



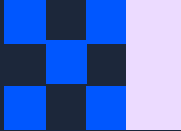
2026 | Q3 Pulse Report

State of Software Delivery



// Contents

Executive summary	3
Section 1: The velocity gap keeps widening	5
The busiest workflows now run 10x the median	
The divergence starts well before the top 5%	
The same pattern appears at the organization level	
The gap is widening at the contributor level, too	
Section 2: Other performance indicators are improving	10
Builds are passing more often	
Workflows are finishing faster	
Teams are doing more validation per run	
Recovery is improving on main	
Section 3: Most teams are adding work without increasing velocity	16
Developers have more branches in flight	
Teams are supporting more projects and workflows	
Feature-branch failures are taking longer to recover	
The typical workflow isn't moving any faster	
Field note: Development spills outside the workweek	
Section 4: Top teams are adding workstreams and increasing delivery velocity	23
Development activity is expanding rapidly	
More of that work is reaching main	
The gains grow toward the top	
Section 5: Merge efficiency still separates the teams pulling ahead	26
Top performers keep their merge efficiency ratio (MER) low	
Other habits of top performers	
Section 6: Where to focus	30
Section 7: Looking ahead to Q4 and 2027	34
Appendix 1: Additional findings	36
Appendix 2: Q3 metrics at a glance	40
Methodology and contributors	41



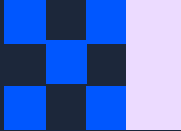
// Executive summary

The [2026 State of Software Delivery](#) documented the biggest increase in development activity we've ever measured. Daily workflow volume grew 59% year over year as AI accelerated code generation. But the gains weren't evenly distributed. The highest-throughput teams were shipping more code, while the median team was treading water or losing ground.

Since Q1, researchers and engineering leaders across the industry have increasingly converged on the same conclusion: as AI accelerates code generation, validation is becoming a critical bottleneck in software delivery. Our [Q2 report](#) showed the gap between top-performing and typical teams widening further. Now, Q3 gives us a clearer picture of where teams are making progress and where they're still falling behind.

Here are the key findings from Q3:

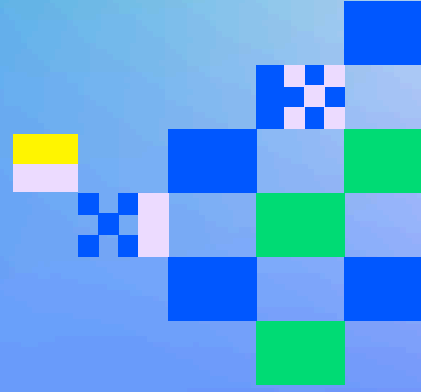
- **The velocity gap keeps widening.** The top 5% of main-branch workflows now run 10.4x more often than the median, up from 9.1x in Q2. The fastest teams continue to pull away.
- **CI performance is improving, but the benefits aren't translating into faster delivery.** Workflows are finishing faster, passing more often, and recovering from failures on main more quickly, even as teams run more validation per workflow. Yet throughput remains flat for the typical team, raising questions about what else is holding delivery back.



- **Most teams are adding workstreams without shipping any faster.** The number of feature-branch workflows active on a typical day has grown 40% since Q1, while median throughput on main has barely moved. More work is in progress, but little of the additional activity is making it to production.
- **Top teams are expanding R&D *and* shipping more.** Alongside adding new workstreams, top teams have increased their total main-branch throughput by 57% since Q1, and each workflow on main runs 31% more often.
- **Merge efficiency remains a major differentiator.** The typical team runs nearly four feature-branch workflows for every run on main, compared with 2.6 for high performers and roughly 1.5 for the Top 20 organizations. Teams that get changes through validation in fewer cycles also tend to ship more.

The patterns we identified in Q1 have become more pronounced. CI is getting faster and more reliable, but most teams are still taking on more work without increasing delivery velocity. The fastest teams are expanding their scope and accelerating delivery at the same time.

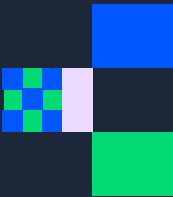
As organizations plan for 2027, the challenge is to turn growing development capacity into software that reaches customers. Generating more code is only part of the equation. The teams getting the most from AI are also finding ways to validate, integrate, and ship it more efficiently.



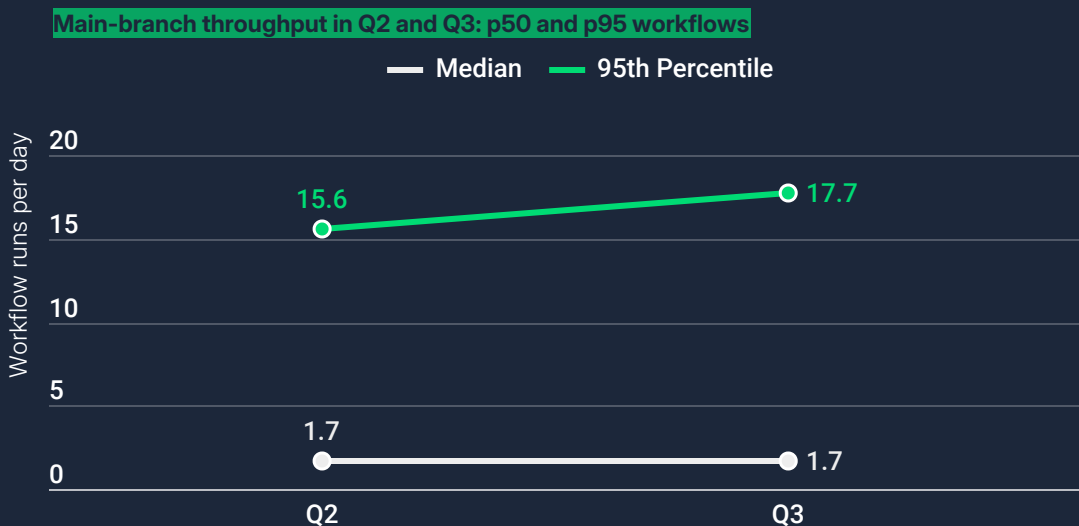
1

The velocity gap is widening

The busiest workflows now run 10x the median



In Q1, we found that AI was increasing development activity across the industry, but the gains in delivery velocity were uneven. By Q2, the top 5% of main-branch workflows were running 9 times as often as the median. In Q3, the gap widened to more than 10x.



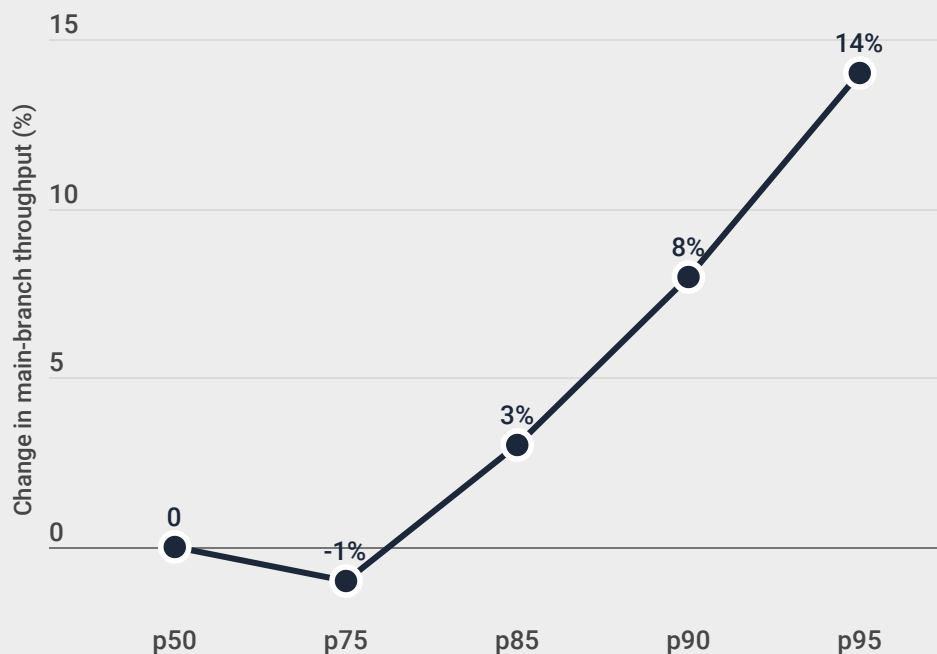
The median main-branch workflow still runs about 1.7 times a day, unchanged since Q2. At the 95th percentile, throughput increased to nearly 18 runs a day.

Main-branch throughput is a useful proxy for how often teams move validated changes toward production. A 10x difference gives the highest-throughput teams many more opportunities to ship, learn from customers, and iterate. Teams holding their current pace are therefore losing ground relative to teams whose delivery systems continue to accelerate.

The divergence starts well before the top 5%

The growing gap between the median and p95 isn't limited to the very top. Since Q2, main-branch throughput has stayed flat for the bottom 75% of workflows. Above that, gains increase steadily: 3% at p85, 8% at p90, and 14% at p95. The top quarter is pulling away from the rest, with the biggest gains concentrated at the top.

Change in main throughput since Q2 by workflow percentile



The pattern suggests that teams with higher delivery velocity are building on their existing advantage. They're increasing throughput while most teams stand still, and the gains get larger the further ahead a team already is.

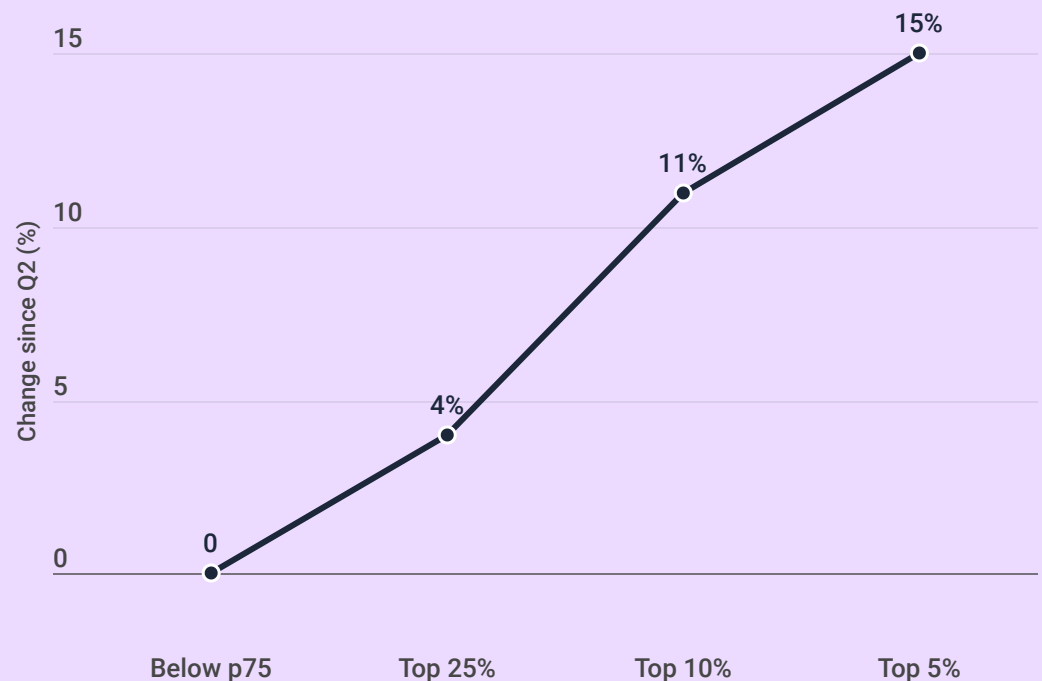


The same pattern appears at the organization level

Workflow-level data tells us where throughput is increasing. Grouping organizations by total main-branch activity across all of their projects shows who is driving the increase.

The top 5% of organizations increased throughput on each main-branch workflow across all of their projects by 15% since Q2. The top 10% increased 11%, and the top 25% increased 4%. Below p75, it didn't move.

Main-branch throughput growth by organization percentile

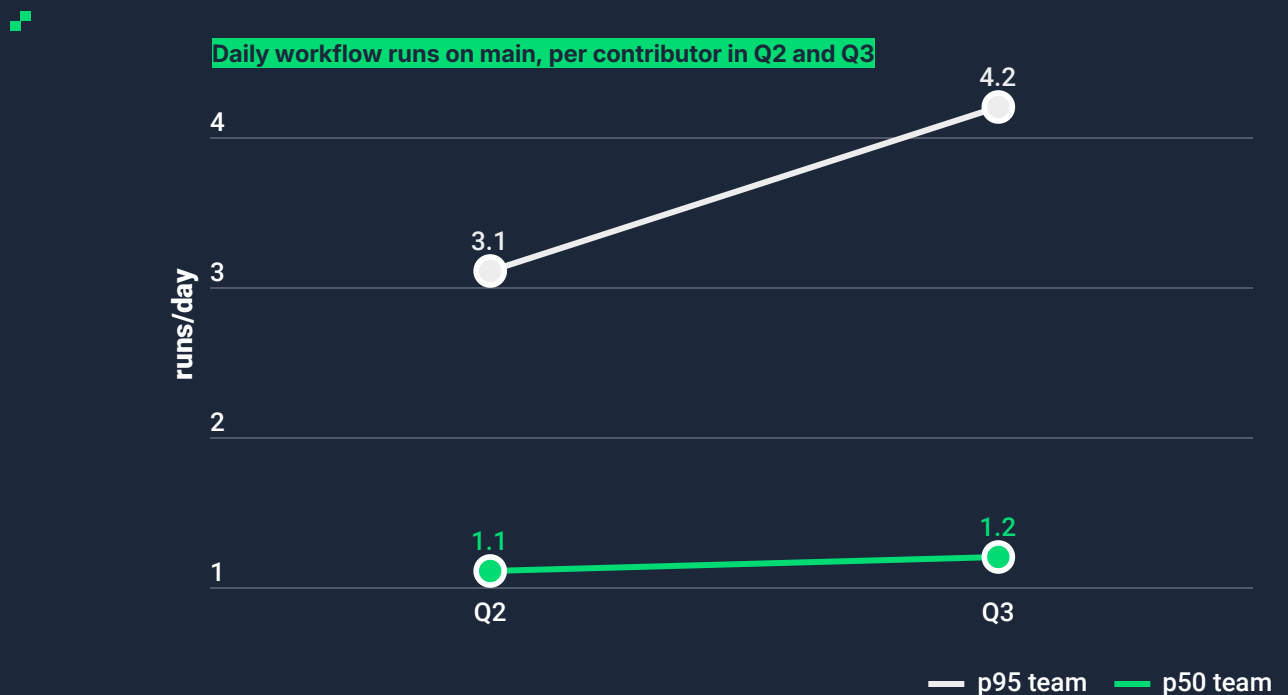


Teams moving the most work through main have continued to increase the pace of each workflow. The same advantage is becoming more pronounced across engineering organizations.

The gap is widening at the contributor level, too

Larger organizations naturally have more people and more opportunities to run CI, so total activity alone can exaggerate differences between teams. But the gap remains when we normalize main-branch activity by contributors, and it widened again in Q3.

The median team now gets 1.2 main-branch workflows per contributor per day, up from 1.1 in Q2. High performers, the teams above the 95th percentile of main-branch throughput, average 4.2, up from 3.1. A contributor on a high-performing team now gets about 3.5 times as much work onto main as a contributor on a median team, up from 2.8 times in Q2.

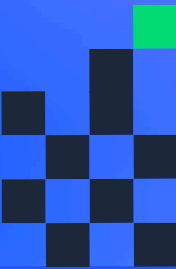


The same divergence appears at the workflow, organization, and contributor levels. Teams at the top are creating more opportunities to ship, learn, and iterate, while the median has made little progress on delivery velocity.

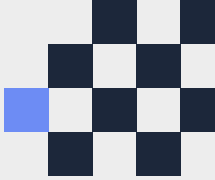
Most teams still aren't seeing faster delivery from AI-assisted development. But the rest of the Q3 data tells a more complicated story.

2

Other performance indicators are improving



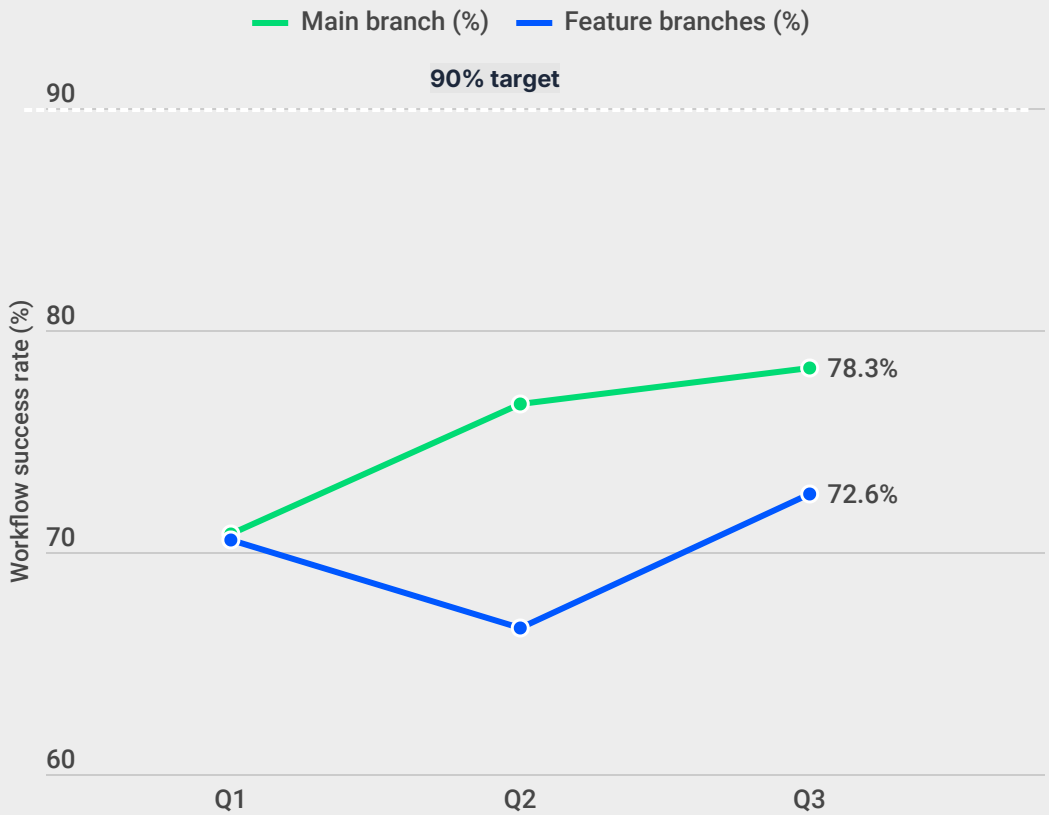
In Q1, main-branch success rates fell to a five-year low while recovery times surged. With more builds failing and taking longer to fix, teams were in a difficult position: spend longer debugging failures or reduce validation to keep delivery moving. By Q3, CI performance is improving on both fronts. Main-branch success is rising, recovery times are falling, and the pressure creating the tradeoff is starting to ease.



Builds are passing more often

Main-branch success rose to 78.3% in Q3, up from 76.7% in Q2 and 70.8% in Q1. Feature branches improved even more sharply, climbing 6.1 percentage points since Q2 to 72.6%.

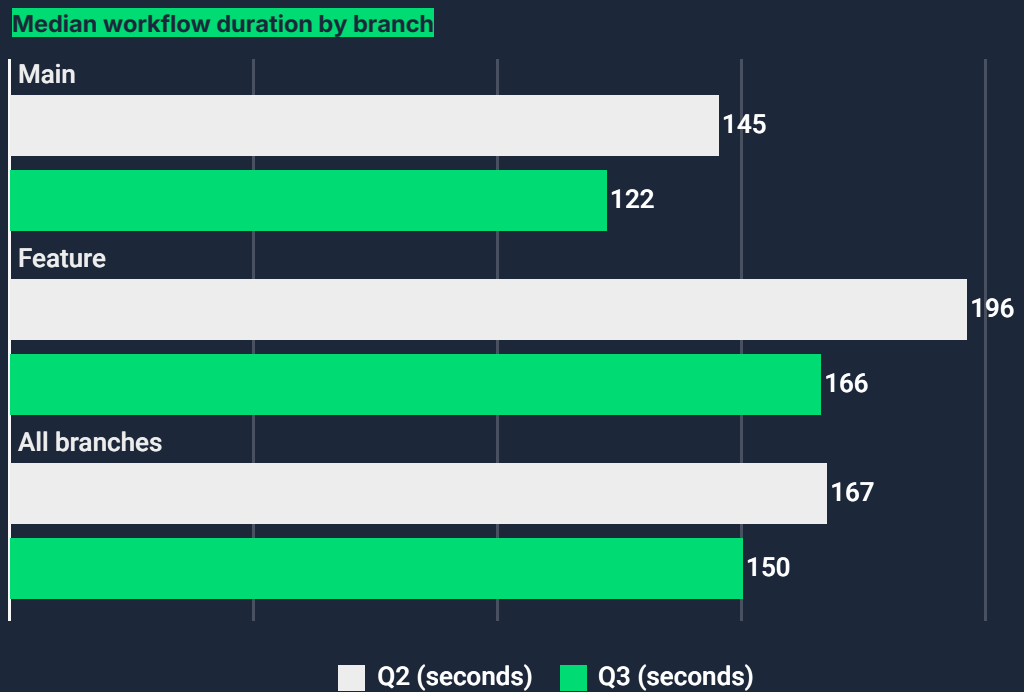
Workflow success rates by branch



Feature branches recovered from a dip in Q2 and now pass more often than they did at the start of the year. Main-branch workflows are still falling short of the 90% success benchmark, but the deterioration we saw earlier in the year across both branches has corrected.

Workflows are finishing faster

Success rates tell us more code is making it through CI. The workflows themselves are also taking less time to run. The typical main-branch workflow now finishes in 2 minutes 2 seconds, 23 seconds faster than in Q2, while feature-branch duration fell by 30 seconds to 2 minutes 46 seconds.

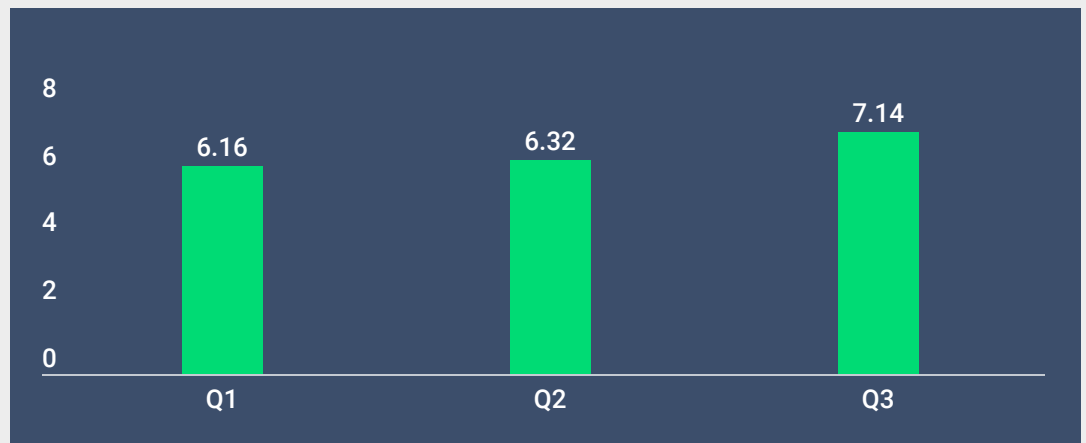


The improvement extends into the slowest workflows as well. At the 95th percentile, duration fell by about 3 minutes across all branches, reducing some of the longest waits developers encounter in the feedback loop.

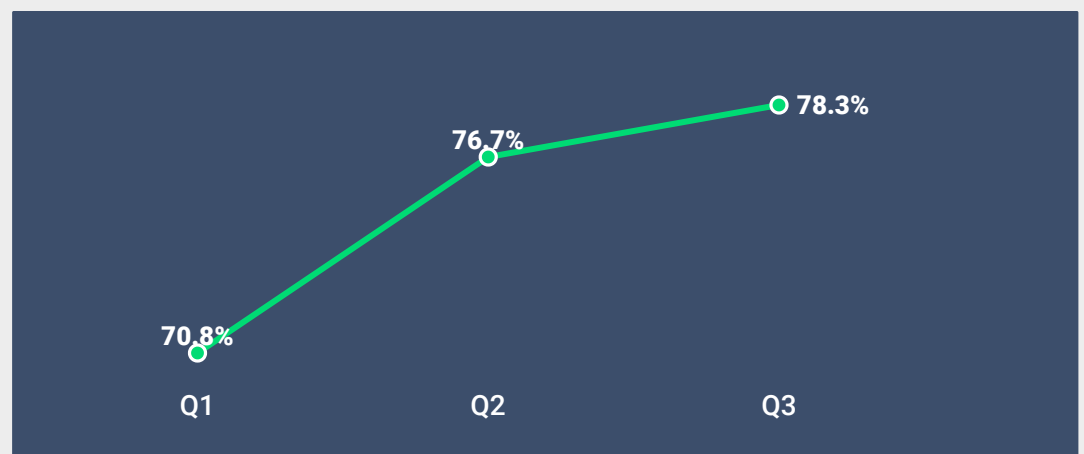
Teams are doing more validation per run

Faster workflows and higher success rates would be less meaningful if teams were getting there by running lighter workflows or removing validation steps that might otherwise catch a failure. Instead, the average main-branch workflow now runs 7.1 jobs, up 15% from 6.2 in Q1.

Jobs per main-branch workflow



Main-branch success rate

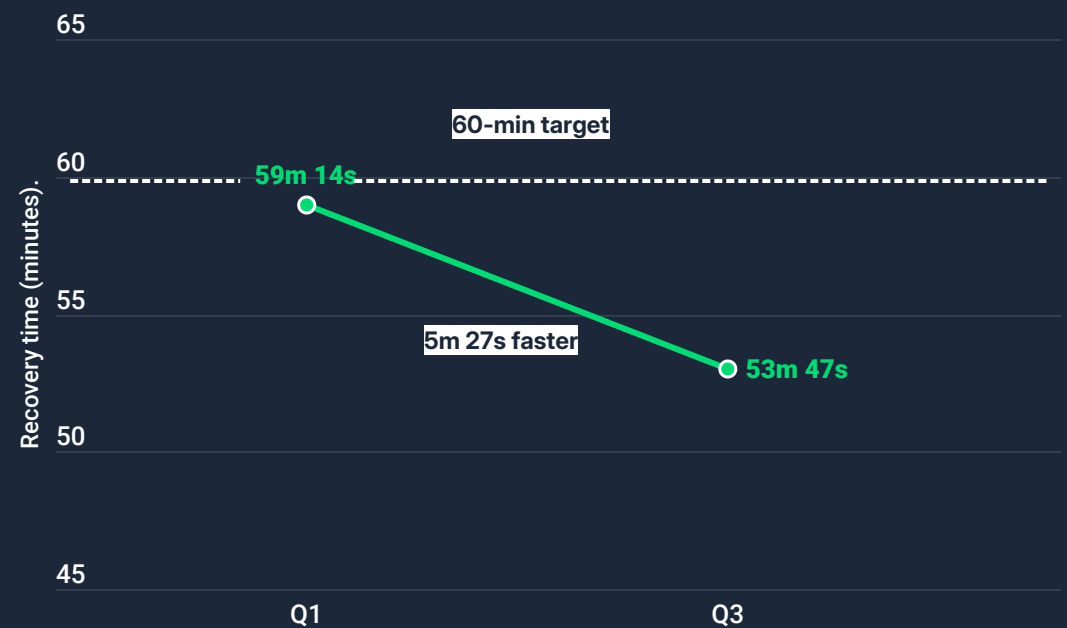


Teams haven't had to choose between debugging longer and validating less. Builds are passing more often and finishing faster, and each run is checking more than it did in Q1.

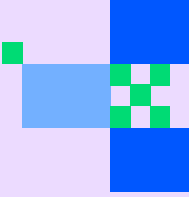
Recovery is improving on main

When a main-branch build does fail, teams are recovering faster. Median recovery time on main fell from about 59 minutes in Q1 to 54 minutes in Q3, putting it below the 60-minute benchmark for production branches.

Median MTTR on the main branch



Main is the path to customers, so every minute it stays broken is a minute changes can't ship. Faster recovery shortens that wait.



The improvements across CI metrics suggest teams are getting better at developing and delivering software with AI. One likely factor is that models are becoming more capable of producing usable code with fewer iterations. Another is that developers have had more time to learn how to work effectively with AI and where human expertise is most valuable.

A third factor is the CI platform itself. In 2026, CircleCI has continued evolving to deliver faster, more targeted feedback throughout the delivery cycle.

Improvements in models, developer effectiveness, and CI are helping address the delivery bottleneck we identified in Q1. Fewer failed runs, shorter feedback cycles, and faster recovery on main mean teams spend less time waiting on CI or untangling failures. The result is more time and attention available to build and ship software.

Yet only a small subset of teams is turning the additional capacity into higher delivery velocity. For the typical team, throughput remains flat. So the question becomes: where is that extra capacity going?

3

Most teams are adding work without increasing velocity

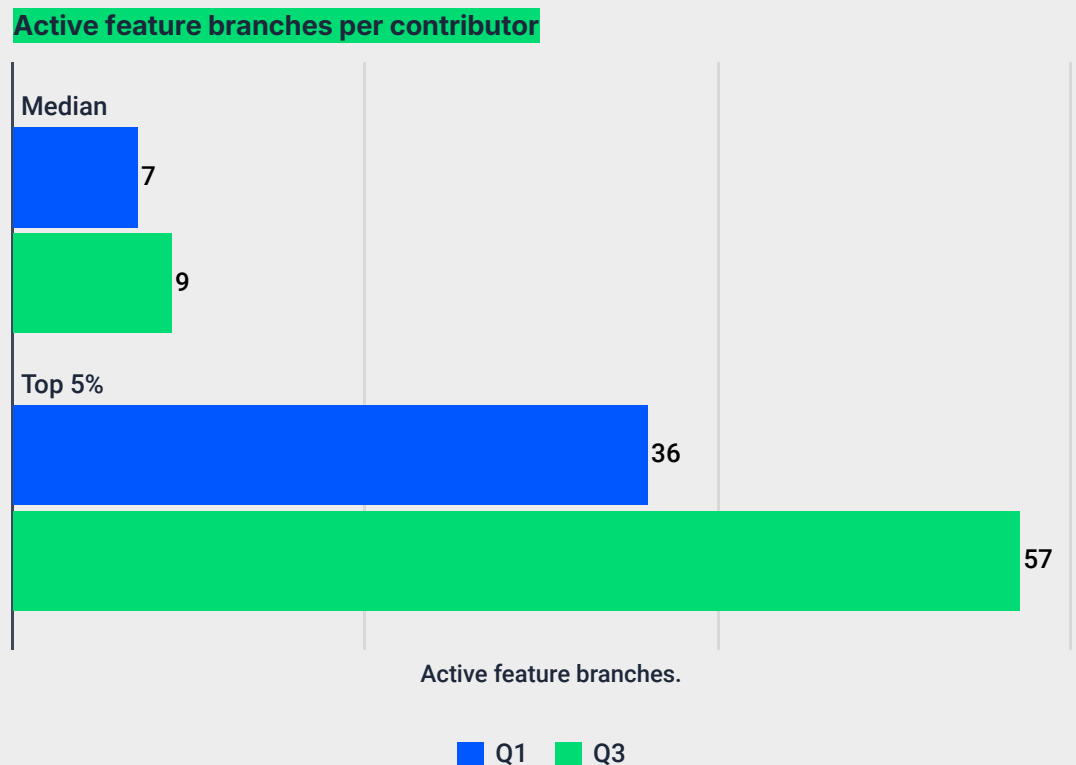
Improvements in delivery metrics are loosening the bottleneck we identified in Q1. So why aren't more teams shipping faster?

The data points to a familiar project management trap: teams are increasing work in progress instead of finishing work at a higher rate.

Developers have more branches in flight

The increase is visible first at the contributor level. Developers are working across more feature branches than they were at the start of the year, with the largest increase among contributors who already had the most concurrent work.

The typical contributor had 9 active feature branches in Q3, up from 7 in Q1. Among the most active 5% of contributors, the number increased from 36 to 57.

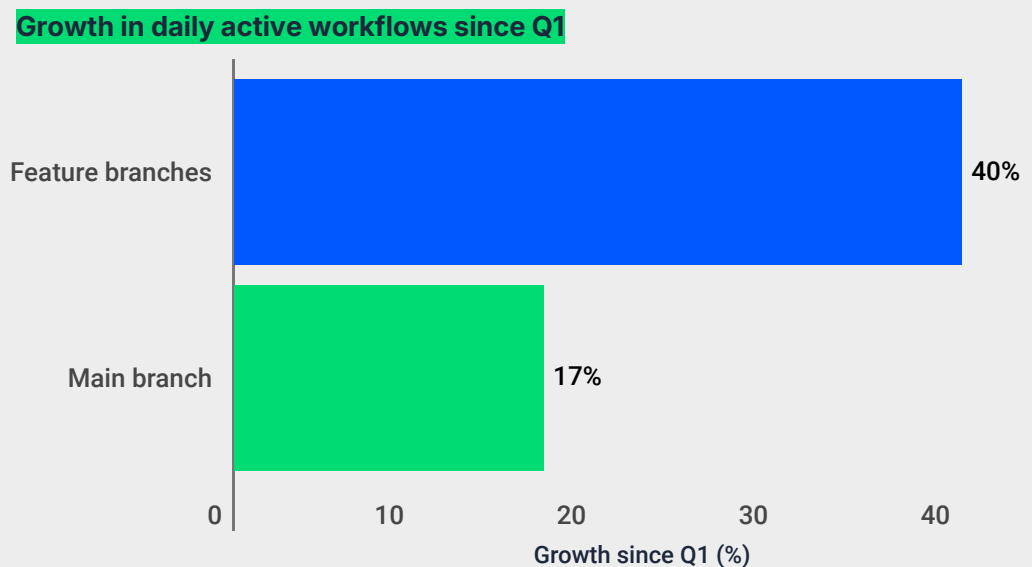


Each additional branch represents another stream of work that still needs review, validation, and attention before it can reach main. More parallel work can increase potential output, but it also leaves developers responsible for keeping more unfinished work moving at once.

Teams are supporting more projects and workflows

The increase in active work extends beyond individual developers. Organizations are spreading development across more projects and running CI across more workflows, with feature-branch activity growing especially quickly.

Since Q1, the number of workflows active on a typical day has grown 40% on feature branches and 17% on main. Teams in the middle of the distribution now support 16% more projects, with 9% more workflows per project on average.



Teams are taking on more work across more parts of the organization at once. Some of the additional work is reaching main, but the much faster growth on feature branches shows how quickly work in progress is accumulating before integration.

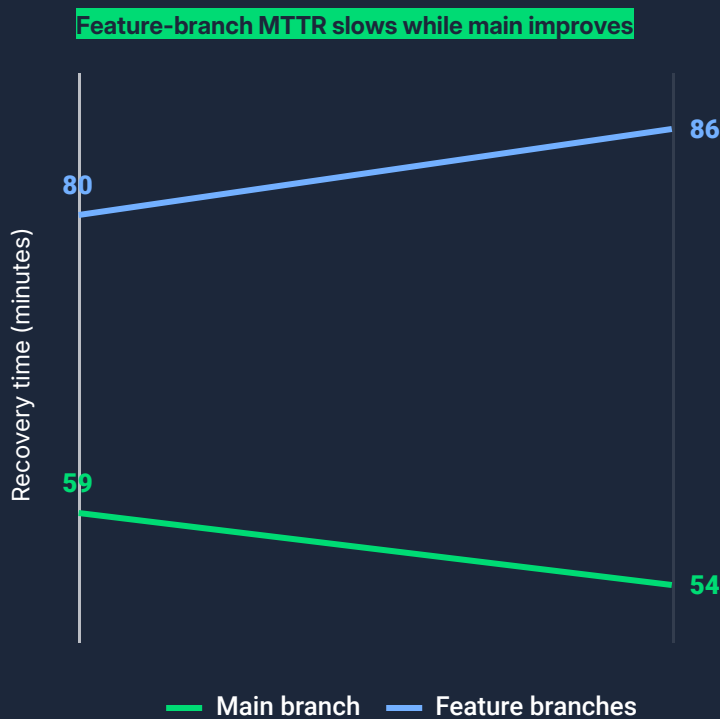
A broader set of active projects can create more opportunities for growth and experimentation. But every new workstream is another queue of work waiting for attention instead of reaching customers.



Feature-branch failures are taking longer to recover

More work in flight also means more failures competing for developer attention. Feature-branch workflows are passing more often than they did earlier in the year, but when a failure occurs, teams are taking longer to get back to green.

Median feature-branch recovery increased from 79 mins 43 secs in Q1 to 85 mins 59 secs in Q3, moving the opposite direction from recoveries on main.

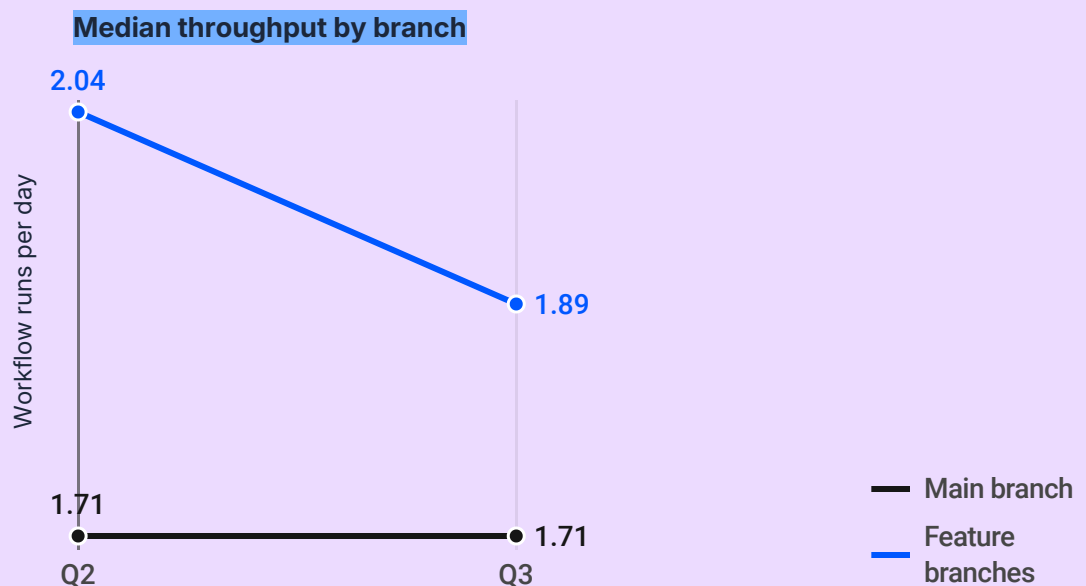


Developers are likely prioritizing the failures that matter most. A broken main-branch pipeline can block an organization's ability to deliver value, and teams are resolving those failures faster. A failed feature branch can wait while a developer turns to another active branch, project, or task.

The typical workflow isn't moving any faster

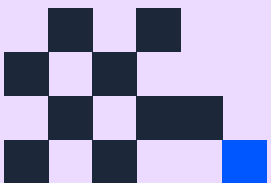
If teams were converting the increase in development activity into faster delivery, we would expect each workstream to move faster as well. For most teams, that isn't happening.

Median main-branch throughput remains 1.71 runs a day, unchanged since Q2. Median feature-branch throughput fell from 2.04 to 1.89 runs a day. At the organization level, teams below p75 have also seen little change in main-branch throughput since Q1.



Total main-branch activity has increased because teams have more workstreams running at once. But within each workstream, delivery velocity has barely moved, and in some cases declined.

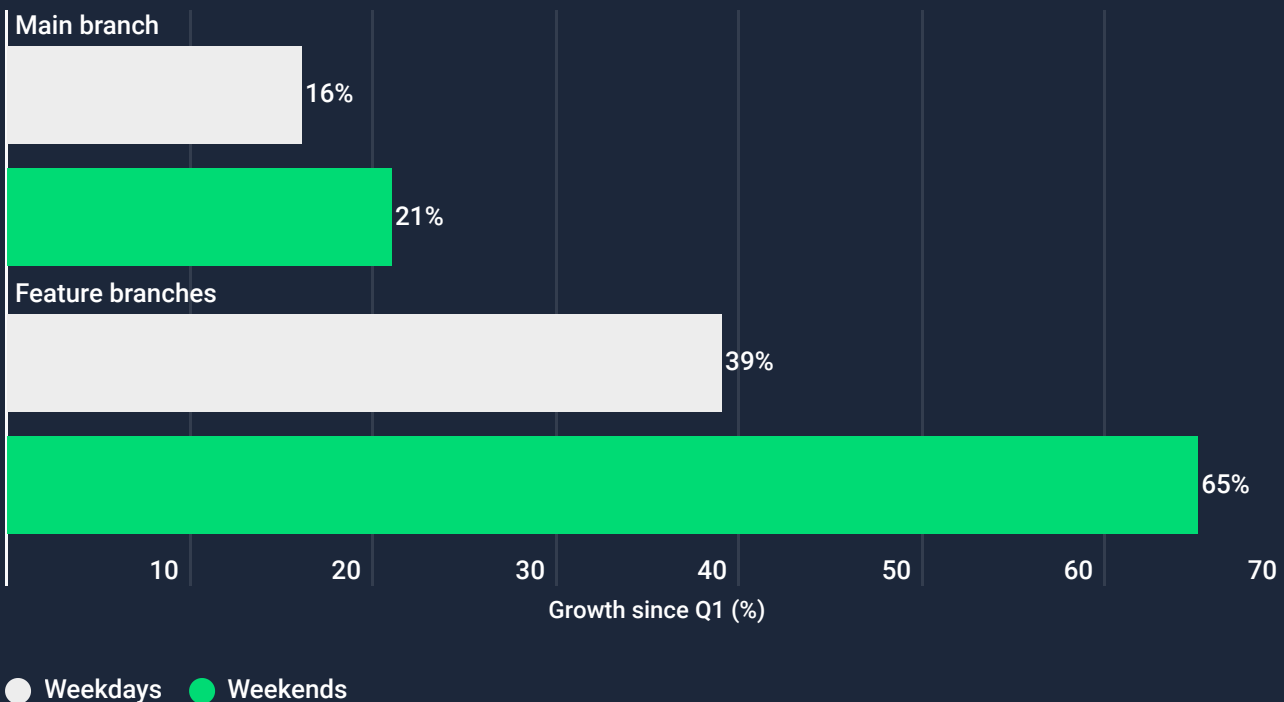
Teams are doing more engineering work than ever, but for most of them, delivery speed hasn't kept pace.



Field note: Development spills outside the workweek

Weekend activity shows how AI is expanding the amount of work teams can keep in progress. Since Q1, the number of distinct workflows active on a typical weekend day has grown 46%, led by feature branches (+65%) while main grew just 21%. Total weekend feature-branch runs increased only 35%, so more feature workflows are active while each runs less often, and far less of the expansion is showing up on main.

Growth in daily active workflows since Q1



Field note

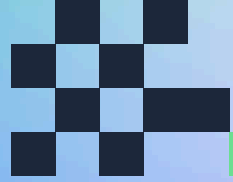
(contd.)

AI and automation account for a significant share of the increase. Weekend branches named for coding agents like **claude/** and **codex/** grew nearly 14x since Q1. Together with dependency bots like Renovate and Dependabot, agents and bots now account for about one in five feature branches running CI on weekends, twice their weekday share. Most agent work carries no identifiable branch name, so the real share is likely higher.

Weekends expose the constraint at the other end of the process. Failed feature-branch workflows take 104 minutes to recover, compared with 79 minutes on weekdays, while main-branch failures recover in about 53 minutes either way. Teams are protecting the critical path to main, but AI is increasing work in progress faster than teams can validate, fix, and move changes onto main. The result is a familiar work-in-progress problem: more capacity to start work without the same capacity to finish it.

14x

Growth in weekend branches named for coding agents since Q1.



4

Top teams are adding workstreams and increasing delivery velocity

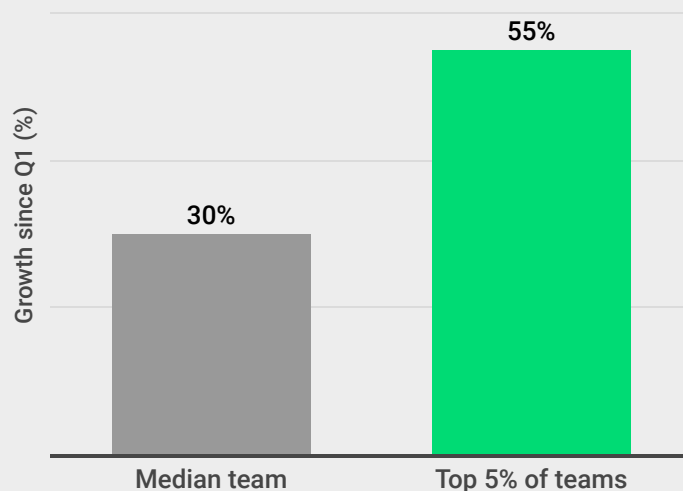
Like the median team, top performers are using their extra capacity to expand the R&D roadmap, taking on more work across more branches and workflows. Top teams, however, are also getting more of that work onto main.

Development activity is expanding rapidly

The most active teams are expanding their scope faster than anyone. Since Q1, top teams are working across 55% more feature branches per team, nearly twice the growth of the typical team. They've done it without adding people.

On its own, that would leave top performers in the same position as everyone else: more work in flight, spread across the same people. What sets them apart is their ability to ship.

Feature-branch growth since Q1



More of that work is reaching main

Top teams are speeding up their workstreams as they add new ones.

Since Q1, top performers have increased throughput on each main-branch workflow by 31%. With the extra workflows they've added since Q1, their total main-branch throughput, meaning all runs on the main branch combined, is up 57% without a measurable increase in headcount.

That's the reverse of the pattern in the previous section. The typical team is adding work without moving any of it faster, while top teams are moving more work through main every day.

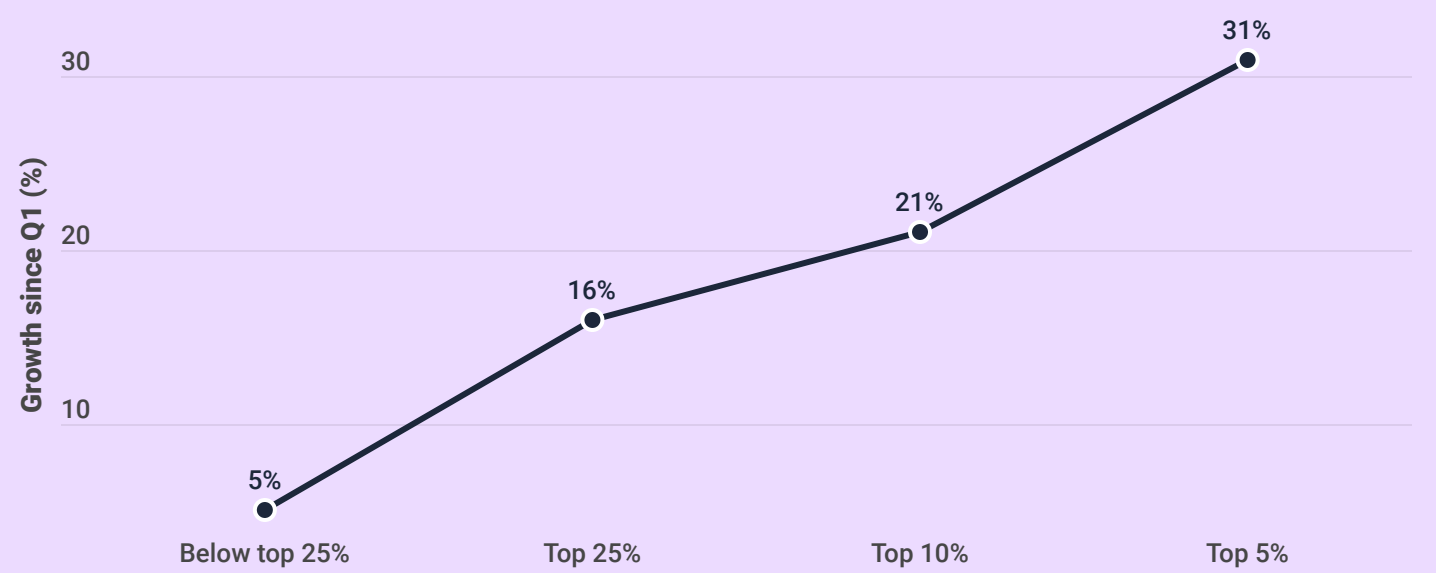


The gains grow toward the top

Earlier, we saw that the most active organizations are increasing their total main-branch activity faster than everyone else. But they're not getting ahead simply by adding more workflows. They're also running each workflow more frequently.

Since Q1, main-branch throughput per workflow has increased 16% among the top quarter of organizations, 21% among the top 10%, and 31% among the top 5%. Below the top quarter, growth was just 5%.

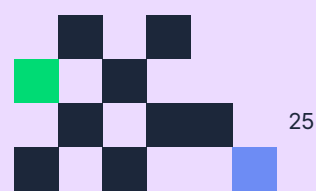
Main-branch throughput growth per workflow since Q1

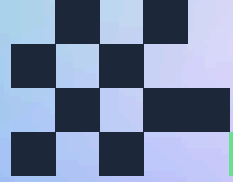


Top cohorts cumulative; ranked by total main runs

The dividing line is around the top quarter of organizations. Most teams are adding projects, branches, and workflows without increasing throughput on any individual workflow. Above p75, teams are expanding their workstreams while also increasing the pace of each one.

The most active teams are making better use of their growing development capacity. They're taking on more work without adding headcount, but unlike the median team, they're also getting more of that work onto main. Their advantage isn't simply that they have more workstreams running. Each workstream is moving faster, and the gains get larger toward the top.





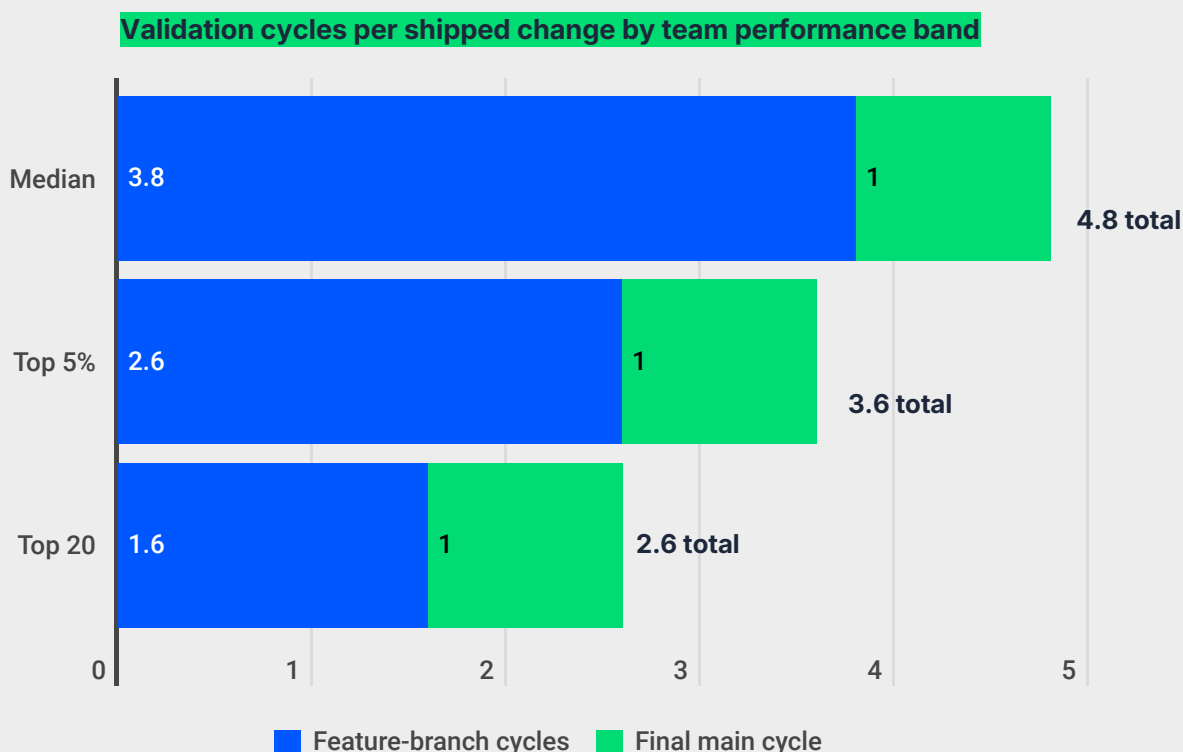
5

Merge efficiency still separates the teams pulling ahead

In Q2, we introduced merge efficiency ratio (MER): the number of feature-branch workflow runs for every workflow run that lands on main. As agentic development enables teams to take on more work across more projects and branches, efficient validation increasingly determines how much of that work reaches production. In Q3, the teams merging the most code into main continue to do so with significantly fewer validation cycles than everyone else.

Top performers keep MER low

At the median, MER is 3.8, virtually unchanged from 3.9 in Q2. Including the final run on main, that's roughly five total CI cycles per shipped change. For the 5% of teams with the most main-branch activity, MER also held steady at 2.6. Among the 20 most active organizations, the ratio was even lower at 1.6, equivalent to fewer than three total CI cycles per shipped change.



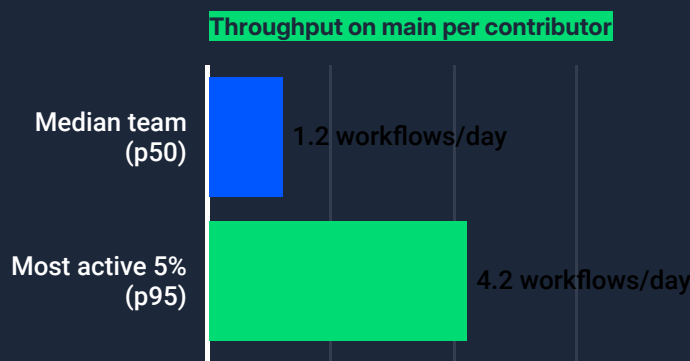
A difference of one or two validation cycles may seem small for an individual change. But across hundreds or thousands of changes a day, extra rounds of testing and fixing add up quickly in both time and cost. For a detailed breakdown of the ROI impact of merge efficiency, see our [Q2 Pulse report](#).

Despite improving CI success rates, MER has barely changed at the median. Reducing unnecessary validation cycles could help more teams turn their expanding development activity into faster delivery.

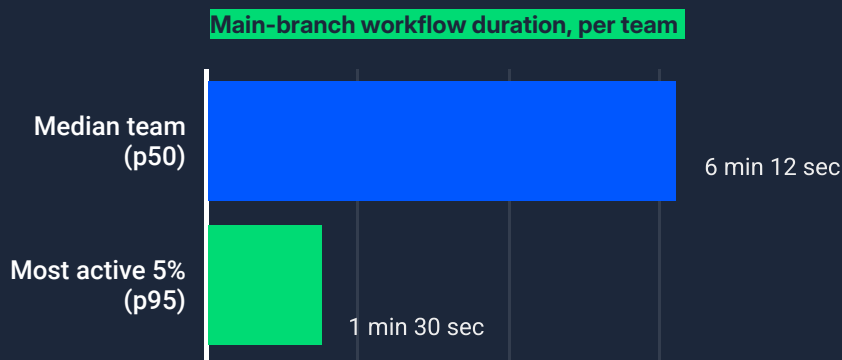
Other habits of top performers

In Q2, we also identified higher delivery output per contributor and more validation outside the commit loop among the 5% of teams with the most main-branch activity. Both advantages persist in Q3, alongside several additional findings about how these organizations validate their code.

- **More delivery output per contributor.** Developers at the most active 5% of teams average 4.2 main-branch workflow runs a day, about three and a half times the 1.2 for the median team. Since Q2, main-branch runs per contributor have increased 38% at the top, compared with 11% for the median team.

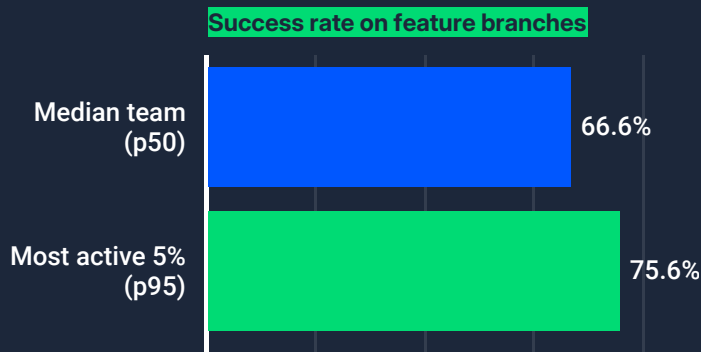


- **Shorter CI feedback loops.** Main-branch workflows finish in 1 minute 30 seconds on the most active 5% of teams, compared with 6 minutes 12 seconds on the median team. Feature-branch workflows are more than twice as fast, at 2 minutes 12 seconds versus 5 minutes 12 seconds.



Other habits of top performers

- **More reliable validation before merge.** Feature-branch workflows succeed 75.6% of the time among the most active 5% of teams, compared with 66.6% for the median team.



- **More validation beyond the commit.** Code pushes trigger virtually all pipelines for the median team, compared with 82.4% at the top. The remaining pipelines are triggered primarily through schedules and API calls.
- **More granular CI workflows.** The most active 5% average 2.6 workflows per project, compared with just 1 for the median team. Spreading validation across more workflows may allow teams to run checks independently and get more targeted feedback.

The organizations moving the most work through main get faster feedback, break the build less often, and need fewer validation cycles to move a change onto main. As development activity grows, these advantages may help explain why their delivery velocity continues to pull ahead.



6

Where to focus for the rest of the year



Teams have more development capacity than ever, but most are still struggling to turn it into faster delivery. The Q3 findings point to four areas where teams can reduce unnecessary work, improve feedback cycles, and get more changes through validation and onto main.

01

Catch problems before the push

Every failure caught during development can save another round trip through CI. Start by identifying the errors that repeatedly cause feature-branch workflows to fail, especially linting, type-checking, build, and unit test failures that could have been caught earlier. Run the relevant checks as code changes, and give coding agents the ability to run them and respond to failures before committing or pushing.

With CircleCI: Chunk sidecars provide coding agents with a fast, CI-aligned validation environment. Agent hooks can trigger checks automatically, helping agents catch and resolve routine failures during development.



02

Make CI failures easier to investigate and resolve

Feature-branch failures now take a median of 86 minutes to resolve, rising to 104 minutes on weekends. As teams spread their work across more branches and projects, developers have more failures to monitor and investigate. Make pipeline status and failure details available where developers and agents are already working. Give agents access to test results and logs so they can investigate failures, propose fixes, and request targeted validation without developers having to gather context manually.

With CircleCI: The [CircleCI CLI](#) and [hosted MCP server](#) give coding agents access to pipeline status, failure logs, test results, and rerun controls, allowing them to help developers investigate and resolve CI failures.

03

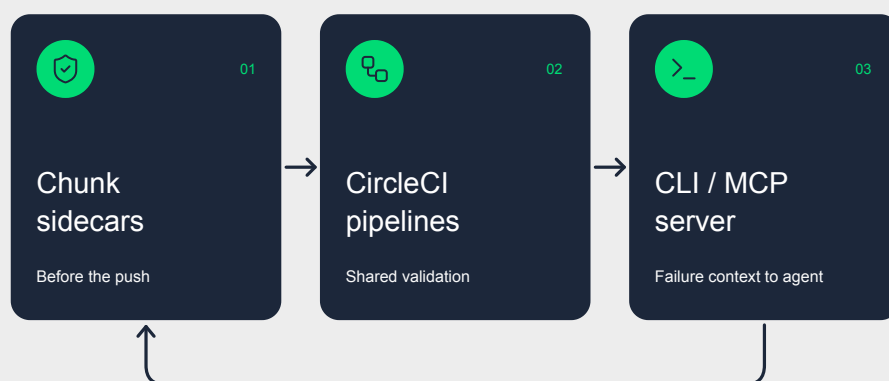
Shorten feedback cycles without reducing coverage

The most active teams finish main-branch workflows about four times faster than the median team. Start by identifying the jobs and tests responsible for the longest waits. Run independent checks in parallel, cache dependencies and build artifacts, and avoid repeating expensive work when a change doesn't require it. Keep the checks needed to validate each change, while moving periodic or nonblocking checks into separate workflows.

With CircleCI: [Smarter Testing](#) supports test impact analysis, dynamic test splitting, and automatic retries for flaky tests, helping teams shorten feedback cycles while maintaining meaningful validation coverage.

Keep AI-generated changes small and easy to merge

Higher CI success rates alone haven't reduced MER for the median team. As agents produce more code across more workstreams, large or long-lived changes create additional review, validation, and integration work. Give agents focused tasks with clear acceptance criteria, keep branches short-lived, and merge completed changes frequently. Track how many validation cycles changes require before reaching main, then look for recurring causes of rework, including oversized diffs, merge conflicts, and failures caught late.



Where to start

Calculate your team's MER for the last month: divide the total number of feature-branch workflow runs by the total number of main-branch workflow runs. If the ratio is near four, look at the changes requiring the most validation cycles and identify what's causing the repeated work. Focus on the most common source of delay first, whether that's failures caught too late, slow feedback, difficult reviews, or long-lived branches. The goal is to help more work reach main with fewer rounds of testing, debugging, and waiting.



7

Looking ahead to Q4 and 2027



Looking ahead to Q4 and 2027

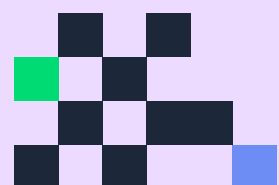
AI has made new work much easier to start, but teams' capacity to finish it hasn't kept pace. The underlying problem is a familiar one in project management: when organizations take on more work than they can complete, progress slows. Coding agents are bringing that challenge to a new scale. It's easier than ever to say yes to an idea and see quick results, but each new workstream still needs review, validation, and attention before it reaches main. As teams spread their attention across more work in progress, they risk finishing less of what they start.

There are encouraging signs elsewhere. Improvements in AI models, CI platforms, and developers' ability to work with coding agents are contributing to higher success rates, faster workflows, and quicker recovery from main-branch failures. Yet most teams aren't getting code onto main any faster, even as they expand development activity. The top-performing teams show it can be done: they're taking on more work while increasing how frequently changes reach main.

As organizations plan for 2027, the challenge is to bring delivery capacity in line with development capacity. That means managing work across more projects and branches, making validation and integration more efficient, and adapting delivery processes so teams can finish more of the work they start. To better understand how organizations are tackling these challenges, we're supplementing CircleCI's platform data with a Q4 industry survey. We'll explore how teams use AI, where they encounter friction, and which practices and investments they're prioritizing for 2027.

Add your voice to the 2026 State of Software Delivery Q4 Pulse report.

[Take the survey](#) and help us understand what's helping teams deliver more of the work they start, and what's still holding them back.





Appendix 1:

Additional findings

The gap between the highest-throughput teams and everyone else continues to widen. But company size, location, and industry explain less of the variation in typical workflow throughput than they did at the start of the year. Across these groups, most teams are now running workflows at a similar pace, even as MTTR and throughput among the most active workflows continue to vary widely.

Company size no longer predicts typical throughput

In Q1, the smallest and largest companies generally performed best, while mid-sized organizations struggled with lower throughput and longer MTTR. By Q3, much of that gap had closed.

Median main-branch throughput now ranges from 1.57 to 1.71 runs a day across every company size, compared with 1.00 to 1.96 in Q1. Main-branch MTTR also improved across nearly every size group, particularly among companies with 21 to 50 employees, where it fell 53%, from 174 to 81 minutes.

The largest organizations still have an advantage at the top. Among companies with 1,001 or more employees, the top 5% of main-branch workflows run 22.9 times a day, compared with 11.6 to 17.3 across other company sizes.

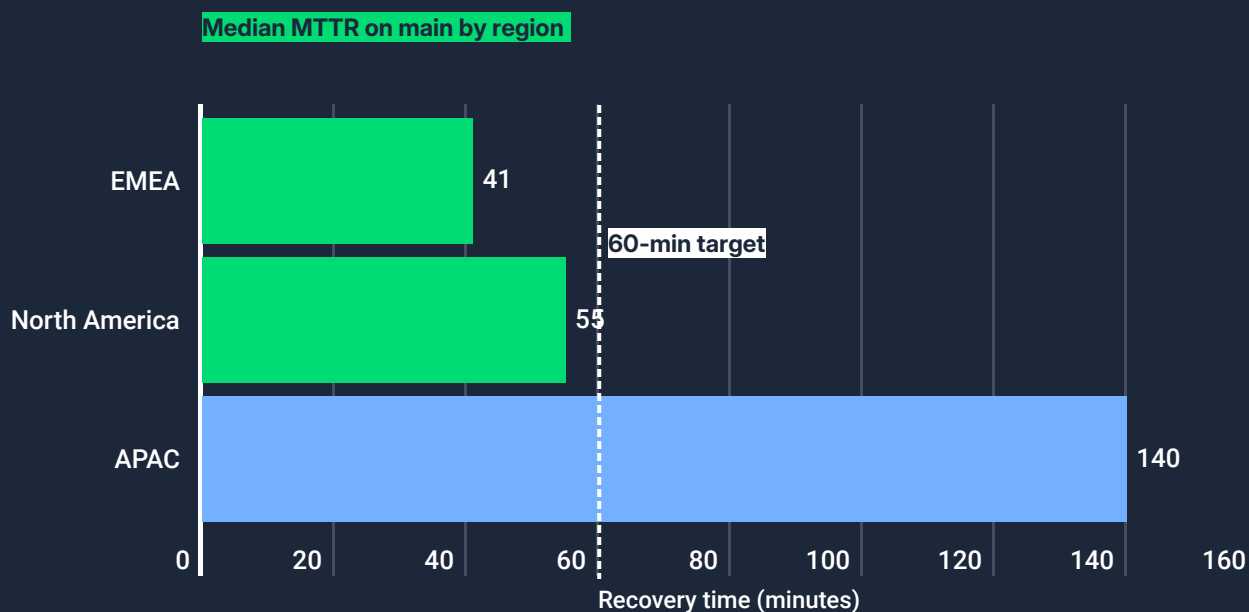
Range of median main-branch throughput by company size



Regional throughput is converging, but MTTR still varies sharply

The largest software markets now have remarkably similar median throughput. In the US, UK, Germany, Canada, and Japan, the typical workflow runs between 1.68 and 1.75 times a day across all branches.

Teams in these markets still differ substantially in how quickly they fix failures. EMEA has the lowest median main-branch MTTR at 41 minutes, followed by North America at 55 minutes and APAC at 140 minutes. Germany, Australia, and Japan take considerably longer, at 216, 233, and 243 minutes respectively.

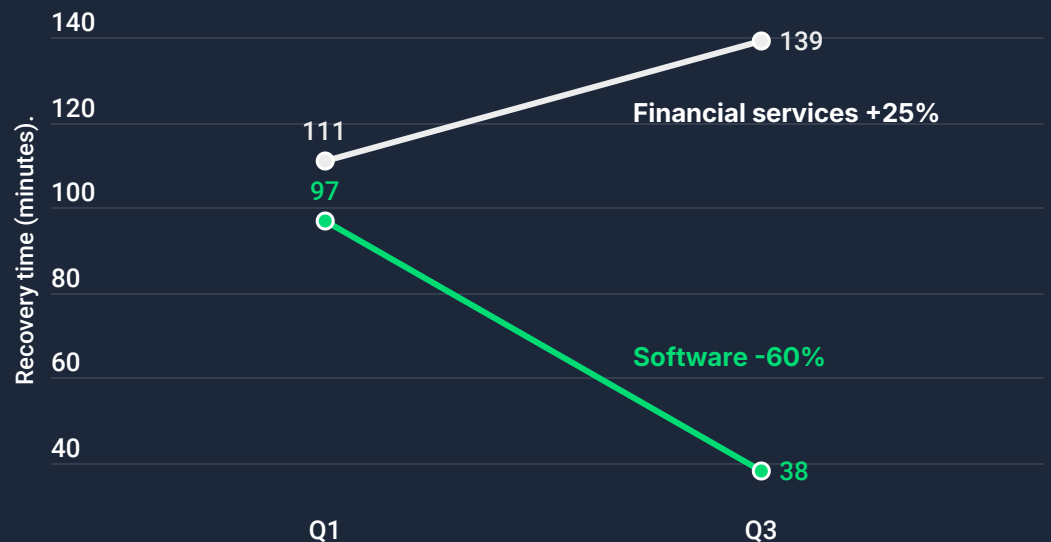


Developers across major software markets are running workflows at similar rates, but teams in some regions still take several times longer to restore broken main-branch pipelines.

Software companies lead a turnaround in MTTR

Software companies have cut median main-branch MTTR by 60% since Q1, from 96 mins 54 secs to 38 mins 23 secs, the fastest among large industries. Financial services moved in the opposite direction, with MTTR rising from 111 mins to 138 mins 40 secs.

Median MTTR on main by industry



Across industries, teams are running workflows at increasingly similar rates, but they're making very different progress on failure recovery. Software companies have substantially shortened the time it takes to restore broken main-branch pipelines, while financial services firms are taking longer than they did at the start of the year.

Company size, industry, and location may shape the challenges organizations face, but none determines how fast they can deliver. The highest-performing teams are showing that faster delivery is possible across different organizational contexts, and that the practices behind their success matter more than the circumstances they operate in.



Appendix 2:

Q3 metrics at a glance

Metric	Main branch	Feature branches	All branches
Throughput (runs/workflow/day)	1.71	1.89	1.71
Success rate	78.3%	72.6%	74.0%
Duration	2m 2s	2m 46s	2m 30s
Mean time to recovery (MTTR)	54 min	86 min	75 min
Merge efficiency ratio	—	—	3.77

Methodology and contributors



To create this report, we analyzed data from over 22 million CircleCI workflows from August 1-28, 2026. We compared them with the windows used in our earlier 2026 reports: September 1-28, 2025 for the annual report (Q1) and March 1-28, 2026 for the Q2 Pulse report.

To ensure benchmarks focus on repeatable software delivery, we restricted workflow-level metrics (throughput, success rate, duration and recovery time) to projects with more than one contributor and workflows that ran at least four times in the window.

Team-level metrics (merge efficiency ratio, activity groups, branches and workflows per team, and per-contributor figures) include all activity in multi-contributor projects with no minimum run count. Activity groups rank teams by main-branch workflow runs in each window. Standard exclusions were applied across all metrics to remove test accounts, abusive projects and internal CircleCI activity.

Branch designations reflect each branch's status when its pipeline ran. Main-branch activity is used as a proxy for work delivered to production. Industry, company size and region data is not available for all organizations.

Data details:

- Every day between August 1, 2026 and August 28, 2026 (compared with the Q1 and Q2 report windows: September 1-28, 2025 and March 1-28, 2026)
- 22,144,586 workflows evaluated
- Workflow-level metrics restricted to projects with more than one contributor and workflows with at least four runs
- Team-level metrics use all activity in multi-contributor projects

Contributors

Report authors: Jacob Schmitt, Ron Powell

Report editor: Erin O'Leary

Report designer: Ruban Tamrakar

Acknowledgments: Jason Lee, Christian Otto, Yael Lior, Bijan Samimi