

— CUSTOMER STORY

Six plants, two clouds, one patch procedure.

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.

The situation — Patching as local folklore

Maintenance windows were tight — production lines don't wait. The procedure itself was the problem:

- 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.

The move — Fork the prebuilt runner, point it at each site

The team forked the prebuilt Patch Window Runner, added their line-safety check before draining any host, and pointed the same definition at all six plants and both Azure environments. Each site keeps its own maintenance window, and the old scripts stayed in place for a month as a fallback nobody ended up needing.

"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

— DEPLOYMENT SNAPSHOT

SECTOR

Discrete manufacturing

ESTATE

6 plants on VMware, 2 Azure environments

START POINT

Forked Patch Window Runner

TYPE

Workflow (deterministic)

TRIGGER

Schedule, site-local windows

CHANGE

Line-safety check added before drain

TRACKING

Success rate per site and overall

INTEGRATIONS

VMware, Azure, ServiceNow, Teams

— BUILT ON

3

AUTOMATION
TYPES

10

BLOCK FAMILIES

1-click

RUN ON ANY
SITE

- **Fork, don't start blank.** Most of the work already exists in the library — change what is different.
- **Pick the right machinery.** Fixed steps, reasoning inside rails, or a standing agent — matched to the job.

— 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

BEFORE WORKFLOW DESIGNER

- Eight scripts for one procedure
- Fixes stuck at the plant that made them
- No idea which version worked best
- New sites meant a new script
- Patch quality tied to who was on site

WITH WORKFLOW DESIGNER

- 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

— WHERE THE CHANGE CAME FROM

The site held the detail

Windows and hostnames moved into site config, so the procedure itself could stay identical.

Forking beat reconciling

Starting from a proven runner was faster than merging eight opinions.

The score retired the copies

A month of results made deleting the old scripts an easy decision.

— WHY THIS MATTERS IN MANUFACTURING

Prebuilt library

Forking

Deterministic workflow

Cross-platform targets

Schedule trigger

Success-rate tracking

Org sharing

One-click run

— BOTTOM LINE

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

3 AUTOMATION TYPES**10** BLOCK FAMILIES**1-click** RUN ON ANY SITE

Customer anonymised at their request; profile and quotation are representative of a deployment in this sector, not a named engagement. Outcomes are illustrative of the operating changes Workflow Designer enables, not measured benchmarks. © 2026–2027 LumiOps.AI.

**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