ETIS EXECUTIVE BRIEF SERIES

Measuring Engineering in the AI Era

Replacing Activity Metrics with Evidence of Flow, Quality, Trust, and Business Outcomes

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-005 | Version 1.0 | July 2026

Executive Thesis

AI makes software activity easier to generate and easier to mismeasure. The enterprise must stop treating code volume, commits, story points, prompt counts, and agent utilization as evidence of productivity. Engineering performance should be measured as the organization's ability to convert intent into valuable, trustworthy, and sustainable outcomes.

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

Executive Summary

Software engineering has always been difficult to measure. Output is intangible, work is highly interdependent, and the most valuable contributions often prevent problems rather than create visible artifacts. Artificial intelligence intensifies the challenge. Developers and agents can now generate code, tests, documentation, and pull requests at a rate that makes traditional activity measures even less meaningful.

Executives will be tempted by readily available telemetry: lines generated, prompts submitted, agent tasks completed, commits, pull requests, story points, or hours saved. These measures can describe activity. They do not establish business value, system quality, engineering health, or trustworthy delivery. Used as targets, they invite gaming and can accelerate the wrong work.

DORA provides a strong outcome-oriented foundation through software delivery performance measures: change lead time, deployment frequency, failed deployment recovery time, change failure percentage, and deployment rework rate [1]. The SPACE framework adds a multidimensional view of developer productivity across satisfaction and well-being, performance, activity, communication and collaboration, and efficiency and flow [2]. More recent Microsoft research extends this thinking into the AI era, showing that AI's effects differ across those dimensions and should not be reduced to one productivity score [3].

ETIS extends these foundations for AI-assisted and agentic engineering. Leaders must measure not only delivery speed, but also evidence completeness, review capacity, context quality, architecture health, AI verification cost, delegated-work success, authority violations, operational readiness, recovery, and business outcomes.

The central discipline is measurement architecture: a coherent set of outcome, flow, quality, trust, economic, and human-capability measures tied to decisions. The purpose is not to create a larger dashboard. It is to help executives determine where to invest, what to stop, which authority to expand, where risk is accumulating, and whether AI is improving the engineering system or merely increasing activity.

The Executive Test

When a metric improves, can leadership explain which customer, business, engineering, or risk decision should change because of it? If not, the metric is probably reporting activity rather than performance.

1. AI Breaks the Remaining Link Between Activity and Value

Activity has never been equivalent to productivity. A large code change may introduce complexity; a small architecture decision may prevent months of rework. A developer who deletes unnecessary code may create more value than one who adds a feature. A reviewer who blocks a dangerous release may produce no visible output while protecting the enterprise.

AI weakens the connection further because artifact production becomes inexpensive. An agent can create thousands of lines, open multiple pull requests, generate extensive tests, and produce polished documentation without proving that the work addresses the right problem or improves the system.

Measurement systems that reward visible production will therefore encourage excess. Teams will create larger changes, more generated artifacts, and more agent activity. The downstream organization will pay through review load, integration difficulty, security exposure, operational fragility, and verification cost.

Measurement Warning

What AI makes cheap should not become the primary measure of value.

2. What Should Not Be Used as Executive Productivity Measures

Measure	Why it is attractive	Why it fails as a performance measure
Lines of code or generated code	Simple, automated, apparently objective	Rewards volume and complexity; ignores deletion, reuse, quality, and value.
Commit or pull-request count	Visible in repositories and easy to compare	Can reward fragmentation, noise, and trivial change; varies by workflow.
Story points and velocity	Familiar Agile planning data	Team-specific estimates; unsafe for cross-team comparison or executive productivity.
Prompt, token, or agent-task volume	Signals AI adoption and tool use	Measures consumption, not useful outcome, quality, or verification burden.
AI acceptance rate	Appears to show tool effectiveness	May reward passive adoption and hide inappropriate or low-quality output.
Hours saved	Easy business case narrative	Usually estimated, difficult to validate, and may reappear as review or rework.
Individual rankings	Creates apparent accountability	Damages collaboration, encourages gaming, and ignores system constraints.

These measures may remain useful diagnostically. A sudden change in pull-request size, prompt cost, or generated-code volume can reveal a workflow shift. The error is treating them as outcomes or using them to rank people and teams.

Goodhart's Law applies directly: when a measure becomes a target, it ceases to be a good measure. In engineering, the damage is amplified because teams can optimize the visible proxy while weakening maintainability, collaboration, safety, and long-term delivery capability.

3. Start with Business and User Outcomes

Engineering exists to improve an organizational or user outcome. Measurement should therefore begin outside the engineering system. The relevant outcomes may include revenue, conversion, customer retention, task completion, service reliability, cycle-time reduction, fraud prevention, safety, cost avoidance, or regulatory performance.

Not every engineering change maps cleanly to immediate revenue. Platform, security, architecture, reliability, and modernization investments often create option value and reduce future cost. Their measures should still connect to a business hypothesis: faster product delivery, fewer incidents, lower cloud cost, shorter onboarding, reduced vulnerability exposure, or improved recovery.

DORA's user-centric guidance is especially important in the AI era: accelerated generation can move a team faster in the wrong direction if user value is unclear [4]. Business outcomes provide the compass; engineering measures explain whether the delivery system can produce them sustainably.

4. Preserve the DORA Foundation

DORA's software delivery performance measures remain the strongest executive baseline for the delivery system [1]. They balance throughput and instability and should be measured at the level of an application or service, not used to rank individual developers.

The five current measures are change lead time, deployment frequency, failed deployment recovery time, change failure percentage, and deployment rework rate [1]. Together they reveal how quickly change moves, how often value is released, how frequently change creates instability, and how efficiently the organization recovers.

AI does not replace these measures. It makes them more necessary. If coding accelerates while change failure, deployment rework, or recovery deteriorates, the organization is producing faster without delivering better.

DORA measure	Executive question	AI-era interpretation
Change lead time	How quickly can an approved change reach users?	Did AI shorten the complete path or only code production?
Deployment frequency	How often can value be released safely?	Can the system absorb more change in small batches?
Change failure percentage	How often does deployed change degrade outcomes?	Is accelerated generation increasing instability?
Failed deployment recovery time	How quickly can service be restored?	Are AI-generated changes understandable and recoverable?
Deployment rework rate	How much deployment effort must be repeated?	Is apparent velocity being consumed by downstream correction?

5. Use a Multidimensional Productivity Model

Developer productivity cannot be represented by a single number. The SPACE framework explicitly rejects that premise and recommends measuring across satisfaction and well-being, performance, activity, communication and collaboration, and efficiency and flow [2].

This matters because AI can improve one dimension while degrading another. A developer may complete implementation faster but spend more time reviewing generated work. Teams may produce more pull requests while collaboration and shared understanding weaken. Individual satisfaction may rise while operational instability grows.

Microsoft's 2025 SPACE-of-AI research reports that developer experiences with AI differ across SPACE dimensions and that real-world impact is more nuanced than simple speed claims [3]. Executives should expect a portfolio of measures and deliberate tradeoffs rather than a universal AI-productivity percentage.

Productivity Principle

A measure is credible only when it reflects the system of work: outcomes, flow, quality, collaboration, and human sustainability—not one person's visible activity.

6. Add AI-Era Engineering Measures

AI introduces new costs, dependencies, and forms of authority. Organizations need additional measures that reveal whether AI-assisted and agentic work is improving the engineering system.

These measures should remain decision-oriented. They are not a standard scorecard to implement everywhere. The appropriate set depends on the use case, risk, and operating model.

Measure family	Illustrative measures	Decision supported

AI verification cost	Review time, rework, rejected output, escaped defects, independent validation effort	Is AI reducing total effort or moving work downstream?
Delegated-work effectiveness	Task success, human intervention, rollback, escalation quality, repeat failure	Which agent authority can expand, remain bounded, or be withdrawn?
Context quality	Freshness, authoritative-source coverage, retrieval precision, missing-context failures	Where should context engineering and knowledge investment focus?
Evidence completeness	Traceable requirement, review, test, provenance, release, and operational evidence	Is the change defensible enough to advance?
Review capacity	Queue age, PR size, reviewer load, time to disposition, finding closure	Is production outrunning human and automated assurance?
Authority integrity	Denied actions, policy violations, privilege exceptions, unowned agents	Is delegated execution operating within approved boundaries?
AI economics	Cost per accepted outcome, model/tool cost, failed-run cost, human oversight cost	Which use cases and configurations create sustainable value?

7. Measure Evidence, Not Document Volume

AI can produce extensive documentation quickly, so artifact count is an increasingly weak indicator of discipline. The stronger question is whether engineering claims are supported by connected, current, credible evidence.

Evidence completeness can be measured through traceability: whether a material change links to authorized intent, architecture or risk decisions, implementation, review, verification, release, and operational ownership. Quality still requires judgment; a complete chain can contain weak evidence. But missing links reveal uncertainty that should be addressed before authority is exercised.

This approach converts governance from process-counting to decision support. Leaders can see which releases, systems, or agents lack sufficient evidence for their risk tier, without rewarding teams for creating documents that do not change decisions.

8. Architecture and Repository Health Need Leading Indicators

Delivery metrics are lagging indicators of an engineering system. Architecture and repository health provide earlier signals of future friction. Useful indicators include dependency growth, ownership gaps, test reliability, change concentration, interface instability, architectural exceptions, outdated documentation, unresolved vulnerabilities, and recurring review findings.

These measures should not become universal thresholds. A monolith and a distributed platform have different structures. The value lies in trend and context: is complexity increasing faster than capability, are ownership and testability keeping pace, and is AI-generated change amplifying weak areas?

Repository health also affects agent performance. Missing instructions, stale architecture, weak tests, and ambiguous ownership create poor context for humans and agents. Context debt becomes measurable when it produces repeated clarification, rework, policy violations, or retrieval failure.

9. Operational Trust Is the Reality Test

Pre-release measures cannot establish operational success. Reliability, recoverability, security, user impact, and agent behavior must be measured in production.

Service-level indicators and objectives connect technical behavior to user reliance. Incident frequency, detection, containment, recovery, repeated causes, and corrective-action closure show whether the organization can live with the systems it releases.

NIST's AI RMF Measure function calls for defined performance limits, uncertainty, transparency metrics, human-AI competency assessment, and ongoing measurement of trustworthiness [5][6]. For AI systems, operational measures may include unsupported-output rate, policy adherence, human escalation, authority violations, harmful outcome signals, model or context drift, and time to intervention.

Operational Principle

A dashboard is not proof. Operational evidence is useful when it changes release, authority, intervention, investment, or retirement decisions.

10. Measure the Human System

Engineering performance depends on people's ability to understand, collaborate, review, learn, and sustain attention. An organization can improve short-term output while damaging the human capability needed for long-term performance.

SPACE includes satisfaction and well-being because developer experience influences retention, quality, and effectiveness [2]. Microsoft's EngThrive model similarly combines speed, ease, and quality with thriving as a guardrail [7].

Executives should monitor cognitive load, interruption, review burden, onboarding, access to feedback, psychological safety, and opportunities to develop expertise. AI adoption that removes all formative work from early-career engineers may improve immediate throughput while weakening the future leadership pipeline.

Survey data should complement telemetry. System data can show delay; people can explain why. Neither should be used as covert individual surveillance.

11. Build a Measurement Architecture, Not a Metric Warehouse

A mature measurement system has layers. Outcome measures show whether the organization creates value. Delivery measures show flow and stability. Engineering-health measures show whether architecture, repositories, and platforms remain sustainable. Trust measures show whether claims, authority, and operations remain defensible. Human measures show whether the organization can continue performing.

Each measure should have an owner, definition, source, segmentation, interpretation, and decision use. Measures should be reviewed as a coherent narrative, not as independent traffic lights. A faster lead time accompanied by rising rework and reviewer exhaustion is not unqualified improvement.

The measurement platform should preserve drill-down without inviting individual ranking. Executives need portfolio visibility; teams need actionable local detail. The same data should support learning rather than punishment.

Layer	North-star question	Examples
Business and user outcomes	Did the change create meaningful value or reduce material risk?	Adoption, task success, revenue, customer impact, loss avoided.
Flow and delivery	Can the organization move change safely and quickly?	DORA measures, queue time, work in progress, batch size.
Quality and engineering health	Is the system becoming easier or harder to	Defects, test reliability, architecture

	change and operate?	exceptions, dependency and repository health.
Trust and governance	Can consequential claims and delegated actions be defended?	Evidence completeness, authority violations, exception age, intervention success.
Human capability	Can people understand, review, collaborate, learn, and thrive?	SPACE/EngThrive measures, review load, onboarding, skill growth, retention.
Economics	Does the full engineering system produce value at sustainable cost?	Cost per outcome, rework, AI verification cost, platform unit economics.

12. A 120-Day Executive Agenda

First, stop using activity measures as executive productivity targets. Retain them only as diagnostic signals with explicit caveats.

Second, establish an outcome hierarchy. Connect enterprise objectives to product outcomes, service outcomes, and engineering-system measures.

Third, implement the DORA delivery baseline at the service or application level. Use trends and context; do not rank individuals.

Fourth, add a small set of AI-era measures for the highest-impact workflows: verification cost, delegated-work success, review capacity, context quality, authority integrity, and unit economics.

Fifth, combine telemetry with structured team feedback using SPACE or a comparable multidimensional framework.

Sixth, create metric governance. Define ownership, calculation, interpretation, privacy boundaries, and the decisions each measure supports.

Finally, review the portfolio quarterly as a system. Ask where speed improved, where instability or verification cost moved, where authority can expand, where trust weakened, and which investments remove the most important constraint.

Conclusion

AI makes engineering measurement more important and more dangerous. It increases the amount of visible activity while weakening the assumption that activity represents value, understanding, or progress.

The right response is not a new universal productivity score. It is a balanced measurement architecture grounded in business outcomes, DORA delivery performance, multidimensional developer productivity, engineering health, trust evidence, operational reality, and economics.

Executives should measure the performance of the engineering system, not the apparent busyness of its participants. The organization succeeds when it can convert intent into valuable change quickly, safely, transparently, and repeatedly—and when AI strengthens that capability rather than obscuring its weaknesses.

ETIS Executive Position

Measure what the organization can responsibly deliver, defend, operate, and improve. Do not confuse accelerated artifact production with engineering performance.

References and Executive Reading

[1] DORA. DORA's software delivery performance metrics, updated 2026. https://dora.dev/guides/dora-metrics/

[2] Forsgren, N., Storey, M.-A., Maddila, C., Zimmermann, T., Houck, B., and Butler, J. The SPACE of Developer Productivity, 2021. https://www.microsoft.com/en-us/research/publication/the-space-of-developer-productivity-theres-more-to-it-than-you-think/

[3] Microsoft Research. The SPACE of AI: Real-World Lessons on AI's Impact on Developers, 2025. https://www.microsoft.com/en-us/research/publication/the-space-of-ai-real-world-lessons-on-ais-impact-on-developers/

[4] DORA. User-centric focus capability guidance, updated 2026. https://dora.dev/capabilities/user-centric-focus/

[5] NIST AI Resource Center. AI RMF Playbook: Measure. https://airc.nist.gov/airmf-resources/playbook/measure/

[6] NIST AI Resource Center. AI RMF Core: Measure. https://airc.nist.gov/airmf-resources/airmf/5-sec-core/

[7] Microsoft Research. EngThrive: Make It Fast and Easy to Do Great Work, 2026. https://www.microsoft.com/en-us/research/publication/engthrive-make-it-fast-and-easy-to-do-great-work/

[8] DORA. State of AI-assisted Software Development 2025. https://dora.dev/dora-report-2025/

[9] William T. O'Connell. Engineering Evidence. ETIS White Paper WP-003, 2026. https://etisframework.org

[10] William T. O'Connell. The Future Software Engineering Organization. ETIS Executive Brief EB-002, 2026. https://etisframework.org

[11] William T. O'Connell. Building an AI Engineering Platform. ETIS Executive Brief EB-004, 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.