DATA REPORT

2026 Engineering Reality Report

Chainguard

Table of Contents

Introducing the 2026 Engineering Reality Report

As engineering organizations are working through fiscal planning, headcount planning, and talent reviews moving into 2026, we wanted to take a look at the modern-day developer experience (the people, processes, and tools that allow engineers to work efficiently) and the reality of being a software engineer to help leaders prioritize their investment areas. We surveyed 1,200 engineers and technology leaders to learn more about how they think about AI and automation, their day-to-day tasks, and where organizations are falling short in areas like tooling, engineering retention, and more.

Toil makes it hard for engineers to build new features and create value for their organizations

72% of engineers said that demands on their time make it difficult to make space for building new features.

Having time to build is key to a positive developer experience

93% of engineers reported finding writing code and building new features to be rewarding, but they only spend 16% of their week on this type of work.

AI and automation are reaching critical mass and improving the developer experience

65% of respondents said that the majority of their organization's common software engineering tasks are either mostly or fully automated. Of those respondents automating a majority of these tasks, 94% reported that they are spending the majority of their time on work that energizes them, compared to only 67% from respondents who do not automate the majority of their tasks.

Tool sprawl reduces productivity

88% of respondents said switching between tools impacts their productivity, with 44% reporting significant loss of focus due to this context switching.

The Developer Experience Dilemma: Where Can Teams Improve?

The survey paints an important picture of the present-day developer experience: it is difficult for software engineers to make time for building new features and delivering code for their organizations. **Seventy-two percent (72%) of engineers state that the demands on their time make it difficult to find space to build new features.** Engineers face pressure to deliver new features and respond quickly to customer needs, but they are also expected to maintain existing systems, adopt new tools, and follow evolving security practices. The result: innovation often takes a back seat to operational demands. One engineer put it succinctly when asked how the developer experience could be improved at their organization:

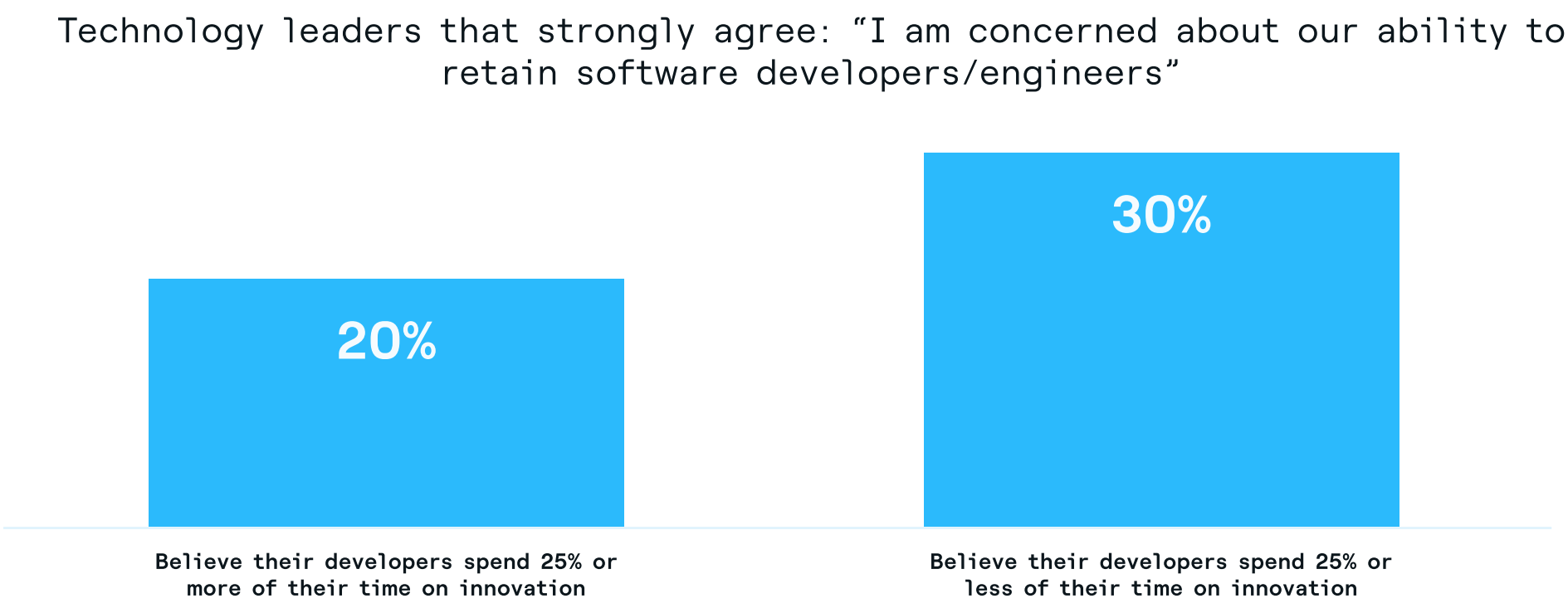
> **"Stop throwing competing priorities at teams and expecting everything to be done at once and at top priority."**

72% of engineers state that there are **so many demands on their time** it is difficult to find space for building new features

Only one in three engineers strongly agreed that they spend the majority of their week on work that energizes them. Instead, **the vast majority of engineers (79%) pointed to code maintenance as a major drain on their time**, leaving fewer hours to build new features, experiment with technologies, or design new systems. For engineers, this imbalance can make the role feel more focused on upkeep than innovation. For leaders, it translates into slower progress toward business goals and risk of disengagement and churn across their engineering teams.

Leaders are beginning to see the consequences. **Two-thirds (66%) of senior technology leaders reported being worried about retaining engineering talent.** When engineers feel bogged down by maintenance and repetitive work, the role becomes less inspiring, making it harder to keep teams intact. At a time when demand for skilled engineers remains high, the stakes of failing to close the experience gap can be significant.

Spending time on innovation improves retention



Simply put, to improve the developer experience, organizations need to reduce the amount of work that feels repetitive and unrewarding. Engineers and senior technology leaders alike pointed to the same obstacles: too many tedious tasks, too much time spent on code maintenance, and workloads that push teams toward burnout. These recurring frustrations make it difficult for engineers to focus on the creative, energizing aspects of their work, and they weigh heavily on overall job satisfaction.

Top reasons engineers don't feel highly positive about their developer experience

- Frequent need for code maintenance (upgrades, breaking changes, patching, vulnerability resolution) distracting from key priorities (38% of respondents)
- Too many tedious tasks that take them away from meaningful, productive work (38%)
- Excessive workload or burnout (35%)
- Limited learning and development opportunities (33%)
- Frequent switching between coding tasks and non-coding tasks (32%)
- Lack of recognition and appreciation (28%)
- Lack of effective tools to allow engineers to do their work (28%)
- Too many tools, creating integration difficulties (27%)

Top reasons senior technology leaders don't feel highly positive about their team's developer experience

- Frequent need for code maintenance (upgrades, breaking changes, patching, vulnerability resolution) distracting from key priorities (43% of respondents)
- Too many tedious tasks that take them away from meaningful, productive work (37%)
- Frequent switching between coding tasks and non-coding tasks (34%)
- Excessive workload or burnout (33%)
- Lack of recognition and appreciation (33%)
- Lack of effective tools to allow engineers to do their work (32%)
- Limited learning and development opportunities (29%)
- Too many tools, creating integration difficulties (25%)

The effects of these challenges go beyond day-to-day frustration. The survey shows that engineers who feel weighed down by maintenance and repetitive work are less likely to describe their experience as highly positive, and leaders are already linking this to retention risks. These issues are compounded by a staggering amount of technical debt at the average organization, with 66% of respondents saying they frequently or very frequently come across technical debt that impacts their ability to deliver work effectively. Burnout remains a looming threat, as engineers juggle competing priorities and feel that the bulk of their time is spent on tasks, like code maintenance and admin, which do not inspire them. In environments where talent is scarce and competition is fierce, this dynamic has the potential to create long-term challenges for organizations trying to hold onto skilled software engineers.

While the survey highlights the scale of the problem, it also shows a shared desire to improve. The alignment between engineers and their leaders suggests that the issue is not one of awareness, but of execution: everyone recognizes that toil is taking engineers away from higher-value work. They agree that creating the conditions for engineers to focus on higher-value work is essential, even if many organizations have yet to find a sustainable way to achieve it. Until that happens, burnout will remain an ever-present risk, fueled by the imbalance between what engineers aspire to do and what their roles currently demand.

Innovation Over Maintenance: What Engineers Really Value

If there is one thing that unites both engineers and technology leaders, it is the shared belief that engineers should be spending more of their time building rather than maintaining. The survey shows a clear ambition to shift the balance of work toward building new features, writing code, and designing systems—work that both groups agree are the most rewarding and impactful. But the current reality tells a different story, with maintenance and repetitive tasks continuing to dominate the engineering workload.

Engineers expressed a strong desire to increase the portion of their time spent on feature development. While 93% of engineers found writing code and building new features to be rewarding, **they report spending only around 16% of their week on this type of work**. Technology leaders voiced similar ambitions, saying they want their engineers to dedicate more time to creating new code. This alignment signals that both sides of the organization see innovation as a critical priority, even if day-to-day demands keep engineers anchored to upkeep.

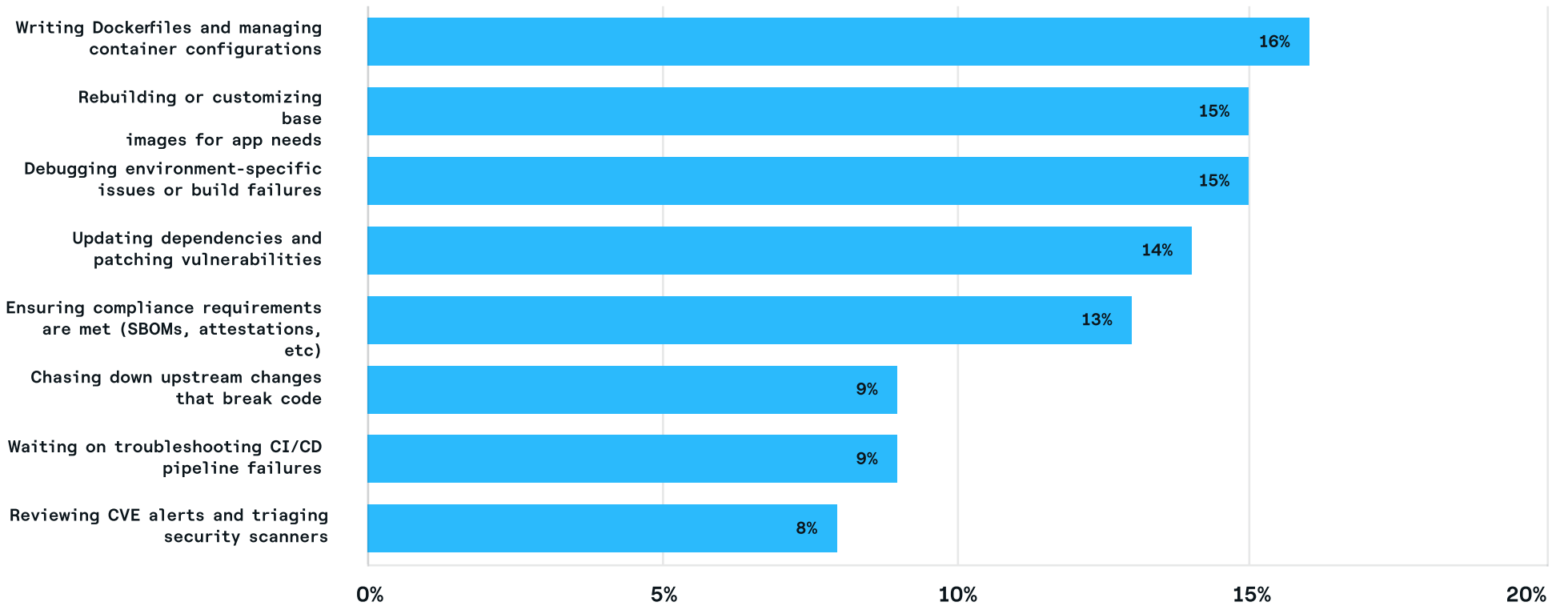
When asked for ways to improve the developer experience, several responses from both technology leaders and engineers mentioned this need for innovation:

> **“Focus on building an in-house development toolchain and self-service platform, freeing developers from tedious non-development tasks and allowing them focus more on writing code”**
>
> Technology Leader, France

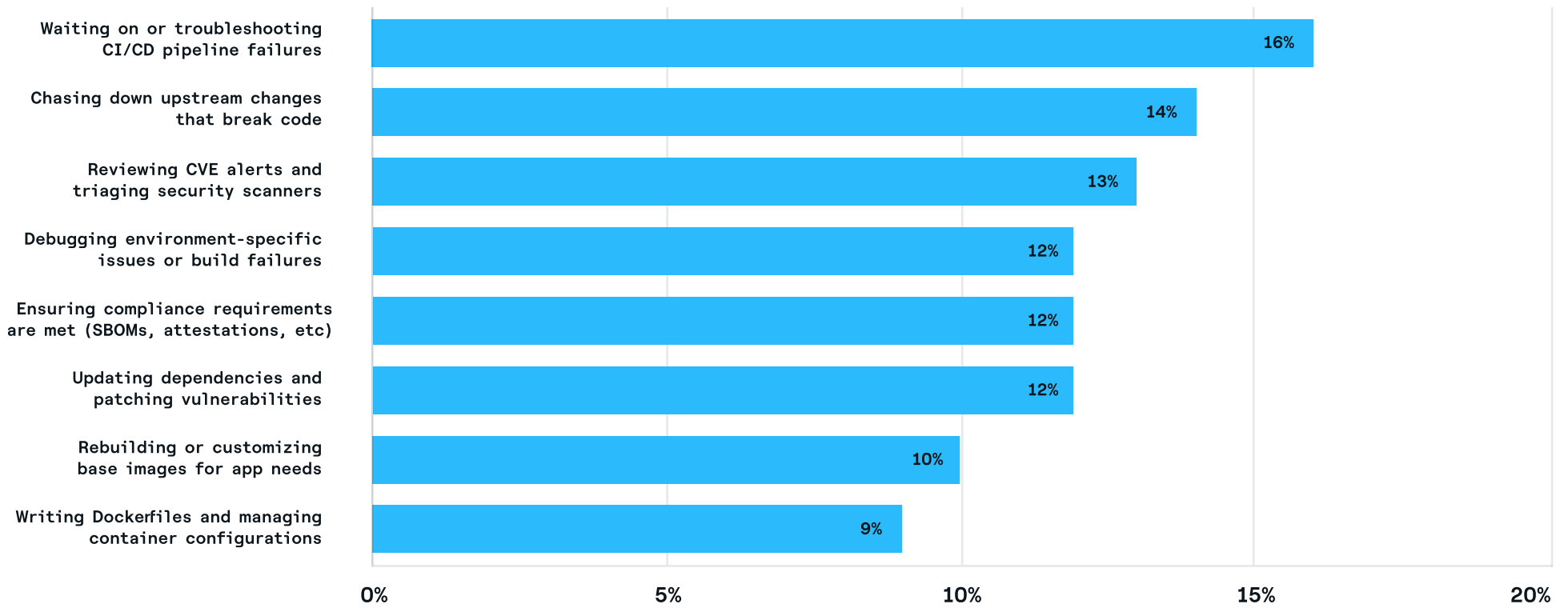
> **“Allow engineers to spend more time building and less time debugging infrastructure troubleshooting”**
>
> Engineer, UK

> **“A consistent, well-documented, and standardized setup would minimize time spent on configuration and troubleshooting”**
>
> Engineer, UK

Day-to-day tasks engineers like doing the most



Day-to-day tasks engineers like doing the least



The survey also explored which types of work engineers find most rewarding, and the results reinforce this theme. Engineers want to spend their time building and developing features, not dealing with cumbersome admin tasks or difficult, time-consuming maintenance work.

Despite the overall consensus, maintenance work continues to dominate engineers' time. Engineers described how frequent patches, fixes, and updates consume significant portions of their time. For many, this not only limits innovation, but also erodes enthusiasm for their roles. Leaders echoed these frustrations, noting that the balance of work is tilted too far toward tasks that do not contribute directly to business growth. The result is a paradox: everyone agrees on the value of innovation, but most can't focus on it enough.

AI and Automation Move Towards Critical Mass

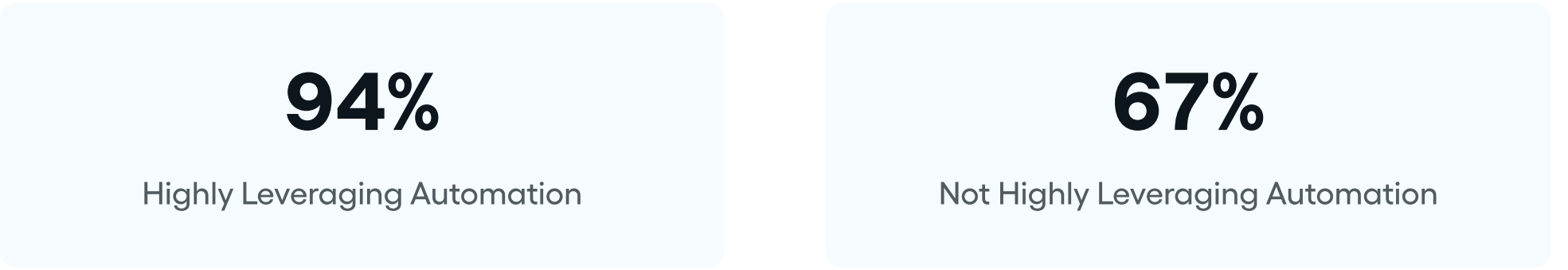
Artificial intelligence (AI) and automation are already deeply embedded in the engineering workflow, and the survey results show that engineers and technology leaders are seeing measurable benefits. For many organizations, these technologies have moved from experimentation to adoption, saving teams time on routine tasks and freeing them to focus on more complex work. Automation technologies, which focus on using pre-defined rules to execute repetitive tasks consistently, are especially prevalent.

Percentage of respondents whose organizations have mostly or fully automated various engineering activities

- Testing, monitoring, and quality assurance (68% of respondents)
- Security patching and vulnerability remediation (67%)
- Incident response and trouble shooting (66%)
- Administrative tasks, meetings, and internal communication (65%)
- Code review (for colleagues, peer reviews) (65%)
- Maintaining code (upgrades, breaking changes, rebuilding or customizing base images, etc.) (64%)
- Writing code, documentation, and building new features (63%)
- System design and architecture (63%)

The survey results indicate strong adoption of automation technologies across most common software engineering tasks, with at least 60% of respondents claiming to mostly or fully automate everything from writing code to dealing with administrative tasks like internal communication. **This number is quite high, and poses an important question: Is all this automation actually making the developer experience better?**

Percentage of respondents who spend the majority of their time on work that energizes them

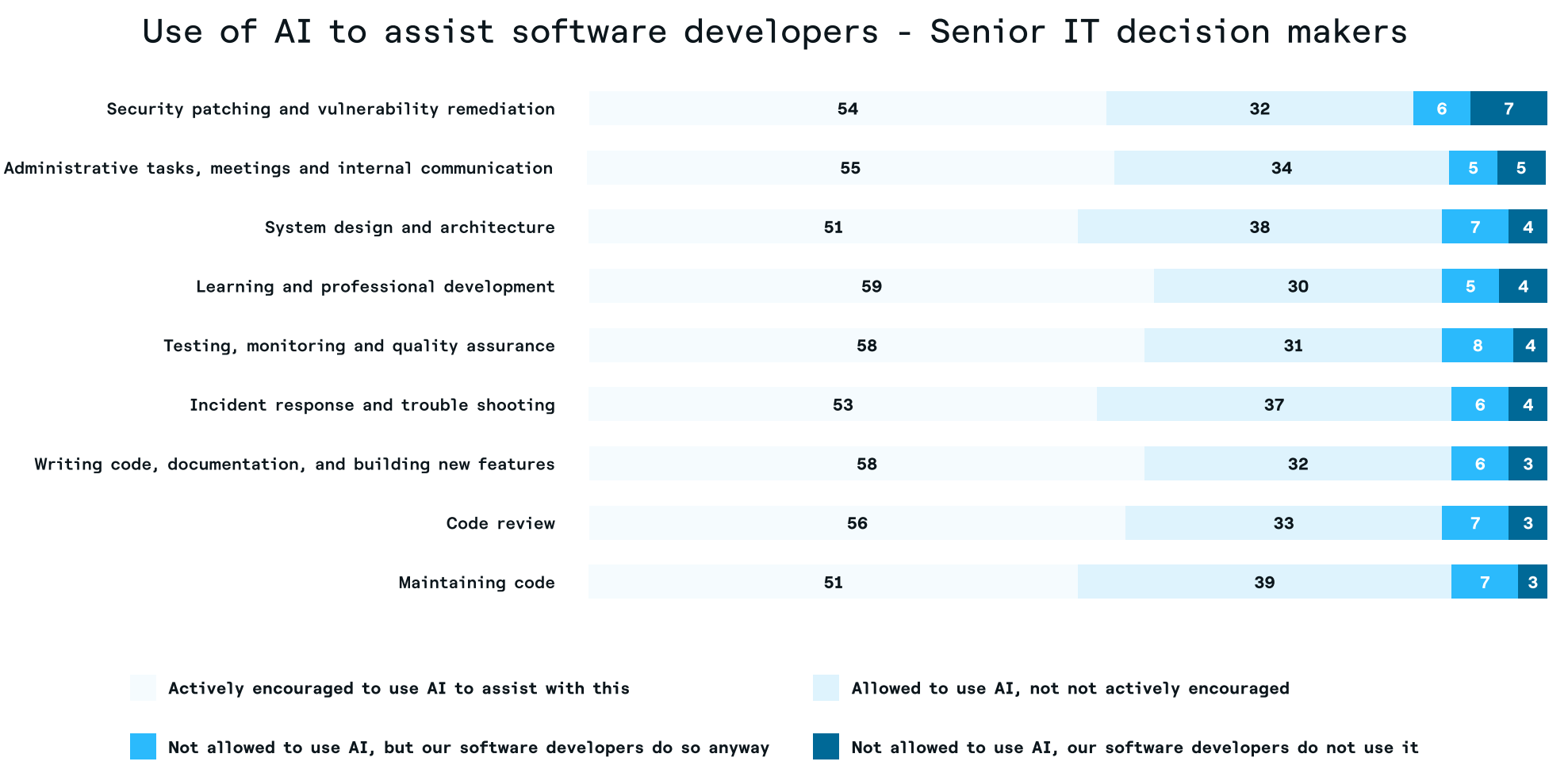


The answer based on the survey results is a resounding “yes.” Perhaps the strongest indicator of a positive impact from automation is how respondents reacted to the statement “I spend the majority of my time on work that energizes me.” **Ninety-four percent (94%) of respondents who said they mostly or fully automated at least half of the tasks highlighted above agreed that they spend the majority of their time on work that energizes them.** When examining respondents who do not mostly or fully automate at least half of the tasks, only 67% agreed on spending the majority of their time on work that energizes them (with only 14% strongly agreeing, compared to 43% for “high automation” individuals).

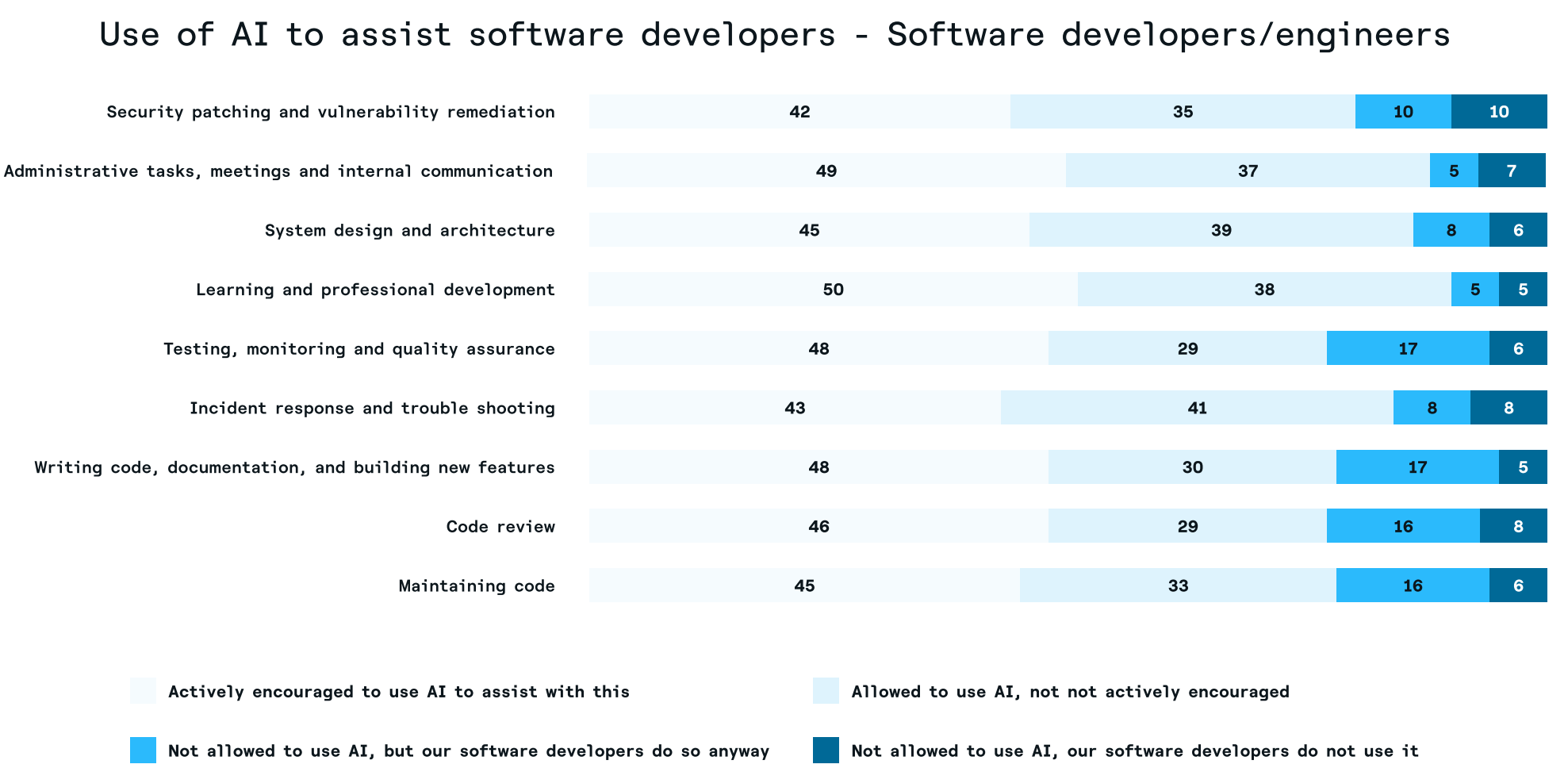
Automation is one of the most effective methods engineering organizations can leverage as they look to improve the developer experience. As mentioned earlier, **repetitive and difficult tasks like upgrading, patching, and vulnerability management often cause toil that takes away from the developer experience and makes engineers less excited about their jobs and less productive as a result.** Automation tooling in these areas can allow engineers to spend time on the aspects of their job they enjoy most, like writing code. Based on the survey results, many organizations already recognize the benefits here, and we expect adoption to continue to accelerate.

AI technologies are also rapidly increasing in adoption and generally proving to be useful. Eighty-nine percent (89%) of organizations reported that engineers are saving at least three hours a week thanks to AI, with 28% saving between five and six hours. These time savings reflect how AI is already reshaping workflows, from generating code snippets to supporting documentation and testing. For teams under pressure to deliver more (sometimes with less resourcing), the ability to reclaim hours each week has the potential to shift the balance of work toward higher-value activities. Many teams are already utilizing AI and automation technologies to do just that, mostly or fully automating activities like testing and incident response to free up time.

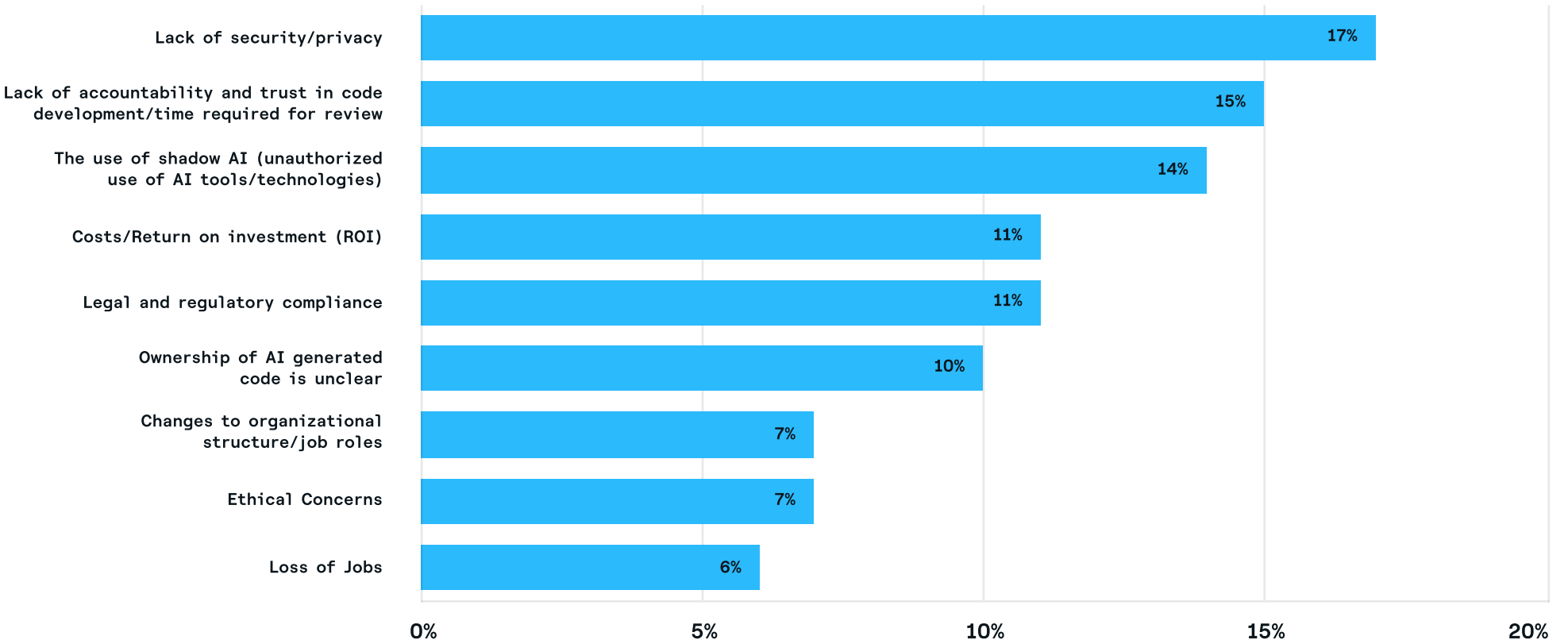
Around half of senior technology leaders state that software developers are encouraged to use AI



Software developers report lower usage of AI than perceived by senior technology leaders



Respondents' top concerns about AI in software development and engineering



Concerns about ethics and job loss typically drive reduced enthusiasm for automation and AI technologies. The traditional narrative, misguided or not, is that organizations are implementing AI tools and strategies to reduce labor costs. According to the respondents of this survey, that is not the case for engineering teams. Of all the reasons why respondents may be concerned about AI adoption in their organizations, ethical concerns and loss of jobs ranked last on the list, with just 7% and 6% of respondents reporting them as reasons to be concerned about AI usage, respectively.

Engineers' enthusiasm for AI is also tempered by mixed signals from leadership. **While 55% of technology leaders said their engineers are encouraged to use AI, only 46% of engineers reported feeling that same level of support.** Only a small proportion of leaders (9%) said that AI is outright prohibited in their organizations, but engineers themselves were more likely to believe that restrictions are in place (22%). In some cases, this disconnect comes from limits on where AI can be applied. Around one in five respondents said engineers are not allowed to use AI for some of the most time-intensive tasks, such as patching vulnerabilities, running tests, or conducting code reviews. For engineers, these restrictions reduce the practical impact of AI and may contribute to skepticism about how fully organizations are prepared to embrace it.

Concerns about trust also loom large. **Engineers highlighted accountability, security, and privacy as their top worries, with over 40% pointing to each as a barrier to full adoption.** Many also raised the issue of “shadow AI” — engineers will often use tools that aren’t approved by their organizations, creating risks around compliance and governance. These findings show that while AI adoption is growing, engineers are aware of the risks that come with it and are hesitant to rely too heavily on tools they do not fully trust.

The survey paints a picture of AI in transition: productivity benefits are real, and adoption is spreading, but cultural, regulatory, and trust barriers are slowing momentum. Leaders are slightly more optimistic than engineers, but both groups recognize that AI is part of the future of software engineering. The challenge ahead lies not in proving its value, but in building the trust and guardrails needed for AI to become a natural, reliable part of the developer experience.

Engineering Tool Sprawl Reduces Productivity

Behind every line of deployed applications is a complex web of tools. Software engineers now rely on a wide range of platforms, repositories, and applications to power their workflows, with open source playing an especially central role. At the same time, the very abundance of tools creates challenges of integration, efficiency, and focus that can hinder productivity rather than enhance it. Almost half (49%) of all respondents agreed that tool-related issues contributed to a negative developer experience. Several engineers were opinionated here:

> **"Simplifying and integrating the development environment and toolchain will greatly improve our developer experience"**
>
> Engineer, UK

> **"It's crazy that we have to switch between five different tools just to deploy a microservice. It would be such a relief if we could consolidate into one coherent workflow"**
>
> Engineer, USA

The challenge is that adoption does not always mean seamless integration. More than half of respondents (57%) said their open source tools are not fully integrated into their workflows. This lack of cohesion means engineers are often left stitching together tools that do not connect smoothly, creating additional manual work and slowing down the development process. Here, a perception gap emerges: Technology leaders tend to be more optimistic about integration, while engineers experience the daily friction of juggling tools that fall short of working seamlessly together.

62%

of engineers said their organization's tools are not fully-integrated

52%

of senior technology leaders said their organization's tools are not fully integrated

That friction is compounded by context switching. **70% of engineers reported using more than five different tools each week, and nearly nine in ten said that switching between them has a negative impact on their productivity.** For almost half, the impact is significant, with 44% reporting a major loss of focus as they navigate between platforms. In practice, this means that engineers often lose valuable time reorienting themselves across fragmented systems rather than building or maintaining code.

Interestingly, and perhaps not surprisingly, respondents who reported mostly or fully automating a majority of their common engineering tasks reported much better integration. Fifty-eight percent (58%) of respondents in this category said their organization's tools were fully integrated. If an organization is planning on using automation or AI for many tasks, proper tool integration is paramount. As this number shows, engineers and technology leaders need to trust that the tools they are using to automate tasks are going to work properly. This is another usage of automation that, when properly integrated, contributes to a positive developer experience by reducing the amount of toil engineers experience day-to-day as they navigate their organization's tool landscape.

87% of engineers reported a negative impact on productivity when switching between multiple tools.

The cumulative effect of tool sprawl is an environment where engineers must balance the benefits of specialized platforms with the inefficiencies that come from fragmentation. Engineers value the flexibility and innovation these tools enable, but they also feel the strain of trying to knit them together into a coherent workflow. Leaders, meanwhile, recognize the potential of specialized tooling but may underestimate the productivity costs of poor integration. As engineering toolchains continue to expand, leaders and practitioners alike need to shift evaluations to account for ease of integration and compatibility alongside core capabilities. The outcome of deep integrations and a high degree of automation maximizes the benefits of robust tooling while minimizing the negative impacts of context-switching.

Key Takeaways

In summary, engineers are not feeling energized by the day-to-day work they are doing. Both audiences reached a broad consensus around the need for better processes around maintenance, tedious admin work, and tooling in order to improve the developer experience and enable engineers to do what they enjoy most: building and innovating.

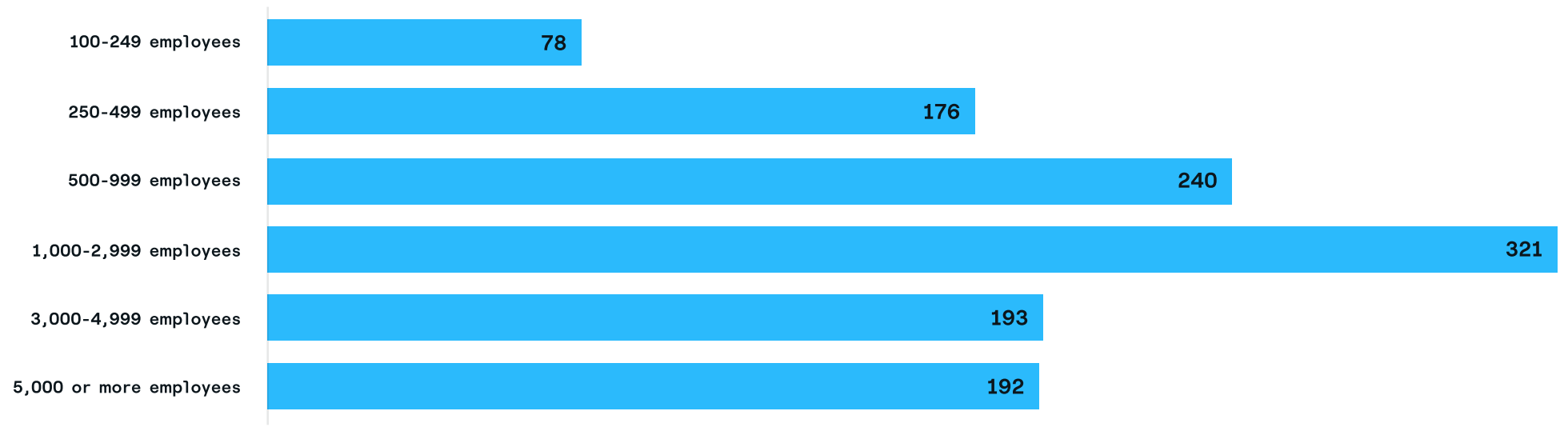
The adoption of AI and automation technologies continues to increase across all sectors surveyed. When paired with proper integration and process, these tools can vastly improve the developer experience by reducing toil and enabling more energizing work. When looking at AI specifically, engineers are seeing measurable benefits in the amount of time they are saving on various tasks, but there is still some confusion and room for improvement as to how and where they can use the technology.

Chainguard has designed products that are tailor-made to reduce the amount of time engineering teams are spending on maintenance and other repetitive tasks, unlocking value through faster innovation, reduced risk, and more. Discover more about Chainguard OS, the operating system that makes it all possible.

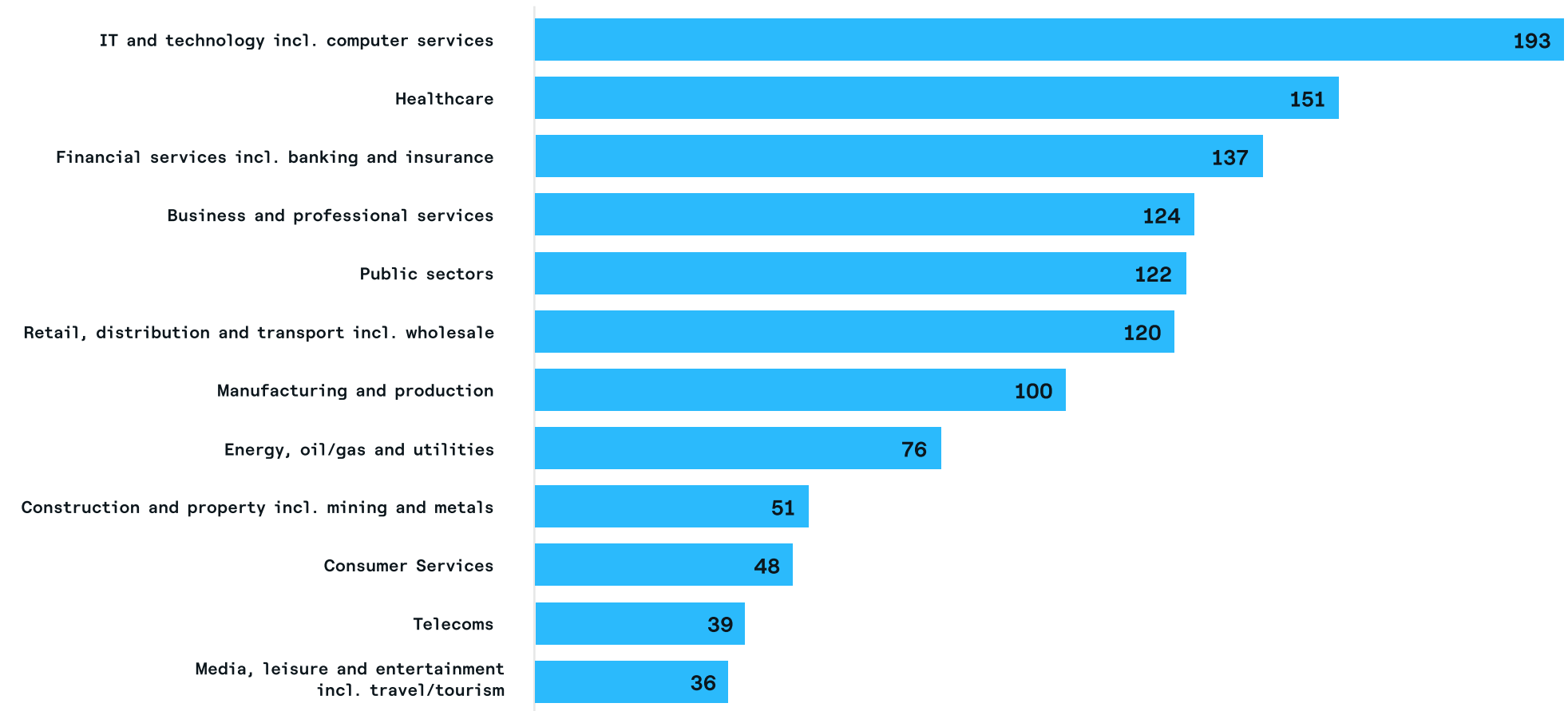
Demographics and Methodology

We funded a third-party research partner to collect 1,200 survey responses in August 2025 from 600 software engineers and 600 senior technology leaders. The respondents were from a variety of different industries, organization sizes and revenue, and had a variety of seniority levels. There were 400 respondents from the United States, 400 from the United Kingdom, 200 from Germany, and 200 from France. Each region's respondents were split evenly between software engineers and technology leaders: 200 software engineers and 200 technology leaders from the United States, 100 software engineers and 100 technology leaders from Germany, etc.

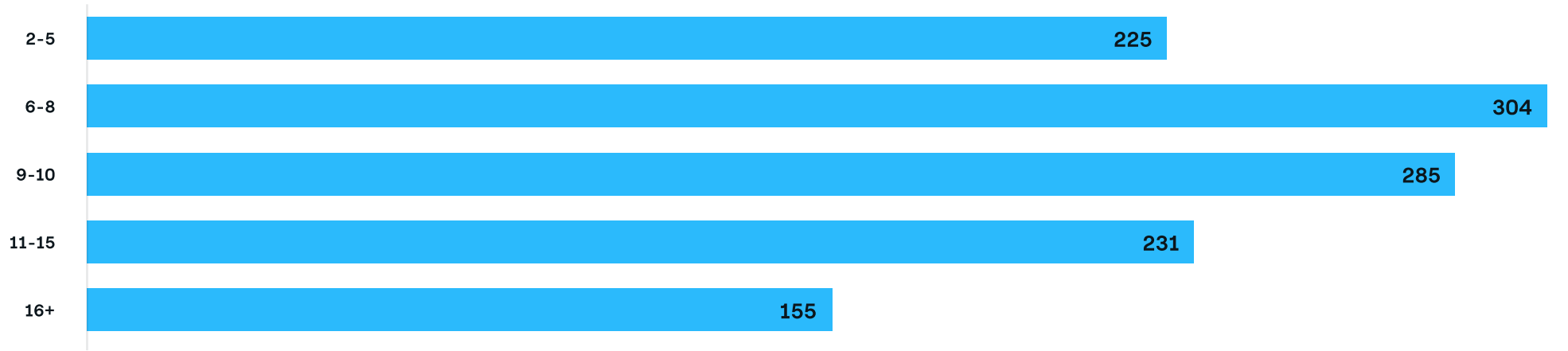
Organization Size



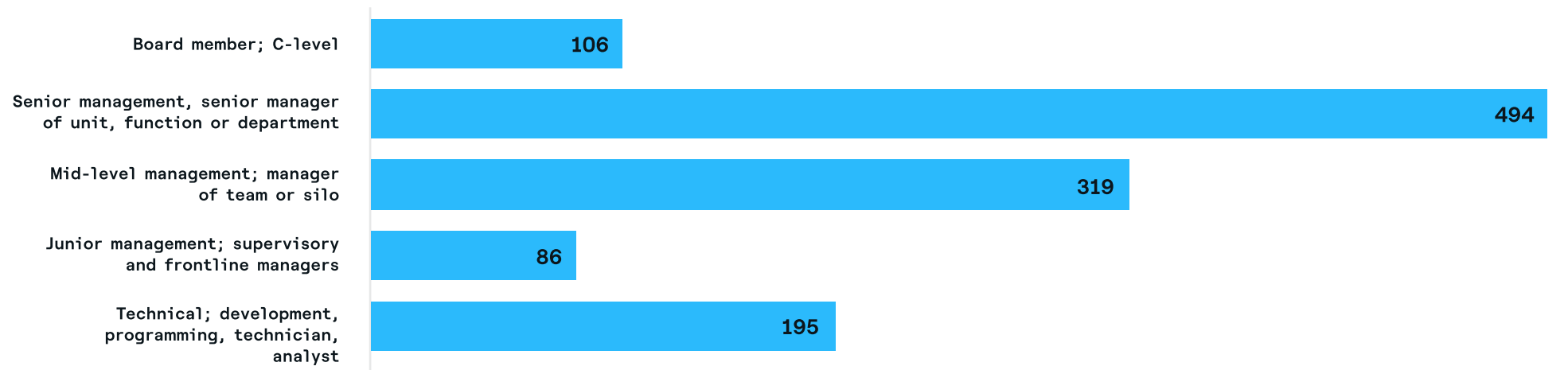
Organization Sector



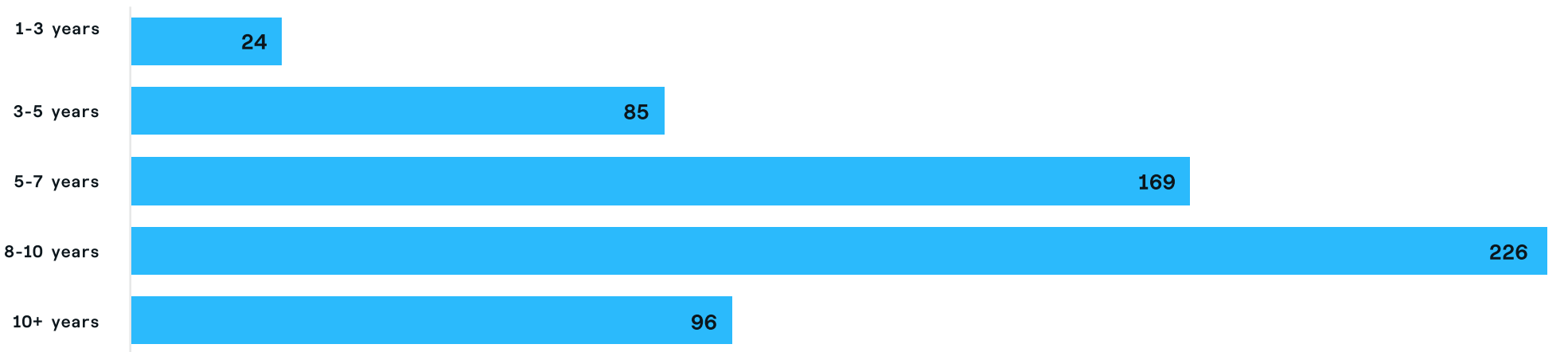
Number of Software Developers/Engineers



Respondent Seniority



Software developer/engineer experience



Organization Revenue