ETIS EXECUTIVE BRIEF SERIES

The Future Software Engineering Organization

From Human-Centered Delivery Teams to Governed Human-Agent Engineering Systems

William T. O'Connell, Ph.D.

Former IBM Distinguished Engineer and CTO, IBM Consulting Quality and Delivery

Creator, Engineering Trustworthy Intelligent Systems (ETIS)

ETIS Executive Brief EB-002 | Version 1.0 | July 2026

Executive Thesis

The software engineering organization of the next decade will not simply be a larger version of today's team with better tools. It will be a governed human-agent system in which people define intent, architecture, risk, and accountability while AI agents perform increasing amounts of bounded analysis, implementation, review, coordination, and operations.

Audience: CTOs, CIOs, heads of engineering, chief architects, product and platform leaders, enterprise risk executives, and board technology committees.

Executive Summary

Software organizations are moving through a structural transition. The first stage of AI adoption equipped individuals with assistants. The next stage gives teams coding agents, review agents, test agents, modernization agents, incident agents, and workflow agents that can act across repositories and platforms. GitHub now describes a unified environment in which developers orchestrate multiple agents, while Microsoft describes a transition from a software factory toward an agent factory [2][5]. The direction is not speculative: delegation, coordination, and agent-mediated execution are becoming product capabilities.

The strategic mistake is to treat this change as a productivity-tool rollout. AI changes the composition of the engineering organization, the unit of work, the shape of management, the architecture of controls, the economics of review, and the skills that remain scarce. DORA's 2025 research characterizes AI as an amplifier: it magnifies the strengths of effective organizations and the dysfunctions of weak ones [1]. Individual coding gains can disappear into downstream disorder when testing, security, architecture, release, and operations cannot absorb the increase in change.

The future organization therefore needs more than AI-fluent developers. It needs clear work architecture, reusable platforms, authoritative context, bounded delegation, evidence-centered delivery, new forms of technical leadership, and workforce systems that develop judgment rather than merely tool familiarity. The winning model is not humans versus agents. It is a deliberately designed human-agent operating system.

For executives, the decision is no longer whether AI will enter engineering. It already has. The decision is whether the organization will redesign around it intentionally or allow local adoption, vendor defaults, and informal delegation to define the future operating model.

The Leadership Question

What work should remain human, what work may be delegated, what evidence must accompany delegated work, and how will the organization preserve accountability as execution becomes increasingly distributed across people and agents?

1. The Organization Is Becoming the Productive Unit

Traditional productivity discussions focus on the individual developer: time saved, code produced, tickets completed, or prompts accepted. That lens is increasingly inadequate. Software outcomes depend on the complete value stream—product definition, architecture, data, implementation, verification, security, release, operations, and learning. Acceleration at one point creates little value if the rest of the system becomes the constraint.

DORA's platform-engineering guidance warns that individual gains are frequently absorbed by downstream disorder [1]. A developer may generate a change quickly, but the organization still has to understand it, review it, integrate it, test it, release it, observe it, and recover from it. If those capabilities are weak, AI increases queue length and coordination cost rather than enterprise throughput.

The relevant unit of transformation is therefore the engineering organization. Leaders must redesign the flow of intent, context, authority, evidence, and feedback across teams. The objective is not maximum agent use. It is a higher-capability organization that can convert accelerated production into reliable business outcomes.

2. From Roles to Work Architecture

Most engineering organizations are designed around stable human roles: product manager, architect, developer, tester, security specialist, release engineer, site reliability engineer, and manager. Agentic systems introduce a new design variable: work can be decomposed and reassigned dynamically among people, specialized agents, deterministic automation, and shared platforms.

This does not eliminate roles. It forces leaders to distinguish role from task. Human accountability remains durable; execution may become fluid. A senior engineer may own a service while delegating code exploration, test generation, documentation updates, and dependency analysis to agents. A release manager may retain the decision right while an agent assembles evidence, detects gaps, and proposes a disposition. A security architect may define policy while automated controls and agents evaluate changes continuously.

Work architecture is the explicit design of this allocation. It defines which tasks are assistive, which may be coordinated by agents, which may be executed under bounded authority, and which remain human decisions because consequence, ambiguity, or accountability requires judgment.

Work class	Primary actor	Executive design requirement
Intent and accountability	Human leaders and accountable owners	Clear purpose, decision rights, accepted risk, named ownership.
Bounded analysis and drafting	Human-agent collaboration	Authoritative context, transparent sources, review expectations.
Repeatable execution	Agents and automation under policy	Identity, least privilege, tests, provenance, exception handling.
Consequential decisions	Humans supported by evidence	Independent challenge, escalation, explicit acceptance.
Continuous operations	Human-agent supervisory model	Telemetry, intervention, recovery, runtime governance.

3. The Developer Role Moves Up the Abstraction Stack

Advanced AI users are already shifting from code producers toward orchestrators of intent, context, delegation, verification, and integration [3]. This does not make technical depth less important. It makes depth necessary for review. Engineers cannot responsibly delegate architecture, implementation, or operations unless they can recognize when the output is wrong, incomplete, insecure, or inconsistent with the system.

The role changes in three ways. First, engineers spend more time defining specifications, constraints, acceptance evidence, and operating boundaries. Second, they direct and review larger units of generated work. Third, they become responsible for the engineering environment that agents use: repositories, instructions, examples, tests, tools, policies, and observability.

The strongest engineers will combine domain knowledge, systems thinking, architecture, review skill, and operational judgment. Coding fluency remains foundational, but it is no longer the complete differentiator. The differentiator becomes the capacity to convert ambiguous business intent into trustworthy system change.

Talent Implication

The organization should stop measuring AI readiness primarily by tool adoption. It should measure whether engineers can specify work, govern context, review generated change, reason about system consequences, and own production outcomes.

4. Technical Leadership Becomes More Important, Not Less

AI increases the volume of technically plausible decisions. That makes architecture, standards, and review more important. Without strong technical leadership, local agent-assisted choices accumulate into dependency sprawl, duplicated capabilities, inconsistent controls, brittle interfaces, and operational complexity.

The future organization needs technical leaders who shape reusable context and paved roads. They define architectural boundaries, approved patterns, platform capabilities, reference implementations, test strategies, policy controls, and evidence expectations. They do not review every line. They design the environment in which good decisions are easier and dangerous decisions are constrained.

This shifts senior technical work from heroic problem solving toward system design for organizational judgment. Distinguished engineers, principal engineers, and chief architects become stewards of the engineering operating system: they connect strategy, architecture, platforms, governance, and operational learning.

5. Platform Engineering Becomes the Organizational Backbone

AI adoption at scale requires a coherent platform. Agents need secure identities, approved tools, repository access, model services, context sources, evaluation harnesses, delivery workflows, observability, and policy enforcement. If every team assembles these independently, the enterprise creates fragmented cost, inconsistent risk, and duplicated learning.

Platform engineering is therefore not merely developer convenience. It is the control and enablement layer for human-agent delivery. DORA's research places platform quality among the capabilities that determine whether AI produces value or disorder [1]. The platform should provide reusable workflows that preserve provenance, enforce review, generate evidence, and make operational feedback visible.

The executive goal is a product-oriented internal platform with clear customers, adoption measures, service objectives, and a roadmap. It should support local innovation while standardizing the controls that should not be reinvented.

Platform capability	Business value	Governance value
Authoritative developer and agent context	Faster onboarding and more consistent output	Reduces hidden assumptions and context drift.
Reusable CI/CD and evaluation	Shorter feedback and lower duplicated effort	Makes verification and evidence repeatable.
Identity and tool brokerage	Safe access to enterprise systems	Enables least privilege and attributable action.
Observability and cost telemetry	Operational control and economic transparency	Supports intervention, accountability, and optimization.
Policy-as-workflow	Fewer manual gates and lower friction	Turns standards into enforceable delivery behavior.

6. Management Changes from Activity Supervision to System Design

Managers have historically coordinated people, priorities, dependencies, and delivery. In a human-agent organization, they must also manage work allocation across agents, review capacity, exception flow, platform constraints, and the quality of organizational context.

Activity metrics become less useful as generation accelerates. Lines of code, ticket counts, prompt volume, and agent utilization can reward noise. Managers need outcome measures: cycle time, change quality, customer impact, review burden, escaped defects, recovery performance, cost, and evidence completeness.

The best managers will design conditions for effective work. They will maintain clear priorities, protect review capacity, reduce unnecessary work, improve team context, develop judgment, and surface systemic bottlenecks. Their role becomes less about tracking production and more about improving the socio-technical system that produces outcomes.

7. The Organization Needs a New Control Model

As agents move from suggestion to execution, the organization must define an authority model. Not every task requires the same control. Drafting a test, changing a database schema, modifying an access policy, and operating a production workflow have different consequences.

Delegation should be explicit, attributable, least-privileged, and reversible where possible. High-consequence changes should require stronger evidence and independent review. Runtime agents require monitoring, intervention, and the ability to reduce or revoke authority when conditions change.

This control model must be embedded in repositories, platforms, identities, and workflows rather than maintained only in policy documents. EB-001 argued that AI changes software governance by moving governance into the engineering control system. EB-002 extends that point: the future organization is viable only when enablement and control are designed together.

Operating Principle

Autonomy should expand only as organizational evidence, platform control, and accountable ownership become strong enough for the consequence.

8. Workforce Strategy Must Preserve the Learning Pipeline

AI can compress tasks traditionally assigned to early-career engineers. If organizations automate those tasks without redesigning development pathways, they risk weakening the pipeline that produces future senior engineers. People learn architecture, debugging, review, and operational judgment through progressively responsible work—not through titles alone.

The World Economic Forum reports that nearly 40 percent of job skills are expected to change and that employers see the skills gap as a leading barrier to transformation [6]. Technical skills in AI, data, and cybersecurity are rising quickly, but analytical thinking, resilience, leadership, and collaboration remain critical [6]. The implication is a blended capability model rather than a purely technical reskilling program.

Organizations should redesign apprenticeship around AI-era work. Early-career engineers can own specifications, tests, evaluations, incident analysis, small-batch changes, review preparation, and operational evidence. Agents can accelerate the work, but managers and senior engineers must still create opportunities for humans to form mental models, encounter failure, and receive meaningful review.

9. An Executive Operating Model

Layer	Executive responsibility	Organizational mechanism
Enterprise	Set strategy, risk appetite, investment, and workforce direction.	Portfolio governance, enterprise architecture, AI policy, funding.
Platform	Provide secure, reusable capabilities for	Internal developer platform, model

	human-agent engineering.	gateway, identity, observability, evidence services.
Product and service teams	Own outcomes, system quality, and operational consequences.	Cross-functional teams, repository-centered workflow, service ownership.
Technical leadership	Maintain architecture, standards, and engineering coherence.	Principal engineer network, review forums, reference patterns, stewardship.
Assurance and risk	Provide independent challenge proportional to consequence.	Risk-tiered review, auditability, exceptions, control validation.
Learning system	Develop people and convert experience into improved capability.	Communities of practice, postmortems, rotations, apprenticeship, skills pathways.

10. A 180-Day Executive Agenda

First, identify where AI is already changing engineering work. Map assistants, agents, automated reviews, generated code, and runtime use by business consequence rather than by vendor. Determine who owns each use and what evidence exists.

Second, define the target operating model. Establish work classes, delegation boundaries, accountable roles, platform responsibilities, and the decisions that remain human.

Third, strengthen the engineering platform. Prioritize identity, repository controls, context management, evaluation, provenance, observability, and intervention before expanding high-authority use cases.

Fourth, redesign measures and incentives. Track flow, quality, reliability, review burden, cost, and learning. Avoid targets that reward artifact volume or superficial agent utilization.

Fifth, protect the talent pipeline. Create AI-era development paths for early-career, senior, managerial, and executive technical roles. Make judgment, review, and operational ownership explicit competencies.

Finally, select a small number of value streams for end-to-end redesign. Do not merely add agents to the existing process. Redesign intent, context, execution, review, release, and feedback as one system, then scale what evidence shows works.

Conclusion

The future software engineering organization will not be defined by how many AI tools it licenses. It will be defined by how well it combines human judgment, agent capability, platform leverage, engineering evidence, and accountable control.

Organizations that treat AI as individual productivity software may achieve local gains while increasing downstream disorder. Organizations that redesign the engineering operating model can increase capacity, shorten learning cycles, improve consistency, and extend the reach of scarce expertise.

The central executive challenge is organizational design. Leaders must decide how work is decomposed, how authority is delegated, how context becomes trustworthy, how evidence is generated, how people develop, and how operational accountability is preserved. The companies that solve that problem will not merely build software faster. They will build a more capable engineering institution.

ETIS Executive Position

The future engineering organization is a governed human-agent system. People remain accountable for intent, architecture, risk, and outcomes. Agents expand execution capacity. Platforms provide context and control. Evidence connects the two.

References and Executive Reading

[1] DORA. State of AI-assisted Software Development 2025 and Platform Engineering capability guidance. https://dora.dev/dora-report-2025/

[2] GitHub. Introducing Agent HQ: Any agent, any way you work, 2025. https://github.blog/news-insights/company-news/welcome-home-agents/

[3] GitHub. The new identity of a developer: What changes and what does not in the AI era, 2025. https://github.blog/news-insights/octoverse/the-new-identity-of-a-developer-what-changes-and-what-doesnt-in-the-ai-era/

[4] GitHub. Octoverse 2025: AI, agents, and the changing development ecosystem. https://github.blog/news-insights/octoverse/octoverse-a-new-developer-joins-github-every-second-as-ai-leads-typescript-to-1/

[5] Microsoft. Transform software development through an agentic platform, 2026. https://developer.microsoft.com/blog/learn-from-microsoft-transform-software-development-through-an-agentic-platform

[6] World Economic Forum. Future of Jobs Report 2025. https://www.weforum.org/publications/the-future-of-jobs-report-2025/

[7] Microsoft Research. New Future of Work: AI is driving rapid change and uneven benefits, 2026. https://www.microsoft.com/en-us/research/blog/new-future-of-work-ai-is-driving-rapid-change-uneven-benefits/

[8] William T. O'Connell. Engineering Digital Colleagues. ETIS White Paper WP-010, 2026. https://etisframework.org

[9] William T. O'Connell. Why AI Changes Software Governance. ETIS Executive Brief EB-001, 2026. https://etisframework.org

About the Author

William T. O'Connell, Ph.D., is the creator of Engineering Trustworthy Intelligent Systems (ETIS), an adjunct professor of computer science at Loyola University Chicago, and a former IBM Distinguished Engineer and CTO for IBM Consulting Quality and Delivery. His career spans four decades of software engineering, enterprise architecture, product and platform development, quality, governance, and large-scale client delivery.

His work focuses on how technology organizations can use artificial intelligence to increase capability without surrendering engineering discipline, operational responsibility, governance, evidence, or human accountability.