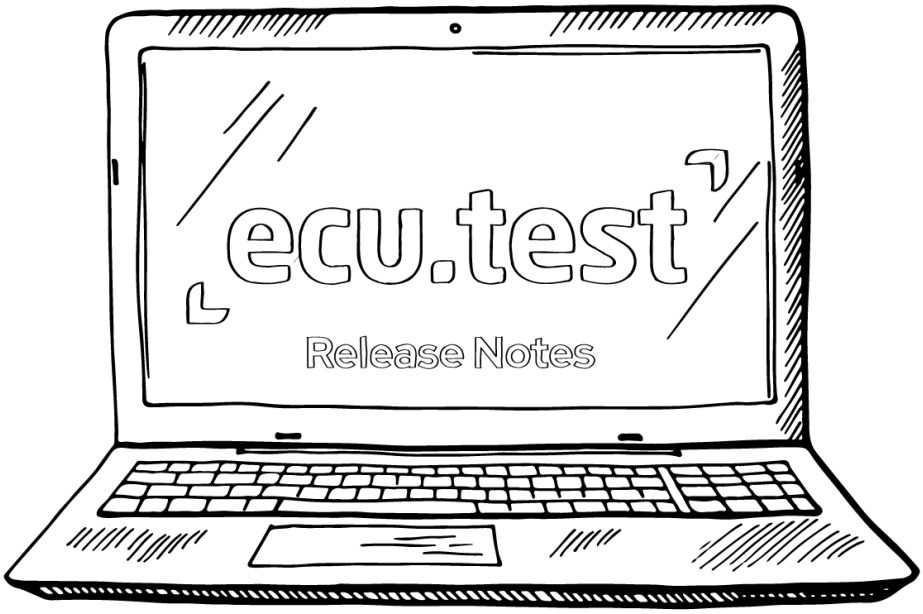
tracetronic



Release Notes

ecu.test 2025.3

trace.check 2025.3

Datum: 30.09.2025

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

tracetronic

Inhalt

tracetronic

Überblick

Mit dem **ecu.test** Release **2025.3** launchen wir gleich **zwei neue Produkte**:

ecu.test *agent*

... ist ein AI-gestützter Assistent, der vollständig in **ecu.test** integriert ist und die Generierung von Testfällen automatisiert.

ecu.test *code*

... ist **ecu.test** in einer Python-Umgebung und damit direkt auf die Anforderungen von ProgrammiererInnen zugeschnitten.

Weitere Highlights des Releases:

- **Modellzeit**
  Die Modellzeit kann von beliebigen als Zeitquelle konfigurierten Modellports bezogen und automatisch an Bus- und Diagnosekomponenten weitergeleitet werden.
- **Nutzerdefinierte Synchronisation von Aufnahmen**
  Eigene Synchronisationsarten können als UserPyModul implementiert werden.

ecu.test *lab* gehört erst seit der **2025.2** zu unserer Produktfamilie, wartet aber schon mit zahlreichen neuen Features darauf, euch zu begeistern, zu unterstützen und zum Ausprobieren zu animieren. Bis Ende 2025 könnt ihr **ecu.test** *lab* noch kostenlos testen. Meldet euch dafür bei unserem Support.

Dein Spezialgebiet in **ecu.test** ist aber die Kommunikation? Wie gut, dass es hierfür gleich sechs neue Features gibt. Stichworte sind: Some/IP, MACSec, CAN-FD und XL-API. Lies mehr dazu im Kapitel 3.9.

In diesem Release gibt zwei zentrale Erweiterungen in Verbindung mit **test.guide**. Zum einen wurde die Coverage-Analyse in **test.guide** integriert. So ist die Auswertung von Coverage-Dateien statt bisher nur lokal jetzt auch zentral möglich. Zum anderen enthalten exportierte Testfall-Kennzahlen in **trace.xplorer**-Dateien jetzt pro Testausführung einen Link, mit dem man direkt zu den zugehörigen Testlaufdetails in **test.guide** gelangt.

Zusätzlich zu den oben beschriebenen Features stehen auch zahlreiche Neuerungen für **ADAS/AD, Multimedia**, **SiL**- und **HiL**-Setups, **Diagnose, Traceanalyse** oder **Usability** zur Verfügung. Ein Blick lohnt sich!

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

1 Highlights in ecu.test 2025.3

ecu.test *agent*

Der **ecu.test** *agent* ist ein AI-gestützter Assistent, der die Erstellung von Testfällen in ecu.test erheblich beschleunigt. Er ist als Service vollständig in **ecu.test** integriert und generiert mit einem einzigen Klick automatisch passende, ausführbare Testschritte aus Testfallspezifikationen. Dazu greift er auf alle Daten zum aktuellen Kontext direkt aus dem Workspace zurück.

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

Wer eine aktive **ecu.test**-Lizenz hat, kann ab sofort 50 Testschritte pro Monat (50 Credits) kostenfrei generieren lassen. Hol dir dafür deinen Token bei unserem Sales-Team und los geht's. Darüber hinaus ist es natürlich auch möglich, größere Credit-Pakete zu kaufen.

Viele Informationen zu unserem neuen Tool erhältst du auch auf der Webseite des **ecu.test** *agent*.

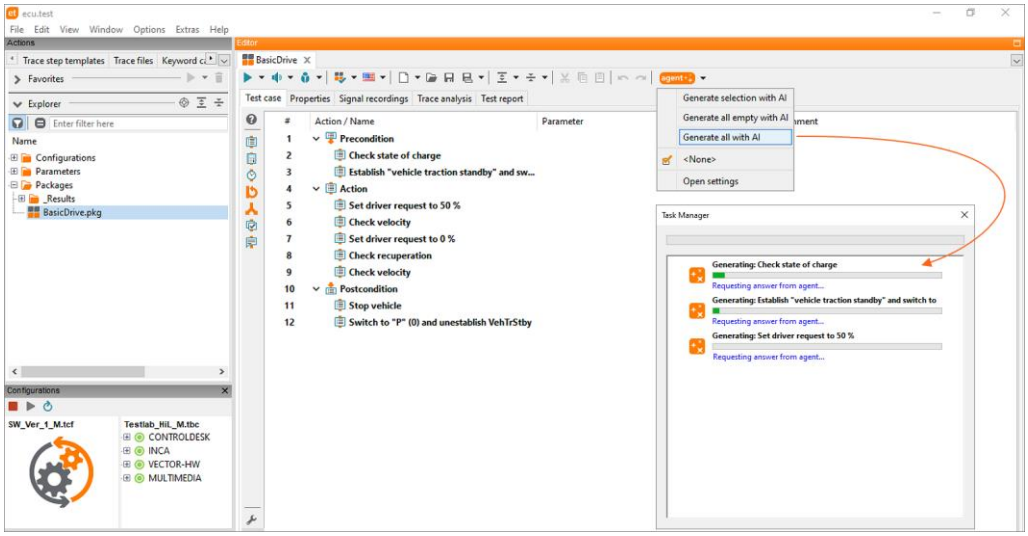


Abbildung 1: Generierung von Testfällen mit dem ecu.test *agent*

ecu.test *code*

Die Workflows zur Testentwicklung und die Testsprache in **ecu.test** wurden so konzipiert, dass sie den Anforderungen von Testingenieuren gerecht werden. Merkmale wie Wiederverwendung und eine intuitiv zu bedienende Oberfläche für die Testfallerstellung haben bei der Entwicklung von **ecu.test** einen hohen Stellenwert.

Da die Bedeutung und Verantwortung von SoftwareentwicklerInnen im Rahmen des Entwicklungsprozesses stetig zunehmen, sehen wir einen wachsenden Bedarf für leichtgewichtige Testlösungen, die in Code umgesetzt werden können.

Mit **ecu.test** *code* werden die Fähigkeiten von **ecu.test** in einer Python-Umgebung bereitgestellt, um den Bedarf aus beiden Welten zu decken. Es ist auf die Anforderungen von ProgrammiererInnen zugeschnitten und kann – ohne die umfangreichen Konzepte von **ecu.test** erlernen zu müssen – eingesetzt werden.

Dabei stellt es keinen Ersatz für **ecu.test** dar. Vielmehr handelt es sich um eine alternative Möglichkeit, von dem Funktionsumfang von **ecu.test** zu profitieren. Es fördert die Zusammenarbeit von IngenieurInnen und EntwicklerInnen und verfolgt den Shift-Left als Ziel.

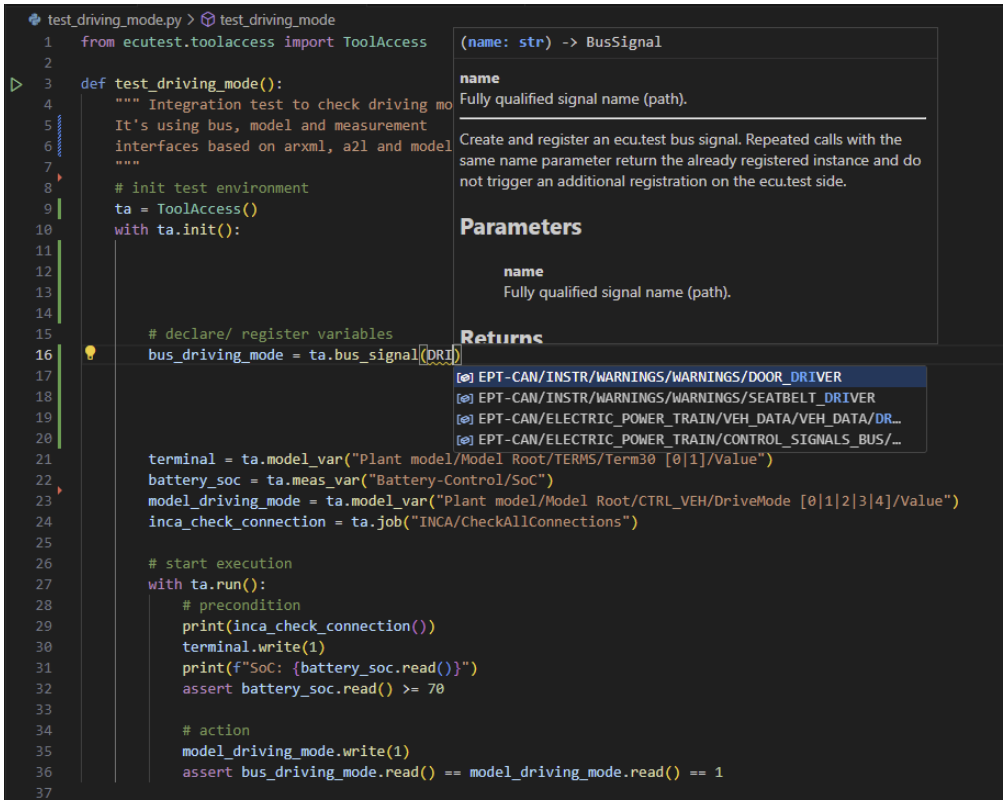


Abbildung 2: Testfallerstellung direkt in der Python-Umgebung

Features

- Erstellung Ausführung von Testfällen aus der einer beliebigen Python-Umgebung
- Zugriff auf alle von **ecu.test** unterstützten Tools
- Schneller Zugriff auf Testgrößen durch Autovervollständigung in Visual Studio Code
- Rückspielen der Ergebnisse durch nahtlose **test.guide**-Integration

Aufbau

- Die Python-Bibliothek ist der Hauptbestandteil von **ecu.test** *code*.
  Sie aktiviert den Zugriff auf die Funktionen von **ecu.test** direkt in der Python-Umgebung und ermöglicht den Zugriff auf **ecu.test**-Tools, Daten und Automatisierungsprimitive.
- Die VS Code-Erweiterung ist ein optionales Add-on. Sie baut auf der Python-Bibliothek auf und bietet umfangreiche Autovervollständigung, Online-Hilfe und optimiertes Authoring um Testfälle entwerfen, parametrisieren und ausführen zu können, ohne VS Code zu verlassen.
- Das pytest-Plugin ermöglicht die nahtlose Integration von Testberichten in **test.guide**.

Getting Started

- Wir freuen uns über Feedback und wollen **ecu.test** *code* anhand deiner Anwendungsfälle weiterentwickeln. Deshalb stellen wir **ecu.test** *code* zunächst auf Anfrage zur Verfügung.
  Sende uns einfach eine kurze E-Mail an support@tracetronic.com. Eine zusätzliche Lizenz zu **ecu.test** ist momentan nicht erforderlich.
- Für die Verwendung von Visual Studio Code kannst du die Erweiterung über den VS Code Marketplace installieren.
- Um schnell loslegen zu können, haben wir weitere Information in der **ecu.test**-Anwenderhilfe unter Erste Schritte zusammengefasst.

Automatische Auswirkung der Modellzeit auf alle busnahen Tools und die Diagnose

et

Mit **ecu.test** unterstützen wir das Testen des kompletten Entwicklungszyklus - vom ersten Softwarestand im SiL, über das Testen von Modellen im MiL, der Integration am HiL bis zur Erprobungsfahrt im Fahrzeug.

Die Testdomänen sind jedoch nicht immer trennscharf. So werden einige Low-cut-Modelle bereits mit virtueller Bushardware getestet. Diese hybriden Testaufbauten bringen neue Herausforderungen, insbesondere beim Zeitverhalten mit. So muss sich die vom Modell vorgegebene Zeit auch auf die einzelnen Kommunikationsschichten auswirken, damit z. B. die Aufnahmen mit dem Modell synchron sind oder Timeouts korrekt angewendet werden können.

Diese Hürde wird jetzt automatisch von **ecu.test** genommen, indem die Modellzeit automatisch an die Bus- und Diagnosekomponenten weitergeleitet werden.

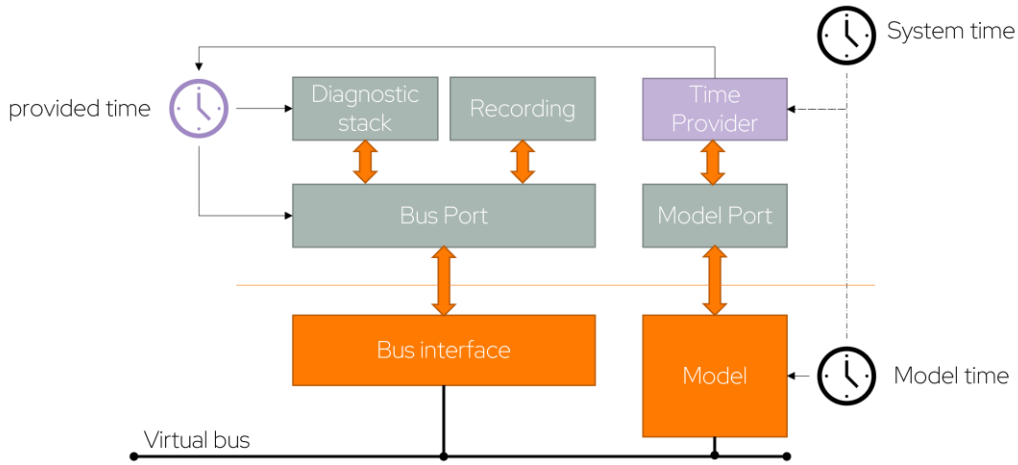


Abbildung 3: Schematische Darstellung der Modellzeitweiterleitung

Um diese zu aktivieren, muss lediglich ein Modellport in der Testkonfiguration (TCF) gewählt werden. Dies führt dazu, dass die Modellzeit automatisch in den Kommunikationskomponenten angewendet wird. Dies betrifft die Bus- und Diagnoseports sowie die tracetronic-Ethernet-Ports.

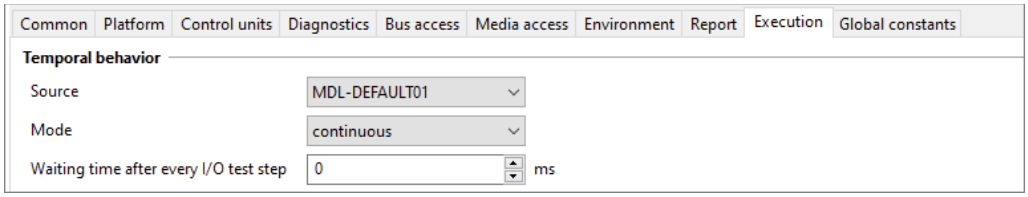


Abbildung 4: Aktivierung des Modellports

Hinweis: Bisher wird lediglich der kontinuierliche Modus unterstützt. Es bestehen jedoch schon Pläne für die schrittweise Ausführung. Sprich uns gerne zu diesem Thema an!

Nutzerdefinierte Synchronisation von Aufnahmen

et tc

Für die Absicherung komplexer Fahrzeugsysteme wird neben dem Zugriff auf einfache Signale auch der Zugriff auf Protokolle und Multimediadaten benötigt. Oft ist es für eine Traceanalyse notwendig, unterschiedliche Aufnahmen mehrerer Quellen nachträglich zeitlich zu synchronisieren. Die Synchronisation muss dabei die komplexen Daten für jeden Anwendungsfall individuell verarbeiten können.

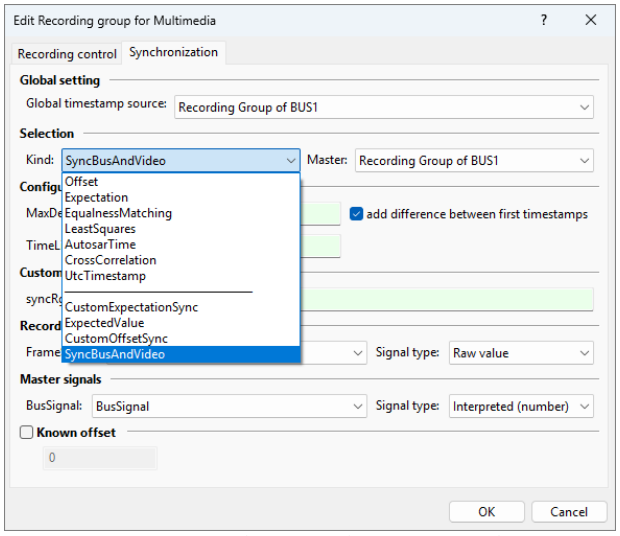


Abbildung 5: Wahl der Aufnahmeart für die nutzerdefinierte Synchronisation

Die nutzerdefinierte Synchronisation ermöglicht es, ähnlich zu Traceschrittvorlagen, auf Signalwerte zuzugreifen, eigene Algorithmen oder individuelle Bibliotheken anzuwenden und einen zeitlichen Offset zu bestimmen.

Bereitgestellt wird die Synchronisation in Form einer selbst implementierten Benutzer-Python-Bibliothek.

2 Usability

Testfall-Coverage: test.guide-Integration

Seit **ecu.test 2024.3** lassen sich Coverage-Metriken für Testfälle analysieren und seit **ecu.test 2025.2** in einem zentralen Bericht zusammenführen – bisher jedoch nur auf Basis lokaler Coverage-Dateien. Das war aufwendig, speziell bei verteilten Teams, vielen Testausführungen und großen Datenmengen.

Mit der neuen Integration in **test.guide** wird dieser Prozess deutlich effizienter: Über individuell definierbare Filter in der eigenen **test.guide**-Instanz können gezielt relevante Testausführungen ausgewählt werden. **ecu.test** lädt die zugehörigen Coverage-Daten direkt aus **test.guide** und erstellt daraus automatisch einen konsolidierten Bericht ohne manuelle Dateiablage.

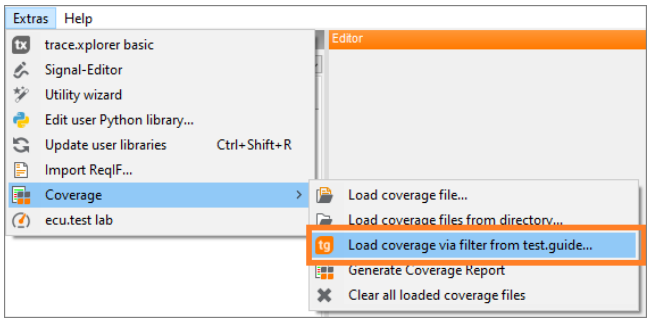


Abbildung 6: Neuer Eintrag im Coverage-Menü unter Extras

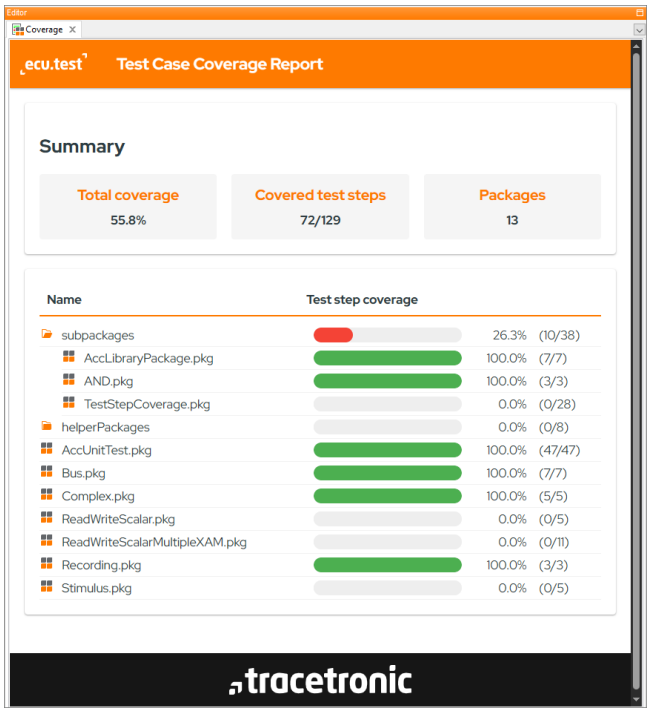


Abbildung 7: Konsolidierter Coverage-Bericht

Die Kombination mit **test.guide** erweitert die verfügbare Coverage-Analyse entscheidend: Sie ermöglicht nun eine zentrale Auswertung über viele Testläufe hinweg und ist ideal für die systematische Absicherung sicherheitskritischer Bibliotheken und die Identifikation ungenutzter Packages im gesamten Projekt.

Mit **ecu.test 2025.3** werden zudem Coverage-Aufnahmen automatisch zu **test.guide** geladen, wenn der Report-Upload aktiv ist. Diese Funktionalität steht ab **test.guide** 1.199.0 zur Verfügung.

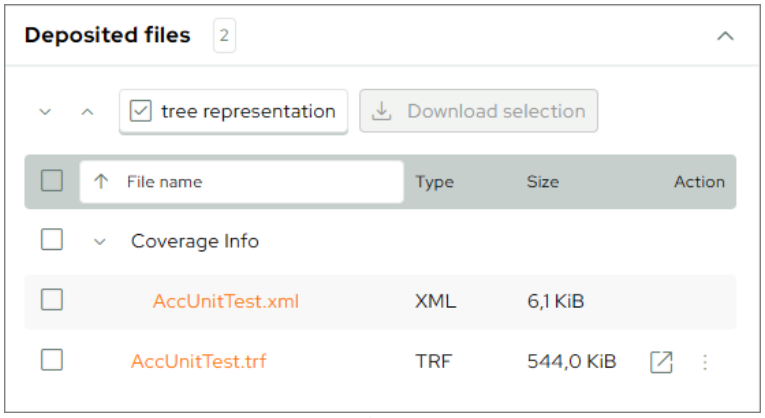


Abbildung 8: Ablage der Coverage-Aufnahmen in test.guide

Testfall Coverage: Verbesserungen im HTML-Report

In den HTML-Dateien des konsolidierten Coverage-Berichts ist es nun möglich, die maximale Anzahl der Top-Level-Elemente festzulegen. Außerdem werden eingebundene **ecu.test**-Bibliotheksworkspaces explizit dargestellt.

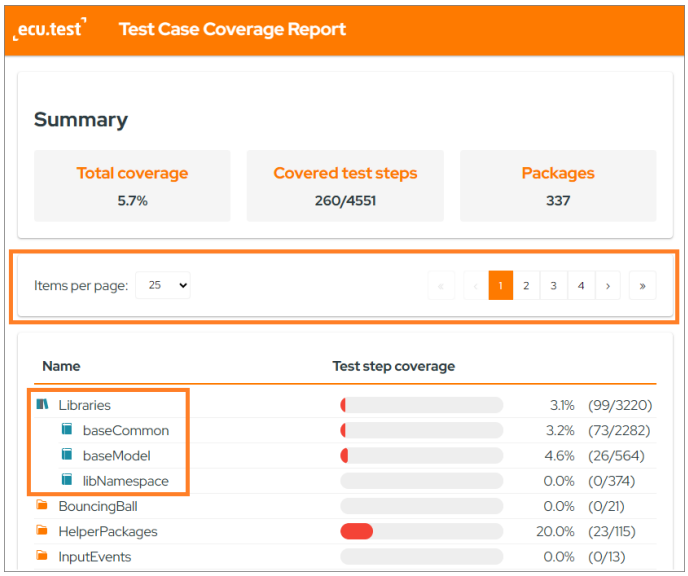


Abbildung 9: Konsolidierter Coverage-Bericht

Direkte Integration der Online-Hilfe in ecu.test

Die neue Online-Anwenderhilfe wurde bereits mit **ecu.test 2025.2** eingeführt. Ab **ecu.test 2025.3** wird sie nun auch beim Aufruf über das Startmenü …

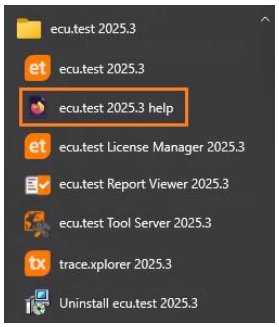


Abbildung 10: Hilfe im Startmenü

… sowie direkt aus dem Tool genutzt.

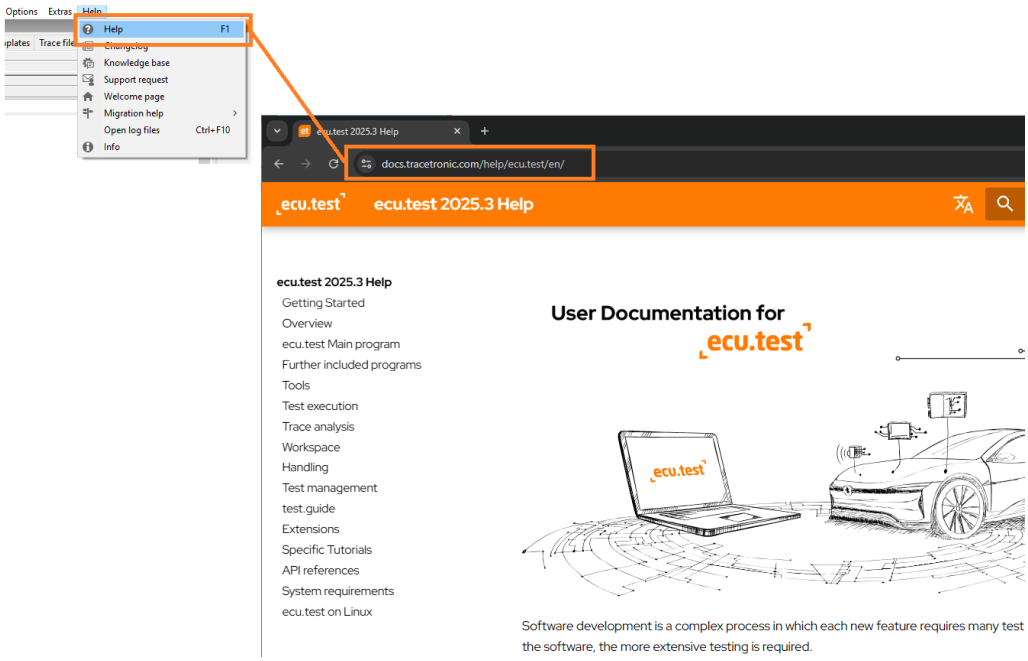


Abbildung 11: Beim Klick auf Hilfe öffnet sich nun primär die Online-Hilfe

Die Sprache wird automatisch erkannt, und falls keine Internetverbindung besteht, sorgt ein lokaler Fallback dafür, dass weiterhin Unterstützung verfügbar ist.

Durch die Online-Anbindung sind die Inhalte jederzeit aktuell, die Suche deutlich verbessert und künftig werden zusätzliche interaktive Funktionen bereitgestellt – für eine noch schnellere und gezieltere Hilfe bei der täglichen Arbeit.

Ethernet-Adapternamen anhand des Namens auswählen

Bislang wurde zur Identifikation eines Netzwerkadapters unter Windows der Hardware-Namen des Adapters einschließlich der MAC-Adresse verwendet. Da sich die MAC-Adresse für jeden Prüfplatz unterscheidet, musste für jeden eine eigene Testbenchkonfiguration (TBC) erstellt werden.

Ab dieser Version wird der Windows-Name zur Identifikation verwendet. Damit können TBCs auf verschiedenen Prüfplätzen wiederverwendet werden, wenn der Windows-Name gleich gewählt wird.

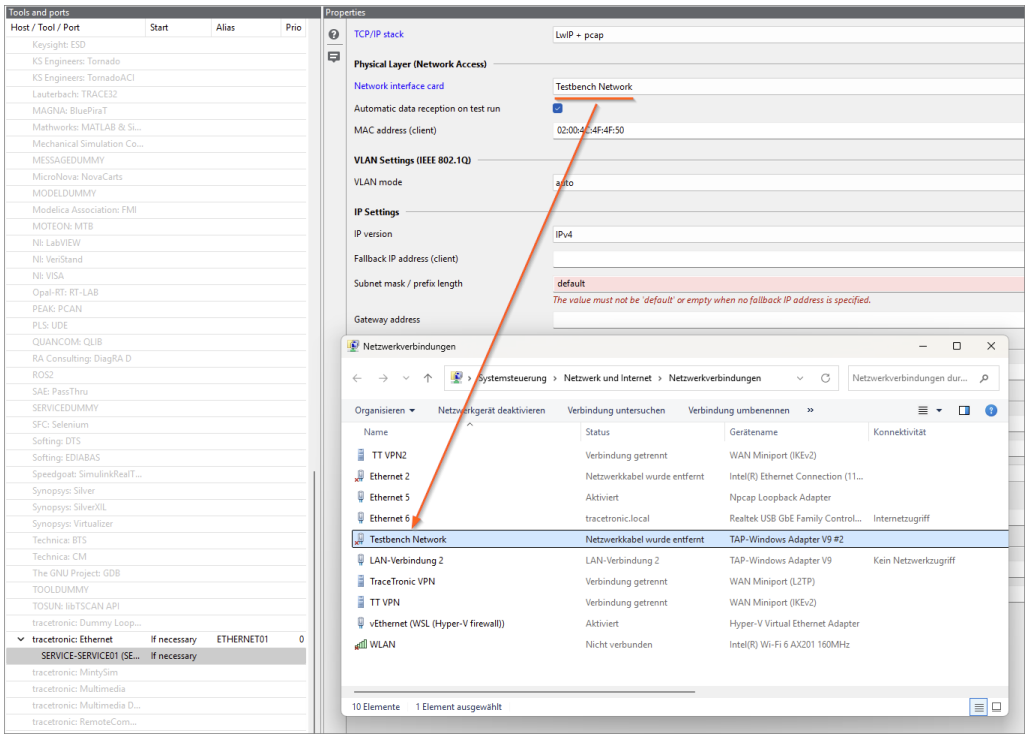


Abbildung 12: Verwendung des Windows-Namens zur Identifikation

Bitte beachten: Eine automatische Migration auf die neuen Bezeichner ist nicht möglich, sodass alle Ethernet-TBCs aktualisiert werden müssen und der jeweilige Netzwerkadapter neu ausgewählt werden muss.

Überarbeitetes Logging für deutliche Performancesteigerung

et tc

Das Logging wurde umfassend optimiert, wodurch insbesondere bei sehr umfangreichem Logging in Testfällen die Ausführung erheblich beschleunigt wird. Die neue Implementierung wird standardmäßig verwendet – sowohl in **ecu.test** als auch im **Tool Server** – und sorgt so für spürbar mehr Performance.

Bei Bedarf kann das Verhalten weiterhin in den Einstellungen unter **Protokoll** angepasst werden, um das bisherige Logging zu nutzen ("Synchrones Schreiben der Ausgabe-Logdateien").

Report Viewer: Umschaltbare Darstellung der Absolutzeit

et tc

Bisher wurden absolute Zeiten im Report immer in der lokalen Zeit des eigenen Rechners dargestellt. Für eine bessere zeitzonenübergreifende Zusammenarbeit und dem Nachvollziehen von Fehlern in externen Logs, ist es jedoch hilfreich die Zeiten nicht selbst umrechnen zu müssen.

So ist es über die Toolbar des Report Viewers nun möglich, zwischen drei Darstellungsformen zu wählen: Lokale Zeit, Zeit des Testhosts, der den Report erstellt hat, oder UTC-Zeit ohne Offset.

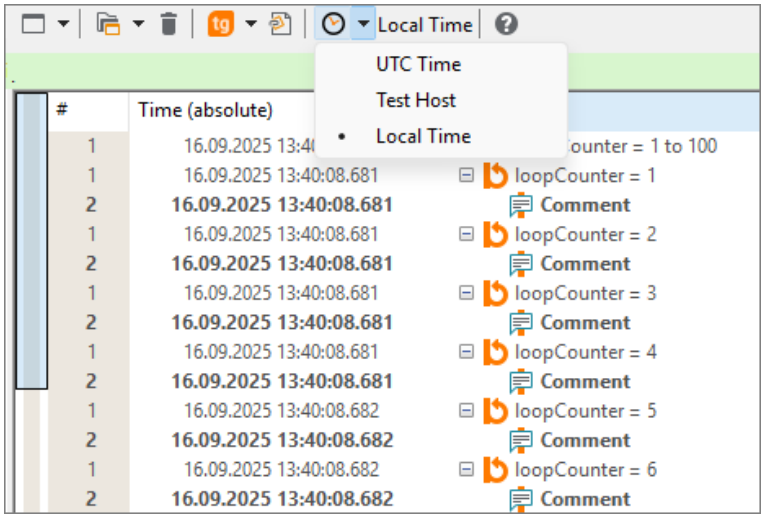


Abbildung 13: Umschaltbare Darstellung der Absolutzeit

Optimierte Git-Anbindung

et tc

Die Git-Anbindung wurde optimiert, was sich positiv auf die erste Ausführung größer Projekte aus versionierten Workspaces auswirkt.

Einstellungsdateien werden robust geschrieben und geladen

et tc

In seltenen Fällen konnte es dazu kommen, dass lokale oder Workspace-Einstellungsdateien fehlerhaft waren. Z. B. konnten externe Skriptlösungen oder ein hartes Beenden von **ecu.test**, während es Einstellungsdateien schreibt, zu einer leeren oder unvollständigen Datei führen.

Mit **ecu.test 2025.3** wurde die Verarbeitung deutlich robuster gestaltet. Das Schreiben wurde atomar gestaltet. Zusätzlich werden beim Versuch eine fehlerhafte Datei zu lesen, die Standardeinstellungen wiederhergestellt.

Ein- und ausblenden von verstecken Dateien im Workspace Explorer

et tc

Bisher hat **ecu.test** grundsätzlich alle Dateien und Ordner ausgeblendet, die mit einem Punkt im Namen beginnen. Um dies, insbesondere bei der Entwicklung und Nutzung von Bibliotheksworkspaces, konfigurierbar zu machen, wurde ein neuer Button im Workspace Explorer ergänzt.

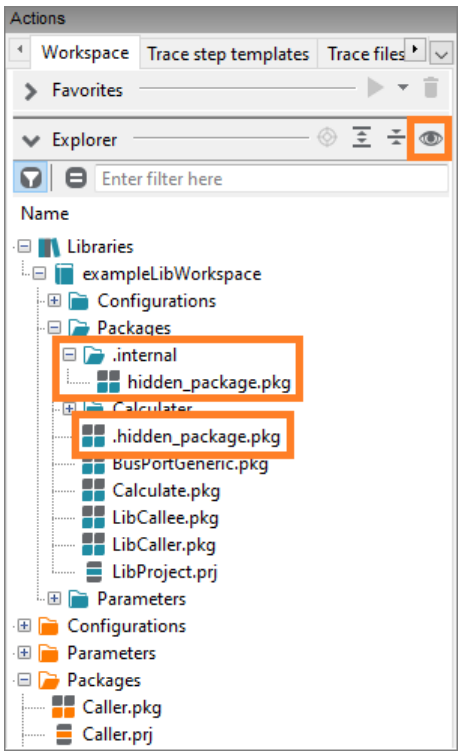


Abbildung 14: Der neue Button, ein Auge, wurde ergänzt, um verstecke Dateien bei Bedarf anzuzeigen.

Parametrierte Testfallausführung mit letzter Parametrierung erneut ausführen

Die gewählten Werte einer parametrierten Packageausführung werden jetzt gespeichert und bei einer erneuten Ausführung wieder angeboten.
Bei einer größeren Anzahl an Parametern und deren Werten kann bei Änderungen einzelner Werte die Zeit für das erneute Setzen eingespart werden.

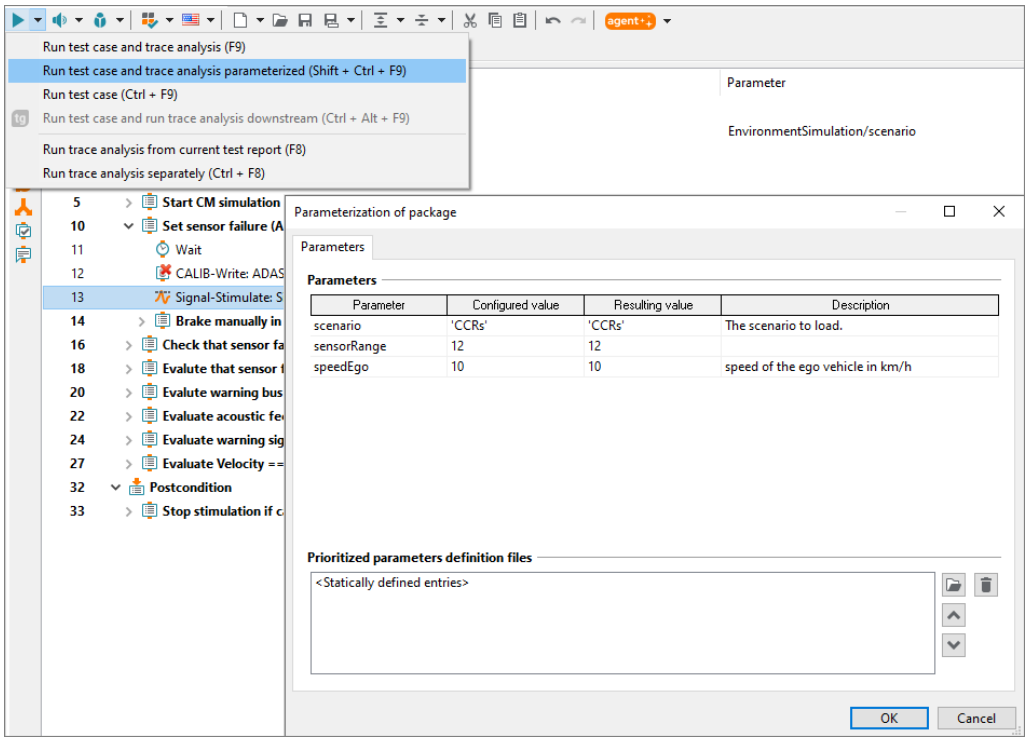


Abbildung 15: Testfall und Traceanalyse parametriert ausführen

ecu.test berücksichtigt die Anzeigeskalierung des Systems

Bei der Verwendung von höher aufgelösten Bildschirmen wird auf eine Anzeigeskalierung von 125% bis 300% verwendet. **ecu.test** ist nun in der Lage diese Einstellung pro Bildschirm zu erkennen und seine Darstellung anzupassen, sodass insbesondere Texte gut lesbar sind.

3 Testaspekte

3.1 ADAS/AD

CarMaker: Echtzeitfähigkeit von Stimulus-Schritten

et

Beim Einsatz von CarMaker an Prüfständen wie xPack oder Scalexio ist es oft notwendig, Signale zu schreiben. Mit dem Stimulus-Schritt gibt es die Möglichkeit, einen Signalverlauf aus Konstanten und auch Rampen zu erstellen und zur Ausführung zu bringen. Mit den bisherigen Versionen von **ecu.test** konnte die Ausführung solcher Signalverläufe noch nicht in Echtzeit gewährleistet werden.

In **ecu.test 2025.3** gibt es eine technische Anpassung, sodass diese Signalverläufe der Stimulus-Schritte auf der Echtzeitplattform zur Ausführung gebracht werden. Dadurch wird gewährleistet, dass die im Stimulus definierten Werte und die Dauer auch exakt eingehalten werden. Für bereits bestehende Stimulus-Schritte benötigt es keine Anpassung bzw. Migration.

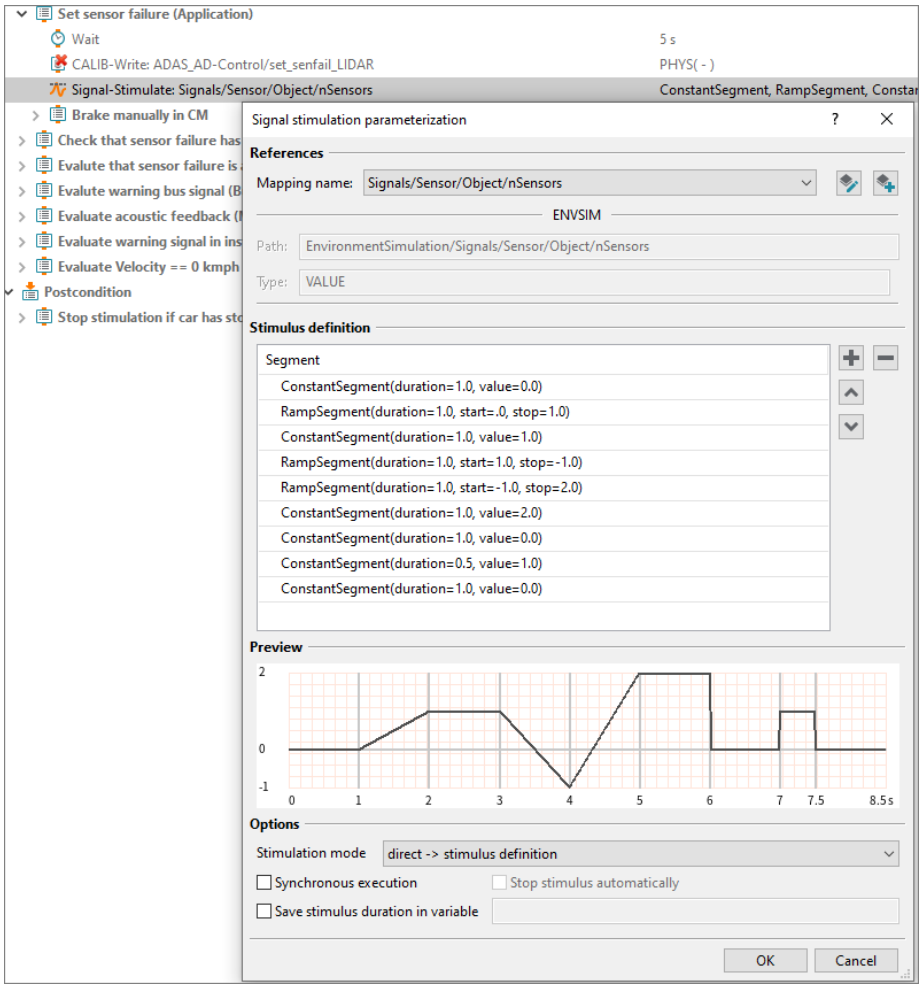


Abbildung 16: Echtzeitfähigkeit von Stimulus-Schritten

ModelDesk: Performanceoptimierung beim Aktivieren und Download von Szenarien und Strecke

et

Bisher wurde beim Ausführen der Tooljobs **Activate** und **Download** das komplette Experiment gespeichert. Bei größeren Experimenten führt dies zu langen Ausführungszeiten der besagten Schritte.

Mit **ecu.test 2025.3** werden Experimente gespeichert, wenn tatsächlich Änderungen durch bspw. das Setzen von Werten vorgenommen wurden. Ohne Änderungen am Experiment werden die Tooljobs **Activate** und **Download** jetzt deutlich schneller ausgeführt.

3.2 Multimedia

SFE: Selenium: Direkte Interaktion mit WebDriver-Objekt

et

Der Zugriff das **WebDriver**-Objekt der Selenium-API ist über den folgenden neuen Tooljob möglich:

- **GetWebDriver**

Für Anwendungsfälle, die durch unsere vorhandenen Jobs nicht abgedeckt sind, bietet dieser Zugriff einen direkten Zugang auf die Selenium-API und damit einen ungeführten, aber breiteren Funktionsumfang.

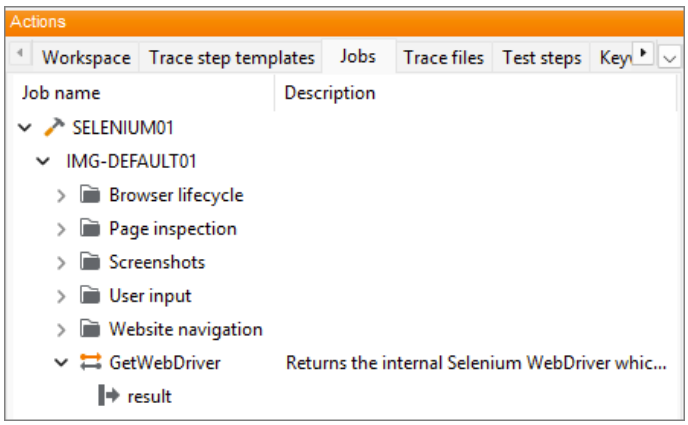


Abbildung 17: Job „GetWebDriver“ zur Interaktion mit WebDriver-Objekt

EMVA: GenICam ist wieder verfügbar

et

Die Inkompatibilitäten sind behoben. Daher steht **GenICam** wieder im selben Funktionsumfang zur Verfügung wie bereits bis **ecu.test 2024.4**.

tracetronic: Multimedia: Audio-Port für RTP-Streams

et

Unsere interne tracetronic: Multimedia-Anbindung bietet nun einen neuen Port zum Lesen und Aufzeichnen von RTP-Audioströmen.

Die Konfiguration erfolgt über eine SDP-Datei. Danach kann der Strom analog zu anderen Audioquellen im Testfall verwendet werden.

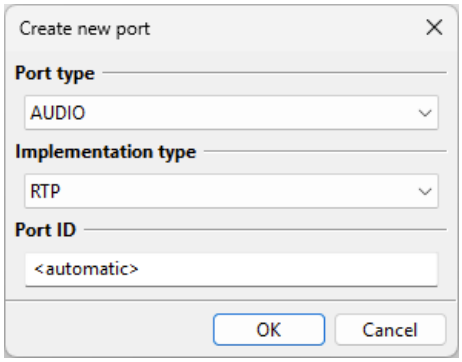


Abbildung 18: Audio-Port für RTP-Streams

Neue Methode "FilterByText" auf TextMatchList

et

Bei der Verwendung von OCR-Algorithmen abseits des mitgelieferten **TesseractOCR** fehlte bisher eine Methode, um eine Liste von im Bild gefundenen Texten nach dem Vorkommen eines bestimmten Textes zu durchsuchen.

Diese Möglichkeit wurde mit **FilterByText** auf der **TextMatchL** ist nun geschaffen.

Abbildung 19: Methode FilterByText

ADB: Screen-Recording

et

Die Toolanbindung an Google: ADB unterstützt nun auch die Aufnahme von Videos über die Signalaufnahme im Package.
So können nun auch kontinuierliche Prüfungen im Rahmen der Traceanalyse ausgeführt werden.

Voraussetzung für die Videoaufnahme ist eine aktuelle Android-Version (Version 11/API-Level 30).

3.3 SiL

Unterstützung des FMI Layered Standard for Network Communication (FMI-LS-BUS)

et

Ab Version **2025.3** unterstützt **ecu.test** den FMI Layered Standard (LS) Bus. Mit dem neuen Porttyp **CAN** wird nun automatisch auf Basis des LS mit der FMU kommuniziert (Teil "low-cut").

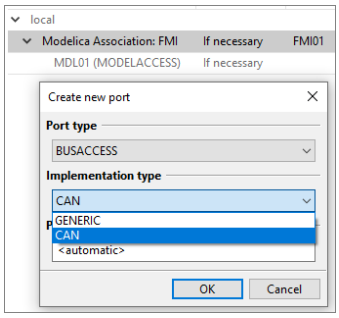


Abbildung 20: Anlegen eines neuen CAN-Ports für Tool FMI, der den LS Bus verwendet

Die umfangreiche und toolunabhängige Busfunktionalität, die **ecu.test** seit vielen Jahren im HiL- und SiL-Bereich anbietet, kommt nun auch bei der Verwendung von FMI zum Einsatz. Dadurch wird die Lücke zwischen modellorientierten (Unit-) Tests hin zu komplexen Integrationstests geschlossen.

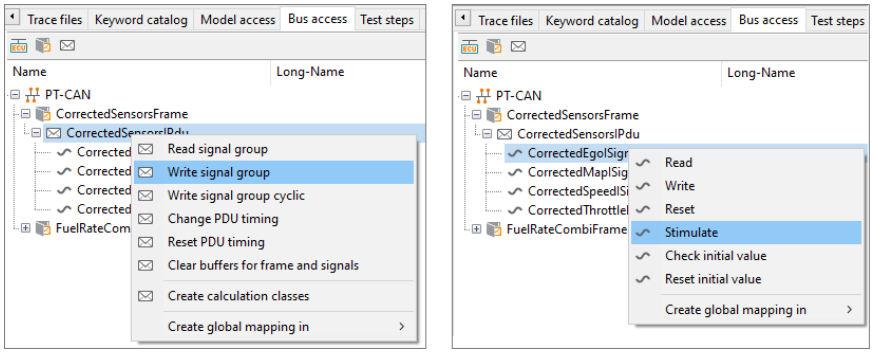


Abbildung 21: Möglichkeiten des Buszugriffs

Unterstützung des FMI Layered Standard for XCP (FMI-LS-XCP)

Neben dem LS Bus kann nun auch der Layered Standard for XCP in **ecu.test 2025.3** verwendet werden. Zusammen mit dem Extra **ecu.test** *calibration* kann ein gemäß LS XCP integrierter XCP-Server aus **ecu.test** heraus genutzt werden.

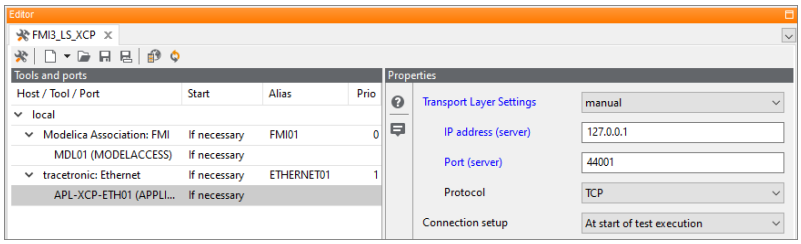


Abbildung 22: Einrichtung von ecu.test *calibration* zur Verwendung von FMI-LS-XCP

Die Einrichtung und Verwendung funktioniert analog zu anderen Mess- und Kalibrierungswerkzeugen. Dank LS XCP und **ecu.test** *calibration* kann Steuergerätesoftware in einer frühen Entwicklungsphase praxisnah genutzt und getestet werden, ohne dass weitere Tools notwendig sind.

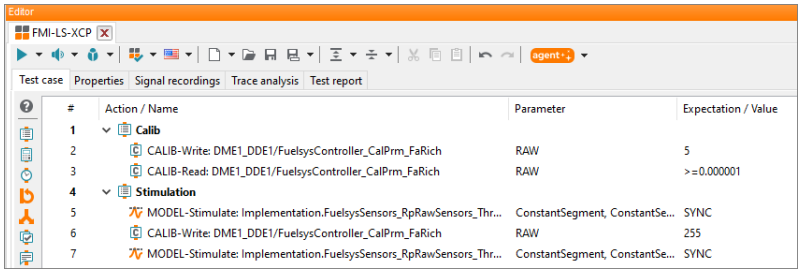


Abbildung 23: Typischer Testfall, der sowohl Modell- als auch Kalibrierzugriffe darstellt

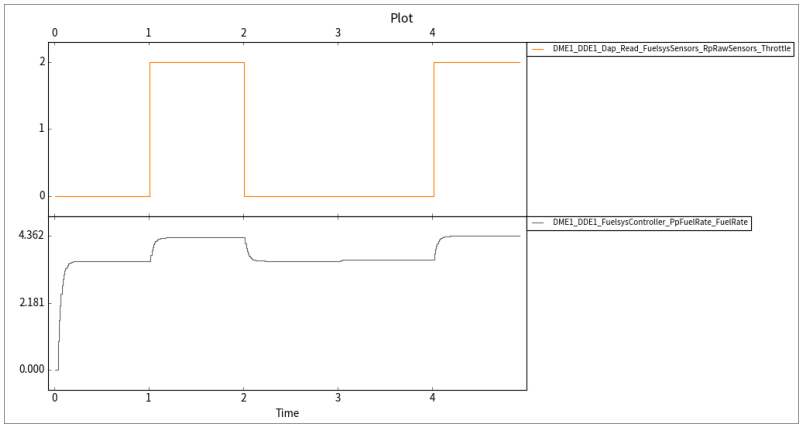


Abbildung 24: Beispiel einer Traceanalyse

Die Einrichtung und Nutzung wird ausführlich in der Anwenderdokumentation beschrieben.

Synopsys: Silver Unterstützung unter Linux

et

Die Toolanbindung Synopsys: Silver steht ab sofort auch unter Linux zur Verfügung.
Dies sollte den Wunsch nach einer Verlagerung von Testaufwänden in Linux-Umgebungen unterstützen, wenngleich eine Unterstützung von SilverSim hierdurch noch nicht gegeben ist.

Es wird also weiterhin eine komplette Silver-Lizenz und eine Ausführung der grafischen Oberfläche vorausgesetzt.

Hinweis: Die Anbindung Synopsys: SilverXIL steht weiterhin nur unter Windows zur Verfügung.

3.4 HiL

CANape: Buszugriff für Socket-Adaptor

et

Um den Zugriff auf Ethernet-Größen über CANape zu ermöglichen, kann der Busport in **ecu.test** jetzt auch für den Zugriff auf das Socket-Adaptor Protokoll verwendet werden.

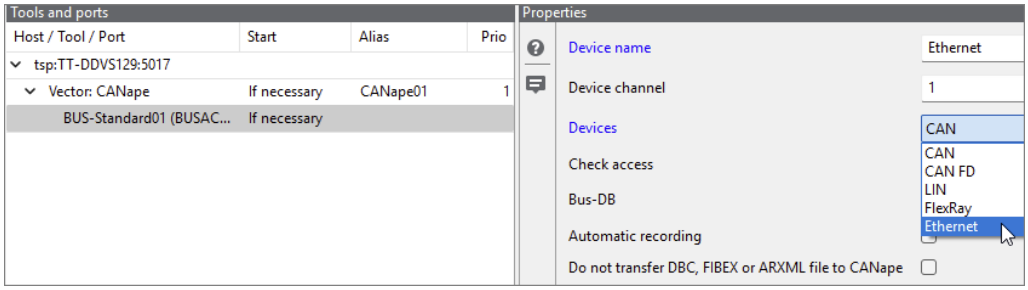


Abbildung 25: Zugriff auf Ethernet-Größen über den Busport

dSPACE: ControlDesk: Service_Manager Port

et

Der Port bietet nun die Möglichkeit der Aufzeichnung skalarer Blattknoten in der Signalaufnahme und dadurch die nachgelagerte Analyse des Signalverlaufs in der Traceanalyse.

Außerdem unterstützt der Port nun Werte in Form von String-Arrays in UTF8 und UTF16. Diese können nun als Text gelesen und geschrieben werden.

dSPACE: ControlDesk: Automatischer Reset des Modells

et

Wir bieten nun eine neue Eigenschaft auf dem Modell-Port an, um das Modell beim Teststart automatisch zurückzusetzen. Diese Option soll insbesondere in SIL-Umgebungen zu einer performanten Bereitstellung einer frisch initialisierten Umgebung beitragen, um Tests unabhängig vom Kontext und ohne aufwändiges Rücksetzen von Simulationen ausführen zu können.

dSPACE: ControlDesk: Optionaler Overwrite bei ImportDCMFile

et

Bisher warf **ecu.test** beim Laden einer DCM/CDFX-Datei einen Fehler, falls bereits eine mit dem gleichen Dateinamen geladen war. Dieses Verhalten ist im Job nun konfigurierbar. Neben dem Bestandsverhalten (default) stehen noch **Überschreiben** und **Import unterlassen** zur Auswahl.

INCA: Experiment nach der Testfallausführung wiederherstellen

et

Damit vorkonfigurierte Experimente in Verbindung mit **ecu.test**-Testfällen kombiniert werden können, wird ein INCA Experiment nach der Testfallausführung nun so wiederhergestellt, wie es vor der Ausführung des Testfalls vorgefunden wurde. Dies umfasst die Wiederherstellung:

- der Testgrößen
- der ausgewählten Messraster
- des Anzeige/Aufnahmestatus

3.5 Testmanagement

Jama: Erweiterung des Beispiel Workflows

et

Mit **ecu.test 2025.2** wurde die neue User-Testmanagement-API eingeführt und für das ALM-Tool Jama ein erster Beispiel-Workflow erstellt.

Dieser Workflow wurde nun überarbeitet: Die Testdurchführung kann in Jama direkt über TestCycles oder – allgemeiner – über TestPlans geplant werden. Da TestPlans den generischeren Ansatz darstellen, wurde der **ecu.test**-Projekt-Import vom reinen TestCycle-Import auf den TestPlan-Import umgestellt.

ecu.test User-Testmanagement-API: Beispiel-Workflow für Polarion

et

Mit **ecu.test 2025.2** wurde die neue User-Testmanagement-API eingeführt. Sie ermöglicht die vollständige Anbindung von **ecu.test** an ein beliebiges ALM-System – ausschließlich in Python.

Für das ALM Polarion steht jetzt ein erster Beispiel-Workflow bereit. Dieser importiert WorkItems aus Polarion nach **ecu.test**. Der Workflow nutzt die Polarion-REST-API und greift über ein Personal Access Token (PAT) sicher und passwortlos auf die Daten zu. Diese neue Lösung kann einfach an die die individuelle Konfiguration der Polarion-Instanz angepasst werden.

3.6 ecu.test *calibration*

Auswahl der Transportschicht aus einer A2L-Datei ermöglichen

et

Eine A2L-Datei kann unterschiedliche Konfigurationen für verschiedene Transportschichten enthalten. Damit die richtigen Parameter verwendet werden, kann die Transportschicht ab sofort ausgewählt werden.

Die Auswahl einer Transportschicht ist für die Verwendung von **ecu.test** *calibration* erforderlich damit die richtigen Verbindungseinstellungen ermittelt werden können. Für andere Mess- und Kalibrierwerkzeuge wirkt sich die Auswahl nicht aus und der Standardwert <ALL> kann beibehalten werden. Die Auswahl von <ALL> stellt alle verfügbaren Messraster bereit.

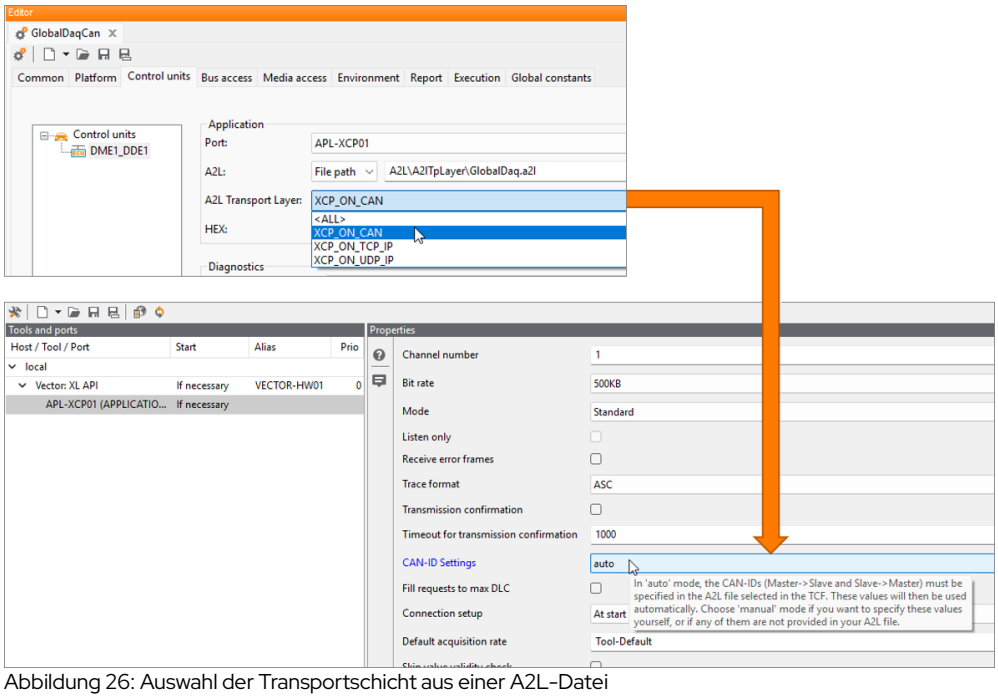


Abbildung 26: Auswahl der Transportschicht aus einer A2L-Datei

3.7 ecu.test *diagnostics*

Umfangreiche Verbesserungen bei der Verwendung von symbolischen Diagnosetestschritten

et

Einige Diagnoseservices sind mit komplexen Parametern und Rückgabewerten bedatet. Diese können tief verschachtelte Strukturen und Arrays sein. Es fällt oft schwer, diese in Gänze zu erfassen.

Mit der **2025.3** listen wir im **Parameter**-Block alle Blattknoten auf, welche der Nutzer befüllen kann. Falls in der Datenbasis ein Defaultwert angegeben wurde, ist dieser schon vorausgewählt. Zudem gibt es einen Button für die Autovervollständigung. Diese Funktion ist insbesondere für Bestandtestschritte interessant.

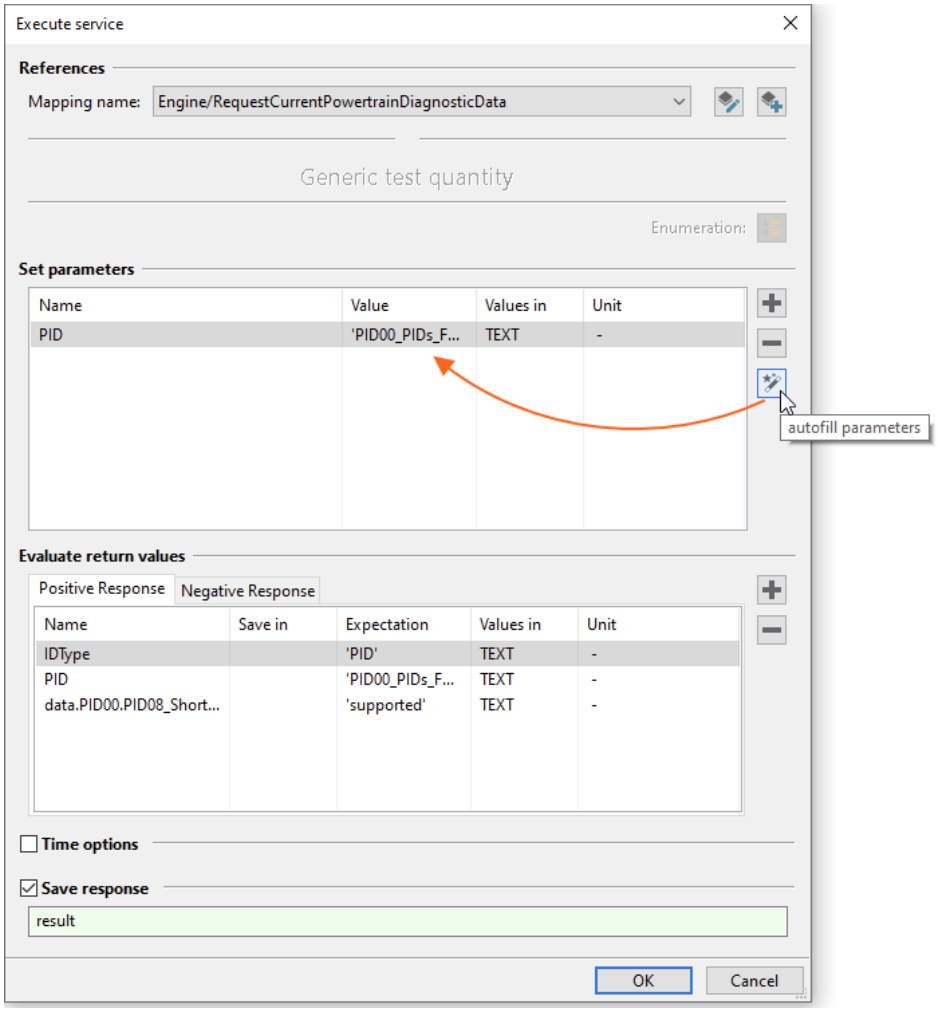
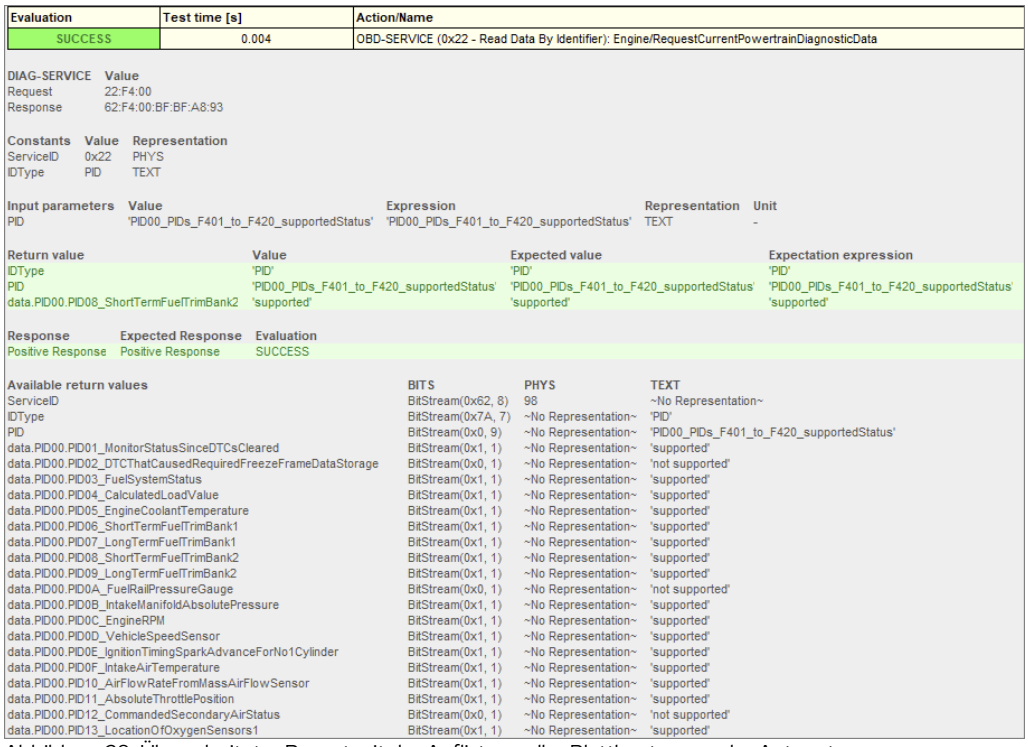


Abbildung 27: Neuer Button zur Autovervollständigung

Auch die Reports wurden überarbeitet. In dem neuen Abschnitt **Verfügbare Rückgabewerte** werden alle Blattknoten aus der Antwort aufgelistet.
Dies schafft einen besseren Überblick insbesondere bei komplexen Rückgabewerten.



Evaluation	Test time [s]	Action/Name
SUCCESS	0.004	OBD-SERVICE (0x22 - Read Data By Identifier): Engine/RequestCurrentPowertrainDiagnosticData

DIAG-SERVICE	Value
Request	22:F4:00
Response	62:F4:00:BF:BF:A8:93

Constants	Value	Representation
ServiceID	0x22	PHYS
IDType	PID	TEXT

Input parameters	Value	Expression	Representation	Unit
PID	'PID00_PIDs_F401_to_F420_supportedStatus'	'PID00_PIDs_F401_to_F420_supportedStatus'	TEXT	-

Return value	Value	Expected value	Expectation expression
IDType	'PID'	'PID'	'PID'
PID	'PID00_PIDs_F401_to_F420_supportedStatus'	'PID00_PIDs_F401_to_F420_supportedStatus'	'PID00_PIDs_F401_to_F420_supportedStatus'
data.PID00.PID08_ShortTermFuelTrimBank2	'supported'	'supported'	'supported'

Response	Expected Response	Evaluation
Positive Response	Positive Response	SUCCESS

Available return values	BITS	PHYS	TEXT
ServiceID	BitStream(0x62, 8)	98	~No Representation~
IDType	BitStream(0x7A, 7)	~No Representation~	'PID'
PID	BitStream(0x0, 9)	~No Representation~	'PID00_PIDs_F401_to_F420_supportedStatus'
data.PID00.PID01_MonitorStatusSinceDTCsCleared	BitStream(0x1, 1)	~No Representation~	'supported'
data.PID00.PID02_DTCThatCausedRequiredFreezeFrameDataStorage	BitStream(0x0, 1)	~No Representation~	'not supported'
data.PID00.PID03_FuelSystemStatus	BitStream(0x1, 1)	~No Representation~	'supported'
data.PID00.PID04_CalculatedLoadValue	BitStream(0x1, 1)	~No Representation~	'supported'
data.PID00.PID05_EngineCoolantTemperature	BitStream(0x1, 1)	~No Representation~	'supported'
data.PID00.PID06_ShortTermFuelTrimBank1	BitStream(0x1, 1)	~No Representation~	'supported'
data.PID00.PID07_LongTermFuelTrimBank1	BitStream(0x1, 1)	~No Representation~	'supported'
data.PID00.PID08_ShortTermFuelTrimBank2	BitStream(0x1, 1)	~No Representation~	'supported'
data.PID00.PID09_LongTermFuelTrimBank2	BitStream(0x1, 1)	~No Representation~	'supported'
data.PID00.PID0A_FuelRailPressureGauge	BitStream(0x0, 1)	~No Representation~	'not supported'
data.PID00.PID0B_IntakeManifoldAbsolutePressure	BitStream(0x1, 1)	~No Representation~	'supported'
data.PID00.PID0C_EngineRPM	BitStream(0x1, 1)	~No Representation~	'supported'
data.PID00.PID0D_VehicleSpeedSensor	BitStream(0x1, 1)	~No Representation~	'supported'
data.PID00.PID0E_IgnitionTimingSparkAdvanceForNo1Cylinder	BitStream(0x1, 1)	~No Representation~	'supported'
data.PID00.PID0F_IntakeAirTemperature	BitStream(0x1, 1)	~No Representation~	'supported'
data.PID00.PID10_AirFlowRateFromMassAirFlowSensor	BitStream(0x1, 1)	~No Representation~	'supported'
data.PID00.PID11_AbsoluteThrottlePosition	BitStream(0x1, 1)	~No Representation~	'supported'
data.PID00.PID12_CommandedSecondaryAirStatus	BitStream(0x0, 1)	~No Representation~	'not supported'
data.PID00.PID13_LocationOfOxygenSensors1	BitStream(0x1, 1)	~No Representation~	'supported'

Abbildung 28: Überarbeiteter Report mit der Auflistung aller Blattknoten aus der Antwort

J1939 aus unterschiedlichen Aufnahmeformaten

et

Die Möglichkeiten zur Prüfung der Diagnosekommunikation im Rahmen der Traceanalyse wurden weiter gestärkt. Bisher konnte J1939 nur aus ASC-Aufnahmen gelesen werden.

Mit **ecu.test 2025.3** ist es nun auch möglich, J1939 aus BLF-, GIN-, MDF4- und TTL-Aufnahmen zu lesen.

Kategorisierung von Jobs für Diagnoseports

Für die Diagnoseports sind in den letzten Jahren eine Vielzahl an Jobs entstanden. Mit der Zeit wurde der **Job**-Reiter dadurch unübersichtlicher.
Mit Version **2025.3** werden die Jobs sinnvoll kategorisiert.

Des Weiteren wurden die Jobbeschreibungen eingekürzt. Eine ausführliche Beschreibung kann weiterhin im Tooltip oder der Hilfe gefunden werden.

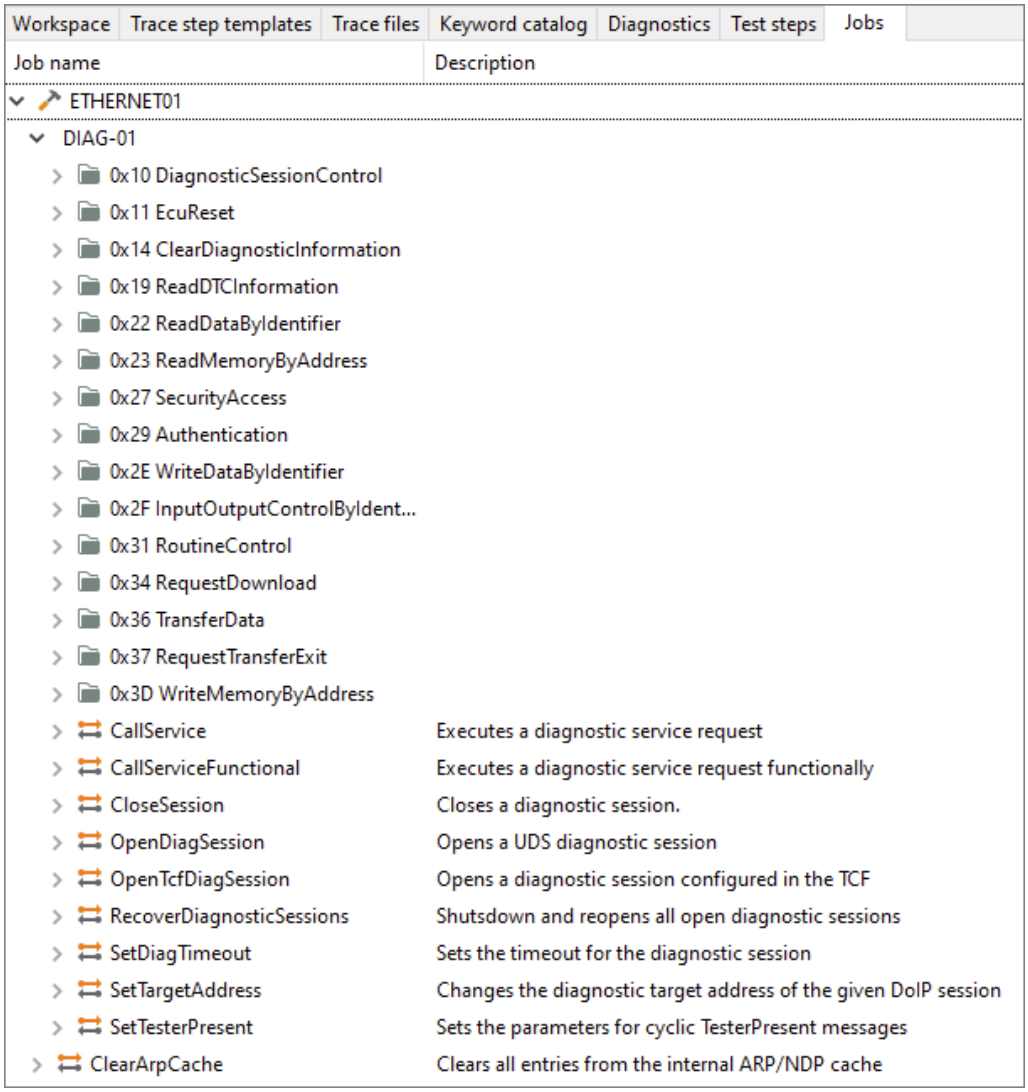


Abbildung 29: Kategorisierung von Jobs für Diagnoseports

3.8 ecu.test *lab*

Verbesserter Edit-Workflow

et

Bisher war der Editiermodus in **ecu.test** *lab* zweistufig aufgebaut. Es gab das Editieren der Größe und Anordnung der Widgets und zusätzlich das Editieren der einzelnen Widgets in einem Editor. Diese beiden Aktionen sind nun zusammengefasst.

Der Edit-Modus wird weiterhin über das Zahnrad in der Titelleiste betreten. Hier können Widgets per Drag and Drop eingefügt, verschoben oder skaliert werden. Sobald ein Widget selektiert wird, öffnet sich dessen Konfiguration auf der rechten Fensterseite. Nun können parallel Änderungen an der Konfiguration des Widgets und am **Arrangement** der Widgets durchgeführt werden.

Bei gültiger Konfiguration des Widgets führt die Selektion eines anderen Widgets zum Öffnen von dessen Konfiguration. Ein Klick auf einen leeren Bereich schließt den Editor. Ist die Konfiguration ungültig, bleibt die Konfiguration geöffnet.

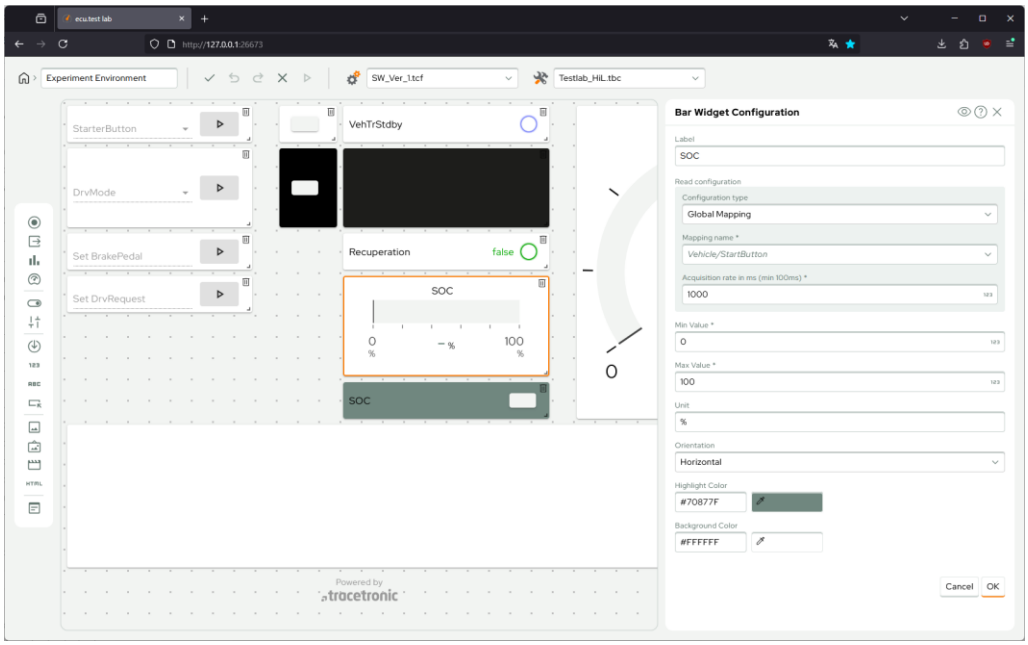


Abbildung 30: Konfiguriertes Dashboard in ecu.test *lab*

Undo/Redo

et

Veränderungen an einem View können nun über Undo/Redo-Buttons in der Titelleiste zurückgenommen und wieder hergestellt werden.

Monitoring-Modus während der Testausführung starten

et

Bisher konnte der Monitoring-Modus in **ecu.test** *lab* nicht aktiviert werden, wenn bereits eine Testausführung lief.
Nun ist dies möglich, insofern der View in lab bereits vor Start der Testausführung geöffnet war oder der View in den Workspace-Einstellungen von lab für den Autostart vorgesehen ist.

Neues „Selection"-Widget

et

Das **Selection**-Widget bietet die Möglichkeit, Textwerte zur Auswahl zu definieren. Diese werden dann in Form eines Dropdowns angezeigt.
Falls die Tool-Schnittstelle nur eine Manipulation mit numerischen Repräsentationen der Werte ermöglicht und das beschriebene Mapping über eine Enum/Vtab/Texttable verfügt, findet eine automatische Umrechnung auf den numerischen Wert statt.

Neues "Call"-Widget

et

Das **Call**-Widget bietet die Möglichkeit als Mapping ein Job-Mapping auszuwählen und diesen auszuführen. Im Gegensatz zu den bisherigen Widgets orientiert sich die Bedienoberfläche des Widgets an der Schnittstelle des Jobs.

In der Konfiguration des Widgets können jene Parameter ausgewählt werden, die bei der Ausführung in der Widget-GUI befüllt werden können/müssen.
Die restlichen Parameter werden in der Konfiguration statisch parametriert und bei der Ausführung automatisch befüllt.

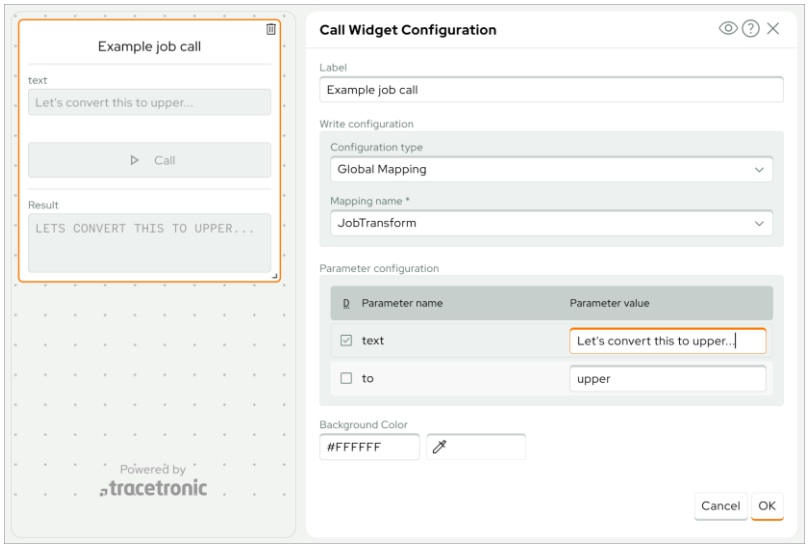


Abbildung 31: „Job Aufrufen"-Widget

3.9 Kommunikation

Neues Simulationsfeature für SOME/IP - Anbieten von Methoden im Testfall

et

Mit dem neu geschaffenen Mechanismus zum Anbieten von Methoden lässt sich nun schnell und einfach eine Service-Simulation aktivieren.

Außerdem profitiert man von den Möglichkeiten des Mappings, Pfade bei Änderung der Systembeschreibungen anpassen zu können. 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.

Aktuell lassen sich nur feste Werte für den Methodenaufruf hinterlegen. Die dynamische Berechnung von Rückgabewerten wird in einer der nächsten Versionen bereitgestellt.

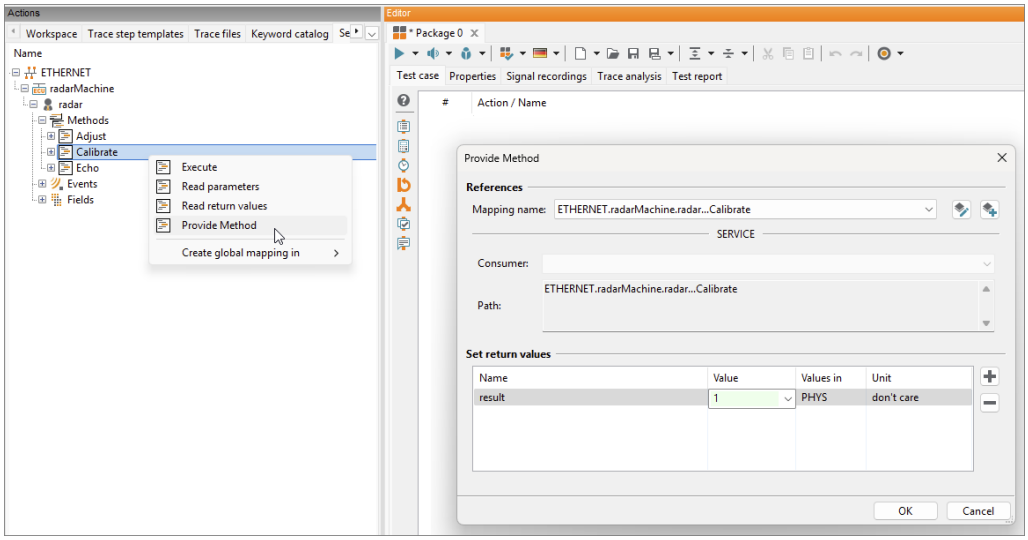


Abbildung 32: Anbieten von Methoden im Testfall

Hardwarenahe Busanbindungen für CAN(-FD): Neuer Job zur Ermittlung der Bus-Ruhe

et

Mit dem neuen Port-Job **GetLastReceptionTimestamp**, auf allen hardwarenahen CAN und CAN-FD Ports, kann der Zeitstempel der zuletzt empfangenen Nachricht abgefragt werden.
Damit kann beispielsweise ermittelt werden, ob und seit wann ein Bus inaktiv ist.

MACsec: Einfachere Konfiguration durch weniger Ports

Das MACsec-Protokoll kann für die kryptographische Absicherung der Ethernet-Bordnetzkommunikation genutzt werden.
Bisher musste für die Absicherung jeder IP-Verbindung jeweils ein Port in der Testbenchkonfiguration angelegt werden.

Neu ist, dass eine ausgehandelte Security Association sich nun auf alle TCP/IP-Stacks mit dem gleichen Device und der gleichen MAC-Adresse auswirkt. Der Zähler für die Packet Number wird gemeinsam genutzt. Die Anzahl notwendiger Ports bei mehreren IP-Verbindungen reduziert sich dadurch.

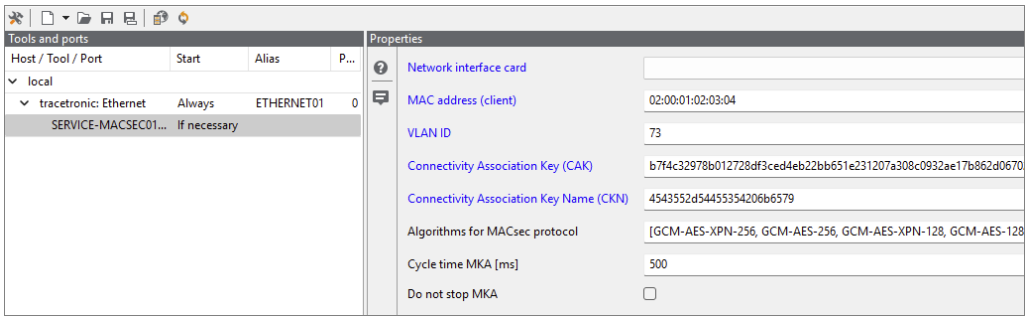


Abbildung 33: Einfachere Konfiguration von MACsec durch weniger Ports

XL-API: Neue Option für CAN-FD zur Abschaltung der Pufferung

Falls ein CAN-FD Bus inaktiv wird, kann es vorkommen, dass noch Nachrichten im Sendepuffer der Vector-Hardware verbleiben. Diese Nachrichten werden von der Hardware automatisch bei Reaktivierung des Busses versendet. Dieses Verhalten kann unerwünscht sein, da die Daten möglicherweise veraltet sind.
Mit der neuen TBC-Option **Sendepuffer leeren**, dass von der XL-API nur für CAN-FD bereitgestellt wird, kann die Nutzung des Sendepuffers unterbunden werden.

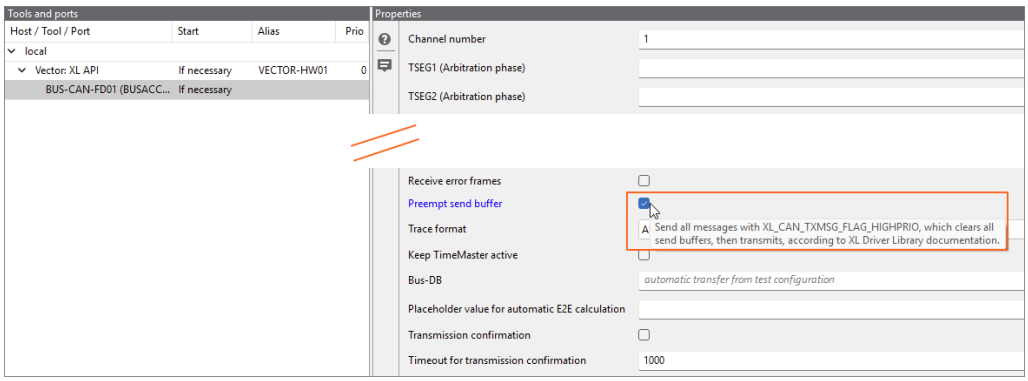


Abbildung 34: Neue Option zu Abschaltung der Pufferung von CAN-FD

XL-API: Neuer Job *GetLinkState*

et

Mit dem neuen Port-Job, der auf dem Ethernet-RAW Port zur Verfügung steht, ist es möglich zu prüfen, ob der Medien-Verbindungsstatus des XL-Netzwerkadapters aktiv oder inaktiv ist.

3.10 Traceanalyse

MDF4: Umgang mit verdrehten Matrizen bei XIL-Aufnahmen

et

Bei der Interpretation von mehrdimensionalen Größen kommt es unter Umständen vor, dass **ecu.test** die Reihenfolge der Dimensionen in umgekehrter Reihenfolge als das über XIL angebundene Tool interpretiert. Damit man dennoch einheitlich per Index auf die gewünschten Elemente zugreifen kann, wurde bereits in vergangenen Versionen eine TBC-Option zum Transponieren von Matrizen hinzugefügt. Bisher griff diese Option jedoch nur bei Testschritten.

Mit **ecu.test 2025.3** wird die TBC-Option zum Transponieren von Matrizen nun auch für aufgezeichnete Matrizen berücksichtigt. Wichtig ist, dass die Aufnahme "live" während der Testausführung über den Tool-Adapter aufgezeichnet. Andernfalls findet diese Option beim Einlesen von Signalen für Traceanalysen keine Verwendung.

MDF4: Verbesserte Performance beim Einlesen

et tc

Der MDF4-Parser wurde insbesondere für Aufnahmen mit Kompression optimiert, sodass Signale um bis zu Faktor 1,7 schneller eingelesen werden können. Auch das Schreiben wurde durch diese Optimierung beschleunigt.
In vielen Tools werden komprimierte MDF4-Aufnahmen geschrieben, sobald MDF 4.1 oder höher gewählt ist.

3.11 trace.xplorer

Verlinkung zu Testfallergebnissen aus test.guide-Kennzahlenexport

et tc

test.guide bietet seit 2023 die Möglichkeit, Testfall-Kennzahlen in eine ASTRACE-Datei zu exportieren, um KPI-Daten von vielen Testläufen im **trace.xplorer** performant visualisieren und im Detail analysieren zu können.

Wollte man zu einem interessanten Datenpunkt die Details des zugehörigen Testlaufs aufrufen, musste dieser durch manuellen Vergleich des Zeitstempels in **test.guide** identifiziert werden. Dieses Vorgehen war umständlich und fehleranfällig.

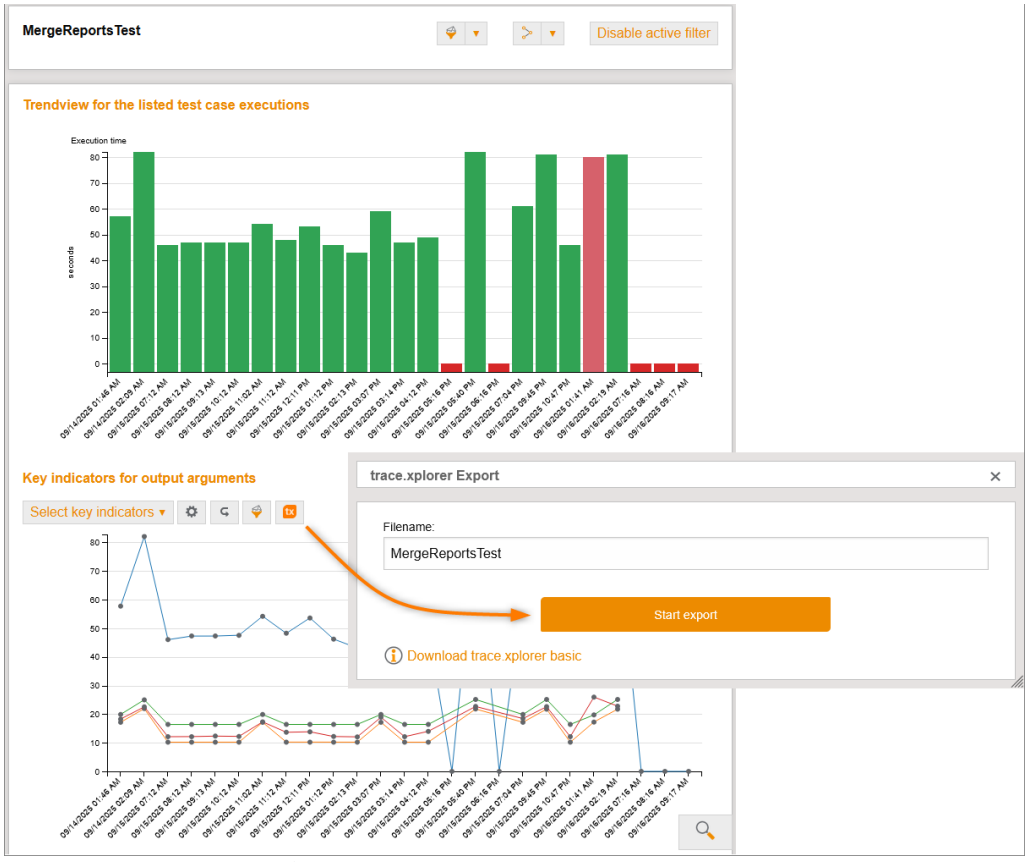


Abbildung 35: Export von Testfall-Kennzahlen aus test.guide in eine trace.xplorer-Datei

Ab sofort enthalten die Kennzahlenexporte pro Testfall ein neues URL-Signal **Link to test case execution**, das für jeden Testlauf einen Link auf dessen Ergebnisseite in **test.guide** speichert.

Um diese Seite aufzurufen, bietet der **trace.xplorer** für URL-Signale den neuen Menüpunkt **Open URL in Browser** im Kontextmenü der Tabellenansicht. Dieser führt ohne lange Suche direkt zu den gewünschten Details des Testlaufs.

Damit das funktioniert, muss die **test.guide**-Instanz, mit der das ASTRACE-Dokument erzeugt wurde, erreichbar sein. Der Anzeigename der gespeicherten Links benennt die ID des Testlaufs (TCE ... Test Case Execution) und kann zum zusätzlichen Abgleich der Information zwischen **trace.xplorer** und **test.guide** dienen.

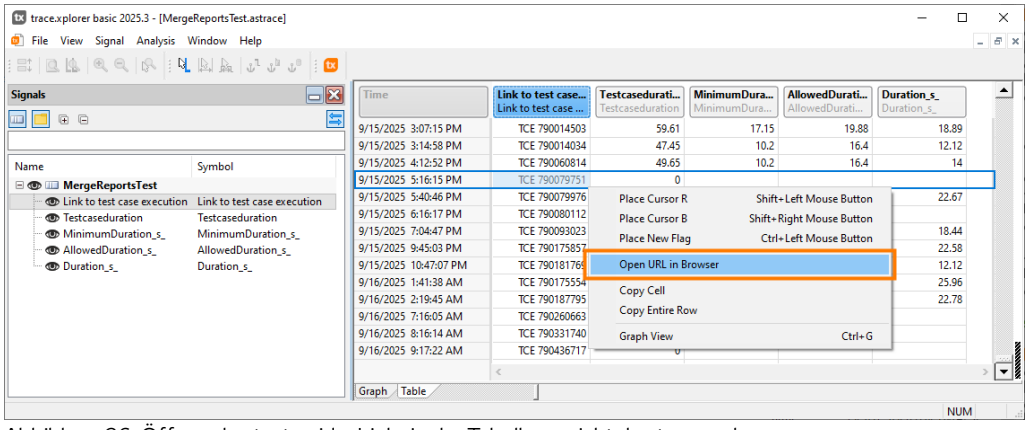


Abbildung 36: Öffnen des test.guide-Links in der Tabellenansicht des trace.xplorers

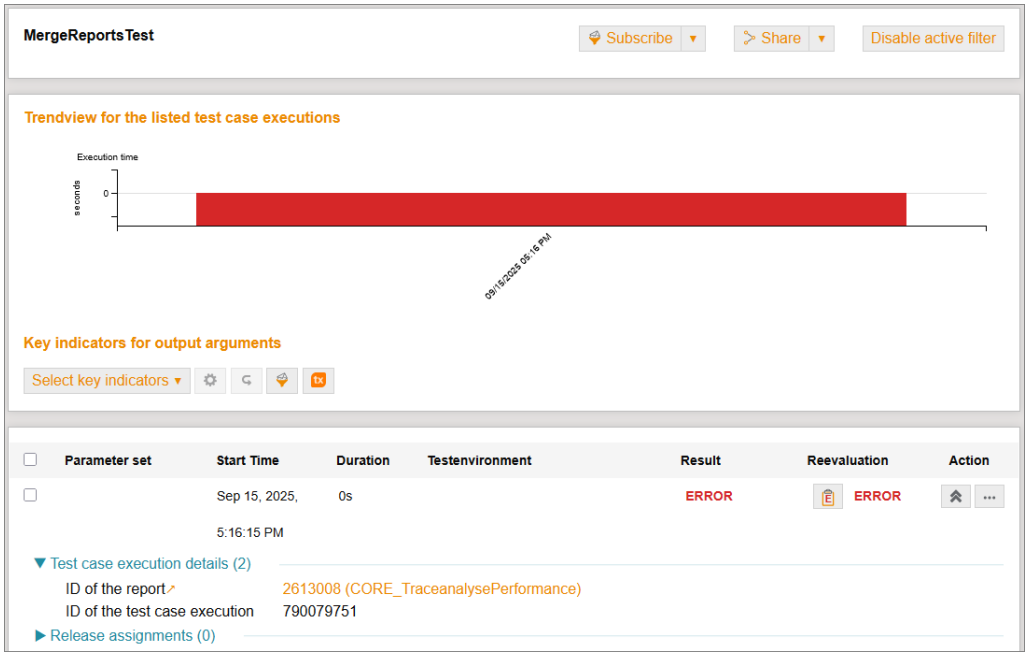


Abbildung 37: Direkter Sprung vom trace.xplorer zu den Ergebnisdetails des Testlaufs in test.guide

Socket-Adaptor-Signaldaten aus PCAP-Dateien importieren

et tc

Mitschnitte einer Socket-Adaptor-Kommunikation landen in PCAP-Dateien. Der **trace.xplorer**-Importfilter für PCAP-Dateien konnte bisher nur SOME/IP-Signaldaten importieren und fragte ausschließlich den **Service**-Namensraum ab. Socket-Adaptor-Kommunikation überträgt jedoch CAN- oder FlexRay-Daten und nutzt dafür den **Buszugriff**-Namensraum. Diesen unterstützt der Importfilter jetzt auch, sodass Socket-Adaptor-Signaldaten in ASTRACE-Dokumente importiert werden können.

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

Alternative Zahlendarstellungen für Ganzzahl-Datentypen

et tc

Alle Ganzzahl-Datentypen ließen sich bisher dezimal und hexadezimal visualisieren. Verbergen sich hinter einem Integer-Signal in Wahrheit aber einzelne Flags, wäre es sehr hilfreich gewesen, die Werte zum Beispiel in Binärdarstellung zu sehen. Genau das ist jetzt möglich.

Die Formatierung der Textausgabe von Ganzzahl-Signalwerten lässt sich im Andockfenster **Eigenschaften** pro Signal individuell konfigurieren.
Es kann das Zahlensystem **(Binär/Dezimal/Hexadezimal/Oktal)** gewählt werden und, ob das zugehörige Präfix angezeigt werden soll. Zur besseren Lesbarkeit lassen sich Ziffern gruppieren.

Die Formatierung beeinflusst die Tabellenansicht, das Andockfenster **Messen** und den CSV-Export.

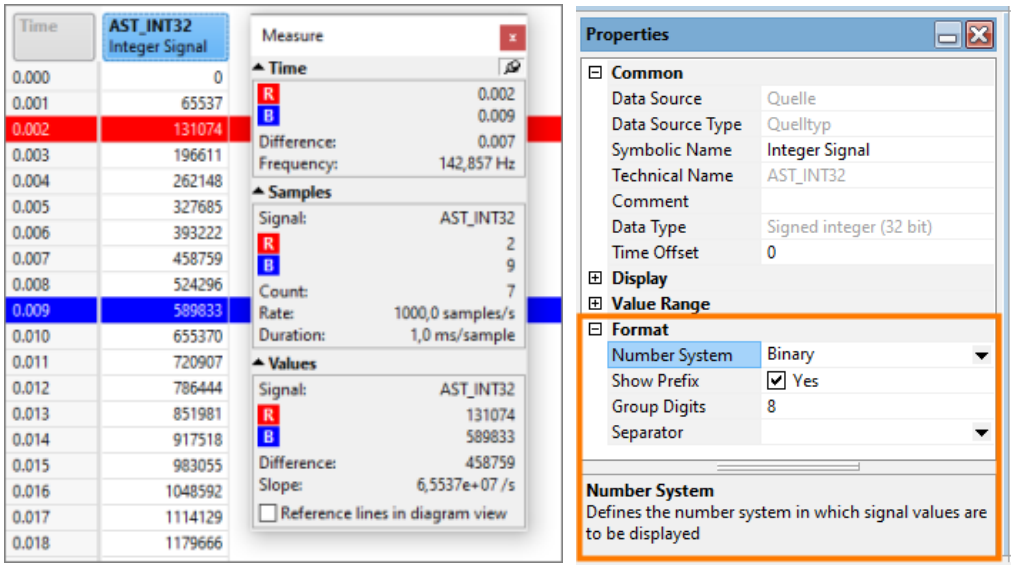


Abbildung 38: Individuelle Formatierung der Textausgabe von Ganzzahl-Signalen festlegen

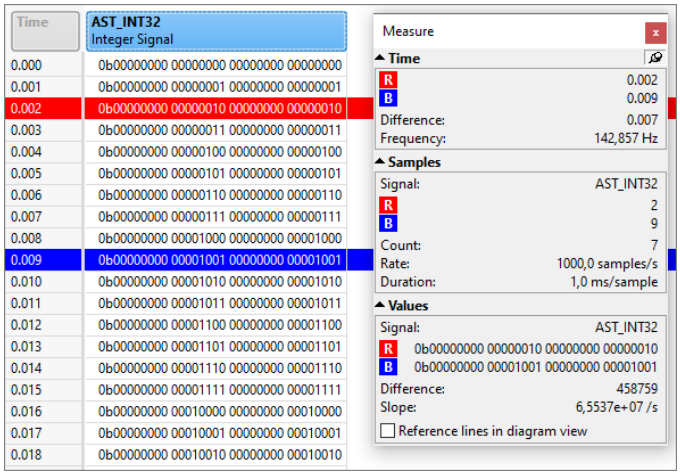


Abbildung 39: Textausgabe von Ganzzahl-Signalen im Binärformat mit Präfix

Kleinere Verbesserungen der Usability

et tc

Neue Option in den Workspace-Einstellungen

Die Auswahl des gewünschten Signalbetrachters erfolgt jetzt – wie auch die Lizenzierung (des **trace.xplorers**) - pro Arbeitsplatz und nicht pro Workspace.

Deswegen wurden die betreffenden Einstellungen umgezogen:

- Die bisherige Kategorie **Signalvisualisierung** existiert nicht mehr.
- Die Wahl des Standard-Signalbetrachters, **trace.xplorer** oder **trace.xplorer** *basic*, ist eine lokale Einstellung in der neuen Sektion **Mitgelieferte Programme**.

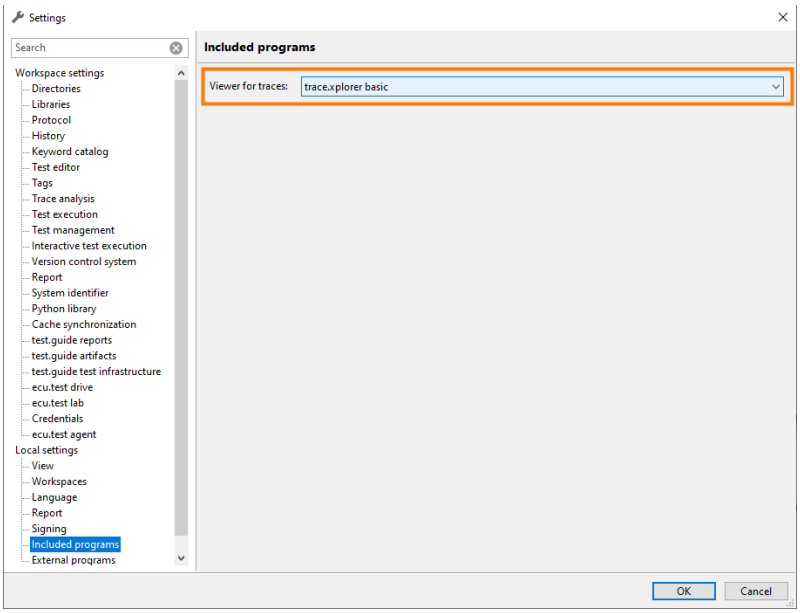


Abbildung 40: Einstellungen für den Standard-Signalbetrachter

- Alle übrigen Optionen der ehemaligen Kategorie **Signalvisualisierung** wurden auf die Seite **Traceanalyse** verschoben.

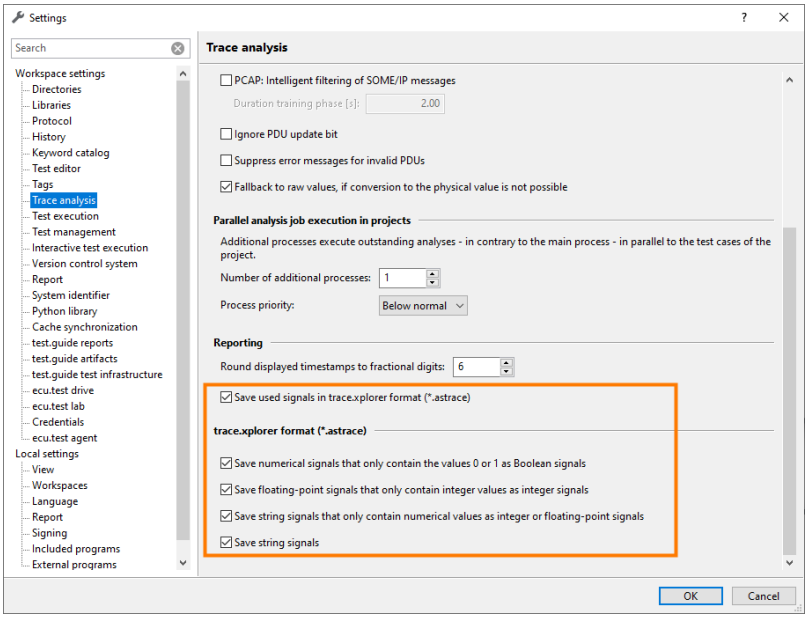


Abbildung 41: Einstellung für das trace.xplorer-Dateiformat

trace.xplorer im Vordergrund anzeigen

Bei der Arbeit mit vielen, gleichzeitig geöffneten Fenstern wird der **trace.xplorer** schnell in den Hintergrund verdrängt.

Wer das Tool gern im Blick behalten möchte, findet jetzt im Systemmenü des Anwendungsfensters einen zusätzlichen Befehl, um das Fenster permanent im Vordergrund anzuzeigen.
Das Menü lässt sich per Rechtsklick auf die Titelleiste aufrufen.

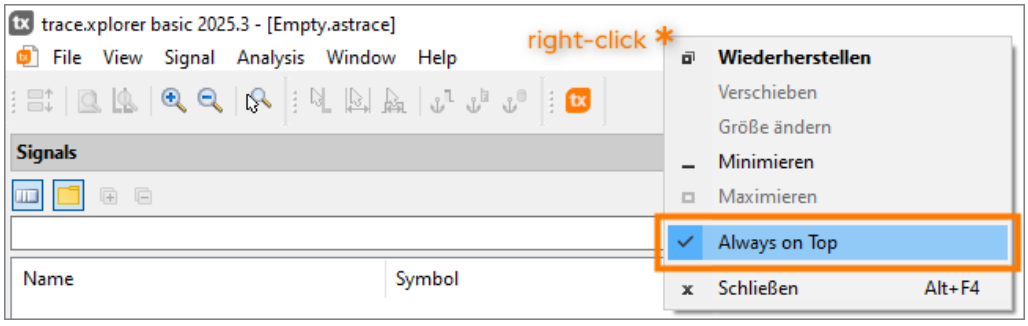


Abbildung 42: trace.xplorer-Anwendungsfenster immer im Vordergrund anzeigen

Datenquellen und Signalgruppen ein- und aufklappen
ASTRACE-Dokumente mit vielen Datenquellen oder tief verschachtelten Signalgruppen waren bisher schwer zu handhaben, weil im Andockfenster **Signale** keine Möglichkeit existierte, ganze Zweige mit wenigen Handgriffen ein- oder aufzuklappen. Stattdessen war man gezwungen, viele einzelne Klicks zu machen.
Solche Dokumente entstehen zum Beispiel beim Kennzahlenexport aus **test.guide** oder beim Importieren von Ethernet-Signalen.

Nun lassen sich Einträge für Datenquellen und Signalgruppen ganz einfach über ihr Kontextmenü oder die neuen Funktionen in der Symbolleiste vollständig ein- und aufklappen.

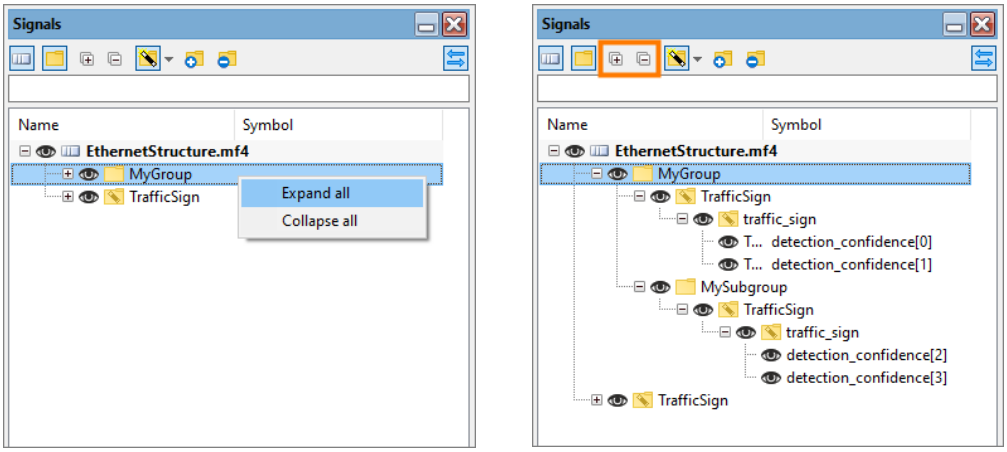


Abbildung 43: Datenquellen und Signalgruppen vollständig ein- und aufklappen über Kontextmenü (links) oder Symbolleiste (rechts)

4 Versionen und Schnittstellen

4.1 Neue Tools und Versionen et

	Provider	Webseite	System	Produktname	Version
1	**dSPACE**	Release-Link	Software für die integrierte Steuergeräteentwicklung	**ControlDesk**	**2025-A**
2	**MOTEON**	Link	Tests und Messungen von eingebetteten Systemen	**Motor Test Bench**	
3	**Synopsys**	Link	Software-in-the-Loop Solution für virtuelle ECUs	**Silver**	**W-2025.09**

4.2 APIs

4.2.1 Internal API

ToolAccess API zum Ausführen von Jobs

et

Um einen Job aufzurufen, war es bis dato nötig im Package ein Mapping anzulegen. Ein generischer Aufruf eines Jobs auf einen beliebigen Tool/Port ist damit nicht oder nur umständlich möglich, da man vor Testfallstart alle Mappings anlegen muss.

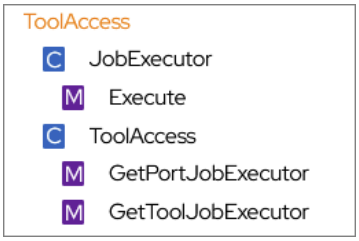


Abbildung 44: ToolAccess API

Mit der ToolAccess API gibt es nun die Möglichkeit, alle Jobs eines Tools beziehungsweise Ports direkt über die API aufzurufen.

Das Anlegen eines Mappings ist nicht mehr nötig.

Hinweis: Die ToolAccess API ist Teil der Internal API.

4.2.2 REST API

DELETE Endpoint bricht Postcondition nicht mehr ab

Wird der REST API Endpoint DELETE während der Ausführung einer Postcondition aufgerufen, werden die laufenden Aktionen der Postcondition nicht abgebrochen. Beim manuellen oder automatisierten Abbruch eines Playbooks in **test.guide** wird dadurch gewährleistet, dass der Prüfstand bzw. die Umgebung via der Postconditions in einen sauberen Zustand zurückgesetzt wird.

4.2.3 UserTool

Differenzierung zwischen internem Eigenschaftsnamen und Anzeigenamen

Bisher haben wir einen Bezeichner für eine Eigenschaft verwendet, der in der Implementierung, aber auch für die Anzeige in der TBC verwendet wurde. Dies schränkte den Gestaltungsspielraum für die Nutzerführung im Editor deutlich ein.

Diese Unterscheidung ist nun durch Definition eines expliziten **DisplayName** möglich.

5 Abkündigungen

5.1 Abkündigungen und Inkompatibilitäten in dieser Version

Überarbeitete Terminologie

et tc

Um die Benutzererfahrung intuitiver zu gestalten, wurden etliche etablierte Begriffe geändert. Die neuen Bezeichnungen sorgen für mehr Klarheit, eine einheitliche Sprache und erleichtern die Orientierung in der gesamten Toolandschaft.

Die Änderungen betreffen sowohl GUI-Elemente in **ecu.test** selbst, aber auch Dateinamen, die Anwenderhilfe und API-Namen.

	Alte Bezeichnung	Neue Bezeichnung
Tool	Diff-Viewer Lizenzmanager Signal-Editor Signal-Viewer TRF-Viewer/Report-Viewer Tool-Server ecu.test-Runner	ecu.test Diff Viewer 2025.3 ecu.test License Manager 2025.3 ecu.test Signal Editor 2025.3 ecu.test Signal Viewer 2025.3 ecu.test Report Viewer 2025.3 ecu.test Tool Server 2025.3 ecu.test Runner 2025.3
Datei-name	Tool-Server.exe tool-server tool-server_runner Tool-Server_out.log LicenseManager.exe TRF-Viewer.exe	tool_server.exe (W) tool_server (L) tool_server_runner (L) tool_server_out.log (W+L) license_manager.exe (W) report_viewer.exe (W)
API-Name	Project parameter generator API Project package generator API Project generator API Model based bus port API Advanced properties of bus related objects Advanced properties of diagnostics related objects Advanced properties of media related objects Advanced properties of DLT logging related objects	Parameter Set Generator API Package Generator API Project Generator API Model-based Bus Port API Advanced properties of bus-related objects Advanced properties of diagnostics-related objects Advanced properties of media-related objects Advanced properties of DLT logging-related objects

*W=Windows, *L= Linux

Hinweis: Die Versionsangabe ist abhängig vom Release. Nicht erwähnte Anteile behalten ihren Namen (z.B. "**ecu.test.exe**" oder andere APIs).

"Alternative call representation" der Package-Eigenschaften

et

Die **Alternative call representation** in den allgemeinen Eigenschaften ist mit **ecu.test 2025.3** entfernt.

Das Feature wurde im Rahmen des schlüsselwortbasierten Testens im Mapping von Referenzpackage-Aufrufen implementiert und findet ab sofort dort seine Anwendung.

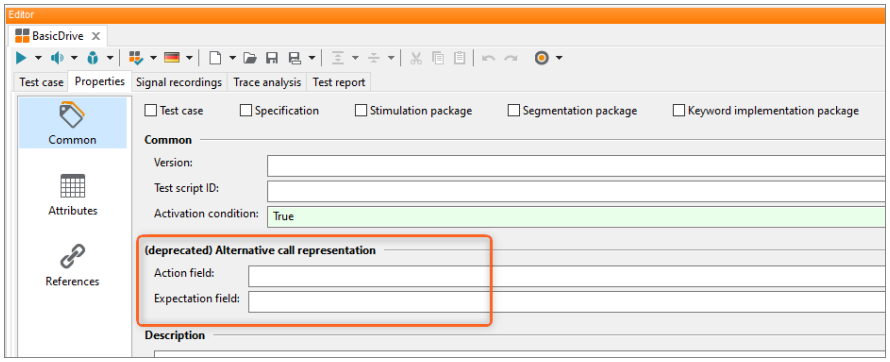


Abbildung 45: Package-Eigenschaften: Alternative call representation

ODX Migration Helper

et

Mit der Version 2023.4 wurden für die Verwendung von ODX/PDX Datenbasen generische Diagnosetestschritte eingeführt. Diese stellen alle Services der ODX/PDX bereit.
Um den Umstieg zu vereinfachen wurde eine Migrationshilfe zum Umstellen der bestehenden Packages auf die neuen Testschritte angeboten. Diese wird mit **ecu.test 2025.3** entfernt.

DiagBrowser: GetUdsName wird durch GetServiceName ersetzt

et

Die Methode zum Abfragen des Services im DiagBrowser **GetUdsName** wurde endgültig entfernt.
Als Alternative steht die folgende Methode zur Verfügung:

- **GetServiceName**

5.2 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.

Fibex-Unterstützung

et

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 mit **ecu.test 2026.2** entfernt. Als Ersatz kann die neue Python-basiere Anbindung genutzt werden.

Abkündigung ReqIf Import

et

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

Alternatives Reportverzeichnis bei separater Unterprojektausführung

et

Bei separater Unterprojektausführung besteht aktuell die Möglichkeit, den Reportordner angeben zu können. Dieses Feature ist zur 2025.3 als depricated 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.

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().