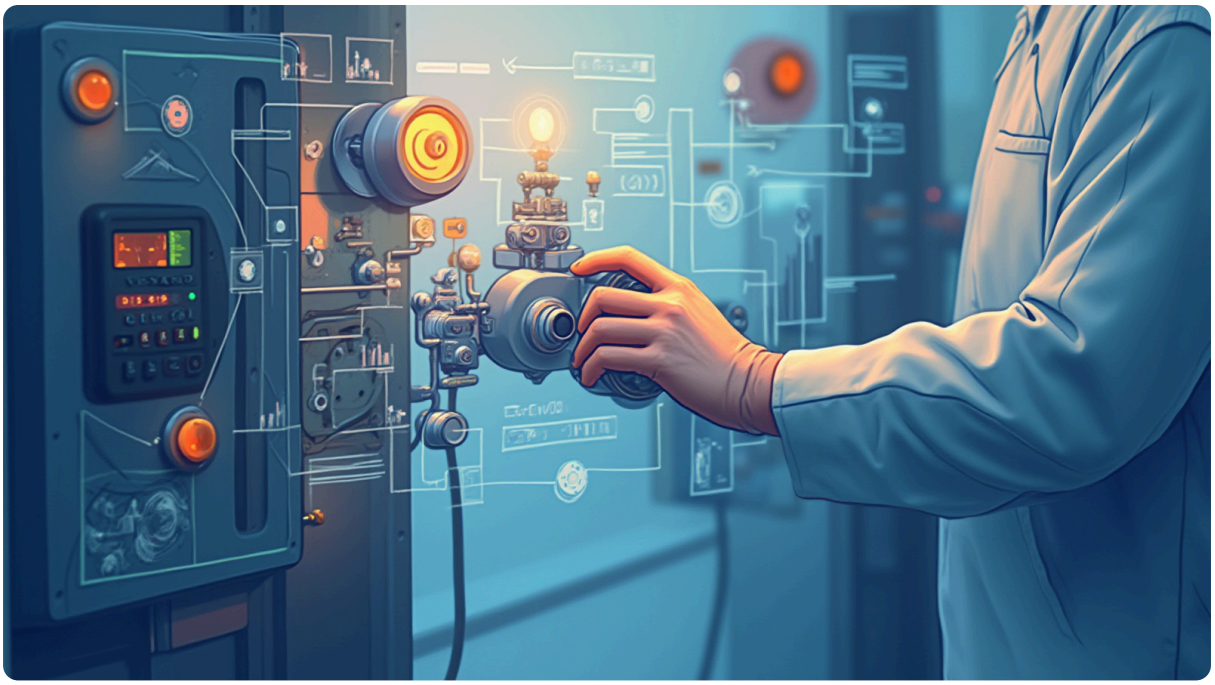
OPERATIONAL RESILIENCE NEWSLETTER

When AI Logic Fails: Building Immunity into Your Operations

Managing the new physical risks introduced by 'black box' software.

AI Risk Management · Operational Redundancy · Process Control · Systemic Failure

> The newsletter for operations leaders who prefer robust fallback protocols over relying on perfect technology.



The Diagnosis: When Code Breaks Things

I’ve spent enough time on shop floors to know what a predictable failure feels like. When a bearing seizes, it makes a noise you can hear from the breakroom; when a belt frays, you see the fraying before the line stops. These are physical failures. They follow the laws of physics: wear and tear lead to degradation, and degradation eventually leads to a stop. You know where to look for the problem because the problem is tangible.

We are entering an era where many manufacturing "failures" are no longer physical—they are logical. When you integrate AI or complex automated logic into your production flow, the point of failure shifts from a piece of metal grinding down to a line of code drifting away from reality.

This shift creates what I call **The Black Box Blindspot**. In this scenario, the machine isn't breaking in a way that alerts the operator; it is simply making "wrong" decisions based on "correct" data within its own limited logic. The system doesn't stop because of a mechanical jam; it continues to run while

producing out-of-spec parts because the underlying model has drifted from the physical requirements of the floor.

A broken motor is an honest failure. A drifting algorithm is a silent one. One tells you exactly where it hurts; the other hides the wound until you reach the customer's doorstep with a shipment that doesn't meet spec. We have to move our focus from just maintaining the machines to auditing the logic that governs them.

Why Relying on AI is Not Enough: The Fallacy of Perfect Data

There is a common misconception in modern manufacturing that if you have "perfect" data, you have perfect process control. This is not true.

Data collection is not the same thing as process validation. You can have a high-resolution sensor feed and an advanced AI model, but if those tools aren't anchored to physical reality, they are just very expensive ways to automate a mistake. We often see teams fall into the trap of trusting the dashboard over the product. They look at the green lights on the screen while the quality of the output is slowly eroding because the "logic" doesn't account for real-world variables like humidity shifts, tool wear, or raw material inconsistencies.

To be clear: **A data point is not a guarantee of quality.**

The difference between a successful implementation and a failed one often comes down to whether the team believes that just because an algorithm *can* make a decision doesn't mean it *should* be allowed to do so without human-validated checks. We must distinguish between "automated execution" and "unsupervised autonomy." You want your machines to be fast, but you need your humans to remain the ultimate arbiters of quality.

The Common Rationalization	The Reality on the Floor
"The AI is monitoring the tolerances in real-time."	The AI sees what it's programmed to see; it doesn't know when a sensor starts drifting or a material changes.
"We have enough data to predict failures."	Data shows you where things *did* go wrong; validation ensures they won't go wrong again today.
"The system is automated, so it’s consistent."	A flawed logic applied consistently results in perfectly produced waste.

The Cost of the Invisible Failure

When a machine breaks physically, the cost is immediate: downtime, lost parts, and an urgent maintenance call. When the logic fails, the costs are often deferred—and those are the ones that kill your margins over time.

The most dangerous consequence isn't just the scrap heap; it’s the **Erosion of Trust**. When a system produces "bad" results but the dashboard says everything is "good," the operators on the floor start to

lose faith in the tools you've given them. They stop trusting the sensors, they start making their own unofficial workarounds, and they begin to ignore the very systems meant to help them.

Furthermore, there is a massive loss of institutional knowledge when we let the "black box" take over entirely. If an operator isn't required to understand *why* a machine makes a certain adjustment, they lose the ability to spot the nuances of the craft. When the algorithm fails, you're left with a workforce that doesn't know how to intervene because they weren't part of the loop; they were just observers of the screen.

The cost of an invisible failure includes:

1. **Delayed Detection:** You don't find out there's a problem until it hits QC or, worse, the customer.
2. **Compromised Quality Integrity:** The "drift" happens slowly enough that it may pass initial checks but eventually pushes the product into a marginal zone.
3. **Cultural Decay:** Operators stop looking at the work and start watching the data, leading to a disconnect between the person and the process.

Building Operational Immunity: The Multi-Layered Buffer Approach

We don't need to be anti-technology; we need to be pro-resilience. To combat logic drift, you must build **Operational Immunity**. This means creating layers of "friction" where necessary to ensure that a failure in the software layer cannot bypass your quality standards.

Think of this as building a series of firebreaks. If one area catches fire (a logic error), the others stop it from consuming the whole plant. You do this through three specific buffers:

1. Physical Guardrails

Never allow a digital decision to override a physical constraint. If a machine's output must stay within a certain tolerance, there should be a hard-coded mechanical limit or a physical "stop" that no software update can bypass. If the AI wants to push a parameter beyond what is safe for the tool life or product integrity, the system must physically prevent it from doing so.

2. Human-in-the-Loop Gateways

Identify the high-risk transitions in your process—points where material changes, tools are swapped, or batch sizes shift. At these points, do not allow "auto-resume." Require a human operator to verify the state of the machine and sign off on the transition. This forces a manual check at the exact moments when logic is most likely to drift.

3. Cross-Model Validation

Don't rely on one algorithm to be both your "pilot" and your "navigator." Use two different methods to measure the same outcome. If an AI model predicts the furnace temperature, have a secondary, simpler,

rule-based system (a simple "if/then" logic) running in the background. If the results of the two systems diverge by more than a set percentage, trigger an automatic alert.

Practical Steps to Harden Your Process Today

You don't need to overhaul your entire IT infrastructure this afternoon. You can start building immunity on the floor tomorrow with these three actions:

1. Define "No-Go" Zones. Sit down with your engineers and identify the top five parameters that, if they drift even slightly, result in a non-conforming product. These are your "hard limits." Ensure these are controlled by simple logic or physical stops, not complex, evolving algorithms.

2. Audit the Exceptions. Review your last three months of "near misses" and scrap reports. Look specifically for instances where the machine was operating within its programmed parameters but produced a bad part. These are your primary targets for manual overrides. If you can't explain *why* it happened using only the data on the screen, that’s where you need to insert a human check.

3. Establish an "Audit of Logic" Schedule. Instead of just doing a standard maintenance check on the machine parts (the gears and belts), schedule a monthly review of the software's logic. Compare what the AI *thought* was happening against what actually happened during production. If there is a gap, it’s time to tighten your guardrails.

The rule for the floor is simple: The more complex the "brain" of the machine becomes, the simpler and more robust the safety net must be. We don't trust the black box; we trust the system that catches the black box when it fails.

Call to Action

What is your plant's documented reaction plan for a sudden software outage? Share this newsletter with another operations leader who needs an honest diagnosis.

Newsletter replies: newsletter@sixsigmakaizen.com · X: @kaizen_6sigma

Stay curious, stay improving,

David Rodgers
SixSigmaKaizen.com

References: When Your AI Logic Fails: Managing the New Physical Dependency