

BEYOND THE WORKING DEMO

The Capability Cliff: When Automating a Process Destroys the Understanding That Ran It

Tracing the roots of a recurring failure — why capable organisations automate what they have not yet understood, and what it costs them

Giovanni Leonardi

March 2025



Contents

1. Executive Summary	3
2. The shape of the cliff	3
3. Codification is not comprehension	4
4. Why capable organisations walk off the edge	5
5. The strongest case for automating first	5
6. What understanding is, and how it leaves	6
7. Keeping the understanding while taking the labour	7

The Capability Cliff: When Automating a Process Destroys the Understanding That Ran It

Transformation · No. 404
March 2025

"Codification captures the part of a process that was already explicit, and loses the part that was tacit."

1. Executive Summary

Across the current wave of enterprise automation, a specific and expensive failure keeps recurring. An organisation automates a process it does not fully understand; the automation encodes what was written down and silently discards what was not; and when conditions change, no one is left who can explain what the system does, or why. I call this the capability cliff: the point at which automating a process destroys the very understanding that would have let the organisation recover it.

This essay traces the roots of the pattern — not the technical faults, which are well catalogued, but the structural and cultural forces that make otherwise capable enterprises walk off the same edge again and again. It argues three things. That the confusion at the heart of the cliff is the mistaking of codification for comprehension. That the forces sustaining it are largely economic and organisational rather than technical, which is why better technology has not cured it. And that the strongest counter-argument — that learning systems make understanding optional — mistakes a genuine advance in one narrow domain for a general licence. The remedy is not to automate less, but to treat understanding as a deliverable in its own right, and to defend it as deliberately as we pursue efficiency.

2. The shape of the cliff

An insurer automates the first-line triage of household claims. On paper the process is tidy: a set of rules that route each claim to fast settlement, to an assessor, or to the fraud queue. The rules are encoded faithfully, the pilot is a triumph, and the automation turns out to be not merely as good as the human triage team but visibly better — faster, more consistent, tireless. The assessors who used to do the work are, over the following year, released as a saving or moved to other things. The business case realises. Everyone moves on.

Then the weather changes — literally. A run of storms shifts the mix of claims, a new product brings a kind of customer the rules never anticipated, and the automation begins

to fail in the quiet way that automations fail: declining claims it should pass, passing claims it should have queried, all with the same untroubled consistency it showed when it was right. The numbers look fine until, some weeks later, they very much do not. And now the organisation reaches for the people who would once have caught this in an afternoon — who would have looked at a handful of cases and said the rule about outbuildings was never meant to apply when the policy is this kind — and finds that those people are gone. The staff who remain can operate the system. They cannot reason about it. Recovery takes the better part of a year and costs more than the original build.

Here is the cruel geometry of it. Before automation, if the triage team had been overwhelmed or mistaken, the organisation could have retrained a new assessor in a matter of weeks, because the knowledge lived in people who could teach it. After automation, the knowledge lived nowhere that could be consulted under pressure. The efficiency was real. So was the loss, and the loss was invisible on every dashboard until the day it wasn't.

That is the cliff. Note carefully what it is not. It is not that automation fails — automation fails constantly, and organisations absorb the failures and recover. The cliff is that this particular failure removes the ground you would have stood on to recover. It is not a steeper hill. It is a different kind of drop.

3. Codification is not comprehension

At the centre of the pattern sits a single confusion, and naming it is most of the work.

To automate a process you must first write it down. Writing it down feels like understanding it — it is a kind of understanding, and a valuable one — and this feeling is where the trouble begins. Because codification captures the part of a process that was already explicit, and loses the part that was tacit: the exceptions the experienced hand makes without thinking, the we don't do it that way for this kind of customer even though the rule permits it, the sense of when a case smells wrong. The written process is a map. The lived process is the territory. And under the pressure of a delivery timeline, with a working demo on the screen, the map is mistaken for the territory with remarkable ease.

What makes the error so seductive is that the automation frequently executes the written rules better than the humans ever did. This is read as proof that the system has understood the process — when in fact it has inherited only the explicit half and quietly amputated the rest. Competence at the codified part is taken as evidence of command over the whole. It is the difference between knowing that and knowing how, and it is precisely the how — unwritten, embodied, resistant to specification — that does not survive the transcription.

“The automation did not fail to learn the process. It learned the half of the process someone had already managed to write down, and we mistook that half for the whole.”

4. Why capable organisations walk off the edge

If the confusion were merely intellectual it would be correctable with a good briefing. It persists because a set of structural forces reward it, and those forces act on exactly the organisations experienced enough to know better.

- The economics of the demo. A working demonstration is cheap, dazzling, and fundable. Understanding is slow, invisible, and almost impossible to put in a business case — you cannot show a steering committee a photograph of comprehension. Capital flows to the thing that can be shown, and the thing that can be shown is the automation, not the understanding beneath it.
- The incentive asymmetry. Automating a process is a visible achievement with a name attached to it and a saving booked against it. Preserving the organisation's understanding of that process is a non-event — it prevents a future problem that, if prevention works, no one will ever see. We reward the visible act and take the invisible capability for granted, until it is gone.
- The flight from tacit knowledge. Holding deep process knowledge is expensive: it means senior people, redundancy in roles, slack in the system. On every efficiency dashboard that expense looks exactly like waste, because from the dashboard essential understanding and mere inefficiency are indistinguishable. So it is optimised away first, and most confidently.
- The retirement inversion. The people who understand a legacy process are often the ones nearest the exit — and automation is frequently justified as a way to de-risk their departure, to capture what they know before they go. This inverts reality. Encoding the explicit fraction of what a departing expert knows, and then releasing them, does not capture their knowledge; it accelerates the loss of the tacit remainder that was the valuable part.
- The precedent we filed under the wrong heading. The previous decade's wave of robotic process automation taught this exact lesson at scale — armies of bots built on top of undocumented processes, breaking the moment the underlying system beneath them shifted, abandoned not because the technology failed but because no one understood the automation well enough to change it in step with the process. We filed that experience under old technology rather than under recurring pattern, and so we are living it again with better tools and the same blind spot.

The pattern that recurs is not a failure of intelligence. Each individual decision — release this role, remove this review step, decline to fund this year of shadow-running — is locally reasonable and defensible on its own terms. The cliff is what their sum produces: a collective-action problem wearing the costume of efficiency.

5. The strongest case for automating first

An essay that only prosecutes its own thesis is a rigged trial, so let me put the most serious objection as forcefully as its advocates would, because it is not weak.

The objection is that I am describing an obsolete kind of automation. Rule-encoding automation, the argument runs, genuinely does lose the tacit half — but the whole promise of learning systems is that they do not need the process written down at all. Show them enough examples of good outcomes and they infer the process, including the parts the humans could never articulate, precisely because they learn from behaviour rather than from rules. On this view, demanding comprehension before automation is a category error and a reactionary brake: the machine can absorb exactly the tacit knowledge that codification destroys, and can often surface regularities the experts themselves never consciously knew they were using. Insisting that humans first understand a process the machine could simply learn is to privilege a slower, worse instrument out of professional sentiment.

There is real truth in this, and any honest treatment has to grant it. Learning from examples does capture tacit structure that no rulebook ever would, and in stable, high-volume, well-instrumented domains it genuinely outperforms the codifiers. The objection is not a strawman. It is the strongest thing that can be said against the whole argument.

But it proves less than it promises, for two reasons. The first: learning from examples captures the tacit knowledge that is present in the data, and is structurally blind to the tacit knowledge that governs when the data itself stops being representative. That second kind — the assessor's sense that this season's claims are unlike any season the training set ever saw — is exactly the knowledge the cliff depends on, and exactly the knowledge a system trained on the old distribution cannot hold, because it is knowledge about the distribution rather than within it.

The second reason is subtler and more uncomfortable. A system that has genuinely absorbed tacit knowledge you could not articulate is, by construction, a system whose reasoning you also cannot articulate. The comprehension problem has not been solved; it has been relocated — out of the process and into the model. You have replaced a process you did not fully understand with an automation you understand even less. For a stable, low-stakes, high-volume task, that may be an excellent trade. For a process that must be defended to a regulator, explained to a court, or changed under pressure when the world moves, it is a terrible one — because you have bought performance at the price of the one thing the situation will eventually demand, which is the ability to say why.

6. What understanding is, and how it leaves

Step back from the mechanics and the deeper subject comes into view: what organisational understanding actually is, and why it erodes so quietly that no one decides to lose it.

Understanding, in the sense that matters here, is not documentation. A process map is not understanding; it is a residue of it. Understanding is a living capability — the capacity, distributed across people, to answer a question that has never been asked before. What happens if we get twice the volume of this odd new claim type? is not answerable from the rulebook. It is answerable only by someone who holds the process as a model in their head and can run it forward against a novel input. That capability is invisible precisely because it only shows itself when something unprecedented arrives, which by definition

is rarely, and always inconveniently.

Because it is invisible, it is dismantled without a decision ever being taken to dismantle it. No one stands up and says let us make ourselves unable to think about our own claims process. Instead a hundred reasonable local choices — this role is redundant now, this review step is pure overhead, this senior hire is a luxury in the current climate — accumulate into a company that can run its processes but can no longer reason about them. The erosion is not a betrayal by anyone. It is the ordinary working of an organisation optimising against the things it can see.

I want to hold a tension here rather than resolve it too neatly, because the argument is easy to misread as a plea to hoard people and romanticise the veteran. It is not. Genuine waste is everywhere. A great deal of what passes for precious tacit knowledge is merely undocumented habit — arbitrary, defensive, occasionally obstructive — and automation rightly exposes it and improves on it. The difficulty, the whole difficulty, is that from the vantage point of the dashboard, essential understanding and mere inefficiency look identical. Telling them apart requires judgement about which is which — and that judgement is itself one more piece of tacit understanding that does not survive being automated.

7. Keeping the understanding while taking the labour

Where this leads is not caution for its own sake, which would be its own kind of failure. Automation is not the enemy of the essay; amputation is. The task is to build a guardrail deliberate enough to let an organisation automate hard and keep its capacity to think.

That begins with making understanding an explicit deliverable rather than an assumed by-product. Before automating a process, an organisation should be able to state plainly what it will still be able to do the day after go-live — not merely operate the system, but reason about it, retrain people into it, and rebuild it if it fails. If the honest answer is nothing but operate it, the business case is incomplete, however good the saving looks.

It means keeping a small number of people genuinely fluent in the process — not as a hedge against the technology breaking, but as the organisation's standing capacity to change its own mind when the world moves. And it means treating the ability to turn the automation off and run the process by hand — degraded, slow, but alive — as a capability worth paying for, in the same spirit and for the same reasons that any serious operation pays for resilience it hopes never to use.

The organisations that will come through this wave well are not the ones that automate the most, nor the anxious ones that automate the least. They are the ones that automate without amputating — that let the machines take the labour while keeping, deliberately and at some real cost, the understanding. The cliff is not the price of ambition. It is the price of mistaking the map for the territory, one entirely reasonable decision at a time.