The Product Pathway

By **Matt Paulin**

Posted on April 2, 2026

How I manage multiple products in tandem

I am my own innovation hub.

Over the last 6 months I have been building non-stop—web apps, games, Android apps, iOS apps. The question I keep coming back to is:

What is the best use of my time today?

When you have more ideas than time, and you have obligations pulling you in all directions, this becomes the whole game.

Product vs Engineering

The way I look at it, there are two forces that have to move together:

- **Product thinking** → Why would anyone use this?
- **Engineering thinking** → Why doesn't this work?

They are tightly coupled, especially early on.

Product is about understanding people.
Engineering is about solving problems.

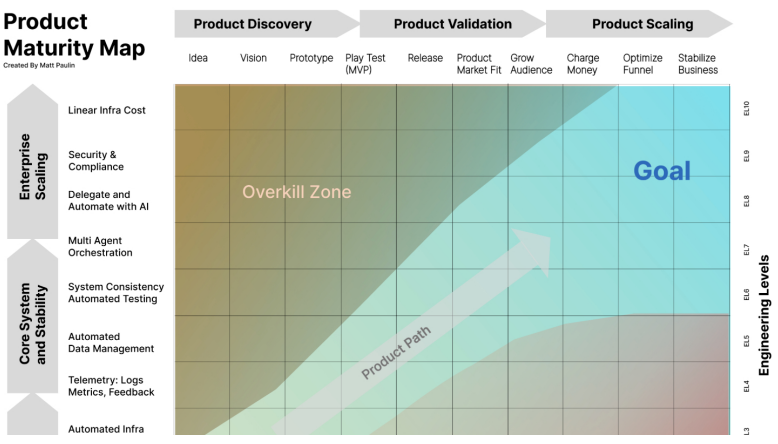
This was the inspiration of previous articles where I have been thinking about the steps toward building a product and the steps it takes to develop the engineering.

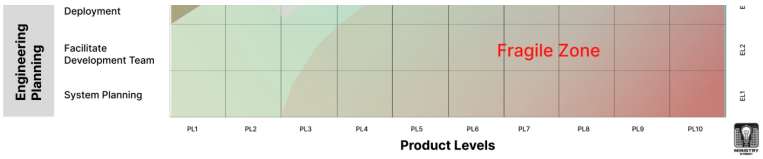
- A **product track** (Idea → Vision → Prototype → MVP → ...)
- An **engineering track** (Planning → Facilitate → Deploy → ...)

The Product Maturity Map

I found it very helpful to put them on a grid so that I can make decisions about what I need to work on, more product work or more engineering work.

- **X-axis (right)** → Product usefulness (does anyone care?)
- **Y-axis (up)** → Engineering maturity (is this well built?)





Download the Product Maturity Map (PDF)

This creates a simple mental model:

- Move **right** → increase usefulness
- Move **up** → improve reliability, scalability, and structure

The goal is to move **through the map in balance**.

How to use this

Start in the bottom-left.

At that point, everything is undefined:

- Who is this for?
- What problem does it solve?
- What does it look like?
- How will it make money?

I usually work through this phase with conversations, sketches, and rough prototypes.

Once you have something to work with, you start making moves.

At any point, you're choosing:

- Move **right** → understand the product better
- Move **up** → build the system better

The map helps you decide which one matters most **right now**.

The Zones

As you move through the map, you'll fall into one of four zones.

Overkill (too early)

> ***Overkill is when you solve problems before you understand what needs to be built.***

This happens when engineering gets ahead of the product.

You might:

- Build full infrastructure before users exist
- Design systems for scale that may never come
- Lock in decisions too early

This creates **code debt before value**.

It's also hard to unwind, especially if people feel attached to what they built.

A useful question here is:

> *What is the last responsible moment to make this decision?*

If you're making it earlier than that, you're probably in Overkill.

Fragile (too late)

This is the opposite problem.

The product is valuable—people want it—but the system can't support it.

This often looks like:

- A "vibe coded" app no one fully understands
- No monitoring, no ownership of infrastructure
- Billing, data, or reliability issues
- Things breaking as soon as real usage shows up

The product has outpaced engineering.

> *If you're in the fragile zone, you waited too long.*

Safe (balanced)

This is where you want to operate.

You're:

- Not overbuilding ahead of demand
- Not underbuilding behind demand
- Moving step-by-step across both axes

There's some tolerance here. You don't need perfect balance.

But you are **aware of the gap** and adjust as needed.

Alive (the goal)

I think of the top-right not as "winning," but as being **alive**.

A product is alive when:

- Users are growing naturally
- Revenue exceeds expenses
- The system is stable

It justifies its existence.

It doesn't need constant intervention to survive.

How I decide what to do next

The most useful thing this map gives me is a simple decision rule.

At any point, I ask:

> *What is the biggest move for this product right now—more engineering or more understanding?*

That question forces a choice.

- If no one is using it → move right (product)
- If it's breaking or unstable → move up (engineering)

You don't need perfect information.
You just need to move in the right direction.

Modern Product Discovery

One observation is with the advent of vibe coding platforms like Base44 and Loveable.ai it suddenly becomes very easy to move to the right on this map and create something people are using quickly.

For example you can use these tools for:

- exploring ideas
- building prototypes
- understanding what might work
- getting all the stakeholders on the same page

The danger is that they look done and the reality is they can also create fragile systems if you rely on them too long.

At some point, you are responsible for:

- your users
- their data
- the reliability of the system

That's when you have gone too far to the right without really understanding the engineering.

Final Thought

This map is not about perfection.

It's about staying aware of where you are and making the next responsible move.

If something feels off, it usually is.

- Too much building → go understand users
- Too much selling → go stabilize the system

Just keep asking:

> ***What is the biggest move for this product right now—more engineering or more understanding?***

← PREVIOUS POST