

# Release Notes

ecu.test 2026.1

trace.check 2026.1

Datum: 03.03.2026  
© 2026 tracetronic GmbH

tracetronic GmbH  
Stuttgarter Str. 3  
01189 Dresden  
[www.tracetronic.de](http://www.tracetronic.de)

## Inhalt

Überblick .....	1
1 Highlights in ecu.test 2026.1 .....	2
2 Sneak Preview .....	6
3 ecu.test extras .....	12
3.1 ecu.test <i>agent</i> .....	12
3.2 ecu.test <i>calibration</i> .....	13
3.3 ecu.test <i>drive</i> .....	13
3.4 ecu.test <i>lab</i> .....	13
4 Usability .....	17
5 Testaspekte .....	18
5.1 HiL .....	18
5.2 Testmanagement .....	18
5.3 Traceanalyse .....	19
5.4 Weitere Testaspekte .....	20
6 Versionen und Schnittstellen .....	21
6.1 Neue Tools und Versionen .....	21
7 Abkündigungen .....	22
7.1 Abkündigungen und Inkompatibilitäten in dieser Version .....	22
7.2 Abkündigungen in zukünftigen Versionen .....	23

# Überblick

**ecu.test 2026.1** treibt moderne Testprozesse für SDV-, Cloud- und KI-Szenarien weiter voran: Plattformunabhängigkeit (Linux, ARM), deutlich schnellere Infrastrukturen, KI-Assistenz im Alltag sowie zahlreiche Verbesserungen bei der Integration, Analyse und Visualisierung von Testdaten. Die wichtigsten Neuerungen im Schnelldurchlauf:

## Umgebungen & Infrastruktur

- **ecu.test Linux GUI**  
Vollständige grafische Oberfläche unter Linux: Testen ohne Windows-Umweg
- **ecu.test Linux Runner & Runner für ARM (Sneak Preview)**  
Effiziente Testausführung auf Linux- und ARM-Ressourcen

## KI & Assistenz

- **ecu.test agent**  
Der Agent ist jetzt zur Generierung von Traceschritten verwendbar.
- **ecu.test agent chat (Preview)**  
Geplanter Copilot für Workflows: Mustererkennung, Reviews, Bibliothekspackages und Zusammenarbeit mit dem **test.spec agent**.
- **test.spec agent (Preview)**  
Generiert aus Requirements Testspezifikationen.
- **TouchInput mit KI-Prompt**  
HMI-Interaktionen per Sprach-Prompt statt Koordinaten.

## Analyse & Diagnose

- **CheckResponse**  
Die Funktion erleichtert Zeitprüfungen in der Traceanalyse.
- **J1939 DM-Services (Preview)**  
Symbolische, generische Diagnose-Testschritte analog OBDOnUDS, ohne zusätzliche Datenbasis.

## Performance im Alltag

- **Paralleler Toolstart** verkürzt die Aufstartzeit umfangreicher TBCs.
- **CAN-FD Timing Helper** ermittelt fehlende Timing-Parameter automatisch und erleichtert so die CAN-FD-Konfiguration.
- **ecu.test code**
  - wird ab sofort über [PyPI](#) bereitgestellt.
  - ermöglicht jetzt direkte Package-Aufrufe.

Alle weiteren zahlreichen Neuerungen und Erweiterungen werden nachfolgend detailliert beschrieben.

**Hinweis:** Die Icons zeigen, für welches Produkt ein Thema relevant ist:

 **ecu.test**  **trace.check**

# 1 Highlights in ecu.test 2026.1

## ecu.test Linux GUI



**ecu.test** steht neben Windows ab sofort auch unter Linux mit vollständiger grafischer Benutzeroberfläche zur Verfügung. Tests können damit direkt in der gewohnten Linux-Umgebung erstellt, ausgeführt und ausgewertet werden, ohne den Umweg über Windows für Anpassungen an Tests und Konfigurationen gehen zu müssen.

Die neue **ecu.test Linux GUI** nutzt **Ubuntu 24.04** und vereinfacht zusammen mit dem **ecu.test Linux Runner** die Entwicklung und Absicherung moderner SDVs. So lässt sich die vorhandene Linux-Infrastruktur effizienter nutzen und es eröffnen sich neue Anwendungsfälle – bis hin zu remote betriebenen, im Browser gestreamten Systemen.

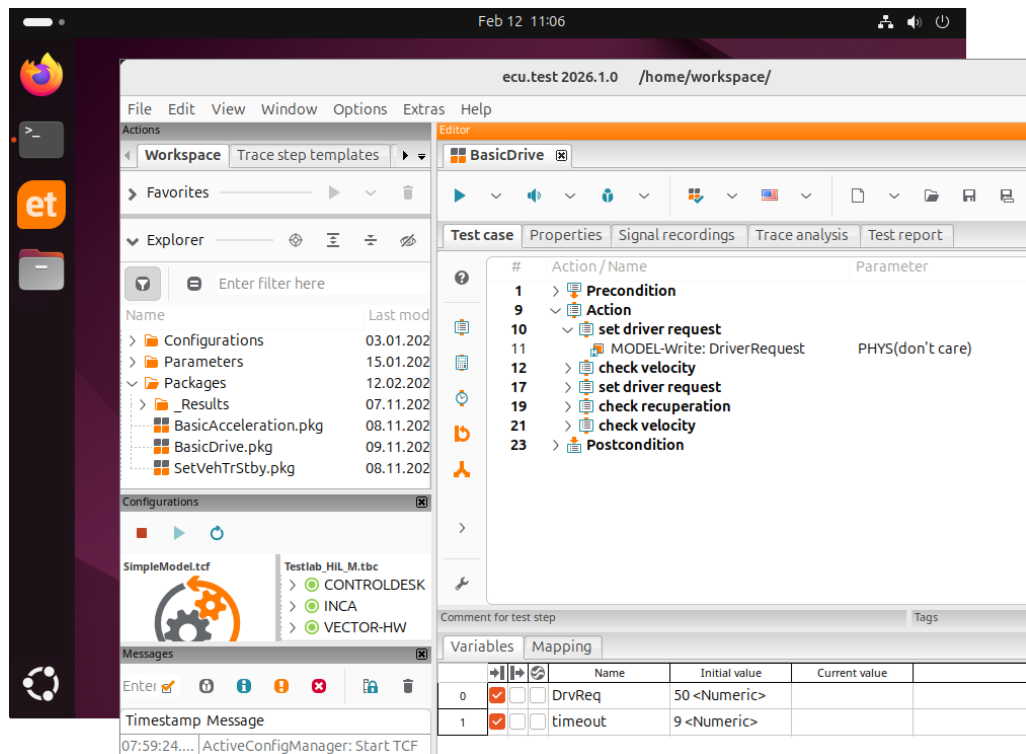


Abbildung 1: ecu.test Linux GUI in Ubuntu 24.04

## CAN-FD Timing Helper



Das neue Utility **CAN-FD Timing Helper** vereinfacht die Testbenchkonfiguration (TBC) für CAN-FD-fähige Bushardware.

Auf Basis der parametrisierten Bitraten, Sample Points und der Oszillator-Frequenz werden die fehlenden Parameter – **Synchronization-Jump** und **Zeit-segmente** – automatisch ermittelt und visualisiert.

Die manuelle Parametrierung in der TBC mit abweichenden Werten ist optional weiterhin möglich, aber nicht mehr zwingend erforderlich.

Zusätzlich können einmal ermittelte Werte per Knopfdruck vollständig kopiert und auf anderen Ports eingefügt werden.

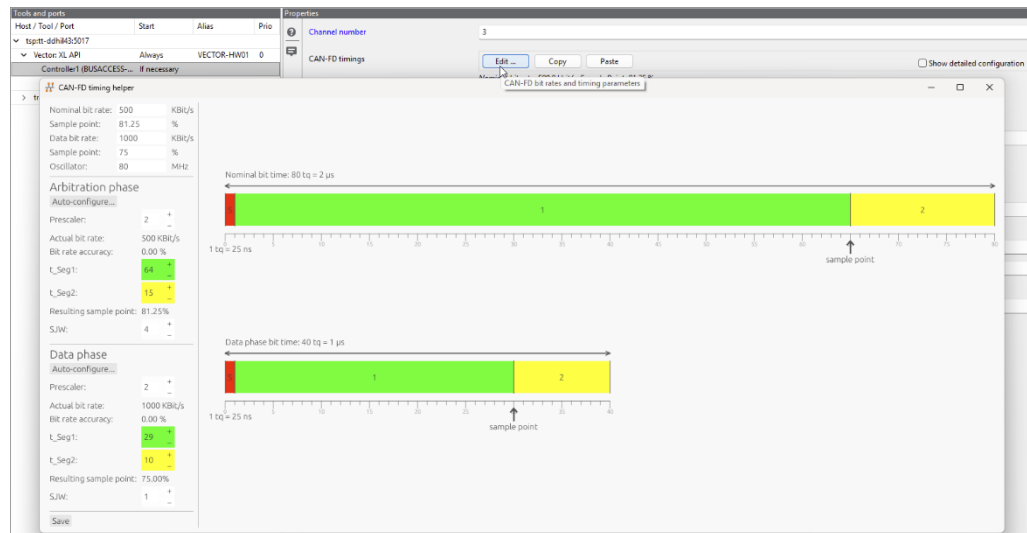


Abbildung 2: CAN-FD Timing Helper

## Paralleler Toolstart



Der Betrieb der Testinfrastruktur ist mit hohen Kosten verbunden. Zeiten, in denen die Testbench und deren Tools konfiguriert werden, müssen von der kostbaren Testzeit abgezogen werden und tragen, wenngleich der Konfigurationsstart nötig ist, nicht zu den eigentlichen Tests bei.

Insbesondere bei umfangreichen TBCs mit vielen Tools, kann sich der Start der Konfiguration hinziehen. Die Aufstartzeit eines Tools hängt dabei stark von den benötigten Konfigurationsschritten, dem Umfang der Konfiguration und der individuellen Implementierung des Tools und dessen API ab.

Bisher wurden jegliche Tools in einer TBC sequenziell gestartet. Mit **ecu.test 2026.1** werden die Tools nun parallel gestartet, was substantielle Zeitersparnisse mit sich bringt.

Die individuell gesetzte Aufstartpriorität jedes Tools, wird dabei weiterhin beachtet.

### ecu.test code ab sofort auf pypi.org



**ecu.test** code wird ab diesem Release über den Python Package Index [PyPI](#) bereitgestellt. Sofern PyPI erreichbar ist, erfolgt die Installation ab sofort ganz einfach via:

- `pip install ecutest_code`

Die Bereitstellung via E-Mail und Seafile-Link wird gleichzeitig eingestellt.

### Package-Ausführung für ecu.test code



Für bestimmte Aufgaben sind evtl. schon vorhandene Hilfspackages im Einsatz, oder die Teilaufgabe lässt sich prägnanter als Package implementieren.

Ab sofort können in **ecu.test** code Packages, die bestimmte Einschränkungen einhalten, aufgerufen werden. Dabei können Parameter übergeben und Rückgabewerte ausgelesen werden. Außerdem kann die Bewertung der Package-Ausführung (SUCCESS, FAILED, etc.) abgefragt werden.

Selbstverständlich können Package-Aufrufe auch mit Zugriffen auf Testgrößen kombiniert werden.

```
pkg = ta.Package("my package.pkg")
with ta.run():
    pkg_result = pkg.run({'param_A': 3.0})
    assert pkg_result.result_code == ExecutionResult.SUCCESS
    print(pkg_result.return_values)
```

Abbildung 3: Package-Aufruf in ecu.test code

## CheckResponse erleichtert Zeitprüfungen in der Traceanalyse



Innerhalb der Traceanalyse war es bisher schwierig, die Prüfung typischer zeitlicher Zusammenhänge zwischen zwei Signalen ohne Verwendung von Trace-schrittvorlagen zu modellieren.

Für Berechnungsschritte und Triggerblöcke steht nun die Funktion **CheckResponse** zur Verfügung. Sie vereinfacht die zeitliche Prüfung deutlich, indem nur zwei Signale und ein *timeout*-Parameter angegeben werden.

Ist der Wert des ersten Signals zu einem Zeitpunkt wahr, prüft **CheckResponse**, dass innerhalb der Zeit *timeout* auch das zweite Signal wahr wird. Gute Praxis ist es, für das erste Signal eine Edge-Funktion anzugeben, da meist der Zustandswechsel eines Signals als Aktivierungsbedingung dient:

- **CheckResponse(RisingEdge(Sig1), Sig2 == 1, 0.5)**

Durch den optionalen Parameter **useHoldValue** lassen sich auch spezielle Fälle abdecken, bei denen das zweite Signal selten abgetastet wird und sich eventuell schon vor dem Aktivierungszeitpunkt im gewünschten Zustand befunden hat.

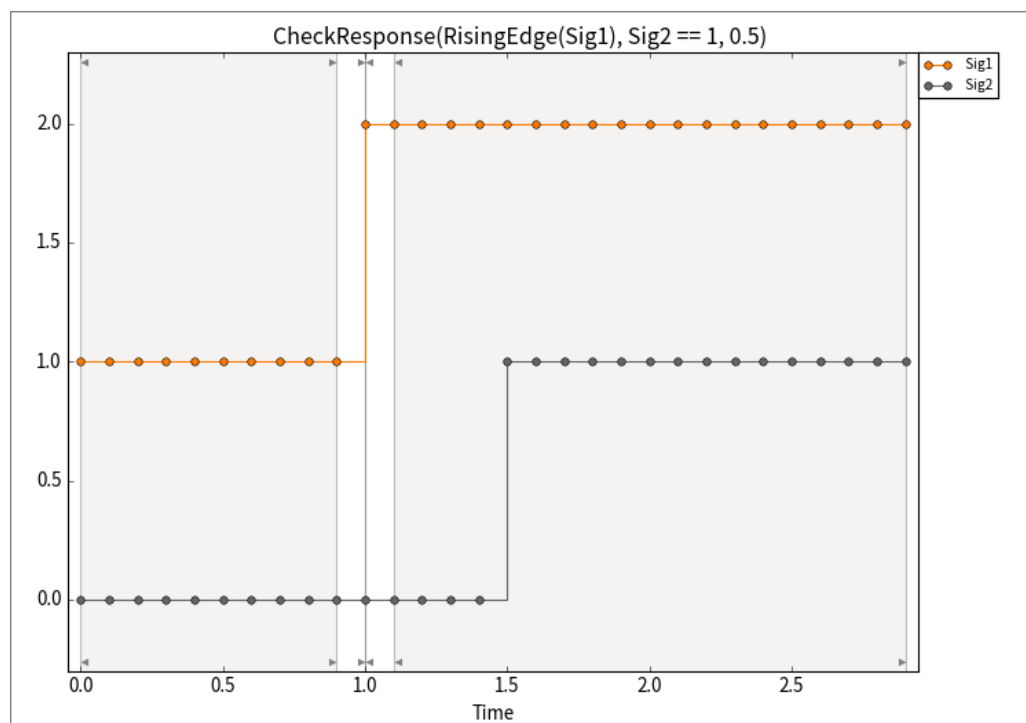


Abbildung 4: Überprüfung an der steigenden Flanke von Sig1, ob sig2 innerhalb von 0,5 s den Wert 1 hat.



Weitere Use Cases bzw. Einsatzszenarien:

- Ausführung von Package-Checks und Generatoren
- Durchführung von Traceanalysen und Dateikonvertierungen
- Betrieb auf Embedded-Hardware für dedizierte Test- oder Analyseaufgaben oder zur Prüfstandsautomatisierung/-überwachung (z. B. via **ecu.test lab**)

### ecu.test agent chat - dein ecu.test Copilot



In den vergangenen beiden Releases haben wir den **ecu.test agent** eingeführt und seitdem kontinuierlich verbessert. Parallel arbeiten wir an einem nächsten großen Highlight – dem **ecu.test agent chat**, eine Art Copilot für **ecu.test**.

Mit diesem neuen Chat kann sich der Nutzer zukünftig bei den verschiedensten Workflows unterstützen lassen. Neben den bisherigen Funktionalitäten für die Testfall- und Traceanalysegenerierung sind folgende neue Workflows bzw. Generierungen denkbar:

- Finden von gleichartigen Implementierungen in Testfällen und automatisierte Ableitung von Bibliothekspackages
- Unterstützung der Signalaufnahmen für die optimierte Zusammenführung der Testfall- und Traceanalysegenerierung
- Review eines Testfalls: Der Agent weist auf Unzulänglichkeiten und auch Unterschiede zwischen Spezifikation und Implementierung hin
- Aktionen im Rahmen eines **ecu.test**-Projektes: bspw. das Hinzufügen von Parametersätzen aus angehängenen Files
- Kommunikation mit dem **test.spec agent**, sodass neue Spezifikationen aus Requirements abgeleitet werden können

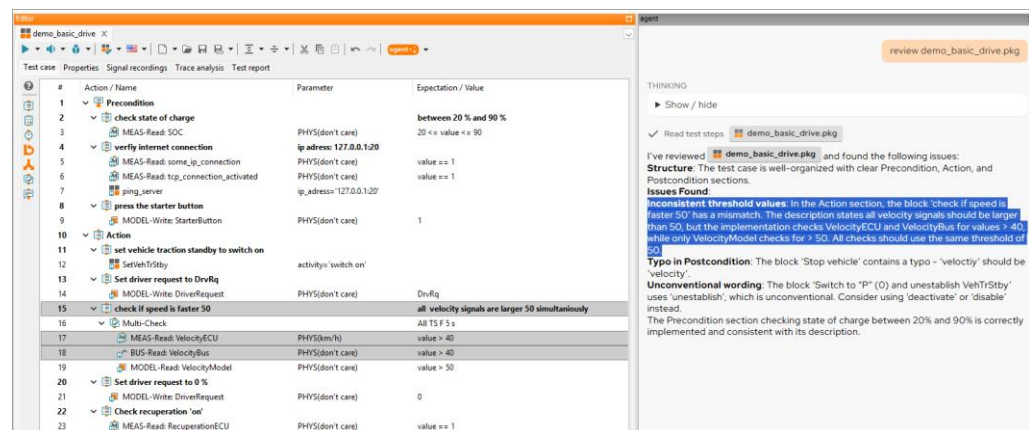


Abbildung 6: Kommunikation via ecu.test agent chat

Diese Aufzählungen sind Beispiele. Lass uns gemeinsam über deine Workflows sprechen. Wo geht bei dir die meiste Zeit verloren?

In **ecu.test 2026.1** ist der **ecu.test agent chat** bereits auf Anfrage erlebbar. Bitte sprich uns dazu über [support@tracetronic.com](mailto:support@tracetronic.com) gern an.

### test.spec agent



Mit dem aktuellen Release wurde der **test.spec agent** konsequent weiterentwickelt. Der Fokus lag auf Qualitätsverbesserungen des gesamten Agentic Workflows: Spezifikationen werden noch konsistenter erzeugt, Testmethodiken wie Positiv-/Negativtests und Äquivalenzklassen werden automatisch berücksichtigt, und Regeln für die Spezifikationsgenerierung lassen sich gezielt definieren und selektiv anwenden.

Neu ist zudem eine beispielhafte Anbindung an Codebeamer, sodass sich Requirements aus dem ALM-System direkt in den Spezifikationsprozess integrieren lassen.

#### Ausblick:

- geplantes Release Ende Juni
- engere Interaktion des **test.spec agent** mit dem **ecu.test agent** zur durchgängigen Testfallgenerierung
- Editor-Optimierungen für die effiziente Pflege von Spezifikationen
- erste Funktionen zur Unterstützung der Traceanalyse und Qualitätsbewertung von Anforderungen und Spezifikationen

Nutze die Preview-Phase, um den **test.spec agent** in deiner eigenen Toolandschaft zu evaluieren und erlebe, wie sich der Aufwand beim Erstellen der Spezifikationen von Stunden auf Minuten reduziert.

Wir unterstützen dich gerne bei den ersten Schritten und freuen uns auf deine [Anfrage!](#)

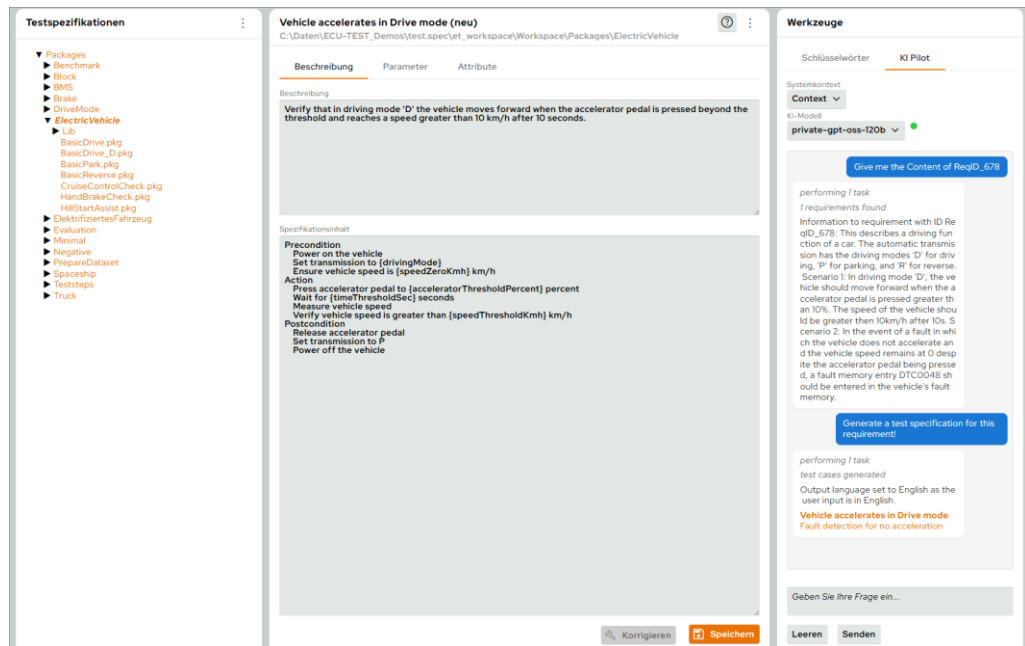


Abbildung 7: test.spec agent

## TouchInput mit KI-Prompt



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.

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.

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 4.5 und ChatGPT gemacht.

Wenn du das neue Feature bereits jetzt ausprobieren möchtest, melde dich bei deinem tracetrionic-Ansprechpartner oder bei unserem Support!

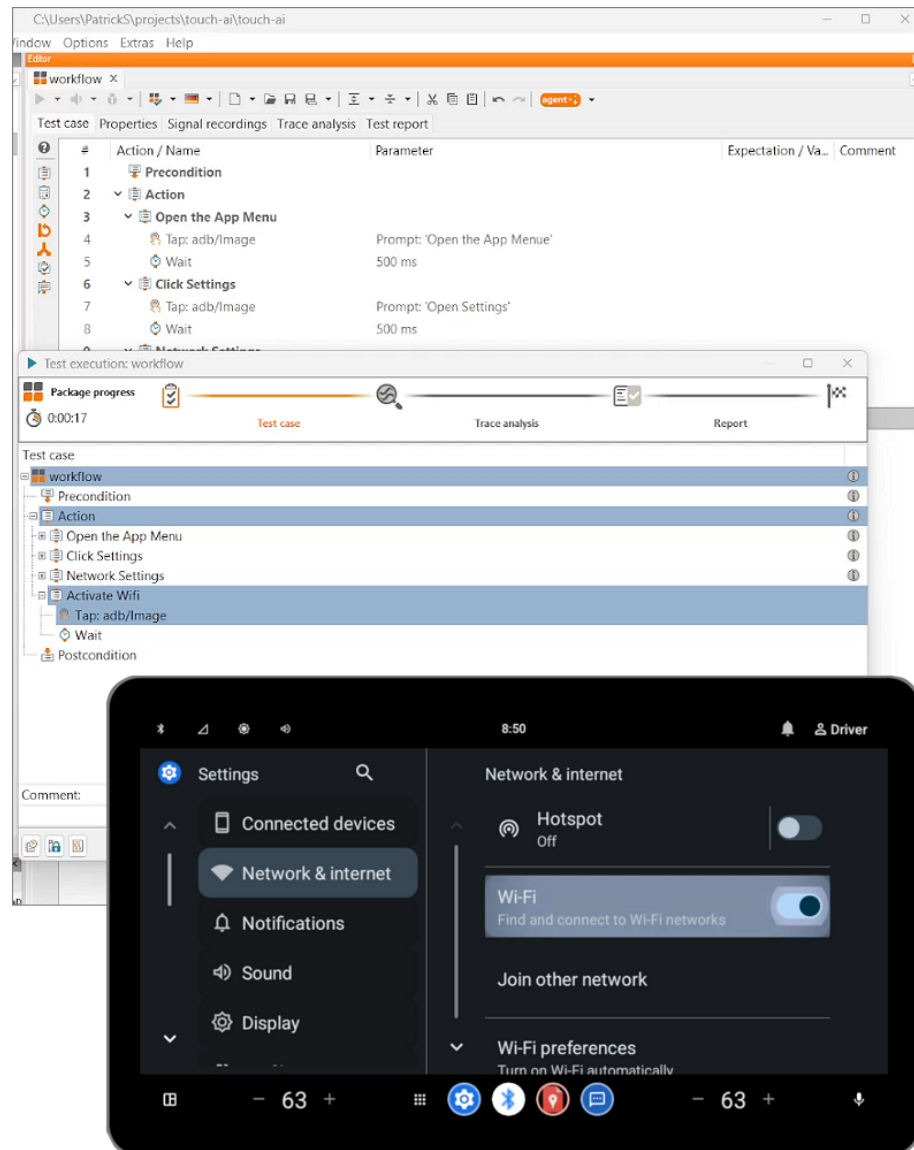


Abbildung 8: Funktion Touchinput mit KI-Prompt



## 3 ecu.test extras

### 3.1 ecu.test agent

#### Unterstützung für die Traceanalyse



Nachdem im letzten Release die Generierung von Traceschritten als Vorschau verfügbar war, steht diese Funktion nun offiziell zur Verfügung.

Der **ecu.test agent** erzeugt jetzt auf Wunsch auch Inhalte für Blöcke. Unterstützt werden aktuell Triggerblöcke, Berechnungsschritte und Plots.

Analog zum Testfall werden Blöcke mit generierten Inhalten mit einem kleinen Symbol markiert:

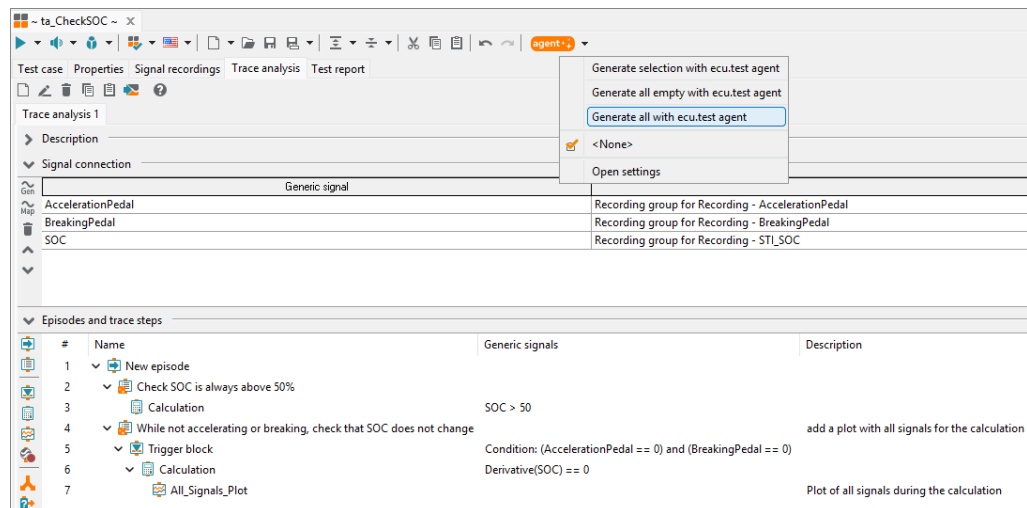


Abbildung 10: Generierung von Traceschritten via ecu.test agent

#### Optimierungen und Unterstützung weiterer Testschritte



Im aktuellen **ecu.test**-Release 2026.1 wurden zahlreiche Optimierungen für die Generierung von Testfällen vorgenommen. Dazu zählen unter anderem:

- Werte von Enumerations werden bei der Generierung berücksichtigt.
- Aktuell nicht unterstützte Testschritte aus Referenzimplementierungen werden als ToDo-Schritt erzeugt.
- Die Zeitoptionen "Wait until true", "Wait for at least" und die Option "Polling cycle" werden bei der Generierung einbezogen.
- Die Generierung von Tool- bzw. Portjobs wurde optimiert.

Weiterhin unterstützt der **ecu.test agent** die folgenden Testschritte:

- Exit
- Comment

## 3.2 *ecu.test calibration*

### Gleichzeitige Aufnahme eines Bus- und eines Kalibrierungs-Ports



Häufig müssen Aufzeichnungen für die Bus- und XCP-Kommunikation über denselben Controller der Messhardware erfolgen. Dies war bislang möglich und führte zu einer Fehlermeldung. Ab diesem Release gilt diese Einschränkung nicht mehr.

## 3.3 *ecu.test drive*

### Shortcuts zum Pausieren und Abbrechen von Testfällen



Die Ausführung eines Tests kann durch Drücken der Leertaste unterbrochen und ein Testfall durch Drücken der Esc-Taste abgebrochen werden.

## 3.4 *ecu.test lab*

### Belegung aller Parameter eines Packages



Bisher konnte in schreibenden Widgets für ein hinterlegtes Package nur ein einziger Parameter belegt werden. Mit dieser Version können nun alle Parameter belegt werden.

In den meisten Fällen hat das zur Folge, dass ein Parameter dynamisch durch die Bedienung des Widgets belegt wird, und die restlichen Parameter entweder mit statischen Werten belegt, oder bewusst mit dem im Package hinterlegten Initialwert ausgeführt werden.

Bei lesenden Widgets gab es bei Packages bisher nur die Möglichkeit zur Auswahl des Rückgabewertes. In diesem Fall können die Parameter nun auch statisch konfiguriert werden.

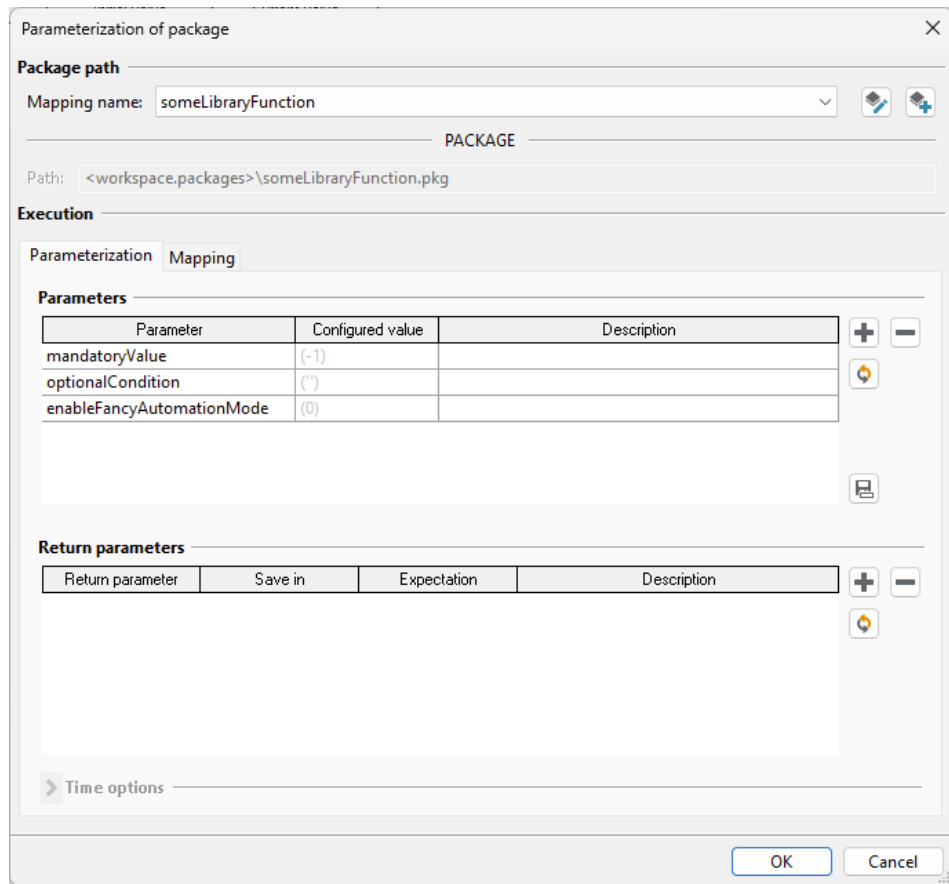


Abbildung 11: Parametrierung eines Packages in ecu.test

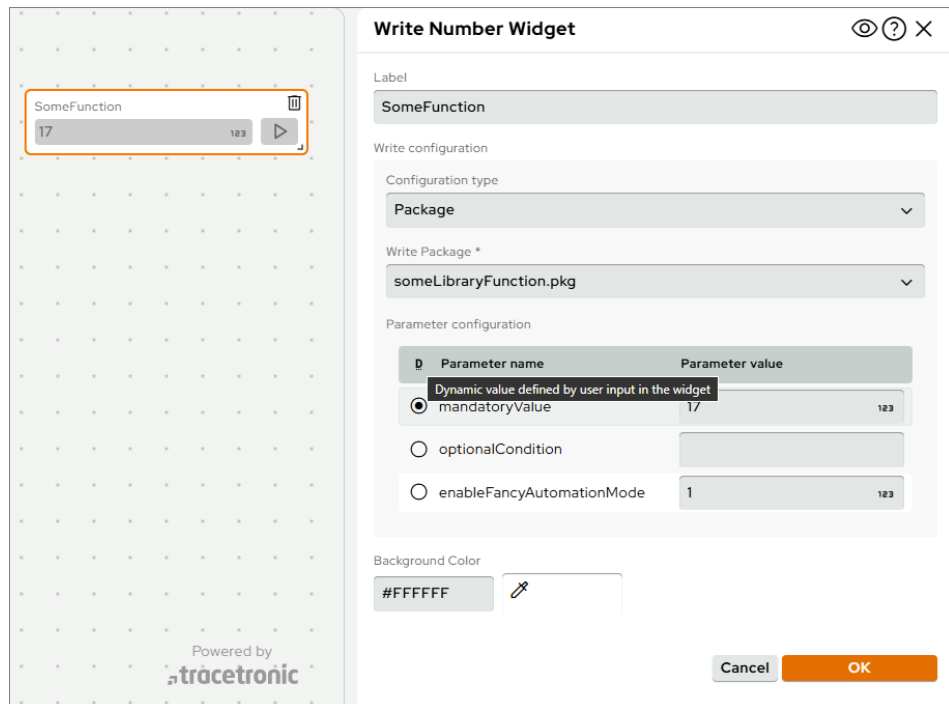


Abbildung 12: Parametrierung aller Packages in ecu.test lab

## Neues Widget: Signal Graph



Seit **ecu.test lab** existiert, gibt es den Wunsch nach einer Darstellung von Signalverläufen. Mit **ecu.test 2026.1** und unserem neuen Widget **Signal Graph** wurde dieser jetzt umgesetzt.

Das Widget basiert auf der Signalvisualisierung des **trace.xplorers** und kann bis zu vier Signale in einem Graphen anzeigen. Dabei kann konfiguriert werden, ob diese sich eine gemeinsame Y-Achse teilen oder jeweils eine eigene Achse erhalten.

Optional können Marker für Signalwerte und eine interpolierte Kurvendarstellung aktiviert werden.

Während der Ausführung lassen sich mit einem Klick in den Graphen die Signalwerte an der Cursorposition anzeigen. Zum besseren Überblick können Signale temporär ausgeblendet werden. Außerdem kann das Zeichnen des Signalverlaufs bei Bedarf pausiert und später (ohne Verlust von Werten) wieder fortgesetzt werden.

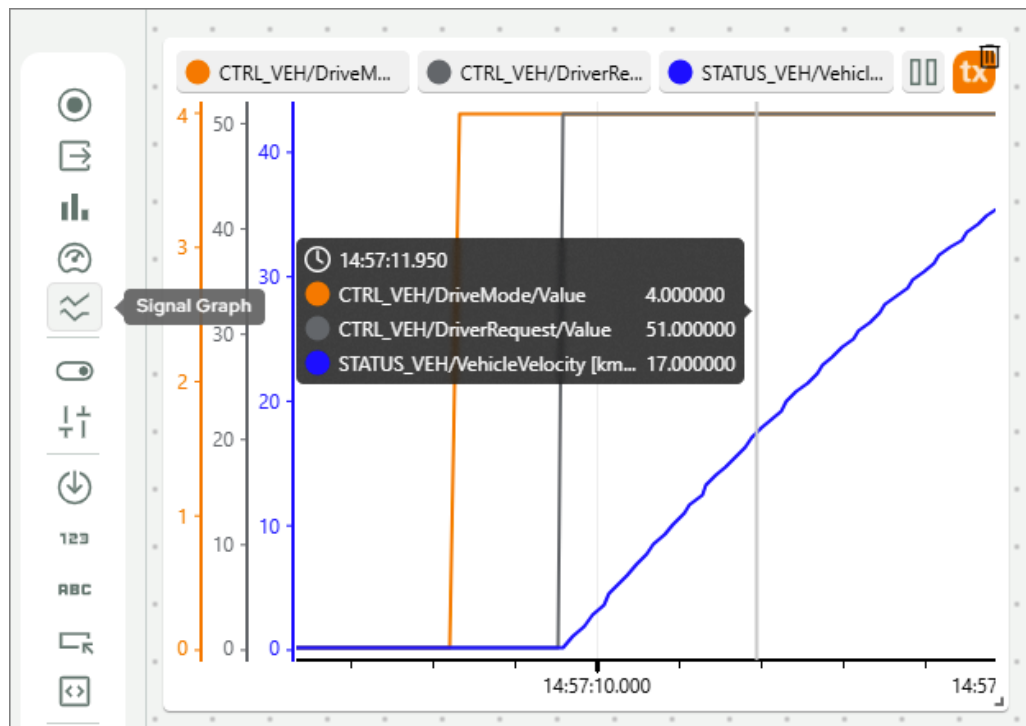


Abbildung 13: Neues Widget zur Darstellung von Signalen

## Mehrere Pages pro View



Um die verfügbare Arbeitsfläche zu vergrößern und besser strukturieren zu können, wurden Tabs, bzw. Pages zu den Views hinzugefügt.

So können mehrere Seiten mit Widgets befüllt und gleichzeitig initialisiert und verwendet werden.

Ein Stoppen des Monitoring-Modus, sowie der Wechsel der View entfallen dadurch.

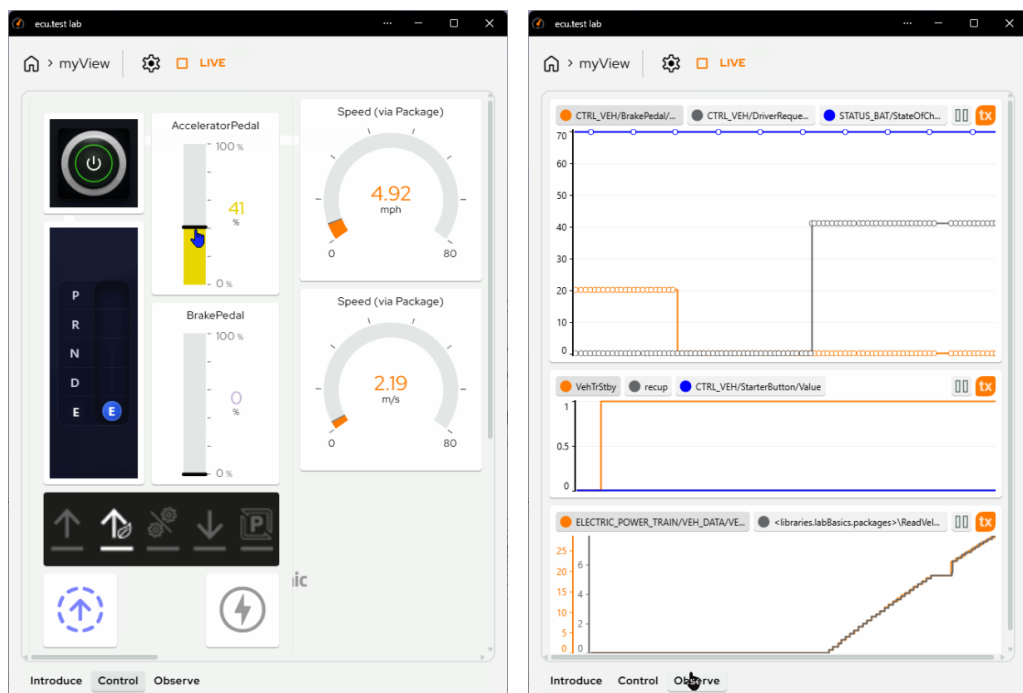


Abbildung 14: Implementierung von Tabs zur Navigation durch mehrere Seiten

## 4 Usability

### Browser-Erweiterung "Open with ecu.test diff"



**Open with ecu.test diff** bietet die Möglichkeit, aus dem Browser heraus Änderung an **ecu.test**-Artefakten im **ecu.test** Diff Viewer zu öffnen.

Mit der nun veröffentlichten Version 1.1.0 wird neben GitHub und GitLab auch **BitBucket Cloud** unterstützt.

Zusätzlich wurde die Einrichtung und einige Fehlermeldungen verbessert.

Die Erweiterung kann über den [Firefox](#) und [Chrome Store](#) bezogen werden.

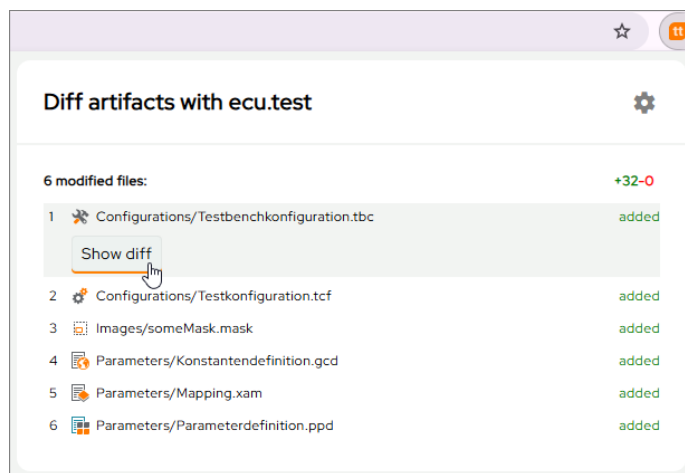


Abbildung 15: Browser-Erweiterung "Open with ecu.test diff"

### Workspace-Auswahl-Dialog: Lizenzen für ecu.test extras



Im Workspace-Auswahl-Dialog, vor dem vollständigen Start von **ecu.test**, können die genutzten Lizenzen für die **ecu.test** Extras angezeigt und bei Bedarf direkt im Lizenzmanager konfiguriert werden.

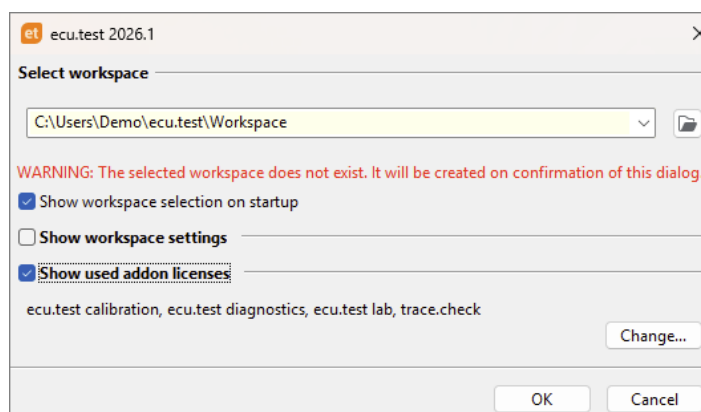


Abbildung 16: Lizenzen für ecu.test extras anzeigen und konfigurieren

## 5 Testaspekte

### 5.1 HiL

#### Unterstützung der Intrepid CAN(-FD)-Bushardwareanbindung



Alle Features der hardwarenahen Busanbindungen inkl. **ecu.test** *diagnostics* und **ecu.test** *calibration* können jetzt mit CAN- und CAN-FD-fähiger Messhardware, die in Verbindung mit der Intrepid neoVI-API angesteuert werden können, verwendet werden.

#### ETAS INCA: Überarbeitetet Umgang mit Datensätzen



Die INCA TBC-Option **Verhalten bei vorhandenem Projekt** für Applikations-Ports legt fest, welche der Konfigurationen für Projekt (A2L-Datei) und Datensatz (HEX-Datei) in **ecu.test** und INCA beim Öffnen des INCA-Experiments verwendet werden.

Da es in Abhängigkeit, ob es einen Datensatz gibt und welcher Datensatz priorisiert verwendet soll, eine Vielzahl von Anforderungen geben kann, wurden die Auswahlmöglichkeiten grundlegend überarbeitet und dokumentiert. Details finden sich in der [Anwenderdokumentation](#).

### 5.2 Testmanagement

#### Beispiel-Workflow für PTC Codebeamer



Basierend auf der neuen User-Testmanagement-API gibt es einen ersten Beispiel-Workflow zum Import von Packages und Projekten aus PTC Codebeamer.

Zusammen mit dem tracetronic-Codebeamer-Client ist der volle Zugriff auf die Codebeamer-REST-API möglich. Die rein Python-basierte Lösung lässt sich einfach an individuelle Workflows anpassen.

## 5.3 Traceanalyse

### Zugriff auf Enum-Werte in Traceschrittvorlagen



In ereignisbasierten Python- und arraybasierten NumPy-Traceschrittvorlagen ist es nun über die API möglich, auf die Enum-Werte eines Signals zuzugreifen.

```
def Process(self, parameters, report, timeAxis, ranges, signals):
    """
    ...
    """
    # E.g. a typical bus status signal with text table
    firstRawValue = signals['EnumSignal'].GetValues()[0]
    textTable, defaultText = signals['EnumSignal'].GetTextTable()
    if textTable:
        textValue = textTable.get(firstRawValue, defaultText) or ''
        print(f'First signal value: {firstValue}, corresponding text: {textValue}')
```

Abbildung 17: Zugriff auf Enum-Werte eines Signals via API

### Neue Standard-Traceschrittvorlage "CalculateInterpolatedSignal"



Selten abgetastete Signale von eigentlich kontinuierlichen (analogen) Messgrößen machen robuste Prüfungen oft schwierig.

Standardmäßig werden Signalwerte per Sample-and-Hold auf eine gemeinsame Zeitachse mit anderen Signalen abgebildet. Beispielsweise erfordert die Verwendung der **Hose**-Funktion aber eine individuelle Interpolation.

Die neue Standard-Traceschrittvorlage **CalculateInterpolatedSignal** ist in der Lage, die Werte eines Signals linear auf die Zeitachse eines Referenzsignals zu interpolieren. Somit lassen sich viele Anwendungsfälle einfach umsetzen.

## 5.4 Weitere Testaspekte

### Verbesserte Unterstützung von Messlisten



Aufbauend auf der in **ecu.test** 2025.4 eingeführten Möglichkeit, XAM-Dateien als Messliste für Signalgruppen zu nutzen, bringt die aktuelle Version weitere praktische Verbesserungen:

- **Bibliotheksworkspaces integriert:** XAM-Dateien können nun direkt aus Bibliotheksworkspaces importiert werden.
- **Reporting:** Für eine bessere Nachvollziehbarkeit, werden für die Aufzeichnung verwendete Messlisten dokumentiert.
- **Einfaches Konfigurieren:** Messlisten lassen sich jetzt einfach per Drag-and-Drop konfigurieren.

## 6 Versionen und Schnittstellen

### 6.1 Neue Tools und Versionen



	Provider	Web- seite	System	Produktname	Version
1	<b>Accurate Technologies Inc.</b>	<a href="#">Release-Link</a>	Kalibrierungs- und Datenerfassungstool	<b>ATI VISION</b>	<b>7.0</b>
2	<b>Lauterbach</b>	<a href="#">Release-Link</a>	Tool zur Analyse, Optimierung und Zertifizierung von eingebetteten System	<b>TRACE32</b>	<b>R.2025.2</b>
3	<b>PLS</b>	<a href="#">Release-Link</a>	Tool zum Debuggen, Nachverfolgen und Testen eingebetteter Software	<b>UDE</b>	<b>2025.2</b>

## 7 Abkündigungen

### 7.1 Abkündigungen und Inkompatibilitäten in dieser Version

#### Python 3.13



**ecu.test 2026.1** basiert auf Python 3.13.

Nach aktueller Einschätzung sollte der Umstieg wenig Auswirkung auf bestehende **ecu.test**-Workspaces haben.

Jedoch sollte eigener Python-Code in Packages, Benutzer-Bibliotheken und externe Python-Bibliotheken auf Python-3.13-Kompatibilität geprüft und gegebenenfalls aktualisiert werden.

- <https://pyreadiness.org/3.13/>

Ob auch eigene Codestellen durch Removals betroffen sind, kann unter folgendem Link nachvollzogen werden:

- <https://docs.python.org/3/whatsnew/3.13.html#removed>

Hilfreiche Informationen zum Umstieg gibt es in der

- [Knowledge Base](#)

#### Port BUSACCESS - GENERIC\_MAPPINGFILE



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, wurde der BUSACCESS - GENERIC\_MAPPINGFILE mit **ecu.test 2026.1** entfernt.

#### Abkündigung FEP2 Anbindung



Mit **ecu.test 2026.1** wurde die FEP2-Anbindung entfernt.

### Ubuntu 20.04 LTS und 22.04 LTS



Mit **ecu.test 2026.1** wurde die Unterstützung für Ubuntu 20.04 LTS und 22.04 LTS entfernt.

### Playbook-Export von ecu.test nach test.guide



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

### Unterstützte Versionen von MATLAB/Simulink in Linux



Im Zuge der Abkündigung der Unterstützung älterer Versionen von Ubuntu mit **ecu.test 2026.1** wurde 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.

### ICMP-RAW Port



Für die Tools Ethernet, XL-API und SIL-Kit wurde der ICMP-RAW Port nach Ankündigung ersatzlos entfernt.

### Alternatives Reportverzeichnis bei separater Unterprojektausführung



Bei separater Unterprojektausführung bestand die Möglichkeit, den Reportordner angeben zu können. Dieses Feature wurde in **ecu.test** Version **2026.1** entfernt.

## 7.2 Abkündigungen in zukünftigen Versionen

### KS: Tornado nur noch über ASAM ACI



Die Toolanbindung wird voraussichtlich mit **ecu.test 2026.2** entfernt. Sie wird durch die neue Anbindung auf Basis von ASAM ACI ersetzt, die seit **ecu.test 2023.3** verfügbar ist.

## Fibex-Unterstützung



Die Fibex-Unterstützung für Bus wird abgekündigt und soll mit **ecu.test 2026.2** entfernt werden. Die Fibex-Unterstützung für DLT bleibt weiterhin erhalten.

## TRF-Dateien lassen sich für Projektberichte nicht mehr erzeugen



Im Zuge der Einführung des PRF-Formats für Projektberichte 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 zu **ecu.test 2026.2** entfernt.

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

## Interface TmUserAdapter



Die folgenden Interface-Methoden werden mit **ecu.test 2026.2** entfernt:

- GetPackageFromTms
- GetProjectsFromTms

Als Ersatz dienen diese neu eingeführten Methoden:

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

## Import im Zusammenhang mit CustomChecks



Eigene Prüfungen verwenden unter Umständen ein Modul, das umgezogen ist. Daher müssen die Importe geprüft und ggf. angepasst werden.

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

## Jama Connect



Der integrierte Test Management Adapter für **Jama Connect** wird mit **ecu.test 2026.2** 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



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

- RequestSeed → SecurityAccessRequestSeed
- SendKey → SecurityAccessSendKey

Die neuen Jobs unterstützen auch die Seed & Key DLLs.

## Echtzeit-Packagereferenzen



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 ausgebliebenen 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-Package-Referenzen. Weitere Features rund um das Thema Real Time Testing (RTT) sind nicht betroffen.

## Bus-Monitoring-Testschritte



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



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



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-Workflows bereitgestellt.

### IBM Rational Quality Manager (RQM)



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-Workflows bereitgestellt.

## HP Application Lifecycle Management (ALM)



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



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**



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.