augment code

The State of AI-Native Engineering in 2026:

[EXCITED] [ANXIOUS] [INVIGORATED]

BY VINAY PERNETI AND EMMA WEBB

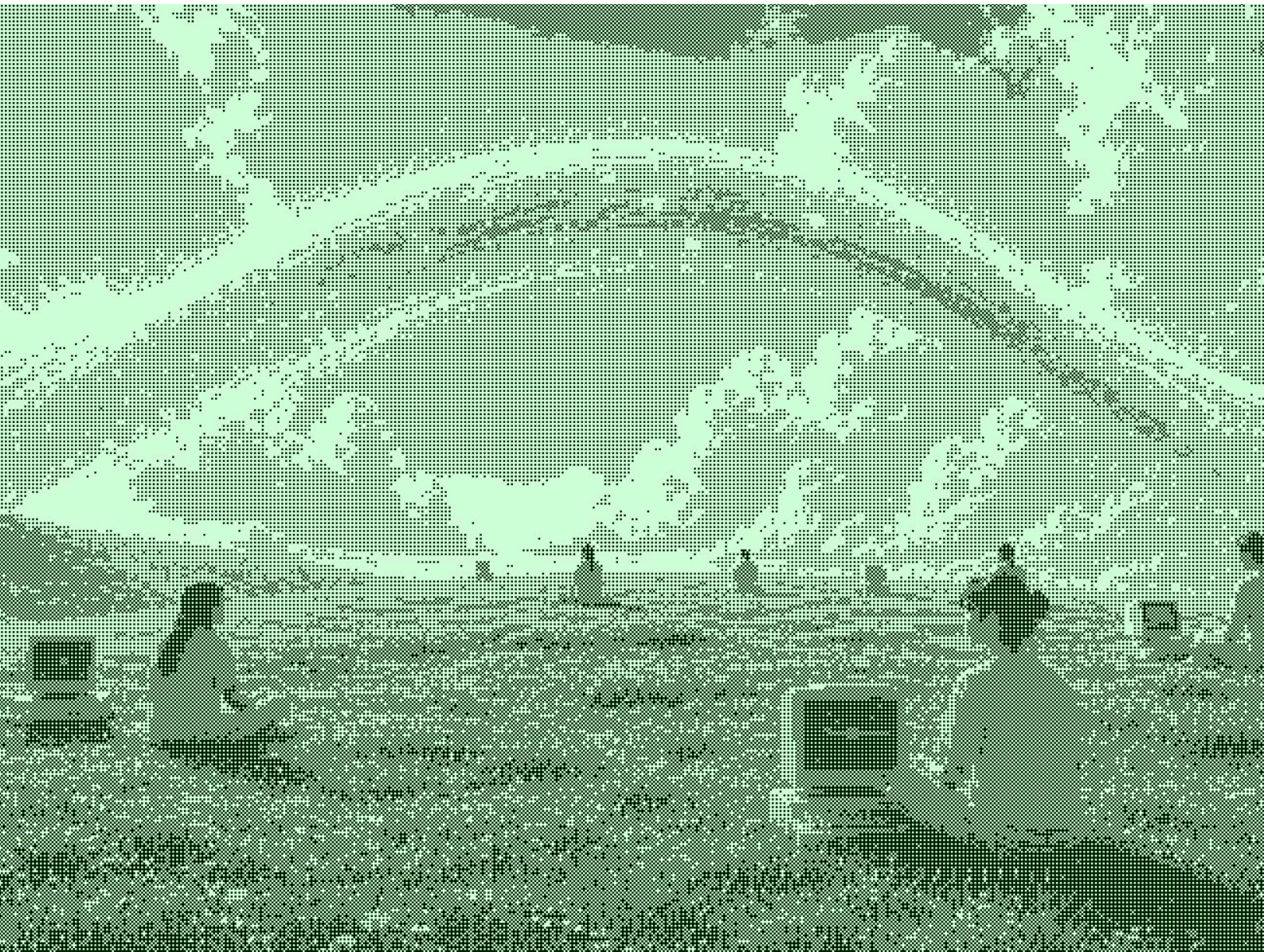
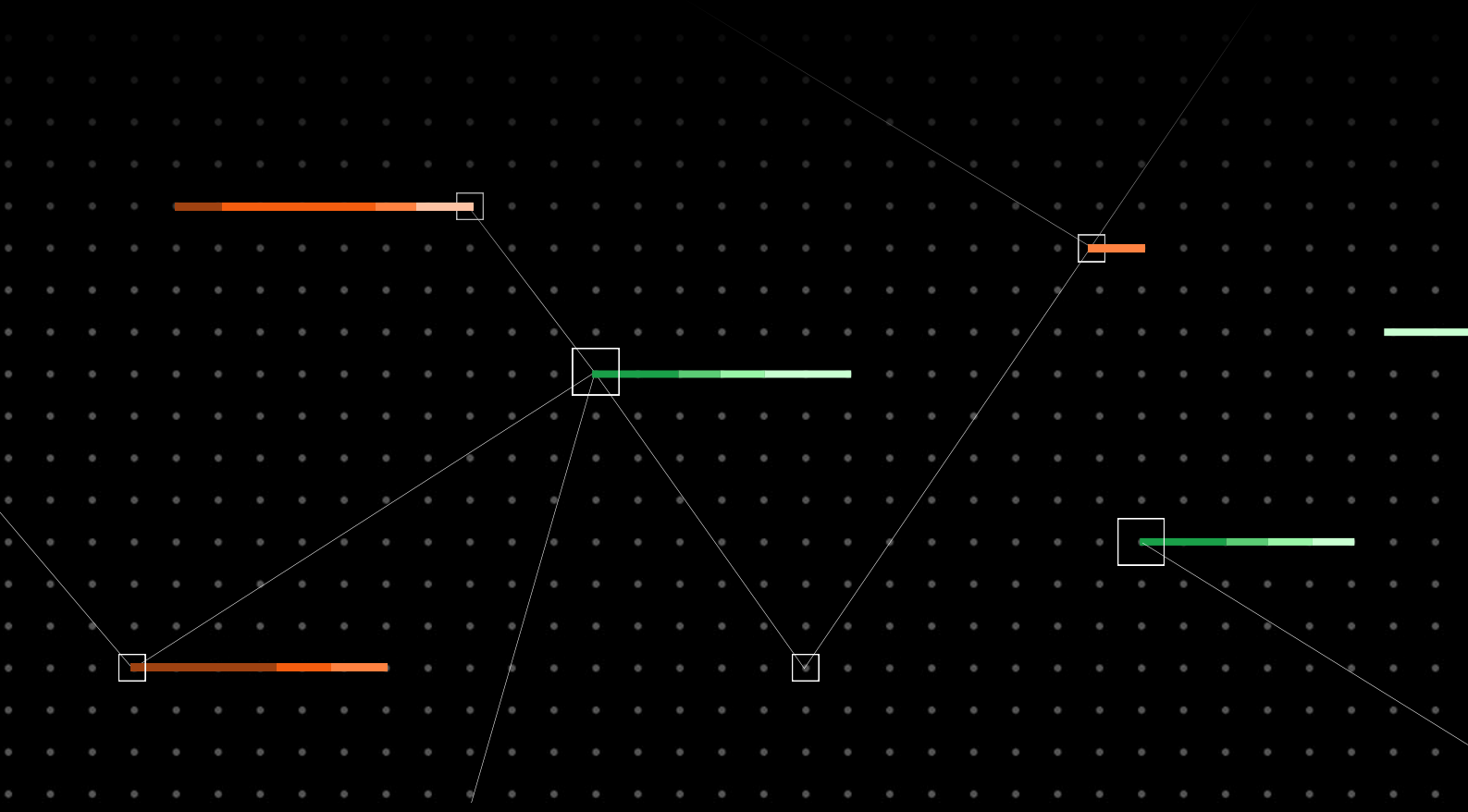


TABLE OF CONTENTS



We surveyed 219 engineering leaders and found an industry holding excitement and dread in the same breath.

Nearly half of all code in these organizations is now AI-generated. Leaders say they're hiring for judgment and systems thinking over coding ability. And they rate their teams as more competitive than a year ago.

However, AI generated code's speed comes at a cost. They also say they don't trust what they're shipping, that nearly half their engineers are worried about the relevance of their own skills, and that role definitions and onboarding have barely changed. When we asked leaders for three words to describe how they feel, many put contradictory emotions in the same answer: "excited, anxious, invigorated." "Insecurity, mistrust, hope." "Optimistic, excited, threatened."

It's the same people, holding both at once. That's what this moment feels like from inside it.
The question engineers are sitting with is not "will I lose my job." It's "will I recognize it." That's a harder question, and a more interesting one, because the answer is something we get to shape.

This isn't a split between optimists and pessimists. It's the same people, holding both at once. That's what this moment feels like from inside it.

WHY AUGMENT CODE RAN THIS SURVEY

We ran this survey because we were having the same conversation and we wanted to dive deeper. Augment has been documenting our own AI-native journey in public, including the course corrections. Six months ago, we pushed hard on agent-generated code. Similar to others, our PR volume went up and confidence in what was shipping went down. Our team had to rethink how engineers stay connected to the codebase.

We took two days off the roadmap to ask the team how they were feeling. The word cloud that came back: frustration, uncertainty, curious, excited, hopeful, agency. One breakout group named the shift directly: from "proud builder" to "proud coordinator." Another raised the fear of skill atrophy: if you stop writing code daily, you lose the understanding required to step in when things break.

Then we started hearing the same things at every dinner, event, and meetup. Other engineering leaders were nodding along. The same questions kept coming up. How do you review code nobody on the team wrote? How do you know if the system still does what you think it does? What do I say to an engineer who asks if their skills still matter? What does "senior engineer" even mean when agents write most of the code?

We wanted to know if this was just our circle, or something broader. It wasn't just us.

DEFINITIONS

"AI NATIVE"

Orgs where agents are writing code that ships to production. Not autocomplete. Not chat-in-the-sidebar.

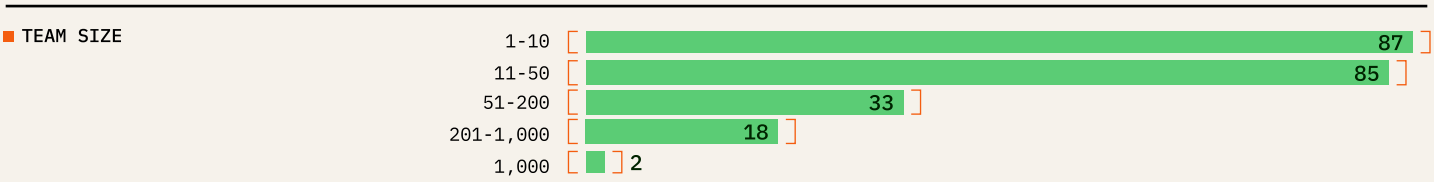
"IDENTITY CRISIS"

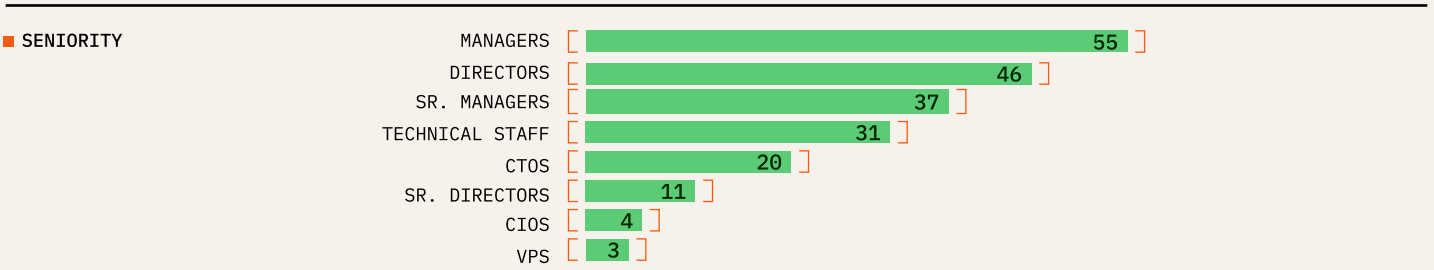
The questions engineers and their managers are asking about craft and identity when the core act of the job, writing code, is increasingly done by something else.

WHO RESPONDED

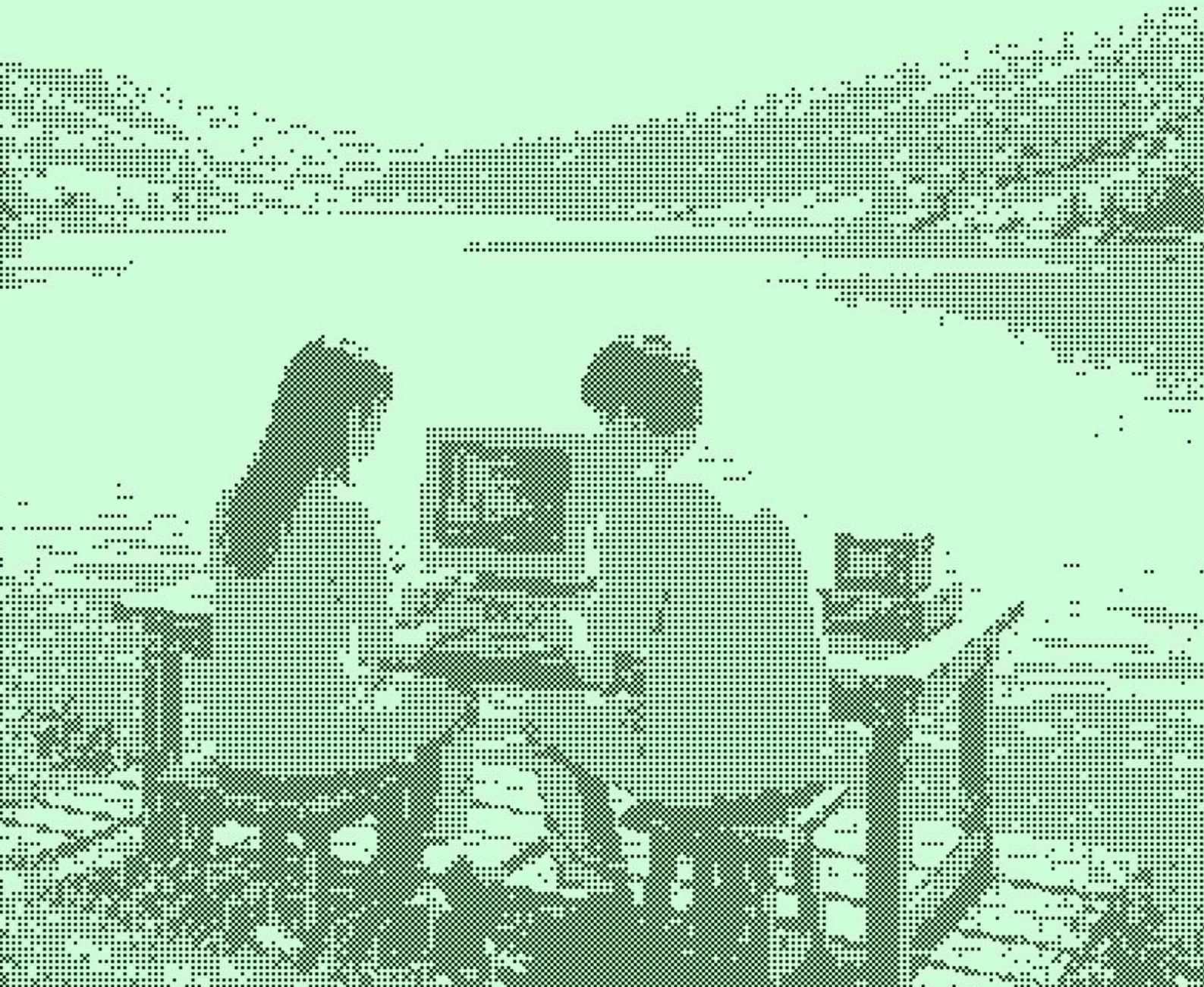
We asked 219 engineering leaders including managers, directors, senior managers, and CTOs. The majority of our respondents lead teams of 1 to 50 engineers, with a tail of larger organizations (51 respondents with 51+ engineers). See breakdowns below.

We recruited through engineering leadership networks, events, and inbound interest. The sample skews toward AI-forward leaders.





The emotional state: two feelings at once.



We asked leaders for three words to describe how they feel about AI and the future of software engineering. The most common pattern was some variation of "curious, cautious, optimistic." The question was open ended, and dozens of responses looked almost identical.

The data is interesting on two levels:
[1] How many responses contain contradictory emotions in the same answer
[2] That this tension was the general consensus among these engineering leaders.

Some of the answers included:

[EXCITED] [ANXIOUS] [INVIGORATED]

[INSECURITY] [MISTRUST] [HOPE]

[OPTIMISTIC] [EXCITED] [THREATENED]

[EXCITED] [WORRIED] [EMPOWERED]

[LIFE-CHANGING OPPORTUNITY] [TBD QUALITY]

[AMAZING] [POWER] [RISK]

[EXCITING] [CONCERNED] [CUSP]

And a consistent minority went to a harder place:

[PRETTY DARN BLEAK]

[POTENTIAL DISASTER WAITING]

[UNCERTAIN] [UNSETTLING] [STRESSED]

[AGITATED] [CURIOUS] [TIRED]

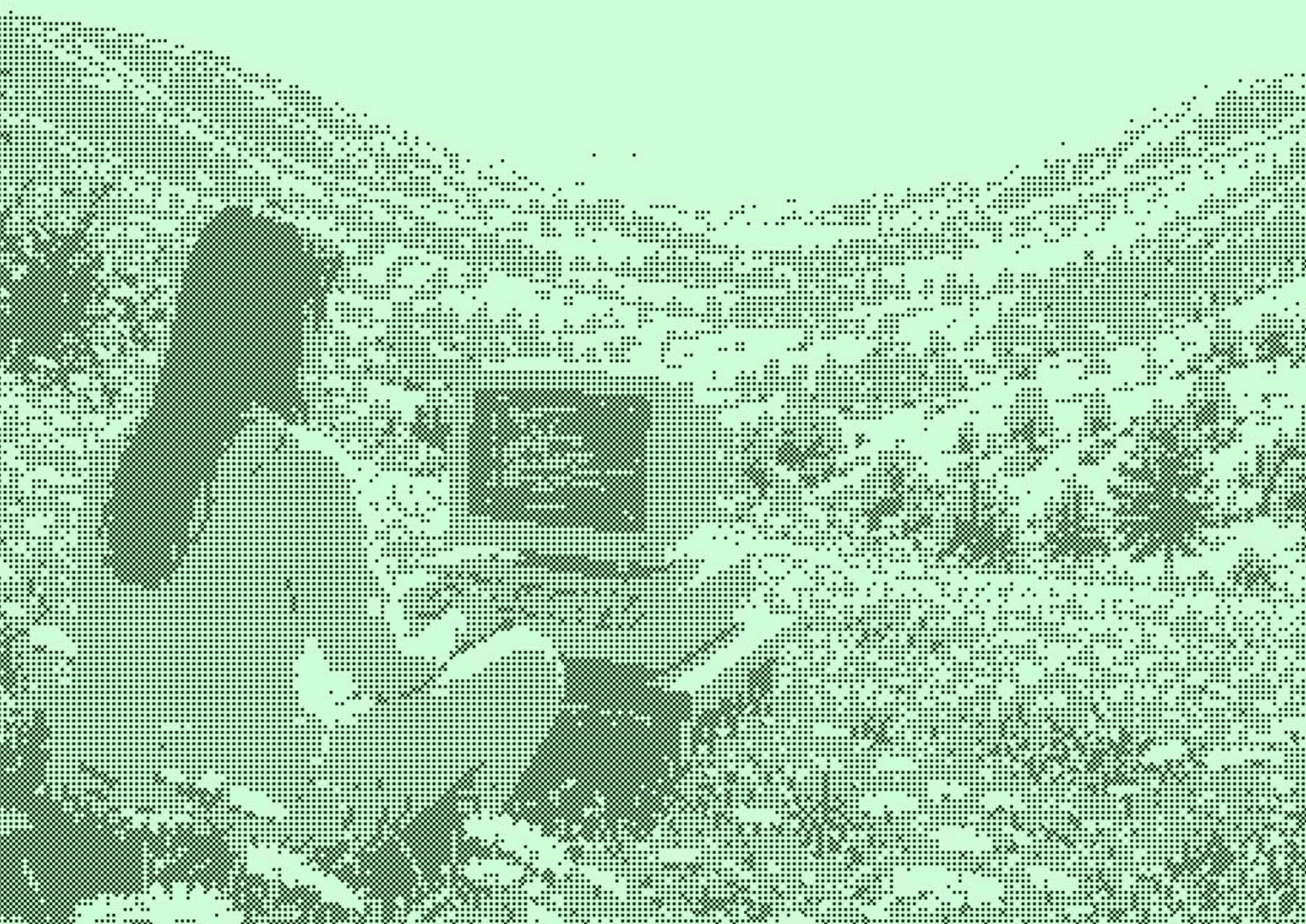
[EXHAUSTED] [ANXIOUS] [CONCERNED]

At smaller sample sizes, this looked like two distinct groups: the optimists and the worried. At 219, the pattern is different. Many of these are the same person. They're holding both feelings at once, not choosing between them.

> “
> A MIXTURE OF EXCITEMENT AND CAUTION. I KNOW THAT THIS TOOL CAN ENABLE GREAT STRIDES AS WELL AS ENABLE HUGE MESSES.
>
> - ONE RESPONDENT

That's the emotional baseline this report is written against. Both, simultaneously.

Where the strain is showing.



The feelings are real. But what are they about, specifically? When we looked at where the anxiety clusters in the data, three patterns emerged. Margaret-Anne Storey's recent research on system health gives us a useful frame. She identifies three layers of debt that compound on each other. Our survey maps to all three.

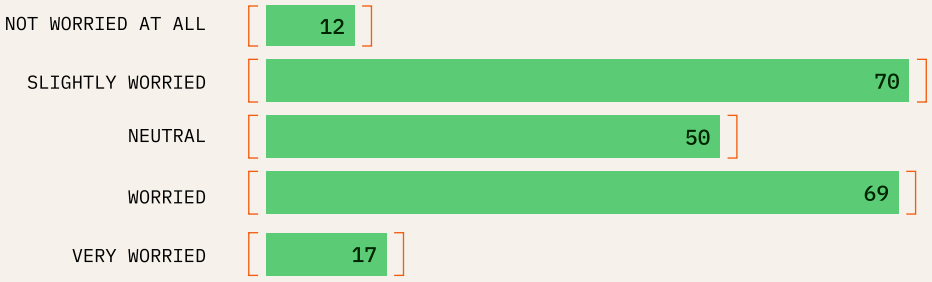
[1]

Technical debt: do you trust the code?

48% of code is now AI-generated.

MEAN	MEDIAN
48.34%	48%

But more code doesn't mean more confidence. 39% of respondents (86/218) are worried or very worried about shipping with confidence as agents write more code and approve PRs. Another 70 are slightly worried. Only 12 said not worried at all.



> "
> IT'S GREAT FOR BEING ABLE TO QUICKLY WRITE CODE. TRADEOFFS INCLUDE NOT BEING FAMILIAR WITH NEWLY WRITTEN CODE.
>
> - ONE RESPONDENT

The volume of AI-generated code also varies by team size in a way that tells its own story. Small and mid-sized teams (1-10 and 51-200 engineers) report the highest rates, above 50%. Large teams (201-1,000) report the lowest at 37%, despite being the most likely to have structured agent workflows: 61% of large teams use agents running workflows with engineer oversight, compared to about a third of small teams. Larger orgs are more deliberate in how they deploy AI, and that governance appears to slow the volume of AI-generated code even as it matures the process around it.

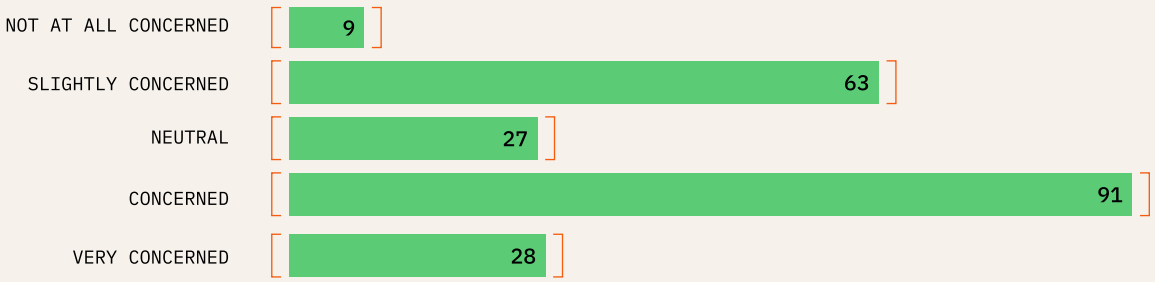
CircleCI's 2026 State of Software Delivery report, based on 28 million CI workflows, found the same pattern from a different angle: average throughput increased 59% year over year, but success rates on the main branch fell to their lowest level in five years (70.8%). More code is being written. More of it is breaking.

[2]

Cognitive debt: does anyone understand the system?

This is the loudest signal in the dataset.

55% of respondents (119/218) are concerned or very concerned about humans losing shared understanding of how the codebase is evolving. Only 9 said not at all concerned.



Storey's research calls this cognitive debt: the gap that forms when shared understanding of a system erodes faster than it is replenished. Technical debt lives in code. Cognitive debt lives in people.

Coding used to be how engineers built system intuition. You'd discover edge cases while writing. You'd feel the friction of a bad abstraction in your hands. When agents do the writing, that understanding doesn't form the same way.

> “
> EXCITING AND ALSO VERY WORRYING, I ALREADY SEE KNOWLEDGE DECLINE AND PEOPLE GETTING ADDICTED TO ITS USAGE.
>
> - ONE RESPONDENT

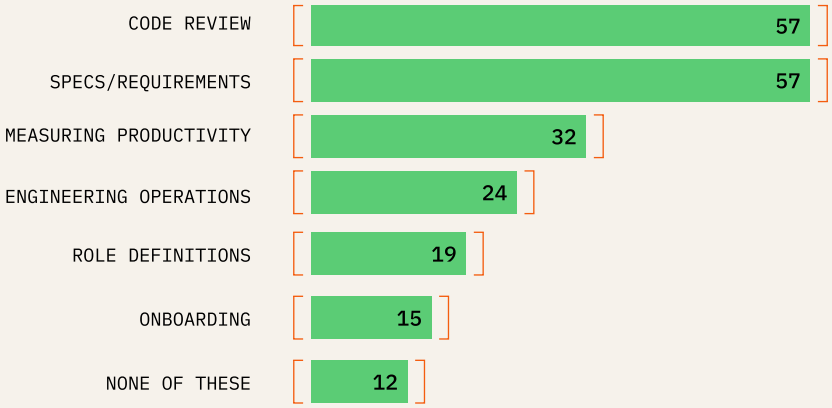
Shaw and Nave at the Wharton School published a paper that names the behavioral risk: cognitive surrender. Different from cognitive offloading, which is strategic (you know what you're delegating). Cognitive surrender is passive. You stop applying your own judgment because the AI-generated output looks right and checking it is expensive. It feels like efficiency right up until something breaks and nobody can trace the cause.

[3]

Intent debt: are the specs, roles, and processes current?

Storey's third layer is intent debt: it accumulates when the goals and constraints that should guide a system are poorly captured or maintained. Our survey shows this debt building fast.

Code review and specs are being reworked in equal measure: 57 respondents have formally changed each. But beyond those two, role definitions (19), onboarding (15), and even how productivity is measured (32) are still barely moving:



The specs, the job descriptions, the career ladders, the things that tell humans and agents what "good" looks like, are still mostly written for the old world.

These three debts compound. When the code is less understood (cognitive debt), and the specs describing what it should do are stale (intent debt), the technical debt becomes invisible until something breaks.

What the data shows: five contradictions.

[1]

Leaders feel more competitive than ever, but they don't trust what they're shipping.

FORWARD-LOOKING CONFIDENCE (12 MONTHS FROM NOW)	RETROSPECTIVE CONFIDENCE (12 MONTHS AGO)
3.96/5	3.66/5

They believe AI is making them stronger. They also report that 39% are worried or very worried about shipping with confidence, that 55% are concerned about codebase comprehension, and that they haven't changed most of their org processes. Both are true at the same time.

[2]

Codebase comprehension is the #1 concern, and it's the thing fewest orgs are doing anything about.

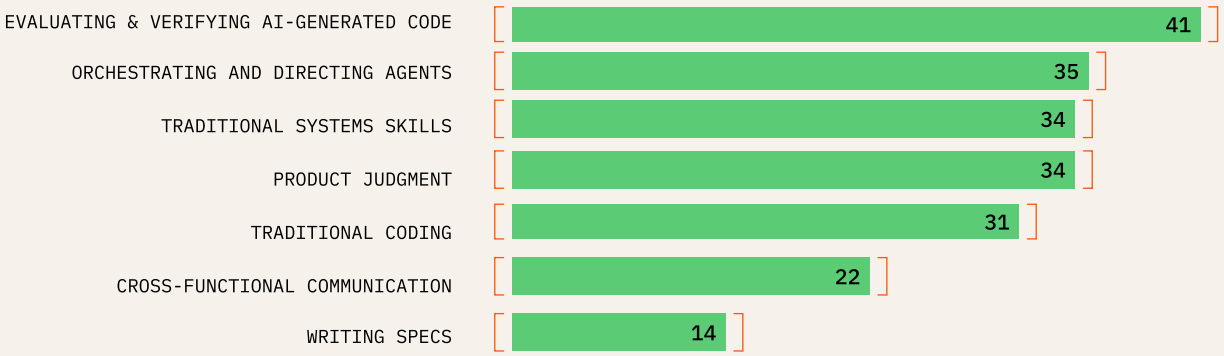
55% flagged it as a top concern

It's the highest-anxiety question in the survey. And yet role definitions (19 respondents changed), onboarding (15), and engineering operations (24) have barely moved. Everyone sees the problem. Almost nobody is building a response beyond code review.

[3]

Orgs know the role is changing, but almost none have changed anything.

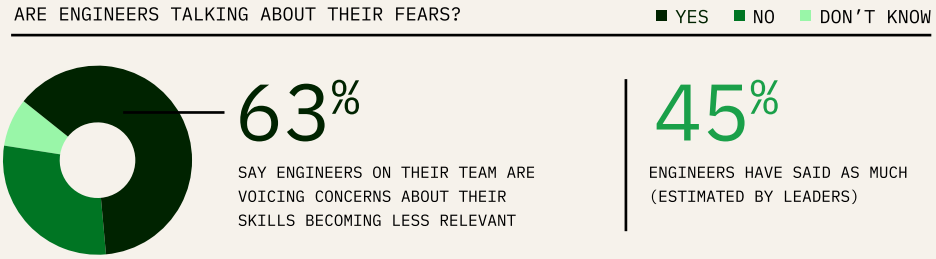
Leaders are hiring for a different job. But only 19 have formally changed role definitions. Only 15 have changed onboarding. The hiring bar has moved. The org structures around it have not. Here's a breakdown of hiring priorities shared by the survey respondonts:



Ajey Gore, writing about what becomes expensive when coding becomes free, described the reorg this implies: the Monday standup changes from "what did we ship" to "what did we validate." Our data doesn't go that far, but it points in the same direction. The skills leaders value most are already judgment, systems thinking, and orchestration.

[4]

Engineers are raising the alarm, and the org response is silence.



But the rate varies sharply by team size. 89% of leaders at large teams (201-1,000 engineers) say engineers are raising these fears, compared to 55% at the smallest teams (1-10). The anxiety conversations scale with the size of the org.

Given how few orgs have changed role definitions (19), measurement (32), or onboarding (15), most managers are absorbing these conversations and then doing nothing.

[5]

Everyone is going "AI-native," and nobody agrees on what that means.

We asked all 219 respondents to define "AI-native" in their own words. The range was enormous. Some describe a full organizational redesign. Others describe autocomplete with extra steps. Leaders are asking their teams to move toward something that has no shared definition.

> AUTONOMOUS AGENTS THAT SERVE AS CORE BUSINESS DRIVERS, NOT AUXILIARY TOOLS.

> WE DON'T USUALLY TRY TO DEFINE IT.

> AI NATIVE CURRENTLY SOUNDS LIKE JUST ANOTHER BUZZ WORD TO ME, SO I'D PREFER NOT TO DEFINE IT AT ALL."

> USING AI WHERE POSSIBLE TO STREAMLINE AND IMPROVE PROCESSES.

> THE GREATEST THING SINCE SLICED BREAD.

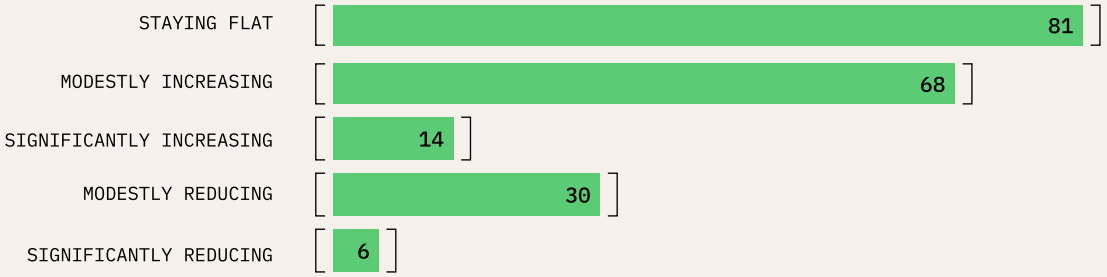
> I WOULDN'T, THAT'S A NONSENSE MARKETING TERM.

> A WORKSPACE WHERE BOTH HUMANS AND AI SHARE OWNERSHIP OF OUTCOMES, NOT JUST TASKS.

> EVERYTHING IS IMPLEMENTED BY AI FROM SCRATCH.

The headcount signal.

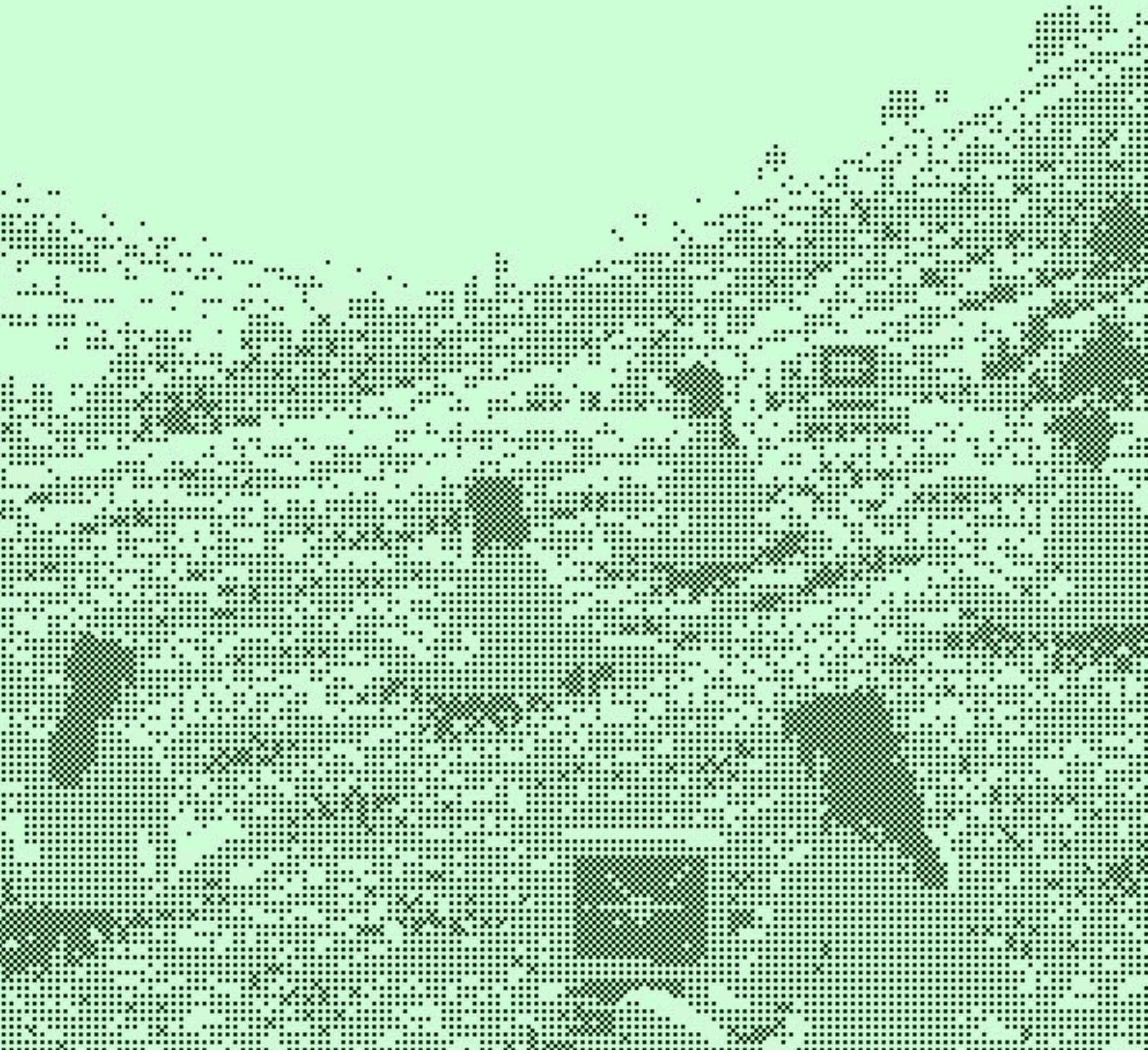
Overall, headcount is stable to growing (in the next 12 months)



But our sample is 79% teams under 50, and those teams are mostly staying flat (~39%) with moderate growth. Mid-sized teams (201-1,000 engineers) tell a more interesting story: 61% plan to grow headcount, the highest of any group, but they also have the highest reduction rate at 22%. Larger orgs are making bigger bets in both directions.

This is not a story about job losses. It's a story about job redefinition, and mid-sized engineering orgs are moving the fastest.

What this maps to in our own experience.



We don't share this because we have the answers. We share it because we've been going through the same thing, and what we've tried might be useful.

ON COGNITIVE DEBT

The more code our agents generated, the less confident the team was in what was shipping. We rethought code review entirely. We've been experimenting with a new process called Intent Review. For medium and high risk PRs, the agent guides the engineer through understanding how the change affects the system. The goal is comprehension, not approval. Most orgs in the survey haven't gotten past naming the concern.

ON THE FROZEN ORG.

Our CEO Matt McClernan wrote that the orgs falling behind "aren't losing because the AI doesn't work. They built their org structure, their decision-making, and their spending philosophy for a different era." We rewrote our hiring criteria around six dimensions: product taste, architectural judgment, agent leverage, communication, ownership, and learning velocity. Raw coding ability is no longer a standalone dimension. But we did this after a deliberate pause. Most orgs in the survey haven't taken that pause.

ON THE PEOPLE SIDE

Our breakout sessions surfaced that spec clarity is now where humans add the most value. Multiple groups independently concluded that spec reviews may matter more than code reviews. One group reframed the aspiration: not "ticket to PR" but "intent to production." This only happened because we gave the team space to say what they were feeling.

ON THE EMOTIONAL GAP

Our internal word cloud mirrors the survey's three-word responses almost exactly. Frustration and excitement. Uncertainty and agency. The gap between the official narrative ("we're excited about AI") and what people say when they're honest. Most orgs haven't created a space to surface this. The survey suggests the feelings are there whether you ask or not.

What you can do now.

This space is moving too fast for a thorough playbook. But these are concrete starting points.

[1] **RETHINK HOW YOU HIRE.** The survey shows leaders are prioritizing judgment, systems thinking, and orchestration over coding ability. But most haven't updated their interview loops, leveling criteria, or job descriptions. If the skill you value most is evaluating AI-generated work, your hiring process should test for it.

[2] **GET GOOD AT WRITING SPECS, BUT DON'T TREAT THEM LIKE DOCUMENTS.** Spec-driven development will fail for the same reason every documentation-first initiative has failed: it asks developers to do maintenance work nobody sees and nobody rewards. The fix is making specs a living contract that both humans and agents read from and write to. When an agent discovers an assumption in the spec was wrong, it updates the spec itself, the way a good junior engineer would update the ticket.

[3] **WRITE DOWN WHAT YOUR TEAM KNOWS.** Agents replicate what's already there. If your conventions, architectural decisions, and reasoning live in people's heads, agents will never learn them. Write guidelines. Write skills files. One survey respondent defined AI-native as "a workspace where both humans and AI share ownership of outcomes, not just tasks." That's only possible if the knowledge is written down in a form agents can consume.

[4] **TREAT AGENT PRODUCTIVITY AS A DISCIPLINE.** Agents do their best work when they have both independence and autonomy. Independence is about context. Autonomy is about action. Agents need both to be productive. The Agent Productivity team's job is to make sure agents have the context to make good decisions and the tools, permissions, and guardrails to act on them.

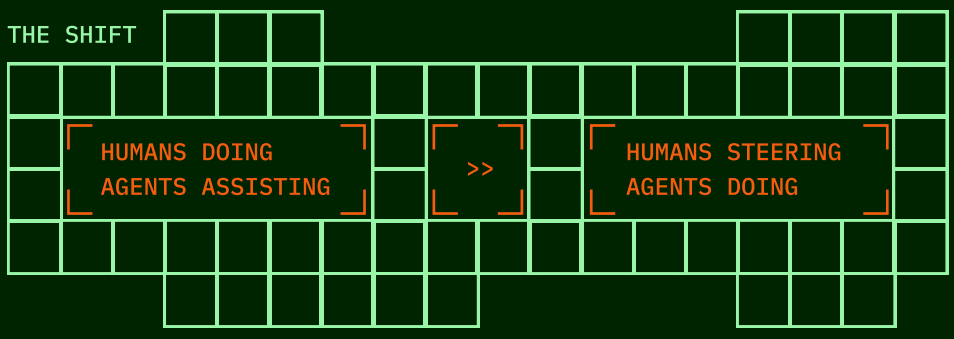
[5] **MAKE SPACE FOR THE FEELINGS.** When you are going really fast, the only way to change direction is to first slow down before reorienting and reaccelerating in the new direction. 63% of respondents said their engineers are voicing concerns about their skills. At large teams, it's 89%. These feelings don't go away if you don't ask. They go underground. They show up as passive resistance, quiet disengagement, people going through the motions while privately wondering if their skills still matter. I wrote about this after Augment's own offsite: one breakout group named the shift from "proud builder" to "proud coordinator." Another raised the fear of skill atrophy. These conversations only happened because the team was given permission to have them. Take time off the roadmap. Ask your team how they're feeling. Name the shift out loud. The orgs that do this will keep their best people. The orgs that don't will lose them to companies that did.

Cosmos

At Augment, we're moving from agents to orchestration.

Every problem in this report traces back to the same gap: individual engineers have adopted agents, but the organization hasn't changed around them. Setups fragment. Expertise gets trapped. The review queue grows. More code gets written and less gets understood. Individual adoption is not organizational change.

That's why we built Cosmos: an operating system for agentic software development. Agents that run anywhere, work across the SDLC, with humans steering where judgment matters. Cosmos is built for teams: shared context and memory, self-improving agent loops, connections to all of your tools, multi-model by default.



[TRY COSMOS FOR YOUR TEAM]

SOURCES AND FURTHER READING

VINAY PERNETI	CONFIDENCE IS THE NEW BOTTLENECK
VINAY PERNETI	THE HARDEST PART ABOUT GOING AI-NATIVE ISN'T JUST TECHNICAL
VINAY PERNETI	WE'RE IN AN EXPONENTIAL
VINAY PERNETI	YOU DON'T HAVE AN AI PROBLEM. YOU HAVE A TRANSFORMATION PROBLEM.
VINAY PERNETI	HOW WE HIRE AI-NATIVE ENGINEERS NOW
AMELIA WATTENBERGER	WHAT SPEC-DRIVEN DEVELOPMENT GETS WRONG
ZIZHUANG YANG	SPECS ARE INFRASTRUCTURE IN THE AGE OF AGENTS
NIKITA SIROHI	PROMPTS ARE INFRASTRUCTURE
MARGARET-ANNE STOREY	THREE LAYERS OF SYSTEM HEALTH
SHAW AND NAVE	TRI-SYSTEM THEORY OF COGNITION
AJEY GORE	THE EXPENSIVE THING
MARTIN FOWLER	FRAGMENTS: APRIL 2
CIRCLECI	THE 2026 STATE OF SOFTWARE DELIVERY

THE SURVEY IS NOW CLOSED AT 219 RESPONSES.
FOR QUESTIONS ABOUT THE METHODOLOGY OR FINDINGS, CONTACT PRESS@AUGMENTCODE.COM