

PRODUCT MANAGEMENT FRAMEWORK

The Opportunity Solution Tree

Outcome → Opportunities → Solutions → Experiments

A Practical Guide to Continuous Discovery & Evidence-Based Roadmapping

Prepared by	Harsh - Product Manager
Use case	Continuous discovery & outcome-driven roadmapping
Category	Product Strategy - Discovery Frameworks

Table of Contents

- 1. What Is the Opportunity Solution Tree? 3
- 2. The Four Levels of the Tree 3
- 3. Core Principles for Building One 5
- 4. Worked Example 5
- 5. How to Build a Tree - Step by Step 6
- 6. When to Use It and Its Limitations 7
- 7. Reusable Template 7

1. What Is the Opportunity Solution Tree?

The Opportunity Solution Tree (OST) is a visual discovery framework created by Teresa Torres to connect everyday product decisions back to a single measurable outcome. Instead of starting from a feature idea, the tree starts from the outcome the team is accountable for, then works backward through customer opportunities unmet needs, pain points, and desires to the solutions worth testing and the experiments that will validate them.

The structure keeps discovery honest: every solution on the tree must trace back through a real customer opportunity to the outcome it is meant to move. This makes it much harder for a team to build features that are interesting to build but disconnected from what actually drives the metric they own.

Why PMs use it

- Keeps the team anchored to a measurable outcome instead of a stream of disconnected feature requests.
- Makes it visually obvious when a solution has no supporting opportunity a sign it's solutioning without evidence.
- Encourages breadth-first exploration of opportunities before jumping to a single solution.
- Creates a living artifact that continuous-discovery teams update weekly as new customer evidence comes in.

2. The Four Levels of the Tree

The tree has four levels, read top to bottom. Each level answers a different question, and each node should trace to exactly one parent node above it.

1 Desired Outcome

The measurable business or customer result the team is accountable for.

- Framed as a metric with a direction and, ideally, a target (e.g., “increase 30-day activation from 42% to 55%”)
- Owned by the team, not imposed as a vague company-wide mission statement
- Sits at the top of the tree there is only ever one per tree

2 Opportunities

Unmet customer needs, pain points, or desires that stand between the customer and the outcome.

- Framed from the customer's point of view, not the business's a need, not a feature
- Sourced from interviews, support tickets, sales calls, and usage data, not brainstorming alone
- Can have child opportunities nested underneath, breaking a broad need into smaller ones

3 Solutions

Concrete product ideas that could address a specific opportunity.

- Multiple solutions should be generated per opportunity before committing to one
- Solutions can range from small UI tweaks to entirely new capabilities
- Each solution must trace to a specific opportunity above it never straight to the outcome

4 Experiments

The smallest test that generates evidence about whether a solution is worth building.

- Can be a prototype test, a fake-door test, a concierge test, or a small-scale A/B test
- Designed to produce a learning fast and cheaply, before committing full build capacity
- Feeds evidence back up the tree a failed experiment can kill a solution or an entire opportunity branch

3. Core Principles for Building One

- Breadth before depth map many opportunities and many solutions before committing to build any single one.
- Opportunities are needs, not features “users want a dashboard” is a solution in disguise; “users can’t tell if setup succeeded” is an opportunity.
- Every node needs a parent a solution with no opportunity above it, or an opportunity with no evidence behind it, is a red flag.
- The tree is never “done” continuous discovery teams revisit and reshape it weekly as new customer evidence arrives.
- One outcome per tree if the team is accountable for multiple outcomes, build a separate tree for each.

4. Worked Example

The diagram below shows a partial tree for a B2B SaaS onboarding team accountable for activation rate.

OUTCOME Increase 30-day activation rate for new accounts from 42% to 55%

OPP Users don’t know which setup steps actually matter

SOL Guided setup checklist with a visible progress indicator

EXP A/B test: checklist flow vs. current unguided flow

SOL AI-generated setup recommendations based on company profile

OPP Users stall mid-setup waiting on IT/admin approval

SOL Self-service admin invite flow

EXP Concierge test: manually fast-track 10 pending approvals

SOL Temporary sandbox access while approval is pending

OPP Users don't see product value before finishing full setup

SOL Demo mode with sample data before real setup begins

Reading the tree: the outcome is fixed and measurable. Three distinct opportunities compete for the team's attention, each grounded in a customer need rather than a feature pitch. The first opportunity already has two candidate solutions, one of which has a scoped experiment ready to run the second solution is still waiting on its own test. The third opportunity has only one solution mapped so far, a signal to generate more before committing engineering time to it.

5. How to Build a Tree Step by Step

1. Write the desired outcome at the top as a specific, measurable metric with a current value and a target.
2. Gather opportunities from ongoing discovery customer interviews, support themes, sales objections, and usage data not a single brainstorm.
3. Phrase each opportunity as a customer need or pain point, not as a feature request in disguise.
4. Break broad opportunities into smaller, more specific child opportunities where useful.
5. For each opportunity the team chooses to pursue, brainstorm multiple candidate solutions before picking a favorite.
6. Design the smallest possible experiment for the most promising solution to generate evidence cheaply.
7. Run the experiment, record the evidence, and update the tree promote, reshape, or prune branches based on what was learned.
8. Revisit the whole tree on a regular cadence (weekly is common) as new discovery evidence arrives.

6. When to Use It and Its Limitations

Best suited for

- Teams practicing continuous discovery who need a living artifact to organize ongoing customer research
- Situations where the roadmap risks becoming a list of disconnected feature requests
- Aligning cross-functional stakeholders around a single measurable outcome rather than a feature backlog

Known limitations

- Takes real discipline to keep current a stale tree is worse than no tree, since it creates false confidence
- Doesn't rank opportunities or solutions on its own pair it with RICE or ICE once candidates are identified
- Requires a steady stream of real customer contact; without ongoing discovery, the opportunities go stale fast
- Can become unwieldy on paper for large trees most teams maintain it in a dedicated whiteboard or discovery tool

PM Practice Tip

- Use the Opportunity Solution Tree to decide what to explore, then use RICE or ICE to decide what to build first once a solution has real evidence behind it the three frameworks are complementary.

7. Reusable Template

Copy the structure below into a whiteboard tool, slide, or discovery tracker. Keep the outcome metric visible at the top of every working session so opportunity and solution discussions stay anchored to it.

Level	Node	Evidence / Notes	Status

Suggested "Level" values: Outcome, Opportunity, Solution, Experiment filter or color-code by level once the tracker has enough rows to need it.