Operational Strategy, Invisible Inventory, and Hard Portfolio Calls

PM Daily Digest

2026-03-10

Operational Strategy, Invisible Inventory, and Hard Portfolio Calls

By PM Daily Digest • March 10, 2026

This brief covers how PMs are tightening strategy for AI-accelerated teams, redesigning products around hidden capability discovery, and making tougher portfolio and career choices - from agent readiness to sunsetting flatliners.

Big Ideas

1) Strategy has to become an operating system, not an annual document

AI is speeding up both builders and PMs. Engineers and designers can do far more with tools like Cursor and Claude Code; PMs can prototype quickly, write evals, and even push PRs into engineering review. That makes directional clarity more important, not less. Aakash Gupta argues that if 9 out of 10 engineers and designers cannot explain the strategy, while a typical 5 engineer / 1 designer / 1 PM team costs about $1.4M fully loaded, the company is burning money. Common failure modes are strategies that are too long, vague, detached from execution, or too static [1].

- **Why it matters:** Faster execution widens the downside of bad direction and narrows the time available to correct it [1].
- **How to apply:** Treat strategy as a short, regularly updated decision-making tool that helps the team choose, sequence, and say no [1].

> "Can your engineer or designer explain the strategy in 30 seconds? Can they make decisions based on it? Does it help them say no to things?" [1]

2) In AI products, the new design problem is capability discovery

Enterprise products have always taught users three things: the interface, the domain, and the benefit. Conversational interfaces make interface teaching almost disappear and make domain teaching easier through plain language, but they make benefit teaching harder because the full capability surface is invisible behind a text field. Users can end up having a functional interaction that uses only a narrow slice of what the product can do, while their prior mental model narrows the questions they ask. Suggested prompts help briefly, but as a small static menu they do little to expand the frame [2].

- **Why it matters:** If capability stays invisible, differentiated product value stays invisible too [2].
- **How to apply:** Design for discovery and judgment: surface the right capability at the right moment, and create feedback loops so the product gets better with use rather than acting like a one-off chat box [2, 3].

> "The interface was the product. The capability is the product now. And capability that stays invisible is as good as absent." [2]

3) Agents are becoming a real user segment

For agent-facing products, Aakash Gupta argues the API, CLI, and MCP server are parallel layers rather than a maturity sequence: API for bulk operations and latency control, CLI for composability, MCP for discoverability and multi-client reach. He also argues agents need discoverability, programmatic auth, structured I/O, idempotency, and rate limits, and that the fix is to treat the agent as a first-class user with a PM who owns the experience [4].

- **Why it matters:** If one of those layers or primitives is missing, agents can route around your product to one that is easier to use [4].
- **How to apply:** Stop treating agent access as a side integration; define the agent journey, owner, and roadmap explicitly [4].

4) AI raises the cost of indecision

Shreyas Doshi highlights a simple tradeoff: a leader who makes a B+ decision today may beat the leader with A+ product sense who takes a week longer. Scott Belsky gives the organizational version of the same idea, calling the backlog of unmade decisions "organizational debt." His prescription is to prompt decisions or at least deadlines, run AI change through protected pilots with learning KPIs, and socialize new ways of working until they become obvious. He expects more process to be offloaded to compute, leaving humans to contribute taste and agency [5, 6].

- **Why it matters:** As more process moves to compute, slow consensus and process buildup become a bigger drag on product velocity [6].
- **How to apply:** Prompt the decision, or at least a deadline for it; use pilots with learning-focused KPIs before hardening new process [6].

Tactical Playbook

1) Build an AI-era strategy that survives contact with execution

1. Start with the seven elements: **Objective, Users, Superpowers, Vision, Pillars, Impact, Roadmap** [1].
2. Treat them as sequential but iterative; loop back as you learn [1].
3. Check for the four failure modes: too long, too vague, too detached from daily work, and too static [1].
4. Pass the 30-second test: an engineer or designer should be able to explain it, make decisions from it, and use it to say no [1].

   "If not, you have a document, not a strategy." [1]

2) Design AI onboarding around benefit teaching, not just interface reduction

1. Separate what the user must learn about the **interface**, the **domain**, and the **benefit** [2].
2. Assume the blank text field hides inventory; identify the capabilities users will never discover on their own [2].
3. Do not rely on a few static suggested prompts to solve discovery; they help briefly but quickly plateau [2].
4. Add an investment loop so the product stores value and improves through feedback and repeated use [3].
5. Use personalization as **persuasion** - helping users do what they want to do - not coercion [3].

3) Run AI adoption as a protected operating change

1. Start with pilots and play, not blanket mandates [6].
2. Give teams learning KPIs so they are rewarded for insight, not punished for early failure [6].
3. Use collapsed-stack teams or dual-role operators where possible to speed tool adoption and decision flow [6].
4. Keep destroying outdated process while new process is created; otherwise organizational debt accumulates [6].
5. Force a decision, or at least a decision deadline, when issues stall [6].

4) Prepare your product for agents in one quarter

1. **This week:** run the five-question audit and ship an `AGENTS.md` file [4].
2. **This month:** stand up a read-only MCP server and list it on PulseMCP [4].
3. **This quarter:** add approval flows, agent analytics, and agent-specific pricing [4].
4. Build the API, CLI, and MCP layers in parallel, not one after another [4].

5. Verify the basics: discoverability, programmatic auth, structured I/O, idempotency, and rate limits [4].

Case Studies & Lessons

1) Teresa Torres chose audience fit over easy revenue

Teresa Torres describes shutting down a $19/month community membership that was growing and generating reasonable revenue because it attracted low-effort questions, cannibalized courses and books, and pulled her away from the audience she wanted: people willing to invest in learning. She removed monthly subscriptions and kept annual only, explicitly accepting slower growth for better audience alignment [7].

- **Lesson:** Revenue can be real and still be strategically expensive if it trains the wrong user behavior or weakens your better products [7].

2) She also cut a product worth 40% of revenue

Torres says her deep-dive courses represented 40% of revenue, but the format had weak B2B fit and unstable cohort economics on the direct-to-consumer side, leading to cancellations, refunds, and administrative overhead. She sunsetted the cohort format and replaced it with two experiments: on-demand consumer courses and a subscription for corporate leaders to coach teams [7].

- **Lesson:** Stable revenue can hide format-market mismatch. The right question is not just "is this profitable?" but "is this the best use of time and team?" [7].

  > "I got to burn the ships." [7]

3) Sold out did not mean optimized

Petra Wille describes rethinking Product at Heart even though the event routinely sold out. The team felt the existing half-day format underused the value of putting about 60 product leaders together, so they did lightweight interviews and redesigned it into a two-day experience despite uncertainty about time commitment and pricing [7].

- **Lesson:** Strong demand is not proof that the current format is best; it may only show that the underlying need is real [7].

4) Portfolio governance ideas worth borrowing

Across the Teresa/Petra discussion, four operating mechanisms stand out: keep a visible **sunsetting** column on the taskboard, use **H1/H2/H3** horizons so replacement bets are already in motion, make sunsetting decisions one level above the product team, and normalize the fact that even successful products have life cycles [7].

Career Corner

1) Show product sense before anyone asks for it

One AI PM candidate stood out by watching three hours of TikTok videos from coaches serving small businesses, then bringing firsthand user insights to the first interview. The point was not the medium; it was the behavior. The candidate bypassed the company's framing, did lightweight user research independently, and demonstrated product sense rather than talking about it [8].

- **Why it matters:** In competitive PM hiring, evidence of judgment beats generic preparation [8].
- **How to apply:** Before interviews, go to the end user, build a small artifact, or bring real research. Do the work before you are asked [8].

2) Build AI fluency on tools that will matter at work

Sachin Rekhi advises PMs to spend their learning cycles on Claude Code rather than OpenClaw if the goal is practical AI fluency in day-to-day work. His reason: Claude Code combines strong agentic capability with broad enterprise adoption, and the related skill set - Skills, CLIs, MCPs, and adjacent workflows - is both productivity-enhancing and marketable [9].

- **Why it matters:** Some enterprises are explicitly hiring more junior AI-native talent to inject this fluency into everyday meetings and challenge legacy process [6].
- **How to apply:** Prioritize tools your current or next employer is likely to sanction, then learn the surrounding workflow surface, not just the interface [9].

3) Management is optional; clear thinking is not

Tony Fadell argues that many people should not be pushed into management just because it looks like the default ladder, especially if they prefer hands-on work, daily wins, or are not energized by people leadership. At the same time, Shreyas Doshi argues that long-term relevance in the AI age depends on evaluating logic rather than superficial tells about whether something "looks AI generated." Scott Belsky adds that the human edge will center more on taste and agency [10, 5, 6].

- **Why it matters:** Career progression is becoming less about title conformity and more about judgment, fluency, and role fit [10, 6].
- **How to apply:** Choose the ladder intentionally, then practice reviewing AI output for reasoning quality instead of style markers [10, 5].

Tools & Resources

- **How to Build Product Strategy in the Age of AI: Step-by-Step with Claude Code** — a compact strategy template: Objective, Users, Superpowers, Vision, Pillars, Impact, Roadmap, plus the anti-pattern check and 30-second test [1].

- **The Interface Was the Product** — useful if you're designing AI-native workflows and need a sharper lens for interface teaching vs. benefit teaching [2].
- **AGENTS.md + read-only MCP + agent analytics/pricing roadmap** — a practical starter set if you expect agents to use your product, not just humans [4].
- **AI Productivity course** — the course link Sachin Rekhi shared alongside his advice on Claude Code fluency [11, 9].
- **The Messy Middle of AI** — Scott Belsky's interview on organizational debt, collapsed-stack teams, pilots, and the role of taste and agency in AI adoption [6].
- **From Building Habits to Breaking Limiting Beliefs with Nir Eyal #beyondbelief** — a useful refresher on the Hook Model, the investment phase, and the persuasion-vs.-coercion boundary for habit-forming products [3].

Sources

1. How to Build Product Strategy in the Age of AI: Step-by-Step with Claude Code
2. The Interface Was the Product
3. From Building Habits to Breaking Limiting Beliefs with Nir Eyal #beyondbelief
4. substack
5. X post by @shreyas
6. The Messy Middle of AI
7. Kill Your Darlings - All Things Product with Teresa & Petra
8. substack
9. X post by @sachinrekhi
10. X post by @tfadell
11. X post by @sachinrekhi