Focus, Working Prototypes, and the New Shape of PM Work

PM Daily Digest

2026-04-02

Focus, Working Prototypes, and the New Shape of PM Work

By PM Daily Digest • April 2, 2026

This issue covers enduring product strategy lessons from Apple and WHOOP, the shift from documents and mockups to instrumented prototypes, and the execution habits PMs need as AI changes team structure. It also includes stakeholder-management tactics, career advice, and practical tools for design, discovery, and collaboration.

Big Ideas

1) Product fundamentals are holding up in the AI era

Tony Fadell says the fundamentals have not changed: start with the user, not the tech; focus by saying no; make hardware, software, and services work together; sweat the details; debate hard and then commit; build for the long term. He adds that winning companies build things people actually use and cannot imagine living without [1].

> "Start with the user, not the tech… Focus is everything… Details are the product." [1]

Scott Belsky makes a similar point through WHOOP: the team lived and breathed the product and its aspirational customer, stayed centered on physical health, readiness, and performance despite pressure to broaden, and cared about design from day one [2].

Why it matters: New tooling changes how teams build, but these sources still point back to the same advantage: sharp focus on a real user problem, a coherent end-to-end experience, and disciplined scope [1, 2].

How to apply: Define the user problem before discussing the technology, write

down what you will *not* build, and review the full experience end to end—including the small details users will remember [1, 2].

2) PM work is moving from mockups and PRDs to working, instrumented artifacts

Sachin Rekhi argues that customer discovery with functional prototypes plus PostHog is much better than asking for feedback on Figma mockups because it reveals what users actually do, not what they guess they might do [3]. His Walkman example is blunt: people called a yellow device sporty, but when they could take one home, they chose black [3].

A related community write-up described an Anthropic-style loop: skip the PRD, build a working prototype in hours, ship it internally, watch usage, and iterate based on what people actually do. The same post cites claims that 90% of code is AI-written and engineers ship 2,000-3,000-line AI-generated PRs [4]. Product School's discussion of the new Figma/Codex integration points in the same direction: design and code can now move back and forth in a round trip workflow, with faster iteration and cleaner handoffs [5].

Why it matters: Discovery, planning, and handoff get stronger when the team can react to behavior in a working product instead of opinions about a static artifact [3, 4, 5].

How to apply: Use working prototypes when you need to learn quickly, observe usage before over-polishing, and keep the loop tight between design and code. But keep the scope situational: commenters warned that this model fits controlled environments better than every organization, and that shipping more can create adoption fatigue if users cannot absorb the changes [6, 7].

3) AI is increasing leverage, but also raising the bar on judgment

At Block, product leadership says a given roadmap will need fewer engineers, designers, and PMs; PMs and designers are already shipping PRs; and Builder-Bot is building some features to 100% completion and many to 85-90% before a human finishes the last 10% [8]. Workflows are also changing from linear execution to managing many agents in parallel, and the organization has shifted to smaller squads of 1-6 people with fewer layers [8].

John Cutler adds the caution: AI lowers unproductive cognitive load and helps more people make progress, but it can also let people get ahead of their skis, while experienced builders feel their days compress into nonstop hard decisions [9]. Product School makes a similar point from the design side: as coding gets faster, the bottleneck can move to thoughtful design and deciding what to build in the first place [5].

Why it matters: The scarce skill is shifting from doing every task by hand to deciding what deserves attention, review, and slower thinking [8, 9, 5].

How to apply: Limit the number of concurrent AI threads you can realistically review, match task difficulty to the skill level of the person using the tools, and protect time for the harder product choices that speed alone will not answer [9, 5].

Tactical Playbook

1) Run behavior-based discovery in five steps

1. Build a functional prototype in a tool such as Bolt, Lovable, Reforge Build, Magic Patterns, or Claude Code [3].
2. Integrate PostHog [3].
3. Instrument the key user actions you care about [3].
4. Review return behavior and action data through DAUs/WAUs, retention curves, and action metrics dashboards [3].
5. Add qualitative context with post-usage surveys, session replays, and heatmaps [3].

Why it matters: This stack turns discovery from 'what do users say?' into 'what did users actually do?' [3].

How to apply: Use surprises—drop-offs, unexpected clicks, low return rates, or mismatches between surveys and behavior—as the starting point for the next round of interviews or iteration [3].

2) Use a facts-vs-stories loop in stakeholder conflict

Pippa Topp describes a common PM failure mode: defensiveness in meetings, where a request triggers a private narrative and the PM reacts emotionally instead of getting curious [10]. Her coaching move is to separate facts from the story you are telling yourself, then create space between stimulus and response [10].

Why it matters: This is a practical way to stop reactive communication before it creates an unproductive back-and-forth [10].

How to apply: When a request bothers you, pause and ask: what happened factually, what meaning am I adding, and what question would help me understand the request better? In forced decisions, acknowledge the discomfort, invite challenge, and if the decision stands, ask for 'disagree and commit' rather than fake agreement. The same empathy skill can be extended beyond customers to stakeholders and peers [10].

3) Fix decision-loss after meetings by standardizing where work lives

One product design discussion described a familiar post-meeting mess: notes, summaries, Jira or Linear tasks, Notion or Docs decisions, and Slack follow-ups all live in different places. Later, teams forget why a decision was made, who owns what, what the next step was, and even which meeting produced the

outcome [11]. A related remote-team post says the root problem is often not low communication volume, but the lack of a shared communication system—status updates in random places, decisions in private chats, unclear async rules, and escalation only after something is already late [12].

Why it matters: Teams often lose execution quality *after* the meeting, not during it [11, 12].

How to apply: Give status updates one home, record every decision with rationale, owner, next step, and meeting origin, make async expectations explicit, and define escalation rules before work slips [11, 12].

4) Coach soft-skill gaps like any other capability

Topp uses the conscious competence ladder to move people from unconscious incompetence to conscious awareness, then practice, and eventually automatic skill [10]. Her methods include life-story reflection, real-time observation, and assessment tools that reveal behavioral patterns and empathy misses [10]. At the team level, she suggests choosing a behavior such as curiosity, defining what good and bad versions of it look like, and measuring it through observation [10].

Why it matters: Emotional intelligence becomes coachable when it is translated into visible behaviors instead of vague feedback [10].

How to apply: Pick one behavior to improve this quarter—curiosity is a good candidate—write down what it looks like in meetings, and review examples in coaching or retros until the pattern becomes easier to notice in real time [10].

Case Studies & Lessons

1) WHOOP stayed narrow and became category-defining

Scott Belsky says WHOOP became an industry-defining product by staying close to its aspirational customer, focusing on physical health, readiness, and performance even when outside pressure pushed broader feature requests, and caring about design from day one [2].

Takeaway: Protecting a narrow performance focus can beat feature sprawl when the target user is clear [2].

2) Block changed both how products ship and how teams are shaped

At Block, PMs and designers shipping PRs is routine, BuilderBot can autonomously merge PRs and often build features to 85-100%, and the company now uses smaller, more fluid squads, fewer layers, and a functional product structure under one head of product [8].

Takeaway: If AI changes throughput, org structure and operating cadence need to change with it—not just individual tool use [8].

3) The Walkman study showed why behavior beats stated preference

In Sony's Walkman study, users described the yellow version as sporty and the black version as boring, but when they had to choose one to take home, everyone picked black [3].

Takeaway: Treat user language as signal, but trust real choice behavior more when the two conflict [3].

4) A junior PM's growth came from self-awareness, not more process

Topp described coaching a junior PM who wanted to retreat back to delivery management because the discomfort of product work felt like a sign she was in the wrong role. By unpacking the stories she was telling herself and connecting them to earlier life experiences, the PM gained confidence, communicated more clearly, built trust, navigated stakeholders better, and later took a more senior role [10].

Takeaway: What looks like role mismatch can sometimes be unprocessed discomfort in a new responsibility set [10].

Career Corner

1) Curiosity is becoming a practical PM skill

Product School's advice to PMs was simple: stay curious and play with the tools. The speakers pointed to concrete PM use cases: collating customer feedback into a prioritized backlog, inspecting merged code changes to draft personal release notes, and querying code behavior to understand business implications [5]. At Block, that shift is already concrete enough that PMs shipping PRs is described as routine [8].

Why it matters: Practical tool fluency now changes how independently a PM can investigate, prototype, and communicate [5, 8].

How to apply: Start with real work you already own—feedback synthesis, release notes, or product logic questions—before expanding into bigger builds or agent workflows [5].

2) Resilience starts with self-belief and balanced empathy

Topp says the missing foundation for stressed product leaders is often self-belief, and that resilience improves when people stop tying every challenge directly to their identity [10]. She also describes her own arc from being judgy and competitive, to becoming so empathetic that leadership suffered, and then finding a better balance between compassion and accountability [10].

Why it matters: PMs need empathy, but not at the cost of avoiding hard calls or accountability [10].

How to apply: Look for places where external validation is driving your behavior, and check whether your empathy is helping the team understand a decision—or just helping you avoid conflict [10].

3) Interview companies as hard as they interview you

One PM candidate described turning down a role at a company that wanted to build a SaaS-plus-AI product but had no roadmap, no firm plan, no target audience, and no real product specifics, while expecting the PM to think up the idea and implement it [13]. The candidate's concern was that this setup pushes too much uncertainty and politics onto one hire without even a rough product direction [13].

Why it matters: Undefined ownership can look like autonomy at first and impossible scope later [13].

How to apply: Before accepting a role, ask what problem the company already understands, who the target user is, what rough product direction exists, and what decisions the PM will actually own [13].

Tools & Resources

1) PostHog + functional prototype stack

This is the most concrete discovery stack in the notes: prototype in Bolt, Lovable, Reforge Build, Magic Patterns, or Claude Code, then layer in PostHog instrumentation, action metrics, retention curves, surveys, session replays, and heatmaps [3].

Use it for: early product discovery where behavior matters more than opinions [3].

2) Figma Codex round-trip workflow

OpenAI and Figma's new integration allows builders to start in design or code and move back and forth, with faster iteration and a cleaner design-to-ship handoff [5]. Product School's discussion also notes that a full design can appear in the product from a single prompt, speeding team velocity and shortening handoffs [5].

OpenAI & Figma on Killing the Design-to-Code Handoff (1:10)

Use it for: teams trying to compress design-to-code cycles without losing the ability to iterate in both directions [5].

3) OpenClaw multi-agent setup

OpenClaw lets you run multiple agents on one machine, each with its own identity, tools, crons, and workspace [14]. The onboarding flow includes files such as `soul.md`, `tools.md`, and `user.md`, and new agents can be added from the terminal [14]. Example workflows in the notes include project-management support for launches, PLG lead qualification, and converting repeated support tickets into documentation issues [14].

Use it for: recurring PM operations where the work is repetitive, schedulable, and easy to route to a specialized agent [14].

4) Collaboration options for Claude-generated docs

A community thread highlighted the pain of drafting docs in repo-based markdown, then needing richer formatting and comments for feedback [15]. Suggested solutions included exporting to Word on OneDrive, pushing directly to Confluence, generating target formats such as tables for Google Docs or Sheets, and using scripts to sync markdown to Google Drive while collaborators work in shared docs [16, 17, 18, 19]. One PM also built a custom tool to get Coda/Notion-style collaboration without leaving the repo [15].

Use it for: teams that want AI-assisted drafting without forcing every reviewer into GitHub-native workflows [15, 19].

Sources

1. X post by @tfadell
2. X post by @scottbelsky
3. X post by @sachinrekhi
4. r/ProductManagement post by u/marsel040
5. OpenAI & Figma on Killing the Design-to-Code Handoff
6. r/ProductManagement comment by u/utzutzutzpro
7. r/ProductManagement comment by u/*CaptRondo*
8. "We're Not Writing Code by Hand Anymore. That's Over." | Owen Jennings & David Haber - The a16z Show
9. AI Is Changing Who Builds Software—But At What Cost? | John Cutler at ShipSummit 2026
10. How to build emotional intelligence in product management - Pippa Topp (CPO, giffgaff)
11. r/product_design post by u/tricky_trick_52
12. r/prodmgmt post by u/Bitrix_24
13. r/ProductManagement post by u/Clear_Measurement_32
14. OpenClaw: The complete guide to building, training, and living with your personal AI agent
15. r/ProductManagement post by u/n3a-a4u
16. r/ProductManagement comment by u/DarkFlowerPewPew
17. r/ProductManagement comment by u/MockStarNZ
18. r/ProductManagement comment by u/Coramoor_
19. r/ProductManagement comment by u/mtn_coffee_drinker