A FIELD GUIDE

How to Install a Product Operating System

Most orgs have the data. Almost none have the system that knows what to do with it. This is a 5-page guide on building that system.

Jenna Minnix
jennakertz.com

2026

THE THESIS

What a product operating system actually is

Most product teams ship features. The good ones ship the right features. The best ones build the *machinery* that figures out which features are right, runs the tests to validate them, and — increasingly — operates parts of itself without human intervention.

That machinery is what I call a product operating system. It's not a tool. It's not a process doc. It's a connected set of capabilities that turn messy organizational signal into reliable product decisions. Most companies don't have one. They have meetings, dashboards, and roadmaps that gesture at the same thing without doing the work.

> *"Most orgs have the data. Almost none have the system that knows what to do with it."*

A product operating system has three layers:

DIAGNOSIS The muscle to ask the real question. Most product orgs report metrics. A good one finds the question the metrics are hiding.

SYNTHESIS The pipeline that turns customer signal into product direction. Without a system, signal arrives in fragments, gets discussed once, and disappears.

EXECUTION The systems and norms that turn decisions into shipped work, validated learnings, and compounding leverage — including the AI agents now doing parts of the work.

This guide is the short version of what I've learned installing this system. The next three pages cover each pillar. The final page covers how I'd install it in 90 days.

PILLAR ONE

Diagnosis

Most CVR investigations are wrong on the first pass because they confuse correlation with cause and trust the metric without interrogating the population behind it.

The first move in any diagnosis isn't to look at the metric. It's to look at *who's in the metric*. A conversion rate is a number. The cohort that produced the number is the actual subject. If the cohort changed, the metric changed — even if nothing in the product did.

> *"You can't fix what you haven't named. A good diagnostic names the thing everyone felt but no one said."*

Three questions I ask before anything else:

- Has the population behind this metric changed? Build segment classification and look at mix shift over the same window as the metric shift. If the mix changed and you didn't notice, that's your answer — and the product fix you were about to ship is the wrong fix.
- Is the data actually trustworthy? Tracking gaps are more common than most teams admit. Trace event definitions back to the code. A bad metric beats no metric only when you know which one you're looking at.
- Are you measuring the right transition in the funnel? Most funnel investigations look at one step. Real funnel pathologies concentrate at a transition you weren't watching.

The point isn't the analysis method. The point is that you can't fix what you haven't correctly diagnosed — and most product orgs skip this layer entirely.

PILLAR TWO

Synthesis

Customer signal is everywhere. Most companies don't have a synthesis problem — they have a *system* problem. Without a pipeline, signal arrives in fragments, gets discussed once, and disappears.

A synthesis pipeline has four components:

- **Continuous collection.** NPS surveys running in-product. Support ticket categorization. Churn exit surveys. Sales call notes tagged consistently. The collection has to be passive — if anyone has to remember to do it, it won't happen.
- **Rolling theme tracking.** Run sentiment analysis on rolling 14, 30, and 60-day windows. The 14-day catches emerging issues. The 30-day shows trend stability. The 60-day shows what's structural vs. transient.
- **Cluster, then interpret.** Use vectorization and clustering to surface themes, then human judgment to interpret what they mean. The machine finds the patterns. The PM decides what they're telling you.
- **Synthesis cadence.** A weekly digest combining qualitative signal with behavioral data — delivered to decision-makers without anyone manually preparing it.

> *"If your roadmap has twelve priorities, you have no priorities. Synthesis is the art of the hard no."*

The methodology compounds. Pricing research conducted using rigorous methodology in year one can still be authoritative five years later in leadership-level decisions. Synthesis isn't a quarterly project. It's an asset that grows in value the longer it runs.

PILLAR THREE

Execution machinery

The best product teams aren't faster because their people are better. They're faster because their systems do more of the work.

Three components of execution machinery:

- **Experimentation infrastructure.** Not just tooling — norms. Structured readouts. Pre-registered hypotheses. Null results called cleanly. A culture where killing a feature is as celebrated as shipping one. The norms are what make experimentation actually inform decisions.
- **Ownership models.** Every roadmap item has a directly responsible individual. Estimates are joint between product and engineering. Resource allocation happens at the exec level with the same people each time.
- **AI agents for the work humans shouldn't be doing.** This is where the biggest leverage gains are happening in 2026.

> *"If your team can only ship when everyone is in the room, you don't have a system. You have a dependency graph shaped like a person."*

I built an autonomous agent in roughly ten minutes by dictating what I wanted to an AI. It runs daily, monitors team communication for tagged feature requests, conducts competitive and demand research, drafts tickets, and generates interactive mockups. It handles the no-brainer features the team never has time for.

The ten minutes is the punchline. The decade of pattern recognition that made it possible is the story. Any PM at any company can build a version of this. What you need is a clear enough mental model of your own workflow that you can describe it to an AI — and the willingness to let an agent ship work without you in the loop on every step.

HOW I'D DO IT

Installing this in 90 days

A real engagement — full-time or fractional — would look roughly like this.

WEEKS 1–3

Diagnose

Build segment classification. Audit data integrity. Identify the two or three real questions the leadership team is asking — usually not the ones in the dashboard.

Output: A diagnostic report that names the actual problems and ranks them by impact and tractability.

WEEKS 4–6

Install synthesis

Set up rolling theme tracking on existing customer signal sources. Build the weekly synthesis digest. Identify the top themes the company has been under-responding to.

Output: A synthesis system that runs continuously, plus a prioritized list of customer-signal-driven product opportunities.

WEEKS 7–10

Build experimentation muscle

Establish experimentation norms. Run two real tests — one on a current roadmap item, one on a synthesis-surfaced opportunity. Build the readout culture.

Output: An experimentation cadence the team will keep running after I leave.

WEEKS 11–13

Ship the first agent

Identify one workflow that's a candidate for AI augmentation. Build a v1 agent for it. Establish the norms for adding more.

Output: One shipped agent and a roadmap for the next three.

"I'm always interested in conversations with companies serious about installing or upgrading their product operating system."

jennakertz.com · jennakertz@gmail.com