

— CASE STUDIES

Five teams. One question: which copy of the runbook works?

Short case studies from manufacturing, insurance, utilities, logistics and higher education — each showing what changed once scattered scripts became one library of automations, scored on whether they work.

5 case studies

5 sectors

3 automation types

— INSIDE THIS BOOKLET

01

Manufacturing

One patch runner for every plant

02

Insurance

No more expired-certificate outages

03

Energy & Utilities

Unauthorised config change, reverted

04

Logistics

Terraform changes with a real approval step

05

Higher Education

Backups proven, not assumed

— WHAT EVERY TEAM BUILT ON

3

AUTOMATION TYPES

10

BLOCK FAMILIES

1-click

RUN ON ANY SITE

18

PLATFORMS TARGETED

None of these teams needed more automation. They needed fewer copies of it, and a way to tell which one works.

Build once, run anywhere — one library of workflows, agentic workflows and agents for every site you run.

— CONTENTS

What these five have in common

Every team here already automated plenty. What they lacked was one version of each procedure, the right machinery for each job, and a number that said what to trust. In four of the five cases, the starting point was a prebuilt automation — forked or run as-is.

- 01 Manufacturing**
Six plants, two clouds, one patch procedure. **p03**
- 02 Insurance**
The certificate that took down the claims portal was the last one. **p04**
- 03 Energy & Utilities**
Unauthorised network changes now get reverted before the next shift. **p05**
- 04 Logistics & Transport**
Infrastructure changes stopped going straight from laptop to production. **p06**
- 05 Higher Education**
A small IT team finally knew its backups would actually restore. **p07**

— THE RECURRING PATTERN

Forking beat building.

Four of five teams started from a prebuilt automation and changed a step or two.

The site held the detail.

Moving hostnames and windows out of the procedure ended per-site copies.

One branch settled the debate.

A confidence threshold or approval let cautious and impatient people both say yes.

The score retired the old scripts.

A few weeks of success rates made deleting duplicates an easy call.

— AT A GLANCE

SECTOR	STARTING POINT	TYPE	PRIMARY DRIVER
Manufacturing	Forked Patch Window Runner	Workflow	Per-site script drift
Insurance	Prebuilt Cert Renewal · Prod	Agentic workflow	Expired-cert outages
Energy & Utilities	Prebuilt Config Drift Rollback	Workflow, gated rollback	Undetected config drift
Logistics	Forked Terraform Plan & Apply	Workflow with approval	Unreviewed infra change
Higher Education	Prebuilt Restore Drill + Report	Workflow	Unproven backups

01 — MANUFACTURING

Six plants, two clouds, one patch procedure.

SECTOR
ManufacturingSTART POINT
Forked Patch Window
RunnerTYPE
WorkflowDRIVER
Per-site script drift

The situation

- Six plant scripts plus two cloud variants, each maintained by whoever was on site.
- A fix made at one plant took months to reach the others, if it ever did.
- Nobody could say which version had the fewest failed windows last quarter.
- A new plant acquisition meant writing a ninth version from scratch.

How it was built

- **Fork.** Started from the prebuilt runner instead of reconciling eight local scripts.
- **Tune.** Added one step: confirm the line is idle before draining any plant host.
- **Point.** Same definition targeted at six plants and two cloud environments.
- **Track.** Success rate watched per site for a month before retiring the old scripts.

What changed

- One definition across every site
- A fix lands everywhere at once
- A success rate on the one procedure
- New plants inherit it on day one
- The same checks, every window

"We didn't have a patching problem. We had eight patching opinions. Now we have one procedure, and the site just tells it when the window is."

— HEAD OF INFRASTRUCTURE, PLANT SYSTEMS

— CAPABILITIES DOING THE WORK

Prebuilt library

Forking

Deterministic workflow

Cross-platform targets

Schedule trigger

Success-rate tracking

Org sharing

One-click run

— A PATCH SCRIPT PER PLANT

The manufacturer ran plant-floor systems on VMware at six sites and its business apps in Azure. Every site had its own patch script, and every script had its own quirks.

Workflow Designer turned eight local patch scripts into one procedure that every plant — and every cloud — runs the same way.

02 — INSURANCE

The certificate that took down the claims portal was the last one.

SECTOR
InsuranceSTART POINT
Prebuilt Cert RenewalTYPE
Agentic workflowDRIVER
Expired-cert outages

The situation

- Expiry dates lived in a spreadsheet that was accurate the day it was made.
- Renewal was manual and spread across three teams with different tooling.
- A claims-portal outage from an expired cert had already reached the press once.
- Nobody wanted fully unattended renewal on customer-facing services.

How it was built

- **Run.** Started with the prebuilt automation as-is on a non-production site.
- **Threshold.** Set a confidence level below which it escalates instead of renewing.
- **Gate.** Customer-facing certificates always require approval, whatever the confidence.
- **Report.** Every run posts what changed to the platform channel and the ticket.

What changed

- Expiring certs found daily, 60 days ahead
- Routine renewals handled end to end
- Customer-facing renewals approved with evidence
- A confidence branch instead of a debate
- A record of every change, posted automatically

"We didn't want a robot renewing the claims portal at 2am on a guess. We wanted it to handle the easy ninety percent and put the tricky ones in front of us with the evidence."

— DIRECTOR OF INFRASTRUCTURE SECURITY

— CAPABILITIES DOING THE WORK

Agentic workflow

Confidence branching

Approval flow control

Schedule trigger

Prebuilt library

Output to Slack & tickets

Success-rate tracking

Cross-platform

— EXPIRED, AGAIN

Every few months, a certificate nobody was tracking expired and something customer-facing stopped working. The fix took ten minutes. Finding out what had broken took the rest of the morning.

Workflow Designer replaced a spreadsheet and a calendar reminder with an automation that renews what it is sure of and asks about the rest.

03 — ENERGY & UTILITIES

Unauthorised network changes now get reverted before the next shift.

SECTOR

Energy & utilities

START POINT

Prebuilt Config Drift
Rollback

TYPE

Workflow

DRIVER

Undetected config drift

The situation

- Drift from the golden baseline was found during quarterly audits, weeks after it happened.
- Comparing running configs by hand took an engineer most of a day per region.
- Some drift was legitimate emergency work that never got written up.
- Regulators wanted evidence that unauthorised change was detected and handled.

How it was built

- **Run.** Pointed the prebuilt workflow at one region first, report-only.
- **Baseline.** Loaded the golden configs the team already maintained per device role.
- **Gate.** Rollback on control-centre devices always goes to the network lead.
- **Widen.** Extended to every region once a month of reports matched expectations.

What changed

- Drift found the next morning
- Every device diffed daily, automatically
- Legitimate changes captured and written up
- A diff report attached to every ticket
- One workflow across both vendors

"Most drift was someone fixing something at 3am and forgetting to tell us. Now we find it the next morning, not the next audit."

— NETWORK OPERATIONS MANAGER

— CAPABILITIES DOING THE WORK

Deterministic workflow

Config diff & rollback

Approval flow control

Schedule trigger

Cross-platform (Cisco, Juniper)

Ticket outputs

Prebuilt library

Success-rate tracking

— THE CHANGE NOBODY LOGGED

On an operational network for a utility, an undocumented config change isn't an annoyance — it's a risk. The team knew their golden baseline. They just couldn't check every device against it every day.

Workflow Designer moved a utility from finding config drift at the quarterly audit to reverting it before the next shift.

04 — LOGISTICS & TRANSPORT

Infrastructure changes stopped going straight from laptop to production.

SECTOR
LogisticsSTART POINT
Forked Terraform Plan &
ApplyTYPE
WorkflowDRIVER
Unreviewed infra change

The situation

- Terraform ran from engineers' machines with their own credentials.
- Plans were sometimes pasted into chat for review, sometimes not.
- Different engineers used slightly different wrapper scripts for the same job.
- Peak season was coming, and a bad apply during it would stop deliveries.

How it was built

- **Fork.** Started from the prebuilt plan-and-apply workflow, not the team's wrappers.
- **Tune.** Added a cost estimate to the posted plan so reviewers saw spend impact too.
- **Gate.** Apply waits for approval from someone other than the requester.
- **Retire.** Deleted the old wrapper scripts once every change went through the library.

What changed

- Every apply through one workflow
- Diff reviewed before anything changes
- One definition; wrappers retired
- Runs on the target site, not a laptop
- A clear record of who approved what

"Nobody lost any speed. The plan still takes a minute. The only difference is that someone else sees it before it goes out — which is what we always said we did."

— PLATFORM ENGINEERING LEAD

— CAPABILITIES DOING THE WORK

Terraform & scripts blocks

Approval flow control

Forking

Manual / direct trigger

Cross-platform (AWS, GCP)

Outputs to Slack & Jira

One-click run

Org sharing

— TERRAFORM APPLY, FROM SOMEONE'S LAPTOP

The logistics company's platform team was quick and trusted. That was part of the problem: infrastructure changes went out from individual laptops, with the review happening afterwards, if at all.

Workflow Designer gave a fast-moving platform team a review step that costs a minute and closes the gap before peak season found it.

05 — HIGHER EDUCATION

A small IT team finally knew its backups would actually restore.

SECTOR
Higher educationSTART POINT
Two prebuilt workflowsTYPE
WorkflowDRIVER
Unproven backups

The situation

- Restore tests were planned every term and quietly skipped when term got busy.
- Incident summaries for leadership were assembled by hand each Monday morning.
- Research groups ran their own storage, with their own ideas about retention.
- The team had no time to build automation from scratch.

How it was built

- **Run.** Both workflows run as-is from the library — no forking needed at first.
- **Point.** The restore drill targets central storage first, then research groups who opt in.
- **Report.** Recoverability gaps land in a weekly report with the affected system named.
- **Summarise.** Leadership gets yesterday's incidents by email each morning, formatted.

What changed

- A restore drill every week, automatically
- Recoverability gaps named and fixed
- Leadership summary sent every morning
- Research groups opting in to drills
- Two automations running within a week

"We're nine people. We were never going to write a restore-test framework. Running two things from a library was something we could actually do in a week."

— HEAD OF INFRASTRUCTURE SERVICES

— CAPABILITIES DOING THE WORK

Prebuilt library

Deterministic workflows

Schedule triggers

Report outputs

Cross-platform

One-click run

Success-rate tracking

Category filters

— BACKUPS NOBODY HAD RESTORED

The university backed everything up every night. What nobody could say with confidence was whether any of it would restore — because restoring took a day nobody had.

Workflow Designer let a nine-person university IT team prove its backups every week — without building anything from scratch.

— THE PATTERN

Automation scales when there is one version of it.

Every team here had working automation before they started. What changed was that each procedure ended up in one place, pointed at every site, with a success rate everyone could see.

WHAT SCATTERED SCRIPTS ASSUME

- That the right copy is the one you found first.
- That a fix in one place reaches the others.
- That the author is still around to ask.
- That "it usually works" is good enough.

WHAT A LIBRARY PROVIDES INSTEAD

- One definition; the site decides where it lands.
- A fix made once, everywhere.
- A catalog anyone can search and run.
- A live success rate on every row.

— ACROSS ALL FIVE TEAMS

5

SECTORS COVERED

3

AUTOMATION TYPES USED

4 of 5

STARTED FROM PREBUILT

1-click

RUN ON ANY SITE

Customers anonymised at their request; profiles and quotations are representative of deployments in each sector, not records of named engagements. Success rates shown come from a sample library and vary by estate and configuration. Third-party names are trademarks of their owners. © 2026–2027 LumiOps.AI.

— IN THEIR WORDS

"We didn't have a patching problem. We had eight patching opinions. Now we have one procedure, and the..."

**HEAD OF
INFRASTRUCTURE,
PLANT SYSTEMS**

"We didn't want a robot renewing the claims portal at 2am on a guess. We wanted it to handle the easy..."

**DIRECTOR OF
INFRASTRUCTURE
SECURITY**

"Most drift was someone fixing something at 3am and forgetting to tell us. Now we find it the next..."

**NETWORK
OPERATIONS
MANAGER**

"Nobody lost any speed. The plan still takes a minute. The only difference is that someone else sees it..."

**PLATFORM
ENGINEERING LEAD**

"We're nine people. We were never going to write a restore-test framework. Running two things from a..."

**HEAD OF
INFRASTRUCTURE
SERVICES**

1

Several copies of one procedure existed

2

A prebuilt automation was forked or run

3

The site took over the local detail

4

The score retired the old copies

— IN CLOSING

You already wrote the automation. Keep one copy and run it everywhere.

Every story here started with procedures the team already had. The work was choosing one version, putting it in the library, and letting the numbers decide.

3

AUTOMATION TYPES

10

BLOCK FAMILIES

1-click

RUN ON ANY SITE

18

PLATFORMS TARGETED

— WHERE TO START

1**Count the copies**

Find the procedure that exists in the most places.

2**Search the library first**

Fork the closest prebuilt automation.

3**Let the score decide**

Run it for a week, then retire the old copies.

— WHERE YOUR FIRST AUTOMATION WILL COME FROM

Personal

Drafts and experiments. Build freely; promote to the org once it proves out.

Org

Your team's runbooks, wired to your systems and reviewed by your people.

UnityOne

Production-tested automations for common ops work. Fork and tune without touching the original.

Every team here changed where their automation lived, not how clever it was.

Stop rewriting the same runbook.

Fork a prebuilt automation, point it at your site, and run it today.

GET IN TOUCH

contact@lumiops.ai

RUN

1-click, any site

LINKEDIN

linkedin.com/lumiops