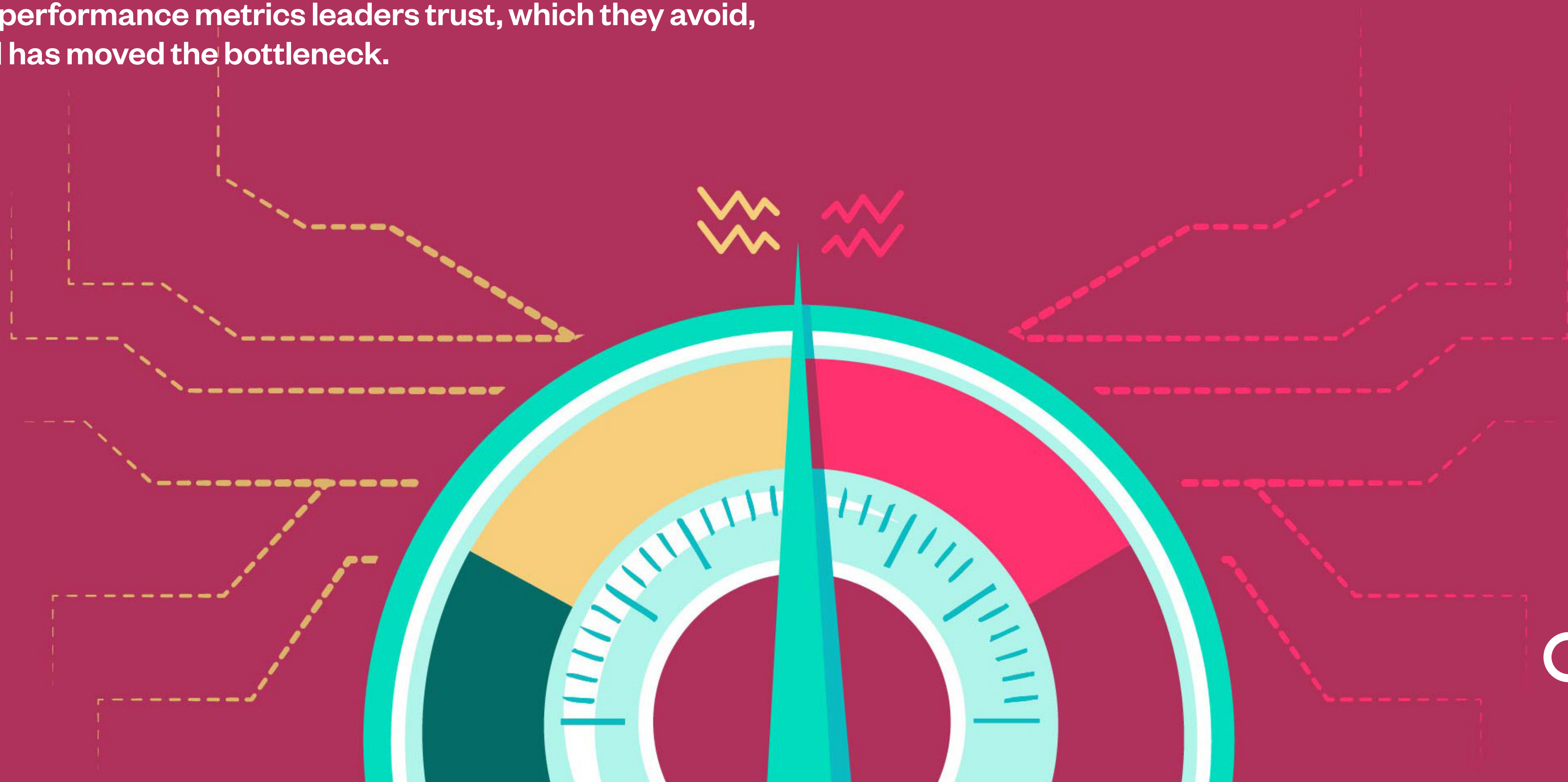


The LeadDev

# Engineering Team Performance Report 2026

Learn which performance metrics leaders trust, which they avoid, and where AI has moved the bottleneck.



O'REILLY®

# Introduction

For the last four years, LeadDev has asked engineering leaders the same question: how do you actually know your team is performing? The latest version of our Engineering Team Performance Report, based on responses from 384 engineering professionals, digs into that question again. The answer, as usual, is messier than any dashboard would suggest.

Our latest survey lands at a poignant moment for the profession. AI has changed what engineers do day to day faster than anyone has figured out how to measure it, and this year's responses make that gap impossible to ignore. Metrics built for a world of manual pull requests and predictable cycle times are now being asked to explain a workflow that barely resembles the one they were designed for.

What emerges from the data is a picture of an industry that's still considering what "performance" even means as AI rewires what engineers spend their time on.

Five themes stood out clearly enough to shape this report, from who's actually accountable for reporting to where the real bottleneck in software delivery has moved.

**Michael Hill**

Editor, LeadDev



# A note from O'Reilly

O'Reilly's mission is to change the world by sharing the knowledge of innovators. For over 45 years, we've inspired companies and individuals to do new things – and do things better – by providing them with the skills and understanding that are necessary for success. By partnering with LeadDev to create this report, we continue in that tradition with you.

At the heart of our business is a unique network of experts and innovators who share their knowledge through us. The O'Reilly learning platform offers exclusive live tech and business events, AI-powered answers you can trust, interactive learning, on-demand courses, a certification experience, videos, and more, making it easier for our customers to develop the expertise they need to get ahead. Our books have been heralded for decades as the definitive place to learn about the technologies that are shaping the future.

Everything we do is to help professionals from a variety of fields learn best practices and discover emerging trends that will shape the future of the tech industry. When you and your teams are ready to learn what's next, you'll find us at [oreilly.com](https://oreilly.com).

**O'REILLY®**

# Key findings

**57%** say choosing which metrics to use is a challenge

**44%** trust their metrics reflect real value

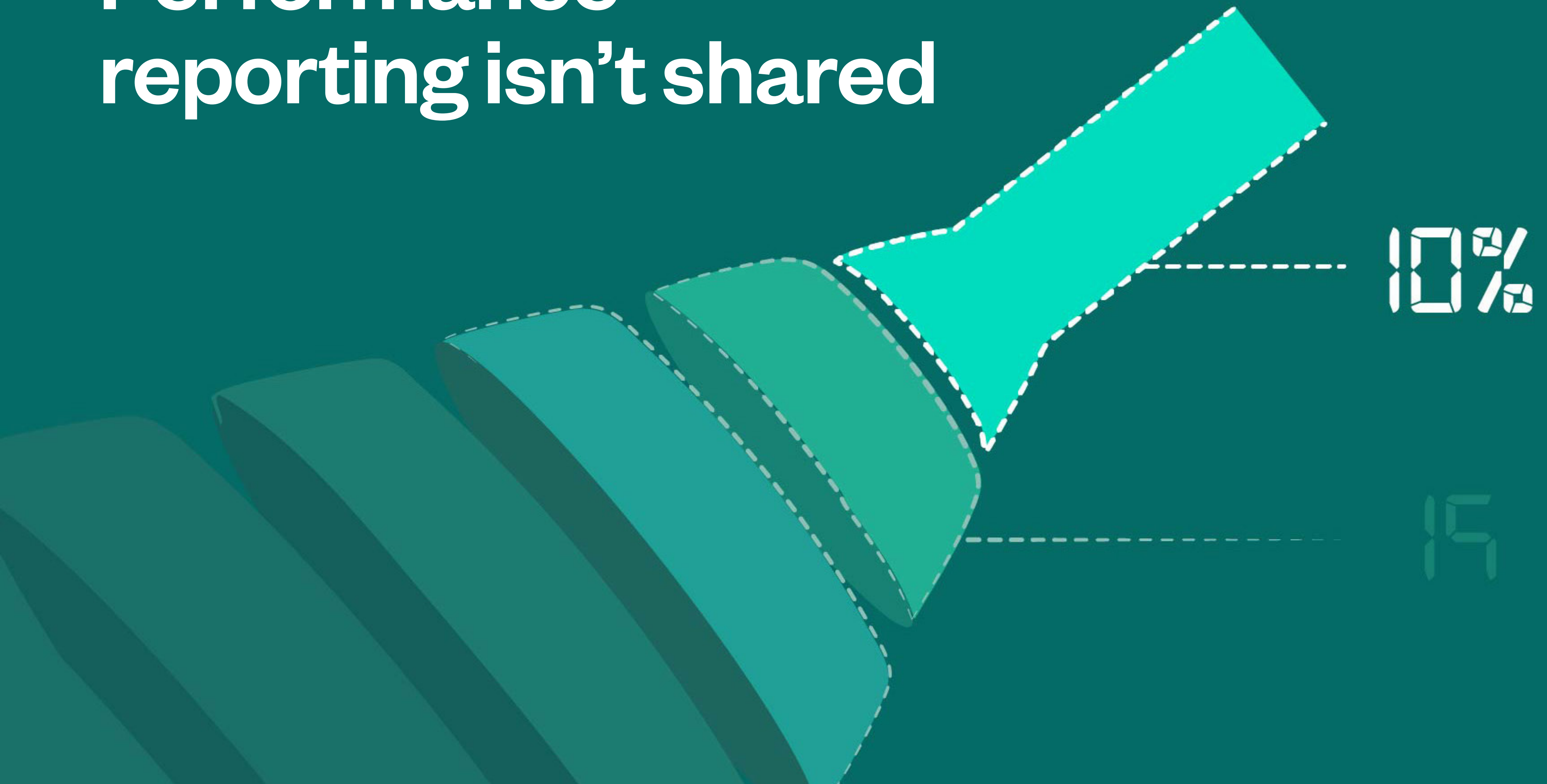
**10** percentage points fewer use developer satisfaction surveys

**72%** more organizations measure performance with PRs (vs 2025)

**54%** say upskilling will focus most on working with agents

**46%** say developers are spending more time building new features because of AI

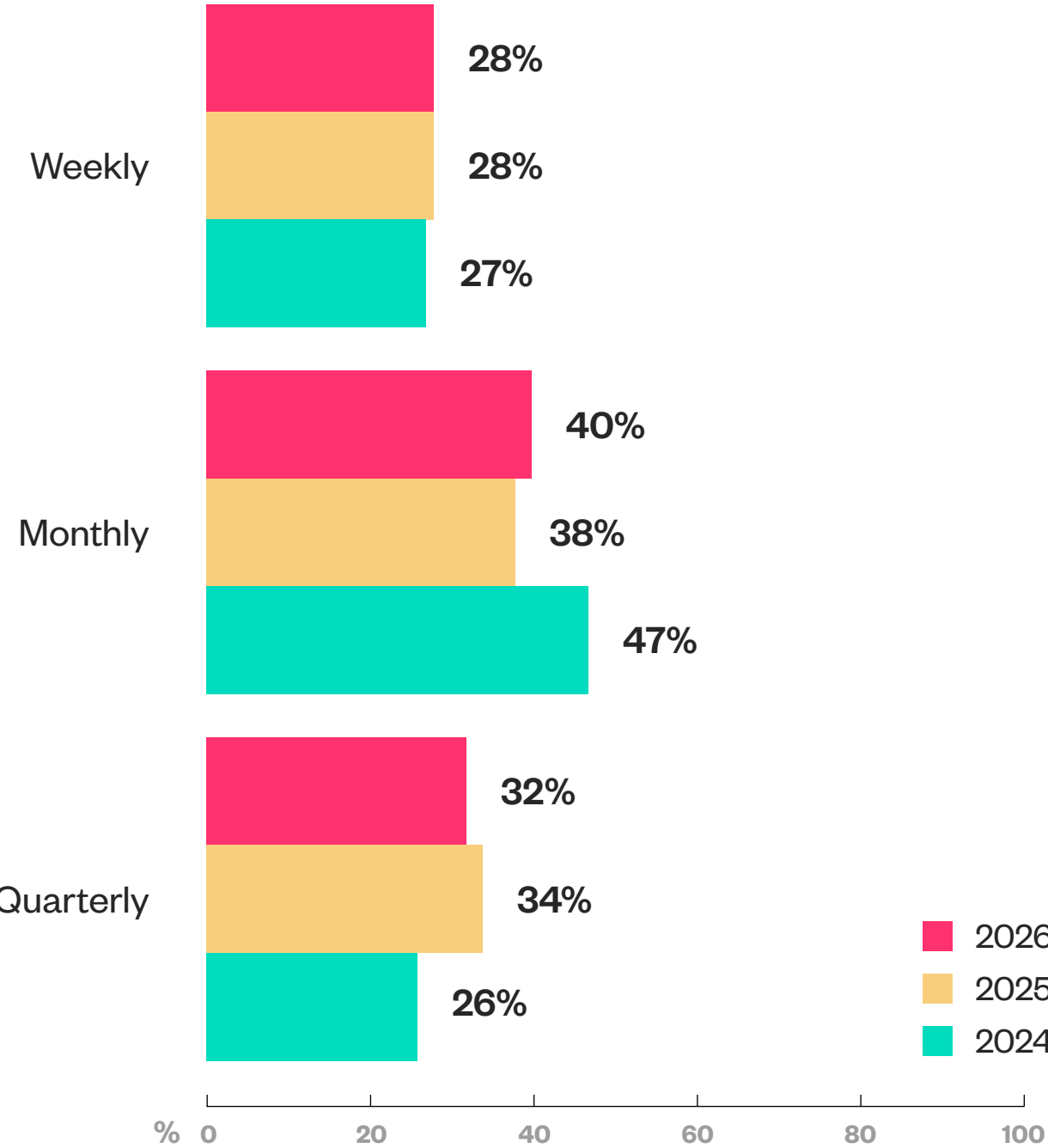
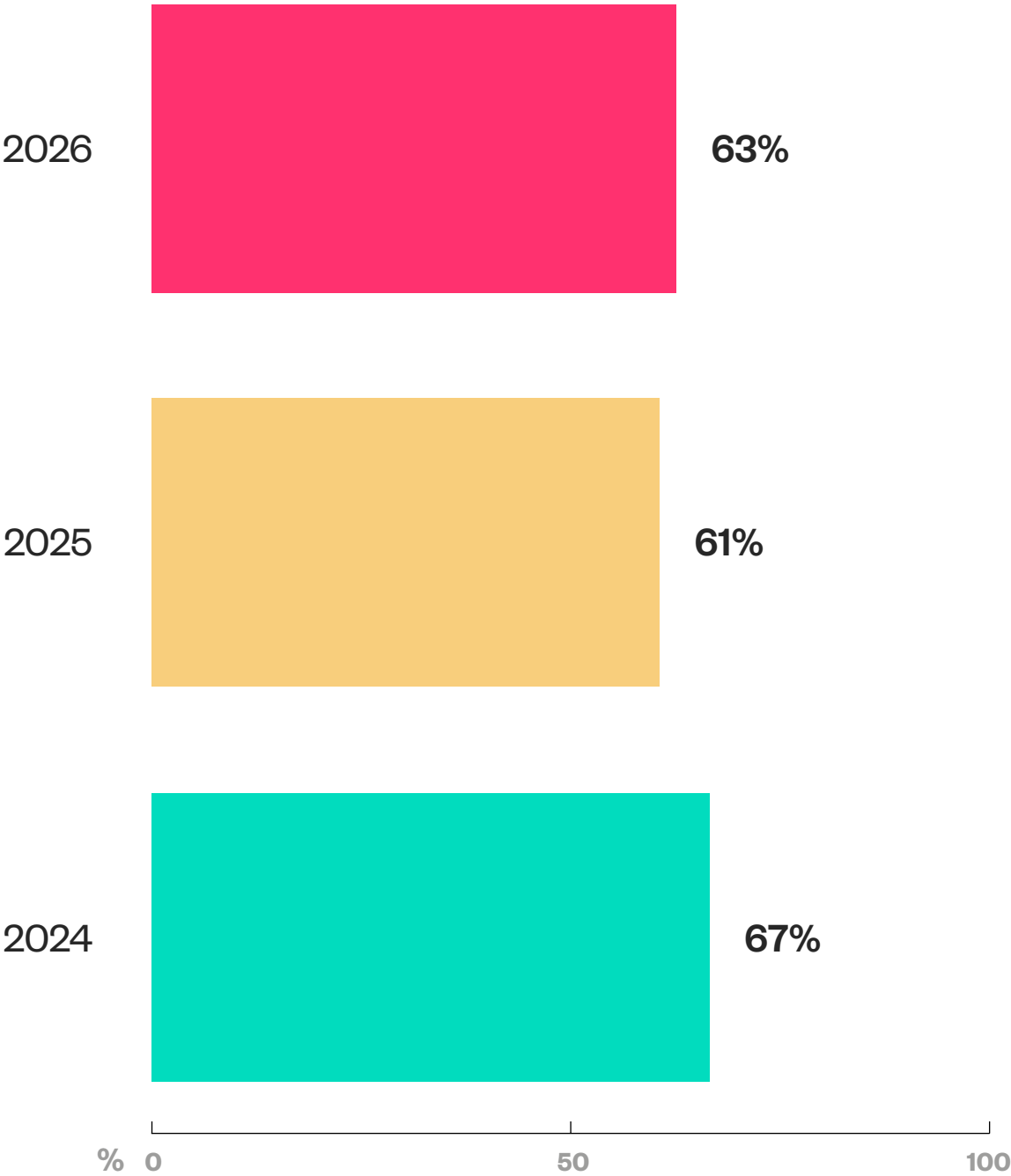
# Performance reporting isn't shared



Managers and tech leads look at team performance differently. Almost three quarters (74%) of managers and executives are required to report on team performance, compared to just 44% of everyone else.

Of those that are personally required to report on team performance, 28% do it weekly, 40% monthly, and 32% quarterly. Tech leads are the most diligent: 48% of them track performance weekly, versus 22% of everyone else involved in reporting.

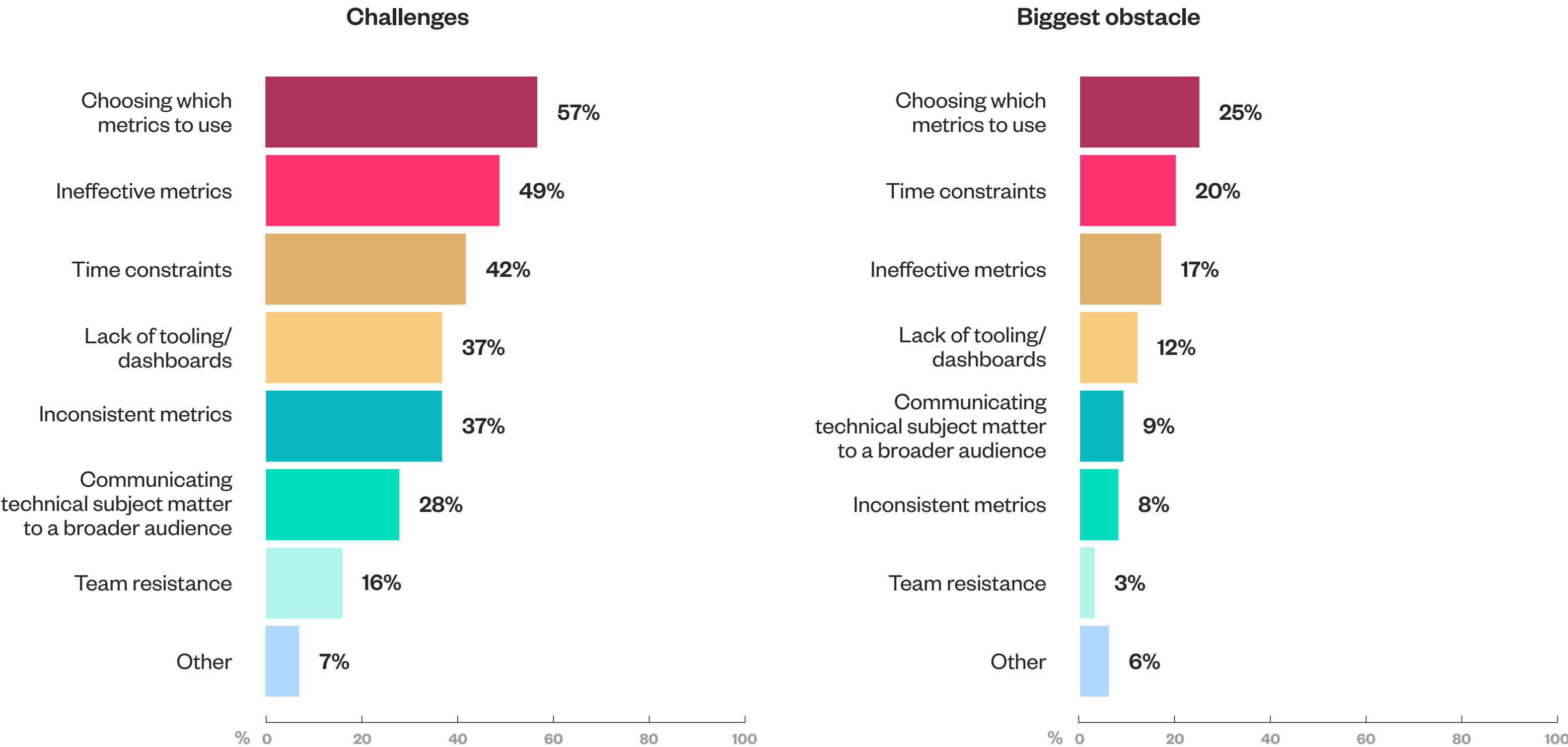
How often are you required to report on team performance?



For the people who do have to report, the process is difficult. Over half (57%) cite choosing which metrics to use as a challenge, while 49% say the metrics they have are ineffective and 42% point to time constraints. Lack of tooling and inconsistent metrics are both challenges for 37% of the respondents.

Managers of managers are the ones struggling most with choosing metrics (71%), while organizations with under 20 engineers are most burdened by time constraints (62%) and large organizations with 500 or more engineers suffer from inconsistency between teams (51%). When asked to name the single biggest obstacle, choosing the right metrics still wins, at 25%, ahead of time constraints (20%) and ineffective metrics (17%).

If required to report on team performance, which of the following challenges do you encounter with reporting on team performance? Which of these challenges is the biggest obstacle to reporting on team performance?



Reporting struggles aside, the friction shows up in how teams work together day to day. More than two thirds (68%) of respondents say aligning priorities across teams is a challenge, whereas 57% point to ownership and accountability and 52% cite a lack of information or visibility.

Tech leads feel this more acutely than anyone: 79% of them flag ownership and accountability as an issue, well above the 57% average. One respondent summed up a version of this

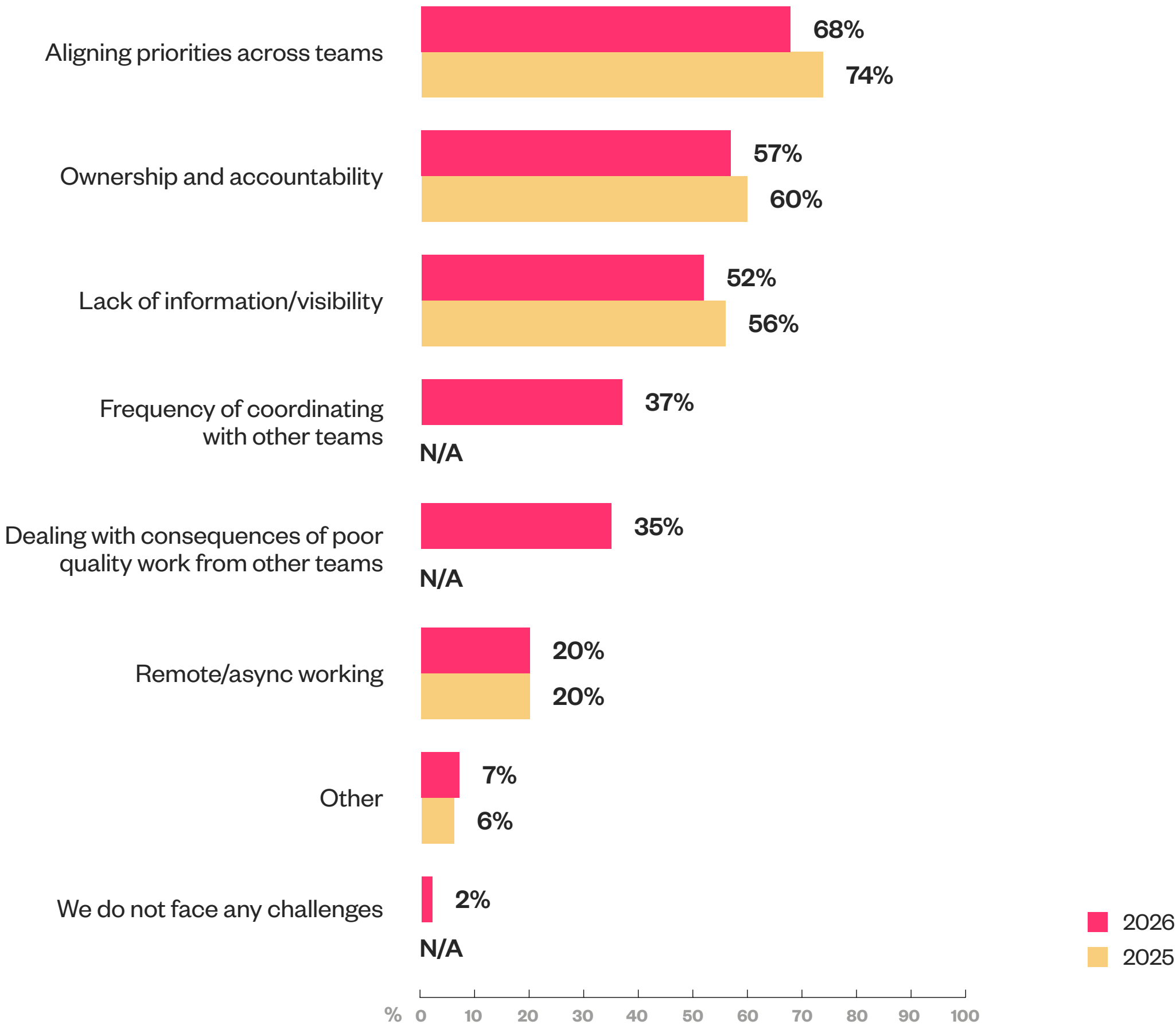
frustration bluntly, describing how non-technical stakeholders keep “designing solutions when you don’t have any clue about software.”

Ask people why they measure performance in the first place, and the answers get more grounded. Almost a quarter (24%) say it’s to improve engineering velocity, 19% to improve quality, and 18% to identify process bottlenecks. Only 14% say it’s about creating accountability and just 13% admit it’s to manage upwards.

### What this means?

Reporting is something most people would rather use as a diagnostic tool than a scorecard. The gap between that instinct and the actual, top-heavy way reporting gets enforced is where a lot of this year’s frustration lives.

### What challenges does your teams face when collaborating internally?





**Metrics still  
don't earn trust**

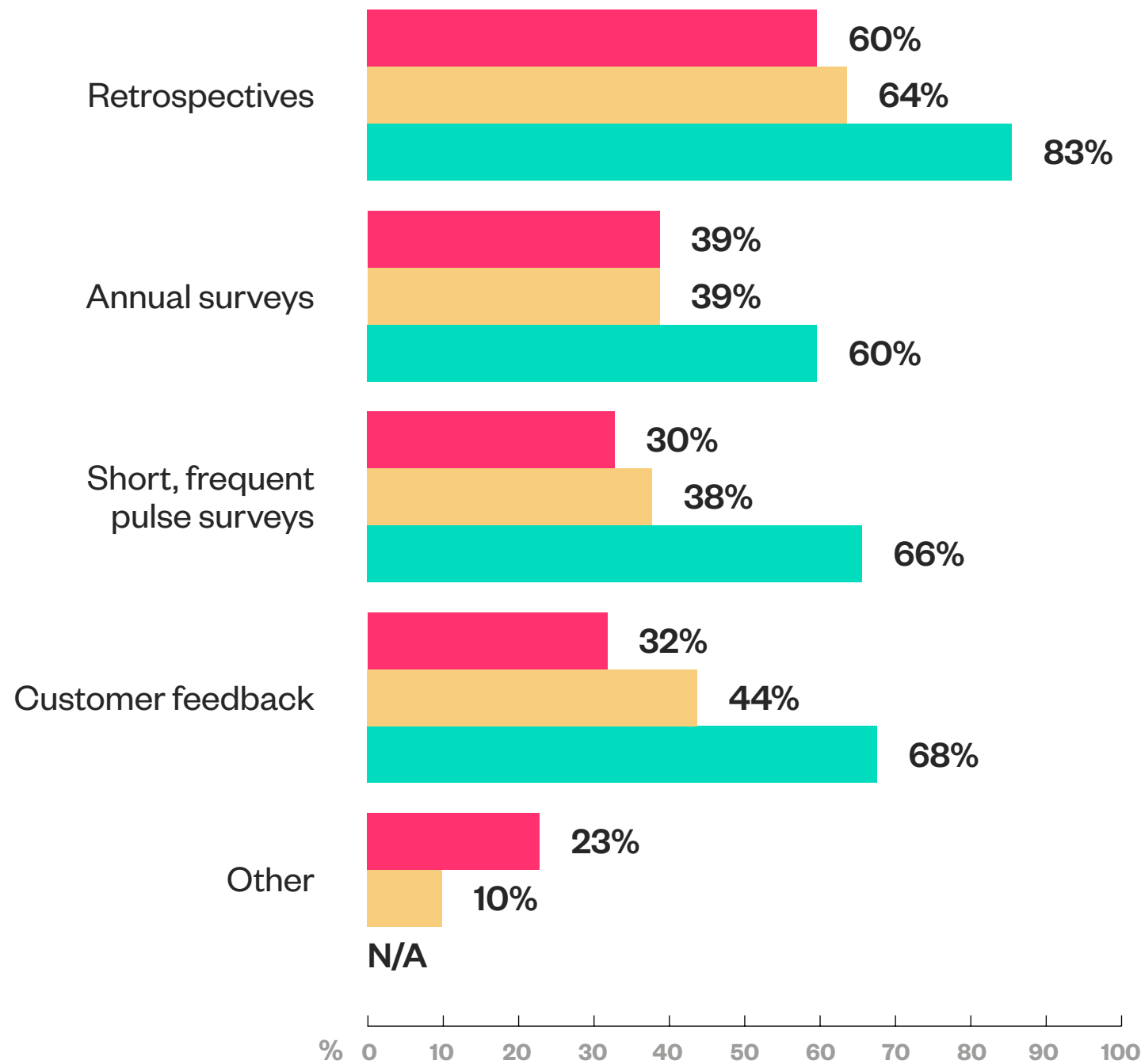
If there are two themes running through the whole survey, they are skepticism toward existing metrics, and a shift toward harder, operational ones like incidents, lead time, and pull requests.

At a high level, many methods of measuring team performance have become less common. Retrospectives are used by 60% of organizations, but that's down from 64% in 2025 and a steep drop from 83% in 2024.

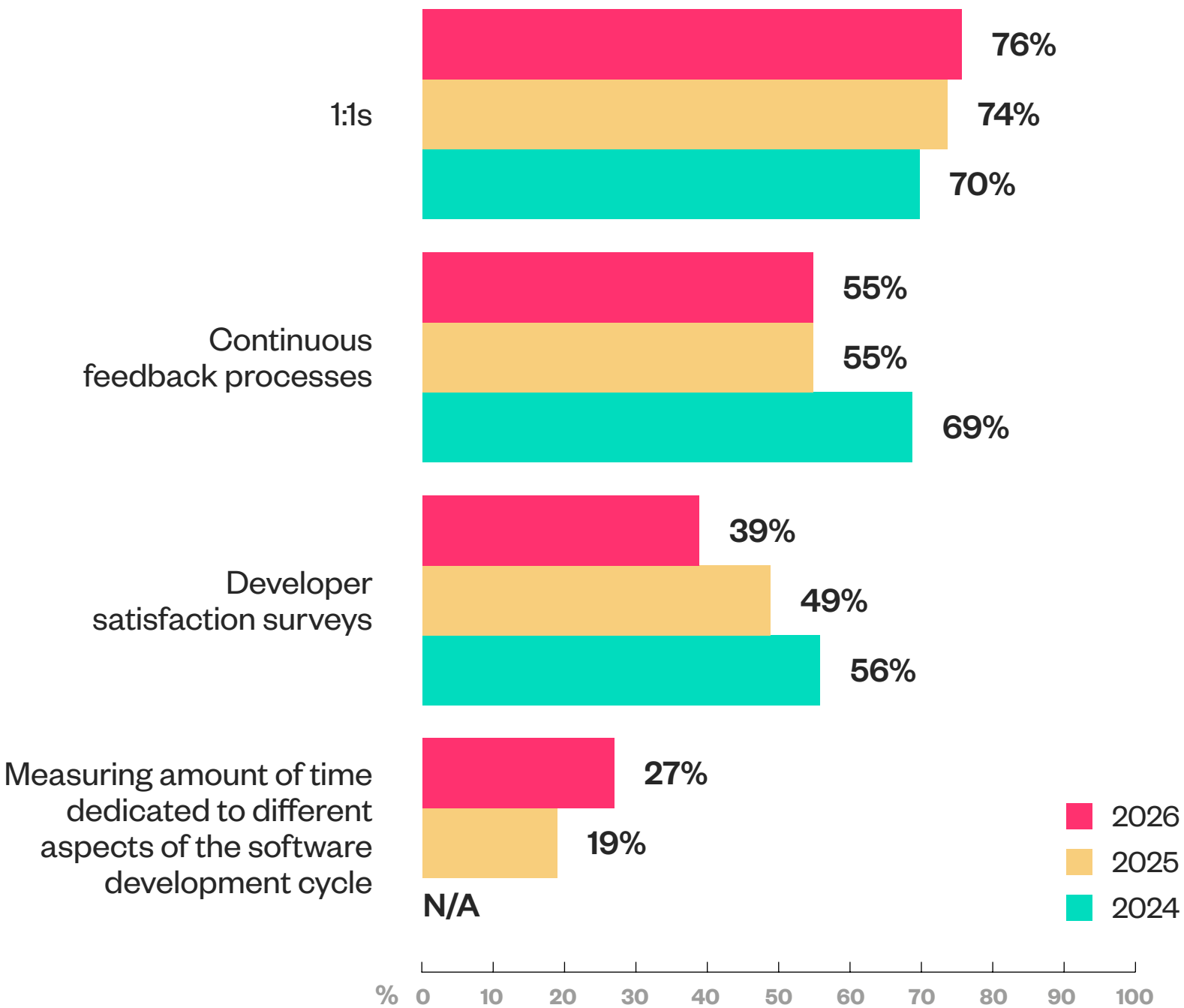
Annual surveys (39%) have held steady, while pulse surveys (33%, down from 38%) and customer feedback (32%, down from 44%) have both declined. However, customer feedback is much more common (55%) at organizations with fewer than 20 engineers, where it's often the only signal available.

When it comes to measuring developer experience specifically, nothing beats a direct conversation. Over three quarters (76%) of respondents say one-on-ones are the most useful method. Developer satisfaction surveys, by contrast, keep losing ground: 56% found them useful in 2024, 49% in 2025, and just 39% this year. At small companies, only one in five respondents still bother citing them.

What methods does your organization typically use to measure team performance?



Which methods have you found the most useful to measure developer experience?

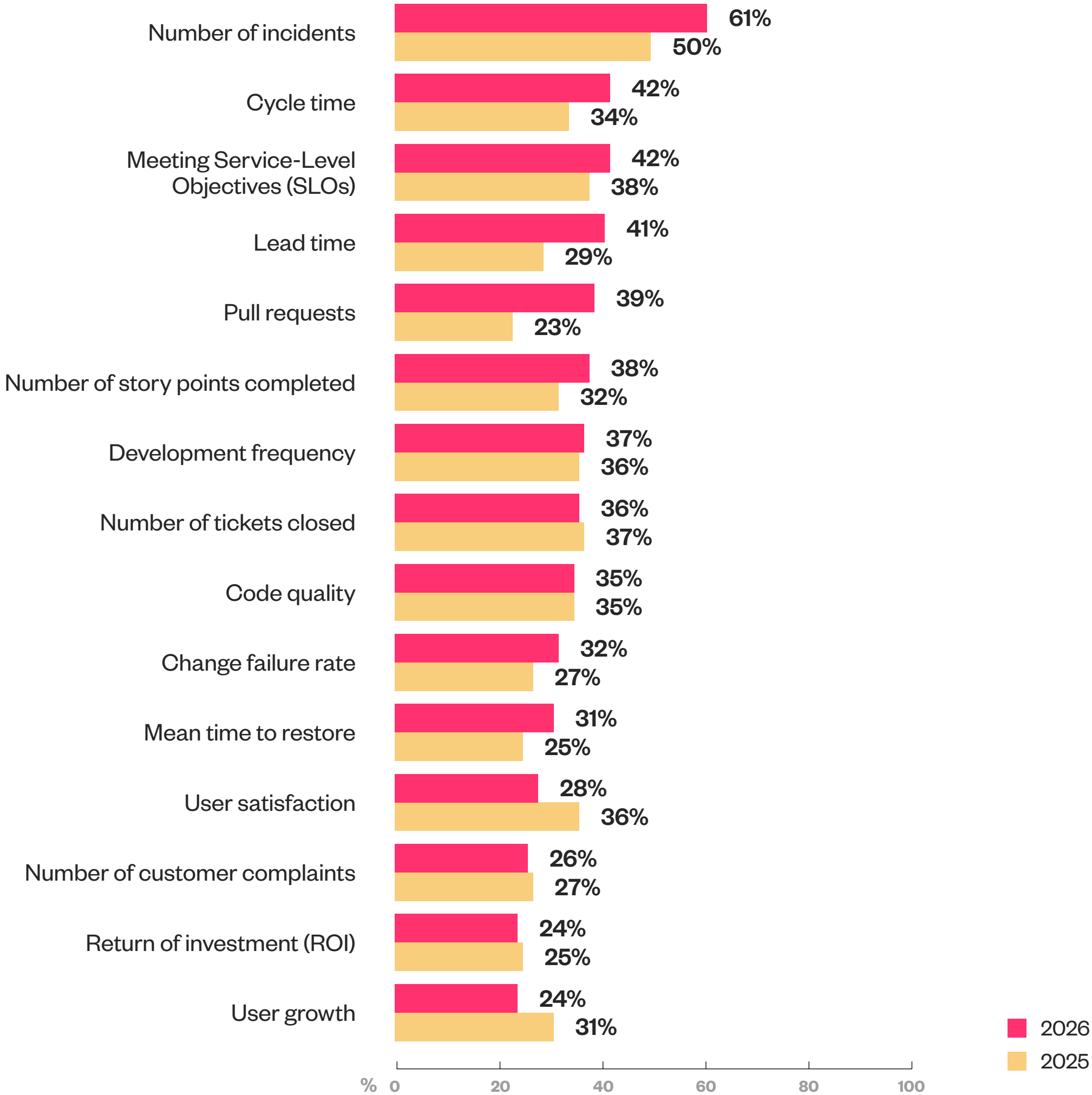


The prevalence of several metrics has also seen changes. Operationally, tracking the number of incidents is more common (61%, up from 50%), as is meeting service-level objectives (42%, up from 38%). Tracking user satisfaction (28%, down from 36%) and user growth (24%, down from 31%) have both fallen out of favor. Team leads are the exception here. User satisfaction remains a favorite, cited by 44% of them.

There have been big jumps for several development-related metrics. On the development side, cycle time (42%, up from 34%), lead time (41%, up from 29%), and pull requests (39%, up from 23%) all saw similar increases.

Managers of managers lean hardest into the more failure-oriented metrics: 78% track number of incidents, 59% track lead time, and 51% track mean time to restore.

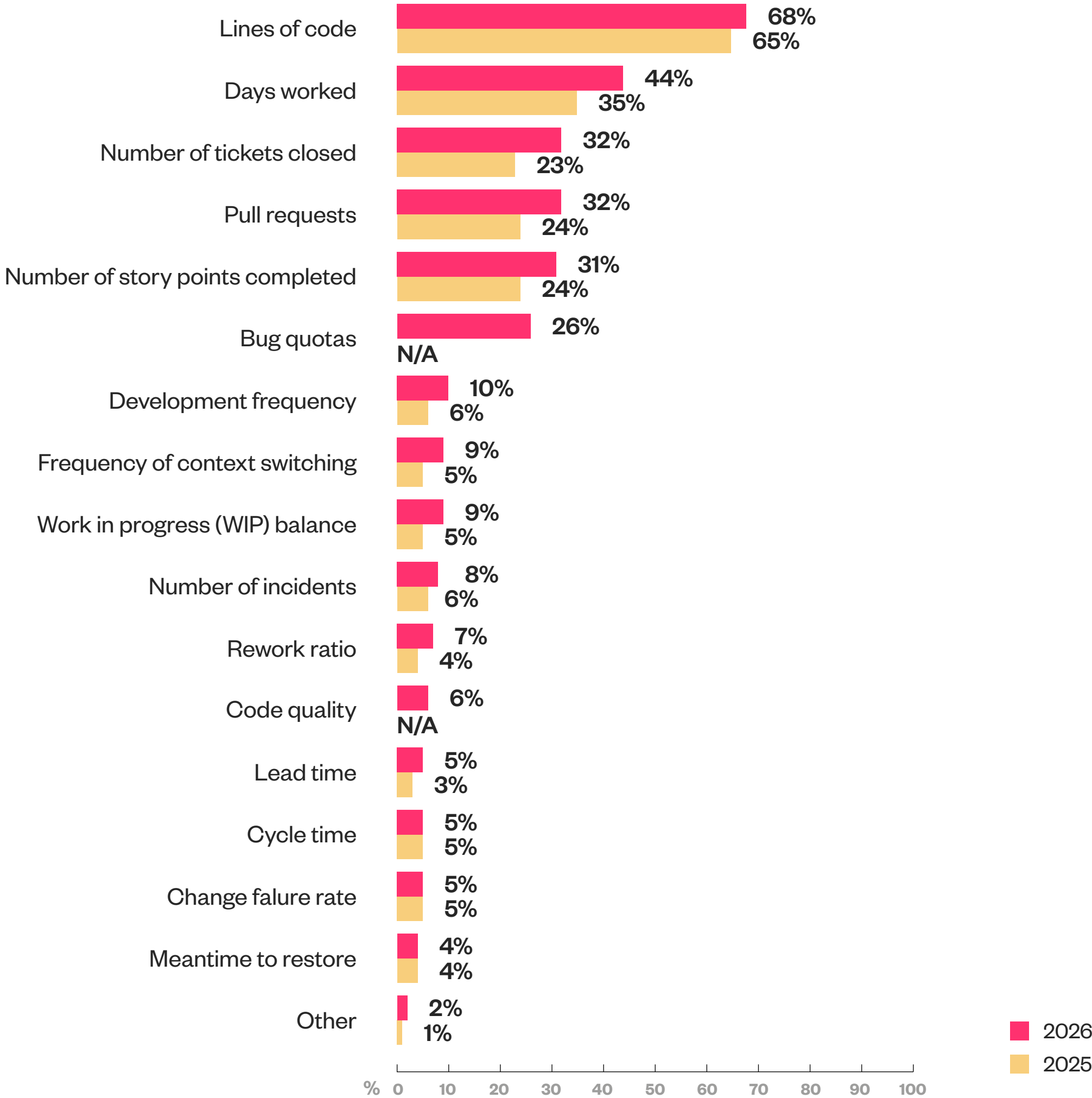
Which of the following metrics are used in your engineering organization to measure engineering team performance?



However, the metrics people avoid are just as telling as the ones they use. Lines of code tops the list by a wide margin, avoided by 68% of respondents, followed by days worked (44%), tickets closed (32%), pull requests (32%), and story points completed (31%). The reasoning, drawn from open-ended responses, splits into a few recurring complaints: metrics get gamed, they reward activity over outcomes, and AI has made several of them meaningless overnight.

As one respondent put it, “with AI it is easy to make yourself look busier than you are.” Another was more direct: “people already game ticket metrics by making every Slack thread a Jira ticket.”

What development-related metrics do you or your team try to avoid?



That skepticism runs deep. Only 44% of respondents believe their current metrics accurately reflect the value their team delivers, though that climbs to 63% among managers of managers, who tend to have more say in which metrics get used. Almost two thirds (65%) say they can tell their team's performance has declined even without their current metrics.

When asked whether the effort spent on reporting is worth what they get back, the honest answer from most people is a shrug. Just 36% agree it's worth it, while 30% disagree and a third sit on the fence.

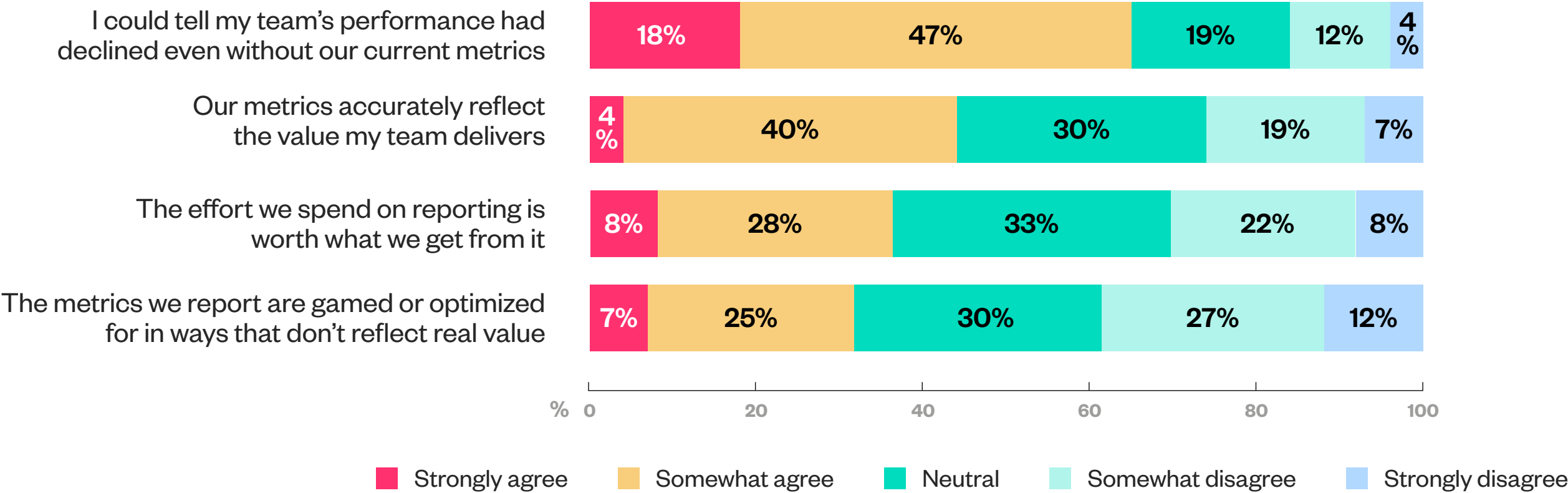
There's a bit of good news buried in here for anyone using DevOps Research and Assessment (DORA) metrics. Among respondents aware of the framework, 50% now rate it effective or very effective, up from 41% in 2025.

The Satisfaction, Performance, Activity, Communication, Efficiency framework (SPACE), on the other hand, hasn't fared as well. Awareness of it has fallen back to roughly where it stood when the framework first launched, with 61% of respondents saying they either don't know it or don't use it, up from 43% in 2024.

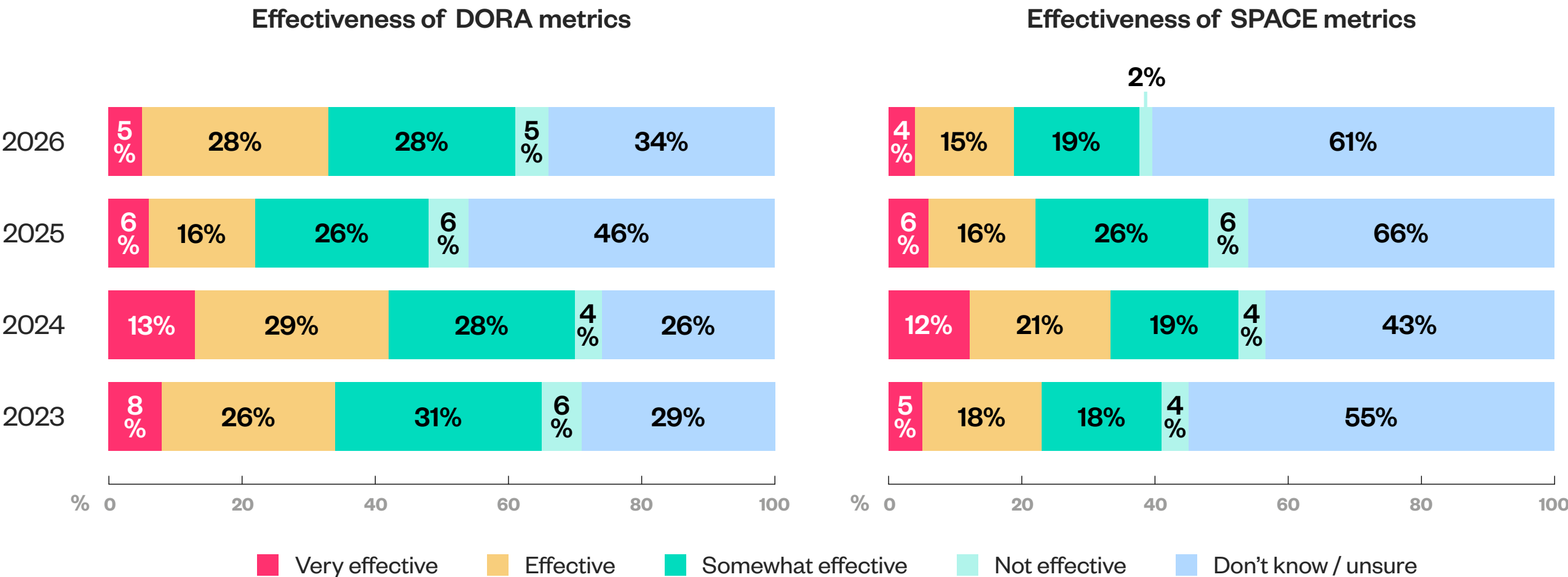
### What this means?

Trust in metrics is low because most still measure activity, not outcomes, and AI has made that gap obvious. If you can't explain why a metric was chosen since AI changed what "output" looks like, it's probably losing trust, not building it.

### How much do you agree with the following?



### How affective are the following at measuring team performance?



Your team is retraining for AI agents



*Loading*

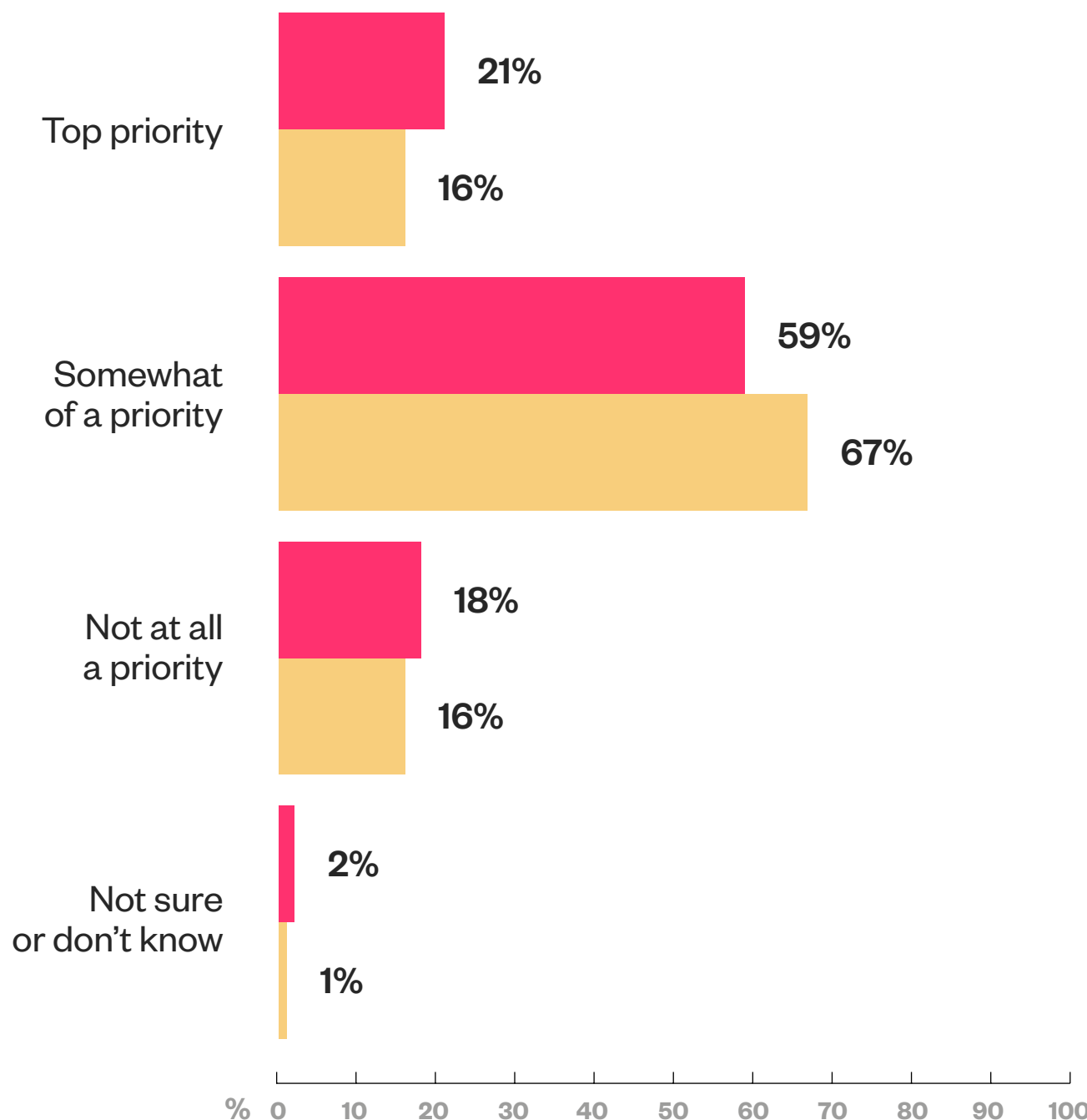
Learning and development gets more lip service than budget. Just 21% of respondents say it's a top priority for engineers at their organization, though that's an improvement on the 16% recorded in 2025.

More than half (59%) call it “somewhat” of a priority, down from 67%, and 18% say it's not a priority at all. Individual engineers are slightly less likely to consider L&D a top priority, while 31% of both CTOs and tech leads say L&D is a top priority for their organization. CTOs want L&D to help retain talent whereas tech leads want to upskill in new strategic areas.

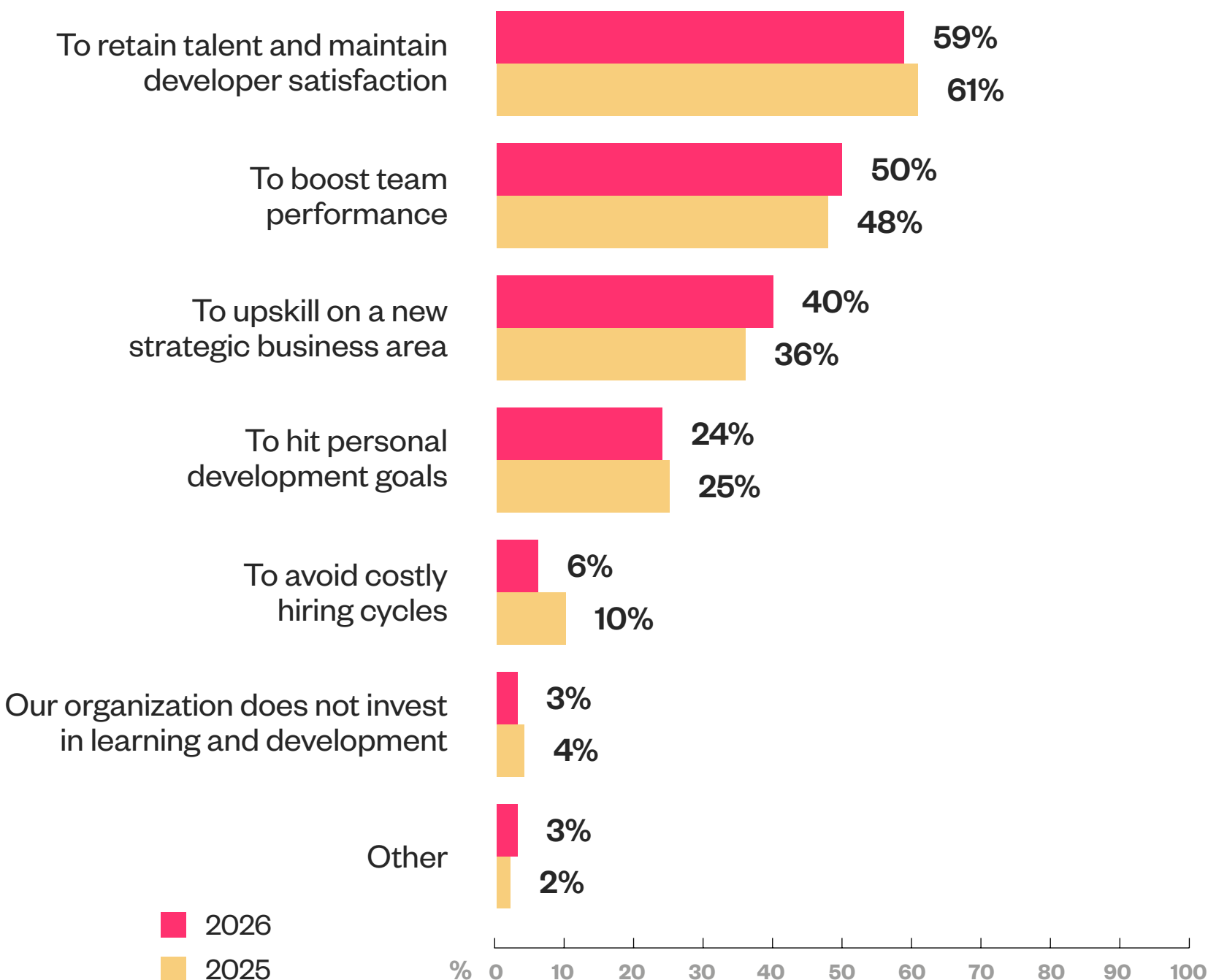
The reasons organizations invest in L&D reveal a real disconnect between leadership and the people actually doing the learning. Over half (59%) cite retaining talent and maintaining developer satisfaction as a top reason, and 50% cite boosting team performance.

Break that down by role, and the gap is stark: 81% of CTOs and managers of managers say retention and satisfaction justifies L&D spending, compared to just 40% of junior and senior engineers. Leadership seems to believe L&D is a retention lever far more than the engineers on the receiving end do.

How much of a priority is learning and development for engineers at your organization?



What are the top two reasons your organization invests in learning and development?



What people want to learn has shifted almost entirely toward AI, with 72% of teams looking to upskill on working with coding agents, whether that's delegation, scoping, or supervision. More than half (54%) want to get better at reviewing AI-generated code and 43% are focused on prompting and context engineering.

Interest in cloud-native technologies, meanwhile, has collapsed, falling from 39% in 2025 to just 21% this year. When forced to pick a single top priority, working with coding agents wins clearly at 54%. Smaller organizations' plans diverge here: at companies with fewer than 100 employees, only 38% prioritize coding agents, while 19% prioritize reviewing AI-generated code instead, roughly double the overall rate.

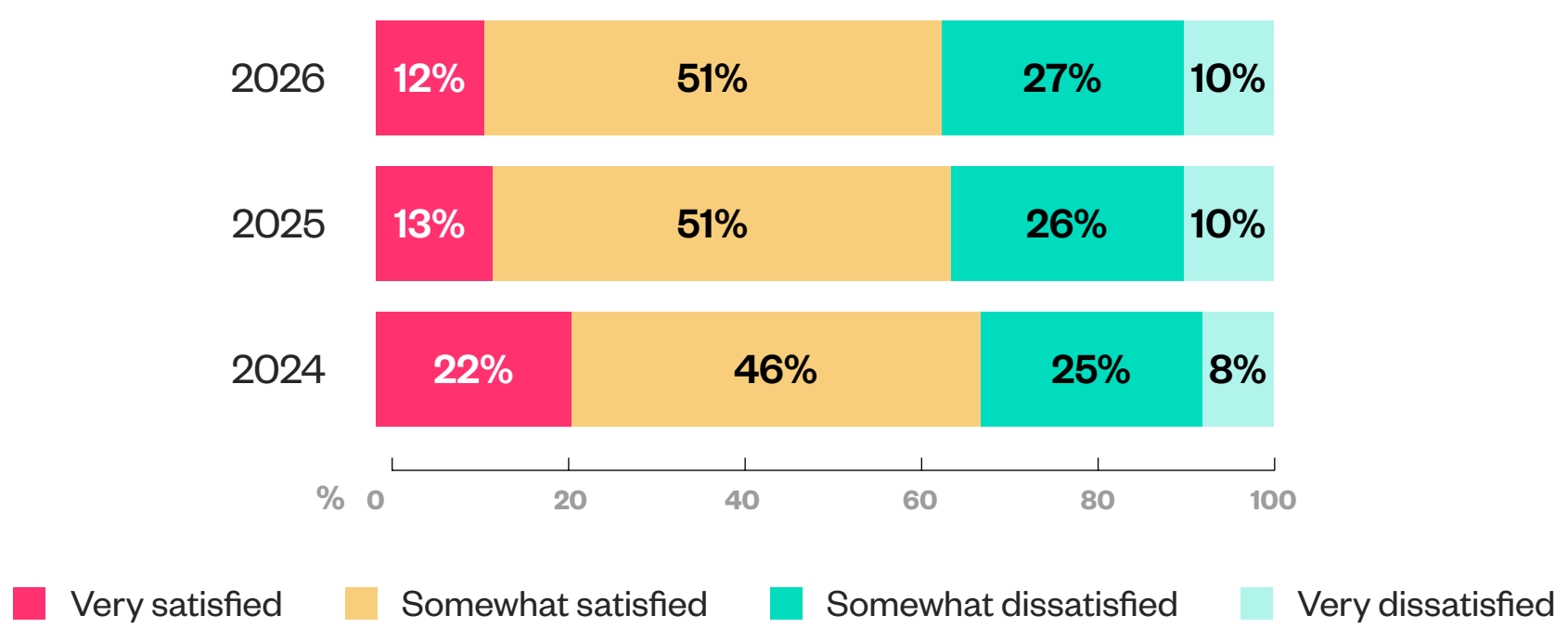
Despite the growing appetite to learn, satisfaction with actual L&D opportunities keeps sliding. It dropped from 68% in 2024 to 63% this year. When asked what would help, the top answer hasn't changed in years: more time for learning and training, cited by 60%, though even that figure is down from 64% in 2025 and 66% in 2024. Employer incentives to complete training (44%) and the ability to learn through AI tools already in use (31%) round out the top requests.

Larger organizations are more likely to want AI baked directly into their learning platforms, with 32% of companies with 500 or more engineers wanting AI agents added to existing L&D tools, compared to the overall average of 21%.

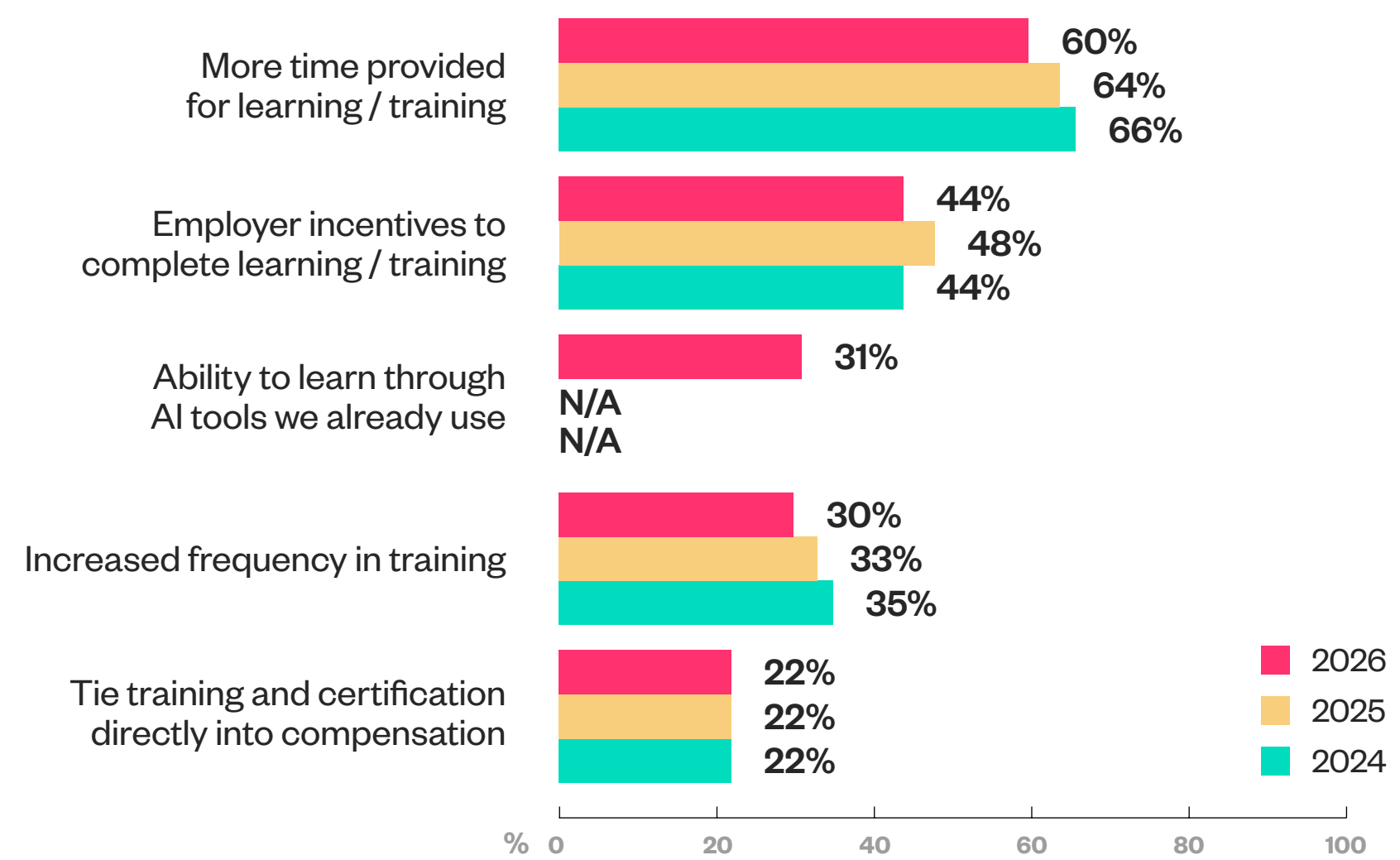
### What does this mean?

Leaders think L&D investment is mostly about retention. Engineers, who see it as a way to stay relevant as AI reshapes the job itself, disagree. That mismatch is likely why satisfaction keeps sliding even as priority scores tick up.

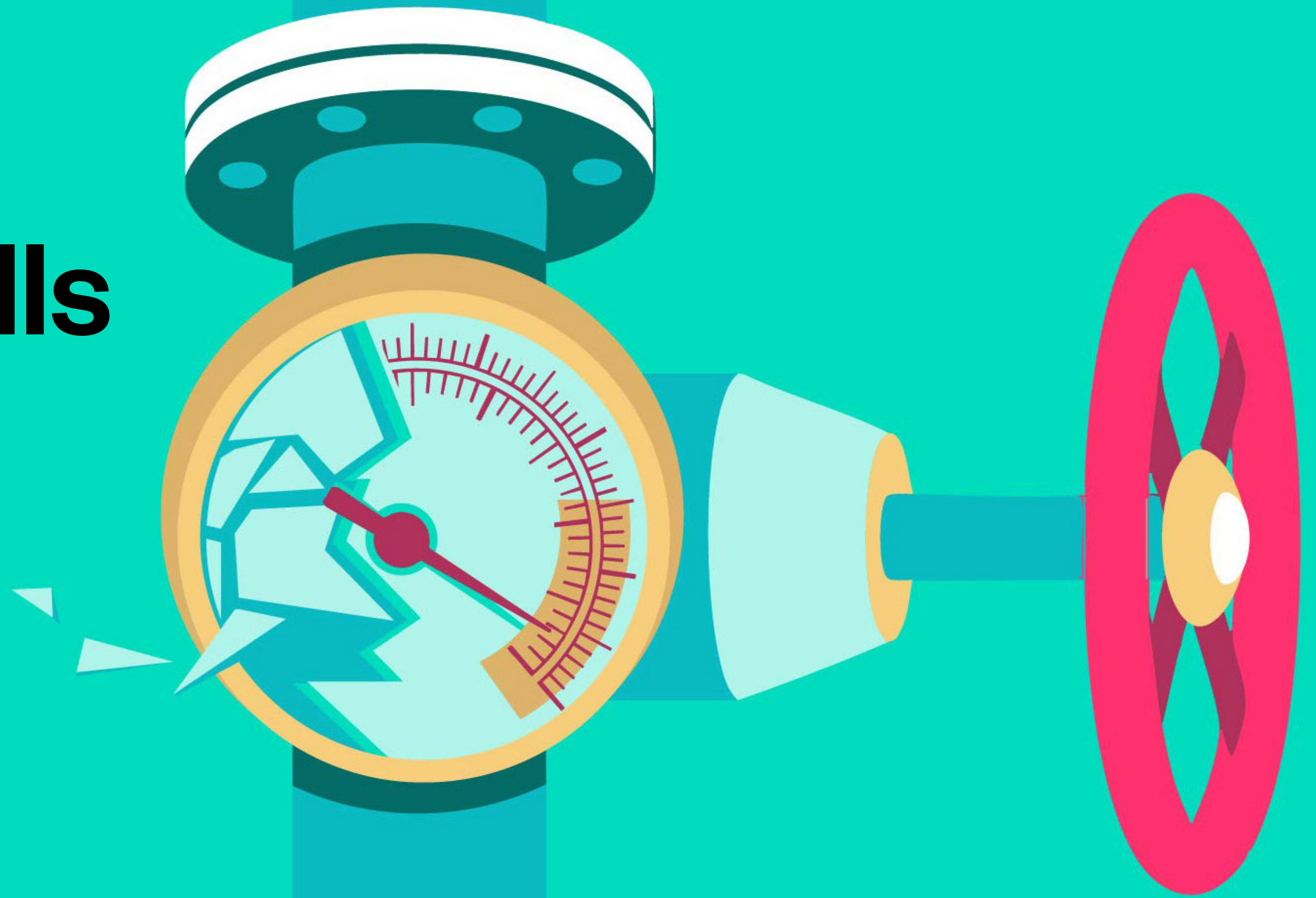
### How satisfied are you with the current learning and upskilling opportunities at your organization?



### What would improve learning and upskilling at your current organization?



**AI is reshaping  
engineering skills**

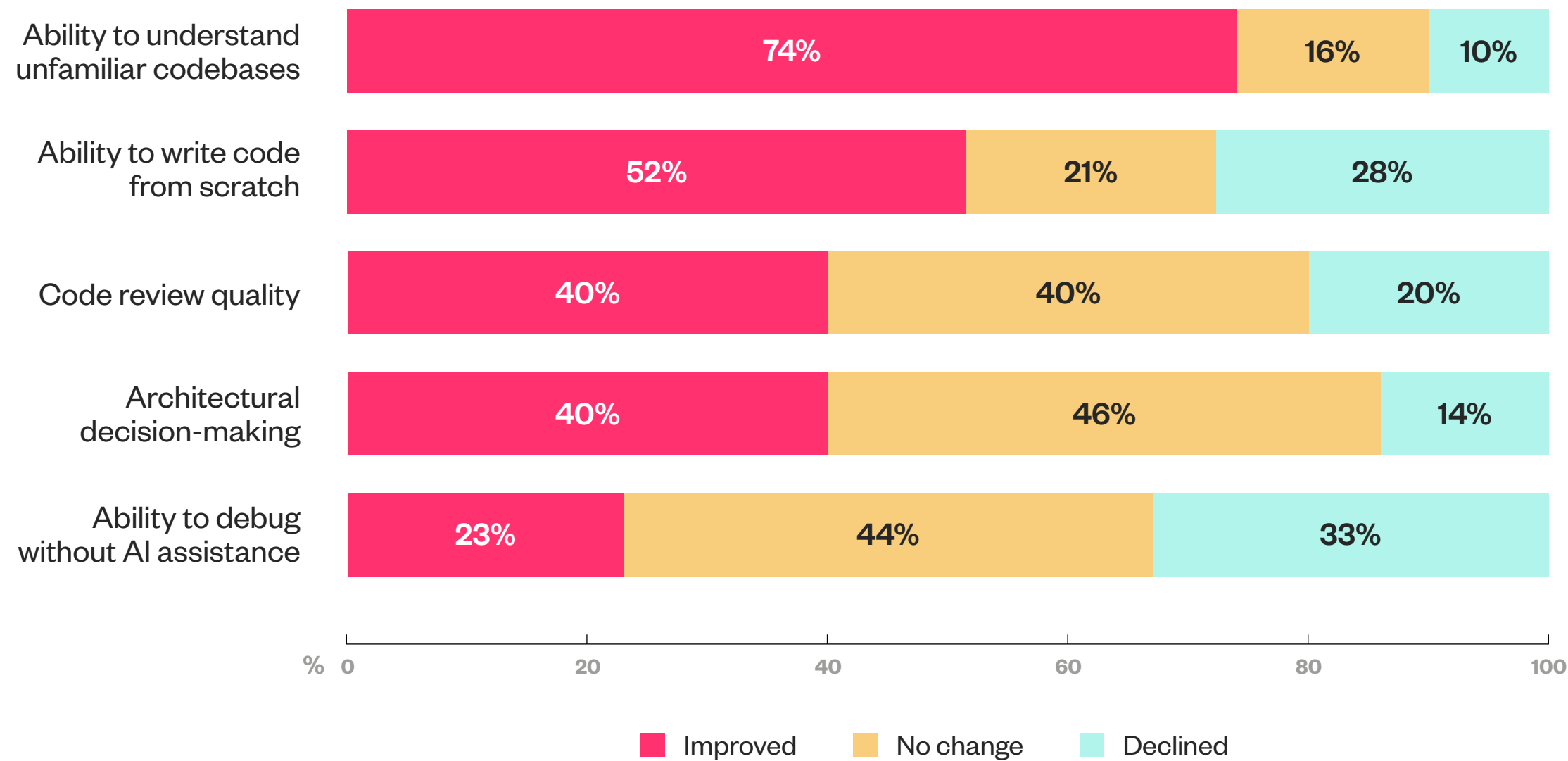


This year's survey captured a clear read on how AI adoption is reshaping engineering skill sets, not just workflows. The picture is more nuanced than either the hype or the doom narrative would suggest.

On the positive side, 74% of respondents say their team's ability to understand unfamiliar codebases has improved over the past 12 months because of AI. Meanwhile, 52% say the same about writing code from scratch and 40% report improvements in both code review quality and architectural decision-making. At smaller organizations, with fewer than 100 employees, that code review improvement figure jumps to 52%.

The one skill bucking the trend is debugging. Only 23% of respondents say their team's ability to debug without AI assistance has improved, while 33% say it's declined. That's a meaningful signal for anyone worried about atrophy. While teams see a net benefit to most developer competencies, they are losing the ability to debug without the assistance of AI tools.

How has AI adoption affected the skills and competencies of your engineering team over the past 12 months?

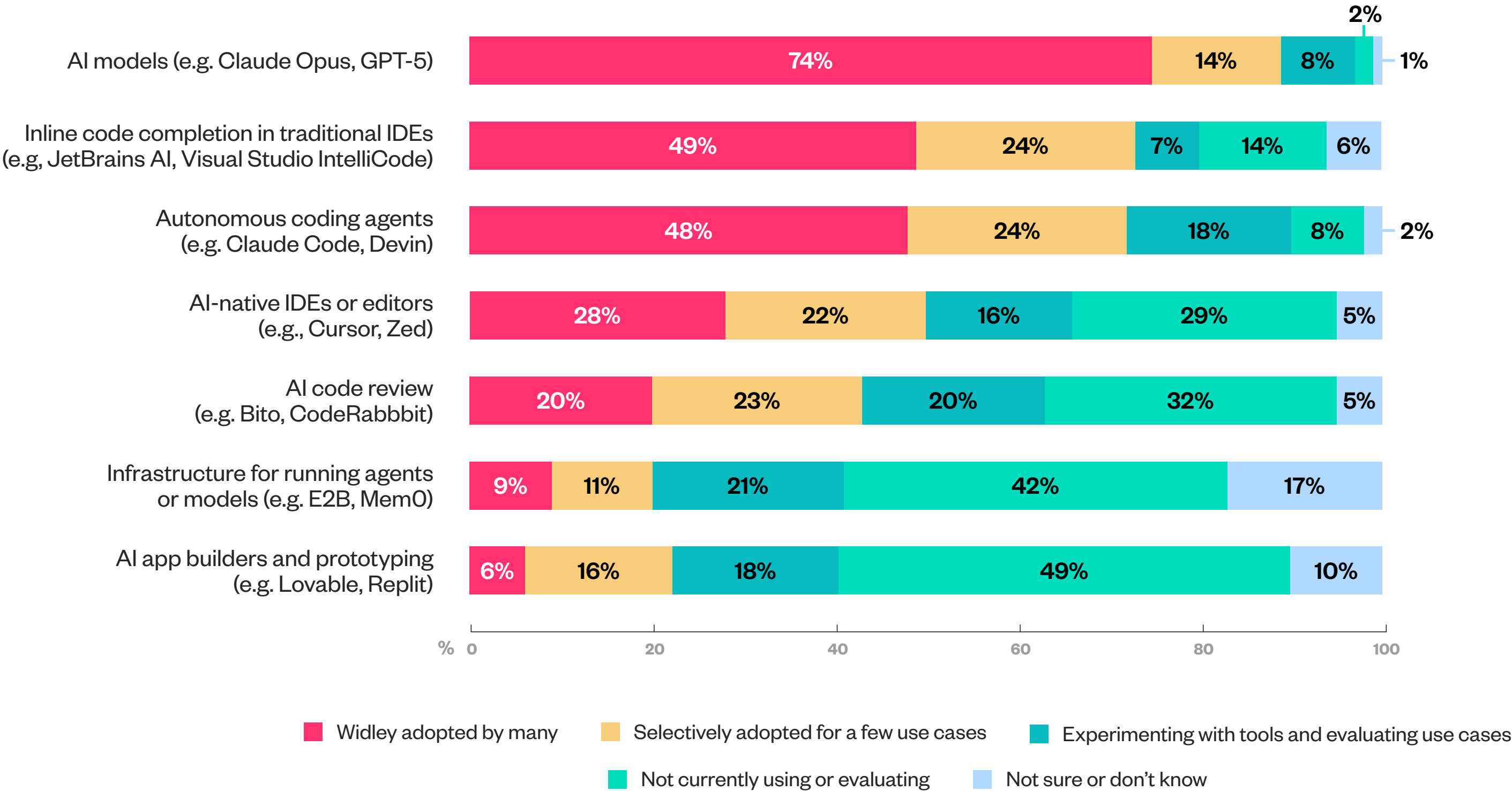


Tool adoption itself is now near-universal for the basics, with 74% of organizations having widely adopted AI models like Claude or GPT-5. Autonomous coding agents aren't far behind at 48% widely adopted. AI-native IDEs (28%) and AI code review tools (20%) are earlier in their adoption curve, though 85% of the people using AI code review are also already using autonomous coding agents, suggesting these tools tend to get adopted together rather than in isolation.

Adoption of inline code completion scales with company size: 31% at organizations with under 100 employees, 48% at mid-sized ones, and 58% at organizations with more than 1,000 employees.

Looking forward, engineers expect AI's biggest impact to land on automated code generation (48%) and code refactoring or migration (39%), with code reviews, documentation, and bug fixing each cited by 30%. That last one is worth a pause. Fixing bugs didn't even crack the top nine areas in last year's survey. This year it's tied for third.

To what extent has your organization adopted AI-powered developer tools or models in the following categories?



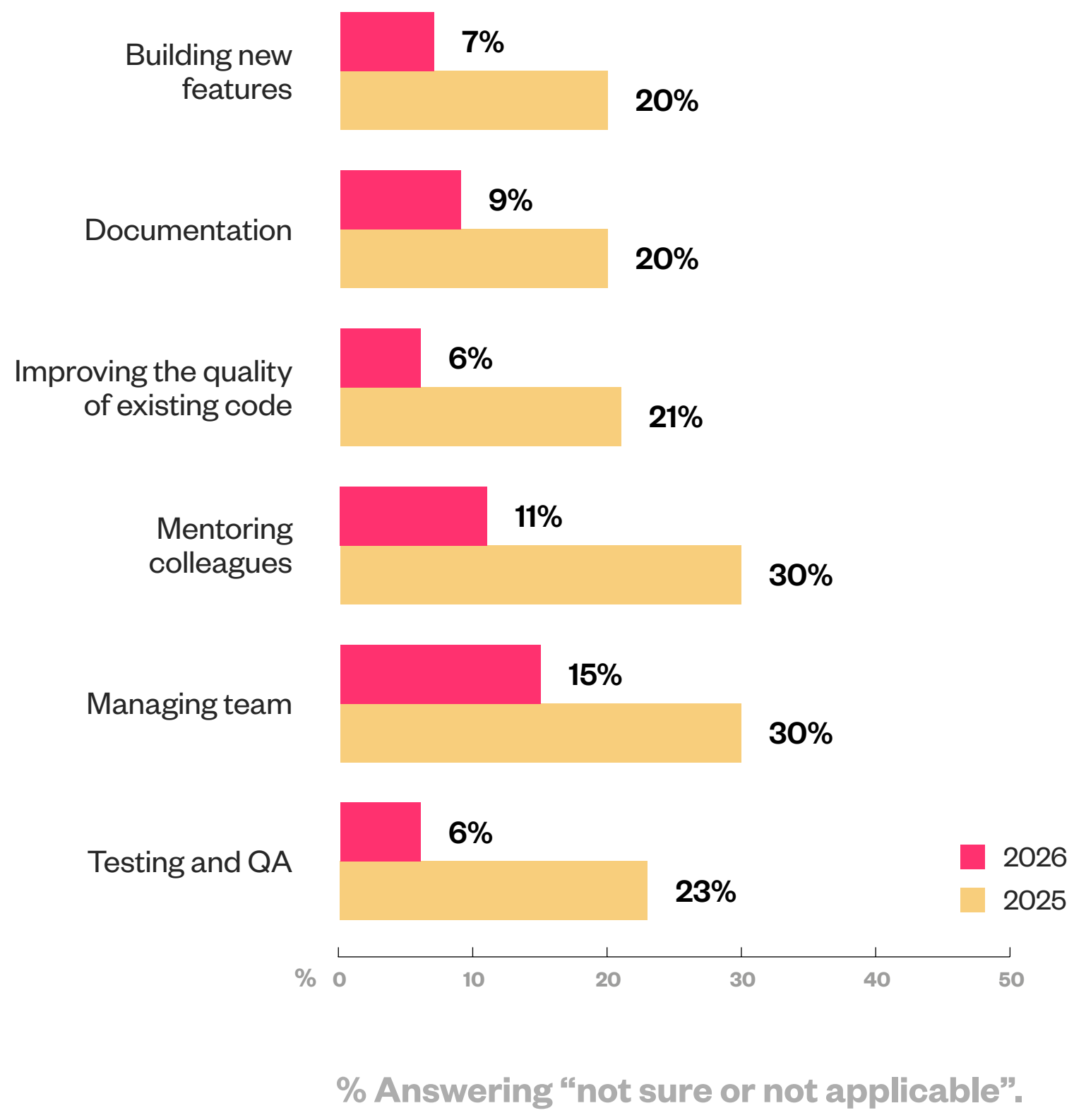
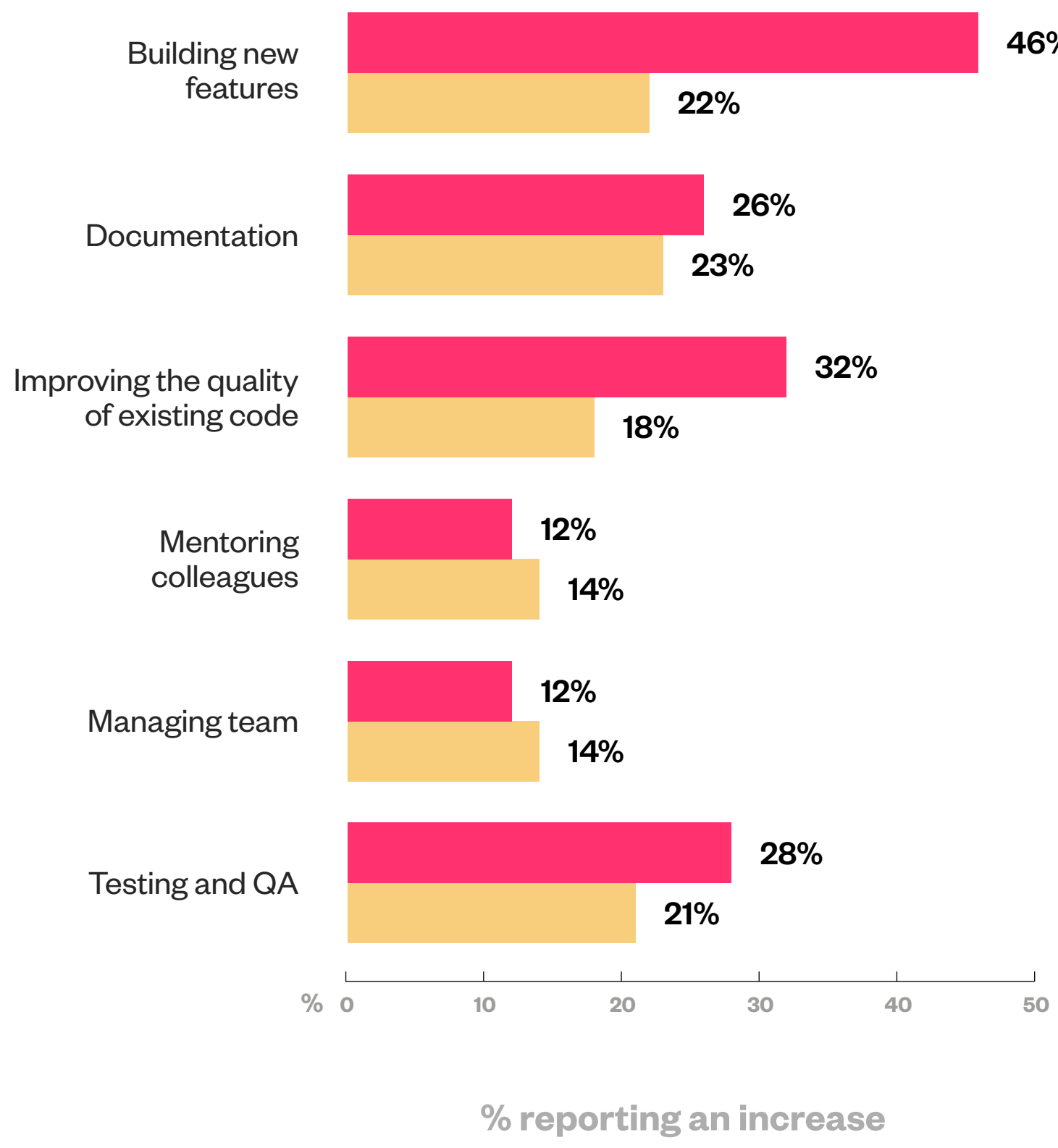
The clearest evidence of AI actually changing daily work shows up in how developers spend their time. A little under half (46%) of respondents say they're now spending more time each week building new features as a direct result of AI-assisted coding, more than double the 22% who said so in 2025.

Time spent improving the quality of existing code has also risen, from 18% to 32%. Uncertainty about the question itself has dropped too, with the share of respondents answering “not sure or not applicable” falling from 24% last year to just 9% this year. People finally seem to have a handle on where the hours are actually going.

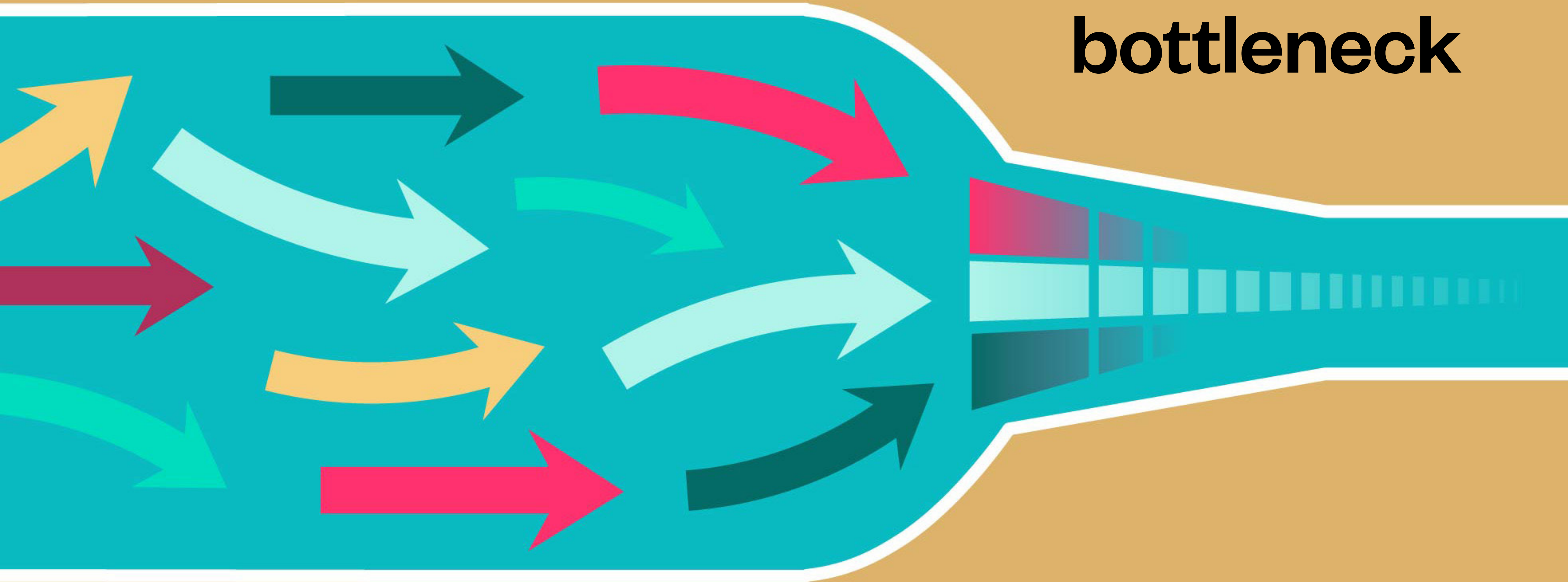
What does this mean?

AI is upgrading the skills tied to reading and building code, but not the ones tied to fixing it when it breaks. That's a bet worth watching as teams trade debugging muscle for speed. They won't know the real cost until AI-generated code fails somewhere it can't explain itself.

As a result of using AI-assisted coding tools, how has the average developer's weekly time allocation changed for the following activities?



**Review is  
the new  
bottleneck**

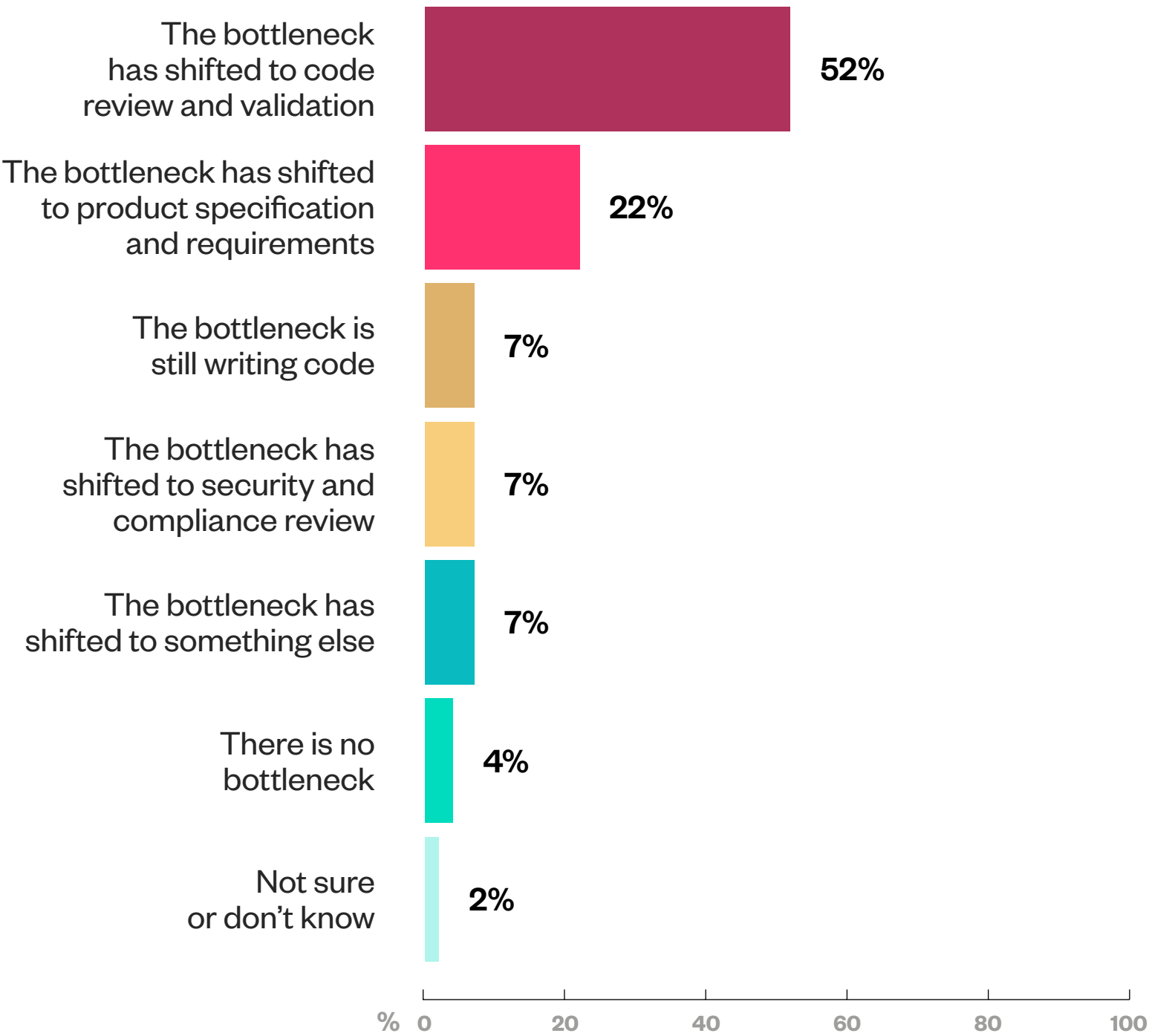


Perhaps the most consequential finding in this year's report is where engineering teams say AI has caused a bottleneck in the software delivery process. Over half (52%) say the biggest bottleneck is now code review and validation. Another 22% point to product specification and requirements. Only 7% think the bottleneck is writing code itself.

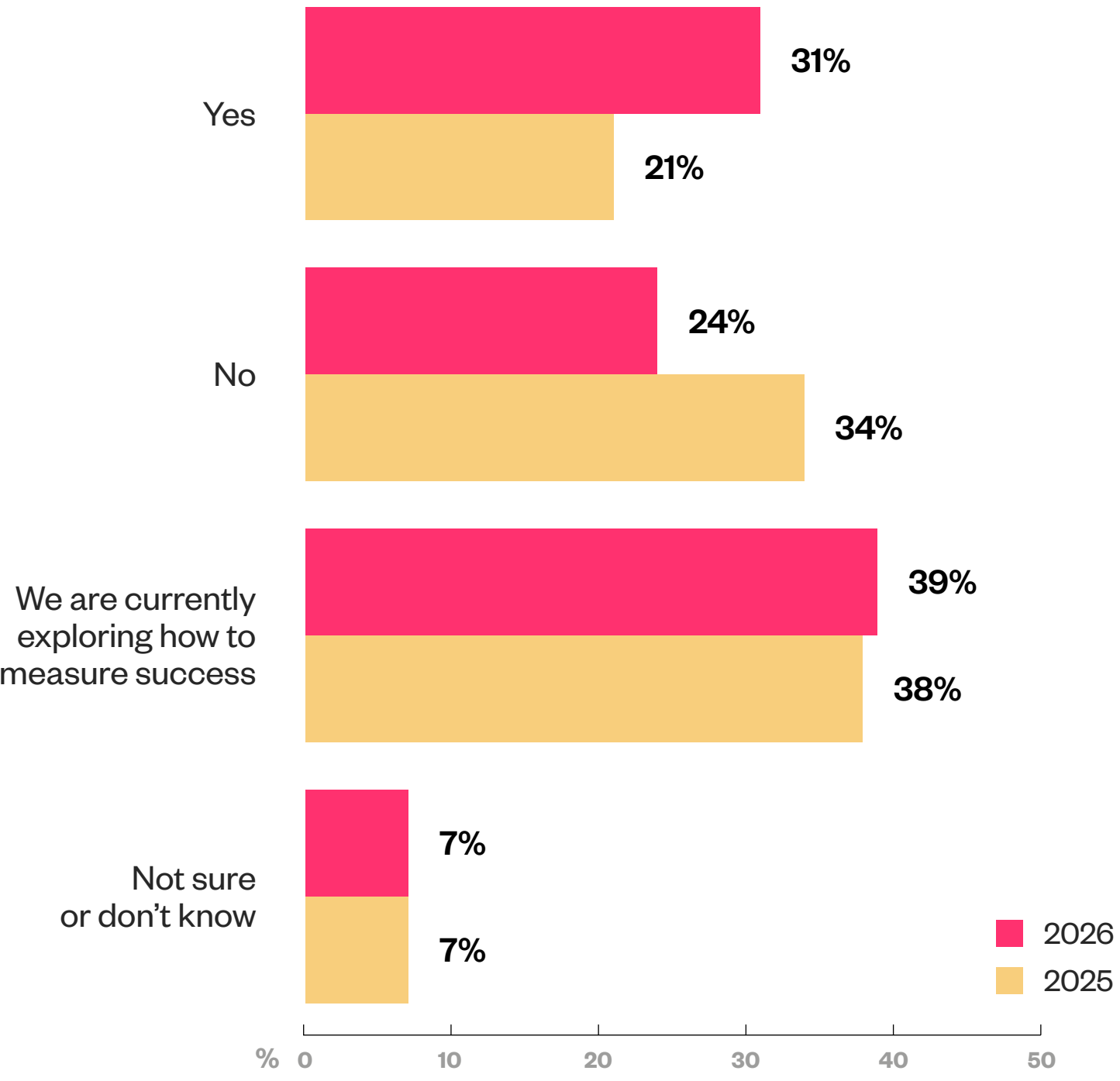
That's a genuine inversion of how software delivery has worked for most of the industry's history. Writing code used to be the expensive, slow part. Now, with AI generating a growing share of that code, the expensive element is deciding whether what got generated is actually correct, safe, and worth shipping.

Measurement, unfortunately, is lagging behind that shift. Only 31% of organizations currently measure the impact of AI-powered developer tools, though that's up from 21% in 2025. Another 39% say they're exploring how to measure success, which suggests most of the industry is still figuring out what "impact" even means in this context.

Due to AI, what is the biggest bottleneck in your team's software delivery process



Does your organization currently measure the impact of AI-powered developer tools or LLMs used within engineering teams?



The gap is worst at small organizations: 48% of companies with fewer than 20 engineers have no plans to measure AI tool impact at all.

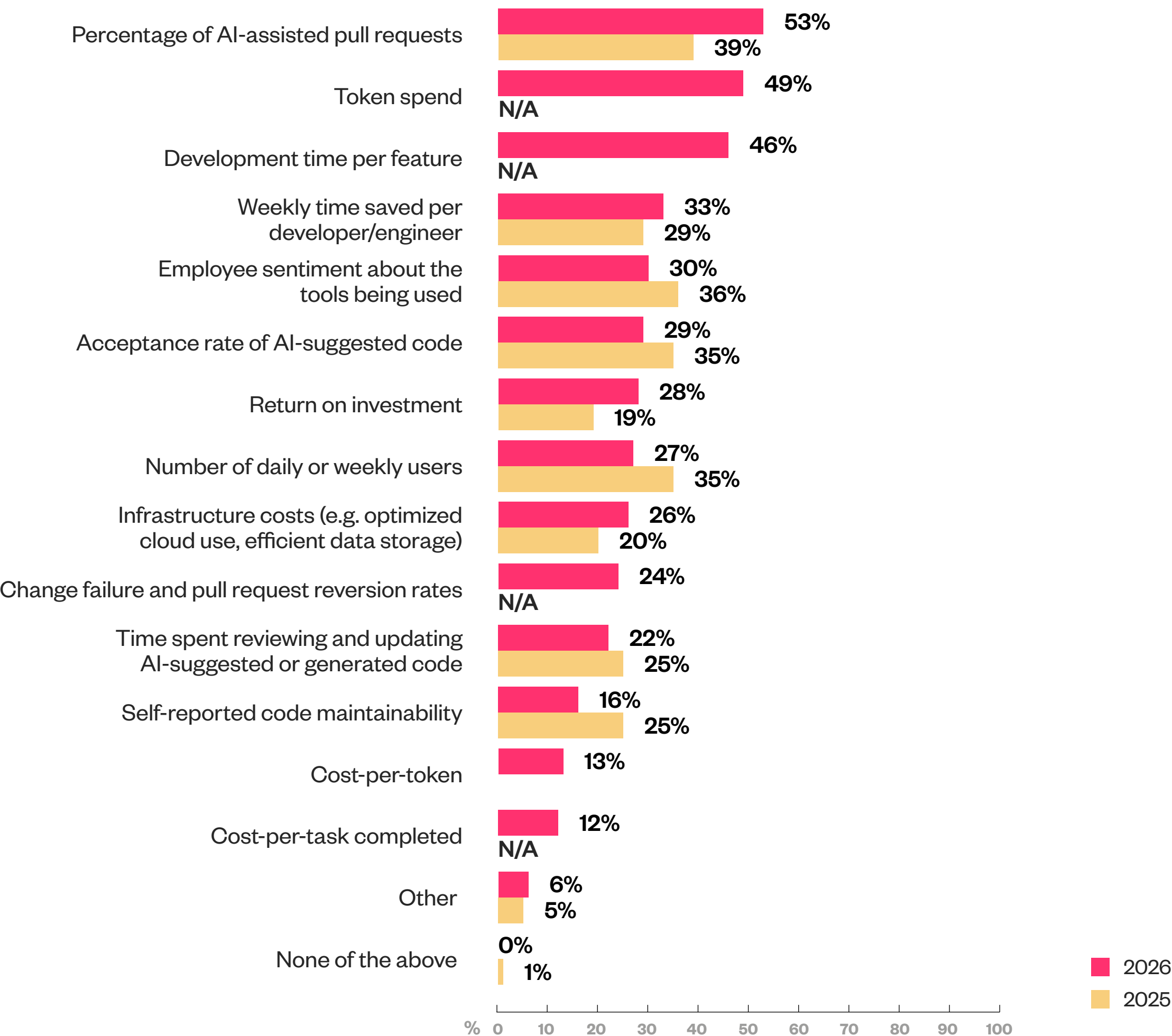
Among the minority who are measuring it, the most common metric is the percentage of AI-assisted pull requests, tracked by 53% of that group, up from 39% last year. Token spend (49%) and development time per feature (46%) follow closely, both new additions to the survey this year.

Weekly time saved per developer (33%) and employee sentiment about the tools (30%) round out the list, though sentiment tracking has actually declined slightly from 36% in 2025, and acceptance rate of AI-suggested code has dropped from 35% to 29%.

### What does this mean?

This section paints a picture of an industry moving fast on adoption and slowly on accountability. Teams know where the friction has moved, but fewer of them have built the measurement infrastructure to manage it.

### If the impact is being measured, what metrics does your organization track that can gauge the impact of AI-powered developer tools or LLMs used within engineering teams?



# Conclusion

Pull all five threads together and a consistent story emerges. Engineering leaders are reporting on performance more because they're required to than because they trust what they're reporting.

The metrics available to them, old standbys like lines of code and newer ones like pull request counts, are losing credibility fast, especially now that AI can inflate almost any activity-based number without adding real value.

Even as engineers scramble to build AI-era skills that didn't exist as job requirements two years ago, investing in learning and development is not a top priority at most organizations. What's more, the entire shape of software delivery has shifted, with code review now the clear bottleneck, while measurement of that shift is still mostly aspirational.

None of this means the industry is in crisis. If anything, the honesty in this year's responses, the willingness to say "our metrics don't reflect real value" or "I could tell performance had declined without the numbers," suggests engineering leaders are clear-eyed about their own measurement gaps.

The challenge for 2027 isn't going to be finding new things to measure. It's going to be building the discipline, and the tooling, to measure the things that actually matter, particularly around AI-assisted work, before the gap between what teams do and what they can prove gets any wider.

# Methodology

The survey was conducted between **17 August** and **14 September, 2026**. After performing data quality checks, this analysis is based on **384 respondents**, of which **300 completed the entire survey**. Due to rounding, not all charts add up to 100%. Margin of error: +/- 5.0% at a 95% confidence level.

