

FIELD WHITEPAPER · DEPLOYMENT STRATEGY

Dirty Data Is a Rational Answer

The economics of broken records: why WMS cutovers and AI break on the dock, and how to deploy with human incentives instead of against them

---

Marcos Stipicich

Deployment Strategist · 13 years in supply chain and operations

Cover: Caspar David Friedrich, The Sea of Ice, 1824

The argument in one page

A better algorithm cannot fix a record that is distorted where it is written.

A warehouse record is written by a person, at a specific moment, at a specific cost. When accuracy costs that person time now, and an error costs someone else money later, the shortcut is not a lapse. It is the rational answer.

So I start a deployment from the point of capture: who types the record, what accuracy costs them, and what happens if it is wrong.

1. **The record breaks where the work happens.** In one retailer's audit of nearly 370,000 inventory records, 65% did not match the shelf, and the error had a structure.
2. **The break travels because the record is trusted.** After a cutover, rules act on the record faster than the floor can contradict it.
3. **Software controls one lever.** It cannot appoint the authority that keeps records true. It can make the correct path faster, and refuse irreversible writes without a named person's approval.

The second half describes the architecture I build for that; the paper closes with its limits.



PART I

When the record leads

Each man holds on to the one in front. The first is already in the ditch.

After a cutover, every decision downstream holds on to the record in front of it.

Pieter Bruegel the Elder, The Parable of the Blind, 1568.

Two weeks of iron by air

I was on the floor for the go-live of a Tier-1 enterprise WMS at the parts distribution centre of a global agricultural-machinery manufacturer, a business present in more than 140 countries, with more than 50,000 SKUs under one roof.

Under the old system, air freight was the exception: urgent orders, charged to the customer. After cutover, the new rules began sending standard replenishment by air, in volume, at the company's cost.

What flew was cast iron: structural parts and dense spare components for tractors and implements. By eye, the freight on each shipment was well above the part's gross margin and, on several, above its sale price. Cost-to-serve is physical: weight and freight mode decided the margin. You did not need a model to see it. You needed to lift the box.

When I reported it, the instruction that came back was to stop questioning the software and execute what it assigned. The new system, the reasoning went, already had it calculated.

It went on for about two weeks. It stopped when the consolidated invoices from the cargo carriers reached senior management. Executives came to the site to reverse the policy, and air freight went back to being an exception approved by hand.

What actually failed

The easy reading is that the software was wrong. The rule was wrong, for two weeks, and a wrong rule at go-live is what hypercare exists to catch. What made this one expensive was that the fastest detector in the building, a person packing cast iron for an air shipment, had no channel to stop it.

In 1983, Lisanne Bainbridge described the structure of this problem in industrial process control. Designers automate the routine tasks on the premise that people are unreliable, and leave the operator only the anomalies they could not automate. In doing so, they remove the daily practice the operator needs to handle exactly those anomalies.

My reading is that a cutover adds an organisational version of that irony: in one night, it moves authority from the person who can see the shelf to the record that cannot. “The system already calculated it” is not a technical statement. It is a decision about who is allowed to be right.

In that operation, the correction did not come from the floor. It came from accounts payable, two weeks late, as an invoice.



PART II

Who pays for the record

Every record starts as a gesture: in heat or in cold, by tired hands, in seconds nobody budgeted.

The accuracy of a record depends on what capture costs the person who does it.

Adolf Menzel, The Iron Rolling Mill, 1875.

Dirty data has a structure

It is tempting to blame bad inventory records on people, or dismiss them as random noise. A large audit points somewhere else.

Nicole DeHoratius and Ananth Raman studied nearly 370,000 inventory records from 37 stores of one retailer, taken from the chain's physical audits. In 65% of them, the recorded quantity did not match what was in the store. The error was not noise. It had a shape.

It grew with an item's unit sales and fell with its cost. It was higher in stores with denser inventory and more variety, and for items routed through the retailer's own DCs rather than shipped by the vendor. Audit practices reduced it. And 26.4% of the variance sat between product categories, against 2.7% between stores.

My reading: if carelessness drove it, the error would follow the stores, where people change. It followed the work: how often an item is touched and how hard it is to count. Random noise averages out; systematic distortion does not.

The scope stays as the study set it: one retailer, 37 stores, nothing about ERP projects.

At -25 °C, the time standard is blind

I also worked the docks and storage halls of a cold-storage distribution centre owned by one of the world's largest animal-protein producers, with chambers at -18 °C and -25 °C. When the shift was short or the flow backed up, I drove conventional and high-reach forklifts and loaded trucks.

Crossing the air curtain into the -25 °C chamber, the doorway mist fogged my glasses, and for something like 20 to 40 seconds I drove without seeing. Near the doors and in the frozen aisles, frost formed a film on the concrete: a high-reach truck's polyurethane wheels would lose traction, slide and stop. The fix was physical. Climb down, find another driver, tow the machine free with a sling.

None of that exists in the time standard. The WMS assumes a constant travel speed from the first metre, and it has no event for an unplanned tow, so the minutes land where it can put them: idle time, or low productivity for the driver.

Nobody falsified anything. The record wrote down the only version of events it had a field for, and every labour standard and performance conversation built on it inherited that version.

The arithmetic of the shortcut

The cold chamber is physics. The more common case is economics, and it fits in one line.

Let s be what the person at the point of capture saves by skipping or bending a transaction: minutes at the dock, a truck that leaves on time, an adjustment that makes the month balance. Let p be the probability that it is detected, and F the penalty if it is. The shortcut is individually rational when:

$$s > p \cdot F$$

This is the risk-neutral form of the economic model of crime that Gary Becker published in 1968. In a typical ERP, p is the share of transactions an audit samples, far below one, and F is close to zero: the later correction, the inventory adjustment, costs nothing to the person who makes it or to their evaluation. With F near zero, any saving wins. The shortcut is not a moral deviation. It is the equilibrium.

Who does the work, who gets the benefit

Steven Kerr named the management side of this in 1975: the folly of rewarding A while hoping for B. In a warehouse it reads like this: pay for units moved per hour, hope for location accuracy, and the operator delivers what is paid and pushes the cost of accuracy onto the next shift. Donald Campbell had stated the general form in 1976: the more an indicator is used to make decisions, the more pressure there is to corrupt it, and the more it distorts the process it was meant to monitor. Economists know its cousin as Goodhart's law.

In 1988, Jonathan Grudin found the same shape while studying why collaborative software failed inside organisations: the people who had to do the extra work a system required were often not the people who got its benefit.

I watched the small version on a picking line for machinery and automotive filters. Dented units kept turning up in the pick faces. Reporting one meant stopping, calling the supervisor, filling in a non-conformity and walking the unit to quarantine: 10 to 15 minutes, against the pick rate each picker was evaluated on. The dented units shipped. Reporting cost the picker their own number; not reporting cost the customer and the returns desk, later.

When exceptions outrun the team

The saving *s* also grows with the queue. Every shift brings exceptions the software cannot resolve alone: a pallet with no label, a count that does not match the advance notice, a unit of measure the item master does not know. When they arrive faster than the team resolves them, the queue never settles, and working around the system becomes the only way to keep the dock open.

The waiting climbs well before that point. In the heavy-traffic approximation John Kingman published in 1961, waiting time grows with load divided by spare capacity:

LOAD ON THE EXCEPTION DESK	50%	80%	90%	95%
Waiting time, relative to 50% load	1×	4×	9×	19×

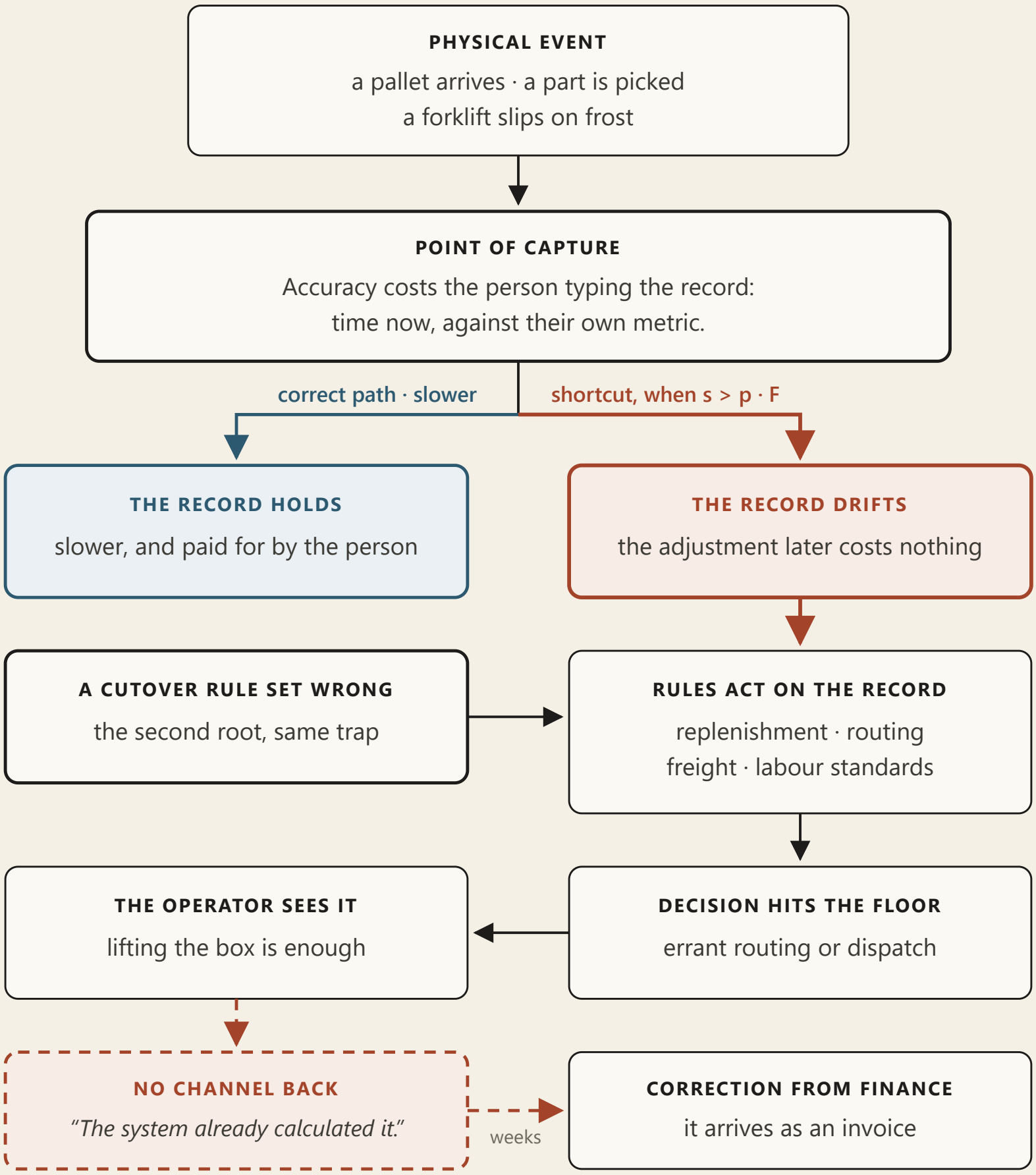
Single-server congestion factor ρ/(1−ρ), holding arrival and service variability constant. Adding staff lowers absolute wait times through pooling, but the nonlinear queue explosion near saturation remains. Model arithmetic, not a measurement from a specific dock.

The design consequence is uncomfortable for business cases built on headcount: cut the exception team after “automating”, without cutting the rate of exceptions, and the shortcut stops being a choice.

What the plan assumes, and what the dock does at 02:00

WHAT THE BOARDROOM MODEL ASSUMES	WHAT HAPPENS ON THE DOCK AT 02:00
The record matches the shelf.	In one retailer's audit, 65% of nearly 370,000 records did not, and the error followed product categories more than stores.
Capture is free; a scan takes a second.	Reporting one dented unit cost a picker 10 to 15 minutes against the pick rate they were evaluated on. The unit shipped.
Travel speed is constant from the first metre.	Entering a -25 °C chamber, I drove for 20 to 40 seconds with fogged glasses.
Time outside a transaction is idle time.	A high-reach truck that slips on frost needs a second driver and a tow. The system books the minutes as idle.
If the system routed it, it was calculated.	After go-live, standard replenishment of cast-iron parts flew for about two weeks, until the freight invoices arrived.
Exceptions are rare, and someone resolves them.	When exceptions arrive faster than the desk clears them, the queue never settles and the workaround keeps the dock open.
A later adjustment fixes the record.	The adjustment costs the person who makes it nothing, which is exactly why the shortcut wins.
Side spreadsheets are resistance to change.	In Huber, Zimmermann, Rentrop and Felden's multiple-case study, 64% of shadow systems were wired directly to the ERP to make up for its rigidity.

How a bad record travels



The trap is identical whether the root was capture arithmetic (s > p · F) or a blind cutover rule: downstream models trust the record, the floor has no channel back, and the correction arrives weeks late from finance.

What changes the result

Exhortation will not move arithmetic. An awareness campaign changes, at best, how people perceive p; not F, and not the physics behind s. Technical training that speeds up exceptions works once the correct path beats the shortcut.

Only three interventions change the sign of the inequality:

1. **Raise p to one, at the border.** Validate at the point of capture, not in an audit months later.
2. **Make F fall on someone with a name.** An executive's override carries that executive's signature.
3. **Make s negative.** Make the correct path the faster one.

The first two need coercive power, which inside a company sits with the board. The third does not: it is a software problem, and the one I build for.

s also includes what the shortcut pays. On a gemba walk, I saw a per-operator production KPI, tied to a bonus, producing mixed, short and damaged orders. I removed it and replaced it with customer satisfaction, measured as zero claims and response time. No metric escapes Campbell; this one starts with the customer, not the floor.



PART III

An architecture that survives the floor

An irreversible write needs a hand on the valve: once the air is out, there is no letting it back in.

Nothing irreversible reaches the system of record without a named person’s approval; everything else keeps a path back.

Joseph Wright of Derby, An Experiment on a Bird in the Air Pump, 1768.

Why big-bang cutovers break live operations

A cutover replaces two things in one night: the record, and the rules that act on it. The old system's flaws were known, and the floor had built compensations around them: the supervisor's spreadsheet, the person who knows which locations are never right. When Huber, Zimmermann, Rentrop and Felden studied shadow systems in organisations running a central ERP, 64% were directly connected to it, living off its data to make up for its rigidity.

On go-live night those compensations are switched off, and the new rules act with full authority and no history. If the business case also cut the exception team, the desk loses capacity just as the new rules produce their first exceptions.

My reading, not a statistic: the first weeks run on faith, and faith mutes the channel that catches errors fastest, the person on the floor. So I deploy in two levels. The first touches nothing. The second writes, but never directly, never irreversibly without a person, and never without a way back.

> **Software status.** The two levels below are built and tested locally; not yet deployed at a client.

Level 0: the forensic audit

Level 0 runs on files the client already has: exports of stock on hand, order lines and receipts, and a physical count when one exists. No network access, no installation, no credentials, no floor time, no meeting with IT security, and nothing goes back.

What I built computes, deterministically and the same way every time:

- **Record accuracy against a count:** variances within tolerance bands, the accuracy rate, their dollar value, and an ABC cycle-count plan.
- **Service actually delivered,** measured from the client's own order lines.
- **Queue and staffing math** for docks and exception desks, variability included.
- **Organisational-friction flags:** an official number that disagrees with an independent one; a target raised right after it was met; a control point with no owner; an informal channel nobody has classified.
- **Cost-to-serve** by customer, channel or SKU, from activity drivers.

What Level 0 refuses to do

What it refuses to do matters as much as what it computes.

- **No number without a basis.** A count sheet without unit costs does not report a zero-dollar impact. An unmeasurable figure is reported as unmeasured, with the reason, and a report that states a number it had no basis for fails verification.
- **No number flattered by a gap.** An order line missing a field is left out of that measure's denominator, and how many were left out is published beside it, so a shrinking denominator stays visible. A lower-bound estimate is labelled as a lower bound.
- **No verdicts on people.** Each friction pattern needs its own minimum of independent evidence before it fires, and even the strongest finding is reported as a question for a person to confirm.

The output is a report in units, or in dollars only where the files carry costs, plus a short list of questions only the floor can answer. It is also the go or no-go for Level 1: if the record cannot be reconciled, nothing should be written on top of it. The gate works per item and location, not for the whole warehouse: Level 1 starts where the count reconciles.

Level 1: a safe-staging layer outside the accounting core

Level 1 is where software proposes changes to a client's system of record: a reorder point, a price, a purchase order. Whether from a rule, an optimizer, or an AI agent, it never writes directly. The write path has five steps:

1. **Stage.** Every proposal becomes a dry-run changeset showing the before and after value of each field. Nothing is written.
2. **Classify.** Each changeset gets a risk tier: read-only, reversible or irreversible.
3. **Approve.** Anything the policy does not allow to apply on its own needs an unexpired approval bound to the exact field-level changes: approving one change cannot authorise another. If the approval is missing, mismatched or expired, the write is refused: the layer fails closed, not open.
4. **Apply once.** Writes are idempotent: the same change, retried, never lands twice.
5. **Audit, and roll back.** Every applied changeset leaves an audit entry. Reversible changes roll back from it; anything that leaves the building, such as an order a supplier has received, cannot, which is why it needs a person's approval first.

Around the write path

For a buyer, the aim is a correct path that is faster than the shadow spreadsheet: the third lever from Part II. That is a design goal, not yet a measured result.

In the one ERP where I built write-back to purchasing, an open-source mid-market system, purchase orders land inside the client's system as drafts and wait there for a person to confirm them. The write is idempotent, and a batch that fails halfway is undone.

Two rules sit around that path. Autonomy is earned, in three tiers: execute and audit, one-click approval, or escalate to a person. A type of decision is promoted only after consecutive cycles in which its backtested precision clears the bar, and only when a person approves the promotion; demotion is automatic and instant. And state is appended, never overwritten: each cycle's snapshot is validated against a schema before it is allowed to land.

Where a decision is genuinely combinatorial, such as which facilities should serve which demand points, I solve it exactly and offline, as a mixed-integer program with the open-source HiGHS solver. I would not connect an exact solver to a record that Level 0 has not reconciled.

Where AI fits, and where it never writes

The cover promises enterprise AI, so here is the line. A language model helps where input is unstructured and a person checks the output; it has no place where the system keeps state.

AI CAN PROPOSE	ONLY DETERMINISTIC CODE
Reading unstructured input: a carrier's email, a driver's note ("pallet tipped at dock 4")	Stock balances and accounting state: added and subtracted, never predicted
Classifying an exception and routing it to whoever can resolve it	Checking each proposal against its risk tier and policy
Drafting the buyer's summary of an order whose quantities code computed	Writing to the system of record: staged, approved, applied once, audited
Explaining a WMS error code, or a proposed change, in plain words	Combinatorial decisions, such as which facility serves which demand: an exact MILP

What the model reads from outside is data, never an instruction. It fills a fixed set of fields, cannot change quantities or recipients, and the person approving sees field-level differences, not the model's prose.

> AI belongs in interpretation and drafting; determinism belongs in state, arithmetic and writes. When an LLM writes directly to the ledger, you have not deployed an agent. You have deployed an uncontrolled mutation engine inside your accounting core.

One click to cancel, one click to save

I have also been on the other side, as a customer. I ordered paper filters and accessories for a coffee maker from an online specialist. Thirty days later nothing had arrived, and no alert had come either. When I asked, the company said one item was out of stock, cancelled the whole order and refunded it. Nobody offered a partial shipment, a substitute or a date.

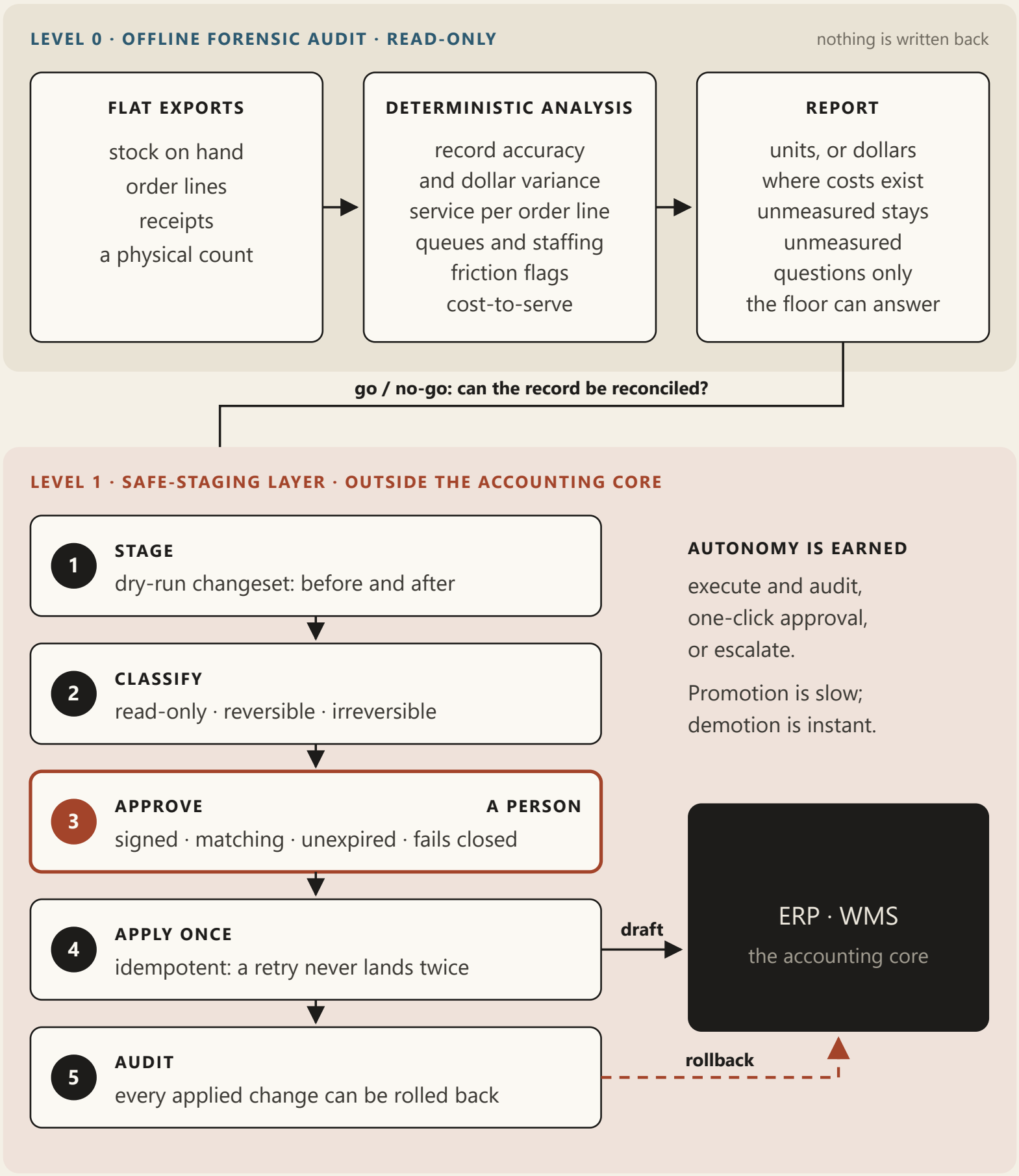
From the outside, the arithmetic is easy to guess. Cancel-and-refund is one action that closes the ticket; saving the order takes several. The books show the refund as zero. What it consumed is not on that line: support time, acquisition cost, often the payment fees, and a repeat customer.

This is where I would put AI:

1. Deterministic code checks what is physically available, and reads the restock date from open purchase orders.
2. An AI agent drafts the partial-shipment proposal and the message to the customer from those facts.
3. A person approves it in one click: as fast as cancelling.

> **Design, not a deployed feature:** the one-click approval exists in my write path; this flow does not.

Two levels: read first, then write



Level 0 writes nothing, and its result decides whether Level 1 starts. Level 1 never writes to the system of record directly: it stages, classifies, waits for a signed approval, applies once as a draft, and keeps the trail that makes rollback possible.

Who uses it first, and how it should feel

Grudin's observation also decides who the first user should be. Software that asks a floor operator to do extra work for someone else's benefit starts with the arithmetic against it. The planner or buyer is a different case: the person who records the decision also benefits when the recommendation is right. That is where I start. The floor operator, where misalignment is largest, comes later.

The second rule comes from Paul Adler and Bryan Borys, who in 1996 separated coercive formalization, which treats the operator as the source of error, from enabling formalization, which helps people diagnose and resolve the exception where it happens. Their four dimensions make a good test for any screen at a point of capture:

- **Repair:** can the person fix it in place?
- **Internal transparency:** can they see why the system did what it did?
- **Global transparency:** can they see how their step affects the rest?
- **Flexibility:** can they depart from the script when the floor requires it?

What these cases do not prove

- **They are one observer's cases.** I chose them because they went wrong, which is exactly how selection bias works. They illustrate a mechanism; they do not measure how often it happens.
- **The 65% belongs to one retailer.** Thirty-seven stores, one chain, and a study that does not discuss ERP. Its value is the structure of the error, not the headline number.
- **The inequality is a model.** It predicts the direction of behaviour when its terms change, not the size of the effect in a given warehouse. The queue figures are the model's too.
- **Enterprise systems do not fail on average.** My reading: for forty years each wave, from ISO 9000 to ERP, SaaS and AI, has promised to tame the floor with a better abstraction. What matters is not the average outcome but the tail. Hendricks, Singhal and Stratman found modest profitability gains after ERP implementations in listed companies. The failure percentages that circulate in slide decks have no known denominator: nobody counts the projects that never made the news. The risk worth designing for is the tail, the deployment that does not merely underperform but takes the operation down with it.

When not to deploy

- **When the problem is authority, not software.** In Chile, electronic invoicing was introduced in 2003 and made mandatory in stages between 2014 and 2018; each document is validated by the tax authority when it is issued, and one that fails validation does not exist as a tax document. Inside a company, the only equivalent of that authority is the board or a regulator. A software vendor cannot take its place; when it tries, it produces the illusion of control.
- **When exceptions already outrun the team.** Fix capacity first. New software adds load before it removes any.
- **When there is nothing to reconcile against.** Without a physical count, dated order lines and a cost column, there is no baseline. Results stay in units; dollar figures would be invented.
- **When nobody owns the cost.** If a bad record's cost lands in a generic account with no name on it, nobody will defend the correct path when it is slower. Name the owner first.
- **When an exact solver would sit on an unreconciled record.** An optimizer fed a wrong record produces a precise wrong answer, faster.

One question for the floor

Before the next cutover, trade one steering-committee slide for a question asked on the floor at 02:00: when an operator sees the system do something absurd, who can stop it, how long does that take, and what does it cost the operator to say so?

If the honest answer is “nobody, two weeks, and their next evaluation”, the deployment has already chosen its correction channel.

It is accounts payable.

About the author

Marcos Stipicich
Deployment Strategist · Supply Chain & Operations

I grew up on a sheep ranch, where my father taught me pragmatism early. I read across disciplines, though theory only comes alive for me on the floor, never behind an office desk. When I walk into an operation, I go in like a beginner—dropping the expert posture so people drop their guard and show how the shift actually runs. And when the crew is short a hand, I get on the forklift or the high-reach myself, pick the lines, and load the truck.

Over thirteen years across Chile, Argentina, and Australia, working and leading on the floor took me from -25 °C cold storage and a USD 15M three-site logistics budget to owning a USD 30M cross-border e-commerce P&L (releasing ~USD 5M in inventory) and supervising floor operations through a 50,000-SKU Tier-1 WMS cutover at a global agricultural-machinery parts centre in Melbourne, where order processing time fell 20%.

Grounded early in C and C++, I architect the systems and direct AI coding agents to ship the Python, TypeScript, and Postgres—owning the math, schema contracts, MILP models, and rollback gates under the hood.

Adapted from my manuscript, The Deployment Chasm.

Available for pre-cutover forensic audits, deployment strategy, and Forward Deployed roles.

linkedin.com/in/marcos-stipicich · github.com/esstipi-debug · marcos.stipicic@gmail.com

Sources

- Adler, P. S., and B. Borys. 1996. "Two types of bureaucracy: Enabling and coercive." *Administrative Science Quarterly* 41(1): 61–89. doi.org/10.2307/2393986
- Bainbridge, L. 1983. "Ironies of automation." *Automatica* 19(6): 775–779. doi.org/10.1016/0005-1098(83)90046-8
- Becker, G. S. 1968. "Crime and punishment: An economic approach." *Journal of Political Economy* 76(2): 169–217. doi.org/10.1086/259394
- Campbell, D. T. 1979 (first circulated 1976). "Assessing the impact of planned social change." *Evaluation and Program Planning* 2(1): 67–90. doi.org/10.1016/0149-7189(79)90048-X
- DeHoratius, N., and A. Raman. 2008. "Inventory record inaccuracy: An empirical analysis." *Management Science* 54(4): 627–641. doi.org/10.1287/mnsc.1070.0789
- Goodhart, C. A. E. 1975. "Problems of monetary management: The U.K. experience." *Papers in Monetary Economics*, Vol. 1. Reserve Bank of Australia.
- Grudin, J. 1988. "Why CSCW applications fail: Problems in the design and evaluation of organizational interfaces." *Proceedings of the 1988 ACM Conference on Computer-Supported Cooperative Work*: 85–93.
- Hendricks, K. B., V. R. Singhal, and J. K. Stratman. 2007. "The impact of enterprise systems on corporate performance: A study of ERP, SCM, and CRM system implementations." *Journal of Operations Management* 25(1): 65–82.
- Huber, M., S. Zimmermann, C. Rentrop, and C. Felden. 2016. "The relation of shadow systems and ERP systems: Insights from a multiple-case study." *Systems* 4(1): 11. doi.org/10.3390/systems4010011
- Kerr, S. 1975. "On the folly of rewarding A, while hoping for B." *Academy of Management Journal* 18(4): 769–783. doi.org/10.5465/255378
- Kingman, J. F. C. 1961. "The single server queue in heavy traffic." *Mathematical Proceedings of the Cambridge Philosophical Society* 57(4): 902–904. doi.org/10.1017/S0305004100036094
- Servicio de Impuestos Internos (Chile). Exempt Resolution No. 45, 2003. Republic of Chile, Law No. 20,727, 2014.

Images. Public-domain paintings: Caspar David Friedrich, *The Sea of Ice*, 1824; Pieter Bruegel the Elder, *The Parable of the Blind*, 1568; Adolf Menzel, *The Iron Rolling Mill*, 1875; Joseph Wright of Derby, *An Experiment on a Bird in the Air Pump*, 1768.

Field cases. The three scenes are my own observations, written down after the fact. Their figures are estimates from memory, not measurements. Employers are described, not named.