EPF-00 · DOCTRINE · SEPTEMBER 2026

Organizational Drift

Why Companies Fail Slowly

Enterprise Process Framework · Opening doctrine paper · Own the Work, Fund the Risk

The claim

Companies rarely die from one bad system, one missed patch, or one loud outage. They die from drift: years of disparate systems and disparate processes pulling away from the way the company claims to work. No single decision looks fatal. The accumulation is.

A firm lasts when work, system, cost, risk, baseline, and time have the same owner. When those six split, the organization grows a second, unofficial operating model. That unofficial model eventually becomes the business.

1. Failure is usually not an event

The stories we tell about corporate failure prefer a climax. A breach. A product that ships late. A plant that stops. A quarter that misses. Those moments are real, and they are almost never where the company was actually lost.

What looks like a sudden failure is usually the first time drift became visible to people who were not living inside it. The tool that "we have always used." The process that exists only in one manager's head. The exception that was granted for ninety days and is now entering its seventh year. The platform a division demanded, never funded, and still does not own. The security control everyone agrees with and nobody is measured on.

Drift is the gap between the operating model on the slide and the operating model in the building. It starts small because small is how work gets done. A team cannot wait. A vendor is already in the door. A leader does not want to be the person who said no. Each of those choices can be defended in isolation. Together they write a second company on top of the first one.

Lasting is not a brand attribute. It is a process property. Firms that last are not the ones with the newest stack. They are the ones that refuse to let unowned work, unowned systems, and unowned risk become the default way of operating.

2. What organizational drift is

Organizational drift is what happens when systems, processes, and incentives are allowed to diverge from the way the company claims to work.

It is not chaos. Chaos would be easier to see. Drift is locally rational. Every workaround has a sponsor. Every extra tool has a user who will defend it. Every skipped control has a deadline that felt more important that afternoon. The organization is not failing to decide. It is deciding, constantly, in fragments, without a rule that keeps those fragments aligned.

Disparate systems

A company does not drift because it has many systems. It drifts because those systems do not share an owner, a baseline, or a lifecycle. One division buys a platform. Another keeps a spreadsheet that quietly becomes the system of record. A third asks an AI how the work "should" be done and arrives at IT with a finished architecture and a demand to implement it. None of these objects is inherently reckless. What is reckless is that nobody is required to carry the cost and the residual risk of keeping them.

Disparate processes

The same pattern shows up in how work actually moves. The documented process is the one written for the auditor. The real process is the one that gets a customer, a hire, an invoice, or a change out the door. When those two stay apart long enough, people stop treating the documented process as real. New employees learn the hallway version. Leaders keep funding the slide version. The distance between them is drift.

Why IT feels it first

Technology is where drift concentrates because technology is where every other function leaves its residue. HR's new hire needs a stack. Sales wants a tool before the quarter ends. Maintenance will not leave the plant system that still "works." Finance wants the lights kept on and the line item cut. Security wants the estate closed. Everybody is correct about their local problem. IT becomes the place where those local problems collide, and then IT is blamed for the collision.

That is why this doctrine series begins with a diagnosis rather than a tool catalog. If you do not name drift, you will keep treating symptoms—more tickets, more staff, more outsourcing, more "alignment meetings"—and the unofficial company will keep growing.

3. How drift compounds

Drift follows a reliable sequence. It does not require villains.

1. Demand arrives without an owner. A division wants an outcome. The request is framed as a technology problem, so it is handed to IT. Ownership of the outcome does not travel with the request.
2. Cost is centralized and then resented. Licenses, endpoints, projects, and residual support sit in an IT expense line. The people who created the demand cannot see what they consume. Leadership sees only an overhead number that looks available to cut.
3. Risk is orphaned. The system is "IT's." The behavior is the user's. The exception is the executive's. When something fails, the only mailbox that looks official is IT's.
4. The exception becomes the estate. A one-off tool, a frozen platform, an unsupported device standard—each is easier to keep than to retire, because retirement has a bill and a political cost. Time has no policy, so yesterday's workaround is tomorrow's architecture.
5. The unofficial model wins. People stop asking what the company standard is. They ask who can get them what they need this week. At that point the firm still has an org chart. It no longer has a single way of working.

This sequence is why "keep the lights on" is not a strategy. Keeping the lights on, while playing whack-a-mole with vulnerabilities, technical debt, and incoming demand, is what a drifted organization looks like from the inside of IT. The threat environment has only made the cost of that posture more obvious. Weaknesses are weaponized in

hours. The human failure mode—an urgent call to action that violates policy—has not changed in decades. Drift makes both worse, because nobody is clearly responsible for closing either.

4. What lasting companies refuse to split

The opposite of drift is not rigidity. It is alignment of ownership. Six things must have the same owner or the unofficial company starts writing itself:

Must stay together	If they split
Work and owner	Outcomes become tickets. Nobody is accountable for the result, only for the activity.
System and owner	IT inherits objects it did not choose and cannot retire.
Cost and demand	Consumption is invisible. IT looks expensive. Demand stays unbounded.
Risk and beneficiary	The people who wanted the tool do not sign what it can do to the firm.
Process and baseline	Autonomy becomes a second enterprise. Integration becomes folklore.
Time and policy	Exceptions age into architecture. Technical and process debt become the estate.

These six alignments are the Enterprise Process Framework. They are not an IT standard. They are a rule for how a company remains one company. Technology is simply the first place the rule is easy to see, because technology is where every function leaves a bill.

5. False fixes

Drift attracts remedies that rearrange the furniture and leave the unofficial company intact.

More central control

IT demands ownership of everything technical. That feels like order. It produces a department that is overworked, underfunded, and universally blamed. Control without funded ownership is just concentrated liability.

More local freedom

Every division is told to “be agile” and choose its own tools. That feels like speed. It produces five estates, five identity models, and a security team that discovers the real architecture after an incident.

Outsourcing the residue

The cost line shrinks. Headcount and tax load move off the books. The provider brings a method that is not yours. Influence over quality drops. Drift does not leave. It changes letterhead.

Another transformation program

A new operating model is announced. The slide is clean. The hallway process does not change, because nobody moved cost, risk, or lifecycle with the announcement. Transformation without ownership is a rebrand of the same split.

The useful move is narrower and harder: put work, system, cost, risk, baseline, and time back in the same hands. Let a professional function manage the estate. Stop asking that function to own the appetite.

6. What this paper is for — and what comes next

EPF-00 exists to name the problem so the rest of the work is not a pile of IT opinions. If you accept the diagnosis, three commitments follow.

- Stop treating every failure as an event. Ask which split of ownership made the event cheap to produce and expensive to see.
- Stop funding demand you cannot see. If a division wants the outcome, it funds the stack, the project, and the residual risk.
- Stop confusing management with ownership. The people who run the estate are not required to be the people who wanted the object.

The next paper, EPF-01, states the rule in full: Own the Work, Fund the Risk. It turns the six alignments into an operating constitution.

The first applied series, IT After Ownership, shows the rule inside technology: the internal company model, funded demand, the controls catalog, owned technical debt, a finite baseline, accountability when something breaks, and why none of that is an outsourcing argument.

None of those papers will matter if the organization still believes it failed on a Tuesday. It didn't. It drifted on a hundred Tuesdays, and then one of them was loud.

7. A definition you can reuse

> Organizational drift is what happens when systems, processes, and incentives are allowed to diverge from the way the company claims to work. No single decision looks fatal. The accumulation is.
>
> Companies do not fail from one bad system. They fail from a thousand unowned ones.
>
> Own the work. Fund the risk. Keep the six together. That is how a company lasts.

— End of EPF-00 —

Next in the doctrine pair: EPF-01, Own the Work, Fund the Risk. First applied paper: IT After Ownership T00, From Cost Center to Company.