



Measure the Foot

How to know what product is worth developing

The Claim

A **pain point** is what a customer experiences when their solution fits their problem badly. A **feature request** is a solution they imagine would help. Neither describes the underlying customer problem, so neither can tell you what to build. They are a starting point for discovery, not its outcome.

A **customer problem** is a **concrete situation**: what someone is trying to achieve, why, in what circumstances beyond their control, and by what criteria they judge the result. **If you describe the situation exactly, the right solution mostly follows by reasoning.**

Problems and Solutions Are Different Kinds of Thing

A customer problem	A solution
Exists as facts in reality	Comes from imagination
Discovered by research	Invented
Changes slowly , so it is a solid basis for long-term strategy	Changes fast with technology and fashion
The vendor cannot change it	Entirely under the vendor's control
Independent of any solution	Its fit depends on the problem ; a poor fit produces pain points and feature requests
Not directly visible while a solution is being used	Plainly visible when it is used

Exactly like a **foot and a shoe**. A blister tells you the shoe doesn't fit the foot. It doesn't tell you the shape of the foot, and you can't design a good shoe based on information about blisters. Shoe shops have measured feet with the Brannock Device since 1925. Software teams still ship a shoe and then ask about blisters.



"The hard part is no longer writing code but finding the problem to solve and understanding it better than the customer alone can."

Mikko Kärkkäinen, CEO of RELEX, *Helsingin Sanomat*, 9 April 2026

The Backlog Test

For tomorrow's hackathon, before your team builds anything:

1. **Pick one person** your challenge is about.
2. **Write down one concrete situation in their life**, without describing any solution: what they are trying to get done, why, in what circumstances they can't change, and how they would judge the result.
3. **Cross out** everything that refers to solutions: tools and how they are used, and the processes, roles or hand-offs of one organisation.
4. **Keep only what reality imposes.**

Example: a family is moving into a new flat on 1 November and needs the internet working on the day they arrive. They can't change the moving date, the kind of connection the building has, or what the previous tenant is doing. Criteria: the connection works on day one, getting it takes little effort, and the family doesn't pay for two connections. "A web form" or "a call centre" are not in the description. They are potential solutions.

If you can't write the description, you don't yet know if you are solving the right problem. The same test works on the top items of your product backlog.

Measuring a Design Before You Build It

A good solution requires the customer to take **only the actions that are genuinely necessary** to reach the best available result. Because the situation is concrete, you can count them: how many actions does a proposed solution need in *that* situation, at a minimum? Fewer necessary actions means a better fit. As a first approximation, this doesn't require users or building the product.

In the RELEX project I watched an experienced user complete a 30-minute workflow in existing retail software. It took about 300 steps. Only three or four were necessary for a human to take in that situation. She asked for one small thing: a button that would save her 2–3 steps. Most of her steps should not have existed.

The best solution is often **smaller and simpler** than what is already on the market.

Where this fits: it works where customer value comes from getting something done, which covers nearly all business software. It works much less where the value is the experience itself, such as a game or a film. It measures functional fit. It is not sufficient to measure whether customers will pay.

Three Starting Points, One Manoeuvre

What it looked like

A feature request: a profitability figure for each product, so the buyer can drop the unprofitable ones

A pain point: the space planner returns 3 weeks later to say the new products won't fit on the shelves

A scope assumption: assortment planning is its own module, because that is how the market sells it

What the situation turned out to be

The buyer decides about a whole category, not one product. Poor profitability has many causes, and most are better fixed without dropping a product. The software had to help diagnose the cause, not just output a number.

Buyer and space planner are judged on different measures with opposing incentives. Show the space and cost impact **while** the buyer decides, so nobody has to be surprised later.

Assortment and pricing are largely overlapping problems. If they are split in separate modules, the tool won't fit the buyer's job. Unusable tools are what failed competitors had shipped.

The manoeuvre is the same every time: **discover, understand and describe the situation.**

What It Did for RELEX

RELEX entered assortment management, a market where competitors' products had become shelfware that retailers bought and did not use.

3–4

people built the first release in 18 months

5

enterprise customers in the first year

~20

customers by the end of year two: on target

Leader

IDC rating on the first release

"The insights and results from Antti's product discovery and concept design work helped us find the right approach. ... The results have been surprisingly good given how little resources we have spent on it."

Tommi Ylinen, Chief Product Officer, RELEX Solutions

Antti Latva-Koivisto



Helps B2B software teams work out what customers actually need, rather than what they ask for. He has spent 25+ years in B2B software at Comptel, Nokia Software and as a consultant to companies like RELEX Solutions.

deductivediscovery.com · newsletter on B2B SaaS product discovery, with a report based on interviews with 31 B2B SaaS executives and founders
linkedin.com/in/anttilk · anttilk@antti.lk · +358 40 578 2747



Read the newsletter