What matters most for embedding AI: **How you adapt**

Moving from AI adoption to AI outcomes: What engineering leaders are doing to get more from their AI tooling.

multitudes

Executive summary

For the last year, we've been following software engineers and engineering leaders through AI rollouts. Our goal was to (1) understand how AI is impacting productivity, quality, and people, as well as (2) identify the strategies that work best for getting the most out of AI as organizations navigate this period of massive change in our industry.

This is the second in a series of whitepapers on our findings. In the first (What matters most for AI rollouts: How you lead), we explored what increases AI adoption (answer: Enablement, or how leaders support their people to use AI) and the impact of AI on productivity and quality.

This second whitepaper focuses on the tactical side – what strategies organizations are using to get more from their AI tooling – as well as emerging changes for the field.

Top findings

1. **For more AI usage, build a learning organization.**
   Learning organizations are ones that have systems that support continuous learning – something we especially need now given the pace of change of AI.

2. **The desired AI outcomes are clear but measuring them is not.**
   While organizations know what they want out of their AI tooling, many aren't measuring outcomes – in part because it's hard to know how to measure them.

3. **Leaders need to adapt processes, not just tooling, to work with AI.**
   Despite the importance of supporting people to get more from AI, the vast majority of the time, people focus on codebase changes rather than people or process ones.

4. **Engineering roles are evolving.**
   Engineers are expected to be more cross-functional, and with non-engineers building more software, there are additional demands on engineers to support these new builders.

Recommendations for leaders

To succeed with AI, this research recommends that you:

- **Build a learning organization** – with dedicated time to learn, regular peer-to-peer learning opportunities, and procurement processes designed for quick tooling decisions and experimentation
- **Choose good-enough metrics for AI adoption and AI outcomes,** and get ahead of your board by tracking progress over time.
- **Ensure that you're adapting your people & processes, not just your codebase** – by considering how your processes (particularly reviews) and engineering roles should adapt.
- **Set guardrails for non-engineers writing software** – from sandboxed environments to guidelines for check-in points with engineers.

About the research

This is the second paper in our AI impact series; the first was titled
What matters most for AI rollouts: How you lead.

In total, this research has spanned:

- 500+ individual contributors (ICs) and 200+ engineering leaders
- 15 months of data – starting in Jan 2025 with telemetry data and ending in March 2026 with 1:1 interviews
- A broad set of data sources, from telemetry data for 500+ engineers to 428 survey responses (across 2 surveys, one deep-dive survey focusing on 4 organizations, and one industry survey covering hundreds of organizations) and 29 interviews

This whitepaper focuses on the survey and interview data. Our survey dataset has 428 responses, with the following demographic breakdowns (Figure 1):

1. 54% individual contributors, 46% leaders
2. 45% of responses from large organizations (with 250+ engineers), 34% from medium-sized ones (20-249 engineers), 21% from small ones (1-19 engineers)

For more detail about each research phase and the demographic groupings, see the appendix.
Also note that this paper uses "AI" to mean "LLMs".

Figure 1: Demographic breakdown of all survey respondents

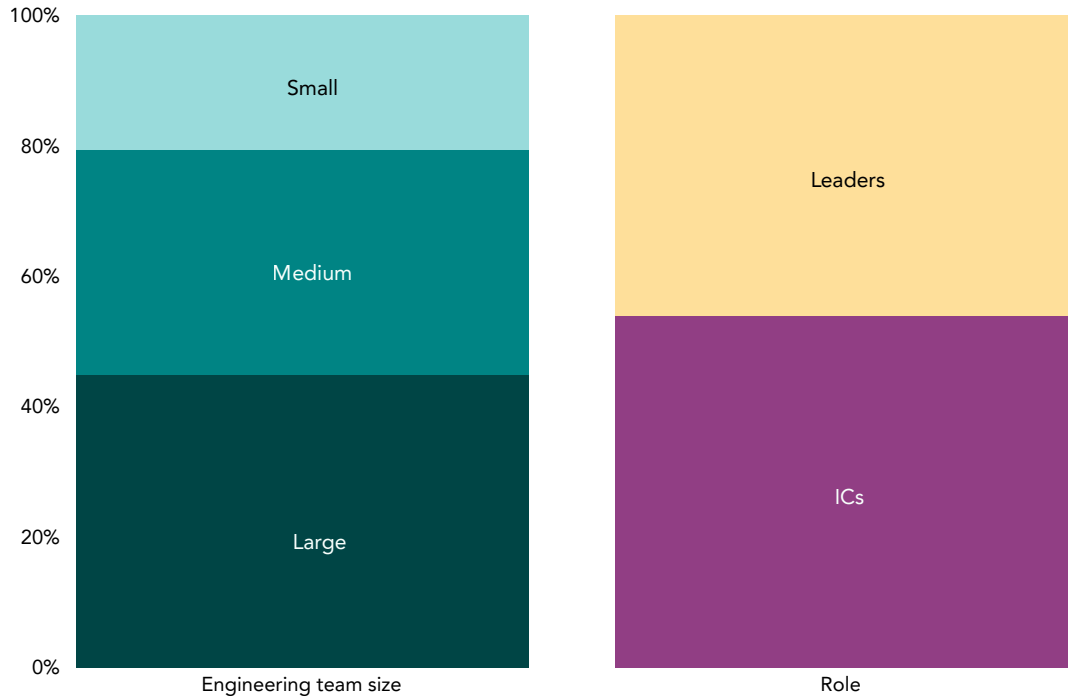

Figure 1: *Demographic breakdown of survey respondents across both the Industry Survey and Deep-dive Survey. N = 428*

A note on model improvements since December 2025

One of the challenges of conducting research on the impact of AI is how quickly the underlying technology is changing. The end of 2025 heralded a particularly big shift in AI technology – with Opus 4.5 launching on November 24, 2025 and GPT 5.2 launching on December 11, 2025. These models both showed marked improvements, particularly for coding use cases. Then Opus 4.6 and GPT-5.3 Codex came out in February 2026 and brought new leaps forward (for a fun summary, check out: WTF happened in 2025?).

We knew that we needed to acknowledge these shifts in this whitepaper. To do so, we conducted ten interviews (5 ICs and 5 leaders) in March 2026 – after the latest model improvements. We asked if there had been changes to how they work since December 2025 and if so, what those changes were; alongside that, we dived into their survey responses. This whitepaper focuses on the trends that were shown in the survey data and validated by these interviews.

Changes since December 2025

1 Companies are moving up the maturity curve.

Companies that hadn't yet rolled out AI tooling to their teams now have. Engineering teams that already had AI tooling are now running agentic workflows. Some are even testing out largely autonomous agents for greenfields projects.

"The big shift since the November models is that these things can do most of the work now. Most of the engineers that I know are agent-first now. They're less likely to touch the code directly. Engineers go around saying, like, hey, what are you still doing in Cursor editing code, just use Claude Code or Codex."
- Head of Engineering

2 Engineers are supporting people outside engineering with LLM usage.

Several engineers and engineering leaders mentioned initiatives in their company to support departments outside of engineering with LLM usage.

"The big thing is that the rest of the company is starting to look at agents. The first pilot project we're doing outside of engineering with agents is lead nurturing outside of our main ICP."
- CTO

"We've definitely seen an explosion of interest from stakeholders across the business that are wanting to get their hands on [coding] tools ... like Claude Code, Codex, or Cursor."
- Security Engineer

3 Claude usage leading to a shift in approved tools.

Claude came up organically in many interviews (note that people often said "Claude" when they meant "Claude Code"). As one Principal Engineer shared, "everyone is Claude-crazy". At some organizations, when leaders found out that ICs were using Claude for personal use, they added it to the approved work tooling list – to not risk leaking company information. Several leaders also shared that Claude was part of a broader push to add more AI coding tools to the approved list.

"People started getting Claude personally, and using it for work stuff. This was our biggest red flag. We needed a policy around this, and we needed an approval process for any additional tooling".
- Director of Engineering

4 Different models provide very different experiences.

The model improvements mean that which model someone is using has a big impact on whether an LLM can do a given task well.

"Whenever someone has an issue with AI tooling, I first ask what model they're on. It's a very different experience on old models – this week, for example, I found out someone was doing a high reasoning task with Haiku; it worked better when I got them to switch to Opus."
- Principal Engineer

Finding 1

For more AI usage, build a learning organization

In our first paper, we found that an organization's practices were the key factor influencing AI adoption. (More here: What matters most for AI rollouts: How you lead.) What worked was (1) sharing the "why" for what the organization was hoping to get from AI usage, and (2) setting clear expectations for what success looks like. We also saw early evidence of the importance of super-users – because engineers want more peer-to-peer learning.

In this whitepaper, we focus on the "what" – the practical actions that organizations are taking to support AI usage. We'll consider what's blocking AI usage, what's accelerating it, and the impact of peer-to-peer as a tool for AI adoption.

Recommendations for leaders

- **Build a learning organization** – carve out time for learning and give yourself time, as a leader, to revisit whether your organization's current practices are speeding you up, or slowing you down.
- **Support peer-to-peer learning.** This could be via Slack/Teams channels for AI sharing, dedicated AI champions, or regular experience-sharing sessions.
- **Work with your procurement teams to speed up approval processes** – through dedicated budgets, easy ways for ICs to ask for tools, and pre-set paths for experimentation.

AI adoption hindered by barriers to learning

The biggest challenges – for both leaders and ICs – all related to keeping up with the changes of AI.

Specifically, we asked survey respondents what they saw as the biggest barriers to AI adoption. This was a free-text question, so the responses came up organically – showing what's top-of-mind for people when they think about AI. In our thematic analysis[1], we saw the following themes (Figure 2):

- **For leaders:** The top barriers were mindset inertia (25% of leaders) and insufficient time for learning (21%).
- **For ICs:** The top barriers were the steep learning curve of AI (20% of ICs), followed by insufficient time to learn (17%).

The first observation is that both leaders and ICs see learning barriers at the top of the list. The pace of change is overwhelming for everyone – which means that moving past mindset inertia, getting time to learn and overcoming the steep AI learning curve are all important for keeping skills sharp. As one Engineering Leader said:

> "It's all pretty overwhelming – it's impossible to keep up."
>
> - Head of Engineering

However, responses diverged around what's slowing organizations down. While leaders were most worried about "mindset inertia", ICs named structural barriers. Compared to leaders, ICs were:

- 4.7x more likely to cite lack of tool availability as a barrier (14% of ICs vs 3% of leaders)
- 2x more likely to point to the steep learning curve of AI (20% of ICs vs 10% of leaders)
- 1.8x more likely to mention security and compliance friction (16% of ICs vs 9% of leaders)
- 40% less likely to cite mindset inertia (15% of ICs versus 25% of leaders).

This shows that leaders were more likely to treat slow adoption as a people issue (as "mindset inertia"), while ICs were more likely to see it as an issue of organizational practices. Some examples of what ICs shared:

Time for learning:

"Having time to adopt a new way of working takes time, and our workload doesn't leave much room to do that."

- Staff Engineer

Steep learning curve:

"I feel that I'm just scratching the surface. I could probably get more out of LLMs if I invested more time, to know better how Claude works, the best way to organize my Skills... However, I'm losing time trying things [with LLMs], and in the end it might not even be the optimal way."

- Senior Software Engineer

Security and compliance friction:

"We have a complicated procurement and governance process. We have some tools, but most we can't connect to anything useful, like MCP servers... We should experiment with what works best for us, but we can't due to governance."

- Staff Engineer

[1]See Appendix for detail on how we ran the thematic analysis

These themes were consistent in our recent interviews – but one thing that has shifted since we ran the survey is the lack of tool availability, which in the interviews seemed to be less of an issue because companies have expanded their tooling lists. As one example, at one organization we spoke to, the policy moved from one coding tool per engineer before December 2025 to two coding tools and two non-coding tools in 2026.

Figure 2: What's been the biggest barrier to encouraging AI adoption on your teams?

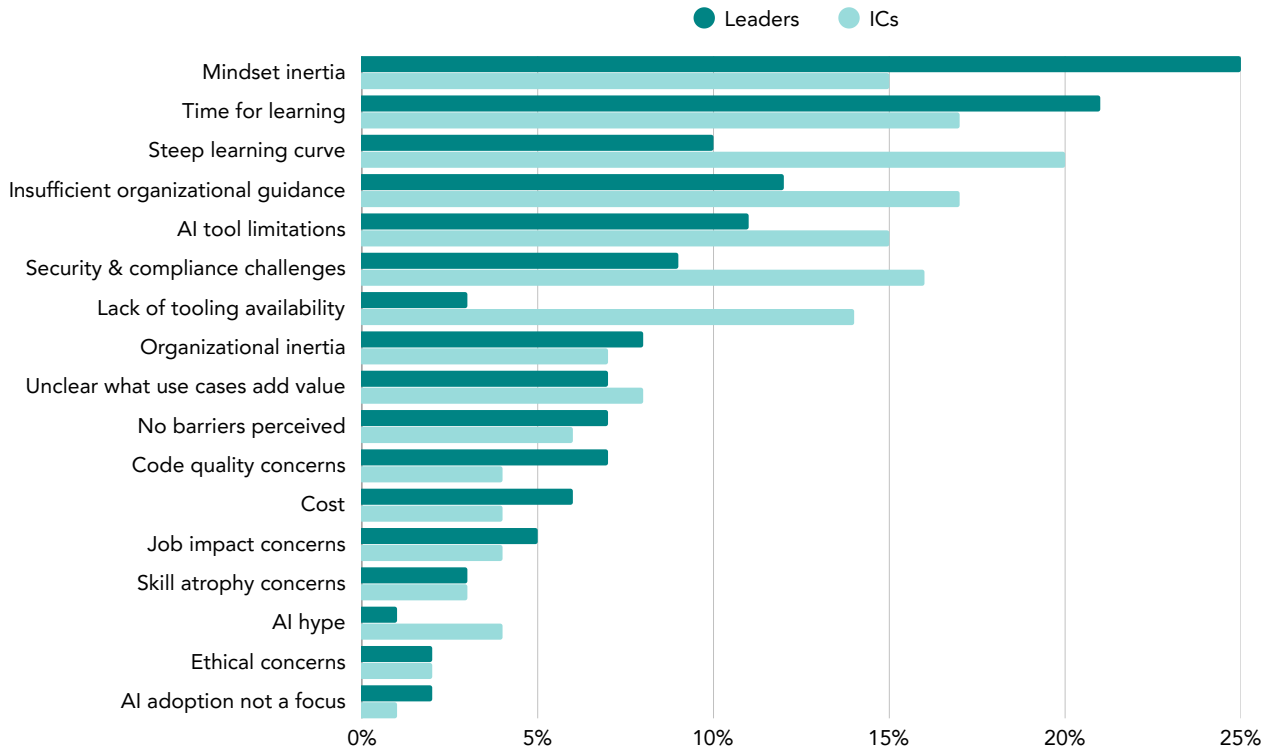

Figure 2: *% of respondents. N = 145 for ICs, N = 150 for Leaders for this chart out of N = 428, Industry Survey and Deep-dive Survey. Free-text responses were grouped into themes; multiple themes possible.*

The rapid pace of change in AI means that organizations need systems in place to support continuous learning. The good news is that there's alignment on this, with both leaders and ICs wanting to reduce the barriers to learning. Even better is that behavioral research shows that beliefs often follow behaviors, so reducing structural barriers to AI tooling (and therefore getting people to use it more) is an easier way to move past the mindset inertia that leaders are worried about. In the section that follows, we dive deeper into what specific practices are supporting AI usage.

Peer-to-peer learning is most effective

Beyond the barriers to adoption, we also asked about accelerants – which initiatives were most effective for increasing AI usage?

To answer this, we asked two questions in our industry survey:

1. Which activities did your organization use as part of the AI rollout? (Multiple-choice question; "other" option to add free-response text.)
2. Which **single** activity was most effective for increasing AI usage? (Single-choice question; "other" option to add free-response text.) (Figure 3)

Figure 3: What activity has been the MOST effective for increasing AI usage?

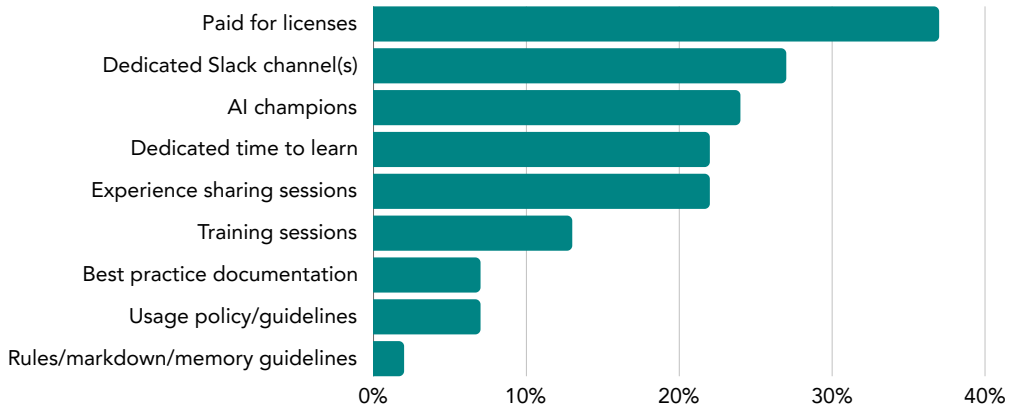

***Figure 3:** % of respondents, out of those whose orgs ran the activity. N = 203 respondents for this chart (orgs using AI tools) out of N = 226, Industry Survey.*

Unsurprisingly, paying for licenses (36%) was the top-rated, given that it is a prerequisite for using AI tooling.

More telling is what followed as the other most effective activities for increasing AI adoption:

- Dedicated Slack channels for sharing AI information (27%)
- AI champions (24%)
- Experience-sharing sessions (22%)

The common thread across all of these is that they are ways for peers to share their learning with others. (We also note that "dedicated time to learn" was next-most-effective activity at 22%, which re-emphasizes the point in the previous section about not having time to learn being a barrier).

Alongside our previous whitepaper findings, we've seen three key reasons why peer-to-peer learning is so effective:

1 Peers know how to make AI tooling work in your own context.

When a peer uses AI on the same codebase, or for a similar task, they've already mitigated concerns about how AI might work for your organization's flavor of complexity.

2 Peers are a better way to keep up with trends.

As one engineering manager said:

"Things change almost daily and it's hard to keep up, so dedicated training is not worth it. Most engineers just want to know how others are using it to get inspiration."

Across a group of peers, there's a better chance that someone has tried the latest model/technique – versus one person or a playbook that gets out of date. Peer-to-peer sharing is also a practical way to address the lack of time for learning mentioned earlier, since it can happen in small chunks throughout the workweek.

3 People trust their peers.

Engineers have stronger trust for their peers, so it means more when they recommend something. This makes peer-to-peer sharing one of the best ways to bring AI skeptics along the journey.

That's why peer-to-peer learning is impactful – but what specifically are people doing to get these outcomes?

Slack/Teams Channels for AI sharing

The most common thing we saw was that a leader often needed to nurture these channels until they reached a critical mass of activity. However, rather than leaders posting themselves, an effective tactic was to encourage others to share their interesting AI work:

> "You've got to deliberately kick it off by getting ICs and early adopters to start conversations."
> - Engineering Lead

Another important theme was encouraging people to share the good and bad of AI tooling – what they liked and didn't:

> "The Slack channel is the cornerstone of our community of practice... It's where people can joke and complain about it. It's a good venue for all the thoughts and feelings people are having about this stuff, and it's where I see the most day-to-day activity."
> - CTO

AI champions

AI champions could be full-time (research participants told us about their "AI engineers" and "AI teams") or part-time, with people doing it in addition to their main role. The key for this was having people opt in to the role rather than a leader assigning it to them.

> "What you want is for people that are keen to tinker with [AI] to figure out what the use cases are, and share it with the wider team as they go. The people that are enthusiastic will lead the way, rather than the people that are being forced to integrate it into their systems when they're not ready for it yet."
> - Engineering Team Lead

Experience-sharing sessions

Experience-sharing sessions work best when they are peer-led and recurring. They could be demos of use cases, pairing sessions, or open discussions on topics voted on by the group.

> "Every week we have a meeting to demo things in engineering. One of the staff engineers was showing us how he used AI to investigate an incident... people wanted to know more – asking, can you share your Skills? Can you share your markdown files?"
> - Senior Software Engineer

> "The topics [for our experience-sharing sessions] fall into 4 categories:
> 1. Success stories: I did this and it was great
> 2. New tool reviews: This looks interesting
> 3. Cautionary Tales: I did this and it was a mistake
> 4. Request for help: I'm wedged here, help plz"
> - CTO

Procurement needs to move at the pace of AI

Last year, we heard something surprising from engineering leaders: They were putting AI tooling – for the company – on their personal credit card.

Curious how widespread this was, we asked about this in our industry survey and found that 33% of leaders had put AI tooling for company usage on their personal credit card. That's not including those using discretionary spend limits on a company card.

This is consistent with the 36% shadow AI usage rate we observed in our deep-dive research, where engineers were using personal accounts on AI tools for work. It makes sense that leaders are responding to the same pressures their teams are feeling – to try the latest tools.

33% of leaders put AI tooling for company usage on their ***personal*** credit card

Of the leaders putting AI tools on their personal credit card for work, 42% explained that procurement friction was the key reason why they were doing so (Figure 4). As one Senior Engineering Manager said, "[I use my personal card] to try new tools, with non-confidential data, without needing the full IT process."

Figure 4: Reason for putting AI tool on personal card

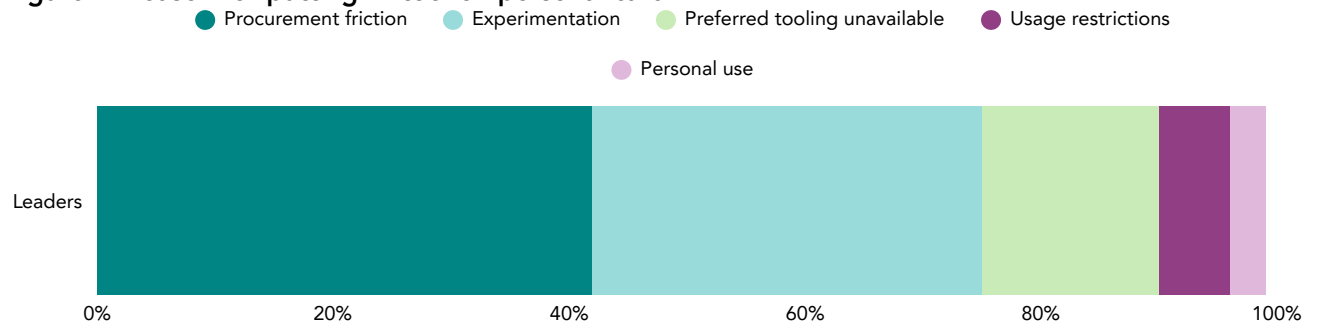

Figure 4: *% of themes. N = 32 for this chart out of N = 226, Industry Survey. Free-text responses were grouped into themes; multiple themes possible.*

To benchmark what good procurement processes look like, we asked in our industry survey how long their most recent AI tool took to receive approval. See the results by organization size in Figure 5.

Figure 5: For your most recent AI tool rolled out, how long did it take to get approval to use it at your organization?

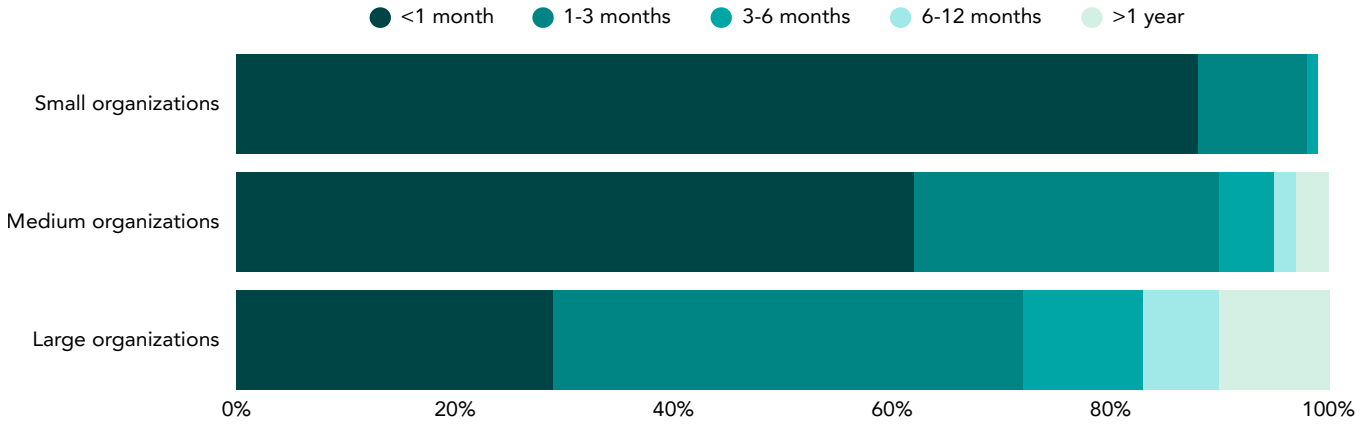

Figure 5: *% of respondents. N = 69 for small orgs, N = 86 for medium orgs, and N = 28 for large orgs for this chart, out of N = 226, Industry Survey*

As expected, smaller organizations are more agile: 88% of small teams approve new tools in under a month, compared to 62% for medium-sized organizations and just 29% for large ones. That said, the fact that almost one-third of large organizations got approval in less than a month shows that it's possible to move quickly even for them.

We also looked at the leaders in the "procurement friction" bucket who were paying for AI tooling at work on their personal card. Surprisingly, almost all had approval times of less than one month, and the longest approval time for anyone in this bucket was 3 months. This shows it's not that their procurement teams were slow. It's that when a promising new AI tool emerges, even a month feels too long. In this environment, "fast" means days, not weeks.

What are companies changing to become more adaptable? Some practices we saw in our interviews:

- **Dedicated budgets:** Dedicated budgets for AI tooling makes approvals faster. Respondents reported AI budgets from $20-$500 USD per person per month.
- **Make it easy to ask:** One company changed their process so that any engineer could raise a tooling request directly with their manager, who would then escalate it. As their VP of Engineering explained, *"it's faster for the engineer to ask their manager, and then all I need to do is approve the budget."* (This company also has dedicated budgets.)
- **Plan for experimentation:** Another company had a "testing protocol" – a process where an engineer can trial several tools for a defined period before committing.

The common theme here is pre-planning for change – and letting engineers drive tooling requests, since they're the ones using these tools day-to-day.

Overall, the key takeaway we draw from this section is the importance of building a learning organization. This term was coined by Peter Senge in his book The Fifth Discipline: A "learning organization" is one that facilitates continuous learning so it can keep evolving. This starts with considering the systems of the organization and how well (or not) they support ongoing adaptation.

Finding 2

The desired AI outcomes are clear but measuring them is not

There are many reasons why organizations are adopting AI – both for the outcomes they hope to achieve and because of pressures from above. As adoption becomes more universal, the conversation will inevitably shift from "are we using AI?" to "what are we getting from AI?" We wanted to understand how prepared engineering leaders are for that shift.

Recommendations for leaders

- **Once your org is clear on the "why" for AI tooling, choose metrics that are a good indicator of your goals.** Most organizations don't have AI OKRs so tracking any metrics is a great start.
- **Done is better than perfect:** Choose good-enough metrics, and then focus on trends over time. Looking at the change in metrics before versus after interventions is also a good way to tease out the impact of your actions instead of confounding factors.
- **Boards are starting to shift their focus from AI adoption to AI outcomes** – get ahead by looking at outcome metrics now.
- **Use metrics and communication to manage up –** including by:
  - Supporting AI learning and development
  - Transparently communicating what's working and not
  - Educating the board to reset expectations as needed

There's a gap between desired AI outcomes and AI metrics

We asked engineering leaders what outcomes they hoped their teams would get from AI tooling. The results were remarkably aligned: 86% want to move faster, 86% want to reduce tedious work, and 64% want to deliver more features (Figure 6). These are concrete, outcome-oriented goals. Given that, we'd expect organizations to be looking at metrics like change lead time (move faster), developer experience (reducing tedious work), or velocity (deliver more features) – but that wasn't what we saw.

Figure 6: What outcomes do you hope your team will get from using AI tooling?

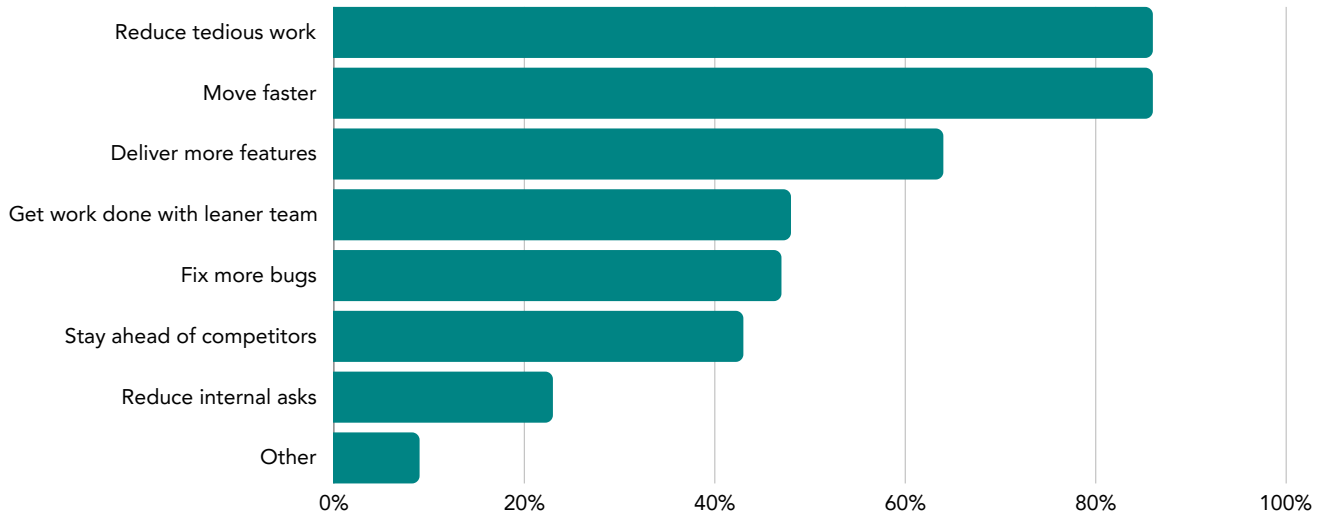

Figure 6: *% of respondents. N = 203 for this chart out of N = 226, Industry Survey*

To understand what metrics people are using, we asked survey respondents whether they had Objectives and Key Results (OKRs) in place to measure the impact of AI tooling. Only 31% reported having AI-related OKRs. As one VP of Engineering told us bluntly when asked about AI metrics: "I don't have any good ones."

Only 31% of respondents reported having AI-related OKRs

Among those who did have AI OKRs, the most common metrics were activity-focused rather than outcome-based: 71% tracked AI feature delivery and 71% tracked AI adoption rates, while only 44% measured velocity and 37% measured speed (Figure 7). The most common metrics show, "Are people using AI?" rather than, "Is AI making us better?".

Figure 7: Do you have OKRs/KPIs related to AI - if so, which ones?

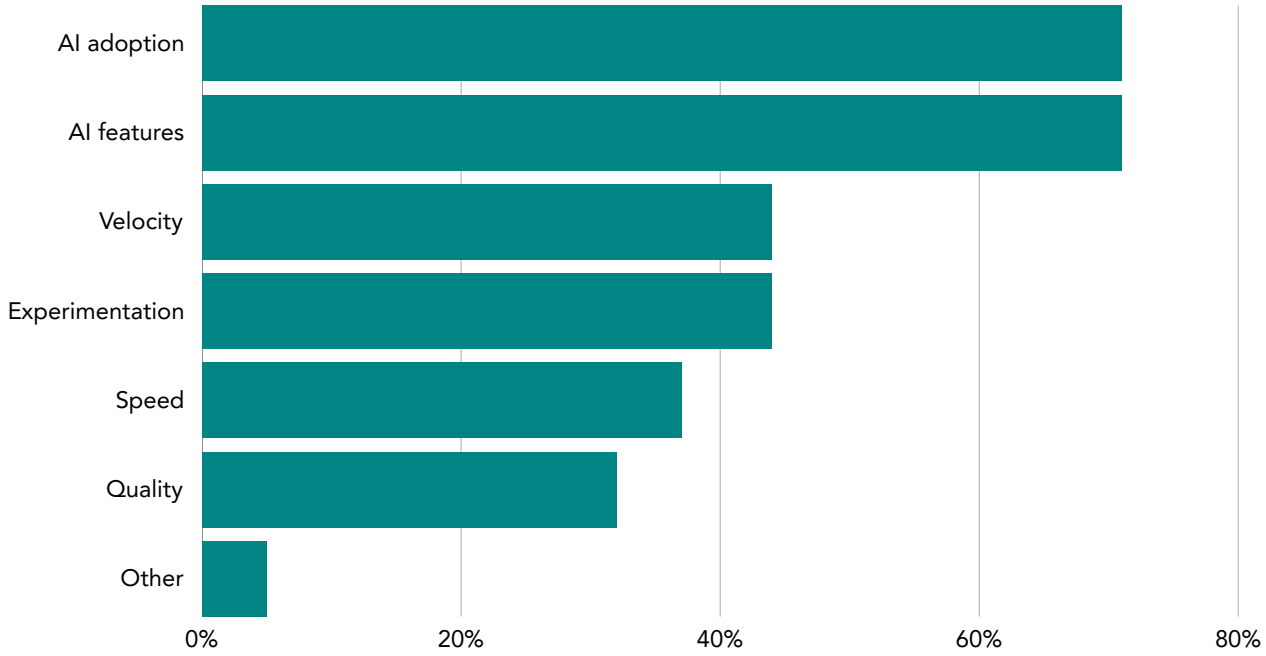

***Figure 7**: % of respondents. N = 63 for this chart out of N = 226, Industry Survey*

The disconnect is worth highlighting. Even though 86% of respondents said they wanted to move faster, only 15% had OKRs related to speed and velocity in place to track whether that was actually happening.

86%
said they wanted
to move faster

Only 15%
had OKRs related to speed and velocity
to track whether that was happening.

This may partly reflect the transition from focusing on AI adoption first and AI outcomes after. We saw this in our interviews. As one leader said:

> "Before, they wanted to know we were doing something. What it's started to turn into is the expectation of something coming out the other side."
>
> - CTO

What people find hard to measure about the impact of AI

Of the respondents with OKRs, 75% had something that they found hard to measure about the impact of AI. The most common challenges were with measuring productivity (26%), capturing human impact (19%), and isolating AI's contribution from other factors (15%) (Figure 8). Responses ranged from not knowing what to measure to knowing what but not knowing how to do it.

Figure 8: What do you find hard to measure about the impact of AI, if anything?

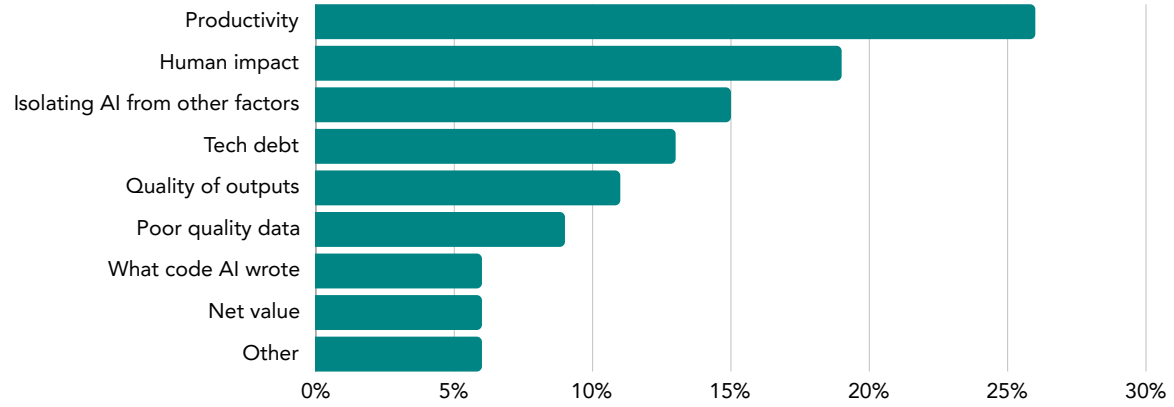

Figure 8: *% of respondents. N = 47 for this chart out of N = 226, Industry Survey. Free-text responses were grouped into themes; multiple themes possible.*

On what respondents found hard to measure about the impact of AI:

Productivity:

"Measuring productivity is not easy – increased PR throughput is not a great measure of the impact of AI."

- Senior Engineering Manager

Human impact:

"We can track velocity, but it's tough to quantify the lift in creativity, morale and strategic thinking that AI frees up time for. Cultural shifts and serendipitous innovation take longer and don't show up in dashboards."

- CTO

Isolating AI's impact:

"You can't measure [AI impact] in isolation. If we get a ticket done quickly, is it because we used AI for writing code and reviews, or is it because we collaborated well from the start and throughout, which results in less rework?"

- Software Engineer

What stands out is the range of things that are difficult – from the perennial challenge of measuring software engineering productivity to broader questions about tracking human impact and cultural change.

On the question of isolating AI's impact from other factors, we can at least suggest some approaches that are methodologically stronger. Pre/post comparisons around a specific intervention are more reliable than simply comparing high and low AI adopters, because that introduces confounding factors. Plus, in a world where nearly everyone is using AI, there will be fewer low adopters to compare against and even more selection bias. See this article for more about lightweight techniques for better measuring the impact of AI.

Overall, we suggest that it matters less what specific metric leaders choose, as long as they're logical and linked to the desired outcomes. The key is to choose something that's feasible to measure and then track trends over time.

Metrics as a tool for upward management

We also wanted to explore the top-down pressures leaders are facing and how they're navigating them.

We asked a series of questions about whether respondents were facing board-level pressure regarding AI, and if so, what form it took. The first finding: 55% of all industry survey respondents said they were facing pressure from the board regarding AI.

55% experience board pressure around AI

> "At the leadership level, there is definitely an appetite [for AI]. Even though they don't understand it, it's a trend they must follow. And they're looking for quick results."
> - Engineering Lead

When we asked leaders what those pressures looked like (Figure 9), the top response was around AI adoption (40%) – consistent with the phase we've been in. But the second-highest was expectations for improved outcomes (39%) – a signal of where things are headed. Among those citing outcome expectations, the vast majority (60%) named productivity as the desired result. We expect these pressures to grow, particularly around demonstrating outcomes and ROI.

Figure 9: What pressures are you facing from the board/execs?

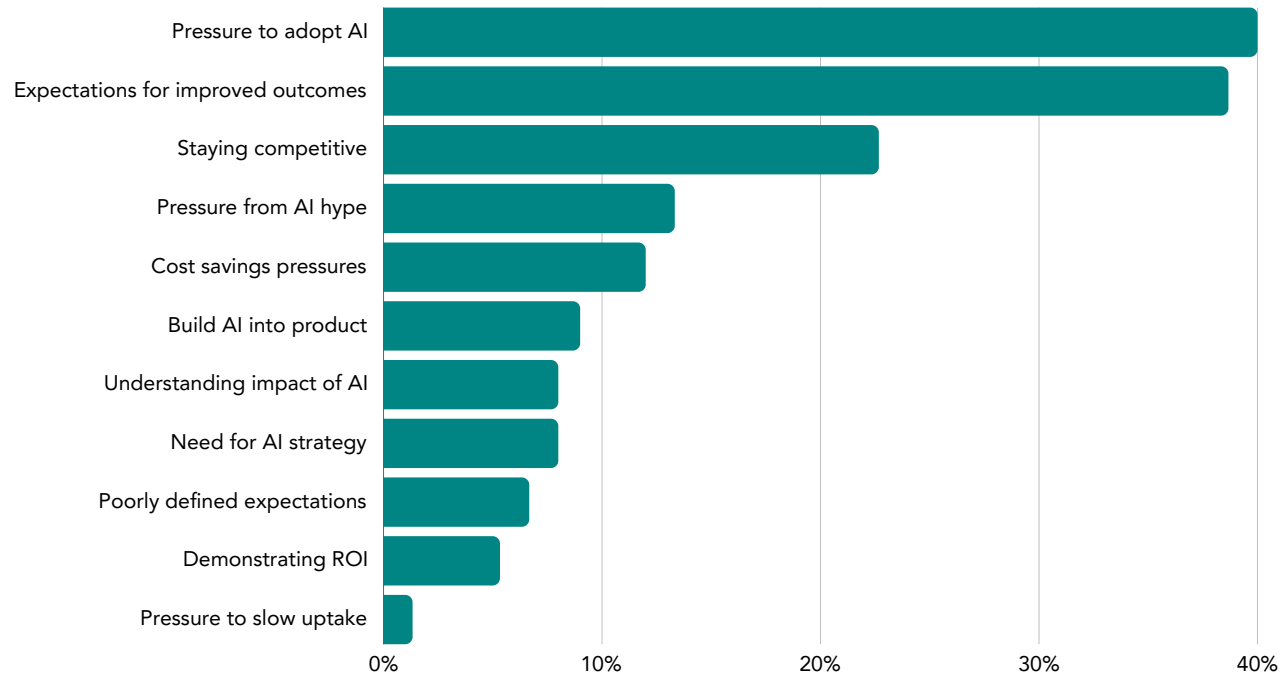

Figure 9: *% of respondents. N = 75 (Leaders only) for this chart out of N = 226, Industry Survey. Free-text responses were grouped into themes; multiple themes possible.*

The next question is what engineering leaders are doing to manage up. Figure 10 shares more.
The top responses were:

- Investing more resources and effort into AI learning & development (22%)
- Transparently communicating both the good and bad (18%)
- Resetting board expectations via education (18%)

Figure 10: How are you managing pressures from the board/execs?

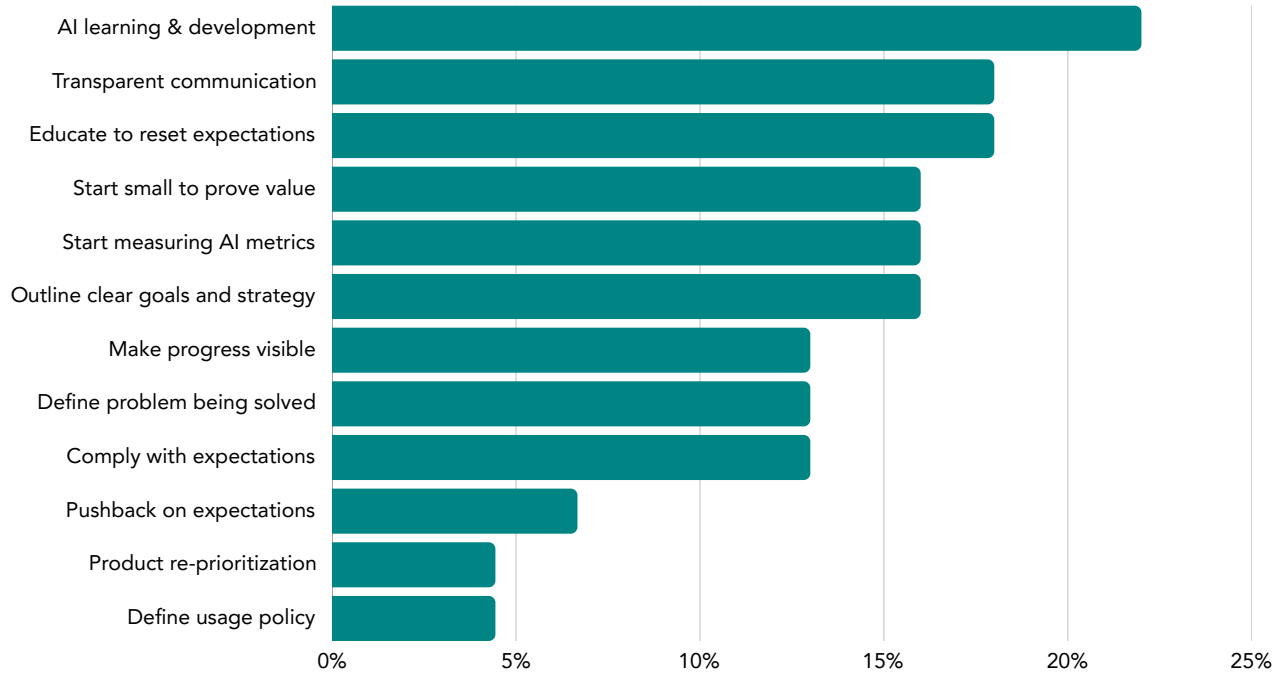

Figure 10: *% of respondents. N = 45 (Leaders only) for this chart out of N = 226, Industry Survey. Free-text responses were grouped into themes; multiple themes possible.*

Some examples of what people said:

Transparently communicating the good and bad:

"[The board expects] a 50% productivity improvement across my team, but I'm not going to get that – so I need a quantification of what I am getting right now."

- CTO

Investing into AI learning & development:

"[The board] expected that with the use of AI tools, delivery and developer productivity should increase. [We're managing this by ensuring] all our team members have hands-on experience with AI. My focus is on learning and development."

- Senior Engineering Manager

Resetting expectations via education:

"As part of our board's AI-first mandate, a significant portion of my time is now dedicated to building foundational AI literacy across our board, executive team and non-technical leadership, ensuring alignment on AI's viable forms, financial implications, and governance."

- VP of Engineering

These are great recommendations for how to guide the board and get ahead of the AI pressures. Boards and execs are still figuring out how best to measure AI impact, so establishing and presenting what you want to measure gives you a chance to shape the narrative.

Finding 3

Leaders need to adapt processes, not just tooling, to work with AI.

As we've seen throughout this paper, adaptation and learning are the name of the game given the pace of change in AI. In this section, we explore what changes people are making to get more from their AI tooling.

Recommendations for leaders

- **If you make any changes to work better with AI, you're ahead of most organizations.** Consider what changes you want to make.
- Despite the importance of people and process changes, most leaders focus first on codebase changes. **We see an opportunity to get more from AI tooling by shifting more focus to people and process.**
- Some people and process changes you might consider:
  - **Reconsidering the role of engineers** – in particular, having them do more product work
  - **Stricter human code reviews** – consider where you need more human input into reviews and if or where AI-only reviews would be appropriate (e.g., for lower-risk changes or in a less-complex part of the codebase)

Changes to work better with AI

We asked survey respondents whether they had made changes to work better with AI tooling – and whether they had made changes to protect their codebase quality from AI's potential impact. The first insight: Most organizations haven't made any changes. Only 41% had made changes to work better with AI tooling, and only 40% had made changes to protect codebase quality.

41%
made changes to work better with AI

40%
made changes to protect codebase quality

For some, this may reflect a view that sound engineering practices haven't fundamentally changed. As one principal engineer put it:

"All the things that we're doing around Claude are the things that we've been talking about for the last 3 years. So we're talking about cutting down PR sizes, doing small chunks of work, test-driven development, small changes, and all we're really doing is trying to accelerate the bit in the middle [with writing code]."

Among those who did make changes, we saw a clear pattern: the vast majority were codebase-focused. Codebase changes accounted for 85% of changes made to work better with AI, and 63% of changes made to protect quality (Figure 11).

Figure 11: Organizational changes to work better with AI

Figure 11: *% of themes. N = 69 for top bar, N = 61 for bottom bar out of N = 226, Industry Survey. Free-text responses were grouped into themes; multiple themes possible.*

This pattern held in our interviews too. Almost without fail, leaders spoke about codebase changes first, and only surfaced people and process changes when we specifically prompted them.

Figures 12 and 13 show the types of changes respondents are making.

Figure 12: Changes so that your people/codebase work better with your AI tooling

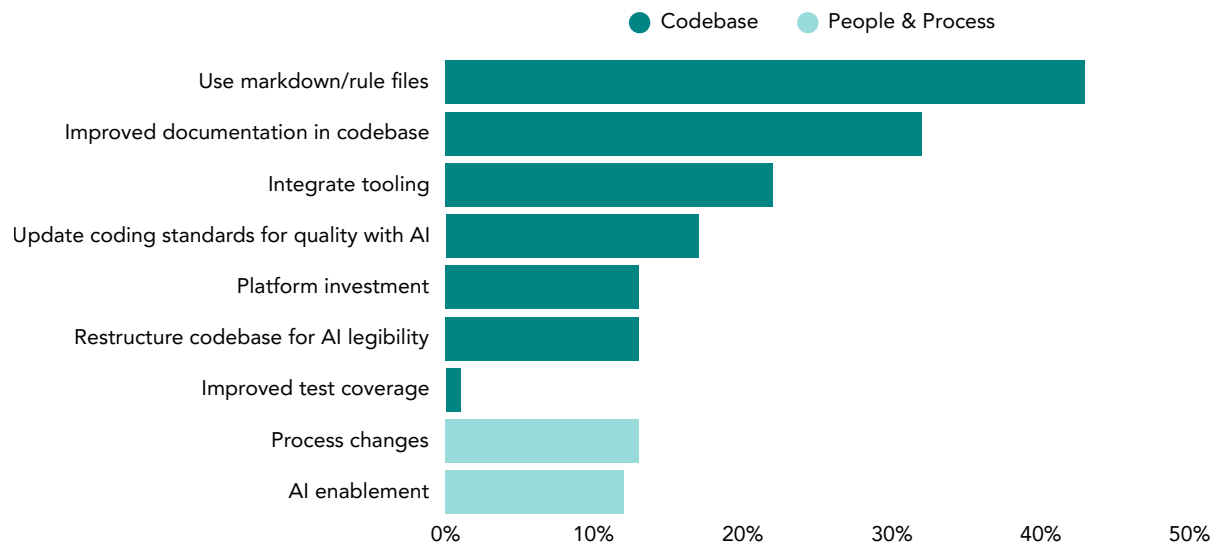

Figure 12: *% of respondents. N = 69 for this chart out of N = 226, Industry Survey. Free-text responses were grouped into themes; multiple themes possible.*

Changes to work better with AI tooling (Figure 12)
The top changes were implementing markdown files (43%), relocating documentation into codebases for AI-searchability (32%), and integrating AI tooling into existing workflows and development infrastructure (22%).

On the documentation side, one CTO described having "developed a global context repository" which contains markdown documentation that explains their business, domain, services and interactions to coding agents.

> "We started thinking about developer experience for agents so have tried to put more pressure into moving our old, slowly grown codebase more towards modern standards and conventions that models are trained on better."
> - Engineering Manager

Not many people mentioned any changes to people or process – process changes were the most common theme, though only mentioned by 13% of respondents. Most process changes focused on stricter enforcement of norms around PR sizes, writing good documentation, and following coding standards. One significant process change was around redefining the role of engineering. An Engineering Manager at a medium-sized organization shared that,

"we have bumped developers up to product engineers, giving them more end-to-end responsibilities. With AI managing boilerplate, refactoring, and code review, they have more time to deliver features, and it makes sense to give them the ownership and tools to do so."

Changes to protect codebase quality (Figure 13)
To protect quality, the most common change was codebase-related: 39% of respondents have increased test coverage.

> "We're measuring and increasing our test code coverage. We've also added linters and AI self reviews before a peer PR review."
> - CTO

The second most common was a people & process change: 31% of people have brought in stricter human code review processes.

> "Two code reviews are required for any code that used AI tools to be written."
> - Tech Lead (Multiple said this)

> "We're doing more pairing on code reviews, which has made it easier to review AI-generated code... Reviewing synchronously has been the most efficient way for us to get through bigger pull requests, and also keeps both people context-loaded on all the projects their team is working on."
> - Engineering Manager

Figure 13: Changes made to protect the quality of your codebase

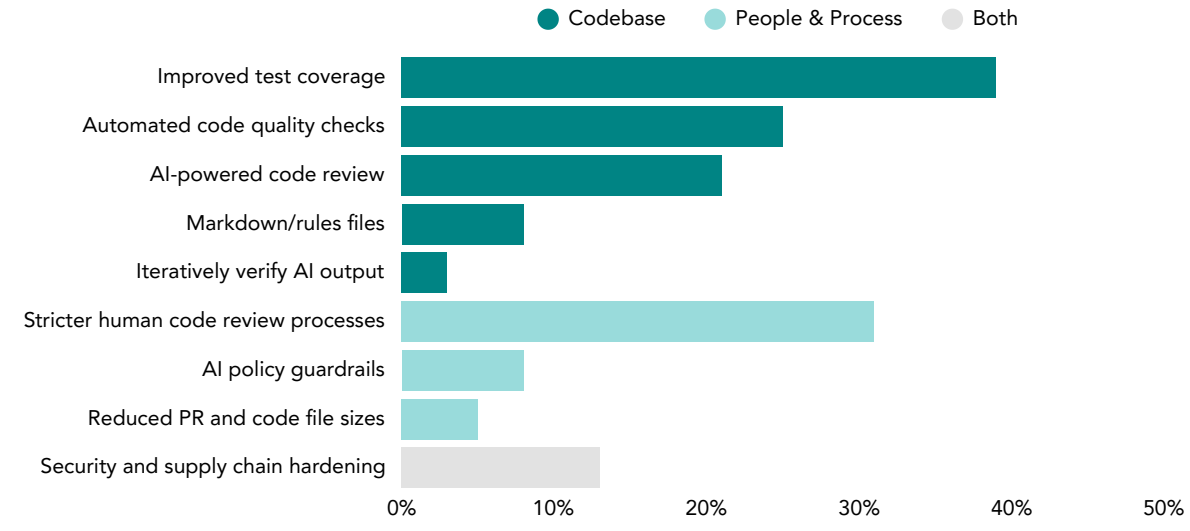

Figure 13: *% of respondents. N = 61 for this chart out of N = 226, Industry Survey. Free-text responses were grouped into themes; multiple themes possible.*

The focus on codebase changes isn't surprising since they're more discrete (you can update a markdown file in a matter of minutes), whereas shifting how humans work can take weeks. However, given our findings that AI enablement is number one for supporting adoption, and what we've seen in this paper about the importance of building a learning organization, it's likely that organizations are leaving value on the table by underinvesting in people and process.

AI doing code reviews

In our first whitepaper, we found that good code review practices can prevent AI quality issues. The industry survey we've shared in this whitepaper validates that – the data above shows that the top people and process change to protect code quality was stricter human code reviews. This also came up in industry interviews, with some people seeing code reviews as more critical than ever. As Principal Engineer said
"when we give Claude to all of the engineers, we're just gonna hit another bottleneck, and that bottleneck is going to be code review."

Organizations are responding in different ways. At one company, after seeing more AI-related incidents, they introduced a new rule: During their first six months in the company, engineers could not approve a PR for merge. The reasoning was that longer-tenured engineers would know the coding standards and codebase architecture well enough to catch risks in AI-generated code.

An engineering leader at another company continues to express the importance of keeping humans in the loop:

> "There's no one in our team who wants to trust AI 100% for generated code yet. They're still at that point where the human needs to be the approver."
> - Engineering Lead

For others, they're looking to remove humans from part or all of the process.

> "We're evaluating if we can find a change to our SDLC that will let us drop some of the core review requirements – so we could release to prod without human reviews, but keep human reviews before something is GA."
> - CTO

> "Now everyone can write code with AI; does every single line that goes into a codebase need to be reviewed by a human? I don't think that's sustainable. We're pushing pretty hard into how we can leverage agents to review code, with our standards in mind – whether it be performance, security, or quality."
> - Head of Platforms

Across all these examples, the common theme is that AI-generated code is making review volumes unsustainable under existing processes. The strategies differ – some organizations are pushing for smaller code changes with the human author holding responsibility for quality, while others are pushing to have AI review AI's work. Most organizations will likely land somewhere in the middle: AI reviews to handle routine or low-risk changes, and human reviews concentrated on changes with the greatest risk. We expect that the specifics will depend on the organization's risk appetite and codebase complexity.

Finding 4

Engineering roles are evolving

Throughout the surveys and interviews, a big theme came out around the uncertainty and stress of living through this period of change – alongside the excitement and optimism, of course. Despite the hype, the worries are real.

One senior IC spoke about the pressure to keep up with AI – saying *"if you don't use it, you will be behind"*. Another senior engineer said he felt "lazy" using AI, worrying, *"It works, but am I shipping good code? Is it something that future engineers – or even I – will understand later?"*

Others spoke about the stress of what it means for jobs. With the "AI-washing" happening with layoffs (with leaders using AI as an excuse for over-hiring or other issues), one engineer pointed out that even if the original layoffs weren't really about AI, the narrative still does damage – because "now you can see everybody else in the industry twitching."

With that in mind, let's look at what people said about how they see AI impacting engineering teams in the future.

Recommendations for leaders

- For organizations hiring, there might be an opportunity in hiring juniors – **because they're AI-native, it's faster to onboard someone with AI, and there's less competition for junior talent**
- **Consider how engineering will need to support non-engineering groups as they start writing more code, and even doing changes to production.** This includes:
  - Sandboxed environments for them to use AI in
  - Defined check-in points with engineers – (1) after creating an implementation plan but before building and (2) as a review once the code has been written

It's not all doom and gloom for junior engineers

One of the most persistent concerns that surfaced across our research – from junior and experienced engineers alike – was about the future of junior roles.

The overall trend gives reason to worry. Junior hiring across the technology sector has declined since 2022, though it remains unclear how much of this is attributable to AI (a Stanford study says yes) versus the broader economic downturn (economic recessions are linked to a decrease in junior hiring).

What is clear is that the perception that AI can replace juniors is real, and it's shaping decisions, both from organizations cutting internships (technology internships have declined 30% since 2023) and from prospective engineers moving from computer science degrees to others (as one example, Princeton has seen a decline in the number of computer science majors over the last two years).

We surveyed leaders about this in our industry survey. 37% of respondents with AI tools said AI had changed how their organization thinks about hiring/working with juniors, and only 25% were hiring junior engineers (Figure 14).

37% of leaders said AI had changed how their organization thinks about working with juniors

Figure 14: AI impact on junior engineers

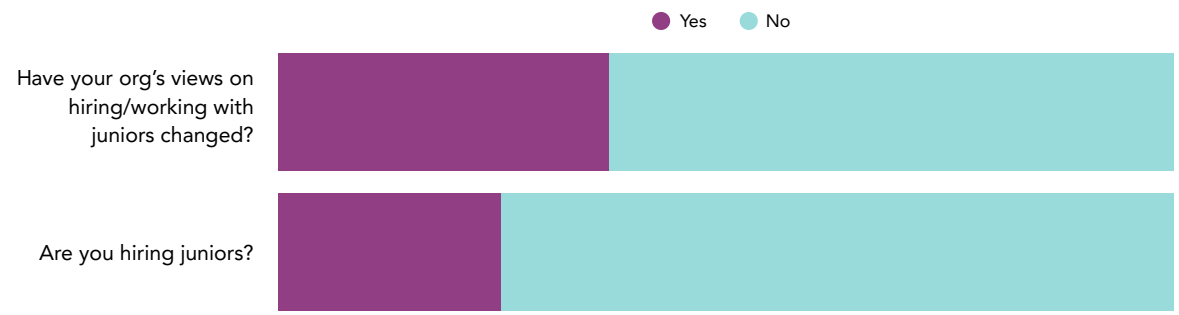

Figure 14: *% of respondents. N = 198 for top bar, N = 203 for bottom bar, Industry Survey*

For those who said that AI had changed how their organization thinks about working with juniors, decreased junior hiring was the most common theme they shared (47% of themes expressed by ICs and 25% of Leaders) (Figure 15). This validates the concern that AI is reducing junior hiring. Beyond this shared recognition, leaders and ICs diverged.

Figure 15: How has your organization's views on hiring/working with juniors changed?

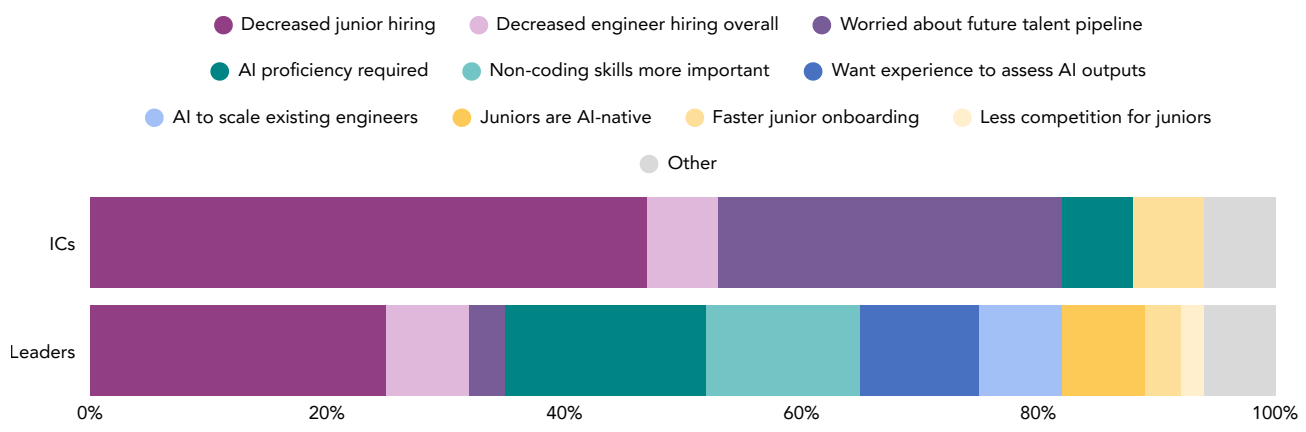

Figure 15: *% of themes. N = 14 for ICs, N = 60 for Leaders for this chart out of N = 226, Industry Survey. Free-text responses were grouped into themes; multiple themes possible.*

For ICs, the next-biggest focus is the future. Nearly a third (29%) of themes expressed by ICs are on concern about the future talent pipeline – the worry being that if organizations stop hiring juniors now, the industry will face a shortage of experienced engineers in the future. In stark contrast, this was only 3% of leader responses.

> "We need industry citizenship - everyone has to start somewhere. If you don't have opportunities, then talent will dry up."
> - Software Engineer

In contrast, leaders are focused on present-day implications. Their second-largest bucket (after "decreased hiring", with 30% of responses) was around how hiring criteria have changed since AI was rolled out – with AI proficiency (17%) and non-coding skills (13%) becoming more important. The third-largest bucket (with 17% total) was about why they aren't hiring juniors – because they want experience to assess AI output (10%) and are using AI to scale existing engineers (7%). Neither of these buckets registered much for ICs.

One bright spot is that some leaders saw junior hiring as an advantage because juniors are AI-native (7%), onboarding juniors is faster now with AI (3%), and there's less competition for juniors (2%).

Putting it all together, we can see a redefinition of what a junior engineering role looks like. The emerging expectation is that junior engineers need to develop AI proficiency as well as skills in other areas beyond software engineering development, including product and systems thinking.

> "We have changed our hiring process to encourage and require AI usage, but also to enable us to validate capability fundamentals are still there without AI usage. There is a greater focus on product and systems thinking than technical capability."
> - Head of Engineering

The broader implication for engineering leaders is that the junior hiring question is not simply a headcount decision – it's a talent pipeline decision with long-term consequences. Organizations that stop investing in junior talent today may find themselves facing a senior engineer shortage later. For organizations hiring, there might be an opportunity to benefit from bringing in AI-native juniors.

More people are building software

The evolving expectations for junior engineers are a prelude to a broader transformation of what software engineering roles look like. As AI makes writing code easier, more people in organizations are building software. We wanted to understand what that redistribution looks like in practice.

In our industry survey, we asked respondents at organizations with AI tools whether non-engineers were writing more code. Half said yes.

50% of organizations have seen an increase in non-engineers writing code

We followed up to ask who these people were and what they were writing code for.

In terms of which non-engineering groups are writing code, product roles were the most common by far, at 56%. Other roles (like operations, GTM, and design) were less than half as common (Figure 16).

In terms of use cases, people were mostly using these tools for prototyping (47%) and building internal tools (29%). Interestingly, the third-most common use case was making codebase changes, at 17% (Figure 17).

Figure 16: Who is writing more code?

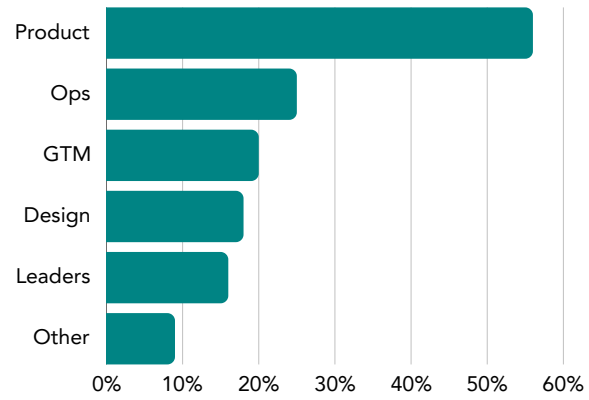

Figure 16: *% of respondents. N = 85 for this chart out of N = 226, Industry Survey. Free-text responses were grouped into themes; multiple themes possible.*

Figure 17: What are non-engineers writing code for?

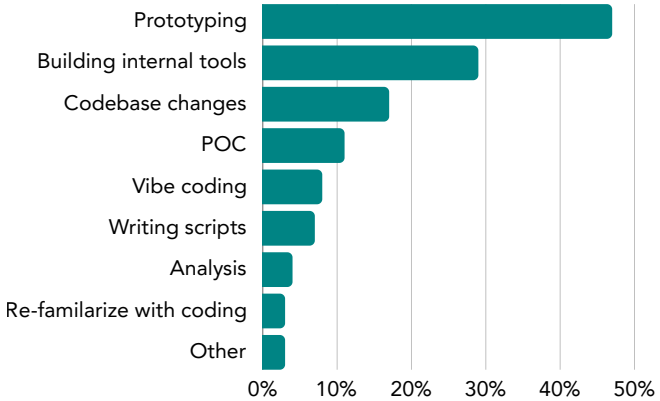

Figure 17: *% of respondents. N = 72 for this chart out of N = 226, Industry Survey. Free-text responses were grouped into themes; multiple themes possible.*

These results suggest product and engineering work are converging. We saw in the survey questions above that juniors are being expected to understand product and systems thinking; conversely, product people are now moving into more engineering work. Many have said that roles are becoming more cross-functional; our data confirms that.

"It's a tragic era for people who got into this field because they love the act of writing code. I think if that's the main reason that you're in this field, you're kind of cooked. Because that era is gone. On the flip side, I think it's a great time to be somebody who likes to build things and turn ideas into something tangible. You can do that at a velocity that is unprecedented."

- Head of Engineering

That said, non-engineers aren't at a place where they are making production code changes without engineering input. In one organization, an engineer shared that their Head of Design had coded up a feature to ship in production:

"Our Head of Design shipped a feature with Claude, which worked but it was overcomplicated as it tried to fit this new feature into the garbage that was already there. So [an experienced developer] stepped in and refactored the code base which made it a lot easier for the model to implement."

- Head of Engineering

The implication for engineering teams is that support requests will only increase from other parts of the business – it's no longer just, "can you build this for me?", it's "can you review what I built?", with the implicit need for engineering to maintain this new work too.

Engineering leaders can get ahead of this by proactively identifying the common use cases for other departments and being clear about the kind of input others should get from engineers when they're using AI tools to write code. One security engineer shared their process for a recent code change they made:

- Use AI to help generate a plan
- Check the implementation plan with an engineer
- Use AI to implement the plan
- Have an engineer do a code review

"We've been working to put together coding sandboxes so people can use [AI] tools [safely]. If they install something malicious or something goes wrong, it's not going to affect their main host."

- Security Engineer

Interestingly, even while organizations pull back on human reviews for some types of code changes, we assert that human input is more important than ever here, both before and after the code gets written, to support knowledge-sharing and maintainability.

From AI adoption to ongoing adaptation

We're moving from a period where success with AI was using AI tooling and into a period where we need to get the desired outcomes with these tools. Where some might have hoped that the pace of change would slow down, if anything, it seems to be speeding up.

This means that from a strategic perspective, we need to be planning for change – planning that there will be a steady stream of new models to learn, new tooling to try, new practices to understand, and shifts in the roles of our engineers and non-engineers.

What this period requires from leaders is to adapt our processes and support our people through change. It's still overwhelming and there's still a lot of uncertainty about what the future holds – but at a minimum, we can help each other (our teams, our organizations, and our industry) figure this out. You've got this.

Appendix

Data and methodology notes

AI usage

This whitepaper was written by humans – specifically, Lauren and Youxiang – with input from the wonderful people named in our acknowledgement. We also did not use AI to generate any of the images you see here.

When we did use AI, it was with human oversight. We used AI:

1. For our thematic analysis for the qualitative survey data; see "Methodology for Thematic Analysis" for more.
2. Occasionally if we were stuck on a word or phrasing, to get suggestions – but all AI wording suggestions were reviewed and written into this paper by a human.

If you're curious about why we added this note, check out The /ai 'manifesto'. (Hat tip to Cassidy Williams for introducing us to this concept.)

About the research

Phase 1 Deep-dive research

Our previous AI impact whitepaper, titled What matters most for AI rollouts: How you lead, was the first in this series, based on following 500+ software engineers at 4 organizations for 10 months through AI rollouts.

Phase 2 Industry breadth

The second phase of our research brings breadth, with new survey and interview insights from hundreds of engineering leaders around the world. This research focused on four key areas:

1. **Rollout activities:** What organizations have been doing as part of their AI rollouts, where are they seeing the most success and challenges.
2. **Goals with AI and measurement:** What outcomes are organizations looking for and how they are tracking progress.
3. **Outcomes with AI:** What are engineers' experiences with these tools to date, and what organizations can do to improve outcomes.
4. **Future of AI:** How AI is shifting hiring, the role of the engineer, and more.

This whitepaper focuses on survey and interview data, collected in 2 phases of research:

- **Phase 1 – Deep-dive research:** Data from 4 organizations, collected over 10 months
  - Survey: 202 responses collected from four organizations over the period July - October 2025
  - Follow-up Interviews: 19 interviews conducted from Aug 2025 to Oct 2025
- **Phase 2 – Industry research:** Data from hundreds of organizations, collected over 6 months
  - Survey: 226 responses collected over the period October 2025 - February 2026
  - Follow-up Interviews: 10 interviews conducted in March 2026

Demographic groups

Organization size

We grouped based on the number of engineers in the company. For deep-dive survey respondents, we knew their organization size from internal company data. For industry survey respondents, we used their self-reported answer to the question "How many engineers work at your company or organization?".

Based on this data, we grouped respondents into three tiers:

- Small organizations: 1-19 engineers
- Medium organizations: 20-249 engineers
- Large organizations: 250+ engineers

Respondent role type

Respondents across both surveys were also categorised into two role types, leaders or individual contributors (ICs). How we grouped roles into each bucket:

- Leaders: Managers of ICs, Engineering Managers, Directors, Vice Presidents, and Executive or C-suite roles.
- ICs: Software engineers of all levels from Associate to Principal, as well as Tech Leads and Analysts.

Methodology for Thematic Analysis

For open-ended survey responses, we used thematic analysis informed by Braun and Clarke's framework and LLM-based thematic analysis approaches. Themes were developed iteratively by the research team through multiple rounds of reading and coding across both datasets. GenAI was used as a tool to help map themes. All AI-generated codings were reviewed and corrected by a human researcher.

Our process was:

1. A human reviewed all responses and developed an initial set of themes.
2. The human prompted an LLM to build its own list, using the initial set of themes developed by the human as a guide in the prompt.
3. We used an LLM to classify each of the responses based on the themes list from step 2.
4. A human reviewed the LLM classifications, making changes to the themes and classifications as needed.
5. If the human updated the list of themes in step 4, we then re-started from step 3. We repeated rounds of LLM and human review until there were no more changes.

To avoid the risks of context-loss and lack of interpretive depth, the development of themes was front-loaded by humans, with human validation at every step involving LLM outputs.

Authors

Lauren Peate
Founder & CEO,
Multitudes

Youxiang Lei
Data,
Multitudes

Acknowledgements

We're grateful for the feedback and support provided by **Dr. Brittany Johnson-Matthews** (Assistant Professor, George Mason University), **Dr. Kelly Blincoe** (Associate Professor of Software Engineering, University of Auckland), **Nathen Harvey** (DORA Lead, Google Cloud), **Dr. Thomas Fritz** (Associate Professor, University of Zurich), and **Dr. Vivek Katial** (co-author of our first AI impact whitepaper).

For more about our research and academic relationships, please visit www.multitudes.com/research.

multitudes

multitudes.com