Harness Engineering, Onboarding Bottlenecks, and the New PM Operating Model

PM Daily Digest

2026-05-26

Harness Engineering, Onboarding Bottlenecks, and the New PM Operating Model

By PM Daily Digest • May 26, 2026

OpenAI's harness model shows how PM work is becoming more executable: PRDs, tests, and rules that agents can ship against. This brief also covers why onboarding remains a hidden growth constraint and how to evaluate feedback tools as part of a connected context graph.

Big Ideas

- **Harness engineering is becoming PM work.** At OpenAI, PMs shipped around **100K lines of production code** through PRDs, tests, docs, and harness rules instead of typing in an IDE [1]. The harness combines an `agents.md` operating loop, a docs tree, tests and lints that encode taste, specialized review agents, and observability/computer-use checks [1]. In one experiment, an internal app reached roughly **1M lines of code** from an empty repo with no human-typed production code; failures were addressed by improving the harness [1]. **Why it matters:** PM leverage is shifting from handoffs to executable artifacts. **Apply it:** write PRD + acceptance tests/evals + decision docs before implementation.

  > "The differentiator is how much of your team's judgment is embedded in your harness." [1]

- **Strategy still starts with the first user.** Paul Graham argues startup ideas should be framed as **idea + early adopters**; if you cannot say who will use the product when nobody else is, move on [2, 3]. Building something you want helps because you and your peers become that first cohort [4]. **Apply it:** add an "early adopters" line to every concept doc.

- **Agentic products need human control surfaces.** The emerging pat-

tern is software that humans and agents use together, with approval flows, summary inboxes, logs, and rollback—not just agent-only CLIs [5]. **Apply it:** specify the human checkpoints for every AI workflow.

Tactical Playbook

- **A safer PM workflow for AI coding agents:**
  1. Stress-test architecture against the PRD before coding [6].
  2. Pull fresh docs and research first [6].
  3. Split work into small components; quality reportedly drops past about **200K tokens** [6].
  4. Generate tests first so the model has a target [6].
  5. Use plan mode, manual approvals, and frequent Git pushes [6].
  6. Restart polluted sessions; store key decisions in markdown and keep context narrow [6].

  Why it matters: these steps directly target common hallucination and context failures.

Case Studies & Lessons

- **Onboarding keeps surfacing as the real bottleneck.** For a workflow tool, the biggest growth lever was reducing time from sign-up to real value, especially first project setup [7]. A separate B2B SaaS complaint shows the opposite pattern: users create a password, verify email, set up TOTP, and import a team, then later delete the password once SSO arrives [8]. **Lesson:** design onboarding for the end state, not the temporary workaround.
- **In marketplaces, vendor onboarding/catalog ingestion can be harder than vendor acquisition.** One founder said vendors liked the internal software once onboarded for orders and inventory, but uploading products, pricing, photos, descriptions, variants, delivery constraints, and categories could still take weeks [9]. Options under consideration included concierge onboarding, internal tooling, PIM tools, POS integrations, and AI/OCR ingestion [9]. **Lesson:** if setup takes weeks, acquisition is not the real growth constraint.

Career Corner

- **AI-native PMs need executable specs.** In one OpenAI example, a PM wrote a markdown PRD for a "skills system," the team reviewed it once, and the feature existed by week end with tests passing [1]. **Practice:** express one feature as PRD + tests + evals an agent can run [1].
- **For internships, signal structured thinking.** Expect product improvement, prioritization, stakeholder conflict, metrics/KPIs, basic agile,

and tech-fluency questions; interviewers are judging communication, curiosity, ownership, and problem solving [10]. **Practice:** answer aloud with a clear structure and success metric.

Tools & Resources

- **Feedback tools still split into three jobs:** analysis (**Kapiche, Unwrap, Chattermill**), collection (**Canny, Productboard**), and behavior (**Pendo, Hotjar**) [11]. The main caution: separate collection and analysis tools force manual stitching across Slack, tickets, surveys, and widgets [12]. A better selection principle is to favor a connected feedback graph so agents can surface patterns and PMs can spend more time on judgment [13].

Sources

1. How PMs Ship 100K Lines of Code at OpenAI with Ryan Lopopolo, Member of Technical Staff
2. X post by @paulg
3. X post by @paulg
4. X post by @paulg
5. X post by @lennysan
6. r/ProductMgmt post by u/InfamousInvestigator
7. r/startups comment by u/tw1x_45
8. r/ProductManagement post by u/Legal_case16
9. r/startups post by u/tonyyayo93
10. r/ProductManagementJobs comment by u/Alarmed_Campaign_338
11. r/ProductManagement post by u/Excellent-Garage8695
12. r/ProductManagement comment by u/canhigher23
13. substack