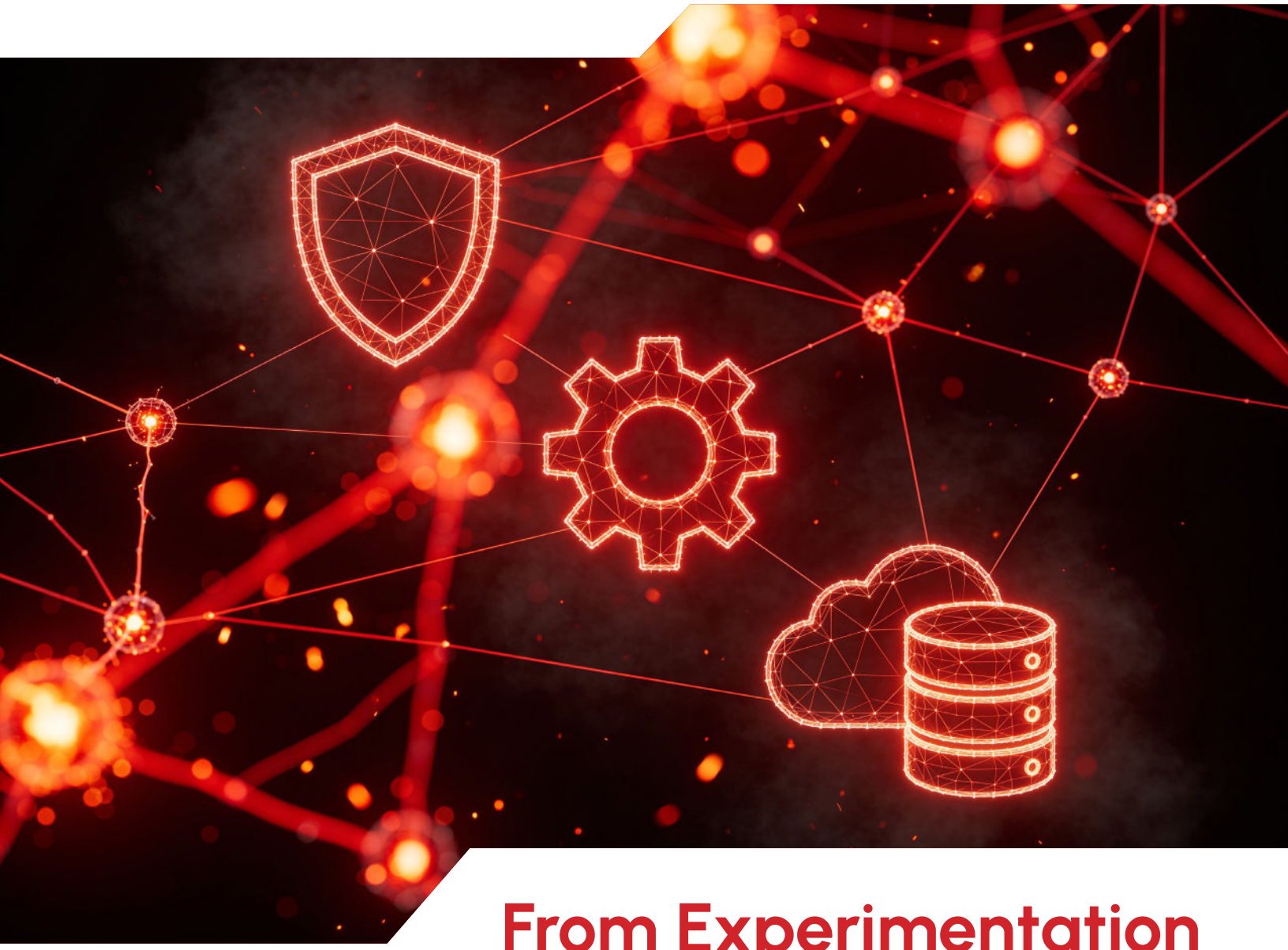
Futurum®

Red Hat



From Experimentation to Execution:

Platform Engineering for Scalable Generative AI

AUGUST 2026

Executive Summary

Enterprise generative AI has moved past the point where model capability was the bottleneck. The constraint now is operational: the ability to deploy, govern, and run AI systems reliably across production environments. Investment in data science capability does not, on its own, produce the platform infrastructure required for repeatable production deployment. As GenAI programs mature, operational execution becomes the test of whether investment converts to business outcomes.

Futurum's 1H 2026 AI Decision-Makers Survey shows roughly 52% of organizations still in awareness or experimentation stages of GenAI maturity. The gap is not model access. It is the absence of platform infrastructure to absorb AI workloads alongside the rest of enterprise IT. Infrastructure, governance, security, and reproducibility are the primary bottlenecks. Rapid adoption of public model APIs outside formal IT channels has created a familiar set of risks: fragmented security postures, unmanaged token spend, and data exposure that bypasses enterprise controls. Futurum sees the pattern repeating early-cloud Shadow IT dynamics.

The move to production introduces workload characteristics that pilot tooling does not address. Production GenAI introduces composite architectures, agentic workflows and inference economics that require runtime governance. These are platform engineering problems, not model selection problems.

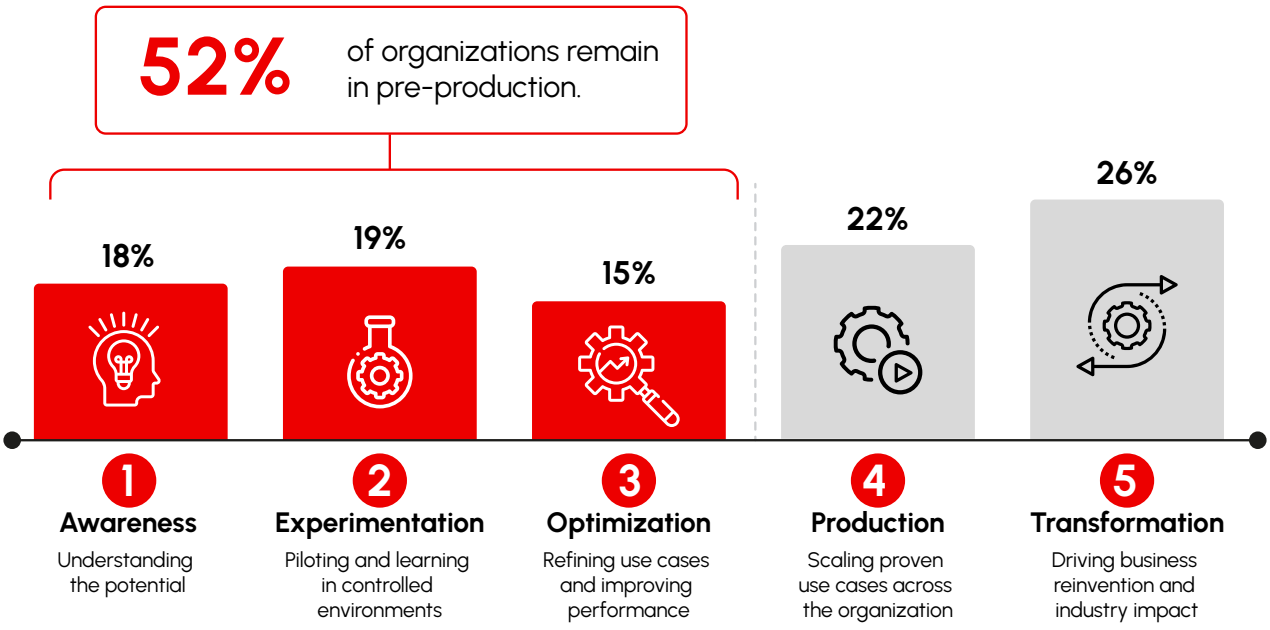
The recommendation is structural. Platform teams should leverage proven commercial technologies and platforms to extend engineering disciplines, GitOps, CI/CD, infrastructure-as-code, observability, and policy-as-code, to absorb GenAI workloads as a new workload class rather than constructing parallel AI stacks. Platform engineering becomes the scaling mechanism for enterprise AI, and the foundation for converting GenAI investment into repeatable business outcomes.

The Shift: GenAI Moves from Models to Systems

The Limits of Experimentation: Early GenAI proofs of concept were designed to validate model capability, not to operate as production systems. They were typically isolated from enterprise integration, governance, and lifecycle requirements. The pilot validates that a model can do something. It does not validate that the organization can run it.

The data confirms the pattern. Futurum's 1H 2026 AI Decision-Makers Survey places roughly 52% of organizations in Awareness or Experimentation/Optimization (Stages 1–3 of Futurum's five-stage GenAI maturity model). They are piloting or fine-tuning but have not standardized GenAI for regular operational use. Pilot-to-production conversion lags pilot density by a wide margin, and the lag widens as more organizations enter early stages without a corresponding rise in mature deployment (Figure 1).

Figure 1. Enterprise GenAI Maturity Distribution



Source: Futurum Research (2026). 1H 2026 AI Decision-Makers Survey.

What Changes in Production: GenAI becomes a composite system: model, retrieval, orchestration, integration, and increasingly, agentic execution. Each layer introduces operational characteristics absent from pilots. Multi-model strategies require routing logic that balances cost, latency, and capability tradeoffs in real time. Inference at scale introduces capacity, cost, and performance constraints that GPU-bound workloads make acute. Agentic workflows increase unpredictability and demand runtime governance, including identity, sandboxing, tool-call authorization, and human oversight.

Organizational Implications: Data science organizations are typically optimized for model development and experimentation, not for production operations. The handoff between data science and platform engineering can produce gaps in tooling, incentives, and risk tolerance when no shared operating model is in place. Without that shared model, the early-cloud Shadow IT pattern tends to reappear: line-of-business teams procure public model API access directly, bypassing enterprise controls for cost, security, and policy. The result is fragmented governance, duplicated infrastructure, and unmanaged risk.

Strategic Takeaway: The bottleneck shifts to security, reproducibility, governance, and integration. Concretely, the choke points move from model selection to GPU scheduling and orchestration, token-level cost attribution, tool-call authorization, and AI-specific observability. Platform engineering, the discipline that already governs how enterprises run mission-critical workloads, becomes the scaling mechanism for GenAI. Organizations that recognize the shift early absorb AI into existing platform disciplines. Those that do not accumulate parallel infrastructure and stall in pilot.

The Platform Imperative: Why GenAI Is an Infrastructure Problem

As GenAI initiatives move from sandbox to production, the engineering profile shifts. Enterprise AI behaves as a distributed system, with the operational characteristics that implies. To scale, GenAI demands a platform engineering mindset.

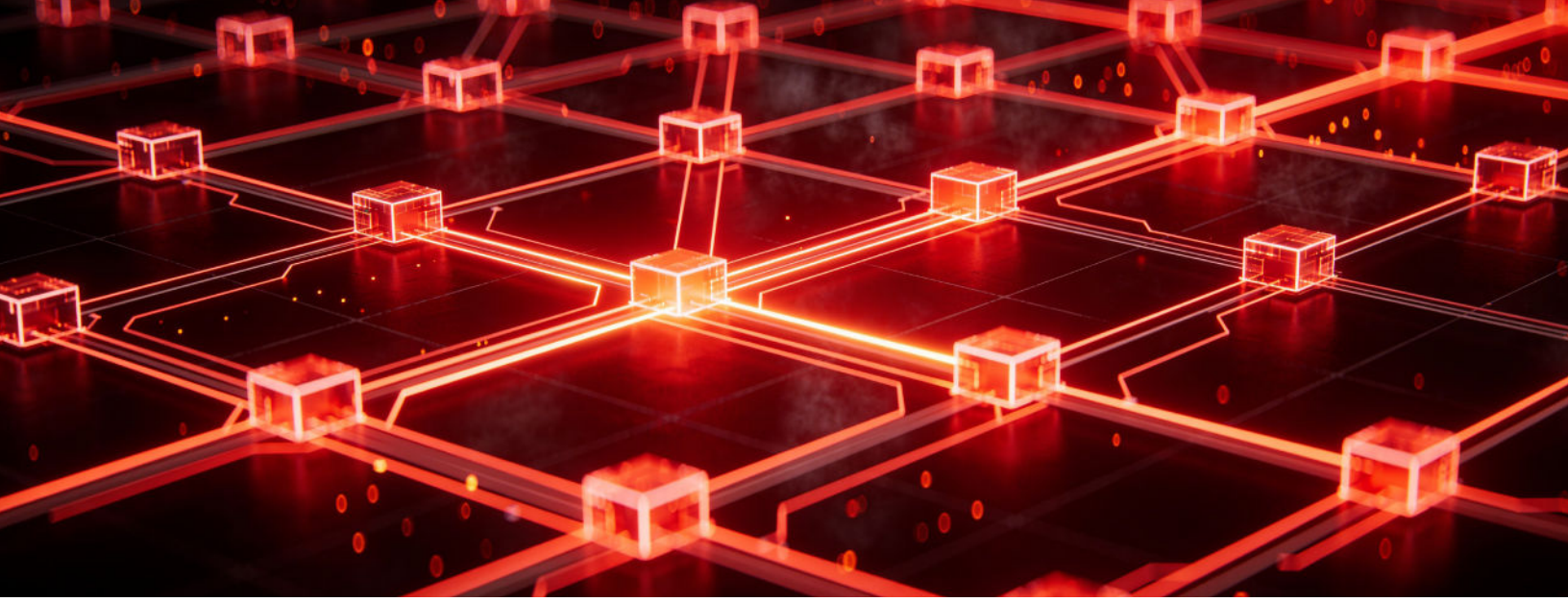
GenAI as a Distributed System: AI workloads require orchestration across compute, storage, APIs, and data pipelines. Complexity increases with retrieval-augmented generation, agents, and multi-step workflows that span multiple services. Agentic systems add tool invocation, state, and the need for execution sandboxing and human-in-the-loop checkpoints to prevent autonomous actions from creating operational havoc.

The Risk of AI Silos: Isolated AI stacks create concrete failure modes that platform-mediated approaches prevent. Unmanaged token spend accumulates without a chargeback to the consuming line of business, producing cost surprises at quarter close. Audit trails disappear when a provider deprecates a model version, changes pricing, or modifies safety filters without enterprise notification. To solve the 'Shadow AI' crisis, enterprises must transition from 'token consumers' into internal 'token providers' running governed AI platforms.

Platform Engineering as an Abstraction Layer: Platform engineering provides standardized deployment paths and reduces cognitive load for AI development teams. Self-service capabilities enable developers and data scientists to consume models, infrastructure, and tools without deep platform expertise. A centralized API gateway brokering access across internal models and public providers, with consistent identity, rate limits, cost attribution, and policy enforcement, gives organizations a single point of control over how models are consumed in a Model-as-a-Service architecture. The same pattern supports regulated and disconnected environments where workload sensitivity requires it.

Infrastructure Economics: GPU scarcity and high costs demand centralized scheduling and utilization controls. Without platform-level orchestration, GPU resources fragment across teams, driving utilization down and cost up. A platform approach improves efficiency, supports chargeback and showback, and avoids the duplication of parallel AI infrastructure and orchestration frameworks — such as vLLM, NVIDIA TensorRT-LLM, SGLang, and llm-d. These solutions dynamically separate prefill and decode phases, reuse KV cache, and maximize distributed GPU throughput.

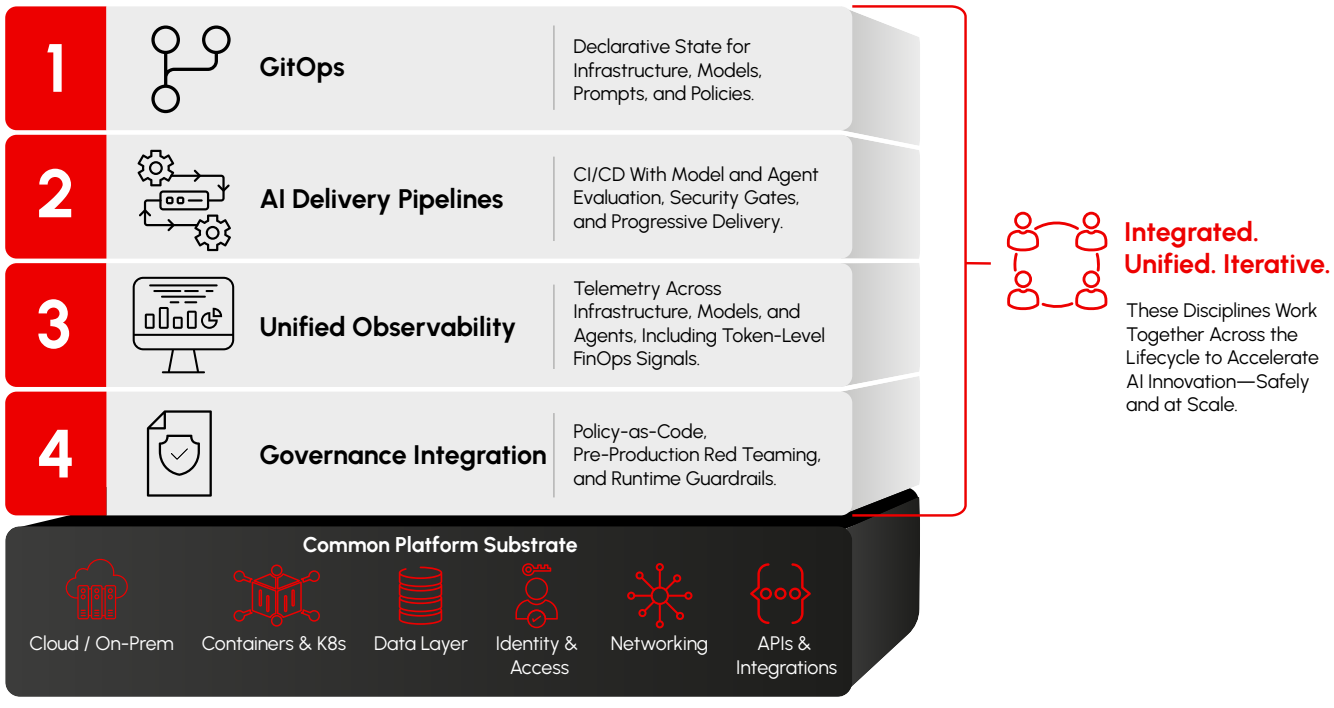
These challenges shift the evaluation criteria for enterprise AI. Success depends less on model selection and more on the platform capabilities that govern deployment, operations, cost control, and observability at scale. The next section examines the technologies and engineering disciplines that make that possible.



Reproducibility and Control: GitOps, Delivery, and Observability for GenAI

Once a GenAI model reaches production, operating it reliably is a platform engineering problem. The disciplines that carry this work, GitOps, declarative delivery, inference runtime management, unified observability, and embedded governance, apply to GenAI the same way as they apply to any other enterprise workload (Figure 2). They are existing disciplines extended to a new workload class.

Figure 2. Operating GenAI as a Platform Workload



Source: Futurum Research

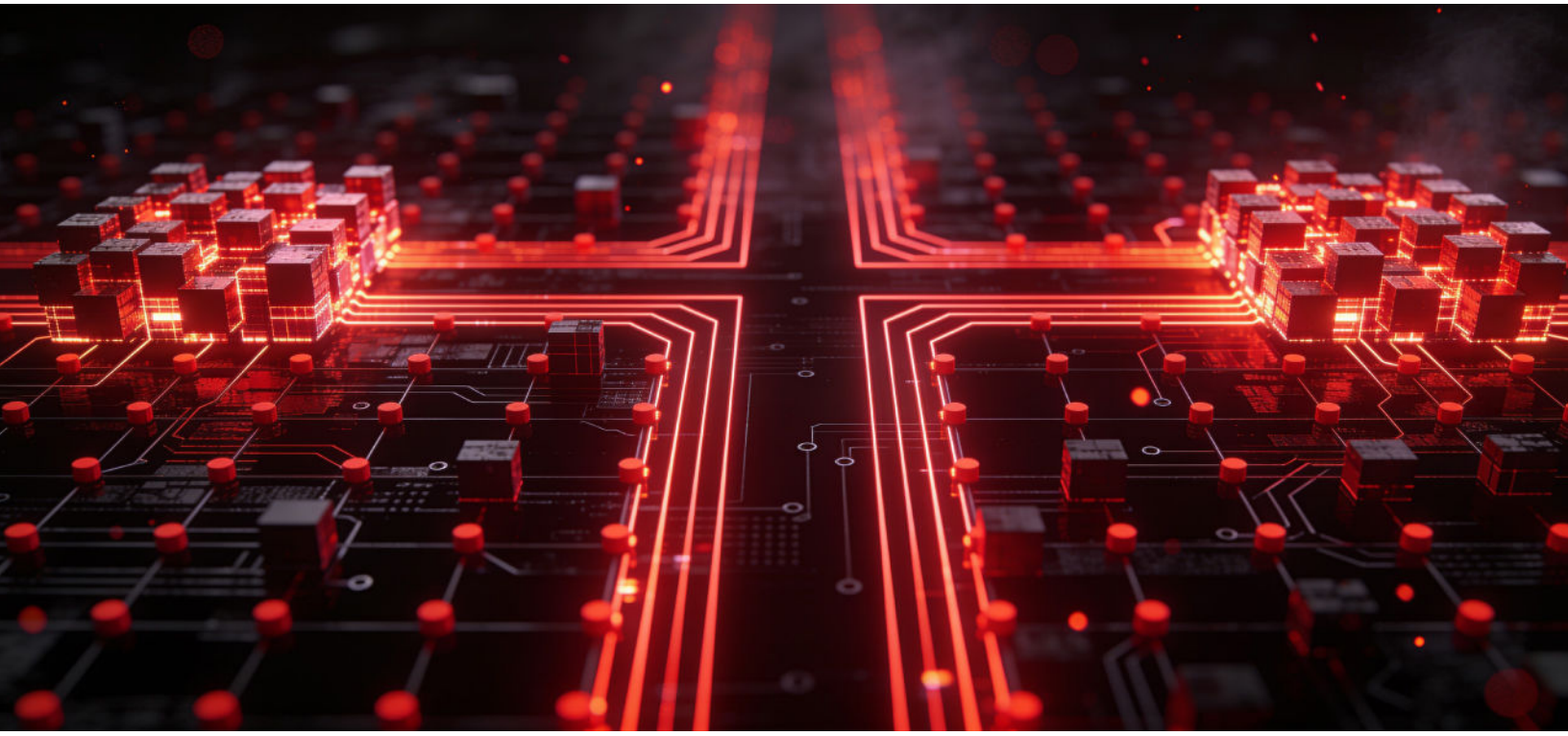
Declarative Platform Model (GitOps): Git becomes the source of truth, not only for infrastructure, but also for generative AI assets. This declarative deployment pattern should extend to model serving, routing configurations, and token quota management. The same Kubernetes-style reconciliation loops that govern application infrastructure govern AI workloads.

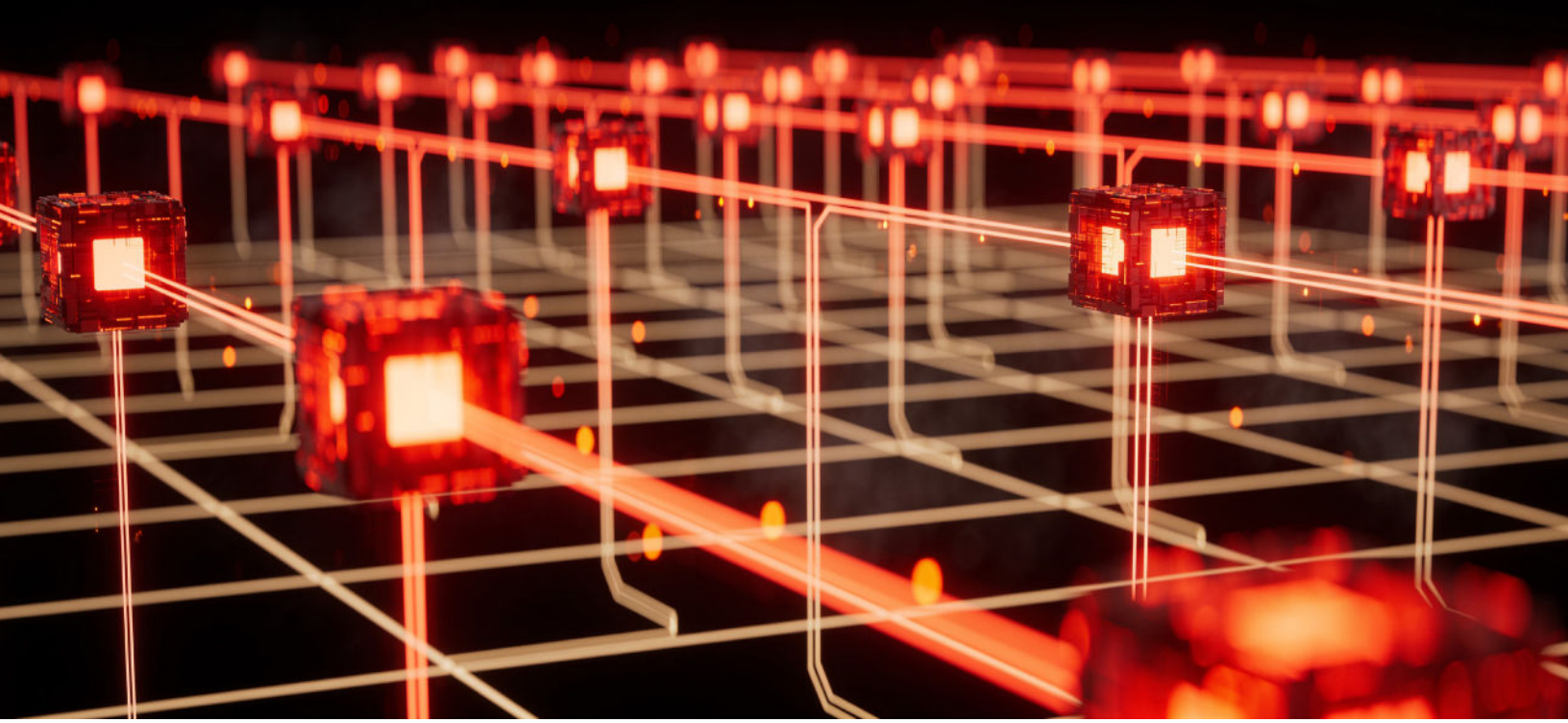
AI Delivery Pipelines: CI/CD pipelines extend with model and agent evaluation, testing, and security gates. Progressive delivery patterns, canary deployments, blue-green rollouts, and feature flags apply to models and prompts as much as to code. Rollback paths must include both application and model versions to enable recovery when behavior degrades.

Model Serving and Runtime: A standardized serving layer handles multi-model and multi-version inference. Routing, autoscaling, and batching shift to the platform layer, where Kubernetes operators and custom schedulers handle them. Inference workloads differ structurally from traditional services. Stateless request patterns combine with stateful serving state, including KV cache reuse and session affinity for prompt caching, creating orchestration requirements that conventional load balancing does not handle. Batch inference belongs in the same serving infrastructure as interactive inference, supporting data-processing workloads that do not require real-time response.

Unified Observability: Because AI systems are unpredictable, traditional monitoring is insufficient. Platform engineers must implement standards-based telemetry, such as OpenTelemetry, to capture deep execution traces. In Futurum's 1H 2026 Software Lifecycle Engineering Decision-Maker Study, AI observability ranks as a top priority at 37.4% and AI agent observability at 30.9%. To ensure auditability and drift detection, this observability stack must continuously log token consumption, reasoning chains, model latency, tool invocations, and user prompts. Token usage becomes a first-class FinOps signal, with showback supporting cost attribution across open-weight model usage and public-API spend.

Governance Integration: Governance happens at two stages. Pre-production automated red teaming and adversarial scanning catch prompt injection, jailbreak, and policy bypass risks in CI/CD. Runtime guardrails act as a proxy to filter sensitive data, enforce business policies, and prevent toxic or non-compliant outputs. Policy-as-code embeds governance in the platform itself rather than treating it as a separate compliance function. Auditability extends across hybrid environments.





Data Pipelines and Model Lifecycle at Scale

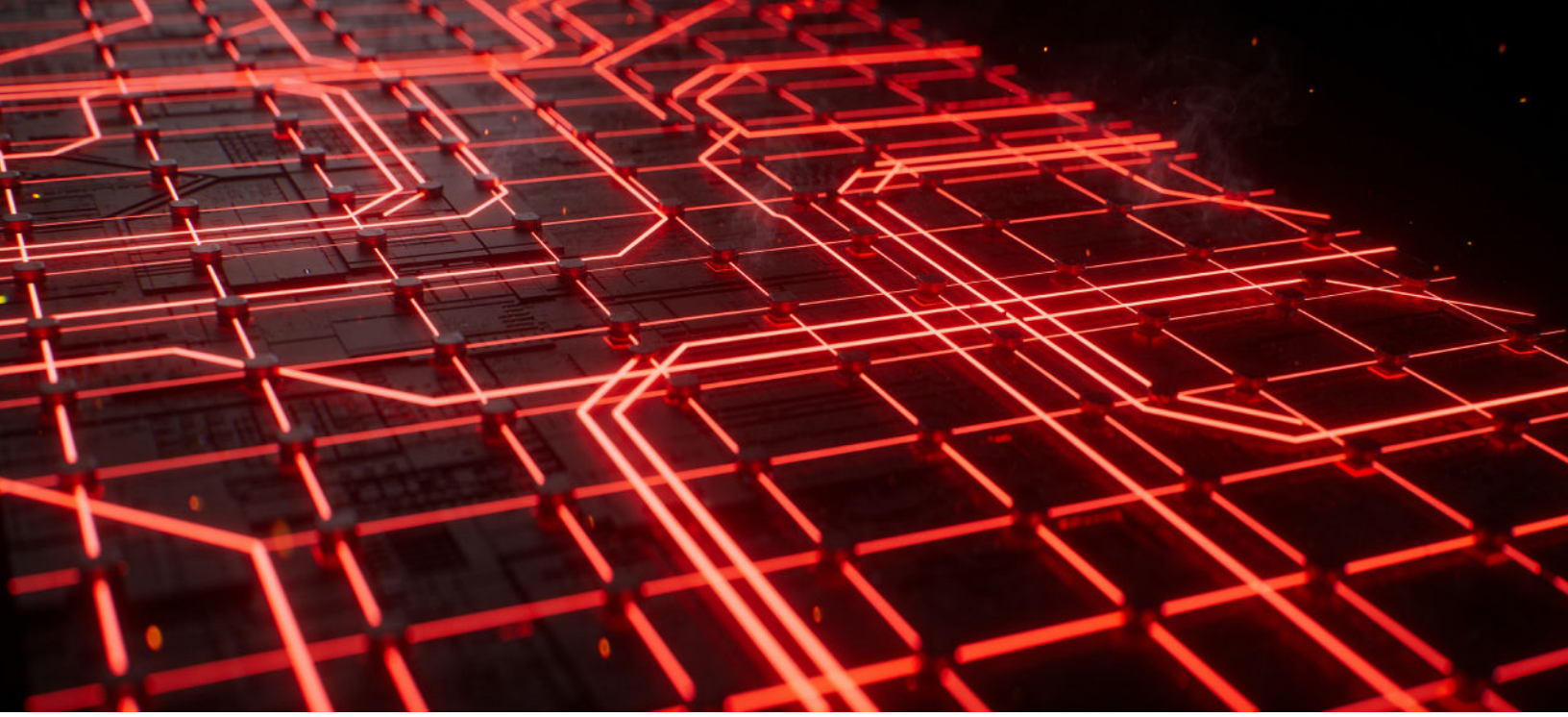
A generative AI model deployed without a continuous, governed flow of fresh enterprise data is more of a demonstration than a business system. Delivery and observability address how systems run. Data pipelines and lifecycle address what feeds them. Models and agentic workflows are living artifacts whose coherence depends on context that stays in sync with the business.

RAG as a Data Engineering Discipline: Retrieval-augmented generation requires scalable ingestion, transformation, and synchronization of enterprise data. Freshness and quality directly impact model performance, and the cost of stale or low-quality context shows up in output reliability. Platform teams should automate the testing of embedding models, chunking strategies, and retrieval techniques programmatically, the same way they automate any other performance tuning exercise.

Data Pipelines as Platform Assets: Pipelines become versioned, testable, and governed like application code. They integrate into platform workflows alongside service deployments, sharing the same CI/CD, observability, and policy infrastructure. Treating pipelines as first-class platform assets eliminates the operational friction that arises when data engineering and platform engineering run on separate tooling.

Lifecycle and Lineage: Model and data versioning enable reproducibility, including under regulatory scrutiny. Feedback loops drive updates as drift, usage, and performance signals accumulate. Lineage tracking spans the full path from source data through embeddings, retrieval, model inference, and downstream action, providing the evidence trail that auditors and operators both require.

Platform Convergence: Data and platform operations share infrastructure and governance. Coordinated scheduling of compute-intensive workloads becomes possible when both share the same control plane and both share the same Kubernetes control plane. GenAIOps is best understood as the evolution of MLOps, not a separate discipline. Utilizing unified lifecycle tooling prevents the operational silos that parallel infrastructure inevitably creates.



Compute Reality: GPUs, Infrastructure, and Execution Constraints

Platform engineering is the operating model for enterprise workloads. GenAI is the newest and most demanding workload class entering that scope, and the top workload targeted for Kubernetes as found in Futurum's 2026 SLE Decision-Maker Study. Treating GenAI as another workload, with specific operational characteristics, prevents the fragmentation of tooling, skills, and governance that specialized AI platforms create.

GenAI as a New Workload Class: GPU dependency introduces cost and performance constraints absent from traditional services. Inference workloads carry latency sensitivity that batch training does not, though both belong in the same serving infrastructure, governed by the same scheduling controls. Inference itself is structurally different from conventional services.

Platform Operating Model Holds: The principles that govern existing platform workloads must be applied to AI. Rather than managing models manually, platform teams should use declarative state, rely on automation for model delivery platforms, and provide AI endpoints to developers as shared services (such as Model-as-a-Service). Policy enforcement should be embedded in areas such as runtime guardrails. Platform maturity is measured by the ability to absorb new workloads without rebuilding the operating model. GenAI tests that maturity. It does not require replacing it.

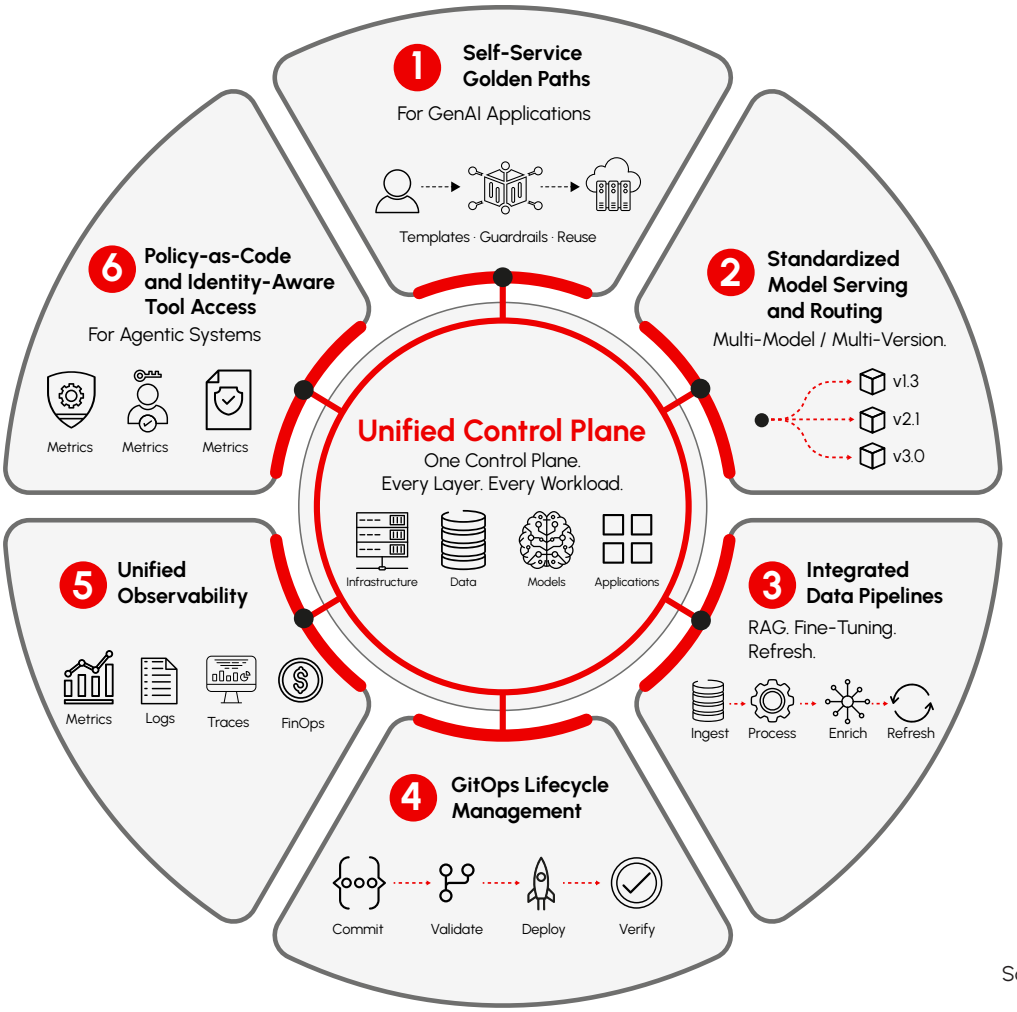
Extension, not Reinvention: Existing platform tools extend for AI-specific needs rather than being displaced. Duplication through parallel AI stacks generates exactly the operational fragmentation that platform engineering exists to prevent.

Hybrid and Workload Placement: Consistent control across cloud, on-premises, and regulated environments is a structural requirement. Workload placement becomes a vendor-neutrality and workload-portability question before it is a technology question. Data locality, regulatory regimes, and cost considerations increasingly influence where AI workloads run, with some enterprises evaluating repatriation from public cloud as inference economics evolve. To avoid lock-in and support workload movement as conditions change, platform teams must build an architecture that provides the flexibility to run any model, on any hardware accelerator, across any hybrid cloud environment.

What Good Looks Like: Platform Capabilities for Scalable GenAI

The target state is a unified platform that operates GenAI workloads alongside the rest of enterprise IT, governed by the same control plane, observability, and policy disciplines. AI systems are treated as composite, production-grade workloads rather than parallel infrastructure projects (Figure 3).

Figure 3. Platform Capabilities for Scalable GenAI



Source: Futurum Research

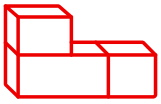
The capabilities cluster around four operational themes. Self-service capabilities let teams deploy GenAI applications combining model, data, and orchestration without re-implementing platform plumbing. Standardized serving and integrated data pipelines support multi-model, multi-version deployment with RAG, fine-tuning, and continuous refresh handled as platform services. GitOps lifecycle management and CI/CD pipelines with embedded evaluation and rollback bring AI workloads under the same delivery discipline as application code. Governance, including policy-as-code, end-to-end auditability, and identity-aware tool access, close the governance loop. For agentic systems specifically, tool access is governed by cryptographic identity and short-lived tokens rather than by prompt-based instructions, which are vulnerable to injection.

Strategic Recommendations

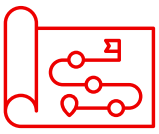
Reframe the Problem: Treat GenAI as a platform and operations challenge that data science teams cannot resolve alone.

Unify the Operating Model: Extend existing platform engineering practices, GitOps, CI/CD, infrastructure-as-code, and observability to AI workloads. Avoid building isolated AI stacks that duplicate tooling and governance.

Operationalize Governance: Embed governance directly into pipelines, runtime controls, and delivery workflows. Treat policy enforcement as continuous and automated, with pre-production red teaming and runtime guardrails as standard pipeline stages.



Invest in Platform Capabilities, not Point Tools: Prioritize reusable platform services, model serving, data pipelines, and observability over fragmented solutions.



Plan for Complexity from the Start: Assume multi-model, hybrid, and GPU-constrained environments. Design for scale, cost control, workload portability, and data locality early.

The Bottom Line

Competitive advantage shifts from model access to operational execution: the ability to deploy, govern, and observe AI systems consistently across environments. The mechanisms that determine which organizations succeed are concrete: GitOps-driven delivery, unified observability with token-level cost attribution, RAG-grade data engineering, GPU-aware scheduling, and embedded governance spanning pre-production red teaming, runtime guardrails, and agent identity controls. Organizations that extend their platform engineering disciplines to absorb GenAI will scale faster and more safely. Those that treat AI as a parallel stack will accumulate complexity and stall at the pilot stage.

Important Information About This Report

AUTHORS

Mitch Ashley
Vice President & Practice Lead, CIO & Technology Buyers, and Software Lifecycle Engineering | Futurum Research

Brad Shimmin
Vice President & Practice Lead, Data Intelligence, Analytics, & Infrastructure | Futurum Research

Nick Patience
Vice President & Practice Lead, AI Platforms | Futurum Research

PUBLISHER

Futurum Research

INQUIRIES

Contact us if you would like to discuss this report and The Futurum Group will respond promptly.

CITATIONS

This paper can be cited by accredited press and analysts, but must be cited in context, displaying author's name, author's title, and "The Futurum Group." Non-press and non-analysts must receive prior written permission by The Futurum Group for any citations.

LICENSING

DISCLOSURES

The Futurum Group provides research, analysis, advising, and consulting to many high-tech companies, including those mentioned in this paper. No employees at the firm hold any equity positions with any companies cited in this document.

Red Hat

ABOUT RED HAT

Red Hat is a leading provider of enterprise open source software, helping organizations build, deploy, and manage applications consistently across hybrid cloud environments. Its portfolio, including Red Hat Enterprise Linux, Red Hat OpenShift, and Red Hat Ansible Automation Platform, provides the foundation for modern application development, infrastructure automation, and AI-enabled operations. Red Hat's approach to scalable AI Ops combines automation, observability, and intelligent operations to help organizations simplify IT management, improve resilience, and operate AI workloads more efficiently at enterprise scale. By enabling consistent platforms and automated operations across on-premises, cloud, and edge environments, Red Hat helps enterprises accelerate innovation while maintaining security, governance, and operational control.

Futurum®

ABOUT THE FUTURUM GROUP

The Futurum Group is an independent research, analysis, and advisory firm, focused on digital innovation and market-disrupting technologies and trends. Every day our analysts, researchers, and advisors help business leaders from around the world anticipate tectonic shifts in their industries and leverage disruptive innovation to either gain or maintain a competitive advantage in their markets.

Futurum®

CONTACT INFORMATION: The Futurum Group LLC | futurumgroup.com | (833) 722-5337