Claude Code Workflows, Jazz Teams, and Reconnecting PM Work to Reality

PM Daily Digest

2026-03-21

Claude Code Workflows, Jazz Teams, and Reconnecting PM Work to Reality

By PM Daily Digest • March 21, 2026

This issue focuses on four shifts in PM practice: AI-native workflows built on local context, product teams moving from rigid orchestration to small-team improvisation, a stronger discipline for linking work to business reality, and the continued importance of direct user understanding. It also includes concrete playbooks for stakeholder access, AI-assisted discovery, PM prototyping, and tool selection.

Big Ideas

1) AI is pushing product teams from "symphony" to "jazz"

Deb Liu argues that many product orgs were built like symphonies: defined roles, structured handoffs, detailed specifications, and careful preparation across product, engineering, design, analytics, research, and other functions [1]. AI changes that operating model by making it possible to spin up a prototype quickly, blur role boundaries, and work inside guardrails with more real-time improvisation [1]. Her leadership implication is equally important: leaders move from coordinating a fixed score to setting themes and guardrails for small, high-trust teams [1].

- **Why it matters:** The premium shifts from rigid orchestration toward faster adaptation and learning [1].
- **How to apply:** Ask whether your team is optimized for orchestration or adaptation, whether roles are too rigid, and whether PMs are waiting for engineering to build ideas that they could prototype first [1].

2) In B2B, product work naturally drifts away from business reality

> "We have to force the work back into connection with reality." [2]

Run the Business frames the gap between what product teams do and what the business needs as the most expensive dysfunction in B2B product companies, and argues that drift toward disconnection is the default [2]. The fix is not just better execution; it is better connection. That means coherent strategy across **legibility, synchronicity, composability, and affordability** [2], plus a metric ladder that links **growth, retention, and margins** to strategic levers and then to team north stars [2].

- **Why it matters:** Without this ladder, teams slide into feature-factory behavior, reactive planning, or "south star" metrics that look good on a dashboard but pull the product in the wrong direction [2].
- **How to apply:** Check whether each team north star clearly maps to a strategic lever and then to business KPIs. Look for the known anti-patterns: direct revenue accountability for product teams, one-hop feature-to-revenue logic, and "orphan" teams that do not tie to any business outcome [2]. The positive endpoint is creating users so successful they become advocates: "manufacturing champions" [2].

3) The strongest AI PM workflows are built on local context, not chatbot memory

Sachin Rekhi argues that Claude Code should be the primary AI productivity tool PMs focus on because it is built to generate PM artifacts such as documents and reports, works well with local markdown files, automates workflows through skills, agents, and commands, can run command-line tools, writes bespoke scripts, and keeps context portable instead of locked inside a proprietary chat history [3]. He says he has already shifted the majority of his product work to it [3].

Dave Killeen shows what this looks like in practice: session-start hooks preload weekly priorities, quarterly goals, and working preferences; markdown files exist for each project, person, and company; meeting transcripts append automatically; mistakes get logged into a reusable file; and AI filters a large information diet down to novel, contrarian signals that matter [4].

- **Why it matters:** Context compounds. By day 30, the system can track relationships, commitments, and meeting history; by day 90, it can start surfacing patterns in the PM's own work [4].
- **How to apply:** Store reusable context locally, inject current priorities at session start, log repeated AI mistakes, and use AI to produce work products and workflows rather than only one-off answers [4, 3].

4) AI raises the premium on direct user understanding

A recurring community theme is that AI gives PMs more data, but not necessarily more understanding. One poster argues that many AI tools act like dashboards: they explain what users did, not why they did it or what they actually need [5]. Another makes the same point more directly: AI can summarize data, but it cannot replace the hard work of talking to users and validating what is real [6]. In the strongest example, AI added value only after the research work was done, by helping identify a novel persona hidden in interview transcripts [6].

- **Why it matters:** Over-relying on AI as a substitute for research can create false confidence and widen the gap between PMs who do the work and PMs who look for a shortcut [6].
- **How to apply:** Keep user conversations human, then use AI on the back end for transcript summarization, clustering, and pattern-finding [6].

Tactical Playbook

1) Build a local-context PM operating system

A practical setup from the Claude Code examples:

1. Create local markdown files for every active **project, person, and company** you need to track [4, 3].
2. Automatically append meeting transcripts into the right file so relationship history and commitments accumulate over time [4].
3. Add session-start hooks that preload your **weekly priorities, quarterly goals, and working preferences** into every new session [4].
4. Maintain a **mistakes file** and inject it at the start of future sessions so repeated AI errors become reusable guardrails [4].
5. Point the system at newsletters, bookmarks, LinkedIn messages, and videos, and ask it to extract only the novel or contrarian signals that matter [4].
6. Use the system to generate the artifacts PMs actually own: plans, reports, backlog outputs, and other deliverables [4, 3].

Why this works: It turns context from something trapped in meetings and chat threads into a reusable operating layer. In Dave Killeen's example, that meant daily planning, backlog management, career tracking, and compressing 120 newsletters down to the 3% that mattered [4].

2) Use AI in discovery without outsourcing discovery

> "If you don't talk to your users yourself, you'll have no idea what's real." [6]

A grounded AI-assisted discovery loop from the community discussion:

1. Conduct the user conversations yourself [6].
2. Use AI to summarize the transcript set and cluster themes [6].
3. Ask AI to look for gaps, anomalies, or a missed persona in the material [6].
4. Go back to the raw interviews to confirm what is real, why it happened, and what users actually need [5, 6].
5. Treat the model as an analysis aid, not a replacement for human interaction or judgment [6].

Why this works: It preserves respect for users and keeps the PM close to reality while still capturing some of AI's speed in synthesis [6].

3) Turn a difficult stakeholder into a predictable operating rhythm

Community advice on dealing with a hard-to-reach internal client converged on making access systematic instead of ad hoc:

1. Set a recurring meeting cadence that matches how often you actually need decisions or clarification [7].
2. Bring an agenda focused on feedback and requirements **before** the urgent moment arrives [7].
3. Frame the meeting as a way to reduce last-minute requests and unplanned interruptions for the stakeholder, not just for you [7].
4. Be explicit that you value their insight and clearly state what you need from the relationship [8].
5. Keep a paper trail of outreach attempts so blockers are documented [9].
6. If the stakeholder is critical, create face time through office visits or regular lunches when feasible [10, 11].
7. Escalate if non-response starts obstructing the work [9].

Why this works: It replaces reactive chasing with a clearer access model and gives you escalation cover if the relationship still fails.

Case Studies & Lessons

1) A non-coding PM turned a parked MCP idea into a production prototype

In sprint planning, a team dropped an MCP server idea for on-call troubleshooting because the estimate was **4-6 weeks** of developer time [12]. Later, a non-coding PM used GitHub Copilot with Opus 4.6 to generate a PRD, refine the design, scaffold the project, and work through five phases including testing, security audits, and bug fixes over **2-3 evenings** [12]. The team was impressed enough to start using the result in production [12].

- **Lesson:** Clear architecture patterns and first-principles decomposition can let PMs prototype ideas far earlier than a full delivery plan would suggest [12].

- **Apply it:** Use AI to produce a phased plan and rough implementation when an idea is strategically important but hard to fund upfront [12].

2) Dave Killeen turned Claude Code into a CPO operating layer

Dave Killeen uses Claude Code for daily planning, backlog management, and career tracking [4]. The system is built from session hooks, auto-updating markdown files, a mistakes log, and AI-driven information filtering [4]. One concrete outcome: he stopped reading 120 newsletters and instead reviewed the **3%** that the system flagged as relevant [4].

- **Lesson:** The leverage came from durable context and continuous correction, not from a single prompt.
- **Apply it:** Treat AI setup as infrastructure. The more your files, hooks, and error logs improve, the better the next session starts.

3) A north-star metric can still point south

Run the Business highlights a Microsoft Windows Update example where a team targeted **90%+** of machines on the latest OS version. When the number stayed stuck around **80%**, the team stopped asking users and forced updates without consent [2]. The metric improved, but the user experience deteriorated into the familiar mid-sentence reboot [2].

- **Lesson:** A green metric can still represent bad product judgment if it is disconnected from customer value and business health [2].
- **Apply it:** Pressure-test every north star by asking what behavior it incentivizes, not just whether it is measurable.

Career Corner

1) Prototyping is becoming part of PM craft

Deb Liu's question is blunt: if you are a PM, are you waiting for engineering to build your ideas, or are you prototyping them yourself? [1] Her broader argument is that PMs can now prototype and run research directly as role boundaries blur [1]. The MCP example shows what that can look like in practice, even for a non-coding PM [12].

- **Why it matters:** The PM who can turn an idea into a testable first version creates faster learning and usually better conversations with engineering.
- **How to apply:** Use AI to draft phased plans, PRDs, and rough prototypes before asking the team for a full build commitment [12].

2) Learn an AI toolchain that produces real work, not just answers

Rekhi's case for Claude Code is career-relevant because it is about artifact generation, local context, automation, and portability, not just chatting with a

model [3]. He says he has moved the majority of his product work into that environment [3].

- **Why it matters:** PM leverage increasingly comes from compressing execution loops around documents, analysis, and workflow automation.
- **How to apply:** Build fluency with a toolchain that can operate on your files, preserve context, and generate deliverables PMs are accountable for [3].

3) Keep direct user understanding as your moat

The community view here is consistent: AI has not replaced the hard work. It can summarize data and surface patterns, but it should not replace direct conversations with users [6].

- **Why it matters:** In an environment where more PMs have access to the same models, first-hand understanding becomes more differentiating, not less [5].
- **How to apply:** Protect time for direct interviews and transcript review, then use AI after the fact to expand your analysis [6].

4) Before chasing a title, check whether the role gives you leverage

Run the Business argues that many "unimpactful" PMs never had control over any of the three real assets in a product company: **product vision, R&D resources, or the P&L** [2].

- **Why it matters:** Sometimes low impact is structural, not personal [2].
- **How to apply:** When evaluating roles, look past title and ask what decisions, resources, or business levers you will actually control.

Tools & Resources

- Claude Code for PMs — Rekhi's companion video to his argument that Claude Code is a better PM productivity layer than generic chatbots because it supports artifact generation, local markdown context, and workflow automation [13, 3].
- pmskilltoolkit.com — a community-shared toolkit with **25** Claude/ChatGPT skills across discovery, roadmapping, competitive battle cards, win/loss debriefs, pricing strategy, and stakeholder politics. The creator says it draws on **1,500+** research sources and has reached **10k downloads** [14].
- From Symphony to Jazz — useful if you are rethinking team shape, role boundaries, and leadership guardrails in AI-heavy product development [1].
- The Most Expensive Dysfunction in B2B Product Companies — a strong reference for reconnecting teams to business reality through KPI laddering and better north stars [2].

- **Watchlist: real-time meeting copilots for internal knowledge retrieval** — multiple community posts describe the same gap: typing questions into ChatGPT or Claude Code during technical meetings is too slow, conversations move on, and transcript review afterward often creates more follow-up questions [15, 16].

Sources

1. From Symphony to Jazz
2. The Most Expensive Dysfunction in B2B Product Companies
3. X post by @sachinrekhi
4. substack
5. r/ProductManagement post by u/fan_ling
6. r/ProductManagement comment by u/aaronin
7. r/ProductManagement comment by u/hbtn
8. r/ProductManagement comment by u/*CaptRondo*
9. r/ProductManagement comment by u/Available_Orchid6540
10. r/ProductManagement comment by u/lic_queens
11. r/ProductManagement comment by u/lic_queens
12. r/ProductManagement post by u/Sufficient-Rough-647
13. X post by @sachinrekhi
14. r/prodmgmt post by u/YamObjective2419
15. r/ProductManagement post by u/geeky_traveller
16. r/ProductManagement post by u/geeky_traveller