State of AI Impact in Engineering: Q3 Report

Published October 2026 | Reporting period: July - September 2026

REPORT

Table of contents

Executive summary i

Introduction ii

Part 1—DX Core 4 Framework 2

Speed 3

PR Throughput 3

Time to first review 4

PR size 5

Effectiveness 6

Distribution of DXI over time 6

Perceived ease of delivery 7

Percent change in DXI factors 8

Quality 10

Change failure rate 10

Failed deployment recovery time 11

Impact 12

Innovation ratio 12

AI work type 14

AI cost 15

AI spend 15

Spend per developer 16

Token spend type 17

Part 2—CAFE(S) Framework 18

Mapping DX Core 4 to CAFE(S) framework 20

Closing thoughts 21

Methodology 22

DX Research Team 24

State of AI Impact in Engineering: Q3 Report

DX Research Team

Executive summary

With AI-adoption and AI-authored code at all-time highs, many engineering organizations are seeing faster code generation not translating into a proportional number of new features and capabilities.

Our Q3 data complicates the obvious story. Organizations generating high volumes of AI code wait 105% longer for a first review than those at the bottom, yet weekly PR throughput keeps climbing fast, up 75.5% over four quarters. Authoring capacity has scaled, and delivery volume has scaled with it. What hasn't kept pace is everything that makes that volume safe to trust.

This report publishes the latest data on the DX Core 4 from 500+ companies and identifies where the delivery pipeline may be breaking down:

1. **Speed gains remain strong, with the first sign of easing.** Median weekly TrueThroughput grew from 1.43 to 2.51 PRs/dev (+75.5%) over the past four quarters. Quarterly growth has held in a narrow, high band, +21.7% (Q4 2025 to Q1 2026), +21.3% (Q1 to Q2 2026), +19.0% (Q2 to Q3 2026), with Q3 marking the first quarter where growth ticked down rather than held steady or climbed. While it is too early to identify whether this is the start of a trend, a quarter of slightly less growth after three straight quarters near 21% is noticeable.
2. **Individual task relief far outpaces systemic gains.** Developers are reporting higher satisfaction with isolated, diagnostic tasks like production debugging (+13.3%), documentation (+10.4%), and build and test (+10.4%). Systemic factors like code maintainability (+0.6%) and clear project direction (+2.0%) moved only marginally — we don't have prior-quarter driver-level data to say whether this is a rebound from an earlier decline or a new high. Perceived ease of delivery, a separate metric, improved by just 3% over the same period — a fraction of the relief developers report on individual tasks. AI is helping developers move through systemic issues faster, but it is not yet closing that gap.
3. **Quality is remaining flat.** Change failure rate (CFR) and failed deployment recovery time stayed roughly flat. Steady failure rates while shipping substantially more is an achievement, suggesting that failures are being absorbed by automated pipelines. The one exception is the median change in CFR from Q2 to Q3 2026 shifted from 0% to +0.5%, a small, not-yet-significant move in the wrong direction, and the metric we'd watch most closely next quarter.
4. **Innovation has stayed stable.** Average innovation, measured by the percentage of time spent on new capabilities and features as opposed to maintenance, has increased by 1%, a statistically significant but small improvement that still does not match up to speed gains.

Because foundation model capabilities are held relatively constant across teams, variance in code quality and review overhead is driven in large part by input context quality. To measure context upstream, this report also uses our newly released CAFE(S) context framework to improve output.

Introduction

Over the past two years, AI has moved from a pilot initiative to a standard component of the engineering toolchain. As reported in our Q2 State of AI Impact in Engineering Report, AI-authored code now accounts for more than half of all new code. Developers frequently report that these tools help them complete individual tasks faster and reclaim hours of their work week.

However, this increase in authoring capacity has not resulted in a proportional increase in software delivery. Our research shows that while the friction of writing code has decreased, the friction of moving that code through the delivery pipeline has intensified. The bottleneck in engineering continues to migrate downstream.

Теперь это перемещение бутылочного горлышка видно и на статистике

The cost of machine-authored code

The current challenge for engineering leaders is an imbalance between production and trust. AI has made it inexpensive to generate a changeset but it has not reduced the cognitive effort required to review, test, and verify one. When AI agents produce code without sufficient architectural context or business intent, they often create larger, more complex pull requests and changesets that lack architectural alignment that are difficult for human peers to absorb. This is an early expression of what Margaret-Anne Storey terms cognitive and intent debt, debt that is invisible while humans hold missing context in their heads, and immediately costly once that context must be made explicit for an AI agent.

This imbalance is visible in the data. As AI adoption scales, we see a widening gap between developer sentiment and system performance. Developers feel more effective in isolation, while the overall ease of delivery has moved only slightly. Time saved during the initial drafting phase is being lost to saturated review queues and extended verification cycles.

PART 1

DX Core 4

Speed | Effectiveness | Quality | Impact

To understand why AI gains are stalling, we look beyond raw output volume to evaluate the state of engineering through the DX Core 4, and identify where the delivery pipeline is breaking down.

Я уже в одном из выпусков Research Insights разбирал этот фреймворк

DX Core 4

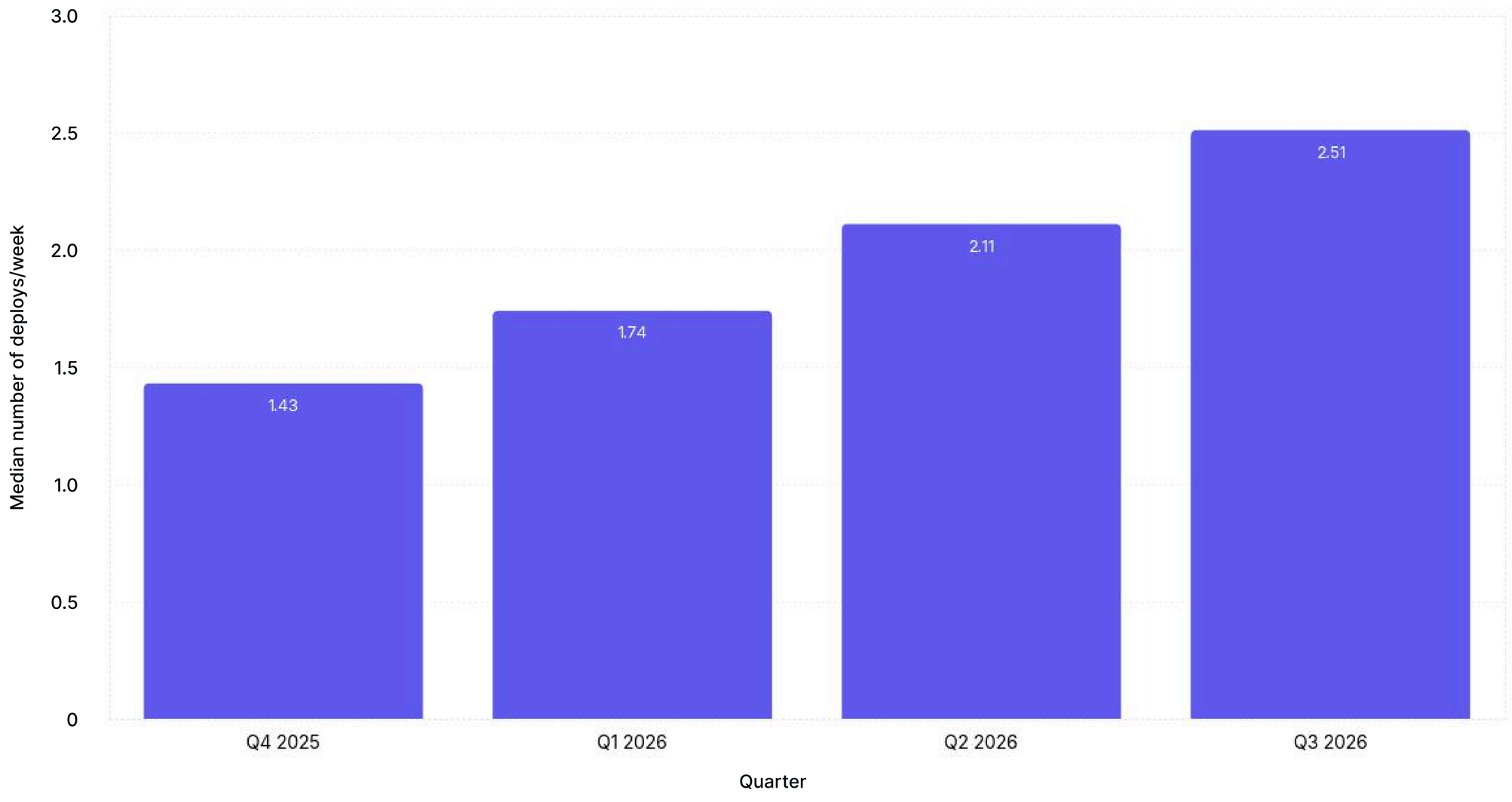
	Speed	Effectiveness	Quality	Impact
Key metric	Diffs per engineer* (PRs or MRs) *Not at individual level	Developer Experience Index (DXI) Measure of key engineering performance drivers, developed by DX	Change failure rate	Percentage of time spent on new capabilities
Secondary metrics	Lead time Deployment frequency Perceived rate of delivery	Time to 10th PR Ease of delivery Regrettable attrition* *Only at organizational level	Failed deployment recovery time Perceived software quality Operational health and security metrics	Initiative progress and ROI Revenue per engineer* R&D as percentage of revenue* *Only at organizational level

Speed

PR throughput continues to show sustained, high growth. Median weekly TrueThroughput grew from 1.43 PRs per dev in Q4 2025, to 1.74 in Q1 2026 (+21.7%), to 2.11 in Q2 2026 (+21.3%), to 2.51 in Q3 2026 (+19.0%). Growth has held remarkably steady near 20–22% a quarter for a full year, with Q3's 19.0% growth the first modest dip below that band.

Median weekly TrueThroughput

Sample from 500+ companies from October 2025 to September 2026



Why TrueThroughput

Raw PR throughput is an effective metric for delivery volume, engineering system flow, and pipeline friction, especially when measured in aggregate within the DX Core 4. However, raw throughput is increasingly distorted with greater AI utilization and code generation. With AI-authored code exceeding 50% and PR sizes doubling, raw PR counts are not reflecting fundamental shifts in PR structure, and may even be overestimating outcomes. TrueThroughput bridges this growing gap by applying complexity weighting, offering another approach to measuring delivery and flow.

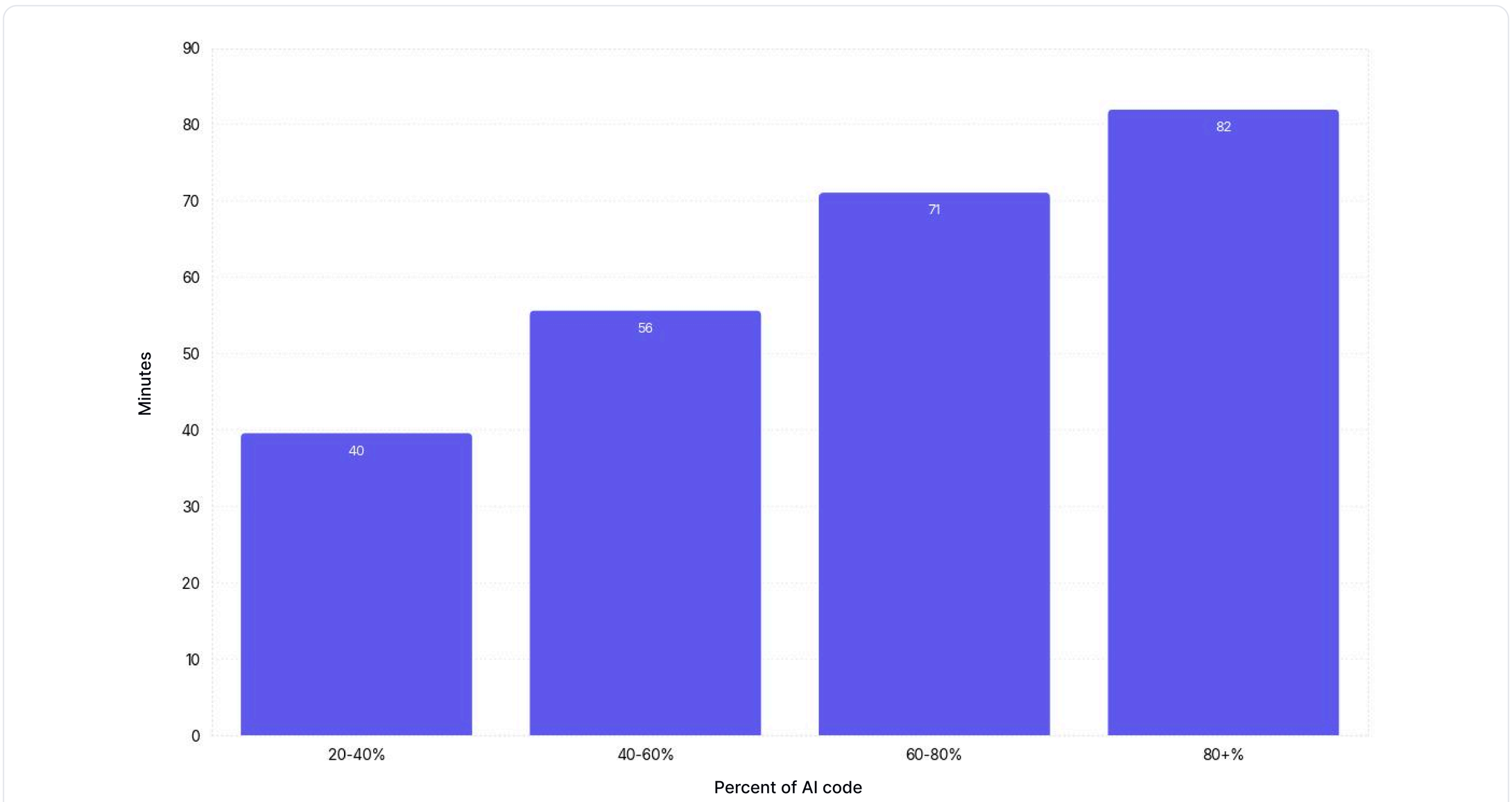
Какой-то свой TrueThroughput :)
Интересно, а есть публичное описание того, как он считается?

To check whether review is keeping pace with this sustained volume growth, we examined the first point in the pipeline where a machine-authored change requires human attention. Analyzing the median time to first review across AI code volume tiers reveals a clear slowdown as AI code volume grows. Organizations at the top of the AI code generation curve wait more than twice as long for a first review as those at the bottom.

AI-authored code share	Median time to first review	Change vs. previous tier	Change vs. lowest tier
20-40%	40 min	-	-
40-60%	56 min	+40%	+40%
60-80%	71 min	+26.8%	+77.5%
80%+	82 min	+15.5%	+105%

Median time to first review by volume of AI code

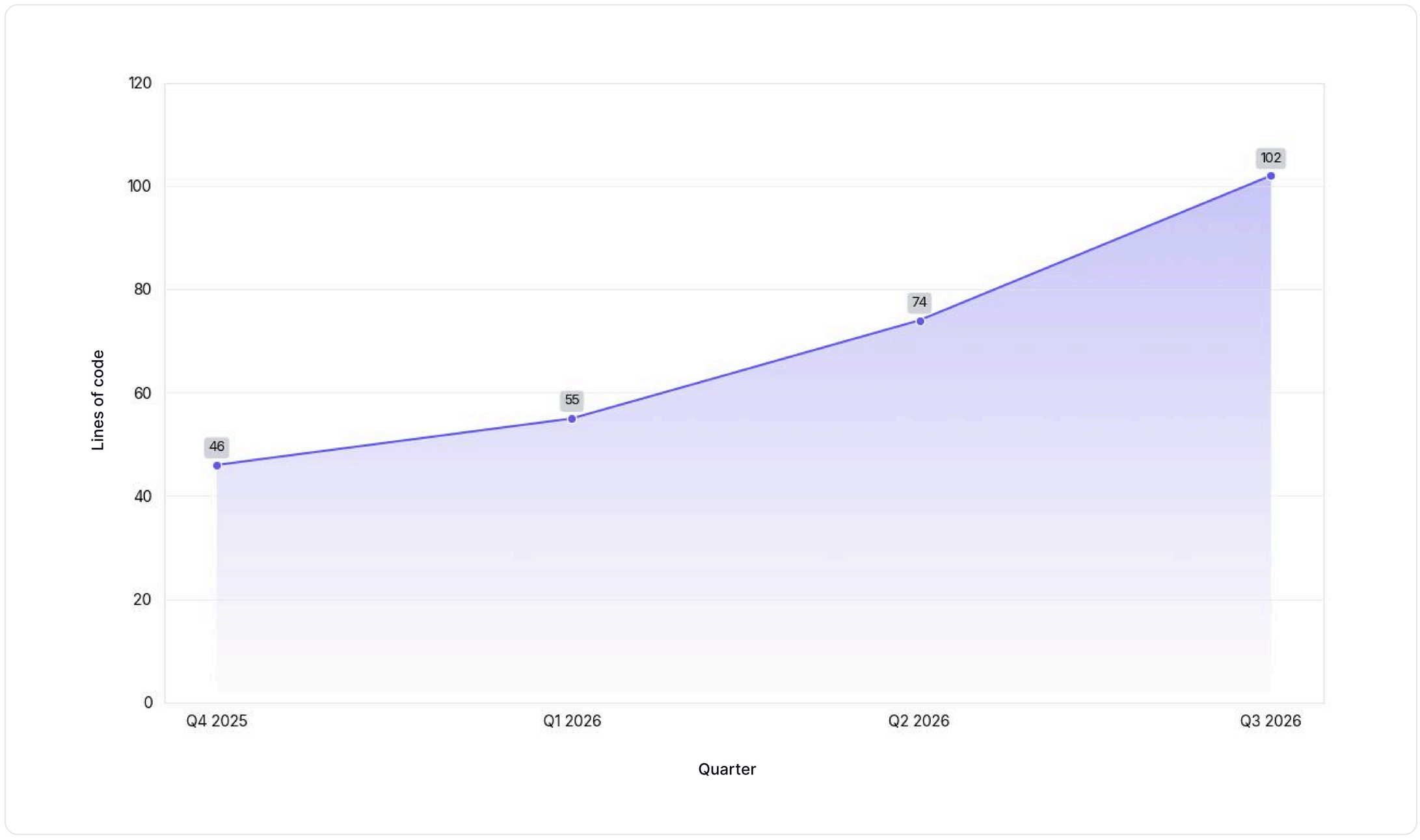
Sample from 500+ companies from July 2026 to September 2026



The data points to review readiness as a growing source of friction that hasn't yet slowed aggregate delivery, but it's worth separating three distinct claims rather than treating "review is the bottleneck" as one finding. First, there's simply more code to review. PR size is up 122% year-over-year (+38% from Q2 2026 alone), and wait time before a first review is 105% longer for the highest-AI-volume organizations.

Median PR size

Sample from 500+ companies from October 2025 to September 2026



This is consistent with a queuing problem—more changesets chasing the same review bandwidth—rather than a quality problem. Second, a harder claim is that AI-generated changesets require more effort per review, not just more reviews. We don't have a direct measure of active review time in this data set, so we can't confirm or rule this out yet, but it is a gap worth flagging rather than an assumption to build the narrative on. Third, some reviewers may be selectively delaying or declining AI-heavy pull requests for reasons distinct from workload. We don't have evidence for this either; it would show up as increased picking/assignment latency independent of PR size, and would need qualitative data (e.g., reviewers describing hesitancy toward AI-authored changes specifically) to distinguish from plain overload.

Taken together, however, the wait-time and PR-size data support the first, capacity-driven explanation. The second and third are open hypotheses, not findings and worth naming explicitly as directions for the next report rather than asserting either way.

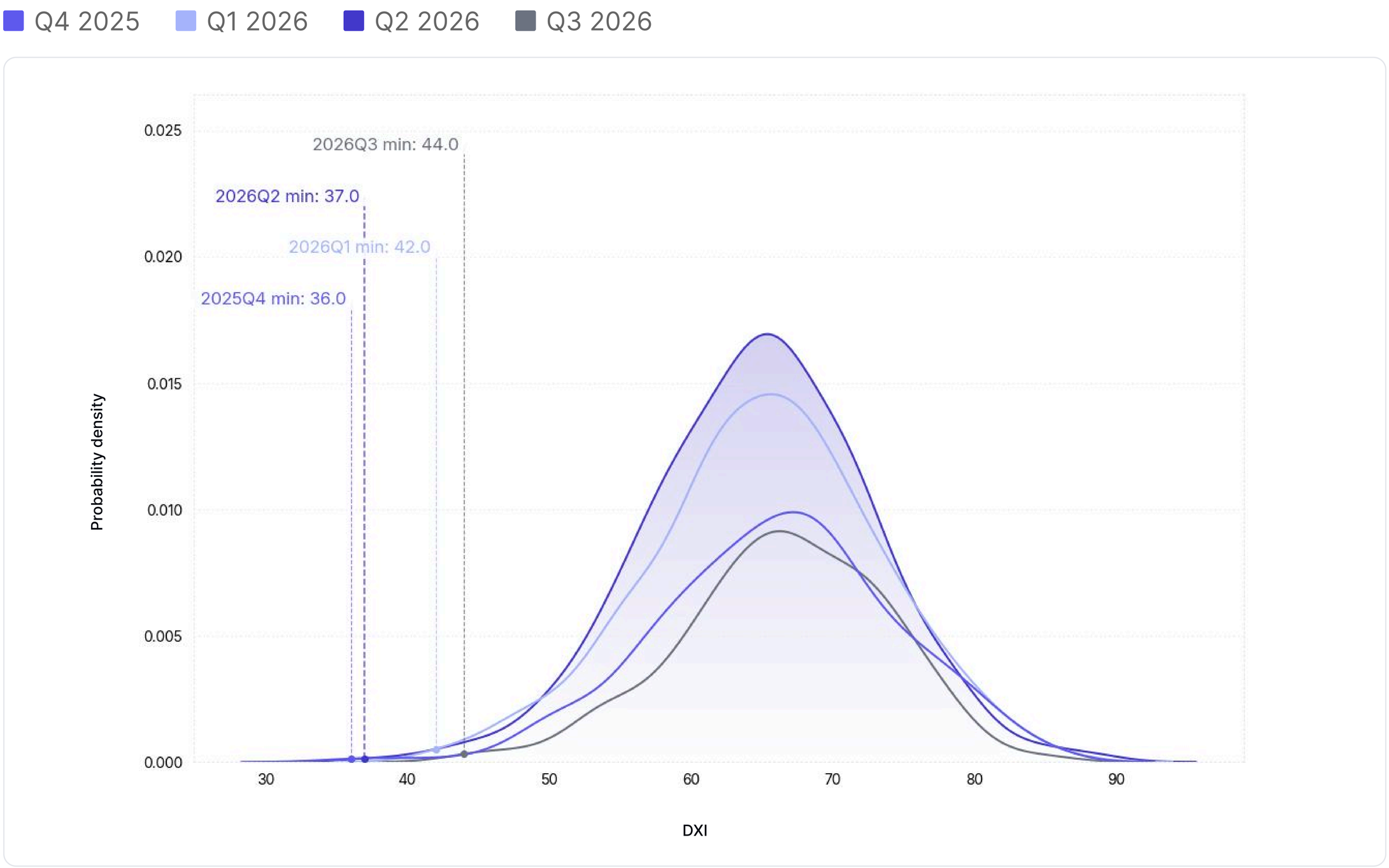
Effectiveness

Evaluating developer experience through sentiment metrics suggests that while AI tools are measurably relieving isolated, high-friction tasks, it is leaving the experience of software delivery largely unchanged. The Developer Experience Index (DXI), which captures developer perception of engineering processes and workflows, has moved only slightly at the median (+1%) though its distribution has notably changed.

Between Q4 2025 and Q3 2026, DXI scores consolidated from a wide spread into a narrower band centered between 65 and 70. The lower tail, representing organizations whose developers reported frustration, has compressed substantially.

Distribution of DXI over time

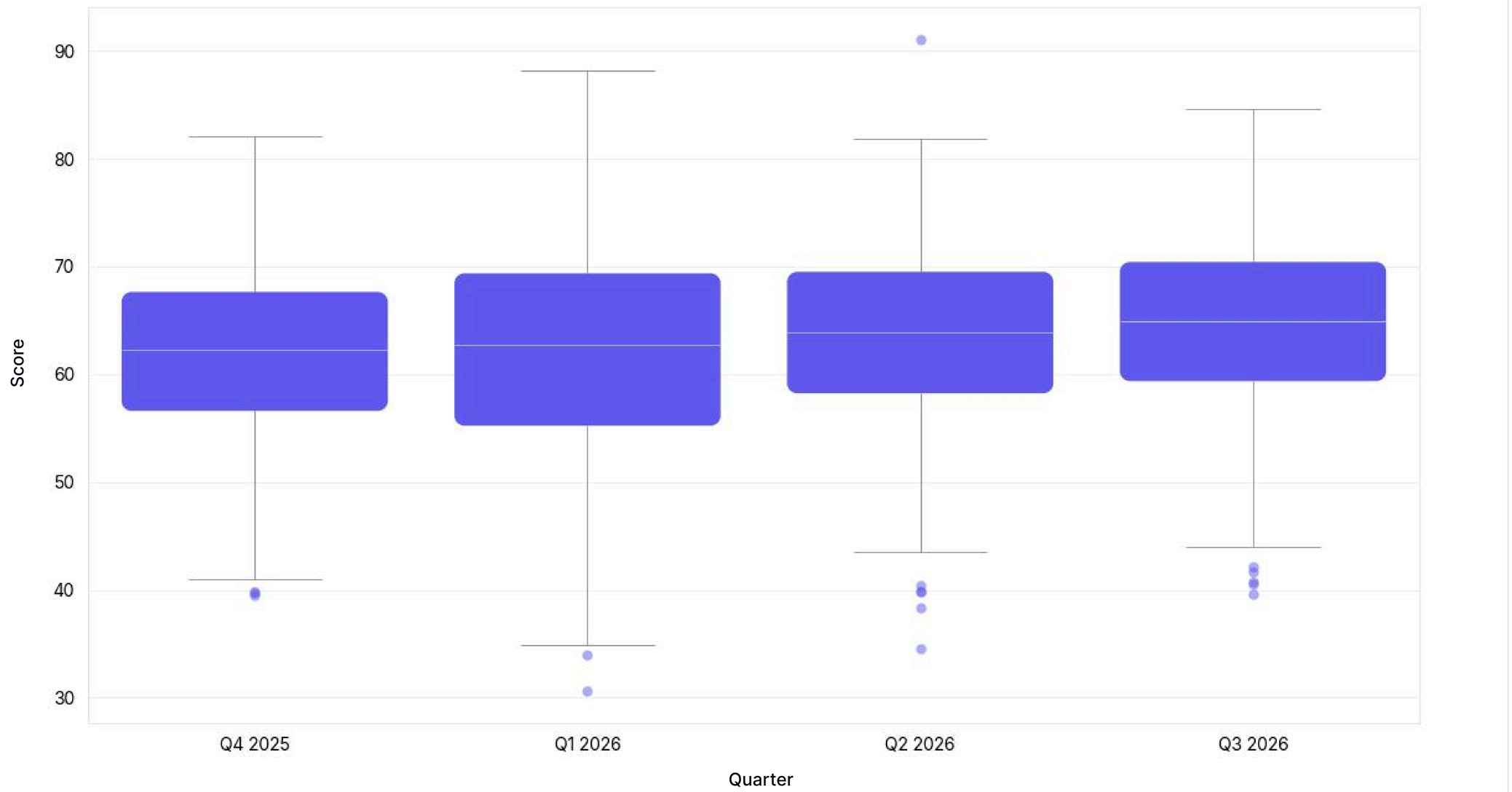
Sample from 500+ companies from October 2025 to September 2026



Raising a floor, however, is not the same as raising a ceiling. Over the same period, the median organization reported only a 3% improvement in perceived ease of delivery from Q2 to Q3 2026[1]. This is a modest one next to the double-digit relief developers report on individual tasks. Developers are generating code faster than ever, and delivery is inching forward too, just not at the same pace.

Perceived ease of delivery score over time

Sample from 500+ companies from October 2025 to September 2026



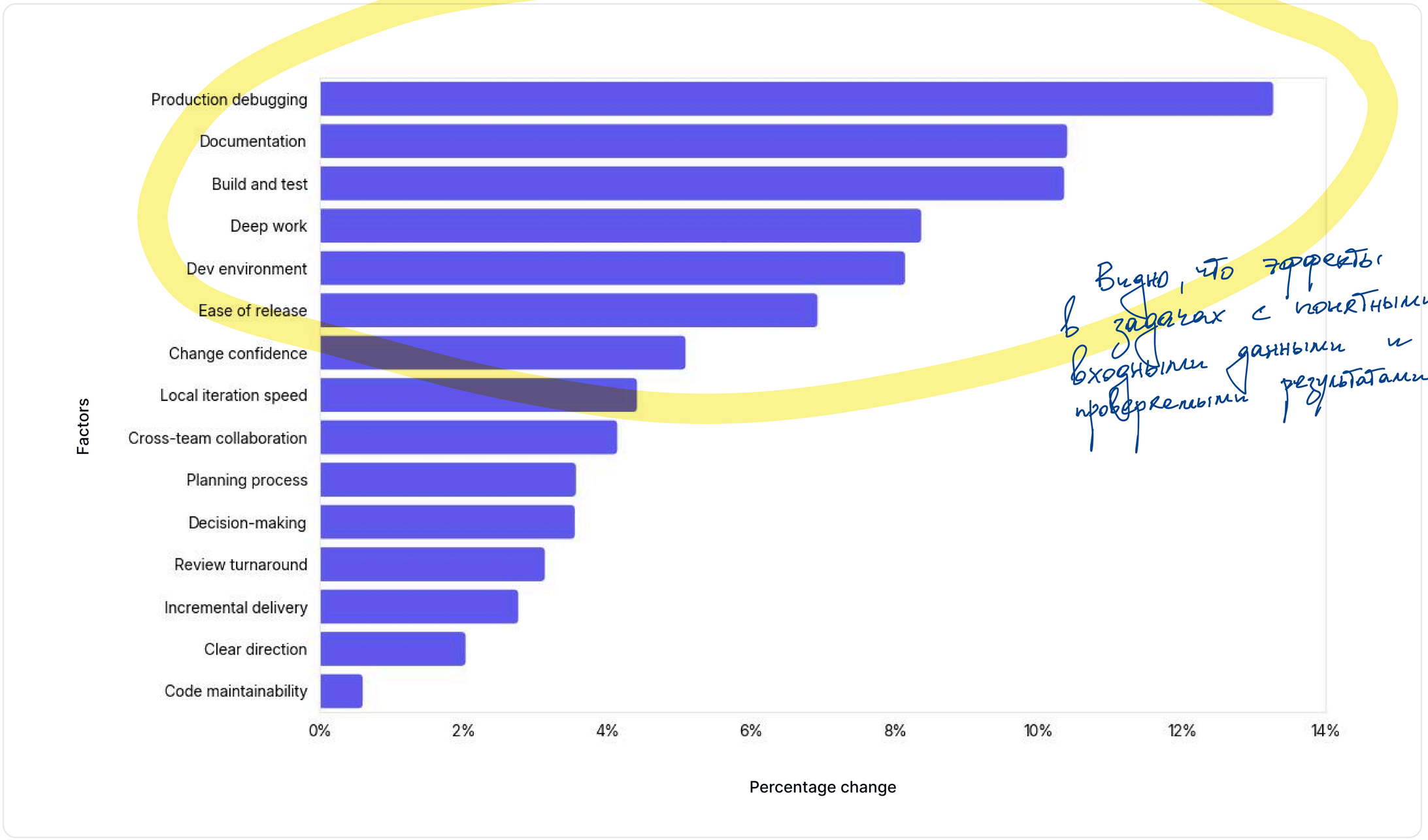
[1]Correction: our Q2 2026 report described "ease of delivery" as both a DXI driver and a secondary Core 4 metric, with the figure shown there reflecting DXI's ease of release driver. Ease of delivery is not a DXI driver. It's a separate survey item, as described above. The 3% figure in this report uses the correctly identified metric and is not a continuation of the Q2 figure.

Breaking DXI into its component factors shows where AI's value is concentrated. The largest gains are all in bounded, diagnostic work: production debugging (+13.3%), documentation (+10.4%), and build and test (+10.4%). These are tasks with a clear input, a verifiable output, and a short feedback loop. The smallest gains are all in structural, cross-cutting concerns: code maintainability (+0.6%) and clear project direction (+2.0%). These are tasks that require holding a whole system in view.

Ranked from largest to smallest gains, the DXI factors sort almost perfectly from local to systemic. AI is helping developers move through existing organizational friction faster. It is not reducing the friction.

Percent change in DXI factors

Sample from 500+ companies from April 2026 to September 2026



Every individual DXI driver improved this quarter, even structural ones like code maintainability and clear project direction, but by smaller margins than diagnostic tasks like debugging. Our previous report, covering Q3 2025 to Q2 2026, showed a mix of drivers rising and falling over that full year. Rather than compare the two periods directly, we'd note that this quarter's uniform improvement is consistent with seasonality as much as it is with an underlying trend, and it is a pattern worth watching as more quarters come in.

These are different layers of the same system. Individual-task experience improved substantially; system-level delivery experience improved only slightly. AI's gains are concentrated at one layer and have barely reached the other.

This gap between individual-task experience and end-to-end delivery experience underscores that AI is making developers more proficient at working around existing friction, without yet reducing that friction by nearly the same amount. Developers are using AI to complete individual tasks quickly, but those gains translate only partially into a smoother path through review queues, test suites, and release gates. AI is making authoring code feel easy and the broader delivery system is improving too, just much more slowly.

Quality

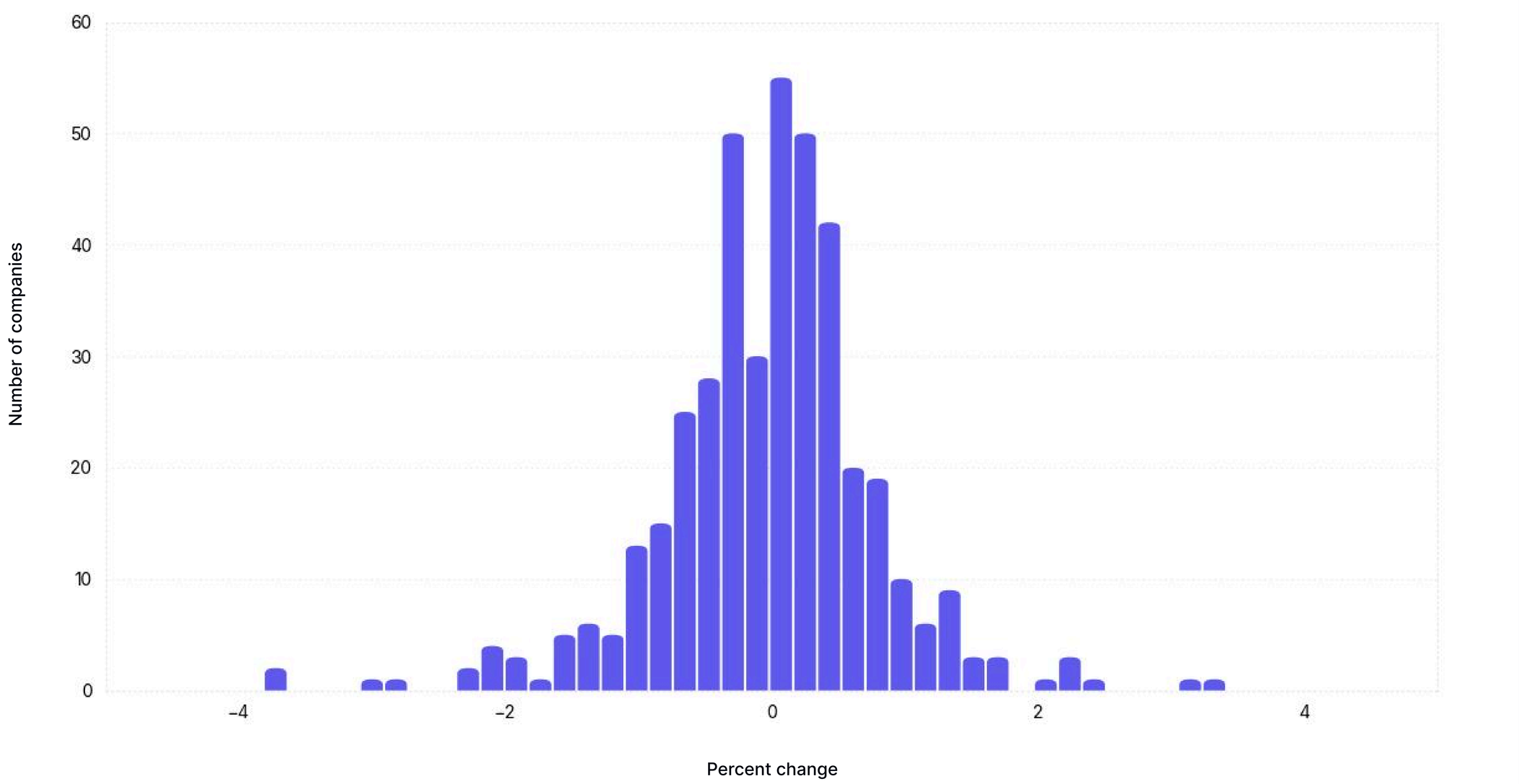
Production stability metrics show little shifts despite a 47% increase in delivery volume over four quarters. Holding failure rates flat while shipping substantially more is not a neutral result, but neither is it the quality collapse that early AI-adoption concerns anticipated. The more accurate read is that automated pipelines are currently absorbing the additional risk, and that we can see the first signs of strain.

Of the quality metrics we track, change failure rate is one of the more diagnostic, reflecting failure frequency directly, where recovery time reflects the combined effect of failure rate and response capacity.

Change failure rate (CFR) offers the one directional signal in this section. In our previous report, the distribution of quarter-over-quarter percentage shifts in CFR sat centered on 0%, indicating no change for the typical organization. The median change in CFR from Q2 to Q3 shifted from 0% to approximately +0.5%. This is a shift in the central tendency of a distribution of quarter-over-quarter relative changes of roughly half a percentage point, against a spread running from -4% to +4%. It is directionally consistent with our expectations, and it is not yet statistically distinguishable from normal quarterly variation.

Distribution of percent change in change failure rate %

Sample from 500+ companies from Q2 (April-June) 2026 → Q3 (July-September) 2026



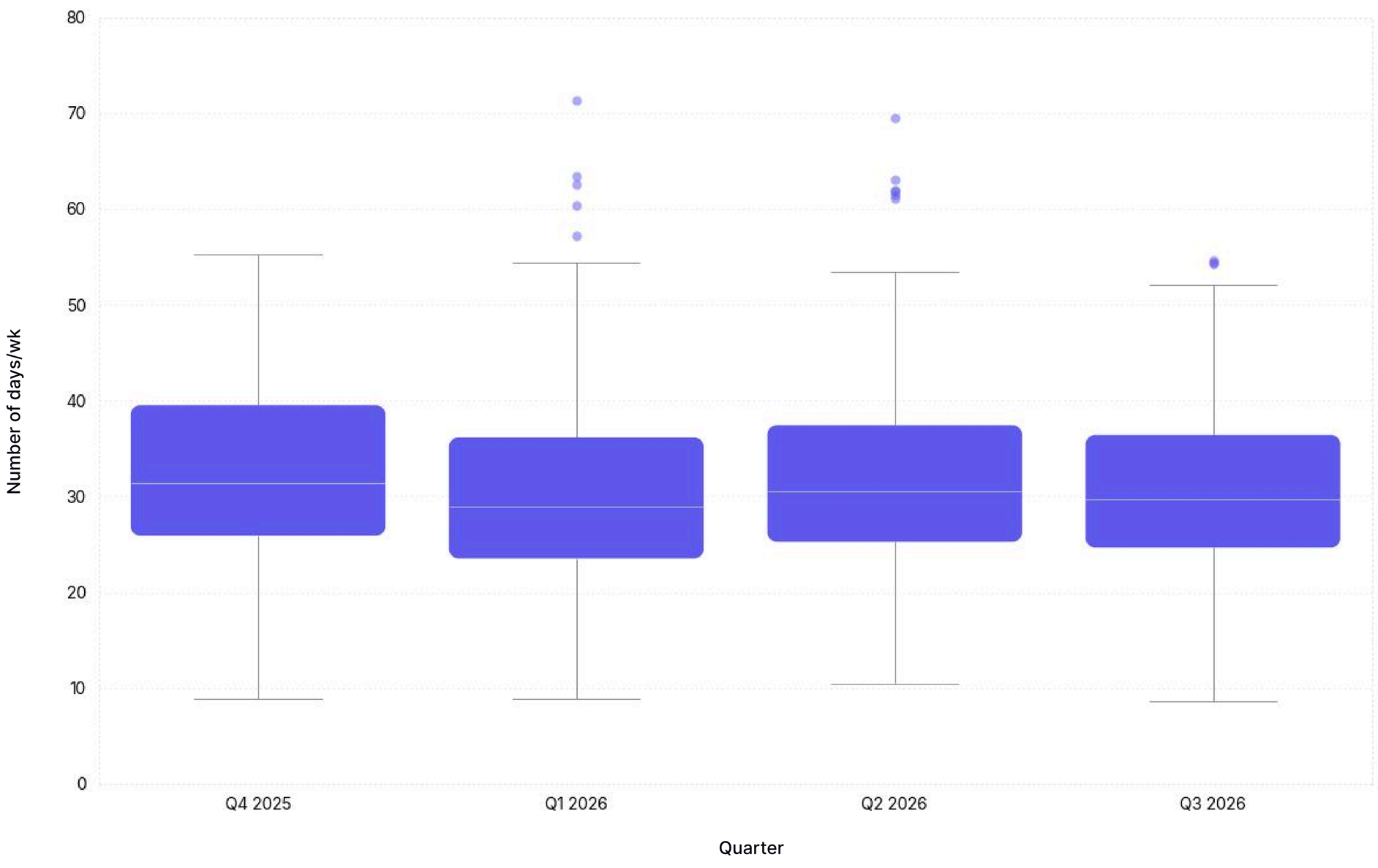
What we can say is that CFR is not improving, that the direction of the signal has changed from flat to marginally positive, and that this is the metric we would watch most closely next quarter. If AI-generated code is introducing subtle defects at scale, CFR drift is where it should surface first and quality debt is by nature lagged, so a Q3 signal this small is consistent with either noise or an early-stage effect.

Notably, the failure mode is not catastrophic. We see no evidence of high AI code volume triggering large production outages. The pattern, to the extent it exists, is a slow accumulation of small, perceptible failures distributed across many teams.

Failed deployment recovery time is a secondary lens on the same question - does slower, weaker signal detection show up downstream once something does fail? Recovery time has remained stable, with the median organization reporting at approximately 30 minutes for each quarter. This stability highlights a growing asymmetry. While AI tools have accelerated velocity, diagnostic and incident remediation tools have not kept pace.

Distribution of failed deployment recovery time over time

Sample from 500+ companies from October 2025 to September 2026



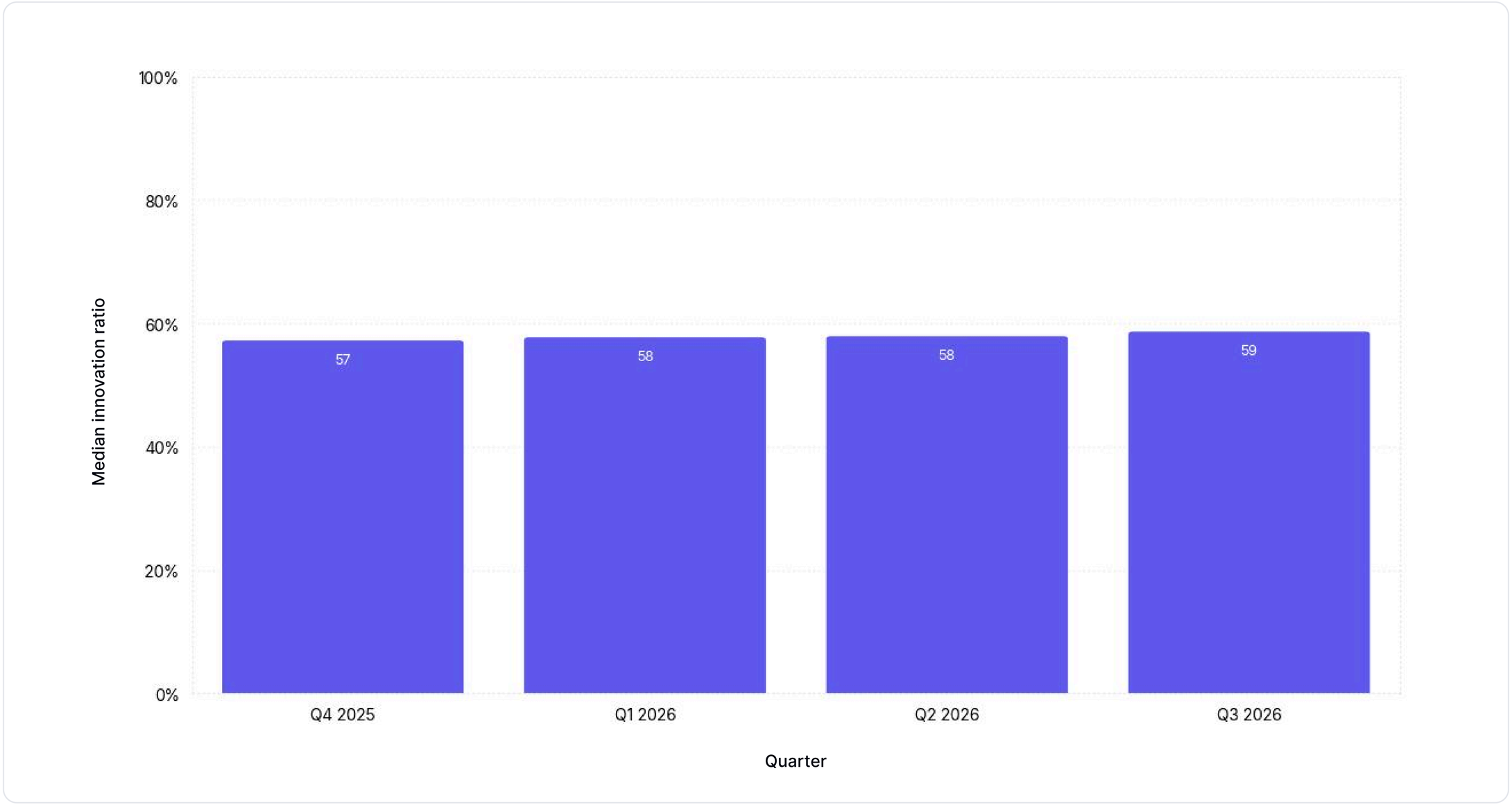
Impact

Один из самых интересных вопросов для Технических директоров.

While AI tools are reportedly saving developers hours, those savings are rarely reaching the product roadmap. Innovation ratio, the percentage of time spent on new feature development opposed to maintenance, support, and bug fixes, has shifted slightly (+1%). This is a small shift, but a statistically significant one ($p = 0.002$). A statistically significant 1-point gain is still far short of what the reported time savings would predict. The signal is reliable, but it is reliably small.

Median innovation ratio

Sample from 500+ companies from October 2025 to September 2026

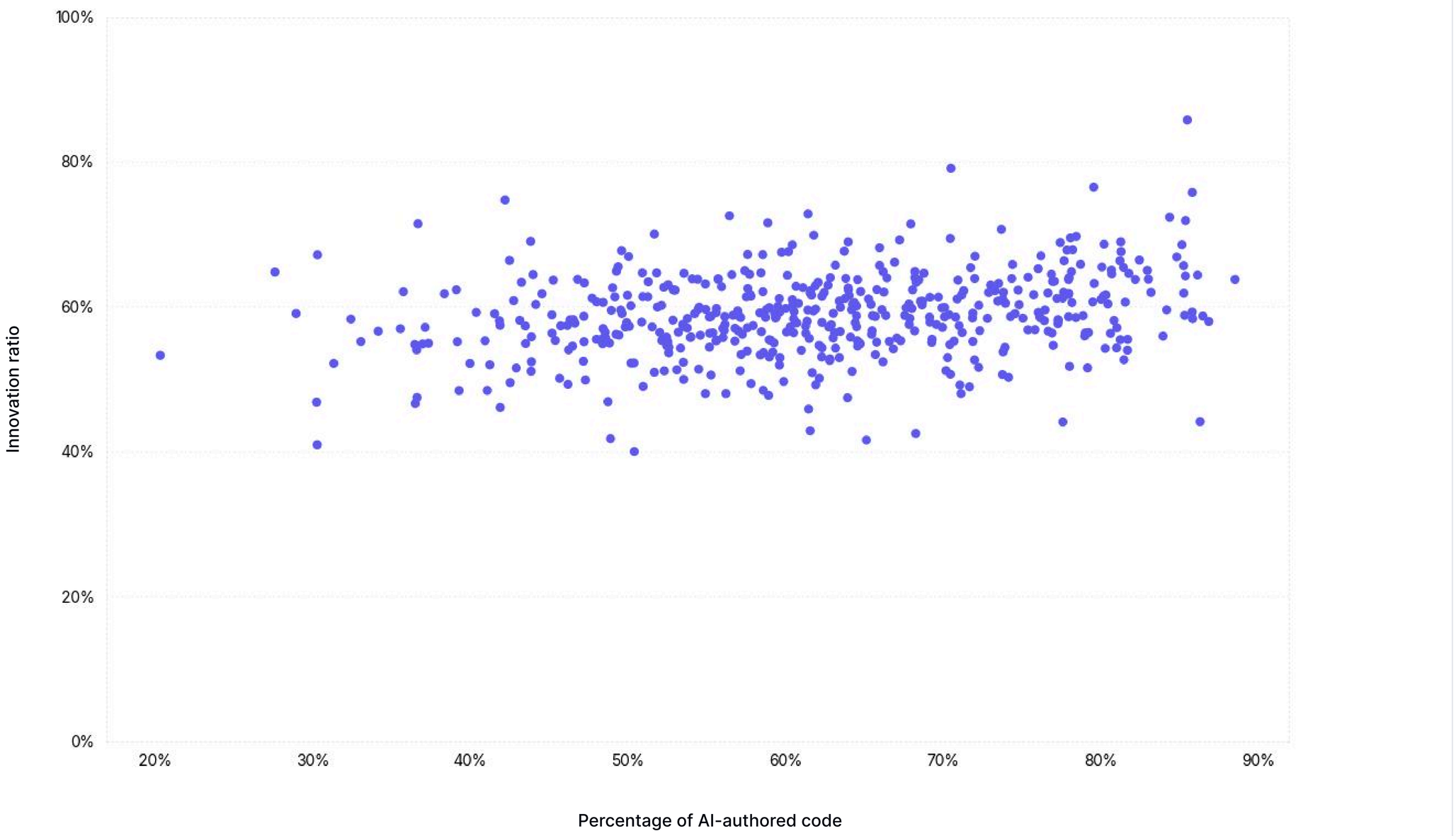


Despite broader tool adoption, overall capacity allocation has shifted minimally. The scatterplot below that maps AI code volume against innovation time reflects this pattern. Reported innovation clusters between 50% and 65% irrespective of code volume. Teams generating a high percentage of automated code allocate roughly the same proportion of time to maintenance tasks as teams with lower utilization.

Странно, я думал, что maintenance можно перекинуть на AI

Mean innovation ratio by AI code volume

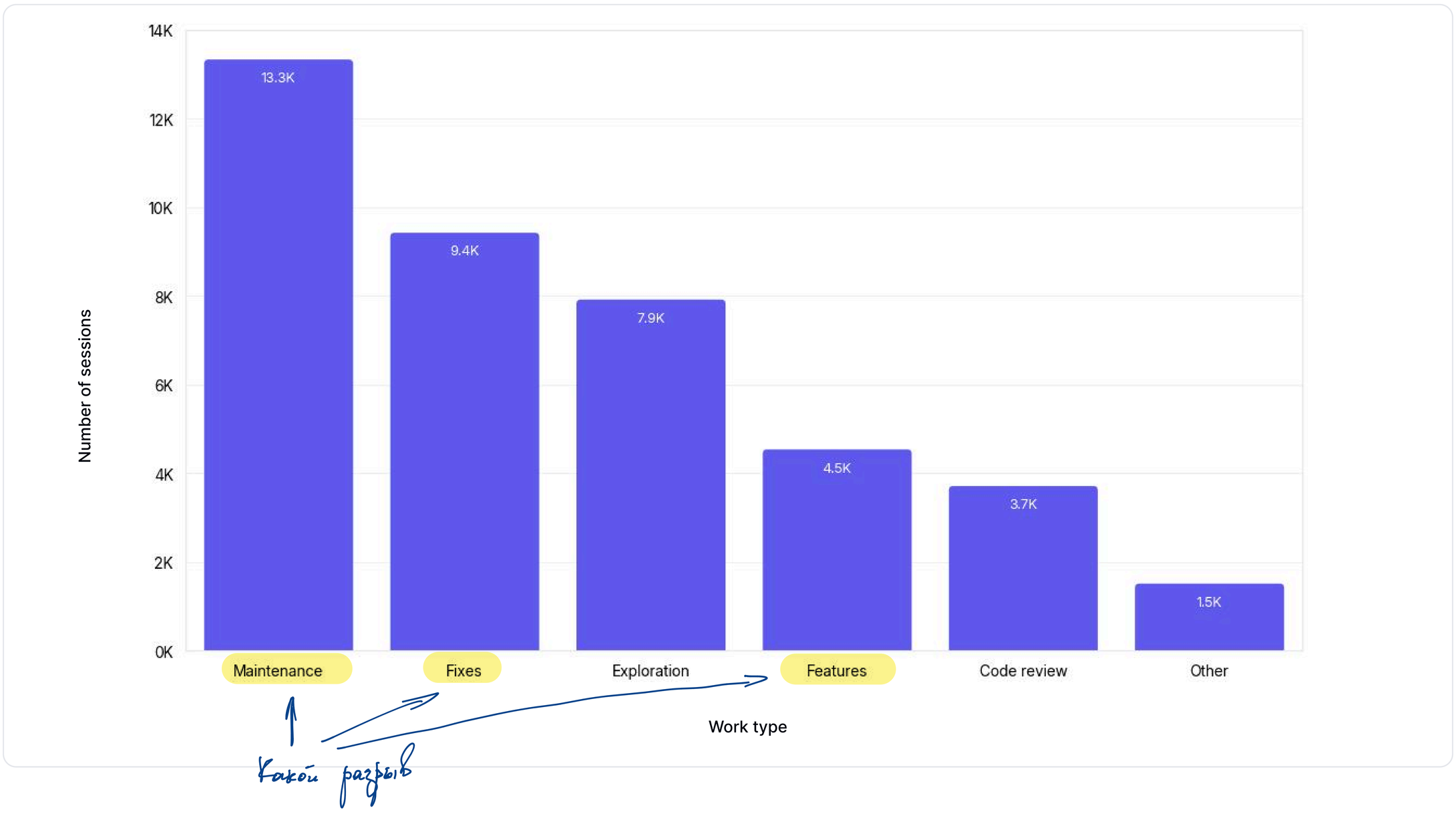
Sample from 500+ companies from July 2026 to September 2026



New telemetry from our AI Code Insights daemon lets us observe what AI is being used for, rather than inferring it from output volume.

AI session by work type

Sample from 500+ companies from July 2026 to September 2026



Roughly 33% of observed AI usage goes to maintenance and 23.3% on fixes, activities that keep existing systems running, compared to 11% that has gone to building new capability. Both feature work and code review are not absent, but are lesser use cases. This session-level breakdown is the clearest direct evidence yet that AI usage is concentrated in upkeep rather than new-feature work, consistent with the flat innovation ratio above.

This aligns with our previously released findings showing that while generation volume correlates with individual AI-related time savings, its relationship with a higher percentage of time spent on new feature work is considerably weaker. Developers report recouping up to 6.1 hours/week as tool usage grows but those hours are not systematically moving into roadmap delivery.

Это большая проблема на фоне значительных трат на AI

Instead, reclaimed time is largely offset elsewhere in the work cycle. When tools produce changesets without full system awareness, the time saved during code generation is absorbed elsewhere in the cycle. Accelerating initial drafting does not automatically alter overall engineering output. While localized speed reduces task-level effort, realization of broader organizational ROI depends on addressing pipeline bottlenecks and information access to convert individual time savings into sustained feature development.

AI cost

> A note on framing: Cost is not a fifth Core 4 dimension, but a dimension from the AI Measurement Framework, placed here to put a dollar figure on the Speed and Effectiveness findings above.

Many of our findings describe a widening gap between authoring and delivery, in units of time and throughput. Cost telemetry reflects a similar pattern. Total AI spend grew 269% from Q4 2025 to Q1 2026, 342% Q1 to Q2, then 106% Q2 to Q3. Spend growth decelerated sharply (from 342% to 106%) while TrueThroughput growth eased only slightly (+21.3% to +19.0%), indicating that cost growth is slowing much faster than delivery growth is. Spend is still climbing fast enough that Q3 alone roughly doubled total spend for the typical account, but the gap between how fast spend is growing and how fast delivered volume is growing narrowed this quarter.

The pace of growth is not reflecting higher prices, but higher volume. The median cost per million tokens held in a narrow band throughout, from $0.58 to $0.73 to $0.78 to $0.71, indicating that the same seats are simply using far more tokens through the pipeline every quarter.

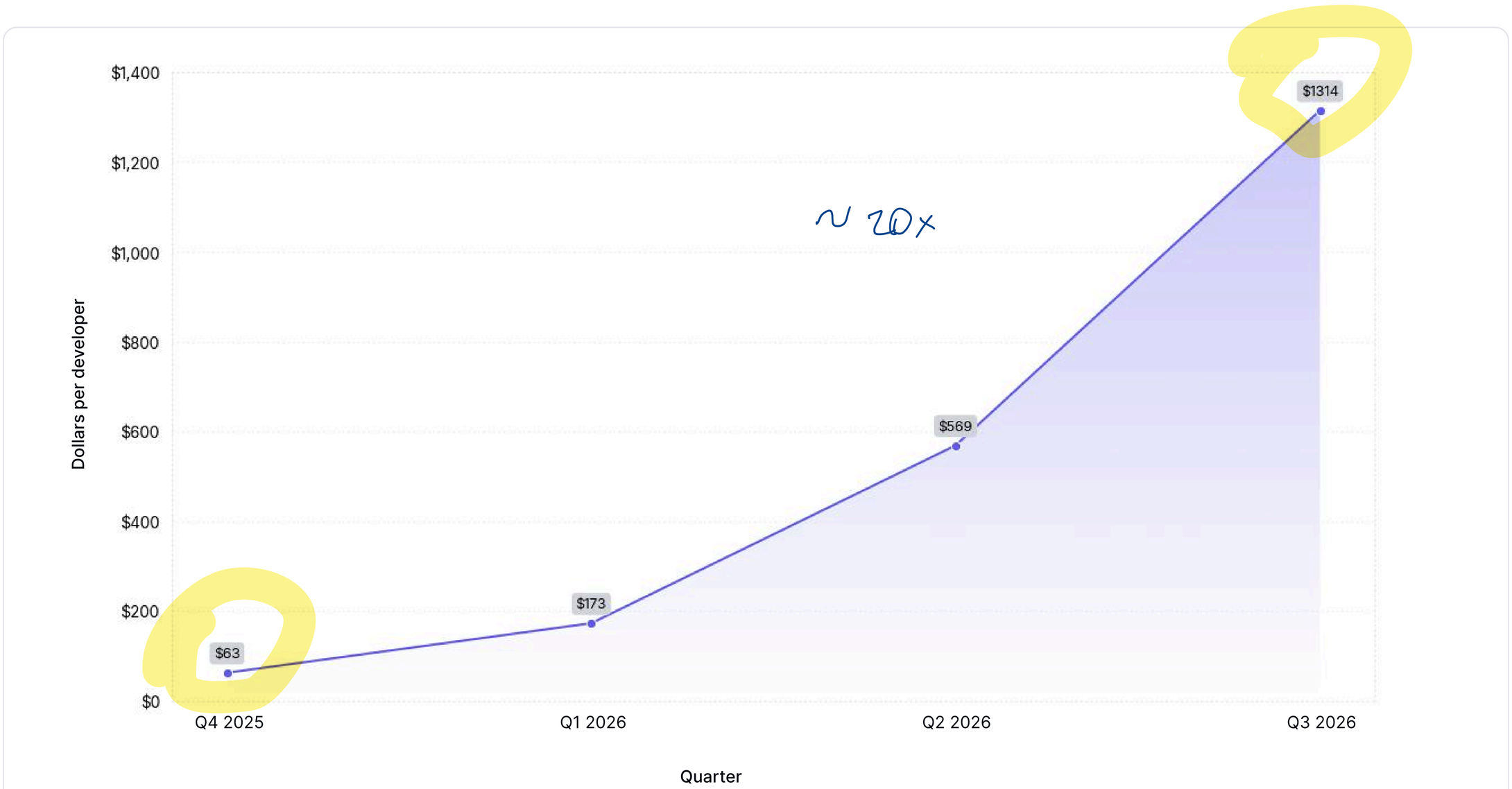
Total AI spend and total token spend

Sample from 500+ companies from October 2025 to September 2026



Spend per developer

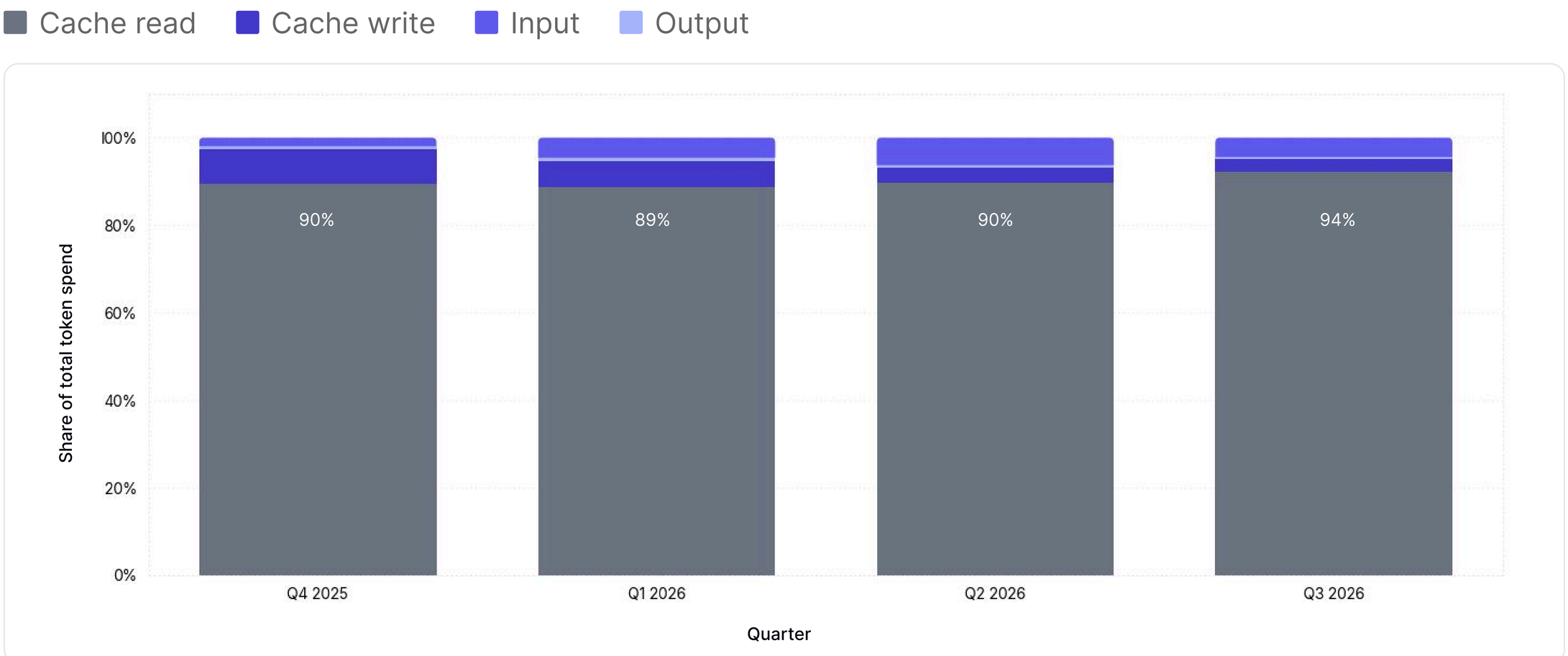
Sample from 500+ companies from October 2025 to September 2026



One shift inside that volume is worth flagging. Cache-read tokens grew from 89.6% to 94.0% of the total mix across the four quarters, while output tokens' share shrank. A growing portion of every dollar spent is going toward re-reading context that was already available, not toward generating new work. Part 2 looks at why that context keeps needing to be re-read in the first place; here, the effect is simpler to state. It shows up as spend that keeps climbing even as less of it produces something new.

Token spend by type

Sample from 500+ companies from October 2025 to September 2026



None of this is cause for alarm on its own: rising AI spend alongside rising AI usage is expected. However, spend has compounded at 2–4x per quarter, while throughput growth has held near a steady 20%. The gap between the two narrowed this quarter as spend growth slowed faster than delivery did.

This cost data shows its own version of the deceleration pattern. Spend growth slowed sharply in Q3, while TrueThroughput growth eased only slightly, remaining in the high-teens. Across the DX Core 4, AI has reduced the cost of producing a change; this quarter's narrowing gap between spend growth and delivery growth is a modest early signal that the cost of trusting one may be starting to improve too, though review wait times, CFR, and ease of delivery haven't caught up yet.

Это интересное наблюдение и гипотеза

Across the DX Core 4, AI has reduced the cost of producing a change without reducing the cost of trusting one. Notably, this isn't a story of some organizations pulling ahead while others fall behind. Throughput, DXI, and innovation ratio all show our industry sample converging rather than spreading out. The open question is why the typical organization, not just a lagging one, is hitting this ceiling. It is not raw model capability — the leading foundation models are broadly available and broadly comparable, and no organization in our sample has a meaningful capability advantage. Something about how work is handed to these systems differs.

Our hypothesis is that one of the differentiators is the quality of context supplied to AI systems and whether the intent, constraints, and architectural reality of a change are explicit enough for a model to reason from rather than guess at. We believe that this is only one major driver and that there are additional credible explanations (i.e. codebase age and size, test coverage, service architecture, review culture, team tenure, and the depth of AI integration) that could each account for meaningful variance.

PART 2

The CAFE(S) framework

Clarity | Actionability | Fidelity | Efficiency | Security

The CAFE(S) framework introduces five durable properties for improving AI agent effectiveness through better context. By measuring the Clarity, Actionability, Fidelity, Efficiency, and Security of inputs provided to AI, leaders can identify why machine-assisted work may be stalling in review and potential paths forward to sustained feature velocity.

CAFE(S): Your Agent Is Only As Good As Its Context — Five properties for evaluating context quality

Clarity SPECIFIC & NON-AMBIGUOUS	**Actionability** DIRECTIVE & GOAL-ORIENTED	**Fidelity** ACCURATE & UP-TO-DATE	**Efficiency** SCOPED & TOKEN-OPTIMIZED	**Security** SAFE & COMPLIANT
Can the agent interpret the request as intended?	Can the agent proceed, and know when it is done?	Can the agent trust that this context is true?	Can the agent focus on what matters?	Should the agent have this context at all?

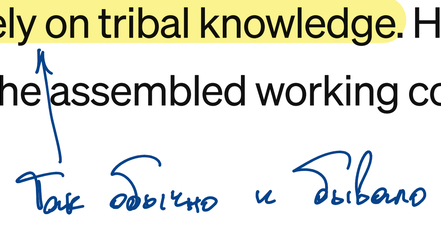
If the differentiator is not capability, spend, tooling, or scale, the most likely remaining candidate is the one thing every organization does differently and few measure: how work is specified before it reaches a model.

> AI does not create a new need for good knowledge management; it dramatically increases the cost of poor knowledge management. -Brian Houck, CAFE(S) framework
>
> Brian Houck, Distinguished Scientist, DX

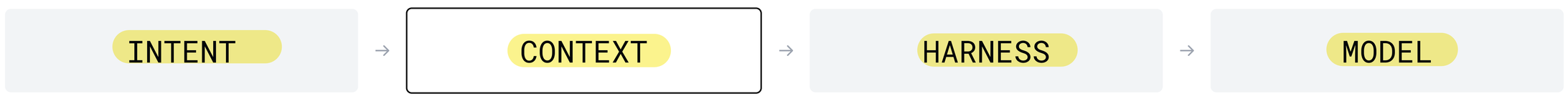


Context has always mattered in software development, but its absence used to surface slowly during onboarding or in a bad architectural decision discovered two quarters later. When delegated to an agent operating at machine speed, the same gap produces a wrong changeset in minutes, reviewed by someone who must reconstruct the missing context to catch it. Worsening developer experience, rising cost per unit of delivery, and increased risk turn what was an abstract concern into an operational one.

This breakdown connects directly to Storey's triple debt model we referenced earlier. Beyond standard technical debt, organizations accumulate cognitive debt – occurring when a team's mental model of a system erodes – and intent debt – the loss of explicit rationale, constraints, and business goals required to evolve software safely over time. An organization can carry intent debt invisibly for years when human engineers rely on tribal knowledge. However, the moment work is delegated to an AI agent, only the explicit knowledge present in the assembled working context is available. Whatever intent was left unstated simply does not exist for the model.



To resolve this friction, we must look at the pipeline through which human intent becomes executed software:



Most organizations have invested heavily in the last two stages and not nearly as much in the first two. Good context gives an agent enough information to reason instead of guess, grounds its work in accurate realities, keeps its attention focused on task-relevant data, and protects it from security exposure. It requires a structured framework to evaluate whether AI agents and developers possess the necessary context to act effectively, which is what we at DX have coined the CAFE(S) Framework. By systematically assessing context along five core dimensions, engineering leaders can dismantle operational drag and unblock capacity for long-term product innovation.

A note on what follows: The mapping in the next section is diagnostic, not evidentiary. It connects each Core 4 symptom to the context failures most likely to produce it, so that leaders have a place to start looking. It is a set of hypotheses derived from the framework, not results measured against it. Instrumenting CAFE(S) across the sample is the work we are doing now, and we hope to report measured results in Q4.

Mapping DX Core 4 friction to CAFE(S) framework

Many of the structural bottlenecks across the Core 4 could be traced back to context hygiene failures during human-to-agent prompting and information retrieval. Evaluating the context supplied to AI agents across the CAFE(S) pillars may reveal how low-quality agent inputs generate downstream delivery friction.

DX Core 4 insight	CAFE(S) diagnosis
Speed Insights: Organizations at the top of the AI utilization curve wait 105% longer for a first review than those at the bottom (82 vs. 40 minutes), even as quarterly TrueThroughput growth has remained near 20% for a full year.	**CLARITY, ACTIONABILITY, & EFFICIENCY** When users feed agents ambiguous prompts (low Clarity) or lack clear boundary constraints (low Actionability), the agent fills in the blanks with plausible but unaligned code. Human reviewers, sensing higher risk and cognitive load, delay picking up these PRs. Furthermore, feeding agents noisy context (low Efficiency) degrade model attention, producing chaotic changesets that sit in longer review queues.
Effectiveness Insights: Perceived ease of delivery improved only 3%, against tactical wins of +13.3% in debugging. Individual relief is far outpacing systemic improvement.	**ACTIONABILITY & FIDELITY** Prompts given to agents often lack explicit, testable completion criteria (low Actionability). The agent generates a non-comprehensive solution, forcing developers to fix subtle edge cases. Because agents are rarely supplied with high-Fidelity context, they generate localized patches that solve immediate surface bugs without respecting broader maintainability.
Quality Insights: Quarter-over-quarter median change in CFR shifted upward by +0.5 ppt, while failed deployment recovery time remained stable at ~30 minutes.	**FIDELITY** When AI models operate on low-Fidelity context (outdated APIs or deprecated patterns), they write syntactically accurate code with subtle runtime edge cases.
Impact Insights: Despite developers reporting up to 6.1 hours per week reclaimed in authoring time, the median innovation ratio moved only from 57% to 59% across four quarters.	**EFFICIENCY** Low context Efficiency may contribute to unnecessary agent work, repeated attempts, and additional human verification. These costs can consume time otherwise available for innovation.

Closing thoughts

The pattern across the Core 4 is mixed: raw delivery volume has grown steadily and quickly for a full year, but the metrics describing how safely and easily that volume moves through review, verification, and release are moving much more slowly, or not at all. The constraint hasn't shown up as less throughput. It is showing up as more strain per unit of throughput.

1. Measure review capacity, not just authoring capacity.

If authoring throughput has grown and review throughput has not, every additional AI seat widens the gap. Track PRs opened per reviewer per week alongside PRs merged per author, and treat the ratio as a capacity metric with a ceiling rather than a background statistic. Most organizations can tell you their adoption rate but very few can tell you how much review headroom they have left.

2. Constrain change size at the source.

Median PR size grew 122% in four quarters, and time to first review rises steeply across AI utilization tiers. Capping change size through convention, tooling, or CI enforcement converts the same authoring volume into more, smaller review units that clear queues faster. This is one of the least expensive interventions available and requires no new platform investment.

3. Shift verification left, and automate it.

Failed-deployment recovery time has not improved in four quarters while change volume rose 75.5%. If human review is saturated and time to first review is effectively fixed, the remaining lever is catching defects before they reach a reviewer. Investment in test generation, static analysis, and automated pre-review verification is where we would expect a better return right now than investment in additional generation capacity.

4. Point AI at the bottleneck, not only at the keyboard.

Our session telemetry shows that code review is one of the smallest observed AI work types, in a pipeline where review is the binding constraint. The next increment of AI value is more likely to come from summarizing changesets, generating review checklists, mapping blast radius, and pre-verifying diffs than from generating more diffs. Organizations that have saturated authoring assistance and not yet applied AI to review are leaving the cheapest capacity on the table.

5. Make intent explicit and machine-readable.

Intent debt, the unstated rationale, constraints, and goals that live in people's heads, is invisible while humans do the work and immediately expensive once agents do. Architectural constraints, ownership boundaries, and acceptance criteria need to exist in the repository, in a form a model can load. This is the intervention with the longest lead time and the largest ceiling, which is why it is worth starting before the measurement is settled.

6. Re-plan around delivery capacity, not generation capacity.

Quarterly throughput growth has held near 20% for a full year, with only a slight easing this quarter. It is a strong pace, but it arrives alongside rising review wait times and a CFR signal moving in the wrong direction, signs this pace may be sustained by absorbing more risk. Plan roadmap commitments against the full friction picture in this report, not just the throughput trendline, and treat review and verification investment as what makes continued growth safe to rely on, rather than overhead competing with it.

Methodology

Sample

The dataset represents a combined sample of developers across 500+ engineering organizations that use the DX platform. Companies range in size from fewer than 50 to over 10,000 employees, represent a wide geographic spread from Europe to Asia to North America, and encompass many industries, including Finance, Retail/eCommerce, and Healthcare. Data covers the period of July 1, 2026 to September 30, 2026. Where trend comparisons are shown, historical data extends back to Q4 2025 (October 1, 2025 to December 31, 2025).

Key metrics and data sources

This report draws on two complementary data sources:

- **System metrics:** Quantitative metrics collected automatically from source control, CI/CD pipelines, AI coding tools, and other engineering systems. Includes TrueThroughput™, PR size, and AI session types.
- **Survey metrics:** Developer feedback and experience data collected through DX's research-based survey instruments. Includes the Developer Experience Index (DXI) and its driver dimensions, self-reported time savings, perceived rate of delivery, perceived software quality, change confidence, code maintainability, ease of delivery, and innovation ratio.

Analytical frameworks

Findings are organized around three frameworks:

- DX Core 4: Measures engineering productivity across four outcome dimensions: Speed, Effectiveness, Quality, and Impact. It synthesizes key principles from DORA, SPACE, and DevEx into a unified methodology designed for executive decision-making
- AI Measurement Framework: Provides context for AI-specific telemetry across three categories: Utilization (adoption and AI-generated code share), Impact (time savings and developer satisfaction), and Cost (total and per-developer AI spend).
- CAFE(S) Framework: A new DX diagnostic framework that evaluates AI context quality along five dimensions: Clarity, Actionability, Fidelity, Efficiency, and Security.

Methodology

Limitations

1. Self-reported time savings and developer experience data are subject to response bias. To mitigate this, we triangulate survey responses with system-level telemetry to validate directional trends, rather than relying on any single data source.
2. Company-level aggregation means individual team variation within organizations is not captured. Where possible, we segment by organization size, industry, and region to surface meaningful differences and avoid overgeneralizing across dissimilar contexts. In other research insights, we also examine the effect of AI on individual behavior to view user-level trends.
3. The sample skews toward organizations that have invested in developer productivity measurement platforms. While this means the findings may not generalize to all engineering organizations, it does ensure a high-quality dataset with consistent measurement practices, making the data especially relevant for organizations evaluating or already using similar platforms.

Data notes

Historical data may not match up to previous reporting on the same time period. This is because DX backfills 24 months of data when onboarding new customers.

Acknowledgements

Special thanks to Collin Green and Sara Gottllieb-Cohen for their contributions and support on this report.

About DX

DX is an engineering intelligence platform designed by leading researchers to help organizations thrive in the AI era. By unifying system metrics with real-time developer and AI agent feedback, DX empowers engineering leaders and platform teams to continuously improve software health and productivity. DX serves hundreds of the world's most innovative companies, including Dropbox, Block, Pinterest, and BNY, to drive higher ROI across their entire engineering workforce.

getdx.com

DX Research team



Gratiana Fu

Research Analyst

Grace is a data scientist and analyst at DX with nearly 10 years of experience turning data into actionable insights.



Brian Houck

Distinguished Scientist

Brian co-authored the SPACE framework, now one of the most widely adopted approaches for measuring developer productivity.



Justin Reock

Deputy CTO

Justin is an engineer, speaker, writer, and software practice evangelist with over 20 years of experience.



Abi Noda

CEO

Abi is the Co-Founder and CEO of DX, where he leads the company's strategic direction and R&D efforts.



Engineering intelligence

Learn more at getdx.com

Copyright © 2026