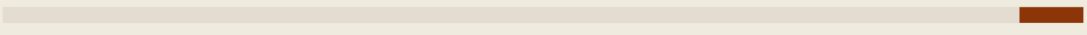
[anthara]

The *AI-Fluent* PDLC.

Almost every engineering organization has adopted AI, while very few have become fluent with it. The gap between those two things is the subject of this paper.



[ABSTRACT]

Read this paper if:

- Nearly everyone on your team has an AI subscription
- Cycle times have not reduced and overall productivity gains are elusive
- You'd like a framework on how to move your team to AI fluency

Most AI spend is concentrated on coding, which is just a fraction of your delivery cycle (6% for the study in this report). Weeks go by in the other 94% of getting your ticket to production. Fluency is when the whole cycle moves faster, without losing trust in what's delivered.

AI fluency is when teams can deliver fast AND trust what they ship. This paper is our analysis of what it takes for teams to achieve this.

It is grounded in DORA's research, experience from our team of AI Forward Deployed Engineers, and our 20+ years of experience building software for highly regulated industries.

Contents

This paper is Incubyte's position on how the entire delivery cycle can be compressed, without losing trust in what ships.

[a] **WHO THIS IS FOR**

CTOs and VPs of Engineering at regulated and health-tech software organizations. A field reading built on public research and our work with regulated teams, not a controlled benchmark. Lifecycle figures are from analysis of one enterprise health-tech client's delivery pipeline.

01

Adoption is not fluency

AI spend concentrates on a small sliver of the delivery cycle associated with writing code. Productivity gains show up only when the entire cycle is compressed.

Usage up, delivery unchanged

Coding is not the constraint

Fluency is speed + trust

DORA and DX are two of the most reputed research teams tracking AI adoption and its effects on software delivery.

[ADOPTION · DORA 2025]

DORA's 2025 survey of nearly 5,000 technology professionals found that **90% now use AI at work**. Most tellingly, higher AI adoption was associated with both higher software delivery throughput and higher delivery instability.

[DELIVERY · DX 2026]

According to DX's February 2026 report, AI usage rose 65% across 400+ companies, while the quantum of work delivered rose just **7.8%**. Because coding is only about 14% of a developer's day, accelerating code generation relocates the constraint downstream to review, verification, and integration.

DORA'S CONCLUSION

AI is fundamentally an amplifier: it magnifies the strengths of high-performing organizations and the dysfunctions of struggling ones.

90%

technology professionals now use AI day to day · DORA 2025

7.8%

throughput increase despite 65% usage rise · DX 2026

14%

of a developer's time is spent on coding · DX 2026

At one of our own health-tech clients, the entire delivery cycle was **11.7 weeks**, of which coding was just **6%**. Tickets spent almost 11 weeks in queues, clarification, review, or rework — the AI budget lands almost entirely on the coding sliver, so the 11.7-week cycle barely moves.

FIG. 01 · AI SPEND CONCENTRATED ON A SMALL SLIVER

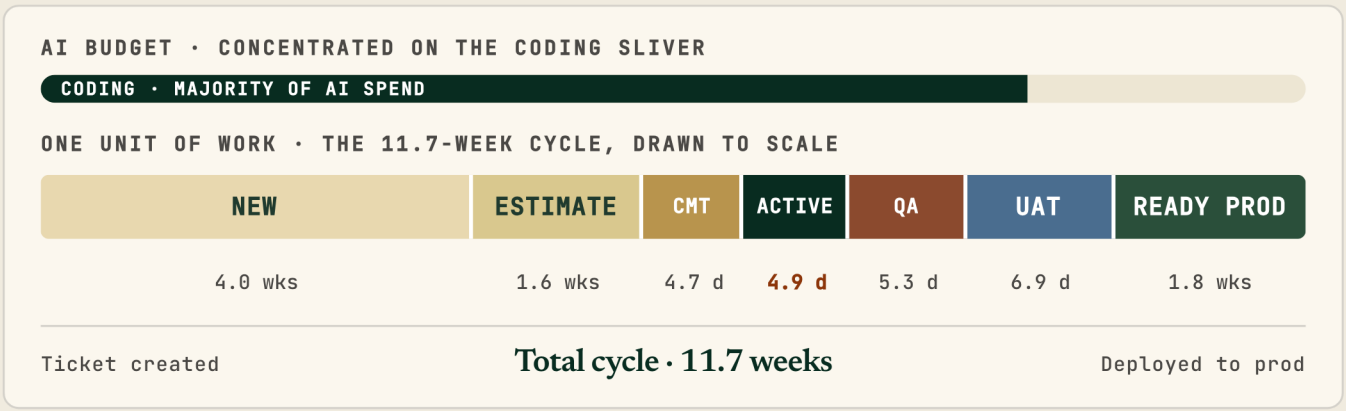


Figure 1. AI has been unable to compress the entire cycle, with faster coding creating bottlenecks upstream and downstream.

[FLUENCY, DEFINED]

AI fluency is when the entire cycle is compressed, without losing trust in what ships.

Adoption is easy to track: usage, licenses, AI-assisted commits. Fluency is measured by how much of the delivery cycle AI compresses, while stability, maintainability, technical debt, and overall system quality *improve* rather than decline.

The next chapter introduces the core organizational framework that turns adoption into fluency.

ONE CAVEAT BEFORE WE GO FURTHER

Since AI amplifies the current state, the system has to be in a state worth amplifying. **Continuous delivery, automated tests, small batches, and loosely coupled architecture** are table stakes that let teams compound their AI gains. Without them, teams are adding code faster than the system can absorb. Everything that follows assumes these foundations are in place.

[OUR TAKE]

> *The whole coding stage is 6% of the cycle. Speeding up coding while leaving the other 94% untouched barely moves the calendar. Fluency is when AI reaches into review, QA, and deployment and the whole thing gets faster, and you can still stand behind what shipped.*

02

The fluency stack

Fluency is built on the cultural foundation of an organization, as it applies to the capabilities of its people and the conduct of its agents.

DORA's chain, extended

Conduct, added for the AI era

DORA's 2025 State of DevOps research published a core organizational model for engineering excellence: capabilities drive software delivery performance, which drives organizational outcomes.

Alongside it, DORA released an AI Capabilities Model with seven capabilities that amplify AI outcomes: a clear and communicated AI stance, healthy data ecosystems, AI-accessible internal data, strong version control practices, working in small batches, user-centric focus, and quality internal platforms.

Our adaptation of the core model abstracts these capabilities and introduces **Conduct** as an additional layer, which together drive engineering performance.

CAPABILITY

What your people can do with AI across the whole lifecycle, well beyond the coding stage. Judgment, breadth, business grounding, documented context.

CONDUCT

What your systems hold your people and your agents to while they work. Standards, regulations, and data boundaries, enforced as the code is generated.

WHAT AN ORGANIZATION IS RESPONSIBLE FOR

Build the capability and conduct layers. What follows are outcomes, not levers.

The next two chapters drill into these layers, breaking down what it takes to build them into your organization's playbook.

[THE SHAPE OF THE STACK]

Culture is the soil. Capability and conduct are the two layers you invest in directly. Engineering performance and business outcomes are the evidence they leave behind, not levers you pull.

The AI-Fluent stack

[FIG. 02]

Layers 1 (Capability), 3 (Engineering Performance), and 4 (Business Outcomes): from DORA's research chain. Layer 2 (Conduct): added for the AI era.

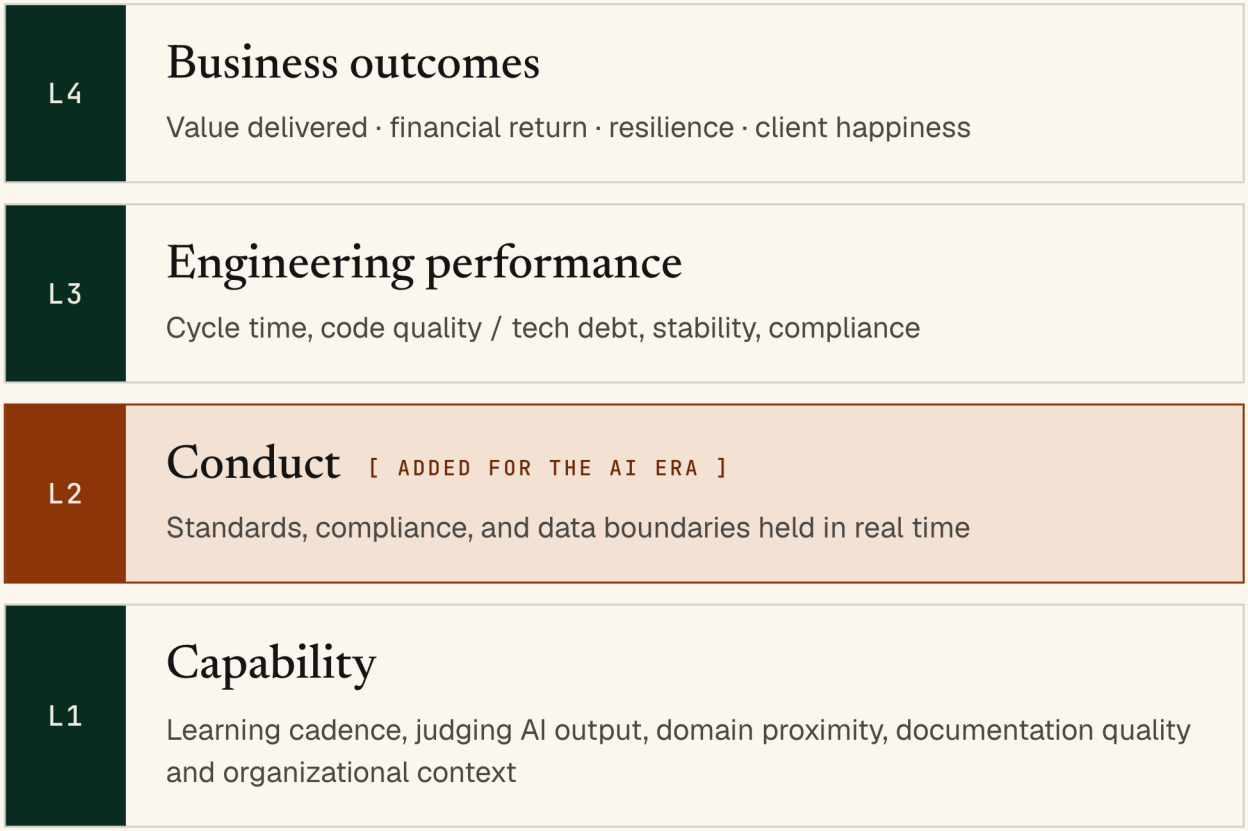


Figure 2. Layers 1 (Capability), 3 (Engineering Performance), and 4 (Business Outcomes) come from DORA's research chain. Layer 2 (Conduct) is added for the AI era. Read it bottom to top: you build the capability and conduct layers, and the outcomes above follow.

The AI-Fluent stack: DORA's chain, with conduct added for the AI era

Each layer, what it is, and the example signals to watch for.

LAYER	WHAT IT IS	EXAMPLE SIGNALS
Culture foundation	The soil everything grows in.	Psychological safety, generative (Westrum) culture, room to fail.
Capabilities	What the org provides and what people know.	Learning cadence, judging AI output, domain proximity, documentation quality and organizational context.
Conduct added for the AI era	Rules enforced while AI generates.	Standards, compliance, and data boundaries held in real time.
Engineering performance	The evidence fluency leaves behind.	Cycle time, code quality / tech debt, stability, compliance.
Business outcomes	Why any of it matters.	Value delivered, financial return, resilience, client happiness.

> [AN ORGANIZATION'S RESPONSIBILITY]
>
> *Build the capability and conduct layers. Engineering performance and business outcomes follow; they're not levers you pull directly.*

03

The capability layer

Capability is what your people can actually do with AI across the whole lifecycle. It is the first layer an organization is responsible for building.

Learning and continuous improvement

Skill and judgement

Domain and Organizational Fluency

Knowledge and documentation systems

This is the part of fluency you can invest in directly. Each area is something you can measure today, that moves a delivery metric you care about months later, and that an Anthara deployment builds. And even these rest on one thing that has to come first: culture.

[FOUNDATION]

Culture decides whether any capability takes root

Culture is the soil the other four grow in. When AI usage is mandated by upper management, teams won't admit the AI is wrong or share what they learned the hard way, if it isn't safe to do so. DORA has measured this for a decade as generative (Westrum) culture, and it remains the strongest predictor of whether any other capability converts into outcomes.

The signal the whole stack depends on is whether engineers can hold each other accountable for what their AI produces. You cannot push sloppy code that raises the team's review burden, and you cannot let your craft slide because AI wrote the code. Culture is being able to call that out. **Measure psychological safety and learning-culture scores.**

03 · THE CAPABILITY LAYER

“

[OUR TAKE]

If it isn't safe to call out sloppy AI-generated code, your review process is theater, and your stability numbers will tell on you.

[DIMENSION 01]

Learning and continuous capability

This is about how much your team collectively knows and whether you keep growing it. People who can name the patterns and architectures they work in can direct and check more of what the AI produces; narrow specialists can only vouch for their own corner.

Learning cadence, generalist ratio, and slack-time usage predict lead time and change-failure rate: teams that keep learning catch AI's mistakes before they ship, and generalists keep work moving instead of queuing behind the one specialist who knows.

Tools alone won't do it. Leaders have to supply the conditions:

[01 · SLACK TIME]

Protected hours for hackathons, learning sessions, and trainings, counted as real capacity on the plan.

[02 · LEARNING BUDGET]

Continuous learning carried as an organizational responsibility, with a real budget and cadence behind it.

[03 · AI-CENTRIC HIRING]

Interviews that test whether a candidate can read and critique a large AI-generated plan, which now matters as much as writing code from scratch.

[04 · 3:1 WITH PAIRING]

Senior-to-junior ratios of 3:1 with active pairing. Teams that protect pairing time on AI-generated code keep their junior pipeline.

“

[OUR TAKE]

Apprenticeship matters more in the AI era. AI is doing the work juniors used to learn from, so you have to build new learning loops on purpose, or you'll have no senior engineers in five years.

[DIMENSION 02]

Skill and judgement

When the capabilities around them are in place, individual engineers develop a recognizable set of skills, and these are what a modern hiring process should test for now.

01 Reading and judging AI output

Evaluating large AI-generated plans, diffs, and code for correctness and fit. This is fast becoming the core engineering skill, and what the hiring process should test for.

02 Spec and intent authoring

Telling AI what to build clearly enough to get the right result: specifications, constraints, and acceptance criteria. This is spec-driven development, the leap past prompt-in, code-out.

03 Architecture and systems judgment

Recognizing good structure when you see it and steering AI toward maintainable design instead of plausible mess.

04 Generalist breadth

A wide surface area across the stack and the SDLC, so one person can direct AI across more of the work and connect the parts.

05 Craft mindset

Caring enough to keep the bar high for AI-generated code, so the quality of the codebase doesn't degrade over time.

Together these are what let a team take AI past the coding stage and into requirements, review, and QA without losing the thread, which is where the 94% actually compresses.

[DIMENSION 03]

Domain and Organizational Fluency

As AI absorbs the purely technical work, the edge that's left is the knowledge AI can't pull from the codebase or a prompt: how the business actually makes money, how this organization actually works, and the rules of the domain it operates in. AI can write the code. It can't tell you which feature matters to the business this quarter, which internal system will quietly break if you touch it, or where the regulatory and compliance landmines sit.

That judgment pays off earliest in the cycle, in scoping and estimation, long before code is written, and it's the part least substitutable by a model. "Almost right" is a judgment about your business and domain, the kind a model can't make for you. Decisions grounded in how the business and the system actually work predict rework rate and value delivered.

66%

cite "almost right, but not quite" as their top AI frustration · Stack Overflow 2025

75%

would still ask a person when they don't trust an AI answer · Stack Overflow 2025

~95%

of enterprise GenAI pilots show no measurable return · MIT NANDA 2025

88% of organizations now use AI in at least one business function, yet by one widely cited analysis roughly 95% of enterprise generative-AI pilots show no measurable financial return. What's missing is rarely the model; it's the distance between the team and the problem worth solving. Teams with that grounding build the right thing the first time, which is the cheapest cycle time there is.

> “
>
> [OUR TAKE]
>
> *AI commoditizes the tech; the edge is what lives outside the codebase, in your business, your systems, and your domain.*

[DIMENSION 04]

Knowledge and documentation systems

Capability lives in more than people's heads. It lives in the systems that capture and retrieve what the organization knows: internal documentation and decision records that both humans and coding agents can pull from. It is also what bridges the handoff and onboarding gaps that sit inside the 94%. DORA identified internal documentation quality as a **force multiplier**: the same AI investment produces materially better outcomes where documentation is strong.

Documentation quality and retrievability predict lower lead time, time-to-restore (MTTR), and onboarding speed, and they cut rework. **In the field:** across the codebases we've reviewed, the teams whose agents behaved best shared one trait, current and written-down context. Tooling budget didn't predict it.

THE PREREQUISITE UNDERNEATH

An **agent-ready codebase.** Coding agents go only as deep as your codebase lets them. Documentation, decoupling, and test coverage are a precondition for fluency, and sit where culture does, underneath the capability layer as a foundation.

CAPABILITY, CODIFIED

Craft's durable form is written down and executable: practices, patterns, and domain rules become reusable **skills, commands, hooks, and shared memory** every agent runs by default. Encode it once; every session inherits it.

“

[OUR TAKE]

What's good for your engineers is good for your agents. Clear documentation and clean code help both; the messy kind drags both down. AI just makes you find out faster which one you've built.

The Conduct Layer

The enforcement half of fluency: standards, regulations, and data boundaries held while AI generates code, so manual reviews are not the bottlenecks.

Coding standards Compliance posture Data and security boundaries

Conduct warrants a layer of its own in the AI era because human reviews can no longer keep pace with what AI generates. Building trust in what AI generates is the way forward. The cost of a leak missing human reviews could be catastrophic for a highly regulated domain, and conduct is what reduces the risk. The capability layer prepared your people and systems for AI; the conduct layer ensures your standards, regulations, guardrails, security rules, and data boundaries are all enforced *while* AI generates code.

[DIMENSION 01]

Coding standards

Standards are your team's agreed-upon way of coding, a definition of what good looks like and how to reliably produce it. An agent will produce something that works or passes tests. Standards steer the agent to produce code your team can continue to own, read, and change months later.

Standards prevent the slow quality erosion teams are experiencing with higher AI use. Turn them into live context so they load with every session, not into review comments that show up too late.

[01 · SESSION-LOADED RULES]

Standards that load into every AI session automatically and render into each tool's native format, so they are present as the code is written.

[02 · ORG + REPO LAYERS]

Clarity around what applies across the company versus to this codebase, so the agent applies the right rules contextually.

[03 · CRAFT WITH AI]

Spec-driven development and working in small batches let you shift left on reviews and keep each unit small enough to reduce cognitive load.

[OUR TAKE]

> *If your standard only lives in review comments, you are paying your best engineers to be a linter.*

[DIMENSION 02]

Compliance posture

For compliance-heavy domains like healthcare or fintech, engineers are trained to ensure their code and conduct is compliant with HIPAA, PCI-DSS, and SOC 2. The cost of a leak goes up the later it is found.

HHS OCR brought 16 HIPAA enforcement actions in 2024 (about $9.4M in penalties), and in late 2024 proposed the first major update to the HIPAA Security Rule in over 20 years (published January 2025). The proposed rule tightens encryption, risk-analysis, and resilience requirements for every system that handles ePHI. This brings AI-enabled teams squarely in scope of audits.

16

HIPAA enforcement actions in 2024, about $9.4M in penalties · HHS OCR

20+ yrs

since the last major Security Rule update; the NPRM brings AI tools into scope · HHS OCR 2025

[01 · REGULATIONS AS GUARDRAILS]

The frameworks that apply to you, written into rule sets the agent works to as it generates.

[02 · AUDIT TRAILS & LOGGING]

Logging every prompt, action, and policy decision as the work happens ensures the evidence is always available.

[03 · VERSION CONTROL PRACTICE]

DORA's strong version control: every change small, attributable, and logged with its reasoning becomes an additional compliance record.

[OUR TAKE]

> *Compliance checked only at review is you relying on the judgement of one human. It takes just one incident to put your business at serious risk.*

[DIMENSION 03]

Data and security boundaries

An agent does not just read; it acts. It calls models, tools, and outside services, and each call can carry sensitive data out of your boundary or take an action you did not intend it to take.

Intentionally govern what is allowed to cross the line. The rules have to be enforced at the point of the call, not in a post-hoc review.

01

In-flight detection

Define what happens when sensitive data is detected before it leaves the network.

02

Tool and action governance

Deliberately define what every tool and MCP call is allowed to do, with destructive operations blocked across the organization.

03

Own the data boundary

Ensure the whole path runs in your VPC or on-prem, so code and data stay inside.

[OUR TAKE]

> *One prompt with real patient data in a public model is enough to lose your AI mandate. Boundaries are what let you go faster with higher trust in autonomy.*

05

Engineering performance

Higher engineering performance is evidence that your capability and conduct layers have been built right.

Throughput and cycle time

Stability

Code quality and technical debt

Compliance and audit-readiness

The four DORA metrics, deployment frequency, lead time for changes, change failure rate, and time to restore, track the system's health and measure a team's ability to ship fast with safety. When capability and conduct are built right, all four move together.

Throughput and cycle time HIGHER

The whole cycle compresses, not just the coding sliver.

Stability HELD

Change failure rate and time to restore hold as throughput climbs.

Code quality & tech debt SUSTAINED

The codebase stays ownable months later, with sustained developer and agent experience.

Compliance & audit-readiness INTACT

Compliance posture stays intact with evidence always available.

An AI-assisted PDLC raises the stakes: much more code being generated makes it even more critical to get early signals from these metrics as adoption grows. These are also the right signals for how the team is trending on its journey from adoption to fluency.

When the Capability and Conduct dimensions are done right, all four DORA metrics move together.

[FIG. 03 · DORA METRICS MAPPED TO CAPABILITY AND CONDUCT DRIVERS]

KEY *Capability dimension* Conduct dimension

Deployment frequency	Small changes ship without waiting on a manual gate. *Skill and judgement* · Coding standards · Data boundaries	HIGHER
Lead time for changes	The 94% compresses: scoping, review, integration. *Business fluency* · *Knowledge and docs* · Coding standards	LOWER
Change failure rate	Standards and compliance enforced as code is written. Coding standards · Compliance posture · Data boundaries	HELD OR LOWER
Time to restore service	Changes stay small, attributable, reversible. *Knowledge and docs* · Compliance posture · Data boundaries	HELD OR LOWER

Foundation: a generative (Westrum) culture is the soil every driver above grows in.

Figure 3. Each DORA metric maps back to specific capability and conduct dimensions.

The next chapter defines our model for fluency and how these performance parameters translate to your team's AI fluency.

SECTION SIX · 06 / 06

06

Reading your fluency

Fluency has two axes: how fast you move across the cycle, and how much of your trust AI has earned.

Fluency is fast plus trusted

Flow, level by level

Earned trust

A team can feel fluent and yet not be. They may be shipping fast, with healthy dashboards and high tool usage. Then a change that sailed through review takes down production, and it turns out nobody really knew what the AI had written. That gap, between feeling fluent and being fluent, is what this chapter helps you see.

Fluency is fast plus trusted

[FIG. 04 · THE FLUENCY QUADRANT]

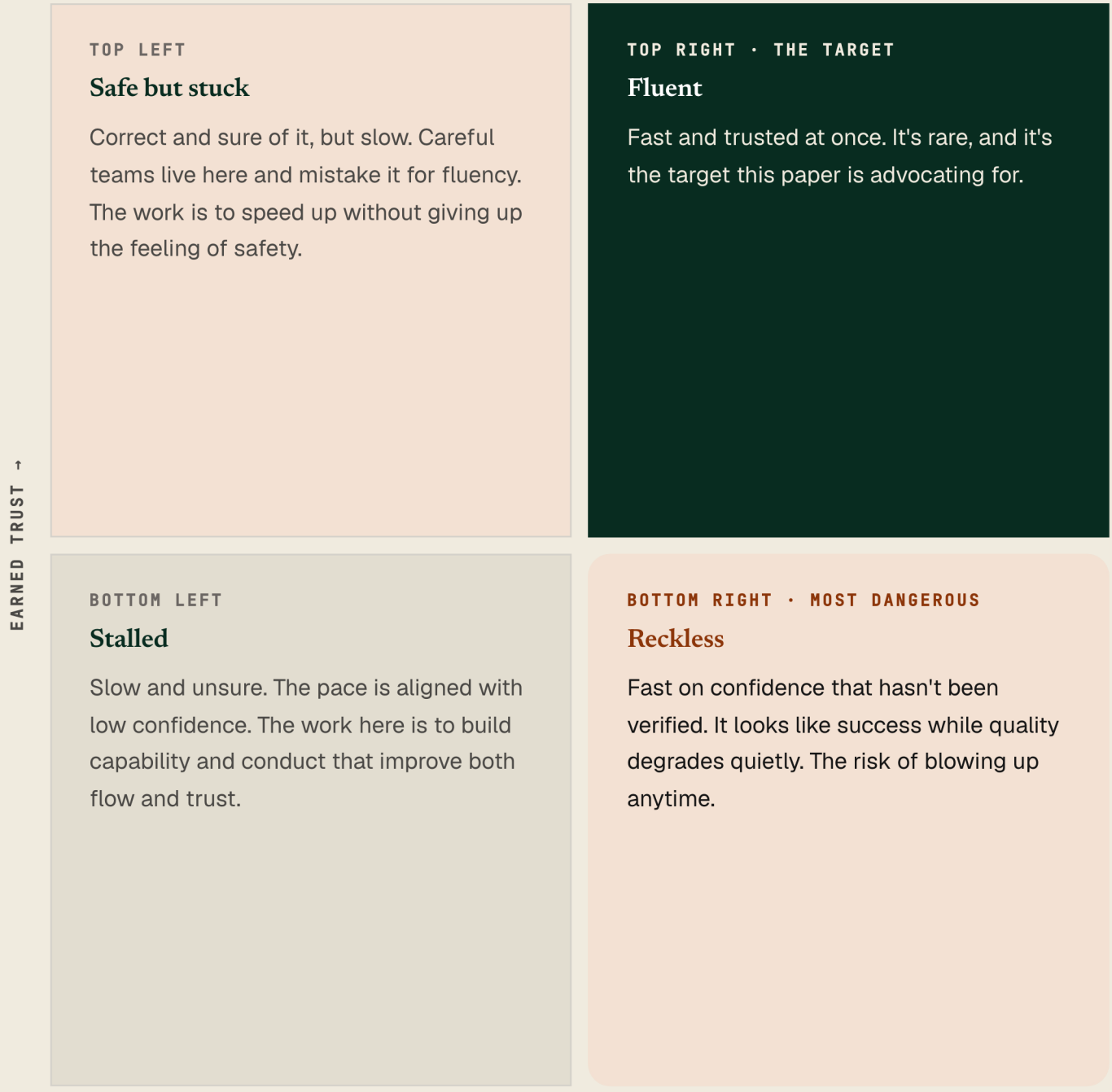


Figure 4. Fluency has two axes: how much of the cycle you've compressed (flow), and how much of your trust AI has earned. The target is the top-right corner.

[AXIS 01 · FLOW]

The whole cycle moving without friction

Work is in flow when a unit of work moves through the whole delivery cycle without stalling: no waiting in a queue, no bouncing back for clarification, no sitting in a review backlog for three days. When work flows, cycle time is low. The biggest impact comes when AI is able to carry a single unit of work on its own, without stalling.

LEVEL	HOW FAR AI CARRIES THE WORK	WHAT IT LOOKS LIKE
1	**Prompting**	Code generated from prompts in the editor. Table stakes now.
2	**Plan mode**	The developer has AI plan the change first, then work against that plan.
3	**Spec-driven**	High-quality specs drive larger batches of work at once.
4	**Agentic from the ticket** THE JUMP THAT MATTERS	Agents pull the spec straight from Jira or Linear, clarify acceptance criteria, and open reviewed PRs.
5	**Autonomous within guardrails**	End-to-end workflows triggered and run on their own, within pre-set guardrails.

THE JUMP FROM 3 TO 4

This is the leap that most compresses the delivery cycle. It's where AI stops living inside the coding stage and starts carrying work across the intake, review, and QA that make up the rest.

[AXIS 02 · EARNED TRUST]

Confidence that matches reality

Confidence on its own tells you nothing, because a team can feel completely sure and be completely wrong. What matters is whether that confidence is calibrated: whether what the team believes about its AI's output matches what the code actually does.

[FIG. 05 · THE EARNED-TRUST SPECTRUM]

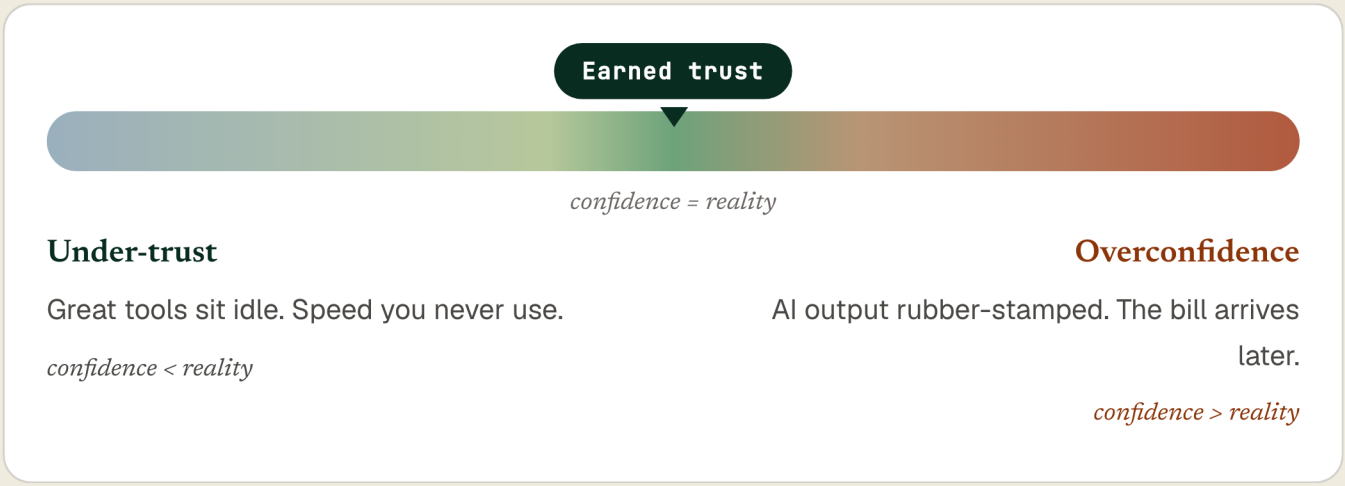


Figure 5. Earned trust is confidence you can defend: verification catches bad outputs, outputs track your AI stance, rework drops, and stability doesn't slip as you speed up.

CHAPTER CLOSING

You reach the top-right by building capability and enforcing conduct as AI works across the cycle. They're the only levers that move flow and trust together — faster delivery you don't have to second-guess.

1. **Fluency is fast + trusted** — the top-right of the quadrant.
2. **Flow is how far AI carries a unit of work across the cycle** — the jump from level 3 to level 4 is the one that matters.
3. **Earned trust is confidence calibrated to reality** — verification catches bad outputs, rework drops, stability holds.

THE OFFER · TWO WAYS TO INSTALL IT

Two ways to install it.

OPTION A

Buy it from Anthara

The conduct layer as a product, enforced inside your own security boundary.

ANTHARA.AI

OPTION B

Build it with Incubyte

The same capability and conduct, built in-house as a service with your team.

INCUBYTE.CO

Methodology and sources

This brief combines public research with field observations from regulated-software engineering teams. The framework follows DORA's capabilities, performance, outcomes model, extended with a conduct layer for the AI era. Field observations are illustrative of patterns we see in engagements and are labeled as such, not presented as a controlled study.

Delivery lifecycle timings (coding about 6% of the cycle inside an approx. 11.7-week cycle) are from analysis of one enterprise health-tech client's delivery pipeline.

References

1 DORA, 2025 State of AI-Assisted Software Development Report (Google Cloud). dora.dev/research/ai/

2 DORA, Accelerate State of DevOps Report 2024 (throughput, stability, cluster distribution, documentation findings). dora.dev/research/2024

3 GitHub, Octoverse 2025 (Copilot adoption among new developers and open-source maintainers). octoverse.github.com

4 Stack Overflow, 2025 Developer Survey (adoption, trust, frustration figures). survey.stackoverflow.co/2025

5 McKinsey, The State of AI 2025 (88% adoption in at least one function); MIT NANDA, State of AI in Business 2025 (~95% of GenAI pilots show no measurable return). mlq.ai/media/quarterly_decks/v0.1_State_of_AI_in_Business_2025_Report.pdf

6 DX, AI and Engineering Velocity: A Longitudinal Analysis, 2026 (65% usage rise, 7.76% PR-throughput rise across 400+ companies). getdx.com/report/ai-and-engineering-velocity-a-longitudinal-analysis/

7 CodeRabbit, State of AI vs Human Code Generation, December 2025 (10.83 vs 6.45 issues per PR across 470 PRs). coderabbit.ai/blog/state-of-ai-code-generation-2025

8 Sonar, false-positive analysis, 2026 (3.2% across 137 million issues). sonarsource.com/blog/how-sonarqube-minimizes-false-positives/

9 METR, Measuring the Impact of Early-2025 AI on Experienced Open-Source Developer Productivity, July 2025. metr.org/blog/2025-07-10-early-2025-ai-experienced-os-dev-study/

10 Peng et al., The Impact of AI on Developer Productivity: Evidence from GitHub Copilot, 2023; Cui, Demirer, Jaffe, Musolff, Peng, and Salz, field experiments on generative AI and developer output.

11 Robbes et al., "Agentic Much? Adoption of Coding Agents on GitHub," arXiv 2026; SWE-PRBench, arXiv 2026. arxiv.org/abs/2601.18341

12 HHS OCR, HIPAA Security Rule NPRM fact sheet (December 2024 / January 2025); HIPAA Journal enforcement tracker. hhs.gov/hipaa/for-professionals/security/guidance/index.html

Ready to move from adoption to fluency? Get in touch with the Anthara team.

[a]

The rigor of healthtech, *at AI's pace.*

Same discipline, every install. We embed engineers trained in the craft that keeps AI-accelerated engineering safe, compliant, and built to last.