ESSAY · JUNE 2025

AI's Impact on Product Management and Development Roles

A practitioner's attempt to sort the real from the hyped, and to ask what's left of these jobs once the tools take over the easy parts.

Marshall Cahill · June 2025

Why I wrote this

I work in product, and over the past eighteen months I've watched the ground shift under my feet. The tools changed first — copilots in my IDE, AI summarizers in my inbox, design assistants my designer friends use to skip the blank page. Then the conversations changed: hiring managers asking for "AI-fluent" PMs, founders pitching "lean teams powered by AI," recruiters wondering aloud whether they still need to fill the junior analyst seat.

I wrote this for myself, mostly. To force myself to sit with the question I keep dodging in casual conversation: what does this actually mean for how I work, what I'm worth, and what comes next? I've tried to be honest about what's real, what's hype, and what I'm still unsure about. The cited research is doing the heavy lifting on the factual side; the framing and the takes are mine. If you're a product manager, engineer, or designer reading this, I hope it saves you the same week of pacing I spent on it.

The shift, in one sentence

AI is changing the unit of contribution. The unit used to be the *task* — write the spec, ship the feature, mock the screen. The unit is becoming the *outcome* — define what should exist, get it built, decide if it's right. Tasks compress; judgment doesn't. The people who do well in this transition are the ones who notice the difference.

That sentence is the spine of everything that follows. Most of the breathless takes I've read about AI and work obscure it by either over-claiming ("AI replaces designers!") or under-claiming ("AI is just a new tool, like Excel"). Neither is quite right. What's actually happening is that AI is collapsing the cost of producing artifacts — drafts, code, mocks, summaries — while leaving the cost of *deciding what's worth producing* roughly where it was. That asymmetry is the entire story.

As Zeinab Abbassi put it in a widely-shared LinkedIn post earlier this year: *"The AI revolution will result in flatter orgs. An individual contributor's future value*

won't come from executing tasks AI can handle — it will emerge from how effectively they lead without formal authority."[1] I think she's directionally right. The fuzzy part is what "leading without authority" looks like day to day, and that's most of what I'm trying to work out below.

How individual roles are changing

Product managers: from data-wrangler to navigator

The traditional PM job was, in large part, an exercise in synthesis under information scarcity. You stitched together survey results, sales call notes, support tickets, and your own intuition, then defended your prioritization. A lot of that synthesis is now cheap.

Tools like Mixpanel and Amplitude have added AI insight layers that surface anomalies and trends without anyone writing a query.[2] Productboard's AI clusters customer feedback into themes you used to spend a week assembling.[3] ChatPRD will draft a serviceable first-pass requirements document from a few bullet points.[3] Marty Cagan, who has spent more time thinking about the PM role than nearly anyone, argues the role isn't being eliminated — it's becoming *more* essential and more demanding, because the work that's left is the work that requires judgment.[4]

I think Cagan is right but I'd push his point further. The PMs I see thriving aren't the ones who picked up the AI tools fastest; they're the ones who noticed that when the cost of generating a roadmap drops to zero, the *quality of the question being asked* becomes everything. Anyone can ask an LLM to generate a list of feature ideas. Knowing which feature actually solves the customer's real problem — and being willing to defend that answer against a chorus of cheaper AI-generated alternatives — is a different and harder skill. It's the difference between being a passenger and a navigator.

Engineers: from code producer to system thinker

Developers were the first cohort to feel this clearly. GitHub's 2022 controlled study of Copilot found developers completed tasks 55% faster with the tool than without it; 88% reported feeling more productive. A related GitHub study on

Copilot Chat found 85% felt more confident in their code quality.[5] The follow-on enterprise deployment with Accenture goes further: developers produced 84% more successful builds, with over 80% of participants adopting the tool and 67% using it five or more days a week.[5]

But ask any senior engineer what's actually changed about their week and you get a more nuanced answer. The boilerplate is faster. The unfamiliar code is more legible. The first draft of a unit test arrives in seconds instead of minutes. What hasn't changed is the part that was always hard: deciding whether the architecture is right, whether the failure mode you haven't considered will bite you in three months, whether the thing you're building should exist at all.

A frontend engineer can now reach into a backend file and make a reasonable change with an AI in the loop, the way a generalist could on a small startup team a decade ago. That's a real expansion of scope. The flip side, less discussed, is that AI-generated code carries a particular kind of risk: it's confident, plausible, and frequently subtly wrong. The engineers I trust most have started talking less about *writing* code and more about *reviewing* it — including code they technically wrote themselves, with an AI's help. The skill that's appreciating in value is the ability to read a diff and feel the wrongness of it.

Designers: from pixel-pusher to curator

Design is the role I'm least confident about, partly because design is the place where the gap between "demo magic" and "actually shippable" is widest. Figma has rolled out AI features. Midjourney and DALL-E can spin up reference imagery in seconds. Galileo and Uizard claim to go from prompt to UI.[3]

But Nielsen Norman Group, who first concluded in April 2024 that AI UX-design tools were "not ready for primetime," revisited the question in May 2025 and found that while the tools have improved, *"we're still nowhere near the AI-powered design tools we've been promised, nor are design professionals yet in danger of being replaced by AI."*[6] The outputs are good enough to ideate against, not good enough to ship without significant rework. That mirrors what every designer I've talked to says: AI is useful for breaking out of the blank canvas, for generating ten variations of a layout when the budget previously allowed one, for translating a rough sketch into something presentable. It is not,

yet, replacing the part of the job where a person decides whether the experience feels right.

If anything, design might be the role where the asymmetry I described earlier — cheap artifacts, expensive judgment — is most visible. When everyone can generate twenty mockups in an afternoon, the designer's value moves further upstream: into research, taste, and the ability to defend a single answer when twenty plausible ones are on the table.

The "AI Product Engineer" idea, and where I think it breaks

The most provocative framing of where this is all heading is the "AI Product Engineer" — a hypothetical individual who, with AI as their cofounder, can handle strategy, design, code, and iteration alone.[7] The argument is straightforward: if AI can fill the gaps in any one person's skill set, then a sufficiently determined generalist can run the whole stack themselves, with no handoffs, no inter-team friction, and total ownership of an outcome.

There's real signal here. Atlassian formalized a version of this pattern years before AI tools matured, calling it the "product engineer": engineers who own outcomes end-to-end, including shaping the problem and talking to customers, not just executing a spec.[11] The trend toward "tiny but high-impact" startups is real: Cursor reportedly reached $100M in annual recurring revenue with a team of around twenty; ElevenLabs crossed $100M with a team in the high double digits; Midjourney built one of the most-used generative-AI products in the world while staying under 40 employees through its first $200M of annualized revenue in 2023.[10] All three companies have grown substantially since those milestones, so this isn't a steady-state claim — it's a claim about how *thin* a team can now be in the early-scaling phase. A decade ago, no software business reached these revenue marks with these headcounts. Today, several have.

Where I think the framing oversells is in the implicit claim that diversity of perspective is a coordination tax to be eliminated. It isn't. A designer, an engineer, and a PM in a room are not just three skill sets in a trench coat — they're three different ways of seeing the same problem. An AI Product Engineer

who can technically do all three jobs will, in most cases, still do them with one person's blind spots. The good version of this future isn't one heroic generalist; it's *small, deliberate teams* — two or three people, each cross-trained — where the diversity of perspective is preserved and the coordination tax is paid down by AI. That's the version I'd bet on.

I want to name a tension in my own position here, because I'm currently operating as one of those solo operators — using AI agents to do work that a few years ago would have required a small team. That's in apparent contradiction with what I just argued. My honest read is that the solo configuration is the right *early-stage* move, useful when you're still mapping the surface area and don't yet know what to hire for, and that the team-of-three is what you graduate into once the bets sharpen. Team-of-one now, team-of-three later. That's also the arc Midjourney, Cursor, and ElevenLabs followed — stunningly thin during early scaling, then growing into something more conventional once the product surface was clearer. The lone operator as the endgame is the part I'd push back on. The lone operator as a phase you pass through is just realism.

What this means for how teams are structured

The role changes are the visible part. The harder change, and the one most companies haven't reckoned with, is what happens to the org chart.

If individual contributors can credibly own broader scopes, you need fewer of them. If you need fewer of them, you need fewer managers to coordinate them. If decisions can be made by the people closest to the work — because they now have the data and tools to do so — the role of the middle manager shifts from gatekeeper to coach. As one technologist put it, "the power center in tomorrow's organizations won't be who controls headcount but who best harnesses AI capabilities."[9]

I think the practical implications are:

A move toward smaller, cross-functional pods. The team-of-three is becoming the team-of-one-with-tools, and the team-of-eight is becoming the team-of-three-with-tools. This isn't a prediction; it's already the default at AI-native startups.

A flattening of middle management. Not the elimination of management — but a redefinition. The valuable manager in this world is the one who teaches their team to delegate to AI well, runs ethical and quality checks on AI output, and creates the conditions for autonomous work. The manager whose job is to relay information up and down a hierarchy is the role most at risk.

A higher floor on individual responsibility. With smaller teams and broader scopes, every person on the team is making decisions that used to belong to someone more senior. That's exciting if you're hungry for ownership. It's exhausting if you're not. Companies will need to decide which culture they're optimizing for, and people will sort themselves accordingly.

A culture problem, not just a structure problem. You can't flatten an org by redrawing the boxes. The shift requires teams to trust each other, to trust the AI's outputs to a degree, and to trust managers to step back. Most companies will get the structure right and the culture wrong, and the gap between those two will be where the next round of organizational dysfunction lives.

A few cases worth knowing

Three are worth highlighting because they each illustrate a different facet of what's happening.

GitHub and Accenture. The clearest evidence we have that AI coding tools meaningfully change developer output at enterprise scale. Accenture's deployment of Copilot showed an 84% increase in successful builds among the developers using it, with over 80% adopting the tool and roughly two-thirds using it five or more days a week.[5] The build-success number is the one that's hard to wave away: it's not a self-report about feeling more productive, it's a measurable lift in whether the code actually compiles and ships. When that metric moves by 84% in a giant consulting firm's day-to-day, the tool is doing real work.

Midjourney. Built one of the most-used generative AI products in the world while keeping the team under 40 people through its first $200M of annualized revenue in 2023.[10] What's interesting isn't the headcount-to-revenue ratio in isolation — Midjourney is a category-defining product riding a generational

wave, and category-defining products are always lean early. What's interesting is *how long* they stayed lean before scaling. The team did grow through 2024, but remained startlingly small relative to the revenue, suggesting that the operational gains from AI tools are real enough to defer the usual hiring rush. Whether this works for companies that aren't riding a generational wave is the open question.

Atlassian's "product engineer." Atlassian formalized this pattern years before AI tools were widely available. The term was coined years earlier by then-SVP of Engineering Jean-Michel Lemieux and codified in 2019 by Distinguished PM Sherif Mansour, who described product engineers as engineers who own outcomes end-to-end — shaping the problem, talking to customers, and shipping the solution rather than just executing a spec.[11] The lesson, if it generalizes, is that the *organizational* move toward individual ownership was already working before AI showed up. AI is now lowering the barrier to entry, turning what was previously a senior-engineer-only mode of working into a default that more people can plausibly operate in.

What I'm taking from this

Writing this clarified four things for me. I'll offer them as recommendations, but really they're notes to myself.

Get comfortable with the tools, but don't mistake fluency for advantage. Six months ago, knowing how to prompt Claude well felt like a meaningful edge. Today, it's table stakes. The actual edge is what you do with the time the tools give back to you. If you fill it with more tasks, you've gained nothing. If you fill it with more thinking, more customer conversations, more time on the hard questions — that's where the leverage compounds.

Invest in the skills AI is bad at. Judgment, taste, narrative, the ability to read a room, the willingness to make a decision under uncertainty and own it. These are not new skills, but their relative value is going up sharply. The cliché that AI will replace people who can't use AI is half true; the more useful framing is that AI will widen the gap between people who can think clearly and people who can't.

Be skeptical of the “lone genius with AI” story. It will sell a lot of books, but it’s not where most of the value will accrue. Small teams of cross-trained people, each strong in one craft and conversant in two others, will outperform individuals in nearly every domain that matters. The career bet is on becoming the kind of person who can be that team member — not on becoming a one-person team.

Notice what gets harder, not just what gets easier. Every productivity gain has a hidden cost. Faster code means more code to review. Cheaper drafts mean more drafts to evaluate. AI-generated insight is only as useful as the human judgment applied to it. The roles that survive and thrive are the ones that meet the new costs head-on, not the ones that pretend they don’t exist.

What could still go wrong

I want to end with what I’m not sure about, because I find I trust writing more when the author admits where they’re guessing.

The biggest open question is whether the productivity gains we’re seeing in narrow studies translate to durable organizational performance. A 55% speedup on a coding task is not a 55% speedup on shipping a product, because shipping a product involves a hundred steps that AI doesn’t touch. We might find, two years from now, that the productivity story was real but smaller than the early numbers suggested.

The second open question is what happens to the talent pipeline. If junior roles compress because one mid-level person with AI can do the work of three juniors, where do the future mid-level people come from? The answer “they’ll just be more productive juniors” is plausible but unproven. We may be optimizing the system today in ways that hollow it out a decade from now.

The third is the quality question. AI tools produce plausible-looking output. Plausible-looking output is dangerous in fields where being subtly wrong matters — code, medicine, law, design decisions that compound. The companies that will look smart in five years are the ones that built quality and review processes commensurate with the new generative speed. The ones that didn’t will produce a lot of polished garbage.

I don’t know how any of this resolves. I’m writing this in June 2025, and the honest answer is that we’re a year or two into a multi-decade shift, and the early data points are suggestive rather than conclusive. What I do believe is that the people who treat this as a serious, evolving question — rather than as either an existential threat or a marketing slogan — are the ones who’ll come through it best positioned. That’s the position I’m trying to occupy. I hope this essay helps you find yours.

Sources

1. Abbassi, Z. (2025). *How AI will change the role of individual contributors.* LinkedIn
2. Egon Zehnder (2024). *How AI Is Redefining the Product Manager’s Role.* egonzehnder.com
3. Product School (2025). *Top 21 AI Tools for Product Managers and Product Teams.* productschool.com
4. Cagan, M. & Baxley, B. (2024). *AI Product Management 2 Years In.* Silicon Valley Product Group
5. GitHub (2022, 2024). *Quantifying GitHub Copilot’s impact in the enterprise with Accenture.* github.blog
6. Nielsen Norman Group (April 2024, updated May 2025). *AI Design Tools Are Marginally Better: Status Update.* nngroup.com (original: Status Update: AI UX-Design Tools Are Not Ready for Primetime)
7. Gottlob, F. (2024). *The Rise of the AI Product Engineer.* Medium
8. Goudet, M. (2024). *AI is Replacing Designers.* Medium
9. Parikh, B. (2025). *AI Will Flatten Enterprises: How Organizational Structures and Roles Must Adapt.* LinkedIn Pulse
10. The VC Corner (2025). *The Billion-Dollar Startup Formula: Why AI-Driven Small Teams Are Beating Giants.* thevccorner.com
11. Mansour, S. (2019). *Product Engineers.* Atlassian Product Craft Blog