PlantPulse® **Platform**

INDUSTRIAL DATA INFRASTRUCTURE

PRODUCT OVERVIEW & PROPOSAL

Scattered plant data, one infrastructure.

An enterprise IIoT data platform that delivers the whole data lifecycle — collection, messaging, storage, processing, analytics and visualisation — as **a single install package**. It replaces the SCADA, historian, gateway and database sprawl.



40,000/s
messages ingested

500,000/s
real-time event evaluations

24+
pre-integrated components

September 2026

Enterprise industrial data infrastructure · On-premises / air-gapped

kopens®

CONTENTS

Contents

01	Why now — the infrastructure problem	3
02	What PlantPulse Platform is	4
03	Seven layers, one package	5
04	Key strengths ①②③	6–8
05	The industrial data lake · **our engines**	9–15
06	Key strengths ④⑤⑥	16–18
07	Key strengths ⑦⑧	19–20
08	Architecture — data flow	21
09	How it is used (live screens)	22–23
10	Security · operations	24
11	Installation · delivery model	25
12	References	26
13	Deployment plan and schedule	27

One-page summary

> Plant data infrastructure is usually **five products stitched together**. PlantPulse Platform replaces that with **a single install**.

40,000/s	500,000/s	24+
message ingestion throughput	real-time event evaluation (CEP)	pre-integrated microservices

THE KEY DIFFERENCE

One product, not a toolbox. Instead of buying and wiring up a collector, a broker, a time-series database, a rule engine and a dashboard separately, seven layers install **already integrated**. Ontology, Unified Namespace, Flow and complex event processing work together on the same data foundation.

What this document covers

What it replaces

Pages 3–5. The problem of scattered infrastructure, product definition, layer composition.

What makes it different

Pages 6–20. Performance, standardisation, **the industrial data lake (pages 9–15)** · **three engines of our own (page 10)**, ISO KPIs, ontology, UNS, Flow, MCP.

How it is actually used

Pages 21–23. Data flow and screens captured from a running system.

What deployment looks like

Pages 24–27. Security and operations, installation, references, schedule.

01 · Background

Why now — the infrastructure problem

A plant rarely suffers from a lack of data. The problem is that the data sits **in five places, in five different shapes**.

PROBLEM 1

Five products, five contracts

Buying SCADA, historian, gateway, database and BI separately means five vendors, five licences, and five parties to blame when something fails.

PROBLEM 2

Gluing them together becomes the project

Every link between products turns into a development item. Change one and the integrations to the other four need reworking.

PROBLEM 3

The same machine looks different at every site

Mapping is done by hand, so structure drifts from line to line and plant to plant. Comparison, aggregation and AI training all stall there.

PROBLEM 4

Even building a KPI is work

To see one OEE figure you write the definition and the calculation from scratch. Definitions differ per site, so nothing adds up.

So what is needed

> One data foundation that **runs the moment it is installed**, on which standardisation, KPIs, automation and AI all **act on the same data**.

PlantPulse Platform is that foundation. More than 24 pre-integrated components install as one package, and the whole pipeline — collect → store → process → analyse → visualise — is running right after installation.

02 · Introduction

What PlantPulse Platform is

Industrial data infrastructure that runs as soon as it is installed. It receives data from edges and equipment, stores, standardises and analyses it, and makes it usable by the floor and by AI in the same context.

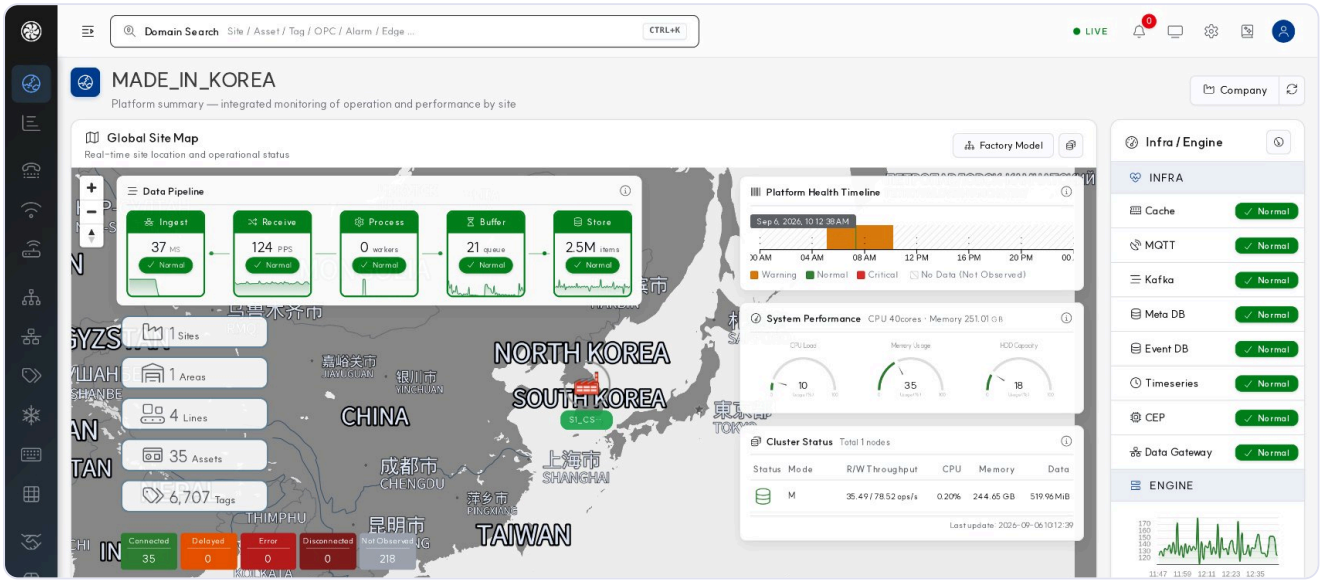


Figure 1. Integrated monitoring — site map, throughput and state of every pipeline stage, system performance, cluster state, infrastructure component health and key metrics on one screen. *Captured from a live instance.*

RECEIVE

Collection · messaging

Takes data from edges, equipment and external systems and puts it on the queue.

KEEP

Storage · standardisation

Stores it in a distributed time-series/event store and organises it into a standard asset model.

USE

Analytics · automation · openness

Builds KPIs, detects with rules, and opens the data to AI and external systems.

What it replaces

Previously bought separately	What the platform contains
SCADA · HMI server	Real-time monitoring screens and canvas dashboards
Historian	Distributed time-series/event store (horizontal scaling)
Message broker	MQTT · messaging broker · Sparkplug B built in
Rule engine · alarm server	Complex event processing (CEP) and advanced alarms
ETL · integration development	No-code Flow automation
BI · reporting	Automatic ISO-standard KPIs and analytics screens

03 · Composition

Seven layers, one package

The layers below are not products bought one by one but **components installed in a single step**. The full pipeline is running within minutes of installation.

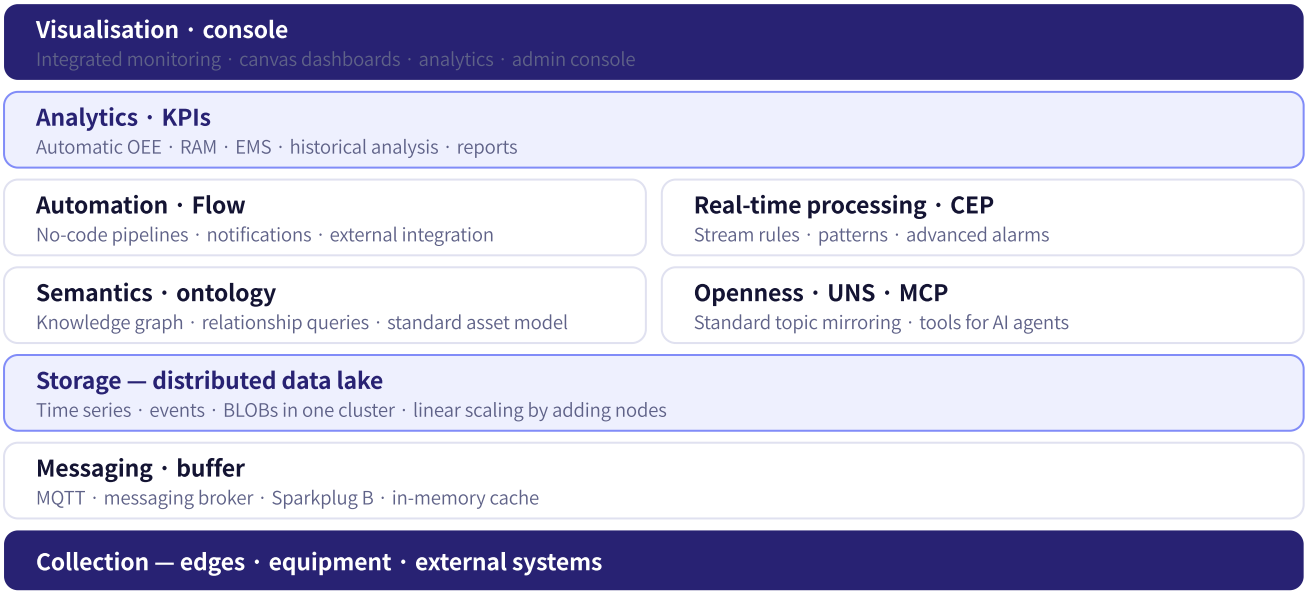


Figure 2. Seven layers. Data flows bottom-up, and the layers are not options to pick from but **one set that is installed together and verified together**.

> **What "pre-integrated" means.** Deploy the layers separately and version matrices, integration and performance tuning become a project every time. Pre-integrated means the vendor has already done that work, and data is actually flowing within minutes of install.

Running right after installation

Data flows

Attach an edge and collect → buffer → store → query works immediately. No storage tuning or schema design up front.

Screens exist

Integrated monitoring, asset explorer, alarms and the cluster console come as standard. No days spent building the first dashboard.

KPIs appear

Configure the asset model and OEE, RAM and EMS are computed to standard definitions at once.

It is open

UNS topics and AI tools are on from the start, so external integration is not a later phase.

All that is left is "fitting it to our plant" — the asset hierarchy, rules per equipment type, alarm thresholds, automation flows. You start by **reflecting the business**, not by standing up infrastructure.

04 · Key strengths

① It handles the incoming volume

Add equipment and data grows in steps, not in a line. PlantPulse Platform is designed for industrial scale in **both ingestion and real-time evaluation**.

40,000/s	500,000/s	Linear scaling
messages ingested on a single server	in-memory CEP event evaluations	add a node = add capacity

Real-time evaluation — complex event processing (CEP)

It does more than catch one tag crossing a threshold: it watches **patterns inside a time window**. Conditions such as “three or more vibration spikes within five minutes followed by a temperature rise” are defined in a dedicated query language and evaluated on the stream instantly.

- **Time-window rules** — judged by the trend, not by a single instant.
- **Asset-domain rules** — standard evaluation rules per equipment type are reused.
- **Roll-ups to aggregates, snapshots and predictions** — raw measurements become meaningful units right away.

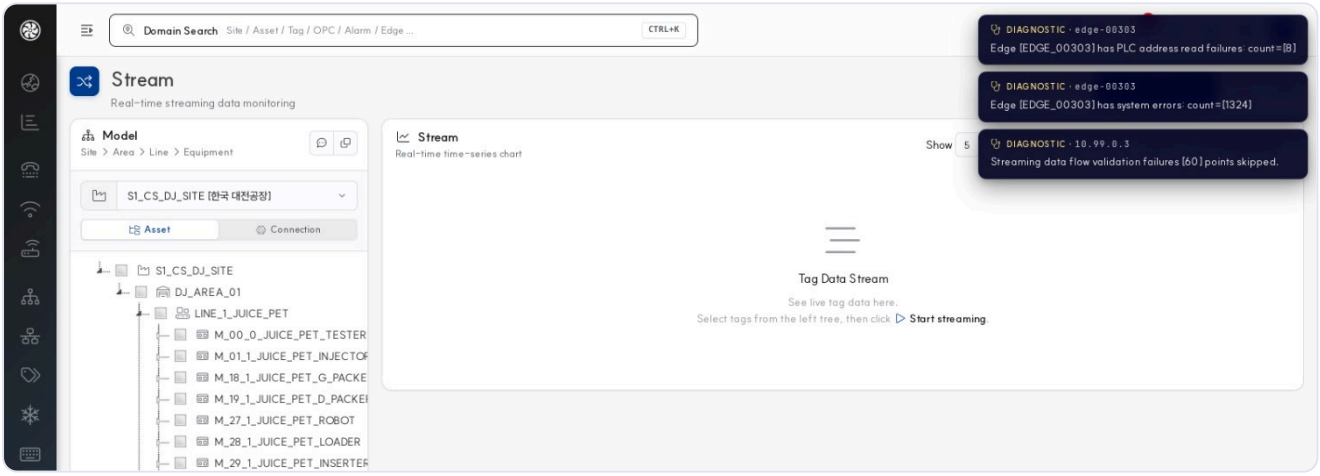


Figure 3. Real-time streaming monitor — data flowing along the asset hierarchy, checked on the spot.

WHY IT MATTERS

Fast ingestion alone backs up the store; big storage alone slows evaluation. **Ingestion, storage and evaluation designed together in one product** keep the bottleneck from piling up on one side.

04 · Key strengths

② Growing means plugging in a node

Industrial data never shrinks. Storage therefore has to be **expandable**, and expanding it must not become a migration project.

Distributed data lake

A **distributed event and time-series store** from the same family global hyperscalers use, embedded and tuned for industrial data. Data is distributed across nodes automatically and replicated, so losing a node loses no data.

- **Linear scaling by adding nodes** — storage capacity and throughput rise together.
- **Time series, events and BLOBs in one cluster** — drawings and images live in the same place.
- **State you can see** — ring topology, node load and data distribution in the console.

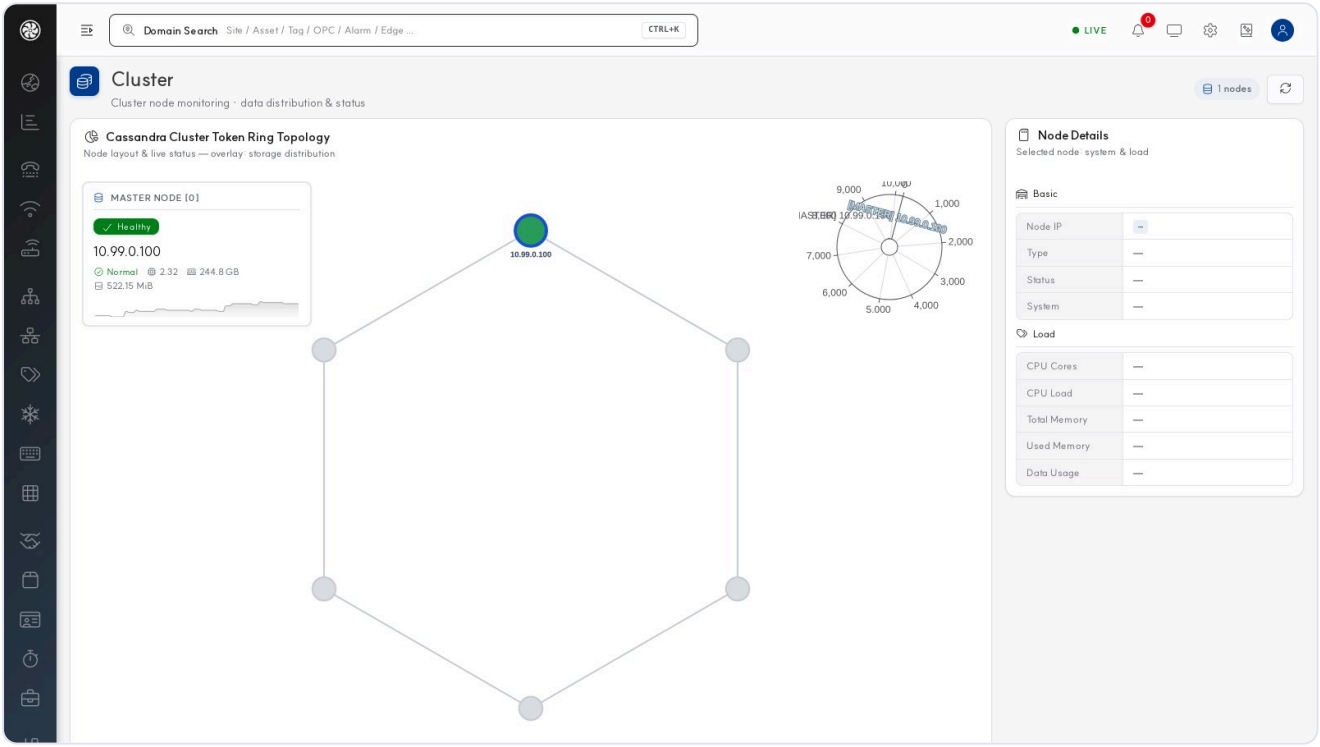


Figure 4. Cluster console — per-node state, throughput, resources, data distribution and ring topology in real time.

> **Scaling an operator can handle.** Distributed stores usually need specialists. This product puts the store inside the product and **makes it visible in the console**, so it can be run without a dedicated team. Structure, tiers and operations are covered in depth in **“The industrial data lake”, pages 9–15**.

04 · Key strengths

③ 100 new machines is not 100 mappings

The real reason data projects drag on is **mapping**. When a person decides the structure for each machine, work grows with the machine count.

Six data classes

Equipment output is split into six kinds, and **rules are defined once per equipment type**. Every new machine of the same type reuses those rules as-is.

Snapshot Current value	**Alarm** Anomaly raised	**Event** State change
Context Work order · conditions	**Aggregate** Period summary	**Command** Control history

- Mapped by **on-screen configuration alone** — no coding, no schema design.
- Stored in the same structure in a **standard asset hierarchy** (site → area → line → equipment → tag).
- Same type means **the same structure across lines and sites** — compared and aggregated immediately.
- Same structure means data is **ready for AI training** as it is.

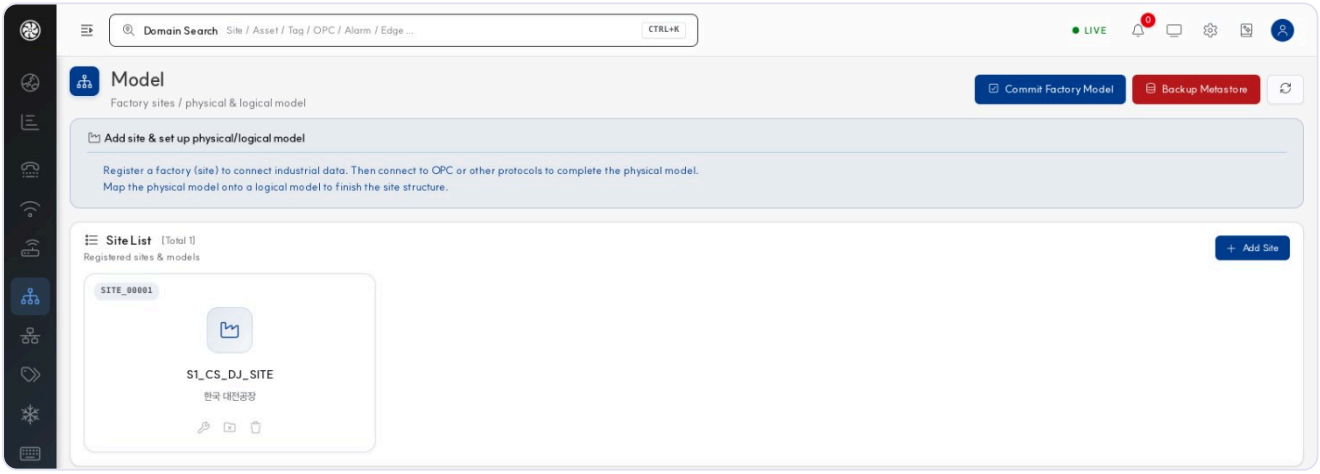


Figure 5. Factory model — the physical and logical structure of a site is defined on screen and changes are committed.

05 · Data lake

The industrial data lake — 16 components in one image

The floor of the platform is **PlantPulse DataLake**. Storage, messaging, time series, real-time processing, analytics, workflow, the query gateway and the admin console ship as **one container image** and come up in a fixed order with a single `pd start`. It is a complete data infrastructure assembled for industrial data.

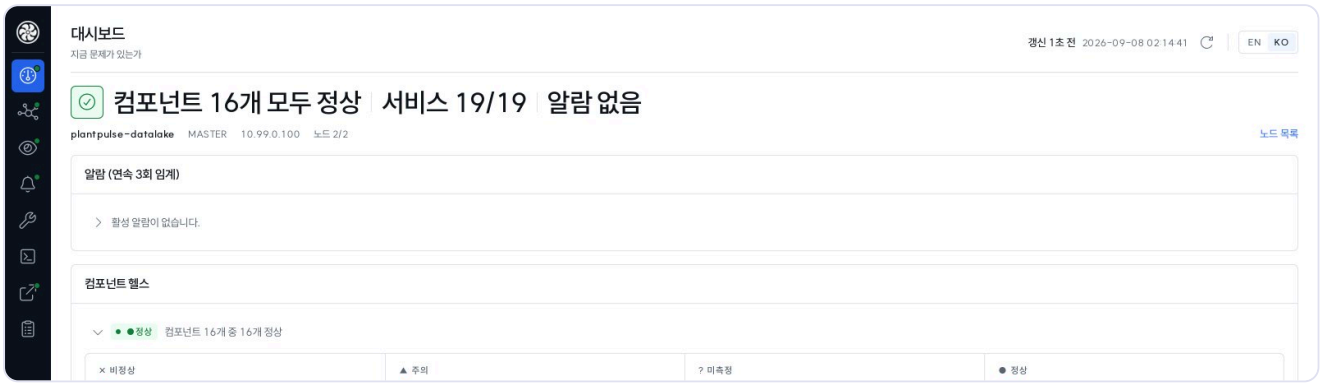


Figure 6. The data lake admin console's first screen answers “is anything wrong right now?” in one line — all 16 components healthy · services 19/19 · no alarms, with a 24-hour timeline per component below. *Captured from a live instance.*

What is inside

Layer	Components	Role
Storage	Aster TSDB (our own build on Cassandra 6.0) · PostgreSQL 18 · Valkey · RustFS	Time-series/event ring / master & metadata / in-memory cache & streams / S3 objects
Messaging	Apache Kafka 4.3 (KRaft) · **Aster MQTT** (our own broker)	Ingest buffer · pub/sub · Sparkplug B · broker auth and ACL
Time series	PlantPulse TSE (time-series engine) · Grafana	Roll-ups · 30+ aggregators · TQL queries · dashboards
Real time	Aster CEP (EQL) — our own engine	Time-window rules · patterns · advanced alarms
Analytics	Apache Spark 4.1 · Kyuubi · Hive · Hadoop · Gravitino	Iceberg lakehouse · SQL analytics · JDBC serving · catalog
Workflow	Temporal · Kestra · Warehouse worker	Daily archive · weekly optimisation · declarative pipelines
Openness	Data Gateway · SQL console	PQL · CQL · SQL · TQL behind one REST endpoint
Operations	Admin console · `pd` CLI · backup toolkit	Status · topology · storage · backup · upgrade

Why “one image”. Install these separately and version combinations, configuration and boot order drift from site to site. The data lake is baked as a verified combination; the image carries a **calendar version and build commit**, and its **signature and checksum** are verified before deployment. The same image is the same data lake at every site.

05 · DATA LAKE

Three engines of our own — Aster TSDB · Aster CEP · Aster MQTT

The heart of the data lake is **where data is stored**, **where it is evaluated** and **where it comes in**. None of the three is a third-party product bolted on: all are **engines KOPENS develops and maintains at source level**. What the plant floor needs goes into the next release, and defects are fixed here rather than waiting on a vendor roadmap.

3	7.1×	0
engines of our own — storage · evaluation · ingest	time-series storage reduction (measured)	separate streaming clusters

STORAGE

Aster TSDB — industrial time-series database

Our own build on Apache Cassandra 6.0. **CQL syntax and the on-disk format are unchanged**, so existing data reads as-is and the time-series features are enabled per table.

- **7.1× smaller storage** — 11.9 → 1.7 B/row (20M rows, measured)
- **21 server-side CQL time-series functions** — bucketing · gap-fill · regression · time-weighted average
- **Tiered storage** — older windows compress automatically and `SELECT` does not change
- **Verification** — 93 release-gate assertions · 49 three-node cluster assertions

EVALUATION

Aster CEP — complex event processing engine

Rules are written in **EQL** (SQL-92 core plus streaming extensions) and evaluated **inside** the CEP server — there is no separate streaming cluster to stand up and operate.

- **SQL:2016 row pattern recognition** — built into the engine, because equipment intervals need «every» row between start and end
- **Answers are cross-checked** — the same rule runs on Apache Flink 2.3.0 and the results are compared
- **152 golden rules** compared on every build, and **rules it cannot run are refused by name** — never a quietly wrong answer

INGEST

Aster MQTT — the door equipment comes in through

The MQTT broker plant sensors and PLCs connect to. Standard MQTT throughout; what we changed is what bites in operation.

- 🔴 **It refuses to start without an authenticator** — a broker whose security extension failed to load used to sit wide open, looking normal
- **It counts «why» a message was dropped** — capacity, back-pressure and policy as seven separate reasons. One number cannot tell you which
- **15 published vulnerabilities** closed by moving the networking library forward

Why owning the engine matters

When	With a third-party product	With our own engine
A feature the plant needs does not exist	File a request, wait for the roadmap	Change grammar and runtime, ship it next release
A defect appears	Work around it until the vendor releases	Find the cause in the source, fix it, pin it with a test
Deploying to an air-gapped site	Licence and supply chain checked separately	Apache-2.0 based, and the supply path is ours

What is worth reusing, we reuse. Kafka, Spark and Iceberg are standards and we use them as they are. We built the three places where general-purpose products do not take the plant floor as it comes: **industrial time series**, **real-time evaluation** and **equipment ingest**.

05 · Data lake

Hot · warm · cold — the path data takes as it ages

Industrial data changes its use over time. A value that just arrived feeds screens and rules; last month's feeds reports; last year's feeds analytics and AI training. The data lake keeps three tiers and moves data between them **automatically** — nobody moves it by hand.



Warm tier — retention per table (defaults)

Data	TTL	Why this length
Live equipment data (per-asset snapshots)	10 days	What screens and rules read — kept short and fast
Tag points (raw measurements)	62 days	Two months of raw data instantly queryable
Aggregates · samples · snapshots	93 days	Quarterly reports and trend comparison
Archive tables	1 year	A buffer readable straight from the ring after moving to cold
System logs · monitoring · batch/audit history	1 – 31 days	Operational diagnostics — no reason to keep long

Cold tier — moved automatically, tidied automatically

- **Every day at 02:20** a Temporal workflow launches a Spark job that moves **8 Aster TSDB tables** to Iceberg. Which tables move is declared as plug-ins (SPI).
- **Every Sunday at 03:00** optimisation — small files merged (128 MB target) · old snapshots expired · orphan files removed. The cold tier keeps itself lean.
- Partitioned by **site / year / month / day** — site isolation and period queries prune at file level. 256 MB files · zstd compression.

05 · Data lake

Token ring — plug in a node and capacity grows

The warm tier is a **token-ring distributed database** from the same family global hyperscalers use. Data is distributed and replicated across nodes automatically, so **adding a node raises capacity and throughput together**, and losing one node loses no data.



Figure 7. Topology screen — 1 master + 1 worker, 4 empty slots. The node detail shows ring state · DC/rack · load (8.37 GiB) · tokens · effective keyspace ownership (48.8%). *Captured from a live instance.*

How to grow — three commands

1. **Start a worker container** — same image, mode WORKER; only storage, analytics and the admin API come up.
2. `pd node status` — confirm the new node shows in the ring as `UN` (Up · Normal).
3. `pd node add` — **raises the replication factor automatically** to the live node count (2 → 2, 3+ → 3) and repairs. If it cannot read the datacenter or node count it **stops rather than guessing**.

What is replicated

Component	Method	On failure
Aster TSDB ring	Token-ring auto-distribution · RF up to 3 · datacenter-aware	Serves with one node lost
PostgreSQL	Master → worker streaming replication · slots retain WAL	Worker holds a current copy
Valkey	Primary → replica	Cache copy ready to promote

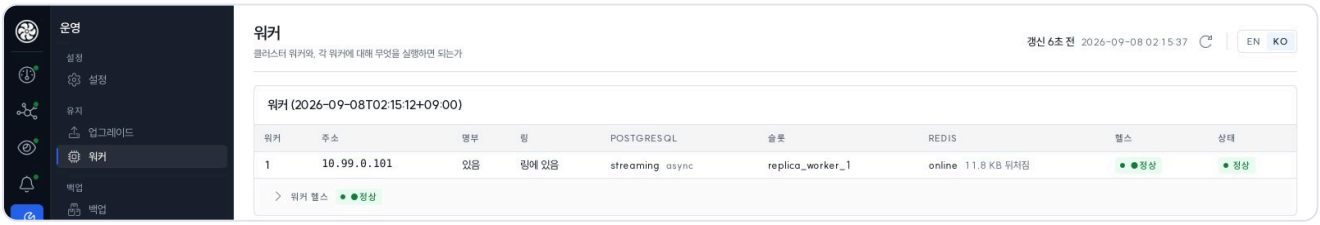


Figure 8. Workers screen — roster · ring · PostgreSQL streaming · slot · Valkey lag · health, judged in one row.

> **1 master + up to 5 workers.** Start with one master; attach workers as tags and retention grow — no migration project, no downtime.

05 · Data lake

Lakehouse — open formats, standard SQL, data that is never locked in

A data lake is judged not by what it can hold but by **what you can get back out**. The cold tier is **Apache Iceberg v2 + Parquet** — an open format any analytics engine reads, not a vendor's — and every tier is open over **standard JDBC**.

WHAT ICEBERG GIVES YOU

- **ACID transactions** — reading during an archive never shows half-written data.
- **Schema evolution** — more tags and columns without rewriting old files.
- **Time travel** — see "the table as it was last week" as-is.
- **Hidden partitioning** — files are pruned without naming partition columns in the query.

THE ENGINES ON TOP

- **Spark 4.1** — runs archiving, optimisation and large-scale analytics.
- **Kyuubi** — multi-user SQL serving · JDBC endpoint.
- **Gravitino · Hive metastore** — Iceberg REST catalog · table metadata.
- **RustFS** — S3-compatible object storage, on-premises.

How to ask — four languages, one gateway

Language	Target	Access
PQL	PostgreSQL — asset model · configuration · master data	Standard JDBC
CQL	Aster TSDB — warm time series · events · alarms	Our own **JDBC driver**
SQL	Spark / Iceberg — cold tier · large-scale analytics	Kyuubi Thrift · our own **Spark SQL JDBC driver**
TQL	Time-series engine — roll-ups · 30+ aggregators · time windows	`jdbc:ts://` — anyone who knows SQL uses it at once

All four also go through the **Data Gateway** (`POST /api/v1/query`) — large results stream as NDJSON from `/query/stream` , and `/cache` returns in-memory current values. A Java client SDK is provided.

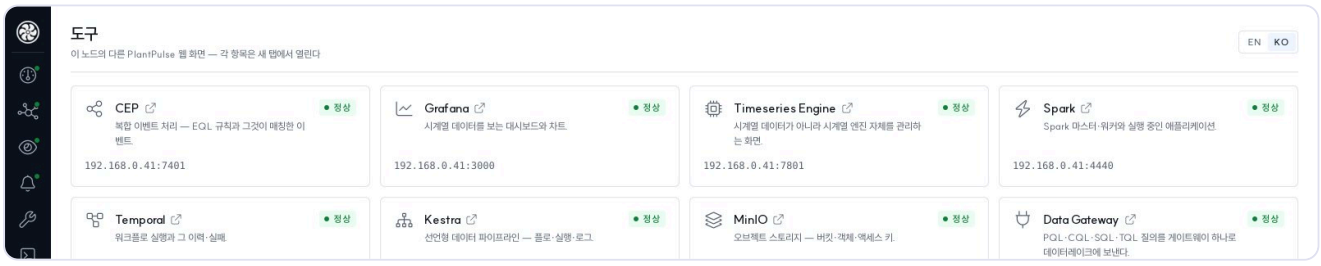


Figure 9. Console "Tools" — CEP · Grafana · TSE · Spark · Temporal · Kestra · RustFS · Data Gateway, each with its own screen and status inside one node.

OPEN, BUT THE GATEWAY STANDS GUARD

DDL blocked · multi-statement blocked · **100,000-row** cap · **1,000 req/s** per IP · request size cap · every query in the **query log** — open to analysts, safe for the operational stores.

05 · Data lake

A data lake you can see — the console and `pd`

Distributed storage usually needs a dedicated team. This product puts the storage inside the product and makes **the first screen answer "is anything wrong right now?"** — 16 components measured against their own thresholds, and 24 hours shown as a strip.

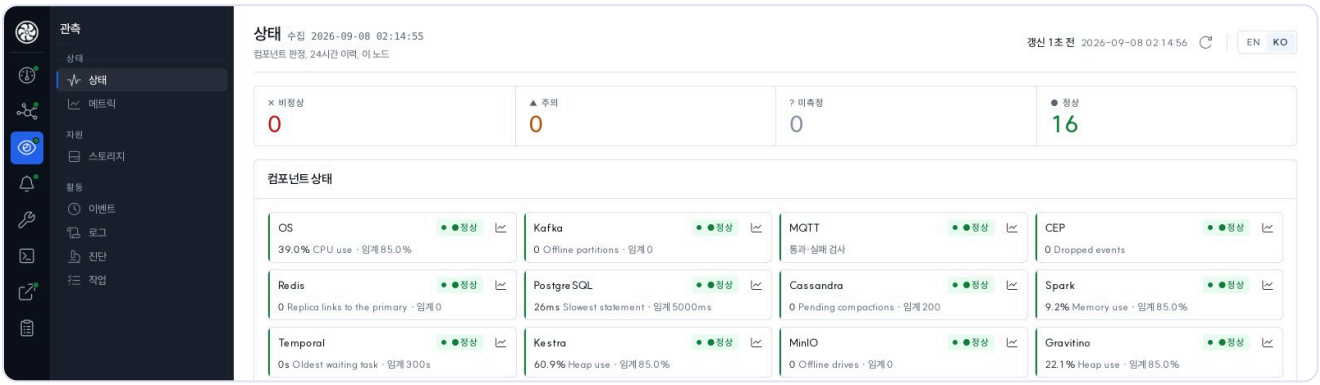


Figure 10. Status screen — each component shows what was measured and what the threshold is (Kafka offline partitions 0, PostgreSQL slowest statement 26 ms, Aster TSDB pending compactions 0, TSE ingest queue 626 …).

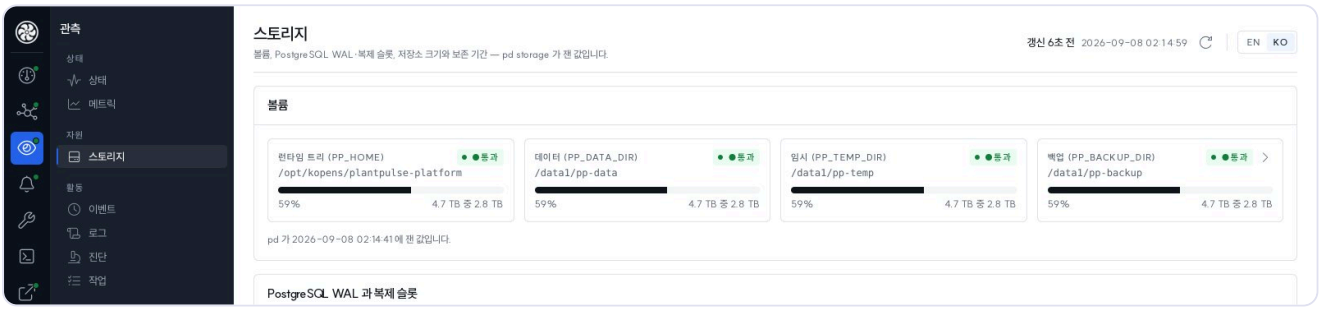


Figure 11. Storage screen — usage of four volumes (2.8 of 4.7 TB), PostgreSQL WAL and replication slots, Aster TSDB and Kafka disk and retention. Verdicts are exactly what `pd storage` measured — the screen invents no thresholds of its own.

Twelve console screens in five groups

Group	Screens
Overview	Dashboard — one-line verdict · active alarms · component health · timeline
Observe	Status · Metrics · Storage · Events · Logs · Diagnostics (`pd doctor`) · Jobs
Operate	Config (template render · diff) · Upgrade · Workers · Backup
Tools · Records	Shortcuts to 8 other web screens · Audit log

`pd` — the single entry point to the data lake

`pd start | stop | restart | status` · `pd config render | diff` · `pd secret rotate` · `pd node add | status | repair` · `pd backup run | restore | schedule` · `pd doctor` · `pd storage` · `pd logs` . Dozens of scattered scripts became one verb, and every command emits the same values with `--json` — console and terminal see the same thing.

> **"Unmeasured" is not "healthy".** A value that could not be read stays grey; it is never filled with 0 or green. A green dot lights only when something was actually measured — the rule that lets operators trust the screen.

05 · Data lake

Safeguards — backup · restore · upgrade

The data lake is the plant's memory. So **backups are on by default**, restore is a button, and an upgrade is one image swap. All three are visible in the console and repeatable with `pd`.



Figure 12. Backup screen — PostgreSQL (pgBackRest) last full and differential, next scheduled runs, the list of backup sets and a **Restore button**. Aster TSDB (Medusa) follows below in the same shape. *Captured from a live instance.*

Default schedule — already running after install

Target	Tool	Cadence
PostgreSQL	pgBackRest	differential daily 01:30 · full every Sunday 01:00
Aster TSDB	Medusa	differential daily 01:20 · full every Sunday 00:10
Retention purge	purge	Every Sunday 03:00 — sets past retention deleted
Cold archive	Temporal + Spark	Daily 02:20 (previous pages) — a second copy independent of backups

- Schedules run as **systemd timers** inside the container; when switched off they **record** “skipped by configuration” — no backup ever fails silently.
- We recommend keeping the backup root (`/data1/pp-backup`) on a **separate disk or NAS**. Sync to external S3 is a configuration item.
- For partial loss there is a **data recovery tool** — it rebuilds per-asset data from raw tag points for a chosen period and set of assets.

Upgrade — swapping one image

You see what is running

The console's Upgrade screen shows image tag · build commit · bake time, and writes out the command to change it.

No configuration drift

Every runtime setting is **rendered from templates at each start**. Hand edits inside the container do not survive.

Reversible

Roll back to the previous image tag and you are done. Data lives in volumes and settings in templates, separate from the image.

Every screen in this chapter was captured from a development instance in continuous operation — backup sets, volume usage and node load are real values. Retention, backup cadence and node count for your environment are agreed during deployment.

06 · Key strengths

④ KPIs appear without being built

Building OEE from the definition up to the calculation logic is a common chore. PlantPulse Platform **generates international-standard KPIs automatically** from collected data.

KPI	Standard	Content
OEE	ISO 22400	Availability × performance × quality computed on ingestion and broken down into the six big losses
RAM	Reliability engineering	MTBF · MTTR · availability · reliability R(t) derived automatically from operating data
EMS	ISO 50001	Energy intensity · peaks · CO_2 estimation aggregated automatically

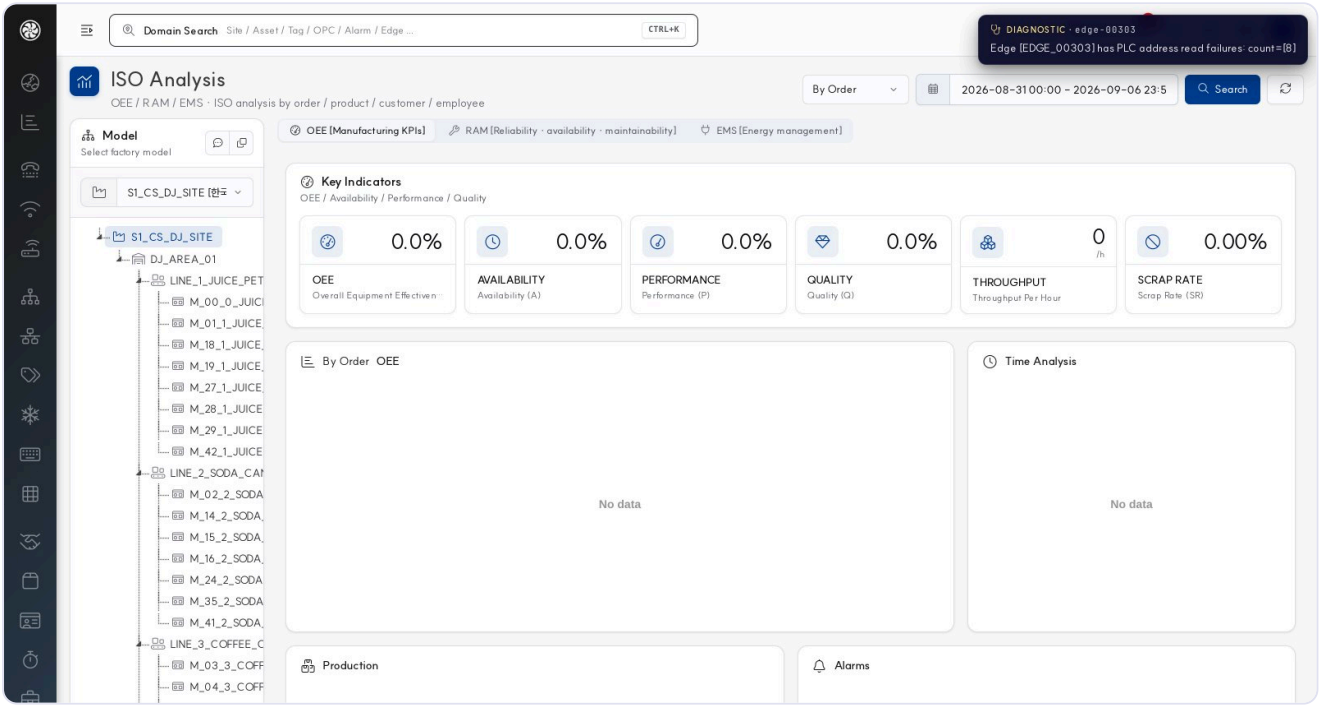


Figure 13. ISO analytics — OEE, RAM and EMS broken down by order, product, customer and operator.

Why “to the standard definition” matters

- **Sites become comparable.** Different definitions per plant mean the numbers cannot be combined.
- **Usable as-is for audits and reporting.** Standard definitions need no separate justification.
- **Nobody maintains KPI definitions.** Calculations do not wobble when staff change.

06 · Key strengths

⑤ Turns the plant into a knowledge graph

Classifying assets into a tree is common. PlantPulse Platform layers **relationships between equipment** on top, turning the whole plant into a queryable knowledge graph.

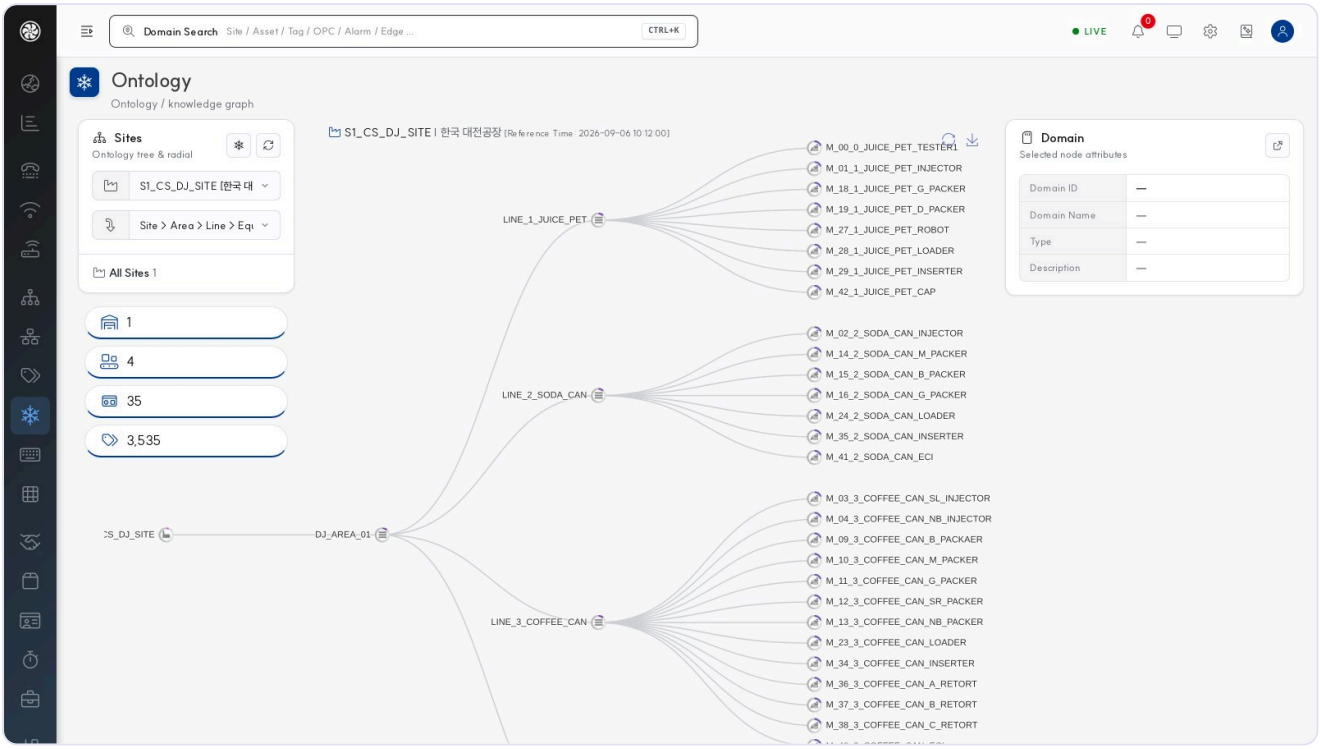


Figure 14. Ontology screen — sites, assets, tags and the relationships between them, explored as a graph.

Questions a tree cannot answer

“If this machine stops, how far does the impact reach?”

Graph traversal along supply, dependency and production relationships answers the downstream impact.

“Is any tag missing its alarm setting?”

Setpoint tags and the alarm registry are linked by relationship, exposing the gaps.

People and AI see the same context

The ontology is stored in a standard knowledge-representation format and open to read-only queries. The built-in AI uses this graph as **evidence** for its diagnoses, so it can explain “why it decided that” in terms of equipment relationships.

> **Why go beyond the standard hierarchy.** A standard asset hierarchy says “where it is” but not “what it affects”. The questions that are actually urgent — failure propagation, root-cause tracing — are mostly the latter.

06 · Key strengths

⑥ The asset hierarchy is the real-time address space

If every external system has to learn each vendor's protocol, integration never ends. The **Unified Namespace** turns the asset tree itself into the **single address scheme** for real-time data.

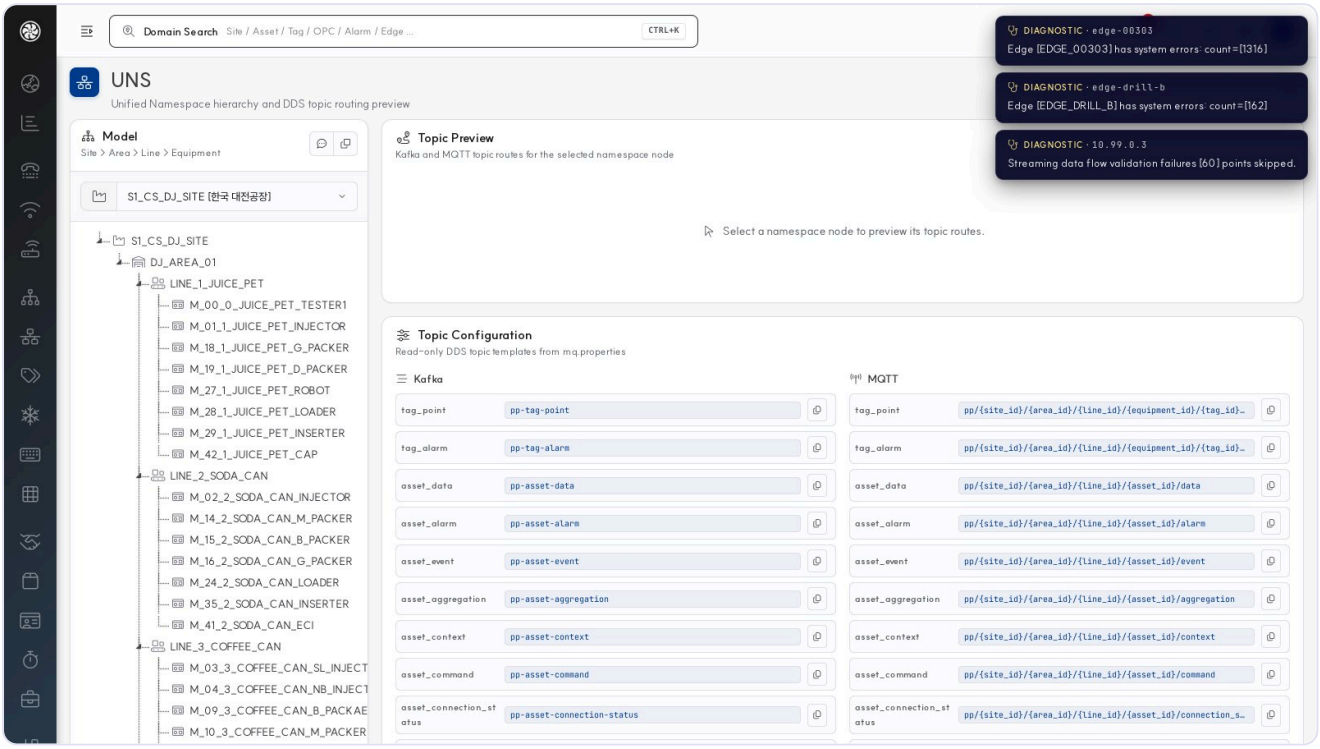


Figure 15. UNS screen — the site → area → line → equipment hierarchy becomes the topic path; preview and copy a path in the console and attach an external system directly.

What is mirrored

Data

Measurements · aggregates · predictions

Occurrences

Alarms · events · commands

State

Connection state · health

Mapped automatically onto three industry-standard topic schemes, so an external system needs to know **one rule** to subscribe to the whole plant.

Integration moves from development to configuration

- New equipment **extends the address space automatically** — no integration target list to maintain.
- Connection and health state use the same scheme, so **subscribers know on their own when a machine is dead**.
- Vendor-specific communication converges into **one standard namespace**.

07 · Key strengths

⑦ Operators build the automation themselves

“When an alarm fires, create a work order, notify the owner and pass it to MES” — flows like this were always development items. **Flow** lets you draw them on screen.

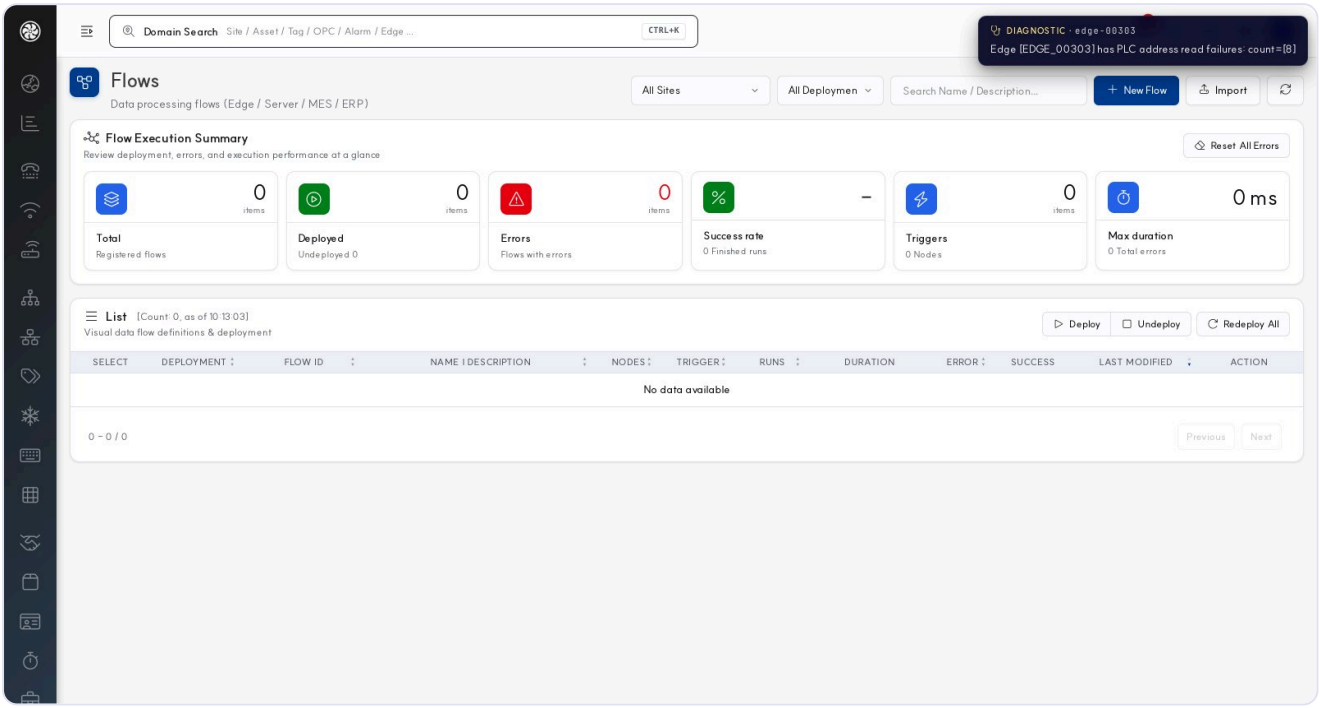


Figure 16. Flow list — data-processing flows targeting edges, servers, MES and ERP are built here and their run history managed.

What can be chained

Kind	Examples
Triggers	Tag values · alarms · asset events · OEE changes · edge state
Processing	Filter · script · aggregate · retry · flow control
Actions	Issue work orders · email/SMS/push notifications · call external systems · control edges

Around 100 node types, connected by drag and drop on a single screen.

Safe even when operators build it

Deploy · roll back

Revert to the previous version if something goes wrong.

Automatic versioning

A snapshot is kept at every change.

Live debugging

Fix it while watching real data flow through.

> **The development queue disappears.** “Can I get this alarm in the messenger too?” stops being a development ticket and becomes a 30-minute job for the person on the floor.

07 · Key strengths

⑧ Open in a form AI can use

Giving an AI database access and letting it **ask in operational terms** are entirely different things. PlantPulse Platform provides the latter through a standard interface.

Operational concepts, not tables

Platform data is exposed as **high-level industrial-operations tools**. Instead of writing SQL, an AI agent asks "OEE for line 3 this shift" and gets the answer.

Kinds of tools provided

Equipment status · time-series queries · alarm queries · KPIs such as OEE · ontology relationship traversal · document search and more — around 100 high-level tools.

Who can connect

Not only PlantPulse AI but **agents built in-house** and any LLM tool that supports the standard connect the same way.

Open, but under control

- Every tool runs **inside the platform's permission model** — being an AI grants no extra visibility.
- Calls are written to the **audit log** — who asked what is traceable.
- **Runs on-premises** — complete even in closed or air-gapped networks.

WHY THIS IS AN ASSET

AI tools keep changing. **Keep the data open through a standard** and the integration does not have to be rebuilt when the tool changes. The structure is not tied to any particular AI product.

08 · Architecture

The path the data takes

One page on how a single measurement from the edge is stored, becomes a KPI, and reaches the screens and the AI.

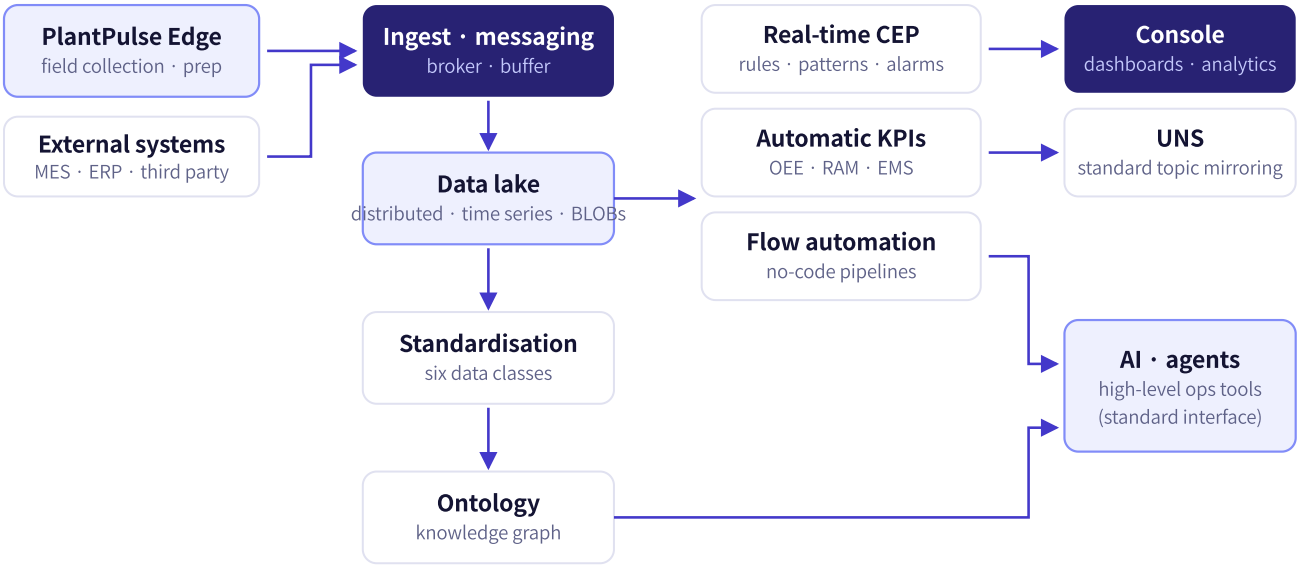


Figure 17. From collection to openness. The point of this structure is that data which has passed through standardisation and the ontology reaches screens, external systems and AI **in the same shape**.

09 · How it is used

The screens an operator sees every day

Every screen below was **captured from a system in live operation**.

Integrated monitoring — the day starts on one screen

Site map, every pipeline stage, infrastructure component health and key metrics on one screen, laid out so that **“is everything normal right now?”** is answered without scrolling.

Data pipeline

Throughput and state of receive → process → buffer → store shown per stage. Where things back up is obvious at a glance.

Infrastructure health

Per-component state for cache, messaging, storage, processing engine and more. When one wobbles, only that row changes colour.

Platform health timeline

Normal, warning and critical periods across the day as a strip — to trace back “since when has it been wrong?”

Key metrics

Connection rate · today's ingestion · alarms · equipment utilisation · availability · overall OEE.

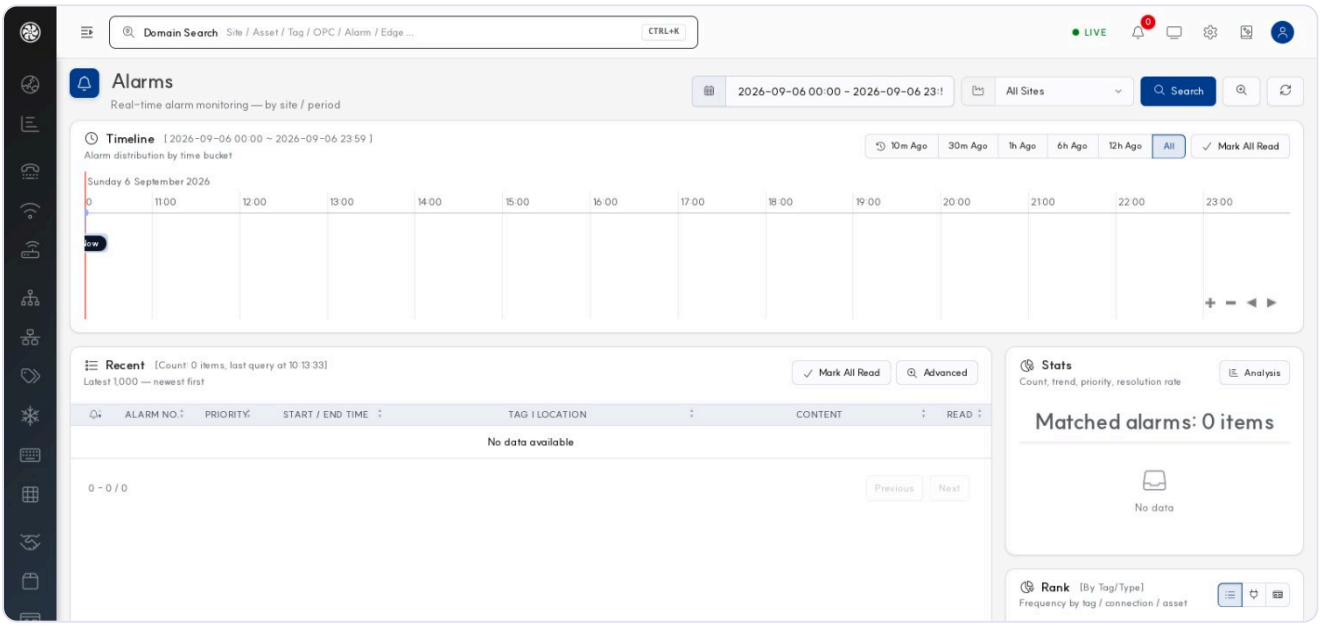


Figure 18. Alarm screen — real-time alarms by site and period. Simple threshold alarms and CEP-generated advanced alarms meet on the same screen.

09 · How it is used

Managing edges and assets

Edge connection state

The connection and collection state of edge gateways in the field, seen centrally. This screen checks **whether the collector itself is alive** before looking at the equipment.

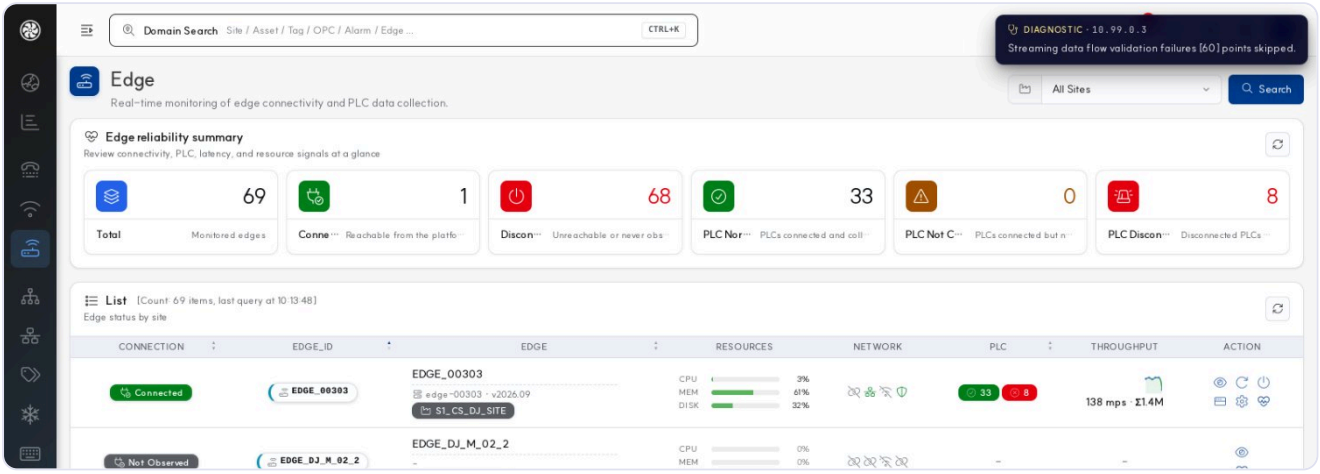


Figure 19. Edge screen — connection state of edges per site and their PLC collection status in real time.

Build your own dashboards

Dashboards beyond the standard screens are composed on the **canvas**. Drag assets and tags onto it to build a screen, then share it.

Asset explorer

Find equipment through the tree and the relationship graph.

Domain search

Sites, assets, tags, alarms and edges from one search box.

Audit log

Who changed what is kept on record.

> **Having many screens is not the goal.** An operator opens three or four screens a day. The rest are paths to dig into when something goes wrong, and the value of this product is that those paths **sit on the same data**.

10 · Security

Security · operations

This is the system where all plant data converges. So **who can see what** and **what gets recorded** matter as much as features.

Access control

- **User and role-based permissions** — visibility divided by site and asset.
- **API tokens** — external integrations only with issued tokens, issued and revoked from the console.
- **AI tools use the same permission model** — an agent sees no more than its user.
- **Audit log** — reads, changes and integration calls are recorded.

Network placement

The platform is **installed inside the customer's infrastructure**. It is complete in a closed network with no outbound connection, and even the AI features can be configured on-premises — data never leaves the site.

High availability

Item	Content
Storage redundancy	Replicated distributed storage — no data loss on node failure
Node expansion	Capacity and throughput grow by adding nodes during operation
Component monitoring	Health checks and a state timeline per infrastructure component
Edge disconnection	Edge-side buffering preserves data across upstream outages

We recommend a security review before deployment. The items above are what the product provides by default; configuration to your security policy (network segregation level, identity integration, log collection scope) is agreed during deployment.

11 · Delivery

Installation · delivery model

Seven layers and more than 24 components ship as **one install package**. There is no installing components one by one and matching versions.

Install

All layers from one package; the full pipeline runs within minutes.

Air-gapped

An offline bundle for environments without internet.

Scale

Add nodes during operation to grow storage and processing.

Sizing criteria

Criterion	Effect
Tag count · interval	Determines ingestion rate per second — the primary basis for node count
Retention period	Determines storage capacity; raw and aggregate retention are set separately
Concurrent users · dashboards	Determines query load
Number of CEP rules	Determines real-time evaluation load

Give us these four and we will size the **recommended node configuration and specification**.

Upgrades

Delivered as regular releases; operations can be pinned to a specific verified version. Upgrades happen **as a package**, not scattered across layers.

RELATIONSHIP WITH THE EDGE

The platform is most natural together with **PlantPulse Edge**. It is not mandatory, though — data arriving over standard protocols from third-party gateways is accepted too.

References

Technology proven on manufacturing floors for 15 years.

162 projects delivered	**34** sites in operation	**642,021** connected sensors
100 TB+ data under management	**15 years** manufacturing-floor experience	**6** awards

Representative deployments

POWER · HEAVY INDUSTRY

Doosan

ICT-converged wind-farm operations system — combined with a digital-twin framework to improve availability through turbine health tracking and predictive maintenance.

FOOD & BEVERAGE

Lotte Chilsung

Smart-factory platform and real-time OEE system for production lines.

SHIPBUILDING · OFFSHORE

Hanwha Ocean

Real-time pipeline for large-scale shipyard OT data — digitalisation of construction and operations data.

LOGISTICS · ELECTRONICS

Samsung Electronics

OT data platform for a next-generation logistics centre — a real-time integrated operations base for logistics equipment and process data.

PUBLIC · TESTBED

AX testbed

Selected as the builder of an industrial digital transformation (AX) testbed.

RESEARCH · TESTBED

KERI

Industrial data platform for the Korea Electrotechnology Research Institute's smart-factory testbed plant — from equipment data collection to real-time monitoring.

Customers we have worked with

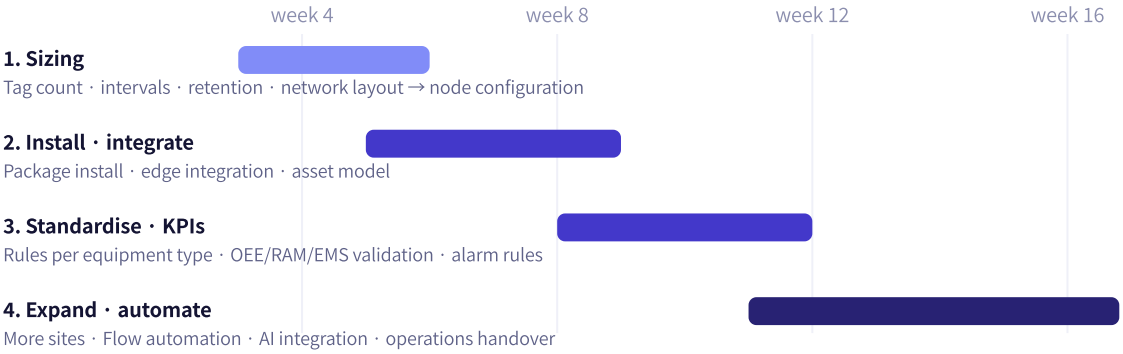
Samsung Electronics · Samsung C&T · LG · Doosan · Doosan Mottrol · Hanwha · Hanwha Ocean · Hanwha Aerospace · Hanwha Engine · Lotte Chilsung · Lotte Chemical · Lotte Fine Chemical · Lotte Wellfood · Amorepacific · Hyundai Rotem · Hyundai Mobis · HD Hyundai Heavy Industries · Hyosung · Hansol · SK Chemicals · CJ Logistics · KAI · KEPCO · KT · Denso · Carrier · Seegene · Shilla · Winia · STX · Iljin · KERI · KEIT · Ministry of Trade, Industry and Energy and others.

Detailed scope and outcomes by industry are provided individually through your sales contact.

13 · Deployment

Deployment plan and schedule

We recommend **four stages** — start at one site, then expand to the enterprise.



What to confirm in the PoC

- At our tag volume, **does ingestion keep up**
- With one equipment-type rule, **do machines of that type attach as-is**
- Does OEE computed to the standard definition **match what the floor believes**
- In a closed network, **is everything — AI included — complete**

Decisions to make together during deployment

Retention policy

Separate retention for raw and aggregate data to design storage cost.

Existing systems

Which to replace and which to run alongside.

Permission model

Visibility by site and asset, and how to integrate corporate accounts.

Where AI runs

On-premises model or external API — whichever fits the network policy.

PlantPulse® **Platform**

INDUSTRIAL DATA INFRASTRUCTURE

GET IN TOUCH

Start your industrial AI transformation today.

Consultation, demo requests, quotations — whatever you need, leave us a message. A specialist replies within one business day, and we can schedule a live demo on the real product right away.

PHONE

+82 1588-5268

EMAIL

webmaster@kopens.com

INQUIRIES · DEMO REQUESTS

kopens.com/contact

COMPANY

kopens.com

PRODUCT

plantpulse.io

DOCUMENTATION

docs.plantpulse.io

THE PLANTPULSE® FAMILY — DEPLOYED TOGETHER, ONE STACK FROM THE FLOOR TO AI

Edge	Platform	AI	Studio
Industrial protocol edge gateway	Enterprise IIoT / DataOps platform	Plant operations AI copilot	AI application builder

KOPENS, Inc.

© 2026 KOPENS. All rights reserved. PlantPulse® is a registered trademark of KOPENS, Inc.

Document Rev. 2026.09 · Figures and screens are as of the date of writing and may change without notice.

kopens®