AI-speed product work: prototypes over decks, data-driven moats, and real-time onboarding interventions

PM Daily Digest

2026-03-02

AI-speed product work: prototypes over decks, data-driven moats, and real-time onboarding interventions

By PM Daily Digest • March 2, 2026

AI-speed shipping is compressing product cycles and changing what "good" looks like—toward near-term prototypes, fast iteration, and trust built through responsiveness. This edition also offers concrete playbooks for real-time onboarding interventions, turning AI defensibility into a data plan, and building career leverage as roles become more builder- and direction-oriented.

Big Ideas

1) AI-speed execution is compressing "vision" into near-term prototypes

One emerging pattern: product teams are relying less on long-range, polished vision decks and more on **3–6 month prototypes** that point fast execution in the right direction [1]. In parallel, craft work is shifting from heavy "mocking/prototyping" toward **pairing with engineering and implementation** [2].

Why it matters: When "shipping scrappy" gets easier, the scarce skill becomes setting direction that prevents lots of fast work from turning into lots of misaligned work.

[1] The design process is dead. Here's what's replacing it. | Jenny Wen (head of design at Claude)

[2] The design process is dead. Here's what's replacing it. | Jenny Wen (head of design at Claude)

How to apply: Make prototypes a *direction-setting artifact* (not just validation), and keep them close enough to reality that engineering can iterate quickly without re-litigating intent [3].

2) For AI products, defensibility increasingly hinges on proprietary data—beyond UI, prompts, or the model

A startup lens argued many AI companies aren't failing on execution/UX—they're failing because they don't have a moat [4][5]. A "fragile moat" pattern: **UI + prompt engineering + the same foundation model as everyone else**, with **no proprietary data** or **unique signal** [6].

> "You can clone UI. You can clone prompts. You can switch models. You can't easily clone years of structured domain data." [7]

The suggested defensibility stack: - **Proprietary training data** (domain-specific corpora others don't have) [8] - **Proprietary evaluation data** (measuring performance in a way competitors can't) [9] - **Proprietary workflow telemetry** (real interaction data that compounds over time) [10]

Why it matters: This reframes "data" from a vague advantage into a concrete roadmap: what you must instrument, store, and use to become a system (not a wrapper) [11].

3) "PMs as builders" is becoming more real—if leadership removes the access bottlenecks

One take: PMs should build **prototypes on the actual design system**, do small front-end polish, and sometimes ship an initial version directly in the codebase [12]. The same note argues that Claude Code "connects your PM directly into the codebase" and cites rapid adoption (including a claim that Claude Code went from zero to **$2.5B ARR in nine months**) [13].

The bottleneck is often organizational: teams get tool access but aren't connected to real systems due to **IT/security/privacy/regulatory constraints**, which "handicaps" strategy work when tools can't access analytics/BI/revenue metrics [14].

---

[3] The design process is dead. Here's what's replacing it. | Jenny Wen (head of design at Claude)
[4] r/startups post by u/Khade_G
[5] r/startups post by u/Khade_G
[6] r/startups post by u/Khade_G
[7] r/startups post by u/Khade_G
[8] r/startups post by u/Khade_G
[9] r/startups post by u/Khade_G
[10] r/startups post by u/Khade_G
[11] r/startups post by u/Khade_G
[12] [Your PMs should be building in the codebase.
[13] [Your PMs should be building in the codebase.
[14] [Your PMs should be building in the codebase.

Why it matters: If builder workflows and agent harnesses are becoming default, competitive advantage shifts to who can safely connect people + agents to real data and real delivery systems.

Tactical Playbook

1) Ship "research previews" without burning trust: a speed-and-responsiveness contract

A design leader described releasing Claude Cowork as a **research preview**—shipping despite flaws because the benefits outweighed the cons, *as long as the team commits to responding and iterating quickly* [15].

Step-by-step 1. **Label the release explicitly** as early/research preview, including known limitations [16]. 2. **Make an explicit promise to iterate** based on feedback (and treat that as part of the launch) [17]. 3. **Demonstrate follow-through quickly** (continuous shipping + visible improvements) to avoid brand trust erosion [18].

[15]The design process is dead. Here's what's replacing it. | Jenny Wen (head of design at Claude)
[16]The design process is dead. Here's what's replacing it. | Jenny Wen (head of design at Claude)
[17]The design process is dead. Here's what's replacing it. | Jenny Wen (head of design at Claude)
[18]The design process is dead. Here's what's replacing it. | Jenny Wen (head of design at Claude)

The design process is dead. Here's what's replacing it. | Jenny Wen (head of design at Claude) (24:52)

2) Design for non-deterministic AI by testing with real models + real users (not just mocks)

A practical argument: with evolving, non-deterministic AI models, you can't mock every state; you need to **use the actual models** and see real users try real use cases to discover what's valuable [19].

Step-by-step 1. Put a working version in front of users using the **actual model behavior** (not a theoretical flow) [20]. 2. Watch users attempt their real use cases; treat "use case discovery" as an expected output of testing [21]. 3. Iterate based on what users *actually attempt and succeed/fail at*, not just what they say they want [22].

3) Add real-time "stuck" interventions to onboarding (so you help users *before* they churn)

A PM building a "Life Guard for SaaS onboarding" observed that standard analytics (e.g., Mixpanel/PostHog) often only show who dropped off *after* the user is already gone [^4]. Their solution: a **stuck detection engine** that triggers interventions (e.g., a Slack alert to reach out, or an automated "Need a hand?" email) when a user hits a "stuck" state [^4].

Step-by-step 1. Define "stuck states" in onboarding (e.g., repeated retries, long idle periods, blocked steps) and detect them in-session [^4]. 2. Route a response: **human outreach (Slack alert)** for high-value accounts, or **automated help** for long-tail users [^4]. 3. Treat interventions as a discovery surface: catalog the stuck patterns and feed them into onboarding fixes and product changes [^4][^4].

4) Turn "moat" into an instrumentation plan (training, eval, telemetry)

If the wrapper-to-system shift depends on proprietary data, make it operational:

Step-by-step 1. Decide which category you can own: - Training data [23] - Evaluation data [24] - Workflow telemetry [25] 2. Identify the user workflow mo-

[19]The design process is dead. Here's what's replacing it. | Jenny Wen (head of design at Claude)

[20]The design process is dead. Here's what's replacing it. | Jenny Wen (head of design at Claude)

[21]The design process is dead. Here's what's replacing it. | Jenny Wen (head of design at Claude)

[22]The design process is dead. Here's what's replacing it. | Jenny Wen (head of design at Claude)

[23]r/startups post by u/Khade_G

[24]r/startups post by u/Khade_G

[25]r/startups post by u/Khade_G

ments that generate that data, and instrument them deliberately (not as an afterthought) [26]. 3. Store and structure it over time—because the claim is that **years of structured domain data** is hard to clone [27].

5) Unblock AI tool value by connecting them to real business metrics (and working through security)

A leadership warning: giving teams tool access but not connecting it to analytics/BI/revenue metrics due to constraints "totally handicaps the team" [28]. The same guidance emphasizes working through IT/security hurdles so teams can connect tools to data sources (analytics, research, support, CRM) and output systems (e.g., Linear/Jira) [29][30].

Step-by-step 1. Inventory the decision-critical systems the team needs (analytics/BI/revenue metrics—not just engagement) [31]. 2. Work with IT/security to enable access paths (MCP/API integrations were called out as essential) [32]. 3. Verify end-to-end usefulness: data in → documents/models out (PRDs, roadmaps, sizing models) [33].

Case Studies & Lessons

1) Claude Cowork: "10 days to ship" was only the final leg—prototypes came first

On Cowork, the team had "a bunch of different prototypes internally," tried many form factors, and then did ~**10 days** to move from internal state to something ready to ship externally [34]. They framed the launch as a research preview and emphasized **trust through speed**—ship early *and* show continuous improvement based on feedback [35].

Takeaways - If a product is going out early, the trust lever is not perfection—it's responsiveness and visible iteration [36]. - Don't let a single shipping metric ("10 days") erase the prototyping/learning investments that made shipping possible [37].

---

[26] r/startups post by u/Khade_G
[27] r/startups post by u/Khade_G
[28] [Your PMs should be building in the codebase.
[29] [Your PMs should be building in the codebase.
[30] [Your PMs should be building in the codebase.
[31] [Your PMs should be building in the codebase.
[32] [Your PMs should be building in the codebase.
[33] [Your PMs should be building in the codebase.
[34] The design process is dead. Here's what's replacing it. | Jenny Wen (head of design at Claude)
[35] The design process is dead. Here's what's replacing it. | Jenny Wen (head of design at Claude)
[36] The design process is dead. Here's what's replacing it. | Jenny Wen (head of design at Claude)
[37] The design process is dead. Here's what's replacing it. | Jenny Wen (head of design at Claude)

2) X platform iteration: growth instincts + fast iteration before polish (plus transparency)

Scott Belsky praised @nikitabier's "leader/product fit" advancing X through **growth instincts + quick iterations before polish**, engaging detractors, and "clearing abusers" with transparency features [^5]. He added that the product is "undeniably better" [^5].

Takeaways - Fast iteration can coexist with trust/safety work if transparency and abuse-handling are treated as product features (not just policy) [^5]. - Leadership behavior can set norms for how a platform is used ("setting [the] bit… through his own use") [^5].

3) SportsFlux: utility vs. retention—feature, product, or ecosystem?

A PM iterating on SportsFlux (a "discovery layer" hub for live sports links) worried about retention once users find a game and leave [^6]. They considered adding **real-time win probability** or **fantasy alerts** to create a second-screen experience [^6]. Replies highlighted that direction depends on monetization [^7], and asked whether cross-provider links imply "illegal streams" [^8].

Takeaways - Retention concerns should be grounded in *why* the user leaves (task completion vs. dissatisfaction) before expanding scope [^6]. - Monetization and legal/ethical constraints can be first-order product inputs—not things to "solve later" [^7][^8].

Career Corner

1) Three "interesting" talent archetypes to hire for (and become)

A hiring lens surfaced three archetypes: - **Strong generalists**: "block-shaped" (multiple ~80th-percentile skills) [38] - **Deep specialists**: a T-shape with an unusually deep spike (e.g., highly technical designers, or extremely strong visual/craft specialists) [39] - **"Craft new grad"**: early career, humble, fast learner; valuable because roles are changing and blank slates adapt quickly [40]

How to apply: In your growth plan, name which archetype you're optimizing for per role—and which archetype you're personally building toward (breadth vs. spike vs. learning velocity) [41].

[38] The design process is dead. Here's what's replacing it. | Jenny Wen (head of design at Claude)

[39] The design process is dead. Here's what's replacing it. | Jenny Wen (head of design at Claude)

[40] The design process is dead. Here's what's replacing it. | Jenny Wen (head of design at Claude)

[41] The design process is dead. Here's what's replacing it. | Jenny Wen (head of design at Claude)

2) Management that persists: direction + real engagement with the work

A design leader argued managers still matter "as long as there is a team," but the future manager is not "pure people management"; they must provide direction and engage with the work [42].

How to apply: If you manage PM/design/eng, take periodic IC rotations or hands-on time so you can empathize with how the work has changed and guide effectively [43].

3) "Low-leverage" work can be high-leverage when leaders do it

An example-driven point: leaders can create outsized impact by choosing seemingly low-leverage tasks (e.g., deep product dogfooding, reproducing bugs with engineers, even putting in PRs) because it builds shared context and signals standards [44]. The same discussion describes "roasting" as a possible signal of psychological safety when balanced with high standards [45].

How to apply: Pick one "hands-on" ritual you do consistently (e.g., dogfood + bug repro), and make it visible and collaborative [46].

Tools & Resources

1) Claude OKR Skill (GitHub): an OKR coach with five modes

A community-shared "Claude OKR Skill" positions Claude as an OKR coach with five modes: **Define**, **Refine**, **Check-in**, **Score**, and **Align** [^9]. It supports multiple OKR frameworks—Classic (Doerr/Google), outcome-based (Cagan/Perri), and hybrid—and defaults to Classic with outcome-oriented language [^9][^9].

- Repo: https://github.com/marfoerst/claude_okr_pm_kill [^9]

How to apply: Use **Refine** to clean up vague targets and "output-disguised-as-outcome" KRs, then use **Align** to flag orphan objectives and overloaded areas before the quarter starts [^9][^9].

---

[42] The design process is dead. Here's what's replacing it. | Jenny Wen (head of design at Claude)

[43] The design process is dead. Here's what's replacing it. | Jenny Wen (head of design at Claude)

[44] The design process is dead. Here's what's replacing it. | Jenny Wen (head of design at Claude)

[45] The design process is dead. Here's what's replacing it. | Jenny Wen (head of design at Claude)

[46] The design process is dead. Here's what's replacing it. | Jenny Wen (head of design at Claude)

2) Tooling stack called out for PM work connected to real systems

One suggested stack: - **Claude Code** (PMs building closer to the production codebase) [47] - **Claude Cowork** (strategy docs/PRDs/roadmaps across Slides/Keynote/PowerPoint/Excel, connected to data + delivery systems) [48] - **MCP/API integrations** to analytics, research, support, and CRM (to avoid "handicapping" decisions) [49][50]

How to apply: If you roll these out, treat "connected to real metrics + real workflows" as the success criterion—not just tool access [51].

Not full features. But prototypes on your actual design system. Polish on simple front-end changes. Sometimes the initial version of a feature. If your product team doesn't have access to these three tools, you're already behind.

1. Claude Code

This is the most powerful way to use an LLM. It connects your PM directly into the codebase. Not a separate prototype environment. Your actual elements, your actual design system. PMs building on top of production components, doing polish on simple front-end work, sometimes shipping the initial version themselves. Anthropic hit $14B in annualized revenue in February 2026, up from $1B fifteen months ago. Claude Code alone went from zero to $2.5B ARR in nine months. That adoption curve tells you how much value teams are extracting. If your PMs aren't connected to the codebase, they're living in the past.

1. Claude Cowork

Once you connect Cowork with MCP and API to your data sources (analytics, user research, customer support, CRM) and your output systems (Linear, Jira, your engineering team's coding agent), it becomes the best harness for creating strategy documents, PRDs, and roadmaps. Nothing else compares right now. It works in Google Slides, Keynote, PowerPoint. It builds Excels and impact sizing models. When Anthropic launched Cowork plugins, Thomson Reuters dropped 15.8% in a single day. LegalZoom fell 19.7%. The market is telling you what this tool means for knowledge work. Your team shouldn't be using ChatGPT or Gemini chat. They should have Cowork.

1. MCP or API into all key tools

People give their teams Code or Cowork access and then don't connect them to systems because of IT constraints, privacy, regulatory, compliance. It totally handicaps the team. If they're not plugged into analytics, business intelligence, actual revenue metrics and not just engagement data, they can't make strategic

[47] [Your PMs should be building in the codebase.
[48] [Your PMs should be building in the codebase.
[49] [Your PMs should be building in the codebase.
[50] [Your PMs should be building in the codebase.
[51] [Your PMs should be building in the codebase.

decisions. The best product leaders are working through the security hurdles and getting their teams connected to everything.

Here's your guides:

1. Claude Code: %5Bhttps://youtu.be/4nthc76rSl8?si=eYyGMAys1oGQzdr2%5D(https://youtu.be/4nthc76rSl8?si=eYyGMAys1oGQzdr2)
2. Claude Cowork: %5Bhttps://www.news.aakashg.com/p/you-should-be-using-claude-cowork-465%5D(https://www.news.aakashg.com/p/you-should-be-using-claude-cowork-465)
3. MCP + PM OS: %5Bhttps://www.news.aakashg.com/p/pm-os%5D(https://www.news.aakashg.com/p/pm-os)

PMs already want to use these tools. The bottleneck is leadership. Work through the IT and security hurdles. The teams that do will be winning in the market.

https://www.news.aakashg.com/p/you-should-be-using-claude-cowork-465](https://substack.com/@aakashgupta/note/c-221745926) [^4]: r/ProductManagement post by u/MammothAd137 [^5]: post by @scottbelsky [^6]: r/ProductManagement post by u/Suspicious-Turn2708 [^7]: r/ProductManagement comment by u/Superbureau [^8]: r/ProductManagement comment by u/iheartgt [^9]: r/ProductManagement post by u/Lazyyyyymaverick

---

Sources

1. The design process is dead. Here's what's replacing it. | Jenny Wen (head of design at Claude)
2. r/startups post by u/Khade_G
3. [Your PMs should be building in the codebase.

Not full features. But prototypes on your actual design system. Polish on simple front-end changes. Sometimes the initial version of a feature. If your product team doesn't have access to these three tools, you're already behind.

1. Claude Code

This is the most powerful way to use an LLM. It connects your PM directly into the codebase. Not a separate prototype environment. Your actual elements, your actual design system. PMs building on top of production components, doing polish on simple front-end work, sometimes shipping the initial version themselves. Anthropic hit $14B in annualized revenue in February 2026, up from $1B fifteen months ago. Claude Code alone went from zero to $2.5B ARR in nine months. That adoption curve tells you how much value teams are extracting. If your PMs aren't connected to the codebase, they're living in the past.

1. Claude Cowork

Once you connect Cowork with MCP and API to your data sources (analytics, user research, customer support, CRM) and your output systems (Linear, Jira, your engineering team's coding agent), it becomes the best harness for creating strategy documents, PRDs, and roadmaps. Nothing else compares right now. It works in Google Slides, Keynote, PowerPoint. It builds Excels and impact sizing models. When Anthropic launched Cowork plugins, Thomson Reuters dropped 15.8% in a single day. LegalZoom fell 19.7%. The market is telling you what this tool means for knowledge work. Your team shouldn't be using ChatGPT or Gemini chat. They should have Cowork.

1. MCP or API into all key tools

People give their teams Code or Cowork access and then don't connect them to systems because of IT constraints, privacy, regulatory, compliance. It totally handicaps the team. If they're not plugged into analytics, business intelligence, actual revenue metrics and not just engagement data, they can't make strategic decisions. The best product leaders are working through the security hurdles and getting their teams connected to everything.

Here's your guides:

1. Claude Code: %5Bhttps://youtu.be/4nthc76rSl8?si=eYyGMAys1oGQzdr2%5D(https://youtu.be/4nthc76rSl8?si=eYyGMAys1oGQzdr2)
2. Claude Cowork: %5Bhttps://www.news.aakashg.com/p/you-should-be-using-claude-cowork-465%5D(https://www.news.aakashg.com/p/you-should-be-using-claude-cowork-465)
3. MCP + PM OS: %5Bhttps://www.news.aakashg.com/p/pm-os%5D(https://www.news.aakashg.com/p/pm-os)

PMs already want to use these tools. The bottleneck is leadership. Work through the IT and security hurdles. The teams that do will be winning in the market.

https://www.news.aakashg.com/p/you-should-be-using-claude-cowork-465](https://substack.com/@aakashgupta/note/c-221745926) 4. r/ProductManagement post by u/MammothAd137 5. post by @scottbelsky 6. r/ProductManagement post by u/Suspicious-Turn2708 7. r/ProductManagement comment by u/Superbureau 8. r/ProductManagement comment by u/iheartgt 9. r/ProductManagement post by u/Lazyyyyymaverick