AI-native product work: agent-ready surfaces, governance realities, and pricing under compute constraints

PM Daily Digest

2026-03-03

AI-native product work: agent-ready surfaces, governance realities, and pricing under compute constraints

By PM Daily Digest • March 3, 2026

This edition covers the biggest PM shifts emerging from AI: enterprise governance and IT re-centralization, designing products for agent users via MCP, and why AI pricing is moving away from flat subscriptions. It also includes tactical playbooks for quality/observability, AI-assisted discovery, value-based metrics, and a practical "marketing content as code" automation pipeline.

Big Ideas

1) AI in the enterprise is shifting power (and responsibility) back toward IT/security

AI tools aren't just creating more "shadow IT"—they're enabling employees to **build unapproved workflows/tools** quickly, turning shadow IT into "shadow software development" [1]. That raises the stakes for governance: prompts and ad-hoc agent workflows are harder to revoke/control than a SaaS license, with implications for security, data governance, and audit trails [1].

At the same time, as AI tools get connected to company data, governance questions become centralized again (what data is accessed, who authorized integrations, whether prompts are stored), pulling IT back into the conversation as the owner of data/access policies [1].

Why it matters: If your product can't credibly address IT/security requirements, it may hit a ceiling that wasn't as hard in a pure PLG era [1].

How to apply: Treat "governable access to data + auditable workflows" as a first-class product requirement (not just an enterprise checklist) [1].

---

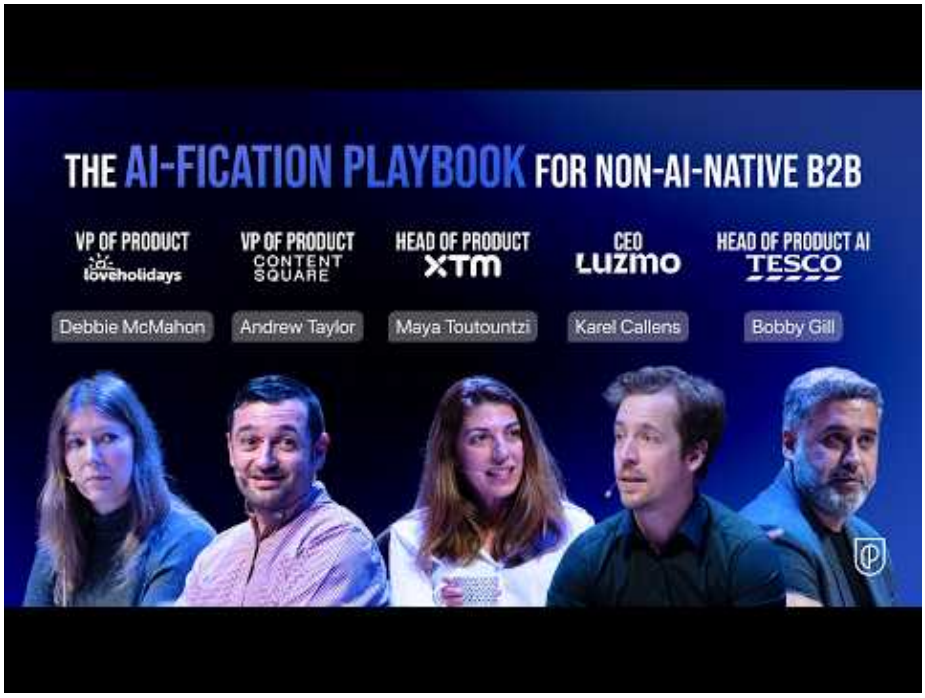
2) "Agent reachability" is becoming table stakes (MCP + agent-ready product surfaces)

As AI orchestration layers become the interface across tools, the question shifts from "do you have an API?" to "are you reachable from the agent?" [1]. MCP (Model Context Protocol) is described as an emerging standard for models interacting with external tools/data sources natively [1], and one speaker called MCP "pretty quickly the de facto way in which you get data out to agents" [2].

This aligns with a broader product shift: in the next few years, many users may not log into your UI the way they do today—they'll have agents do that on their behalf, pushing teams to see their product less as an interface and more as infrastructure [2].

Why it matters: Products that aren't built for agent consumption risk becoming less relevant as workflows move to AI layers [2, 1].

How to apply: Design for agents and humans in parallel—prioritize structured outputs, reliable tool access, composable functions, and strong observability (including confidence levels / finished vs. pending states) [2].



#productcon London'26 | Bringing AI Capabilities to Non-AI Native Products (15:27)

---

3) AI product pricing is being reshaped by extreme cost variance across users

Traditional SaaS economics assume 70–80% gross margins because one more subscriber costs almost nothing; AI products incur compute cost on every prompt, making your best users your most expensive users [3]. One pricing analysis of 50 top AI startups highlights that in most AI products, the P90 user costs **10–40×** more than the P50 user (while paying the same subscription), creating a growing subsidy from light to heavy users [3].

The same analysis argues that "pure flat pricing is dying," with nearly half of companies using two or three models simultaneously [3].

Why it matters: Flat subscriptions can hide margin cliffs until usage patterns shift (or agents increase volume), forcing abrupt pricing changes.

How to apply: Before setting price, estimate your **cost distribution from P10 to P90**; if you can't, you're not ready to price [3].

---

4) Product focus remains a durable counterweight to complexity

Tony Fadell frames the renewed love for iPod as a lesson about focus: no notifications, no feeds, no algorithm—"1,000 songs in your pocket" and "one job. Done well." [4]. He emphasizes the lesson isn't about whether Apple revives the device; it's about why focus matters [4].

Why it matters: As teams add AI capabilities (and as interfaces collapse into chat/agents), "one job done well" can still be a differentiator users understand immediately.

How to apply: Pressure-test roadmap additions against a simple question: do they strengthen the core job, or blur it into a multi-job bundle?

Tactical Playbook

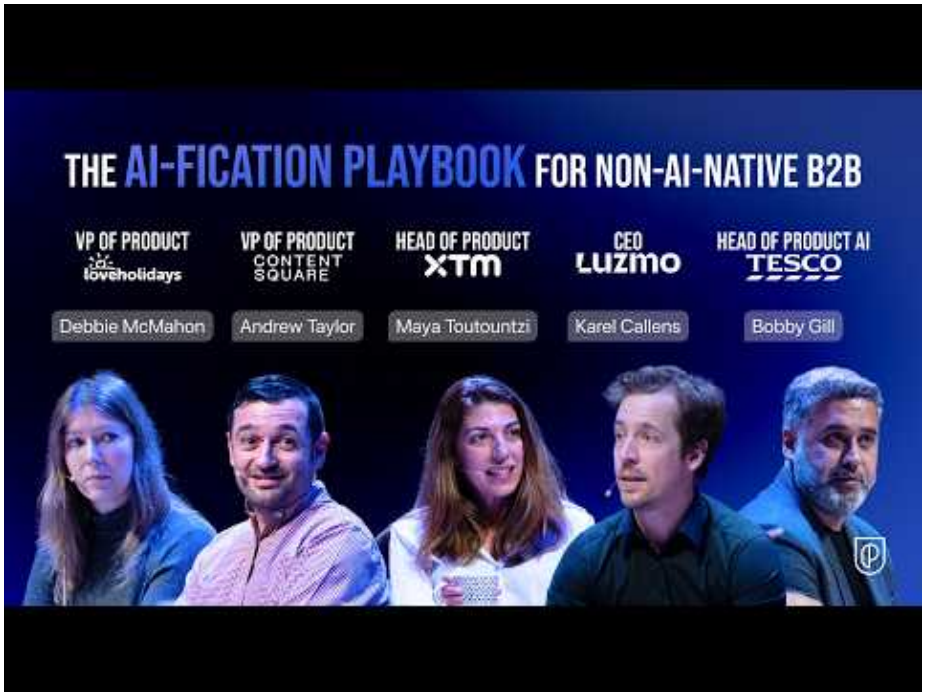
1) Reduce AI skepticism by making quality observable—and building tight feedback loops

A concrete playbook for overcoming skepticism: **obsess about quality** and operationalize feedback loops in beta. One team (Contentsquare's Sense Analyst) added thumbs up/down on every AI response; every thumbs down pinged

the team in Slack along with traces/logs, enabling fast prompt/system iteration and measurable improvement [2].

A related approach: make AI observable with mechanisms like quality scoring that explains performance, start with low-risk content and human-in-the-loop, then iterate while showing how each version improves over time [2]. For reliability, constrain model behavior (e.g., don't let it generate SQL freely if it can hallucinate; instead constrain it through a query engine) and make generation traceable so users can see how results were produced and adapt them [2].

Step-by-step 1. **Instrument per-response feedback** (thumbs up/down) and route negatives to the owning team with traces/logs [2]. 2. **Make outputs inspectable** (quality signals + "how it was generated") to reduce black-box fear [2]. 3. **Constrain high-risk actions** with guardrails and explicit operating boundaries [2].



#productcon London'26 | Bringing AI Capabilities to Non-AI Native Products (4:17)

---

2) Use AI to unblock discovery (where PMs are becoming the bottleneck)

Sachin Rekhi argues that AI has accelerated delivery, but not discovery—and cites Andrew Ng's view that PM is becoming the bottleneck [5]. His response is

to build AI workflows directly into discovery, including: analyzing surveys, automating survey programs, automating "feedback rivers," developing interview scripts, synthesizing interviews, AI-moderated interviews, generating synthetic feedback, discovery via prototypes, analyzing metrics, and automating metric analysis [5].

Step-by-step 1. Start by **centralizing customer inputs** (surveys + feedback streams) and run structured analysis on them [5]. 2. Use AI to **draft and synthesize interviews** (script → synthesis), keeping humans responsible for what decisions are made from the outputs [5]. 3. Add AI-assisted **metrics analysis** to close the loop between what customers say and what they do [5].

Where to see it live: Rekhi is demo'ing these workflows on March 5th in Mountain View (registration link shared) [5, 6].

---

3) Make dashboards reflect value (not just activity)

A recurring critique: many SaaS dashboards track activity (clicks, sessions, feature usage) rather than user value [7]. The thread challenges teams to define and measure "customer value created" [7] and notes that logging in 20 times without getting closer to the user's outcome isn't success [7].

Step-by-step 1. Write the **user outcome** your product exists to deliver, in plain language. 2. Define what "getting closer" looks like (progress signals) so "20 logins" can't be mistaken for success [7]. 3. Add at least one metric that directly answers: "Did users achieve/advance the outcome?" (not just "Did they use the product?") [7].

---

4) Keep launch marketing current by treating marketing content as code (and automating the pipeline)

A PM case study describes falling behind on the "software → marketing" work (website updates, docs, screenshots, blog posts) even though the team ships features quickly with coding agents [8]. Their fix: treat marketing content like code in a single git project, eliminate separate tooling (CMS, demo software), and use Claude Code/Codex to generate feature descriptions in Markdown and keep website copy in YAML [8]. They also had agents write Playwright scripts for automated screenshots/videos (light/dark mode, realistic cursor movement), with assets linked to YAML and kept up to date by agents [8].

On release, a "marketing skill" kicks off the process; the PM reviews/edits diffs, iterates with coding agents, then deploys via git push—Cloudflare deploys the site and GitBook picks up docs, with a human-in-the-loop command center called Nimbalyst orchestrating the pipeline [8]. Reported outcome: the website

stays current, docs match the product, and screenshots reflect the up-to-date UI [8].

Step-by-step 1. Move marketing/docs assets into a **single versioned repo** (treat content as code) [8]. 2. Automate "truth maintenance" (docs + copy + screenshots/videos) with agents + Playwright, then require **human review on diffs** before deploy [8]. 3. Deploy through a repeatable path (git push) so updates ship as reliably as code changes [8].

Case Studies & Lessons

1) Pricing failures (and mitigations) when heavy users become economically dominant

- **Replit:** Gross margins reportedly went from **36% to -14%** in two months after launching a more autonomous AI agent that consumed more LLM resources than pricing covered—while product, pricing, and team otherwise stayed the same [3].
- **Cursor:** Moved from a flat "500 requests/month" model to a credit pool; one developer burned the full monthly allocation in a day, generating a **$7,225 invoice**, followed by a CEO apology and a plan rename from "Unlimited" to "Extended" twelve days after launch [3].
- **Anthropic:** Used tiers ($17/$100/$200) mapped to different user personas (casual, power user, developer replacing an IDE), plus weekly rate limits affecting <5% of subscribers to push heaviest users to the API where per-token pricing covers compute [3].

Key takeaways 1. Model changes and autonomy shifts can change unit economics fast—even without "new features" in the traditional sense [3]. 2. If power users discover more ways to use the product, subsidy dynamics can worsen over time [3]. 3. Persona-based tiers + rate limits can be used to segment willingness-to-pay and steer high-cost usage to usage-based rails [3].

---

2) Enterprise adoption friction is shifting from "learning curve" to existential trust

One enterprise lens argues user resistance used to be dominated by a "literacy tax" (upskilling to learn the new tool) [1]. In an AI-native context, that resistance can flip to a psychological fear—"am I training my replacement?"—requiring trust-building and explicit positioning around augmentation vs. replacement [1].

Separately, conversational interfaces can reduce the UI-learning problem, but introduce a new challenge: benefit discovery. Without a browsable UI surface,

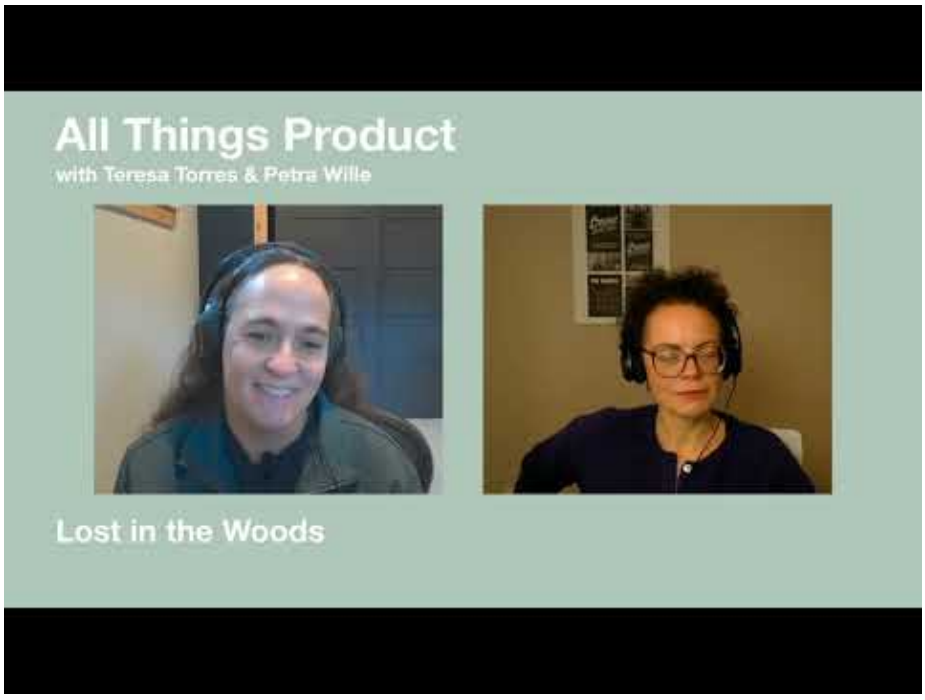
product teams must make capabilities legible through a blank text field—users won't discover value unless they know what to ask [1].

Key takeaways - Adoption work must include **trust design**, not just training and documentation [1]. - "Capability discoverability" becomes a product design problem in conversational UX [1].

---

3) A lightweight "getting un-lost" framework for product teams

In *All Things Product*, Teresa Torres and Petra Wille use *Lost Person Behavior* analogies to describe how product teams get "lost in the woods" and recover [9]. Examples include: - **Settle in place (do nothing):** sometimes correct in IT-mindset environments where teams lack business context and should escalate rather than wander (e.g., teams configuring third-party systems) [9]. - **Chasing shortcuts:** risky without terrain familiarity; discovery should help validate whether a shortcut is actually getting you where you want to go [9]. - **Over-relying on intuition ("taste"):** combine intuition with data/customer centricity and use judgment across conflicting inputs, rather than "blindly deciding north" [9].



Lost in the Woods - All Things Product with Teresa & Petra (14:56)

Career Corner

1) If you feel like the "lone champion," build change through shared habits (not just advocacy)

Teresa Torres notes that it can feel like you're the lone champion pushing for change, which is why she's organizing a 2026 group read of *Continuous Discovery Habits*—one section per month—to help people build real habits (not just read concepts) [10]. The program includes monthly reading guides with reflection questions and exercises, short videos to share with teammates, and quarterly live discussion sessions [10].

This month's reading covers Chapter 4: why visualizing what you know builds team alignment, plus a collaboration pattern that explores individual perspectives before forming a shared team perspective [10].

How to apply: Use "visualize what you know" as a practical alignment ritual: share what each function believes is true before converging on a unified team view [10].

---

2) Treat AI-assisted discovery as a core PM skill—not a side experiment

If delivery accelerates while discovery lags, PMs risk becoming a throughput bottleneck [5]. Rekhi's list of AI discovery workflows is a practical checklist for where to build leverage (surveys, interviews, prototypes, metrics) [5].

How to apply: Pick one discovery workflow to systematize (e.g., survey analysis or interview synthesis) and turn it into a repeatable motion your team can run every cycle [5].

Tools & Resources

1) PM Skills Marketplace (open source): a structured "skills + commands" system for PM work

The Product Compass launched an open-source **PM Skills Marketplace** on GitHub with **65 PM skills** and **36 chained workflows** across **8 Claude plugins**, designed for Claude Code and Claude Cowork (skills also compatible with other AI assistants) [11]. Repo: https://github.com/phuryn/pm-skills [11].

- **Skills** are building blocks (domain knowledge, frameworks, guided workflows) that load automatically when relevant [11].
- **Commands** are user-triggered workflows invoked with `/command-name` that chain skills into an end-to-end process (e.g., `/discover`) [11].

Installation (Claude Cowork): Customize → Browse plugins → Personal → "+" → Add marketplace from GitHub → enter `phuryn/pm-skills` (installs all 8 plugins) [11].

---

2) AI workflows for discovery (live demo)

Sachin Rekhi is demo'ing ten AI discovery workflows live on March 5th in Mountain View [5]. Registration link shared: https://events.ticketleap.com/tickets/danolsen/sachin [6].

---

3) Pricing + AI agents reading for PMs

Aakash Gupta shared an "AI agents guide for PMs" resource link: https://www.news.aakashg.com/p/ai-agents-pms [3].

Sources

1. Enterprise SaaS Truisms, Revisited
2. #productcon London'26 | Bringing AI Capabilities to Non-AI Native Products
3. substack
4. X post by @tfadell
5. X post by @sachinrekhi
6. X post by @sachinrekhi
7. r/ProductManagement post by u/Serious-Hope2707
8. r/prodmgmt post by u/StravuKarl
9. Lost in the Woods - All Things Product with Teresa & Petra
10. X post by @ttorres
11. PM Skills Marketplace: The AI Operating System for Better Product Decisions