2026 Agentic Coding Trends Report

How coding agents are reshaping software development

Claude

Contents

Foreword: From assistance to collaboration 3

Foundation trends: The tectonic shift 4

- **Trend 1:** The software development lifecycle changes dramatically 5

Capability trends: What agents can do 7

- **Trend 2:** Single agents evolve into coordinated teams 8
- **Trend 3:** Long-running agents build complete systems 9
- **Trend 4:** Human oversight scales through intelligent collaboration 10
- **Trend 5:** Agentic coding expands to new surfaces and users 11

Impact trends: What agents may change in 2026 12

- **Trend 6:** Productivity gains reshape software development economics 13
- **Trend 7:** Non-technical use cases expand across organizations 14
- **Trend 8:** Dual-use risk requires security-first architecture 15

Priorities for the year ahead 16

Foreword

From assistance to collaboration

In 2025, coding agents moved from experimental tools to production systems that ship real features to real customers. Engineering teams discovered that AI can now handle entire implementation workflows: writing tests, debugging failures, generating documentation, and navigating increasingly complex codebases.

In 2026, we predict these gains will extend far beyond incremental improvements to existing tools or models. We expect single agents to become coordinated teams of agents. Tasks that took hours or days may now be completed with minimal human intervention. And engineers who, only a few years ago, wrote every line of code will increasingly orchestrate long-running systems of agents that handle implementation details so they can focus on architecture and strategy.

Yet a critical nuance has emerged from studying how developers actually work with AI: this transformation is fundamentally collaborative. **Research** from our Societal Impacts team reveals that while developers use AI in roughly 60% of their work, they report being able to "fully delegate" only 0-20% of tasks. AI serves as a constant collaborator, but using it effectively requires thoughtful set-up and prompting, active supervision, validation, and human judgment—especially for high-stakes work.

Inspired by our own experiences working with customers, this report identifies **eight trends** we predict will define agentic coding in 2026. These predictions fall into three categories: **foundation trends** that we believe will reshape how development work happens, **capability trends** that look to expand what agents can accomplish, and **impact trends** that we anticipate will affect business outcomes and organizational structures.

These predictions reflect what we're seeing with customers today, not certainties about tomorrow. We offer them as a framework for thinking about the year ahead, knowing the future will surprise us.

Significantly, these trends illustrate how the gap between early adopters and late movers is widening. Organizations that figure out how to scale human oversight without creating bottlenecks are better positioned to maintain quality while moving faster. Teams that master agent coordination across the software development lifecycle today can ship features in hours instead of days. Companies that **extend agentic coding beyond engineering teams** to less technical roles stand to unlock productivity gains across their entire organization.

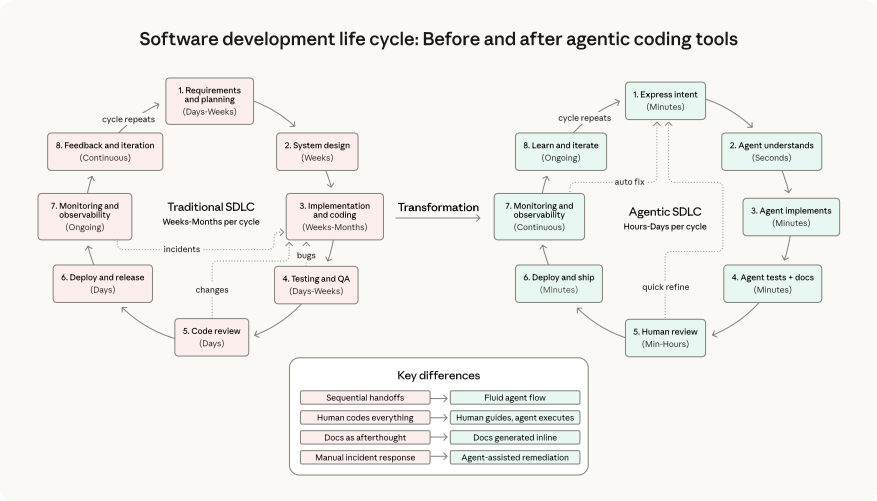
The patterns emerging in 2026 suggest software development is evolving toward a model where human expertise focuses on defining the problems worth solving while AI handles the tactical work of implementation.

Let's dive in.

Foundation trends: The tectonic shift

Trend 1

The software development lifecycle changes dramatically



Traditional SDLC stages remain, but agent-driven implementation, automated testing, and inline documentation collapse cycle time from weeks to hours. Monitoring feeds directly back into rapid iteration.

The way we interact with computers is undergoing one of its most significant changes since the graphical user interface. From machine code to assembly to C to modern high-level languages, each abstraction layer reduced the gap between human thought and machine execution.

The most recent step in this evolution was human/machine conversation. In 2025, agentic AI changed how a large swath of developers write code. 2026 is poised to be the year when the systemic effects of this evolutionary shift reconfigure the software development lifecycle and reshape software engineering roles.

Predictions

- **Evolution of abstraction:** Most of the tactical work of writing, debugging, and maintaining code shifts to AI while engineers focus on higher-level work like architecture, system design, and strategic decisions about what to build.
- **Engineering role transformation:** Building software used to mean primarily writing code, although software engineering roles always involved many other skills. Now, being a software engineer increasingly means orchestrating agents that write code, evaluating their output, providing strategic direction, and ensuring the system as a whole solves the right problems correctly.
- **Expedited onboarding to dynamic project staffing:** Traditional timelines for onboarding to a new codebase or project will collapse from weeks to hours, changing how companies think about talent deployment and project resourcing.

The collaborative reality

While agents handle more implementation work, the nature of this shift reveals something important: engineers are becoming more "full-stack" in their capabilities rather than being replaced. **Our research** shows engineers can now work effectively across frontend, backend, databases, and infrastructure—areas where they may have previously lacked expertise—because AI fills in knowledge gaps while humans provide oversight and direction.

This capability expansion enables tighter feedback loops and faster learning. Tasks that once required weeks of cross-team coordination can become focused working sessions. Engineers describe using AI for tasks that are easily verifiable,

well-defined, or repetitive, while keeping high-level design decisions and anything requiring organizational context or "taste" for themselves.

Role transformation: From implementer to orchestrator

In 2026, the value of an engineer's contributions shifts to system architecture design, agent coordination, quality evaluation, and strategic problem decomposition. The primary human role in building software is orchestrating AI agents that write code, evaluating their output, providing strategic direction, and ensuring the system solves the right problems for the right stakeholders. Engineers who master orchestration can shepherd multiple features through development simultaneously, applying their judgment across a broader scope than individual implementation previously allowed.

Onboarding revolution

In 2025, the traditional timeline for onboarding to a new codebase or project began to collapse from weeks to hours. In 2026, we anticipate organizations learning how to use this capability to its fullest, changing how companies think about talent deployment and project resourcing.

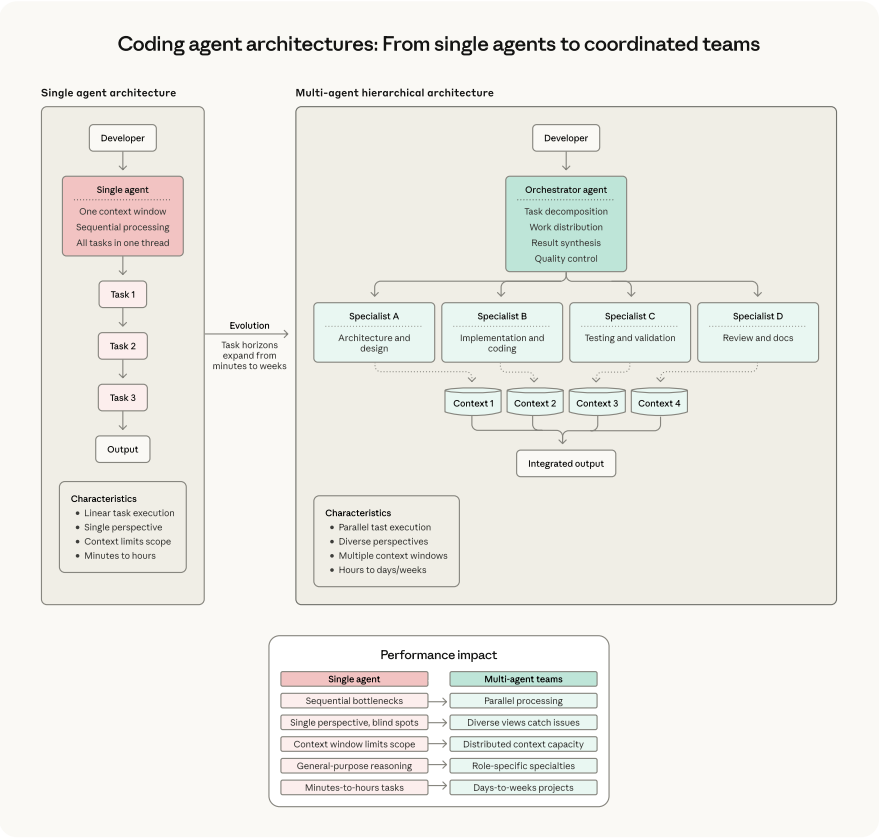
One manifestation we envision is dynamic "surge" staffing. Businesses will be able to surge engineers on-demand onto tasks requiring deep codebase knowledge. Organizations can start staffing projects dynamically, bringing in specialists for specific challenges and shifting resources without the traditional productivity dip.

Augment Code, a startup building AI-powered software development tools for systems like networking platforms, databases, and storage infrastructure, flattened the learning curve for engineers joining a new codebase or project by using Claude to provide contextual code understanding. One enterprise customer finished a project that their CTO had initially estimated would take 4 to 8 months in just two weeks using Augment Code, powered by Claude.

Capability trends: What agents can do

Trend 2

Single agents evolve into coordinated teams



Single-agent workflows process tasks sequentially through one context window. Multi-agent architectures use an orchestrator to coordinate specialized agents working in parallel—each with dedicated context—then synthesize results into integrated output.

We predict that organizations in 2026 will be able to harness multiple agents acting together to handle task complexity that was difficult to imagine just a year ago.

This ability will require new skills in task decomposition, agent specialization, and coordination protocols, along with development environments that show the status of multiple concurrent agent sessions and version control workflows that handle simultaneous agent-generated contributions.

Prediction

- **Multi-agent systems replace single-agent workflows:** Organizations adopt multi-agent workflows that maximize performance gains through parallel reasoning across separate context windows.

Fountain, a frontline workforce management platform, achieved 50% faster screening, 40% quicker onboarding, and 2x candidate conversions using Claude for hierarchical multi-agent orchestration.

Their Fountain Copilot serves as the central orchestration agent to coordinate specialized sub-agents for candidate screening, automated document generation, and sentiment analysis. This architecture enabled one logistics customer to cut the time required to fully staff a new fulfillment center from one or more weeks to less than 72 hours.

Trend 3

Long-running agents build complete systems

Early agents handled one-shot tasks that took a few minutes at most: fix this bug, write this function, generate this test. By late 2025, increasingly adept AI agents were producing full feature sets over the course of several hours. In 2026, agents will be able to work for days at a time, building entire applications and systems with minimal human intervention focused on providing strategic oversight at key decision points.

Predictions

- **Task horizons expand from minutes to days or weeks:** Agents evolve from handling discrete tasks that complete in minutes to working autonomously for extended periods, building and testing entire applications and systems with periodic human checkpoints.
- **Agents handle the messy reality of software development:** Long-running agents plan, iterate, and refine across dozens of work sessions, adapting to discoveries, recovering from failures, and maintaining coherent state throughout complex projects.
- **Economics of software development change:** When agents can work autonomously for extended periods, formerly non-viable projects become feasible. Technical debt that accumulated for years because no one had time to address it gets systematically eliminated by agents working through backlogs.
- **Path to market accelerates:** Entrepreneurs use agents to go from ideas to deployed applications in days instead of months.

At **Rakuten**, engineers tested Claude Code's capabilities with a complex technical task: implement a specific activation vector extraction method in vLLM, a massive open-source library with 12.5 million lines of code in multiple programming languages. Claude Code finished the entire job in seven hours of autonomous work in a single run. The implementation achieved 99.9% numerical accuracy compared to the reference method.

Trend 4

Human oversight scales through intelligent collaboration

Perhaps the most valuable capability developments in 2026 will be agents learning when to ask for help, rather than blindly attempting every task, and humans stepping into the loop only when required. This isn't about removing humans from the process—it's about making human attention count where it matters most.

Predictions

- **Agentic quality control becomes standard:** Organizations use AI agents to review large-scale AI-generated output, analyzing code for security vulnerabilities, architectural consistency, and quality issues that would overwhelm human capacity.
- **Agents learn when to ask for help:** Rather than blindly attempting every task, sophisticated agents recognize situations requiring human judgment, flagging areas of uncertainty and elevating decisions with potential business impact.
- **Human oversight shifts from reviewing everything to reviewing what matters:** Teams maintain quality and velocity simultaneously by building intelligent systems that handle routine verification while escalating genuinely novel situations, boundary cases, and strategic decisions for human input.

The collaboration paradox

Research from Anthropic's internal studies reveals an important pattern: while engineers report using AI in roughly 60% of their work and achieving significant productivity gains, they also report being able to "fully delegate" only a small fraction of their tasks. The apparent contradiction resolves when you understand that effective AI collaboration requires active human participation.

Engineers describe developing intuitions for AI delegation over time. As models improve, this is shifting quickly, but historically, they tended to delegate tasks that are easily verifiable—where they "can relatively easily sniff-check on correctness"—or are low-stakes, like quick scripts to track down a bug. The more conceptually difficult or design-dependent a task, the more likely engineers keep it for themselves or work through it collaboratively with AI rather than handing it off entirely.

This pattern has important implications: even as AI capabilities expand, the human role remains central. The shift is from writing code to reviewing, directing, and validating AI-generated code. As one of our engineers put it: "I'm primarily using AI in cases where I know what the answer should be or should look like. I developed that ability by doing software engineering 'the hard way.'"

At CRED, a fintech platform serving over 15 million users across India, engineers implemented Claude Code across their entire development lifecycle to accelerate delivery while maintaining quality standards essential for financial services. The Claude-powered development system has doubled their execution speed—not by eliminating human involvement, but by shifting developers toward higher-value work.

Trend 5

Agentic coding expands to new surfaces and users

The earliest wave of agentic coding focused on helping professional software engineers work faster within familiar environments. In 2026, agentic coding is poised to expand into contexts and use cases that traditional development tools could not reach, from legacy languages to new form factors that democratize access beyond traditional developers.

Predictions

- **Language barriers disappear:** Support expands to less-common and legacy languages like COBOL, Fortran, and domain-specific languages, enabling maintenance of legacy systems and removing adoption barriers for specialized use cases.
- **Coding democratizes beyond engineering:** New form factors and interfaces open up agentic coding to non-traditional developers in fields like cybersecurity, operations, design, and data science. Tools like **Cowork**, designed for non-developers to automate file and task management, signal this shift is already underway.

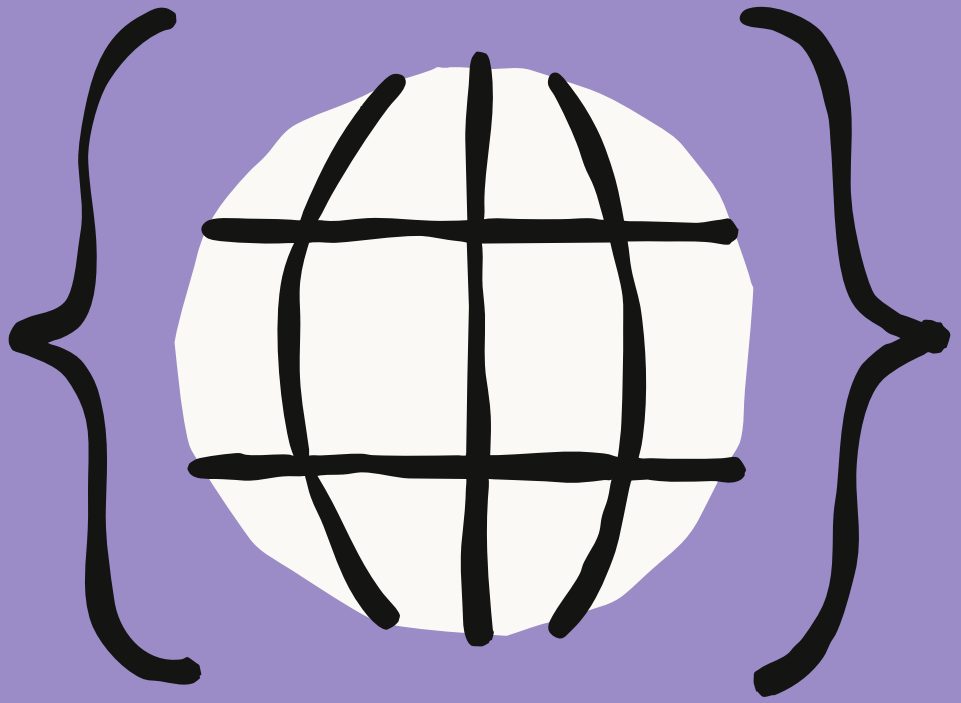
Everyone becomes more full-stack

Analysis of how different teams use AI reveals a consistent pattern: people use AI to augment their core expertise while expanding into adjacent domains. Security teams use it to analyze unfamiliar code. Research teams use it to build frontend visualizations of their data. Non-technical employees use it for debugging network issues or performing data analysis.

This expansion challenges the long-held assumption that serious development work can only happen in an IDE or that only professional engineers with specialized tools can use code to solve problems. The barrier that separates "people who code" from "people who don't" is becoming more permeable.

At **Legora**, an AI-powered legal platform, agentic workflows are integrated throughout their legal technology platform, demonstrating how coding agents extend into domain-specific applications.

"We have found Claude to be brilliant at instruction following, and at building agents and agentic workflows," said Max Junestrand, CEO of Legora. The company uses Claude Code to accelerate their own development while providing agentic capabilities to lawyers who need to create sophisticated automations without engineering expertise.



Impact trends: What agents may change in 2026

Trend 6

Productivity gains reshape software development economics

Organizations that intelligently integrate agents into their software development lifecycle will see timeline compression that affects what projects are viable and how quickly companies can respond to market opportunities.

Predictions

- **Three multipliers drive acceleration:** Agent capabilities, orchestration improvements, and better use of human experience compound to create step-function improvements rather than linear gains as each enables the others.
- **Timeline compression changes project viability:** Development that once took weeks now takes days, making previously unviable projects feasible and enabling organizations to respond to market opportunities more quickly.
- **Economics of software development shift:** Total cost of ownership decreases as agents augment engineer capacity, project timelines shorten, and faster time to value improves return on investment.

Productivity through output volume, not just speed

Internal research at Anthropic reveals an interesting productivity pattern: engineers report a net decrease in time spent per task category, but a much larger net increase in output volume. This suggests that AI enables increased productivity primarily through greater output—more features shipped, more bugs fixed, more experiments run—rather than simply doing the same work faster.

Notably, about 27% of AI-assisted work consists of tasks that wouldn't have been done otherwise: scaling projects, building nice-to-have tools like interactive dashboards, and exploratory work that wouldn't be cost-effective if done manually. Engineers report fixing more "papercuts"—minor issues that improve quality of life but are typically deprioritized—because AI makes addressing them feasible.

At **TELUS**, a leading communications technology company, teams created over 13,000 custom AI solutions while shipping engineering code 30 percent faster. The company has saved over 500,000 hours with an average of 40 minutes saved per AI interaction.

Trend 7

Non-technical use cases expand across organizations

We anticipate that one of the most significant trends in 2026 will be steady growth in agentic coding used by functional and business-process teams to create their own solutions to problems they experience, and improvements to processes they use every day.

Predictions

- **Coding capabilities democratize beyond engineering:** Non-technical teams across sales, marketing, legal, and operations gain the ability to automate workflows and build tools with little or no engineering intervention or coding expertise.
- **Domain experts implement solutions directly:** The hands-on experts who understand problems deeply gain confidence in using agents to initiate solutions themselves, removing the bottleneck of filing a ticket and then waiting for development teams.
- **Productivity gains extend across entire organizations:** Problems not worth engineering time get solved, experimental workflows become trivial to attempt, and manual processes get automated.

Zapier, a leading AI orchestration platform, has made agents accessible to all their employees. Design teams use Claude artifacts to rapidly prototype during customer interviews, showing design concepts in real-time that would normally take weeks to develop. The company achieved 89 percent AI adoption across the entire organization with 800-plus AI agents deployed internally.

How Anthropic uses Claude Code

Our legal team reduced marketing review turnaround from two to three days down to 24 hours by building Claude-powered workflows that automate repetitive tasks like contract redlining and content review. Using Claude Code, a lawyer with no coding experience built self-service tools that triage issues before they hit the legal queue, freeing attorneys to focus on strategic counsel instead of tactical busywork.

The result: lawyers reduced the potential for being a bottleneck and could devote their time to other, more pressing matters.

Trend 8

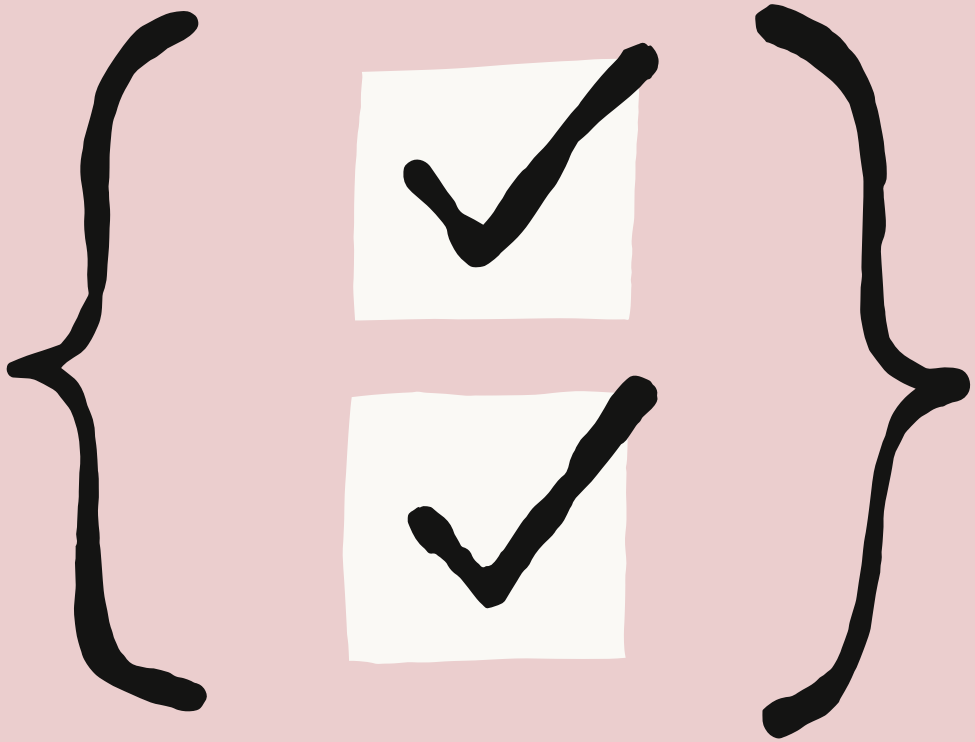
Agentic coding improves security defenses—but also offensive uses

Agentic coding is transforming security in two directions at once. As models become more powerful and better aligned, building security into products becomes easier. Now, any engineer can leverage AI to perform security reviews, hardening, and monitoring that previously required specialized expertise. But the same capabilities that help defenders are also capable of helping attackers scale their efforts.

Predictions

- **Security knowledge becomes democratized:** With improved agents, any engineer can become a security engineer capable of delivering in-depth security reviews, hardening, and monitoring. Engineers will still need to consider security and consult with specialists, but it will become easier to build hardened and secure applications.
- **Threat actors scale attacks:** While agents will benefit defensive uses, they will also benefit offensive uses too. In order to defend against this dual-use technology, it will become more important for engineers to build in security from the start.
- **Agentic cyber defense systems rise:** Automated agentic systems enable security responses at machine speed, automating detection and response to match the pace of autonomous threats.

  The balance favors prepared organizations. Teams that use agentic tools to bake security in from the start will be better positioned to defend against adversaries using the same technology.



Priorities for the year ahead

Priorities for the year ahead

These eight trends are poised to define agentic coding in 2026 all converge on a central theme: software development is shifting from an activity centered on writing code to an activity grounded in orchestrating agents that write code—while maintaining the human judgment, oversight, and collaboration that ensures quality outcomes.

The research is clear: AI is a constant collaborator, but using it effectively requires active supervision and validation, especially in high-stakes work. While more routine coding tasks can be delegated to AI, humans are still reviewing the code. It's not "fully delegated" but highly collaborative. This distinction matters for how organizations approach AI adoption and how they think about the evolving role of engineers.

For organizations planning their 2026 priorities, four areas demand immediate attention:

1. **Mastering multi-agent coordination** to handle complexity that single-agent systems cannot address
2. **Scaling human-agent oversight** through AI-automated review systems that focus human attention where it matters most
3. **Extending agentic coding beyond engineering** to empower domain experts across departments
4. **Embedding security architecture** as a part of agentic system design from the earliest stages

Organizations that treat agentic coding as a strategic priority in 2026 will define what becomes possible, while those that treat it as an incremental productivity tool will discover they are competing in a game with new rules. The key to success lies in understanding that the goal isn't to remove humans from the loop—it's to make human expertise count where it matters most.

Claude

claude.ai