Capgemini

Beyond vibe coding: Accelerated and scalable delivery of ***business applications*** in an agentic era

Why enterprise AI platforms define the next decade of application delivery

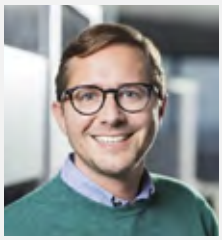
Vibe coding accelerates the build. Enterprise AI platforms decide whether what you build can scale, change, and govern the evolution over time.

content

Disclaimer

The information contained here is general in nature and is not intended, and should not be construed, as professional advice or opinion provided to the user. It is not legal advice. Capgemini is not a law firm, and we recommend that users seeking legal advice consult a lawyer. This document does not purport to be a complete statement of the approaches or steps necessary for a business to accomplish any particular business, legal, or regulatory goal, which vary according to individual factors and circumstances. The document is provided for informational purposes only. It is not a recommendation of any particular approach and should not be relied upon to address or solve any particular matter. The text of this document was originally written in English; any translation is provided as a convenience and Capgemini disclaims responsibility for translation inaccuracies. The information is provided on an as-is basis. Capgemini disclaims any and all representations and warranties of any kind concerning the information provided here and will not be liable for any direct, indirect, special, incidental, or consequential loss, or loss of profits, arising in any way from the information contained here. Where charts or tables do not cite an external source, the data is derived from Capgemini surveys, engagement experience, or internal analysis.

Authors



Stephan Kolarik

Vice President | Global DCX Customer Process Leader

Dinesh Karanam

Global Head of Customer Process Management and Contact Center Transformations | Financial Services

Niklas Jansson

Senior Director | CTO, Customer Process Management

Contributors

Shib Sankar Chakraborty

Global Pre-Sales Leader for Intelligent Automation and Enterprise Platforms, DCX, India

Ritesh Arora

Customer Process Management Leader for EMEA | Financial Services

Srimayee Majumdar

Customer Process Management Leader for APAC | Financial Services

01

Executive summary

Three forces are colliding inside enterprise IT, and most leadership teams are misinterpreting their convergence. Here is the short version.

Three forces reshaping delivery

AI adoption is now a race, not a choice. Gartner expects 40% of enterprise applications to embed task-specific AI agents by the end of 2026, up from under 5% in 2025. IDC puts worldwide AI and agentic AI IT spend on track to exceed $1.3 trillion by 2029, with agentic AI alone accounting for more than a quarter of all IT spend.

Platforms already proved their economics. Low-code has evolved beyond a category and become a default capability. Gartner expects 75% of new enterprise applications to be built on low-code platforms by 2026. The model-driven idea has proven durable, even as the label fades from analyst vocabulary.

Vibe coding changed the build. Natural language now turns into running software in hours rather than quarters. McKinsey finds that most organizations achieve at least a 25% productivity lift from AI in software work, and a Jellyfish-OpenAI study reports deep-adoption teams above 110%. The shift in speed is real and worth paying attention to.

The misconception

The temptation, especially in the boardroom, is to read this as a clean generational swap. Vibe coding absorbs low-code, agents absorb workflows, platforms fade. That reading is partially correct, but dangerously incomplete. Gartner is blunt that more than 40% of agentic AI projects will be cancelled by the end of 2027 due to cost, unclear value, and weak controls. The build velocity is real. The operational assurance is not. This is where most of the five-year cost resides.

The Capgemini thesis

Vibe coding and AI development environments are genuine accelerators across discovery, prototyping, scaffolding, and exploratory engineering. Their advantage narrows as work moves toward production, where determinism, audit, scaling, and continuous change dominate cost.

Enterprise AI platforms close that gap by combining deterministic process orchestration with agentic reasoning under a single governed control plane. Capgemini's position is that the winning move is not choosing between speed and control, but designing both into a single delivery model. The following sections provide evidence for this position.

Five executive conclusions

1. **Treat AI-generated code as a prototype, not a production asset.** Veracode found security flaws in 45% of AI-generated code, and the Cloud Security Alliance found that AI-assisted commits expose secrets at twice the rate of human commits. The discipline is simple: use vibe coding to move fast in design time, and never ship unreviewed AI-generated code into a regulated workload.
2. **Code generation was never the critical path.** AI returns 30 to 50% of effort in build, but less than 15% in operate and change. Roughly 70% of five-year total cost of ownership sits in the phases that code assistants barely reach.
3. **AI needs structure to deliver predictable ROI.** The winning pattern, now formalized by Forrester as the agent control plane and adaptive process orchestration, is a deterministic backbone with agentic reasoning at selected nodes.
4. **Model-driven platforms reduce long-term change cost.** Pure code generation, even when AI-accelerated, accrues technical debt unless it is anchored in a governed model. Over five years, platform-based delivery wins decisively for mission-critical work.
5. **The platform decision is now an economic decision.** Enterprises that route AI velocity through a governed platform get predictable cost, audit-grade governance, and change that stays additive. Leading enterprise AI orchestration platforms illustrate what a complete enterprise AI platform looks like; Capgemini's role is to make that pattern real inside each client's regulatory perimeter, on the workflows that carry the most risk and value.

If the question is how fast you can build, vibe coding wins. If the question is how fast you can scale and change safely at scale over five years, the platform wins. Most enterprise CIOs are making a five-year decision while being marketed a one-quarter narrative.

02

The two enterprise choices

In the agentic era, enterprise leaders are not really choosing between low-code, vibe coding, and agentic AI. They are making two decisions about how AI enters the enterprise, and over a five-year horizon those two decisions will shape cost, risk, and customer outcomes far more than any single tool selected this quarter.

Choice 1: How will the enterprise run AI at scale?

This is a run-time decision. AI is moving out of pilots and into live operations, where it touches customers, money, and regulated decisions. The question is whether that activity runs on a coherent platform or accumulates as a collection of independent agents that each behave on their own terms.

Path A: Homegrown or fragmented agent platforms	Path B: Orchestration-first enterprise AI platform
The result is agent sprawl, nondeterministic behavior, duplicated controls, and governance that is hard to predict or evidence.	AI agents operate inside deterministic workflows, with governed decisioning, auditability, and consistent customer outcomes.

Choice 2: How will the enterprise build AI-enabled systems?

This is a build-time decision. AI can write and assemble software faster than any prior tooling. The question is whether that speed produces systems the enterprise can maintain, or a backlog of artifacts that someone has to re-engineer before they are safe to run.

Path A: Build from scratch with AI-generated code and copilots	Path B: Accelerate transformation on a governed enterprise platform
This creates speed, but also potential vulnerabilities, inconsistency across components, compliance gaps, and long-term sustainment risk.	AI accelerates delivery on a governed platform, so the resulting systems stay maintainable, scalable, compliant, auditable, and easier to change over time.

The point is not low-code versus vibe coding versus agentic AI. The real question is whether the enterprise pursues AI acceleration through fragmented build and run models, or through a governed enterprise AI platform that combines speed with control.

Build-time and run-time AI: two dimensions, governed differently

The two choices above rest on a distinction worth making explicit, because most confusion in agentic strategy comes from collapsing it. AI shows up in two very different places in the enterprise, and each carries a different risk profile.

Build-time AI accelerates how systems are created

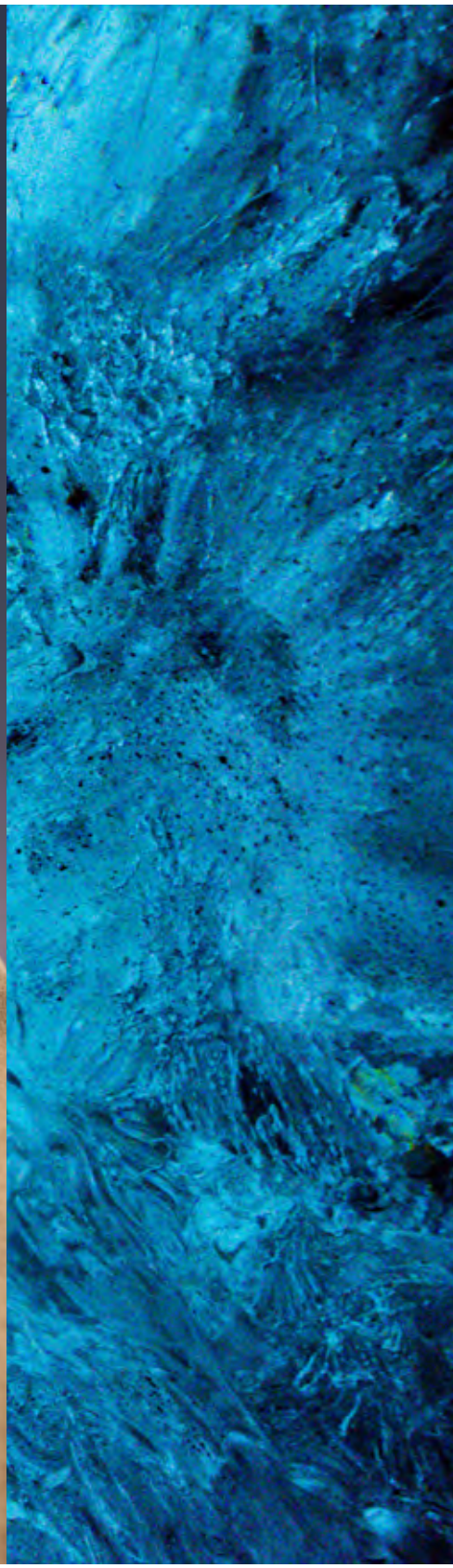
It speeds design, prototyping, requirements interpretation, scaffolding, test generation, documentation, and overall developer productivity. This is where AI-assisted development, coding copilots, and emerging "vibe coding" patterns operate. Its primary value is reducing time to the first working artifact and enabling faster, more efficient transformation.

Run-time AI changes how the business operates

It lives inside live operations as AI agents, decisioning, summarization, intent detection, recommendations, autonomous or semi-autonomous actions, and process orchestration. Its primary value is better customer outcomes, operational efficiency, faster decisions, and service execution at scale. It also demands a higher level of governance, because when run-time AI fails the consequences land on customers, compliance, financial outcomes, and brand trust.

Dimension	Build-time AI	Run-time AI
Where it acts	How systems are created	How the business operates
What it includes	Vibe coding, AI coding assistants, AI-assisted platform development, scaffolding, test and documentation generation, design and requirements support	AI agents, decisioning, summarization, intent detection, recommendations, autonomous and semi-autonomous actions, process orchestration
Primary value	Speed to first artifact and faster transformation	Better customer outcomes, operational efficiency, faster decisions, scalable service execution
Governance posture	Important, but failures mostly surface during development and review	Critical, because failures reach customers, compliance, financial outcomes, and brand trust

Enterprises need both. The mistake is governing them as if they were the same thing. Build-time AI can be reviewed before anything ships. Run-time AI acts on the business as it runs and has to be governed accordingly.



03

The lifecycle reset: Why this conversation matters now

Application delivery has gone through three eras in recent memory. The first was hand-coded enterprise software, dominated by Java, .NET, and the system-integrator labor model. The second was the low-code era, where graphical, model-driven environments shortened time-to-first-application from years to weeks. The third, which we are inside now, is the AI-native delivery era, where the interface to software creation is increasingly based on natural language and the runtime is increasingly autonomous.

Each transition triggered the same recurring debate. Will the new paradigm replace the old? In every prior case, the honest answer turned out to be no. It absorbed it. Low-code did not eliminate hand coding. It compressed the share of delivery that needed it. AI-native delivery is doing the same thing to low-code. Gartner's forecast supports this directly: low-code as a label is fading, but model-driven capability is appearing in more enterprise platforms than ever before.

Three signals worth paying attention to

Signal 1. The market math is enormous and unevenly distributed. The low-code platforms market is growing at roughly 16 to 19% CAGR. The AI and agentic AI segment is growing at roughly 32% and is on a trajectory to exceed $1.3 trillion in worldwide IT spend by 2029. The chart below puts both markets on the same axis. While both grow, they don't grow at the same scale.

Two markets, one architecture problem Low-code vs AI / agentic IT spend, 2023 - 2029

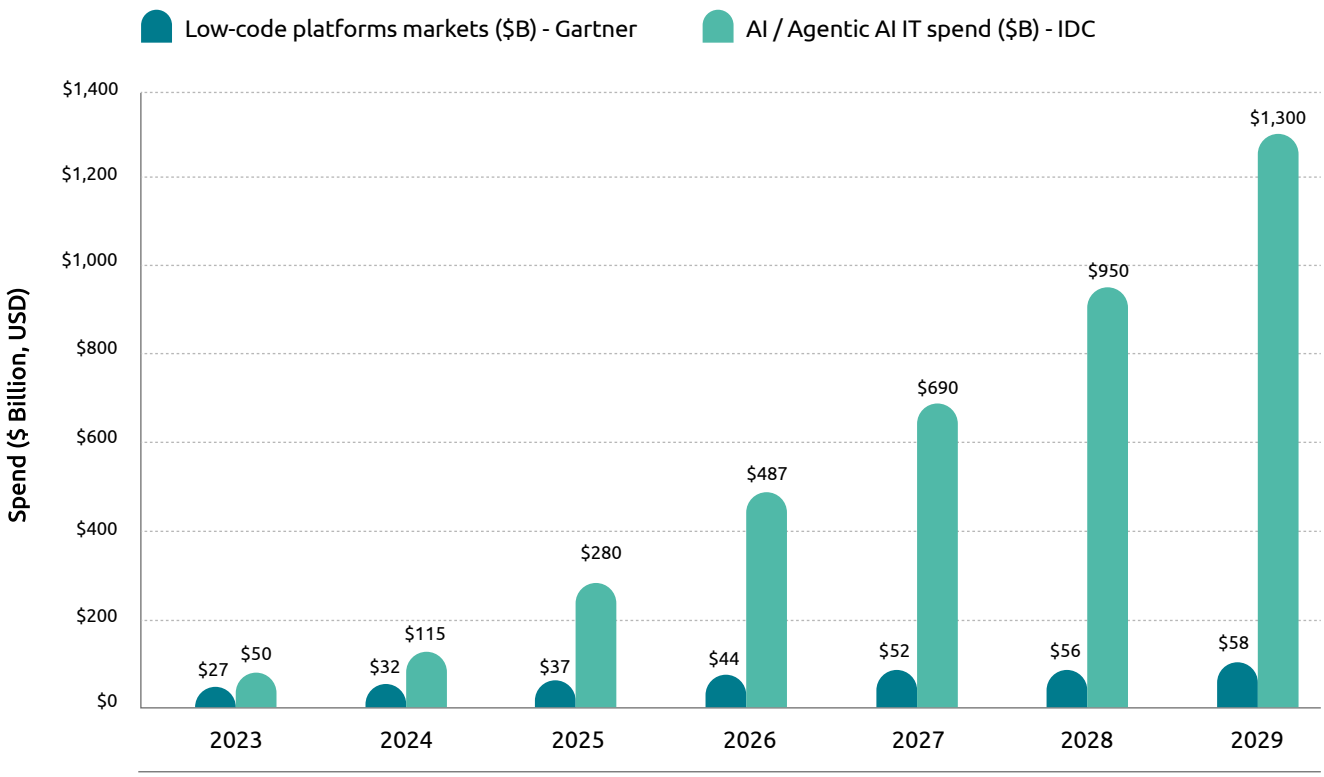


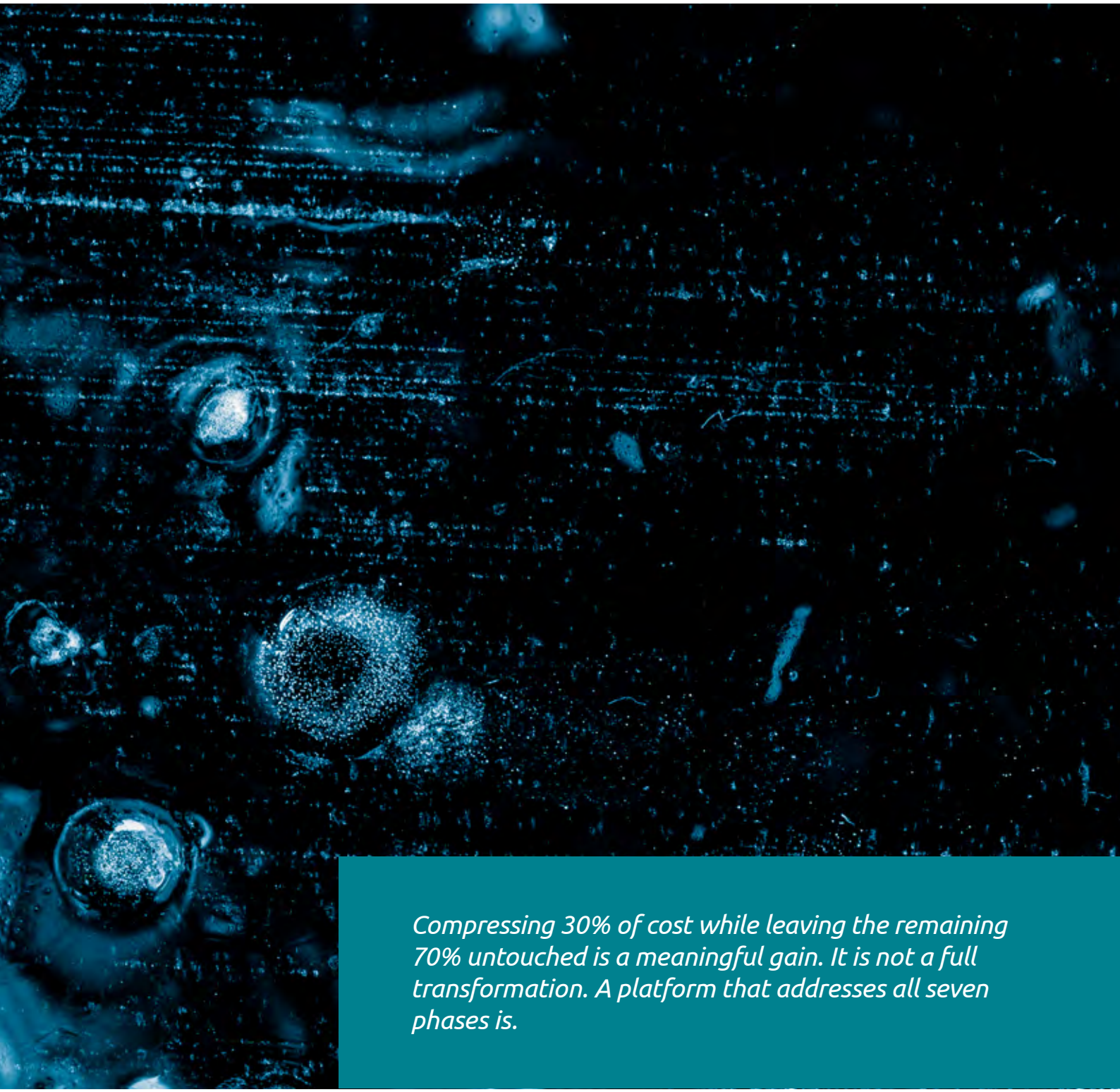
Figure 1. Low-code platforms versus AI and agentic IT spend, 2023 to 2029. Both markets grow; the agentic curve outpaces low-code by roughly 20x in absolute terms. Source: Gartner Forecast - Low-Code Development Technologies, Worldwide (2024); IDC AI IT Spending Forecast (Aug 2025).

Signal 2. Vibe coding has produced viral demos and a worrying production track record. Multiple external scans of deployed vibe-coded applications report a high incidence of exposed secrets, missing authentication, and unsafe defaults. Veracode's 2025 GenAI Code Security Report tested 100 leading large language models across 80 curated tasks and found that AI-generated code contains security vulnerabilities 45% of the time. A December 2025 Tenzai study tested 15 vibe-coded production applications across five major AI coding tools and found 69 vulnerabilities, with every single application missing CSRF protection and security headers.

Signal 3. Even AI-native organizations are doubling down on systems of record and orchestration. Forrester's Predictions 2026 report describes the AI bubble deflating and expects only a small proportion of organizations to currently tie AI initiatives to measurable impact. McKinsey's State of AI 2025 reports that 94% of organizations are not yet seeing significant value from AI investment, with only 5.5% seeing real financial returns. This is not a failure of the technology. It is a failure of the operating model around it

Why this matters for the lifecycle

If you look at the traditional enterprise application lifecycle as seven phases, discovery, design, build, test, deploy, operate, and change, vibe coding meaningfully compresses the first three. It barely touches the last four. Over a five-year life of an application, the last four phases account for roughly 70% of total cost. That is the lens through which the rest of this document is written.



Compressing 30% of cost while leaving the remaining 70% untouched is a meaningful gain. It is not a full transformation. A platform that addresses all seven phases is.

04

The end-to-end application lifecycle lens

This section walks through each phase of the application lifecycle in platform-neutral language. It compares the productivity contribution of vibe coding, AI coding assistants, and model-driven enterprise platforms.

Where AI delivers real productivity, and where it does not Effort reduction by lifecycle phase, %

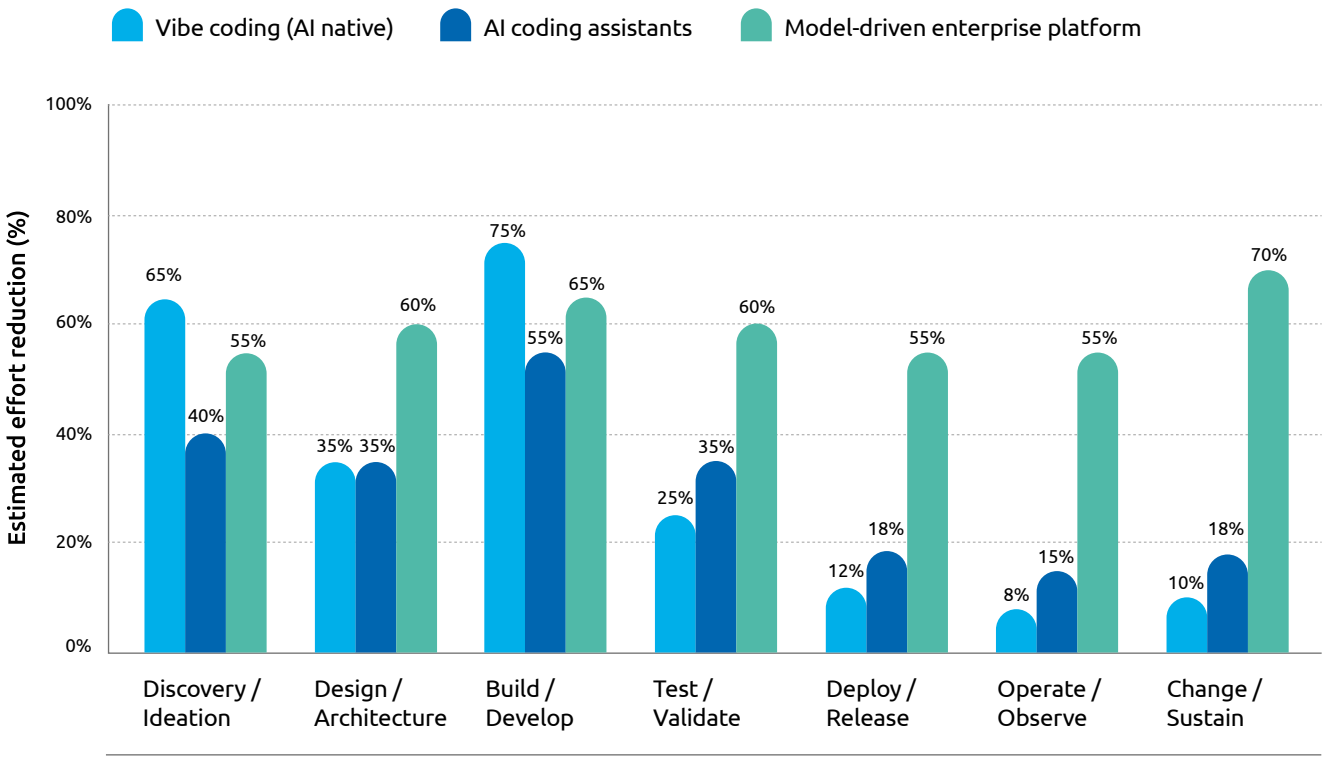


Figure 2. Effort reduction by lifecycle phase, by delivery approach. The shaded region on the right is where vibe coding productivity collapses. Source: Capgemini analysis based on McKinsey Developer Productivity studies (2023, updated 2025), the Jellyfish-OpenAI study (Aug 2025), and engagement evidence.

The pattern is consistent across engagement evidence and analyst data. AI-driven development gives back roughly 30 to 50% of effort across the early lifecycle. The remaining 60 to 70%, which dominates total cost of ownership, requires platform capability that vibe coding does not provide on its own.

Discovery and ideation

This is the phase where business intent gets translated into something an engineering team can act on. Historically, this phase has always been paper-heavy: requirements, user stories, process diagrams, and signoffs. Here's what's new. A business analyst can describe an idea in natural language and have a working prototype within hours. This shift is genuinely transformative for stakeholder alignment, because seeing a working artifact accelerates feedback by a factor of five to ten compared with document-based review. McKinsey's research shows generative AI can produce documentation in half the time, write new code in nearly half the time, and refactor existing code in nearly two-thirds the time. Where it falls short is depth. When the problem requires domain understanding, compliance constraints, or integration with multiple systems of record, generated artifacts miss critical context. That is the difference between a prototype that demonstrates the idea and a design that survives the journey to production.

Design and architecture

This is where business design becomes solution design: data model, integration topology, decisioning logic, security boundaries, identity model, and operating environment. It is also where vibe coding begins to exhibit structural limits. Generated applications tend to embed implicit dependencies, assume defaults that do not match enterprise reality, and rarely reflect the integration patterns required for systems of record. AI platforms with strong context engineering can suggest architectures from prior implementations, but they fall short on industry-specific regulatory constraints, which usually live in tribal knowledge rather than training data. Model-driven platforms win this phase decisively. The architecture is itself a model artifact, which means it can be reasoned about, version-controlled, and modified without rewriting the application beneath it.

Build and develop

This is where the application gets created: forms, processes, business rules, decisioning, integrations, UI, and automated tests. It is vibe coding's home turf. Productivity gains are real, particularly for greenfield internal tools without significant integration scope. The Jellyfish-OpenAI study of August 2025 found companies with 80 to 100% developer adoption of AI coding tools saw gains above 110%. For regulated, deeply integrated, mission-critical applications, the gains shrink due to the necessity for the platform to continue to handle integration, decisioning, identity, and case lifecycle.

Test and validate

This is where the application is tested for functional correctness, performance, security, compliance, and accessibility. Generated tests cover happy paths and basic boundary conditions. Edge cases, negative paths, performance under load, and regulatory test scenarios still need engineering effort. The Tenzai study of 15 vibe-coded production applications found that every single one lacked CSRF protection and security headers, and that all five tested tools introduced server-side request forgery vulnerabilities. The takeaway here is not that the tools are broken, but that test coverage does not automatically extend to the contextual security concerns that matter in production.

Deploy and release

This is where the application transitions from build to running in a controlled environment, including continuous integration and delivery, environment promotion, configuration management, and release governance. Deployment is largely outside the vibe coding loop. Generated applications still need to be packaged, deployed, configured, and integrated with existing pipelines and release governance. Some AI platforms provide hosted runtimes that simplify deployment for simple cases, but enterprise release management with environment promotion, approval gates, and rollback usually still requires platform-level engineering.

Operate and observe

This is where the application runs in production: performance, availability, security monitoring, incident response, capacity, and continuous evolution. It is where pure vibe coding tends to break down at enterprise scale. Operating discipline, observability, alerting, capacity planning, secure operations, and identity and access management require deliberate engineering that is largely orthogonal to how the application was built. Forrester's 2026 predictions describe a coming era where process intelligence will rescue up to 30% of failed AI initiatives by grounding automation in how work actually happens, not how it is documented.

Change and sustain

This is where the application evolves over its productive life: new requirements, regulatory

change, integration change, performance tuning, experience refresh, and eventually decommissioning. Change in a vibe-coded application is the largest open question in the industry today. Re-prompting the system may produce a new version, but tracking what changed, why, and whether it is safe to deploy is still an active research area. Based on current evidence, production vibe-coded applications in regulated enterprise environments tend to transition into traditional engineering models for sustainment, though tooling for in-place AI-anchored change is improving. This is exactly the phase where model-driven platforms have a structural advantage: because the application is a model rather than a body of code, change is largely additive.

The end-to-end picture is consistent across the evidence we have today. Vibe coding tends to win phases 1 through 3, where idea-to-artifact velocity dominates. In regulated enterprise environments, governed platforms tend to win phases 4 through 7, where audit, change cost, and operational assurance dominate. Most enterprise total cost of ownership lives in phases 4 through 7.

05

Vibe coding: Promise, reality, and the production gap

The productivity story around vibe coding is real, and enterprises should capture it. Used well, it compresses the distance from idea to working software in a way no prior approach has, and that speed has genuine value across the early lifecycle. The risk story is less understood, partly because the field is new and partly because the people who have lived through the operational consequences are not the loudest voices in the discourse. This section quantifies the gap between the two, so leaders can capture the speed without inheriting the exposure.

The quantified production-readiness gap

The data is now consistent across multiple independent studies and security firms. The chart below summarizes the picture.

The production-readiness gap Quantified risk across AI-generated and agentic code

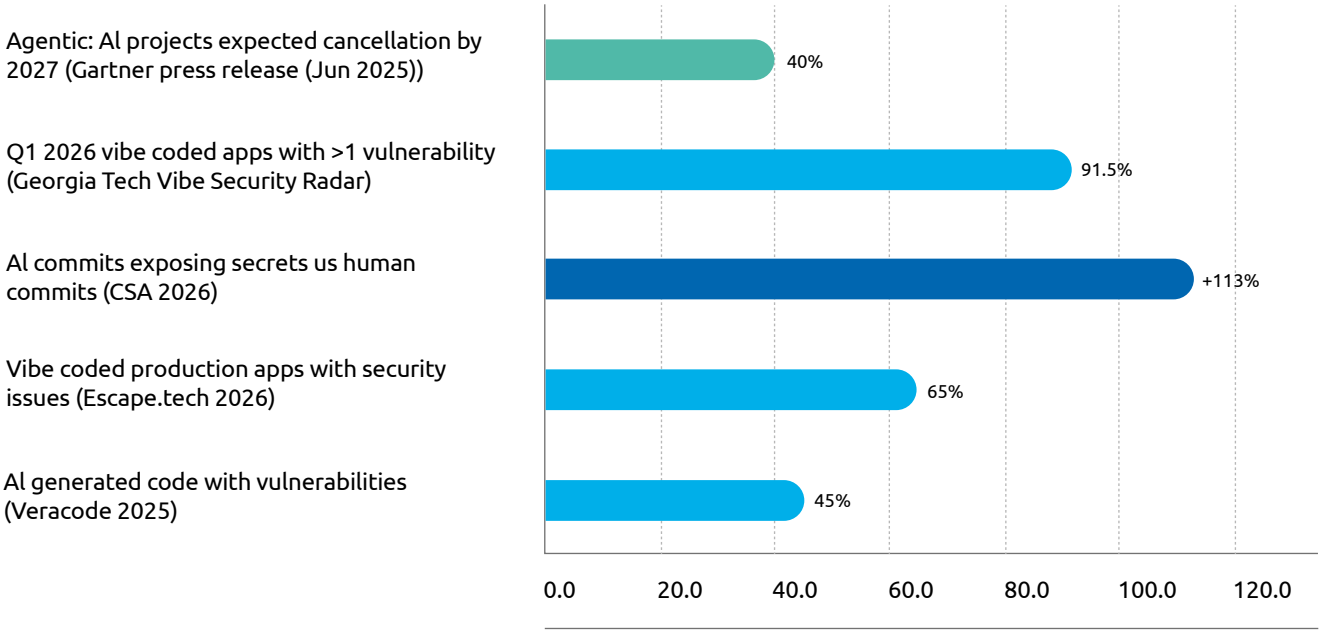


Figure 3. Quantified risk across AI-generated and agentic code. Each data point is drawn from a public study or analyst publication, 2025 to Q1 2026. Sources: Veracode GenAI Code Security 2025; Escape.tech (Mar 2026); Cloud Security Alliance Research Note (Mar 2026); Georgia Tech Vibe Security Radar (Q1 2026); Gartner press release (Jun 2025).

What the major studies found

Veracode GenAI Code Security Report 2025: This report tested 100 leading large language models across 80 curated tasks. It found that AI-generated code contains security vulnerabilities 45% of the time. The study found models failed to secure code against highly common attacks such as cross-site scripting and log injection in 86% and 88% of cases respectively, with no real improvement across newer or larger models.

Tenzai 15-application study, December 2025: This study compared five major vibe coding tools by using predefined prompts to build the same test applications. Found 69 vulnerabilities across 15 applications. Every application lacked CSRF protection and security headers, and each tool introduced server-side request forgery vulnerabilities. A clean sweep of basic security failures.

Escape.tech production scan, Q1 2026: Escape scanned more than 1,400 vibe-coded production applications and found that 65% had security issues and 58% contained at least one critical

vulnerability. Across a broader sample of 5,600 applications, researchers found over 2,000 vulnerabilities, more than 400 exposed secrets, and 175 instances of exposed personal data.

Cloud Security Alliance Research Note, March 2026: The Cloud Security Alliance found that AI-assisted commits expose secrets at more than twice the rate of human-only commits, 3.2% versus 1.5%. Public code repositories saw a 34% year-over-year increase in hard-coded credentials discovered in 2025, the largest single-year jump on record. The OWASP Top 10 for LLM Applications 2025 specifically called out overreliance and sensitive information disclosure as emerging top-tier risks.

Georgia Tech Vibe Security Radar, Q1 2026: This assessment of more than 200 vibe-coded applications found 91.5% contained at least one vulnerability traceable to AI hallucination or unsafe defaults. Georgia Tech estimates the actual figure of disclosed vulnerabilities from AI-generated code is five to ten times higher than what is currently being detected.

Where the costs hide

- **Security debt:** External scans of deployed applications repeatedly find a meaningful incidence of hard-coded secrets, missing or weak authentication, and unsafe default configuration. Speed of generation has not been matched by speed of security review.
- **Architectural drift:** Generated code is often functionally correct but architecturally inconsistent across components. Two prompts for similar functionality can produce structurally different implementations – increasing the difficulty of estate-wide refactoring.
- **Regression fragility:** Many vibe-coded applications regress when underlying model or framework versions change. Without an anchored model representation, the implementation surface is the source of truth, and small changes break in opaque ways.
- **Reproducibility gap:** Reproducing application behavior in a different environment, or auditing why a specific output occurred, is harder when the artifact is generated rather than configured.
- **Knowledge concentration:** When the original prompt author leaves, replacing that individual is more difficult than replacing a developer with a documented codebase. The prompt history and tacit understanding often live only with that one person.
- **Platform-layer vulnerabilities:** Vibe coding platforms are themselves attack surfaces. The Base44 platform vulnerability allowed unauthenticated attackers to access any private application built on it. CVE-2025-54135 allowed remote code execution on Cursor users' machines. CVE-2025-55284 allowed data exfiltration from Claude Code through DNS requests. These are disclosed vulnerabilities, not theoretical ones.

The Lovable case study, a structural warning

Lovable is the fastest-growing software startup in history by several measures: $4 million of annual recurring revenue in its first four weeks, $10 million in two months, $200 million raised at a $1.8-billion valuation in July 2025, then $330 million at $6.6 billion valuation in December 2025. It is also the company that left thousands of customer projects exposed for 48 days before patching, then publicly denied a breach, blamed its documentation and bug-bounty partner, and apologized for the apology. Lovable is not uniquely insecure – it is representatively insecure. The structural incentives of the category reward growth and feature velocity over governance and security review. The lesson is not to avoid these tools, which are genuinely useful, but to put a governed operating model around them before production. That is exactly the gap the rest of this paper addresses.

Where the picture will likely improve

The above evidence describes the state of the field today. However, it does not dictate its ceiling. AI-native engineering tooling is on a steep maturity curve, and several developments are already visible. Secure-by-default code generation is improving across newer model releases, particularly on reasoning models where security pass rates have moved measurably. SDLC integrations are emerging that wrap AI-generated code in automated security scanning, secret detection, and policy enforcement before it reaches a branch. Hybrid development models, where AI assistance is bounded by a governed model rather than free-form code emission, are becoming the architecturally preferred pattern across enterprise platforms. It is also worth noting that traditional enterprise software engineering produced its own long catalog of security failures over thirty years – ranging from buffer overflows to authentication bypasses to credential mismanagement – that the industry only addressed through deliberate process discipline rather than tooling alone. Vibe coding is at the beginning of a similar discipline curve. This section doesn't argue that these tools are unsafe by nature, but that the operating model around them has not yet caught up with the velocity they enable. That gap will narrow. In the meantime, treating AI-generated code as a high-fidelity prototype that needs deliberate review before production is the prudent posture for regulated work.

The implication for enterprise leaders

Vibe coding is best treated as an accelerator inside a governed delivery process, not as a standalone delivery model. What it does well is collapsing the time from idea to first working artifact. What it does poorly is sustaining that artifact across years of regulatory change, integration churn, and personnel turnover.

Treat vibe-coded artifacts the way you would treat a high-fidelity prototype produced by a vendor: useful for alignment and validation, expected to be rebuilt or substantially re-engineered before production, and never deployed into a regulated workload without a governed platform underneath.

06

Agentic AI: Maturity gap and the control plane imperative

Agentic AI is on the most aggressive adoption curve in enterprise technology since the public cloud. It is also a market where analyst data and operational reality have begun to visibly diverge. Both are useful inputs to strategy.

The adoption curve

Enterprise agentic AI adoption curve
Deployment, pilot, and embedded-app penetration, 2024 - 2030

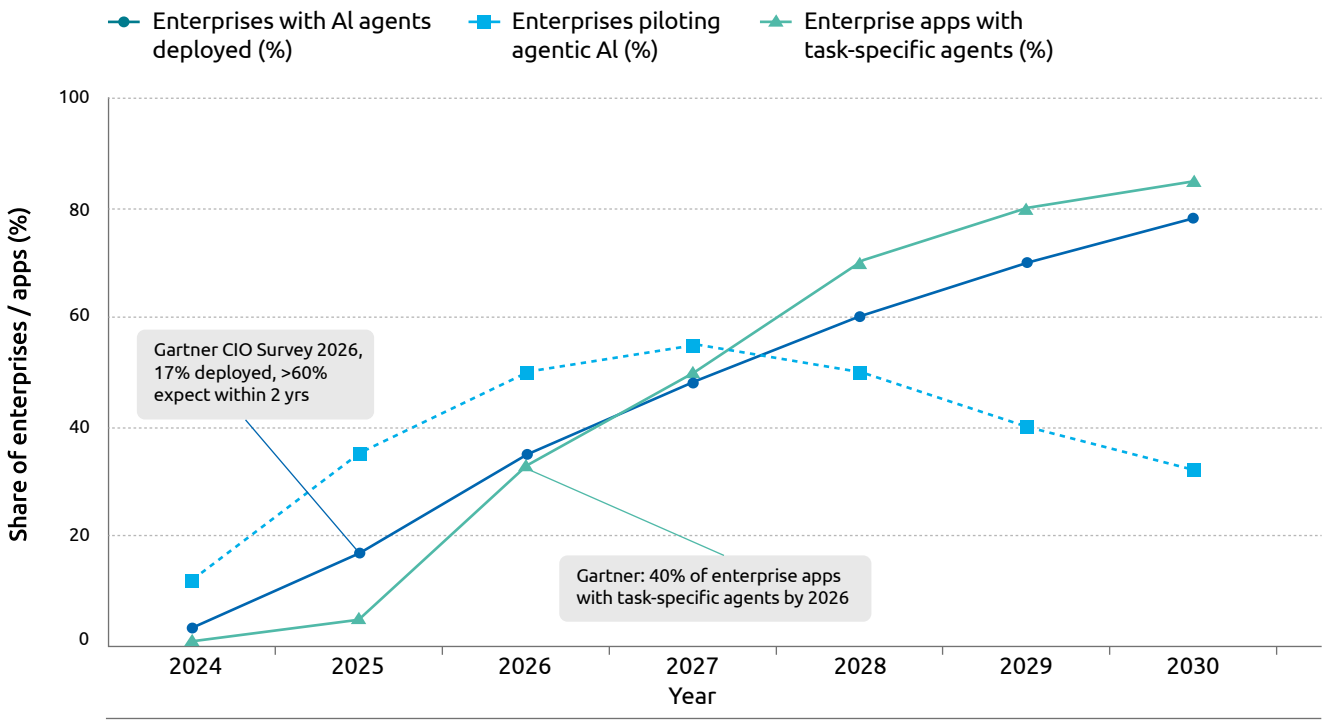


Figure 4. Enterprise agentic AI adoption: deployment, pilots, and embedded-app penetration through 2030. Sources: 2026 Gartner Hype Cycle for Agentic AI; Gartner CIO Survey 2026; Gartner press release (Aug 2025); IDC FutureScape 2026. Figures for 2027 to 2030 are extrapolated.

The numbers are striking. According to Gartner's August 2025 forecast: 40% of enterprise applications will be integrated with task-specific AI agents by the end of 2026, up from less than 5% in 2025. The 2026 Gartner CIO and Technology Executive Survey found only 17% of organizations have deployed AI agents to date, yet more than 60% expect to do so within the next two years. By 2028, Gartner expects at least 15% of day-to-day work decisions to be made autonomously through agentic AI. By 2029, IDC predicts 70% of enterprises will deploy agentic AI as part of IT infrastructure operations. By 2035, Gartner's best-case projection has agentic AI driving roughly 30% of enterprise application software revenue, surpassing $450 billion.

The dose of reality

Gartner's own analysis, June 2025: more than 40% of agentic AI projects will be canceled by the end of 2027, driven by escalating cost, unclear business value, or inadequate risk controls. The 2026 Hype Cycle for Agentic AI explicitly places agentic AI at the Peak of Inflated Expectations. This is not the analysts hedging, it is a structural property of the category. Agents are fundamentally nondeterministic. The same inputs can produce different outputs across runs. Though acceptable in creative and exploratory contexts, it is rapidly unacceptable as work moves toward regulated decisions, customer-impacting actions, or actions on real money. McKinsey and OpenAI's joint research on developer productivity reaches the same conclusion from a different angle; giving developers AI tools does not, by itself, move the needle. The companies that unlock real value are the ones that re-architect how they build software and embed AI across the entire lifecycle, not just for coding.

The control plane imperative

The most important architectural development of the past twelve months is the formal emergence of the agent control plane as a distinct enterprise capability. Forrester announced its evaluation of the agent control plane market in December 2025 and defined it precisely: an enterprise control plane that inventories, governs, orchestrates, and assures heterogeneous AI agents across vendors and domains. It plays a similar role similar to the control plane found in cloud or network architecture, sitting outside both the build and the orchestration layers.

The three-plane enterprise architecture

Build plane x Orchestration plane x Agent control plane (Forrester, 2025 - 2026)

AGENT CONTROL PLANE (governance, identity, audit, FinOps for agents)

- Inventory and registry of agents across vendors and domains
- Policy enforcement, identity propagation, observability, cost attribution
- Vendor-agnostic, sits outside both build and orchestration

ORCHESTRATION PLANE (adaptive process orchestration)

- Deterministic and agentic flows under a single state machine
- Case management, decisioning, SLAs, audit trails, human-in-the-loop
- Forrester APO category; this is where leading adaptive process orchestration platforms sit

BUILD PLANE (AI platforms, agent frameworks, vibe coding tools)

- Model access, agent frameworks, tool integration, evaluation pipelines
- Where vibe coding and AI coding assistants accelerate prototypes
- Output is artifacts (code or designs), not governed runtime

Figure 5. The three-plane enterprise architecture: build, orchestration, and agent control plane, as defined by Forrester research, 2025 to 2026.

- **Build plane:** Provides model access, agent frameworks, tool integration, vector stores, evaluation pipelines, and the surrounding infrastructure to create agents. This is where vibe coding tools, AI coding assistants, and agent frameworks sit.
- **Orchestration plane:** Expressed by what Forrester now calls Adaptive Process Orchestration. This plane models processes, defines routing and decision logic, composes integrations, and ensures tasks, data, and actions move through a process in a controlled, observable sequence. This is where Enterprise Platforms, the broader BPM-derived market, and the new orchestration landscape category sit.
- **Agent control plane:** This plane supervises the entire heterogeneous agent landscape in a vendor-agnostic way: identity propagation, policy enforcement, cost attribution, observability, and lifecycle. The Forrester thesis is that this plane must sit outside the build and orchestration environments, because governance applied inside the same layer that runs the agents creates a structural conflict of interest.

What this means for strategy

Three things follow from this architecture taking hold.

1. **Vendor consolidation will favor platforms that already span build and orchestration.** The control plane category will mature quickly, but enterprises that already have a governed orchestration layer will integrate control planes more cleanly than those that do not.
2. **The orchestration plane is the new center of gravity.** It is where determinism, decisioning, audit, and service-level agreements are enforced, and where agents are invoked with bounded scope and structured inputs and outputs. That is the analyst consensus, and increasingly the consensus in enterprise architecture itself.
3. **Agent governance becomes a board-level concept within three years.** Forrester's AEGIS framework, released in 2026, treats agent governance as a peer to identity, observability, and data security. Boards already ask who can demonstrate that an agent took a defensible action in a regulated context. That question is answered by the control plane, or it is not answered at all.

The protocol layer is emerging fast. Two standards have moved from research papers into production architecture inside twelve months: model context protocol (MCP), for exposing tools and processes to AI agents, and agent-to-agent (A2A), for delegation between agents. Together, they are becoming the wiring of the control plane.

The strategic implication is direct. Enterprise platforms that publish their workflows as governed MCP endpoints turn the orchestration plane into a callable surface for any external agent, on any vendor stack, under enterprise identity and audit. Those that do not will find themselves on the wrong side of where agentic traffic actually flows.

If you want to know which platforms are positioned for the next five years, ask which ones already have audit trails, decisioning, case management, and now agentic process orchestration as native concepts. Agents can be added to those backbones. Backbones cannot be retrofitted onto agent frameworks without rebuilding them.

07

The Capgemini view: Customer first in the agentic era

Customer first organizations continuously work to improve their value propositions and customer journeys. In the agentic era, the differentiator is the ability to sense, decide, and act quickly across channels and functions without sacrificing trust. Traditional transformation, with workshops to document as-is processes and incrementally design to-be flows, remains useful but is no longer sufficient on its own. New AI-powered tools change how organizations jointly blueprint the way they will run the business. What matters most is that leaders reimagine customer outcomes end-to-end, then decide where autonomy is appropriate, where human judgment must remain, and how the organization will govern both.

We define agentic process automation as the transformation of customer-experience operations through autonomous, reasoning, goal-oriented AI agents working in concert with rule-based automation and human specialists. It does not replace established process

digitization or automation. Instead, it extends them by adding a new mode of interaction and by delegating selected decisions and orchestration steps to agents under explicit guardrails, approvals, and auditability. The core leadership question is no longer can we build an agent, but can we delegate decisions safely at scale.

Our vision of the new enterprise architecture

After early experimentation, clients are converging on what they call agentic control planes: governance structures and technical layers that oversee customer flows across front, middle, and back office, ensuring auditability, recoverability, and accountable decision making. The Capgemini Agentic Enterprise Stack below is how we describe the target architecture. It is the glue that breaks organizational and IT silos, with the semantic layer and the agent control plane as its vital organs.

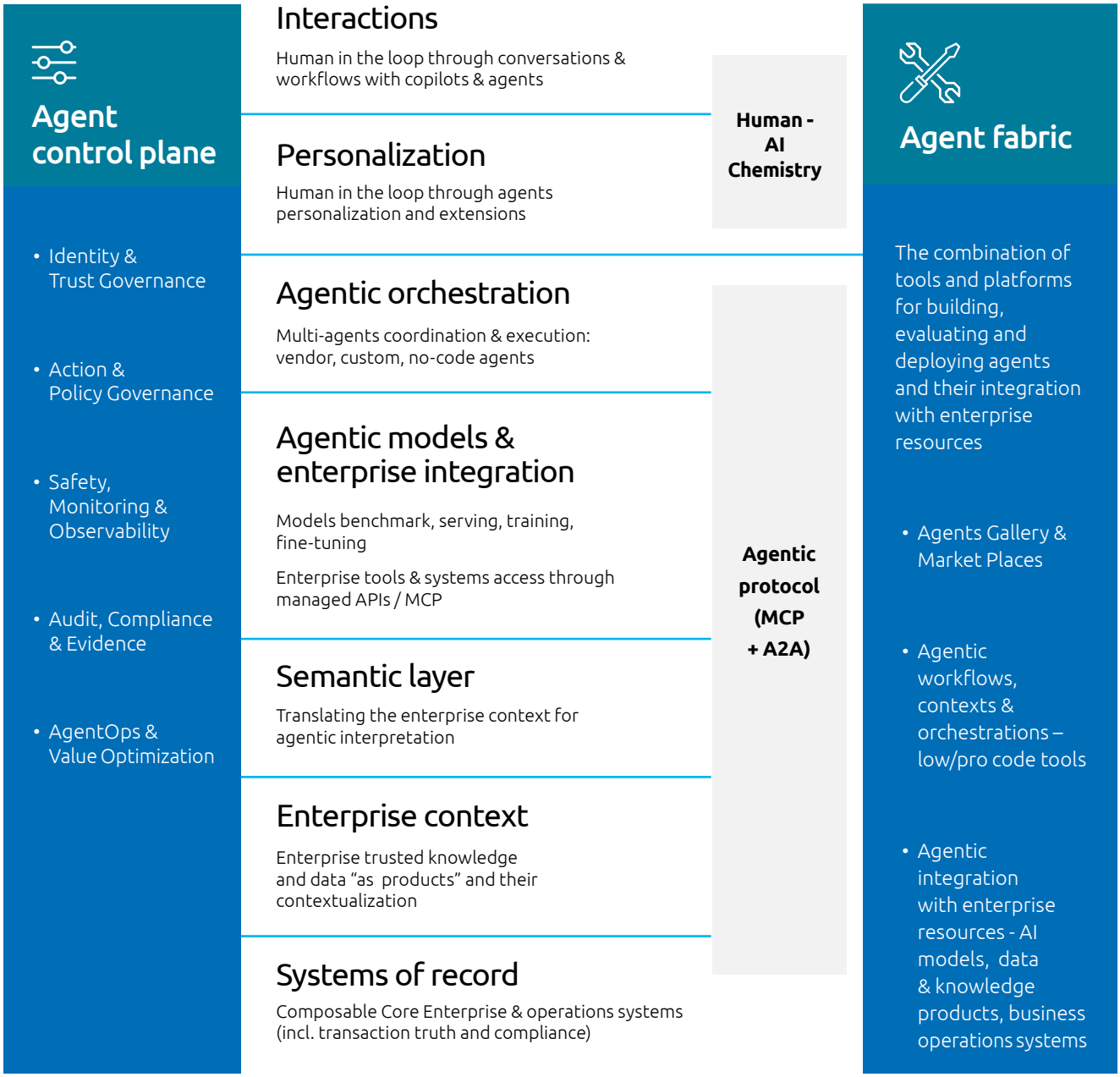


Figure 6. The Capgemini Agentic Enterprise Stack. Seven layers, with a vendor-agnostic agent control plane on one side and an agentic protocol layer (MCP and A2A) on the other.

Read the stack from the bottom up. Systems of record hold the transaction truth. Enterprise context and the semantic layer translate business logic and KPIs into something agents can interpret reliably. The build plane provides models, tools, and connectors. The orchestration plane coordinates multiple agents. The tools and interaction layers are where low-code and vibe-code tooling, copilots, and conversational channels live. The agent control plane runs alongside all of it, handling identity, policy, observability, audit, and value optimization. The agentic protocol layer, MCP and A2A, is the wiring that lets agents, tools, and workflows call one another. The point of the picture is simple: vibe coding and low-code live in the upper layers, determinism and governance live in the lower layers, and the control plane plus the protocol layer are what hold the two together.

Enterprise orchestration and the key enablers

Despite the agentic enthusiasm, defined business processes and workflows remain essential, particularly in regulated environments. What changes is the execution model. Alongside humans and deterministic automation, agents now contribute to outcomes by coordinating decisions and actions across systems within defined boundaries. Agentic process automation combines goal-oriented agents with established workflow automation, rules engines, and robotic process automation. Agents interpret intent, assemble context, recommend options, and execute actions where permitted, while human specialists retain authority for high-risk decisions and exceptions.

Three enablers are foundational to delivering customer-first outcomes with agentic automation. Personalized conversations mean tailored recommendations and next-best actions based on customer context and intent, real-time sentiment and intent signals mean interactions that adapt to customer emotion and service-risk indicators, and 24/7 intelligent support means always-on self-service and assisted service with consistent knowledge and escalation paths. For safety and compliance, agents need explicit guardrails: policies, thresholds, approvals, and audit trails. The design goal is safe autonomy – automation that is explainable, recoverable, and governed through human oversight.

The orchestration maturity curve

Organizations do not move directly from basic automation to full autonomy. The journey requires a deliberate evolution of orchestration capability, from rule-based task automation toward agent-driven, policy-aware decisioning at enterprise scale. Each stage builds on the previous one, expanding scope, intelligence, and governance.

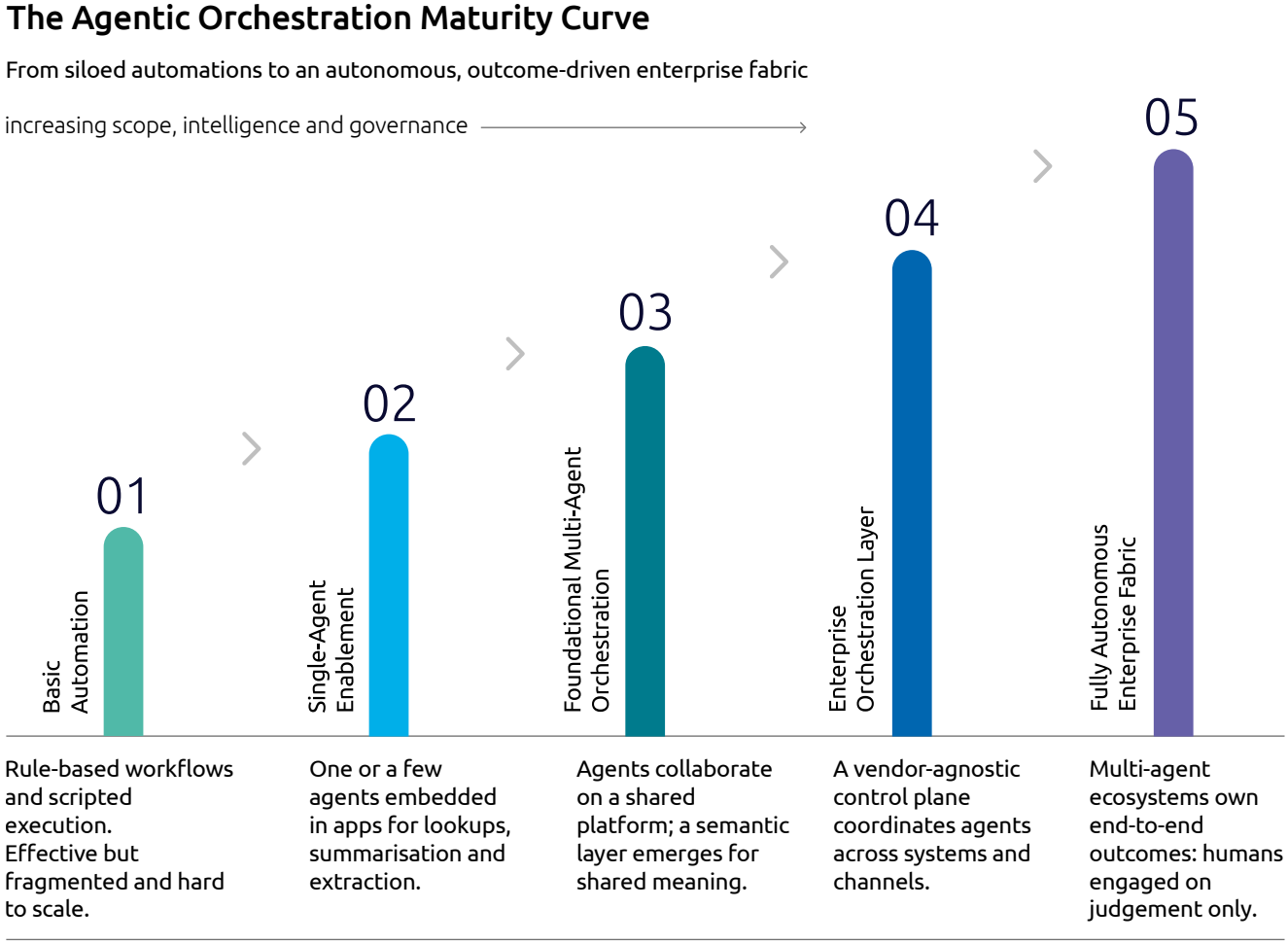


Figure 7. The agentic orchestration maturity curve. Five stages from basic automation to a fully autonomous enterprise fabric.

At the foundation, basic automation focuses on rule-based workflows and scripted execution. It is effective for individual processes but manually coordinated, fragmented across systems, and difficult to scale. Single-agent enablement embeds one or a few agents into applications for localized execution, such as lookups, summarization, or document extraction. Intelligence accelerates individual tasks but stays isolated. Foundational multi-agent orchestration brings multiple agents together on a shared platform, coordinated by a basic orchestration layer, with a semantic layer providing shared meaning. The enterprise orchestration layer becomes the control plane for intelligent operations: vendor-agnostic, coordinating agents across systems, platforms, and channels, with shared semantics, standard protocols, and central registries. The most advanced stage, the fully autonomous enterprise fabric, has multi-agent ecosystems operating across channels with end-to-end accountability for well-defined outcomes, where humans are engaged only when judgment or escalation is required.

The core insight: Autonomous agentic automation is not achieved by adding intelligence to workflows. It is achieved by establishing an enterprise-grade orchestration fabric that governs how agents reason, collaborate, and act at scale.

The Capgemini value proposition

Capgemini helps organizations operationalize agentic automation through an approach that combines domain expertise, integration capability, and industrialized delivery with AI governance and strong technology partnerships. We combine these to make agentic enterprise orchestration real, with an absolute focus on value creation. Our value proposition rests on five pillars. Underpinning all five are deep partnerships with leading enterprise platform providers; the platform supplies the engine, and Capgemini makes it run inside the client's regulatory perimeter and business reality.

1. **Domain-led transformation:** we start from the business process, leveraging our industry and process expertise across banking, insurance, telecommunications, and the public sector to find the high-impact decisions and set responsible autonomy boundaries.
2. **Enterprise orchestration:** we integrate agents with workflows, rules, APIs, and systems of record so automation runs end to end, not as a set of disconnected point solutions.
3. **Trust and governance by design:** control-plane patterns, auditability, human override, and compliance-ready evidence trails are built in from the start, so agentic capability arrives inside a governed structure rather than getting retrofitted around it later.
4. **Industrialized delivery:** accelerators, reusable process assets, and a phased rollout from pilot to scale turn a single proof case into a repeatable program with measurable outcomes, instead of a rebuild each time.
5. **Operating-model enablement:** change management, reskilling, and new roles embed the capability in how teams actually work and how performance is judged, so adoption is sustained at enterprise scale.

The Capgemini Value Proposition for Agentic Process Automation

Five pillars that turn agentic ambition into governed enterprise outcomes

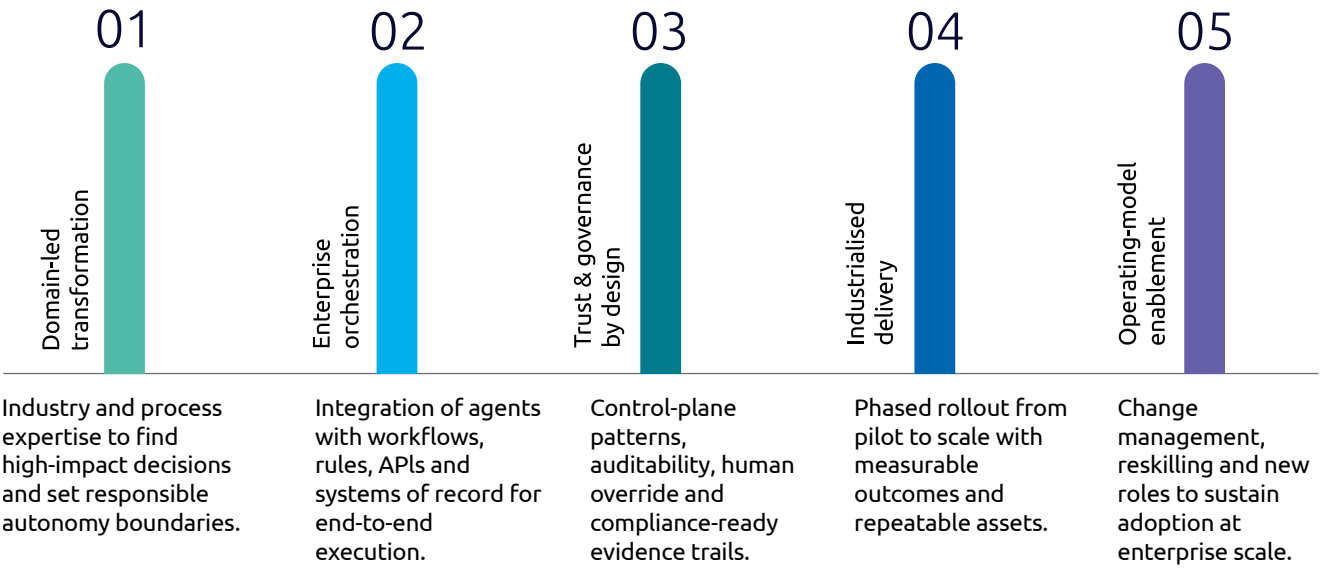


Figure 8. The Capgemini value proposition for Agentic Process Automation: five pillars from domain-led transformation to operating-model enablement.

08

The hybrid architecture as the scalable pattern

The replacement narrative is appealing because it is simple. A new paradigm arrives, the old one becomes obsolete, the organization migrates. In reality, every major IT transition of the past three decades has been an absorption rather than a replacement. Mainframes did not disappear when client-server arrived. Client-server did not disappear when web architectures arrived. On-premises systems did not disappear when SaaS arrived. They became one component in a larger architecture.

Agentic AI and vibe coding are following the same pattern. The scalable architecture is hybrid, and the analyst consensus has now formalized it. Gartner's October 2025 announcement on the Business Orchestration and Automation Technology category states it directly: by 2030, 70% of enterprises will pivot to a consolidated automation platform that orchestrates business processes, AI agents, bots, APIs, and human actions, up from 5% today.

The decision framework

When deciding which delivery model applies to a given application, three questions narrow the choice quickly.

Question	If yes	Implication
Will this application execute regulated decisions or carry direct customer impact?	Lean toward platform-based orchestration with agents at selected nodes	Audit, identity, and decisioning explainability are non-negotiable. Generate the artifact inside a governed model.
Can the decision logic be fully enumerated in advance?	The application is a process. Use process orchestration as the backbone	Determinism wins where the decision space is bounded. Agentic reasoning adds value at the leaves, not the trunk.
Will this application live for more than three years?	Model-driven platforms dominate on total cost of ownership	Sustainment cost dominates over five years. Architecture that supports additive variation beats architecture that requires regeneration.

Where each delivery model belongs

Delivery model	Best fit	Risk if used outside fit
Vibe coding	Prototyping, internal tooling, scaffolding, design-time exploration, throwaway pilots, rapid alignment with business stakeholders	Security debt, sustainment failure, regulatory exposure
AI coding assistants	Engineering productivity inside a governed delivery process: test scaffolding, documentation, code review, refactor	Productivity decay if treated as a substitute for engineering discipline
Agentic AI platforms (build plane)	Reasoning nodes inside larger workflows: document understanding, summarization, recommendation, intent and exception triage	Nondeterministic output where determinism is required; ungoverned agent sprawl

Where each delivery model belongs

Delivery model	Best fit	Risk if used outside fit
Adaptive process orchestration (orchestration plane)	Production case management, regulated workflows, mission-critical orchestration, long-lived applications, multi-agent coordination	Slower for one-off prototypes if used without the platform's design-time AI assistance.
Agent control plane	Cross-vendor, cross-domain governance of heterogeneous agent estates	Premature investment if the agent estate is still small and homogeneous
Pure custom code	Niche performance-critical components and integrations not covered by a platform	Slow build, high sustainment cost, weak governance unless deliberately engineered

The hybrid pattern in practice

The hybrid pattern is not a compromise. It is the architecture that has consistently won at enterprise scale across every wave of IT modernization, because it lets each layer do what it does best.

- **Systems of record remain the transaction truth:** core platforms that hold canonical state, ERP, core banking, policy administration, claims, customer master, stay governed, deterministic, and audit-anchored.
- **Adaptive process orchestration coordinates work:** it routes work between systems of record, applies decisioning, manages service-level agreements, owns audit trails, and routes between humans, bots, and agents.
- **Agentic AI sits at reasoning nodes inside processes:** Agents are invoked with bounded scope and structured inputs and outputs, under the platform's identity, audit, and governance perimeter, with outputs converted into structured signals the deterministic orchestration can route and govern. In 2026, the protocol question is settling on MCP for tool exposure and A2A for agent-to-agent delegation.
- **The agent control plane provides vendor-agnostic governance:** it inventories agents across vendors and domains, propagates identity, enforces policy, observes behavior, and attributes cost.

The practical implication is that hybrid is the operating architecture, not the compromise position. Build-plane velocity, orchestration-plane determinism, and a control plane that supervises both belong together. Treating them as competing categories rather than as complementary layers is the source of most of the failed agentic AI projects analysts are now writing about.

09

Enterprise AI economics: From token cost to cost per outcome

Most architecture debates end the moment someone asks what it costs to run. For a growing number of CIOs, enterprise AI economics is the conversation, ahead of any reference diagram. The questions are blunt and operational. What does one AI-assisted interaction cost? What does an agent cost to run for a year? What does a business outcome cost once you net out the model calls behind it? A clear economic answer earns more executive attention than a clean architecture picture, and the two are more connected than they look. The cost for tokens is becoming increasingly complex to understand fully and as adoption increases the topic of token costs are becoming increasingly important.

AI-cost at build-time risks becoming a shocking surprise

As elaborated on earlier, AI can accelerate design and shorten build times. At the same time, the cost for the tokens used at build-time risks

becoming a shocking surprise. The need for larger context windows for complex applications and multiple agents collaborating on the build might consume your token budget quicker than you initially had planned for. Modern agentic workflows solve a task with a team of agents using their own context windows and parallel executions leading to token consumption at exponential rate. Token usage at build is also highly dependent on the skills and experience of the human operator working in collaboration with the build-agents and a small shift in initial conditions, prompt design, and model choice can end up making a big difference in token costs.

What actually drives the run-cost of AI

Token consumption is not only a build-time cost. Once a system is live, every model call becomes a run-time operating cost that recurs with volume. Multi-agent designs compound the effect, because agents call models, tools, workflows, and other agents repeatedly, and each hop adds tokens. Large context windows, retrieval-augmented generation, retries, continuous evaluation, and monitoring all push the number higher. Custom agentic architectures tend to produce run costs that are hard to forecast, because spend grows with prompts, context, model calls, and interactions that nobody fully controls. The spending does not stop when the system ships. That's when it starts.

The metrics that now matter

Boards are starting to govern AI usage the way they govern any other consumption, which means by measuring it properly. Three numbers are becoming standard. Cost per interaction tells you what a single AI-assisted step costs at the point of use. Cost per agent tells you what an autonomous worker costs to keep running, including the calls it makes on your behalf. Cost per outcome tells you what it costs to actually close a case, settle a claim, or resolve a dispute, which is the figure the business truly cares about. Around these sits a discipline quickly earning its own name, FinOps for AI: cost attribution by team and use case, usage limits, policy controls, auditability, and continuous optimization. The point is to treat AI spend as a managed line item rather than a surprise on next quarter's cloud bill.

Why architecture becomes an economic decision

Here is the connection. The architecture argued throughout this paper is the same one that makes AI spend easier to attribute and contain. Deterministic orchestration reserves token-consuming reasoning for the steps that genuinely need judgment, and relies on workflow, rules, and decisioning where deterministic execution is cheaper and more reliable. Fewer unnecessary model calls mean lower and more predictable cost. Model-driven generation creates governed rules instead of open-ended code. Integrations are controlled, avoiding open agent loops. Workflows use models only where they add real value.

Capgemini's view is that this is where the economic case and the control case converge. None of this guarantees a fixed cost, and we do not claim it does. What a governed platform delivers is visibility and bounds: spend you can see, attribute, cap, and optimize, rather than spend that grows with every retry and every extra agent hop. For regulated, customer-facing work, predictable economics is not a nice-to-have. It is what keeps an AI program funded past its first budget cycle.

The enterprises that win on AI economics will not be the ones that spend the least. They will be the ones that can see, attribute, and bound what they spend, and route every expensive token to a decision that was worth making.

10

Current and future market trends 2026 to 2030

The table below is a reference that captures the analyst consensus on eleven market dimensions across the next five years, drawn from Gartner, Forrester, IDC, and McKinsey publications cited throughout this paper.

Dimension	Current state (2025 to 2026)	Direction of travel (2027 to 2030)
Low-code platform market size	$37 - 44.5B (Gartner forecast). LCAP segment alone $16.5B by 2027 at 16.3% CAGR.	$58.2B by 2029 (Gartner). The term "low-code" is fading from analyst vocabulary; the capability is now a default feature of enterprise platforms.
AI and agentic AI IT spend	Roughly 115 to 280 billion dollars globally (IDC). Platform-as-a-service expanding over 37% year over year in 2026.	1.3 trillion dollars by 2029, with agentic AI accounting for more than 26% of worldwide IT spending.
Enterprise apps with task-specific agents	Below 5% in 2025, 40% by end of 2026 (Gartner).	Multi-agent ecosystems by 2028. Half of knowledge workers expected to create, govern, and deploy agents on demand by 2029.
Enterprises with AI agents deployed	17% in early 2026, with more than 60% expecting deployment within two years (Gartner CIO Survey 2026).	60 to 78% by 2028 to 2030. Banking and insurance leading at roughly 63% production rate by 2027.
Agentic AI project success rate	More than 40% expected to be cancelled by end of 2027, on cost, unclear value, and weak controls (Gartner).	Success rates improve as control plane and orchestration categories mature.
AI-generated code in production	60% of all new code AI-generated by end of 2026 (Gartner); 45% contains vulnerabilities (Veracode).	Security tooling matures around AI-generated code. Agentic security emerges as its own category.
Control plane and governance	Emerging as a distinct concept. Forrester evaluation announced December 2025. Multiple vendors entering the category.	Becomes the operational center of gravity alongside identity and observability. Regulators issue specific guidance on agentic governance.
Change model	Code-anchored for traditional apps, model-anchored for platforms, re-prompt and regenerate for vibe coding.	Model-anchored change dominates for production. AI assistance generates diffs against the model rather than against code.

Dimension	Current state (2025 to 2026)	Direction of travel (2027 to 2030)
Regulatory posture	EU AI Act and sectoral regulators issuing frameworks. Enforcement building.	Audit-grade evidence requirements become standard. Platforms producing native audit trails gain structural advantage.
Vendor landscape	Multiple vendors across overlapping categories: hyperscalers, AI platform startups, enterprise platforms, vibe coding tools.	Consolidation around platforms combining deterministic orchestration with native agentic AI under a control plane. Pure-play vibe coding tools likely absorbed.
Commercial model	Per-seat licensing for platforms, consumption pricing for AI services, project-based delivery cost.	Outcome-aligned commercials become more common at the platform layer. Delivery shifts toward managed-service models.

Two implications that are easy to underestimate

First, the AI bubble is deflating in the way analysts describe these things, not the technology, the commercial expectations. Forrester's 2026 predictions describe the next phase as the era of frumpy but functional AI: less glamorous, more valuable. McKinsey's data shows 94% of organizations not yet seeing significant value from AI. This is the predictable second act of a hype cycle. The platforms that survive it will be the ones that already deliver measurable value through deterministic orchestration and embed AI as augmentation rather than substitution.

Second, the control plane becomes a board-level concept within three years. Today most CIOs treat it as an architecture detail. Boards do not. Boards ask who can demonstrate that an agent took a defensible action in a regulated context. That question is increasingly answered by the control plane and the orchestration plane working together.

11

Industry implications: Banking, insurance, telco, public sector

The argument changes shape across industries. Agentic process automation is not a one-size-fits-all capability. Its value emerges only when it is tailored to the specific decision patterns, risk profiles, and operational realities of each sector. The four short views below consolidate the pattern.

Banking and capital markets

Disputes, onboarding, KYC, complaints, lending origination, and servicing are all process-heavy, regulated, and customer-impacting. Banks operate under intense regulatory scrutiny while facing sustained pressure to reduce cost-to-serve. Fragmented system landscapes and inconsistent data quality often prevent a unified customer view. Vibe-coded onboarding is a regulatory liability. The Federal Reserve, FCA, MAS, and ECB have all signaled that audit-grade explainability of decisioning is non-negotiable for customer-facing workflows. The pattern we recommend is platform-based orchestration with embedded agentic intent detection, document understanding, and decisioning. Agentic architectures enable compliance-by-design: intelligent agents orchestrate KYC and AML activities end-to-end, coordinating evidence collection, routing cases, handling exceptions, and maintaining fully auditable decision trails.

The Capgemini Research Institute's work on agentic AI underscores that the enterprises capturing real return are those that deploy inside a governed architecture, not those bolting agents onto existing systems.

Insurance

Underwriting, claims first notice of loss, claims adjudication, and policy servicing all require auditable decisioning, line-of-business variation, and regulatory accommodation. Insurers face pressure to accelerate time-to-market for new products and journeys while operating with fragmented data that limits underwriting precision. Model-driven inheritance, which allows base applications with country, product, and regulatory variation as additive layers, reduces change cost by a factor rarely available in any other architecture pattern. Agentic insurance lifecycles support onboarding, underwriting, and claims through automated context assembly, intelligent document processing, and guided decision support, reducing cycle times while preserving control. AI agents add value in document understanding and exception triage, but they sit on top of a governed claims process, not in place of it. Forrester's Predictions 2026 notes that process intelligence will rescue up to 30% of failed AI initiatives by grounding automation in how work actually happens.

Telecommunications

Order management, service activation, billing disputes, and customer-experience workflows benefit from the same hybrid pattern. Telecom operators face high operational cost driven by network incidents, fault management, and outage-related call spikes. Agentic models enable experience-driven operations, converging network management and customer engagement into a closed-loop operating model. Autonomous network operations centers apply predictive detection, root-cause analysis, and guided remediation under human oversight, while intelligent customer-service agents increase containment for routine inquiries and provide targeted assistance for complex cases. Real-time next-best-action decisioning is increasingly the differentiator in business-to-consumer telco experience. Forrester reports 73% of organizations exploring multi-agent implementations in 2026, with telco among the leading sectors.

Public sector

Citizen services, benefits administration, case adjudication, and regulatory enforcement workflows are the canonical use case for governed orchestration with selective AI assistance. Public services are often characterized by complex forms, long processing times, and fragmented responsibilities across agencies. Agentic solutions reimagine citizen services by guiding form completion, automating eligibility checks, and routing cases with full auditability. Cross-agency orchestration breaks down silos by coordinating data and tasks while preserving data privacy and sovereignty. The auditability and explainability requirements in this sector make ungoverned agentic deployments effectively non-viable. The EU AI Act's high-risk category covers most public-sector decisioning workflows, and the supply-side response has been consolidation around platforms with native audit trails and explainable decisioning.

Across all four industries the pattern is the same, and it is industry-specific only in its details. Regulated decisioning, auditability, and integration depth dictate the architecture; agentic capability adds value at the reasoning points inside that architecture. Capgemini's role across these industries is making the pattern operationally true inside each client's regulatory perimeter.

12

What enterprise AI platforms should provide

The hard part of agentic transformation is not deciding it matters. It is deciding what an enterprise AI platform must actually do to carry the weight that will land on it. Everything in this paper up to this point has argued that the architecture which scales is hybrid: deterministic orchestration as the backbone, agentic reasoning at selected nodes, and a control plane supervising the estate. That argument is incomplete without a working definition of the platform itself. The capabilities below are the ones we look for in client engagements, regardless of vendor. They function as a rubric, not a wish list: each item solves a failure mode this paper has already documented, and a platform that misses two or three of them will struggle to absorb the velocity of vibe coding without inheriting its production-readiness gap. Read the list as a structural test. Anything that passes deserves a closer look. Anything that does not will eventually become the type of integration project the platform was supposed to prevent.

Deterministic process orchestration as a native primitive: The platform should treat workflows, case management, routing, and service-level agreements as first-class objects, not as components engineered on top of a generic application platform. This is the layer that enforces audit, identity, and predictability.

Agentic execution under governance: The platform should support AI agents as governed participants in workflows, with bounded scope, structured inputs and outputs, and the ability to be invoked from inside a deterministic process. Agents should operate inside the platform's identity and audit perimeter, not outside it.

Lifecycle governance and audit-grade evidence: Every case, decision, rule, and action should produce audit evidence at a defensible grain, generated as a byproduct of execution rather than as a separate engineering effort. Regulators ask who took what action; the platform should answer that question without a custom observability stack bolted on after the fact.

Decisioning as a first-class capability: Real-time, next-best-action decisioning, eligibility, and pricing logic should be native and explainable, not stitched together from a separate rules engine. Decisioning is increasingly the layer that governs when an agent is invoked in the first place.

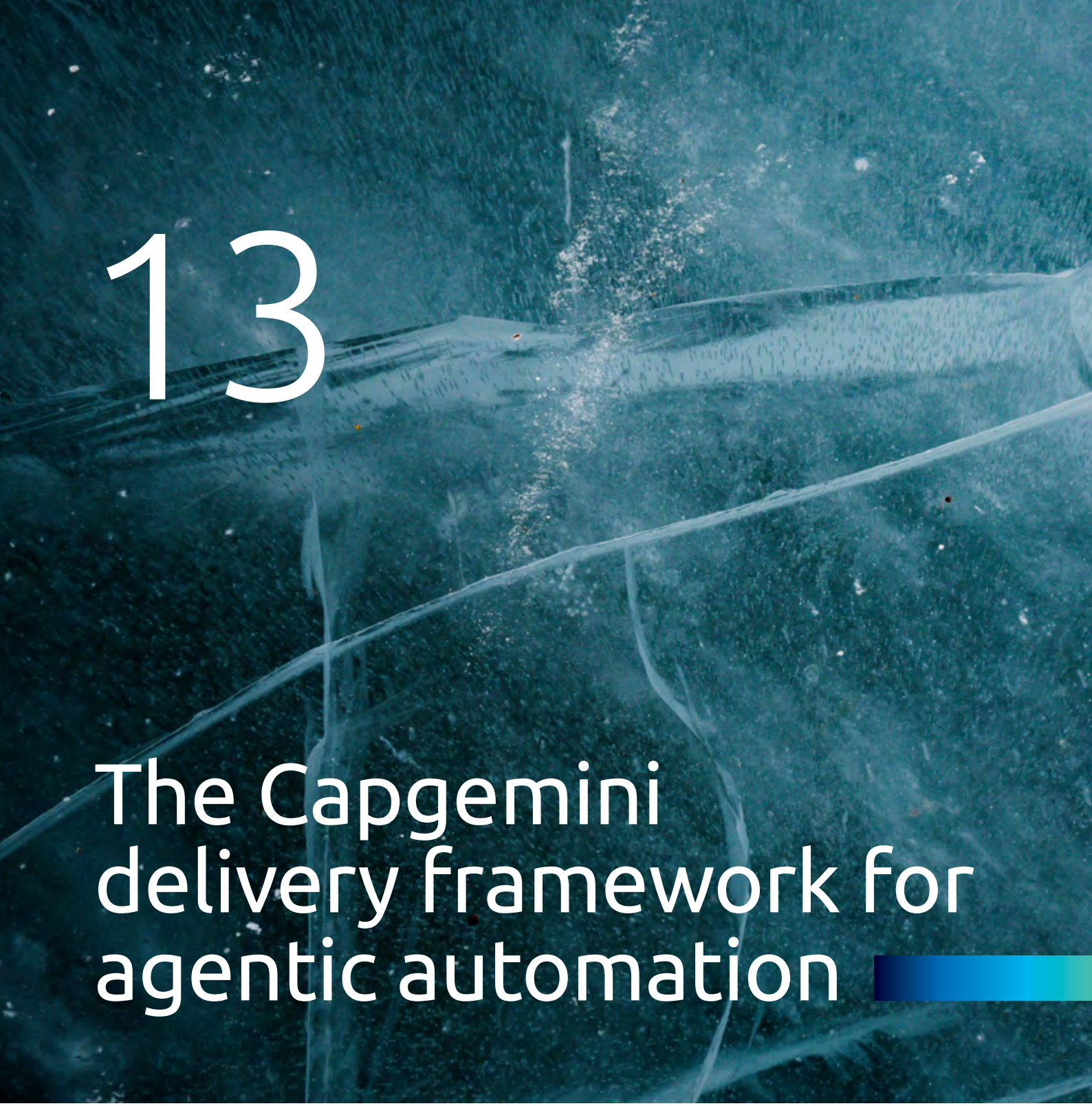
Model-driven architecture, not generated code: The application artifact should be a model that can be reasoned about, version-controlled, and modified, not a body of generated code that has to be regenerated to change. Model-driven inheritance, where country, line-of-business, and regulatory variation become additive layers rather than forks, is the single largest sustainment lever in long-lived applications.

Open protocol support for agentic interoperability: The platform should expose its workflows and tools through MCP for external agent invocation, and support A2A for agent-to-agent delegation, under enterprise identity and audit. Workflows that cannot be called by external agents will be on the wrong side of where agentic traffic flows.

Agent supervision and observability: Every agent invocation, decision, and tool call should be observable in detail. Drift detection, intervention metrics, reversal paths, and confidence signaling are platform capabilities, not bolt-on projects.

Hybrid development that absorbs AI coding velocity: The platform should let AI coding assistants and vibe-coding tools contribute productively to development, but on a governed substrate where the artifact is a model rather than free-form code. The right architecture absorbs velocity from the AI ecosystem without inheriting its production-readiness gap.

The next section applies this rubric to a representative enterprise AI platform example. Multiple enterprise platforms meet these criteria to varying degrees, and the evaluation is deliberately a worked example rather than a vendor selection. Other platforms in the Business Orchestration and Automation Technology (BOAT) and adaptive process orchestration categories satisfy several of these criteria today; the question for any specific enterprise is which of them satisfy the criteria most relevant to that enterprise's regulatory perimeter, integration estate, and existing investment posture.



13

The Capgemini delivery framework for agentic automation

This is where Capgemini's contribution is most distinct. A platform supplies capability. It does not tell an enterprise how to deploy agentic automation safely, in what sequence, or how to expand autonomy without losing control. The framework below is the operating model we bring to exactly that problem. It is refined across client engagements, independent of any single vendor's tooling, and it is the part of the work a technology purchase cannot replace.

Agentic automation requires a different delivery model than classic automation. The governing idea is to design the goals first, orchestrate the decisions next, and only then let agents act within guardrails. Throughout every phase, one question stays in front of every deliverable: can we responsibly delegate more decision power without losing control, trust, or accountability.

Capgemini Delivery Framework for Agentic Automation

Design the goals, orchestrate the decisions, then let agents act within guardrails

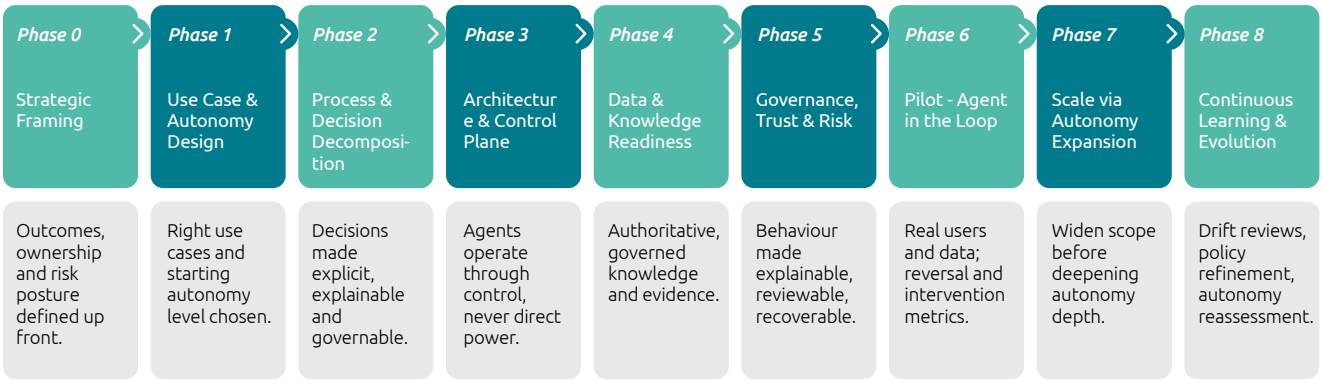


Figure 19. The Capgemini delivery framework for agentic automation, Phases 0 to 8, with the autonomy ladder that expands stage by stage.

Autonomy levels

Autonomy is not a switch. It is a ladder, and most successful deployments climb it slowly. We use five levels, and most programs start at Levels 1 to 2 and remain below Level 3 in the first year.

Level	Description
Level 0	Assist only. The agent makes recommendations; a human takes every action.
Level 1	Execute with approval. The agent proposes an action; a human approves before it runs.
Level 2	Execute with monitoring. The agent acts; a human monitors and can intervene.
Level 3	Conditional autonomy. The agent acts within defined conditions; escalation is automatic outside them.
Level 4	Full autonomy. Rare, reserved for well-bounded, low-risk, fully evidenced decisions.

Objectives and results by phase

The framework runs from strategic framing through to continuous evolution. Each phase has a clear objective and a defined set of results, so a program can be governed against evidence rather than enthusiasm.

Phase	Objective	Results
Phase 0 - Strategic framing	Establish intentional delegation with clear outcomes, ownership, and risk posture	Outcomes defined; autonomy boundary documented; sponsor assigned; non-goals documented
Phase 1 - Use case and autonomy design	Select the right use cases and starting autonomy level to ensure adoption and trust	Use-case shortlist with success criteria; autonomy level set; override and approval model agreed
Phase 2 - Process and decision decomposition	Make decisions explicit, explainable, and governable before automating execution	Decision inventory; exceptions; human triggers; success signals; traceable rationale
Phase 3 - Architecture and control plane	Ensure agents operate through control, not direct power	Orchestrator; guardrails; override; audit and traceability; no direct system access
Phase 4 - Data and knowledge readiness	Provide authoritative, governed knowledge and evidence tracking	Systems of record; memory model; approved sources; evidence linked to outcomes; ownership
Phase 5 - Governance, trust, and risk	Make agent behavior explainable, reviewable, and recoverable	Explainability model; logging; escalation; bias and drift monitoring; confidence signaling

Phase	Objective	Results
Phase 6 - Pilot, agent in the loop	Validate real-world performance and trust within controlled scope	Real users and data; instrumentation; reversal and intervention metrics; go, adjust, or stop decision
Phase 7 - Scale via autonomy expansion	Scale impact without eroding trust by expanding scope before autonomy depth	Widened boundaries; additional agents; autonomy increased on evidence; new roles; KPI shift
Phase 8 - Continuous learning and evolution	Sustain performance, trust, and compliance over time	Drift reviews; policy refinement; controlled prompt updates; autonomy reassessment; audits

Developing agentic ecosystems and scaling them

The future of workflow automation is a shift from passive digital assistants to proactive, autonomous systems that continuously interpret customer and process data, reason independently, and initiate action in an always-on mode. This requires organizations to rethink how applications are designed and built. The objective is to convert business data and functionality into scalable, agent-agnostic tools, potentially paired with modular agents, that can be accessed across enterprise applications through agentic frameworks using MCP servers and the A2A protocol. To stay competitive, organizations need an AI-first mindset when building new applications or modernizing existing ones.

No organization becomes fully agentic overnight. Success requires a phased transformation grounded in business-domain expertise, deliberate autonomy boundaries, and governance that enables safe scaling. Business domain is key. Let common sense rule and make it real.

14

Recommendations for leaders

If you take one practical view away from this document, take this set. These are the recommendations Capgemini gives to financial services CIOs, COOs, and CDOs navigating the platform, agentic AI, and vibe coding question right now.

1. Separate design-time speed from run-time risk in your strategy

Use vibe coding and AI assistants to compress design and prototyping. Do not let that speed cross into production delivery without platform-grade governance. Establish an explicit rule in your delivery model: prototypes are throwaway, production runs on the platform. The Veracode and Cloud Security Alliance data make this argument quantitatively. Use it to set the policy.

2. Build, or standardize on, your Enterprise AI Platform

Pick a platform that combines deterministic orchestration with native agentic AI, audit-grade governance, decisioning, and lifecycle management. Prioritize platforms that demonstrate strong capabilities across orchestration, automation, decisioning, process intelligence, governance, and customer engagement, with analyst Leader positions across orchestration, digital process automation, AI decisioning, real-time interaction management, process intelligence, and process mining. The cost of standardization is real, but lower than the cost of an ungoverned agent estate.

3. Engineer the control plane from the start

Identity, policy, auditability, observability, lifecycle management, and human-in-the-loop are platform capabilities, not project artifacts. Forrester's December 2025 announcement of the agent control plane market category is a signal: the next twelve months will be defined by who builds this layer first. Treat these capabilities as foundational and build them once. They pay back across every subsequent agent deployment.

4. Scale scope before scaling autonomy

Use staged autonomy levels: assist, recommend, act with approval, act with notification, act autonomously. Increase the breadth of use cases before deepening autonomy. Most organizations are better served by ten agents at recommend level than one agent at full autonomy. Gartner's projection that more than 40% of agentic AI projects will be cancelled by 2027 comes largely from organizations that scaled autonomy before scope.

5. Measure decision quality, not automation volume

Track interventions, reversals, time-to-resolution, error severity, and customer impact. Avoid vanity metrics like the percentage of cases automated, which can mask declining decision quality. McKinsey's data on the productivity paradox is the explicit warning: 94% of organizations not yet seeing significant value, largely because they measure activity rather than outcome.

6. Invest in semantic truth and data lineage

Agents are only as good as the data they reason over. Governed definitions, traceable lineage, and access-controlled data products are prerequisites for trustworthy agentic deployment. Forrester estimates process intelligence alone will rescue up to 30% of failed AI initiatives by grounding automation in how work actually happens.

7. Modernize the legacy estate deliberately

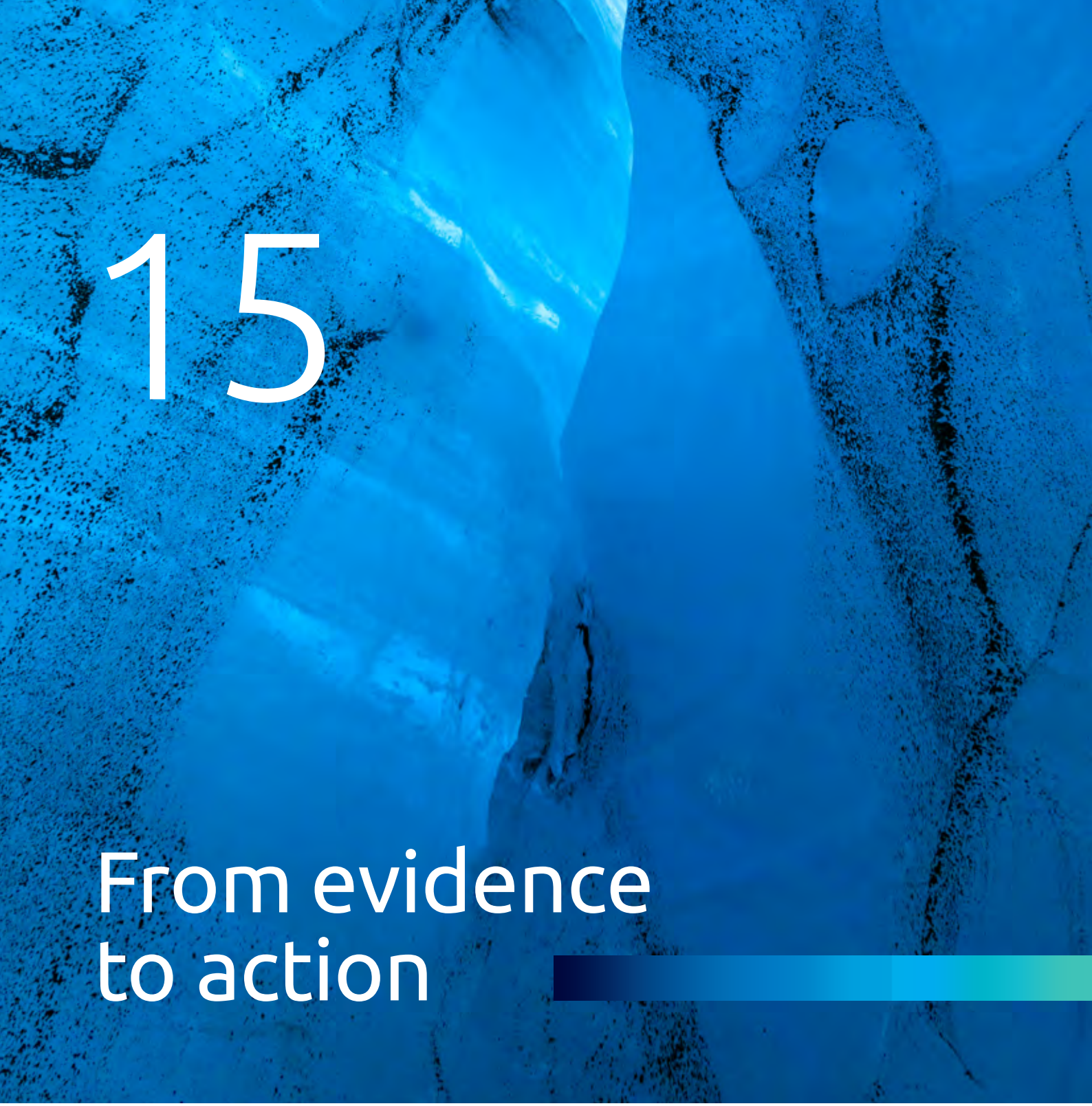
Legacy systems are the largest reservoir of unmodernized process value in most enterprises. Address them with a structured modernization program using Blueprint legacy-analysis agents, rather than incidental rewrites. This converts technical debt into modern, AI-ready process assets, and is increasingly the precondition for meaningful agentic deployment at scale.

8. Build the talent stack you will need

The teams that succeed in this transition combine platform architects, agent designers, control plane engineers, process designers, and domain experts. Pure engineering teams and pure AI teams both produce predictable failure modes at enterprise scale. Gartner's stage-of-evolution model is direct: by 2029, at least half of knowledge workers will be expected to create, govern, and deploy agents on demand. The talent shift is already underway.

9. Choose the right type of AI and automation for each use case

Separate design-time from run-time decisions and prioritize deterministic workflows, rules, and orchestration by default, while reserving AI-driven reasoning and multi-agent patterns only for high-value, context-dependent tasks. This way, you actively control cost and balance it against complexity and driving business outcomes.



15

From evidence to action

The argument of this paper can be reduced to one decision. An enterprise can let AI velocity arrive as ungoverned code in repositories its operations team will eventually inherit, or it can route that velocity through a governed platform that turns the speed into a sustainable model. Everything else, the analyst data, the security research, the lifecycle phase analysis, the industry patterns and platform evaluation criteria are evidence for why that single decision matters more than the surface conversation about which AI tool to standardize on. The five years on the other side of that decision look very different depending on which side a CIO is now standing on.

What action actually looks like

The most common failure mode at this point is not disagreement with the thesis. It is agreement followed by paralysis, because the implications cut across architecture, sourcing, talent, security, and operating model at the same time. A pragmatic move-to-action sequence breaks the paralysis without overcommitting prematurely.

First, name the run-time perimeter. The single most useful early decision is also the cheapest: write down which application categories must run on a governed platform, which can run on AI-assisted custom code with enhanced review, and which can be left to design-time exploration. This is a one-week conversation between the CIO, the chief risk officer, and the head of engineering. It costs nothing and immediately changes how every subsequent procurement request is evaluated.

Second, pick one regulated, high-impact workflow as the proof case. Instead of a greenfield internal tool or a back-office pilot, select a workflow that touches a customer, sits inside a regulatory perimeter, has documented decision logic, and runs at meaningful volume. Examples include KYC, claims first notice of loss, dispute handling, complaint resolution, and eligibility adjudication. Build it on the governed platform using the rubric in the previous section. Instrument it for both productivity and risk metrics from day one. The point is not the workflow. The point is producing one defensible reference inside the enterprise that the rest of the organization can argue from rather than against.

Third, stand up the control plane before the agent estate grows. Identity propagation, policy enforcement, observability, cost attribution, and lifecycle management for agents are platform capabilities, not project artifacts. Building them once, early, and when the agent count is small costs much less than retrofitting them later when the estate is sprawling across vendors and clouds. The Forrester December 2025 evaluation of the agent control plane category exists because every enterprise that has skipped this layer is now reverse engineering it under regulatory pressure.

Fourth, measure what actually matters. Track intervention rates, reversal rates, time to resolution, decision quality, and customer impact. Avoid the vanity metric trap where the share of cases automated is mistaken for value created. McKinsey's State of AI 2025 reports 94% of organizations not yet seeing significant value from AI investment, and the cause is almost always measurement focused on activity rather than outcome. Establishing outcome metrics now, before the agent estate scales, prevents the same problem later.

Fifth, make the commitment visible. Organizations move when leadership commits publicly enough that retreat is harder than progress. That means an architecture position that is written, signed off, and shared with risk and audit. It means a sourcing decision that is communicated to engineering teams as a direction, not a suggestion. And it means a talent plan that acknowledges the new roles, platform architects, agent designers, control plane engineers, process designers, that this transition requires.

Capgemini's role in the move from evidence to action

The delivery framework in Section 14 and the operational recommendations in Section 15 are not abstract ideas. They are the operating model Capgemini brings to client engagements across banking, insurance, telecommunications, and the public sector. We work with the leadership team to set the run-time perimeter, with the architecture team to instrument the proof case, with risk and compliance to stand up the control plane, and with the operating model team to make the new measurement framework real inside the existing performance management process. The objective in every engagement is the same: the enterprise emerges with one production-grade reference, a working governance posture, and the institutional muscle to scale the second, third, and fourth deployments without reproducing the analysis work from scratch.

The window for this decision is open now, and it will not stay open indefinitely. The enterprises that build the governed substrate over the next eighteen months will scale agentic capability under control. The ones that defer the decision will inherit an ungoverned estate that costs more to remediate than it would have cost to architect properly. The choice between those two outcomes is not made in a single board meeting. It is made in the next four or five procurement, architecture, and operating model decisions, each of which will either reinforce the governed-substrate posture or quietly walk away from it. The framework in this paper is the basis on which to make those decisions consistently.

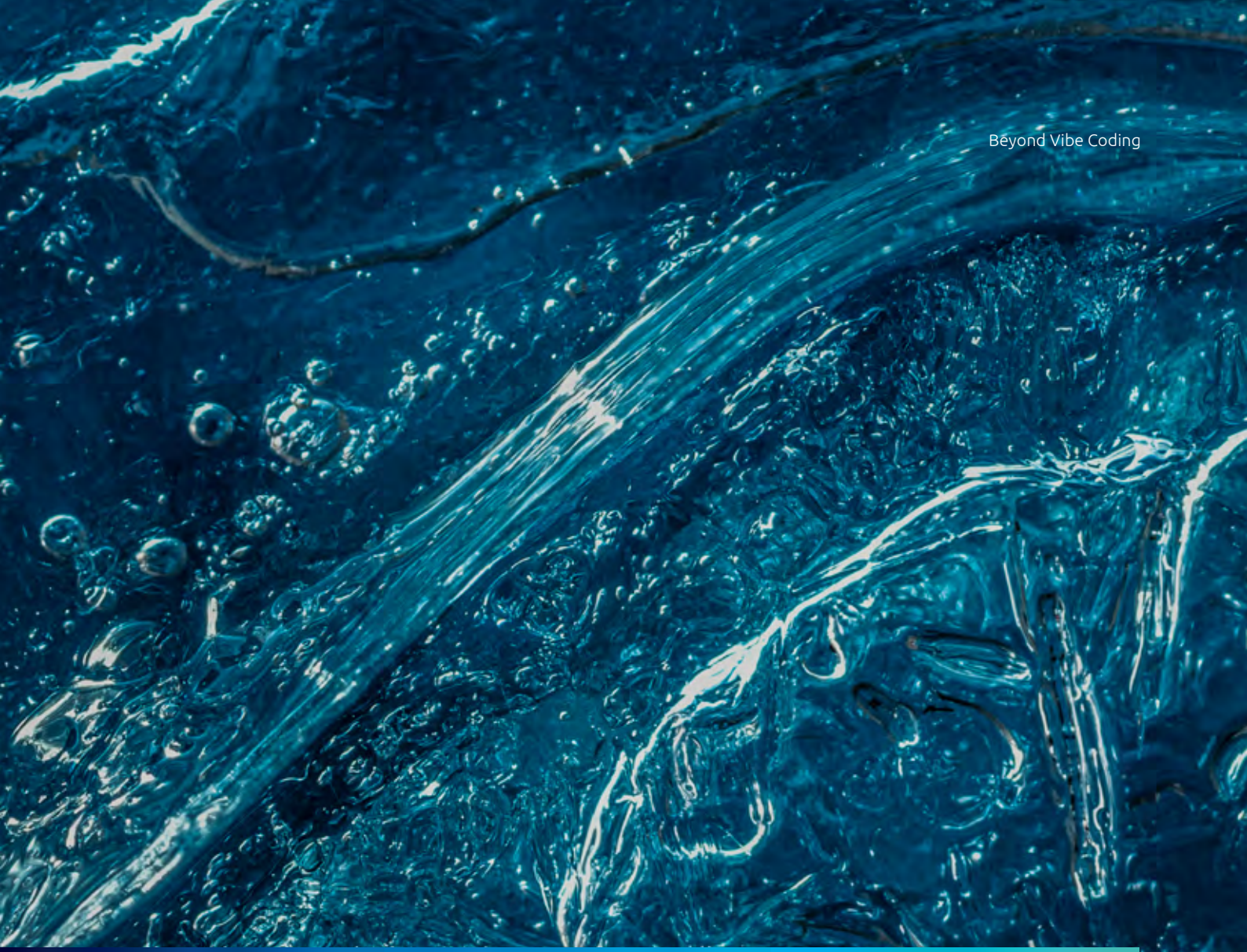
The next decade will not be won by the organizations that generate the most code. It will be won by the organizations that orchestrate decisions, govern agents, and continuously adapt their business processes at scale.

16

References and sources

All third-party data in this paper is drawn from publicly available analyst and research publications. Where a chart or claim is not externally attributed, it reflects Capgemini engagement experience or internal analysis.

1. Gartner, Forecast Analysis: Low-Code Development Technologies, Worldwide (2024).
2. Gartner, Magic Quadrant for Business Orchestration and Automation Technologies (October 2025).
3. Gartner, Magic Quadrant for Process Intelligence (May 2026); Magic Quadrant for Process Mining Platforms (April 2025).
4. Gartner, press release on agentic AI enterprise software revenue and adoption (August 2025).
5. Gartner, analysis on agentic AI project cancellation rates (June 2025); 2026 Hype Cycle for Agentic AI.
6. Gartner, 2026 CIO and Technology Executive Survey.
7. IDC, Worldwide AI and Agentic AI IT Spending Forecast (August 2025); IDC FutureScape 2026.
8. McKinsey, Measuring the Impact of AI in Software Development (December 2025); The AI Revolution in Software Development (November 2025); Unleashing Developer Productivity with Generative AI (2023, updated 2025); State of AI 2025.



9. Forrester, Predictions 2026: Artificial Intelligence, Automation and Robotics; Agent Control Plane evaluation announcement (December 2025); Adaptive Process Orchestration landscape (2026); AEGIS framework (2026).
10. Forrester, Wave evaluations: Digital Process Automation Platforms (Q3 2025), AI Decisioning Platforms (Q2 2025), Real-Time Interaction Management (Q4 2025).
11. Veracode, GenAI Code Security Report (2025).
12. Tenzai, 15-application vibe coding security study (December 2025).
13. Escape.tech, vibe-coded production application security scan (Q1 2026).
14. Cloud Security Alliance, Research Note on AI-assisted commits and secret exposure (March 2026); OWASP Top 10 for LLM Applications (2025).
15. Georgia Institute of Technology, Vibe Security Radar assessment (Q1 2026).
16. Jellyfish and OpenAI, developer AI adoption and productivity study (August 2025).
17. Capgemini Research Institute, Rise of Agentic AI (2025); AI in Action: How Gen AI and Agentic AI Redefine Business Operations (2025).

About Capgemini

Capgemini is an AI-powered global business and technology transformation partner, delivering tangible business value. We imagine the future of organizations and make it real with AI, technology and people. With our strong heritage of nearly 60 years, we are a responsible and diverse group of over 420,000 team members in more than 50 countries. We deliver endto-end services and solutions with our deep industry expertise and strong partner ecosystem, leveraging our capabilities across strategy, technology, design, engineering and business operations. The Group reported 2025 global revenues of €22.5 billion.

Make it real
www.capgemini.com