Quix Pipelines

Automate all your vehicle test analysis workflows in one unified environment

[WHAT IT IS]

Quix Pipelines turn manual steps into automated workflows

Quix Pipelines automate test analysis workflows so that they are consistent, repeatable and run one unified environment. Each workflow step runs in a Quix App container, which behaves like a lightweight virtual machine, and listens continuously for output from the step before it. As soon as new data arrives, the next step starts processing it. The code for each step is stored in a version-controlled repository alongside the definition of the whole pipeline, so you can see exactly what changed and when.

[THE PROBLEM]

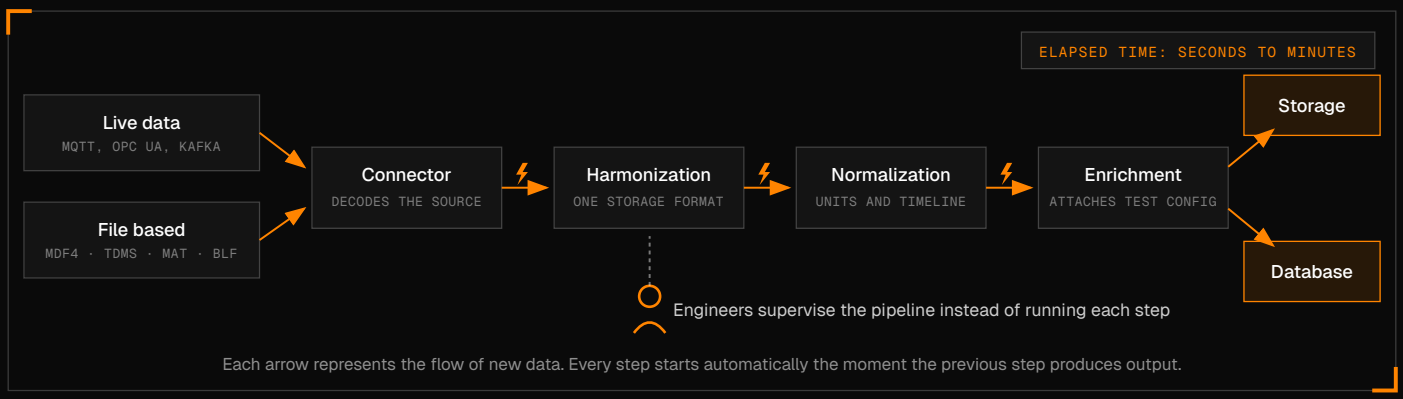
Vehicle test data analysis is slowed down by unnecessary friction

Vehicle testing has a data problem. The route from test to decision involves many steps, and many of them are already automated individually. One engineer converts the files from the DAQ, another checks that the sensors were valid, another computes the derived channels, and each step waits until an engineer notices that the previous step has finished.

These steps are not chained together as a repeatable workflow. Engineers run ad-hoc scripts that live on different PCs. Highly skilled engineers then spend hours on manual data cleanup, clicking, copying and typing in terminals, which slows analysis down for no good reason.

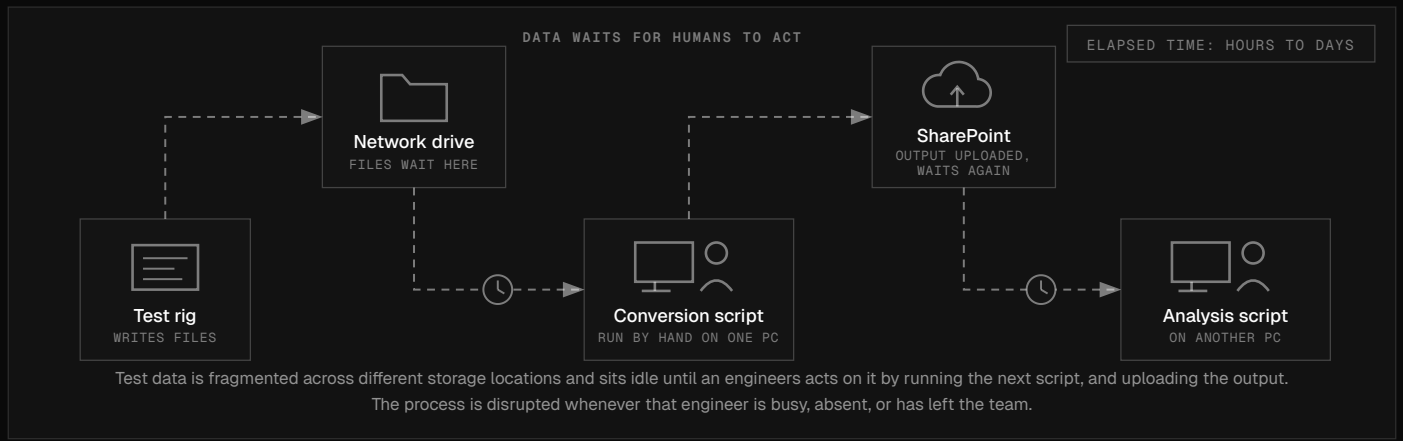
Quix Pipelines solve this problem by allowing engineers to move distributed scripts in to one centralized workspace that the whole team can access.

WITH QUIX PIPELINES: test data is automatically cleaned and analysed in one unified enviroment



THE SAME STEPS, CONNECTED

WITHOUT QUIX PIPELINES: engineers move and process data across fragmented locations using ad-hoc scripts



Quix Pipelines also include ready-made components for each phase of a data processing workflow: ingestion, processing and storage.

[01] Ingestion

Every pipeline starts with an input connector matched to the data you want to ingest. Quix has a connector library covering two modes:

LIVE DATA

Continuous streams such as vehicle telemetry and high-frequency sensor channels.

FILE BASED

Data recorded from a DAQ or logger and exported to formats such as MDF/MF4, TDMS, MAT and BLF. A file watcher polls a network location for new files and ingests them into Quix.

Whether the source is a live stream or a file, it reaches the pipeline as a continuous data stream.

[02] Processing

Processing is where the transformations happen. Some of them you always want to run first, to bring the data into a consistent state. Enrichment, harmonization and normalization are all ready-made building blocks.

HARMONIZATION

Converts different formats (MDF4, CSV, JSON) into a single storage format with a standard structure, and reconciles naming conventions.

NORMALIZATION

Scales numerical ranges so that comparisons are mathematically valid. Aligns timestamps on a fixed millisecond timeline.

ENRICHMENT

Attaches the test configuration to every record, using the dynamic configuration manager. Includes metadata such as test campaign details, and which settings were used to run the test.

You can write any other transformation you need and feed the cleaned data into simulations or ML models. Quix also includes common transformations that downsample, time-slice, aggregate (maximum, minimum, mean) and calculate derived metrics.

[03] Storage

Ready-made sink connectors write data to any destination at the end of the pipeline: back into Quix Lake, into your own object storage, or into your own database for others to query.

[04] Next steps

SEE IT WITH YOUR OWN DATA

Book a product demonstration and watch a pipeline capture your test data, tagged with its configuration.

BOOK A DEMO

READ THE DOCUMENTATION

Connectors and samples

The catalogue of pre-built sources and destinations

Dynamic Configuration Manager

How data is enriched with versioned configuration

Pipeline descriptor (quix.yaml)

The infrastructure-as-code format behind every pipeline