Taste at Speed, Real Operating Systems, and an AI-Era Hiring Reset

PM Daily Digest

2026-03-14

Taste at Speed, Real Operating Systems, and an AI-Era Hiring Reset

By PM Daily Digest • March 14, 2026

Prototype-first PM work is getting more concrete, with Anthropic offering a vivid example of high-volume iteration, AI-assisted coding, and human-gated release discipline. This issue also covers why context and rituals matter more as AI speeds output, plus how PM hiring loops are changing as take-home signal degrades.

Big Ideas

1) Taste at speed is becoming a real PM advantage

Aakash Gupta's framing is that the emerging skill is **taste at speed**: the ability to evaluate working software quickly, kill most of it, and ship the survivors [1]. In that model, AI does not just speed building; it changes the bottleneck from **can we build it** to **should we ship it** [1]. The workflow contrast is sharp: the older path runs from idea to PRD to design to engineering to QA to ship in **8-12 weeks**, while the AI-era loop described here is **idea → 5 prototypes → evaluate → kill 4 → spec the survivor → ship** in **1-2 weeks** [1].

> "The spec didn't disappear. It moved from step 2 to step 6." [1]

- **Why it matters:** the leverage comes from filtering, not from shipping everything faster; the cited **80% kill rate** is the point [1].
- **How to apply:** for ambiguous problems, require multiple working prototypes, review them against **empathy, simulation, strategy, taste, and creative execution**, then spec only the winner [1].

2) Context and rituals are becoming the real operating leverage

John Cutler argues that the first place to inspect a product organization is its **rituals**: the daily and weekly interactions people have through meetings, dashboards, Slack, and other tools [2]. The weak spot is often the layer between front lines and leadership, where information has to move fluidly across the organization [2]. He also warns that AI makes it easy to generate documentation, but more documentation does not create intentionality [2]. Leah Tharin makes the complementary product point: **context is the real value**, not the model, and a jobs lens like "I want to listen to music on the go" opens a much broader solution space than a demographic profile [3].

> "Frameworks are models and all models are useful but wrong." [2]

- **Why it matters:** without shared context and deliberate rituals, faster output just creates faster drift. Cutler also points to a collective memory problem where teams keep re-documenting old issues because context is co-created over time [2].
- **How to apply:** build living context around recurring customer challenges, not just one-off deliverables, and deliberately design how information moves **up, down, and across** the org [3, 2].

3) AI speed makes alignment and work-shape clarity non-negotiable

Scott Belsky argues AI creates a stronger case for **talent density** and **far more alignment than usual** because teams can now move very quickly in the wrong direction [4]. Tony Fadell's reminder is simpler: knowing the **destination** helps people self-prioritize and decide what and how to build [5]. Cutler adds that most organizations have **parallel motions** at once—some work is large, coordinated, and governance-heavy, while other work is highly iterative—and pretending everything should run through one process is damaging [2].

- **Why it matters:** more prototyping capacity increases the cost of fuzzy goals and one-size-fits-all process.
- **How to apply:** make the destination explicit, separate high-coordination work from iterative work, and align your operating rhythm to each motion rather than forcing one template across both [5, 2, 4].

Tactical Playbook

1) Run a prototype-first decision loop

1. **Start with multiple working options.** The model here is five fast prototypes, not one polished plan [1].
2. **Evaluate against five lenses.** Check empathy, simulation, strategy, taste, and creative execution while looking at working software [1].
3. **Kill aggressively.** An **80% kill rate** is framed here as a feature, not a failure [1].

4. **Write the spec after you have a winner.** In this flow, the spec follows the prototype, not the reverse [1].
5. **Keep a human gate before production.** Anthropic still requires an engineer to approve changes before anything goes live [1].

- **Why it matters:** it compresses false certainty early and moves discussion onto concrete artifacts.
- **How to apply:** pilot this on one fuzzy feature before greenlighting a full PRD or roadmap commitment.

2) Use challenge memory in discovery

1. **State the customer challenge in job terms.** Prefer "listen to music on the go" over a demographic profile [3].
2. **Capture the surrounding context.** Tharin's argument is to build memory around the challenge, not just a single job statement [3].
3. **Feed better context into the system.** More correct context improves the odds of better decisions and of spotting bad data early [3].
4. **Revisit old knowledge before reopening old problems.** Cutler points to teams repeatedly documenting the same issue because collective memory is weak [2].

- **Why it matters:** it broadens solution space and reduces rediscovery waste.
- **How to apply:** keep one artifact per problem area with the job to be done, prior evidence, edge cases, and what the team already learned.

3) Repair the operating system through rituals

1. **Map the current rituals.** Start with the daily and weekly interactions people actually have, not the official process deck [2].
2. **Design information cadence intentionally.** Cutler's advice is to get information moving up, down, and across the org deliberately; documentation alone is not enough [2].
3. **Name parallel motions.** Separate large, coordinated efforts from fast iterative streams so each gets the right governance [2].
4. **Label relationships honestly.** Do not force work into fake linear hierarchies when frontline teams can sometimes move business metrics directly [2].
5. **Treat new habits as repeated experiments.** One kickoff meeting or spreadsheet rarely survives without sustained reps [2].

- **Why it matters:** many execution problems are really information-flow and habit-formation problems.
- **How to apply:** redesign one recurring meeting and one update channel before adding another template or framework.

Case Studies & Lessons

1) Anthropic's Claude Code workflow pairs extreme speed with explicit review gates

Boris Cherny is described as shipping **20-30 PRs a day** using **five parallel Claude instances**, with a third of his code potentially started from the iOS app and **100%** of his own code written with Claude Code [1]. This sits inside an unusually technical culture where everyone shares the title **Member of Technical Staff** and PMs, designers, data scientists, and even finance code [1]. Company-wide, Claude Code writes about **80% of code**, and productivity per engineer is cited as up **200%** since launch even as Anthropic **tripled** headcount [1]. The process is not review-free: every PR is first reviewed by Claude Code, which catches about **80% of bugs**, and a human engineer still does the second pass and approves anything before production [1].

- **Lesson:** the interesting move is not just AI-assisted coding; it is AI-assisted coding plus automated review plus human approval.
- **Boundary condition:** Gupta notes this setup fits a **small, senior team** with deep shared context, where the product is the AI tool itself, though the prototype-first discipline can translate beyond Anthropic [1].

2) Anthropic's product work uses volume to improve judgment, not just output

The reported iteration counts are unusually high: agent teams went through "probably **hundreds of versions**" before shipping; condensed file view saw about **30 prototypes** followed by a month of internal dogfooding; the terminal spinner had roughly **50-100 iterations**, with about **80% not shipping** [1]. One example is plugins: Daisy reportedly used a swarm of **a couple hundred agents** over a weekend, producing about **100 tasks** and an implementation that became "pretty much the version of plugins that we shipped" [1].

> "And it's a filtering function, not an acceleration function. The 80% kill rate is the whole point." [1]

- **Lesson:** faster tools matter most when the team is willing to discard most versions.

3) PM hiring loops are being redesigned because old homework signals got cheaper to fake

Andrew Chen says homework has become a common interview step for PMs and other knowledge roles because it can surface real work output [6]. His update is that, in recent weeks, these responses have been flooded with "AI slop"—long, meandering documents instead of a short, high-signal point of view [6]. His proposed fixes are a **recorded presentation**, where candidates sign off on what they wrote and can be questioned later, and a true **work trial** reserved

for the end of the funnel [6]. He also cautions that overly structured formats can alienate top-end talent [6].

- **Lesson:** when a signal becomes easy to mass-produce, move evaluation closer to live reasoning or real work.

Career Corner

1) Build taste reps now, before the gap compounds

The clearest career signal in this set is volume of evaluation. A PM who reviews **15 prototypes a week** builds judgment faster than one reviewing **one spec a month**; over six months, Gupta argues that becomes a widening taste gap and then a career gap [1]. He also says PMs who start building these reps now will have a massive head start [1].

- **Why it matters:** the compounding advantage comes from pattern-matching on working software.
- **How to apply:** keep a log of prototypes you killed, what you learned, and which evaluation lens changed your mind.

2) Nontraditional backgrounds still map well to core PM work

In community discussion, PMs pointed to several transferable skills from psychology, behavior analysis, and research backgrounds: observing people use a product, noticing nonverbal cues, asking better questions, and using surveys and statistics to understand how broad a problem is [7]. One experienced PM described these as among the most important things PMs do [7], while a former cognitive neuroscience researcher said they successfully switched into product and enjoy the work [8]. Another poster said the technical side of a specific product or industry looked like the stimulating challenge, not a blocker [9].

- **Why it matters:** PM hiring still rewards human observation and research judgment, not just AI fluency.
- **How to apply:** translate prior work into PM language: user observation, hypothesis formation, research design, and statistical interpretation.

3) Candidates and hiring managers both need an AI-era interview upgrade

For candidates, the implication of Chen's post is straightforward: concise thinking and live defense now matter more than polished take-homes alone [6]. For hiring managers, recorded walkthroughs can scale better than full work trials, while the most realistic evaluation should still happen late in the funnel [6]. Chen's warning against overly structured formats is also a reminder not to optimize the process so tightly that you filter out strong candidates [6].

- **Why it matters:** interview loops are becoming tests of reasoning ownership, not document generation.

- **How to apply:** if you're a candidate, practice explaining trade-offs live; if you're hiring, keep one live defense step in the loop.

Tools & Resources

- **There's a New PM Skill. It's Called Taste at Speed** — the clearest source here on prototype-first PM work, spec-after-prototype sequencing, and Anthropic's reported metrics [1]
- **Claude Code / Cowork** — examples of AI-assisted building workflows to watch; Cowork is described as a full product for non-engineers built in about **10 days** and part of Anthropic's push to bring this style to non-engineers [1]
- **The missing layer in government tech? A real operating system.** — John Cutler on rituals, information flow, and why a real operating system is more than a framework [2]

The missing layer in government tech? A real operating system. (27:17)

- **10 print Hello World** — Leah Tharin on context as value, jobs framing, and building memory around customer challenges [3]
- **Second Axis** — a community example of tools targeting the "messy middle" between idea and execution by generating docs, tickets, and edge cases from a feature idea; treat this as a category to watch rather than a vetted recommendation [10]

Sources

1. There's a New PM Skill. It's Called Taste at Speed
2. The missing layer in government tech? A real operating system.
3. 10 print "Hello World"
4. X post by @scottbelsky
5. X post by @tfadell
6. X post by @andrewchen
7. r/ProductManagement comment by u/GeorgeHarter
8. r/ProductManagement comment by u/techatypically
9. r/ProductManagement post by u/Dizzy-Ad4286
10. r/prodmgmt post by u/Ancient_Swimming1911