violis.io
by Bertrandt

Whitepaper - Ausgabe 2026

Vom Prüfstandsengpass zur skalierbaren Teststrategie.

Wie virtuelles Software-Testing Hardware-Prüfstände entlastet, Tests parallelisiert und Testzyklen von Wochen auf Stunden verkürzt.

Double your speed – half the time.

by BERTRANDT
Engineering Excellence trifft Plattformlogik mit violis

INHALT

Was Sie in diesem Whitepaper erwartet

Vorwort · Warum virtuelle Absicherung den Takt vorgibt

Executive Summary · Fünf Kernaussagen

01 Der Paradigmenwechsel zum Software-defined Testing

02 Das Universum: HiL, SiL, MiL, PiL

03 Trends und Entwicklungen

04 Schluss mit Prüfstands-Engpässen

05 Anatomie einer Virtual-Testing-Plattform

06 Branchenperspektive: wer profitiert?

07 Anwendungsfälle und Praxisbeispiele

08 ROI und Business Case

09 Umsetzungsplan in 6 Schritten

10 Mythen und Vorbehalte

11 Ausblick

12 violis.io im Fokus

Anhang · Glossar · Quellen · Service-Portfolio

Warum virtuelle Absicherung heute den Takt vorgibt

Software ist nicht länger nur ein Baustein moderner Produkte, sie ist ihr Wesen.

Ob Fahrzeug, Flugzeug, Zug, Medizingerät, Industrieanlage, vernetzte Kaffeemaschine oder smarte Heizungssteuerung: was diese Systeme leisten, entscheidet der Code, nicht die Mechanik. Mit dieser Verschiebung wächst die Verantwortung für die Software-Absicherung sprunghaft, ebenso die Anforderungen an Testumgebungen, Geschwindigkeit und Reproduzierbarkeit.

Hardware-in-the-Loop-Prüfstände bleiben unverzichtbar

Klassische HiL-Prüfstände bleiben in absehbarer Zeit unverzichtbar. HiL bedeutet: die reale Steuerelektronik läuft in einer simulierten Umgebung, die Sensoren, Aktoren und Busse nachbildet. Solche Prüfstände sind allerdings teuer, knapp und werden schnell zum Engpass. Verzögerungen von Tagen oder Wochen sind nicht die Ausnahme, sondern die Regel.

Softwaredefinierte Produkte, komplexe Controller-Architekturen, Over-the-Air-Updates und vernetzte Systeme aller Branchen fordern ein Tempo, das physische Prüfstände allein nicht mehr liefern können. Die Antwort lautet: virtualisieren, wo immer es Sinn ergibt. HiL wird nicht ersetzt, sondern intelligent ergänzt.

Der Schlüssel heißt Software-in-the-Loop (SiL): die Steuerungssoftware läuft als virtueller Controller auf handelsüblichen Rechnerumgebungen – auf Servern, Workstations oder in der Cloud, ohne spezielle Echtzeit-Hardware. Kombiniert mit Cloud-Skalierung und CI/CD-Integration entsteht daraus eine neue Klasse der Absicherung. Mit violis.io liefert Bertrandt genau diese Plattform: offen, modular, Cloud-affin und containerbasiert.

---

70 %

des Testvolumens lassen sich virtuell abbilden

100x

schnellere Ausführung durch Parallelisierung

-52 %

TCO über drei Jahre gegenüber reinem HiL

Automotive als Geburtsstätte: warum violis.io im härtesten Umfeld gereift ist

violis.io ist nicht in einem Labor entstanden, sondern im wohl anspruchsvollsten technischen Umfeld der industriellen Softwareentwicklung: der Automobilindustrie. Hier treffen extreme Varianten-Vielfalt, sicherheitskritische Funktionen, regulatorischer Druck (ISO 26262, ISO 21434, ASPICE), enge Release-Zyklen und eine über Jahrzehnte gewachsene Tool-Landschaft aufeinander.

Genau dort hat sich violis.io als Benchmark und Taktgeber etabliert. Über Jahre intensiver Projektarbeit mit OEMs und Tier-1-Lieferanten ist eine Plattform entstanden, die sich hochdifferenzierten Anforderungen stellt: integrierte Tests über mehrere Steuergeräte hinweg, nahtlose Anbindung an bestehende Toolchains, Wiederverwendung etablierter HiL-Testfälle, lückenlose Nachweisführung.

Das Automobil gehört zu den komplexesten Produkten der Welt. Ansätze, die hier erfolgreich erprobt werden, tragen auch in der Luftfahrt, im Schienenverkehr, in der Medizintechnik, in der Industrieautomation und bei vernetzten Konsumprodukten. Ehrlich betrachtet ist die Transformation im Automotive selbst noch nicht abgeschlossen: die klassische Industrie holt gegenüber Software-first-Herstellern auf, die ihre Fahrzeuge von Beginn an wie eine Software-Plattform entwickeln und testen.

Genau dieser Wandel muss weitergehen. Bertrandt fördert ihn mit Lösungen wie violis.io – aktiv und nutzbar für jede Branche, die Software auf Hardware entwickelt.

Für wen dieses Whitepaper geschrieben ist

Für alle, die Software für Hardware entwickeln: vom OEM bis zum Gerätehersteller, vom Tier-1 bis zum Komponentenlieferanten.

Viel Freude bei der Lektüre.

Das violis.io-Team bei Bertrandt

Fünf Kernaussagen auf einen Blick

Virtuelles Software-Testing ist heute keine Zukunftsoption mehr, sondern Pflicht für jede Organisation, die softwaregetriebene Produkte entwickelt.

01 **Software-defined Products sind der Treiber.** Software bestimmt Funktion, Nutzererlebnis und Differenzierung – im Fahrzeug, im Flugzeug, im OP-Saal, in der Smart Factory oder in der Espressomaschine.

02 **HiL bleibt – als Spezialist, nicht als Engpass.** Hardware-Prüfstände konzentrieren sich auf Major-Releases und systemkritische Tests. Das Tagesgeschäft wandert in die virtuelle Welt.

03 **SiL plus Cloud plus CI/CD ergeben eine neue Geschwindigkeitsklasse.** Continuous Integration und Continuous Testing mit virtuellen Controllern bringen Testzyklen von Wochen in Stunden.

04 **Branchenübergreifender Transfer ist real.** Methoden aus der Automobilindustrie lassen sich auf jede Branche übertragen, die Software auf Hardware ausführt. Toolchains und Vokabular unterscheiden sich, die Methodik nicht.

05 **violis.io ist die offene Plattform für diesen Übergang.** Bertrandt bietet kein weiteres geschlossenes Tool, sondern eine Infrastruktur, die bestehende Werkzeuge, Workflows und IT einbindet.

Die Wirkung: ein Vielfaches an Durchsatz, mindestens halbierte Time-to-Release, deutlich geringere Hardware-Investitionen und eine Testabdeckung, die mit klassischen Mitteln schlicht nicht zu erreichen ist.

ABBILDUNG 1

Die Wirkung auf einen Blick

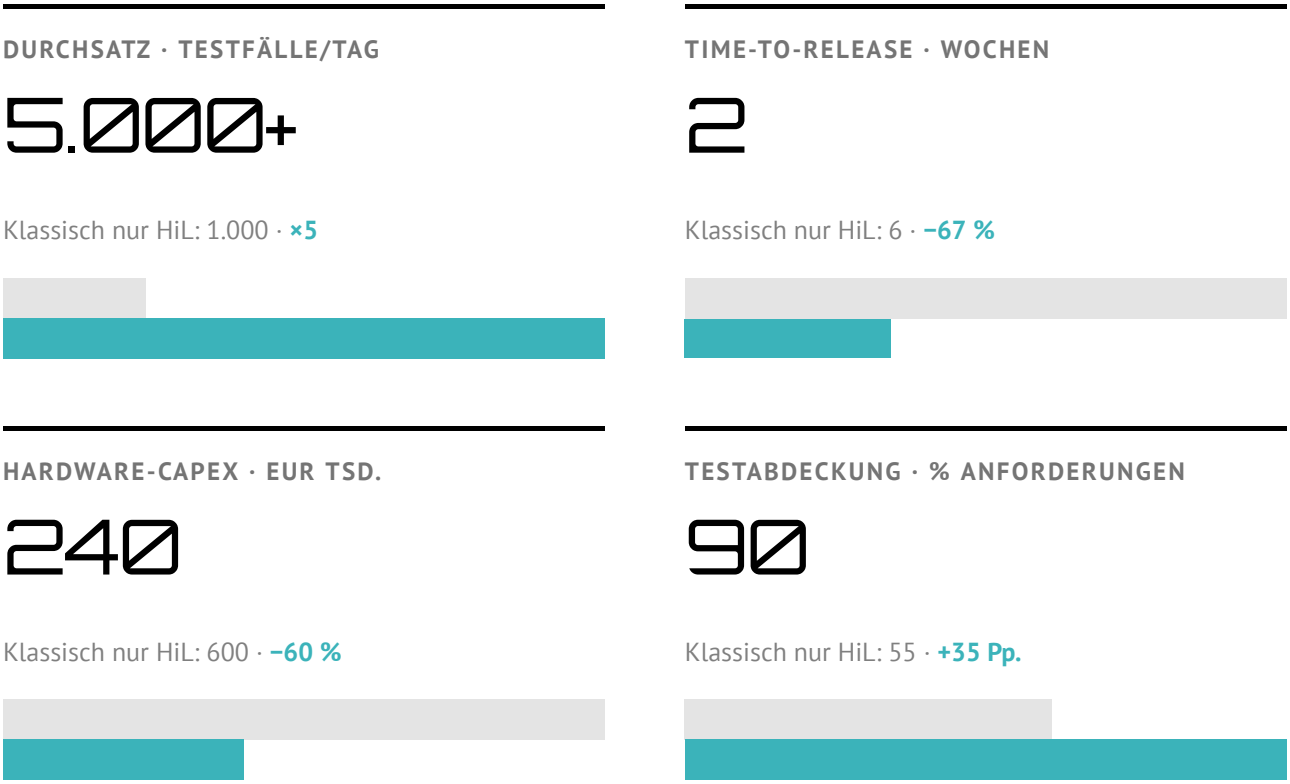


Abbildung 1: Die Wirkung auf einen Blick. Vergleich von vier Kernkennzahlen: Durchsatz, Time-to-Release, Hardware-CAPEX und Testabdeckung. Klassisches HiL-Setup gegenüber der Kombination aus HiL und violis.io-Plattform.

Quelle und Einordnung: Bertrandt-Projektdaten 2024–2026.

EXECUTIVE SUMMARY · AUF EINEN BLICK

NUTZEN

- Kürzere Time-to-Market durch parallele virtuelle Testausführung.
- Geringere Investitionen in Prüfstands-Hardware (CAPEX zu OPEX).
- Risiko-Reduktion durch frühe Fehlererkennung im Entwicklungszyklus.
- Skalierbarkeit, die mit der Komplexität moderner Software mitwächst.

PRAXIS-EMPFEHLUNG

- Klein anfangen: ein Anwendungsfall, ein Team, ein messbares Ziel.
- Bestehende HiL-Testfälle wiederverwenden, nicht neu erfinden.
- Plattformwahl an offenen Standards ausrichten (FMI, AUTOSAR, OpenSCENARIO).
- Erfolg mit klaren Kennzahlen messen: Durchsatz, Abdeckung, Zeit pro Release.

01

Der Paradigmenwechsel

Vom Hardware-Fokus zum Software-defined Testing

Software wird zum primären Werttreiber – branchenübergreifend. Klassisches Testen skaliert nicht mit. Und spät erkannte Software-Fehler verursachen die teuersten Folgekosten.

1.1 Software ist nicht mehr Teil des Produkts. Software ist das Produkt.

Moderne Fahrzeuge enthalten mehr als 100 Steuergeräte und über 100 Millionen Zeilen Code. Das Phänomen beschränkt sich nicht auf das Auto: ein modernes Flugzeug, ein OP-Roboter, eine SPS in der Smart Factory, eine vernetzte Premium-Espressomaschine oder eine intelligente Heizungssteuerung folgen derselben Logik. Funktionen wie Konnektivität, Update-Fähigkeit, Abo-Funktionen oder personalisierte Bedienoberflächen sind rein softwaredefiniert. Die Hardware bleibt wichtig, wird aber zunehmend austauschbar. Was bleibt: Software, Algorithmen, Daten.

1.2 Die SDV-Welle und ihre Folgen für andere Branchen

Was in der Automobilindustrie als „Software-defined Vehicle“ diskutiert wird, läuft parallel überall: Software-defined Aircraft, Train, Medical Device, Appliance. Die Methoden, mit denen Automotive seine Komplexität bewältigt – Modularisierung, Virtualisierung, kontinuierliche Absicherung – werden zur Blaupause für alle, die vernetzte, intelligente, sicherheits- oder qualitätskritische Produkte bauen.

1.3 Warum klassisches Testen an seine Grenzen kommt

- **Skalierung:** mehr Varianten, mehr Märkte, mehr Releases – aber physische Prüfstände skalieren nicht mit.
- **Tempo:** quartals- oder monatsweise Release-Zyklen kollidieren mit Hardware-Verfügbarkeit und Aufbauzeiten.
- **Frühe Fehlererkennung:** Probleme, die erst bei der Hardware-Integration auftauchen, kosten ein Vielfaches gegenüber einer frühen virtuellen Erkennung.

1.4 Software-Bugs als Top-Treiber der Gewährleistungskosten

Software gehört heute zu den größten Verursachern von Gewährleistungs- und Rückrufkosten. Ein einziger größerer Bug kann dreistellige Millionenbeträge kosten. Frühe, kontinuierliche und virtuelle Absicherung ist damit nicht nur eine Engineering-Frage, sondern strategische Risikoreduktion.

Die Software-Explosion in modernen Produkten

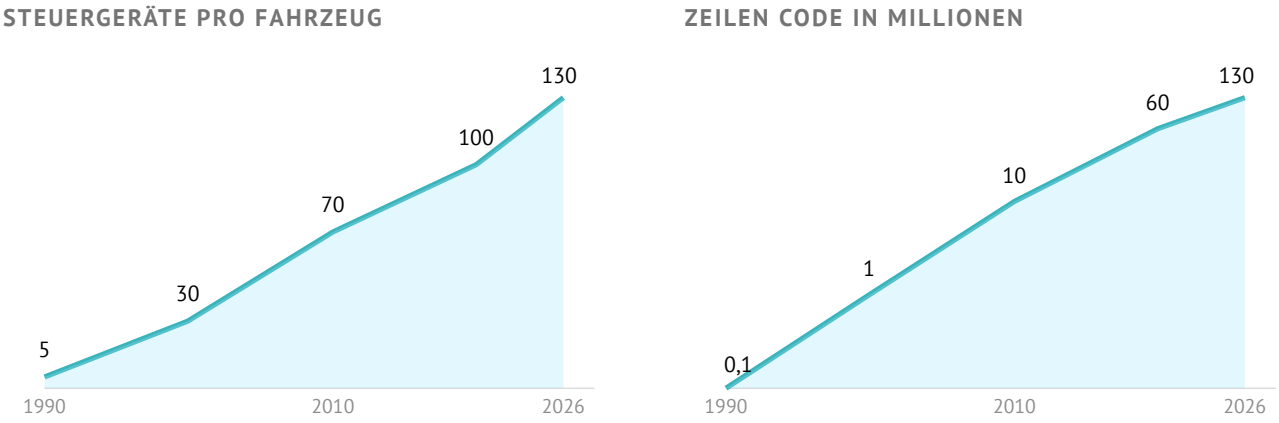


Abbildung 2: Die Software-Explosion in modernen Produkten. Zunahme der Steuergeräte pro Fahrzeug und der Codezeilen zwischen 1990 und 2026 (rechts logarithmisch). Die gleiche Entwicklung gilt für Flugzeuge, Medizingeräte, Industrie-Controller und vernetzte Konsumprodukte.

Quelle und Einordnung: Richtwerte auf Basis von IoT Analytics, Software-defined Vehicles Adoption Report 2026, und Bertrandt-Branchenerfahrung.

1. DER PARADIGMENWECHSEL · AUF EINEN BLICK

NUTZEN

- Software wird zum primären Werttreiber, branchenübergreifend.
- Klassisches Testen skaliert nicht mit wachsender Software-Komplexität.
- Spät erkannte Software-Fehler verursachen die teuersten Folgekosten.

PRAXIS-EMPFEHLUNG

- Software-Anteil im eigenen Produkt beziffern: Steuergeräte, Controller, MCUs.
- Testaufwand pro Release messen und über 12 Monate dokumentieren.
- Engpass-Analyse: wo verbringt das Team heute die meiste Wartezeit?

02

HiL, SiL, MiL, PiL

Die Testhierarchie neu gedacht

HiL und SiL sind keine Konkurrenten, sondern Ergänzungen. SiL trägt das Volumen. HiL konzentriert sich auf Echtzeit, Buskommunikation und sicherheitskritische Major-Releases.

2.1 Die klassische V-Modell-Welt und ihre Engpässe

Das V-Modell der Absicherung sieht vier Testebenen vor: Model-in-the-Loop (MiL), Software-in-the-Loop (SiL), Processor-in-the-Loop (PiL) und Hardware-in-the-Loop (HiL). Klassisch ist HiL die dominante Stufe – mit dem bekannten Engpass: zu wenige Prüfstände, zu viele Teams, zu lange Stillstände.

2.2 HiL: Goldstandard mit Verfügbarkeitsproblem

HiL bleibt der Goldstandard für Echtzeitprüfungen am realen Steuergerät unter realistischen Bus- und Lastbedingungen. HiL ist aber teuer (sechsstellige Investitionen je Prüfstand), wartungsintensiv, ortsgebunden und im Schichtbetrieb umkämpft. Das Ergebnis: Ingenieure warten – oft länger, als sie testen.

2.3 SiL als Skalierungshebel, nicht als HiL-Ersatz

SiL führt Controller-Software als virtuelle Komponente auf Standard-Rechnerumgebungen aus: lokal, im Rechenzentrum oder in der Cloud. Vorteile: praktisch unbegrenzte Skalierung, schnellere Iteration, parallele Ausführung tausender Testfälle, einfache CI-Anbindung.

2.4 HiL wird durch SiL ergänzt, nicht ersetzt

Der entscheidende Denkwechsel: SiL trägt das Volumen der Regressions- und Variantentests. HiL konzentriert sich auf Echtzeitverhalten, hardwarenahe Buskommunikation, EMV-Effekte und sicherheitskritische Major-Releases.

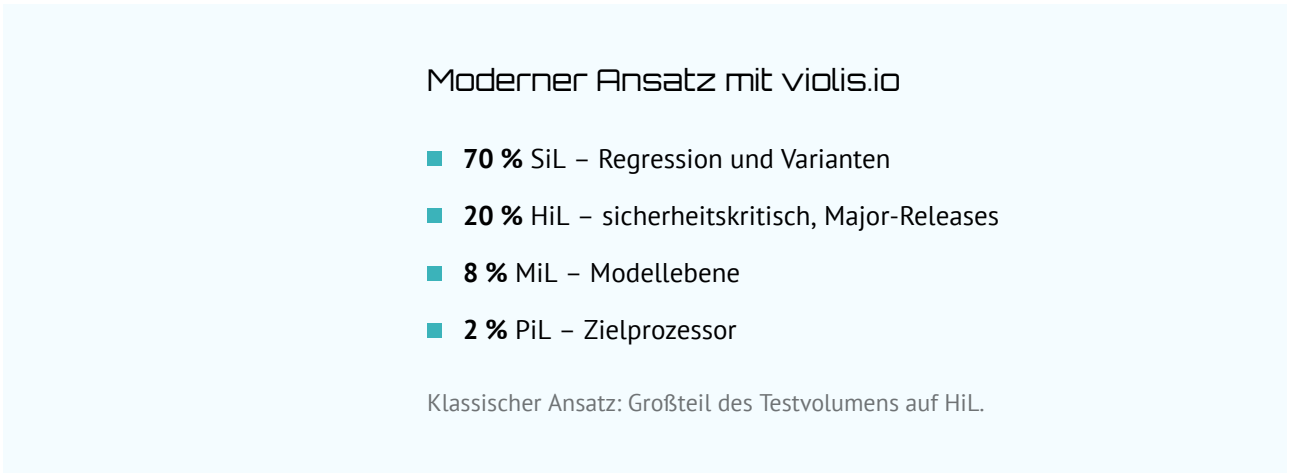


Abbildung 3: Verteilung des Testvolumens im V-Modell. Klassische Setups verlagern den Großteil des Testvolumens auf HiL. Moderne Ansätze verlagern rund 70 % der Tests auf SiL.

Quelle und Einordnung: Beispielhafte Verteilung auf Basis der Validierungsmethodik und Projekterfahrung von Bertrandt; keine Messwerte. Kontext: VDA 710 (07/2022) und VDA 711 (03/2025), Software-in-the-Loop (SiL) Standardisierung.

2.5 MiL und PiL im Zusammenspiel

MiL prüft die Funktion auf Modellebene (etwa in Simulink). PiL führt den generierten Code auf realer Zielhardware unter Simulationsbedingungen aus. Beide bleiben Bausteine einer durchgängigen Toolchain, orchestriert durch eine intelligente Testplattform.

2.6 Begriffe jenseits der Automotive-Welt

Nicht jede Branche spricht von „ECU“ oder vECU. Wer Software für Hardware entwickelt, hat meist einen eigenen Begriff.

Automotive

ECU, vECU, HPC

Aerospace

Avionik-Rechner, LRU, Flight Control Computer

Industrieautomation

SPS, Industrie-Controller, Edge Device

Medizintechnik

Embedded Controller, medizinischer Steuerrechner

Consumer

MCU, Controller, Smart Module, IoT-Device

In diesem Whitepaper

„Controller“ bzw. „virtueller Controller“ als branchenübergreifender Platzhalter

2.7 Software-defined Testing (SDT) als Orchestrator

Software-defined Testing meint den Ansatz, alle Testebenen, Werkzeuge und Datenquellen in eine softwaredefinierte, automatisierbare Plattformlogik zu überführen. Testfälle entscheiden nicht mehr starr, wo sie ausgeführt werden. Die Plattform entscheidet datengetrieben.

2. HIL, SIL, MIL, PIL · AUF EINEN BLICK

NUTZEN

- HiL und SiL ergänzen sich, sie sind keine Alternativen.
- SiL skaliert linear mit Cloud-Ressourcen, HiL bleibt physisch begrenzt.
- Die Methodik gilt branchenübergreifend, nur die Begriffe wechseln.

PRAXIS-EMPFEHLUNG

- Klare Aufgabenteilung festlegen: was testet HiL, was testet SiL?
- Testfall-Kategorien einführen: Regression, Varianten, Echtzeit, Safety.
- Bestehende Testfälle bewerten: welche lassen sich sofort virtualisieren?

03

Trends und Entwicklungen

Was die Industrie bewegt

ZONALE ARCHITEKTUREN | VIRTUELLE CONTROLLER | CLOUD-NATIVE | EU-SOUVERÄNITÄT | CI/CD/CT | AGENTIC AI | DIGITAL TWIN | SHIFT-LEFT | OTA | SAFETY & SECURITY | OFFENE STANDARDS

3.1 Zonale Architekturen und HPC

Die Konsolidierung dutzender Steuergeräte in wenige Zentralrechner ist in vollem Gange. Hypervisor-basierte Architekturen erlauben den parallelen Betrieb mehrerer VMs mit AUTOSAR Adaptive, Classic und Linux – Controller werden grundlegend emulierbar.

3.2 Virtuelle Controller als Standardbaustein

Virtuelle Controller sind keine Forschungsobjekte mehr, sondern Standard-Artefakte jedes Daily Build – vom Application-Container bis zur vollständigen Emulation eines Zentralrechners.

3.3 Cloud-native, TISAX und EU-Souveränität

Bedenken zu IP-Schutz sind durch zertifizierte Rechenzentren und Self-Hosting adressiert. Souveräne Hyperscaler wie StackIT bieten Betrieb in Deutschland unter deutschem Recht; alternativ läuft die Plattform vollständig On-Premise.

3.4 CI/CD/CT-Pipelines werden Standard

GitLab-, GitHub- oder Jenkins-Pipelines binden Controller-Erzeugung, Co-Simulation, Testausführung und Reporting in einen einzigen automatisierten Fluss ein.

3.5 KI und Agentic AI in der Testfallerzeugung

Generative und agentische KI unterstützen Erzeugung, Priorisierung und Auswertung von Testfällen. Ingenieure werden von Routinearbeit entlastet, Edge Cases systematischer abgedeckt.

3.6 Digital Twin trifft virtueller Prüfstand

Digitale Zwillinge ganzer Systeme laufen in Co-Simulation mit virtuellen Controllern. Aus dem Prüfstand wird eine plattformbasierte Absicherungsumgebung.

3.7 Shift-Left: testen, bevor die Hardware existiert

Software kann Monate vor Verfügbarkeit der finalen Hardware abgesichert werden – ein entscheidender Hebel für die Time-to-Market.

3.8 OTA erzwingt kontinuierliche Absicherung

Wenn Updates jederzeit ausgerollt werden können, muss auch jederzeit getestet werden können. Kontinuierliche virtuelle Absicherung ist die einzige skalierbare Antwort.

3.9 Cybersecurity und Functional Safety

ISO 26262, ISO 21434, IEC 62304, DO-178C, EN 50128, IEC 61508 und ASPICE verlangen lückenlose, dokumentierte Absicherung. Virtuelle Plattformen leisten das bei sauberer Umsetzung zuverlässiger als reine Hardware-Aufbauten.

3.10 OpenSCENARIO, FMI/FMU, AUTOSAR, QEMU

Offene Standards bilden das Fundament: OpenSCENARIO für Szenarien, FMI/FMU für portable Modelle, AUTOSAR für Steuergeräte-Software, QEMU als Emulator. Wer diese Bausteine beherrscht, bleibt toolunabhängig.

Offene Standards schlagen geschlossene Tools

OpenSCENARIO	Offener Standard zur Beschreibung von Fahr- und Verkehrsszenarien.
FMI / FMU	Portables Modellformat für Co-Simulation über Werkzeuggrenzen hinweg.
AUTOSAR	Standardisierte Architektur für Steuergeräte-Software – Classic und Adaptive.
QEMU	Open-Source-Emulator für Linux-basierte Controller und Firmware.

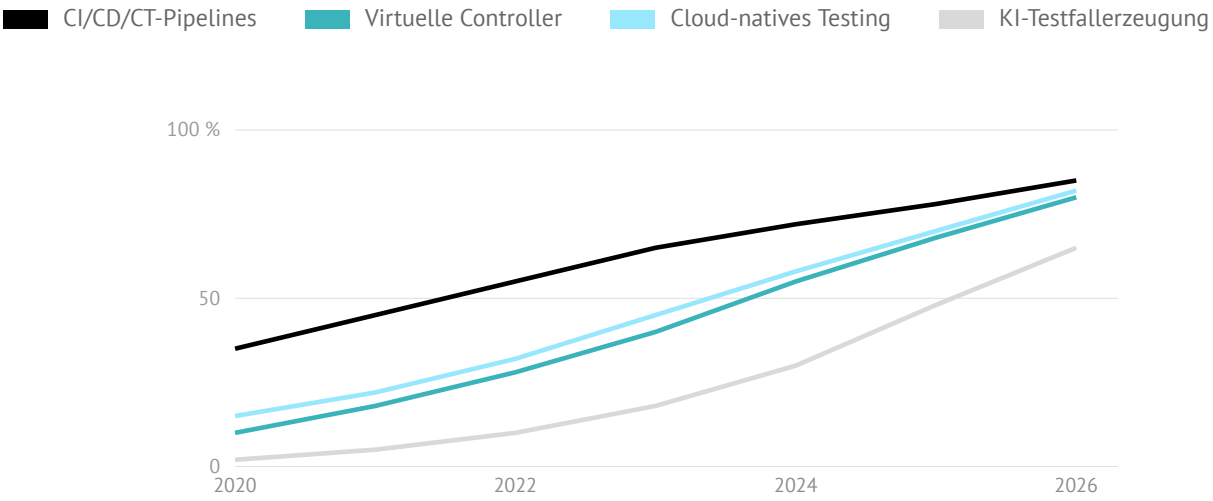


Abbildung 4: Adoption digitaler Testtrends 2020–2026. Indikative Entwicklung, Anteil der Unternehmen mit produktivem Einsatz (geschätzt). CI/CD/CT-Pipelines führen die Adoptionskurve an, gefolgt von virtuellen Controllern und Cloud-nativem Testing. KI-gestützte Testfallerzeugung holt schnell auf.

Quelle und Einordnung: Qualitative Einschätzung von Bertrandt auf Basis eigener Projekterfahrung; keine Messwerte. Kontext: IoT Analytics, Software-defined Vehicles Adoption Report 2026; Eclipse Foundation / Eclipse SDV Working Group, Open-Source-Studienreihe 2024–2025.

3. TRENDS UND ENTWICKLUNGEN · AUF EINEN BLICK

NUTZEN

- Zonale Architekturen und Cloud-Skalierung eröffnen neue Testmöglichkeiten.
- CI/CD/CT wechselt vom Software-Konzept zum Embedded-Standard.

PRAXIS-EMPFEHLUNG

- Frühzeitig auf offene Standards setzen – spätere Migration kostet ein Vielfaches.
- CI-Pipeline und virtuelle Testumgebung gemeinsam planen, nicht nacheinander.

04

Schluss mit Prüfstands-Engpässen

Die Geschwindigkeits-Story

250–600 €

vollkalkulierte Kosten einer HiL-Stunde

> 80 %

typische HiL-Auslastung bei OEM und Tier-1

24/7

Verfügbarkeit der virtuellen Testbench

4.1 Was eine HiL-Stunde im Konzern wirklich kostet

Eine HiL-Stunde kostet vollkalkuliert – inklusive Abschreibung, Wartung, Räumen, Personal und Software-Lizenzen – schnell 250 bis 600 Euro. Engpässe verstärken diese Kosten unsichtbar: jede ungenutzte Stunde ist eine Stunde, in der Engineering-Kapazität blockiert bleibt.

4.2 Wenn der Prüfstand zum Engpass wird

In typischen OEM- und Tier-1-Umgebungen sind HiL-Auslastungen von über 80 % Standard, mit dem bekannten Effekt: kleine Tests werden verschoben, Ingenieure arbeiten an Workarounds, Releases rutschen. Das eigentliche Problem ist nicht der Prüfstand selbst, sondern das Fehlen einer Alternative.

4.3 Wie virtuelle Prüfstände 24/7 verfügbar werden

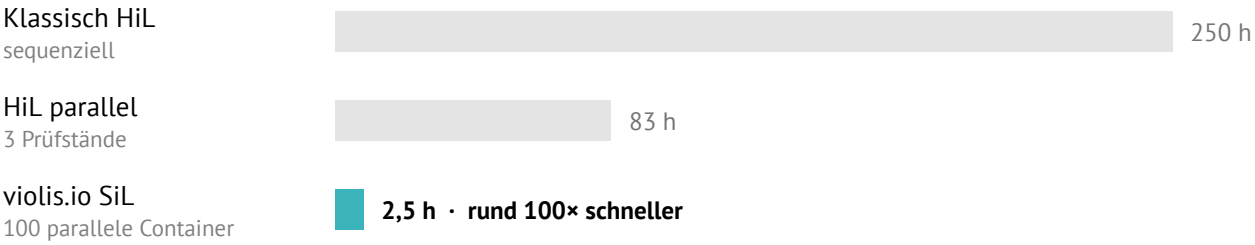
Eine virtuelle Testbench läuft im Browser, ist in Minuten bereitgestellt und steht jedem Ingenieur dauerhaft zur Verfügung. Debugging-Werkzeuge wie CANoe, VEOS oder Silver sind vorinstalliert. Nach manuellem Eingriff übernimmt die Pipeline wieder.

4.4 Skalierung: vom einzelnen Prüfstand zur Cloud-Flotte

Was physisch unmöglich ist – 200 Prüfstände parallel zu betreiben – ist in der Cloud trivial. Auto-Scaling fängt Lastspitzen automatisch ab, Pay-per-Use macht die Kosten transparent.

4.5 Rechenbeispiel: 5.000 Testfälle in Stunden statt Wochen

Ein typisches Tier-1-Release mit 5.000 Testfällen:



Selbst gegenüber drei parallelen HiL-Prüfständen ist SiL mehr als 30× schneller.

Abbildung 5: Wall-Clock-Zeit für 5.000 Testfälle. Ein repräsentatives Tier-1-Release. Klassisches HiL-Testen dauert Wochen; parallelisierte SiL-Ausführung in violis.io ist in Stunden fertig.

Quelle und Einordnung: Modellrechnung auf Basis eines Bertrandt-Referenzprojekts (2025); Annahme: 3 min Laufzeit je Testfall auf HiL.

Wartezeit ist die teuerste Position im Testbudget

Ingenieure warten heute oft länger, als sie testen. Genau diese Wartezeit ist der Hebel, den virtuelles Testen zuerst bewegt.

4. SCHLUSS MIT PRÜFSTANDS-ENGPÄSSEN · AUF EINEN BLICK

NUTZEN

- Engineering-Zeit, die heute am Prüfstand wartet, wird zu produktiver Zeit.
- Cloud-Skalierung ermöglicht Testmengen, die physisch nicht denkbar sind.
- Pay-per-Use öffnet das Investitionsmodell für Mittelständler und Start-ups.

PRAXIS-EMPFEHLUNG

- Wartezeiten am Prüfstand vier Wochen lang messen, Baseline schaffen.
- Kosten pro Testfall berechnen – klassisch versus virtualisiert.
- Erstes Quick-Win-Projekt mit 50 bis 100 Testfällen identifizieren.

„Stell dir vor: jeder Ingenieur hat jederzeit einen eigenen Prüfstand und kann Testimplementierungen direkt absichern. Alle Varianten können getestet werden, während der eigentliche Hardware-Prüfstand sich auf Major-Releases fokussiert."

Nic Eckstein · Project Manager Cloud & IT Processes, Bertrandt

05

Anatomie einer Virtual-Testing-Plattform

Sechs Bausteine, ein Stack

Container-Architektur, Co-Simulation, Tool-Unabhängigkeit, REST-APIs, virtuelle Testbench und Datenorchestrierung.

5.1 Container-basierte Architektur

Jedes Simulationswerkzeug, jeder virtuelle Controller, jede Test-Engine läuft als eigener Docker-Container. Das macht die Plattform cloud-agnostisch: ob AWS, Azure, GCP, StackIT, On-Premise oder hybrid – es ist derselbe Stack.

5.2 Co-Simulation-Framework

Ein virtueller Bus (CAN, FlexRay, LIN, Automotive Ethernet; in anderen Branchen ARINC, EtherCAT, PROFINET, Modbus, MQTT) verbindet beliebig viele Controller, Anlagenmodelle und Test-Tools. Simulationen werden per Konfiguration aufgebaut, nicht per Verkabelung.

5.3 Tool-Unabhängigkeit

Vector CANoe, dSPACE VEOS, Synopsys Silver, ecu.test, PROVEtech:TA, MATLAB/Simulink, QEMU: alle gängigen Werkzeuge werden out of the box unterstützt. Für industrielle Welten lassen sich TwinCAT, CODESYS, Keil oder branchenspezifische Simulatoren über offene Schnittstellen anbinden.

5.4 REST-APIs und Pipeline-Vorlagen

Jede Plattformfunktion ist per REST-API ansprechbar. Fertige Vorlagen für GitLab, GitHub Actions und Jenkins sorgen für einen schnellen Start; eigene Plug-ins entstehen gegen die offenen Endpoints.

5.5 Virtuelle Testbench: Debugging im Browser

Wenn ein Testfall scheitert, öffnet der Ingenieur die virtuelle Testbench direkt im Browser – mit identischer Toolumgebung. Nach dem Fix übernimmt die automatisierte Pipeline wieder.

5.6 Datenorchestrierung und Analytics

Testergebnisse aus SiL, HiL, Fahrzeug- oder Feldtests und Cloud laufen in einer zentralen Datenebene zusammen. Analytics zeigen, welche Testumfänge auf welcher Instanz am besten laufen – inklusive automatisierter Empfehlungen.

5. ANATOMIE EINER PLATTFORM · AUF EINEN BLICK

NUTZEN

- Eine offene, container-basierte Plattform schlägt jedes geschlossene Tool auf Dauer.
- Bestehende Werkzeuge können weiter genutzt werden, kein Umschulen nötig.

PRAXIS-EMPFEHLUNG

- Bestehenden Tool-Stack inventarisieren: was muss angebunden werden?
- Schnittstellen-Anforderungen (REST, FMI, AUTOSAR) früh festlegen.

ABBILDUNG 6

Relative TCO-Verteilung

Drei-Jahres-Vergleich, indexiert mit HiL-CAPEX = 100

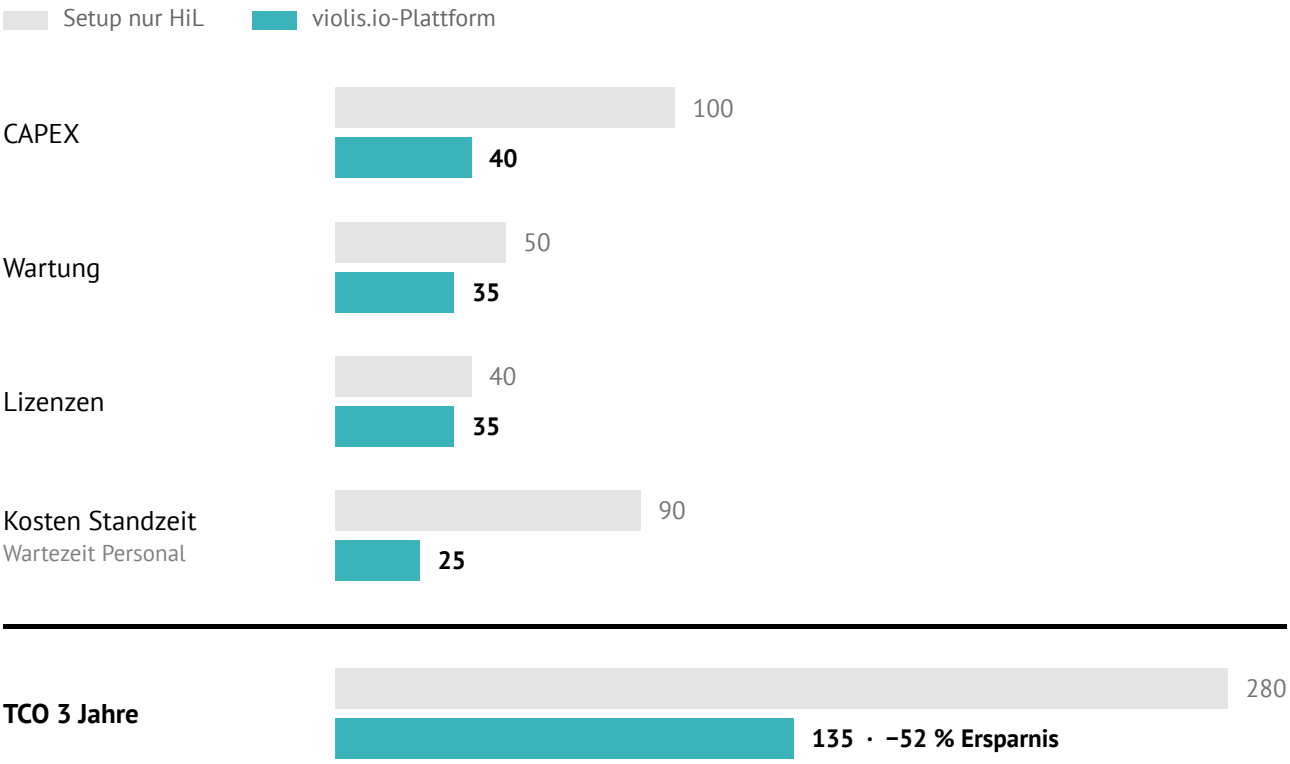


Abbildung 6: Relative TCO-Verteilung. Drei-Jahres-TCO-Vergleich, indexiert mit HiL-CAPEX = 100. Die Kosten für Wartezeiten des Personals sind der größte versteckte Einsparhebel.

Quelle: Bertrandt Kundenfallstudien; interne Kostenmodellierung.

Der größte Kostenblock steht nicht in der Investitionsrechnung.

Standzeit – also wartendes Engineering-Personal – schlägt im klassischen Setup mit Index 90 zu Buche und fällt mit violis.io auf 25.

06

Branchenperspektive

Wer profitiert – außer Automotive?

Die Methodik des virtuellen Software-Testings ist branchenunabhängig. Werkzeuge, Normen und Begriffe unterscheiden sich, die Architektur bleibt vergleichbar.

Faustregel: läuft Ihre Software auf Mikrocontrollern, SoCs, FPGAs oder SPS, gilt das Folgende für Sie.

6.1 Automotive

Steuergeräte-Software, zonale HPCs, Domain Controller. **Begriffe:** vECU, ECU-Verbund, Restbus-Simulation. **Normen:** ISO 26262, ISO 21434, ASPICE.

6.2 Aerospace & Defence

Flugsteuerung, virtuelle Avionik-Integration, UAV-Software. **Begriffe:** Avionik-Rechner, Flight Control Computer, LRU. **Norm:** DO-178C.

6.3 Bahn und Schienenverkehr

ETCS, Türsteuerungen, Brandschutz, Energieverteilung, Leit- und Sicherungstechnik. **Normen:** EN 50128, EN 50657, EN 50129.

6.4 Medizintechnik

Patientenmonitoring, Infusionspumpen, Beatmung, bildgebende Systeme, chirurgische Roboter. **Normen:** IEC 62304, FDA, MDR.

6.5 Energie und Smart Grid

Wechselrichter, BMS, Netzschutz, Wallboxen, Elektrolyseure. **Normen:** IEC 61850, IEC 61508, Netzanschlussrichtlinien.

6.6 Industrieautomation

SPS-Software, Robotik, MES-Komponenten, Werkzeugmaschinen. **Normen:** IEC 61131, IEC 61508. Digital-Twin-Tests vor der realen Anlage.

6.7 Wearables und Diagnostik

Glukose-Sensoren, EKG-Patches, smarte Implantate, Point-of-Care-Geräte. Wochen virtueller Laufzeit in Stunden Rechenzeit.

6.8 Consumer und Smart Home

Küchengeräte, Espressomaschinen, Saugroboter, Beleuchtung, Heizungssteuerungen. OTA-Updates und Cloud-APIs sind Standard.

6.9 Telko, Netzwerk, IoT

Router, Edge Devices, Gateways, 5G-Komponenten. **Begriffe:** NFV, Virtual Network Function, Edge Compute Node.

6.10 Land- und Baumaschinen

Forst- und Kommunalfahrzeuge. **Begriffe:** Implement Controller, ISOBUS-Knoten, Telematik-Controller. Lange Lebenszyklen, hohe Variantenvielfalt.

6.11 Defence, Robotik, Sicherheit

Unbemannte Systeme, autonome Plattformen, Zutrittskontrollen, Brandmeldeanlagen. **Treiber:** Safety, Cyber-Resilienz, Einsatzvarianz.

Gemeinsamer Nenner

Ein eingebetteter Prozessor, eine Firmware, eine Umgebung aus Sensoren, Aktoren und Bussen. Nur das Vokabular wechselt.

6. BRANCHENPERSPEKTIVE · AUF EINEN BLICK

NUTZEN

- Jeder Hersteller, dessen Software auf einem Controller läuft, gehört zur Zielgruppe.
- Branchen unterscheiden sich in Werkzeugen und Normen, nicht in der Methode.

PRAXIS-EMPFEHLUNG

- Eigene Werkzeuge und Normen einer der skizzierten Familien zuordnen.
- Mit dem Vokabular der eigenen Branche arbeiten.

ABBILDUNG 7

Reifegrad und Potenzial je Branche

Automotive zeigt den größten ungenutzten Hebel. Auch Medizintechnik, Energie und Land-/Baumaschinen bieten großes Potenzial.

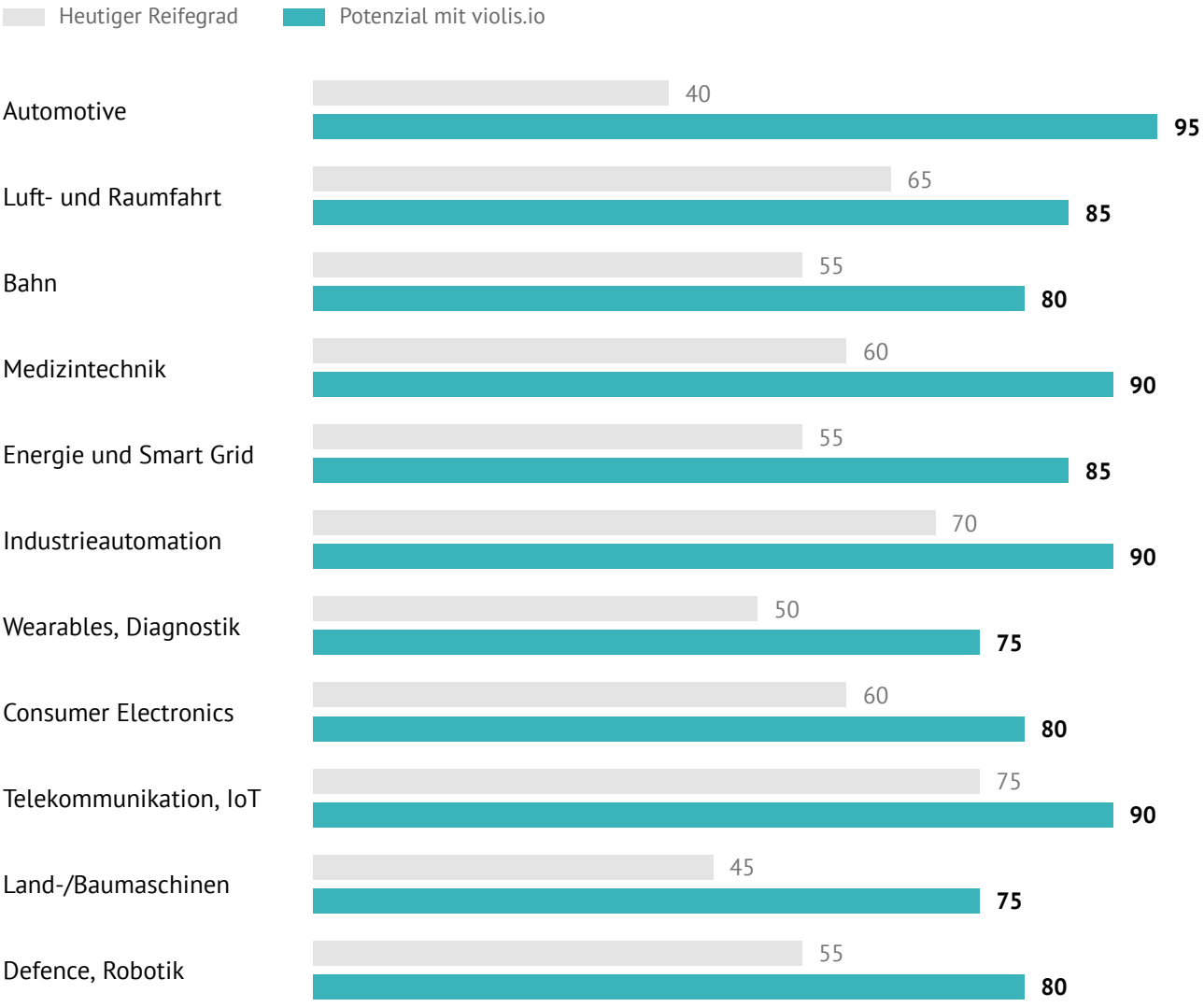


Abbildung 7: Reifegrad und Potenzial je Branche. Index 0–100: 0 = keine virtuelle Testabsicherung, 100 = vollständig integrierte, virtualisierte Testprozesse.

Quelle und Einordnung: Qualitative Einschätzung von Bertrandt auf Basis eigener Projekterfahrung; keine Messwerte. Kontext: Omdia SDV Survey 2026 (Automotive), Tahera et al. (2023), Proceedings of the Design Society 3, DOI: 10.1017/pds.2023.231 (branchenübergreifende Perspektiven auf virtuelles Testen).

Anwendungsfälle und Praxisbeispiele

Sieben Szenarien aus laufenden Projekten

7.1 Klima- und Energiesysteme im Fahrzeug

Ein virtueller Verbund aus Controllern und thermischen Modellen erlaubt Lastszenarien – Sommer, Winter, Schnellladen, Reichweitenoptimierung – die in realer Fahrzeugerprobung Wochen brauchen würden.

7.2 Multi-Controller-Integrationstest

Mehrere virtuelle Controller werden über virtuelle Busse gekoppelt, inklusive Restbus-Simulation. Komplette Funktionsketten werden in der Cloud durchgetestet – ADAS, Energiemanagement, SPS-Verbünde, Avionik-LRUs.

7.3 HiL-Testfälle 1:1 in SiL wiederverwenden

Bestehende, abgesicherte HiL-Testfälle werden ohne Anpassung virtuell ausgeführt. Das schützt Investitionen in bestehende Testfall-Bibliotheken.

7.4 Daily Build zu Cloud-basiertem Testen

Aus jedem Daily Build wird automatisch ein virtueller Controller erzeugt, in eine Co-Simulation integriert und gegen Hunderte Testfälle laufen gelassen. Ergebnisse stehen am nächsten Morgen im Dashboard.

7.5 Fehleranalyse: Browser statt Reise zum Prüfstand

Ein Ingenieur in München öffnet die virtuelle Testbench eines Teams in Wolfsburg, Detroit oder Shanghai im Browser. Reisen, Versand und Zeitzonen-Themen entfallen.

7.6 Vernetztes Konsumprodukt: Premium-Espressomaschine

Statt jede Variante an einer realen Maschine zu testen, laufen virtuelle Brüheinheits-Modelle nachts in der Cloud durch Tausende Brühprofile. Sensor- und Aktorkurven fließen in die Qualitätsdatenbank.

7.7 Medizinprodukt: Beatmungsgeräte-Familie

Zehn Marktvarianten werden virtuell mit synthetischen Patientenmodellen betrieben. Edge Cases – Husten, Atemstillstand, Tubus-Trennung – lassen sich reproduzierbar testen, ohne reales Risiko.

ABBILDUNG 8

Bearbeitungszeiten: vorher und nachher

Typische Durchlaufzeiten gängiger Absicherungsaufgaben fallen von Tagen auf Stunden.

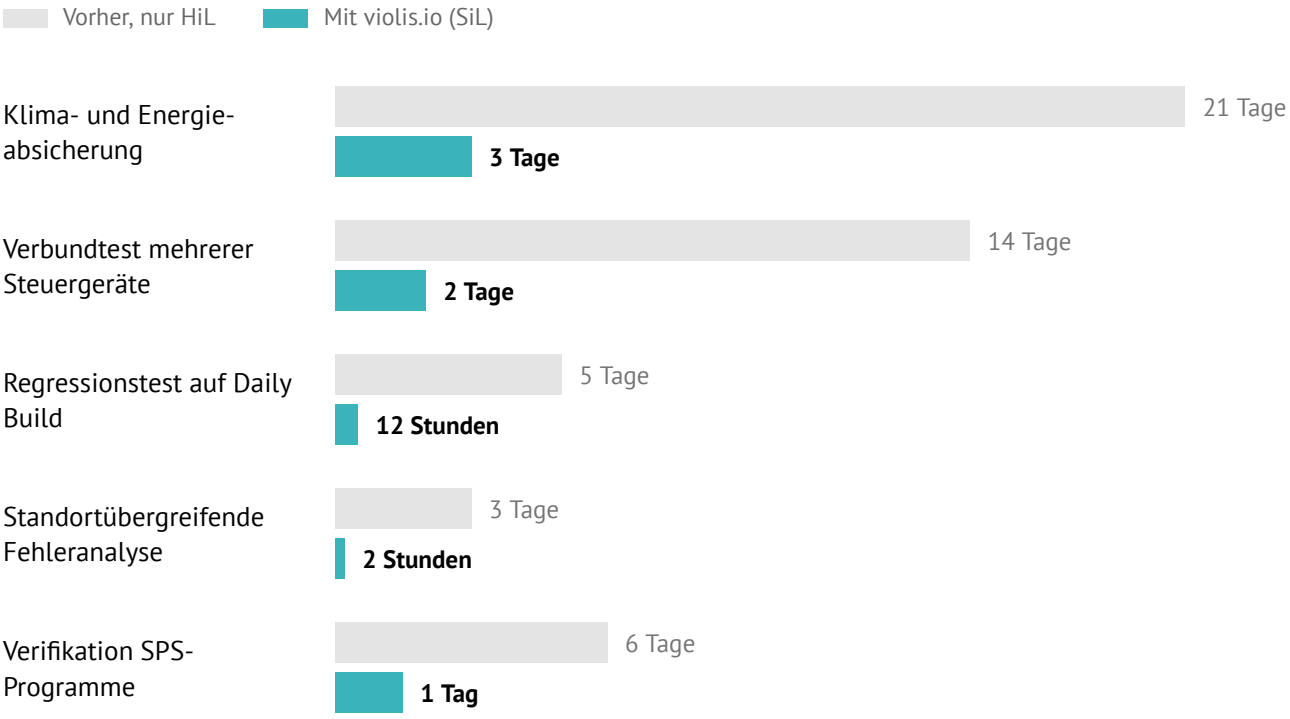


Abbildung 8: Bearbeitungszeiten in Anwendungsfällen. Beispielhafte Durchlaufzeiten gängiger Absicherungsaufgaben vor und nach der Verlagerung von HiL auf virtuelle Controller.

Quelle und Einordnung: Beispielwerte auf Basis von Bertrandt-Projekterfahrung; keine Messwerte. Zum Vergleich: Zäh et al. (2006) belegen für die virtuelle Inbetriebnahme durchschnittlich 75 % kürzere reale Inbetriebnahmezeiten.

7. ANWENDUNGSFÄLLE · AUF EINEN BLICK

NUTZEN

- Virtuelles Testen erschließt Anwendungsfälle, die physisch nicht abbildbar sind.
- Standortübergreifende Zusammenarbeit wird deutlich einfacher.
- Anwendungsfälle entstehen branchenübergreifend.

PRAXIS-EMPFEHLUNG

- Use-Case-Inventur: welche Tests sind heute teuer, langsam oder riskant?
- Ein Showcase pro Quartal aufsetzen – sichtbare Erfolge schaffen Akzeptanz.
- Erkenntnisse zwischen Standorten teilen.

KAPITEL 08

ROI und Business Case

Was virtuelles Testen tatsächlich liefert

8.1 KPI-Rahmen

Durchsatz

Ausgeführte Testfälle pro Zeiteinheit.

Testabdeckung

Anteil abgesicherter Anforderungen.

Time-to-Release

Vom Commit zum freigegebenen Build.

TCO

Hardware, Personal, Lizenzen, Räume, Energie.

8.2 CAPEX zu OPEX

Statt sechsstelliger Investitionen in zusätzliche Prüfstände wandert die Testinfrastruktur in nutzungsbasierte OPEX-Strukturen. Das verbessert Bilanz und Skalierbarkeit – nach oben wie nach unten.

8.3 Time-to-Market: 30–50 % kürzer

Konservativ verkürzen virtuelle Testpipelines die Entwicklungszeit um 30 %. Bei hoher Variantenvielfalt oder starkem OTA-Fokus sind 40 bis 50 % dokumentiert.

8.4 Talent und Skills

Der Markt für Embedded Software ist überzeichnet – die Nachfrage übersteigt das Angebot oft um Faktor drei. Virtuelle Plattformen vervielfachen die effektive Produktivität pro Kopf.

8.5 Nachhaltigkeit

Weniger Hardware-Aufbauten bedeuten weniger Material, weniger Energie, weniger Geschäftsreisen, weniger Logistik. Virtuelles Testen zahlt auf ESG-Ziele ein.

8.6 Beispielrechnung: mittelständischer Gerätehersteller

12 Releases pro Jahr, drei HiL-Prüfstände, 20 Embedded-Ingenieure.

KLASSISCH

60 % Prüfstand-Wartezeit · 12 Releases/Jahr · 250.000 EUR CAPEX je zusätzlichem Prüfstand

MIT VIOLIS.IO

20 % Wartezeit · 18 Releases/Jahr · kein zusätzlicher Prüfstand nötig

ERGEBNIS

+50 %

mehr Releases bei stabiler Hardware-Investition

ABBILDUNG 9

Kumulierte TCO über 36 Monate

Der Break-even wird typischerweise im ersten Betriebsjahr erreicht.

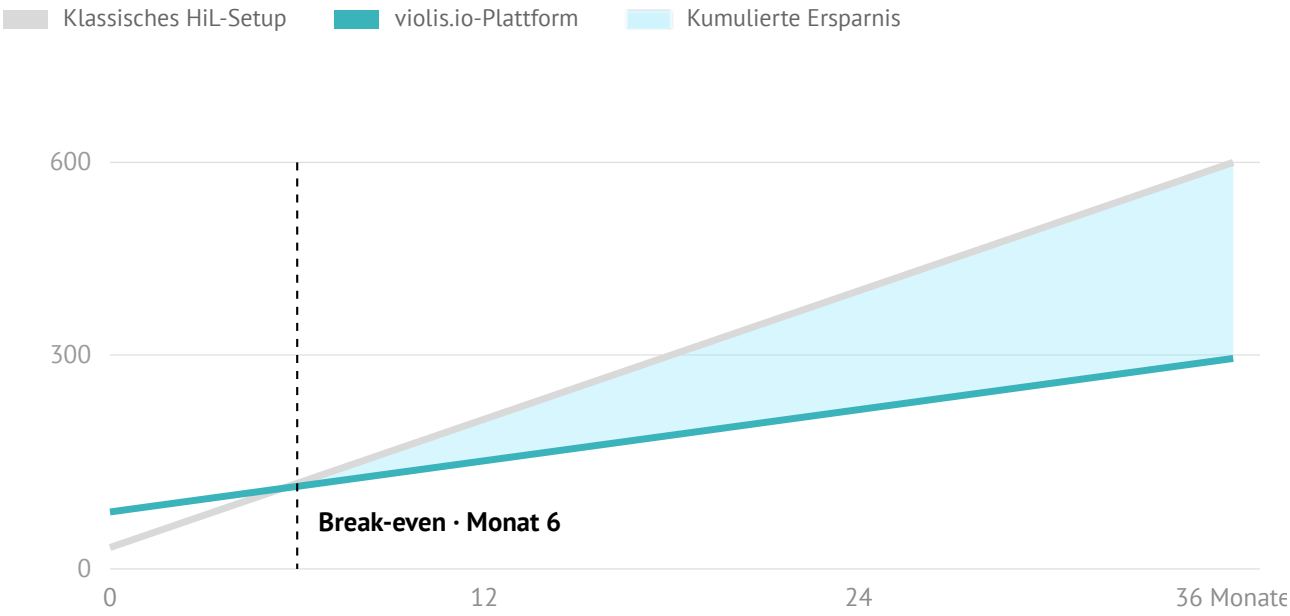


Abbildung 9: Kumulierter TCO über 36 Monate. Kumulative Kostenkurve eines klassischen HiL-Setups im Vergleich zur violis.io-Plattform, indexiert. Die getönte Fläche zeigt die kumulierte Ersparnis.

Quelle und Einordnung: Bertrandt Kunden-TCO-Modelle; indexierte Modellrechnung, keine Messwerte.

8. ROI UND BUSINESS CASE · AUF EINEN BLICK

NUTZEN

- Klare Kennzahlen machen den Nutzen sichtbar und kommunizierbar.
- OPEX-Modelle entlasten die Bilanz und schaffen Flexibilität.
- Auch mittelständische Hersteller profitieren, nicht nur Großkonzerne.

PRAXIS-EMPFEHLUNG

- KPI-Baseline vor dem PoC erheben, sonst lässt sich Erfolg nicht nachweisen.
- TCO-Vergleich über 3 bis 5 Jahre durchspielen, nicht nur Year-1-Effekte.
- CFO früh einbinden – der Business Case ist der Hebel für breite Einführung.

In 6 Schritten zur virtuellen Testumgebung

1. **Status quo analysieren**
   Prüfstands-Engpässe quantifizieren: welche Releases sind durch HiL-Verfügbarkeit begrenzt? Wie viel Engineering-Zeit geht in Wartezeiten verloren?

2. **Use-Cases priorisieren**
   Welche Testfälle sind häufig, repetitiv und gut virtualisierbar? Welche eignen sich für einen ersten Proof of Concept?

3. **PoC aufsetzen**
   Mit vorhandenen virtuellen Controllern oder Simulationen starten. Auch ohne fertigen virtuellen Controller lässt sich mit Komponenten, Modellen oder Restbus-Simulationen beginnen.

4. **CI/CD-Integration**
   Anbindung an GitLab, GitHub oder Jenkins, Definition der Stages, automatisiertes Reporting in Standardwerkzeugen.

5. **Skalierung, Rollout und Schulung**
   Ingenieure werden mit der virtuellen Testbench vertraut gemacht. Pipeline-Vorlagen beschleunigen die Übernahme.

6. **Optimieren und messen**
   Durchsatz, Testabdeckung, Time-to-Release und TCO werden monatlich gemessen und kommuniziert.

Checkliste: sind Sie bereit für virtuelles Testen?

- Wir haben dokumentierte HiL-Engpässe oder Wartezeiten am Prüfstand.
- Wir nutzen bereits CI/CD-Pipelines – oder planen sie.
- Mindestens ein Steuergerät oder Subsystem könnte als virtueller Controller bereitgestellt werden.
- Es gibt Testfälle, die heute manuell oder halbautomatisch laufen.
- Das Management unterstützt die schrittweise Modernisierung der Testlandschaft.

Drei oder mehr Häkchen? Dann lohnt sich ein Gespräch über einen PoC mit violis.io.

Iterativ, nicht „Big Bang“

8–12	26	1
Wochen maximale PoC-Laufzeit. Länger verliert Momentum.	Wochen von der Status-quo-Analyse bis zum vollen Betrieb.	Champion pro Team. Ohne interne Fürsprecher kein Rollout.

9. UMSETZUNGSPLAN · AUF EINEN BLICK

NUTZEN

- Der Weg in die virtuelle Welt ist iterativ, kein „Big Bang“.
- Quick Wins schaffen interne Akzeptanz und Budgetspielraum.
- Externe Begleitung verkürzt die Lernkurve deutlich.

PRAXIS-EMPFEHLUNG

- PoC mit maximal 8 bis 12 Wochen Laufzeit aufsetzen.
- In jedem Team einen Champion benennen.
- Erfolge sichtbar machen: Demos, Dashboards, kurze Videos im Intranet.

„Klein anfangen: ein Anwendungsfall, ein Team, ein messbares Ziel.“

KAPITEL 10

Gängige Mythen und Vorbehalte

Ehrliche Antworten auf die fünf häufigsten Einwände

Mythos 1: Virtuelles Testen ersetzt nicht die Realität.

STIMMT – IST ABER NICHT DAS ZIEL

HiL und Feldtests bleiben unverzichtbar. Virtuelles Testen ergänzt sie und übernimmt rund 70 % des Testvolumens, das heute physische Ressourcen blockiert.

Mythos 2: Wir haben noch keinen virtuellen Controller.

FALSCH

Auch ohne fertige Virtualisierung lassen sich Simulationen, Software-Komponenten oder Modelle einbinden. Der Aufbau virtueller Controller ist eine eigene Disziplin und kann parallel starten.

Mythos 3: Unsere IT wird das nicht erlauben.

LÖSBAR

Hosting im eigenen Rechenzentrum, in einer Private Cloud, auf souveränen Hyperscalern wie StackIT oder in einer TISAX-zertifizierten externen Cloud. Container-Architektur macht den Stack portierbar.

Mythos 4: Wir werden in einem Tool-Ökosystem eingesperrt.

MIT VIOLIS.IO AUSDRÜCKLICH NICHT

Bertrandt ist kein Lizenzhersteller. Die Plattform unterstützt alle gängigen Werkzeuge und offenen Standards: FMI, OpenSCENARIO, AUTOSAR, QEMU.

Mythos 5: Das lohnt sich nur für Großkonzerne.

PAY-PER-USE ÄNDERT DIE RECHNUNG

Auch mittelständische Tier-2-Zulieferer und Spezialisten in Medical, Rail, Smart Home und Industrieautomation nutzen virtuelle Testplattformen wirtschaftlich.

Virtual Testing übermorgen

Fünf Entwicklungen, die den nächsten Zyklus prägen

KI-gestützte Testfall-Priorisierung

Agentic AI entscheidet, welche Testfälle in welcher Reihenfolge auf welcher Instanz laufen – gesteuert durch Code-Änderungen, Risikoanalyse und Historie.

Selbstheilende Testumgebungen

Tests, die an Infrastrukturproblemen scheitern, werden automatisch wiederholt. Tests mit echten Software-Fehlern eskalieren. Die Plattform unterscheidet das selbständig.

Vollständig virtuelle Produkte

Fahrzeug, Flugzeug, Zug, Medizingerät, Kaffeemaschine, Produktionslinie – jeweils als Digital Twin, der synchron mit der realen Flotte läuft.

Branchenübergreifende Standards

Initiativen wie Eclipse SDV und die VDA-Standardisierung schaffen Referenzarchitekturen. Wissen wandert leichter zwischen Automotive, Aerospace, Medical und Smart Home.

Regulatorische Anerkennung virtueller Nachweise

Regulierungsbehörden akzeptieren zunehmend virtuelle Absicherungsnachweise – ein wichtiger Hebel für ADAS, autonomes Fahren und Medizingeräte. Voraussetzung: lückenlose, prüffähige Toolchains.

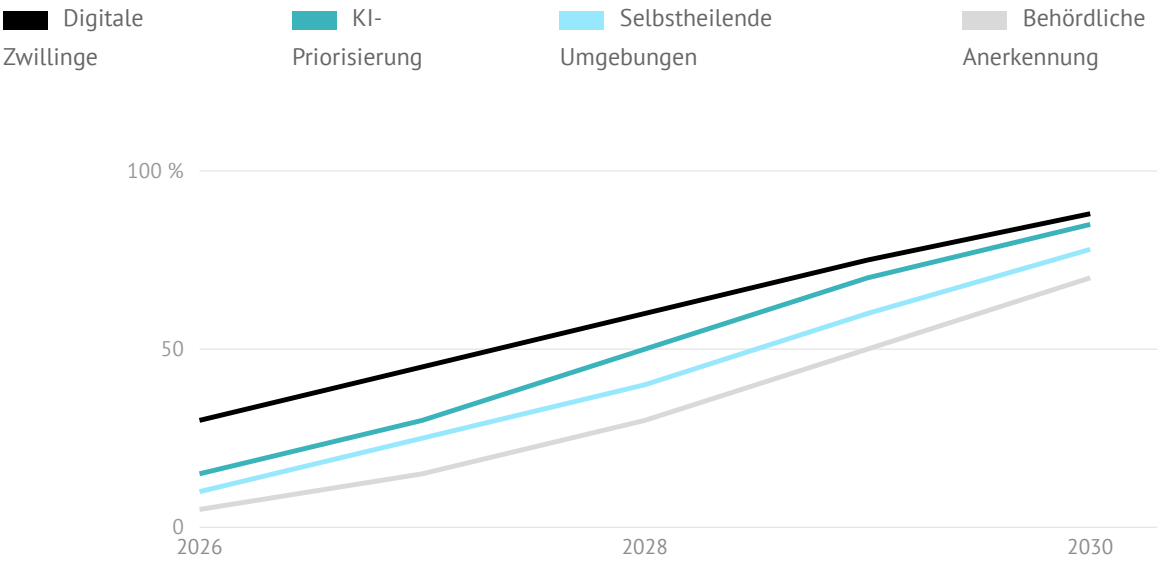


Abbildung 10: Virtual-Testing-Trends 2026–2030. Indikative Projektion der Verbreitung (geschätzter Anteil der Unternehmen mit produktivem Einsatz) von Digital Twins, KI-gestützter Priorisierung, selbstheilenden Testumgebungen und regulatorischer Akzeptanz virtueller Nachweise.

Quelle und Einordnung: Qualitative Einschätzung von Bertrandt; keine Messwerte. Kontext: IoT Analytics, Software-defined Vehicles Adoption Report 2026; Eclipse SDV Working Group; VDA 710/711 (SiL-Standardisierung).

KAPITEL 12

violis.io im Fokus

Die Plattform für die Transformation

violis.io virtualisiert den gesamten Testprozess – von der Simulation bis zur Auswertung – ohne bestehende Toolumgebungen zu brechen. Kein weiteres Tool, sondern die Infrastruktur, auf der modernes Testen läuft.

- Unabhängige Lösung – Bertrandt ist kein Lizenzhersteller mit Tool-Bindung.
- Mehrere virtuelle Controller im Verbund testen, inklusive ganzer Systeme.
- HiL-Testfälle 1:1 wiederverwenden.
- Skalierbar ohne zusätzliche Hardware-Investitionen.
- Transparente Abrechnung im Pay-per-Use-Modell.
- Unterstützung aller gängigen Werkzeuge – Vector, dSPACE, Synopsys, ecu.test, QEMU.
- Modulare Umgebung auf Basis von Docker-Containern.
- Virtuelle Testbench direkt im Browser nutzbar.
- Hosting in eigener Cloud, auf StackIT oder in der TISAX-zertifizierten Bertrandt Cloud.

Vom PoC zum Rollout

Bertrandt begleitet jeden Schritt: von der ersten Analyse über den PoC bis zum unternehmensweiten Rollout. Jahrzehnte Erfahrung im hardwarenahen Testen treffen auf moderne Cloud- und Software-Kompetenz.

Ihre Ansprechpartner

Nic Eckstein · Project Manager Cloud & IT Processes
Felix Strauß · Lead Expert XiL-Technologies

Unit Electronics & Virtual Testing Solution, Bertrandt AG

100	95	100	95
Tool-Unabhängigkeit	Cloud-Skalierung	Wiederverwendung HiL-Testfälle	CI/CD-Integration

Glossar

SDV · Software-defined Vehicle – Fahrzeug, dessen Funktionen primär durch Software definiert werden.

SDT · Software-defined Testing – Testmethodik mit virtualisierten, plattform-orchestrierten Testabläufen.

vECU · Virtuelles Steuergerät – Software-Repräsentation eines Steuergeräts für SiL-Tests.

Virtueller Controller · Branchenübergreifender Begriff für die Software-Repräsentation eines eingebetteten Steuerrechners.

ECU / Controller · Eingebetteter Steuerrechner mit eigener Firmware.

LRU · Line Replaceable Unit – Avionik-Begriff für eine austauschbare Funktionseinheit.

MCU · Microcontroller Unit – Mikrocontroller in eingebetteten Systemen.

SiL · Software-in-the-Loop – Software-Test auf Standard-Rechnerumgebung ohne Zielhardware.

HiL · Hardware-in-the-Loop – Test des realen Steuergeräts in simulierter Umgebung.

MiL · Model-in-the-Loop – Test auf Modellebene, z. B. in Simulink.

PiL · Processor-in-the-Loop – Test generierten Codes auf dem Zielprozessor.

OpenSCENARIO · Offener Standard zur Beschreibung von Fahr- und Verkehrsszenarien.

ISO 26262 / 21434 · Normen für Funktionale Sicherheit bzw. Cybersecurity im Fahrzeug.

DO-178C · Norm für Avionik-Software.

EN 50128 · Norm für Bahnsignal-Software.

IEC 61508 / 61131 · Normen für Industrieautomation und SPS-Programmierung.

HPC · High Performance Computer – zentraler Hochleistungsrechner im Fahrzeug.

ANHANG B · QUELLEN

- Bertrandt AG: Interne Projektdaten und Erfahrungswerte 2024–2026 (nicht veröffentlicht).
- Bertrandt Innovation Day: Software-defined Testing.
- dSPACE: KI-Lösungen für die SDV-Entwicklung, CES 2026.
- Eclipse Foundation / Eclipse SDV Working Group (2024–2025): Studienreihe zu Open-Source-Software in der Automobilindustrie: Driving Innovation & Building Safer Cars with Open Source Software (2024); Driving Efficiency and Sustainability (2024); Challenges Facing Open Source Software in the Automotive Ecosystem (2025).
- Eclipse SDV Working Group: Top SDV-Trends 2026.
- IoT Analytics: Software-defined Vehicles Adoption Report 2026.
- IPG Automotive: Virtual Engineering Workbenches, CES 2026.
- Omdia (2026): The 2026 SDV Reality Check: The Great Recalibration. Software-Defined Vehicle Survey 2026, gesponsert von Sonatus.
- Tahera, K.; Jowers, I.; Loureiro-Koechlin, C.; Eckert, C.; Lockett, H. (2023): Industrial Perspectives on the Adoption of Virtual Testing. In: Proceedings of the Design Society 3 (ICED23), S. 2305–2314. DOI: 10.1017/pds.2023.231.
- VDA (2022): VDA 710 – Software-in-the-Loop (SiL) Standardisierung. Version 07/2022.
- VDA (2025): VDA 711 – Software-in-the-Loop (SiL) Standardisierung, Phase II. Version 03/2025.
- Zäh, M. F.; Wünsch, G.; Hensel, T.; Lindworsky, A. (2006): Feldstudie – Virtuelle Inbetriebnahme. In: wt Werkstattstechnik online, 96 (10), S. 767–771.

ANHANG C · SERVICE-PORTFOLIO

- Virtueller Controller / vECU-Erzeugung.
- violis.io-Plattform mit CI/CD-Anbindung.
- SiL Operation: Integration, Simulation, Testen.
- Prozess- und IT-Consulting (ASPICE, IEC 62304, DO-178C).
- Souveräne Betriebsoptionen: StackIT, On-Premise, hybrid.

violis.io
by Bertrandt

IHR NÄCHSTER SCHRITT

Ein realer Anwendungsfall. Ein kompakter PoC. Keine langfristige Bindung.

Sie möchten konkret bewerten, ob virtuelles Testen für Sie funktioniert? Bertrandt spielt einen realen Anwendungsfall aus Ihrem Unternehmen mit violis.io durch.

Bertrandt AG · Unit Electronics & Virtual Testing Solution
© 2026 Bertrandt AG. Alle Rechte vorbehalten. violis.io ist eine Marke der Bertrandt-Gruppe.
Version 4.0 DE (03. August 2026) · Konzeption und Redaktion: Saupe Communication.