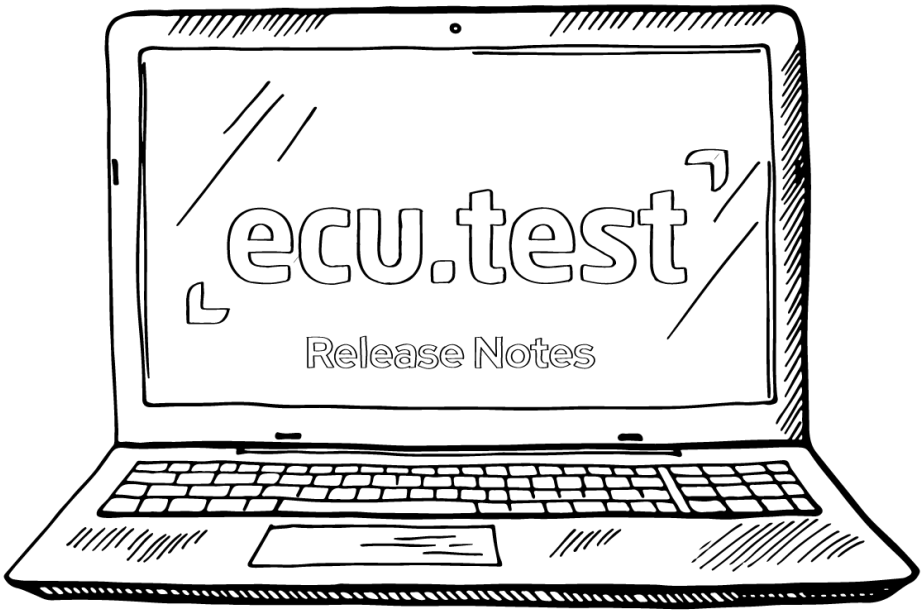
tracetronic



Release Notes

ecu.test 2025.2

trace.check 2025.2

Datum: 27.05.2025

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

tracetronic

Inhalt

tracetronic

Überblick

Mit dem **ecu.test** Release 2025.2 launchen wir eine neue Produkterweiterung: **ecu.test** *lab*. Das Add-on lässt sich – plattformunabhängig – zur Steuerung und Visualisierung von Prüfständen via Weboberfläche nutzen, ohne tiefes Know-how oder komplexe Testumgebungen zu benötigen. Es zielt vor allem auf frühe Entwicklungsphasen und ein exploratives Testverhalten in SiL-Umgebungen ab, das sich live nachvollziehen lässt. Lest mehr dazu im Anwenderhandbuch.

Weitere Highlights des Releases:

- **Anwenderhilfen im Web**
  Die Produktdokumentationen stehen nun auch online zur Verfügung
- **Szenarien- und Testfallworkflows**
  Engere Verknüpfung mit CarMaker-Umgebungen für ganzheitliche Szenarien- und Testfallentwicklung sowie Analyse
- **OBDonUDS**
  Vollständige Unterstützung des OBDonUDS-Standards
- **Open with ecu.test diff**
  Browsererweiterung, um Änderungen aus GitHub oder GitLab direkt in **ecu.test** mit **ecu.test** Diff zu vergleichen
- **Bibliotheksworkspaces**
  Unterstützung für TBC- und TCF-Konfigurationsdateien
- **Test Case Coverage**
  Neue Funktion zur Erstellung von Coverages
- **SOME/IP-Erweiterung**
  Auswahl spezifischer Consumers in der Service-Kommunikation

Weiterlesen lohnt sich!
Denn neben unseren Highlights enthält dieses Release eine Vielzahl weiterer Features und Verbesserungen, die sich quer durch alle zentralen Themenfelder ziehen: von neuen **Diagnose**- und **Kommunikation**smöglichkeiten über **API**-Erweiterungen bis hin zu smarten Neuerungen in **Multimedia**, **SiL**- und **HiL**-Setups sowie der **Traceanalyse**.
Mit zahlreiche kleinen, aber wirkungsvollen Usability-Optimierungen sorgen wir dafür, dass ihr noch effizienter testen könnt und ecu.test euch im Alltag von maximalem Nutzen ist.

Und nicht zuletzt: Freut euch auch auf unsere Sneak Preview!

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

1 Highlights in ecu.test 2025.2

ecu.test lab erhöht die Zugänglichkeit komplexer Prüfplätze

ecu.test ***lab*** ist eine von **ecu.test** bereitgestellte Weboberfläche zur Visualisierung und Steuerung des Verhaltens eines Prüfplatzes sowie des Prüflings.

Egal ob Windows oder Linux, lokal oder remote, CAN, XCP, Modell oder Multimedia – **ecu.test** *lab* läuft dort, wo es gebraucht wird und kommuniziert über alle in **ecu.test** verfügbaren Schnittstellen direkt auf Basis bestehender Mappings oder Packages.

Jetzt ausprobieren!

Der volle Funktionsumfang von **ecu.test** *lab* ist mit jeder **ecu.test**-Lizenz bis Ende 2025 verfügbar.

ecu.test *lab* kann aus **ecu.test** heraus konfiguriert und gestartet werden.
Der Zugriff erfolgt dann über einen beliebigen Webbrowser auf demselben Rechner, oder remote über das Netzwerk.

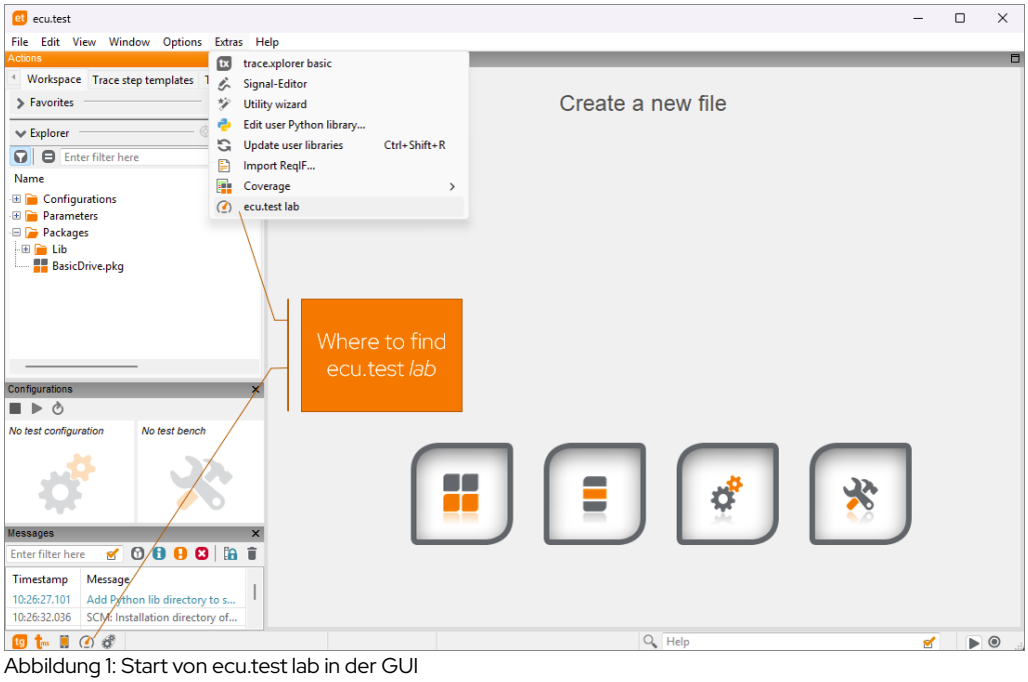


Abbildung 1: Start von ecu.test lab in der GUI

Die grafische Oberfläche kann abhängig von den Bedürfnissen der Nutzenden oder der Nutzergruppe konfiguriert werden.
Dafür stehen verschiedene Widgets zur Verfügung, die Messwerte oder Zustände des Prüfstands anzeigen oder deren Manipulation ermöglichen.

Die Widgets sind immer mit Mappings, Packages und Ausdrücken (Expressions) in **ecu.test** verknüpft. Für den Wertezugriff greift **ecu.test** direkt auf die verfügbaren Toolschnittstellen zu, führt das verknüpfte Package aus oder evaluiert den angegebenen Ausdruck.

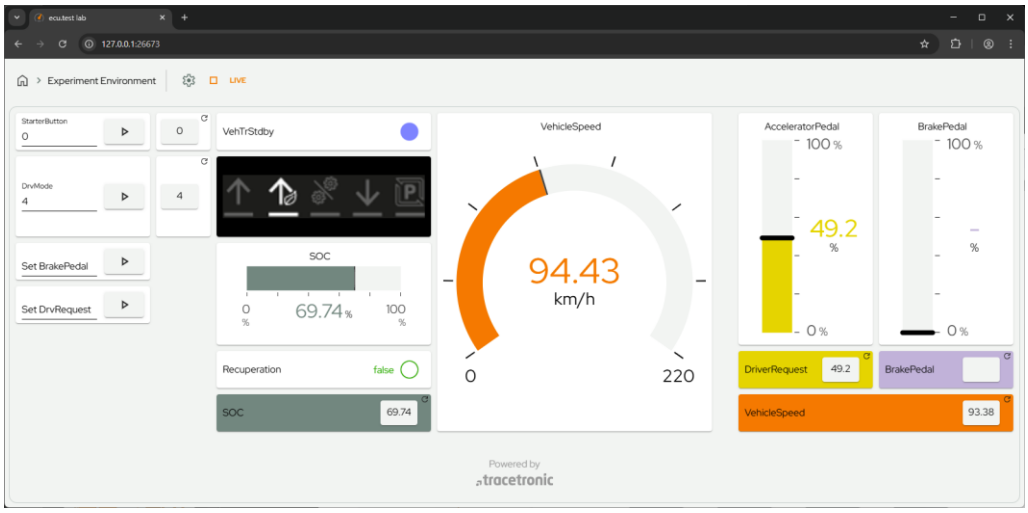


Abbildung 2: Benutzerdefinierte View von ecu.test lab

Hinweis: Mehr Details zu **ecu.test** *lab* gibt es im **Anwenderhandbuch**.

Online-Version der Anwenderhandbücher

et tc

Die **ecu.test**- und **trace.check**-Anwenderhandbücher sind ab sofort auch online abrufbar. Beginnend mit Version 2025.2, werden alle Handbuchbestandteile der letzten vier Releases zur Verfügung gestellt.

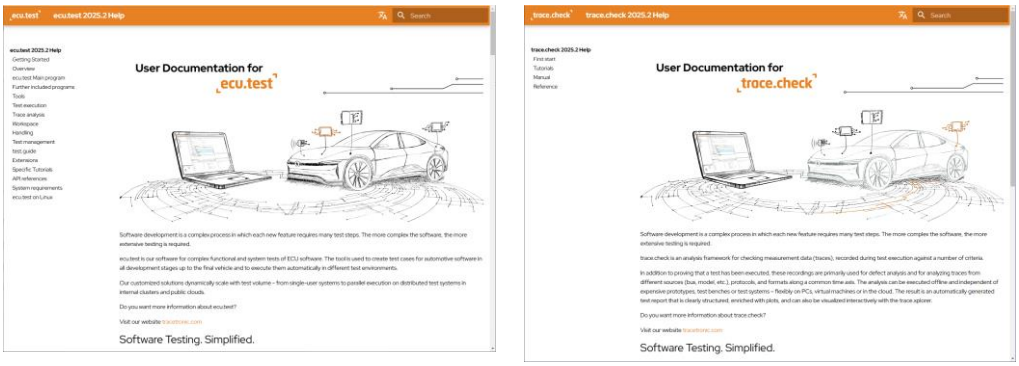


Abbildung 3: Online-Anwenderhilfen ecu.test und trace.check

Zukünftig werden die Online-Handbücher auch direkt mit **ecu.test/ trace.check** verknüpft, sodass immer die aktuellste Version genutzt wird. Auch kann ein schneller Blick in die Handbücher nun auch ohne vorherige Installation erfolgen.

Zusätzlich wurde die Ablagestruktur der Offline-Handbücher modernisiert. Dabei wurden u. a. Dateinamen vereinheitlicht (nun durchgängig englisch) und kleinere strukturelle Anpassungen vorgenommen.

Hinweis: Verlinkungen innerhalb der Hilfe haben sich dadurch geändert!

Szenarien- und Testfallworkflow mit ecu.test und scenario.architect in CarMaker-Umgebungen

et

Eine der größten Herausforderungen beim szenarienbasierten Testen ist das Zusammenspiel von Testfällen und Szenarien. Mit **ecu.test 2025.2** bieten wir für CarMaker einen durchgängigen Workflow, um sowohl Szenarien als auch Testfälle ganzheitlich erstellen, anpassen und in diversen Testumgebungen ausführen zu können.

Die nachfolgende Grafik verdeutlicht die Interaktion zwischen **ecu.test** und dem **scenario.architect**.

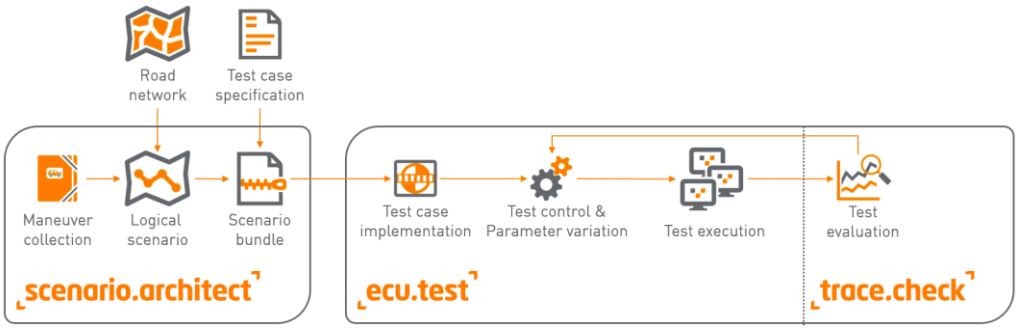


Abbildung 4: Schematische Darstellung der Interaktion zwischen ecu.test und scenario.architect

In **ecu.test** können neue Szenarien angelegt und bestehende angepasst werden. Dafür bietet der Szenarienbaum im Tab **Umfeldsimulation** neue Kontextmenüeinträge.

Beim Erstellen oder Ändern öffnet sich der **scenario.architect**. In diesem Tool können Anpassungen vorgenommen oder Szenarien komplett neu entworfen werden. Nach der Fertigstellung übernimmt **ecu.test** das Artefakthandling und sorgt dafür, dass die notwendigen Szenarien direkt zur Umfeldsimulation weitergeleitet werden. Über die Testfälle werden die entsprechenden Szenarien geladen und ausgeführt.

Der Mehrwert entsteht durch die Kopplung der Toolings. Durch die Integration können Szenarien, Testfälle und Analysen sehr schnell, intuitiv und iterativ entwickeltet und ausgeführt werden.

Abbildung 5: Interaktion zwischen ecu.test und scenario.architect

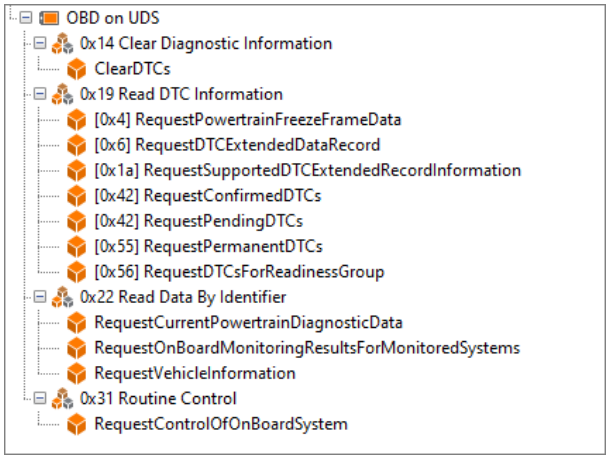
Weitere Details sind im Tutorial des Anwenderhandbuchs zu finden.

Hinweis: Der Workflow ist für die Verwendung von OpenSCENARIO-Dateien konzipiert. Die Funktionen stehen aktuell für die Umfeldsimulationen VTD und jetzt auch CarMaker zur Verfügung.

Vollständige Unterstützung von OBDonUDS

Mit **ecu.test** *diagnostics* kommt eine vollständige Unterstützung des OBDonUDS-Standards. Highlight sind die symbolischen Diagnoseservice-Test-schritte, die einfach per Drag and Drop in den Testfall gezogen werden können.

Dazu muss nur die OBDonUDS-Option in den **Diagnose**-Einstellungen der Testkonfiguration (TCF) aktiviert werden. Die Testschritte erscheinen im Tab **Diagnose** nachdem die Konfiguration gestartet wurde – nativ und ohne, dass eine Diagnosedatenbank benötigt wird.



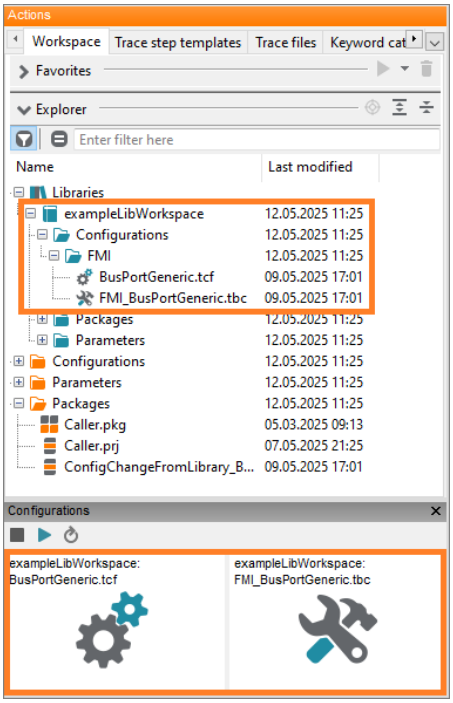
Die Services sind für alle Diagnose-Ports verfügbar.

- IsoTp über CAN
- FrTp über FlexRay
- DoIP über Ethernet

Abbildung 6: Vollständige Unterstützung des OBDonUDS-Standards

Bibliotheksworkspaces: Unterstützung von Konfigurationsdateien (TBC/TCF)

et tc



Testbenchkonfigurations- und Testkonfigurationsdateien (TBC- und TCF-Dateien) lassen sich nun zentral verwalten und werden im **Workspace**-Explorer innerhalb des Konfigurationsverzeichnisses angezeigt.

Sie können per Doppelklick im Editor geöffnet und bearbeitet werden, oder auch via Drag and Drop in das Konfigurationen-Fenster gezogen werden.

Abbildung 7: TBC- und TCF-Dateien im Workspace-Explorer und Konfigurationenfenster

Im **Konfigurationswechsel**-Projektschritt wurde die Auswahl von TBC- und TCF-Dateien erweitert: Im Dropdown-Menü der jeweiligen Datei sind neben lokalen nun auch Bibliotheksdateien wählbar, was die Konfiguration deutlich flexibler macht. Beim Laden der Konfiguration werden die Bibliotheksreferenzen automatisch aufgelöst.

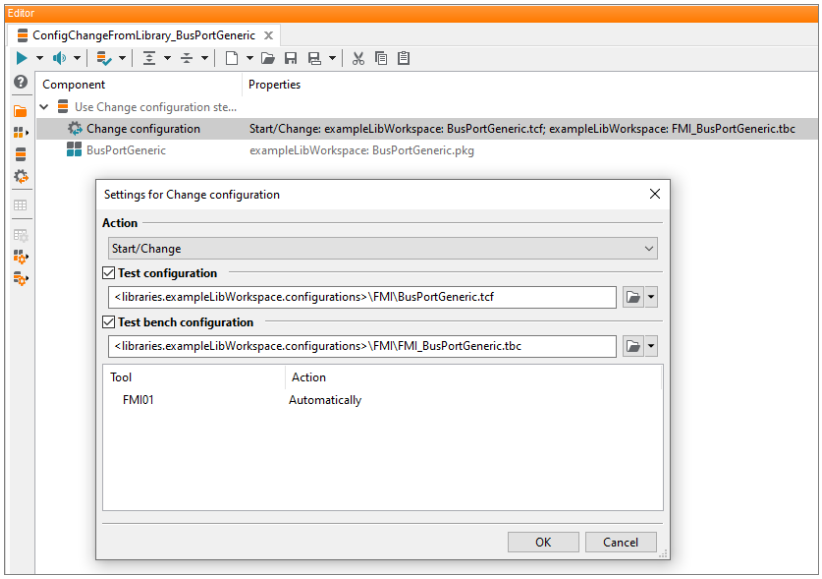


Abbildung 8: Auswahl von TBC- und TCF-Dateien im Konfigurationswechselprojektschritt

Für eine bessere Unterscheidung zwischen lokalen und Bibliotheks-Konfigurationsdateien wurden die Icons nach gewohntem Schema angepasst und werden nun orange bzw. petrol dargestellt.

Hinweis: Die Object-API bietet neue Methoden zur Nutzung im Konfigurationswechsel-Projektschritt. Das bestehende Tutorial wurde dahingehend erweitert.

Erstellung von Test Case Coverage-Übersichten

et

Seit **ecu.test** 2024.3 können Coverage-Metriken für Testfälle erzeugt werden, um die Nutzung und Absicherung von Packages im Rahmen des Qualitätsmanagements systematisch zu analysieren.

Diese Funktion unterstützt insbesondere bei der Identifikation ungenutzter Abschnitte und bei der Absicherung sicherheitskritischer Bibliotheken – vergleichbar mit Code-Coverage-Analysen in der Softwareentwicklung.

Mit dem aktuellen Release wird diese Funktion nun erweitert: Die Coverage-Daten mehrerer Testausführungen lassen sich konsolidieren und Workspace-weit in einem übersichtlichen Bericht darstellen.

Damit entfällt die Notwendigkeit, Coverage-Informationen mühsam durch einzelne Packages in der GUI nachzuvollziehen.
Die kompakten HTML- und JSON-Reports bieten eine zentrale Sicht auf Testabdeckung, Testlücken und Verbesserungspotenzial – als Grundlage für fundierte Testplanung, gezielte Wartung und durchgängige Qualitätssicherung im gesamten Projekt.

Die Erstellung erfolgt über einen neuen Menüeintrag im bereits vorhandenen **Coverage**-Menü.

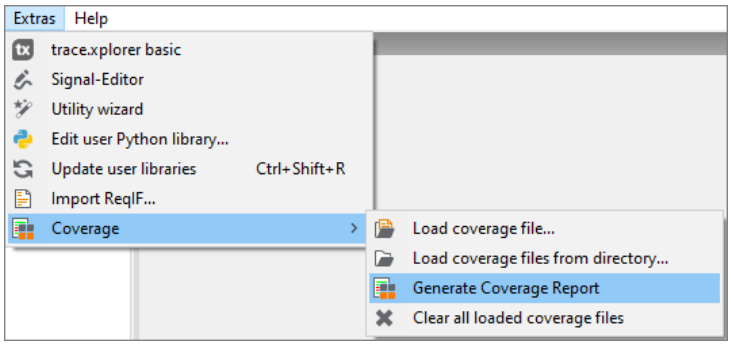


Abbildung 9: Erstellung des Coverage Reports

Daraufhin wird ein HTML-Report erzeugt und direkt in **ecu.test** angezeigt. Die Links zu Packages sind klickbar und öffnen direkt das Package im Tool selbst.

Die HTML-Datei kann auch im Browser angezeigt werden. Für eine automatisierte Auswertung bzw. Verwendung der Daten kann die ebenfalls erzeugte JSON-Datei verwendet werden.

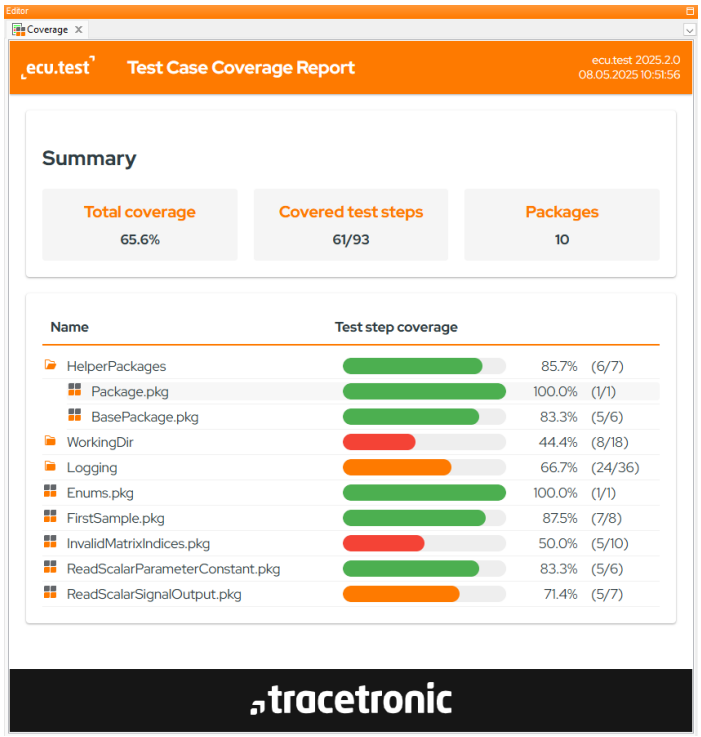


Abbildung 10: Erstellte Coverage-Übersicht

Browser-Erweiterung zum Öffnen von Diffs im ecu.test Diff-Viewer

et tc

Für Chrome, Edge und Firefox steht nun die Browser-Erweiterung **Open with ecu.test diff** zur Verfügung.

Damit können Änderungen an Packages und anderen Artefakten aus GitHub und GitLab heraus, direkt mit einem installierten **ecu.test** verglichen werden.

Hinweis: Der Vergleich kann ab dieser Version auch ohne Lizenz geöffnet werden.

Das Hinzufügen der Erweiterung kann über die Browser erfolgen (Chrome+Edge, Firefox).

Browser-Erweiterungen werden oft über die IT verwaltet und benötigen eine explizite Freigabe. Zur leichteren Prüfung ist der Quellcode der Erweiterung Open Source gestellt (GitHub).

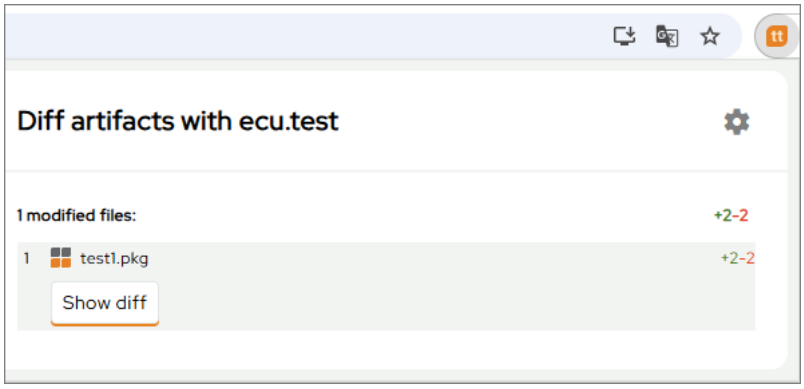


Abbildung 11: Verwendung der Erweiterung im Browser

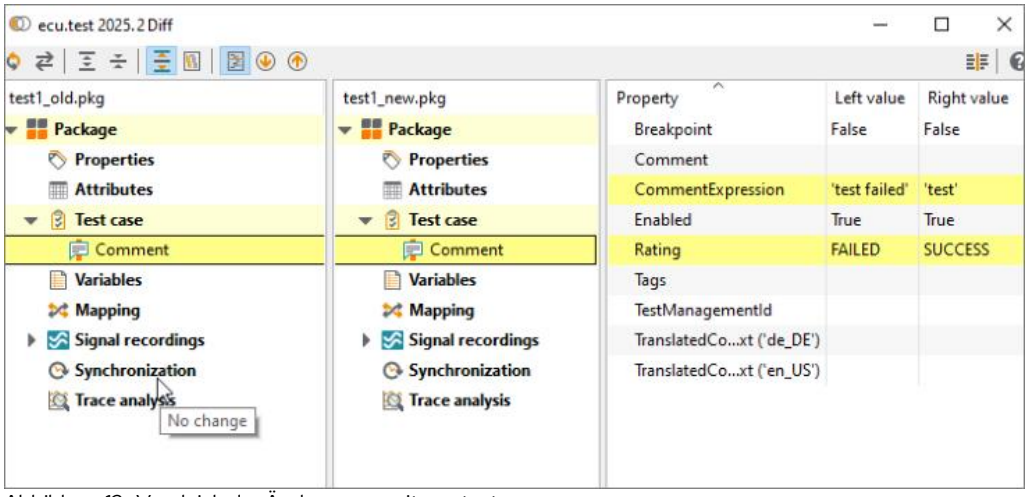


Abbildung 12: Vergleich der Änderungen mit ecu.test

Service communication via SOME/IP: Auswahl des Consumers

et

Für Anwendungsfälle, bei denen es eine Rolle spielt, welcher Service-Consumer einen Service nutzt, besteht ab sofort in den Mapping-Items der Ethernet-Testschritte die Möglichkeit, das konsumierende Steuergerät (Consumer) auszuwählen.

Durch Auswahl eines Consumers wird beim Testfallstart automatisch der Netzwerkadapter mit den TCP/IP-Einstellungen des konsumierenden Steuergeräts aus der ARXML konfiguriert.

Mit diesem Feature entfällt die Notwendigkeit, pro Consumer jeweils einen Port in der Testbenchkonfiguration anzulegen und manuell zu konfigurieren.

Für Bestandstestfälle oder falls die Consumer-Auswahl leer gelassen wird, greift weiterhin die IP-Konfiguration in den TBC-Einstellungen **Fallback IP-Einstellungen**.

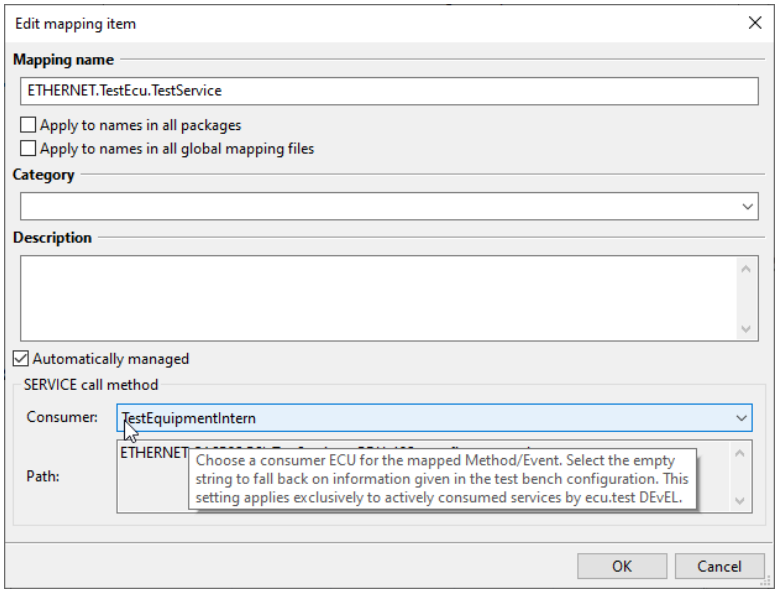


Abbildung 13: Auswahl eines Consumers

Die neue Option wirkt sich nur in Verbindung mit dem aktiven Service-Port und den in **ecu.test** integrierten TCP/IP-Netzwerkstack aus.

Diese Erweiterung wirkt sich zudem auf die Anbindung des ServiceManager-Ports in dSPACE-Modellen aus. Hier kann der Platzhalter **consumerNode** nun auch genutzt werden, um eine Abbildung auf Modellvariablen zu definieren.

2 Usability

Einführung eines technischen Users für den Vault

et tc

Mit Hilfe des Vaults können Zertifikate oder Credentials sicher im Workspace abgelegt und anschließend im Testfall verwendet werden. Mit **ecu.test** 2025.2 wurde der Vault um die Option eines technischen Nutzers erweitert. Dieser kann den Vault nicht einlesen oder editieren, sondern lediglich für die Testfallausführung aufschließen.

Der Vault wird dazu, wie gewohnt, mit einem Passwort angelegt. Unter der neuen Option **Show Tester Key** wird ein generierter Schlüssel aufgeführt, der an den technischen Nutzer übergeben werden kann. Der Vault erkennt automatisch, ob er mit einem Admin-Passwort oder einem Tester Key geöffnet wurde und schränkt den Zugriff automatisch ein.

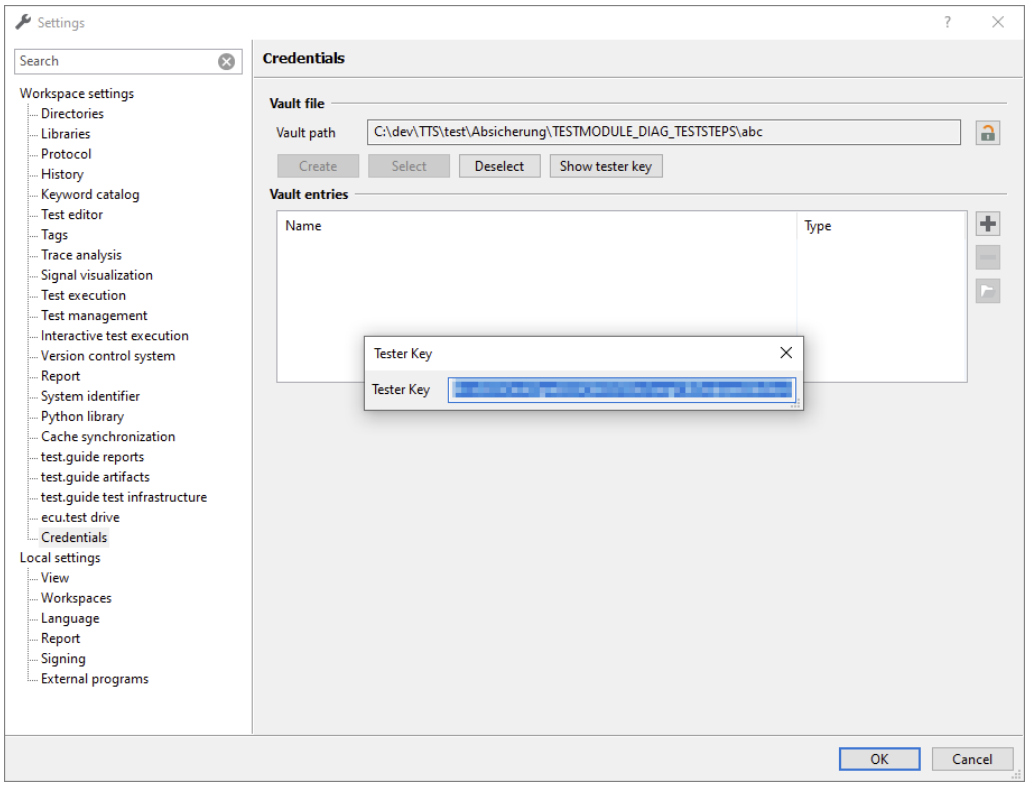


Abbildung 14: Hinterlegen eines Tester-Key-Vaults für den technischen User

Neue File-Cache-Technologie

et tc

Der File-Cache von **ecu.test** dient als lokaler Zwischenspeicher für die im Workspace verwendeten Dateien und Metadaten. Er dient insbesondere dazu, redundante Dateioperationen zu vermeiden und beschleunigt damit den Start und die Ausführung von Testfällen sowie die generelle Verwendung der GUI. Die Verwendung findet im Hintergrund automatisch statt.

Durch eine Aktualisierung der verwendeten Technologie und weiteren Optimierungen, ist die Erstellung des Caches nun bis zu vier Mal schneller als zuvor, was insbesondere in großen Workspaces einen deutlichen Effekt hat.

Copy and Paste in Testbenchkonfigurationen

et

Es ist nun möglich, Tools (inklusive ihrer Ports) und einzelne Ports aus einer TBC in eine andere zu kopieren, indem sie einfach per Copy and Paste (alternativ: Strg-C + Strg-V oder Kontextmenü) überträgt.

Die Object-API wurde analog dazu angepasst und ermöglicht es nun, Ports und Tools wiederzuverwenden:

- Tool.Clone
- Tool.InsertPort
- Port.Clone
- ToolHost.InsertTool

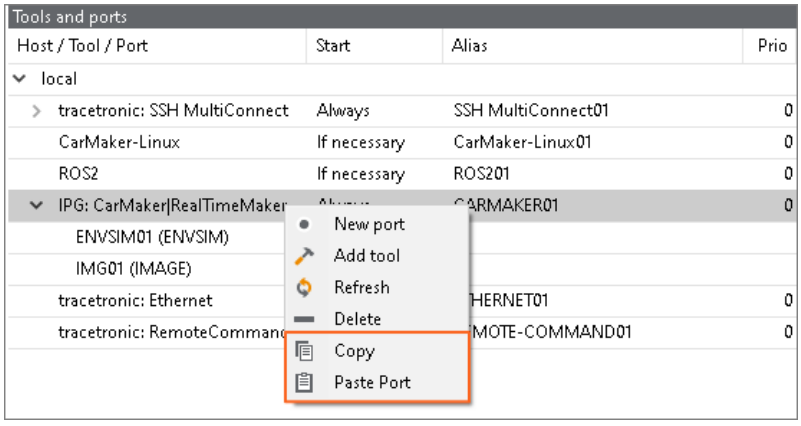


Abbildung 15: Neue Copy-and-Paste-Funktion in der TBC

Neues Attribute-Panel an weiteren Stellen verfügbar

et tc

In vorherigen Versionen wurde das Attribute-Panel für den Package-Editor überarbeitet. Das neue Panel bietet nun eine übersichtliche tabellarische Darstellung, Möglichkeiten Attribute zu filtern und eigene Listen anzulegen.

Mit **ecu.test** 2025.2 wurden auch andere Stellen des Programms umgestellt: Die Bearbeitung von Projektattributen und das allgemeine Bearbeiten von Attributen aus dem Workspace-Explorer heraus bieten nun auch die Vorteile des neuen Attribute-Panels.

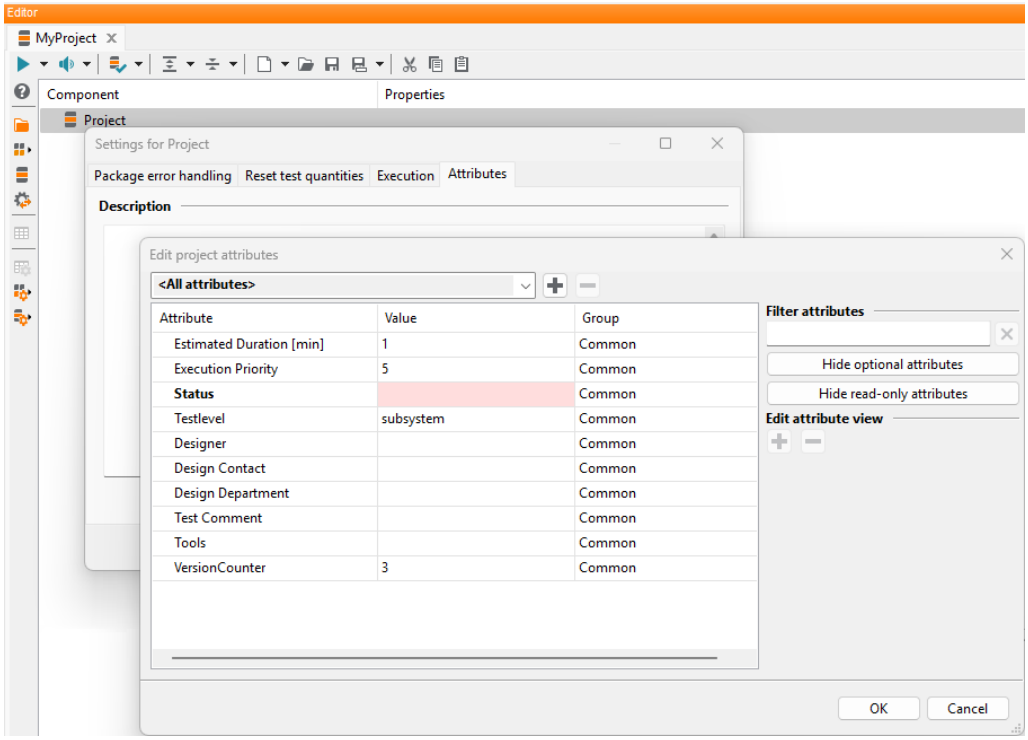
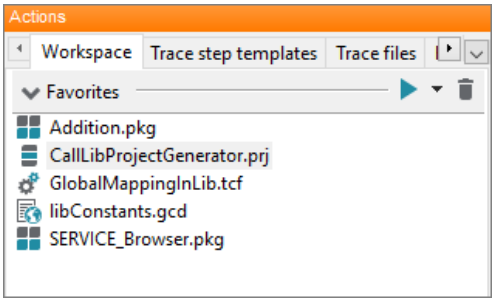


Abbildung 16: Neues Attribute-Panel

Bibliotheksworkspaces: Unterstützung in Favoriten

et tc



Um eine bessere Unterscheidung zwischen lokalen Artefakten und Bibliotheksartefakten zu ermöglichen, werden Bibliotheksanteile nun auch in den Favoriten mit passenden Icons in petrol angezeigt.

Abbildung 17: Favoriten mit Unterstützung von Bibliotheksworkspaces

Verbesserungen für das Dependency Management von Python-Bibliotheken

et tc

Die überarbeitete Verwaltung von externen Python-Bibliotheken wurde weiter verbessert. So werden nun die Abhängigkeiten aus Bibliotheksworkspaces berücksichtigt und gemeinsam mit den in der **requirements.txt** definierten Bibliotheken auf Konflikte geprüft und installiert.

Der Installationsprozess der Python-Bibliotheken wurde insgesamt verbessert, sodass nun zum Beispiel auch Quelltextbibliotheken installiert werden können, wenn auf dem System die für den Bau notwendigen Vorraussetzungen gegeben sind.

3 Testaspekte

3.1 Multimedia

Göpel: Auswahl eines LVDS-Kanals

et

Es ist nun möglich, für Framegrabber von Göpel den zu verwendenden LVDS-Kanal zu konfigurieren. Bisher verwendete **ecu.test** immer automatisch Kanal 0.

Konfiguration kameraspezifischer OpenCV-Einstellungen auf Image-Port

et

Kameraspezifische Einstellungen, wie eine feste Brennweite oder manueller Fokus, können nun als OpenCV-Eigenschaften auf dem **Image**-Port der Kamera konfiguriert werden.

Diese Einstellungen sind identisch zu jenen, die über den Job **SetOpenCVProperty** gesetzt werden können, werden aber vorab und außerhalb des Testfalls angewendet.

3.2 SiL

ASAM: XIL und abgeleitete Anbindungen: Neuer Job zum Pausieren der Simulation

et

Neben den Jobs zum Starten und Stoppen der Simulation, stellen wir nun auch einen Job zum Pausieren der Simulation bereit:

- **PauseSimulation**

Achtung: Kommen diese Jobs bei der Verwendung der Simulationszeit als Testfallzeit (Stepwise bzw. Continuous Modus) zum Einsatz, kann ein Pausieren/Stoppen der Simulation den Ablauf der Testausführung deutlich beeinflussen und ein Fortschreiten auch komplett verhindern.

3.3 HiL

INCA: Experiment ohne Data-Set laden

Insbesondere bei SiL-Anwendungsfällen möchte man für bestimmte Devices, die auch ohne Data-Set funktionieren, auf die Angabe dieses Data-Set verzichten.

In Verbindung mit INCA ab Version 7.5.3 und ab **ecu.test** 2025.2, wird ohne Angabe einer Hex-Datei automatisch ein leeres Data-Set erzeugt und verwendet. Auf die Angabe einer Hex-Datei in der Testkonfiguration kann verzichtet werden.

IPG: RealtimeMaker

Der IPG RealtimeMaker kommt bei der Ansteuerung der IPG XPack HiL's zum Einsatz. Mit **ecu.test** besteht somit die Möglichkeit, die Interfaces wie z. B. die Buskommunikation oder die IO-Ansteuerung zu automatisieren.

Der Funktionsumfang der RealtimeMaker-Anbindung entspricht der CarMaker-Anbindung, mit Ausnahme der Szenariensteuerung.

Hinweis: IPG RealtimeMaker wird über die CarMaker Tool-Anbindung unterstützt.

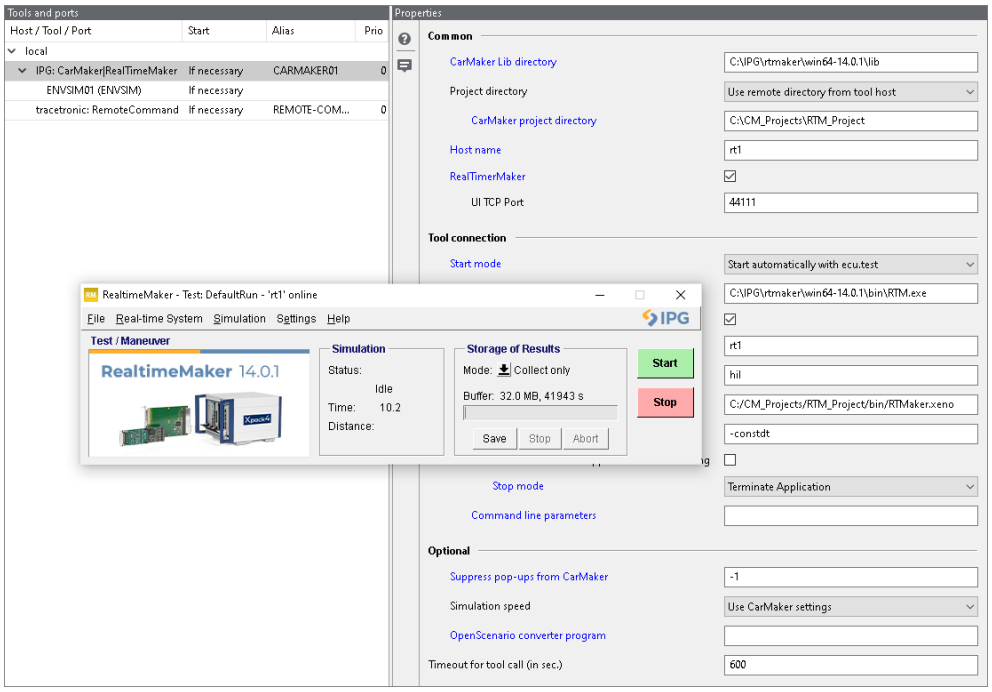


Abbildung 18: Unterstützung des IPG RealtimeMaker über die CarMaker Tool-Anbindung

3.4 Testmanagement

Jama connect: Neue Möglichkeit zur Anbindung an ecu.test

et

Bisher war die Anbindung von Jama connect fest in **ecu.test** integriert und konnte mit Hilfe der ALM-Hooks an spezifische Workflows angepasst werden.

Mit der Einführung der neuen **UserTestmanagement-API** bieten wir nun einen Python-basierten Muster-Workflow für Jama connect, der deutlich mehr Möglichkeiten zur Anpassung bietet. Damit ist eine noch flexiblere und individuellere Anbindung von Jama an **ecu.test** möglich.

3.5 ecu.test *calibration*

Aufnahme vieler Messgrößen

et

Mit dem neuen zweistufigen Aufnahmemodus ist jetzt die Aufnahme einer großen Anzahl von Messgrößen möglich.

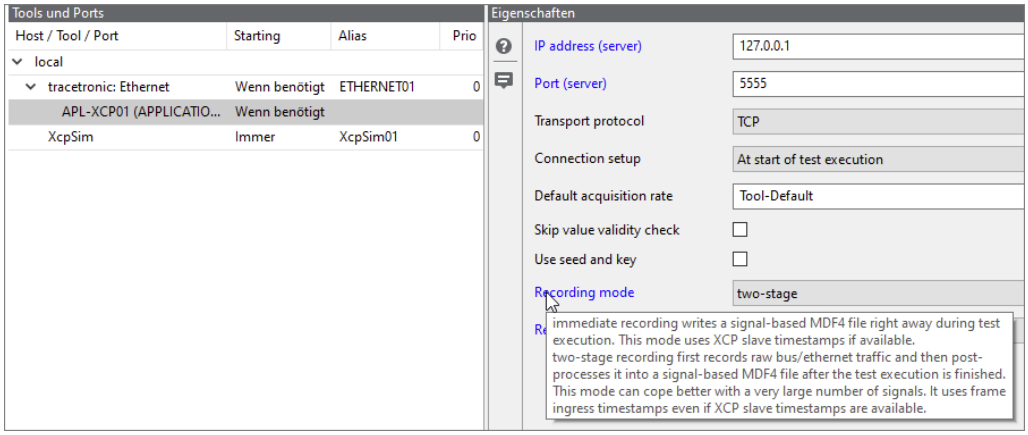
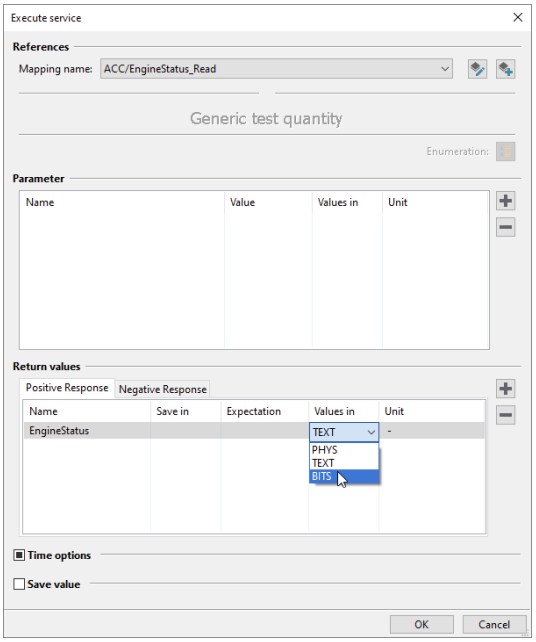


Abbildung 19: Neue Eigenschaft für XCP: zweistufiger Aufnahmemodus

3.6 ecu.test *diagnostics*

Bitcode-Repräsentation in generischen Diagnosetestschritten



Die Rückgabewerte der generischen Diagnoseschritte sind um die Bitcode-Repräsentation (BITS) erweitert worden.

Diese entspricht dem Bitstrom der Daten wie sie auf der untersten Ebene des Übertragungsmediums gesendet werden. Daneben werden weiterhin die umgerechneten Werte (PHYS und TEXT) angeboten.

Über das Auswahlmenü kann zwischen den Repräsentationen gewechselt werden.

Abbildung 20: Wechsel zwischen den Repräsentationen über das Auswahlmenü

Im Report wird die Repräsentation ebenfalls in einer übersichtlichen Tabelle aufgeführt.

Abbildung 21: Darstellung der Repräsentation im Report

DoSoAd für Vector: XL API

Das Transportprotokoll **DoSoAd** ist nun auch für das Tool **Vector: XL API** verfügbar.

Dazu wird in der TBC der neuen Port **DoSoAd** gewählt, der sich innerhalb des Porttyps DIAGNOSTICS befindet.

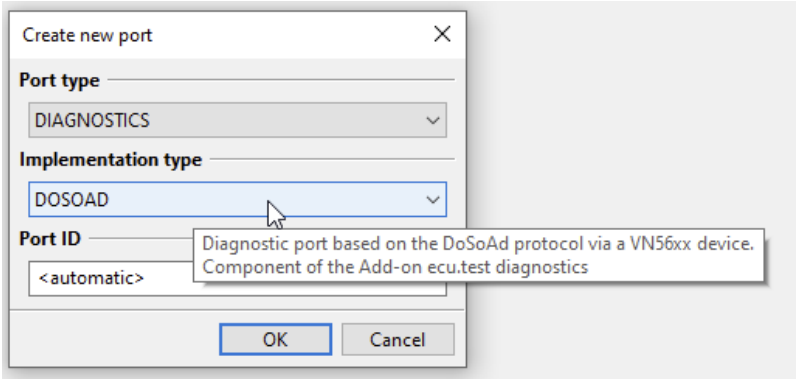


Abbildung 22: Konfiguration des DoSoAd-Ports

Der Port bietet dir alle bereits bekannten Jobs an. Natürlich kannst du auch alle symbolischen Testschritte aus deiner Diagnosedatenbank mit dem Port verwenden.

UDS über CAN in der Traceanalyse

UDS-Botschaften aus einer geladenen ODX oder PDX können nun in der Traceanalyse ausgelesen werden. Dabei werden die in der TCF hinterlegten Parameter zur Kommunikation berücksichtigt.

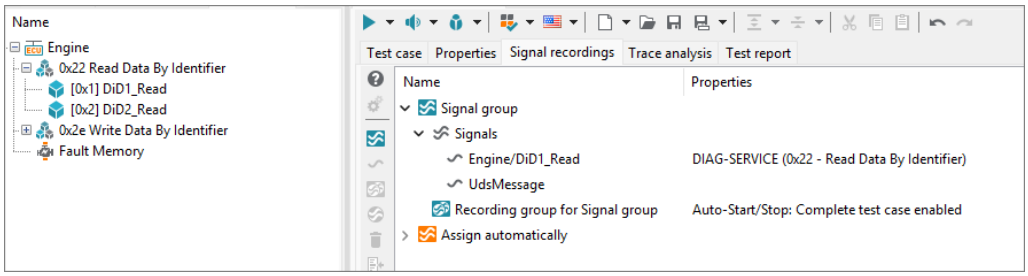


Abbildung 23: UDS-Botschaften in der Traceanalyse

Außerdem ist es möglich, über das Pseudosignal **UdsMessage** auch ohne Datenbank auf das Protokoll zuzugreifen. Auch hier werden die in der TCP hinterlegten Parameter zur Kommunikation benötigt.

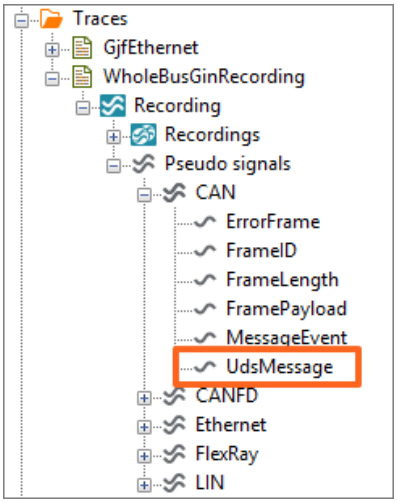


Abbildung 24: Pseudosignal *UdsMessage*

Wird das Signal einer Signalgruppe bzw. dem Mapping hinzugefügt, ist der "Systembezeichner" für das erzeugte Mappingziel standardmäßig leer.

Dadurch werden alle in der TCF hinterlegten ECUs berücksichtigt. Wird nur die UDS-Kommunikation einer einzelnen ECU benötigt, kann der Systembezeichner passend gesetzt werden.

3.7 Kommunikation

Logger-Anbindungen: Lesen und Analyse von Strom- und Spannungswerten

et

Loggerprotokolle wie ASAM CMP, PLP und TECMP unterstützen neben dem Zugriff auf die Buskommunikation auch das Messen von Strom- und Spannungswerten an definierten Messpunkten. Diese Werte können über das jeweilige Loggerprotokoll per Ethernet übertragen werden.

ecu.test bietet ab sofort einen Modell-Port zum Lesen der Messwerte im Testfall und zur Aufzeichnung an. Die Analyse der Messwerte aus PCAP-Aufzeichnungen in der Traceanalyse wird ebenfalls unterstützt.

Um die Konfiguration zu vereinfachen, können mit einer YAML-basierten Konfiguration auf einem einzigen Port mehrere Strecken gemessen werden.

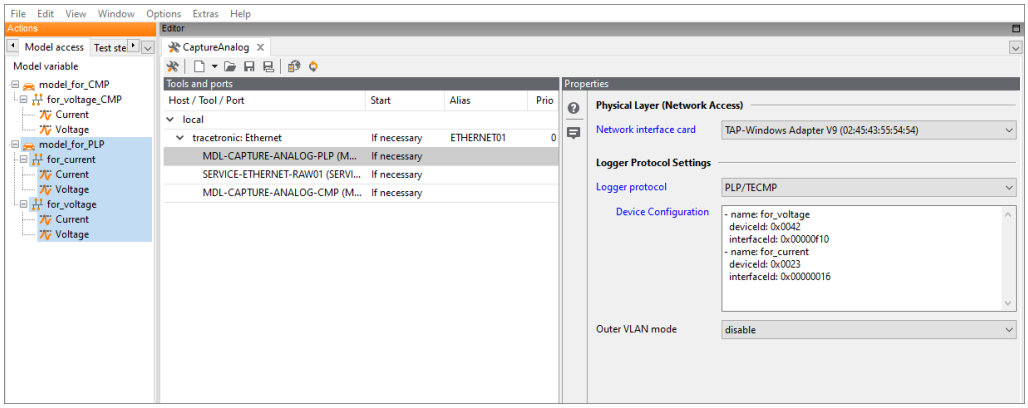


Abbildung 25: Konfiguration des Loggerprotokolls

SOME/IP Simulation: Aktiv lassen

Die Option **Simulierte Services aktiv lassen** auf SOME/IP-Service-Ports lässt nicht nur die skriptbasierte Simulation aktiv, sondern ab sofort auch das zyklische Senden, das mittels Testschritt **Send event** aktiviert wurde.

Damit bleibt eine zuvor aktivierte Simulation auch zwischen der Ausführung mehrerer Testfälle aktiv.

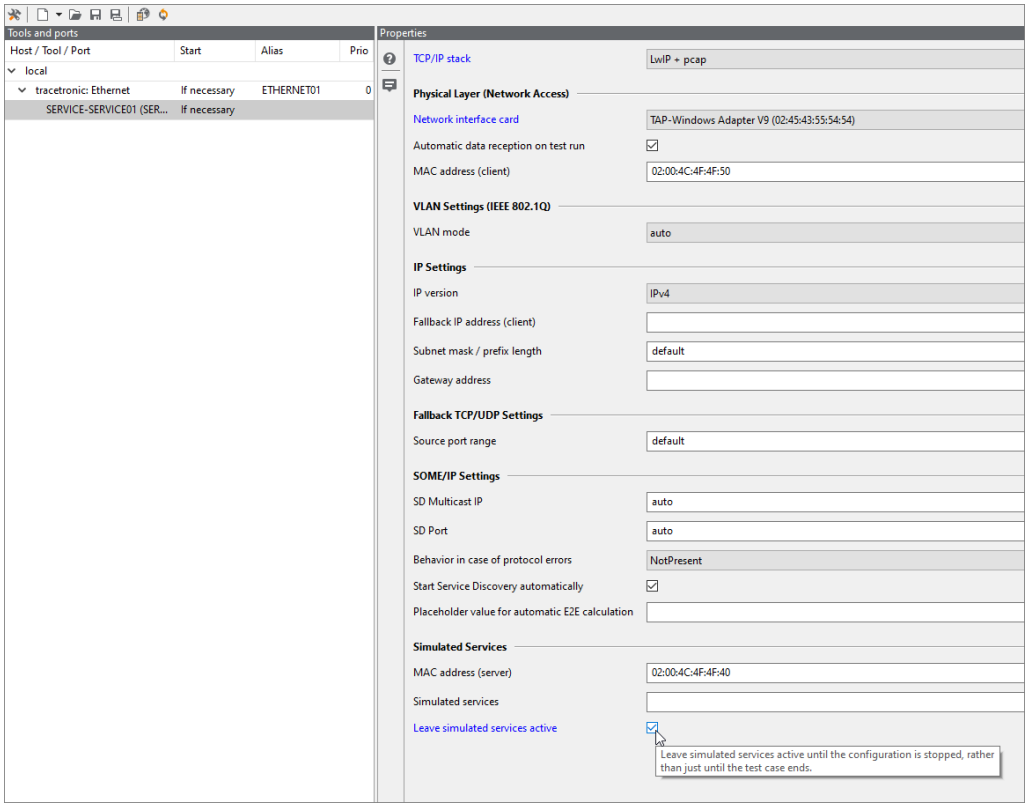


Abbildung 26: Neue Option auf SOME/IP-Service-Ports: Simulierte Services aktiv lassen

J1939: Bus-Lesen und Traceanalyse für lange Botschaften

et

Das Lesen von Bus-Signalen oder Signalgruppen aus Botschaften, die länger als 8 Byte sind und die über das J1939-Transportprotokoll versendet werden, ist ab sofort möglich.

Dafür muss in der Testkonfiguration der hardwarenahen Bus-Ports lediglich der **J1939-Modus** für **Extended CAN-IDs** aktiviert werden.
Die Verwendung der Traceanalyse von ASC-Dateien wird ebenfalls unterstützt.

Hinweis: Die vormals existierende Option **Ignore upper 3 bits (J1939 priority)** wird automatisch in den neuen Modus migriert.

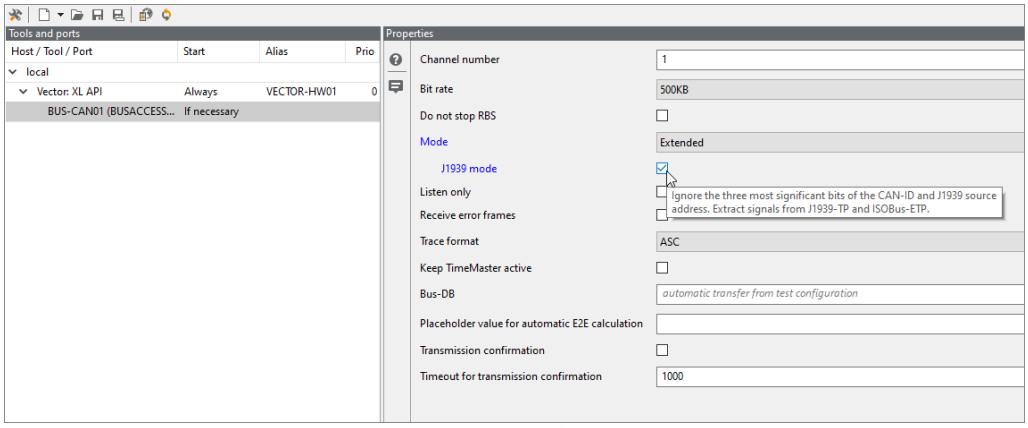


Abbildung 27: Aktivierung des J1939-Modus in der Testkonfiguration der hardwarenahen Bus-Ports

3.8 Traceanalyse

Erweiterte Unterstützung von GIN-Logger Aufzeichnungen

et tc

Das GIN-Aufnahmeverzeichnis wird nun als eine Aufnahme bereitgestellt. Außerdem werden nun GJF-Dateien von Ethernet-Loggern unterstützt. Somit kann nun wie bei anderen botschafts-basierten Formaten eine Vielzahl von Protokollen interpretierten Signalen (bei geladener Datenbank) analysiert werden.

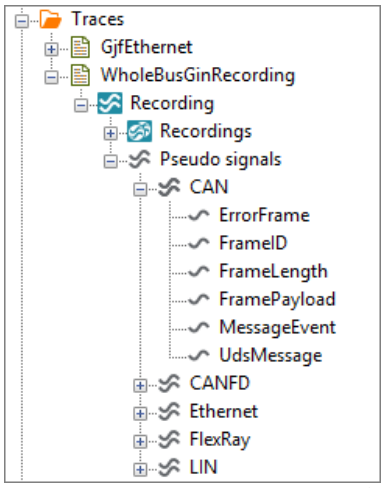


Abbildung 28: Darstellung der GIN-Aufnahmen mit verfügbaren Pseudosignalen für den Zugriff auf Protokollebene.

Performance-Optimierung für signalbasierte Trace-Zusammenführung

et tc

Die Verarbeitung von Signalen aus Busaufzeichnungen wie z. B. ASC wurde optimiert, sodass die Trace-Zusammenführung deutlich beschleunigt wurde.

ROS2: Unterstützung von Strukturinformationen in rosbag-Aufzeichnungen

et tc

Die ROS2-Traceanalyse wurde erweitert, um das neue rosbag-Aufzeichnungsformat zu unterstützen, bei dem Strukturinformationen (IDLs) in der SQLite-Datenbank gespeichert werden.

Durch diese Änderung sind keine zusätzlichen MSG-Dateien, in denen bisher die Strukturinformationen gespeichert wurden, erforderlich, wenn die Strukturinformationen bereits in der SQLite-Datenbank vorhanden sind.

4 Versionen und Schnittstellen

4.1 Neue Tools und Versionen et

	Provider	Web-seite	System	Produkt-name	Version
1	**National Instruments**	Release-Link	Software für Embedded-Softwaretests für Hardware-in-the-Loop-Anwendungen	**VeriStand**	**2025**
2	**Synopsys**	Link	Software-in-the-Loop-Lösung für virtuelle ECUs	**Silver**	**W-2025.03**
3	**Vector**	Release-Link	Programmierschnittstelle für Vector-Hardwareprodukte	**XL Driver Library**	**25.20.14.0**

Unterstützung von Npcap 1.81 für Ethernet-Tests unter Windows

et

Achtung: Bei der Verwendung von **Npcap 1.70 und neuer** in Verbindung mit **802.1q tagged VLAN** kann es zu Einschränkungen bei der Aufzeichnung von selbst versendeten Paketen kommen (abhängig vom verwendeten Netzwerktreiber)

4.2 APIs

4.2.1 Allgemein

Package Check: Neuer Parameter „recursive"

et tc

Die APIs zum Ausführen von Checks wurden um den Parameter **recursive** erweitert, um die Prüfung von referenzierten Artefakten zu steuern.
Diese Erweiterung ist in den REST- und COM-API verfügbar und ermöglicht eine flexiblere und effizientere Prüfung von Workspaces.

4.2.2 Command Line Interface (CLI)

Angabe des Report-Ordners im execute-Command

et tc

Der execute-Command unterstützt jetzt die Option **`--reportdirectory`**, mit der das Zielverzeichnis für den generierten Report angegeben werden kann.

- **Absolute Pfadangabe**: `--reportdirectory /abs/Path`
  - legt den Report im angegebenen absoluten Verzeichnis ab
- **Relative Pfadangabe**: `--reportdirectory rel/Path`
  - speichert den Report relativ zum aktuellen Arbeitsverzeichnis (analog zum Verhalten von `--workspace`)

Diese Erweiterung ermöglicht eine flexible Weiterverarbeitung der Reports gemäß individuellen Anforderungen in der eigenen CI/CT-Umgebung.

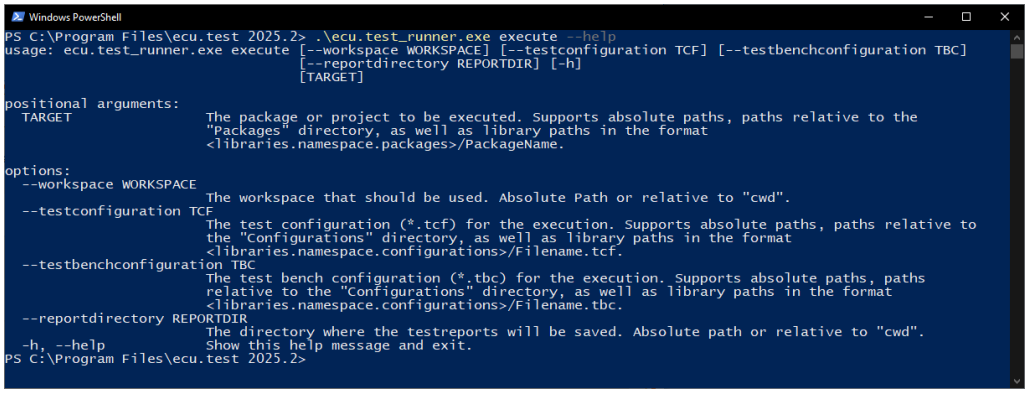


Abbildung 29: Angabe des Report-Ordners im execute-Command

Angabe von Bibliotheksworkspace-Referenzen

et tc

Im Command Line Interface (CLI) wurde die Unterstützung der sogenannten Spitze-Klammer-Notation für den Zugriff auf Bibliotheksworkspace-Artefakte eingeführt:

- `<libraries.libname.packages>/package.pkg`

Diese Notation ermöglicht es, sowohl Packages und Projekte als auch Konfigurationen referenziert aus eingebundenen Bibliotheksworkspaces aufzurufen.

Damit unterstützen Bibliotheksworkspace-Referenzen auch:

- TARGET
- `--testconfiguration`
- `--testbenchconfiguration`

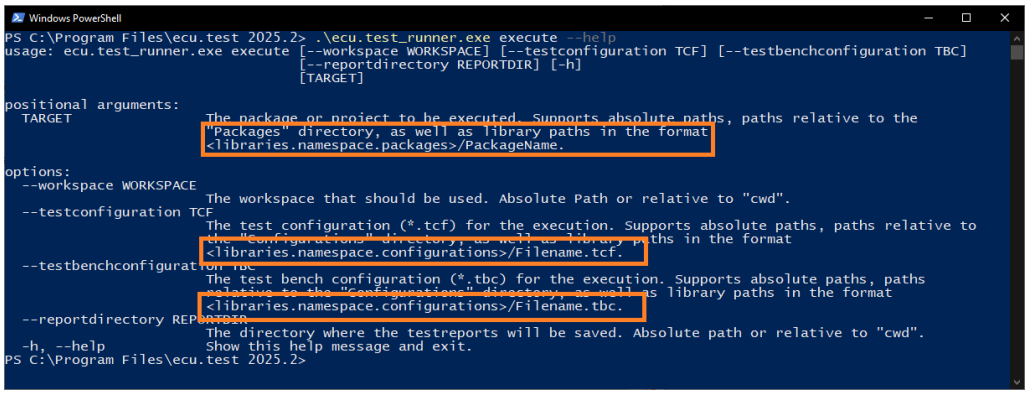


Abbildung 30: Angabe von Bibliotheksworkspace-Referenzen im execute-Command

4.2.3 REST-API

REST-API: Angabe von Bibliotheksworkspace-Referenzen

et tc

Für die REST-API wurde die Unterstützung der sogenannten Spitze-Klammer-Notation für den Zugriff auf Bibliotheksworkspace-Artefakte eingeführt:

- `<libraries.libname.packages>/package.pkg`

Diese ermöglicht es, sowohl Packages und Projekte als auch Konfigurationen referenziert aus eingebundenen Bibliotheksworkspaces aufzurufen.
Konkret betrifft dies die Endpunkte für

- `execution (put)`
- `configuration (put).`

Entsprechende Pfadangaben sind auch in Playbooks in **test.guide** möglich.

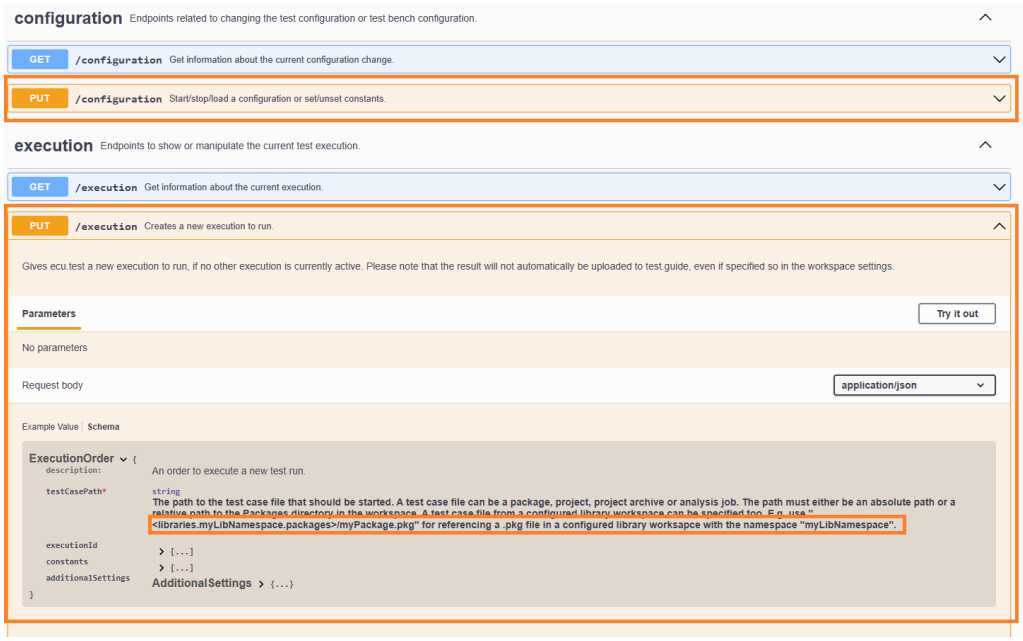


Abbildung 31: Angabe von Bibliotheksworkspace-Referenzen in der REST-API

4.2.4 UserTool

Flexiblere Nutzung von Job-Parametern und Modellauswahl in der TCF

et

- Job-Parameter mit Auswahllisten können nun nicht nur Strings, sondern auch Bool und Int beinhalten.
- Es ist nun möglich, eine Auswahl von Modellen bereitzustellen und das konkret zu startende Modell in der Testkonfiguration auszuwählen.

class UserModelPort ¶

Bases: UserPort

This class describes the interface of a user model port. Classes that fulfill this interface can be used as model port of a user tool.

classmethod GetModels()

Optional (does not have to be implemented)

This method is called to get all available models for test configuration. If it's implemented, a new property "Model" will be added to your port properties which contains a selected model from this list.

Returns

list of available models.

Return type

list[str]

Abbildung 32: Dokumentation UserModelPort

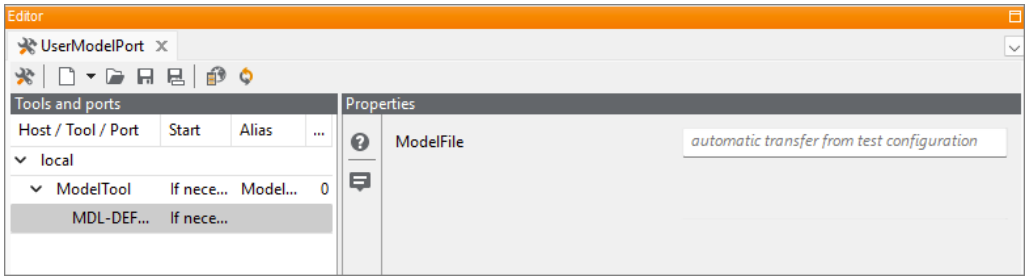


Abbildung 33: TBC-Einstellungen des Modell-Ports

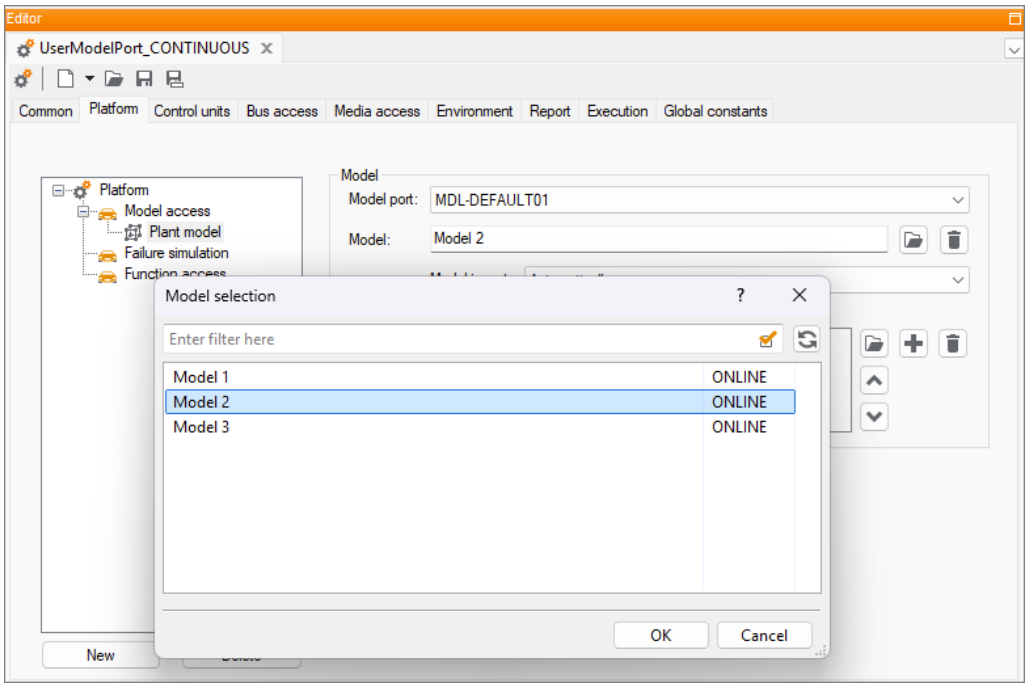


Abbildung 34: TCF des konfigurierten Modells

4.2.5 UserTestmanagement

Neue API für eigene Anbindungen an Testmanagementsysteme

et

Mit der **UserTestmanagement-API** gibt es jetzt die Möglichkeit, eine Anbindung an ein Testmanagementsystem komplett in Python zu implementieren.

4.3 Sneak Preview

AI-basierte Testfallerstellung mit ecu.test *agent*

Der **ecu.test** *agent* ist ein AI-gestützter Assistent, mit dem die Erstellung von Testfällen in **ecu.test** erheblich beschleunigt wird.

Er ist vollständig in **ecu.test** integriert, holt sich die Daten zum aktuellen Kontext direkt aus dem Workspace und generiert auf Knopfdruck passende, ausführbare Testschritte zu definierten Anweisungen im Testfall.

Das zugrunde liegende AI-Modell ist frei wählbar. Wir unterstützen proprietäre Sprachmodelle wie ChatGPT, Gemini und Claude, aber Open Source Modelle wie Lllama, Mistral und Qwen.

Der **ecu.test** *agent* kann auch vollständig on-premise betrieben werden, damit sensible Testdaten zu keinem Zeitpunkt das eigene Unternehmensnetzwerk verlassen.

Wir helfen gern dabei, ein für dich passendes Setup mit einer fundierten Datenbasis aufzubauen. Melde dich bei unserem Support.

Hinweis: Mehr Informationen zum **ecu.test** *agent* gibt es auch auf unserer Homepage.

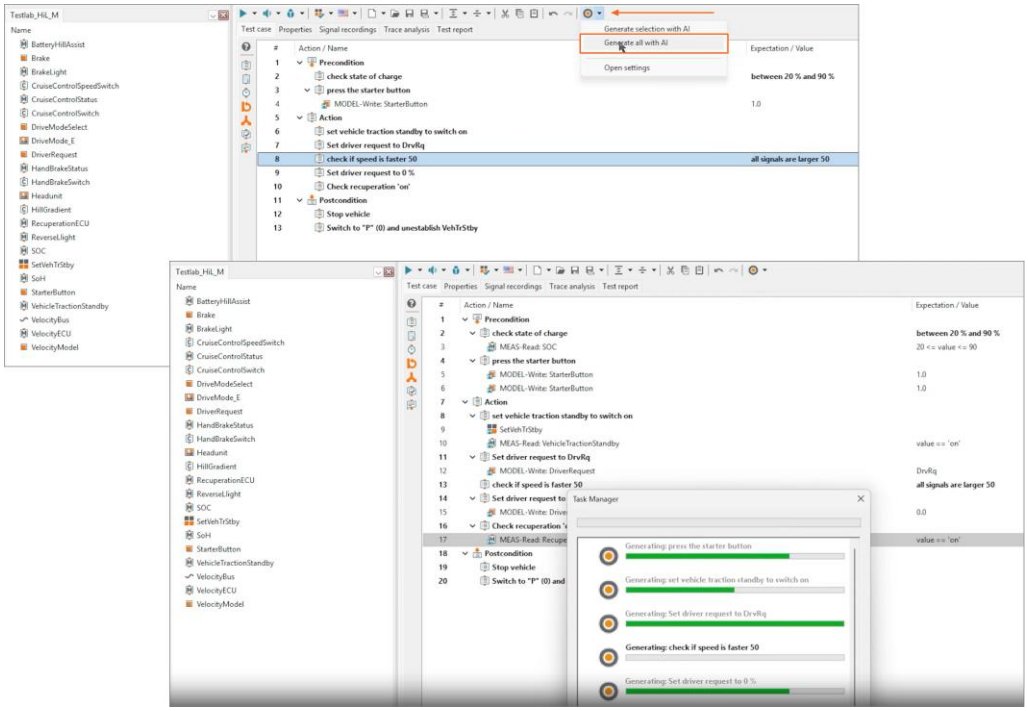


Abbildung 35: AI-basierte Testfallgenerierung mit dem ecu.test *agent*

Python-basierte Testfallerstellung mit ecu.test *code*

Mit **ecu.test** *code* testen Entwickelnde ihren Softwarecode selbst – in ihrer gewohnten Programmierumgebung und ohne das Tool wechseln zu müssen.

Das Python-basierte Framework integriert sich nahtlos in Visual Studio Code und bietet über Code-Completion leichtgewichtigen Zugriff auf **ecu.test** und damit auf mehr als 80 Tools und Automotive-Schnittstellen.

Entwickelnde schreiben Steuergerätecode, definieren passende Tests, führen beides in einer SiL-Umgebung aus und erhalten sofort verwertbare Ergebnisse direkt in der Programmierumgebung oder vom CI-System. Fehler im Code lassen sich so frühzeitig erkennen und direkt beheben.

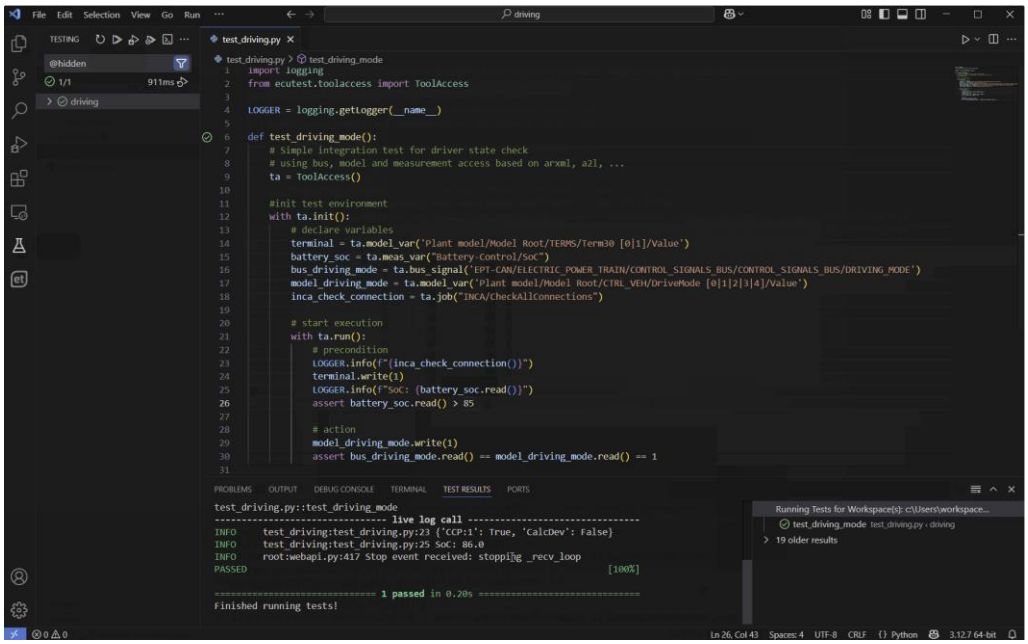
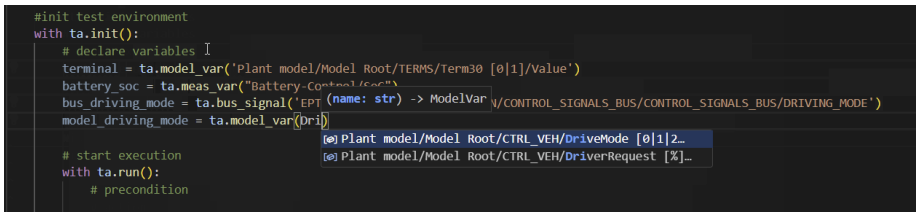


Abbildung 36: Erstellung von Testfällen mit ecu.test code

Vorteile:

- Zugriff auf alle von **ecu.test** unterstützten Tools und Automotive Schnittstellen über Code-Completion



- Leichtgewichtige und schlanke API, die ohne **ecu.test**-Vorkenntnisse verwendet werden kann
- Vollständige Integration in die tracetronic-Toolkette, u. a. **test.guide**
- Flexible Integration in CI-Umgebungen und Python-Testframeworks, wie unittests, pytest, , ...
- Kompatibel für Linux und Windows

Hinweis: Wir haben dein Interesse geweckt? Dann komm auf uns zu!
Wir stellen dir **ecu.test** *code* gerne auch schon vor dem offiziellen Release vor.

ecu.test Linux GUI für Ubuntu 24.04 LTS

Mit **ecu.test** 2023.1 wurde erstmals eine native Linux-GUI bereitgestellt. Damit war es grundsätzlich möglich, Tests auch unter Linux zu entwickeln, auszuführen und ggf. zu debuggen.

Verfügbare Tools, z. B. ADAS/AD-Tools, MATLAB/Simulink oder diverse Netzwerkschnittstellen konnten ohne Umwege wie Remote-Toolserver oder Virtualisierungen genutzt werden.

In der aktuellen Version wurde die Linux-GUI auf **Ubuntu 24.04 LTS** aktualisiert und in Zusammenarbeit mit Pilotanwendern umfassend getestet und stabilisiert.

Ein Blick auf die **ecu.test** Linux-Version lohnt sich – insbesondere für alle, die bereits mit Linux-basierten Toolchains arbeiten oder ihre Testumgebung plattformunabhängiger gestalten möchten.

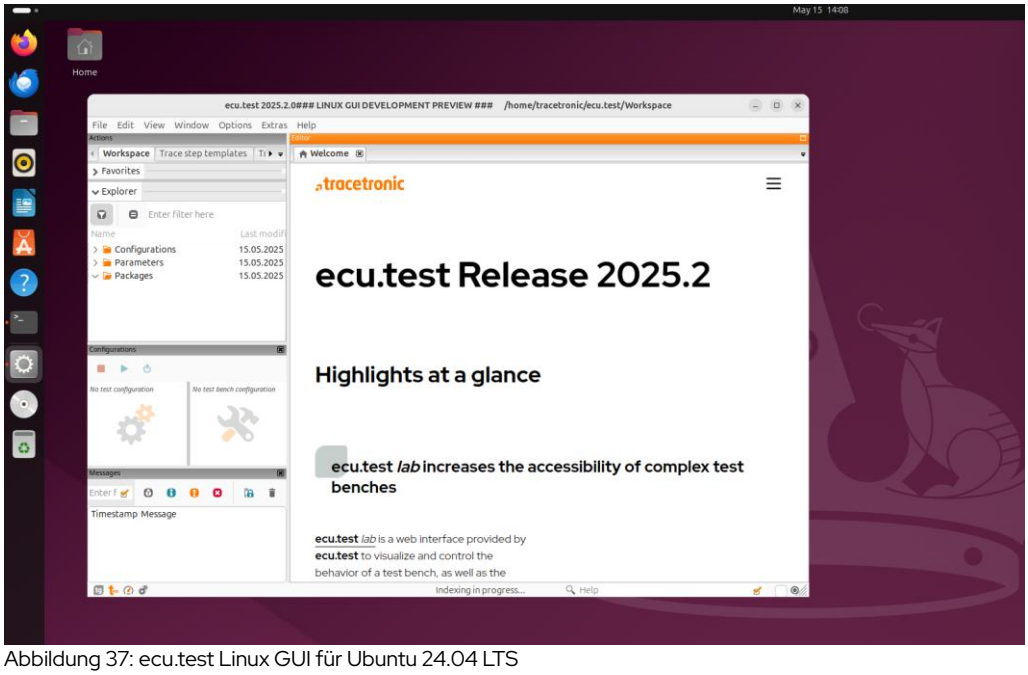


Abbildung 37: ecu.test Linux GUI für Ubuntu 24.04 LTS

5 Abkündigungen

5.1 Abkündigungen und Inkompatibilitäten in dieser Version

CARLA-Anbindung

et

Mit **ecu.test 2025.2** wird die CARLA-Anbindung entfernt.

SILVER XIL

et

Unterstützung für Versionen W-2024.9 und älter entfernt. Durch inkompatible Änderungen an der XIL-Schnittstelle von Silver ging die Unterstützung von W-2025.03 mit dem Ende der Unterstützung älterer Versionen einher.

BusBrowser API

et tc

Die Methode **GetPhysResolution()** wurde entfernt.
Bitte stattdessen die Methode **GetLinearCoeffs()** verwenden.

5.2 Abkündigungen in zukünftigen Versionen

"Alternative call representation" der Package-Eigenschaften

et

Die "Alternative call representation" in den allgemeinen Eigenschaften eines Packages gilt seit der Version 2024.3 als abgekündigt und wird mit **ecu.test 2025.2** endgültig entfernt.
Das Feature wurde im Rahmen des schlüsselwortbasierten Testens im Mapping von Referenzpackage-Aufrufen implementiert und findet ab sofort dort seine Anwendung.

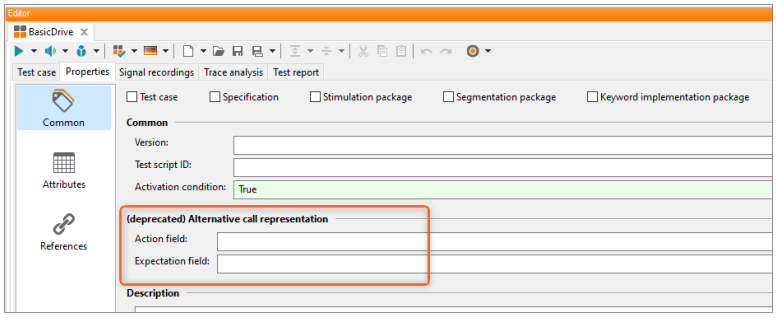


Abbildung 38: Package-Eigenschaften: Alternative call representation

Odx Migration Helper

et

Der **Odx Migration Helper** wird mit **ecu.test** 2025.3 entfernt.

DiagBrowser: GetUdsName

et

Die Methode **GetUdsName** wird mit **ecu.test** 2025.3 entfernt.

Abkündigung ReqIf Import

et

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

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.

Fibex-Unterstützung

et tc

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

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.

Abkündigung der integrierten test management Anbindung für Jama

et

Die fest in **ecu.test** integrierte Anbindung an Jama wird durch den neuen Python-basierten Ansatz ersetzt.

5.2.1 Zukünftig entfernte API-Methoden

et tc

Alter Befehl	Neuer Befehl	Anmerkungen
Report API		
ReportItem. GetActivity	ReportItem.GetLabel()	Veraltet seit Version 6.4: Name und Aktivität sind durch Label zu ersetzen.
ReportItem. SetActivity	ReportItem.SetLabel()	Veraltet seit Version 6.4: Name und Aktivität sind durch Label zu ersetzen.
ReportItem.GetName	ReportItem.GetLabel()	Veraltet seit Version 6.4: Name und Aktivität sind durch Label zu ersetzen.
ReportItem.SetName	ReportItem.SetLabel()	Veraltet seit Version 6.4: Name und Aktivität sind durch Label zu ersetzen.
ReportItems.ReportItem. GetActivity	ReportItem.GetLabel()	Veraltet seit Version 6.4: Name und Aktivität sind durch Label zu ersetzen.
ReportItems.ReportItem. SetActivity	ReportItem.SetLabel()	Veraltet seit Version 6.4: Name und Aktivität sind durch Label zu ersetzen.

ReportItems. ReportItem.GetName	ReportItem.GetLabel()	Veraltet seit Version 6.4: Name und Aktivität sind durch Label zu ersetzen.
ReportItems. ReportItem.SetName	ReportItem.SetLabel()	Veraltet seit Version 6.4: Name und Aktivität sind durch Label zu ersetzen.

Alter Befehl	Neuer Befehl	Anmerkungen
Object API		
API for Reports		
ReportAnalysisEpisode. GetActivity	ReportAnalysisEpisode. GetLabel	Veraltet seit Version 2020.2, nutze bitte: GetLabel().
ReportAnalysisEpisode. GetName	ReportAnalysisEpisode. GetLabel	Veraltet seit Version 2020.2, nutze bitte: GetLabel().
ReportAnalysisStep. GetActivity	ReportAnalysisStep.GetLabel	Veraltet seit Version 2020.2, nutze bitte: GetLabel().
ReportAnalysisStep.GetName	ReportAnalysisStep.GetLabel	Veraltet seit Version 2020.2, nutze bitte: GetLabel().