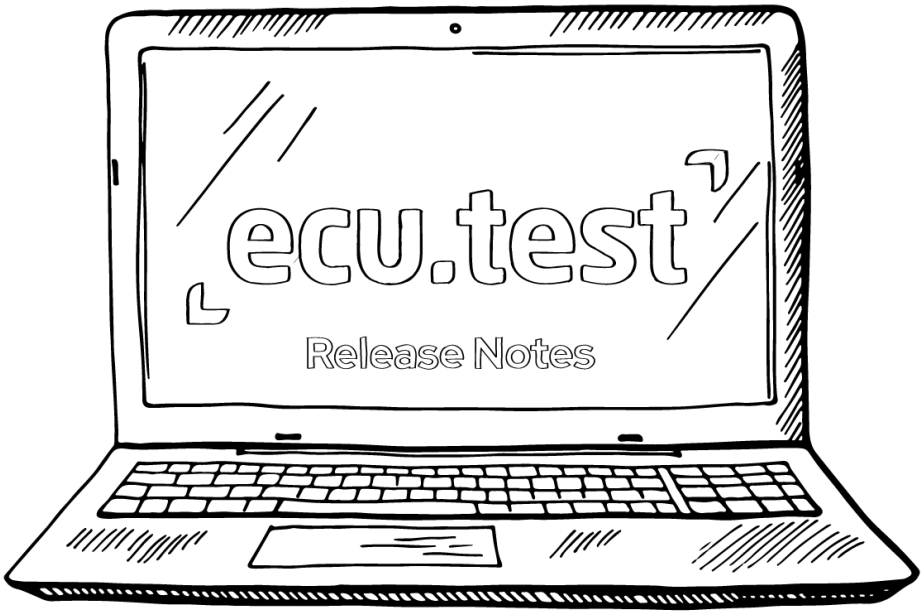
tracetronic



Release Notes

ecu.test 2025.4

trace.check 2025.4

Date: 12/09/2025

tracetronic GmbH
Stuttgarter Str. 3
01189 Dresden
www.tracetronic.com

tracetronic

Content

Overview

The new **2025.4** release provides smarter automation, more flexible workflows and powerful new features that make working with **ecu.test** and **ecu.test** extras even more efficient.

The **ecu.test** *agent* now generates test cases with new, more robust, realistic, and precise functions. For example:

- loops in specifications are automatically detected and implemented,
- signals are identified faster and more reliably,
- generated code blocks are visually marked, and
- a wider range of LLMs is also supported.

Early Access: AI-supported trace analysis – describe what you want to analyze and the agent generates the analysis automatically. More information about the **ecu.test** *agent* is available here.

The **ecu.test** *lab* offers greater performance and flexibility for individual use.

- Package execution is significantly faster thanks to the new logic
- Reuse of created views in other views
- Integration of your own widgets via API

Trace analysis has become noticeably more powerful when handling large amounts of signals. For example, **measurement lists can now be defined as XAM files** and added directly to the signal recording. This is significantly more efficient than dragging and dropping all signals into the **Signal recording** tab.

This quarter, we are celebrating the **launch of one:cx** – our modular platform for test and integration automation. It consolidates fragmented tools, data, and processes within a centralized testing ecosystem. AI-supported, cloud-based, and infinitely scalable. The result is continuous innovation, fewer errors, and significantly shorter development cycles. Visit our website for more information.

Sneak preview: We are introducing the **test.spec** *agent*, another AI-powered assistant that automatically generates test specifications from requirements. Additionally, we offer multi-FMU simulations for complex integration and system tests are. Both are available upon request!

Note: Icons are used to indicate for which product a topic is relevant:
ecu.test **trace.check.**

1 Highlights in ecu.test 2025.4

ecu.test *agent*

The current release focuses on significantly improving the quality of automatic test case generation. Below is an overview of the most important new features:

- **Support for loops**
  The agent can now recognize repetitive sequences (loops) in the specifications and generate the corresponding test steps. This increases the completeness and realism of the generated test cases.
- **Visual marking of generated blocks**
  Each code block generated by the **ecu.test** *agent* is now marked with a special icon. This makes automatically generated sections immediately recognizable and easier to check or adjust.
- **Optimized signal detection**
  The detection logic for signals (e.g., event triggers, state changes) has been revised. Signals are identified faster and more reliably, resulting in more precise test sequences.
- **Enhanced ecu.test agent connector**
  The connector now supports a wider range of large language models (LLMs). This allows different LLM backends to be seamlessly integrated, increasing flexibility and scalability.
- **Use of tool and port jobs without global mapping**
  The agent can now use tool and port jobs that were not previously defined in the global mapping. This enables a leaner configuration and reduces the effort required to create mapping entries.

These improvements make the **ecu.test** *agent* deliver more robust, easier-to-understand test cases that are better adapted to the respective development environment.

AI-supported trace analysis with ecu.test *agent*

We've made getting started with trace analysis even easier! Now, you can describe the desired analysis and have it generated with AI support.

Be among the first to test this feature! Send a short email to our support team, and we will activate your early access. We look forward to your feedback!

ecu.test *lab*

et

Improved package execution

Previously, packages could already be executed from **ecu.test** *lab*. However, execution was limited to a few widget types and was accompanied by significant latency. Thanks to new execution logic in **ecu.test**, it is now possible to execute packages from **ecu.test** *lab* much faster.

This opens up the possibility of using packages for more use cases in **ecu.test** *lab*. As a result, packages are now also available for reading values in bar, slider, gauge, and the new custom widgets.

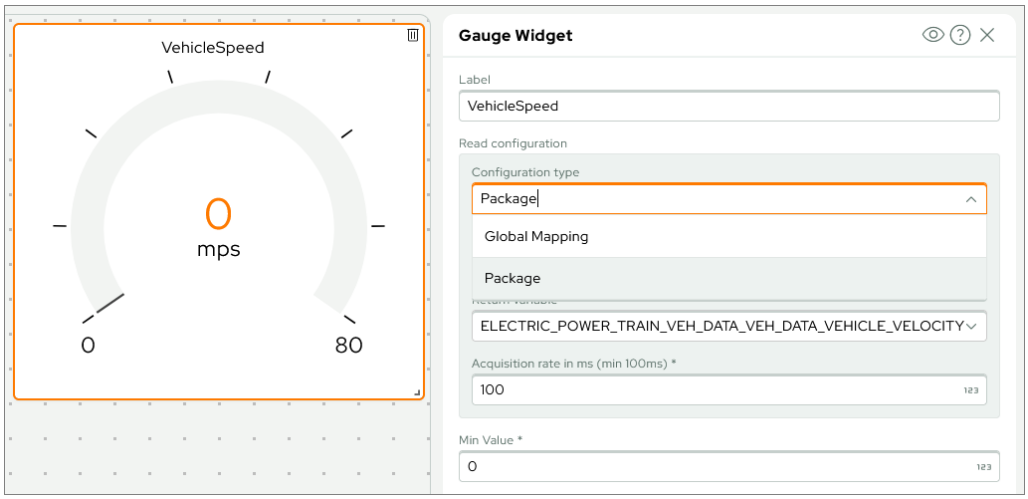


Figure 1: Improved package execution

Custom Widgets

ecu.test *lab* (short: *lab*) already offers a selection of widgets that will continue to expand in the future. Due to the immediate need for more design options and demand for individual widgets, we are now opening *lab* for your own **custom widgets**.

How it works:

- Widgets are created in HTML/JavaScript and stored in the **ecu.test** workspace.
- An API provided by us is used to connect to *lab* in order to supply the widget with measured values and to transfer actions in the widget directly to the test system.
- **Custom widgets** are automatically displayed in the *lab* sidebar, just like the standard widgets, and can be configured there.

Available interfaces:

- Globale Mappings
- Packages

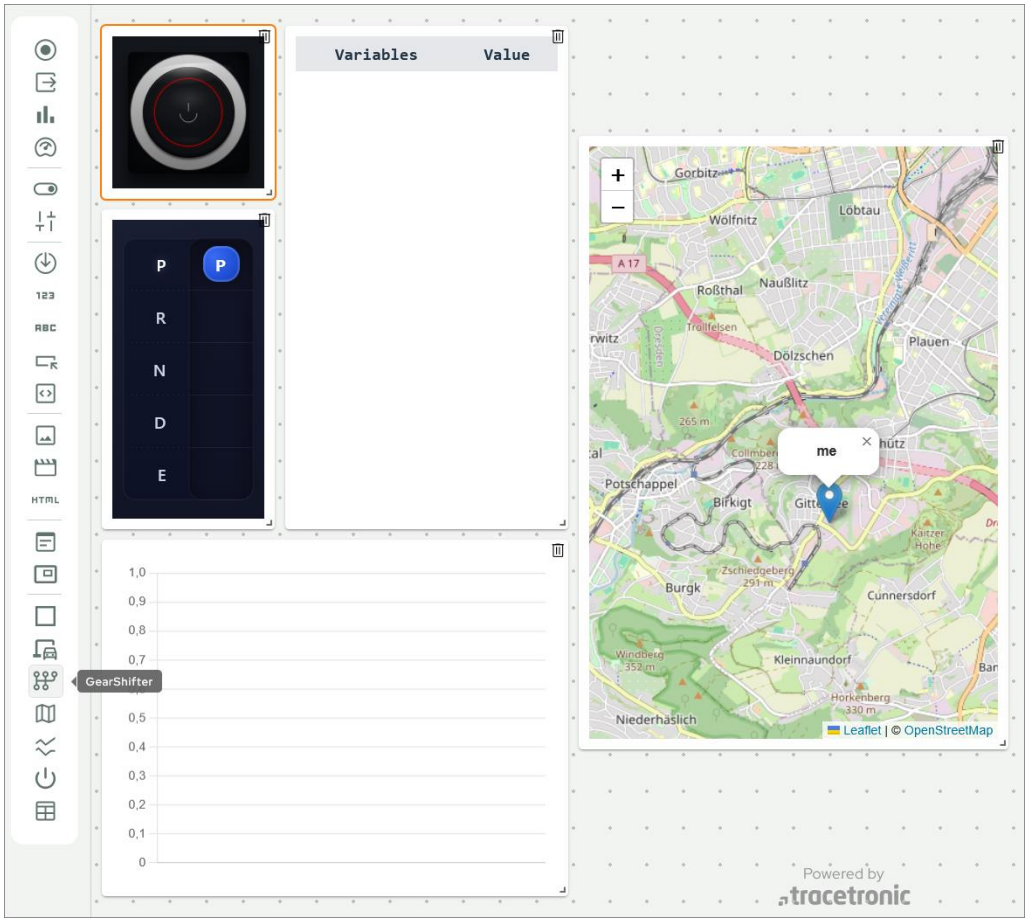


Figure 2: Integrating of custom widgets

View Reference Widget

When creating the second view, many users realize that, despite the different use case, many of the same basic controls are needed in every view.

In response to this request, we are enabling the referencing of a (library) view in another view.

How it works:

- The (library) view is created, configured, and maintained like any other view.
- This view can be integrated into any other view using the **View Reference Widget**, even if it is stored in a library workspace.
- The content of the view cannot be changed as a reference, but it can be used without restriction outside of edit mode.

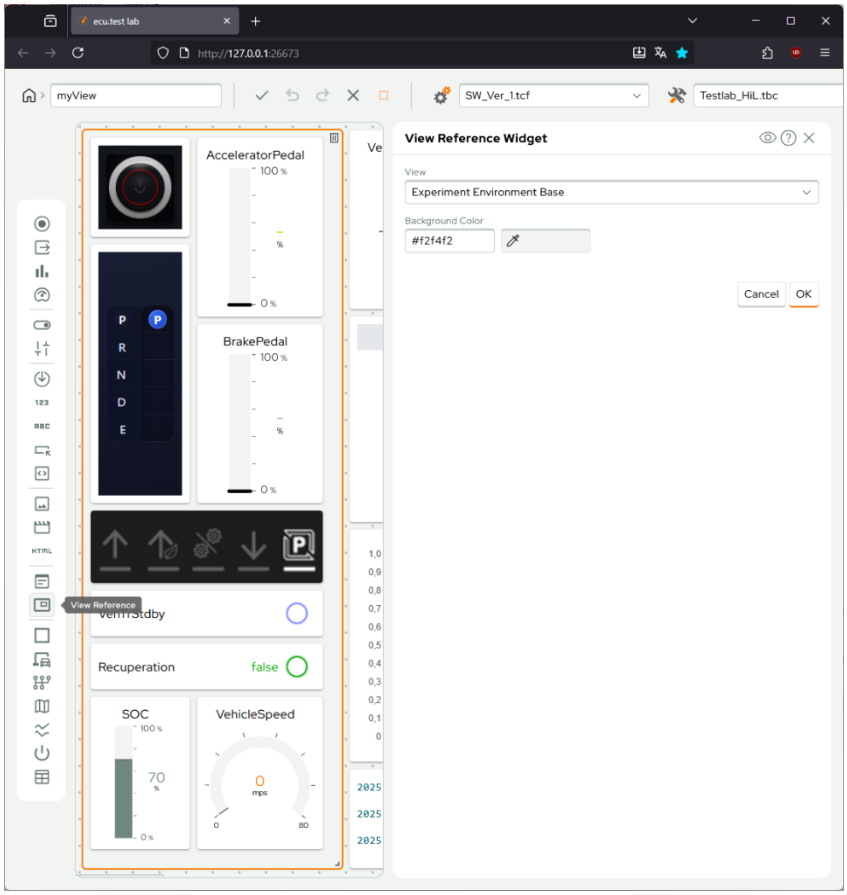


Figure 3: Referencing a (library) view in another view

Specification of measurement lists for signal groups

et

In **ecu.test**, signals to be recorded can be added in the **signal recording** area. This can be done either via drag & drop or via Object API.

With a large number of measured variables, this procedure can be cumbersome and inefficient. For this reason, it is now possible to write the signals as mapping variables to an XAM file (already known as the data format for global mappings) and add this file to the signal recording. This procedure causes all signals in this file to be registered for the respective recording group.

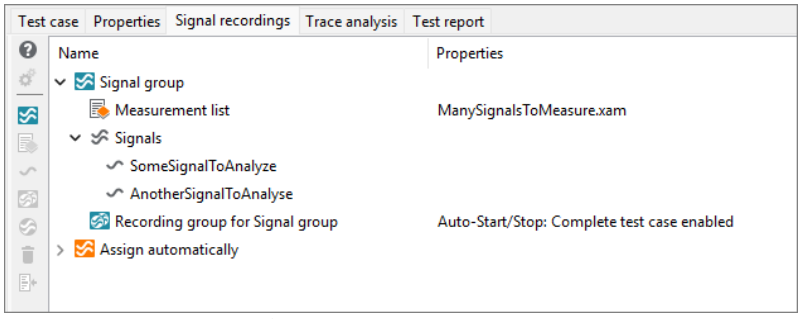


Figure 4: Measurement lists for signal groups

ecu.test in Japanese

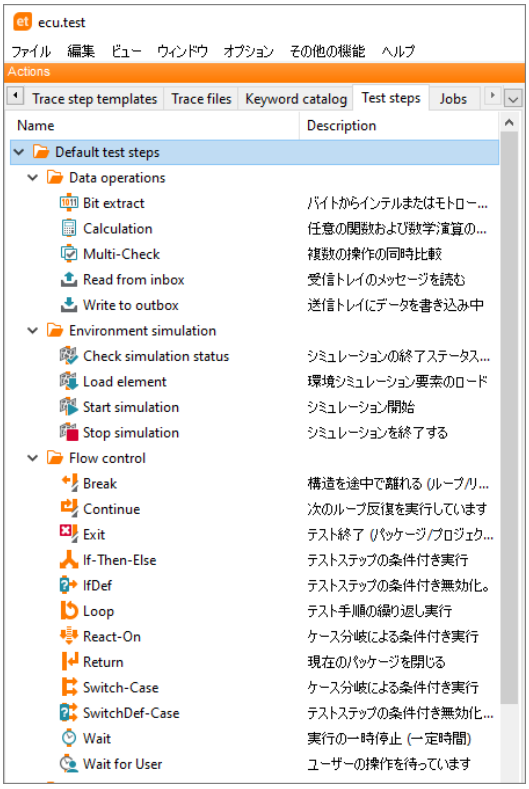


Figure 5: ecu.test with Japanese GUI language

Since June 2025, tracetronic has also been represented in Japan. In line with this, our software now supports Japanese as a fourth language in addition to English, German, and Chinese. Both the user interface and the test case language can now be used entirely in Japanese.

So, Japanese users can use our software in their native language right from the start, which makes it easy and intuitive to use.
At the same time, we are strengthening our international orientation and laying the foundation for further local adaptations.

Figure 6: Japanese informational texts in the GUI

2 Sneak Preview

Our latest AI service: test.spec *agent*

Are you familiar with this scenario? Working through requirements, writing specifications, and manually deriving test cases? What used to take hours of routine work can now be significantly simplified and accelerated.

The test.spec agent starts at the beginning of the testing pipeline by using an AI assistant to generate test specifications from requirements. Then, the **ecu.test** *agent* generates test cases from these specifications in a matter of seconds, saving even more time. It's a novelty!

The MVP of the test.spec agent is a powerful, AI-based tool that automatically delivers specifications and can be actively integrated into development processes.

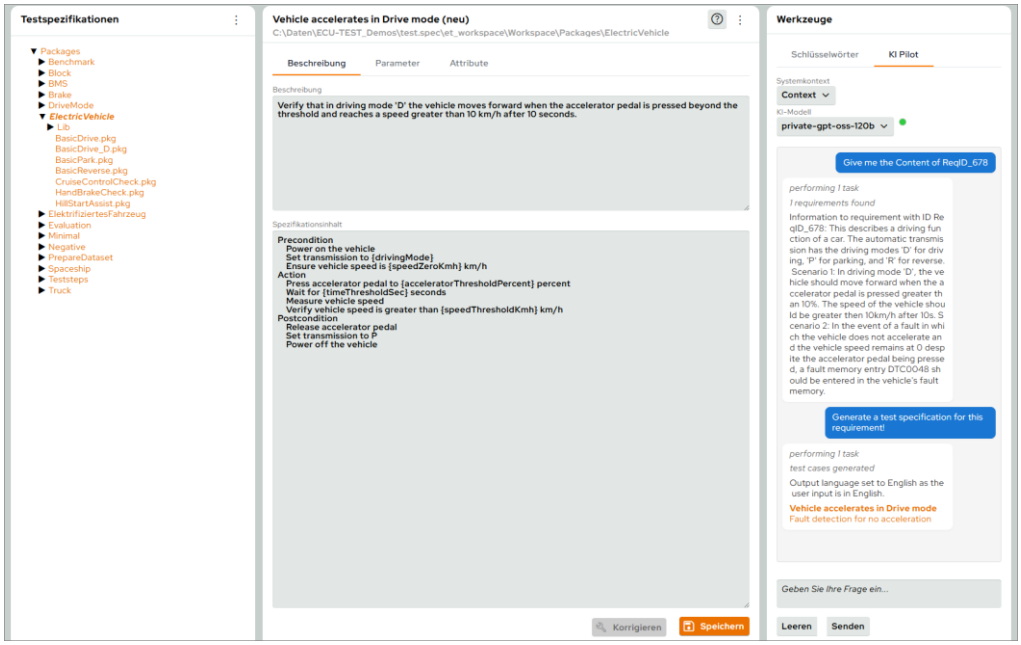


Figure 7: test.spec *agent*

Before:

1. Read requirements
2. Search library
3. Research references
4. Write specification
5. Review
6. Make adjustments
7. Finished

Duration: approx. 2.5 hours

Now:

1. Upload requirement
2. Generate specification
3. Check, ask questions
4. Export to **ecu.test**
5. Finished

Duration: approx. 15 minutes

What are the advantages of the test.spec *agent*?

From hours to minutes	Complete test specifications are automatically generated from requirements, including consideration of suitable library packages and references from ALM systems.
Interactive communication	The agent works like an experienced team member: it generates specifications, displays relevant requirement content, considers library packages, and answers queries.
Fewer errors and less rework	The agent uses existing libraries and company-specific knowledge bases to create comprehensive, reusable specifications. This enables faster validation, higher test coverage, and lower error rates.
Consistent, reliable data	Access to ALM systems, existing test case references, and other sources is defined by the Model Context Protocol (MCP) standard. This guarantees the quality and consistency of test specifications by using proven data sources.
Well-founded quality	Improve specification quality through data-driven deep research instead of considering requirements in isolation.
Seamless integration	Combined with the ecu.test agent, the tooling can be easily integrated into existing test and development environments without additional tools and complicated interfaces.

Take part in shaping test.spec *agent*!
The current MVP is functional, but we're continuing to develop it with you to ensure it offers the best possible benefits. How? With your feedback, use cases, and wishes. Every idea will be incorporated directly into the next version.

Want to see how it works with your requirements? Let us show you a demo!

- We'll give you a live demo with your requirements and your test environment.
- You will have access to the current MVP, including an installation guide and a quick start tutorial.

Let's accelerate your testing processes together.
We look forward to your request!

Simultaneous simulation of multiple FMUs in ecu.test

With support for the simultaneous simulation of multiple FMUs, known as co-simulation, complex model systems can now also be mapped directly in **ecu.test**. This enables, for example, the transfer of outputs from one FMU as inputs to another or the coupling of multiple models via the FMI Layered Standard Bus.

This is a fundamental building block for coupling multiple vECUs for validation in integration and system tests.

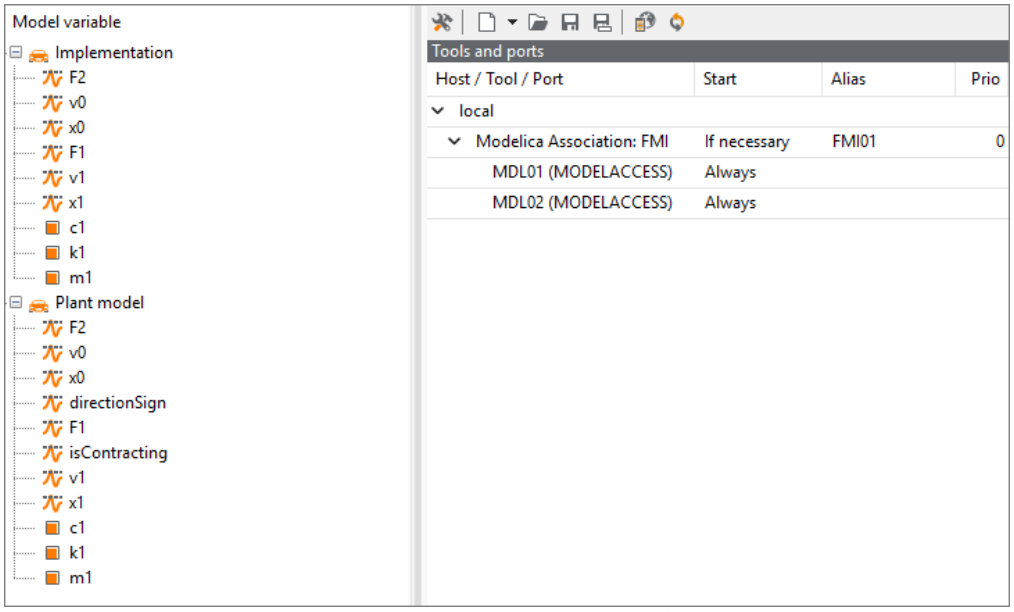


Figure 8: In TBC and TCF, multiple FMUs can be configured, the sizes of which can be accessed in Model Access.

The implementation is based on the SSP standard (System Structure and Parameterization). Signal flows and connections are defined centrally via an SSD file and taken into account in the simulation.

The new function can be used independently of the LS bus and supports scenarios with and without bus communication between the FMUs.

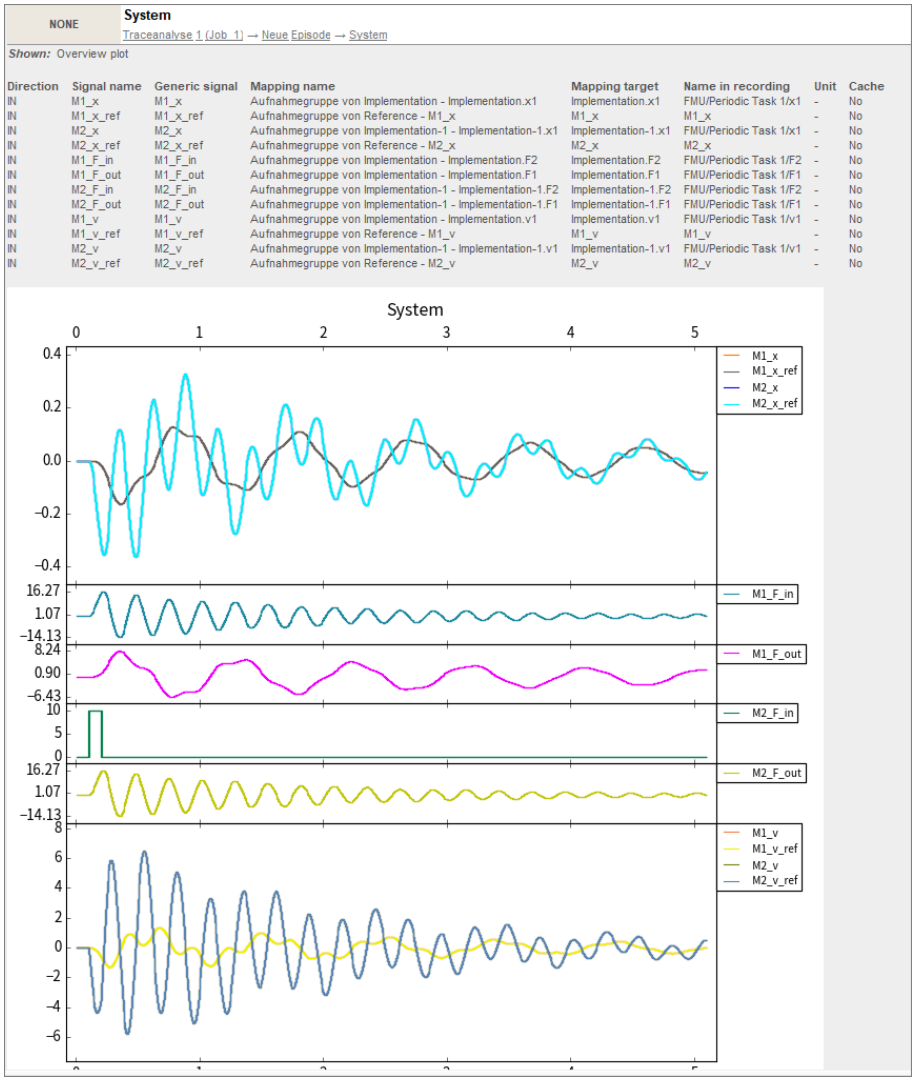


Figure 9:The coupling of two spring-mass dampers

Further information on use cases, implementation, and current limitations is available from us on request.

Additional help formats for AI chatbots

The user documentation is now available in several AI-compatible formats. Upon request, different versions can be provided, for example as PDF files with or without images.

This allows us to support modern usage scenarios such as chatbots and facilitates further processing of the content. Individual in-house solutions are therefore no longer necessary.

3 Usability

Improvements around test case coverage

Test case coverage has been enhanced once again with new features that significantly improve usability. These features allow library packages to be validated more effectively:

- **REST API for parameterizing and generating reports:** New REST endpoints have been introduced to enable coverage reports to be loaded, deleted, and generated via API. The API is based on the previous GUI concept. This simplifies automation and integration into existing workflows and enables integration into the workflow automation from **test.guide**.

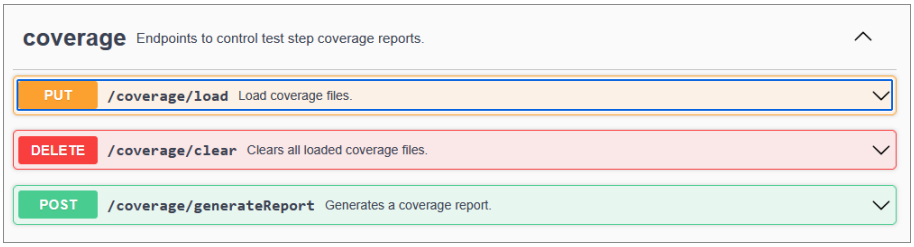


Figure 10: New REST API endpoints for coverage generation

- **Improved pagination:** In the pagination in the coverage report, it is now possible to display “all” entries at once. The term **items per page** in the report has been rephrased to make it easier to understand and use.

- **Filtering in the coverage report:** The coverage report now supports flexible filtering of packages and folders. The filtered elements are displayed in their original structure, while irrelevant elements can be hidden.

- **Checking the coverage report for package revision:** Coverage report generation now checks whether the revision of the analyzed package matches the loaded coverage files (via Git revision, modification date, or hash). If there are any discrepancies, a warning is issued, and coverage is no longer generated for unsaved or modified packages.

- **Display in the report which coverages are loaded:** All loaded coverage files are now clearly displayed in the coverage report. This makes it immediately apparent which files have been included – including the absolute paths. This is especially helpful when loading coverage reports from **test.guide**!

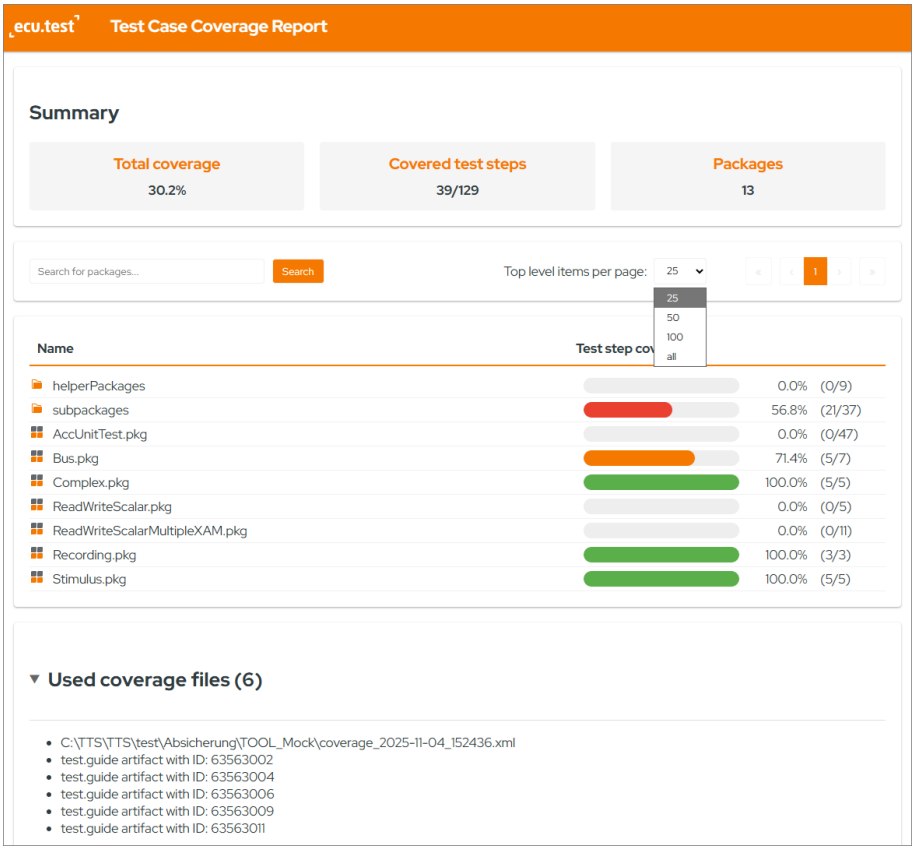


Figure 11: Revised coverage report

Manual termination of recovery packages

et

Recovery packages allow you to reset the test bench if tests fail with an ERROR. Until now, it has not been possible to terminate the execution of these recovery packages. In the current release, this termination is now possible.

Configurable priority of REST API constants

et tc

The priority of global constants set via the REST API can now be configured as desired in the test configuration.

This allows these constants to be specifically assigned to the priority list and makes them available when loading additional constants, enabling them to be overloaded or evaluated in a script (ConstantProvider.py), for example.
This allows dependencies between constants to be resolved more flexibly.

This ensures more transparency and control when working with global constants in complex test sequences, e.g. with **test.guide**. The configuration is done in the test configuration editor (TCF) and can also be done via Object API.

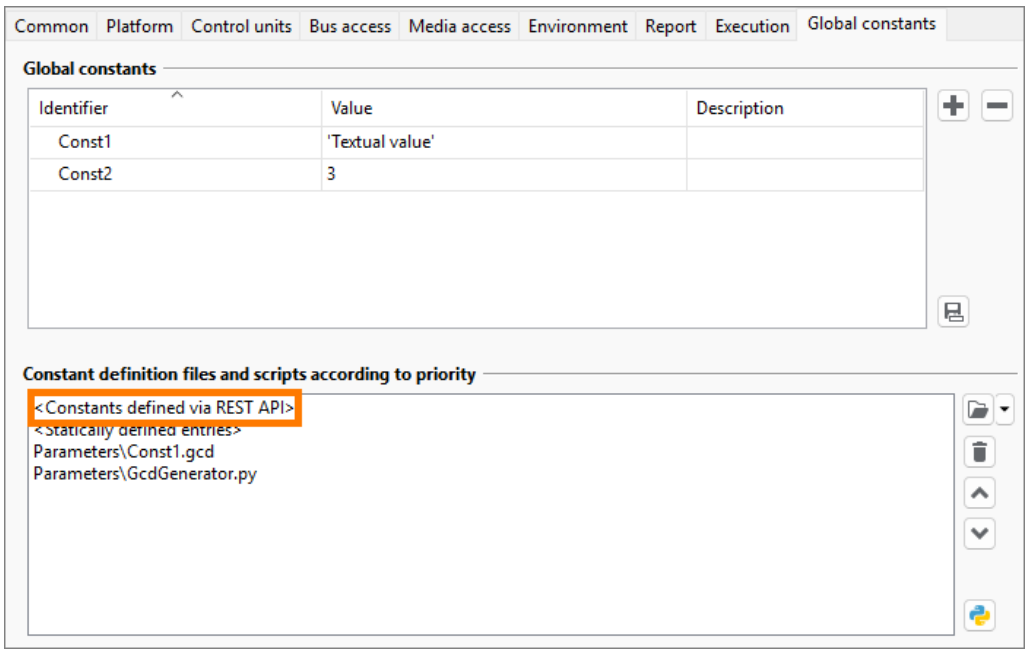


Figure 12: Priority of global constants transmitted via REST API in the TCF

Library workspaces: Support for selective project execution

For several releases now, **ecu.test** projects can also be managed in library workspaces, which increases collaboration and reusability. These projects can now also be used with the selective project execution feature of **ecu.test**.

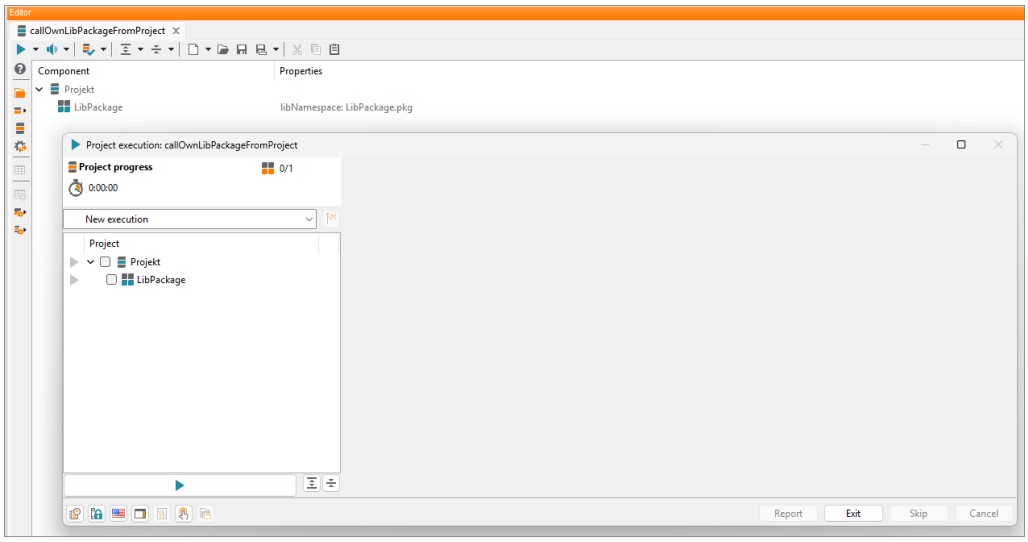


Figure 13: Selective project execution with projects from libraries

4 Test aspects

4.1 Multimedia

Extensive improvements to Tool Gen<I>Cam

et

Extensive improvements have been made to the standard GenCam interface for camera control.

First, performance has been enhanced when using multiple cameras simultaneously. Additionally, users can select different pixel formats for the camera. RAW (Bayer pixel format), RGB, and Mono are a few of the available options. RAW offers higher quality, while Mono saves bandwidth.

To facilitate camera setup, UserSets can now be accessed in the configuration. These are settings profiles stored in the camera that describe settings such as exposure time and zoom. They are created with the associated camera software.

Additionally, it is now possible to execute properties, such as triggers, directly via the job in the test case. This makes the camera even more flexible in the test case.

Integrating user-defined models for object recognition

et tc

Object recognition is a classic use case for AI models. With user-defined object recognition, you can apply your own models to images in the test case via a new interface.

You can create your own object detectors for this purpose. Image objects can be passed to these detectors via the internal API. The detectors then provide a list of recognized objects.

For more information, see the user documentation under user-defined object recognition.

4.2 HiL

Playback of ASC recordings on CAN(-FD) bus hardware

et

There can be various reasons for wanting to play back existing recordings: perhaps the device under test needs to be stimulated again with previously recorded communication, or a test case needs to be developed and tested offline without a real remote device being available.

The following new jobs have been added to enable playback of existing recordings:

- **StartPlayback**
- **StopPlayback**

When playing back ASC frames, the transmission timestamps are adhered to as accurately as possible.

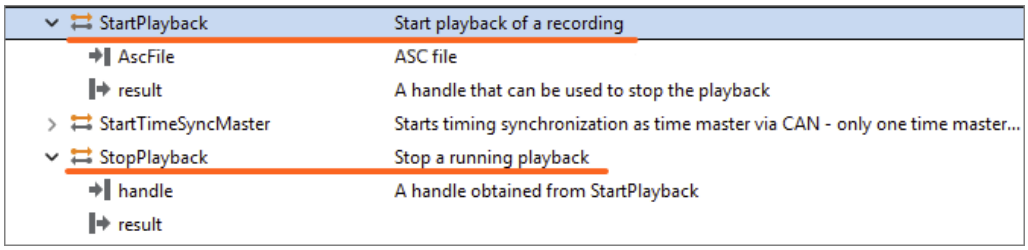


Figure 14: New jobs for playing back existing recordings

Support for ZLG CAN-FD bus hardware connection

et

All features of hardware-related bus connections, including **ecu.test** *diagnostics* and **ecu.test** *calibration*, can now be used with CAN-FD-capable measurement hardware for CAN and CAN-FD from ZLG.

Note: Measurement hardware that only supports CAN is not supported at the moment.

Support for PEAK CAN-FD measurement hardware

et

In addition to CAN, we now also support CAN-FD-capable measurement hardware for PEAK. This means that all features of the hardware-related bus connections, including **ecu.test** *diagnostics* and **ecu.test** *calibration*, can also be used.

4.3 Test management

Jama Connect example workflow: Hierarchical display in the import GUI

et

In Jama Connect, test cases are displayed hierarchically, but the import dialog of the **ecu.test** sample workflow previously only showed a flat list. This made navigation cumbersome and could lead to long loading times.

The sample workflow has been expanded so that the import dialog now supports the same hierarchical structure as in Jama Connect. With lazy loading, subcategories are only loaded when you open the corresponding parent entry. This reduces the initial loading time and makes navigation much clearer and faster.

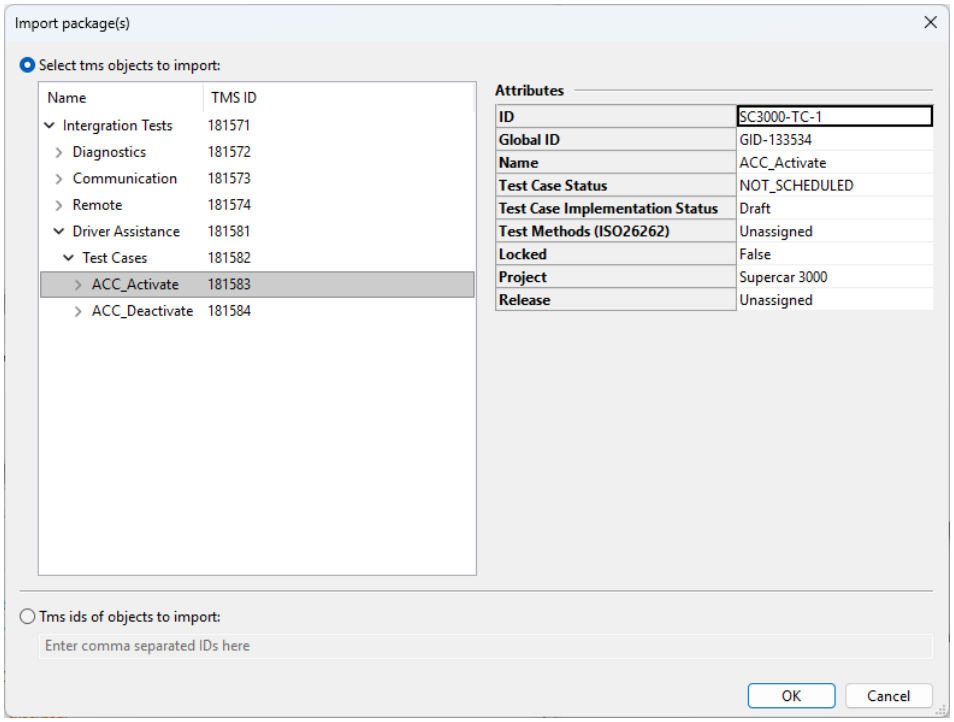


Figure 15: Import dialog supports the same hierarchical structure as in Jama Connect

4.4 ecu.test *calibration*

Specifying the source port for XCP over Ethernet

et

If the control unit only accepts XCP connections from selected source ports, these port ranges can now be specified in the test bench configuration.

4.5 ecu.test *code*

Support of recordings

et

Signal recordings are often necessary to understand and analyze the test sequence. Therefore, **ecu.test** *code* now allows you to create, start, and stop these recordings.

```
# define a recording
rec = ta.recording("C:\\temp\\recording", [terminal, model_driving_mode])

# start execution
with ta.run():
    # start recording
    rec.start()

    # action
    model_driving_mode.write(1)

    # stop recording
    rec.stop()

# finalize recording
recordingInfos = recording.finish()
```

Figure 16: Erstellen von Signalaufnahmen aus ecu.test *code*

4.6 ecu.test *diagnostics*

Separation of the session layer from the application layer

et

To maintain a non-default session, the **TesterPresent** service must be sent periodically. This service plays a special role in the standard, as unlike the other services, it is not processed sequentially but must be sent at any time and without delay.

Technically speaking, the **TesterPresent** service is part of the session layer. This special feature was implemented with 2025.4. This makes it possible to send **TesterPresent** requests independently of other pending requests. This addresses edge cases in particular.

4.7 ecu.test *drive*

Open ecu.test *drive* via the Extras menu

et

ecu.test *drive* can now be opened directly in the browser via the **Extras** menu.

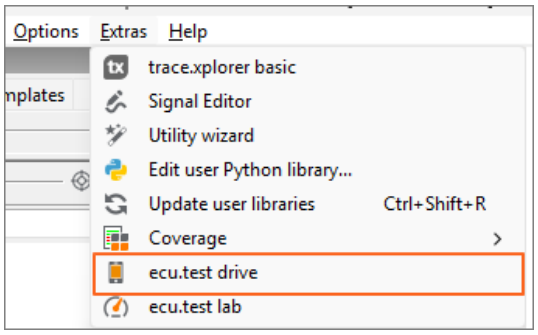


Figure 17: Open ecu.test *drive* via the Extras menu

Improved display of actual and target values

et

The actual value is now rounded to three decimal places when used in block texts and time options. This prevents uneven display due to rounding errors and improves readability.

Manual expectations, such as "min ≤ value ≤ max," are evaluated as much as possible to increase traceability for the test end.

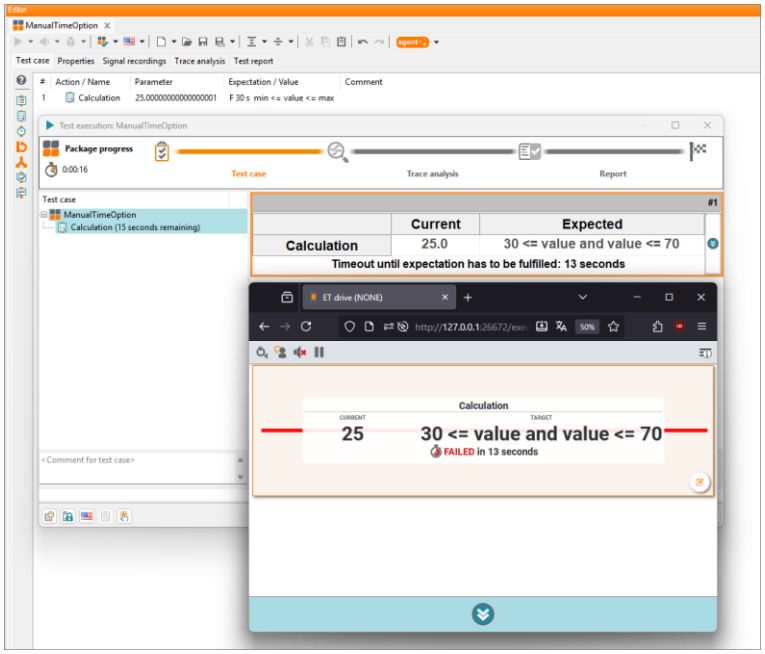


Figure 18: Improved display of actual and target values

Pausing of test cases

et

Test cases can now be paused in **ecu.test** *drive* using a button in the title bar. The pause takes effect after the current test step is completed and does not have a time limit.

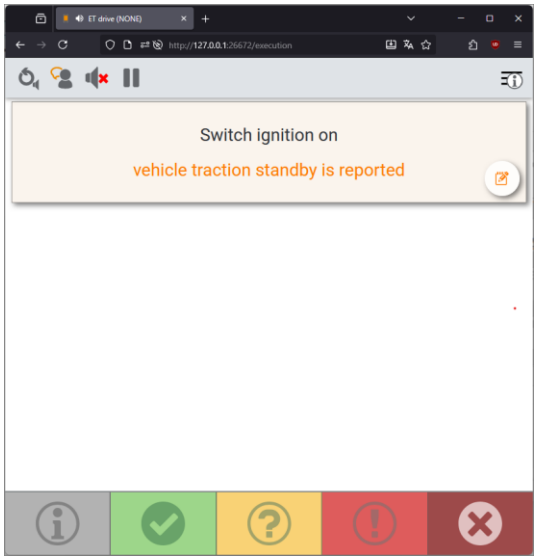


Figure 19: Pausing test cases

4.8 ecu.test *lab*

Support for library workspaces

et

Previously, packages and configurations could only be referenced from the current workspace. Now, artifacts from referenced library workspaces are also available for selection.

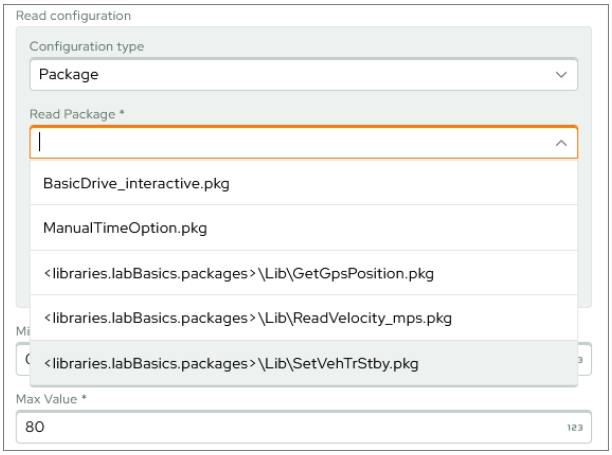


Figure 20: Support for library workspaces

In addition, **custom widgets** and views (as references) are loaded from library workspaces and offered for selection.

4.9 Communication

New simulation feature for SOME/IP – Dynamic calculation of return values for service methods

et

The mechanism for offering methods introduced in **ecu.test** 2025.3 is now able to calculate return values dynamically depending on the method input values. The familiar code completion mechanism is available for using this function.

The function can be used with the SERVICE-PCAP port on the tracetronic: Ethernet tool or the SERVICE port on the Vector: XL-API tool.

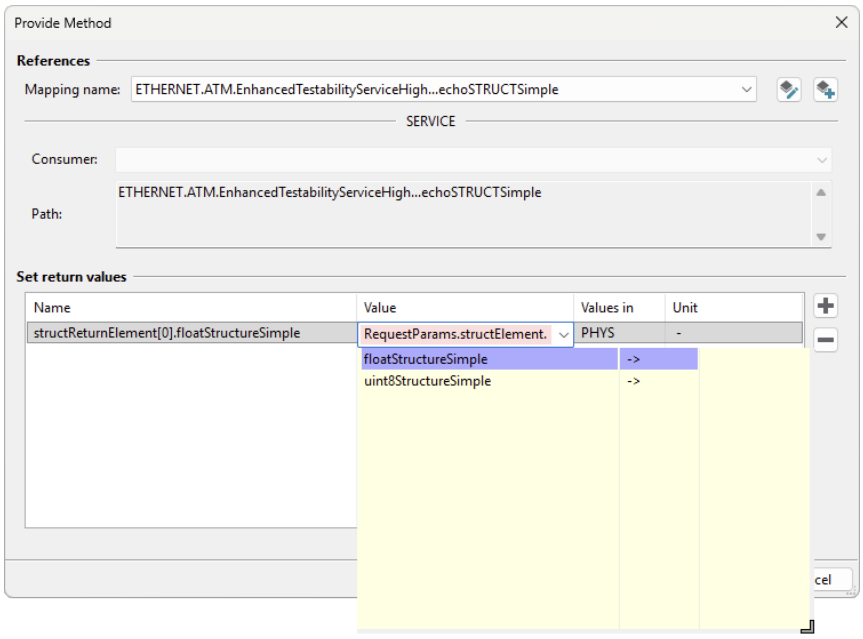


Figure 21: Dynamic calculation of return values for service methods

DLT performance

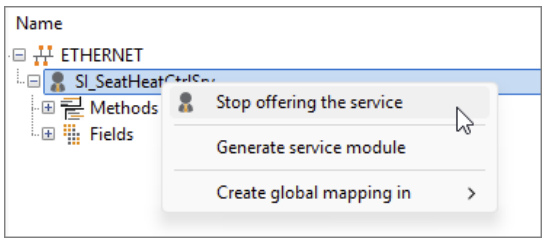
et

The latency for accessing passively read DLT communication has been reduced. At the same time, data throughput during recording in DLT format has been noticeably accelerated.

This improvement has a particularly positive effect at high data rates.

Stopping an offered SOME/IP service

et



In order to test the response of a service consumer in the event that the service provider discontinues its service, there is a new service test step that ensures that the service is no longer offered.

Figure 22: Stop offering the SOME/IP-Services

As a consequence, the sending of events is stopped and method requests are no longer responded to.

4.10 Trace analysis

MDF4: Ethernet recordings with unsorted data streams

et tc

For CANape Ethernet recordings, data streams may be stored in an unsorted order within the recording.

ecu.test now supports these and reads them correctly.

4.11 trace.xplorer

New capture interface in Wireshark for CAN-FD via PEAK

et tc

During setup and commissioning of real or virtual test benches, the need for manual testing or debugging arises regularly. In particular, test bench engineers and integrators want to be able to monitor and analyze the data traffic of Ethernet and classic buses live.

Since **ecu.test** 2024.3, **ecu.test** offers the option of using the free and established network analysis software **Wireshark** for this purpose.

The **trace.xplorer** allows additional capture interfaces to be created in Wireshark for capturing Ethernet, CAN, CAN-FD, or LIN with this tool.

The range of available interfaces has now been expanded with the option of capturing CAN-FD data traffic via hardware measurement boxes from PEAK-Systems.

This interface can be easily set up via a dialog in **trace.xplorer**.

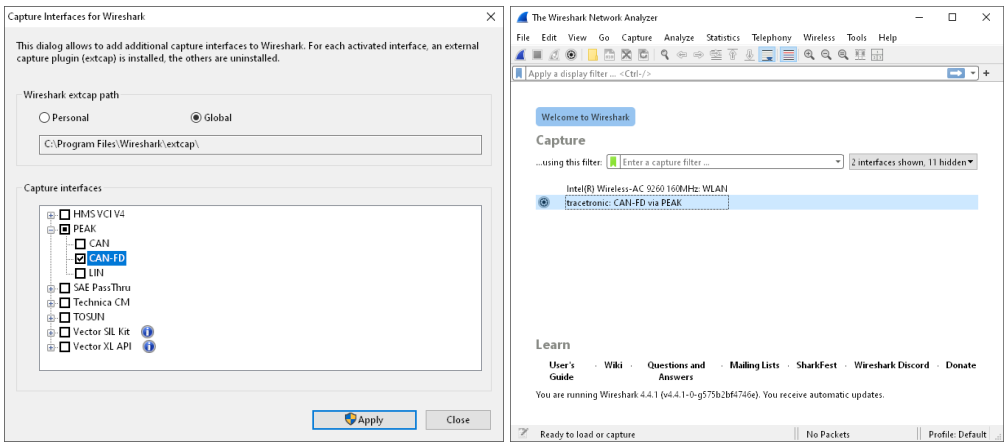


Figure 23: Configuring the capture interface in trace.xplorer (left) and using it in Wireshark (right)

Note: Using this feature requires a **trace.xplorer** license and a suitable measurement box.

Absolute time information available in ASTRACE files

et tc

Testers want to read a timestamp from the test step in the report and then go directly to the corresponding position in the recorded traces.
The easiest way to do this is to use absolute timestamps, which can already be displayed for test steps in the test report.

However, ASTRACE files generated by the trace analysis and signal export trace step previously contained only relative timestamps (measurement time).

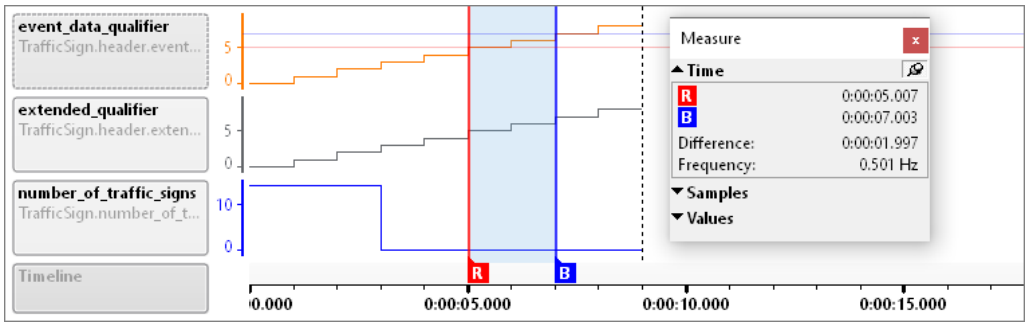


Figure 24: Document view with relative timestamps (measurement time)

This has been changed. If available, the (synchronized) absolute time information (date and time) from the trace analysis is now also written to the ASTRACE files.

In **trace.xplorer**, each user can decide for himself which timeline he wants to see: relative or absolute time. This can be changed either via the context menu of the timeline or in the program options.

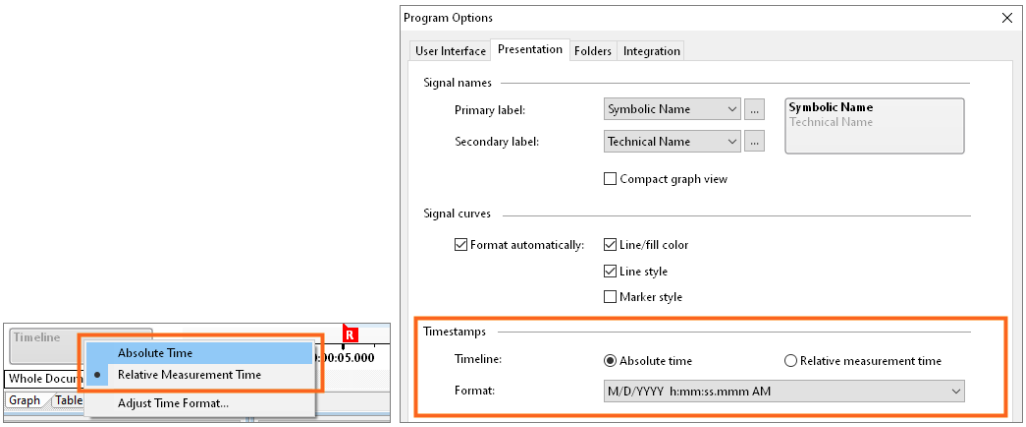


Figure 25: Options for switching between relative and absolute timeline

The selected setting applies to all documents opened by the user. It affects many areas of the application, for example, the graph and table views, the flag list, or the **Measure** toolbox.

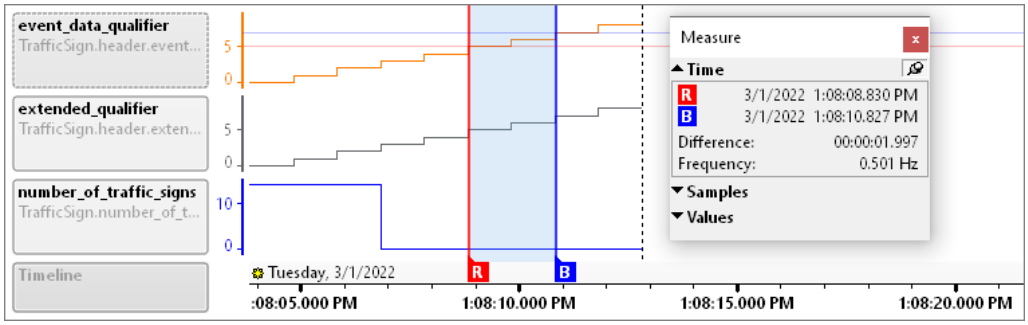


Figure 26: Document view with absolute timestamps (date and time)

Quickly show and hide flag list entries in views

et tc

The flag list sometimes contains a large number of entries for markers (cursors, flags, chain and signal dimensionings, shades), all of which are visualized in different ways in the views.

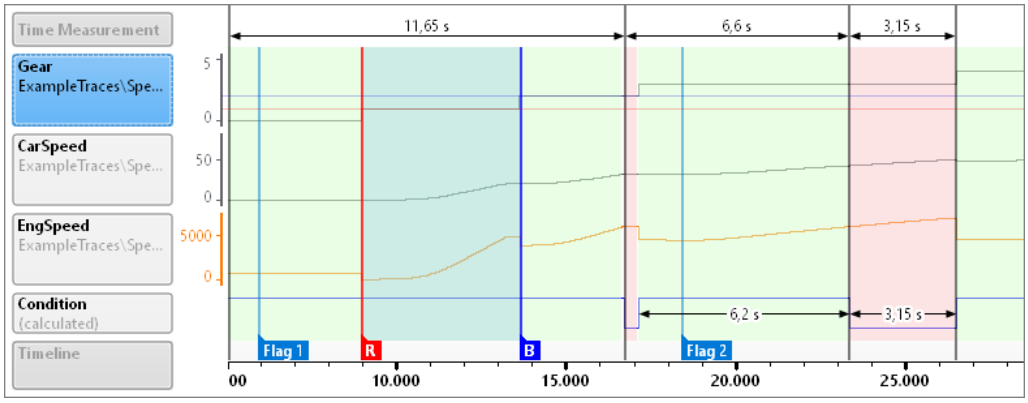


Figure 27: Graph view with all markers visible

To help you keep your focus, the flag list offers two new buttons in its toolbar for quickly showing and hiding markers. Clicking one of the buttons hides or shows **all markers**, regardless of their type or current visibility.

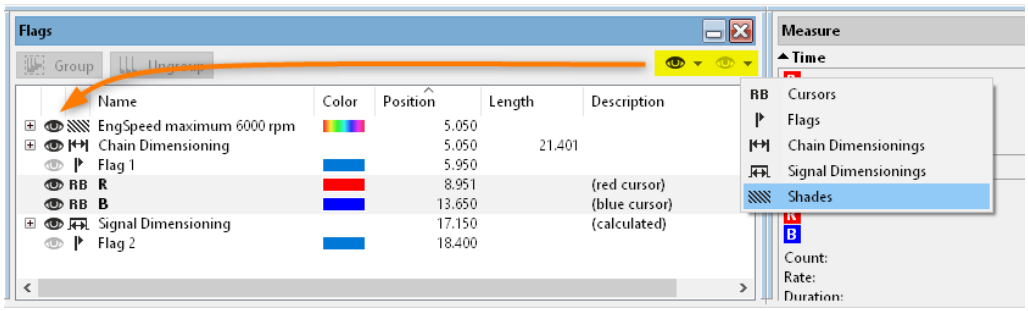


Figure 28: Flag list with buttons for showing and hiding markers

Each button has an additional context menu. This menu contains entries for all available marker types, each of which only changes the visibility of all **entries of the selected type**.

This allows you to show or hide specific markers with a single click, without having to search for and handle these entries individually.

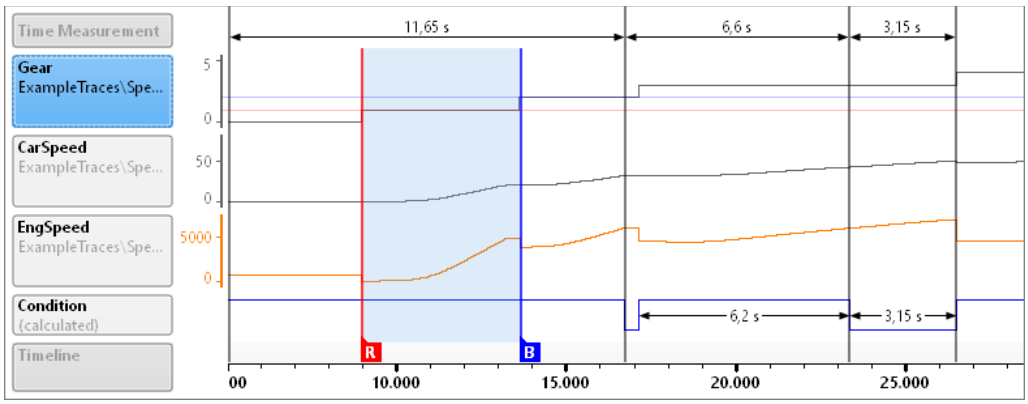


Figure 29: Graph view with flags and shades hidden

5 Tools and Interfaces

5.1 New tool versions

et

	Provider	Website	System	Product name	Version
1	**dSPACE**	Release link	Sensor-based realistic simulation for ADAS/AD in real time	**AURELION**	**24.2 until 25.2**
2	**Vector**	Release link	Development, testing, and analysis tools for automotive SiL and HiL projects	**CANoe/ CANalyzer**	**19SP3**
3	**Vector**	Release link	Measurement, calibration, diagnostics, and flash software	**CANape**	**23**

5.2 APIs

5.2.1 Internal API

Querying mappings via the Internal API

et tc

The new **scope** parameter allows you to retrieve mapping names or mapping targets for a specific area using:

- **api.TestEnvironment.ExecutionInfo.GetMappingNames**
- **GetMappingTargetPath**

For example, with **scope**="global," it is possible to access the mapping target of a global mapping, even if the mapping itself is not used in the current test case.

5.2.2 Test Management API

Extended Hierarchical Import Functionality for ecu.test Test Management API

et

Many test-management tools organize test cases in nested folders, packages or modules that mirror the product's logical structure.

To expose this hierarchy, the import methods of the **ecu.test** Test Management API have been extended with tree-aware functionality, allowing the same folder-and-package layout to be displayed during both package and project imports.

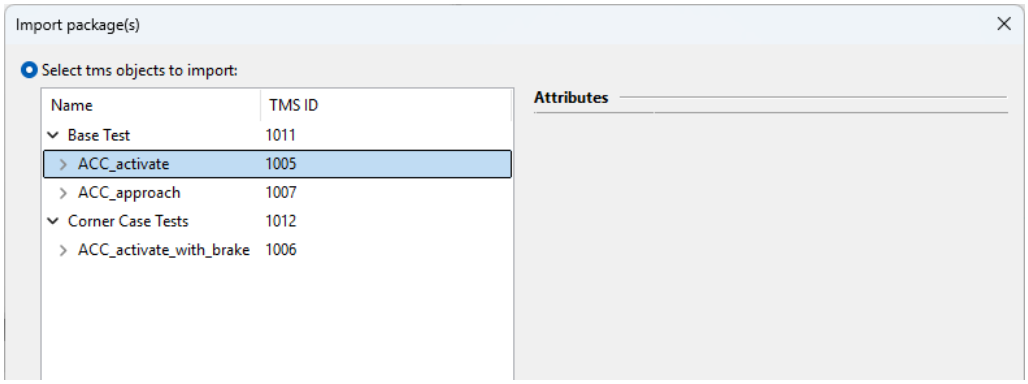


Figure 30: Hierarchical folder and package structure when importing packages and packages

5.2.3 UserTool API

Python library for IDE support of ecu.test-specific user code

et tc

ecu.test can be flexibly extended for different workflows using custom code. This often requires implementing specific interfaces. A first step toward more sustainable development of these modules is the newly available Python library.

You can install it in your own Python environment, which enables autocompletion and type checking in your IDE thanks to the provided interfaces.
The library is available as a prototype in the installation directory under "res" as a Python wheel. Currently, it supports custom checks and user-defined synchronization of recordings.

6 Discontinuations

6.1 Discontinued features and incompatibilities in this version

Fibex support

et

Fibex support for Bus is being discontinued and will be removed with **ecu.test 2025.4**. Fibex support for DLT will continue to be provided.

Discontinuation of the ReqIf Import

et

With **ecu.test 2025.4**, the functionality for importing ReqIf data as a package and project attribute will be removed.

Job SetFlowControl

et

The SetFlowControl job has been completely removed from CAN (-FD) and FlexRay ports.
The following jobs can be used as alternatives:

- **SetFlowControlCTS**
- **SetFlowControlWait**

6.2 Discontinued features in future versions

KS: Tornado only via ASAM ACI

et

The tool connection will be removed with **ecu.test 2026.1**. It will be replaced by the new connection based on ASAM ACI, which is available since **ecu.test 2023.3**.

Discontinuation of the Integrated Test Management Connection for Jama

et

The permanently integrated connection to Jama in **ecu.test** will be removed with **ecu.test 2026.2**. Please use the new Python-based connection instead.

Jobs RequestSeed und SendKey

et

The **RequestSeed** and **SendKey** jobs are replaced.

- RequestSeed → SecurityAccessRequestSeed
- SendKey → SecurityAccessSendKey

The new jobs also support the Seed & Key DLLs.

Alternative report directory for separate subproject execution

et tc

Currently, it is possible to specify the report folder for separate subproject execution. This feature is marked as deprecated in version **2025.3** and will be removed in **ecu.test 2026.1**.

Discontinuation of the FEP2 connection

et

The FEP2 connection will be removed in **ecu.test 2026.1**.

Playbook-Export from ecu.test to test.guide

et

The playbook export from **ecu.test** to **test.guide** will be removed in **ecu.test 2026.1**.

Port BUSACCESS – GENERIC_MAPPINGFILE

et

The newly introduced port type, "BUSACCESS - MODEL BASED," is much more flexible, maintainable, and intuitive to configure. To clarify the available options and guide users to the best solution, we are removing "BUSACCESS - GENERIC_MAPPINGFILE" in **ecu.test 2026.1**.

Support for Ubuntu 20.04 LTS and 22.04 LTS

et

With **ecu.test 2026.1**, support for Ubuntu 20.04 LTS and 22.04 LTS will be discontinued.

Supported versions of MATLAB/Simulink in Linux

et

As part of the discontinuation of support for older versions of Ubuntu with **ecu.test 2026.1**, support for MATLAB/Simulink up to and including R2023b will be removed under Linux. These versions do not offer support for Ubuntu 20.04. Support for Windows remains unchanged for versions from R2015b onwards.

The ICMP-RAW port is being discontinued for the Ethernet, XL-API, and SIL-Kit tools.

et

The port will be removed with **ecu.test 2026.1**.