise in duplication (*StratoFactory estimate*). Agentic leverage on the operating lifecycle **20%** — the reduction in QA and maintenance *labor hours* once agents do test generation, defect fixing, dependency upgrades and migration work, set below the 12.9–21.8% field range measured for authoring work (*StratoFactory estimate*); token spend per remaining maintenance hour rises 35%. Model and API spend **$16,000 per engineer-year during build and $5,000 during maintenance**, set at roughly 8% of loaded salary in 2026 and anchored between the observed collapse in inference unit prices and Gartner's forecast that AI coding cost overtakes developer salary by 2028 (*StratoFactory estimate*). SaaS list price **$38 per user per month** with 4% annual escalation; implementation at 0.85× first-year subscription. Integration $145,000–$175,000 depending on option. Downtime and incident provisions $22,000–$40,000 per year by option, reflecting the negative AI-stability relationship. Governance, model evaluation and agent supervision are costed only where agents are operated. **The model contains no J-curve**: it assumes AI benefits accrue evenly from the first month, where the evidence describes a productivity dip during rollout. That omission makes the model optimistic toward both AI-assisted options.

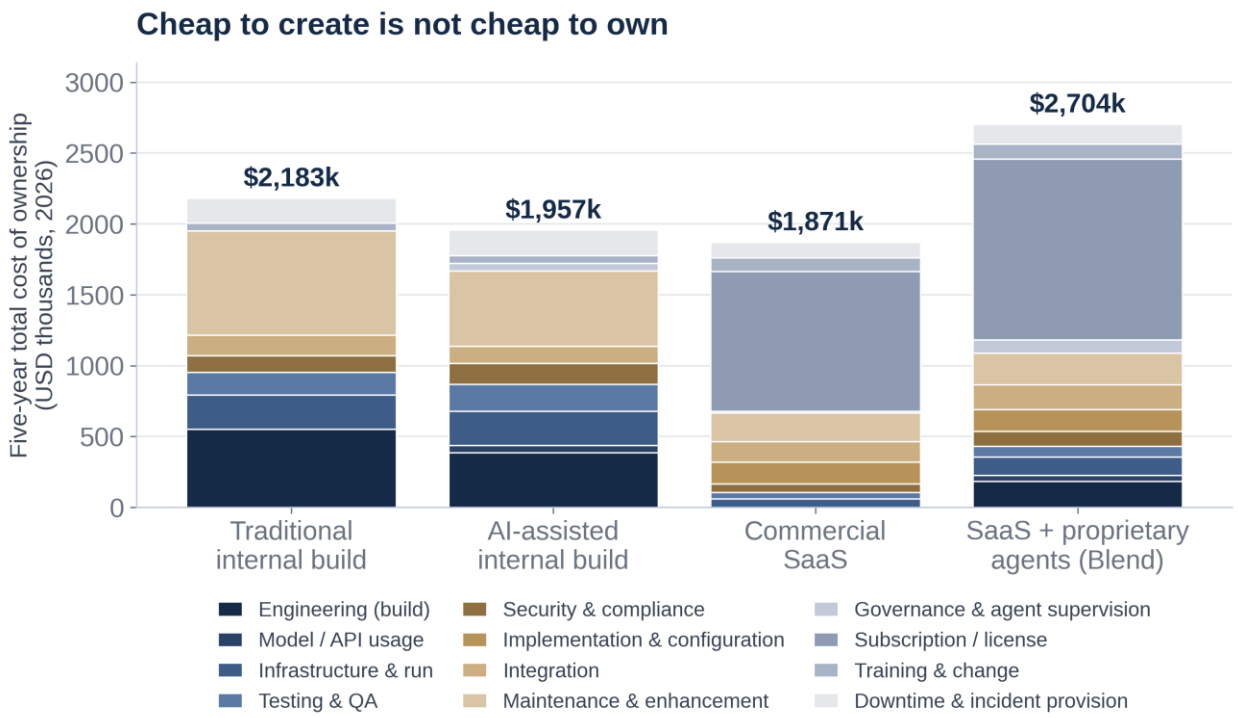

*Figure 2 (StratoFactory analysis). Five-year total cost of ownership by delivery model for the constructed scenario described above, in thousands of 2026 US dollars, for a **single application**. Cost lines are stacked in the order listed in the legend. Figure 4 extends the same model to a portfolio. This is a model output, not market data; all inputs are stated in the assumptions box.*

What the model says

In the base case, **commercial SaaS is cheapest at $1.87m over five years**, followed by the AI-assisted internal build at $1.96m, the traditional build at $2.18m and the blend at $2.70m — $936, $978, $1,091 and $1,352 per user-year respectively.

The most important number is not the ranking. It is the gap between two percentages. **AI assistance cuts the build-engineering line by 30%, from $553,000 to $387,000, and cuts the five-year total cost of ownership by 10.4%.** Three effects are modeled separately. Construction falls $166,000 and integration a further $22,000. Against that, model and API consumption, governance and agent supervision, testing, security and a higher incident provision add back $169,000 — every one of them a cost that AI creates or enlarges. Those two effects almost cancel: on the build phase alone, the AI-assisted option is **1.9%** cheaper than the traditional one. What produces a real saving is the third effect: agents also work the *operating* lifecycle. Maintenance falls $208,000, of which $132,000 comes from the assumed 20% agentic leverage on maintenance hours and $75,000 from a smaller delivered code base; QA labor falls a further $48,000. **Operating-lifecycle leverage contributes $185,000 of the $226,000 total saving — 82% of it.** This is the paradox stated arithmetically: *a 30% reduction in the cost of producing software is a 10% reduction in the cost of owning it, and four-fifths of even that comes from the operating years rather than the build.*

The second observation is that the blend is the most expensive option and still frequently the right answer. It carries a subscription and an engineering line and pays for governance twice. What it buys is control of the differentiating layer without ownership of the commodity layer — an option value the model cannot price.

What actually moves the build-versus-buy decision

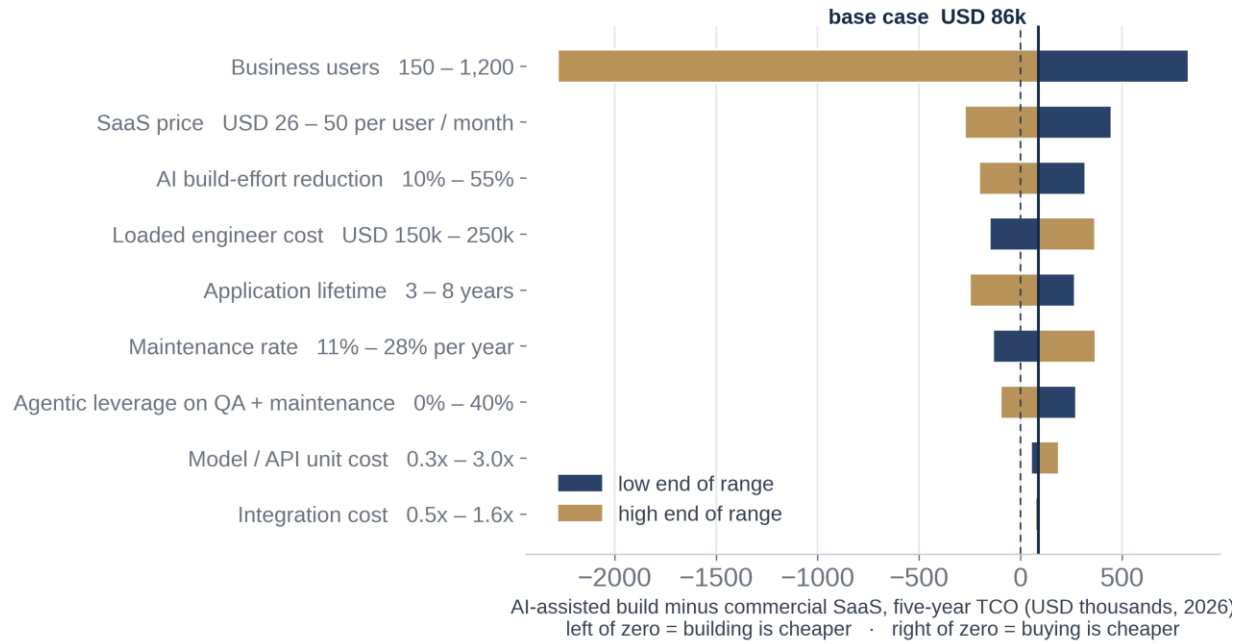

Figure 3 (StratoFactory analysis). Sensitivity of the build-versus-buy margin. Each bar shows the five-year TCO of the AI-assisted internal build minus that of commercial SaaS as one input is moved across the stated range, all others held at base. Base case is $86k in favor of buying. Method: one-at-a-time deterministic sensitivity on the model in Figure 2; no correlation between inputs is modeled, and the ranges are judgmental rather than distributional.

What actually moves the decision

The sensitivity analysis produces an uncomfortable result for the prevailing narrative. **The dominant variable is not AI capability. It is seat count.** Moving from 150 to 1,200 users swings the margin from $825k in favor of buying to $2,280k in favor of building, because subscription scales with seats while build cost does not. SaaS unit price comes next ($447k to −$275k), then a tight cluster of near-equal levers — AI build-effort reduction across a 10–55% range ($318k to −$205k), loaded engineer cost, application lifetime and maintenance rate — with agentic operating leverage a little behind. **Model and API unit cost, the variable most discussed, is the second least influential input tested**, moving the margin $134k across a tenfold range; integration cost matters least. Two of the top three levers are commercial terms, not technology.

Three thresholds follow, all of them **per application**. The seat threshold runs the opposite way to intuition, because subscription cost scales with users and build cost does not: **below roughly 430 users an AI-assisted internal build is more expensive than buying at $38 per user per month, and above it the build is cheaper** — by $506,000 at 600 users and by $1.69m at 1,000. For a traditional build the same crossover sits near 505 users. Second, the **lifetime threshold is about 5.8 years**: below it buying wins, above it maintenance on the build is outweighed by accumulated subscription. Third, agentic leverage on the operating lifecycle decides the sign of the base case on its own: at zero leverage the build is $270,000 worse than buying, at 40% it is $99,000 better. Two cautions attach to the seat threshold. It assumes list pricing, and enterprise agreements discount steeply at four-figure seat counts, so the real crossover sits higher than $38 per user implies. And it is a cost threshold only — the tests in Section 5 routinely override it, as the archetypes below show.

The portfolio effect: what a single-application model cannot see

Everything above prices one application in isolation, and that is the model's most serious limitation. SaaS is bought one business domain at a time; an internal build is not. The second application on an internal platform inherits identity, the data model, the integration fabric, CI/CD, observability and increasingly the agent orchestration layer. The second commercial product arrives with its own identity model, its own data model and its own integration surface. **A single-application TCO comparison is therefore structurally biased toward buying**, and Figure 2 should be read with that bias declared.

The portfolio effect: reuse changes the answer one application cannot

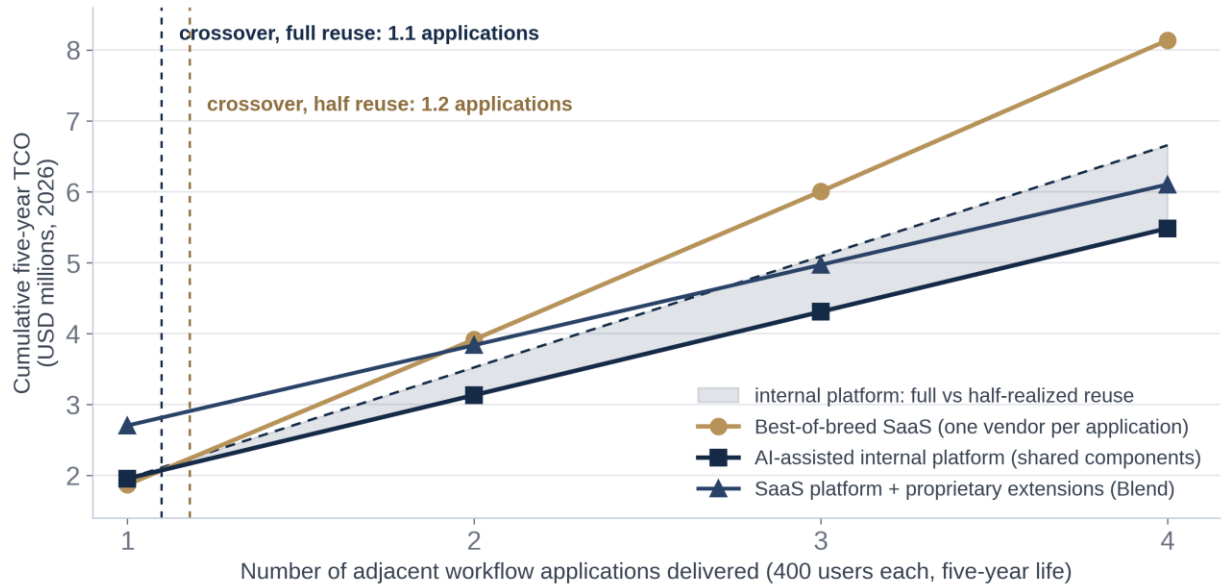

Figure 4 (StratoFactory analysis). Cumulative five-year TCO for one to four adjacent workflow applications in the same value chain, each serving 400 users. Method: the Figure 2 base case extended with marginal-application coefficients — internal reuse of build 65%, QA 70%, integration 45%, security 60%, infrastructure 35%, governance

30%; SaaS integration rising 15% per additional vendor plus $28,000 per vendor per year of vendor management. The shaded band shows the internal platform if only half the assumed reuse is realized. Modeled, not observed.

Modeling four adjacent workflow applications changes the answer. The marginal application on an internal platform costs **$1.18m over five years against $2.05m for the next best-of-breed product** — the internal marginal cost is flat because the platform is already paid for, while the SaaS marginal cost *rises* with each additional vendor. Cumulative cost crosses over at **1.1 applications**, or **1.2** if only half the assumed reuse is realized — that is, at the second application. At four applications the internal platform totals $5.5m against $8.1m for best-of-breed SaaS, a 33% difference, with the blend between them at $6.1m. The SaaS side of that comparison is not a straw man: on the benchmark cited in Section 2, large enterprises add applications faster than they retire them, a third of licenses go unused, and most spend sits outside IT — estate fragmentation is not a risk of the buy strategy, it is its observed default.

But the reuse this rests on is an assumption, and it is the assumption most at odds with the rest of the evidence in this briefing. Cross-file reuse is down 35% and refactoring activity down 70% since 2023 (GitClear, 2026, observed data). Agent-written code, as it is currently produced, copies rather than reuses — which is precisely the behavior that destroys the portfolio effect. The crossover in Figure 4 is not a property of building; it is a property of a platform team, a component registry and enforced reuse standards. Without those, the internal line rotates toward the dashed one and the answer reverts to the single-application result. **The portfolio case is the strongest economic argument for building, and it is the argument most exposed to how AI-assisted software is actually being written.**

Dimension	Archetype A — industrial manufacturer, 1,000–3,000 employees	Archetype B — services / distribution enterprise, 5,000+ employees
Software users	~250 knowledge workers; the rest are shop-floor or field	~4,200 knowledge workers across many functions
Where differentiation sits	Narrow and deep: production scheduling, quality exception handling, engineering configuration	Broad and shallow: client onboarding, pricing exceptions, service orchestration
What the cost model says	At ~250 seats, buying is $529k cheaper per application over five years — the cost model says BUY	At ~4,200 seats, building is roughly $11m cheaper per application at list price — the cost model says BUILD
What the tests say	Advantage test passes, corpus test passes, portfolio leverage strong — BUILD the two or three narrow workflows anyway	Advantage test fails for horizontal processes, corpus test fails for legal, tax and payroll, verification and change capacity binding — BUY anyway
Dominant cost line	Maintenance and integration into ERP/MES; subscription is minor	Subscription and change management; engineering is minor
Recommended posture	BUILD the two or three differentiating workflows on bought infrastructure; BUY everything horizontal	BUY the estate; BLEND on the two client-facing platforms; ORCHESTRATE the back office
Highest-return first move	Orchestrate order-exception handling across the existing ERP and MES	Orchestrate accounts payable and service ticket triage across the existing estate

Dimension	Archetype A — industrial manufacturer, 1,000–3,000 employees	Archetype B — services / distribution enterprise, 5,000+ employees
Portfolio leverage	Strong — the three workflows share ERP/MES data, identity and shop-floor context; one platform serves all three	Weak horizontally — the functions share little; strong only inside the two client-facing platforms
Binding constraint	Verification capacity — a small engineering group cannot review at agent speed	Change management — 4,200 users make process change, not code, the expensive item
What would change the answer	Failure to stand up a platform team — without enforced reuse the portfolio leverage disappears and the cost model, which already says BUY, decides	Enterprise-agreement discounting failing to materialize at 4,200 seats, which would make an $11m per-application cost gap impossible to override

Table 1. Two constructed archetypes (StratoFactory analysis). Analytical constructions used to show how the same model produces opposite answers under different structural conditions. These are not case studies, and no real company data is used.

The framework is not a ranking of options but a function of structure: the manufacturer's differentiating processes are narrow, deep, adjacent and served by few people; the services enterprise's are broad, shallow and served by many. The same model, on the same assumptions about AI, recommends opposite strategies.

7. What Is Exposed, and What Is Not

Substitution risk is not a property of AI capability. It is a property of the software category. This section scores seventeen enterprise software categories against nine criteria and produces two exhibits: a positioning map and a scored index. Both are **StratoFactory analysis**, and the methodology and its limits are stated in full.

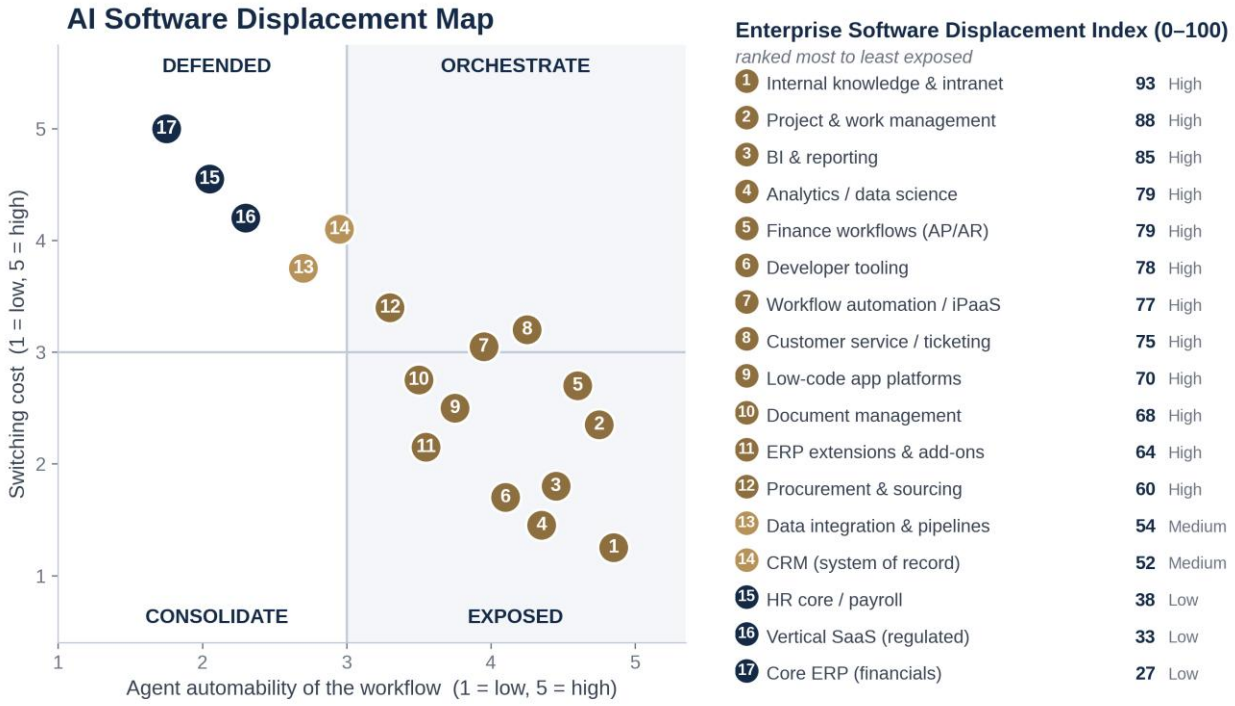

Figure 5 (StratoFactory analysis). AI Software Displacement Map: agent automability of the workflow against switching cost, with each category colored by its Displacement Index band. Positions are analyst judgments on a 1–5 scale, not measured market data. Method: automability and switching cost scored independently of the Index; the Index is shown in the key for reference.

Method, and what it cannot tell you

Each category is scored 1–5 on eight resisting factors — workflow complexity, differentiation value, vendor data advantage, regulatory burden, integration depth, switching cost, network effects, system-of-record status — and on one enabling factor, agent accessibility and API quality. Resistance is their weighted mean, with switching cost and system-of-record status weighted highest at 0.15 and network effects lowest at 0.08. The Index is 100 × (0.65 × normalized inverse resistance + 0.35 × normalized enablement), banded High at 55 and above, Medium 38–54, Low below 38.

The limitations are material. The scores are analyst judgments, not survey data; the weights are chosen, not fitted; the categories are coarse and contain internal variation larger than some of the gaps between them; there is no time dimension, so a High score indicates exposure, not a schedule; and no revenue at risk is estimated, because the only published figure of that kind — $234bn, roughly 20% of enterprise application SaaS spend by 2030 (Gartner, 2026, analyst forecast) — carries no public methodology to calibrate against.

Category	Cx	Df	Da	Rg	In	Sw	Ne	SR	Ag	Index	Band
Internal knowledge & intranet	1	2	1	2	2	1	2	1	5	**93**	High
Project & work management	1	2	1	1	2	2	3	2	5	**88**	High
BI & reporting	2	2	2	2	3	2	1	1	5	**85**	High

Category	Cx	Df	Da	Rg	In	Sw	Ne	SR	Ag	Index	Band
Analytics / data science	3	4	2	2	3	2	1	1	5	**79**	High
Finance workflows (AP/AR)	2	1	2	4	3	3	1	2	5	**79**	High
Developer tooling	3	3	2	1	3	2	3	2	5	**78**	High
Workflow automation / iPaaS	2	3	1	2	4	3	2	2	5	**77**	High
Customer service / ticketing	2	2	2	3	3	3	2	3	5	**75**	High
Low-code app platforms	2	3	1	2	3	3	2	2	4	**70**	High
Document management	2	1	2	4	3	3	1	3	4	**68**	High
ERP extensions & add-ons	3	4	2	3	4	2	1	2	4	**64**	High
Procurement & sourcing	3	2	3	3	3	3	4	3	4	**60**	High
Data integration & pipelines	4	3	2	3	5	4	1	3	4	**54**	Medium
CRM (system of record)	3	3	3	3	4	4	3	4	4	**52**	Medium
HR core / payroll	4	1	3	5	4	5	2	5	3	**38**	Low
Vertical SaaS (regulated)	4	4	4	5	4	4	3	4	3	**33**	Low
Core ERP (financials)	5	3	4	5	5	5	2	5	3	**27**	Low

Figure 6 (StratoFactory analysis). Enterprise Software Displacement Index. Columns: Cx workflow complexity · Df differentiation value · Da vendor data advantage · Rg regulatory burden · In integration depth · Sw switching cost · Ne network effects · SR system-of-record status · Ag agent accessibility and API quality. Each scored 1–5; Index = 100 × (0.65 × normalized inverse resistance + 0.35 × normalized enablement); higher = more exposed to agentic substitution. Scores are analyst judgments, not measured data; see the methodology and limitations in the text.

What the scoring says

The exposed end is dominated by thin layers over data someone else owns: internal knowledge and intranet (93), project and work management (88), BI and reporting (85), analytics (79), finance workflows such as AP and AR (79), developer tooling (78), workflow automation (77) and customer service ticketing (75). What these share is not simplicity but **absence of a defensible substrate** — they read and present data held elsewhere, their workflows are rule-bounded, their APIs are good, and leaving them is cheap.

The defended end is the opposite: core ERP financials (27), regulated vertical SaaS (33) and HR core and payroll (38). These carry statutory obligations, multi-jurisdiction compliance maintained continuously, deep integration, high switching cost and system-of-record status. The mechanism is the code-or-corpus test from Section 5: what the customer buys is a maintained body of jurisdictional rules, templates and filing integrations that no amount of cheap code creation reproduces — which is why the criterion scored as *vendor data advantage* in Figure 6 is doing most of the work in these rows. An agent can operate these systems far more easily than it can replace them, which is precisely the ORCHESTRATE quadrant of Figure 5.

Where BUILD still loses — the case against the thesis

This is the section a report arguing that AI shifts software toward building must not skip. The strongest evidence against the thesis is not one finding but five, and together they are formidable.

First, internal builds fail more often than purchases. The 33%-versus-67% production-deployment gap (MIT NANDA, 2025, survey) is the most direct available test of the proposition and it fails it. The sample is small and conference-recruited, and the study has been widely criticized, but no larger contrary dataset has been published.

Second, agentic projects are being canceled at scale. Over 40% are forecast canceled by end-2027 on grounds of escalating cost, unclear business value and inadequate risk controls, and Gartner judges only about 130 of thousands of self-described agentic vendors genuine (analyst forecast, 2025).

Third, systems of record retain their moats. The Bain analysis that documented software indices falling 15% in weeks and 25% from twelve-month highs in early 2026 concluded that "for most enterprise systems of record, those moats remain strong" — switching barriers, mission criticality and data access — with average gross retention still around 90% or better (analyst, 2026).

Fourth, seats are not disappearing. Microsoft passed 30 million paid Copilot seats in FY2026 and grew Productivity and Business Processes revenue 14%; Salesforce grew subscription revenue 12% with current remaining performance obligation up 14%; ServiceNow grew subscription revenue 24.5% (vendor-reported filings, 2026). The agent era is, so far, being sold by the seat.

Fifth, full automation has been publicly reversed. Klarna's assistant handled work described as equivalent to 700 agents and 2.3 million chats in its first month; the company then resumed hiring human agents, its chief executive stating that AI was cheaper but "lower quality" and that "investing in the quality of human support is the way of the future" — remarks given to Bloomberg and reported by *Entrepreneur* in May 2025. Klarna did not return to its prior cost base — but it established that the automation frontier has a quality boundary that shows up in customer outcomes before it shows up in cost models.

These categories are not marginal: systems of record, safety-critical and heavily regulated software, deep multi-jurisdiction compliance, network-effect products and platforms requiring 24/7 support all remain firmly on the BUY side, and nothing in the 2025–2026 evidence base moves them.

And the case against this briefing's own conclusion

The section above is the case against the naive thesis that AI shifts software toward building. Symmetry requires the other one, because this briefing's conclusion — buy most of the estate, build narrowly, blend widely, orchestrate first — could also be wrong. Five things would overturn it.

Adoption is moving the other way. 32% of organizations decided against buying and built instead, 48% among high performers (McKinsey, 2026, survey), and 35% of one internal-tooling vendor's customers say they have already replaced at least one SaaS product with a custom build (**vendor-reported**, 2026). Revealed preference is not evidence of success, but it is not nothing either. **The model's own arithmetic points at building past a low threshold.** Cumulative cost crosses over at barely more than one application on a shared platform, and at 4,200 seats a build is cheaper by roughly $11m per application at list price (*StratoFactory analysis*) — the conclusion to buy rests on the two tests in Section 5 overriding the cost model, not on the cost model. **Capability is moving faster than the calibration.** Issue resolution on SWE-bench Verified went from 1.96% to about 78% in thirty months, and the largest independent productivity dataset records an inflection from around December 2025 (academic, 2026). A model calibrated on 2025–26 measurement will understate 2027–28 if that continues. **The optimistic claim, if even half true, settles it**: threefold to fivefold productivity with 60% smaller teams (consultancy claim, 2026, no method disclosed) would move every threshold in Section 6 decisively toward building. **And the verification constraint may be temporary** — it is a tooling gap, and tooling gaps close.

What would settle this is not a better forecast but two observations: whether internal-build production rates rise above the 33% baseline, and whether code-reuse signals reverse. Both are stated as indicators in Section 10.

8. Agentic AI and the SaaS Business Model

The structural shift is from *human → interface → workflow → outcome* to *agent → API → workflow → outcome*. If the interface stops being where work happens, the seat stops being a coherent unit of value, and every pricing model built on it becomes an accounting convention rather than an economic one.

The forecasts say this is imminent: pure seat-based pricing obsolete by 2028 with about 70% of vendors refactoring around consumption, outcomes or capability (IDC, 2025); a third of user experiences shifting from native applications to agentic front ends by 2028 (Gartner, 2025); a customer whose 500 seats become 450 as agents absorb tasks (Bain, 2026).

The disclosed numbers say something more complicated. Microsoft sold past 30 million paid Copilot seats — an agentic product priced per seat. In its quarter to July 2026, Salesforce reported Agentforce ARR above $1.5bn growing more than 240%, and combined Agentforce and Data 360 ARR of nearly $3.9bn growing over 210%, alongside 7.0 billion cumulative agentic work units with 3.2 billion in the quarter alone — a consumption metric reported next to, not instead of, subscription revenue, which itself grew 12%. ServiceNow crossed $1bn in AI annual contract value with agentic deployments up ninefold in nine months, targeting 30% of ACV from AI by 2030 (all vendor-reported, 2026). Note the scale: Agentforce ARR is roughly 3.2% of Salesforce's FY27 revenue guidance midpoint, up from 2.6% a quarter earlier. Agent revenue is growing very fast from a small base; seat and subscription revenue keeps growing underneath it.

Does the enterprise still need a seat for every human?

The answer differs by pricing model. Under **seat pricing**, no: if an agent does the work, the license is dead weight — and the 36% unused-license rate (Zylo, 2026) suggests many seats were dead weight before agents arrived. Under **usage pricing** the question is meaningless: an agent doing the work of three people consumes roughly the volume of three, often more, because agents retry, verify and over-fetch. Under **outcome pricing**, the enterprise pays for resolved tickets or matched invoices and seat count is irrelevant to both parties.

This is why the deterioration in retention is the most informative signal available. Median gross revenue retention fell to 84% in 2026 from 88%, and the 75th percentile from 95% to 91%, attributed to renewal scrutiny against AI alternatives, seat rationalization as companies run leaner, and a growing dependence on expansion — 40% of net new ARR at the median company now comes from expansion rather than new logos (Benchmarkit via The SaaS CFO, 2026, observed data). That is not logo churn — it is customers keeping the vendor and buying fewer seats. **Seat compression is arriving before vendor displacement**, and it arrives as a retention problem rather than a churn problem.

Destruction, redistribution, or expansion?

Three readings are available and the evidence supports the middle one. **Destruction** would require agents to replace applications rather than operate them; the 17% deployment rate, the >40% forecast cancellation rate and the intact moats around systems of record all argue against it. **Expansion** — that agents drive so much new consumption that total software spend rises — is supported by the 15.5% growth in software spending in 2026 and by Gartner's forecast that agentic AI reaches roughly 30% of enterprise application software revenue by 2035, above $450bn, from 2% in 2025 (analyst forecasts, 2026 and 2025). **Redistribution** is the reading that reconciles the rest: the same release that put $234bn of enterprise application SaaS spend at risk by 2030 describes the result as a "metamorphosis" of the SaaS market rather than its destruction, with legacy share cannibalized by incumbents and entrants alike (Gartner, 2026, analyst forecast). Spend moves from per-seat interface licenses toward consumption, data and outcome contracts; it moves from workflow and reporting tools toward systems of record and orchestration platforms; and it moves, in part, from software line items to inference line items.

Convergence: the differentiation risk that is actually new

There is a version of the objection that bought software cannot differentiate which none of the above answers, and it concerns the future rather than the present. Historically the residual differentiation on packaged software lived in configuration, process design and the people running it — which is what the 73%-versus-25% workflow-redesign gap measures (McKinsey, 2026, survey). If the vendor now ships the agent, and the agent encodes the process, that residual is absorbed into the product and distributed to every customer including your competitors: competitive variance compresses toward the vendor's default playbook, and accumulated process knowledge becomes an input to a model the enterprise does not own — the same mechanism by which data accumulated inside a product becomes a vendor moat (Bain, 2026, analyst). The likely resolution is not that enterprises build instead. Vendors face rising churn risk if every customer looks identical, which is why 30% of enterprise application vendors are expected to launch their own Model Context Protocol servers (Forrester, 2025, analyst forecast). **Convergence risk is a strong argument for owning the agent and the process, and a weak argument for owning the application.**

The vendors' countermove, and what may hold

Incumbents have responded on five fronts: embedding copilots, shipping task-specific agents, launching agent platforms, exposing the agent-grade interfaces described above, and repricing. Gartner expects 40% of enterprise applications to include task-specific agents by end-2026, from under 5% in 2025, and half of ERP vendors to launch autonomous governance modules (analyst forecasts, 2025).

The structural advantages that may hold are the ones from Section 2: switching costs measured in years, proprietary data accumulated inside the product, workflow integration into processes the customer cannot easily describe let alone rebuild, multi-jurisdiction compliance, an ecosystem of skills, and the trust of being the system auditors already accept. Agents erode none of them. What they erode is the *interface* — and the interface was never where the moat was.

9. Architecture, Procurement and the CIO's Mandate

If agents become significant consumers of enterprise software, the architecture that supported human users is insufficient in specific, enumerable ways. Applications built for humans assume a session, a screen, a person who notices when something looks wrong, and an audit trail keyed to a named user. Agents break all four assumptions.

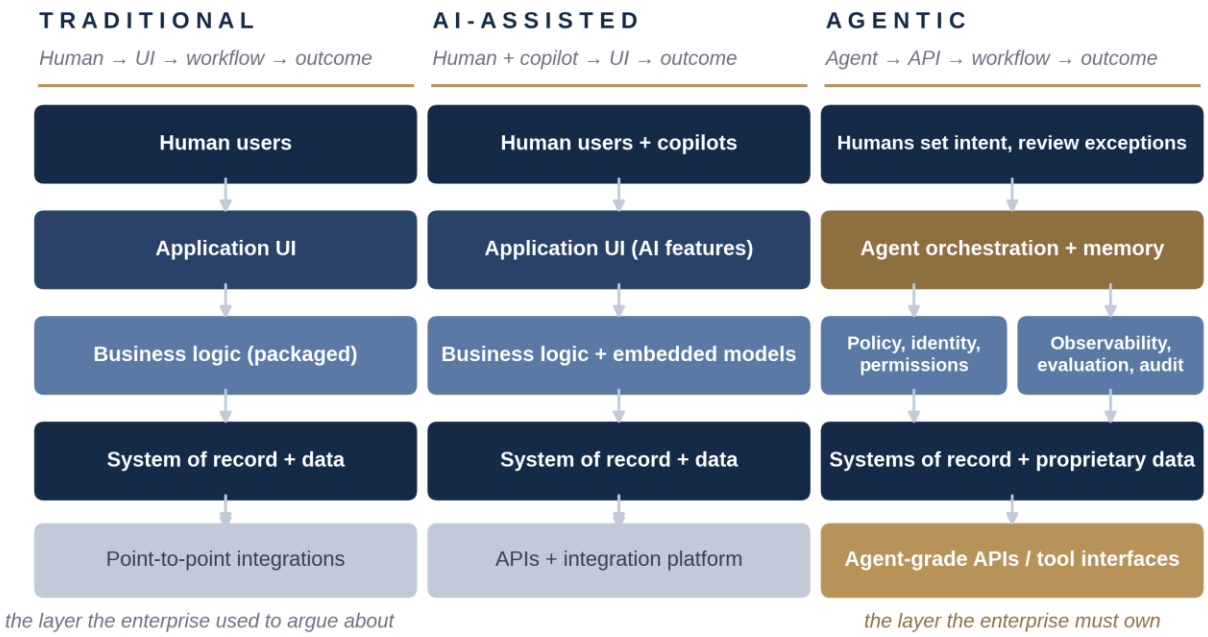

Figure 7 (StratoFactory analysis). Traditional, AI-assisted and agentic enterprise architecture. Conceptual diagram; the layers shown are architectural roles, not products, and the progression is a design pattern rather than an observed migration path.

What the transition demands

Seven requirements follow from the evidence. **APIs must become agent-grade** — complete, semantically documented, idempotent, rate-limited — because an agent's error mode is volume, not confusion. **Systems of record must remain authoritative**, because an agent acting on stale state amplifies the error rather than noticing it. **Identity and permissions must extend to non-human actors**: at 109 machine identities per human, this is no longer theoretical (Palo Alto Networks, 2026, survey). **Observability must cover reasoning, not just requests. Agent memory needs a retention policy** satisfying the EU AI Act's six-month log-retention duty on deployers of high-risk systems (2026). **Orchestration must be a governed layer**, not a collection of scripts. And **interoperability must be assumed contested**, which makes vendor-hosted agent interfaces a strategic dependency, not a convenience.

Owning the inference layer: FinOps, governance and the risk you inherit

Two forms of control are systematically missing from build-versus-buy business cases, and both favor building. The first is **financial control of inference**. Inside an embedded-AI product the enterprise buys tokens at the vendor's markup, on the vendor's model, under the vendor's caching and retry policy, and sees an invoice line rather than a decision it can change. Owning the workflow means owning model routing, context budgets, caching, smaller-model substitution for routine steps and per-team consumption limits — and capturing the price deflation directly. Inference prices fell 9× to 900× per year depending on task (Epoch AI, 2025, observed data), but the enterprise captures that only where it controls the routing decision, which is why 78% of IT leaders reported unexpected consumption or AI charges (Zylo, 2026, survey). The honest scale, though, is that model and API spend is **2.5% of five-year TCO** in the base model: control of it does not flip a decision today. It is an option on the line Gartner expects to exceed the average developer's salary by 2028 (analyst forecast, 2026), and should be valued as an option, not a saving.

The second is **governance, residency and audit control**, and it is the argument that matters most in regulated industries. Building keeps data, prompts, embeddings, logs and model choice inside the enterprise boundary, with no subprocessor chain to inventory and no vendor release cycle that can change model behavior under a validated control. That is not a marginal consideration: over 20% of organizations have already been breached through AI models or applications (IBM, 2026, observed study), and the API-and-plug-in exposure quantified in Section 4 is precisely the third-party surface that building removes. The regulatory position cuts both ways and should be stated precisely. A firm that builds a high-risk system becomes a **provider** under the EU AI Act, taking on conformity assessment, technical documentation and registration; a firm that buys is a **deployer**, owing human oversight and six-month log retention but relying on a provider it does not control (Holland & Knight analysis of the EU AI Act, 2026). **Building converts third-party risk into first-party risk.** For a regulated enterprise with a mature security function, model-governance capability and an audit trail it already owns, that is the better trade, and it is a real argument for building that no cost model captures. For an enterprise without those, the same evidence points the other way: 44% of AI-generated code fails security testing (Veracode, 2026) and internal builds reach production about a third of the time (MIT NANDA, 2025). Control is only an advantage to an organization equipped to exercise it.

The constraints that could stop it

This transition is not inevitable, and three constraints could stall it. The first is **verification capacity**, as established in Sections 3 and 4: security pass rates stuck at 56% and refactoring down 70% describe an organization that cannot safely absorb more machine-generated change. The second is **data readiness**: IDC forecasts a 15% productivity loss by 2027 for companies that do not prioritize AI-ready data (analyst forecast, 2025), and the integration depth that protects incumbents also obstructs agents. The third is **governance and liability**: the breach exposure set out above attaches to the deployer, not only to the vendor (Holland & Knight analysis of the EU AI Act, 2026). A single well-publicized agent-caused loss in a regulated process would change enterprise risk appetite faster than any capability release.

From "build or buy?" to "which layer should we own?"

The procurement question has to be re-specified by layer, because the answer differs at each. At the **data layer**, own — always, and including the right to extract. At the **system-of-record layer**, buy, and negotiate for interface quality rather than features. At the **business-logic layer**, own what is differentiating and buy what is statutory. At the **workflow layer**, own, because this is where competitive process knowledge lives and where agents will operate. At the **agent layer**, own the orchestration and the evaluation; rent the models. At the **interface layer**, expect it to commoditize, and stop paying for it. At the **infrastructure layer**, rent.

Eight tests resolve each layer decision: strategic differentiation, economic value proportionate to the cost of ownership, data advantage no vendor can replicate, switching cost if the decision is wrong, AI-substitutability over five years, integration cost in each direction, regulatory exposure and to whom it attaches, and organizational capability. The last is the one most often skipped and the one the failure data most strongly supports weighting.

10. Predictions 2026–2028, Limitations, and the Next Twelve Months

These are StratoFactory predictions — judgments, not sourced facts. Each is stated so that it can be judged right or wrong by the end of 2028.

- **P1. Seat-based pricing will not be obsolete by the end of 2028; it will persist as a floor with consumption riders above it.** *Rationale:* every major vendor monetizing agents in 2026 did so alongside, not instead of, seat revenue. *Evidence:* 30m paid Copilot seats; Agentforce ARR above $1.5bn reported next to 12% subscription growth at the same vendor; ServiceNow's 24.5% subscription growth (vendor-reported, 2026). *Indicator:* at least three of Microsoft, Salesforce, SAP, Oracle, ServiceNow and Workday still publishing per-seat list prices for flagship suites in Q4 2028. Directly contrary to IDC's 2028 obsolescence forecast.
- **P2. The share of enterprises reporting any EBIT impact from AI will remain below 50% through 2027.** *Rationale:* the constraint is workflow redesign and verification capacity, neither of which improves at model-release cadence. *Evidence:* 37% in 2026, flat on 2025; 6% at the 5%-or-more threshold (McKinsey, 2026, survey). *Indicator:* the same measure in the 2027 or 2028 McKinsey State of AI survey. *Timeframe:* 31 December 2027.
- **P3. At least one Fortune 500 or equivalent enterprise will publicly reverse a build-instead-of-buy decision on maintenance or security grounds and return to a commercial product.** *Rationale:* the 2026 cohort of AI-built internal applications reaches its second maintenance year in 2028, when duplication and refactoring debt surface. *Evidence:* internal-build production rate of 33% (MIT NANDA, 2025); duplication at a record high and refactoring down sharply (GitClear, 2026). *Indicator:* a named reversal in an earnings call, filing or vendor case study by 31 December 2028.
- **P4. Model and API consumption will become a separately disclosed enterprise IT budget line, exceeding 5% of application software spend in at least one large published benchmark.** *Rationale:* unbudgeted lines get their own reporting once they cause cancellations. *Evidence:* 78% of IT leaders reporting unexpected consumption or AI charges and 61% cutting projects as a result (Zylo, 2026); Gartner's forecast that AI coding cost overtakes developer salary by 2028. *Indicator:* a named benchmark reporting AI/inference spend as a distinct category above 5% of application software spend. *Timeframe:* 31 December 2028.

- **P5. Displacement will appear first as seat compression inside retained vendors, not as vendor replacement: median gross revenue retention for private B2B software will fall below 82% in a recognized benchmark before any top-ten enterprise application vendor loses absolute subscription revenue.** *Rationale:* switching costs bind even when seat demand falls. *Evidence:* GRR down from 88% to 84%, attributed to seat rationalization (Benchmarkit via The SaaS CFO, 2026); gross retention near 90% while net retention stalled (Bain, 2026). *Indicator:* published GRR below 82% with no top-ten vendor reporting a subscription revenue decline. *Timeframe:* 31 December 2028.

Limitations

What the models omit, and their scope. Three arguments here are made qualitatively because no defensible number could be derived from the retrieved evidence: the value of controlling inference routing, data residency and audit ownership (Section 9); the value of a maintained domain corpus that AI does not reproduce (Section 5); and convergence risk, where a vendor-shipped agent erases the process differentiation configuration used to protect (Section 8). They point in different directions and none appears in Figures 2, 3 or 4. The model also contains no J-curve, though the evidence describes a real rollout dip, which makes it optimistic toward the AI-assisted options. Separately, the base TCO model prices a single application over five years, so it cannot see cross-application reuse or estate fragmentation and is biased toward buying; Figure 4 extends it, but its reuse coefficients are StratoFactory estimates with no published benchmark behind them, and the $28,000 per additional vendor per year of estate overhead is an estimate anchored to observed license waste. A cost model that omits these things is not neutral; it is silent, and a business case built only on those figures would be too.

Weak evidence and interested parties. The 33%-versus-67% internal-versus-external deployment comparison comes from a study of 52 interviews and 153 conference-recruited responses, not a probability sample; the gross-revenue-retention decline is reported through a secondary source that does not disclose sample size; the technical-debt baseline is from 2020 and predates generative AI. Several findings come from parties with a direct commercial interest and are labeled as such wherever they appear: GitHub's Copilot study, Retool's survey of its own customers, and Veracode's and GitClear's benchmarks. The Microsoft dose-response study is written by Microsoft employees studying Microsoft engineers using a Microsoft product; its method is unusually transparent, which is why it is used, but the affiliation is material.

Contradictory evidence, presented rather than reconciled. Four contradictions run through this briefing and none has been averaged away: AI makes developers 40.5% more efficient (observational, n=16,223), 12.9–21.8% more productive (the same firm's field data), or slowed them 19% and now perhaps 4% (randomized, n=16); enterprises are choosing to build (32%) while internal builds reach production a third of the time; $234bn of application spend is "at risk" while the same firm forecasts agentic AI at 30% of application software revenue; and inference unit prices fall by orders of magnitude while total AI coding cost is forecast to exceed a developer's salary. Each pair is explained by the difference in what is measured.

Data that could not be obtained. No primary vendor pricing document for agent, credit or consumption packaging could be retrieved, so this briefing makes **no quantified claim about any specific vendor's agent pricing**. No credible current figure for application maintenance as a share of IT budget was retrievable. No traceable primary source for Model Context Protocol adoption was found, so the only interoperability figure cited is an analyst forecast. A widely circulated claim that non-seat deals reached 50% of ServiceNow's new business does not appear in ServiceNow's own release and is excluded.

Format and scope. Two production constraints are recorded rather than resolved silently. This briefing is set on a **US Letter** template, not A4, so page geometry follows the template. And at the template's typography — 11 pt Arial, 6.5-inch column, 1.25 line spacing — a body of roughly 11,700 words with seven exhibits occupies about 32 pages, not the 12–14 originally estimated; roughly 5,300 words would fit that page count. Analytical completeness was treated as binding and the page count as its consequence.

What would change first. If the EBIT-impact figure rose above 50%, the central claim of Section 4 would be in trouble and the build case would strengthen sharply. If the security pass rate for AI-generated code moved decisively above 80%, the verification constraint would loosen. If code-reuse signals reversed, the portfolio crossover in Figure 4 would become a reliable planning assumption rather than a conditional one. If a top-ten vendor reported an absolute subscription decline attributable to agent substitution, the redistribution reading in Section 8 would give way to the destruction reading.

The next twelve months

Four actions follow from the evidence rather than the forecasts. **Measure verification capacity before expanding generation capacity** — review throughput, security pass rate on internally generated code, mean time to detect defects in AI-authored changes — and **measure component reuse** alongside it, because the portfolio case for building depends on reuse and the observed trend runs the wrong way. **Instrument model and API consumption as a budget line now**, before it becomes a variance. **Run one orchestration program on the existing estate**, in a high-volume rule-bounded back-office process, measured on cost per completed transaction rather than adoption. And **re-specify the next three procurement decisions by layer**, asking which layer must be owned rather than whether the whole thing should be built.

11. Conclusion: What to Own, What to Buy, What to Let AI Build

The governing question was whether an enterprise still needs to buy as much software once AI makes software cheap to build. On the evidence assembled here the answer is: **yes, for most of the estate, and for a reason that has nothing to do with how hard code is to write.** The vendor's proposition was never cheap code. It was the amortization of every non-code cost across a customer base, and AI has barely touched that amortization while adding two new cost lines of its own.

BUY standardized, complex, regulated, deeply integrated or non-differentiating software — and buy it hardest where the value sits in a maintained corpus rather than in code. Legal, tax, payroll, statutory reporting, trade compliance and comparable specialist domains are the clearest buy in the whole estate, because AI lowers the cost of writing the application and does nothing about the content, the jurisdictional maintenance or the liability. Note that cost points the other way on seats: the Section 6 model puts the per-application crossover near 430 users, and above it building is cheaper because subscription scales with seats while build cost does not. Buy above that line anyway where the two tests say so — this is the part of the estate where parity is the objective, and the objection that bought software cannot differentiate is correct and mostly irrelevant.

BUILD where the workflow is genuinely differentiating, the enterprise holds data no vendor can access, the user population is small relative to the value of the process, and — the condition most often omitted — the organization can verify what it builds. Build as a **platform, not as an application** — this is the single most important qualification in the briefing. The case is far stronger for the second, third and fourth adjacent workflow than for the first, because shared identity, data model, integration fabric, components and agent orchestration are paid for once and reused, while every additional commercial product adds its own stack and its own integration surface. An enterprise that builds one thing should usually have bought it; an enterprise building four adjacent things on one platform is in a different economic position entirely. It stands or falls on reuse actually happening, which is an engineering-governance choice, not a consequence of deciding to build. Build also where control is the point rather than cost — where routing, model choice and token consumption must be owned, or where data residency, subprocessor exposure and audit trail decide the matter. The 33% internal-build production rate is the price of ignoring the verification condition, and the collapse in code reuse is the price of ignoring the platform one.

BLEND in most enterprise situations, and specifically wherever the advantage test passes but the corpus test fails. It is the most expensive option for a single application and closes most of that gap by the fourth, because it buys compliance, support, domain content and an upgrade path while retaining ownership of the workflow, the data model and the agents. It is also the only answer to convergence risk: if the vendor ships the agent, every competitor runs your process. The premium on application one is the price of being able to change the top layer without changing the bottom one.

ORCHESTRATE first. This is the conclusion the evidence supports most strongly and the one the current debate most neglects. Deploying agents across the existing estate requires no replacement decision, monetizes integration already paid for, is cheap to reverse, and attacks the back-office processes where measured AI returns actually concentrate. The categories in the ORCHESTRATE quadrant of Figure 5 — the ones that are automatable but expensive to leave — are where most enterprises will find their 2026–2028 return.

Where the evidence was too thin for a confident answer. Three questions in this briefing remain genuinely open. Whether agent-era pricing settles on consumption or on outcomes cannot be determined from the disclosures available, because no vendor publishes the mix. Whether AI-built software is more expensive to maintain over a full lifecycle is not yet observable, because the first substantial cohort of AI-built enterprise applications has not reached its third maintenance year. And whether the enterprise EBIT gap closes is the central unresolved question of the entire field: two consecutive years of flat readings is a pattern, not yet a verdict.

12. Sources and Methodology

Method

This briefing is built on sources retrieved and read on 3 September 2026, after a sweep for the most recent edition of every recurring publication cited. Where a source publishes on a quarterly or annual cycle, the latest edition available at that date is used and any superseded edition is noted in the evidence table. No statistic, forecast or quotation appears in the text unless the underlying source was opened in the course of the research; where a source could not be reached, the claim was cut rather than approximated. Every figure carries the year of the underlying data, and all currency is US dollars. Where credible sources disagree, both are presented with the methodological reason for the divergence; conflicting figures are never averaged. A full evidence table, including the contradictory findings, the sources that could not be obtained, and the derivation of every StratoFactory estimate, accompanies this briefing.

Claims are labeled throughout as observed data, survey response, vendor-reported claim, analyst forecast, academic finding, StratoFactory analysis, StratoFactory estimate or StratoFactory prediction.

Third-party material is used sparingly and by attribution. Where a publisher's study contains a fuller set of results than is needed here, this briefing reports only the figures its argument rests on and characterizes the rest in words, so that the summary complements rather than substitutes for the original. Every exhibit in this briefing is original work; none reproduces a third-party chart or table. The rights position is set out in Section 13.5. Figures 2, 3, 4 and 5 and Table 1 are model outputs or analyst judgments and are labeled StratoFactory analysis; they are not market data and must not be read as such.

Sources

- **Gartner.** Worldwide IT Spending Forecast, 27 Jul 2026 ($6.37tn; software $1,468bn). $234 Billion in Enterprise Application Software Spend Is at Risk from Agentic AI, 1 Jul 2026. AI Coding Costs Will Surpass Average Developer's Salary by 2028, 24 Jun 2026. 60% of Organizations Will Adopt Smaller Software Engineering Teams by 2029, 7 Jul 2026. Enterprise AI Coding Agents Market, 20 May 2026. 2026 Hype Cycle for Agentic AI, 15 Apr 2026. Over 40% of Agentic AI Projects Will Be Canceled by End of 2027, 25 Jun 2025. 40% of Enterprise Apps Will Feature Task-Specific AI Agents by 2026, 26 Aug 2025. Top Predictions for IT Organizations and Users in 2026 and Beyond, 21 Oct 2025. gartner.com.
- **McKinsey & Company.** The State of AI: Global Survey 2026, 25 Aug 2026 (n=1,719 across 97 nations, fielded 4 May–8 Jun 2026). Tech debt: Reclaiming tech equity, 6 Oct 2020 (n=50 CIOs) — dated, used as a pre-AI baseline only. mckinsey.com.
- **MIT NANDA.** The GenAI Divide: State of AI in Business 2025 — Challapally, Pease, Raskar and Chari; 52 interviews, 153 survey responses, 300+ public initiatives, Jan–Jun 2025.
- **DORA / Google Cloud.** 2025 State of AI-assisted Software Development, 23 Sep 2025 (n≈5,000, 100+ hours of interviews). ROI of AI-assisted Software Development, 11 May 2026 (modeled 500-engineer organization; 39% first-year and 727% three-year return; J-curve; greenfield 35–40% vs complex legacy ≤10%) — figures via InfoQ's coverage, the full report being gated. cloud.google.com; infoq.com.
- **METR.** Measuring the Impact of Early-2025 AI on Experienced Open-Source Developer Productivity, 10 Jul 2025 (RCT, 16 developers, 246 issues). We are Changing our Developer Productivity Experiment Design, 24 Feb 2026 (revised estimates and disclosed selection bias). metr.org.
- **Heilman, A., Kyllo, A. and Murphy-Hill, E. (Microsoft).** GitHub Copilot and Developer Productivity: An Observational Dose-Response Analysis. arXiv:2606.00438, 30 May 2026 (n=16,223 engineers, 43 weeks). Author affiliation is material.
- **Stack Overflow.** 2025 Developer Survey, published 29 Dec 2025 (n=49,000+). stackoverflow.blog.
- **Veracode.** 2026 GenAI Code Security Report, 28 Jul 2026 (100+ models, four testing snapshots; 56% security pass rate; Python 63%, Java 30%). Supersedes the Spring 2026 edition. Vendor-run benchmark; method disclosed. veracode.com; businesswire.com.

- **GitClear.** The Maintainability Gap: 2026 AI Code Quality Research, Jan 2026 (623m code changes, 2023–2026). Vendor-run; correlational. gitclear.com.
- **GitHub.** Quantifying GitHub Copilot's impact in the enterprise with Accenture, 13 May 2024. **Vendor-sponsored**; sample size not disclosed. github.blog.
- **Stanford HAI.** 2026 AI Index Report, Chapter 4 — Economy (enterprise adoption 88%; corporate AI investment $581.69bn in 2025; software-developer employment ages 22–25 down ~20% from 2024). hai.stanford.edu.
- **IDC.** FutureScape: Worldwide Predictions 2026, 23 Oct 2025 (seat-based pricing obsolete by 2028, 70% of vendors refactoring; 15% productivity loss by 2027 without AI-ready data). idc.com.
- **Forrester.** Predictions 2026: Enterprise Software, 29 Nov 2025, read via secondary summary (30% of enterprise app vendors to launch MCP servers; half of ERP vendors to launch autonomous governance modules). The full report is paywalled.
- **Bain & Company.** Why SaaS Stocks Have Dropped — and What It Signals for Software's Next Chapter, 9 Feb 2026. bain.com.
- **Zylo.** 2026 SaaS Management Index, 29 Jan 2026 ($75bn of spend, 40m licenses, n=218 IT leaders). Vendor-run. zylo.com.
- **Benchmarkit**, 2026 B2B SaaS & AI-Native Metrics report, reported via The SaaS CFO, 30 Jun 2026 (median GRR 84%). Sample size not disclosed in the accessible source.
- **Vendor disclosures and filings.** Salesforce, Record Second Quarter Fiscal 2027 Results, 26 Aug 2026. ServiceNow, Second Quarter 2026 Financial Results, 22 Jul 2026. Microsoft, FY2026 Fourth Quarter Results, 29 Jul 2026. All **vendor-reported**.
- **IBM.** 2026 Cost of a Data Breach, 29 Jul 2026 (n=602 organizations, Mar 2025–Feb 2026). newsroom.ibm.com.
- **Palo Alto Networks.** 2026 Identity Security Landscape, 14 May 2026 (109 machine identities per human), read via Help Net Security.
- **Epoch AI.** LLM inference prices have fallen rapidly but unequally across tasks, 12 Mar 2025. epoch.ai.
- **US Bureau of Labor Statistics.** Occupational Outlook Handbook: Software Developers, Quality Assurance Analysts and Testers, last modified 27 Aug 2026 (median wage $135,980, May 2025). bls.gov.
- **Panorama Consulting Group.** 2026 ERP Report (n=170, Jan 2025–Jan 2026). panorama-consulting.com.
- **Retool.** 2026 Build vs. Buy Report, 17 Feb 2026 (n=817). **Vendor-sponsored survey of the vendor's own customers.**
- **Holland & Knight.** US Companies Face EU AI Act's Possible August 2026 Compliance Deadline, 28 Apr 2026 — legal analysis of the primary regulation. hklaw.com.
- **Klarna** company statements reported via Entrepreneur, 9 May 2025.

13. Disclaimer and Notices

This briefing is published for general distribution. The following notices apply to all readers and form part of the document.

13.1 Nature of this document and authorship

This briefing reflects the independent professional analysis of Piotr Jurowiec and is published by StratoFactory.com. Views expressed are those of the author and do not represent those of any employer, client, vendor or other organization. It is a research briefing, not a commissioned study, and it was prepared without input from any of the companies named in it.

13.2 Modeled analyzes — the TCO model and the Displacement Index

Sections 6 and 7 present two constructed analytical models. The total-cost-of-ownership comparisons in Figures 2, 3 and 4, the archetypes in Table 1, and the AI Software Displacement Map and Enterprise Software Displacement Index in Figures 5 and 6 are **StratoFactory analysis**. They are built from stated assumptions, not from observed transactions, and no real organization's cost data was used in their construction. The archetypes are analytical constructions, not case studies, and any resemblance to a specific company is coincidental. Index scores are analyst judgments on ordinal scales; the weights are chosen rather than statistically fitted; and the resulting numbers are comparative devices, not measurements. Applied to a specific organization, every input would change, and with it every conclusion. These models must not be used as a substitute for a costed business case, a vendor evaluation or a valuation input.

13.3 Forward-looking statements

The predictions in Section 10, and every forecast, scenario, threshold and recommendation elsewhere in this briefing, are forward-looking statements. They reflect judgment about uncertain future events as at September 2026 and are stated in falsifiable form precisely because they may prove wrong. Analyst forecasts quoted from third parties are their forecasts, not established facts, and are labeled as such. Actual outcomes will differ, potentially materially, and no undertaking is given to update any prediction as circumstances change.

13.4 Third-party data and vendor-reported figures

This briefing cites publicly available research, regulatory analysis, academic work, corporate filings, press releases and vendor disclosures. Where a figure originates with a party holding a commercial interest in the result — software vendors reporting on their own products, security and code-quality firms publishing benchmarks in their own market, and survey research commissioned by suppliers of the products surveyed — it is labeled **vendor-reported** or vendor-run at the point of use in the text, not merely in the source list, and readers should weight it accordingly.

13.5 Copyright, quotation, trademarks and non-endorsement

Original material. All text in this briefing is original. Every figure and table is the author's own work: Figures 2, 3 and 4 are outputs of a cost model built for this briefing, Figures 1, 5 and 6 and Table 1 are original analytical constructions, and Figure 7 is an original diagram. No third-party chart, table, figure, diagram, image, logo or layout has been reproduced.

Quotation. Direct quotation from third-party works is limited to short extracts — the longest is fourteen words — used to identify a source's own characterization of its findings, and each is attributed in the text to the organization that published it. Such quotation is made for the purposes of criticism, review, analysis and reporting, with acknowledgement of source, and no extract is of a length or nature that could substitute for the original work.

Data and findings. Statistics, survey results, benchmark figures and analyst forecasts attributed to third parties are individual factual references reported for commentary and analysis. Where a source publishes a larger set of results than is discussed here, only the figures material to this briefing's argument are reported and the remainder is described rather than reproduced; no source's dataset, results table or compilation has been reproduced in whole or in substantial part, and readers who need the full findings should obtain them from the original publisher. The underlying works remain the property of their authors and publishers, whose rights are acknowledged.

Trademarks. Company, product and service names appear solely to identify the organizations and products discussed. All trademarks and registered trademarks are the property of their respective owners. Nothing in this briefing is authorized, sponsored, endorsed by, or affiliated with any organization named in it, and no such association is claimed or implied. Analyst-firm forecasts are the property and opinion of the firms that issued them and are reproduced here only as short attributed references.

Corrections and rights enquiries. If any rights holder considers that material has been used beyond fair quotation or fair use, or that a figure has been misattributed, the author will correct or remove it promptly on request.

13.6 Data currency and verification

All sources were retrieved and read on 3 September 2026, and the briefing states the evidence position as of that date. Figures reflect the most recent publication available at that time; several — notably worldwide IT spending forecasts — were revised more than once during 2026, and the direction should be given more weight than the level. Survey findings, vendor disclosures and analyst forecasts are subject to revision or restatement by their publishers. This document is not updated after publication.

13.7 No investment, procurement, legal or tax advice

Nothing in this briefing constitutes investment advice, a recommendation to buy, sell or hold any security, or an offer or solicitation in respect of any financial instrument. Nothing in it constitutes procurement advice, a recommendation to select, retain or terminate any vendor, product or contract, or legal, regulatory, accounting or tax advice. References to the EU AI Act and other regulation are summaries of publicly reported analysis and are not legal opinions; the regulatory position described was itself subject to pending legislative change at the time of writing. Readers must take their own professional advice before acting.

13.8 No warranty; no liability

This document is provided strictly "as is", without warranty of any kind, express or implied, including any warranty of accuracy, completeness, merchantability, fitness for a particular purpose or non-infringement. To the fullest extent permitted by law, neither the author nor StratoFactory.com accepts any liability for any loss or damage, direct or indirect, arising from reliance on this briefing or on any third-party source cited in it.

13.9 Sharing, corrections, and independence

Readers are welcome to share this briefing, quote brief portions with attribution, and reference the analysis in their own commentary and strategy discussions; substantial reproduction requires permission. Corrections are welcomed and will be acknowledged, and where a cited source is shown to have been misread the correction will be published. The author and StratoFactory.com received no compensation from any vendor, advisory firm, analyst house or other organization in connection with this briefing, and hold no position that would be materially affected by its conclusions. Several sources cited are commercial parties whose interests are noted in the text.