2015 The Product Is Never Finished

Product Book · https://kb.jeffreysummers.com/docs/2015-the-product-is-never-finished/

Most operators think of their product as something they built.

The menu they designed. The concept they launched. The experience they engineered before they opened the doors. Something that, once built, gets maintained — updated occasionally, adjusted when costs force it, changed when Guests complain loudly enough.

That's the wrong model.

The product isn't something you built. It's something you're building — continuously, in real time, in response to a living ecosystem that keeps moving whether you're paying attention or not.

The Guest changes. Their expectations shift, their price sensitivity moves, their habits evolve with the market around them. The cast changes. The people who deliver the experience today are different from the people who delivered it two years ago — and what they're capable of, what they believe, and what they'll bring to the floor tomorrow is determined by how you've been developing them. The competitive environment changes. The vendor relationships change. The community around the restaurant changes. The cost structure changes.

The product that was right for 2019 isn't automatically right for 2026. Not because it was wrong then — but because the ecosystem it lives inside kept moving, and the product has to move with it.

The operator who treats the product as finished is operating on a snapshot. The operator who treats it as constantly evolving is building something that stays current with the reality it's operating in.

The Ecosystem Is a Living Organism

A restaurant is not an isolated business. It is a node in a relational network — a living ecosystem where every actor is connected to every other actor through the product and the experience it delivers.

The Guest who comes every Friday built their expectation around the whole — not one element of it. The cast member whose identity is tied to what the restaurant stands for. The vendor who built

their supply relationship around the menu and the volume. The community organization sponsored. The neighboring business that benefits from the traffic pattern. The landlord whose property value is partly built on the anchor presence. The regular who has been telling their friends what to order for three years.

None of these relationships exist in isolation. They exist as a network built around a premise — and the product is the most visible expression of that premise. When the product changes, the signal moves through the entire network simultaneously. Not just to the Guests. To every actor who built their relationship around what the restaurant is.

This is what most operators miss when they make product decisions. They think they're changing a menu item, a price point, a format, a service model. They're actually sending a signal to an ecosystem. And the ecosystem responds — sometimes with enthusiasm, sometimes with confusion, sometimes with quiet drift away from a brand that no longer feels like what it was.

The product is alive because the ecosystem is alive. Every decision that touches the Guest experience is a decision about the whole network, not just the element being changed.

Finding What Needs to Change

The living product doesn't change randomly. It changes in response to constraints — the specific conditions that are preventing the product from delivering what the ecosystem needs it to deliver.

The discipline is finding them before they become crises.

Pick the most persistent problem in the product. The menu item that keeps generating waste without generating loyalty. The service model that produces friction at the same point every shift. The price point that keeps coming up in Guest conversations in ways that suggest something isn't landing. The experience element that was designed for a Guest who may no longer be your Guest.

Now drill.

Level 1: What is the visible symptom? State it precisely. One specific, observable result.

Level 2: Why does this symptom keep appearing? What has to be true for it to keep reproducing itself?

Level 3: What constraint is producing that mechanism? What underlying condition — a system, a standard, a Guest expectation, a cost structure, a belief about what the product should be — makes the mechanism possible?

Level 4 (if needed): Why does that constraint exist? Was it designed deliberately or did it develop by default? Has anyone ever actually questioned it — or has it been accepted as the way things are?

By the time you reach level three or four, the real constraint is usually visible. And it's almost never what the original symptom suggested.

The cast member closest to the problem almost always knows something the operator doesn't — because they've been living with the constraint while the operator has been managing the symptom. Ask them. The answer they give is level one. Ask why again. And again. By the third or fourth answer, you're usually at the real constraint — and the person who helped find it is now invested in solving it, because they were part of the diagnosis.

Incremental by Default — Until It Isn't

Once the constraint is identified, the instinct is to solve it completely. That's almost always wrong.

The operator who changes the constraint all at once disrupts the living ecosystem that has to keep functioning while the change is happening. Guests haven't adjusted their expectation. Cast members haven't adapted their behavior. Vendors haven't realigned their supply. The network that was built around the old constraint hasn't been replaced — it's been destabilized.

The default is incremental. One change per period. One adjustment at a time. The direction is set by the constraint. The pace is set by what the ecosystem can absorb without breaking what works.

Over time, incremental changes compound into the non-incremental result the constraint identified as possible. The insight is non-incremental. The practice is relentlessly incremental.

Except when it isn't.

Some constraints require a hard reset. The cast member who has been poisoning the culture for two years doesn't get incremental correction. The menu architecture generating structural food cost problems doesn't get trimmed one item at a time if the whole model is wrong. The positioning sending the wrong Guest to the wrong experience doesn't get adjusted — it gets rebuilt from the premise.

The discipline is knowing which category the constraint falls into. Most product problems compound best through incremental change. Some require a hard stop. The operator who defaults to incremental on everything avoids the discomfort of the hard reset. The operator who defaults to full resets on everything creates constant disruption without the compounding that incremental

change produces.

The constraint tells you what needs to change. Your judgment tells you how fast.

Taking Everyone Along for the Ride

Whether the change is incremental or a hard reset, the principle is the same: you cannot change one element of a living ecosystem in isolation.

Cracker Barrel tried to modernize. The modernization effort didn't fail because the new menu items were bad. It failed because the cumulative signal of the changes told every actor in the network that the restaurant was no longer fully committed to being what they had built their relationship around. The Guest didn't leave because of one item. They left because the brand stopped believing in itself — and they felt it before anyone said it out loud.

Hooters faces the same dynamic. The bar model that held for 15 years didn't just have Guest loyalty. It had cast loyalty, vendor relationships, community presence, and franchise operators who built their businesses around a specific premise. The family dining pivot doesn't just ask Guests to adjust. It asks every actor in the network to rebuild their relationship around something different — while the competitors those Guests, cast members, and vendors might turn to were built for the new premise from the start.

Taking everyone along for the ride doesn't mean asking permission. It means communicating the direction, explaining the reason, and giving every actor in the ecosystem enough context to understand where the product is going and why — so the network moves together instead of fracturing under the weight of a change nobody understood.

The product improves in small moves, not large ones. The operator who waits for the right moment to overhaul the menu, rebuild the service sequence, or redesign the experience is not being strategic — they are using the scale of the change as a reason to avoid making any change at all. The overhaul never comes. The drift continues.

The discipline is incremental. One menu test per quarter — not a rebuild, a test. One new item, one modified preparation, one price adjustment on one category. Run it on a Tuesday before it goes on a Saturday. Run it at one table before it changes the room. Run it in one daypart before it touches the week. If it works, it earns its place. If it doesn't, the cost was one Tuesday.

The operator who compounds through small, tested, deliberate moves is not less innovative than the one planning the overhaul. They are more innovative — because their ideas are actually

running, actually being read, actually producing information about what the Guest responds to. The overhaul is still being planned. The small move is already teaching them something.

Incremental change by default isn't timidity. It's respect for the relational ecosystem that makes the product work. The hard reset is a last resort — not because change is bad, but because in a living ecosystem, nothing changes in isolation. Every constraint is connected to every actor. Change the constraint without taking everyone along for the ride and you don't just lose Guests. You lose the network.

IP Terms

By Design Or By Default — Every outcome in your operation is the result of a deliberate decision or the absence of one. The product that drifts from what the ecosystem needs is operating by default. The operator who actively monitors the ecosystem, drills to the real constraint, and changes the product deliberately — at the right pace, with every actor accounted for — is building by design.

Guest Experience (GX) — The complete arc of what a Guest feels, perceives, and remembers from the moment they consider your restaurant to the moment they decide whether to return. The product is the primary vehicle of the GX. When the product stops evolving with the ecosystem, the GX drifts — and the Guest notices before the operator does.

The Relational Loop — Operator → Cast → Guest → Cast → Operator. A closed circuit. Product decisions move through this loop in both directions. A change that the operator makes without running it through the loop first — without understanding how the cast will deliver it and how the Guest will receive it — produces an outcome the operator didn't intend and the Guest didn't expect.