HCLSoftware

Why platforms matter more in the age of AI:

A strategic briefing for HCLSoftware customers and partners



Suhas Bhat
Senior Director,
Product Management

HCL **Mosaic Vibe Studio**

Table of contents

01. Executive summary
    Key takeaways
02. Starting with what is true
    Volume has changed permanently
03. The strategic reframe: build cost vs. ownership cost
04. The Vibe Coding security crisis
05. Automation bias: why human review is not enough
06. The numbers: evidence that cannot be dismissed
07. The integration explosion: the N^2 problem
08. The API sprawl crisis: 5× growth, 80% undocumented
09. What AI does not generate
10. The six platform jobs in the AI age
11. The change management tax
12. The accountability gap
13. The knowledge concentration risk
    Cognitive debt: the system understanding that AI is consuming
14. The ownership continuity argument
15. The speed argument: platform enables faster change
16. The agent sprawl time bomb
17. The counter-intuitive truth: AI is driving platform adoption
18. The 10x risk: what walking away looks like
    Token bleed: the runtime cost that arrives after the demo
    The ROI Case: Platform Guardrails Deliver 29% Higher AI returns
19. Industry convergence: the market has decided
20. Partner vs. end-customer value
21. Navigating strategic considerations in AI platform adoption
22. Metrics to prove the platform case
23. Bridging speed and control: the HCLSoftware approach
24. Strategic next steps: assessing your AI governance posture
25. Take action on your AI strategy
26. Source notes



01 · Executive summary

Enterprise AI has crossed the line from experiment to infrastructure, and the crossing happened faster than most governance functions were built to absorb. Google now reports that 75% of all new code deployed internally is AI-generated. 84% of professional developers use AI coding tools. These are not forecasts. They describe the current operating baseline.

The most important question to answer is not whether AI makes application development faster. It does, and defending against that fact is a losing position. The right defense, and the right offense, is to answer the question that comes after: what happens to the applications that AI creates?

For partners, this shift is an improvement. AI raises build throughput and protects delivery margins. For the end customer, the enterprise that owns the resulting applications, the shift creates a structural problem.

Application creation is accelerating faster than the governance, security, and operational infrastructure needed to manage what gets created.

The strategic message for every customer conversation is this: HCL Mosaic Vibe Studio accelerates creation. HCL Volt MX Foundry turns the resulting application estate into a governed, reusable, enterprise delivery system. These are not competing propositions. They are sequential ones. AI solves the build problem. The platform solves the ownership problem.

Key takeaways

Creation vs. Ownership:
AI has solved the build problem and created an ownership problem. Creation is now cheap. Governance, security, integration, and lifecycle maintenance are not.

Quadratic Complexity:
Point-to-point integration across 300 applications produces 44,850 connection pairs. A platform hub reduces the same estate to 300 governed connections.

Human Review Fails at Scale:
Research across 302,579 AI-authored commits found that 22.7 percent of AI-introduced bugs survive human review due to automation bias.

Measurable Financial Exposure:
Shadow AI breaches average $4.63 million. The EU AI Act carries maximum penalties of 35 million euros or 7 percent of global annual turnover.

The Industry Consensus:
90 percent of enterprises now operate internal developer platforms, and 62 percent name governance as the primary barrier to scaling AI.

> AI makes apps cheap to create. Platforms make them safe to own.



02 · Starting with what is true

Any credible conversation with a customer about platforms in the AI age must begin by acknowledging what is real. AI productivity gains are not aspirational. They are measured and documented. A controlled GitHub Copilot study found 55 percent faster task completion. Adoption is near-universal: 84 percent of professional developers now use AI coding tools in some capacity.

The question is not whether AI creates applications faster. It does. The question the platform answers is fundamentally different: what happens to those applications after they are created? Creation is the beginning of the cost, not the end of it. The platform is the infrastructure that makes creation sustainable.

Volume has changed permanently

Google's own corporate tracking confirms that 75% of all new code deployed internally at Google is now AI-generated and reviewed by engineers. Across the global technology sector, approximately 46% of all production code is actively written by AI assistants daily. The barrier to creating software has not been lowered, it has been eliminated.

Consider an enterprise already managing 150 applications across 1,000 repositories. This is not a gradual trend to watch. It is the operating environment for the next 18 months. AI-assisted development will not hold that estate at 150 apps and 1,000 repositories. It will push it to 300 apps and 2,000 repositories, and it will do so faster than any governance structure built application by application can track. The platform is the only mechanism that scales governance at the same rate as creation.

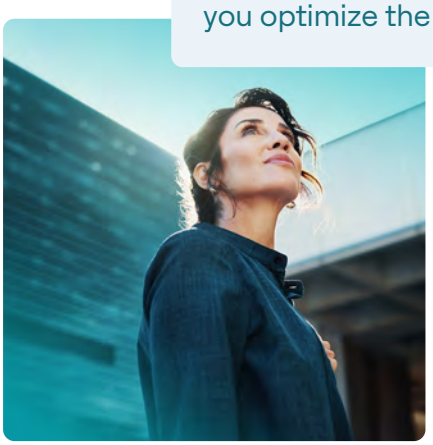
03 · The strategic reframe: Build cost vs. ownership cost

Before AI arrived as a practical tool in enterprise software delivery, the primary question was whether the organization could build required applications fast enough to meet business demand. The bottleneck was creation. Platforms helped, but the argument for them was primarily about reuse and consistency, they made builders more productive.

In the AI age, the bottleneck has moved. The question is no longer whether the organization can build applications fast enough. It can. And increasingly, it will produce more of them than it can safely manage. The new bottleneck is ownership: governance, security, operational continuity, and the organizational capacity to maintain what has been created.

AI attacks build cost. The platform attacks lifecycle cost. These are different problems, and a strategy that addresses only one of them is incomplete. The organization that optimizes for build alone gets faster at creating things that are progressively harder to own.

> When generation is expensive, you optimize the builder. When generation becomes cheap, you optimize the estate.



04 · The vibe coding security crisis

This is the section that should change the tenor of the conversation, because the evidence is no longer theoretical. In May 2026, Red Access scanned 380,000 publicly accessible web assets and found 5,000 corporate-purpose assets with sensitive data and no basic security controls. The majority of these had been built with AI assistance.

An independent study of 5,600 vibe-coded applications in production found that zero had CSRF protection, zero had proper security headers, and zero had properly scoped access policies. These applications are not outliers. They are the median output of ungoverned AI-assisted development.

This pattern has been seen before, and the industry knows how it ends. The AWS S3 misconfiguration crisis of 2017 through 2019 followed an identical trajectory. Anyone could create an S3 bucket. The defaults were open. The tools were accessible. And within two years, thousands of organizations had publicly exposed sensitive data through buckets they had created and forgotten. AWS eventually changed the defaults, after the breach wave.

The vibe coding crisis is the same pattern at a larger and more dangerous attack surface. The difference between a misconfigured S3 bucket and a poorly governed AI-generated application is significant: an application has business logic, user data, integration connections, and in many cases direct access to core enterprise systems.

The broader shadow AI context reinforces the severity. Menlo Security's 2025 research found that 68 percent of employees use unsanctioned AI tools in their work, and 57 percent input sensitive corporate data into those tools. The exposure is not hypothetical. It is already in motion.

> AI didn't create shadow IT. It made shadow IT instant.



05 · Automation bias: Why human review is not enough

The assumption that skilled engineers reviewing AI-generated code will catch its problems is empirically wrong.

A large-scale 2026 empirical study titled "Debt Behind the AI Boom" analyzed 302,579 verified AI-authored commits across more than 6,000 corporate repositories. The data is consistent: over 15% of all AI-generated commits introduce immediate structural code defects or security smells, with code smells accounting for 89.1% of issues. More critically, 22.7% of these AI-introduced bugs survive all the way into the final production codebase.

Why? Human reviewers suffer from automation bias: clean-looking, fast-generated AI code signals competence. Engineers approve pull requests without rigorous verification because the code looks right. This is not a failure of skill, it is a documented cognitive pattern.

When code is generated at machine speed and volume, human review alone cannot be the quality gate.

The IBM Institute for Business Value confirms the downstream effect: 81% of executives report that technical debt is already actively constraining their AI success. 69% believe it will render some AI initiatives completely financially untenable.

Platform-enforced architectural guardrails at the infrastructure layer are the only reliable defense. A platform does not replace code review, it makes code review work by constraining what the code can do before it reaches the reviewer. Integration patterns are pre-approved. Identity is centralized. Security policies are embedded. The reviewer is validating business logic, not reconstructing governance from scratch for every commit.

06 · The numbers: Evidence that cannot be dismissed

The evidence on AI-generated code security is not a collection of isolated warnings, it is a consistent body of findings across multiple independent research efforts, and its consistency is precisely what makes it actionable.

45%
of AI-generated code contains vulnerabilities from the OWASP Top 10.

30%+
Increase in technical debt following ungoverned AI tool adoption.

$4.63 M
The average financial cost of a shadow AI data breach.

Veracode's GenAI Code Security Report found that nearly half of all AI-generated code contains well-documented and widely recognized application security risks. This rate is unchanged across multiple testing cycles, which means the problem is structural, not incidental.

The situation worsens at scale. AI-assisted developers produce commits at three to four times the rate of human developers, but they introduce security findings at ten times the rate. Iteration makes it worse: five rounds of AI-assisted refinement on the same code increases critical vulnerabilities by 37.6 percent.

The debt picture is equally clear. Technical debt increases 30 to 41 percent after AI tool adoption without governance controls in place. This is the structural cost of AI-accelerated development without platform oversight: the applications get built faster, but the hidden cost of owning them accumulates at a higher rate.

The financial consequences are no longer abstractions. IBM and TechTarget research published in 2026 found that the average shadow AI breach costs $4.63 million, which is $670,000 above the standard breach baseline. Sixty-five percent of these breaches expose PII, compared to 53 percent in typical incidents. EY's Responsible AI Pulse Survey found that 99 percent of organizations reported financial losses from AI-related incidents, with 64% losing more than $1 million and an average realized loss of $4.4 million.

The regulatory dimension adds a further layer of financial exposure. The EU AI Act carries maximum fines of 35 million euros or 7 percent of global annual turnover for prohibited AI practices, a benchmark that applies to any organization operating in or serving European markets.



07 · The integration explosion: The N^2 problem

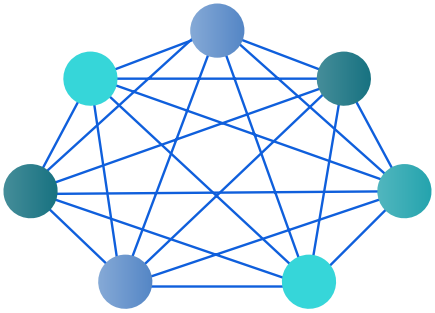
The mathematics of integration without a platform are unforgiving. When applications communicate with one another through point-to-point connections rather than through a governed platform hub, the number of integration pairs required grows as $N(N-1)/2$, where N is the number of applications. Each integration pair must be independently built, tested, versioned, documented, and maintained. The math compounds quickly.

The table below makes the arithmetic concrete. The difference between a governed and an ungoverned estate does not become catastrophic at 10 applications. It becomes catastrophic at scale...and AI is driving scale.

Number of Apps	Without platform (integration pairs)	With platform (connections)
10	45	10
50	1,225	50
150	11,175	150
300	44,850	300

A platform hub reduces the integration problem from quadratic to linear. At 150 applications, the difference is the gap between 11,175 unmanaged pairs and 150 governed connections.

Without Platform

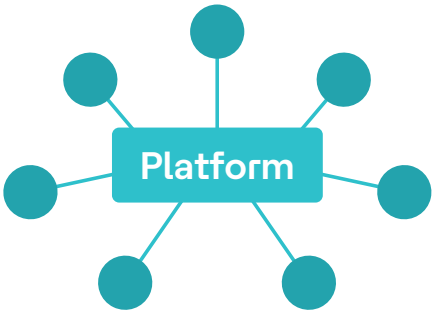


7 apps = 21 integration pairs

150 apps = 11,175 pairs

CHAOS

With Platform



7 apps = 7 connections

150 apps = 150 connections

CONTROL

> N apps without a platform equals N(N−1)/2 integration problems.
> With a platform: N.

08 · The API sprawl crisis: 5× growth, 80% undocumented

The N^2 integration problem has a security dimension that makes it even more urgent than the integration math alone suggests.

Structural data compiled by Boomi and IDC reveals that organizations running generative AI applications are now managing approximately 5× the number of APIs compared to non-AI enterprises. Because AI agents and applications generate point-to-point connections rapidly and without central visibility, only 10–20% of these APIs are properly documented or discoverable.

The result is an explosion of what the industry calls "shadow and zombie APIs", active connections to enterprise systems that security teams cannot see, cannot govern, and cannot revoke. Globally, poorly governed interfaces are projected to cost businesses over $100 billion in annual losses. 31% of all cyberattacks targeting corporate transactional systems now specifically exploit these undocumented, unmonitored APIs.

This is the integration explosion made dangerous. Every AI-generated app that creates its own connections to enterprise systems is adding to this invisible attack surface. A platform acts as the singular hub ensuring every connection is documented, encrypted, versioned, and governed by default. The N^2 problem is not just an efficiency problem – it is a security liability that compounds with every ungoverned app added to the estate.

09 · What AI does not generate

AI generates working code. It does not generate the 6 enterprise requirements that surround working code and determine whether that code can be safely deployed, reliably operated, and durably maintained.

Security Policies

Code arrives without the authentication patterns, role-based access controls, secrets management practices, and security scanning infrastructure that enterprise deployment requires. And in an ungoverned environment, they are added inconsistently, or not at all.

Audit Trails

An enterprise operating under any meaningful regulatory framework needs to know who accessed what data, when they accessed it, and why. AI-generated applications do not produce compliance evidence unless the platform provides them as infrastructure.

Integration Contracts

When an AI-generated application integrates with an ERP system, a CRM, or a data warehouse, it does so through hand-wired API calls that reflect the developer's understanding of the target system at the time of development. When the target system changes, which it will, those hand-wired connections break.

Operational Monitoring

Uptime monitoring, performance measurement, error rate tracking, alerting, and rollback capability are not present in AI-generated code. They are capabilities that must be built or provided. A platform provides them as shared infrastructure.

Compliance Evidence

The controls documentation, data handling records, PII inventory, and policy enforcement reports that regulatory compliance requires must be assembled by someone, and in an ungoverned estate, they must be assembled individually for every application, for every audit.

Portfolio Ownership

What AI generates is a collection of individual artifacts, applications, services, code repositories, each of which was understandable to the person who built it and is progressively less understandable to everyone else.

10 · The six platform jobs in the AI age

Identity and Access
Every application in the estate needs to know who is using it and what they are permitted to do. In an ungoverned estate, each application implements this independently: different identity providers, different session management, different access control models. A platform centralizes identity so that every application in the estate shares a consistent, auditable, and updateable access model.

Integration and Service Contracts
Backend systems, ERP, CRM, databases, data warehouses, do not remain static. They upgrade, they deprecate APIs, they change authentication requirements. A platform governs the connections between applications and backend systems through versioned service contracts that can be updated in one place and inherited by all dependent applications.

Governance and Compliance
Regulatory requirements, whether under national AI governance frameworks, sector-specific regulations, or internal corporate policy, require evidence that applications handle data appropriately, that access is controlled, and that changes are tracked. A platform produces this evidence as a byproduct of normal operation.

Reuse and Consistency
The organizational value of a software estate is determined not just by the sum of what each individual application does, but by the extent to which the knowledge, components, and patterns embedded in one application can be reused in the next. A platform provides the shared component library, the design system, and the architectural patterns that make reuse the default rather than the exception.

Runtime and Operations
Running applications in production requires infrastructure: deployment pipelines, monitoring, alerting, rollback capability, scaling, and incident response. A platform provides this infrastructure at the estate level, so that every application inherits operational capability rather than building it from scratch.

Portfolio TCO Control
Total cost of ownership for a software estate is not just the cost of building the applications. It is the ongoing cost of supporting, securing, upgrading, and eventually retiring them. A platform gives the organization a single point of control over that ongoing cost, one upgrade cycle, one security update, one compliance review, rather than one per application.

11 · The change management tax

Consider a single system change event: an ERP upgrade, a new authentication provider, or the deprecation of a critical API. For a platform-governed estate, this event requires one update. The platform's service contracts absorb the change, and all dependent applications inherit the update without modification.

For an ungoverned estate, the same event requires one update per application - each of which must be individually identified, individually modified, individually tested, and individually deployed. For an estate of 150 or more applications, this is already an expensive problem. If the estate grows to double the number of applications, a plausible outcome in a two-to-three-year AI-assisted development horizon, the change management tax becomes prohibitive.

This is not an abstract efficiency argument. It is the direct financial consequence of architectural decisions made today. Organizations that believe they are saving money by avoiding platform governance are deferring a compounding cost, not eliminating it.

12 · The accountability gap

When an AI-generated application causes a security incident, a data breach, or a compliance failure, the question of accountability is unambiguous: the AI tool is not liable, the developer who prompted it is not individually liable at the organizational level, and the enterprise that deployed it owns the consequence. Accountability sits with the organization that made the deployment decision.

A platform creates the accountability infrastructure that prevents this gap from materializing. It ensures that applications are deployed through governed pipelines, that access controls are documented and enforced, that data handling is auditable, and that the organization can demonstrate, on demand, that it took appropriate care in the deployment and operation of its AI-assisted software estate.

13 · The knowledge concentration risk

AI-assisted applications tend to be understood, at any meaningful depth, only by the person who built them. This is a structural feature of AI-assisted development, not a criticism of the developers who use these tools. The speed of AI-assisted development compresses the design and debugging phases that build institutional understanding of how a system works. The result is applications that function but are not transferable.

With each organization having hundreds of code repositories already in the estate, this problem is not hypothetical, it is visible today. Every repository that is maintained primarily by institutional memory rather than documented architecture is a latent risk. When the team member who holds that memory leaves, the risk becomes an incident.

Platform patterns solve this by establishing a common architecture that any competent team member can navigate. An application built on HCL Foundry's patterns, using its integration layer, its identity management, its deployment pipeline, is legible to any engineer who understands the platform, regardless of who originally built it.

Cognitive debt: The system understanding that AI is consuming

A pattern is emerging across engineering organizations that directly extends the knowledge concentration risk argument: developer satisfaction with AI coding tools is rising, while the depth of engineers' understanding of the systems they work on is falling.

Because AI removes the friction of debugging and tracing logic, the activities that build deep understanding of how systems behave, engineers are losing their mental models of the systems they ostensibly "build."

The industry calls this cognitive debt: the accumulated gap between what a system does and what the team responsible for it understands about how it does it. When the developer who built an AI-assisted application leaves, the cognitive debt is total: nobody understands the application's logic, its failure modes, or its dependencies, because the development process never required anyone to.

A platform addresses cognitive debt structurally. When application behavior is inherited from platform patterns, standardized identity, governed integration contracts, shared component libraries, the behavior is documented at the platform level, not embedded in ephemeral AI prompt sessions. Any new team member inherits a system whose governance is readable and whose integration points are centrally documented. The platform is the institutional memory that AI-assisted development fails to build.

14 · The ownership continuity argument

Without a platform, the customer does not own a software estate. They own a collection of software artifacts. The distinction is significant. A collection of artifacts, applications on different technology stacks, with different identity models, different integration patterns, and different operational tooling, is not an asset. It is a liability that accumulates with every new application added.

With Foundry, the customer owns a governed estate. Service connections are documented contracts. Integrations are versioned and reusable. Identity is centralized and consistent. Observability spans the entire portfolio. The estate is something that can be handed from one team to the next, audited, upgraded, and extended, because its architecture is legible.

This is the deepest value proposition for the platform in the context of an AI-driven build environment: AI accelerates creation, but creation is not ownership. Ownership requires governance, continuity, and the organizational capacity to manage what has been created over time. The platform is how the customer converts AI-generated artifacts into owned, governed, enterprise assets.

The platform converts partner work into customer assets.

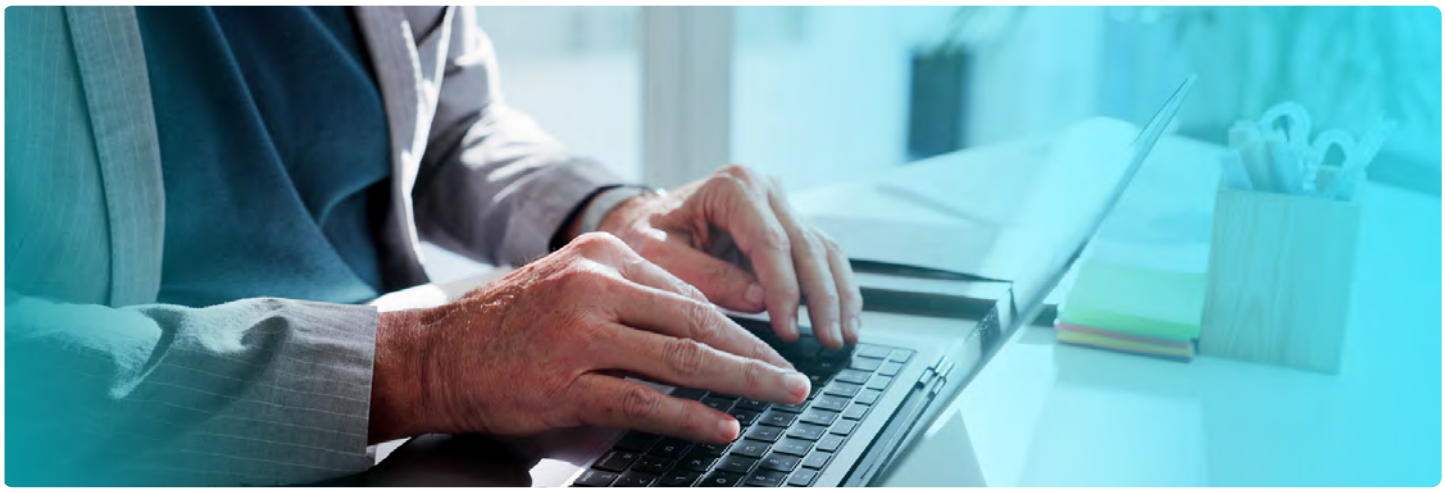
15 · The speed argument: Platform enables faster change

The platform is often positioned defensively: as a governance layer, a safety mechanism, a risk control. This framing is accurate but incomplete. The platform also enables faster change, and this is a commercial argument that resonates differently than a risk argument.

When business conditions shift, a new regulatory requirement, a new market opportunity, a major competitor move, the organization that responds fastest has a structural advantage. An enterprise with a governed estate can respond to that shift by updating shared components, governed integrations, and centralized policies, and having those updates propagate across the entire application estate.

An ungoverned enterprise must respond application by application.

An organization without a platform must build the infrastructure before it can build the application. Every new requirement comes with a setup cost that compounds the delivery timeline. Conversely, an organization building with HCL Mosaic Vibe Studio can start creating immediately. Because developers no longer waste time hand-wiring boilerplate security, identity models, or integration contracts, the platform actively accelerates time-to-production. Governance doesn't slow developers down; it removes the friction that keeps them from shipping.



16 · The agent sprawl time bomb

150K
AI agents projected to run in a typical Fortune 500 enterprise by 2028.

50%
of these agents currently operate in isolated silos with zero unified governance.

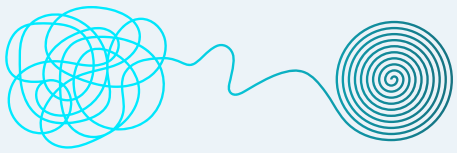
40%
of agentic AI projects will be canceled by 2027 due to governance failures.

The next wave of AI deployment is not individual applications, it is autonomous agents. Industry research finds that Fortune 500 enterprises will go from fewer than 15 AI agents to more than 150,000 by 2028. This is not a gradual adoption curve. It is a step change in the scale and complexity of what must be governed.

The governance picture for this coming wave is not encouraging. Fifty percent of agents currently operate in isolated silos with no coordination and no unified governance. Twenty-seven percent of the APIs these agents consume are ungoverned. Gartner predicts that more than 40 percent of agentic AI projects will be canceled by 2027, with governance failure as the primary cause. Only 54 percent of enterprises have centralized AI agent governance in place today.

An agent operates differently from a traditional application: it acts autonomously, makes decisions, calls external services, modifies data, and triggers downstream processes. An ungoverned agent is not a misconfigured application, it is an autonomous actor with enterprise system access and no accountability framework. The risk is categorically different from anything that came before.

A platform that already manages identity, integration, and policy for the application estate is the only viable foundation for governing agent deployments at this scale. Organizations that build their agent infrastructure on ungoverned, point-to-point foundations are not just creating application debt. They are creating autonomous liability.



AI creates app abundance.
Abundance without a platform becomes sprawl.

17 · The counter-intuitive truth: AI is driving platform adoption

The most common objection to the platform argument in the AI age is an intuitive one: if AI makes application creation cheap and accessible, surely it makes the specialized tooling of a platform less necessary, not more. This objection is empirically wrong, and the data that refutes it is recent.

The DORA 2025 Report found that 90 percent of enterprises now have internal developer platforms. The reason for this acceleration is not coincidental. As AI increases the volume and velocity of application creation, the governance burden of managing the resulting estate exceeds what any non-platform approach can handle. The platform is not the alternative to AI adoption, it is the infrastructure that makes AI adoption sustainable.

The Stanford 2026 AI Index reinforces this finding from a different angle. Sixty-two percent of organizations cited governance as the number one barrier to scaling AI adoption, outranking technical limitations by 24 points. The limiting factor in AI deployment is not the AI. It is the organizational infrastructure to deploy and manage it responsibly. That infrastructure is the platform.

The cheaper code becomes, the more expensive inconsistency becomes.

18 · The 10x risk: What walking away looks like

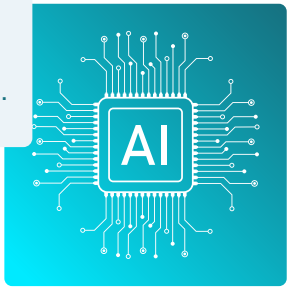
The decision to move away from platform governance is not a decision to maintain the current state. It is a decision to accept a rapidly deteriorating trajectory as AI accelerates application creation without the infrastructure to manage what gets created.

Today, on the platform, the estate consists of 150 managed applications, 1,000 repositories governed under a common framework, a centralized identity model, 150 managed integration contracts, and the operational infrastructure: monitoring, alerting, deployment pipelines, rollback capability, that makes the estate maintainable.

In a two-to-three-year AI-accelerated future without the platform, the same business pressures that today produce 150 applications produce 300 or more. Those 300 applications require individually managed identity, individually maintained integration contracts, individually assembled compliance documentation, individually operated monitoring infrastructure, and individually handled change events. The math is not linear. It is compounding.

The cost difference between these two states is not the cost of the platform. The cost difference is the compound cost of doing everything the platform does, individually, for every application that AI creates. The platform is cheap by comparison to its alternative.

> The platform doesn't add cost. It prevents cost from compounding.



Token bleed: The runtime cost that arrives after the demo

A significant corporate cost center has emerged that is absent from most enterprise AI business cases: uncontrolled token burn.

When individual business units build standalone AI-assisted applications with direct, point-to-point connections to LLM APIs and enterprise backends, those applications make un-cached, un-optimized, and unbudgeted calls on every execution. The cost is not visible at build time, it is a variable runtime expense that scales with usage, multiplies with every new app, and arrives as an infrastructure bill that no one planned for.

By 2028, the cost of running AI coding tools and AI-assisted applications will overtake the average developer's salary due to soaring LLM token consumption and consumption-based infrastructure licensing. This is not a distant projection - organizations deploying AI applications today are already encountering unbudgeted runtime costs that undercut the savings from faster development.

A platform addresses token bleed by acting as a shared control plane: caching frequently used data paths, standardizing model interactions across the app estate, applying token budgets at the governance layer, and ensuring that the variable cost of AI execution is visible, bounded, and optimized. The enterprise that runs 150 ungoverned AI apps is not running 150 apps, it is running 150 independent, unoptimized cost centers with shared liability and no shared cost control.

The ROI case: Platform guardrails deliver 29% higher AI returns

The IBM Institute for Business Value's research provides the clearest commercial framing of the platform argument: this is not a cost decision, it is an ROI decision.

Enterprises that explicitly account for and manage technical debt via systematic platform guardrails project 29% higher AI ROI than those that do not. Conversely, bypassing platform governance results in an 18-29% decline in realized ROI due to post-deployment maintenance, remediation, security incidents, and integration failures that accumulate as the app estate grows.

The mechanism is straightforward. The cost of building an AI-assisted app is compressed, this is the benefit the customer can see. The cost of running, governing, securing, integrating, and maintaining that app across its operational lifetime is the cost the customer cannot see until it arrives. Platform guardrails move the invisible cost forward and make it manageable; ungoverned development defers the cost and makes it compounding.

Framed for a CFO: the platform is not an infrastructure expense added to the cost of AI development. It is the mechanism that preserves the ROI that AI development promises. Without it, the savings on Day 1 are real; the bill on Day 100 is larger.

19 · Industry convergence: The market has decided

The case for platform governance in the AI age is not an HCLSoftware argument. It is the conclusion that the entire enterprise technology industry has independently reached, across multiple major vendors, research organizations, and enterprise deployments.

At Google Cloud Next 2026, the central message was governance. Not models, not compute, not feature releases, governance as core infrastructure, with models repositioned as a commodity. The framing was deliberate: the differentiator in the AI age is not the AI. It is the infrastructure that governs how the AI is used.

Bain's 2026 enterprise AI research named 'the agentic enterprise control plane' as the defining architectural requirement of the year, the infrastructure layer that governs agent identity, permissions, integrations, and accountability. ServiceNow launched 'AI Control Tower' as a named product category at Knowledge 2026.

JPMorgan Chase's AI deployment provides the most concrete data point. The firm has committed $1.8 billion to AI investment, deployed more than 450 use cases, and made AI available to more than 200,000 employees. The architecture is explicitly governance-first: centralized identity, governed integrations, and policy enforcement at the platform layer.

This convergence matters in a customer conversation because it removes the objection that platform governance is a vendor preference or a legacy architectural bias. It is where the entire industry has arrived independently.

20 · Partner vs. end-customer value

The platform delivers distinct value propositions to the partner and the end customer. These propositions are compatible and complementary, but they should be articulated separately, because they address different decision-makers with different priorities.

AI productivity is the partner's build advantage. Platform governance is the customer's ownership advantage. Sell them separately.

Partner: build advantage	Customer: ownership control
Higher build throughput through reusable accelerators, shared components, and AI-assisted development on a governed foundation.	Lower lifecycle risk through governed security, centralized identity, documented integrations, and auditable compliance evidence.
Fewer late-project surprises: integration patterns and security controls are established at the platform level, not discovered during integration test.	Auditability and regulatory confidence: platform-governed apps produce compliance evidence as a byproduct of normal operation, not as a manual pre-audit exercise.
Delivery margins protected by reuse: each new application built on shared platform infrastructure costs less to deliver than a greenfield bespoke build.	Team continuity: platform patterns allow any team member, new partner, or future developer to inherit and support any application in the estate.
Capability differentiation: partners who deliver governed AI-accelerated apps on Foundry are selling a fundamentally different offering from those who deliver ungoverned AI-assisted builds.	Portfolio TCO control: platform cost is largely fixed; per-app overhead without a platform is variable and compounds with every new application.

21 · Navigating strategic considerations in AI platform adoption

As enterprises transition from manual coding to AI-assisted development, evaluating the role of platform governance is a critical step. Leadership teams must weigh the immediate gains of AI acceleration against the long-term operational costs of maintaining the resulting application estate.

The following table outlines recurring strategic considerations that arise during this architectural shift, contrasting common market assumptions with empirical data and operational realities.

Strategic consideration	Architectural reality
If AI builds apps, why do we need a platform?	AI creates abundance. Abundance without a platform becomes sprawl. The question is not whether AI builds apps, it does. The question is whether the organization can govern, secure, and maintain what AI creates. The platform is the answer to that question.
Can't AWS or cloud-native services do this?	Yes, but then you build and maintain the platform yourself: identity management, integration contracts, policy enforcement, governance reporting, deployment pipelines, and monitoring infrastructure. Forrester puts the failure rate for build-your-own governance at 75%.
Why should the customer care about developer productivity?	They shouldn't and we agree. What they care about is faster time-to-change when business conditions shift, lower lifecycle risk as the application estate grows, and the organizational capacity to maintain what gets built. The platform addresses all three.
Does this create lock-in?	The alternative: ungoverned applications built by individual developers with individual tooling creates deeper lock-in to specific individuals. Foundry's patterns are the portable architecture. The lock-in question applies more forcefully to the ungoverned alternative.
Won't platform cost more as we scale?	Platform cost is largely fixed. Per-application overhead without a platform is variable and grows with every new application. At 10 applications, the difference is manageable. At 300, the platform is the cheaper option by a significant margin.
Is this just middleware?	It is the control plane for the AI application factory. Identity, integration, policy, runtime, observability, governance, and reuse are the capabilities that determine whether the organization owns its AI-generated estate or is owned by it.
Our engineers review AI code, so we catch the problems.	Research across 302,579 commits shows that 22.7% of AI-introduced bugs survive human review due to automation bias. Engineers trust clean-looking AI code. Platform guardrails at the infrastructure layer are the only reliable quality gate.
AI development is saving us money, why add platform cost?	IBM data: enterprises managing AI technical debt via platform guardrails deliver 29% higher AI ROI. Bypassing the platform costs 18–29% in real ROI through post-deployment maintenance and remediation, money you don't see until Day 100.
Won't our API management handle the integration sprawl?	AI apps generate 5× more APIs than non-AI enterprises. Only 10-20% are documented. API management governs what you can see. A platform prevents the 80% you can't see from being created in the first place.

22 · Metrics to prove the platform case

Governance arguments require evidence, and evidence requires metrics. The following table defines the measurements that allow partners and customers to demonstrate platform value in concrete terms, before and after platform adoption.

Metric	What it measures	Platform advantage
Time-to-Production	Days from approved use case to governed production release	Reusable patterns eliminate repeated setup work; integration and identity are pre-built
Integration Reuse Rate	Percentage of applications using Foundry service contracts vs. one-off custom connections	Direct measure of integration debt avoided; each point of reuse is a point-to-point connection that was not created
Audit Hours Saved	Time spent assembling compliance evidence per release, before and after platform adoption	Platform automates evidence generation; manual assembly is unscalable at 150+ applications
Support Cost per App	Run and support hours per application per release cycle	Platform reuse reduces marginal cost of each new application; shared patterns mean fewer unique support scenarios
Security Posture	Critical vulnerabilities, exposed secrets, and policy violations per release, caught before production	Platform-level security controls catch issues before deployment; ungoverned bespoke development catches them after breach
Reuse Ratio	Percentage of applications using shared components, themes, and workflows vs. custom implementations	Multiplier on platform investment; each shared component is a cost avoided in every application that uses it
Change Propagation Time	Time to implement one system-level change across all dependent applications	Platform: one update, all applications inherit. Without platform: one update per application, multiplied by the size of the estate

23 · Bridging speed and control: The HCLSoftware approach

The architectural imperatives discussed throughout this whitepaper represent the core design philosophy behind HCL Mosaic Vibe Studio. By pairing AI-accelerated application creation with the Volt MX Foundry backend services layer, the platform governs what gets created. Speed and control are no longer competing priorities. They are dual properties of a unified environment.

The mapping between the strategic challenges of the AI age and HCLSoftware's platform capabilities is direct:

Governance by default, not review:
Manual code review scales linearly, while AI generation scales exponentially. HCL Mosaic Vibe Studio applies governance at the infrastructure layer so applications inherit policy rather than implementing it independently. This supports a platform-mediated operating model where a central team sets the defaults and delivery teams build at their own pace within those guardrails.

Deterministic Security:
Ungoverned AI-assisted development produces applications with missing security controls at a rate human review cannot reliably catch. HCL Foundry constrains what generated code can do by enforcing centralized identity, automated secrets handling, and pre-approved integration patterns. Reviewers can then focus on validating business logic rather than reconstructing security architecture per commit.

Automated Compliance:
Manual evidence assembly cannot survive an estate of 300 or more applications. The platform makes compliance a byproduct of routine operation. Automated audit trails, access records, and change histories provide the enterprise with defensible evidence to satisfy documentation rules, data protection inquiries, and sector-specific supervisory audits.

AI Lifecycle Management:
The pre-AI estate did not require model lifecycle tracking, prompt governance, or inference residency controls. HCLSoftware routing solves this by directing model access through a governed platform service rather than relying on per-application credentials. The enterprise can definitively state which model version an application uses, what context it transmits, and exactly where inference occurs.

Platform Consistency:
Cognitive debt is the hidden cost of AI-assisted development. Consistency is the antidote. Applications built on shared HCL Foundry are instantly legible to engineers who did not originally write them, ensuring the estate remains transferable across teams, deployment partners, and future generations of staff.

Enterprise Scale:
Estate growth naturally compounds integration pairs, shadow APIs, agent sprawl, and token bleed. A platform hub architecture holds integration connections at a linear scale rather than a quadratic one. It keeps interfaces documented by default, absorbs system updates centrally, extends identity controls to autonomous agents, and bounds the runtime costs of LLM token consumption. For existing HCLSoftware customers, this governed architecture allows organizations to securely apply AI acceleration directly to the mission-critical applications and digital experiences they have already built.

The objective is not to slow down AI-assisted development. The objective is to ensure that the resulting speed delivers an enterprise asset the organization can actually own, govern, and trust.

24 · Strategic next steps: Assessing your AI governance posture

If the architectural challenges outlined in this paper align with the current trajectory of your software estate, the most useful next step is an immediate operational assessment.

Before evaluating specific vendor platforms, initiate an internal inventory of the AI-assisted applications currently running in your production environment. For each application, test whether your organization can rapidly answer three fundamental questions:

1. **Data Lineage:** Exactly what corporate or customer data does this application touch?
2. **Access Control:** Who, or what autonomous agent, is authorized to access it?
3. **Model Dependency:** Which specific AI model version does this application depend on to function?

An inability to answer these questions reliably and rapidly across the portfolio indicates a structural governance gap. The results of this simple inventory will tell you more about your actual security posture than any theoretical briefing.



25 · Take action on your AI strategy

Governance cannot wait for the application estate to sprawl beyond repair. Transition from passive observation to active control with HCLSoftware.

Explore HCL Mosaic Vibe Studio: See a live demonstration of how HCL Mosaic Vibe Studio delivers AI-accelerated development with enterprise-grade governance.

Learn more

Contact Us

26 · Source notes

Developer Productivity and AI Adoption
GitHub Copilot Controlled Study: 55% faster task completion. GitHub Research, 2023. github.blog/research.

Developer AI tool adoption (84%): Stack Overflow Developer Survey 2024; GitHub Octoverse 2024.

Google internal AI code generation: 75% of new code AI-generated and engineer-reviewed. Google corporate reporting, 2026.

Global production code written by AI (46%): Industry aggregate, multiple sources, 2026.

Code Volume and Automation Bias
IBM Institute for Business Value: 81% of executives report technical debt constraining AI success; 69% believe it will render some AI initiatives financially untenable.

Vibe Coding and Security Research
Menlo Security (2025): 68% of employees use unsanctioned AI tools; 57% input sensitive corporate data. Menlo Security State of Browser Security Report, 2025.

Code Security Research
Veracode GenAI Code Security Report (2025): 45% of AI-generated code contains OWASP Top 10 vulnerabilities. Rate unchanged across testing cycles. Veracode, 2025.

AI-assisted commits at 3–4x rate, security findings at 10x rate; 37.6% increase in critical vulnerabilities after five AI-assisted refinement rounds. Cloud Security Alliance AI Security Report, 2026.

Financial Impact
Shadow AI breach cost $4.63M average, $670K above standard breach baseline; 65% PII exposure vs. 53% typical. IBM Cost of a Data Breach Report 2026 / TechTarget Shadow AI analysis, 2026.

EY Responsible AI Pulse Survey: 99% reported financial losses; 64% lost >$1M; average realized loss $4.4M. EY, October 2025. 975 C-suite leaders, 21 countries.

IBM IbV: enterprises with platform guardrails project 29% higher AI ROI; bypassing governance costs 18–29% in realized ROI. IBM Institute for Business Value, 2026.

API sprawl: 5× API growth in AI enterprises; 10–20% documented; $100B+ global losses from ungoverned APIs; 31% of cyberattacks exploit undocumented APIs. Boomi/IDC joint research, 2026.

Regulatory
EU AI Act: maximum fine €35M or 7% of global annual turnover for prohibited AI practices. Official EU regulation text, effective August 2024.

Japan METI AI Governance Guidelines: Ministry of Economy, Trade and Industry, AI Governance Guidelines v1.1 (2023) and 2024 updates. meti.go.jp.

Platform Adoption and Agent Scale
Stanford 2026 AI Index: governance cited by 62% as the #1 barrier to scaling AI, outranking technical limitations by 24 points. Stanford Human-Centered AI Institute, AI Index Report 2026.

Industry Convergence
Google Cloud Next 2026: governance as core infrastructure, models as a commodity. Google Cloud Next keynote, April 2026.

Bain: 'The agentic enterprise control plane comes into view.' Bain Technology Report, 2026.

ServiceNow Knowledge 2026: AI Control Tower launch. ServiceNow Knowledge 2026 conference, May 2026.

JPMorgan Chase: $1.8B AI investment, 450+ use cases, 200,000+ employees, governance-first. JPMorgan Chase Annual Report 2025 and CIO public statements.

Integration Mathematics and Build-Your-Own Governance

Forrester Research: 75% failure rate for build-your-own governance infrastructure. Forrester Wave: Enterprise Platform Engineering, 2025.

Debt Behind the AI Boom study:
"Debt Behind the AI Boom: A Large-Scale Empirical Study of AI-Generated Code in the Wild." Singapore Management University et al. arXiv:2603.28592, April 2026. arxiv.org/abs/2603.28592

Red Access vibe coding scan:
RedAccess Security Research, May 2026. Scan of 380,000 publicly accessible web assets; ~5,000 corporate-purpose assets found leaking sensitive data. Covered by The Hacker News (May 2026) and multiple security outlets.

5,600 vibe-coded apps study (CSRF / security headers):
Cloud Security Alliance Research Note: "Vibe Coding Security Crisis: Credential Sprawl and SDLC Debt." March 2026. labs.cloudsecurityalliance.org (public PDF available)

DORA 2025:
DORA State of AI-Assisted Software Development Report 2025. Google / DORA Research. dora.dev/dora-report-2025 — confirms 90% of organizations now use internal developer platforms.

Gartner agent sprawl (150K agents, 40% canceled):
Gartner Press Release: "Gartner Identifies Six Steps to Manage AI Agent Sprawl." April 28, 2026. gartner.com/en/newsroom/press-releases/2026-04-28-gartner-identifies-six-steps-to-manage-artificial-intelligence-agent-sprawl
Gartner Press Release: "Gartner Predicts Over 40% of Agentic AI Projects Will Be Canceled by End of 2027." June 25, 2025. gartner.com/en/newsroom/press-releases/2025-06-25-gartner-predicts-over-40-percent-of-agentic-ai-projects-will-be-canceled-by-end-of-2027

Gartner token cost / developer salary prediction:
Gartner Press Release: "Gartner Predicts AI Coding Costs Will Surpass Average Developer's Salary by 2028 as Token Consumption Surges." June 24, 2026. gartner.com/en/newsroom/press-releases/2026-06-24-gartner-predicts-ai-coding-costs-will-surpass-average-developer-salary-by-2028-as-token-consumption-surges

HCLSoftware

About HCLSoftware

HCLSoftware, the software division of HCLTech, is a global enterprise software product business. By orchestrating Xperience (X), Data (D), and Intelligent Operations (O) using Agentic AI, HCLSoftware unlocks insights, streamline processes, and accelerates business outcomes across 130+ countries, serving a large base of global customers including the Fortune 100/500/G2000. HCLSoftware develops, markets, sells, and supports 60+ transformative software products spanning five core areas: Xperience, Data, Intelligent Operations, Lotus, and Specialized Software. Its engineering excellence is consistently recognized as a Leader by Gartner® across multiple categories.

Learn more at **hcl-software.com**