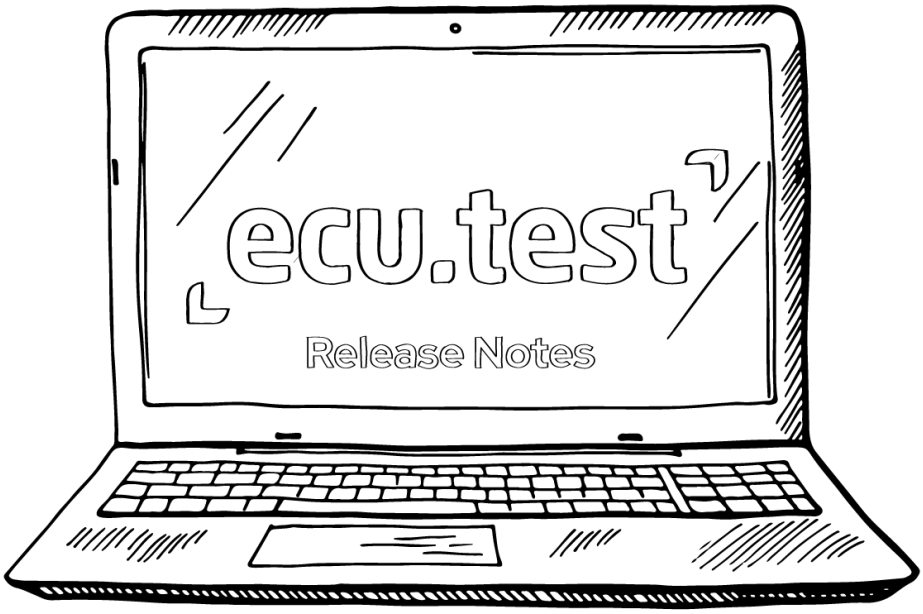
tracetronic



Release Notes

ecu.test 2026.2

trace.check 2026.2

Datum: 09.06.2026
© 2026 tracetronic GmbH

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

tracetronic

Inhalt

tracetronic

Überblick

Mit dem **ecu.test**-Release **2026.2** wurden die KI-Funktionen weiter ausgebaut, zahlreiche **ecu.test** Extras erweitert und an vielen Stellen die tägliche Arbeit spürbar vereinfacht.

ecu.test *agent*

- Durchgängiger Workflow von der Spezifikation bis zur Traceanalyse
- Agentische Multimedia-Testschritte
- Intelligenter Copilot für **ecu.test** (Preview)

ecu.test Extras

- ***calibration***: prüft XCP-Verbindungen automatisch.
- ***code***: unterstützt native Python-Datentypen und Konstanten.
- ***diagnostics***: bietet zahlreiche neue API-Methoden im DiagBrowser.
- ***drive***: läuft nun auch unter Linux.
- ***lab***: überzeugt mit neuen Widget-Funktionen.

ecu.test-Konfiguration

- **Offline-Modus**: hält HiL und andere Testressourcen frei und ermöglicht parallele Entwicklung.
- **Selektiver Toolneustart:** startet bei TBC-Änderungen nur die betroffenen Tools neu.

Performance im Alltag

- **Linux GUI** mit grafischem Lizenzmanager und Tool-Server.
- **Bibliotheksworkspaces** unterstützen Templates, Reportgeneratoren und zeigen den aktiven Git-Branch.
- **Modelfilter & Aliase** reduzieren Modelbäume aufs Wesentliche.
- **Repeat-Until-SUCCESS** als neuer Testschritt für robuste Wiederholungen.
- **EasyInsert** findet auch AUTOSAR-Service-Methoden und -Events.
- **Projektreport-Workflows** mit PRFZ-Export und Offline-Verfügbarkeit.

Außerdem gibt es Neuerungen für HiL, Toolanbindungen, Testmanagement oder die Traceanalyse. Details folgen auf den nächsten Seiten. Viel Spaß beim Weiterlesen!

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

1 Highlights in ecu.test 2026.2

1.1 ecu.test *agent*

Durchgängiger Workflow von Spezifikation zu Testfall und Traceanalyse

Mit dem aktuellen Release vervollständigt der **ecu.test** *agent* die durchgängige, KI-gestützte Erstellung von Testfällen.

Ausgehend von einer Testfall- und Traceanalyse-Spezifikation übernimmt der **ecu.test** *agent* nun die nahtlose Generierung aller Implementierungsschritte – von der Testfallgenerierung über das Recording bis hin zur Traceanalyse.

Was bislang noch manuellen Aufwand erforderte, lässt sich damit vollständig Agent-basiert abdecken.

Die Aufnahmesteuerung wird dabei durch die Testschritte **Start/Stop trace recording** unterstützt. Bei der Signalgenerierung greift der **ecu.test** *agent* auf die bewährten Mechanismen zur automatischen Signalgruppenzuordnung zurück.

Die Aufnahme- und Signalgenerierung, die in vorherigen Versionen noch nicht verfügbar war, ist damit nun fester Bestandteil dieser durchgängigen Lösung.

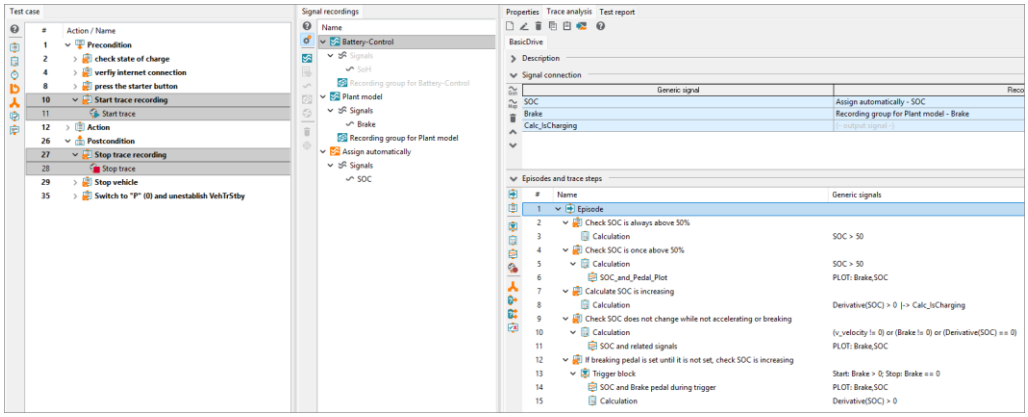


Abbildung 1: Von Testfall bis zur Traceanalyse – ein durchgängiger Workflow

Agentische Multimedia-Testschritte

Mit dem Testschritt **TouchInput** bietet **ecu.test** bereits die Möglichkeit, verschiedene Aktionen (Klicken, Swipen, Rotieren usw.) zum Interagieren mit einer HMI oder einer Webseite durchzuführen. Bisher mussten dazu jedoch feste Koordinaten vorgegeben werden.
Mit dem Bild-Lesen-Testschritt können Erwartungen an den Inhalt von Bildern gestellt werden. Diese müssen allerdings 100% mit der Referenz übereinstimmen.

Insbesondere beim Testen von HMI-Elementen ist dieses Vorgehen nicht sonderlich robust. Wenn Positionen und Größen der Elemente sich während des Entwicklungsprozesses ändern, müssen die manuell erstellten Testfälle mühsam angepasst werden.

Mit der neuen KI-Funktion in den **TouchInput**-Testschritten, kann statt der Koordinaten einfach ein Prompt mitgegeben werden. Für das Testen einer HMI sind beispielsweise folgende Prompts denkbar:

- Drücke den Button „Radio“.
- Ich möchte telefonieren.
- Ich möchte Musik hören.

Mit der neuen Bild-Lesen Funktion können komplexe Erwartungen gestellt werden. Referenzbilder können ebenfalls dem Model übergeben werden:

- Ein blaues WLAN-Symbol ist in der oberen rechten Ecke zu sehen.
- Das Referenzbild taucht zweimal auf.
- Die Sprache im Bild ist chinesisch.

Das Model analysiert das Bild und gibt die Koordinaten zurück - egal welche Sprache eingestellt ist, wie die Anordnung der Elemente ist oder was auf ihnen abgebildet ist. Damit lassen sich sehr robuste Testschritte erstellen.

Das Model wird über den **ecu.test** *agent* connector angebunden und kann auf die individuellen Bedürfnisse zugeschnitten werden. Wir haben bereits gute Erfahrungen mit Claude 4.6 und ChatGPT gemacht.

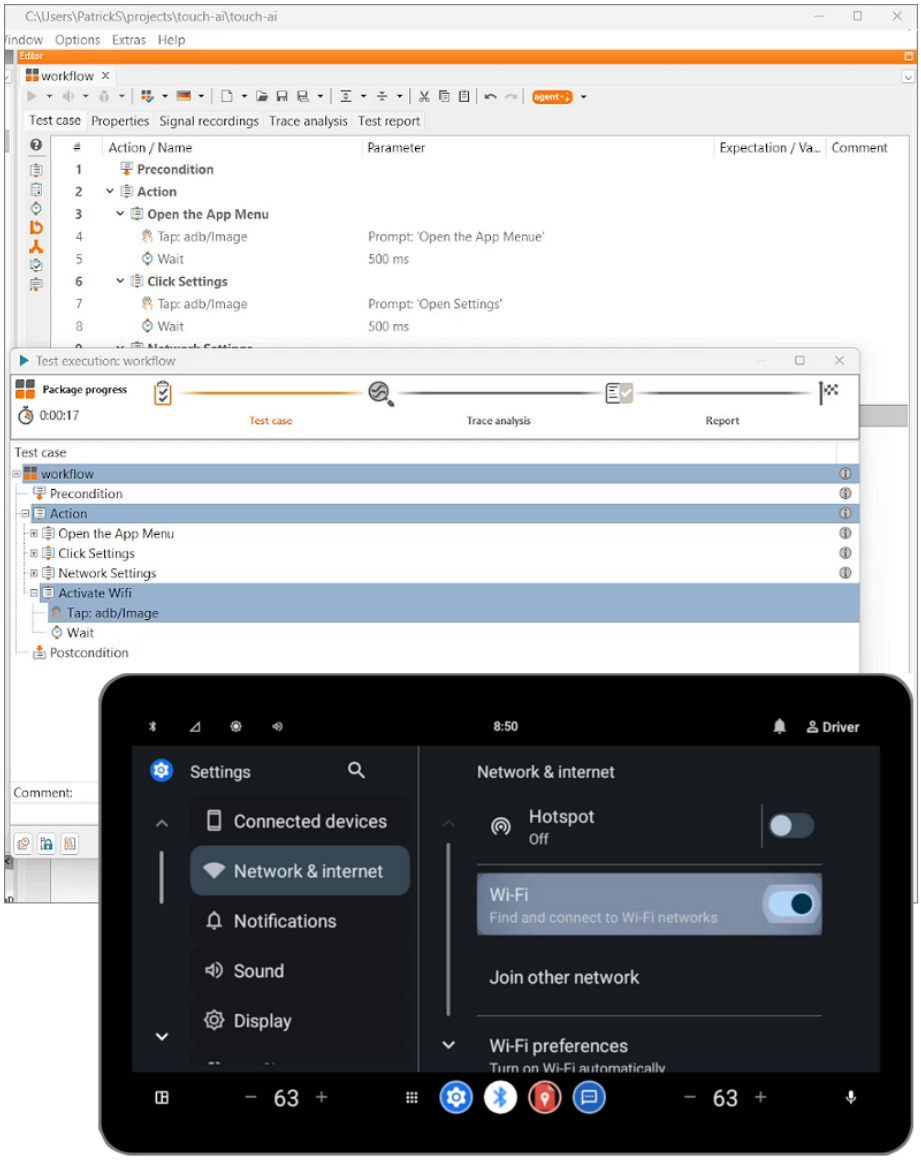


Abbildung 2: TouchInput-Testschritt mit KI-Prompt

1.2 ecu.test-Konfiguration

Selektiver Toolneustart bei geänderter Testbench-Parametrierung

Der Start einer **ecu.test**-Konfiguration kann je nach eingesetzten Tools und der Größe der zu ladenden Datenbasen unter Umständen einige Zeit dauern. Bisher war so, dass sobald eine Einstellung einer bereits gestarteten Testbenchkonfiguration (TBC) geändert wurde, die gesamte Konfiguration erneut gestartet werden musste.

Ab dieser Version wird beim Speichern einer TBC gefragt, ob nur die geänderten Tools neu gestartet werden sollen. Dies kann gerade bei der Einrichtung oder Änderung eines Prüfplatzes erforderlich sein und zu signifikanter Zeitersparnis führen.

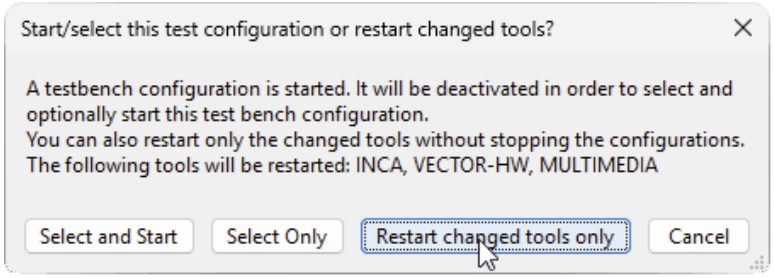


Abbildung 3: Nur geänderte Tools neu starten

Starten einer Konfiguration im Offline-Modus

et tc

Die Testfallerstellung in großen Teams wurde verbessert, indem **ecu.test** befähigt wurde, die wertvollen Testressourcen so effizient wie möglich zu verwenden. Damit kann mehr Zeit für die Ausführung von Tests verwendet werden und parallel an Tests entwickelt werden.

Damit ein Testfall ausgeführt werden kann, muss die Konfiguration gestartet werden. Dieser Start blockierte bislang die dahinter liegende Ressource (z. B. den HiL). Weitere Anwendende sowie der **ResourceAdapter** von **test.guide** konnten in dieser Zeit nicht mit dem HiL interagieren, um z. B. Tests auszuführen.

Mit dem neuen Offline-Modus kann nun die Konfiguration gestartet werden, ohne die Testressource zu blockieren.

Achtung: Einige Funktionen sind im Offline-Modus nicht vorhanden oder nur eingeschränkt nutzbar. Es werden zum Beispiel keine Online-Anfragen getätigt. Alle Einschränkungen können der Hilfe entnommen werden.

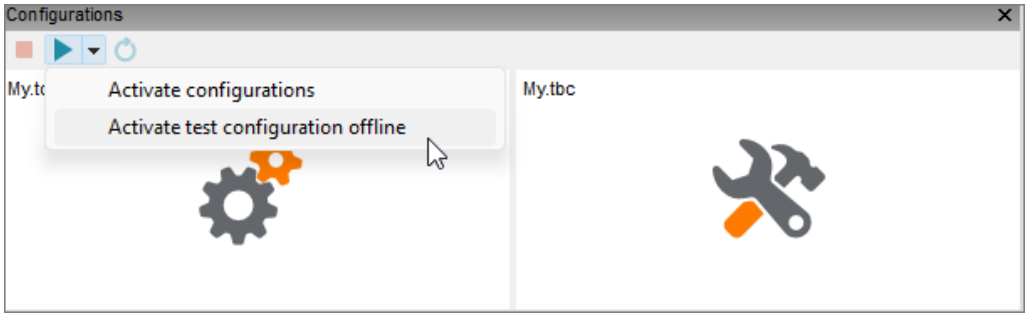


Abbildung 4: Testkonfiguration offline aktivieren

2 Sneak Preview

ecu.test *agent* Chat

In den letzten Releases haben wir den **ecu.*test*** *agent* kontinuierlich weiterentwickelt und ausgebaut. Parallel arbeiten wir bereits am nächsten großen Highlight: dem **ecu.*test*** *agent* Chat – einem intelligenten Copiloten, der künftig in den verschiedensten Workflows unterstützend zur Seite steht.

Der **ecu.*test*** *agent* Chat ist bereits im Release 2026.2 auf Anfrage erlebbar. Meldet euch dazu einfach per E-Mail an support@tracetronic.com. Wir freuen uns auf den Austausch!

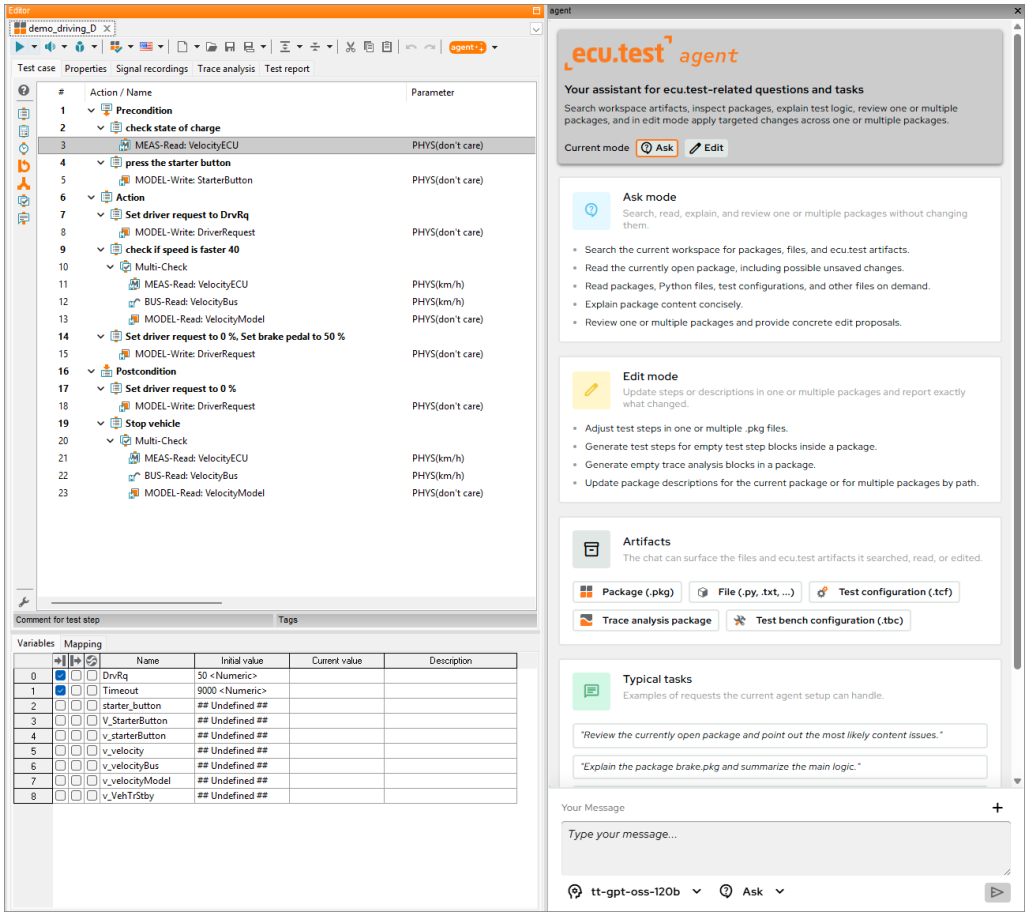


Abbildung 5: ecu.test agent chat

Artefaktkompatibilität zwischen verschiedenen ecu.test-Versionen

Das Aktualisieren von **ecu.test** und Workspaces wird künftig deutlich reibungsloser werden. Ziel ist es, dass verschiedene **ecu.test-**Versionen parallel betrieben werden können, ohne sich gegenseitig zu beeinträchtigen, insbesondere in

Umgebungen, in denen mehrere Teams einen gemeinsamen Workspace, z. B. Bibliotheksworkspaces, nutzen.

Ein zentrales Problem heute: Sobald ein Team ein Artefakt mit einer neueren Version eincheckt, können andere Teams, die noch eine ältere Version verwenden, damit nicht mehr arbeiten. Das erzeugt Druck, Updates stets gleichzeitig und organisationsübergreifend auszurollen, was in der Praxis kaum realistisch ist und zu selteneren Updates führt.

Ab 2026.3 wird es eine umfassende Prüfung beim Speichern von Artefakten geben, die auf Basis einer im Workspace eingestellten **ecu.test**-Zielversion arbeitet. Das verhindert, versehentlich Artefakte inkompatibel für ältere **ecu.test**-Versionen zu machen.
Die kompatible **ecu.test**-Version wird im Workspace-Explorer angezeigt.

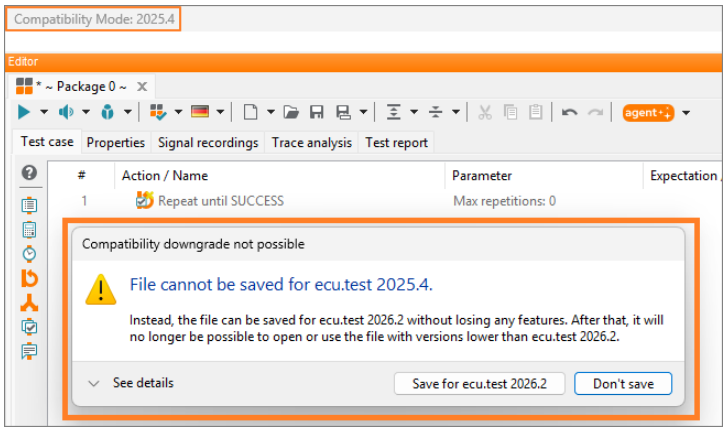


Abbildung 6: Kompatibiltätsmodus prüft, ob der Testfall in älteren ecu.test-Versionen kompatibel bleibt.

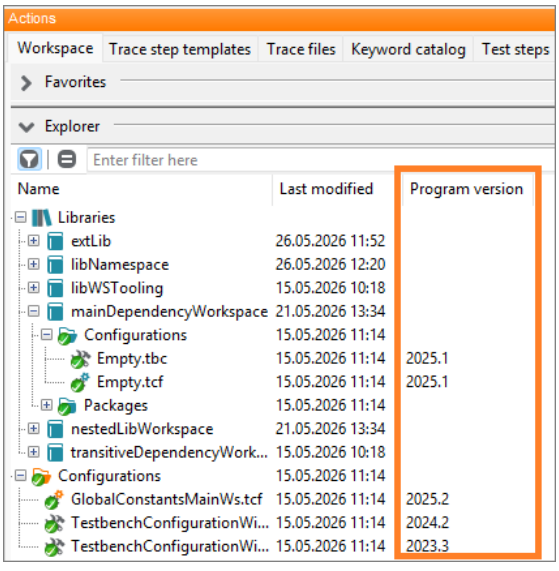


Abbildung 7: Anzeige der kompatiblen ecu.test-Version zu einem Artefakt im Workspace-Explorer

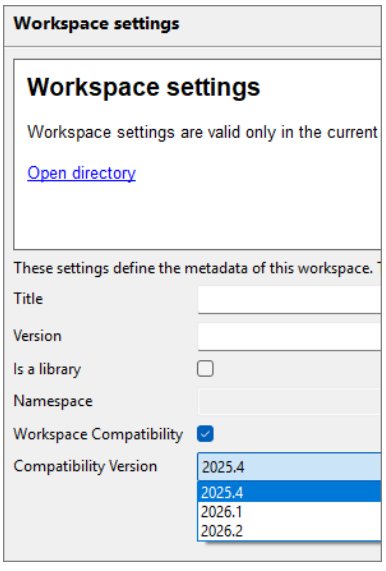


Abbildung 8: Konfiguration des Kompatibilitätsmodus in den Einstellungen

Ausführung lokaler Packages über test.guide

et tc

Demnächst können lokal bearbeitete Packages über die **test.guide**-Ausführungsverteilung direkt getestet werden. Daraus ergeben sich folgende Vorteile:

- **Git-basierter** Workflow
- **Lokale Bearbeitung ohne Prüfstand** mit dem neu eingeführten Offline-Modus der Testkonfiguration (TCF)
- **Flexible Ausführung**, die Prüfstände nicht unnötig blockiert
- **Zeitnahe Rückmeldung** zum Testfallergebnis während man bereits weiterarbeiten kann

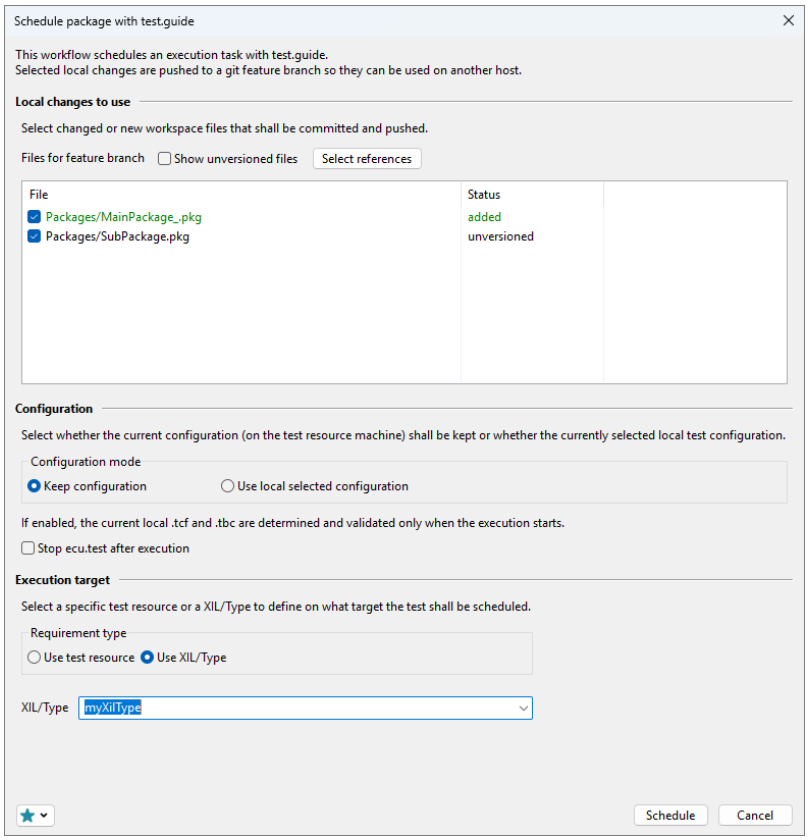


Abbildung 9: Ausführung lokaler Packages über test.guide

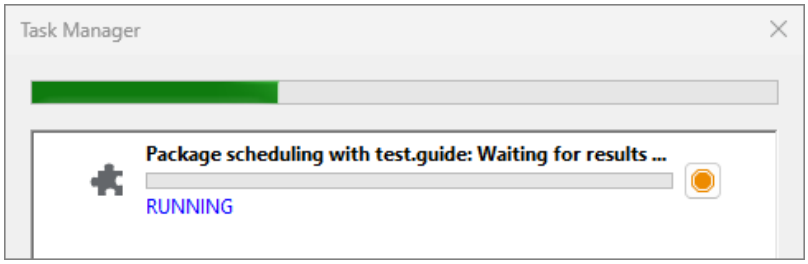


Abbildung 10: Fortschrittsanzeige bis zur Fertigstellung des Ergebnisses

3 ecu.test Extras

3.1 ecu.test *agent*

Globales Mapping für Signale der Traceanalyse

Der **ecu.test** *agent* berücksichtigt nun auch Signale des globalen Mappings. Er fügt relevanten Mappings Signalgruppen hinzu und erstellt die generischen Signale der Traceanalyse. Ein wesentlicher Schritt zu vollständig ausführbaren Packages mit Traceanalysen.

3.2 ecu.test *calibration*

Automatische Verbindungsprüfung und Reconnect für XCP-Slave Zugriff

Ab sofort steht eine neue Option zur Verfügung, mit der die Verbindung zum **XCP-Slave** vor jedem Lese- bzw. Schreibzugriff automatisch geprüft wird. Ist die Option aktiviert, überprüft das System vor jeder Lese-/Schreiboperation den Verbindungsstatus zum **XCP-Slave**. Wird erkannt, dass die Verbindung nicht mehr besteht, erfolgt automatisch ein Reconnect-Versuch. Ist dieser erfolgreich:

- werden alle Puffer geleert.
- wird die zuletzt verwendete Kalibrierseite wiederhergestellt.
- werden Messungen und Aufzeichnungen fortgesetzt.
- und anschließend der eigentliche Testschritt ausgeführt.

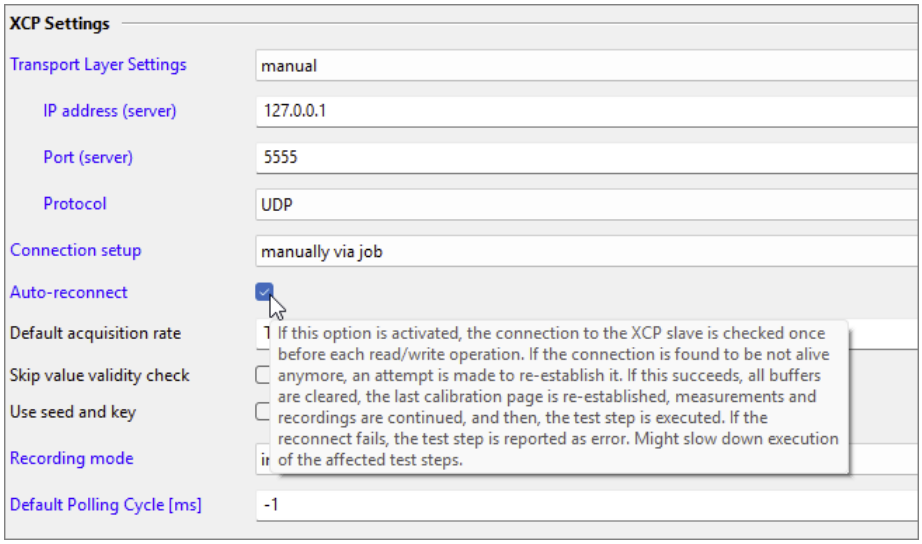


Abbildung 11: Funktion zum Auto-Reconnect aktivieren

Verhalten bei fehlgeschlagenem Reconnect

Schlägt der Reconnect-Versuch fehl, verhält sich das System wie folgt: Lesende Testschritte liefern den Status ***~NotPresent~*** zurück, schreibende Testschritte werden als Fehler gemeldet.

Hinweis: Durch die zusätzliche Verbindungsprüfung kann sich die Ausführungszeit der betroffenen Testschritte geringfügig erhöhen. Die Funktion sollte daher gezielt dort aktiviert werden, wo eine erhöhte Robustheit der XCP-Kommunikation erforderlich ist.

3.3 ecu.test *code*

Unterstützung für "bytes"

et

Für bestimmte Testschritte und Jobs existiert in **ecu.test** der Datentyp **ByteStream.** Damit lassen sich Bytefolgen parametrieren, berechnen und auswerten. Dieser Datentyp ist **ecu.test**-spezifisch und existiert in **ecu.test** *code* nicht.

Überall dort, wo **ByteStream** in **ecu.test** benötigt wird, kann nun aus **ecu.test** *code* heraus der Python-Basisdatentyp **bytes** verwendet werden. Dadurch wird eine Vielzahl von Testschritten befähigt, wie z. B. die Jobs **SendFrame()** und **ReadFrame()**, generisch auf Bushardware zuzugreifen.

```
21  def test_driving_mode():
22      # init test environment
23      ta = ToolAccess()
24      with ta.init():
25          bus_send_frame = ta.job("BUS01/SendFrame")
26          bus_read_frame = ta.job("BUS01/ReadFrame")
27
28          # start execution
29          with ta.run():
30              # using jobs with binary data
31              frameId = 0x1f4
32              frameData = b'\x12\x34\x56\x78\xab\xcd\xef\x12'
33              bus_send_frame(frameId, frameData)
34              assert(bus_read_frame(frameId) == frameData)
35
36  if __name__ == "__main__":
37      pytest.main()
38
```

Abbildung 12: SendFrame() und ReadFrame() für generischen Zugriff auf Bushardware

Unterstützung für die Datenstrukturen Vector, Matrix, Curve und Map

et

Analog zu **bytes** können in **ecu.test** *code* jetzt auch verschachtelte Listen für komplexe Datenstrukturen, die in **ecu.test** als Vector, Matrix, Curve und Map abgebildet werden, genutzt werden.

Möglich ist dies für Job-Aufrufe und Lese- und Schreib-Testschritte.

```
# working with complex data types (Vector, Curve, Matrix, Map)
m = Map(x_axis=[0, 10], y_axis=[11, 22, 33], values=[[0.1, 0.2, 0.3], [1.5, 2.5, 3.5]])
engine_params.write(m)
m2 = engine_params.read()
assert m2.shape == (2, 3)
assert m2.values[0][0] == 0.1
assert m2.x_axis[0] == 0
assert m2.y_axis[0] == 11
assert m2 == m  # equality check of the whole map checks values and both axes
```

Abbildung 13: Unterstützung für komplexe Datenstrukturen

Unterstützung für Konstanten in pytest

et

Um die Wartbarkeit, Lesbarkeit und Verständlichkeit von Testfällen zu erhöhen, kann es sinnvoll sein, Konstanten zu verwenden, die an einer zentralen Stelle definiert werden. Dies kann ab sofort mit dem folgenden Marker erreicht werden:

- **pytest.mark.constants(...)**

Der Marker akzeptiert Keyword-Argumente und kann entweder per Dekorator auf eine Testfunktion bzw. Testklasse angewendet oder durch das Setzen von **pytestmark** auf Modulebene verwendet werden.

Zusätzlich stellt die **Fixture constants** die per **pytest.mark.constants**(...) gesetzten Konstanten als Attribute zur Verwendung innerhalb des Tests bereit.

Bei aktiviertem Upload zu **test.guide** werden alle Konstanten an **test.guide** übertragen und dort dokumentiert.

```python
import pytest

# Makes GLOBAL_CONSTANT available to all tests in this module
pytestmark = pytest.mark.constants(GLOBAL_CONSTANT="myValue")

# MAX_RETRIES is only available in test_constants
@pytest.mark.constants(MAX_RETRIES=3)
def test_constants(constants):
    for i in range(constants.MAX_RETRIES):
        do_something()

    # GLOBAL_CONSTANT is available via the pytestmark variable
    assert constants.GLOBAL_CONSTANT = "myValue"

# constants markers can be combined
# constants are reported even if the constants fixture is not used
@pytest.mark.constants(ARCH="x86", OS="linux")
@pytest.mark.constants(BUILD_TYPE="release")
def test_something():
    assert True
```

Abbildung 14: Unterstützung für Konstanten in pytest

3.4 ecu.test *diagnostics*

Umfangreiche DiagBrowser Erweiterungen

et tc

Zur erleichterten Erstellung von Diagnose-Testschritten sowie der Interaktion mit der geladenen Diagnose-Datenbank (z. B. der ODX) wurde der **Diag-Browser** um zahlreiche API-Methoden erweitert.

- **DecodeUdsRequest**: zum Decoden eines Diagnose-Payload in eine menschenlesbare Struktur
- **EncodeUdsRequest**: zum Encoden einer Struktur in einen Diagnose-Payload
- **GetSubfunction**: zum Auslesen der Subfunction (wenn vorhanden)
- **GetId**: zum Auslesen des DID/RID des Service (wenn vorhanden)
- **GetDescription**: zum Auslesen der in der Datenbasis hinterlegten Beschreibung

Weitere Informationen können der API-Hilfe entnommen werden.

3.5 ecu.test *drive*

Synchrone/asynchrone und alternative Sprachausgabe bei Block-Testschritten

et

Bisher konnte nur zentral für alle Block-Testschritte entschieden werden, ob mit der Ausführung fortgefahren wird, bevor die Anweisungen vorgelesen wurden.

Jetzt kann diese Einstellung individuell pro Block vorgenommen werden. Dies ermöglicht mehr Flexibilität bei der Formulierung von interaktiven Testfällen.

Zusätzlich kann einem Wort in eckigen Klammern ein alternativer Text nachgestellt werden, der dann in *drive* statt dem eigentlichen Wort vorgelesen wird. So ist eine kompakte textuelle Darstellung bei gleichzeitig klar verständlicher Sprachausgabe möglich.

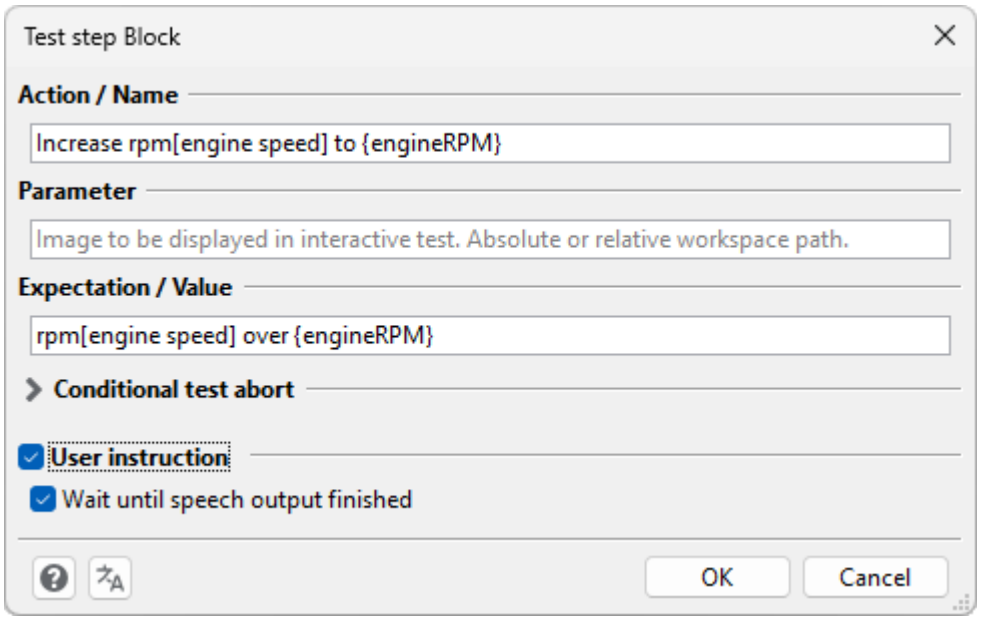


Abbildung 15: Block-Testschritt - Sprachausgabe

Unterstützung von ecu.test unter Linux mit und ohne GUI

et

ecu.test *drive* kann nun auch mit einem unter Linux laufenden **ecu.test** verwendet werden. Dabei spielt es keine Rolle, ob die Variante mit Benutzeroberfläche oder der Runner verwendet wird.

Testausführung in *drive* löst keine interaktive Ausführung in ecu.test aus

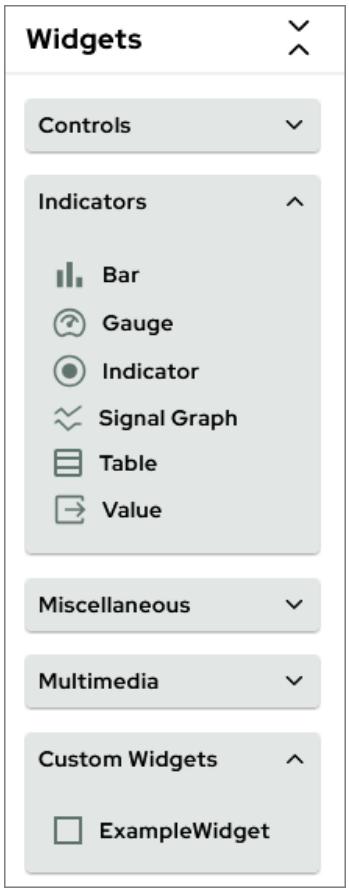
Eine in *drive* ausgelöste Testausführung führte bislang immer auch zu einer interaktiven Ausführung in der Oberfläche von **ecu.test**. Dies führte zu doppelter Darstellung, ggf. doppelt vorgelesenen Anweisungen und zu Schwächen in der Synchronisation von *drive* mit der Testausführung.

Daher wurden die beiden interaktiven Ausführungsmodi entkoppelt. Die Steuerung erfolgt jetzt ausschließlich über *drive*.

Eine interaktive Ausführung aus **ecu.test** heraus ist mit der integrierten interaktiven Oberfläche weiterhin möglich.

3.6 ecu.test *lab*

Benamung und Gruppierung von Widgets



Bisher wurden die verschiedenen Widgets nur in Form von Icons mit Tooltip angeboten. Dabei ging, insbesondere bei intensiver Nutzung von **Custom Widgets**, schnell die Übersicht verloren.

Nun wird zu jedem Widget der Name mit angezeigt und in Gruppen organisiert.

Für **Custom Widgets** besteht die Möglichkeit, eigene Gruppen zu spezifizieren und somit noch weitreichender anhand von beliebigen Kriterien in Gruppen zusammenzufassen.

Abbildung 16: Gruppierte Widgets

Copy/Paste von Widgets

et

Ab sofort ist es möglich, Widgets wie aus anderen Anwendungen gewohnt per **Strg+C** zu kopieren und mit **Strg+V** an beliebiger Stelle, auch auf anderen Pages in derselben, oder einer anderen View, wieder einzufügen.

Hinweis: Der bisherige Workflow mit **Strg+Drag** zum Duplizieren von Widgets entfällt!

Interaktion mit Widgets setzt Live-Modus voraus

et

Bisher war es möglich, einzelne Aktionen auf manchen Widgets auch ohne Monitoring-Modus auszulösen.
Da alle anderen Widgets bei diesem Workflow jedoch nicht aktualisiert wurden, bot dieser Weg wenig Mehrwert und führte regelmäßig zu Irritation bei Anwendenden, die vergessen hatten den Monitoring-Modus zu aktivieren.

Ab sofort ist die Bedienung von Widgets nur möglich, wenn der **Live-Modus** (ehemals Monitoring-Modus) gestartet ist. Darauf wird auch beim Versuch, ein Widget mit deaktiviertem **Live-Modus** zu bedienen, hingewiesen.

Um Anwendungsfälle, in denen Informationen nur bedarfsgesteuert aktualisiert werden sollen, weiterhin gerecht zu werden, ist die Abtastrate bei allen Widgets nun optional. Wird sie deaktiviert, erscheint automatisch ein **Refresh**-Button in der oberen rechten Ecke des Widgets. Ein Klick darauf löst dann eine einmalige Aktualisierung des Zustands des Widgets aus.

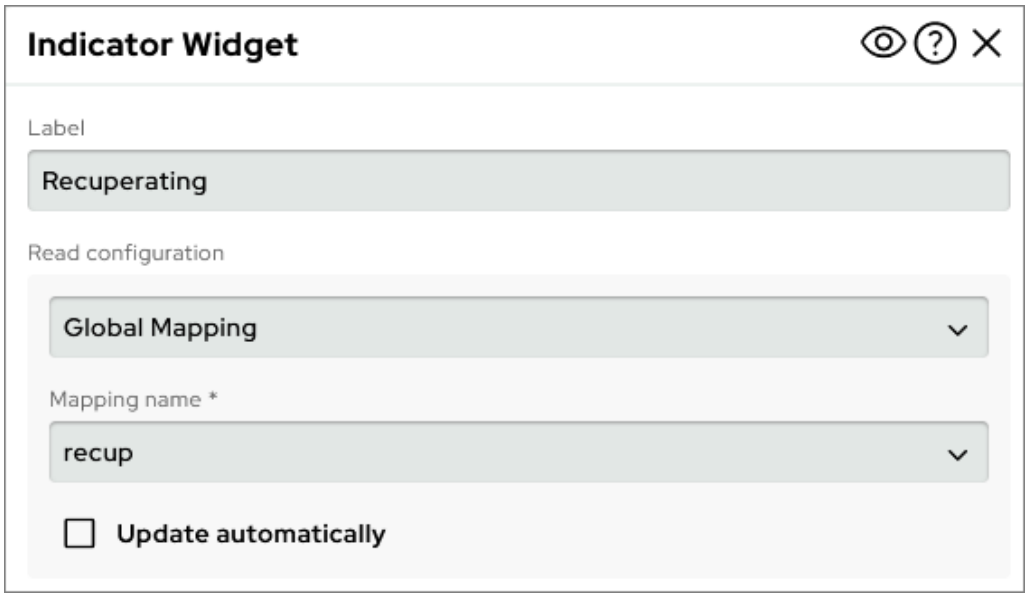


Abbildung 17: Indicator Widget

Verbesserte Verwaltung und Verfügbarkeit von Views und Pages

et

Schon zuvor konnten Pages aus referenzierten Workspaces über das Widget **Page Reference** eingebunden werden. Nun kann eine via Workspace bereitgestellte View komplett verwendet werden.

Ein Bearbeiten ist in diesem Fall nicht möglich. Die Kontrolle über die angebotenen Umfänge bleibt beim Ersteller der View im Kontext des Bibliotheksworkspaces. Außerdem werden die einzelnen Views nun in separaten Dateien abgelegt unter:

- **<workspace>/lab/views**

Dies erlaubt eine einfachere Verteilung und Verwaltung von Views und trägt zur Robustheit bei Fehlern in einzelnen Views bei.

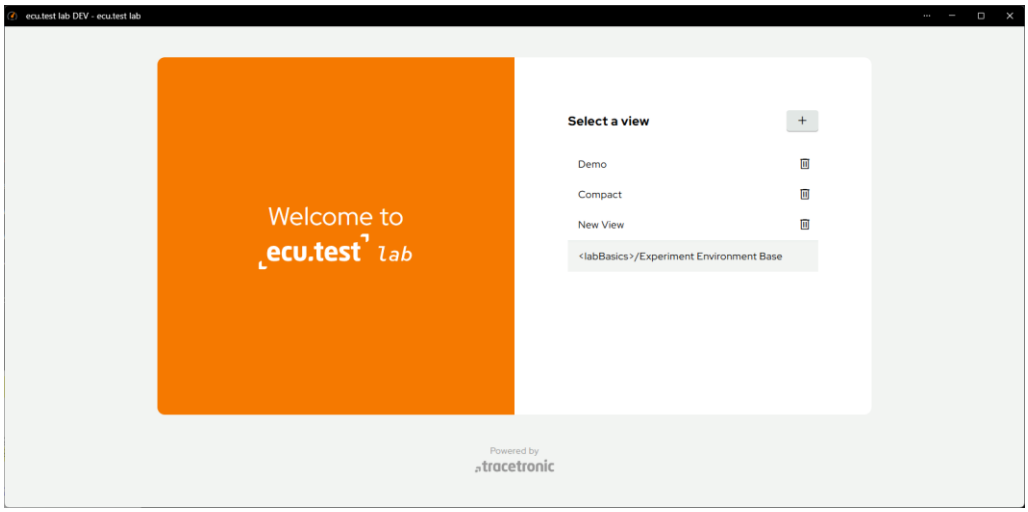


Abbildung 18: Willkommensseite mit verbesserter Verwaltung und Verfügbarkeit von Views und Pages

Erweiterungen für Custom Widgets

et

Signale können in der **widget.json** nun mit einer Minimalanzahl von **0** definiert werden. Dies ermöglicht die Erstellung von Widgets mit optionalen Signalen.

Die **WidgetApi** bietet nun die Methode **GetFile**, die es erlaubt den Inhalt einer Datei aus dem folgenden Verzeichnis auszulesen:

- **<workspace>/lab/shared/**

So kann z. B. eine umfangreichere Parametrierung oder Spezifikation des Widgets als separates Artefakt im Workspace ausgeliefert und vom Widget referenziert werden.

Package-Ausführung über Call-Widget

Mehrere Parameter sind beschreibbar, alle Rückgabewerte sichtbar. Ideal für komplexere Statuswechsel oder -abfragen.

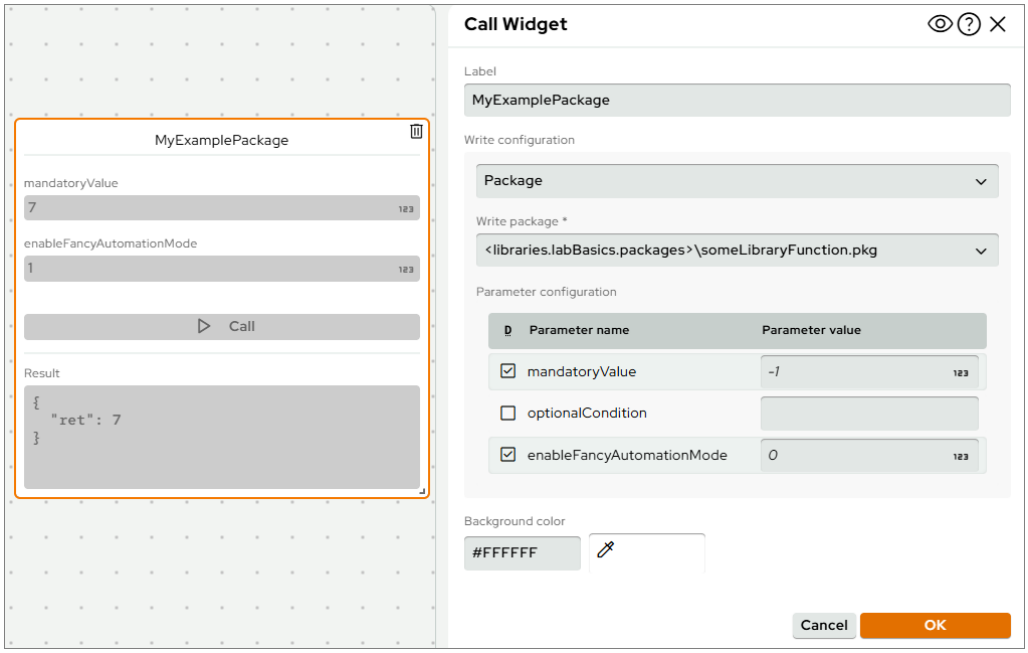


Abbildung 19: Packageausführung über Call-Widget

Jobausführung in allen Mapping-fähigen Widgets

Alternativ zu direkten Signalzugriffen können nun auch Jobs ausgeführt und so Tools und Ports direkter angesprochen werden. Eine zyklische Ausführung wird ebenfalls unterstützt.

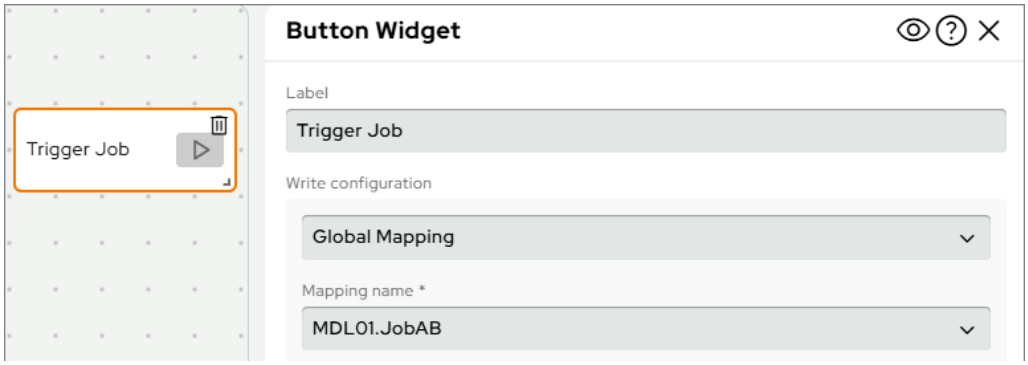


Abbildung 20: Jobausführung in allen Mapping-fähigen Widgets

Direkte Interaktion mit Bild in den Widgets Image und Video

et

Direktes Klicken auf das angezeigte Bild oder Video kann eine mit den entsprechenden Pixelkoordinaten parametrierte Package-Ausführung auslösen.
Im Package kann die Stimulation des Systems auf beliebigem Weg erfolgen.

So ist eine direkte Interaktion mit den angezeigten Bildschirminhalten möglich.

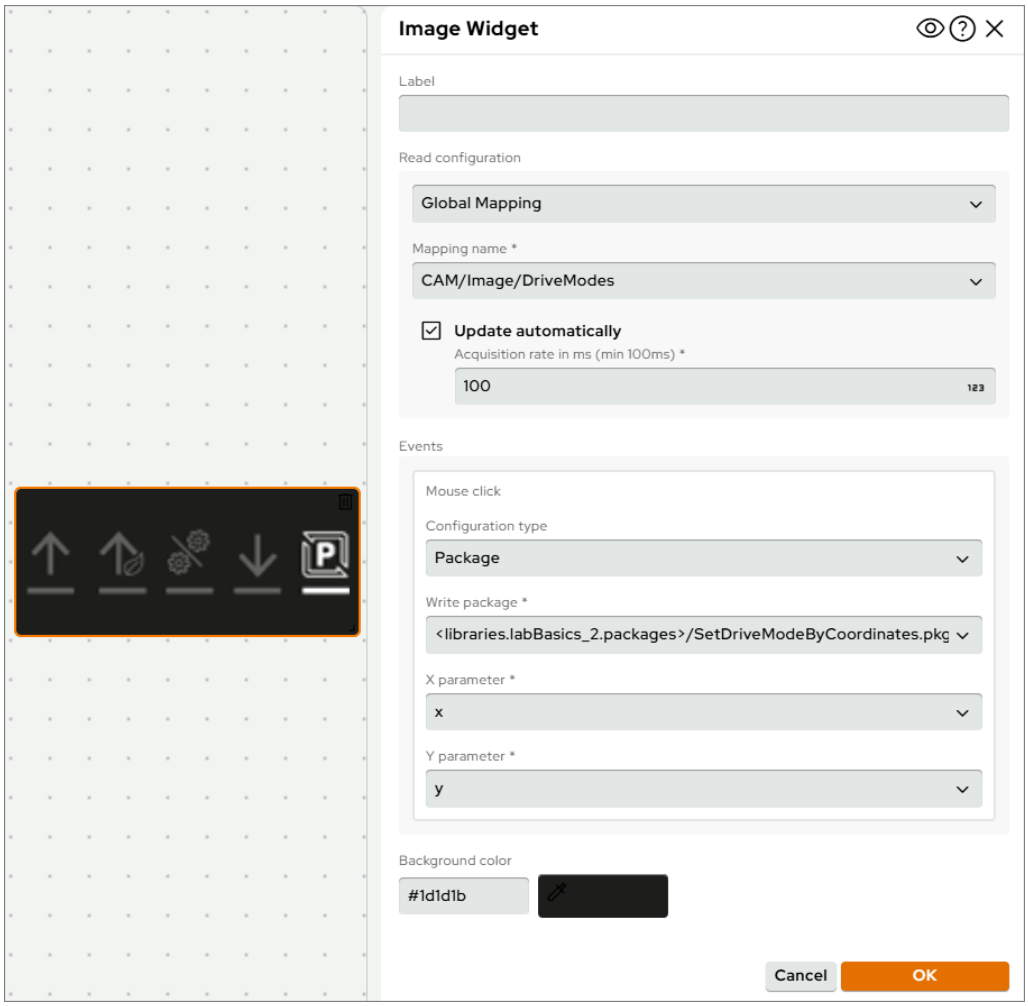


Abbildung 21: Direkte Interaktion mit Bild in den Image und Video Widgets

Tabellen-Widget

Eine kompakte Auflistung von Werten war bislang nur über ein Custom Widget realisierbar. Mit dem Widget **Table** bieten wir diese Möglichkeit nun direkt an.

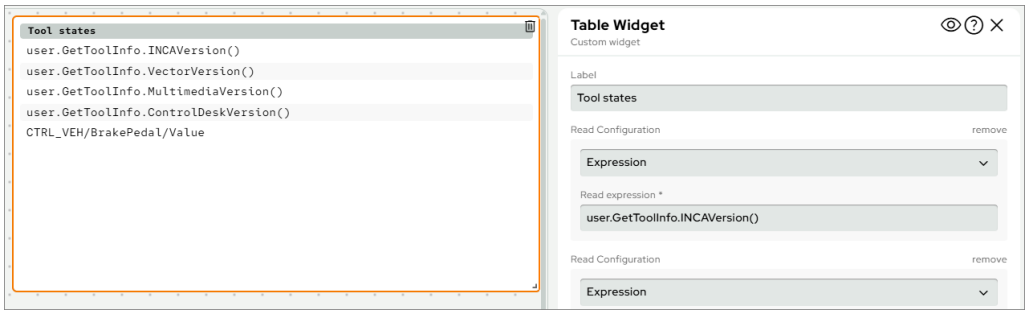


Abbildung 22: Kompakte Auflistung von Werten direkt im Table Widget

4 Usability

Verbesserungen für die ecu.test Linux GUI

Nach dem Release der **ecu.test** GUI für Linux mit **ecu.test 2026.1** gibt es diverse Neuerungen und Verbesserungen für Linux-Nutzer:

- Der Lizenzmanager steht nun auch als GUI zur Verfügung, sodass alle nötigen Lizenzeinstellungen auch ohne die Verwendung der Konfigurationsdatei oder von Umgebungsvariablen zur Verfügung stehen.
- Der Tool-Server wird nun analog zu **ecu.test** inkl. eines Startmenü-Icons installiert.
- **ecu.test**-Artefakte wie Packages oder Reports sind mit **ecu.test** verknüpft, sodass sie, die aus Windows bekannten Icons haben und sich bei einem Doppelklick direkt öffnen.

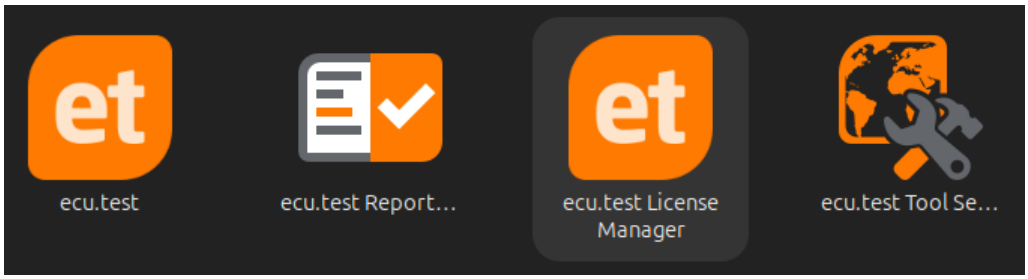


Abbildung 23: Lizenzmanager und Tool-Server für ecu.test unter Linux

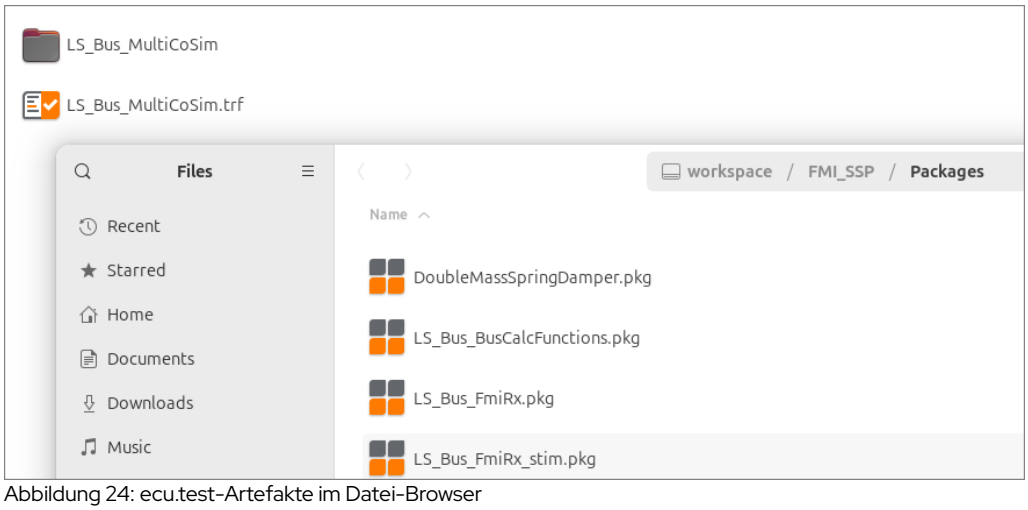


Abbildung 24: ecu.test-Artefakte im Datei-Browser

Bibliotheksworkspaces: Unterstützung von Templates und Reportgeneratoren

et tc

Bibliotheksworkspaces in **ecu.test** unterstützen nun auch die Verwaltung von Templates für Projekte, Packages und Analysepackages.

Diese werden wie gewohnt im Templates-Verzeichnis des Workspaces abgelegt und stehen dann im lokalen Workspace zur Verfügung.

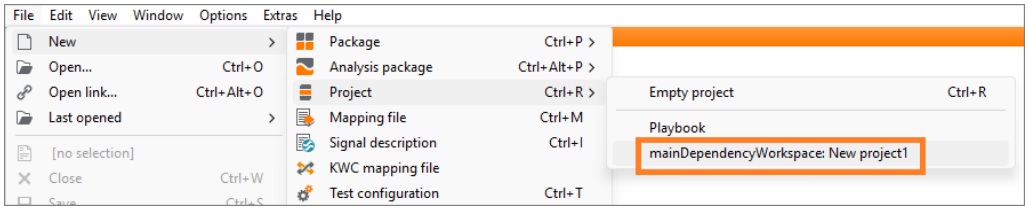


Abbildung 25: Auswahl eines Projekttemplates im Dateimenü

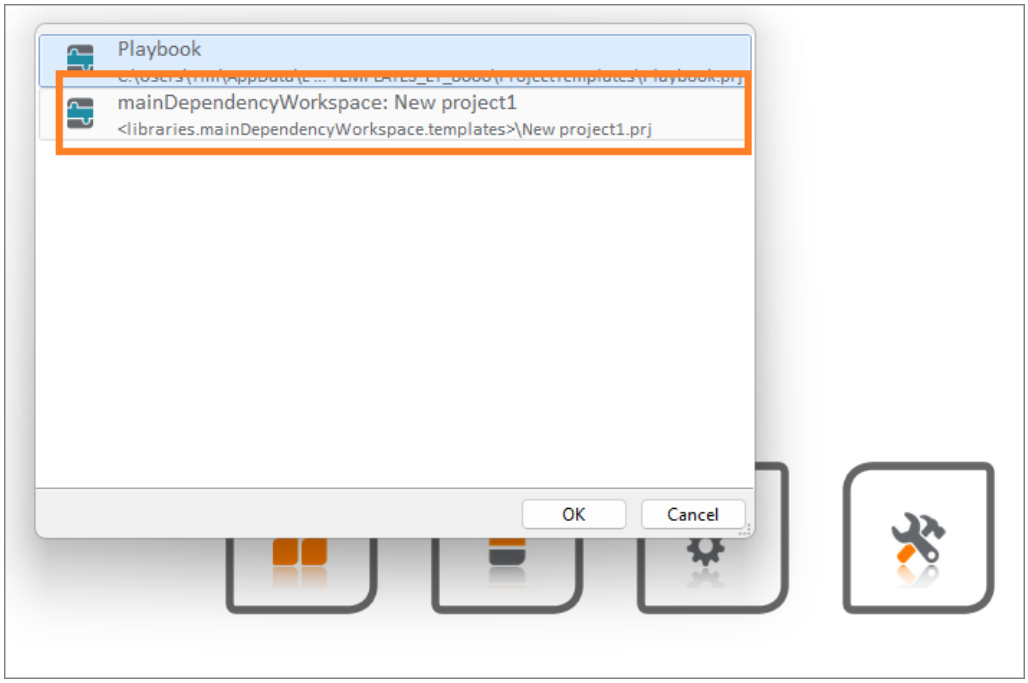


Abbildung 26: Auswahl eines Projekttemplates im Hauptprogramm

Außerdem können nun Reportgeneratoren in Bibliotheksworkspaces verwaltet und in einer TCF ausgewählt werden.

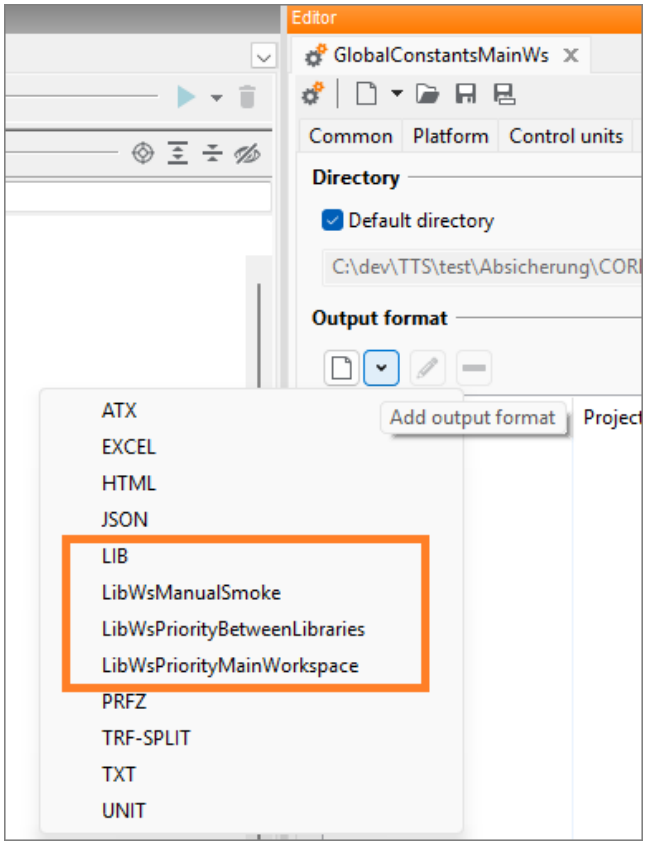


Abbildung 27: Auswahl eines Reportgenerators in der TCF

Bibliotheksworkspaces: Anzeige des Git-Branches

et tc

Der aktive Git-Branch eines Bibliotheksworkspaces wird nun in der Spalte **Beschreibung** angezeigt, sodass der verwendete Branch auf einen Blick ersichtlich ist.

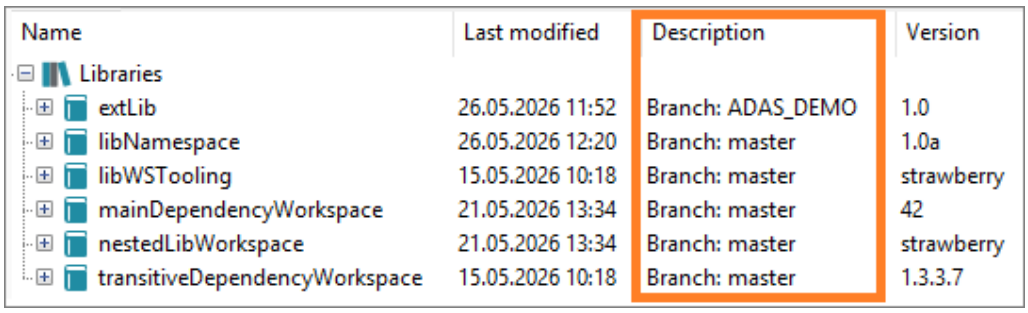


Abbildung 28: Darstellung des GIT-Branches im Workspace-Explorer

Nutzerdefinierte Restrukturierung von Modelgrößen

et tc

Einige Modelle nehmen eine enorme Größe ein - in manchen Fällen gibt es hunderttausende Variablen. Typischerweise wird im Testfall nur eine kleine Auswahl benötigt - trotzdem wird immer das komplette Model eingelesen.

Zudem kann es vorkommen, dass sich Modelgrößen bei einem Versionssprung innerhalb der Struktur verschieben. In diesem Fall ist das Mapping nicht mehr gültig und muss mühsam in den Testfällen nachgepflegt werden.

Mit **ecu.test 2026.2** kann dem Model in der TCF ein User-Script beigelegt werden, in dem nutzerdefinierte Filter sowie Aliase angewendet werden. Über einfache Python-Logik können so Modelbäume aufs Wesentliche reduziert werden oder neue Views für bestimmte Modelgrößen erstellt werden.

Weitere Informationen und Beispiele sind der Anwenderdokumentation (Kapitel: **Tab Modellzugriff**) zu entnehmen.

```python
MODULE_TYPE = 'USER_MODEL_MODIFIER'

class ExampleUserModelModifier:

   def GetVariableAlias(self, variablePath: str) -> str | None:
      return 'myAlias' if variablePath == 'path/to/the/variable' else None

   def IgnoreVariable(self, variablePath: str) -> bool:
      return True if variablePath == 'path/to/the/variableToBeIgnored' else False
```

Abbildung 29: User-Script zur Restrukturierung von Modellbäumen

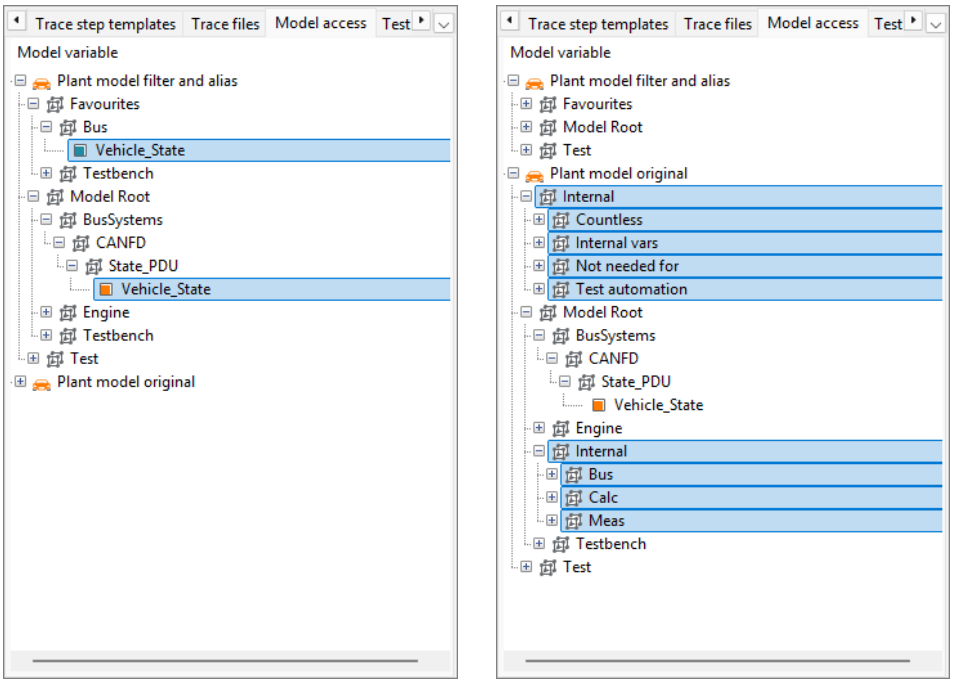


Abbildung 30: Darstellung reduzierter Modellbäumen durch Aliase (links) und Filter (rechts)

Neuer Testschritt "Repeat-Until-SUCCESS“

et

In nicht kontrollierbaren Umgebungen, insbesondere beim manuellen Testen im Fahrzeug, kann es nötig werden, eine Testsequenz mehrfach zu wiederholen, um einen gewissen Systemzustand zu erreichen. Diese Wiederholungsmöglichkeit erspart einen kompletten Abbruch und Neustart des Testfalls.

Der neue Testschritt **Repeat-Until-SUCCESS** ermöglicht genau das. Das wiederholte Ausführen einer Sequenz von Testschritten bis deren Gesamtbewertung SUCCESS lautet, oder eine maximale Anzahl an Versuchen erreicht wurde.

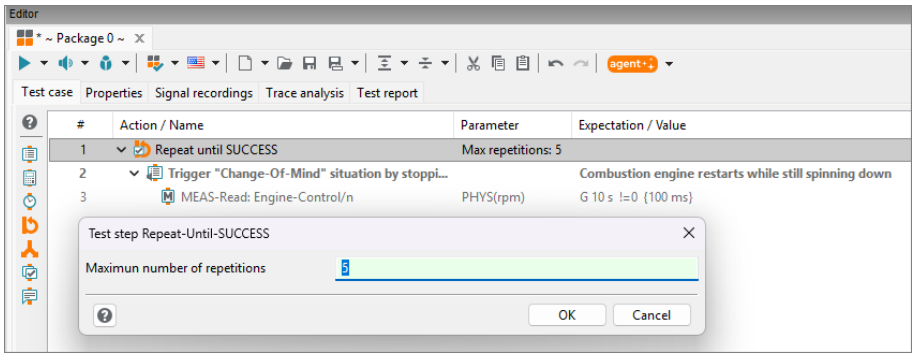


Abbildung 31: Testschritt "Repeat-Until-SUCCESS“

Schnelle Suche von Service-Methoden und Events

et tc

Mit dem **ecu.test** Suchmechanismus **EasyInsert** (**Strg+Leertaste**) können ab sofort auch Service-Methoden und -Events basierend auf AUTOSAR-Beschreibungsdateien schnell und einfach gesucht und gefunden werden.

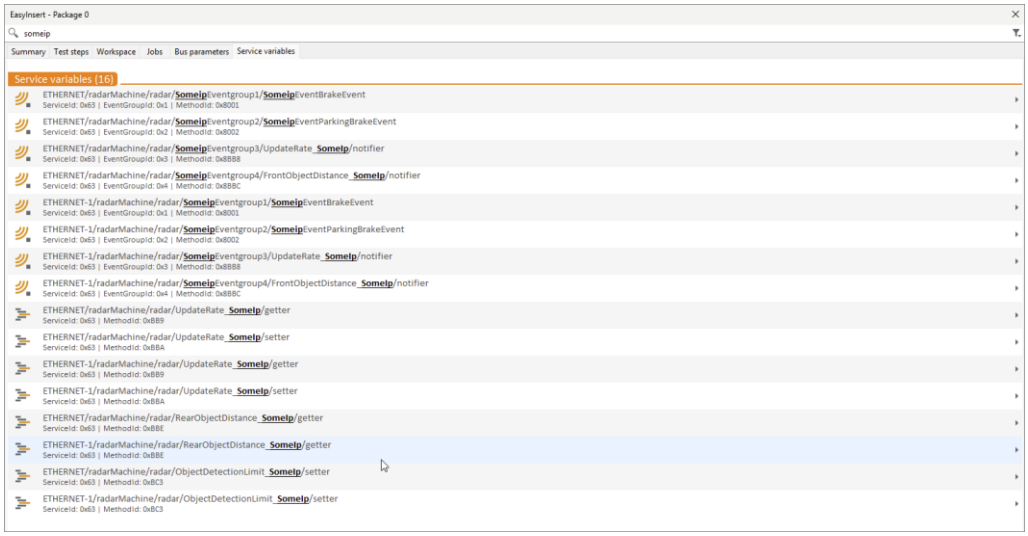


Abbildung 32: EasyInsert zur Suche nach Service-Methoden und -Events basierend auf AUTOSAR-Beschreibungsdateien

Systemzertifikate für HTTPS-Verbindungen

et tc

Beim Aufbau von verschlüsselten Serververbindungen (HTTPS) werden jetzt Zertifikate aus dem Systemzertifikatspeicher verwendet, um die Vertrauenswürdigkeit der Verbindung zu überprüfen. Dies trifft nicht auf benutzerdefinierte Implementierungen zu.

Verbesserte Performance bei Auflösen von Python-Abhängigkeiten

et tc

Das Auflösen und Installieren von Python-Abhängigkeiten über die **requirements.txt** wurde deutlich verbessert und bietet dank eines workspace- und plattformabhängigen Cache-Mechanismus eine wesentlich bessere Performance.

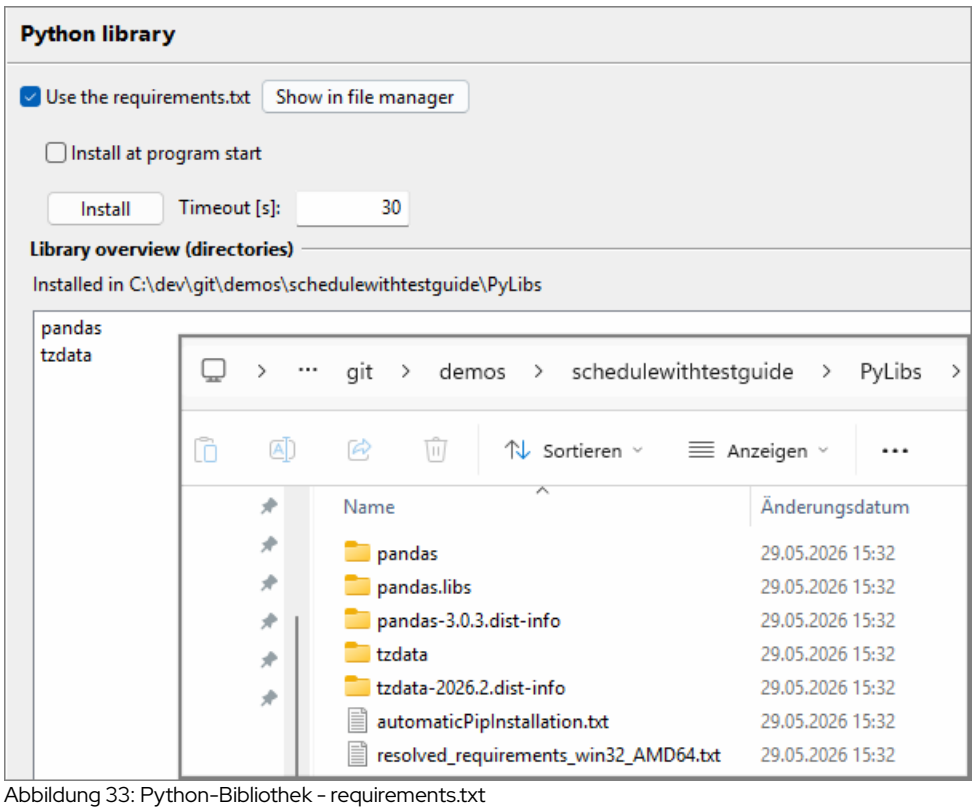


Abbildung 33: Python-Bibliothek - requirements.txt

Erweitere Workflows von Projektreport

et tc

Der **PRFZ-Export** steht jetzt als **Reportgenerator** zur Verfügung.

Beim Öffnen eines Reports aus **test.guide** wurden außerdem die Download-Abläufe verbessert:

- Der Download vieler Dateien erfolgt nun **im Hintergrund** und mit **einem zentralen Fortschrittsbalken**.
- Gewünschte **Reportartefakte** wie Aufnahmen können jetzt direkt über das **Kontextmenü im Reportbaum** heruntergeladen werden.
- Dadurch lässt sich ein Report **vollständig offline verfügbar** machen.
- **Zusätzliche Downloads beim Klick auf einen Subreport entfallen.**

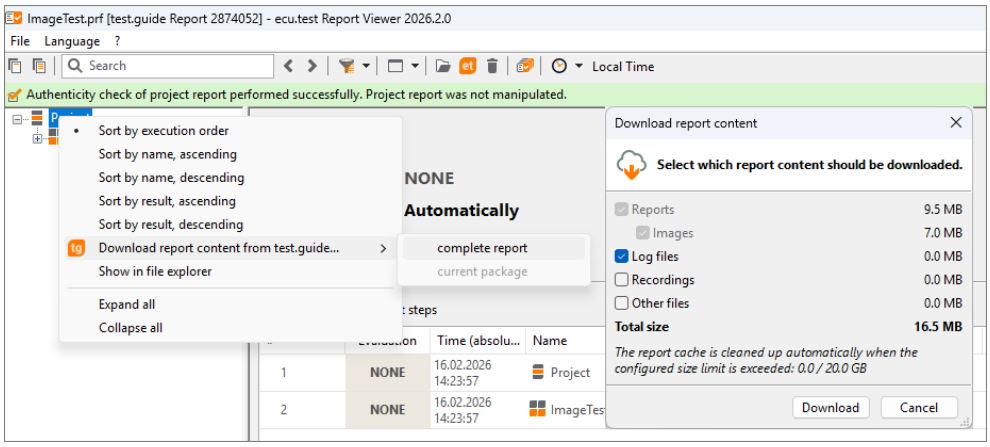


Abbildung 34: PRFZ-Export als Reportgenerator

Testkonfiguration: Angabe von Konstanten in der Bus-Kanalauswahl

et tc

Für Bus- oder Service-Beschreibungsdateien wie z. B. AUTOSAR XML ist es erforderlich den Kommunikations-Cluster auszuwählen. Um eine **ecu.test** Testkonfiguration für unterschiedliche Bus-Konfigurationen wiederverwenden zu können, kann es erforderlich sein, den Cluster zu parametrieren.

Ab sofort kann eine über globale Konstante im Eingabefeld der Testkonfiguration dafür angegeben werden.

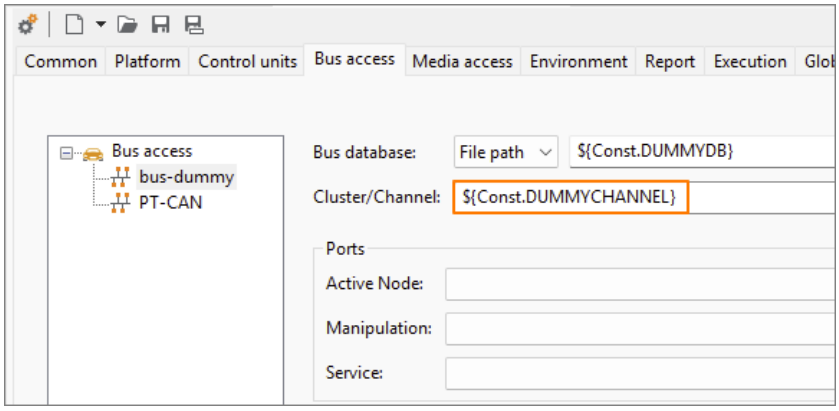


Abbildung 35: Angabe von Konstanten in der Bus-Kanalauswahl

Neues Tutorial für den Einstieg in ecu.test

et

Die bisherigen Einstiegskapitel der Dokumentation wurden konsolidiert, modernisiert und zu einem zusammenhängenden Tutorial ausgebaut, das inhaltlich die bisherige Basic-Schulung abdeckt. Es richtet sich an neue Anwendende und ermöglicht einen direkten, praktischen Einstieg anhand mitgelieferter Beispielartefakte und Übungsprojekte.

Das Tutorial ist als autodidaktischer Einstieg konzipiert und führt Schritt für Schritt durch die grundlegenden Konzepte und Arbeitsabläufe in **ecu.test**. Die Kapitel werden in kommenden Releases kontinuierlich erweitert.

Für komplexere Anwendungsfälle oder eine tiefergehende Einarbeitung steht weiterhin ein individuelles Training zur Verfügung, das gezielt auf konkrete Tools, Projekte und Fragestellungen eingehen kann.

5 Testaspekte

5.1 HiL

Hochperformante Verarbeitung von Bus-Kommunikation aus den Ethernet-Loggerdatenströmen ASAM CMP, PLP, TECMP

et

Die Kommunikation klassischer Busse wie CAN(-FD), FlexRay und LIN kann in **ecu.test** auch aus gebündelten Logger-Datenströmen extrahiert werden. Mit diesem Release wurde die Verarbeitungsleistung deutlich erhöht. Dadurch lassen sich auch bei Datenraten im Gigabit-Bereich verlustfrei Traces in **ecu.test** aufzeichnen und Signale lesen.

Die Verbesserung muss nicht aktiviert werden. Sie wird automatisch für alle Ethernet-Bus-Capture-Protokolle verwendet.

Weitere Protokolle, die im Testfall passiv analysiert werden können, wie DLT und SOME/IP, werden mit dem nächsten Release auf die neue Technologie umgestellt.

ViGEM (CCA): Dateifilter für DownloadLatestRecordings

Der Job **DownloadLatestRecordings** für ViGEM unterstützt jetzt Filter über **Include**- und **Exclude**-Patterns (Glob-Syntax).
Damit lassen sich gezielt bestimmte Dateien und Ordner beim Download ein- oder ausschließen. Das erleichtert z. B. das Ausschließen großer Daten wie MIPI-Videos. Ohne gesetzte Patterns bleibt das Verhalten wie bisher.

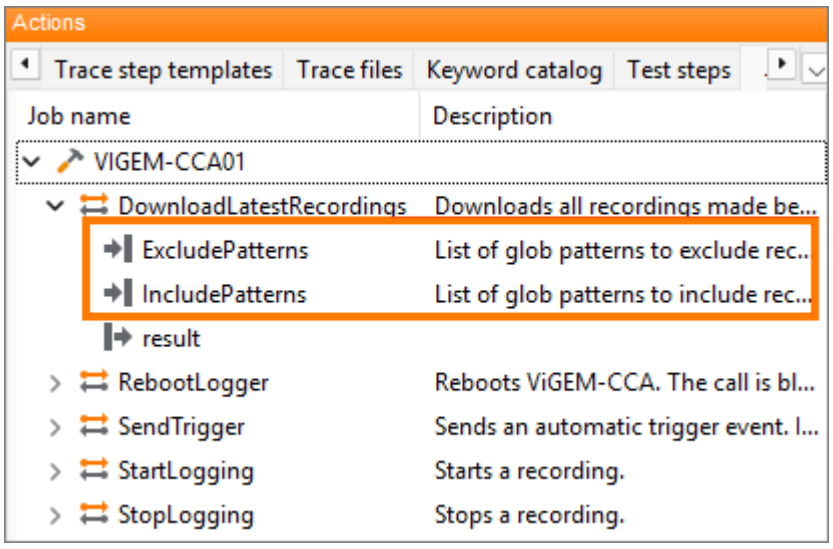


Abbildung 36: DownloadLatestRecordings-Job unterstützt Filter Include- und Exclude-Patterns

Star Electronics FL3X RBS: Neue Toolanbindung

et

Abbildung 37: Star Electronics FL3X RBS

Mit diesem Release steht in **ecu.test** das neue Tool **FL3X** zur Verfügung.

Es ermöglicht die Anbindung der **FlexDevice**-Familie von **Star Electronics** als Busmanipulationshardware.

Zudem erweitert es damit die Möglichkeiten zur Restbussimulation und Signalmanipulation in automatisierten Testabläufen.

Das neue Tool stellt einen Busmanipulationsport bereit und bietet den einen busunabhängigen Zugriff auf alle klassischen Bussysteme (CAN, LIN, FlexRay). Unterstützt werden insbesondere:

- Schreiben und Manipulieren von Signalen durch Setzen fester Werte, Freeze, Offset sowie Factor-, Rampen- und Drift-Manipulationen
- Anwendung der Signal-Manipulationen
- Aktivieren und Deaktivieren einzelner Botschaften
- Zugriff auf globale Variablen des **FlexDevice** via Jobs

5.2 Testmanagement

Erweiterung des Jama Beispiel-Workflows

et

Der Beispiel-Workflow nutzt nun die neue Object API, um Attribut-Spezifikationsdateien zu generieren. Attribute werden nicht mehr nur aus **Jama** in **ecu.test** übernommen, sondern können direkt in einem Package oder Projekt auf der **ecu.test**-Seite gesetzt oder geändert werden.

Der Export von Packages aus **ecu.test** zurück in **Jama** wird ebenfalls unterstützt, und alle obligatorischen Attribute können in den Packages festgelegt werden, bevor der Export gestartet wird.

Codebeamer 3 Support

et

Da der aktive Support für Codebeamer 2.x ausläuft, wurde der Beispiel-Workflow und die tracetronic-Codebeamer-Bibliothek aktualisiert, um die Kompatibilität mit **Codebeamer 3** zu gewährleisten.

Dadurch wird die weitere Funktionsfähigkeit sichergestellt und es können die Funktionen der neuen Plattform genutzt werden.

5.3 Traceanalyse

Autovervollständigung und Typehints in externen IDEs verbessert

et tc

Die bereitgestellten Interfaces für NumPy-Traceschrittvorlagen wurden aktualisiert, um sicherzustellen, dass z. B. in **VS Code** direkt nach deren Einbinden die Autovervollständigung und Typehints funktionieren.

Die Anwenderhilfe wurde um Hinweise zur korrekten Einrichtung in **VS Code** und **PyCharm** erweitert.

Unterstützung von ZSTD-Kompression in MDF 4.3

et tc

Der MDF 4.3 Standard wurde im September 2025 veröffentlicht. Inzwischen adaptieren erste Tools Teile des Standards. Mit **ecu.test 2026.2** wird nun das Lesen von ZSTD-komprimierten Signaldaten unterstützt.

Parallele Analysejob-Abarbeitung: Verbesserte Performance unter Windows

et tc

Unter Windows profitiert die Projektausführung mit paralleler Analysejob-Abarbeitung von einer schnelleren Ausführung.

5.4 trace.xplorer

Kontext von Traceschritten aus Reports im trace.xplorer aktualisieren

et tc

Viele Anwendende wünschten sich, dass der Kontext einzelner Traceschritte beim Auswerten von Reports schneller umgeschaltet werden kann. Bislang musste für jeden neuen Traceschritt der **trace.xplorer** geschlossen, der nächste Traceschritt im Report gewählt und der **trace.xplorer** wieder geöffnet werden. Der zuvor angezeigte Traceausschnitt (Scrollposition und Zoomfaktor) ging dabei verloren und musste händisch wiederhergestellt werden. Der Workflow erforderte viele Klicks und erzeugte viel Unruhe auf dem Bildschirm.

Um die Situation zu verbessern, arbeiten die Reportansichten von **ecu.test** und dem **Report Viewer** ab sofort enger mit dem **trace.xplorer** zusammen.

Wird der **trace.xplorer** aus dem Report heraus aufgerufen, visualisiert er wie bisher die am aktuellen Traceschritt beteiligten Signale sowie eventuell erzeugte Bewertungen. Bei der Auswahl eines anderen Traceschrittes bleiben nun die geöffnete **trace.xplorer**-Instanz und der angezeigte Traceausschnitt erhalten. Es wird lediglich der visualisierte Traceschritt-Kontext, also die eingeblendeten Signale und Bewertungen, aktualisiert.

Insbesondere bei Verwendung von mehreren Bildschirmen lässt sich auf diese Weise die enthaltene Reportinformation viel effizienter als vorher analysieren.

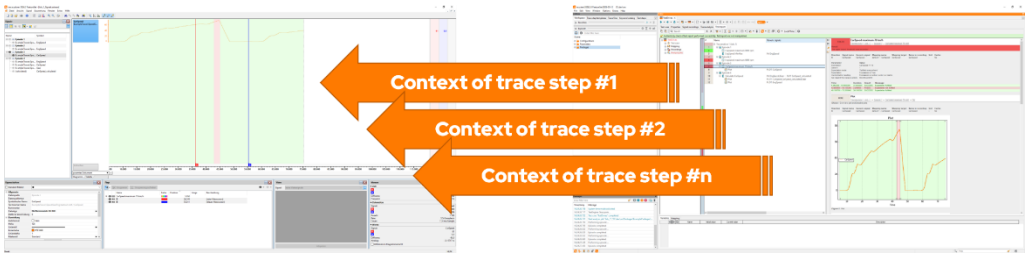


Abbildung 38: Angezeigte Informationen werden im trace.xplorer automatisch aktualisiert, sobald im Testreport ein neuer Traceschritt ausgewählt wird

Über die neue Symbolleiste **Testreport** gelangt man auf einen Klick zur Reportansicht des Aufrufers (**ecu.test** oder **Report Viewer**). Außerdem lässt sich die automatische Aktualisierung der Kontextinformationen bei Bedarf an- und ausschalten.

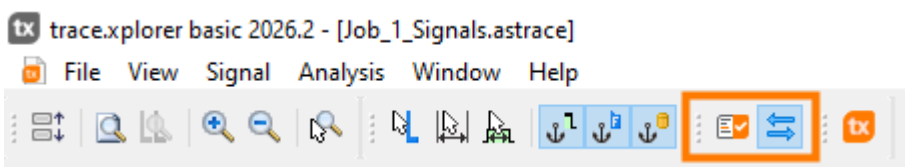


Abbildung 39: Neue Symbolleiste für die Verbindung mit einer Testreportansicht

Importieren von Rohwerten aus MDF4

et tc

Beim Importieren von Signalen aus MDF4-Tracedateien kann es passieren, dass bestimmte Signale nicht zur Auswahl angeboten werden, obwohl **ecu.test** diese anzeigt.

Eine mögliche Ursache dafür wurde nun beseitigt: Der MDF4-Importfilter gestattet jetzt auch den Import von Signalen, deren Rohwerte nicht in physikalische Werte umgerechnet werden können, weil die verwendete Konvertierungsvorschrift nicht unterstützt wird. In solchen Fällen werden statt den physikalischen Signalwerten die Rohwerte importiert.

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

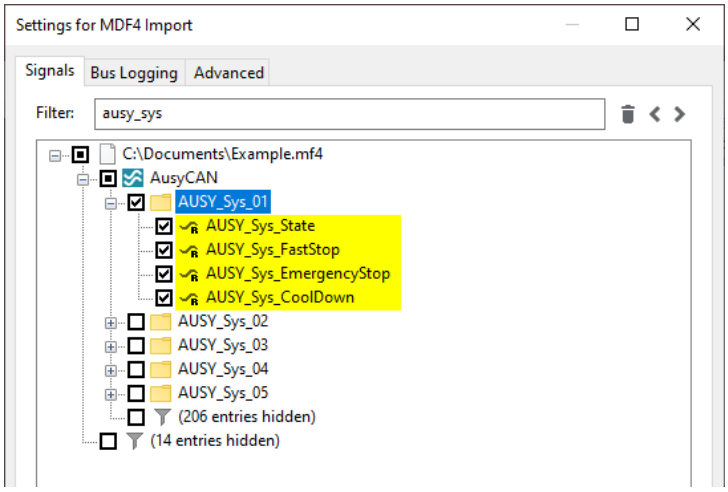


Abbildung 40: Rohwert-Signale sind im Auswahlbaum an ihrem Icon (Signalkurve plus 'R') erkennbar

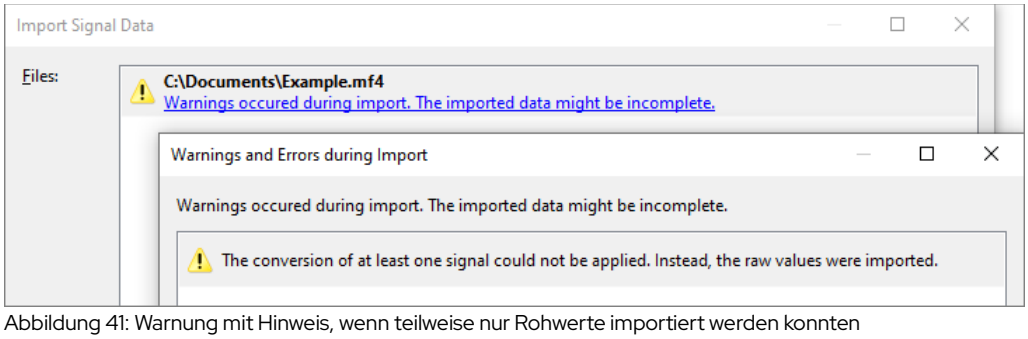


Abbildung 41: Warnung mit Hinweis, wenn teilweise nur Rohwerte importiert werden konnten

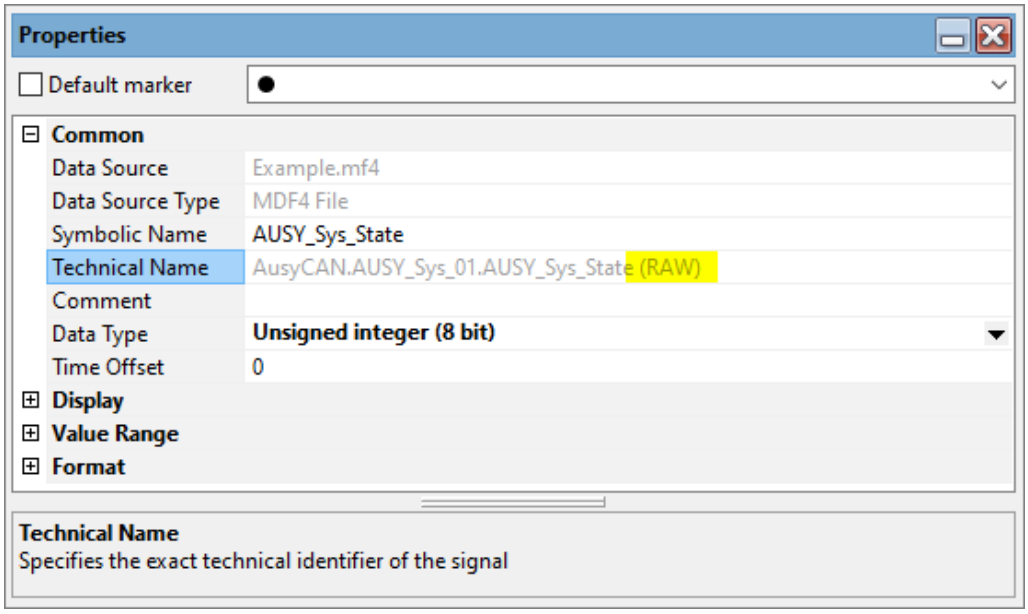


Abbildung 42: Technische Namen von Rohwert-Signalen besitzen nach Import das Suffix '(RAW)':

Signalwertänderungen schnell finden

et tc

Wer bisher Sample-Zeitpunkte suchte, zu denen sich der Wert eines Signals änderte, konnte nur für Bool-Signale eine Suchbedingung formulieren, um nach steigenden oder fallenden Flanken zu suchen.

Ab sofort steht für alle Signaldatentypen die neue Vergleichsoperation **Wertänderung** zur Verfügung, um beliebige Signalwertänderungen zu finden. Auch Unterbrechungen im Signalverlauf (Datenlücken) werden hierbei als Änderung verstanden.

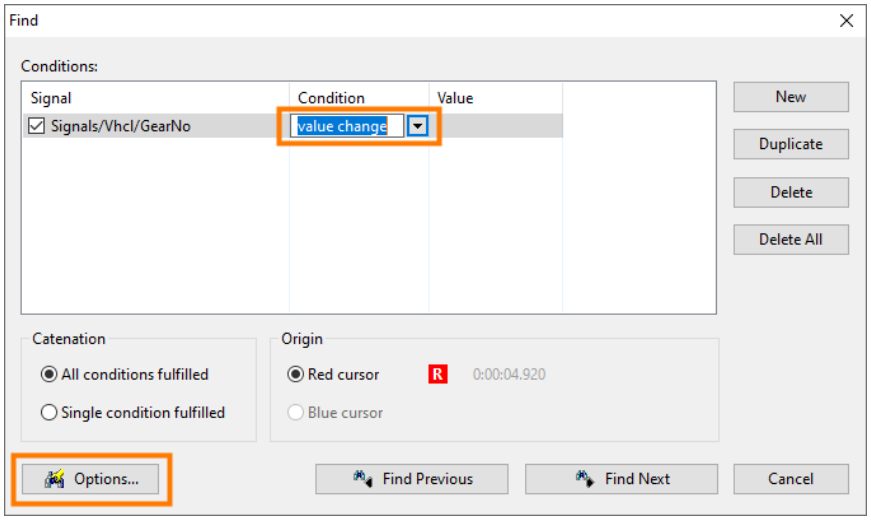


Abbildung 43: Suche bietet neue Vergleichsoperation **Wertänderung** und Optionen in separatem Dialog

Um diese Suchfunktion bestmöglich verwenden zu können, steht sie über die zugehörige Symbolleiste **Suchen** und den Shortcut **Alt+F3** zur Verfügung. Ein Aufruf über diese beiden Wege sucht nach Änderungen im aktiven (gestrichelt umrandeten) Signal und setzt den roten Cursor an passende Fundstellen.

Suchoptionen, wie das Fortsetzen der Suche an Dokumentgrenzen, werden berücksichtigt. Die Suchoptionen wurden deshalb in einen eigenen Dialog ausgelagert, der ebenfalls über die Symbolleiste geöffnet werden kann.

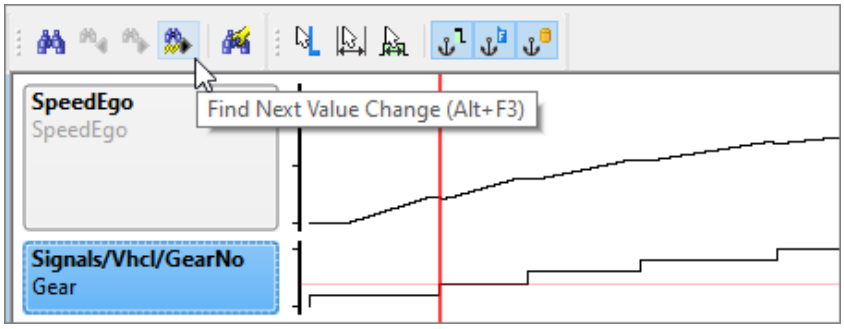


Abbildung 44: Symbolleiste erlaubt die Suche nach Wertänderungen und das Einstellen der Suchoptionen

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

Kleinere Verbesserungen der Usability

et tc

Die **Flagliste** enthält mitunter sehr viele Einträge für Markierungen (Cursors, Flags, Maßketten, Signalbemaßungen und Schattierungen), die alle in der ein oder anderen Form in den Ansichten visualisiert werden. Da kann schnell die Übersicht verloren gehen - sowohl in der Liste als auch in den Ansichten.

Um die Übersicht zu behalten, bietet die **Flagliste** in ihrer Symbolleiste eine neue Schaltfläche zum Filtern von Einträgen getrennt nach ihrer Art. Über ein Kontextmenü lässt sich steuern, welche Markierungen bei aktiver Filterung sichtbar bleiben sollen. Die Filterung hat nur Auswirkung auf die **Flagliste**, nicht auf die Ansichten.

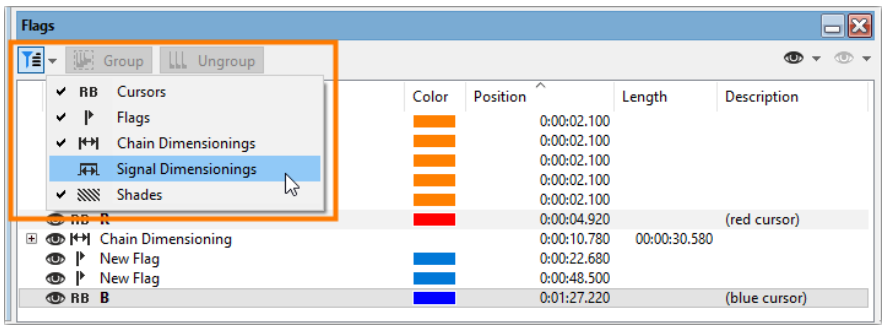


Abbildung 45: Neue Funktion zur Filterung von Einträgen in der Flagliste

Bei der Suche nach der Ursache eines Testfehlschlags kann es entscheidend sein, die Abtastzeitpunkte eines Signals zu kennen, insbesondere bei selten abgetasteten Signalen.

Der **trace.xplorer** bietet dafür nun die Möglichkeit, sich mit einem Klick Samplemarker anzeigen zu lassen. Ist die neue Option **Standard-Marker** im Andockfenster **Eigenschaften** aktiviert, werden die Abtastwerte von allen Signalkurven mit dem gewählten Markersymbol gekennzeichnet, für die der Markerstil **Standard** gesetzt ist. Signale, die bereits einen speziellen Markerstil besitzen, bleiben davon unberührt.

Ob und wie die Samples standardmäßig markiert werden, ist eine Benutzereinstellung, die für alle Dokumente wirksam ist.

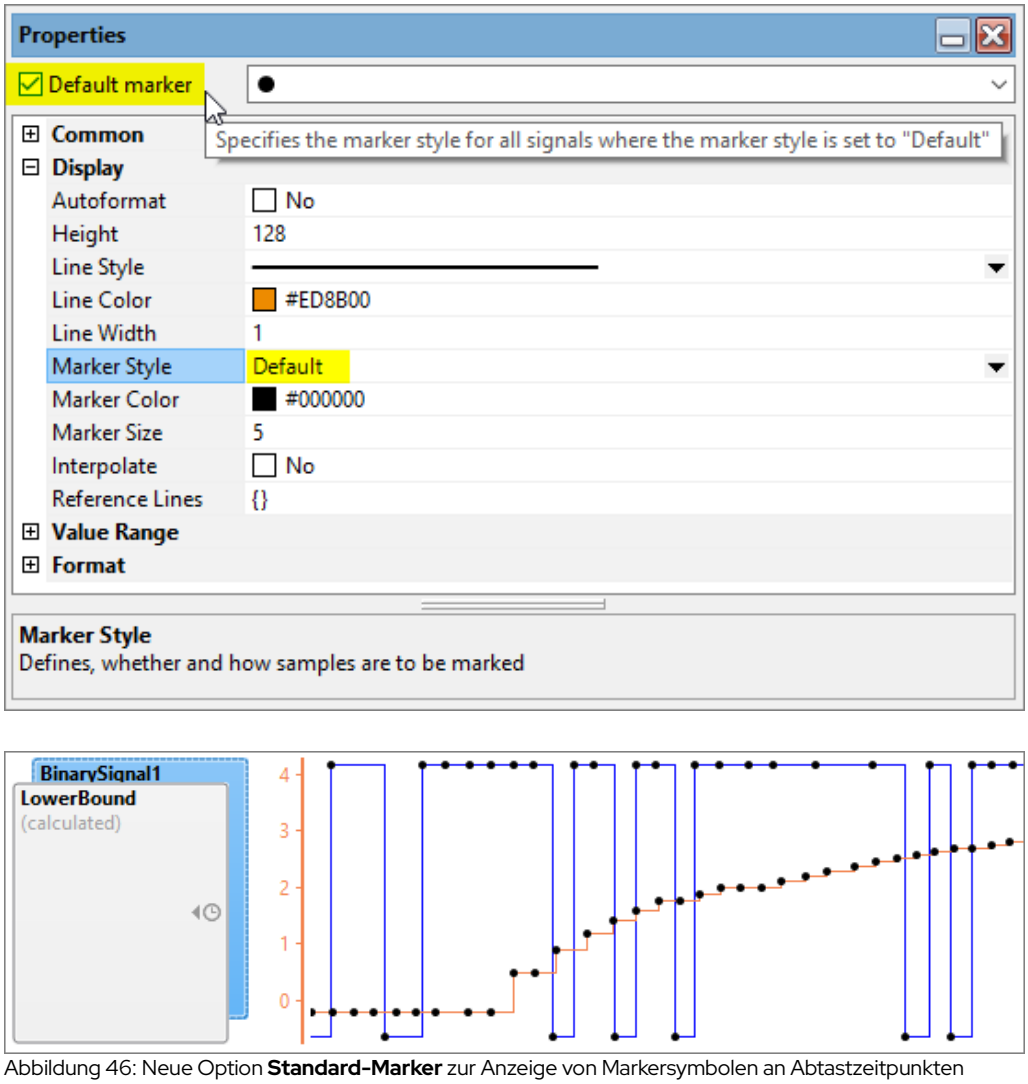


Abbildung 46: Neue Option **Standard-Marker** zur Anzeige von Markersymbolen an Abtastzeitpunkten

Die bestehenden Möglichkeiten zum Kopieren von Inhalten aus der Tabellenansicht in die Zwischenablage wurden erweitert. Bisher ließen sich schon die Daten der aktuellen Zelle (neuer Shortcut **Strg+C**) sowie einer einzelnen Zeile kopieren.

Nun bietet das Kontextmenü der Signalköpfe zusätzliche Befehle zum Kopieren von ganzen Signalen. Ausgewählte Spalten können entweder vollständig oder nur der durch die Cursors markierte Zeitbereich kopiert werden. Mit zwei Optionen lässt sich steuern, ob auch der Tabellenkopf und die Zeitspalte enthalten sein sollen.

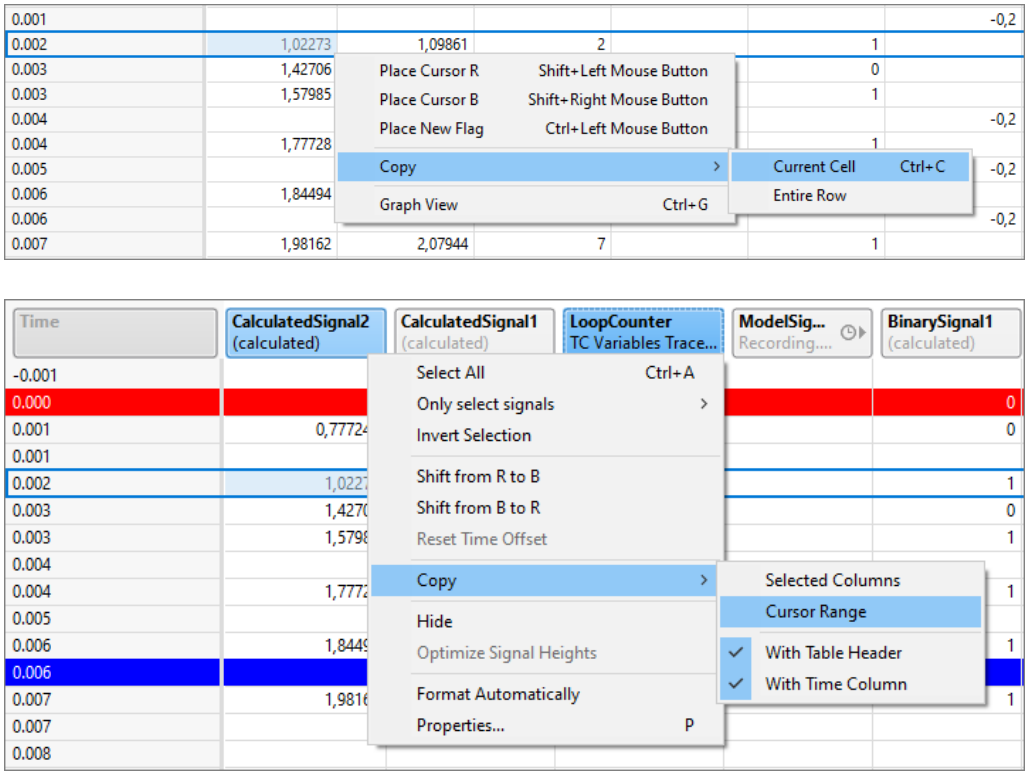


Abbildung 47: Kopieren von Signaldaten aus Tabellen per Rechtsklick auf Zellen (oben) oder den Tabellenkopf (unten)

5.5 Weitere Testaspekte

Optimierung der ecu.test Linux Runner-Installer

et

Das DEB-Paket und das UBI9-Image wurden maßgeblich verkleinert, sodass sie sowohl beim Download als auch in der Installation deutlich weniger Platz benötigen. So wurden u. a. die darin abgelegte Hilfe sowie weitere, für die Konsolenausführung irrelevante Anteile entfernt. Die Kompatibilität bleibt vollständig gewährleistet.

Standardeinheit für Zeiterwartungen via Workspace-Einstellungen

et

In den Workspace-Einstellungen kann die Standardeinheit für Zeiten von neu angelegten Testschritten konfiguriert werden. Somit kann für die wichtigsten Stellen im Testfall, wie z. B. Zeiterwartungen und den Wait-Testschritten, eine Eingabe in Sekunden statt Millisekunden vorgesehen werden.

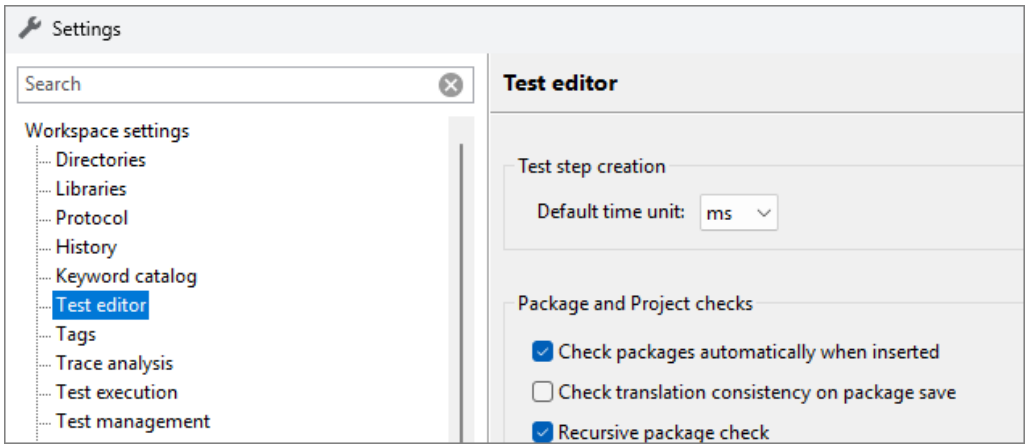


Abbildung 48: Workspace -Einstellungen > Test-Editor – Einstellen einer Standard-Zeiteinheit

Aufgezeichnete Aufnahmen standardmäßig nach Signalgruppe benennen

et

Achtung: Der Standardname für erzeugte Aufnahmedateien wurde geändert!

Die Aufnahmen werden nun **nach ihrer Signalgruppe** benannt, statt anonym durchnummeriert zu werden. So lassen sich insbesondere Busaufzeichnungen wie **A-CAN.asc** deutlich einfacher im Dateisystem identifizieren.

Verbesserungen Default-Aufnahmekonfiguration

et tc

Die Default-Aufnahmekonfiguration ermöglicht eine zentrale Verwaltung von Signal- und Aufnahmegruppen und deren Einstellungen. Dadurch werden die eigentlichen Testfälle deutlich übersichtlicher und sind schneller zu erstellen.

Die Anwenderhilfe wurde erweitert und geht nun ausführlich auf die Konfigurationsmöglichkeiten ein. Dabei helfen sowohl eine Feature-orientierte Tabelle als auch anwendungsfallbezogene Beispiele.

Auch die Verwendung selbst wurde weiter verbessert. So werden Signale standardmäßig der **Automatisch zuordnen**-Gruppe hinzugefügt, falls eine Default-Aufnahmekonfiguration geladen ist (statt ggf. individuelle Signalgruppen zu erstellen).
Außerdem wird eine aktivierte **Tracezusammenführung** nun übernommen und die **Packageprüfung** berücksichtigt nun die Default-Aufnahmekonfiguration korrekt.

Standard-Parametersatzgenerator für x Ausführungen

et tc

Ein neuer Parametersatzgenerator ermöglicht es, einem Package eine gewünschte Anzahl von Wiederholungen auszuführen.

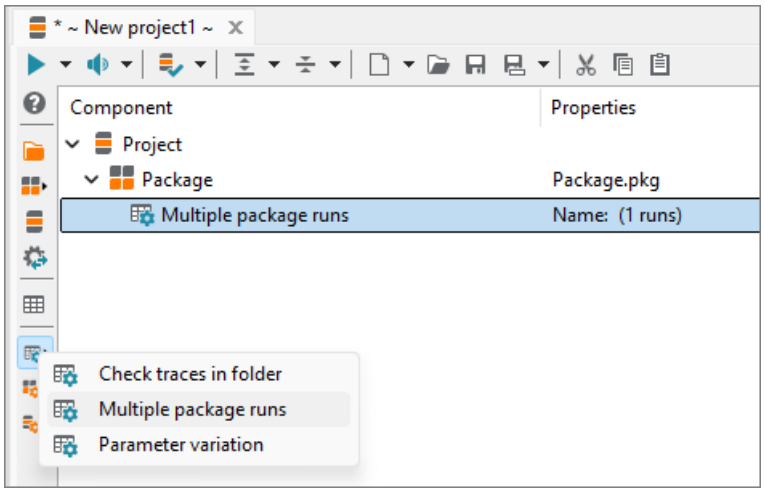


Abbildung 49: Parametersatzgenerator zur Einstellung einer bestimmten Anzahl an Wiederholungen

6 Versionen und Schnittstellen

6.1 Neue Tools und Versionen

et

	Provider	Web-seite	System	Produktname	Version
1	**IPG**	Release-Link	Die Simulations-lösung für den virtu-ellen Fahrversuch	**CarMaker**	**15**
2	**MathWorks**	Release-Link	Plattform für Pro-grammierung und numerische Berech-nungen	**MATLAB/ Simulink**	**R2025b unter Linux**
3	**Vector**	Release-Link	Offene Simulation-sumgebung für vir-tuelle Testfahrten von PKW und Nutz-fahrzeugen.	**DYNA4**	**R9 und R10**

6.2 APIs

6.2.1 Object API

Neue Methoden zur Manipulation der pkgattr.spec und prjattr.spec

et

Bisher konnten Package- und Projekt-Attributspezifikationen nur durch manu-elle Änderungen an den Dateien angepasst werden. Für benutzerdefinierte Testmanagement-Adapter ist jedoch eine automatische Ableitung und Ände-rung der Attribute aus dem ALM erforderlich.

Die neue Object API **AttributeSpecApi** ermöglicht das programmgesteuerte Lesen, Ändern und Speichern von Attributen. Dies erlaubt eine dynamische Synchronisation mit dem ALM, reduziert manuelle Aufwände und erhöht die Konsistenz der Test-Daten.

6.2.2 REST API

Neue Endpunkte für die Testfallgenerierung mit ecu.test *agent*

Für die Generierung von Testfällen via des **ecu.test** *agents*, stehen jetzt neue REST-API-Endpunkte zur Verfügung.
Generierungen lassen sich einreihen und deren Generierungsstatus abfragen.

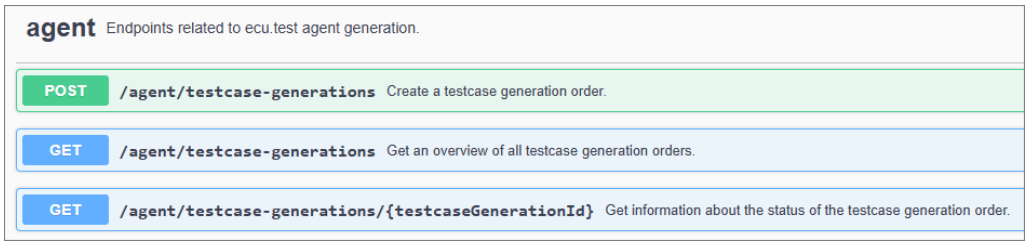


Abbildung 50: Neue REST-API-Endpunkte für den ecu.test *agent*

7 Abkündigungen

7.1 Abkündigungen und Inkompatibilitäten in dieser Version

KS: Tornado nur noch über ASAM ACI

et

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

Interface TmUserAdapter

et

Die folgenden Interface-Methoden wurden entfernt:

- GetPackageFromTms
- GetProjectsFromTms

Als Ersatz dienen diese neu eingeführten Methoden:

- **FetchChildrenForPackageImport**
- **FetchChildrenForProjectImport**

Jama Connect

et

Der integrierte Test Management Adapter für **Jama Connect** wurde entfernt. Als Ersatz können benutzerdefinierte **Test Management Adapter** eingesetzt werden. Diese Adapter bieten deutlich mehr Flexibilität und ermöglichen eine optimale Anpassung an den jeweiligen Workflow.

Als Hilfestellung für die eigene Implementierung wird ein Muster-Workflows bereitgestellt.

Jobs RequestSeed und SendKey

et

Um den UDS Service Security Access mit der **SeedAndKey-DLL** nativ in **ecu.test** zu unterstützen, wurden die veralteten Jobs *RequestSeed* und *SendKey* ersetzt.

- RequestSeed → **SecurityAccessRequestSeed**
- SendKey → **SecurityAccessSendKey**

Die neuen Namen entsprechen nun besser der UDS-Spezifikation.

Kompatibilität
Die neuen Jobs verfügen über dieselben Funktionen wie die alten. Sie unterstützen ebenfalls **SeedAndKey**-DLLs.

Auswirkungen
Wer die alten Jobs noch verwendet, muss auf die neuen umsteigen.

Die Option „Port starten: NIE“ wurde entfernt

et

In der TBC gab es eine Startoption „**Nie**“ für einen Port, um zu verhindern, dass der Port beim Start des Testfalls aktiviert wird. In der Vergangenheit hat diese Option zu einiger Verwirrung geführt.

Kompatibilität
Da es keinen bekannten Anwendungsfall gibt, bei dem explizit entschieden wird, einen Port nicht zu starten, wurde die Option entfernt. Konfigurationen, die diese Option verwendet haben, werden nun automatisch auf die Startoption **IF NECESSARY** aktualisiert.

Auswirkungen
Die Änderung sollte keine spürbaren Auswirkungen haben. Es wird empfohlen, immer die Startoption **IF NECESSARY** zu verwenden.

7.2 Abkündigungen in zukünftigen Versionen

Tektronix UTA 12

et

Das Tool **Tektronix UTA 12** ist nicht mehr auf dem Markt und wird zu **ecu.test 2026.4** abgekündigt.

QUANCOM

et

Der API-Schnittstelle **QLIB** für **Quancom** ist nicht mehr kompatibel zu aktuellen Geräten und wird zu **ecu.test 2026.4** abgekündigt.

Fibex-Unterstützung

et

Die Fibex-Unterstützung für Bus wird abgekündigt und soll mit **ecu.test 2026.3** entfernt werden. Die Fibex-Unterstützung für DLT bleibt weiterhin erhalten.

TRF-Dateien lassen sich für Projektreporte nicht mehr erzeugen

et

Im Zuge der Einführung des PRF-Formats für Projektreporte ist das TRF-Format obsolet. War es bisher während der Migrationsphase noch möglich, über die Workspace-Einstellungen das TRF-Format zu aktivieren, wird diese Fallback-Einstellung voraussichtlich zu **ecu.test 2026.3** entfernt. Bis dahin wird basierend auf erhaltenem Feedback an einer Verbesserung einiger Workflows mit und ohne **test.guide** gearbeitet.

Diese Abkündigung betrifft ausschließlich Projektreporte. Package-Ausführungen werden weiterhin im TRF-Format gespeichert.

Import im Zusammenhang mit CustomChecks

et

Eigene Prüfungen verwenden unter Umständen ein Modul, das umgezogen ist. Daher müssen die Importe geprüft und ggf. angepasst werden. Die Abkündigung erfolgt mit **ecu.test 2026.3**.

- **alt: tts.core.common.check.Constants**
- **neu: tts.interface.customCheck.Constants**

Echtzeit-Packagereferenzen

et

dSPACE hat mit **ControlDesk 2024-B** ihre Schnittstelle zur Ansteuerung der Echtzeitumgebung geändert. Seit diesem dSPACE-Release ist die Ausführung von Echtzeit-Packagereferenzen mit **ecu.test** nicht mehr möglich.

Aufgrund des ausgebliebenem Anwenderfeedbacks zu diesem Umstand nehmen wir an, dass die Funktionalität weitestgehend ungenutzt ist und entfernen sie mit **ecu.test 2026.3**.

Achtung: Die Abkündigung bezieht sich lediglich auf Echtzeit-Packagereferenzen. Weitere Features rund um das Thema Real Time Testing (RTT) sind nicht betroffen.

Bus-Monitoring-Testschritte

et

Die Bus-Monitoring-Testschritte zum Prüfen von **Timings** und **DLC** können deutlich besser mit den Mitteln der Traceanalyse und den mitgelieferten Analysebausteinen umgesetzt werden.

Ab **ecu.test 2026.3** wird das Einfügen neuer Testschritte nicht mehr unterstützt. Die Unterstützung der Ausführung wird frühestens in **ecu.test 2027.1** entfernt.

Basler: Pylon

et

Die Kameras der Firma Basler sind kompatibel zum GenICam Standard, was eine eigene Toolanbindung unnötig macht. Deshalb wird das Tool **Basler: Pylon** zur **ecu.test 2026.3** entfernt.

Bei Verwendung einer Kamera von Basler kann alternativ die GenICam-Toolanbindung verwendet werden.

Siemens Polarion

et

Der integrierte Test Management Adapter für **Siemens Polarion** wird mit **ecu.test 2027.1** entfernt.

Als Ersatz können benutzerdefinierte Test Management Adapter eingesetzt werden. Diese Adapter bieten deutlich mehr Flexibilität und ermöglichen eine optimale Anpassung an den jeweiligen Workflow.

Als Hilfestellung für die eigene Implementierung wird ein Muster-Workflow bereitgestellt.

IBM Rational Quality Manager (RQM)

et

Der integrierte Test Management Adapter für **IBM Rational Quality Manager (RQM)** wird mit **ecu.test 2027.1** entfernt.
Als Ersatz können benutzerdefinierte Test Management Adapter eingesetzt werden. Diese Adapter bieten deutlich mehr Flexibilität und ermöglichen eine optimale Anpassung an den jeweiligen Workflow.

Als Hilfestellung für die eigene Implementierung wird ein Muster-Workflow bereitgestellt.

MotionDesk

et

Die Anbindung dSPACE: MotionDesk wird in **ecu.test 2026.3** entfernt, da kein offizieller Support mehr seitens dSPACE besteht. Als Ersatz steht Aurelion zur Verfügung.

HP Application Lifecycle Management (ALM)

et

Der integrierte Test Management Adapter **für HP Application Lifecycle Management (ALM)** wird mit **ecu.test 2027.1** entfernt.

Um nach dieser Version weiterhin mit HP Application Lifecycle Management (ALM) interagieren zu können, muss ein benutzerdefinierter Testmanagementadapter mithilfe der **ecu.test** Test Management API implementiert werden.

PTC Integrity

et

Der integrierte Test Management Adapter für **PTC Integrity** wird mit **ecu.test 2027.1** entfernt.

Um nach dieser Version weiterhin mit PTC Integrity interagieren zu können, muss ein benutzerdefinierter Testmanagementadapter mithilfe der **ecu.test** Test Management API implementiert werden.

Interaktive Testausführung in ecu.test

et

Die Anwendungsfälle der interaktiven Testausführung innerhalb von **ecu.test** und der Web-Applikation **ecu.test** *drive* überlappen sich größtenteils.

ecu.test *drive* ist die deutlich flexiblere Lösung aufgrund der

- intuitiveren Nutzerführung,
- besseren Anpassbarkeit an die Anwendungsgebiete im Fahrzeug,
- Verwendbarkeit mit **ecu.test** Runner und Remote durch Entkopplung von **ecu.test**-GUI

Daher konzentrieren wir uns in Zukunft auf diese Lösung und werde die interaktive Testausführung mit **ecu.test 2027.1** entfernen.