TECHFLOW INDUSTRIAL · 7 STEPS GUIDE

Build Your AI Data Layer

In 7 steps - from scattered PLC data to a factory that AI understands

> *"You're missing one layer - and AI can't fix what it can't understand."*

For PLC/SCADA engineers who want to make their factory AI-ready without vendor lock-in and without €40K in licences

Reading time: 15-20 min · Hands-on time step 1: 30 min

TECHFLOW24.COM

Build Your AI Data Layer

In 7 steps - from scattered PLC data to a factory that AI understands

> *"You're missing one layer - and AI can't fix what it can't understand."*

This guide is for you if you're a PLC/SCADA engineer or OT architect watching AI initiatives stall in your factory - and you want to know how to fix it. Not by paying €40,000/year for AVEVA Connect. Not by buying yet another historian. By building the missing layer yourself, in 7 steps.

Reading time: 15-20 min. Hands-on time for step 1: 30 min.

The problem in short

Your factory generates terabytes of data from PLCs, SCADA, historians and ERP - and yet you can't name three decisions taken this week based on that data.

That's **the missing layer**: not more dashboards or tools, but **the semantic layer that turns raw machine data into information that a human or AI can act on - within the time window where it still matters**.

This guide shows what that layer looks like. In 7 steps.

Before you start - the Solve test

The biggest mistake in industrial data projects: start with the data and hope action eventually follows. Do it the other way around. Start with **what physically needs to change on the factory floor?** - then build backwards.

We call that the Solve test. Before you build, write one concrete event in this template:

> *"I want to enable [WHO] to [DECIDE OR DO WHAT] within [HOW FAST] after [TRIGGER]."*

Example from BakeryWorks Utrecht (the fictional bakery we use as reference):

> *"I want to enable the shift supervisor to validate the oven temperature correction within 4 minutes after: tunnel oven zone 3 drops below 195°C for >30 sec while product is in zone 3."*

The 5 criteria to test every Solve event against:

1. **Closed loop** - does the action feed back into the system so you know afterwards what happened?
2. **Owner identified** - is there one specific person (or system) responsible?
3. **Time-bound** - does the action have a window where it still matters?
4. **Measurable outcome** - can you afterwards verify that something physically changed?
5. **Traceable to source** - can you trace the action back to the raw machine data?

If even one of the five is "no", you're building a data museum. Not a data layer.

> Important: *Solve is not an 8th step. Solve is the test you apply to the entire 7-step build. It's what makes the build complete.*

Good. With your Solve event on paper, here are the 7 steps.

Step 1 - Connect

Pull raw signals out of your PLC/SCADA/sensors - no interpretation, no modelling. Just: get the data out of its silo.

Open-source toolset: OPC-UA for modern PLCs, MQTT via **MonsterMQ** for the edge, Modbus TCP for legacy.

> **Trap:** *"let's connect everything just in case". Connect only what your Solve event needs. Three to five tags, not two hundred.*

Step 2 - Condition

Clean up raw data: deduplication, smoothing, time-align (UTC + ISO 8601), engineering unit conversion.

Raw PLC data is raw - a sensor sends every second, even when nothing changes. A DCS publishes in local time without DST correction. Conditioning is not the goal; it's a filter that decides what's worth carrying downstream.

> **Trap:** *"standardise everything". Conditioning is a filter, not a report.*

Step 3 - Model

This is the heart of the entire build. Raw tags get meaning when you place them in a **Unified Namespace (UNS)** - a hierarchical language the whole factory shares. Under the hood: ISA-95 as ontology (Enterprise → Site → Area → Work Center → Equipment → Tag).

A tag no longer reads `plc-oven-a/db100/dbw14 = 2400`. It reads:

```
bakery-works-utrecht/line-a/baking/tunnel-oven-01/zone-3/temperature = 240.0°C
```

A human understands it instantly. So does an AI agent. Without the Model step you still have bytes - just cleaner ones.

> **Trap:** *treating ISA-95 as a compliance document. It's a working language, not a report.*

Step 4 - Store

Storage: time-series for measurements + event store for Solve events.

MongoDB for flexible documents - works fine as both at once, one database. Retention policy per topic; don't keep everything for 7 years.

> Trap: *storing everything at raw PLC poll frequency. Nobody needs 1Hz data from 2018.*

Step 5 - Orchestrate

Event-driven workflows. A sensor changes → a trigger fires → an action (or a decision-for-human) is executed.

This is where data becomes **information**. Data-driven (collect everything, analyse later) is not the same as event-driven (define what should happen, collect what's needed for that). Event-driven beats data-driven because it builds Solve events, not reports.

Toolset: **N8N** for workflows. Not for data routing - that's MonsterMQ's job.

> **Trap:** *Node-RED. For production OT, N8N's flow versioning, error handling and UI are more mature.*

Step 6 - Visualize

Dashboards aligned per persona. Operator, shift supervisor, planner, plant manager - each their own view.

Grafana for time-series + drilldowns. One dashboard per persona, no "everything-in-one". Threshold overlays directly in the chart (Solve-trigger lines). Live events feed as a table at the bottom.

> **Trap:** *PowerBI as the visualisation layer in OT - too slow for real-time, too heavy for sector-engineer maintenance, too vendor-locked.*

A dashboard without a decision is wallpaper. One with a decision is a control room.

Step 7 - Distribute

APIs for downstream consumers - other apps, BI tools, AI agents, mobile apps for operators.

FastAPI for REST endpoints. WebSocket for live streaming. Webhooks for outgoing notifications. These endpoints are the **products** your data layer delivers. On top of them you build Teams bots, mobile apps, AI agents.

> Trap: *publishing every endpoint "for later". Publish only what a real consumer asks for - anything else is technical debt waiting to compound.*

Your data layer is only valuable if it's actually used. An endpoint without a consumer is dead code.

Solve - the test, revisited

You now have a working data layer on a fictional bakery. But - is it Solve? Walk through it:

CRITERION	BAKERYWORKS SOLVE-A	STATUS
Closed loop	Shift supervisor acks in Teams, response lands in MongoDB	✅
Owner identified	Duty shift supervisor pulled from ERP shift rota	✅
Time-bound	4 min response, then escalate to plant manager	✅
Measurable outcome	Batches saved per quarter × €batch value	✅
Traceable to source	Action log → MES → MQTT → PLC tag	✅

Five ticks. Working data layer.

> Remember: *a factory with perfect ISA-95 models and beautiful dashboards but no Solve events doesn't have a data layer. It has a* data museum*.*

What now?

Three ways forward:

OPTION 1 - BUILD IT YOURSELF IN 6 MONTHS

All code is open source. The stack runs on an €8/month VPS. You have years of OT experience - you can do this. What it takes: - 14 weeks full-time or 6 months part-time - No architecture review (you learn by failing) - No community of peer engineers on the same journey - Self-teaching the Python/Docker/REST API skills

OPTION 2 - JOIN THE LIVE WORKSHOP

60-min live online workshop. Next session: June 2026. Live demo of the IDP stack (Connect → Distribute), you fill in a 16-item AI-readiness scorecard on the spot, and you leave with a personalised gap analysis for your plant.

OPTION 3 - THE ONLINE PROGRAM - BUILD YOUR AI DATA LAYER

14 weeks, one module per week: - Weekly 60-min review call on YOUR architecture - Community of 20-25 PLC/SCADA engineers on the same journey - ISA-95 templates per sector (brewing, chemicals, metals, food, pharma) - 30+ N8N workflow templates - Skills-bridge module: Python, Docker, C#, REST, XML - in OT context - Capstone: one Solve event from start to finish on YOUR plant

Founding cohort price: €49/mo lifetime lock-in (first 25 spots). Then €97/mo.

Fill in the AI-readiness scorecard →

16 questions (5 min). You get an immediate personalised gap analysis of your skills (Python, Docker, MQTT, OPC-UA, vendor playbooks) and AI affinity. First step towards the online program.

Quick links

Open-source stack The working Docker stack (MonsterMQ + MongoDB + Grafana + Traefik) from this guide: github.com/cftservices/idp-os

Community Facebook group "Industrial Data Platform" - engineers building this together: facebook.com/groups/1282445730514883

Waitlist Reserve a spot in the founding cohort (€49/mo lock-in, first 25): techflow24.com/contact

About TechFlow

TechFlow Industrial builds **the missing AI data layer** for industrial factories - an open-source data architecture (MonsterMQ + MongoDB + Grafana + Docker) that connects PLC, SCADA, MES and ERP and turns raw machine data into AI-ready information. Built on 15 years of shop-floor experience in factories (Siemens PLC/SCADA, historians, DCS). **Build Your AI Data Layer** is the flagship program for PLC/SCADA engineers who want to step up to Industrial Data Architect.

techflow24.com

> *"You're missing one layer - and AI can't fix what it can't understand."*