



PRODUCT BROCHURE

Bringing the VDA5050 Community together.

Analyzer, Testbench, Monitoring: one platform, the way you need it.

One platform, three perspectives: the vendors behind the fleet controller, the vehicle manufacturers and the operators whose fleets have to run.

250

checks for VDA5050 and M2X

15

checks for VDMA LIF

1.1 – 3.0

supported VDA5050 versions

A PRODUCT OF



logiq-vda5050.com
support@logiq-vda5050.com

At a glance

Three areas, one check catalogue: the analyzer reads traffic that happened, the testbench produces some, and monitoring never stops. One platform, one shared body of evidence.

Analyzer

A log file or a capture against VDA5050 1.1 through 3.0, M2X and VDMA LIF: verdict, score and coverage, every finding tied to its message. All of it in a signed package the recipient can verify independently. Beside the standard it counts custom rulesets and somebody else's requirements.

Testbench

The other direction: the tool plays the fleet controller and sends real orders. A certification run plans over your own layout what the vehicle has to show, before anything is sent.

Monitoring

Connectors listen without end and fold the traffic into quarter-hour cells: availability, faults, speed and braking places per machine over months. Preventive maintenance sits on top of them.

Contents

The interface is in the standard. Real life is not.	3	What the fleet did with its time	17
One platform, three perspectives	4	One order, one vehicle, sent now	18
Custom rulesets	5	A certification run is a document before it is a drive	19
Publishing requirements	6	A twin is measured, not typed	20
Four ways to the traffic	7	A fleet stands ready for somebody else to drive	21
How checking works	8	Watching without end: the connector	22
Coverage: what the verdict is worth	9	Fleet Health: the machines, not the protocol	23
Declaration against reality	10	Preventive maintenance	25
Report and traceability	11	Unattended operation and reporting	26
Findings in context	12	Teams, projects and plans	27
From finding to acceptance	13	Operations and security	28
Layout and visualization	14	Service and support	29
Live view	15	Technical data	30
Replay	16	Check catalogue	32

01 The interface is in the standard. Real life is not.

The vehicle and the fleet controller come from different vendors. The specification describes the messages. It does not prove that both sides read them the same way. When they do not, three questions cost real time.

1

“Who is right?”

When something breaks, it is one vendor's claim against another's. Without evidence, the argument takes days.

2

“What was checked?”

A green tick without a scope is worthless. An acceptance test has to state which rules ran, which did not apply and why.

3

“What happened there?”

The fault appears once, on a Friday evening. Without a recording that keeps its context, all that is left is to reproduce it, and hope.

What this tool puts against that

The same traffic, the same check catalogue and the same verdict for all three parties - with the message every finding rests on, and a scope printed beside the result rather than missing beneath it.

FLEET CONTROLLER

Every onboarding starts with the same weeks spent settling what the integration manual already said.

VEHICLE MANUFACTURER

The vehicle is judged in the customer's plant, where mistakes are most expensive and have an audience.

OPERATOR

An incident leaves two accounts standing against each other, and neither can be checked.

02 One platform, three perspectives

VDA5050 is a promise between three parties. The fleet-controller vendor talks to vehicles it did not build. The vehicle manufacturer delivers into sites it does not run. And the operator relies on both meaning the same protocol. The analyzer gives all three the same view of the same traffic and each role the tools it needs.

The fleet-controller vendor

Writes the requirements down once and hands them out as a file instead of explaining them again in every onboarding. Drives new vehicle types against virtual fleets before any test rig exists. And decides about its own badge: nothing counts as passed until the vendor releases it.

The vehicle manufacturer

Measures on its own test rig against the published requirements of every target fleet controller, as often as it takes and with nobody watching. What is still missing comes back as named lines, not as a rejection. The manufacturer's own house rules run as a custom ruleset next to the standard.

The fleet operator

Watches the site live, receives reports on a schedule and holds evidence during an incident instead of two competing claims. Acceptance ends in a signed package that still verifies months later.

Why one platform

Onboarding weeks are born where each side points its own tools at its own recordings. On one platform, the requirement, the measurement and the evidence are the same files. Engineers from both sides meet where a decision needs them. Not to discover that two systems read the same message differently.

03 Custom rulesets

No fleet fails on VDA5050 alone. Every integration manual holds sentences that are binding on this site and invisible to the standard: “pickFromBin needs a binld of the form BIN-0000”. A custom ruleset brings such sentences into the same engine that judges the standard.

Data, never code

The file is descriptive JSON against a published schema, interpreted by seven fixed rule types. Nothing in it is executed. Otherwise the integrity line in every report, which says that only signed code runs, would no longer be true.

Checked at upload

Exactly one thing is verified: that the file is understood. Valid against the schema, every path readable, every regular expression compilable, no id colliding with the built-in checks. Whether the rules are wise remains the author's business.

Two verdicts side by side

A custom rule never moves the VDA5050 verdict. The report shows both: the standard alone and the standard together with your rules. The first is settled before the first custom finding exists.

Frozen with the report

An immutable copy of the rulesets used is stored next to the report files; the audit block lists name, version and SHA-256 of each. Replacing the upload later does not change what an existing report was measured against.

Availability and limits

From the Project plan with three rulesets, unlimited from Team. Up to 200 checks and 2 MB per file. Uploading takes a superuser, because a ruleset changes what every member's runs are judged against.

FLEET CONTROLLER

The quirks of your interface become checkable instead of being explained anew in every project.

VEHICLE MANUFACTURER

House rules per series run at every customer, as a file rather than as knowledge in a commissioning engineer's head.

OPERATOR

Site standards such as naming schemes or forbidden operating modes are measured in every run.

04 Publishing requirements

Every onboarding brings a fleet-controller vendor and a vehicle manufacturer together, and the first weeks usually repeat what the previous onboarding already covered: the same requirements explained on the phone, the same questions answered on site. The platform turns that into a file.

Written once instead of explained every time

The vendor puts its requirements into a file: which checks must pass, how much coverage a run needs, which actions a vehicle has to declare. It travels with the interface manual instead of being repeated in every call.

The manufacturer measures before anybody travels

The manufacturer records a shift on its own test rig and holds it against that file, as often as it likes. What is still missing comes back as named lines, not as a rejection somebody has to interpret.

The badge is the vendor's decision

The vendor approves the link between the two sides; only then does a report carry its badge. Nobody grants themselves a pass. The vendor never sees the manufacturer's recordings.

Commissioning starts on known ground

What reaches the site is what a recording cannot answer. Questions that used to cost a trip are settled before the vehicle is on the truck.

What it costs and what it saves

The savings sit in the part that repeats: the introduction calls that start from zero with every manufacturer, the trips that settle one question, and the weeks between a complaint and its evidence.

FLEET CONTROLLER

One requirements baseline for every manufacturer, maintained in one place and valid everywhere.

VEHICLE MANUFACTURER

Every target platform measurable in advance; the trip starts once the lines are green.

OPERATOR

New vehicle types arrive pre-checked. The site is no longer a test ground.

05 Four ways to the traffic

Depending on what you have: a file from the field, a running broker or a site nobody is allowed to touch.

1 Import a log file

Recordings in .log, .txt, .json or .ndjson. The importer detects the message type, reports lines it could not read and keeps the field order of the source. What you see in the tool is character for character what is in your file. Several files at once are read and listed first; then you decide whether they stay separate or merge into one log.

2 Capture live

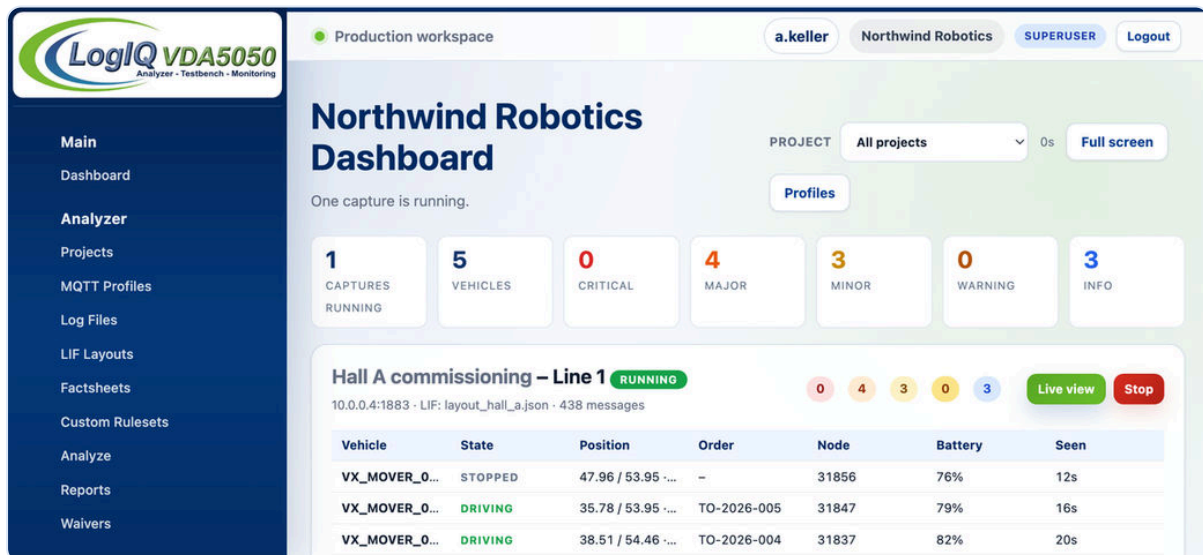
A capture profile holds broker, credentials, topics, target version, check selection and optionally a LIF layout. Started profiles record in the background and analyse every message as it arrives; several profiles run at the same time.

3 Transparently in between

The same profile can mirror the recorded traffic onto a second broker. The analyzer then sits between vehicles and fleet controller as a transparent proxy without changing operations. That helps when only one network segment is reachable.

4 Connected permanently

A connector is the same profile without an end: no daily limit, it returns by itself after an interruption and starts with the service.



One workspace for imports, analyses, layouts and running captures.

FLEET CONTROLLER

Check the log exports of your own system without touching a site.

VEHICLE MANUFACTURER

The test rig records itself; every shift becomes an analysable run.

OPERATOR

Read access to the broker is enough. Nothing gets imported at all.

06 How checking works

Checking is a chain: a message must first match the schema at all, then its structure, then its history. Only then can anybody judge whether order and state belong together.

Schema

Every message against the official JSON schema of the target version, with deviations handled as declared corrections. All four versions 1.1 through 3.0 are kept separate, because each speaks its own dialect.

Structure

Required fields, data types, ISO 8601 timestamps in UTC, semantic version, collections as arrays. Enumerations are checked against the values the target version allows.

History

headerId gaps and regressions, timestamp regressions, node and edge sequences, action lifecycles from dispatch to final state, battery and safety reporting.

Correlation

Is a dispatched order acknowledged? Does lastNodeSequenceId stay monotonic within the order? This correlation runs per vehicle.

Severities

Critical	The message is not specification-conformant; the other side may reject it.
Major	A clear deviation with an effect on operations.
Minor	A deviation without immediate effect, but reportable.
Warning	A safety contradiction without a clause in the standard: loud in the verdict, no deduction.
Info	Context, not a defect.

A fleet is not one vehicle times ten

VDA5050 counts headerIds per topic, and the topic carries manufacturer and serial number. That is exactly how the analyzer judges: every sequence, every lifecycle, every order-state correlation per vehicle. Read as one stream, two vehicles that both start at headerId 1 would look like one sender jumping backwards. That would be hundreds of findings that are not.

Three numbers, not one

The findings produce a score per check area, an overall value and a verdict (PASS, WARNING, FAIL). Next to them stands coverage: how much of the standard the recording actually put to the test. A flawless score on a third of the rules proves almost nothing. Only one of the two numbers tells you that.

07 Coverage: what the verdict is worth

A score says how much of what was checked has passed, not how much there was to check. That is why every run carries a second number.

What it is measured against

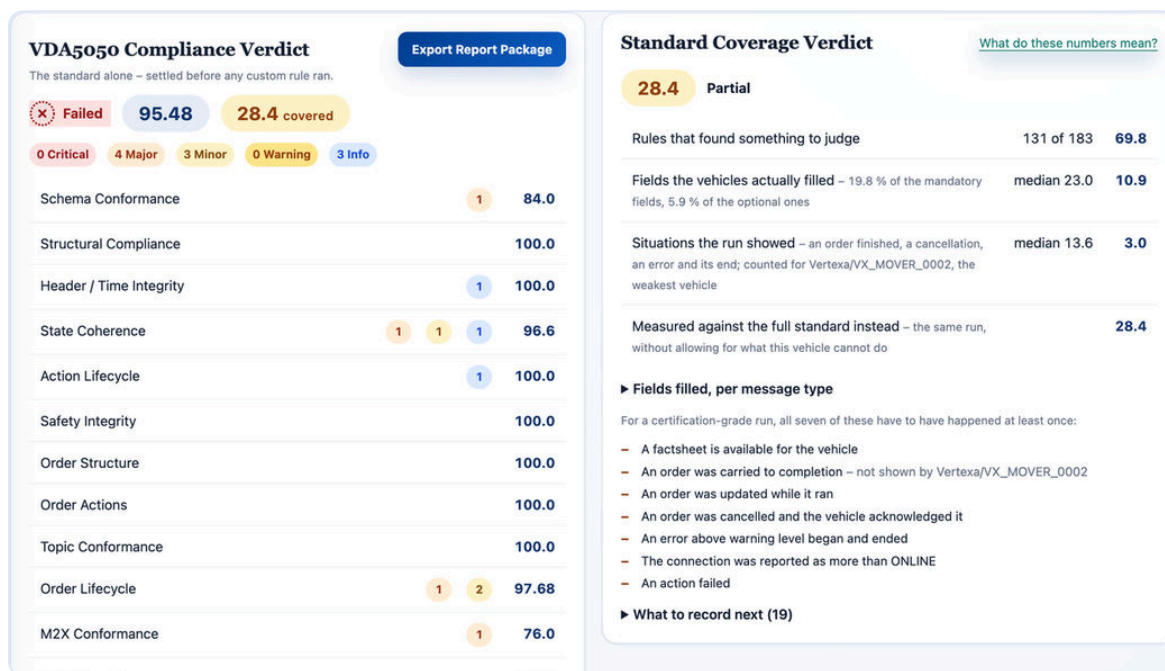
Against what this vehicle is supposed to do: everything the target version describes, minus what its factsheet declares no hardware for. A tugger without a lift is not marked down for never lifting.

Three questions

Did the checks have anything to judge? Did the vehicles fill the fields of the standard, required and optional reported separately? And did the run show the situations that only occur when something is actually done?

Seven gates

A factsheet present, an order driven to completion, one changed while driving, one cancelled with acknowledgement, an error with a beginning and an end, a connection state beyond ONLINE, a failed action. Every open gate names the vehicles holding it open.



Two verdicts side by side: what passed and how much could be checked at all.

Neither number ever moves the other

A run that covers little is not marked down for it; it is reported as covering little. With several vehicles the weakest one counts, never the average. Otherwise a busy vehicle covers for one that merely logged on.

What to record next

For every gap, the report says what the next capture has to show to close it. The number becomes a work instruction instead of a complaint.

08 Declaration against reality

Every vehicle declares in its factsheet what it can do. 24 checks hold the traffic to that declaration and answer a different question than the rest of the catalogue: not “does this message follow the standard” but “does it fit this vehicle”.

Actions

Every actionType in orders and instantActions must exist in the vehicle's catalogue: in the right scope, with a declared blockingType and with the parameters the vehicle expects.

Optional parameters

The specification is strict here: what is not listed counts as unsupported. The analyzer reports optional fields that are sent although the vehicle never declared them.

Limits

Message length, length and character set of identifiers, nodes and edges per order, actions per node, nodeStates and actionStates per state. Both directions are measured against the declared limits.

Timing

Minimum spacing between orders, minimum and regular spacing between states. Checked with a tolerance set per analysis, because the clocks of the systems involved do not run in sync.

Who caused it?

Every finding in this group names the side that caused it: the fleet controller demanding something undeclared, or the vehicle not honouring its own declaration. The most common commissioning argument ends in an attribution, not in claims.

No factsheet, no claim

If the factsheet is missing, or does not declare an area, the affected check reports “no data”, never “passed”. The report names every vehicle and whether it was measured: a vehicle that was skipped must not look like one that passed.

FLEET CONTROLLER

Proof that the fleet controller demanded nothing undeclared, per finding and per vehicle.

VEHICLE MANUFACTURER

The factsheet turns from a data sheet into a measuring stick: what is declared gets measured.

OPERATOR

During an incident the causing side is in the report, not in the room.

09 Report and traceability

A report that only lists errors is no use for an acceptance. The analyzer puts next to the result how it came about.

Active

The check ran and judged:
PASS or FAIL, with the number
of findings.

No data

The check ran but found no
message it applies to. It does
not pass silently; it is reported
as “no data”.

Deselected

Switched off on purpose:
visible in the report, not
hidden.

All 183 Active 131 No data 50 Switched off 2					
Rule	Check	ID	State	Result	Findings
SCHEMA CONFORMANCE					
VDA	State message conforms to the VDA5050 JSON schema	SCHEMA_STATE_VIOLATION	Active	FAIL	1
VDA	Order message conforms to the VDA5050 JSON schema	SCHEMA_ORDER_VIOLATION	Active	PASS	–
VDA	Connection message conforms to the VDA5050 JSON schema	SCHEMA_CONNECTION_VIOLATION	Active	PASS	–
VDA	instantActions message conforms to the VDA5050 JSON schema	SCHEMA_INSTANTACTIONS_VIOLATION	No matching messages	–	–
VDA	Visualization message conforms to the VDA5050 JSON schema	SCHEMA_VISUALIZATION_VIOLATION	No matching messages	–	–
FS	Factsheet conforms to the VDA5050 JSON schema	SCHEMA_FACTSHEET_VIOLATION	No matching messages	–	–
STRUCTURAL CHECKS					
VDA	Required fields are present	STRUCT_REQUIRED_MISSING	Active	PASS	–
VDA	headerId is an integer	STRUCT_HEADER_ID_TYPE	Active	PASS	–
VDA	timestamp is ISO-8601 UTC	STRUCT_TIMESTAMP_FORMAT	Switched off	–	–
VDA	version matches the selected target	STRUCT_VERSION	Active	PASS	–

The scope of one run: active, without matching data, deselected. Nothing passes silently.

Audit block

Every run records what it is made of: project, log file, LIF layout and factsheets, target version, the messages by type and every vehicle by name. Plus tool and catalogue version, schema revisions, a checksum over the input and the number of checks in total, enabled and judged. That makes a result repeatable months later, when nobody remembers which file was meant.

Output

A report exports as one signed package: the report as JSON and Markdown, the analysed file, the cleaned messages and every reference. Plus a manifest with the SHA-256 of each file and an Ed25519 signature over it. The recipient verifies with the public key on logistixpert.com that it is ours and unchanged.

10 Findings in context

A finding is only useful once you can retrace it. Every entry leads to the message that triggered it.

The spot

The finding names check, severity, field path and the message it occurred in. That message is shown with the field highlighted, not as a line number.

The context

Previous, current and next message sit side by side and scroll in step. Switchable between the vehicle's stream and the log, where the neighbours often belong to another robot.

The view

Filters by severity and by vehicle; grouping by check or by message, depending on whether you are hunting a pattern or a single incident.

The diff

Two consecutive state messages of the same vehicle, field by field, changes highlighted, without laying two JSON documents next to each other by hand.

All Critical Major Minor Warning Info DISPOSITION All Open Dispositioned 2 Waived 1 VEHICLE All LogIQ-QA/lhs_pick_11889 Vertexa/VX_MOVER_0001									
Severity	Rule	ID	Vehicle	Message	Field	Description	Recommendation	Disposition	
Major	VDA	SCHEMA_STATE_VIOLATION	Vertexa/VX_MOVER_0001	141	operating Mode	Schema violation: 'BOGUS' is not one of ['AUTOMATIC', 'SEMIAUTOMATIC', 'MANUAL', 'SERVICE', 'TEACHIN']	Make the state payload conform to the VDA5050 2.1.0 JSON schema.	Accepted	
Minor	VDA	LAST_NODE_REGRESSION_derived	Vertexa/VX_MOVER_0001	190	lastNode SequenceId	lastNodeSequenceId decreased within orderId.	Keep lastNodeSequenceId monotonic for each order.	To fix	
Info	VDA	LAST_NODE_JUMP	Vertexa/VX_MOVER_0001	191	lastNode SequenceId	lastNodeSequenceId jumped by more than 4.	Check whether intermediate node updates were missed.	Accepted - waiver	

The findings table: ruleset badges, filters and the recorded disposition per finding.

11 From finding to acceptance

Every acceptance ends with a list: accepted deviation, false positive, will be fixed. That list used to live in a spreadsheet next to the tool. Now it lives in the report, and none of it ever moves a verdict.

A disposition per finding

Status, mandatory justification and author, right on the row. A new run of the same file inherits the decisions on demand. Whoever carries them over is named next to the original author.

Standing waivers

One rule per check and vehicle for deviations that recur in every recording, with an expiry date and managed on its own page. A single disposition beats any rule.

Compare two runs

Firmware update, new configuration: with the same input, new and resolved findings one by one; with different recordings, the movement per check. The page says which mode applies.

All of it in the package

Dispositions and applied waivers travel in the signed package, marked as statements of the accepting side and covered by the signature.

LogIQ VDA5050
Analyzer - Testbench - Monitoring

Production workspace | a.keller | Northwind Robotics | SUPERUSER | Logout

Compare Runs

#40 (2026-08-03 19:40:00) → #41 (2026-08-04 09:10:00) · capture_hall_a_452.ndjson | All Runs | Open newer report

Run #40 – baseline
2026-08-03 19:40:00

Failed **95.48** Pass custom

0 Critical 4 Major 3 Minor 0 Warning 3 Info

Run #41 – newer
2026-08-04 09:10:00 · score -0.29

Failed **95.19** Pass custom

0 Critical 4 Major 4 Minor (+1) 0 Warning 3 Info

Two runs of the same recording: both verdicts, and only the checks that moved.

12 Layout and visualization

Orders refer to a track layout. When order and layout do not fit, the vehicle rejects. Without a picture the search then takes long.

1

Check the LIF

15 rules per VDMA LIF 1.0.0: schema, node and edge integrity, reachability in the network, station references. The result is its own report with a layout drawing. Layouts in the coming LIF 2.0.0 are already read as a preview.

2

Order over layout

The path of an order is drawn over the imported layout. The comparison reports node by node and edge by edge whether the order fits the layout, and names every missing node and edge when it does not.

3

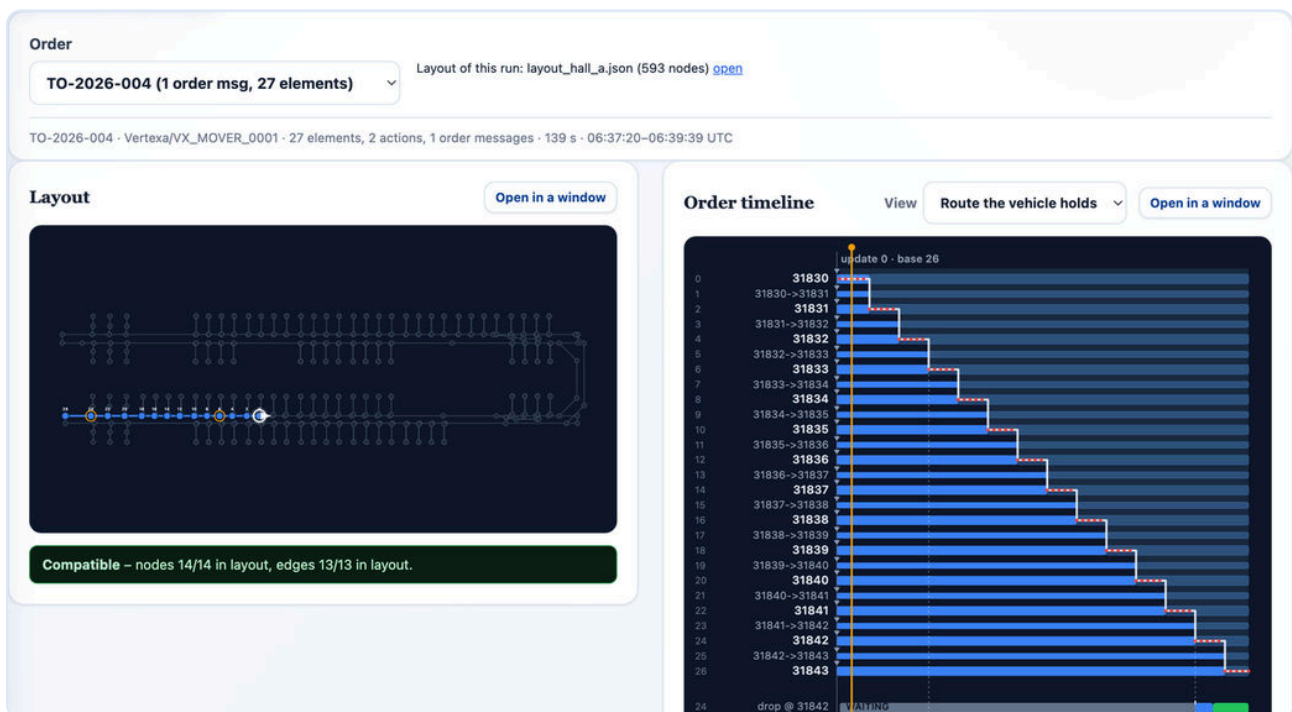
State history

Next to the order path, the visualization plots selected state fields over time: position, battery or operating mode.

4

Stays with the report

The layout is chosen at analysis time and stays attached to the report. Whoever opens it later sees the same basis.



An order path over the LIF layout, with the comparison verdict.

13 Live view

While a profile records, you can watch the site at work, on the same layout the reports use.

- Every vehicle on the LIF layout in its own colour; the last dispatched order as a path, the last reported position as a marker.
- A vehicle table with order, last node, position, battery, state and the age of the report.
- Findings appear as they happen, filterable by severity and vehicle.
- A state diff of a selected vehicle next to it; refresh every second.



Two vehicles with their own routes and last known position.

The actual point

The map draws the order that was sent. The table shows what the vehicle itself reports. When the two disagree, it is right there. That gap is where most commissioning defects live.

FLEET CONTROLLER

Watch your orders arrive during commissioning: target and actual on one map.

VEHICLE MANUFACTURER

Explain your vehicle's behaviour on site without debug access to somebody else's systems.

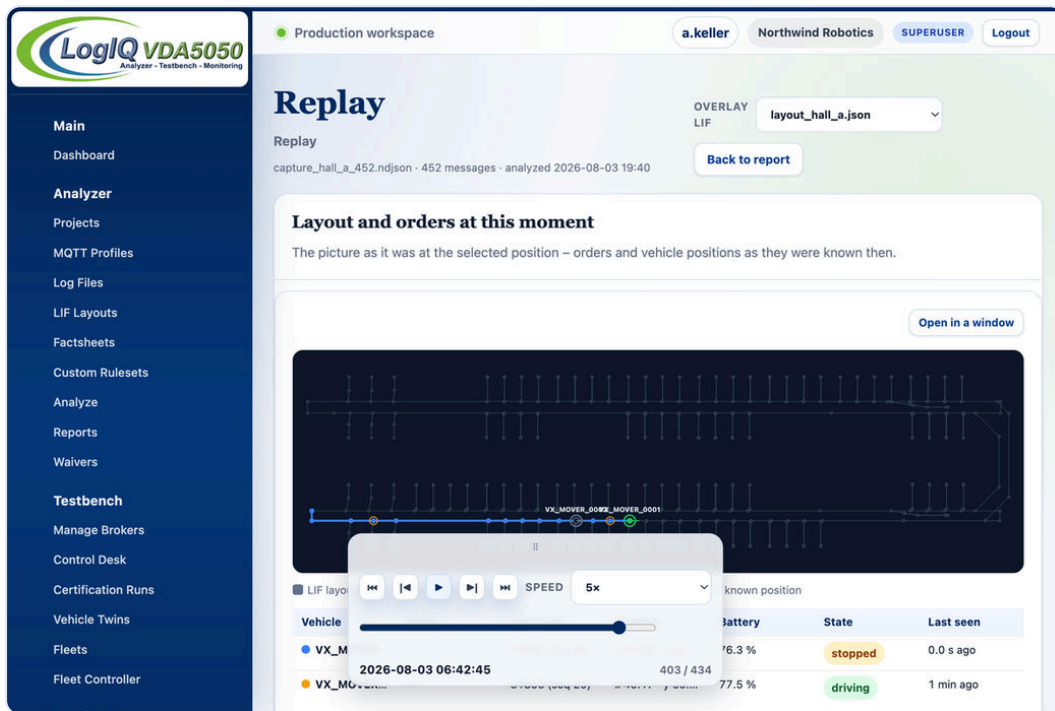
OPERATOR

A view into the running site without touching it.

14 Replay

The same view, but for everything that already happened: finished captures and imported log files can be played back.

- Forward, backward, step by step, up to twenty-fold speed.
- The gaps are the real message gaps, capped so long pauses do not stall a demonstration.
- The findings list shows only what was known at the chosen moment. No knowledge from the future.
- The LIF overlay can be switched without changing the report.



Replay at position 402 of 433, with the findings known up to that moment.

What it is used for

The debrief. An incident can be played to the supplier instead of described; everybody in the room sees the same sequence, with the same findings in the same places.

15 What the fleet did with its time

Next to the conformance verdict, the same file answers an operational question: how much time was each vehicle available, working, in manual mode, in error?

1

Five states, no overlap

Manual, Error, Active, Idle and, where it occurs, Other. Exactly one at any moment, so the shares add up to the span and can be held against each other.

2

Availability without gloss

A warning is not an outage. A vehicle reporting an unsafe pose while it keeps working is available. Everything above that counts.

3

Active means: it works

Not: it drives. A vehicle standing at a station picking is neither travelling nor idle.

4

Any window

A shift, an hour, the half hour before the standstill. The window is cut before counting, not filtered afterwards.

Vehicle	Recorded span	Availability	Error periods	Manual	Error	Active	Idle	Other	Orders	Finished	Cancelled	Open
vertica/VX_MOVE_0001	2026-08-03 08:30:00 - 2026-08-03 06:45:15	100.0%	0	0.0%	0.0%	71.5%	28.3%	0.2%	6	5	0	1
vertica/VX_MOVE_0002	2026-08-03 08:42:08 - 2026-08-03 06:47:26	100.0%	0	0.0%	0.0%	0.0%	100.0%	0.0%	0	0	0	0

One shift, five vehicles: the shares per vehicle, and under the table every error period with its beginning and end.

What it is used for

The conversation after the shift. Not “the site ran badly” but: this vehicle spent 62 % of the hour in error, that one 21 % in manual, with the error list and the minute it began right next to it.

FLEET CONTROLLER

Throughput arguments with numbers: the fleet's waiting time is measured, not felt.

VEHICLE MANUFACTURER

Proof that the vehicle worked, when it stood still and because of whom.

OPERATOR

Availability per vehicle and shift, from the same data as conformance.

16 One order, one vehicle, sent now

Everything so far reads traffic that already happened. The testbench is the other direction: it plays the fleet controller itself, sends real VDA5050 orders and watches a vehicle answer - against a real vehicle on a real broker, or against the built-in virtual one.

1

By hand, before anything is planned

Pick a connection, build an order or an instant action on its layout, send it, watch what comes back. The page for the question whether this vehicle does this at all, before anybody plans a run that assumes it does.

2

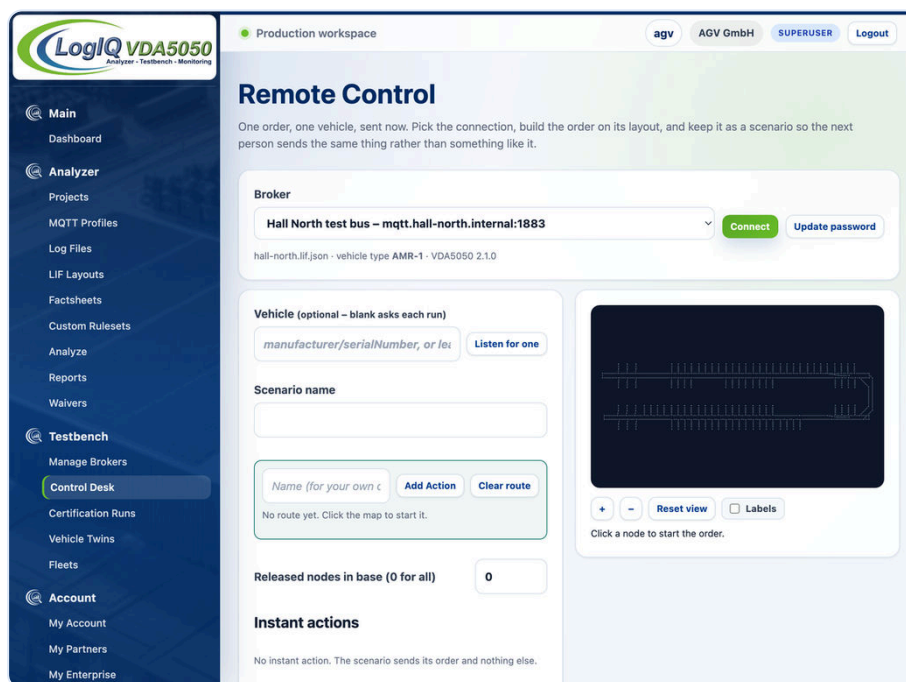
What is offered is what the vehicle can do

Instant actions come from the vehicle's own factsheet where one is known. What you can send is what the vehicle said it supports rather than what the standard allows in general.

3

Stored once, never asked again

A connection carries address, port, credentials, layout, vehicle type and version. The layout is not optional: without nodes there is nowhere to send a vehicle, so a broker without a layout cannot carry a run.



The control desk: an order built on the chosen broker's layout.

FLEET CONTROLLER

Try a new vehicle out before an integration date is set.

VEHICLE MANUFACTURER

Check that your own implementation answers an order the way you meant it to.

OPERATOR

Address one machine deliberately, without going through the fleet controller.

17 A certification run is a document before it is a drive

An ordered list of segments, each carrying the scenarios it exists to prove, built from your own layout and your own vehicle. What separates it from a sequence of orders is that it has to provoke.

1 A document before it is a drive

A certification run is an ordered list of segments, each carrying the scenarios it exists to prove, built from your layout and your vehicle. It has to provoke: hold a horizon back, cancel a running order, ask for a factsheet. It never sends anything the standard forbids.

2 The matrix says in advance what can be shown

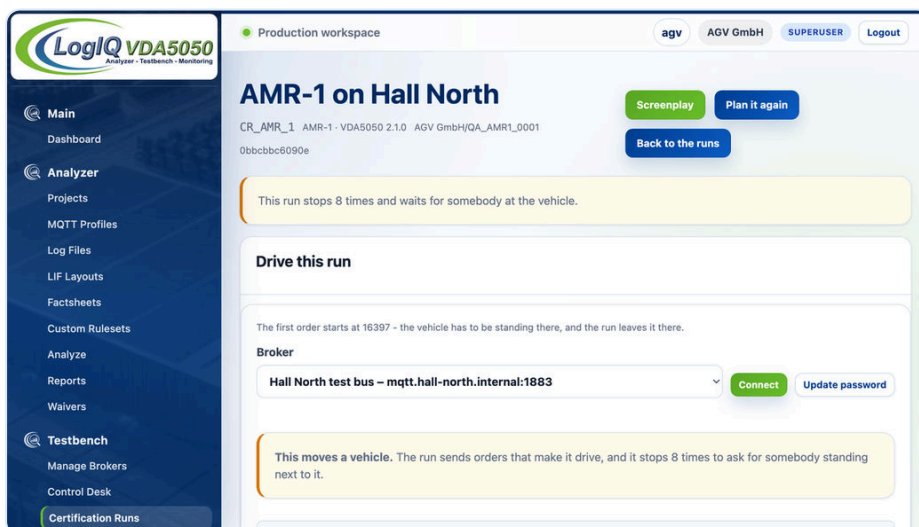
Per scenario it answers whether this layout and this vehicle can show it at all, with the reason. Two of the seven gates cannot be sent by any fleet controller: an emergency stop is pressed and a fault provoked. Those become stops.

3 Twins and fleets

A twin is a behaviour model of a real vehicle, measured from a recording of it rather than typed, and checkable against the recording it came from. A fleet puts such vehicles on a broker under their makers' names for a vendor to drive their own software against.

4 What a figure will carry

Vehicle-intrinsic figures are medians over traversals that were never held up and can be compared. System-level figures carry the layout and the dispatcher with them; the run page says which of the two every figure is.



A certification run: what it drives, and where it waits for somebody at the vehicle.

FLEET CONTROLLER

Drive new vehicle types against your own fleet controller without building a test rig.

VEHICLE MANUFACTURER

Put your vehicle through a planned proof before your customer does.

OPERATOR

Compare two fleets under identical work before committing to a purchase.

18 A twin is measured, not typed

A vehicle twin is a behaviour model of one real machine: how fast it drives, how long it takes over a node, what its states say on the way. Measured from a recording of that vehicle - or uploaded as a twin document.

1 A twin nobody has checked is a claim

So a measured twin can be checked: a short plan is built over a layout you hold, driven against the twin, and what it does is compared with the recording it came from.

2 One figure with the figures behind it

The result is a percentage, and the readings that could not be compared are named rather than passed over. What could not be checked stands where a gap would otherwise be.

3 Everything downstream stands on it

A fleet of twins is only worth driving against if the twins behave like the machines they stand for.

LogiQ VDA5050
Analyzer - Testbench - Monitoring

Production workspace

agv AGV GmbH SUPERUSER Logout

Vehicle twins

This vehicle, not a vehicle: what a maker's AGV declares, how long its actions take and where it departs from the specification. Yours to write, your partners' to drive against.

PROJECT All projects

Upload new twin Create new twin from recording

AGV GmbH

Twin	From	Series	Firmware	Version	Measured	Fidelity	Shared with	Project
☐ AMR_2	yours	AMR-2	3.2.1	2.1.0	unmeasured	not checked	private	Hall North co... ⌵
Hall North AMR-2 (aisle unit)								
☐ AMR_1	yours	AMR-1	3.2.1	2.1.0	unmeasured	not checked	private	Hall North co... ⌵
Hall North AMR-1 (racking unit)								

Twins: vendor, series, firmware - and the fidelity reached.

FLEET CONTROLLER

Rely on the fleet your software is being driven against.

VEHICLE MANUFACTURER

Measure your vehicle once and use it as often as you like afterwards.

OPERATOR

See how an offered vehicle type actually behaves, before procurement.

19 A fleet stands ready for somebody else to drive

A fleet is a name and a list of twins with how many of each. Starting one puts those vehicles on a broker under their makers' names and leaves them there. A fleet-controller vendor points their own software at them and drives; we command nothing.

1

A broker per fleet

Where a broker service is configured, a fleet asks for a broker of its own and gives it back a quarter of an hour after the last connection closes. Two fleets that must not see each other are never put on the same one.

2

Recorded without remembering to

Starting a fleet also starts a capture of it, named after the fleet, and stopping the fleet stops the capture. What the other side drove is an analysable run afterwards.

3

Whole or not at all

If one of the twins is no longer shared with you, the start is refused with the reason rather than half a fleet appearing on the bus. The identities those vehicles publish are yours to choose.

LogiQ VDA5050
Analyzer - Testbench - Monitoring

Production workspace agv AGV GmbH SUPERUSER Logout

Fleets

A fleet is a name and a list of twins with how many of each. Starting one puts those vehicles on a broker under their makers' names; point a fleet controller at them and drive.

PROJECT All projects

Put a new fleet together

Fleet	Vehicles	Layout	Made of	Who may start it	Project	On a broker
☐ Aisle fleet B	6	hall-north.lif.json	2 x AMR_1 4 x AMR_2	yours alone	none	▶ ✎ ◀
☐ Racking fleet A	6	hall-north.lif.json	4 x AMR_1 2 x AMR_2	yours alone	none	▶ ✎ ◀

External broker

There is nowhere for a fleet to stand: no broker service is set up and no broker is configured in the system settings. An administrator does one or the other – or you name a broker of your own below.

Assign a fleet to an external Broker

Fleets: which twins, how many of each, over which layout.

FLEET CONTROLLER

Drive your fleet controller against a mixed fleet that does not exist yet.

VEHICLE MANUFACTURER

Put your vehicle at a vendor's disposal without shipping hardware.

OPERATOR

Try a planned fleet size before buying it.

20 Watching without end: the connector

A capture profile records a shift and stops. A customer installation does not, and neither does the question you are being paid to answer about it. A connector holds one broker connection open indefinitely, returns by itself after an interruption and starts with the service.

1 The gaps are the measurement

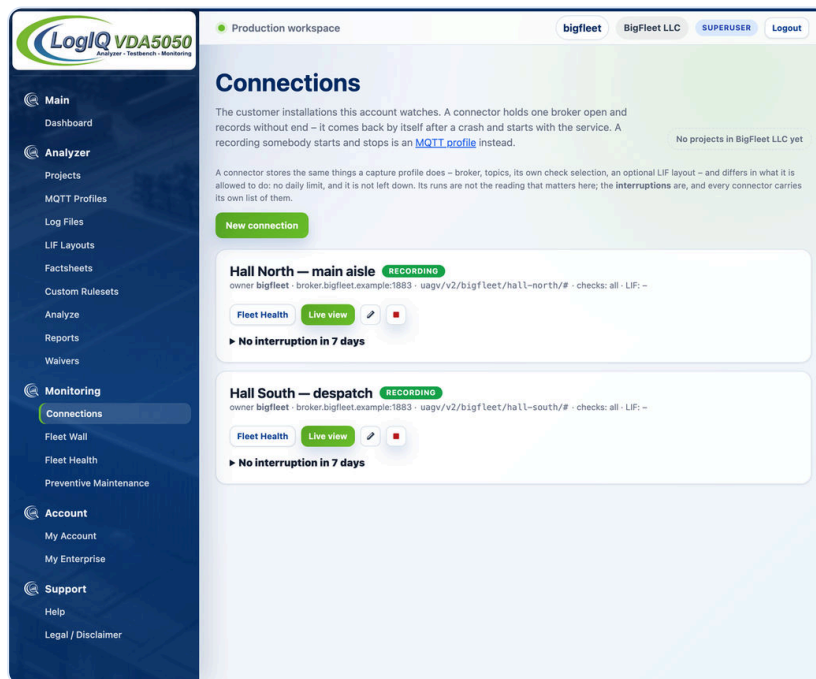
A connector's runs break where the connection broke, so the number of them says more about the network than about the fleet. What is listed instead is every interruption between two runs of one recording, with how long it lasted. A recording somebody stopped and started by hand is a new recording, not an interruption.

2 Quarter-hour cells rather than traffic

As it records, every state message is folded into cells per vehicle: how long the machine drove, how long it stood, what it reported, where, how fast, where it braked. That is why the pages show months without holding months of traffic. A few hundred megabytes carry years.

3 Measured, not declared

A connector is billed as monitoring on every day it records; a capture profile only on a day it ran longer than the configured threshold. The difference is measured rather than asserted.



Connections: what runs permanently and where the connection was gone.

FLEET CONTROLLER

Keep several customers' installations in view without staffing each one.

VEHICLE MANUFACTURER

See how your own vehicle behaves in the field over months.

OPERATOR

Have your own site measured continuously instead of by impression.

21 Fleet Health: the machines, not the protocol

The same recording, a third question: not whether the messages are right and not where the time went, but how the vehicles themselves are doing, read out of the traffic every compliant vehicle already publishes.

1

The odometer nobody exposes

Distance, load cycles, hours driven against hours recorded. Service intervals in usage rather than in weeks.

2

The battery, before it fails

Charge cycles, the deepest discharge and the sag between resting voltage and voltage under load, a proxy for internal resistance.

3

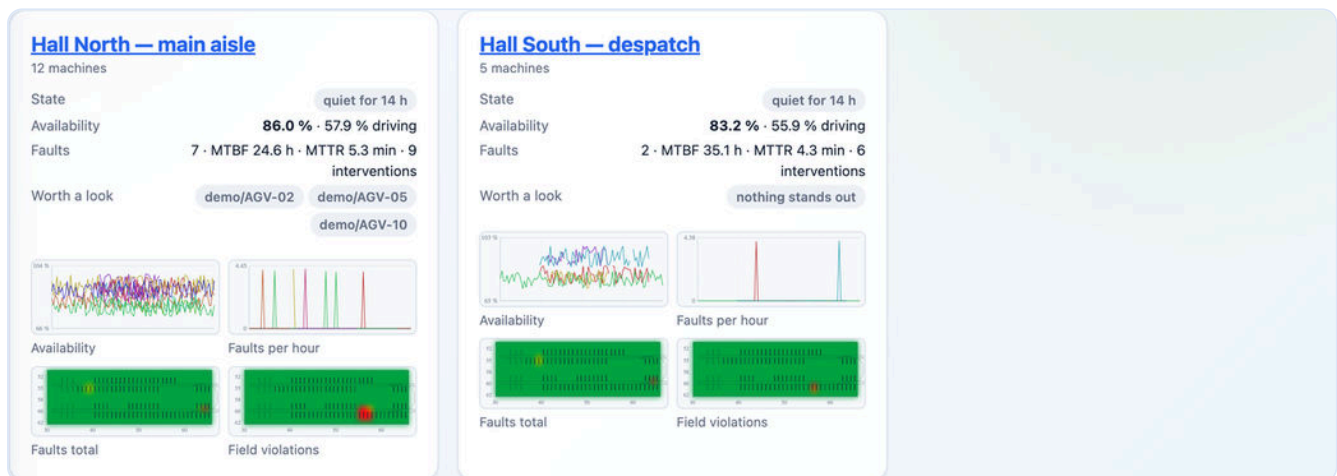
Near misses

Every protective-field violation and every emergency stop as an event. Nothing else in the plant records one.

4

Faults and what they cost

MTBF and MTTR per vehicle, and over the fleet a pareto of the error types, commonest first.



The fleet wall: every fleet with the same two pictures, redrawn every 15 minutes: availability as a line per machine, faults as a map of the building. The page answers one question: which fleet wants looking at?

FLEET CONTROLLER

Evidence of how often a vehicle type really stood still in the field, and on what.

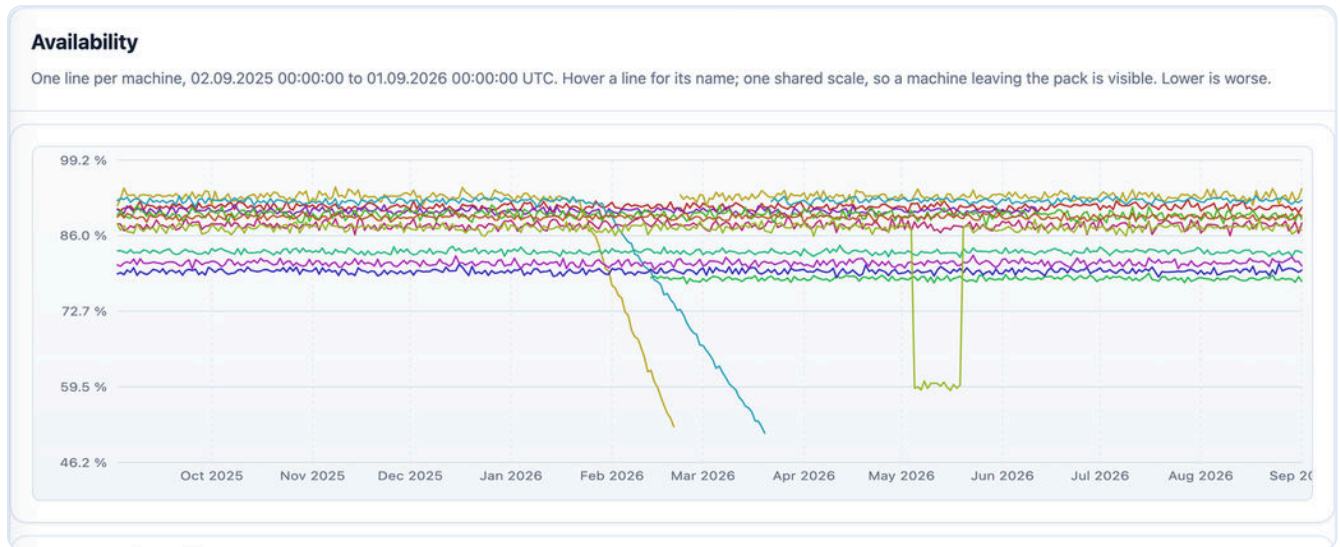
VEHICLE MANUFACTURER

Availability and fault picture per vehicle, from the same data as conformance.

OPERATOR

Maintenance by usage rather than by calendar, and the near misses on record.

21 Fleet Health: the lines and the building



One line per machine on one shared scale, over a year. A machine leaving the pack is visible before anybody goes looking for it, and so is the service day after which it comes back.



Where in the building: protective-field violations per square metre, drawn on the layout the fleet actually drives. A bad junction gets a name, not just a coordinate.

What it is used for

Acceptance, and planning the maintenance: the report says which maintenance-relevant fields a fleet sends at all, written while there is still something to hold back.

VDA5050 carries no temperatures, no motor currents and no wear values. This is measured operating statistics with named assumptions, not condition monitoring, and not sold as it.

22 Preventive maintenance

What is wearing out, read from how the machines behaved. VDA5050 carries no temperatures, no motor currents and no wear values, so this says that a machine is getting worse and roughly how fast, never which bearing.

1 Counters from the last service

Driving hours, distance, load cycles, charge cycles, rotation. Nothing in the traffic marks a service, so a service is entered by hand. That is the one thing this page cannot learn by watching.

2 Trends against the machine's own past

A reading whose recent quarter has left the spread of that machine's own earlier weeks. Voltage sag under load rises with internal resistance, months before usable capacity visibly drops. Interventions per hour find the fault that produces no error code at all, because a person is quietly compensating for it.

3 Warning to failure

The interval between the first warning of a type on a machine and the first failure of the same type on the same machine, learned from the whole recorded history. A warning stays open until that happens. Recording a service does not close it, because nothing in the traffic connects the two.

Machine	Driving hours h	Distance km	Load cycles	Charge cycles	Rotation turns	Counting from
demo/AGV-05	5,015 125 %	23,707 158 %	103,163 115 %	1,197 100 %	358,947 120 %	02.09.2025 first cell – not a service
demo/AGV-09	5,030 126 %	21,209 141 %	103,482 115 %	1,174 98 %	359,505 120 %	02.09.2025 first cell – not a service
demo/AGV-10	4,683 117 %	17,996 120 %	95,066 106 %	1,229 102 %	335,219 112 %	02.09.2025 first cell – not a service
demo/AGV-01	3,422 86 %	14,752 88 %	70,675 78 %	796 68 %	245,785 82 %	02.09.2025 first cell – not a service
demo/AGV-03	2,641 66 %	11,551 77 %	54,191 60 %	580 48 %	189,516 63 %	02.09.2025 first cell – not a service
demo/AGV-11	3,052 76 %	8,342 56 %	61,597 68 %	788 66 %	217,785 73 %	02.09.2025 first cell – not a service

Preventive maintenance: counted since the last recorded service.

FLEET CONTROLLER

Tell the customer which machine needs attention next.

VEHICLE MANUFACTURER

Learn from the field how long a warning on your series is worth.

OPERATOR

Plan maintenance by use rather than by calendar.

23 Unattended operation and reporting

A capture across a weekend brings a hundred thousand messages. Nobody reads those, so the analyzer speaks up by itself.

1

Report on a schedule

Every X hours, a digest of what stood out since the last report. A quiet period is reported too, so silence stays unambiguous.

2

Alert on a threshold

Or only once enough has accumulated: from X findings, optionally Critical only, Critical and Major, or all severities.

3

Recipients per profile

The addresses hang on the capture profile. Different sites can have different distribution lists.

4

Resumable

A restart picks the analysis up where it stood. Nothing is judged twice, nothing skipped.

What the email says

Period, trigger and the number of new findings, broken down by severity, by vehicle and by check, with a direct link into the running capture.



Excerpt of a threshold alert.

24 Teams, projects and plans

Test results do not belong on one laptop. The analyzer knows groups, projects and clear permissions.

1 Enterprise

All members see what the group has produced; every list names the owner. Shared visibility is not shared deletion: only superusers may delete other people's data.

2 Superuser

Manages membership and projects and may delete others' data. Deleting accounts stays with the administration.

3 Fallback superuser

Exactly one per enterprise. When a member closes their account, their logs, layouts, reports and captures pass to this person instead of being lost with the account.

4 Projects

Logs, layouts, profiles and reports can be assigned to a job and filtered by it. A completed project disappears from the lists without being deleted, and returns when reopened.

5 Plans

Different plans are available for all customer types, from fixed packages to SaaS monitoring.

The screenshot shows the LogiQ VDA5050 administrator interface. The sidebar on the left contains navigation links: Main, Dashboard, Analyzer, Projects, MQTT Profiles, Log Files, LIF Layouts, Factsheets, Custom Rulesets, Analyze, Reports, Waivers, Testbench, Manage Brokers, Control Desk, Certification Runs, Vehicle Twins, Fleets, Fleet Controller, Monitoring, Connections, Fleet Wall, Fleet Health, and Preventive Maintenance. The main content area is titled 'All Runs' and shows a search bar with the placeholder 'File, user, email, enterprise or project...'. Below the search bar, there are two sections: 'Halberg Logistics' and 'Northwind Robotics'. Each section displays a summary of runs and a table of run details. The 'Halberg Logistics' section shows one entry for user 't.brandt' with a LIF file 'layout_hall_b.json'. The 'Northwind Robotics' section shows two entries for user 'a.keller' with reports 'capture_hall_a_452.ndjson' and 'acceptance_run_final.ndjson'.

Kind	Name	Detail	Project	Created
LIF	layout_hall_b.json	64 nodes · 70 edges · score 96.0	-	2026-07-28 08:00:00
REPORT	capture_hall_a_452.ndjson	452 messages · score 93.1 2 3 5	Hall A commissioning	2026-08-04 09:10:00
REPORT	acceptance_run_final.ndjson	359 messages · score 88.0 0 1 4	Hall A commissioning	2026-08-03 17:02:00

The administrator's view across enterprises, users and artefacts, with full-text search.

25 Operations and security

Recordings from a site are sensitive. The analyzer is built so they never have to leave the plant.

In your own house

Docker Compose, one file, one command. On your own hardware, in your own network or as a hosted instance. The choice changes nothing about the feature set.

Encrypted connection

MQTT over TLS, optionally with the site's own CA certificate. The mirror target has the same switch. Otherwise the copy would be protected worse than the original.

Credentials encrypted

Broker passwords are stored encrypted, not in plain text. They stay usable for reconnects but cannot be read.

No phoning home

In your own network the installation carries a signed licence file, checked locally. After that, fourteen days of grace, then read-only. Everything recorded stays reachable.

Reports under your own name

With Unlimited, report, export and email report carry the customer's name and logo. The note naming the tool behind them stays underneath.

Addresses nobody can enumerate

Reports, runs, imports and layouts hang on identifiers that cannot be guessed. Every access additionally checks who owns the entry.

Roles with limits

Member, superuser, administrator. Who may delete what is defined in one place in the code and checked on every action.

Closing an account

Users can delete their own account. Without an enterprise, everything goes with it; in a group, the work stays with the group and passes to the fallback superuser.

Proven quality

3774 automated tests run before every release, including regression tests against fixed sample data.

Data and retention

Deleting a log does not delete the reports made from it: a report is the record of a test run and outlives its source. Retention periods for generated source files, uploads and reports are set per installation by the administration.

26 Service and support

A checking tool is only as good as the version of the standard it checks against and as the way to reach us when something is wrong.

The standard does not stand still

VDA5050 has reached 3.0 and will go further; M2X and VDMA LIF are still being written. When the committee changes something, we build it in, and an installation with a running licence gets it as an update rather than a new purchase.

Checks without a new image

The check catalogue, the profiles and the schemas travel in a signed package that is applied in the field. A tightened check therefore reaches an installation that is not rebuilt with every release.

Hosted is already up to date

On our instance there is nothing for you to do. For an installation of your own we ship a new sealed image, and your licence file stays exactly as it is.

One case, not a mail thread

Every report becomes a ticket in the portal with a number, a history and a state. Read it back later and the question, the answer and the attachments are in one place instead of across three mailboxes.

Mail tells you, nothing more

Your inbox gets a number, a subject and a link. The case itself, the attachments and the report package stay in the portal, and you fetch them there over an encrypted connection.

Report it without leaving the tool

»Report a problem« sends the version and the installation along, so nobody has to ask. The signed report package comes too if you want it: on our hosted service the box is already ticked, on your own installation you tick it yourself.

Why the route matters

A recording from your plant shows which vehicles went where and when. That is operational data, and mail is the wrong way to move it: a message rests on other people's servers along the way and then sits in a mailbox nobody opens again. So the only thing mail carries is word that something is waiting for you.

27 Technical data

Specifications	VDA5050 1.1–3.0 · M2X 0.9.0 (preview) · VDMA LIF 1.0.0 and 2.0.0 (preview)
Checks	250 for VDA5050 and M2X, 15 for LIF; individually selectable, per analysis, per version and per capture profile
Testbench	Certification runs, measured twins, fleets on a broker; a virtual vehicle for the dry run
Monitoring	Connectors with no daily limit, quarter-hour cells per vehicle, fleet wall, fleet health with the building map, preventive maintenance
Requirements	Requirement files, check profiles and layout rules per vendor; partners measure against them and hand reports back
Custom rulesets	Descriptive JSON against a published schema, seven rule types, up to 200 checks per file; judged separately (from the Project plan)
Acceptance	A disposition per finding, standing waivers per check and vehicle, carry-over into repeat runs, comparison of two runs; verdicts stay untouched
Message kinds	order, state, connection, instantActions, factsheet, visualization; M2X: load handling, door, elevator, signal light, charger, transport order
Input formats	.log, .txt, .json, .ndjson: raw recordings, MQTT captures and NDJSON exports
Output	One signed package per report: JSON and Markdown, the source file, the cleaned messages, all references, a manifest with SHA-256 and an Ed25519 signature
Live operation	MQTT 3.1.1/5, plain or over TLS (own CA possible), several profiles at once, optional mirroring to a second broker
Notification	Email via SMTP: capture reports on a schedule or a threshold, and on request one message per finished analysis
Operation	Docker Compose (application, analysis workers, PostgreSQL); hosted or as an installation of your own; analyses run through a queue, oldest first
Usage	Web application, nothing installed on the workstation; interface in English
Data storage	Files in one data directory, the database in PostgreSQL; retention periods configurable per installation

27 **Technical data: what it needs and what it will not claim**

Prerequisites

- A machine or VM with Docker; enough disk for long-running captures.
- Read access to the site's MQTT broker; mirroring additionally needs write access to the target broker.
- An SMTP account for email reports; without one, mails are written to a file.

Limits and assumptions

- The analyzer judges what it sees. Messages that never reach the broker cannot be judged.
- A replay shows up to 20,000 messages; longer runs are shortened, and the tool says that it shortened them.
- M2X follows a preview of the specification; results are marked as preview.
- Attribution to vehicles needs manufacturer and serial number in topic or message.

28 Check catalogue

Every check in the tool, grouped by area. Identifiers and short texts match the application and the report; each check there also cites its clause in the specification.

Behind every identifier stands the VDA5050 versions the check applies to. “1.1–3.0” means all four supported versions. A check with no version judges another document: M2X or the layout.

The shape of a message: schema, structure, header and time

Schema Conformance (8)

[SCHEMA_STATE_VIOLATION](#)

1.1–3.0 State message conforms to the VDA5050 JSON schema

[SCHEMA_ORDER_VIOLATION](#)

1.1–3.0 Order message conforms to the VDA5050 JSON schema

[SCHEMA_CONNECTION_VIOLATION](#)

1.1–3.0 Connection message conforms to the VDA5050 JSON schema

[SCHEMA_INSTANTACTIONS_VIOLATION](#)

1.1–3.0 instantActions message conforms to the VDA5050 JSON schema

[SCHEMA_VISUALIZATION_VIOLATION](#)

1.1–3.0 Visualization message conforms to the VDA5050 JSON schema

[SCHEMA_ZONESET_VIOLATION](#)

3.0 zoneSet message conforms to the VDA5050 JSON schema

[SCHEMA_RESPONSES_VIOLATION](#)

3.0 responses message conforms to the VDA5050 JSON schema

[SCHEMA_FACTSHEET_VIOLATION](#)

2.0 2.1 3.0 Factsheet conforms to the VDA5050 JSON schema

Structural Checks (6)

[STRUCT_REQUIRED_MISSING](#)

1.1–3.0 Required fields are present

[STRUCT_HEADER_ID_TYPE](#)

1.1–3.0 headerId is an integer

[STRUCT_TIMESTAMP_FORMAT](#)

1.1–3.0 timestamp is ISO-8601 UTC

[STRUCT_VERSION](#)

1.1–3.0 version matches the selected target

[STRUCT_VERSION_FORMAT](#)

1.1–3.0 version is semantic [Major].[Minor].[Patch]

[STRUCT_ARRAY_TYPE](#)

1.1–3.0 state collections are arrays

Identity Checks (1)

[IDENTITY_VERSION_CHANGED](#)

1.1–3.0 One vehicle keeps one version

Header / Time Checks (4)

[HEADER_DUPLICATE](#)

1.1–3.0 No duplicate headerId values

[HEADER_REGRESSION](#)

1.1–3.0 headerId increases monotonically

[HEADER_GAP](#)

1.1–3.0 headerId leaves no gaps

[TIMESTAMP_REGRESSION](#)

1.1–3.0 timestamp increases or stays equal

The state of the vehicle: nodes, actions, driving, safety

Node / Edge Checks (9)

[STATE_ENTRY_TYPE](#)

1.1–3.0 nodeStates/edgeStates entries are objects

[STATE_ENTRY_INCOMPLETE](#)

1.1–3.0 nodeId/edgeId and sequenceId are present

[STATE_SEQUENCE_TYPE](#)

1.1–3.0 sequenceId is an integer

[STATE_SEQUENCE_REGRESSION](#)

1.1–3.0 sequenceIds increase within a message

[NODE_EDGE_SEQUENCE_OVERLAP](#)

1.1–3.0 Nodes and edges do not share a sequenceId

[LAST_NODE_REGRESSION](#)

1.1–3.0 lastNodeSequenceId does not decrease per orderId

[LAST_NODE_JUMP](#)

1.1–3.0 lastNodeSequenceId does not jump by more than 4

[STATE_RELEASED_TRAJECTORY_CHANGED](#)

1.1–3.0 A released trajectory is not replanned

[STATE_IDLE_INCONSISTENT](#)

1.1–3.0 A robot without an order reports nothing to do

Action Lifecycle Checks (7)

[ACTION_ENTRY_TYPE](#)

1.1–3.0 actionStates entries are objects

[ACTION_REQUIRED_MISSING](#)

1.1–3.0 actionId and actionStatus are present

[ACTION_STATUS_INVALID](#)

1.1–3.0 actionStatus uses an allowed enum value

[ACTION_INVALID_TRANSITION](#)

1.1–3.0 Action transitions are allowed

[ACTION_NOT_TERMINAL](#)

1.1–3.0 Actions reach a terminal state

[ACTION_BLOCKING_VIOLATED](#)

1.1–3.0 A blocking action runs alone

[ACTION_BLOCKING_DRIVING](#)

1.1–3.0 The robot stands still for a blocking action

Operating Mode Checks (1)

[STATE_OPERATING_MODE_INVALID](#)

1.1–3.0 operatingMode uses an allowed enum value

Driving / Velocity Checks (1)

[STATE_NOT_LOCALIZED_NO_ERROR](#)

3.0 Losing localization is reported as an error

The state of the vehicle: nodes, actions, driving, safety (continued)

Driving / Velocity Checks (7)

[STATE_DRIVING_WHILE_NOT_LOCALIZED](#)
3.0 A robot that lost localization does not drive

[STATE_DRIVING_WITH_BLOCKING_ERROR](#)
3.0 A blocking error stops the order

[PAUSED_AND_DRIVING](#)
1.1–3.0 paused=true and driving=true are not both active

[VELOCITY_MISSING_WHILE_DRIVING](#)
1.1–3.0 Velocity is present when driving=true

[DRIVING_ZERO_VELOCITY](#)
1.1–3.0 driving=true comes with a velocity above zero

[NOT_DRIVING_WITH_VELOCITY](#)
1.1–3.0 driving=false comes with no velocity

[STATE_SHARED_PATH_NOT_AT_POSITION](#)
3.0 A shared path starts where the robot is

Battery Checks (3)

[BATTERY_CHARGE_RANGE](#)
1.1–3.0 battery charge is between 0 and 100

[POWER_RANGE_INCONSISTENT](#)
1.1–3.0 The estimated reach follows the state of charge

[BATTERY_CHARGE_JUMP](#)
1.1–3.0 Battery charge moves in steps a battery can take

Safety Checks (4)

[SAFETY_FIELD_MISSING](#)
1.1–3.0 safetyState contains the eStop and fieldViolation fields

[SAFETY_ESTOP_VALUE_INVALID](#)
1.1–3.0 emergency-stop value uses an allowed enum

[ESTOP_WHILE_DRIVING](#)
1.1–3.0 The emergency stop is clear while driving

[FIELD_VIOLATION_WHILE_DRIVING](#)
1.1–3.0 No field violation while driving

Error Reporting Checks (6)

[STATE_ERROR_LEVEL_MISMATCH](#)
1.1–3.0 A predefined error is reported at the level the standard gives it

[STATE_ERROR_REFERENCE_MISSING](#)
3.0 A predefined error names what it was about

[STATE_ERROR_TYPE_CASE](#)
3.0 errorType is written as an enumeration (3.0)

[STATE_ERROR_STRUCTURE](#)
1.1–3.0 errors contain errorType and a valid errorLevel

[STATE_VEHICLE_REPORTS_ERRORS](#)
1.1–3.0 Errors the vehicle reported about itself

[MISSION_ORDER_CLEARED_BY_AN_ERROR](#)
3.0 An error is not a reason to throw the order away

Orders: structure, updates, graph, trajectory

Order Structural Checks (6)

[ORDER_REQUIRED_MISSING](#)
1.1–3.0 Required order fields are present

[ORDER_HEADER_ID_TYPE](#)
1.1–3.0 Order headerId is an integer

[ORDER_TIMESTAMP_FORMAT](#)
1.1–3.0 Order timestamp is ISO-8601 readable

[ORDER_VERSION](#)
1.1–3.0 Order version matches the selected target

[ORDER_ARRAY_TYPE](#)
1.1–3.0 nodes and edges are arrays

[ORDER_MIN_ONE_NODE](#)
1.1–3.0 An order carries at least one node

Order Update Checks (8)

[ORDER_UPDATE_STITCH](#)
1.1–3.0 An order update begins where the vehicle can reach

[ORDER_SEQUENCE_STABILITY](#)
1.1–3.0 A released sequencId keeps naming the same element

[ORDER_UPDATE_DUPLICATE](#)
1.1–3.0 No duplicate orderUpdateId values

[ORDER_UPDATE_REGRESSION](#)
1.1–3.0 orderUpdateId increases monotonically

[ORDER_UPDATE_GAP](#)
1.1–3.0 orderUpdateId leaves no gaps

[MISSION_DUPLICATE_ORDER_ACCEPTED](#)
1.1–3.0 An order resent unchanged is ignored, not executed again

[MISSION_DUPLICATE_ORDER_NOT_REPORTED](#)
3.0 Ignoring a resent order is announced

[MISSION_ORDER_UPDATE_AFTER_CANCEL](#)
3.0 A cancelled order gets no more updates

Order Graph Checks (8)

[ORDER_NODE_EDGE_CARDINALITY](#)
1.1–3.0 Edge count is one below node count

[ORDER_SEQUENCE_TYPE](#)
1.1–3.0 node and edge sequencId values are integers

[ORDER_SEQUENCE_PARITY](#)
1.1–3.0 Node and edge sequencId parity is valid

[ORDER_SEQUENCE_ORDER](#)
1.1–3.0 Nodes and edges are sorted by sequencId

[ORDER_DUPLICATE_ID](#)
1.1–3.0 sequencId and actionId are unique

[ORDER_EDGE_ENDPOINT_UNKNOWN](#)
1.1–3.0 Edges reference existing nodes

[ORDER_GRAPH_CHAIN](#)
1.1–3.0 Graph sequence is continuous

[ORDER_RELEASED_CONTIGUITY](#)
1.1–3.0 Released base is contiguous

Order Position Checks (1)

[ORDER_POSITION_MISSING](#)
1.1–3.0 Node positions contain coordinates and mapId

Order Action Checks (3)

[ORDER_ACTION_REQUIRED_MISSING](#)
1.1–3.0 Order actions contain required fields

[ORDER_BLOCKING_TYPE](#)
1.1–3.0 blockingType uses allowed values

[ORDER_ACTION_TYPE_NOT_PREDEFINED](#)
1.1–3.0 Order actions use a predefined actionType

Order Trajectory Checks (5)

[ORDER_TRAJECTORY_STRUCTURE](#)
1.1–3.0 Edge trajectory structure is valid

[ORDER_CORRIDOR_STRUCTURE](#)
2.1 3.0 Edge corridor structure is valid

[ORDER_CORRIDOR_ZERO_WIDTH](#)
2.1 3.0 A corridor of zero width is left out instead

[ORDER_CORRIDOR_REF_POINT](#)
2.1 3.0 The corridor reference point uses an allowed value

[ORDER_CORRIDOR_RELEASE_BEHAVIOR](#)
3.0 releaseLossBehavior uses an allowed value

Maps, zones and requests

Zone Checks (4)

[ZONE_SET_ENTRY_INCOMPLETE](#)
3.0 Every reported zone set names itself, its map and its status

[ZONE_SET_STATUS_INVALID](#)
3.0 zoneSetStatus uses an allowed enum value

[ZONE_SET_ENABLED_DUPLICATE](#)
3.0 At most one zone set is enabled per map

[ZONE_SET_ID_DUPLICATE](#)
3.0 A zoneSetId is reported once

Maps, zones and requests (continued)

Zone Checks (10)

[ZONE_DIRECTED_AGAINST_TRAVEL](#)
3.0 A directed zone is not driven against

[ZONE_BIDIRECTED_CROSSED](#)
3.0 A bidirected zone is driven along, not across

[ZONE_BLOCKED_ENTERED](#)
3.0 A blocked zone is not driven into

[ZONE_BLOCKED_NOT_REPORTED](#)
3.0 Being in a blocked zone is said out loud

[ZONE_DIRECTION_ZONES_OVERLAP](#)
3.0 Zones that name a direction do not overlap

[ZONE_SPEED_LIMIT_EXCEEDED](#)
3.0 A speed limit zone is driven at its speed

[MISSION_ZONE_REQUEST_IN_A_MODE_THAT_CLEARS_THEM](#)
3.0 A mode that ends fleet control ends its zone requests

[ZONE_SET_NOT_ADOPTED](#)
3.0 A transferred zone set reaches the state

[ZONE_SET_ENABLED_ON_ARRIVAL](#)
3.0 A new zone set arrives disabled

[ZONE_SET_DUPLICATE_NOT_REPORTED](#)
3.0 A zone set that is already known is refused out loud

Map Checks (10)

[MAP_ENTRY_INCOMPLETE](#)
2.1 3.0 Every reported map names its id, version and status

[MAP_STATUS_INVALID](#)
2.1 3.0 mapStatus uses an allowed enum value

[MAP_ENABLED_DUPLICATE](#)
2.1 3.0 At most one version of a map is enabled

[MAP_VERSION_DUPLICATE](#)
2.1 3.0 A map version is reported once

[MAP_POSITION_NOT_ENABLED](#)
2.1 3.0 The robot stands on a map it has enabled

[MAP_DELETED_UNASKED](#)
2.1 3.0 Maps disappear only on request

[MAP_ENABLE_NOT_REFLECTED](#)
2.1 3.0 A finished enableMap shows in the maps array

[MAP_DELETE_NOT_REFLECTED](#)
2.1 3.0 A finished deleteMap removes the map

[MAP_DOWNLOAD_NOT_REFLECTED](#)
2.1 3.0 A finished downloadMap adds the map

[MAP_DOWNLOAD_DUPLICATE_NOT_REJECTED](#)
2.1 3.0 A download of a map that is already there is refused

Request Checks (14)

[REQUEST_ENTRY_INCOMPLETE](#)
3.0 A request names itself, its kind, its resource and its status

[REQUEST_STATUS_INVALID](#)
3.0 requestStatus uses an allowed enum value

[REQUEST_ID_NOT_UNIQUE](#)
3.0 A requestId names one active request

[REQUEST_NOT_STATED_AS_REQUESTED](#)
3.0 A request begins as REQUESTED

[REQUEST_REPLANNING_WITHOUT_TRAJECTORY](#)
3.0 A replanning request carries the path it asks about

[RESPONSE_LEASE_WITHOUT_GRANT](#)
3.0 A lease is only put on a grant

[REQUEST_LEASE_NOT_EXPIRED](#)
3.0 A lease that ran out is reported as EXPIRED

[REQUEST_SEVERAL_GRANTED_KEPT](#)
3.0 One granted path per zone, the others let go

[MISSION_REQUEST_KEPT_AFTER_CLEARING](#)
3.0 Clearing the order removes the requests with it

[REQUEST_EDGE_NOT_IN_ORDER](#)
3.0 An edge request names an edge of the order

[REQUEST_CORRIDOR_NOT_BASE](#)
3.0 A corridor is released only for the base

[REQUEST_NOT_ANSWERED](#)
3.0 A stated request is answered

[REQUEST_DECISION_NOT_REFLECTED](#)
3.0 A decision reaches the requestStatus

[REQUEST_NOT_STATED](#)
3.0 A response answers a request that was stated

Layout Compatibility (4)

[LAYOUT_MAP_MISMATCH](#)
2.1 3.0 The orders were driven on the layout they are judged against

[LAYOUT_ORDER_NODE_MISSING](#)
own specification Every ordered node exists in the layout

[LAYOUT_ORDER_EDGE_MISSING](#)
own specification Every ordered edge exists in the layout

[LAYOUT_ORDER_EDGE_NOT_FOR_VEHICLE_TYPE](#)
own specification Every ordered edge is one this vehicle's type may drive

Connection, instant actions, factsheet and visualisation

Connection Checks (7)

[CONN_STATE_SEQUENCE](#)
1.1–3.0 connectionState transitions are plausible

[STATE_INTERVAL_EXCEEDED](#)
1.1–3.0 State is published at least every 30 seconds

[CONN_HIBERNATION_STATE_SENT](#)
3.0 No state messages while HIBERNATING

[CONN_REQUIRED_MISSING](#)
1.1–3.0 Required connection fields are present

[CONN_HEADER_ID_TYPE](#)
1.1–3.0 Connection headerId is an integer

[CONN_VERSION](#)
1.1–3.0 Connection version matches the selected target

[CONN_STATE_INVALID](#)
1.1–3.0 connectionState uses an allowed enum value

Instant Action Checks (7)

[IA_REQUIRED_MISSING](#)
1.1–3.0 Required instantActions fields are present

[IA_HEADER_ID_TYPE](#)
1.1–3.0 instantActions headerId is an integer

[IA_VERSION](#)
1.1–3.0 instantActions version matches the selected target

[IA_ACTION_REQUIRED_MISSING](#)
1.1–3.0 Instant actions contain required fields

[IA_BLOCKING_TYPE](#)
1.1–3.0 Instant action blockingType uses allowed values

[IA_ACTION_TYPE_NOT_PREDEFINED](#)
1.1–3.0 Instant actions use a predefined actionType

[IA_DUPLICATE_ACTION_ID](#)
1.1–3.0 Instant action IDs are unique

Factsheet Checks (3)

[FS_REQUIRED_MISSING](#)
2.0 2.1 3.0 Required factsheet objects are present

[FS_HEADER_ID_TYPE](#)
2.0 2.1 3.0 Factsheet headerId is an integer

[FS_VERSION](#)
2.0 2.1 3.0 Factsheet version matches the selected target

Visualization Checks (4)

[VIS_REQUIRED_MISSING](#)
1.1–3.0 Required visualization fields are present

[VIS_HEADER_ID_TYPE](#)
1.1–3.0 Visualization headerId is an integer

[VIS_VERSION](#)
1.1–3.0 Visualization version matches the selected target

[VIS_REFERENCE_STATE_AHEAD](#)
3.0 The referenced state exists

Missions and transport

Mission Lifecycle Checks (51)

[MISSION_EDGE_RELEASED_TO_TWO_VEHICLES](#)

1.1–3.0 One edge carries one vehicle at a time

[MISSION_STATE_UNKNOWN_ORDER](#)

1.1–3.0 State references a known orderId

[MISSION_ORDER_SUPERSEDED_ACTIONS_OPEN](#)

1.1–3.0 A new order follows a finished one

[MISSION_ORDER_NOT_ACKED](#)

1.1–3.0 State acknowledges the orderId after an order

[MISSION_ORDER_ACK_SLOW](#)

1.1–3.0 Order is acknowledged within the timeout

[MISSION_NODE_NOT_IN_ORDER](#)

1.1–3.0 nodeStates contain only nodes from the order

[MISSION_EDGE_NOT_IN_ORDER](#)

1.1–3.0 edgeStates contain only edges from the order

[MISSION_NODE_SEQUENCE_NOT_IN_ORDER](#)

1.1–3.0 nodeStates report the sequenceId the order gave

[MISSION_EDGE_SEQUENCE_NOT_IN_ORDER](#)

1.1–3.0 edgeStates report the sequenceId the order gave

[MISSION_ACTION_NOT_IN_ORDER](#)

1.1–3.0 actionStates contain only actions from the order

[MISSION_ORDER_UPDATE_NOT_REJECTED](#)

1.1–3.0 Stale order updates are rejected

[MISSION_ORDER_ACTION_NOT_REFLECTED](#)

1.1–3.0 Order actions are reported as actionStates

[MISSION_TRAVERSAL_INCONSISTENT](#)

1.1–3.0 Traversed nodes leave the nodeStates

[MISSION_ZONESET_MISMATCH](#)

1.1 2.0 2.1 State reports the zone set of its order

[MISSION_FACTSHEET_REQUEST_UNANSWERED](#)

2.0 2.1 3.0 A factsheetRequest is answered with a factsheet

[MISSION_HORIZON_ACTION_NOT_WAITING](#)

3.0 Horizon actions are reported as WAITING

[MISSION_ACTION_STARTED_EARLY](#)

1.1–3.0 An action starts at its own node, not before

[MISSION_CANCEL_NOT_EXECUTED](#)

1.1–3.0 A finished cancelOrder leaves no order behind

[MISSION_CANCEL_NO_ORDER_NOT_REPORTED](#)

1.1–3.0 A cancelOrder with nothing to cancel says so

[MISSION_CANCEL_NO_ORDER_REFERENCE](#)

1.1–3.0 The noOrderToCancel warning names its action

[MISSION_CANCEL_NO_ORDER](#)

1.1–3.0 cancelOrder without an order fails

[MISSION_ORDER_SENT_IN_A_MODE_THAT_TAKES_NONE](#)

1.1–3.0 No order is sent to a robot in a mode that takes none

[MISSION_INSTANT_ACTION_NOT_CLEARED_BY_ORDER](#)

1.1–3.0 A new order clears the instant actions it is allowed to clear

[MISSION_UNSUPPORTED_PARAMETER_ORDER_ACCEPTED](#)

3.0 An order with a field the robot cannot use is not taken over

[MISSION_INSUFFICIENT_MEMORY_ORDER_ACCEPTED](#)

3.0 An order that does not fit is not taken over

[MISSION_UNKNOWN_MAP_NOT_REPORTED](#)

3.0 Refusing an order over an unknown map is announced

[MISSION_START_OUT_OF_RANGE_NOT_REPORTED](#)

3.0 Refusing an order that starts elsewhere is announced

[MISSION_MODE_REFUSAL_NOT_REPORTED](#)

3.0 An order arriving in the wrong mode is refused out loud

[MISSION_NO_ROUTE_ORDER_ACCEPTED](#)

3.0 An order the robot cannot route is not taken over

[MISSION_INSTANT_ACTION_SENT_IN_A_MODE_THAT_TAKES_NONE](#)

1.1–3.0 No instant action is sent to a robot in a mode that takes none

[MISSION_INTERVENED_ORDER_NOT_RESUMED](#)

3.0 Coming back from INTERVENED continues the order

[MISSION_HIBERNATION_ORDER_NOT_CLEARED](#)

3.0 startHibernation clears the order like a mode change does

[MISSION_LAST_NODE_KEPT_IN_A_MODE_THAT_ENDS_IT](#)

3.0 A mode that ends the route forgets the node it stood on

[MISSION_MANUAL_ORDER_NOT_CLEARED](#)

1.1–3.0 A mode that clears the order leaves no route behind

[MISSION_CLEARED_ORDER_CAME_BACK](#)

1.1–3.0 A cleared order does not come back on its own

[MISSION_MODE_ACTIONS_NOT_SETTLED](#)

3.0 The mode is reported after its actions have settled

[MISSION_ORDER_ID_DROPPED](#)

1.1–3.0 Clearing an order keeps its identity

[MISSION_BASE_ACTION_DROPPED](#)

1.1–3.0 Base actionStates are never dropped

[MISSION_HORIZON_TRAVERSED](#)

1.1–3.0 The robot stays within the released base

[MISSION_EDGE_ACTION_OFF_EDGE](#)

1.1–3.0 An edge action runs only on its edge

[MISSION_ORDER_MAP_UNKNOWN](#)

2.1 3.0 An order names maps the robot holds

[MISSION_ORDER_START_OUT_OF_RANGE](#)

1.1–3.0 A new order starts where the robot is

[MISSION_UNDOABLE_ORDER_ACCEPTED](#)

1.1–3.0 An order with actions the robot cannot do is not taken over

[MISSION_ORDER_ERROR_NOT_REPORTED](#)

1.1–3.0 Refusing an order with undoable actions is announced

[MISSION_MALFORMED_ORDER_ACCEPTED](#)

1.1–3.0 A malformed order is not taken over

[MISSION_MALFORMED_ORDER_NOT_REPORTED](#)

1.1–3.0 Refusing a malformed order is announced

[MISSION_ORDER_UPDATE_ERROR_NOT_REPORTED](#)

1.1–3.0 Refusing a stale order update is announced

[MISSION_CANCEL_ACTIONS_PENDING](#)

1.1–3.0 A finished cancelOrder leaves no action running

[MISSION_ORDER_ACCEPTED_WHILE_BUSY](#)

3.0 A new order is accepted only when the robot is idle

[MISSION_OTHER_ORDER_ACTIVE_NOT_REPORTED](#)

3.0 Refusing a second order is announced

[MISSION_INSTANT_ACTION_NOT_REFLECTED](#)

1.1–3.0 instantActions are reflected in state

Transport Checks (8)

[TRANSPORT_QOS_UNEXPECTED](#)

1.1–3.0 Each topic is published at the QoS the standard gives it

[TRANSPORT_NOT_RETAINED](#)

1.1–3.0 The connection and factsheet messages are retained

[TOPIC_STRUCTURE](#)

1.1–3.0 Topic follows the suggested level structure

[TOPIC_IDENTITY_MISMATCH](#)

1.1–3.0 Topic names the same vehicle as the payload

[TOPIC_VERSION_MISMATCH](#)

1.1–3.0 Topic major version matches the message version

[TOPIC_HEADER_SEQUENCE](#)

1.1–3.0 headerId counts up per topic

[TOPIC_CHARSET](#)

1.1–3.0 Topic levels use permitted characters

[IDENTITY_FIELD_CHARSET](#)

1.1–3.0 manufacturer and serialNumber can carry a topic

Capabilities declared in the factsheet

Capability Checks (24)

[CAP_ACTION_NOT_DECLARED](#)

2.0 2.1 3.0 Action type is declared in the factsheet

[CAP_ACTION_SCOPE_INVALID](#)

2.0 2.1 3.0 Action is used within its declared scope

[CAP_ACTION_BLOCKING_TYPE_INVALID](#)

2.0 2.1 3.0 blockingType is one the vehicle declares

[CAP_ACTION_PARAM_UNDECLARED](#)

2.0 2.1 3.0 Action parameters are declared

[CAP_ACTION_PARAM_REQUIRED_MISSING](#)

2.0 2.1 3.0 Required action parameters are sent

[CAP_ACTION_PARAM_TYPE_MISMATCH](#)

2.0 2.1 3.0 Action parameter values match the declared type

[CAP_ACTION_CANCEL_NOT_ALLOWED](#)

3.0 cancelOrder respects cancelAllowed

[CAP_ACTION_PAUSE_NOT_ALLOWED](#)

3.0 startPause respects pauseAllowed

[CAP_ZONE_TYPE_NOT_SUPPORTED](#)

3.0 A zone set only uses zone types the robot understands

[CAP_ZONE_ACTION_STATE_MISSING](#)

3.0 A zone action is reported in the array made for it

[CAP_OPTIONAL_PARAM_UNSUPPORTED](#)

2.0 2.1 3.0 Only declared optional parameters are sent

[CAP_OPTIONAL_PARAM_REQUIRED_MISSING](#)

2.0 2.1 3.0 Parameters declared REQUIRED are sent

[CAP_OPTIONAL_PARAM_UNKNOWN_PATH](#)

2.0 2.1 3.0 Declared optional parameters exist in the schema

[CAP_LIMIT_MSG_LEN](#)

2.0 2.1 3.0 Messages stay within the declared length

[CAP_LIMIT_ID_LEN](#)

2.0 2.1 3.0 Identifiers stay within the declared length

[CAP_LIMIT_ID_NUMERICAL](#)

2.0 2.1 3.0 Identifiers are numeric when declared so

[CAP_LIMIT_LOAD_ID_LEN](#)

2.0 2.1 3.0 loadId stays within the declared length

[CAP_LIMIT_TOPIC_LEN](#)

2.0 2.1 3.0 Topic elements stay within the declared length

[CAP_LIMIT_ARRAY_LEN_ORDER](#)

2.0 2.1 3.0 Orders respect the declared array limits

[CAP_LIMIT_ARRAY_LEN_STATE](#)

2.0 2.1 3.0 States respect the declared array limits

[CAP_TIMING_ORDER_INTERVAL](#)

2.0 2.1 3.0 Orders keep the declared minimum interval

[CAP_TIMING_STATE_INTERVAL_MIN](#)

2.0 2.1 3.0 States keep the declared minimum interval

[CAP_TIMING_STATE_INTERVAL_DEFAULT](#)

2.0 2.1 3.0 States arrive within the declared default interval

[CAP_TIMING_VISUALIZATION_INTERVAL](#)

2.0 2.1 3.0 Visualization keeps the declared interval

Further checks

M2X Error Checks (4)

[M2X_FAILED_REQUEST_WITHOUT_ERROR](#)

own specification A failed request says why

[M2X_ERROR_TYPE_NOT_UPPER_SNAKE](#)

own specification Error types are written in UPPER_SNAKE_CASE

[M2X_ERROR_LEVEL_NOT_PREDEFINED](#)

own specification A predefined error carries the level the table gives it

[M2X_ERROR_REFERENCE_MISSING](#)

own specification A predefined error carries the reference its row names

VDMA LIF 1.0.0 and 2.0.0 (preview)

Schema Conformance (1)

[SCHEMA_LIF_VIOLATION](#)

own specification LIF file conforms to the LIF JSON schema

LIF Structural Checks (5)

[LIF_DUPLICATE_LAYOUT_ID](#)

own specification layoutId is unique

[LIF_DUPLICATE_NODE_ID](#)

own specification nodeId is unique across the file

[LIF_DUPLICATE_EDGE_ID](#)

own specification edgeId is unique across the file

[LIF_DUPLICATE_STATION_ID](#)

own specification stationId is unique across the file

[LIF_DEVIATION_RANGE_INVERTED](#)

own specification allowedDeviationXY minima do not exceed their maxima

LIF Graph Checks (5)

[LIF_EDGE_START_NODE_UNKNOWN](#)

own specification Edge startNodeId is part of the same layout

[LIF_EDGE_END_NODE_UNKNOWN](#)

own specification Edge endNodeId references an existing node

[LIF_ISOLATED_NODE](#)

own specification Every node is connected by at least one edge

[LIF_STATION_NODE_UNKNOWN](#)

own specification Station interactionNodeIds reference existing nodes

[LIF_ROTATION_CONFLICT_AT_NODE](#)

own specification Rotation allowances agree where edges meet

LIF Vehicle Type Checks (2)

[LIF_DUPLICATE_VEHICLE_TYPE_NODE_PROPERTY](#)

own specification Only one vehicleTypeNodeProperty per vehicle type per node

[LIF_DUPLICATE_VEHICLE_TYPE_EDGE_PROPERTY](#)

own specification Only one vehicleTypeEdgeProperty per vehicle type per edge

LIF Trajectory Checks (2)

[LIF_TRAJECTORY_KNOT_VECTOR_SIZE](#)

own specification Trajectory knotVector has the expected size

[LIF_CORRIDOR_ZERO_WIDTH](#)

2.1 A corridor of zero width is left out instead

M2X 0.9.0 (preview)

M2X Schema Conformance (6)

- [M2X_SCHEMA_LHS_VIOLATION](#)
own specification Load handling station messages conform to the M2X JSON schema
- [M2X_SCHEMA_DOOR_VIOLATION](#)
own specification Door messages conform to the M2X JSON schema
- [M2X_SCHEMA_ELEVATOR_VIOLATION](#)
own specification Elevator messages conform to the M2X JSON schema
- [M2X_SCHEMA_SIGNAL_VIOLATION](#)
own specification Signaling device messages conform to the M2X JSON schema

- [M2X_SCHEMA_CHARGING_VIOLATION](#)
own specification Charging device messages conform to the M2X JSON schema
- [M2X_SCHEMA_ORDER_VIOLATION](#)
own specification Transport order messages conform to the M2X JSON schema

M2X Lifecycle Checks (6)

- [M2X_REQUEST_NOT_REFLECTED](#)
own specification Requests appear in the entity state
- [M2X_MULTIPLE_ACTIVE_REQUESTS](#)
own specification Only one request is active per entity

- [M2X_REQUEST_INVALID_TRANSITION](#)
own specification Request status transitions are allowed
- [M2X_INTERACTION_SEQUENCE](#)
own specification Interaction steps follow the specified order
- [M2X_STATE_HEARTBEAT_MISSED](#)
own specification Load handling stations publish state at least every 30 seconds
- [M2X_REQUEST_TYPE_NOT_SUPPORTED](#)
own specification Requests match the addressed interface



The next step

1

Demonstration

A walkthrough on a bundled demo package: one clean and one faulty data set with a matching layout. Ready to show without access to your site.

2

Pilot on your data

One recording from your installation, analysed and discussed together. The result is a report you keep.

3

Operation

Hosted at logiq-vda5050.com or in your own network with Docker.



Matthias Merz

LogistiXpert

m.merz@logistixpert.com

logistixpert.com