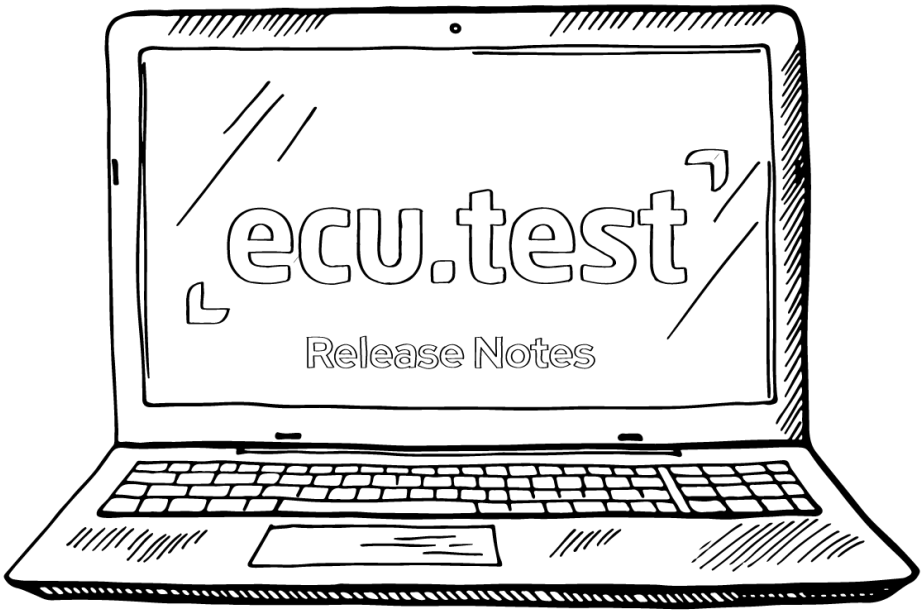
tracetronic



Release Notes

ecu.test 2025.4

trace.check 2025.4

Datum: 09.12.2025

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

tracetronic

Inhalt

Überblick 1

1 Highlights in ecu.test 2025.4 2

2 Sneak Preview 7

3 Usability 11

4 Testaspekte 15

4.1 Multimedia 15

4.2 HiL 16

4.3 Testmanagement 17

4.4 ecu.test *calibration* 17

4.5 ecu.test *code* 18

4.6 ecu.test *diagnostics* 18

4.7 ecu.test *drive* 19

4.8 ecu.test *lab* 20

4.9 Kommunikation 21

4.10 Traceanalyse 22

4.11 trace.xplorer 22

5 Versionen und Schnittstellen 26

5.1 Neue Tools und Versionen 26

5.2 APIs 26

5.2.1 Internal API 26

5.2.2 Test Management API 27

5.2.3 UserTool API 27

6 Abkündigungen 28

6.1 Abkündigungen und Inkompatibilitäten in dieser Version 28

6.2 Abkündigungen in zukünftigen Versionen 28

Überblick

Das neue Release **2025.4** ermöglicht intelligentere Automatisierung, flexiblere Workflows und leistungsstarke Features für eine noch effizientere Arbeit mit **ecu.test** und den **ecu.test** Extras.

Der **ecu.test** *agent* generiert Testfälle mit neuen Funktionen jetzt noch robuster, realitätsnäher und präziser. Beispielsweise werden

- Schleifen in Spezifikationen automatisch erkannt und umgesetzt,
- Signale schneller und zuverlässiger identifiziert,
- generierte Code-Blöcke visuell gekennzeichnet oder
- ein breiteres Spektrum an LLMs unterstützt.

Neu im Early Access: KI-gestützte Traceanalyse – einfach beschreiben, was analysiert werden soll und der Agent generiert die Analyse automatisch.
Die kompakten News zum **ecu.test** *agent* gibt es hier.

Mehr Performance und Raum für individuelle Nutzung gibt es bei **ecu.test** *lab*:

- deutlich schnellere Package-Ausführung durch neue Logik
- Wiederverwendung erstellter Views
- Einbinden eigener Widgets via API

Die Traceanalyse ist beim Handling großer Signalmengen spürbar performanter geworden. So können jetzt **Messlisten als XAM-Dateien** definiert und direkt in der Signalaufnahme hinzugefügt werden. Das ist deutlich effizienter als alle Signale per Drag & Drop in den Tab **Signalaufnahme** zu ziehen.

In diesem Quartal feiern wir auch den **Launch von one:cx** – unserer modular aufgebauten Plattform für die Test- und Integrationsautomatisierung. Sie vereint fragmentierte Tools, Daten und Prozesse in einem zentralen Testökosystem. AI-gestützt, cloud-basiert und unbegrenzt skalierbar. Das Ergebnis: kontinuierliche Innovation, weniger Fehler und deutlich kürzere Entwicklungszyklen. Mehr Infos dazu gibt es auf unserer Webseite.

Sneak Preview: Mit dem **test.spec** *agent* führen wir einen weiteren KI-gestützten Assistenten ein, der Testspezifikationen automatisch aus Requirements generiert. Dazu: Multi-FMU-Simulationen für komplexe Integrations- und Systemtests. Schon jetzt ist beides auf Anfrage verfügbar!

Alle weiteren zahlreichen Neuerungen und Erweiterungen und werden nachfolgend detailliert beschrieben.

Hinweis: Die Icons zeigen, für welches Produkt ein Thema relevant ist:
ecu.test **trace.check**

1 Highlights in ecu.test 2025.4

ecu.test *agent*

Im aktuellen Release liegt der Fokus auf einer deutlich gesteigerten Qualität bei der automatischen Testfallgenerierung. Die wichtigsten Neuerungen:

- **Unterstützung von Schleifen**
  Der Agent kann nun wiederholende Abläufe (Loops) in den Spezifikationen erkennen und entsprechende Testschritte generieren. Das erhöht sowohl die Vollständigkeit als auch die Realitätsnähe der erstellten Testfälle.
- **Visuelle Kennzeichnung generierter Blöcke**
  Jeder vom **ecu.*test*** *agent* erzeugte Code-Block wird jetzt durch ein spezielles Icon markiert. Dadurch sind automatisch generierte Abschnitte sofort erkennbar und lassen sich einfacher überprüfen oder anpassen.
- **Optimiertes Finden von Signalen**
  Die Erkennungslogik für Signals (z. B. Event-Trigger, Zustandswechsel) wurde überarbeitet. Signale werden schneller und zuverlässiger identifiziert, was zu präziseren Testabläufen führt.
- **Erweiterter ecu.test agent connector**
  Der Connector unterstützt nun ein breiteres Spektrum an Large Language Models (LLMs). Dadurch können unterschiedliche LLM-Backends nahtlos eingebunden werden, was Flexibilität und Skalierbarkeit erhöht.
- **Nutzung von Tool- und Port-Jobs ohne globales Mapping**
  Der Agent kann jetzt Tool- und Port-Jobs verwenden, die nicht zuvor im globalen Mapping definiert wurden. Das ermöglicht eine schlankere Konfiguration und reduziert den Aufwand für das Anlegen von Mapping-Einträgen.

Mit diesen Verbesserungen liefert der **ecu.test** *agent* jetzt robustere, leichter nachvollziehbare und besser an die jeweilige Entwicklungsumgebung angepasste Testfälle.

KI-gestützte Traceanalyse mit ecu.test *agent*
Der Einstieg in die Traceanalyse geht ab sofort noch leichter! Jetzt kann die gewünschte Analyse beschrieben und mit KI-Unterstützung generiert werden.

Gehöre zu den ersten, die dieses Feature testen!
Schreib eine kurze Mail an unseren Support und wir schalten dir den Early Access frei. Wir freuen uns auf dein Feedback!

ecu.test *lab*

Verbesserte Packageausführung

Bisher konnten bereits Packages aus **ecu.test** *lab* ausgeführt werden. Die Ausführung war jedoch auf wenige Widget-Typen beschränkt und ging mit deutlicher Latenz einher. Durch eine neue Ausführungslogik in **ecu.test** ist es nun möglich, Packages aus **ecu.test** *lab* heraus deutlich schneller auszuführen.

Damit lassen sich Packages auch für mehr Anwendungsfälle in **ecu.test** *lab* einsetzen. Dadurch sind Packages nun auch für das Lesen von Werten in Bar, Slider, Gauge und den neuen Custom Widgets verfügbar.

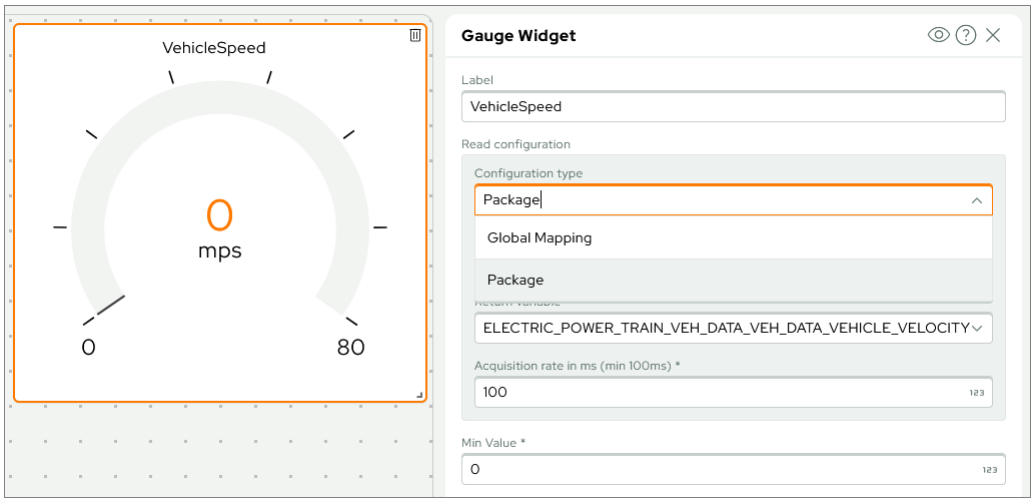


Abbildung 1: Verbesserte Packageausführung

Custom Widgets

ecu.test *lab* (kurz: *lab*) bietet bereits eine Auswahl an Widgets, die auch zukünftig kontinuierlich erweitert wird. Da der unmittelbare Bedarf nach mehr Gestaltungsmöglichkeiten und die Nachfrage nach individuellen Widgets groß ist, öffnen wir *lab* jetzt für eigene **Custom Widgets**.

So funktioniert's:

- Widgets werden in HTML/JavaScript erstellt und im **ecu.test**-Workspace abgelegt.
- Über eine von uns bereitgestellte API wird die Anbindung an *lab* vorgenommen, um das Widget mit Messwerten zu versorgen und um Aktionen im Widget direkt ins Testsystem zu übertragen.
- **Custom Widgets** werden wie die Standard-Widgets automatisch in der *lab*-Seitenleiste angezeigt und können dort konfiguriert werden.

Verfügbare Schnittstellen:

- Globale Mappings
- Packages

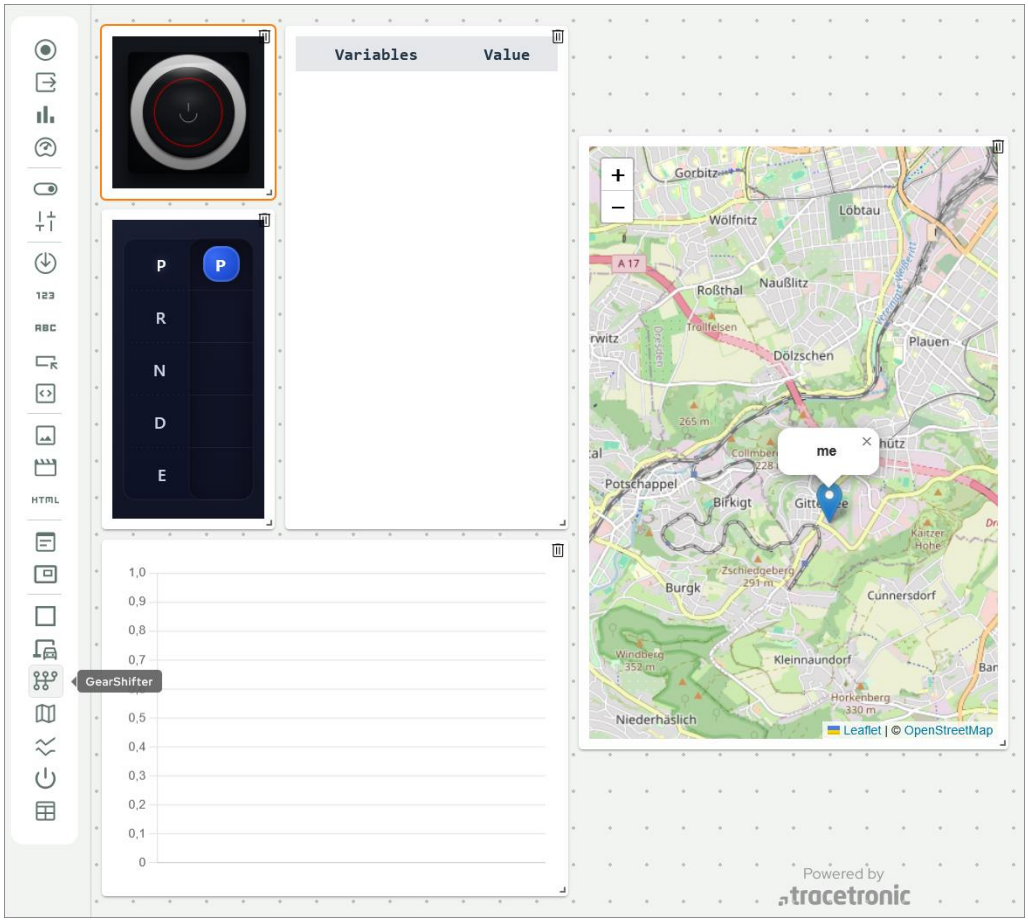


Abbildung 2: Einbinden von Custom Widgets

View Reference Widget
Bereits mit der Erstellung der zweiten View bemerken viele Anwendende: Der Anwendungsfall ist zwar ein anderer, aber viele grundlegende Bedienelemente werden eigentlich in jeder View benötigt.

Diesem Wunsch kommen wir jetzt nach und ermöglichen das Referenzieren eines (Bibliotheks-)View in einer anderen View.

So funktioniert's:

- Der (Bibliotheks-)View wird als ganz regulärer View erstellt, konfiguriert und gewartet.
- Dieser View kann über das **View Reference Widget** in beliebige andere Views eingebunden werden – auch wenn er in einem Bibliotheks-Workspace abgelegt ist.
- Der Inhalt der View ist als Referenz nicht veränderbar, die Bedienung außerhalb des Editiermodus funktioniert jedoch uneingeschränkt.

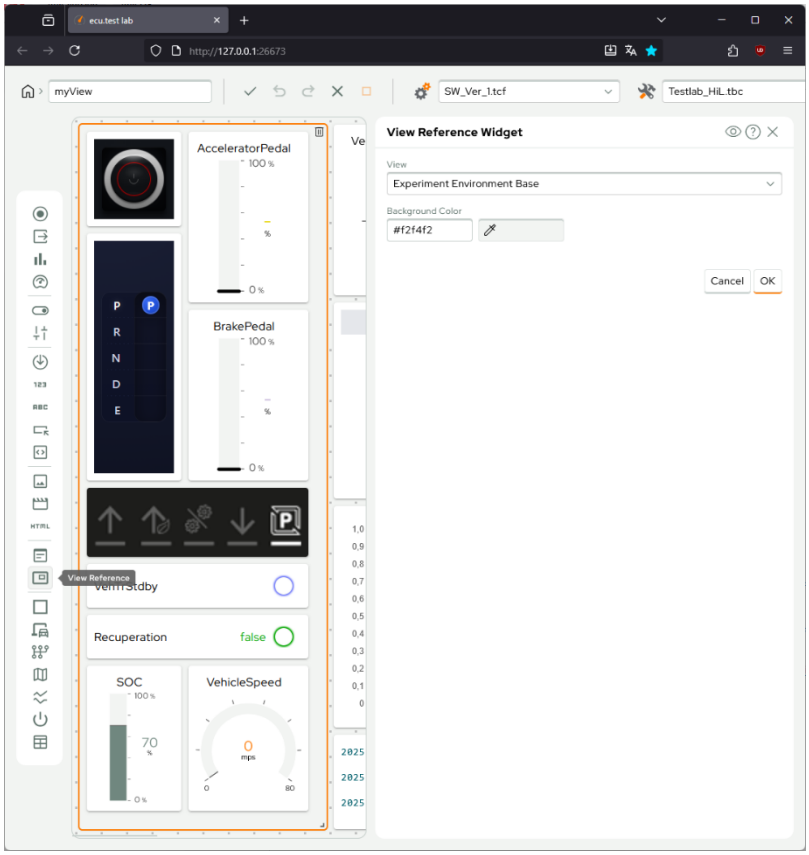


Abbildung 3: Referenzieren eines (Bibliotheks-)View in einer anderen View

Angabe von Messlisten für Signalgruppen

In **ecu.test** können aufzuzeichnende Signale im **Signalaufnahme**-Bereich hinzugefügt werden. Dies erfolgt entweder via Drag&Drop oder via Object API.

Bei einer großen Anzahl an Messgrößen kann dieses Vorgehen umständlich und in-performant sein. Aus diesem Grund gibt es ab sofort die Möglichkeit, die Signale als Mapping-Größen in eine XAM-Datei (bereits bekannt als Datenformat für globale Mappings) zu schreiben und diese Datei in der Signalaufnahme hinzuzufügen. Dieses Vorgehen bewirkt, dass alle Signale dieser Datei für die jeweilige Aufnahmegruppe registriert werden.

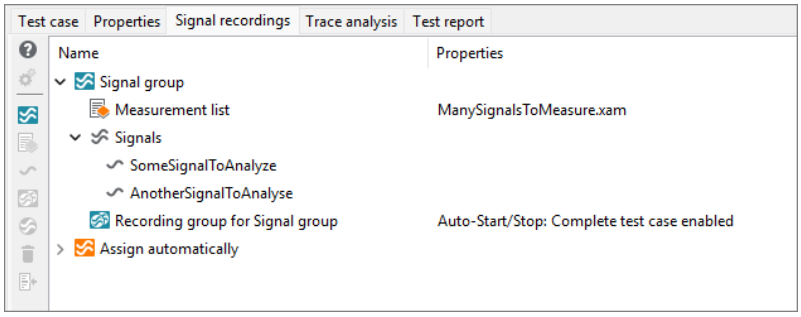


Abbildung 4: Messlisten für Signalgruppen

ecu.test in Japanisch

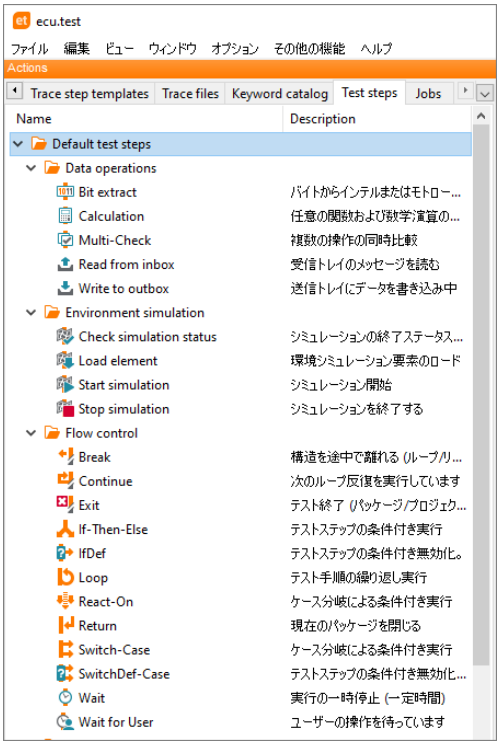


Abbildung 5: ecu.test mit japanischer GUI-Sprache

Seit Juni 2025 ist tracetronic auch in Japan vertreten. Passend dazu unterstützt unsere Software nun Japanisch als vierte Sprache neben Englisch, Deutsch und Chinesisch. Sowohl die Benutzeroberfläche als auch die Testfallsprache können damit vollständig in Japanisch genutzt werden.

Damit können japanische Anwenderinnen und Anwender unsere Software vom ersten Tag an in ihrer Muttersprache nutzen – für einen reibungslosen Einstieg und eine intuitive Bedienung. Gleichzeitig stärken wir unsere internationale Ausrichtung und legen den Grundstein für weitere lokale Anpassungen.

Abbildung 6: Japanische Infotexte in der GUI

2 Sneak Preview

Unser neuester AI-Service: test.spec *agent*

Kennst du das? Requirements durcharbeiten, Spezifikationen schreiben, Testfälle manuell ableiten. Was sonst Stunden dauert und mit Routinearbeiten verbunden ist, lässt sich jetzt erheblich vereinfachen und beschleunigen.

Der **test.spec** *agent* setzt ganz vorne in der Testpipeline an, indem er die Testspezifikationen durch den KI-Assistant aus den Requirements generieren lässt. Anschließend generiert der **ecu.test** *agent* Testfälle aus diesen Spezifikationen in Sekundenschnelle und spart damit zusätzlich massiv Zeit. Ein Novum!

Mit dem **MVP** des **test.spec** *agents* steht bereits heute ein leistungsstarkes, KI-basiertes Werkzeug bereit, das nicht nur automatisiert Spezifikationen liefert, sondern aktiv in Entwicklungsprozesse eingebunden werden kann.

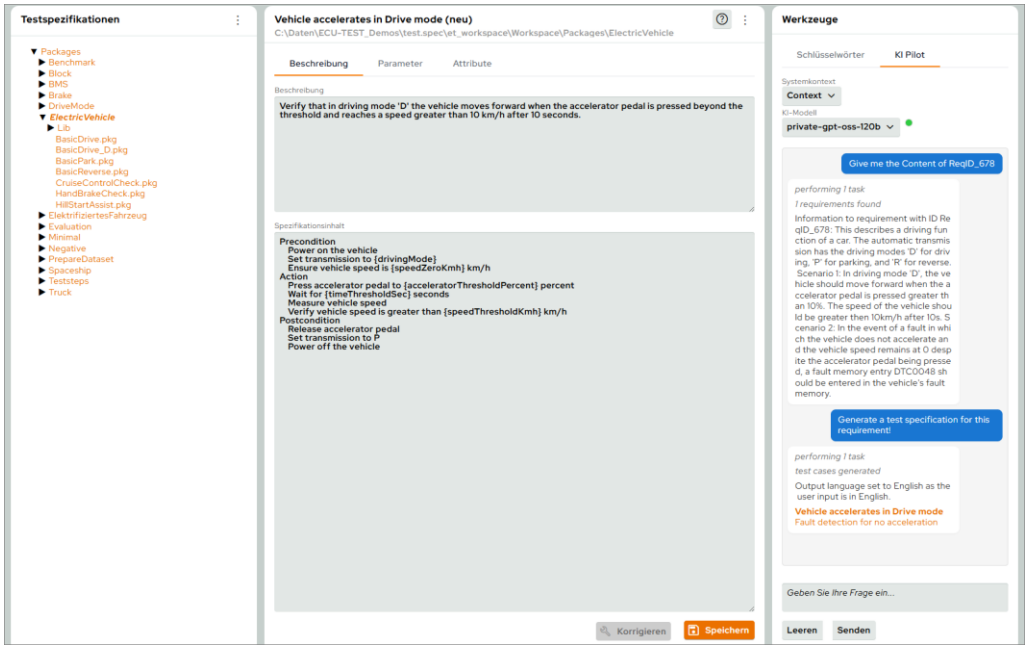


Abbildung 7: test.spec agent

Vorher:

1. Requirement lesen
2. Bibliothek suchen
3. Referenzen recherchieren
4. Spec schreiben
5. Review
6. Anpassungen vornehmen
7. Fertig

Dauer: ca. 2,5h pro Spezifikation

Jetzt:

1. Requirement hochladen
2. Spec generieren lassen
3. Prüfen, Rückfragen stellen
4. Nach **ecu.test** exportieren
5. Fertig

Dauer: ca. 15 Minuten

Welche Vorteile bietet der test.spec *agent*?

Von Stunden auf Minuten	Aus Requirements werden automatisch vollständige Testspezifikationen generiert, inklusive der Berücksichtigung passender Bibliothekspackages und Referenzen aus ALM-Systemen.
Interaktive Kommunikation	Der Agent arbeitet wie ein erfahrenes Teammitglied: erzeugt Spezifikationen, zeigt relevante Requirement-Inhalte, berücksichtigt Bibliothekspakete und beantwortet Rückfragen.
Weniger Fehler und Nacharbeiten	Der Agent nutzt vorhandene Bibliotheken und unternehmenseigene Knowledge-Bases, um vollumfängliche, wiederverwendbare Spezifikationen zu erstellen. Dadurch sind eine schnellere Validierung, höhere Testabdeckung und geringere Fehlerraten möglich.
Einheitliche, zuverlässige Daten	Der Zugriff auf ALM-Systeme, bestehende Testfall-Referenzen und weitere Quellen ist über den Model Context Protocol (MCP)-Standard definiert. Dadurch kann die Qualität und Konsistenz der Testspezifikationen durch bewährte Datenquellen garantiert werden.
Fundierte Qualität	Erhöhung der Spezifikationsgüte durch datengestützte Recherche (Deep Research) statt isolierter Betrachtung der Anforderungen.
Nahtlose Integration	Kombiniert mit dem **ecu.test** *agent* lässt sich das Tooling mühelos in bestehende Test- und Entwicklungsumgebungen einbinden, ohne zusätzliche Tools und komplizierte Schnittstellen.

Gestalte den test.spec *agent* mit!
Der aktuelle MVP ist bereits funktionsfähig – aber wir entwickeln ihn MIT dir weiter. Wie? Mit deinem Feedback, deinen Use cases, deinen Wünschen. Jede Idee fließt direkt in die nächste Version ein!

Willst du sehen, wie das mit deinen Requirements funktioniert? Dann lass dir von uns eine Demo zeigen!

- Live-Demo mit deinen Requirements und deiner Testumgebung
- Zugang zum aktuellen MVP – mit Installations-Guide und Schnellstart-Tutorial

Lass uns gemeinsam Testprozesse beschleunigen – wir freuen uns auf deine Anfrage!

Gleichzeitiges Simulieren mehrerer FMUs in ecu.test

et

Mit der Unterstützung für die gleichzeitige Simulation mehrerer FMUs, s. g. Co-Simulation, lassen sich nun auch komplexe Systeme von Modellen direkt in **ecu.test** abbilden.

Dies ermöglicht beispielsweise die Übergabe von Outputs einer FMU als Inputs an eine andere, oder die Kopplung mehrerer Modelle über den FMI Layered Standard Bus.
Für die Kopplung mehrere vECUs zur Absicherung in Integrations- und System-tests ist dies ein elementarer Baustein.

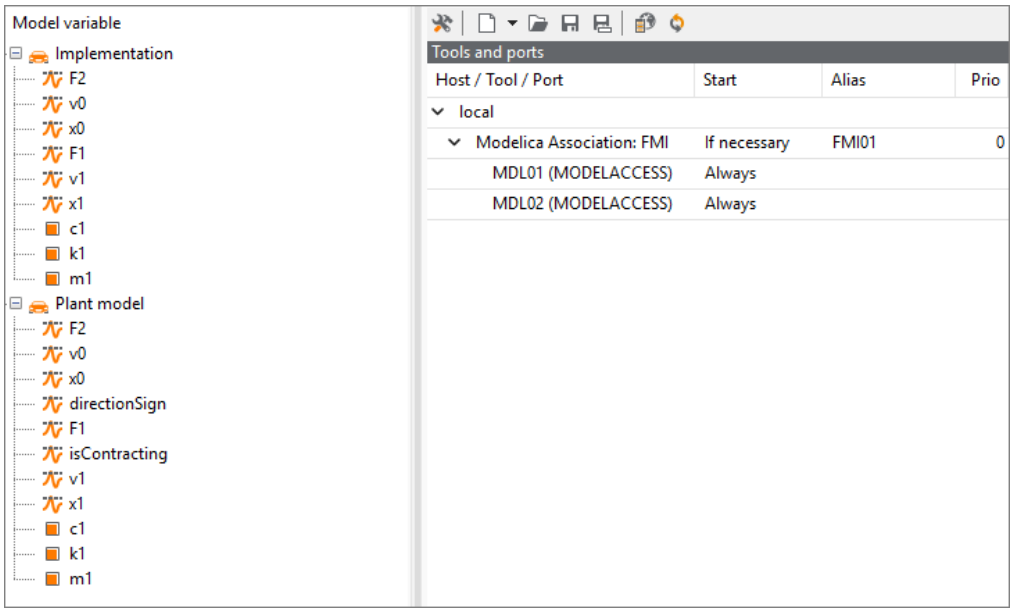


Abbildung 8: In TBC und TCF können mehrere FMUs konfiguriert werden, auf deren Größen im Model Access zugegriffen werden kann.

Die Umsetzung basiert auf dem SSP-Standard (System Structure and Parameterization). Signalflüsse und Verbindungen werden zentral über eine SSD-Datei definiert und in der Simulation berücksichtigt.

Die neue Funktion ist unabhängig vom LS-Bus verwendbar und unterstützt sowohl Szenarien mit, als auch ohne Bus-Kommunikation zwischen den FMUs.

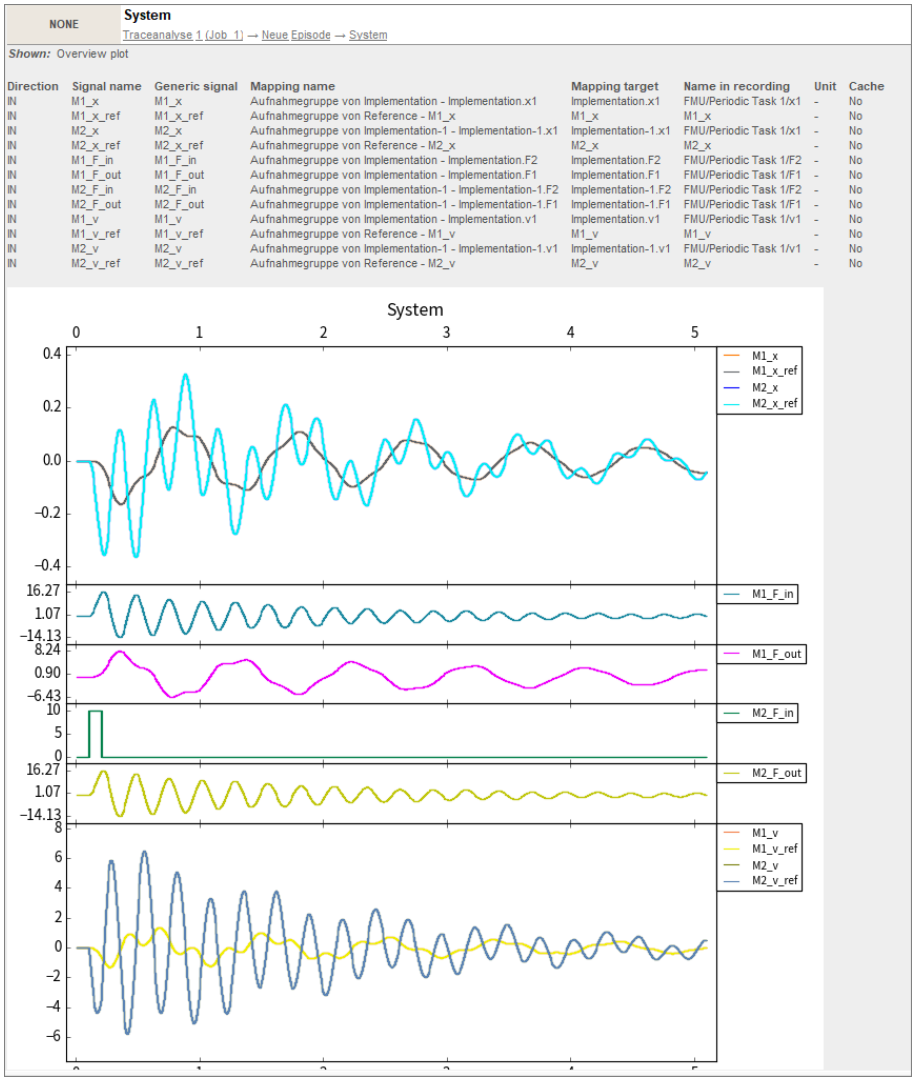


Abbildung 9 Die Kopplung zweier Feder-Masse-Dämpfer

Weitere Informationen zu Anwendungsfällen, Umsetzung und derzeitigen Einschränkungen bekommt ihr von uns auf Anfrage.

Zusätzliche Hilfe-Formate für KI-Chatbots

et tc

Die Anwenderhilfe ist jetzt in mehreren, KI-geeigneten Formaten verfügbar. Auf Anfrage können verschiedene Versionen, zum Beispiel als PDF mit und ohne Bilder, bereitgestellt werden.

Damit unterstützen wir moderne Nutzungsszenarien wie z. B. Chatbots und erleichtern die Weiterverarbeitung der Inhalte. Individuelle Eigenlösungen sind dadurch nicht mehr nötig.

3 Usability

Verbesserungen rund um die Test Case Coverage

et

Die Test Case Coverage erhält erneut etliche neue Features, mit denen die Nutzung nochmals deutlich verbessert wird. Dadurch können Bibliothekspackages nun noch besser abgesichert und die aktuelle Nutzung von Packages ganz allgemein analysiert werden.

- **REST API zum Parametrieren und Generieren des Reports:** Neue REST-Endpunkte wurden eingeführt, um das Laden, Löschen und Generieren von Coverage-Reports per API zu ermöglichen. Die API orientiert sich dabei am bisherigen GUI-Konzept. Dies vereinfacht die Automatisierung und Integration in bestehende Workflows und ermöglicht eine Einbindung in die Workflowautomatisierung von **test.guide**.

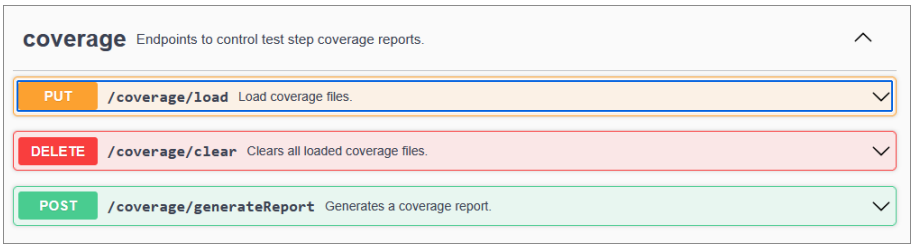


Abbildung 10: Neue REST-API-Endpunkte zur Coverage-Erstellung

- **Verbesserte Pagination:** In der Paginierung des Coverage-Reports ist es jetzt möglich, „alle“ Einträge auf einmal anzuzeigen.
  Die Bezeichnung **Items per page** wurde verständlicher formuliert, um die Bedienung zu erleichtern.

- **Filtern im Coverage-Report:** Der Coverage-Report unterstützt nun ein flexibles Filtern von Packages und Ordnern. Die gefilterten Elemente werden in ihrer ursprünglichen Struktur dargestellt, während irrelevante Elemente ausgeblendet werden können.

- **Prüfen des Coverage-Reports auf Package Revision:** Die Coverage-Report-Generierung prüft jetzt, ob die Revision des analysierten Packages mit den geladenen Coverage-Dateien übereinstimmt (über Git-Revision, Modifikationsdatum oder Hash). Bei Abweichungen wird eine Warnung ausgegeben und für ungespeicherte oder modifizierte Packages wird keine Coverage mehr erstellt.

- **Anzeige im Report, welche Coverages geladen sind:** Im Coverage-Report werden jetzt alle geladenen Coverage-Dateien übersichtlich angezeigt. Dadurch ist direkt ersichtlich, welche Dateien einbezogen wurden – inklusive der absoluten Pfade.
  Besonders hilfreich, wenn man die Coverage-Reports aus **test.guide** lädt!

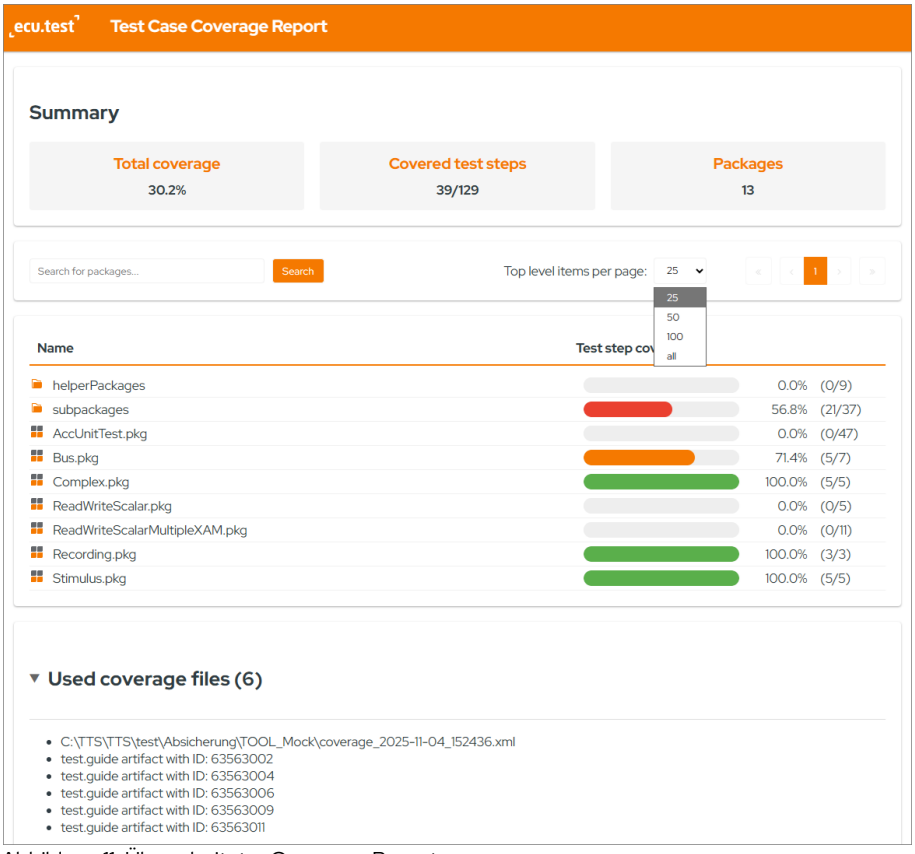


Abbildung 11: Überarbeiteter Coverage-Report

Bibliotheksworkspaces: Unterstützung der selektiven Projektausführung

et

Seit einigen Releases können **ecu.test**-Projekte auch in Bibliotheksworkspaces gepflegt werden, sodass Zusammenarbeit und Wiederwendbarkeit erhöht wird. Diese Projekte können nun auch mit der selektiven Projektausführung von **ecu.test** genutzt werden.

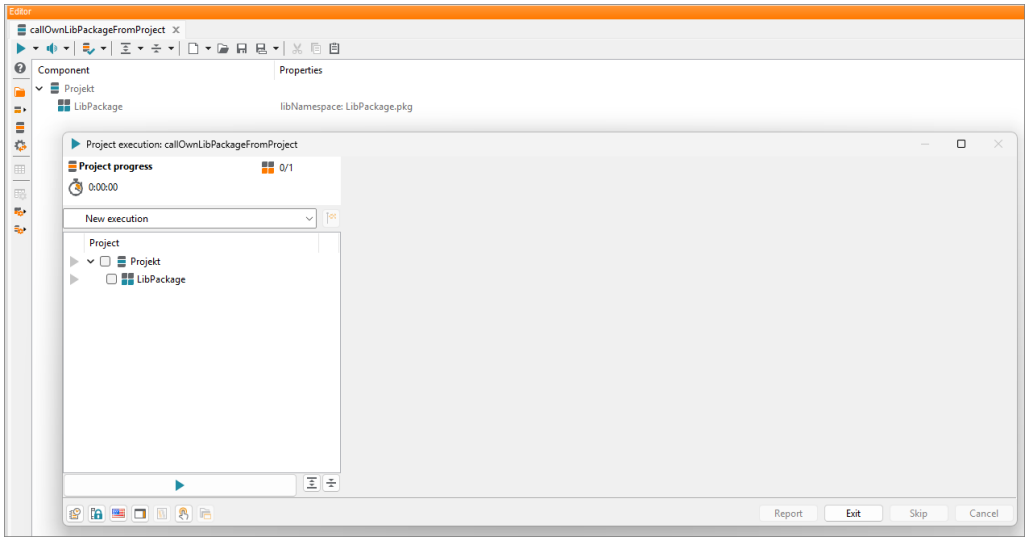


Abbildung 12: Selektive Projektausführung mit Projekten aus Bibliotheken.

Konfigurierbare Priorität der REST-API-Konstanten

et tc

Die Priorität von über die REST API gesetzten globalen Konstanten ist jetzt in der Testkonfiguration frei einstellbar.

Damit lassen sich diese Konstanten gezielt in der Prioritätenliste einordnen und stehen bereits beim Laden weiterer Konstanten zur Verfügung, sodass sie z. B. überladen oder in einem Skript (ConstantProvider.py) ausgewertet werden können.

Einerseits lassen sich Abhängigkeiten zwischen Konstanten so flexibler lösen. Andererseits sorgt dies für mehr Transparenz und Kontrolle beim Arbeiten mit globalen Konstanten in komplexen Testabläufen, z. B. via **test.guide**.

Die Konfiguration erfolgt im Editor der Testkonfiguration (TCF) und kann auch via Object API erfolgen.

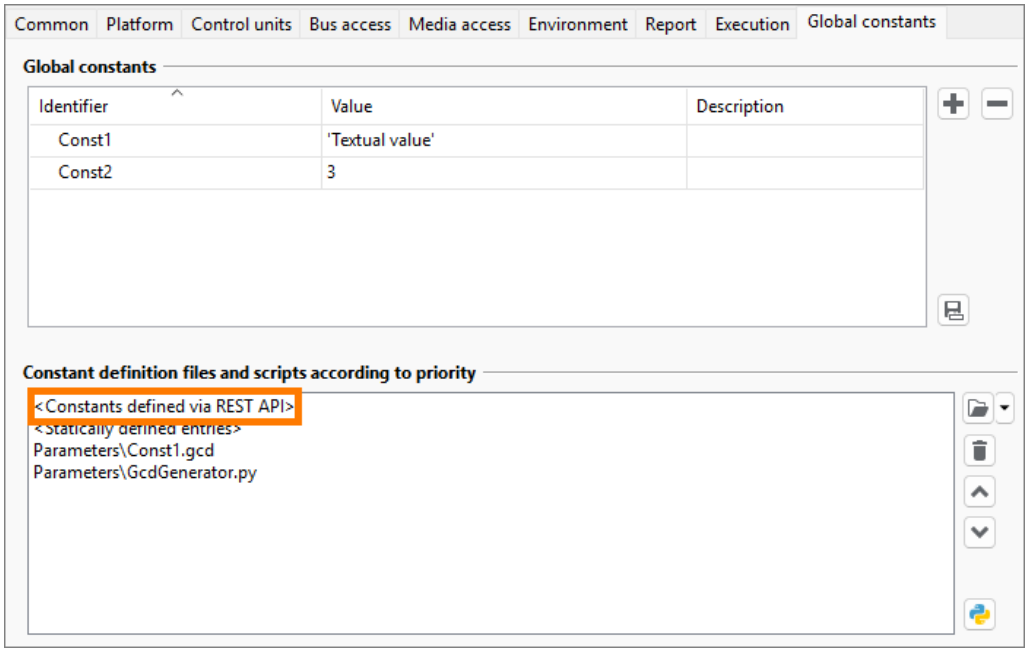


Abbildung 13: Priorität der via REST API übermittelten globalen Konstanten in der TCF

Manueller Abbruch von Recovery Packages

et

Mit Recovery Packages besteht die Möglichkeit, den Prüfstand zurückzusetzen, wenn Tests mit ERROR fehlschlagen.

Die Ausführung dieser Recovery Packages konnte bislang nicht abgebrochen werden. Mit dem neuen Release ist dieser Abbruch nun möglich.

4 Testaspekte

4.1 Multimedia

Umfangreiche Verbesserungen bei Tool Gen<I>Cam

et

Für die Standardschnittstelle Gen<I>Cam zur Ansteuerung von Kameras wurden weitreichende Verbesserungen implementiert.

Zunächst wurde die Performance bei der gleichzeitigen Verwendung von mehreren Kameras verbessert.
Des Weiteren können für die Kamera verschiedene Pixelformate ausgewählt werden. Es stehen z. B. RAW (Bayer Pixel Format), RGB oder Mono zur Verfügung. Formate wie RAW bieten dabei eine höhere Qualität, Mono hingegen spart Bandbreite ein.

Für eine einfachere Einrichtung der Kameras kann nun in der Konfiguration auf UserSets zugegriffen werden. Dies sind hinterlegte Einstellungsprofile in der Kamera, die z. B. die Belichtungszeit oder den Zoom beschreiben. Sie werden mit der zugehörigen Kamerasoftware erstellt.

Außerdem ist es jetzt möglich, Properties, wie z. B. Trigger, direkt per Job im Testfall auszuführen. Damit kann die Kamera noch flexibler im Testfall verwendet werden.

Einbinden von nutzerdefinierten Modellen zur Objekterkennung

et tc

Objekterkennung ist ein klassischer Anwendungsfall für die Nutzung von KI-Modellen. Mit der benutzerdefinierten Objekterkennung gibt es eine neue Schnittstelle, um eigene Modelle auf die im Testfall gelesenen Bilder anzuwenden.

Dazu können eigene Objekt-Detektoren erstellt werden. Über die Internal API können die Bildobjekte an diese Detektoren weitergegeben werden. Anschließend liefern diese dann eine Liste von erkannten Objekten.

Die genaue Verwendung kann der Anwenderhilfe unter Benutzerdefinierte Objekterkennung entnommen werden.

4.2 HiL

Playback von ASC-Aufnahmen auf CAN(-FD) Bus-Hardware

et

Es gibt unterschiedliche Gründe, bestehende Aufnahmen wiederzugeben. Beispielswiese wenn das Device-under-Test mit vormals aufgezeichneter Kommunikation erneut stimuliert werden soll. Oder um einen Testfall offline entwickeln und testen zu können, ohne dass eine real vorhandene Gegenstelle verfügbar ist.

Damit die Wiedergabe existierender Aufnahmen möglich ist, gibt es jetzt die neuen Jobs:

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

Beim Playback der ASC-Frames werden die Sende-Zeitstempel so genau wie möglich eingehalten.

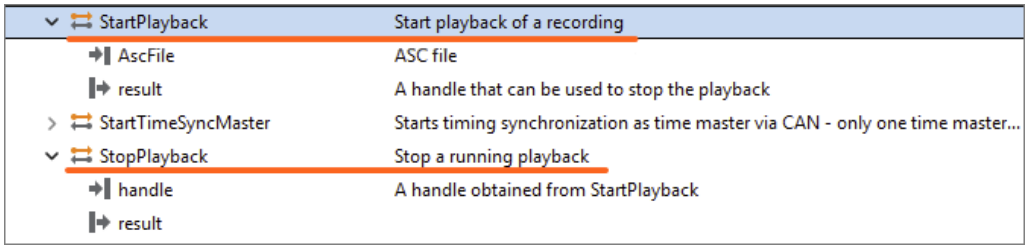


Abbildung 14: Neue Jobs zur Wiedergabe existierender Aufnahmen

Unterstützung der ZLG CAN-FD-Bushardwareanbindung

et

Alle Features der hardwarenahen Busanbindungen, inkl. **ecu.test** *diagnostics* und **ecu.test** *calibration*, können jetzt mit CAN-FD-fähiger Messhardware für CAN und CAN-FD von ZLG verwendet werden.

Hinweis: Messhardware, die ausschließlich CAN unterstützt, funktioniert derzeit noch nicht.

Unterstützung der PEAK CAN-FD Messhardware

et

Neben CAN unterstützen wir für PEAK jetzt auch CAN-FD-fähige Messhardware. Damit können ebenfalls alle Features der hardwarenahen Busanbindungen, inkl. **ecu.test** *diagnostics* und **ecu.test** *calibration*, verwendet werden.

4.3 Testmanagement

Jama Connect Beispiel: Hierarchische Darstellung in der Import-GUI

et

In Jama Connect werden Testfälle hierarchisch dargestellt, doch der Import-Dialog des **ecu.test**-Beispiel-Workflows zeigte bisher nur eine flache Liste. Das machte die Navigation umständlich und konnte zu langen Ladezeiten führen.

Der Beispiel-Workflow wurde daher so erweitert, dass der Import-Dialog jetzt die gleiche hierarchische Struktur wie in Jama Connect unterstützt.

Mit Lazy-Loading werden Unterkategorien erst geladen, wenn der entsprechende übergeordnete Eintrag geöffnet wird (Lazy-Loading). Dadurch sinkt die Anfangsladezeit und die Navigation wird wesentlich übersichtlicher und schneller.

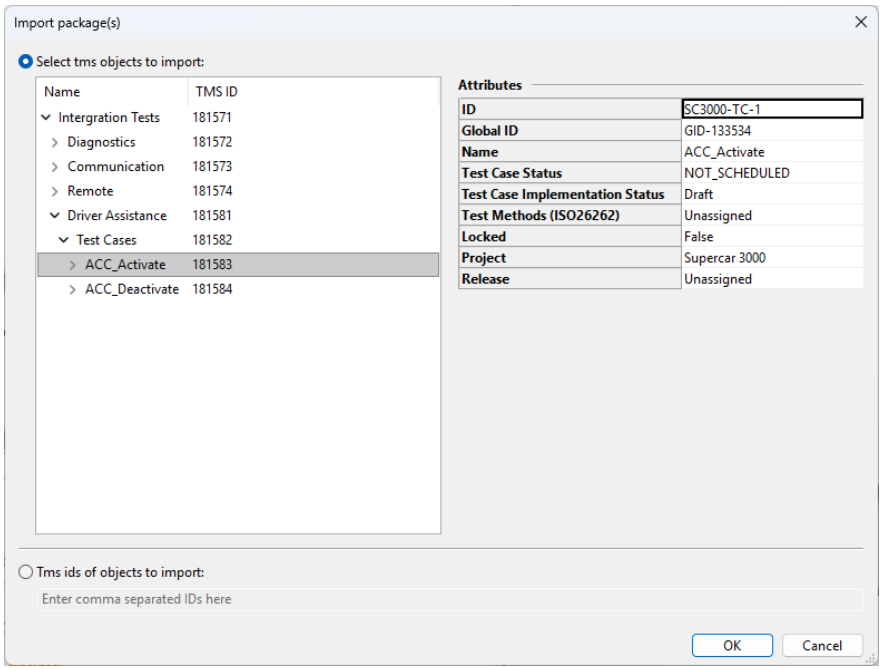


Abbildung 15: Import-Dialog unterstützt gleiche hierarchische Struktur wie in Jama Connect

4.4 ecu.test *calibration*

Angabe des Quellports bei XCP over Ethernet

et

Falls vom Steuergerät nur XCP-Verbindungen ausgewählter Quellports akzeptiert werden, können diese Portbereiche ab sofort in der Testbenchkonfiguration angegeben werden.

4.5 ecu.test *code*

Unterstützung von Aufnahmen

et

Um den Testablauf nachvollziehen und analysieren zu können, werden häufig Signalaufnahmen benötigt. Für diesen Zweck ermöglicht **ecu.test** *code* jetzt auch das Anlegen, Starten und Stoppen von Aufnahmen.

```
# 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()
```

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

4.6 ecu.test *diagnostics*

Trennung des Session Layers vom Application Layer

et

Um eine Non-Default Session aufrecht zu halten, muss der Service **TesterPresent** zyklisch gesendet werden. Dieser Service nimmt im Standard eine besondere Rolle ein, da er im Gegensatz zu den anderen Services, nicht sequenziell abgearbeitet wird, sondern jederzeit und ohne Verzögerung gesendet werden muss.

Technisch gesehen ist der **TesterPresent**-Service Teil des **SessionLayers**. Diese Besonderheit wurde mit der 2025.4 implementiert. Damit ist es möglich, dass **TesterPresent** Requests unabhängig von weiteren noch ausstehenden Requests versendet werden.

Insbesondere Randfälle, in denen eine parallel stattfindende Kommunikation länger als das Session Timeout dauert, sind damit adressiert.

4.7 ecu.test *drive*

ecu.test *drive* über das Extras-Menü öffnen

et

ecu.test *drive* kann ab sofort über das **Extras**-Menü direkt im Browser geöffnet werden.

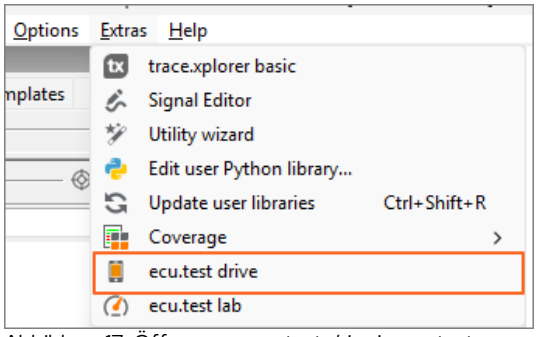


Abbildung 17: Öffnen von ecu.test *drive* in ecu.test

Verbesserung der Darstellung von Ist- und Sollwerten

et

Der Ist-Wert wird nun bei der Verwendung in Blocktexten und in Zeitoptionen auf drei Nachkommastellen gerundet dargestellt. Dies verhindert eine sehr unruhige Darstellung bei Rundungsfehlern und unterstützt die Lesbarkeit.

Manuelle Erwartungshaltungen wie "min <= value <= max" werden nun, soweit möglich, ausgewertet, um die Nachvollziehbarkeit der Erwartung für Testende zu steigern.

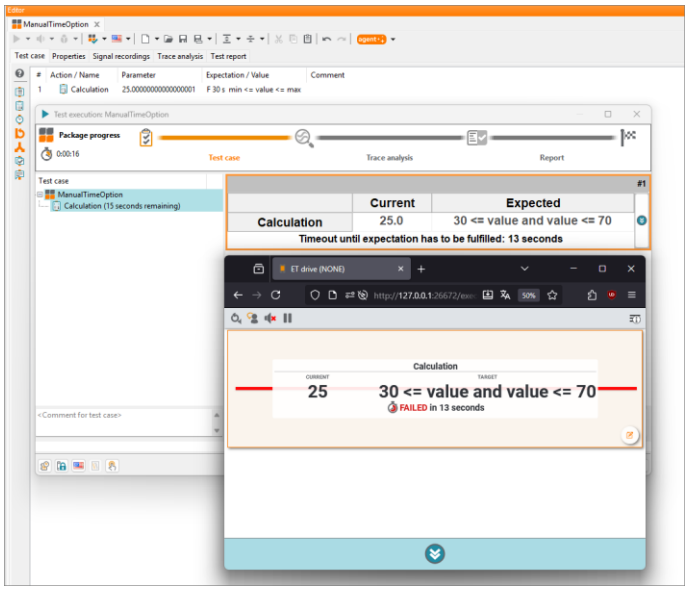


Abbildung 18: Darstellung von Ist- und Sollwerten

Pausieren von Testfällen

et

Testfälle können in **ecu.test** *drive* nun über einen Button in der Titelleiste pausiert werden. Die Pause tritt nach Abschluss des aktuellen Testschritts in Kraft und ist zeitlich unbeschränkt.

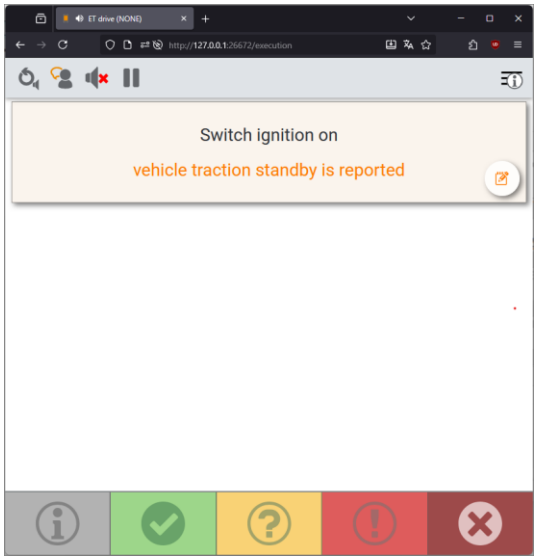


Abbildung 19: Pausieren von Testfällen

4.8 ecu.test *lab*

Unterstützung von Bibliotheksworkspaces

et

Bisher konnten Packages und Konfigurationen nur aus dem aktuellen Workspace referenziert werden. Ab sofort stehen auch Artefakte aus referenzierten Bibliotheksworkspaces zur Auswahl.

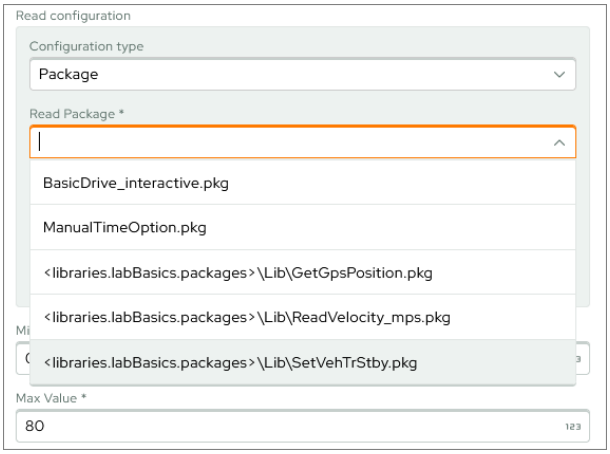


Abbildung 20: Unterstützung von Bibliotheksworkspaces

Außerdem werden **Custom Widgets** und Views (als Referenz) aus Bibliotheksworkspaces geladen und zur Wahl angeboten.

4.9 Kommunikation

Neues Simulationsfeature für SOME/IP – Dynamische Berechnung von Rückgabewerten bei Service-Methoden

et

Der in **ecu.test** 2025.3 eingeführte Mechanismus zum Anbieten von Methoden ist nun in der Lage, Rückgabewerte dynamisch in Abhängigkeit der Methoden-Eingabewerte zu berechnen.

Für die Verwendung der Funktion steht der bekannte Mechanismus für die Code-Vervollständigung zur Verfügung.
Die Funktion kann mit dem SERVICE-PCAP-Port auf dem Tool **tracetronic: Ethernet** oder dem SERVICE-Port auf dem Tool **Vector: XL-API** verwendet werden.

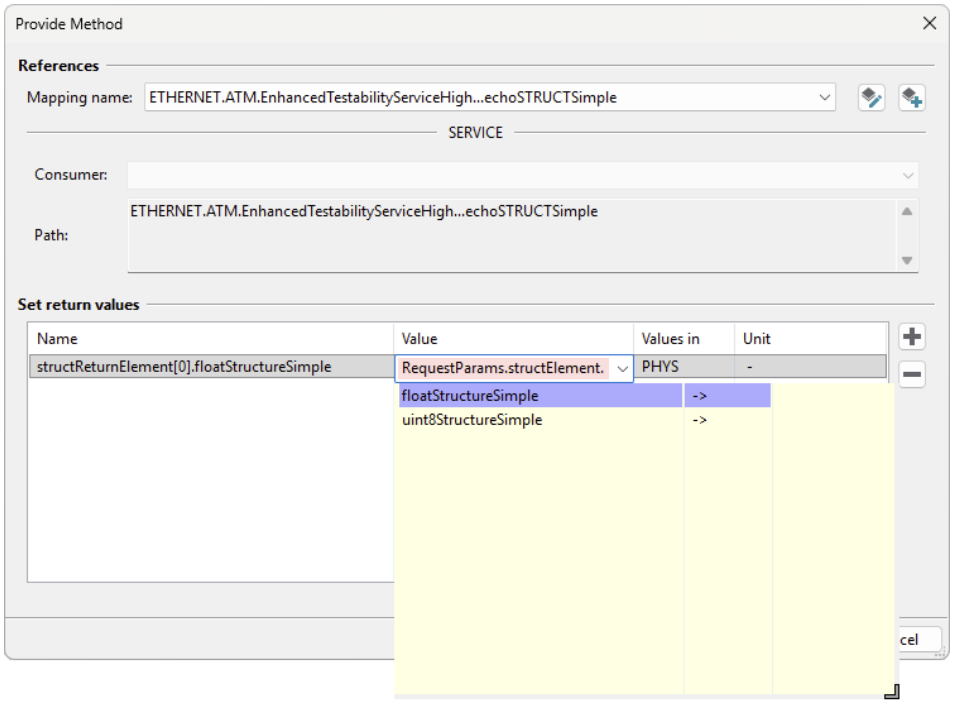


Abbildung 21: Dynamische Berechnung von Rückgabewerten bei Service-Methoden

DLT-Performance

et

Der Latenz für den Zugriff auf passiv mitgelesene DLT-Kommunikation wurde verringert. Gleichzeitig konnte der Datendurchsatz bei der Aufzeichnung im DLT-Format spürbar beschleunigt werden.
Die Verbesserung wirkt sich insbesondere bei hohen Datenraten positiv aus.

Abschalten eines angebotenen SOME/IP-Services

et

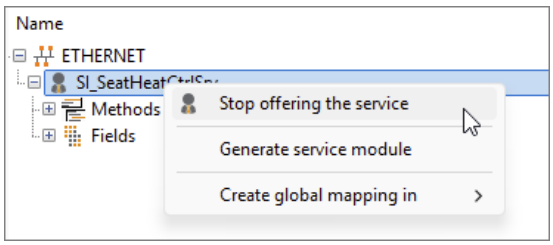


Abbildung 22: Abschalten des angebotenen SOME/IP-Services

Um die Reaktion eines Service-Consumer für den Fall zu prüfen, dass der Service-Provider seinen Dienst einstellt, gibt es einen neuen Service-Testschritt, der dafür sorgt, dass der Service nicht mehr angeboten wird.

Als Konsequenz wird das Versenden von Events gestoppt und nicht mehr auf Methoden-Requests geantwortet.

4.10 Traceanalyse

MDF4: Ethernet-Aufnahmen mit unsortierten Datenströmen

et tc

Für Ethernet-Aufnahmen durch CANape kann es vorkommen, dass die Datenströme unsortiert in der Aufnahme abgelegt werden.
Diese werden nun von **ecu.test** unterstützt und korrekt eingelesen.

4.11 trace.xplorer

Neue Aufzeichnungsschnittstelle in Wireshark für CAN-FD über PEAK

et tc

Während der Einrichtung und Inbetriebnahme von realen oder virtuellen Prüfplätzen entsteht regelmäßig Bedarf für manuelles Testen oder Debuggen. Insbesondere Prüfplatzverantwortliche und IntegratorInnen wünschen sich dann, den Datenverkehr von Ethernet und klassischen Bussen live mitlesen und analysieren zu können.

Seit **ecu.test** 2024.3 kann dafür die freie und etablierte Netzwerkanalyse-Software **Wireshark** genutzt werden. Über den **trace.xplorer** lassen sich in Wireshark zusätzliche Aufzeichnungsschnittstellen hinzufügen, um mit diesem Tool Ethernet, CAN, CAN-FD oder LIN zu erfassen.

Das Angebot verfügbarer Schnittstellen wurde nun um die Möglichkeit zur Erfassung des Datenverkehrs von CAN-FD über Hardware-Messboxen von PEAK-Systems erweitert. Die Einrichtung dieser Schnittstelle kann ganz einfach über den Dialog im **trace.xplorer** vorgenommen werden.

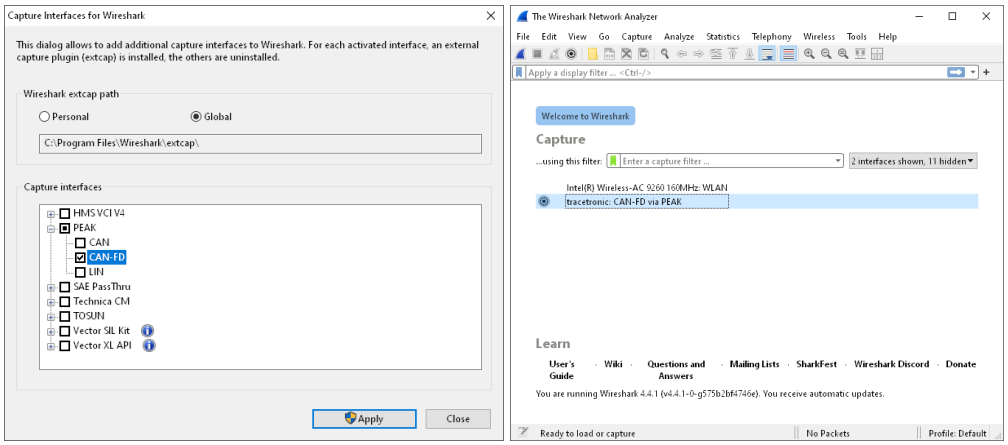


Abbildung 23: Konfiguration der Aufzeichnungsschnittstelle im trace.xplorer (links) und deren Verwendung in Wireshark (rechts)

Hinweis: Die Nutzung dieses Features erfordert eine **trace.xplorer**-Lizenz und eine passende Messbox.

Absolute Zeitinformationen in ASTRACE-Dateien verfügbar

et tc

Testende wollen gern vom Testschritt im Report einen Zeitpunkt ablesen und dann direkt an der richtigen Stelle in die aufgezeichneten Traces schauen. Die einfachste Möglichkeit hierfür bieten absolute Zeitstempel, die man sich schon für Testschritte im Testreport anzeigen kann.

ASTRACE-Dateien, die von der Traceanalyse und dem Signalexport-Traceschritt erzeugt werden, enthielten bisher aber ausschließlich relative Zeitstempel (Messzeit).

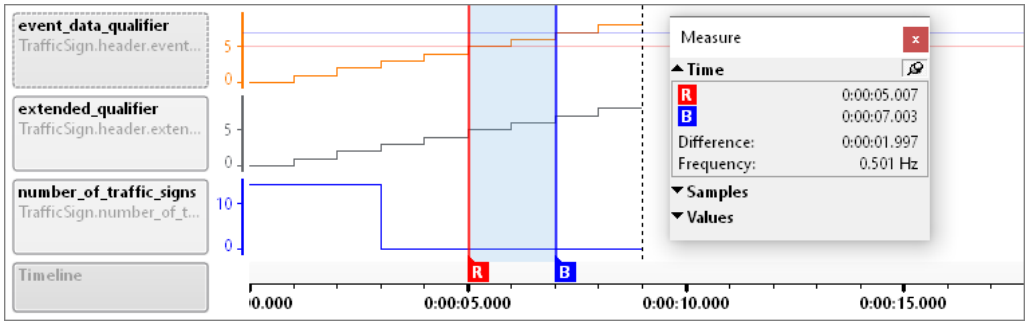


Abbildung 24: Dokumentansicht mit relativer Zeitachse (Messzeit)

Dies ändert sich jetzt. Sofern verfügbar, wird auch die (synchronisierte) absolute Zeitinformation (Datum und Uhrzeit) von der Traceanalyse in die ASTRACE-Dateien geschrieben.

Im **trace.xplorer** können Anwendende entscheiden, welche Zeitachse sie sehen möchten: die relative oder absolute Zeit. Die Umschaltung erfolgt entweder über das Kontextmenü der Zeitachse oder in den Programmeinstellungen.

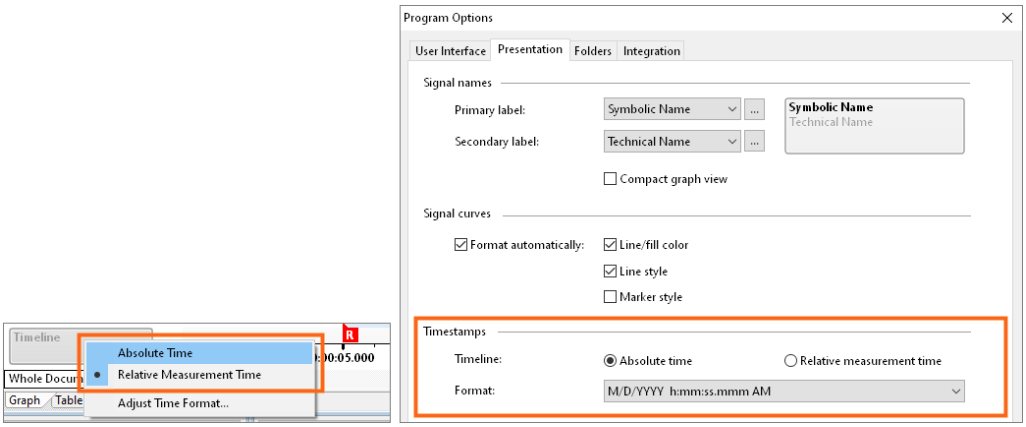


Abbildung 25: Möglichkeiten zum Umschalten zwischen relativer und absoluter Zeitachse

Die gewählte Einstellung gilt übergreifend für alle Dokumente, die von den Nutzenden geöffnet werden.
Sie wirkt sich auf viele Bereiche der Anwendung aus, zum Beispiel auf die Diagramm- und Tabellenansichten, die Flagliste oder das Andockfenster **Messen**.

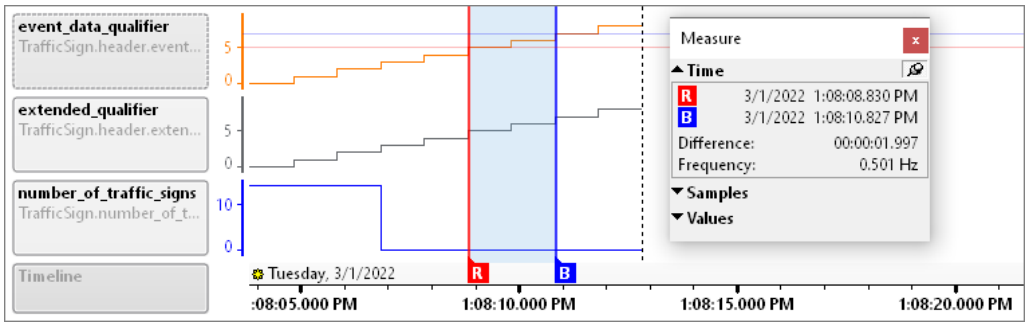


Abbildung 26: Dokumentansicht mit absoluter Zeitachse (Datum und Uhrzeit)

Flaglisten-Einträge in Ansichten schnell aus- und einblenden

et tc

Die Flagliste enthält mitunter sehr viele Einträge für Markierungen (Cursors, Flags, Maßketten, Signalbemaßungen und Schattierungen), die alle in der ein oder anderen Form in den Ansichten visualisiert werden.

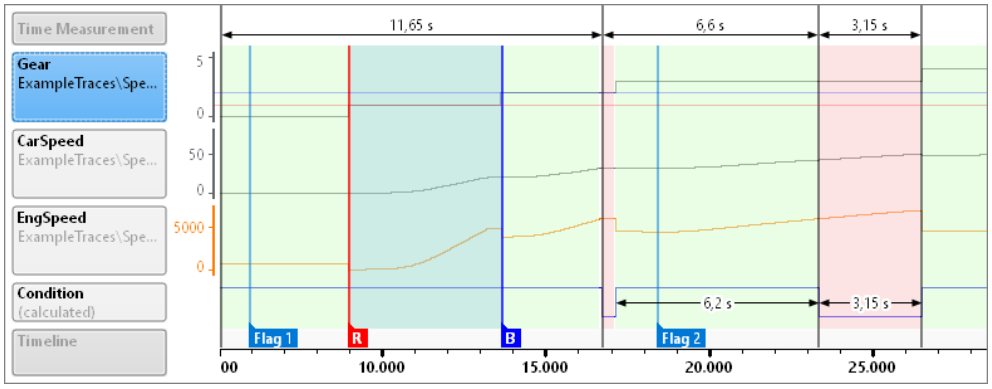


Abbildung 27: Diagrammansicht mit allen eingeblendeten Markierungen

Um die Übersicht zu behalten, bietet die Flagliste in ihrer Symbolleiste zwei neue Schaltflächen zum schnellen Aus- und Einblenden von Markierungen. Wird auf eine der Schaltflächen geklickt, werden **alle Markierungen** entweder aus- oder eingeblendet, unabhängig von ihrem Typ oder ihrer aktuellen Sichtbarkeit.

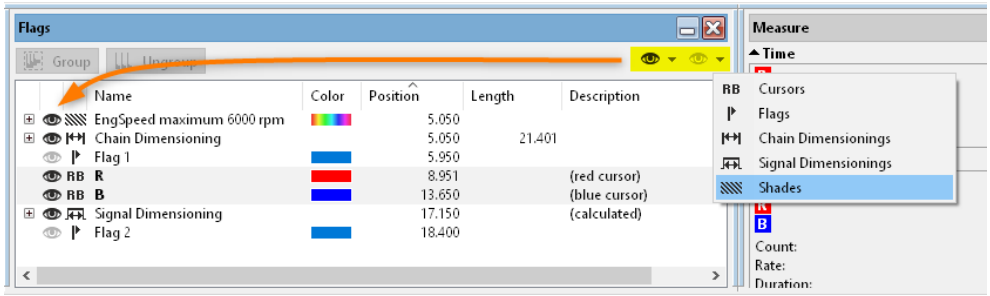


Abbildung 28: Flagliste mit Schaltflächen zum Aus- und Einblenden von Markierungen

Jede Schaltfläche besitzt ein zusätzliches Kontextmenü. Darin befinden sich Einträge für alle verfügbaren Markierungstypen, die jeweils nur die Sichtbarkeit aller **Einträge des ausgewählten Typs** ändern.
Damit lassen sich bestimmte Markierungen auf einen Klick ein- oder ausblenden, ohne diese Einträge vorher einzeln auswählen zu müssen.

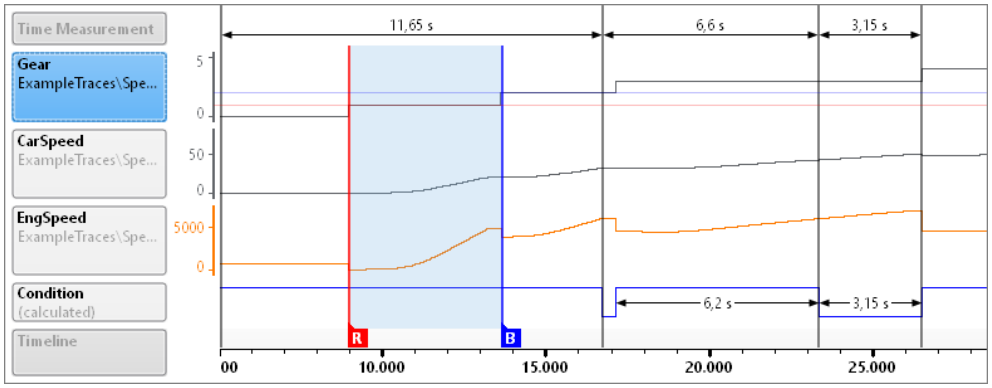


Abbildung 29: Diagrammansicht mit ausgeblendeten Flags und Schattierungen

5 Versionen und Schnittstellen

5.1 Neue Tools und Versionen

et

	Provider	Webseite	System	Produktname	Version
1	**dSPACE**	Release-Link	Sensorgestützte realistische Simulation für ADAS/AD in Echtzeit	**AURELION**	**24.2 bis 25.2**
2	**Vector**	Release-Link	Entwicklungs-, Test und Analysewerkzeuge für automotive SiL und HiL-Projekte	**CANoe/ CANalyzer**	**19SP3**
3	**Vector**	Release-Link	Mess-, Kalibrier-, Diagnose- und Flash-Software	**CANape**	**23**

5.2 APIs

5.2.1 Internal API

Abfrage von Mappings über die Internal API

et tc

Mit dem neuen Parameter **scope** lassen sich Mappingnamen bzw. Mappingziele für einen bestimmten Bereich anhand folgender Syntax abrufen:

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

So ist es z. B. mit **scope**="global" möglich, auf das Mappingziel eines globalen Mappings zuzugreifen, auch wenn das Mapping selbst im aktuellen Testfall nicht verwendet wird.

5.2.2 Test Management API

Erweiterte hierarchische Importfunktion für die ecu.test Test Management API

et

Viele Testmanagement-System organisieren ihre Artefakte in verschachtelten Ordnern, Paketen oder Modulen, die die logische Struktur des Produkts widerspiegeln.

Um diese Hierarchie sichtbar zu machen, wurden die Importmethoden der **ecu.test** Test Management API um baumorientierte Funktionen erweitert, sodass sowohl beim Import von Paketen als auch beim Import von Projekten dasselbe Ordner- und Paketlayout angezeigt werden kann.

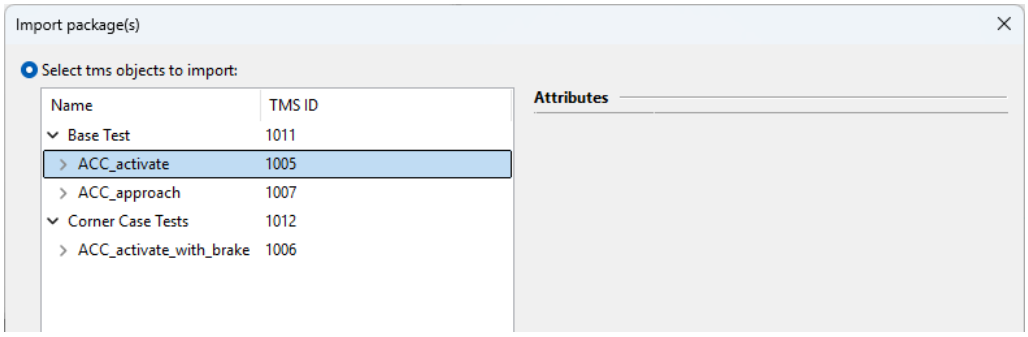


Abbildung 30: Hierarchische Ordner- und Paketstruktur beim Import von Paketen und Paketen

5.2.3 UserTool API

Python-Bibliothek zur IDE-Unterstützung von ecu.test-spezifischem User-Code

et tc

ecu.test lässt sich durch selbst geschriebenen Code flexibel für unterschiedliche Workflows erweitern. Oft müssen dabei bestimmte Interfaces implementiert werden. Ein erster Schritt für eine nachhaltigere Entwicklung dieser Module soll eine bereitgestellte Python-Bibliothek darstellen. Diese lässt sich in die eigene Python-Umgebung installieren und ermöglicht durch die bereitgestellten Interfaces in der eigenen IDE eine Autovervollständigung und Typprüfung.

Sie steht als erster Prototyp im Installationsverzeichnis unter "res" als Python-Wheel zur Verfügung. Aktuell werden die Custom Checks und die nutzerdefinierte Synchronisation von Aufnahmen unterstützt.

6 Abkündigungen

6.1 Abkündigungen und Inkompatibilitäten in dieser Version

Fibex-Unterstützung

et

Die Fibex-Unterstützung für Bus ist abgekündigt und mit **ecu.test 2025.4** entfernt. Die Fibex-Unterstützung für DLT bleibt weiterhin erhalten.

ReqIf Import abgekündigt

et

Die Funktionalität zum Import von **ReqIf**-Daten als Package- und Projekt-Attribut ist mit **ecu.test 2025.4** entfernt.

Job SetFlowControl

et

Der Job SetFlowControl für CAN(-FD) und Flexray Ports wurde vollständig entfernt.
Als Alternative können die folgenden Jobs verwendet werden:

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

6.2 Abkündigungen in zukünftigen Versionen

KS: Tornado nur noch über ASAM ACI

et

Die Toolanbindung wird voraussichtlich mit **ecu.test 2026.1** entfernt. Sie wird durch die neue Anbindung auf Basis von ASAM ACI ersetzt, die seit **ecu.test 2023.3** verfügbar ist.

Abkündigung der integrierten Test Management Anbindung für Jama

et

Die fest in **ecu.test** integrierte Anbindung an Jama wird mit **ecu.test 2026.2** entfernt. Als Ersatz kann die neue Python-basiere Anbindung genutzt werden.

Jobs RequestSeed und SendKey

et

Die Jobs **RequestSeed** und **SendKey** werden ersetzt.

- RequestSeed → SecurityAccessRequestSeed
- SendKey → SecurityAccessSendKey

Die neuen Jobs unterstützen auch die Seed & Key DLLs.

Alternatives Reportverzeichnis bei separater Unterprojektausführung

et tc

Bei separater Unterprojektausführung besteht aktuell die Möglichkeit, den Reportordner angeben zu können. Dieses Feature ist zur 2025.3 als deprecated markiert und wird zur **ecu.test** Version 2026.1 entfernt.

Abkündigung FEP2 Anbindung

et

Mit **ecu.test 2026.1** wird die FEP2 Anbindung entfernt

Playbook-Export von ecu.test nach test.guide

et

Mit **ecu.test 2026.1** wird der Playbook-Export von **ecu.test** nach **test.guide** entfernt.

Port BUSACCESS - GENERIC_MAPPINGFILE

et

Der neu eingeführte Porttyp BUSACCESS - MODEL BASED ist weitaus flexibler, wartbarer und intuitiver in der Konfiguration.
Um die Übersichtlichkeit bei den verfügbaren Optionen zu erhöhen und die Nutzer zur bestmöglichen Lösung zu führen, entfernen wir den BUSACCESS - GENERIC_MAPPINGFILE zu **ecu.test 2026.1**.

Unterstützung Ubuntu 20.04 LTS und 22.04 LTS

et

Mit **ecu.test 2026.1** wird die Unterstützung für Ubuntu 20.04 LTS und 22.04 LTS abgekündigt.

Unterstützte Versionen von MATLAB/Simulink in Linux

et

Im Zuge der Abkündigung der Unterstützung älterer Versionen von Ubuntu mit **ecu.test 2026.1** wird unter Linux auch der Support von MATLAB/Simulink bis inkl. R2023b entfernt, da diese keinen Support für Ubuntu 20.04 bieten. Der Support unter Windows ab R2015b bleibt unverändert.

Für die Tools Ethernet, XL-API und SIL-Kit wir der ICMP-RAW Port abgekündigt

et

Mit **ecu.test 2026.1** wird der Port entfernt.