Building the AI-Native Organization

A Fireside Chat on AI-Native Product, Engineering & Culture
MOD = Moderator · TL = Thought Leader

MOD There's a phrase floating around leadership circles right now — "AI-native." Everyone says they want to be AI-native. But when you push on what that actually means, you get a lot of vague hand-waving. So let me ask you directly: what does AI-native actually look like in practice?

TL It starts with a distinction most people miss. There's a massive difference between AI-assisted and AI-native. AI-assisted means you're doing the same work the same way, just with a smarter tool in the corner. AI-native means the work itself is different. The workflows are different. The team structures are different. The product architecture is different. You can't get there by bolting AI onto what you already have. You have to rebuild.

MOD That sounds expensive and disruptive. Why not just get really good at AI-assisted?

TL Because the market won't wait. Here's what's actually happening. The web is splitting in two. On one side, you have discovery — generic, AI-driven experiences. You ask a question, you get an answer, no brand involved. On the other side, you have brand experience — owned surfaces where the brand controls the relationship, the context, the delight. One leader I respect calls this "Disneylandification." When a user enters your brand's world, they should feel like they walked into Disneyland — not like they're reading a brochure. That means actionable experiences, trust signals that actually change behavior, and a context layer that makes personalization real instead of guesswork. You don't build that with AI-assisted. You build that with AI as infrastructure — AI that understands the user, the content, and the moment, all at once.

MOD So the product has to change. What about the people building it?

TL That's where it gets really interesting. The engineering role is inverting. I'll give you a concrete example. One engineer — real person, real company — shipped over three hundred pull requests in a single month. No overtime. No team behind him. One person.

MOD Three hundred PRs from one person. How is that even possible?

TL He runs ten to fifteen AI agent sessions in parallel. Five terminals on his machine, another five to ten in the browser. Each one working on a separate task. His workflow is three steps: plan, execute, verify. He spends his time getting the plan right with the AI. Then the agents write the code, run the tests, commit, push, and open the pull request — automatically. His job is to verify the output and steer the fleet. He doesn't write code. He manages agents.

MOD And you're saying that's the future for all engineers?

TL It's the future for the best ones. And here's the part that surprises people — it's a harder job, not an easier one. Writing code is a well-understood skill. Decomposing a problem into fifteen parallel tasks, choosing the right delegation strategy, spotting when an agent is drifting, verifying output at speed — those are new muscles. The bottleneck is no longer typing code. The bottleneck is thinking clearly. Design, architecture, judgment. That's what matters now.

MOD So how do you know where your team is on this curve? Not everyone's at three hundred PRs a month.

TL There's a maturity framework I find very useful. Four levels. Level one — AI User — you use AI sometimes, your workflow hasn't changed. Level two — AI First — AI is your default starting point for most tasks. Level three — AI Native Individual — you've restructured your entire workflow around AI. That three-hundred-PR engineer is a textbook Level three. Level four — AI Native Team — the whole team operates this way, with shared tooling, shared norms, and shared infrastructure that makes AI-native the default even for someone who just joined.

MOD Where are most teams today?

TL Most of the industry is at Level one. A few individuals on any given team have reached Level two. Level three is rare. Level four barely exists. And here's the critical insight: getting people to Level three is a training problem — you teach them new workflows, new tools, new habits. Getting a team to Level four is a systems problem. Shared agent configurations. Shared prompt libraries. Automated pipelines so agents can ship without human bottlenecks. Documentation that captures mistakes so they're never repeated. Those are fundamentally different investments, and almost everyone is only making the first one.

MOD Let's talk about what stops organizations from making the jump. I'm guessing it's not the technology.

TL It's not. The tools exist today. The number one blocker is fear. Fear of breaking something. Fear of expensing a tool without permission. Fear of running an experiment that fails. Fear of looking like you're not doing "real work" because you're planning and verifying instead of typing. In most organizations, the immune system kicks in the moment someone tries to work differently. The process says no. The procurement cycle says wait. The culture whispers don't rock the boat.

MOD So how do you fix that?

TL You name it and you kill it — explicitly. Not with a vague innovation memo. With a specific, loud, leadership-backed mandate. One organization I know calls it "Be Pirates." Sanctioned at the director level and above. Experiment freely. Expense the tools. Break the rules if the rules are the blocker. And when someone experiments and it works, celebrate it publicly. When it doesn't work, celebrate the learning. Both outcomes compound.

MOD "Be Pirates." I like that. But doesn't that create chaos?

TL It would, without structure. You need the permission and the pipeline. A co-innovation process that takes ideas from discovery to prototype to production. A place to experiment safely — an agents playground, essentially — where teams can try things without risking live systems. And you need visibility. Small teams can become silos fast. So you make work visible by default, not by effort. Status gets harvested automatically. Demos get generated automatically. The human stays at the creative and decision layer, and the system handles the rest.

MOD You keep coming back to this idea that the human role goes up, not away.

TL That's the emotional contract, and it matters more than any framework. The goal is not to squeeze more out of people. It's to elevate what people spend their time on. Delegate the grind. Keep the creative, strategic, high-judgment work. If the transformation feels like burnout, it's failed — regardless of how much output you're getting. The standard I hold is: ten times the throughput while having more fun, not less. If you can't achieve both, something in the system is broken.

MOD That's a high bar. How do you make "fun" part of a serious transformation strategy?

TL By designing for it explicitly. There's a culture model I think about — nine principles. Five are individual: AI-first, push boundaries, write and share, fail fast, stay skeptical. Those get your people to Level three. But then you need four collective ones for Level four: build for the team, not just yourself. Make work visible by default. Have fun doing it. And elevate — don't grind. That last one is the test. If you're doing something an agent could do, stop and delegate. Spend your time at the highest level of thinking you're capable of. That's not a soft aspiration. That's an operating principle.

MOD Let's close with this. If someone listening to this is an engineering leader, a product leader, a VP — what do they do Monday morning?

TL Three things. First — measure honestly. Where are your people on the four-level framework? Not where they think they are. Where their workflows prove they are. That's your baseline. Second — invest in your Level three proof points. Find your strongest engineers. Give them permission, tools, and time to completely rethink how they work. You need people on your own team who can show what's possible with real numbers, not slides. Third — start building Level four infrastructure now, even if most people aren't ready for it yet. Shared configurations. Automated last-mile pipelines. Institutional memory that agents can learn from. These are the systems that turn individual talent into team-level capability. And one more thing — make usage your north star. Track it weekly, not monthly. Because at the end of the day, the question isn't whether you're building the right thing. The question is whether anyone is using it. Usage is the leading indicator of everything else — value, adoption, retention, loyalty. If people aren't using it, nothing else matters.

MOD The shift to AI-native — not a technology upgrade, but a simultaneous reinvention of product, engineering, and culture.

TL Exactly. And the organizations that figure that out first don't just win. They make it structurally impossible for the rest to catch up.

MOD That's a strong note to end on. Thank you.

Based on emerging practices in AI-native product development and engineering, 2025.