PML | PRODUCT MATURITY LEVEL

FIELD GUIDE • PML 2.0

Measure today. Evolve tomorrow.

A practical guide to running the Product Maturity Level assessment.

10 domains | **50** statements | **5** maturity levels



Sandro Maia de Araújo

Foreword by Paulo Lisboa

PML — PRODUCT MATURITY LEVEL

The PML Field Guide

A Practical Guide to Assessing and Improving Product Maturity

Sandro Maia de Araújo

Creator of Product Maturity Level

First Edition · 2026
Framework version PML 2.0.0

PUBLICATION DETAILS

Title	**The PML Field Guide**
Subtitle	A Practical Guide to Assessing and Improving Product Maturity
Author	**Sandro Maia de Araújo**
Edition	First Edition, 2026
Publisher	Independently published by the author
Language	English
Framework	Product Maturity Level (PML) 2.0.0

© 2026 Sandro Maia de Araújo. All rights reserved.

This publication is an educational and practical resource designed to help product teams assess and improve product management practices. PML is an independently developed self-assessment framework. Its results support reflection, learning and continuous improvement; they are not a professional certification, an external audit, an individual performance evaluation, or evidence that a team is objectively superior to another.

References to third-party publications, frameworks and organizations do not imply their endorsement of PML.

https://www.productmaturitylevel.com/

Foreword

One question I like to ask product teams is simple: What was the last decision you changed after learning something new about your customers?

The answer often tells me more about a team's maturity than an entire presentation about its processes. It reveals whether the team has access to customers, whether it can challenge its own assumptions, and whether it has the autonomy to change direction when better evidence emerges. And often, it reveals something about the organization itself. What kinds of decisions is it willing to delegate?

Having led Product teams in Brazil and Mexico, I've seen how teams within the same company can operate under very different conditions. In one area, a team had clarity about the problem and the freedom to test different approaches. In another, equally capable people faced dependencies that made even small decisions difficult. Prescribing the same improvement plan for both would mean ignoring what each team actually needed to address.

This brings me to a distinction I consider important. Mastering frameworks helps organize the work. Creating value requires making choices based on what we learn. A team can conduct great interviews, maintain a well-structured roadmap, and still spend months building a feature whose expected outcome nobody can clearly explain. Maturity becomes evident when a team can articulate which problem it has chosen to solve, why that problem is worth solving now, and what would make it reconsider its approach.

That's why, before discussing where a team needs to go, it's worth understanding where it stands today. What decisions

can it make? Where does work get stuck? What happens when something is released and the results fall short of expectations? A structured assessment gives teams and leaders a shared language to explore these questions. And when those conversations become a regular practice, it becomes easier to choose a concrete improvement, experiment with it, and return a few weeks later to see whether it made a difference.

It was a real pleasure to be invited to write this foreword. I have tremendous admiration for Sandro Maia. As a professional, I admire the seriousness with which he approaches Product Management and his willingness to ask questions that help us see what often goes unnoticed in our daily work. As a friend, I admire his generosity in sharing what he knows and the genuine care he shows for people. Sandro is an exceptional professional and an exceptional person. I'm proud to call him my friend.

Congratulations, Sandro, on putting this conversation into the hands of those who can actually do something with it. I hope that, by the time readers close this book, they'll have a meaningful question to bring to their next conversation with their team.

Paulo Lisboa

Head of Product | AI & Data | Driving Growth, Revenue & Scalable Products in LATAM | MBA Professor

A Note from the Author

Across my career in product management I have worked with different teams, different organizations and very different product challenges. One question kept surfacing in all of them, and it was never comfortable: **how do we know whether we are actually getting better at building and managing products?**

Most teams can describe what they have delivered. They can show roadmaps, releases, dashboards and a list of initiatives that closed. Whether the way they work has improved is a much harder thing to answer, and in my experience it usually goes unanswered — not because people do not care, but because nobody wrote down where they started.

That gap is what led me to create the Product Maturity Level. I wanted something practical: a way for a team to reflect on how it works, name its strengths without embarrassment, see its gaps without defensiveness, and leave the conversation with a path it actually believes in.

PML was never meant to judge professionals, rank teams or prescribe a single correct way of working. There is no such way, and a framework claiming otherwise would be selling something. Its purpose is narrower and, I think, more useful: better conversations, and better-informed decisions about how to improve.

This guide brings that philosophy together with the practical part — how to run the assessment, how to read a result honestly, and how to turn it into something that changes next quarter rather than a screenshot in a deck.

My hope is that it helps you and your team move past measuring maturity and into making the improvements that matter. **Because product maturity is not about doing more. It is about getting better at what truly creates value.**

Sandro Maia de Araújo
Creator of Product Maturity Level

Acknowledgements

To my parents, for all the lessons, values, and examples that have guided me throughout my life. For teaching me, from an early age, the importance of honesty, dedication, hard work, and, above all, never stopping learning. So much of who I am today is a reflection of everything you have given me.

To my wife, Priscila, my partner in life and my greatest supporter. Thank you for your patience, love, and understanding of my somewhat *workaholic* nature, always immersed in new projects and deeply passionate about what I do. You understand my restless spirit, support my ideas — even the boldest ones — and encourage me to keep pursuing new challenges. Your support gives me the confidence and peace of mind to turn ideas into reality. You are an essential part of every achievement.

To my sons, João and Arthur, who are among my greatest sources of inspiration. You remind me every day that learning, growing, and discovering new paths are journeys that never truly end. I hope my own journey inspires you to pursue what you believe in with courage, curiosity, and passion.

This book was born from my passion for Product Management, but it was only possible because of the people who walk beside me, believe in my ideas, and encourage me to keep growing.

To all of you, my love and deepest gratitude.

If this book is about growth, it is because I have been fortunate enough to have people by my side who have taught me, inspired me, and helped me grow every single day.

Contents

Foreword

A Note from the Author

Acknowledgements

PART I — WHY THIS EXISTS

1 The problem this solves

2 Who this is for

3 What product maturity actually is

4 Why maturity work usually fails

5 What PML is, and what it is not

PART II — MEASURING

6 How maturity is measured

7 Running the assessment

8 Reading your result

9 A result, read end to end

PART III — IMPROVING

10 Turning the result into change

11 The rhythm

12 Running it across several teams

PART IV — REFERENCE

13 The 10 domains

14 What this instrument cannot tell you

15 Questions that come up

References & Further Reading

About the Author

About Product Maturity Level

PART I — WHY THIS EXISTS

CHAPTER 1

The problem this solves

Ask a product team whether they got better this year and watch what happens. You will get anecdotes — a launch that went well, a quarter that went badly — and underneath them an honest shrug. The team worked hard. Whether the way they work improved is a question nobody can answer, because nobody wrote down where it started.

That gap produces four problems, and you have probably met all of them.

Improvement debates are opinion against opinion

One person is certain the problem is discovery. Another is certain it is delivery. Both are describing something real they have seen, and neither has evidence the other accepts. The argument is settled by seniority or by volume, which is a terrible way to choose what a team spends six months on.

Advice arrives without a sense of where you are

The literature is written for everyone, so it is addressed to no one. A team with no written strategy and a team running quarterly strategy reviews get told the same thing: do continuous discovery, measure outcomes, talk to users. For one of them it is the next step. For the other it is three steps ahead, and attempting it produces a ceremony that collapses in a month.

Symptoms get treated instead of causes

Deadlines slip, so the team is asked to improve estimation. Six months later deadlines still slip, because the cause was never estimation — it was that priorities changed mid-sprint, which is a prioritization problem wearing a delivery costume. Without a map of practice areas, the loudest symptom chooses the cure.

Teams cannot learn from each other, even side by side

In most organizations one team has already solved what another is struggling with, and neither knows. There is no shared vocabulary to notice it with: discovery means five different things in five squads, so the conversation never starts.

None of these is a motivation problem. They are all the same missing thing — a shared, specific description of how the team works today, that two people can look at and disagree about productively.

The cost of not measuring is not zero

It is paid as repeated debate, as improvement effort aimed at the wrong domain, and as the quiet erosion of the case for investing in how the team works at all. A head of product who cannot show that practice improved will eventually be asked to stop spending time on it.

CHAPTER 2

Who this is for

PML is for the people who are responsible for *how* a team works, not only for what it ships. That responsibility is usually shared and rarely written down, which is part of why it goes unexamined for years.

Three readers

The product manager or team lead

You can feel that something in the way you work is costing you, and you cannot get agreement on what. A baseline turns that feeling into ten specific readings, which is the difference between “we should be more data-driven” and “nobody on this team can see whether the last release changed anything”. One of those sentences can be acted on.

The leader with several teams

You are asked where to put coaching, hiring and process effort, and the honest answer today is built from whoever spoke to you most recently. Ten domains across eight teams replaces that with a map — and, more valuably, shows you which team has already solved the thing another is stuck on.

The team itself

Maturity is a property of a group, so the group is the real audience. Product, design and engineering answering together produces something none of them had alone: a shared, specific description of their own way of working, including the parts they disagree about.

Five moments worth measuring

Roles say who might benefit. Moments say when it actually pays, and these are the five where it reliably does.

You are inheriting a team or an org

A structured read in your first weeks, from the team rather than filtered through whoever is most comfortable talking to a new boss. It is also the only chance you will ever get to record a baseline you did not influence.

The same theme keeps coming back in retros

A recurring complaint that never gets fixed usually means the team is naming a symptom. Ten domains make it harder to keep circling the same symptom and easier to find what sits upstream of it.

You grew faster than your practices

Two teams became six and nobody decided which habits should come along. Practice goes uneven quietly, and a reading per team shows exactly where it diverged.

Something is about to change the way you work

A reorganization, a new operating model, a coaching engagement. Measure before it starts. A before-and-after is the only honest way to find out whether it worked, and the "before" is impossible to collect afterwards.

You need to justify time spent on how you work

Investment in practice is the first thing cut when it cannot be shown to move anything. Two readings a quarter apart are the cheapest evidence you will ever assemble that it did.

It also works outside classic product teams

Platform, internal tooling and service teams use it regularly. Usually one or two domains genuinely do not apply — pricing is the common one — and that is exactly what *Not applicable* exists for. The other eight still describe real practice, and the per-domain reading stays useful even when the overall number deserves more caution.

When not to do this

During a genuine crisis. An outage, a deadline that will be missed, a round of layoffs. The answers will describe the crisis rather than the way you normally work. Wait until the week is ordinary.

If the result will reach a performance conversation. Even the suspicion is enough to corrupt the data. If you cannot promise it will not, do not run it.

If nobody will own the follow-up. An assessment with no owner for what comes next produces a good twenty minutes and no change. That is not harmless; it teaches the team that this kind of exercise goes nowhere.

There is one case that looks like a reason not to measure and is not: a team with little authority over how it works. The result will document constraints the team cannot lift, which feels pointless from inside. It is in fact the most useful thing that team can produce — a specific, non-emotional account of where it is being held back, which is far more persuasive upward than frustration is.

What you actually need to start: a team willing to answer honestly, twenty minutes in the same room or call, and one person who will own what comes after.

CHAPTER 3

What product maturity actually is

Maturity is the degree to which good practice is **systematic rather than heroic**. It asks one question of everything a team does well: would it still happen if the person who usually does it were on holiday?

That is the whole idea. A team where discovery happens because one PM insists on it is less mature than a team where discovery happens because it is simply how work starts — even if the first team currently does better discovery. The first result depends on a person. The second depends on the system, and systems survive reorganizations.

> **Maturity is about variance, not peak performance**
>
> A mature team is not the one with the best quarter. It is the one whose bad quarter is still decent, because the outcome does not rest on heroics. This is why maturity models measure practice rather than results: practice is what makes a result repeatable.

The idea is older than product management. It comes from the Capability Maturity Model developed at Carnegie Mellon's Software Engineering Institute in the late 1980s, whose central claim was that an organization's capability predicts the *variance* of its outcomes, not just their average. PML applies that claim to product practice: not whether you shipped something good, but whether the way you work makes shipping something good likely again.

Four things maturity is not

> **Not seniority**
>
> A team of experienced people can run entirely on improvisation — and often does, precisely because they are good enough to get away with it. Experience raises the ceiling; maturity raises the floor.

Not size or age

Large organizations accumulate process, which is not the same as practice. A twelve-year-old product org can be systematically mature or can have twelve years of ceremonies that nobody believes in.

Not tooling

Buying an experimentation platform does not make a team mature at experimentation. The tool is evidence of intent; the practice is whether decisions actually change because of what it reports.

Not speed

Shipping quickly is a capability, and a valuable one. It is also the capability most often mistaken for the whole picture — which is how teams end up shipping faster and faster in a direction nobody validated.

Higher is not always better

Each level costs more than the last, and the return does not keep pace. A five-person team at an early-stage company that deliberately runs lightweight practice in most domains, with real rigour in the one or two that decide its survival, is making a sound call — not an immature one.

The useful question is never *how do we get everything to five*. It is *which domains does our strategy actually depend on, and are we mature enough there*. A deliberate 3 is a better answer than an accidental 4.

CHAPTER 4

Why maturity work usually fails

Most teams that assess their maturity never act on it. The failures are predictable enough to list, which means they are avoidable enough to plan around.

It becomes a target

The moment a maturity score is reported upward as a goal, it stops measuring anything. People do not even have to be cynical about it — they simply answer a little more generously each round, and within two cycles the number only describes the pressure on the people answering.

Everything gets fixed at once

A result lists ten domains and every one of them has room to improve. Treated as a to-do list it yields ten half-started initiatives and no change. One domain, properly, beats five domains superficially — every time.

Benchmarking arrives before trust

Comparing teams is genuinely useful, and it is the second thing you do, not the first. Open with a league table and you have taught everyone that the right answer is a high one.

It is measured once a year

An annual cadence is too slow to learn from. By the time the second reading arrives, nobody remembers what was decided after the first, and the comparison becomes trivia rather than feedback.

Nobody owns the actions

The single most common ending: a good conversation, a screenshot in a deck, and nothing in anyone's week. An improvement without a name and a date attached is a wish.

The antidote is small and boring

Say out loud that no one is being graded. Pick one domain. Give two actions an owner and a date. Measure again in a quarter. That is the entire method, and the rest of this guide is detail in service of it.

CHAPTER 5

What PML is, and what it is not

PML is a mirror, not a grade. It asks your team to describe how it actually works today, across 10 areas of product practice, and gives that description a shape you can look at, argue with, and act on.

The value is not the number. The value is the conversation the number starts, and the fact that next quarter you can tell whether anything changed.

It is not any of these

Not a performance review. It describes practices, not people. A low score is a statement about how the team works, not about who works there.

Not an audit. Nobody checks your answers. An inflated result only costs you the usefulness of your own result.

Not a ranking. When several teams measure, the point is to find who already solved what you are stuck on — not to sort them.

Not a certification. There is no level to achieve and nothing to pass. The reading exists to inform a decision, and a reading that informs no decision was not worth taking.

This matters more than it sounds. A maturity assessment that people suspect will be used against them produces one predictable outcome: everybody scores four. The data looks encouraging and tells you nothing. Say out loud, before you start, that there is no expectation to score high everywhere.

PART II — MEASURING

CHAPTER 6

How maturity is measured

You answer 50 statements — 5 per domain — on a five-point agreement scale, from *strongly disagree* to *strongly agree*. Each statement describes an observable practice, not an opinion about quality.

A domain score is the average of its answers. The overall score is the average of the 10 domain scores — not of all 50 answers. That is deliberate: it keeps every domain weighing the same, so a strong delivery practice cannot quietly compensate for an absent discovery practice.

Ad-hoc 1.0 – 2.0

Practices are informal and reactive.

Processes and practices are informal, inconsistent, reactive, or may not exist. Outcomes depend on individual effort and the team is mostly responding to whatever arrives next.

Emerging 2.0 – 3.0

Good practices exist, but they depend on individuals.

Some practices exist but they are applied inconsistently and depend heavily on individuals. The team knows what good looks like; it is not yet the default way of working.

Defined 3.0 – 4.0

Clear practices exist and are shared across the team.

Clear processes and practices exist and are understood by the team, although improvement is still required. The way of working is repeatable and no longer depends on a single person.

Managed 4.0 – 4.5

Practices are consistent, measured and integrated.

Practices are well established, consistently applied, measured, and integrated into the way the team operates. Decisions are traceable to evidence and outcomes are actively managed.

Optimized 4.5 – 5.0

Continuous improvement and organizational influence.

The team demonstrates advanced product practices, continuous improvement, experimentation, learning, and organizational influence. It sets the standard others in the organization learn from.

The bands are anchored to what people actually clicked. Below 2.0 the team on average disagrees that these practices exist. Above 4.5 it agrees strongly, across the

board. That anchoring is what lets you defend a level in a room full of sceptics.

Optimized should be rare

Reaching the top band requires near-unanimous strong agreement across every statement in a domain. That is correct for a maturity model. A framework where most teams land at level four is a framework that has stopped measuring anything.

CHAPTER 7

Running the assessment

Do it together. Book twenty minutes, share your screen, and answer as a group with one person typing. This single choice does more for the quality of the result than anything else in this guide.

Answered alone, the result describes one person's view of the team. Answered together, the disagreements surface — and the disagreements are the most useful thing the exercise produces. When two people rate the same practice three points apart, you have found something worth an hour of conversation.

A twenty-minute run sheet

0–2 min	Say who will see this and what happens next. Say that nobody is being graded and that low answers are the useful ones.
2–4 min	Agree the window — the last three months — and agree that one person types while everyone answers out loud.
4–18 min	Read each statement aloud, take the room's first instinct, move on. Do not debate to consensus; note the disagreement and keep going.
18–20 min	Look at the two statements that split the room hardest. Those, not the score, are what you book the follow-up conversation about.

Three rules for answering

1. **Describe what happens, not what is documented.** A process that exists in a wiki and nowhere else is not a practice.
2. **Answer for the last three months**, not for the best quarter you ever had and not for the week everything broke.
3. **Do not negotiate towards the middle.** If the room disagrees, pick the answer the evidence supports and note the disagreement — it is a finding, not a problem.

When a statement does not fit

Two of the answer options are not scores, and the difference between them matters. Use them honestly: neither one lowers your result.

⊖ Not applicable

The practice genuinely does not belong in your context. An internal platform team with no pricing is not failing at pricing.

Not for a practice that is relevant but missing. That is a low score, and it is the most valuable kind of answer the instrument collects.

ⓘ I don't know

The practice may well exist; you cannot see it from where you sit. Choose this instead of guessing.

A cluster of these in one domain is itself a finding: it usually means the work is happening somewhere the team has no visibility into.

Why honesty is free here

Answers that are not scores are left out of the arithmetic rather than counted as low ones. If a domain ends up with too little evidence, PML reports it as *not enough information* instead of inventing a score. Nothing in the result punishes you for admitting uncertainty.

CHAPTER 8

Reading your result

Read the shape before the number. The radar chart is the result; the overall score is a summary of it. A team at 3.4 with an even shape and a team at 3.4 with two spikes and two holes have almost nothing in common, and need opposite advice.

Scored

Enough statements were rated for the number to carry weight. Read it normally.

Provisional

Scored, but on thin evidence. Treat it as a starting point for a conversation rather than a diagnosis, and be careful comparing it with another team.

Not enough information

No score is shown at all. The domain appears as a gap in the radar — an absence of evidence, which is not the same as the worst possible answer.

That distinction is worth holding on to. A hole in the shape means *we could not see this*. A low score means *we looked, and it is not happening*. The first is answered by finding out; the second by changing how you work. Confusing them sends a team off fixing the wrong thing.

Your result also names your strongest domains. Do not skip past them. They are where your credibility comes from when you ask for time to fix something else, and — if other teams are measuring too — they are what you have to offer.

Three shapes worth knowing

The delivery spike

Strong delivery and technical practice, weak discovery and outcomes. This team builds well and cannot tell you whether what it built mattered. Very common, and usually comfortable, because the visible work all looks successful.

The flat middle

Everything between 2.8 and 3.4, nothing standing out either way. Usually a team with real practices that none of them have been pushed far enough to become an advantage. The move is to choose one domain to be genuinely good at, not to lift the average.

The leadership gap

Solid practice everywhere except strategy and leadership. Often a strong team inside an organization that does not give it the context or the authority it needs. The actions here usually live outside the team, which is worth naming honestly rather than treating as the team's failing.

CHAPTER 9

A result, read end to end

Reading a result is a skill, and the fastest way to learn it is to watch. Atlas is a fictional team — an internal logistics tool, seven people, eighteen months old — that has just finished its first assessment.

Atlas — first baseline		Overall **3.1**
Product Delivery & Continuous Improvement	Managed	**4.4**
Technical Sustainability & Reliability	Managed	**4.0**
Prioritization & Investment Decisions	Defined	**3.6**
Product Vision & Strategy	Defined	**3.4**
Product Leadership & Operating Model	Defined	**3.2**
Customer Understanding & Adoption	Emerging	**2.6**
Product Discovery & Validation	Emerging	**2.4**
AI-Enabled Product Management	Emerging	**2.2**
Outcomes & Product Analytics	Ad-hoc	**1.8**
Business Value & Economics	not enough information	

What the shape says

This is the delivery spike. Atlas ships reliably and maintains its code well — 4.4 and 4.0 are genuinely strong — and has almost no instrumentation for knowing whether any of it worked. Outcomes sits at 1.8, discovery at 2.4.

The overall 3.1 is the least interesting number on the page. It averages a real strength with a real hole and reports the midpoint, which describes no part of this

team accurately.

The trap

The obvious move is to attack Outcomes, the lowest score. It is also the wrong first move, and for a specific reason: analytics on a product whose discovery is weak will measure features nobody validated. You would instrument the wrong things beautifully.

Discovery at 2.4 is the upstream constraint. Fix it and the analytics work gets a question worth answering; fix analytics first and you buy dashboards.

What Atlas actually did

They chose **one domain** — Product Discovery & Validation — and took two actions from the Foundational band, because at 2.4 that is where they are:

- Talk to five users of the last thing shipped and write down what they actually do. Owner: the PM. Date: end of month.
- Before the next significant build, write the one assumption that would sink it, and how you would find out cheaply. Owner: the tech lead. Date: next planning.

They also left Business Value & Economics alone. It came back as *not enough information* — nobody on the team could see the numbers — and the honest next step there was a conversation with finance, not an improvement initiative.

> **One quarter later**
>
> Discovery moved from 2.4 to 3.0, and — unprompted — Outcomes moved from 1.8 to 2.3, because having a validated assumption gave them something worth measuring. The overall score moved from 3.1 to 3.3, which nobody cared about. The two domains that moved are the story.

PART III — IMPROVING

CHAPTER 10

Turning the result into change

Pick one domain. Not three, and certainly not all ten. A maturity result is a map of everything that could be better, and treated as a to-do list it produces ten half-finished improvements and no change.

The domain to choose is rarely the lowest one. It is the lowest one *that is blocking something you already care about*. A weak discovery practice matters enormously if you keep shipping things nobody uses, and barely at all this quarter if your problem is that releases keep breaking.

> **Three questions that pick the domain**
>
> **1.** Which low domain is upstream of the others? Discovery feeds outcomes; strategy feeds prioritization. Fixing downstream first wastes the work.
>
> **2.** Which one is costing us something we can name this quarter? If you cannot name the cost, the team will not sustain the effort.
>
> **3.** Which one can we actually move? A gap caused entirely by decisions above the team is real, but it is an escalation, not a retrospective action.

Match the advice to where you are

Each domain chapter carries three bands of recommendations. They are sequential, not alternatives.

Foundational when you are Ad-hoc or Emerging

Establish the practice at all. Start here even if it feels basic — skipping ahead is the most common way a maturity effort stalls.

Scaling when you are Defined

Make the practice consistent and measurable, so it survives people leaving and quarters getting busy.

Optimizing when you are Managed or Optimized

Turn the practice into a source of advantage: evidence thresholds, reallocation, influence beyond the team.

Choose two or three actions, give each one a name and a date, and put them where the team already looks. PML keeps an action plan for exactly this, so the result does not end its life as a screenshot in a slide deck.

One honest test

If nothing about next month looks different after the assessment, the assessment did not happen. A score that produced no decision is a score you paid twenty minutes for and did not spend.

CHAPTER 11

The rhythm

Reassess about once a quarter. Sooner and you measure noise — practices take weeks to become habits and longer to feel normal. Much later and you lose the thread between what you decided and what moved.

Compare with your own past first. The delta between your two readings is a cleaner signal than the distance between you and another team, because it holds the context constant: same product, same constraints, same people.

Expect movement to be modest and uneven. A domain going from 2.4 to 3.0 in a quarter is a real result. A domain jumping two full levels usually means the first reading was pessimistic or the second was generous, and it is worth asking which.

Two readings only compare if they used the same instrument

A maturity framework evolves. When the statements or the domains change, a before and after stop being comparable, and PML will say so rather than show you a difference that is really a change of ruler. Your past results stay exactly as they were measured.

CHAPTER 12

Running it across several teams

Measuring one team tells that team something. Measuring eight tells the organization something — but only if it is set up to produce learning rather than a league table. The difference is mostly in how it is introduced.

Say what happens with the results, before anyone answers

Who sees them, what they will be used for, and what they will never be used for. Ambiguity here is answered by people's worst assumption, and their worst assumption inflates every score you are about to collect.

Put every team on the same instrument and the same window

Same framework version, same three-month window, all within a couple of weeks of each other. Readings taken six months apart under different conditions do not compare, however much they look like they do.

Ask teams to name their product by team name

Otherwise you end up with eight radars and no reliable way to tell whose is whose. It sounds trivial until you are looking at the spreadsheet.

Publish a map, not a ranking

The artefact to share is *who is strong where*, not *who is ahead*. Same data, completely different behaviour afterwards: one invites a conversation, the other invites a defence.

Spend the strengths

The highest-return thing an organization can do with eight results is book one session where the strongest team in a domain spends ten minutes explaining how they actually work. That exchange is the whole reason to measure more than one team.

What not to do with the aggregate

Do not set an organizational maturity target. Do not put it in anyone's objectives. Do not report an average across teams as a health metric — it hides exactly the variation that makes the data useful, and it converts the instrument into a target within one cycle.

PART IV — REFERENCE

CHAPTER 13

The 10 domains

In canonical order, which never changes — it is what keeps the shape of a radar comparable between readings and between teams. Each chapter holds what the domain covers, what good looks like, where teams usually fall short, the exact statements you will be asked, and what to do next at each stage.

DOMAIN 1 OF 10

Product Vision & Strategy

Clarifies the problem, future direction, distinctive value and strategic adaptability.

WHY IT MATTERS

Without a stated problem and a direction worth defending, every stakeholder request arrives looking equally urgent. Strategy is what lets a team decline work without it feeling arbitrary.

What good looks like

- The team can name the user problem the product exists to solve, without consulting a document.
- The vision describes a change in the world rather than a list of things to build.
- Strategy is revisited when evidence moves, not only when a plan misses.

Where the gaps usually are

- The product is described by its features rather than by the problem it addresses.
- Nobody can say what makes this product a better choice than the realistic alternatives.
- Strategy is written once and then referenced only in planning season.

What you will be asked 5 statements

1. We can explain the main user problem our product exists to solve.
2. Our product vision describes the future change we want to create, not just features to deliver.
3. Our product strategy is linked to relevant organizational objectives.
4. We can explain our product's distinctive value compared with realistic alternatives.
5. We review our product strategy when meaningful new evidence or context changes emerge.

How to move forward

FOUNDATIONAL Ad-hoc or Emerging

- Write one paragraph naming the user problem, who has it, and what it costs them today.
- State the distinctive value in a sentence that a competitor could not honestly copy.
- Ask three people on the team to describe the product direction in their own words, then fix the gaps you hear.
- Link each item currently in flight to an organizational objective, and surface the ones that link to none.

SCALING Defined

- Turn the vision into three outcome-based goals with a measurable target each.
- Add explicit non-goals, so the strategy constrains decisions instead of only inspiring them.
- Put a standing agenda on a quarterly review: what changed, what we learned, what we now believe.
- Record the assumptions the strategy rests on, so you know what would make you change it.

OPTIMIZING Managed or Optimized

- Attach evidence thresholds to the major bets: what result would make you double down, and what would make you stop.
- Run a pre-mortem before funding a significant commitment, and write down the failure modes you accept.
- Track leading indicators for the strategy itself, separate from delivery progress.
- Use the strategy review to move investment between bets, not only to report on them.

DOMAIN 2 OF 10

Customer Understanding & Adoption

Understands the people served, their barriers, first value and sustained value.

WHY IT MATTERS

Knowing who the product is for is only half of it. The other half is knowing what stands between those people and the value you promised — before adoption, and long after it.

What good looks like

- The main groups of users are distinguished, with their different problems described.
- There is a regular supply of credible evidence about needs, not only at launch moments.
- The team knows what a user has to accomplish to get value for the first time.

Where the gaps usually are

- Users are described as one undifferentiated group.
- Evidence about needs arrives in bursts, usually just before a big decision.
- Nobody tracks whether users keep getting value after the first weeks.

What you will be asked 5 statements

1. We distinguish the main groups of people who use or depend on our product.
2. We regularly gather credible evidence about users' needs.
3. We identify important barriers that prevent users from achieving their goals.
4. We understand what users need to accomplish to experience their first meaningful value.
5. We evaluate whether users continue to obtain meaningful value after initial adoption.

How to move forward

FOUNDATIONAL Ad-hoc or Emerging

- Name the two or three groups who depend on the product and write one sentence on what each is trying to do.
- Book a recurring slot for user conversations and protect it when delivery gets busy.
- List the barriers you already know about, from support tickets and sales calls, before commissioning new research.
- Define, concretely, what 'first value' means for a new user, and how you would know it happened.

SCALING Defined

- Put the evidence somewhere the whole team reads, rather than in a researcher's folder.
- Instrument the path to first value and find where people stop.
- Separate the barriers you can remove from the ones you must explain, and treat them differently.
- Look at whether users who adopted six months ago are still getting value, and from what.

OPTIMIZING Managed or Optimized

- Run continuous discovery with the whole team exposed to users, not summaries of users.
- Track sustained value as a product measure with an owner, alongside acquisition.
- Use segment-level differences to decide what to build, not only to describe your users.
- Close the loop: show the team which recent decisions changed because of something a user said.

DOMAIN 3 OF 10

Product Leadership & Operating Model

Evaluates decision rights, collaboration, autonomy, stakeholder alignment and capability development.

WHY IT MATTERS

Most product delay is not technical. It is ambiguity about who decides, disciplines arriving too late to influence anything, and disagreements settled by seniority rather than evidence.

What good looks like

- People know who owns which decision, so decisions rarely stall waiting for permission.
- Design and engineering are in the room early enough to change the shape of the work.
- Disagreements with stakeholders are resolved against stated goals and evidence.

Where the gaps usually are

- Decisions wait because nobody is certain whose call it is.
- Other disciplines are consulted once the solution is already chosen.
- The team has responsibility for outcomes but little authority over approach.

What you will be asked 5 statements

1. Ownership of significant product decisions is clear to the people involved.
2. Relevant disciplines are involved early enough to influence product decisions.
3. Our team can make product decisions autonomously within agreed boundaries.
4. We resolve stakeholder disagreements using clear product goals and evidence.
5. We improve product management capability through feedback and shared learning.

How to move forward

FOUNDATIONAL Ad-hoc or Emerging

- Write down who decides what, for the five decisions that most often stall.
- Move the point at which engineering and design join from solution review to problem framing.
- Agree the boundaries the team can act within without asking, and put them in writing.
- When a disagreement is resolved, record what evidence settled it.

SCALING Defined

- Give each recurring decision a named owner and a stated input list, so consultation is predictable.
- Replace status reporting to stakeholders with a short written update on goals, evidence and open questions.
- Make the team's autonomy boundaries explicit enough that crossing one is obvious to everyone.
- Build a habit of shared learning — a short regular session where the team improves how it works, not only what it ships.

OPTIMIZING Managed or Optimized

- Review the decision boundaries periodically and widen them where the team has earned it.
- Develop product capability deliberately: pairing, feedback on real decisions, rotation through problem types.
- Make the operating model itself something the team can change, with evidence.
- Let the team's way of working be documented well enough that another team could adopt it.

DOMAIN 4 OF 10

Prioritization & Investment Decisions

Evaluates opportunity selection, trade-offs, opportunity cost, decision communication and planning.

WHY IT MATTERS

Prioritization is an investment decision under scarcity. A team that never names what it is declining is not prioritizing — it is queueing, and the queue decides.

What good looks like

- Priorities are argued from their expected contribution to outcomes.
- Decisions to defer or decline are visible, not silent.
- When priorities change, the people affected hear why.

Where the gaps usually are

- Everything is accepted and nothing is explicitly declined.
- Opportunity cost is never discussed: the question is whether something is worth doing, never what it displaces.
- Plans communicate dates and features but not the outcome they are meant to produce.

What you will be asked 5 statements

1. We prioritize opportunities according to their expected contribution to product outcomes.
2. We explicitly consider opportunity cost when making product investment decisions.
3. We make visible decisions about which opportunities to defer or decline.
4. When priorities change, the reasons are clear to the people affected.
5. Our product plans communicate intended outcomes, rather than only delivery commitments.

How to move forward

FOUNDATIONAL Ad-hoc or Emerging

- For the next three decisions, write the outcome each is expected to move before committing.
- Keep a visible list of what you decided NOT to do this quarter, and why.
- When a priority changes, send one short note to the people it affects, with the reason.
- Before saying yes to new work, name what it displaces.

SCALING Defined

- Adopt one consistent way of comparing opportunities, and apply it to everything above a size threshold.
- Make opportunity cost explicit in the decision itself: the alternatives considered and why they lost.
- Express plans as outcomes with confidence levels rather than fixed feature dates.
- Review declined opportunities periodically; some become right later, and it is cheaper to revisit than to rediscover.

OPTIMIZING Managed or Optimized

- Reallocate investment between bets during the quarter when evidence moves, not only at planning time.
- Track how often a prioritized item produced the outcome it promised, and feed that back into the method.
- Treat capacity for unplanned work and technical health as a deliberate allocation, not as spillage.
- Make the trade-off conversation routine enough that stakeholders bring their own opportunity cost argument.

DOMAIN 5 OF 10

Product Discovery & Validation

Frames problems, identifies risks, validates assumptions and examines alternatives.

WHY IT MATTERS

The expensive mistake is not building something badly. It is building the right-looking thing for a problem you never checked, and discovering it after it ships.

What good looks like

- The underlying problem is investigated before a significant solution is committed to.
- The team can name the assumptions most likely to make a solution fail.
- Validation effort is proportionate to the risk being carried.

Where the gaps usually are

- Solutions arrive already chosen, and discovery becomes justification.
- Risk is assessed by how large the build is, not by how uncertain the assumption is.
- Only one approach is ever on the table, however uncertain the problem.

What you will be asked 5 statements

1. We investigate the underlying problem before committing to a significant solution.
2. We identify the assumptions most likely to invalidate a proposed solution.
3. We use credible user evidence to test our understanding of significant problems.
4. We test uncertain solution assumptions using methods proportionate to their risk.
5. When uncertainty is high, we compare plausible approaches before selecting a solution.

How to move forward

FOUNDATIONAL Ad-hoc or Emerging

- Before the next significant build, write the problem statement and what evidence supports it.
- List the assumptions that would make the solution worthless if wrong, and rank them.
- Test the riskiest assumption with the cheapest method that could falsify it.
- When uncertainty is high, sketch a second plausible approach before picking the first.

SCALING Defined

- Run discovery continuously alongside delivery rather than as a phase in front of it.
- Match method to risk explicitly: a conversation for some assumptions, a prototype or experiment for others.
- Keep evidence attached to the decision it informed, so later readers can see the basis.
- Make it normal and cheap to abandon an approach that the evidence did not support.

OPTIMIZING Managed or Optimized

- Maintain a live map of the assumptions behind the current strategy, with their confidence and evidence.
- Measure how often validated assumptions held after launch, and recalibrate your methods.
- Compare alternatives on the dimension that matters — risk reduced per unit of effort.
- Share discovery findings across teams so one team's evidence stops another's repeat work.

DOMAIN 6 OF 10

Product Delivery & Continuous Improvement

Assesses value increments, flow, release readiness, post-release response and lifecycle evolution.

WHY IT MATTERS

Delivery maturity is not speed. It is the ability to release something small, know whether it worked, respond when it did not, and retire what stopped earning its place.

What good looks like

- Changes ship in increments that either deliver value or produce useful feedback.
- Recurring obstacles to flow are named and addressed rather than endured.
- Problems that reach users after release are detected rather than reported by them.

Where the gaps usually are

- Work ships in large batches, so feedback arrives late and in aggregate.
- The same delivery obstacles recur without anyone owning their removal.
- Capabilities accumulate; nothing is ever retired.

What you will be asked 5 statements

1. Where feasible, we deliver product changes in increments that provide value or useful feedback.
2. We identify and address recurring obstacles that slow product delivery.
3. We assess product changes against agreed readiness criteria before release.
4. We detect and respond to problems that affect users after release.
5. We improve or retire product capabilities when evidence shows they no longer serve users well.

How to move forward

FOUNDATIONAL Ad-hoc or Emerging

- Take the next large piece of work and find a smaller slice that still produces a signal.
- Write down the three obstacles that most often slow a release, and pick one to remove this month.
- Agree what 'ready to release' means and apply it consistently.
- Decide how you would learn that a release broke something for users, and set that up.

SCALING Defined

- Make release a routine, low-ceremony event so batch size is a choice rather than a constraint.
- Track where work waits, not only how long it takes, and attack the waiting.
- Review capabilities periodically against evidence of use, and retire what no longer serves.
- Close the loop between a post-release problem and the readiness criterion that would have caught it.

OPTIMIZING Managed or Optimized

- Treat flow improvement as standing work with a visible owner and measured effect.
- Make reverting or disabling a change as routine as shipping one.
- Use lifecycle decisions — improve, replace, retire — as a regular portfolio conversation.
- Let delivery data inform strategy: what the team can ship reliably shapes what it should promise.

DOMAIN 7 OF 10

Outcomes & Product Analytics

Assesses measurable goals, outcome metrics, trustworthy data and learning from evidence.

WHY IT MATTERS

A team that measures only what it completed will always look productive. Measuring outcomes is what makes it possible to be wrong in public, and therefore to learn.

What good looks like

- Success criteria are defined before a significant initiative is committed to.
- Key measures describe what changed for users or the business, not what was finished.
- The limits of the data are understood before conclusions are drawn from it.

Where the gaps usually are

- Measures count output: items shipped, points completed, releases made.
- Success is defined after the fact, usually favourably.
- Numbers are trusted without anyone having checked how they are collected.

What you will be asked 5 statements

1. We define measurable success criteria before committing to significant product initiatives.
2. Our key product measures reflect outcomes, rather than only work completed.
3. We assess the reliability and limitations of product data before relying on it.
4. We compare observed results with the outcomes expected from released changes.
5. We reconsider product decisions when credible outcome evidence challenges our assumptions.

How to move forward

FOUNDATIONAL Ad-hoc or Emerging

- For the next initiative, write the expected outcome and how you will measure it, before building.
- Pick a small set of measures the team genuinely watches, and drop the rest from reporting.
- Write down, for your main metric, exactly what it counts and what it misses.
- After the next release, compare what happened with what you expected, and record the difference.

SCALING Defined

- Plan instrumentation as part of the work rather than adding it after launch.
- Make post-launch evaluation a scheduled step, not something that happens when someone remembers.
- Distinguish measures that guide decisions from measures that only report status.
- When data and belief disagree, write down which one you acted on and why.

OPTIMIZING Managed or Optimized

- Run structured experiments with stated hypotheses where the question justifies the cost.
- Review the reliability of key data sources on a cadence, and publish known limitations.
- Let outcome evidence trigger reconsideration of committed decisions, visibly.
- Hold the team to learning as an output: what did we believe, what do we believe now, what changed it.

DOMAIN 8 OF 10

Technical Sustainability & Reliability

Considers technical constraints, debt, reliability, security and proportionate scalability.

WHY IT MATTERS

Technical reality is a product input, not an implementation detail. Teams that discover constraints after committing spend their credibility explaining delays they could have predicted.

What good looks like

- Technical constraints are part of the conversation while opportunities are still being evaluated.
- Technical debt is described by its impact and carried into investment decisions.
- Reliability, security and privacy expectations are explicit and proportionate.

Where the gaps usually are

- Engineering is asked to estimate a solution that was shaped without them.
- Debt is discussed as a feeling rather than as a cost with consequences.
- Scalability is either ignored or over-engineered, with no evidence either way.

What you will be asked 5 statements

1. We consider relevant technical constraints when evaluating significant product opportunities.
2. We account for technical debt in product investment decisions.
3. We understand the reliability requirements that matter to our users.
4. We account for relevant security and privacy risks in product decisions.
5. We consider realistic changes in demand when making scalability decisions.

How to move forward

FOUNDATIONAL Ad-hoc or Emerging

- Bring an engineer into the evaluation of the next significant opportunity, before the solution is chosen.
- List the three pieces of technical debt that most slow the team, described by what they cost.
- Write down the reliability users actually need — not an aspiration, the real threshold.
- Name the security and privacy risks relevant to the next change, however briefly.

SCALING Defined

- Schedule debt reduction as work with an expected benefit, so it competes honestly for capacity.
- Agree explicit expectations for performance, reliability and security, and check changes against them.
- Base scalability decisions on a realistic demand range, stated out loud.
- Record significant technical decisions with the trade-off accepted, so later readers inherit the reasoning.

OPTIMIZING Managed or Optimized

- Treat architecture as something that enables the next year of strategy, and review it against that.
- Make the cost of debt visible in the same terms as feature investment.
- Rehearse failure — know how the product degrades and what users experience when it does.
- Let reliability and security expectations be revisited as the product's role changes.

DOMAIN 9 OF 10

Business Value & Economics

Connects product investment to organizational value, benefits, costs and cost of delay.

WHY IT MATTERS

A team that cannot explain how its product creates value will have that explanation supplied by someone else, usually at budget time, usually unfavourably.

What good looks like

- The team can explain how the product contributes to organizational value or mission.
- Significant investments carry an estimate of the benefit expected.
- Realized benefit is compared with what was invested.

Where the gaps usually are

- Value is argued entirely in user terms, with no commercial or mission case.
- Nobody on the team knows what the product costs to build and run.
- Delay is treated as free: postponing something has no stated consequence.

What you will be asked 5 statements

1. We can explain how our product contributes to organizational value or mission outcomes.
2. We estimate expected benefits when evaluating significant product investments.
3. We understand the main costs of building and operating our product.
4. We compare realized benefits with the resources invested when evaluating product value.
5. We consider the consequences of delaying important product improvements.

How to move forward

FOUNDATIONAL Ad-hoc or Emerging

- Write one paragraph on how this product creates value for the organization, and check it with someone in finance or strategy.
- Find out the main costs of building and running the product, even approximately.
- For the next significant initiative, state the benefit you expect, in whatever unit fits.
- When something is postponed, say what the delay is expected to cost.

SCALING Defined

- Attach an expected benefit to every initiative above a size threshold, and revisit it after delivery.
- Build a working relationship with finance and agree the numbers you will both use.
- Make cost of delay part of the prioritization argument rather than an afterthought.
- Report outcomes in business terms alongside product metrics.

OPTIMIZING Managed or Optimized

- Compare realized benefit with investment as a routine review, and let the pattern inform future estimates.
- Give product a real voice in pricing and packaging, where those exist.
- Model the economics of the main alternatives before committing, not after.
- Treat the product's unit economics as something the team owns and can move.

DOMAIN 10 OF 10

AI-Enabled Product Management

Assesses appropriate, responsible and outcome-oriented AI support to product management work.

WHY IT MATTERS

This domain is about judgement, not adoption. A team that declines AI for good reasons and a team that uses it carefully are both mature; the immature position is using it without knowing whether it helped.

What good looks like

- The team can say where AI helps its product work and where it does not.
- AI output is verified before it informs a significant decision.
- Whether AI-assisted practice actually improved the work is assessed, not assumed.

Where the gaps usually are

- AI is used because it is available, with no defined need behind it.
- Generated claims are passed into decisions without checking.
- Nobody has asked whether the AI-assisted way is better than the way it replaced.

What you will be asked 5 statements

1. Our team recognizes appropriate and inappropriate uses of AI in product management work.
2. When permitted and useful, we apply AI to address a defined product management need.
3. We verify AI-generated claims before relying on them in significant product decisions.
4. Our AI practices follow the data-handling and security requirements relevant to our work.
5. We evaluate whether AI-assisted practices meaningfully improve the quality or efficiency of our work.

How to move forward

FOUNDATIONAL Ad-hoc or Emerging

- List two tasks in your product work where AI plausibly helps, and two where it plausibly does not.
- Check what your organization permits before adopting anything, and work inside it.
- Agree a simple rule: any AI-generated fact that enters a decision gets verified first.
- Pick one defined need and try AI against it, rather than adopting it generally.

SCALING Defined

- Make verification explicit in the workflow, so it does not depend on individual caution.
- Apply the data-handling rules that already govern your other tools to AI use as well.
- Compare the AI-assisted result with the previous way of working on something you can measure.
- Share what worked and what did not across the team, so adoption is collective rather than personal.

OPTIMIZING Managed or Optimized

- Evaluate AI-assisted practices on quality and effort with the same rigour you apply to product changes.
- Retire AI use that did not demonstrate benefit, as readily as you adopted it.
- Keep a clear boundary between where AI accelerates work and where human judgement must own the decision.
- Revisit the boundary as the tools change, rather than settling it once.

CHAPTER 14

What this instrument cannot tell you

Every measurement has edges, and a guide that hides them is selling something. These are PML's.

It is self-reported

Nothing is verified against your repository, your tickets or your customers. The result is as good as the honesty in the room — which is the argument for answering together, and for saying up front that nobody is being graded.

Every statement is positively phrased

There is a known tendency to agree with statements as presented. Without reverse-worded items to brake it, a team answering quickly will drift upward. If your scores come out high and even across every domain, suspect this before celebrating.

It measures practices, not outcomes

Mature practices make good outcomes more likely; they do not guarantee them. A team can run excellent discovery on a product the market does not want. Read a high score as a well-built engine, not as proof you are driving somewhere good.

It describes a team, not a person

Nothing here supports a judgement about an individual, and a result used that way is being misread. The unit of maturity is the way a group works.

None of this makes the exercise less useful. It makes it a conversation starter with a number attached, which is what it was built to be.

CHAPTER 15

Questions that come up

Our team is only a few months old. Is it too early?

No, and an early baseline is worth more than a late one. A new team scoring low everywhere is not a failure, it is an accurate description of a team that has not had time to build practices yet. The value is that in six months you can prove you built them.

We are not a classic product team. Does this still apply?

Mostly. Platform, internal tools and service teams usually find that one or two domains genuinely do not fit — pricing is the common one — and that is what Not applicable is for. If more than two domains are not applicable, the result will still be useful per domain, but treat the overall score with care.

Who should answer: the PM, or the whole team?

The whole team, in one sitting, with one person typing. A PM answering alone produces a description of the PM's view. The team answering together produces the disagreements, which is the most valuable output of the exercise.

What if people worry this will be used against them?

Then it will be, in effect — because they will answer accordingly and you will get a flattering, useless result. Say explicitly who sees it and what happens next, before you start. If the honest answer is that it feeds a performance conversation, do not run it; find another instrument.

Should we aim for level 5 everywhere?

No. The cost of each level rises and the return does not rise with it. A small team deliberately operating at Defined in most domains, with two at Managed where it matters to their strategy, is making a good decision — not an immature one.

How long until a score moves?

A deliberate change in one domain typically shows up in the next quarterly reading. If nothing moved after two readings, the problem is usually that the actions were never owned by anyone, not that the practice is hard.

References & Further Reading

Grouped by how each item relates to this work, because a single list would imply they all contributed equally and they did not.

FOUNDATIONS

Cited directly in the text.

Paulk, M. C., Curtis, B., Chrissis, M. B., & Weber, C. V. (1993). Capability Maturity Model for Software, Version 1.1 (CMU/SEI-93-TR-024). Software Engineering Institute, Carnegie Mellon University. https://apps.dtic.mil/sti/pdfs/ADA263403.pdf

The origin of the idea that capability predicts the variance of outcomes, not only its average. Cited in Chapter 3.

RESEARCH REVIEWED FOR THE PML 2.0 REVISION

Reviewed while redesigning the framework for version 2.0. The original framework came out of the author's own practice, not from these reports, and none of them validates PML.

ProductPlan. The State of Product Management Report 2026. https://www.productplan.com/ebooks/the-state-of-product-management-report-2026

Atlassian. State of Product 2026. https://www.atlassian.com/software/jira/product-discovery/resources/state-of-product-2026

Product Focus. Annual Survey of the Product Management Profession, 2025/26 edition. https://www.productfocus.com/product-management-resources/profession-survey/

Galante, L., & Barlien, S. State of Platform Engineering Report: Volume 4. Platform Engineering. https://platformengineering.org/reports/state-of-platform-engineering-volume-4
The publisher states no publication year for this volume, so none is given here.

Merge. The 2026 State of Product Integrations. https://www.merge.dev/sopi-2026

Delight Path. The 2026 State of Product Leadership Report. https://www.delightpath.com/product-leadership-report-2026

Framna. The state of product development 2026. https://framna.com/the-state-of-product-development-2026-report

Productboard. The State of AI in Product Management. https://www.productboard.com/ebook/the-state-of-ai-product-management/
Undated by the publisher. Based on a survey of 379 product professionals.

National Institute of Standards and Technology (2023). Artificial Intelligence Risk Management Framework (AI RMF 1.0) (NIST AI 100-1). https://www.nist.gov/publications/artificial-intelligence-risk-management-framework-ai-rmf-10
Informed the responsible-use framing of the AI domain.

Google Cloud / DORA (2025). State of AI-assisted Software Development 2025. https://dora.dev/research/2025/dora-report/

FURTHER READING

Relevant to the subject and worth your time. Listed without any claim that they were used to build PML.

Perri, M. (2018). Escaping the Build Trap: How Effective Product Management Creates Real Value. O'Reilly Media.

Torres, T. (2021). Continuous Discovery Habits.
No publishing imprint is stated by the author, so none is given here.

Cagan, M., & Jones, C. (2020). EMPOWERED: Ordinary People, Extraordinary Products. Wiley.

Cagan, M. (2024). TRANSFORMED: Moving to the Product Operating Model. Wiley.

Product Focus. Product Management Maturity Model. https://www.productfocus.com/blog/product-management-maturity-model
A related maturity framework, with three levels across five dimensions.

Kniberg, H., & Lindwall, K. (2014). Squad Health Check model — visualizing what to improve. https://blog.crisp.se/2014/09/16/henrikkniberg/squad-health-check-model
A related team self-assessment model, not a validation of PML. Originally published on the Spotify Labs blog, which is no longer online; this is the authors' own reposting.

About the Author

Sandro Maia de Araújo is a product management professional with international experience in digital products, B2B platforms, APIs, integrations and product strategy. His work focuses on connecting customer needs, technical capabilities, organizational ways of working and measurable outcomes.

He created Product Maturity Level to help product teams understand how they work today, identify where improvement would matter most, and build stronger product practices over time.

About Product Maturity Level

Product Maturity Level is an independent framework for assessing how a product team actually works. It is a self-assessment of observable practices, not a ranking, a certification or an audit.

A team answers 50 statements across 10 domains, and receives a reading on five maturity levels — per domain, not only overall. Statements that do not apply, and statements the team genuinely cannot answer, are recorded as such rather than scored, so a thin domain reports as insufficient evidence instead of as a failure.

It is designed to be answered together, in one sitting, by the people who do the work. The intended output is a conversation, one chosen area to improve, and a second reading a quarter later that shows whether anything moved.

Run the assessment

Free, with no sales conversation attached.

productmaturitylevel.com

Measure today. Evolve tomorrow.

productmaturitylevel.com

PML — Product Maturity Level · First Edition 2026 · PML 2.0.0