Normkonforme Werkzeugvalidierung – zielgerichtet und effizient

Standard-compliant tool validation – targeted and efficient

Katharina Prochazka | David Seider | Benedikt Wenzel

Die Normen EN 50129 und EN 50716 fordern eine Qualifizierung von Entwicklungswerkzeugen im Bahnbereich, um Fehler im Endprodukt zu verhindern. Grundlagen dazu erläutert ein vorangegangener Beitrag der Autoren [1]. Die Werkzeugvalidierung ist eine anerkannte Qualifizierungsmethode für kritische T2/T3-Werkzeuge. Der vorliegende Beitrag beschreibt die normativen Anforderungen, veranschaulicht die praktische Umsetzung anhand zweier Beispiele und gibt Tipps für eine zielgerichtete Durchführung.

1 Einleitung

Bei der Entwicklung von Bahnanwendungen (im Bahnumfeld eingesetzten Systemen bzw. Softwareanwendungen) werden häufig Werkzeuge (Tools) genutzt, die bestimmte Funktionen im Entwicklungsprozess ausführen oder unterstützen. Insbesondere bei sicherheitsrelevanten Anwendungen ist es daher wichtig, dass diese Werkzeuge keine Fehler in das Endprodukt einbringen, weshalb auch die Normen EN 50129 [2] und EN 50716 [3] für sicherheitsrelevante Anwendungen Anforderungen an die Werkzeugnutzung definieren. Die EN 50129 befasst sich dabei mit Werkzeugen in der Systementwicklung, die EN 50716 mit Werkzeugen in der Softwareentwicklung. Die Vorgaben sind in beiden Normen vergleichbar, allerdings sind die Anforderungen der EN 50716 an einigen Stellen detaillierter. Zudem verweist die EN 50129 für Softwareaspekte auf die damalige Softwareentwicklungsnorm EN 50128 [4]; diese wurde inzwischen durch die EN 50716 abgelöst. Daher konzentriert sich dieser Beitrag auf EN 50716.

Werkzeugklasse	Definition	Beispiele
T1	Werkzeug ohne Einfluss auf die Bahnanwendung.	Texteditor ohne Codegenerierung, Dateiablagesystem, Werkzeuge für Konfigurationsmanagement.
T2 (in EN 50129: Werkzeuge, die die Sicherheit indirekt beeinflussen)	Werkzeug, das für Tests oder Verifizierung der Bahnanwendung oder Prüfung von Konfigurationsdaten der Bahnanwendung genutzt wird. → Fehler im Werkzeug verursachen keinen Fehler in der Bahnanwendung, aber könnten einen eventuell bestehenden Fehler nicht aufdecken.	Werkzeug für statische Codeanalyse, Testwerkzeuge, Prüfwerkzeuge für Planungs-/Projektierungsdaten.
T3 (in EN 50129: Werkzeuge, die die Sicherheit direkt beeinflussen)	Werkzeug mit direktem Einfluss auf die Bahnanwendung bzw. ihre Daten. → Fehler im Werkzeug können unmittelbar zu Fehlern in der Bahnanwendung führen.	Entwurfswerkzeuge, Compiler, Codegenerator, Werkzeuge zur Berechnung von Parametern/Projektierungsdaten für sicherheitsrelevante Funktionen.

Tab. 1: Werkzeugklassen gemäß EN 50716 mit Beispielen

The EN 50129 and EN 50716 standards require the qualification of development tools in the railway sector so as to prevent errors in the final product. The fundamentals of this topic have already been explained in a previous article by the authors [1]. Tool validation is a recognised qualification method for critical T2/T3 tools. This article describes the normative requirements, illustrates the practical implementation using two examples and provides guidance for targeted and efficient accomplishment.

1 Introduction

Tools are often utilised to perform or support specific functions in the development process during the development of railway applications (i.e. systems or software applications used in the railway environment). It is therefore important for safety-relevant applications in particular that these tools do not introduce any errors into the final product, which is why the EN 50129 [2] and EN 50716 [3] standards also set requirements for tool usage in safety-relevant applications. EN 50129 deals with tools in system development, while EN 50716 deals with tools in software development. The specifications in both standards are comparable, although the requirements of EN 50716 are more detailed in some places. Furthermore, EN 50129 refers, for software aspects, to the then-applicable software development standard EN 50128 [4], which has since been replaced by EN 50716. This article will, therefore, focus on EN 50716.

Class	Definition	Examples
T1	a tool with no influence on the railway application.	a text editor without code generation, file storage system, tools for configuration management.
T2 (in EN 50129: tools that indirectly influence safety)	a tool used for testing or verifying a railway application or for checking the configuration data in a railway application. → an error in the tool does not cause an error in the railway application, but could fail to detect a potentially existing error.	a static code analysis tool, test tools, checking tools for planning/configuration data.
T3 (in EN 50129: tools that directly influence safety)	a tool with direct influence on the railway application or its data. → errors in the tool can lead directly to errors in the railway application.	design tools, compilers, code generators, tools for calculating parameters/configuration data for safety-relevant functions.

Tab. 1: The tool classes according to EN 50716 with examples

Die Softwareentwicklungsnorm EN 50716, welche neben der Norm EN 50128 [4] auch die EN 50657 [5] abgelöst hat, klassifiziert Werkzeuge je nach ihrem Einfluss auf die Bahnanwendung bzw. deren Daten als Werkzeuge der Klasse T1, T2 und T3 (Tab. 1). Für jede Toolklasse stellt die Norm Anforderungen, die zusätzlich berücksichtigen, ob die Werkzeuge für die Entwicklung von Funktionen mit Sicherheits-Integritätslevel (Safety Integrity Level, SIL) 1–4 oder Basisintegrität verwendet werden.

Für T3-Werkzeuge und unter Umständen auch für T2-Werkzeuge, die bei der Entwicklung von Funktionen mit SIL 1–4 verwendet werden, ist eine Werkzeugqualifizierung erforderlich. Die Qualifizierung ist der Prozess, der nachweist, dass ein Werkzeug geeignet ist und korrekt funktioniert; dies beinhaltet die Begründung, warum das Werkzeug geeignet ist und wie geplant im Entwicklungsprozess eingesetzt werden darf.

Die EN 50129 und EN 50716 lassen verschiedene Wege für die Qualifizierung zu, darunter die Argumentation auf Basis erfolgreicher Nutzungshistorie („proven in use“), Tooldiversität (Verwendung zweier verschiedener Tools und Vergleich der Ergebnisse) oder die Verifizierung des Werkzeug-Outputs bei jeder Verwendung. Ein weiterer Ansatz, der auf die Absicherung des Werkzeugs selbst abzielt, ist die Werkzeugvalidierung. Bei der Validierung wird das Werkzeug einer umfassenden Prüfung in Form von Tests und Analysen unterzogen. Sie ist besonders in folgenden Fällen sinnvoll:

- bei Werkzeugen, bei denen eine positive Nutzungshistorie nicht verfügbar oder nachweisbar ist (z. B. wegen geringer Einsatzhäufigkeit bahnspezifischer Funktionen)
- bei Werkzeugen, die in mehreren Entwicklungsprojekten mit verschiedenen Toolketten eingesetzt werden sollen
- bei eigens neu entwickelten Werkzeugen, für die eine Spezifikation mit moderatem Aufwand erstellt werden kann
- bei komplexen Werkzeugen mit einem hohen Risiko der Fehlbedienung, für die konkrete, prüfbare Anforderungen existieren
- bei Werkzeugen, für welche keine der anderen Qualifizierungsmethoden mit einem angemessenen Aufwand anwendbar ist (siehe Abschnitt 3.1.1).

2 Normative Vorgaben zur Werkzeugvalidierung

Die zentralen Anforderungen an die Werkzeugvalidierung sind in der Norm EN 50716 im Abschnitt 6.7.4.5 (bzw. in der EN 50129 im Abschnitt 6.3 unter dem Punkt „c) Analyse und Tests der Tools“) festgelegt. Die Norm gibt dabei folgende Punkte vor, lässt aber bewusst Spielraum für die konkrete Umsetzung:

- Angabe des Umfangs der Validierung: Das validierte Werkzeug und die zugehörige Dokumentation (z. B. Handbuch) müssen mit ihren konkreten Versionen identifiziert werden. Außerdem muss angegeben werden, welche Funktionen des Werkzeugs geprüft wurden.
- Testfälle: Bei der Validierung genutzte Testfälle und deren Ergebnisse müssen aufgezeichnet werden.
- Prüfumgebung: Die bei der Werkzeugvalidierung genutzten Werkzeuge und Geräte sind anzugeben.
- Validierungsbericht: Die Aktivitäten und Ergebnisse der Validierung müssen in einem Bericht dokumentiert werden inklusive eines abschließenden Urteils über die Eignung des Werkzeugs. Falls die Validierung nicht bestanden ist, sind die Gründe anzuführen.

Diese Vorgaben zeigen, dass eine sorgfältige Dokumentation aller durchgeführten Schritte entscheidend ist. Es muss eine zum Werkzeug passende Validierungsstrategie gewählt werden, z. B. ein Testkonzept für die systematische Abdeckung der Anforderungen. Tests sind ein zentraler Bestandteil, da sie im Allgemeinen einen stärke-

The EN 50716 software development standard, which has replaced the EN 50128 [4] and EN 50657 [5] standards, classifies tools as Class T1, T2 and T3 tools according to their influence on a railway application or its data (tab. 1). The standard sets out requirements for each tool class that also consider whether the tools are used for the development of functions with Safety Integrity Level (SIL) 1–4 or Basic Integrity.

Tool qualification is required for T3 tools and, under certain circumstances, also for T2 tools used in the development of functions with SIL 1–4 classification. Qualification is a process that demonstrates that a tool is suitable and functioning correctly; this includes the justification for why the tool is suitable and that it may be used as planned in the development process.

EN 50129 and EN 50716 permit various qualification paths, including argumentation based on a successful use history (“proven in use”), tool diversity (using two different tools and comparing the results) or the verification of the tool output with each use. Another approach, which aims to secure the tool itself, involves tool validation. During validation, the tool is subjected to a comprehensive examination in the form of tests and analyses. This is particularly useful in the following cases:

- for tools where a positive use history is not available or cannot be proven (e.g. due to the infrequent use of railway-specific functions).
- for tools that are to be used in multiple development projects with different toolchains.
- for newly developed tools for which a specification can be created with moderate effort.
- for complex tools with a high risk of incorrect operation, for which concrete, testable requirements exist.
- for tools where none of the other qualification methods is applicable with reasonable effort (see section 3.1.1).

2 Normative requirements for tool validation

The central requirements for tool validation are defined in section 6.7.4.5 of the EN 50716 standard (and in section 6.3 of EN 50129 under item “c) Tool proven by analysis and testing”). The standard specifies the following points, but deliberately leaves room for the specific implementation:

- specifying the validation scope: the validated tool and its associated documentation (e.g., manual) must be identified with its specific versions. The tested tool functions must also be stated.
- the test cases: the test cases used for tool validation and their results must be documented.
- the test environment: the tools and equipment used for the tool validation must be specified.
- the validation report: the validation activities and results must be documented in a report, including a final judgment on the suitability of the tool. If the validation is not passed, the reasons must be stated.

These requirements show that careful documentation of all the performed steps is crucial. A validation strategy must be chosen that is appropriate for the tool, for example, a test concept for the systematic coverage of the requirements. Tests are the central component, as they, in general, provide stronger evidence than purely static analyses. At the same time, it is often not practical to test every conceivable function of a complex tool.

ren Nachweis als rein statische Analysen liefern. Gleichzeitig ist es oft nicht praktikabel, jede erdenkliche Funktion eines komplexen Werkzeugs zu testen.

3 Praxisbeispiele

Um die Umsetzung der normativen Vorgaben in der Praxis zu veranschaulichen, werden im Folgenden zwei Beispiele für Werkzeugvalidierungen beschrieben, wobei das erste systematisch dargestellt wird, während beim zweiten der Fokus auf zusätzliche Besonderheiten gelegt wird.

3.1 Validierung des Tools „Gamma-sim" zur Berechnung von ETCS-Bremskurvenparametern nach EN 17997

Das erste Beispiel betrachtet die Validierung des Tools „Gamma-sim", welches bei der Firma quattron GmbH intern entwickelt wurde (siehe Projektbeschreibung [6]). Es ist Teil des Prozesses zur Erstellung von ETCS-Bremskurvenparametern für sogenannte Gamma-Züge.

3.1.1 Werkzeug und Anwendungsfall

Das Tool „Gamma-sim" wird genutzt, um mittels Monte-Carlo-Simulationen ETCS-Bremskurvenparameter für Gamma-Züge zu berechnen. Die Berechnungen basieren auf den mathematischen Grundlagen der Norm EN 17997 [7]. Konkret leitet „Gamma-sim" den fahrzeugspezifischen Korrekturfaktor K_{dry}(C,V,EBCL) für trockene Schienen unter Berücksichtigung stochastischer Schwankungen ab. Dieser Faktor gibt das Verhältnis der sicheren zur nominalen Schnell-

3 Real-world examples

Two real-world examples of tool validations are described below in order to illustrate the implementation of the normative requirements in practice. The first example is described in detail, while the second one focuses on additional specifics.

3.1 Validating the "Gamma-sim" tool for calculating ETCS braking curve parameters according to EN 17997

The first example considers the validation of the "Gamma-sim" tool, which has been developed internally at quattron GmbH. It is part of the process for creating ETCS braking curve parameters for so-called gamma trains (see project description [6]).

3.1.1 The tool and its use

The "Gamma-sim" tool is used to calculate ETCS braking curve parameters for Gamma trains by means of Monte Carlo simulations. The calculations are based on the mathematical principles of the EN 17997 [7] standard. Specifically, "Gamma-sim" derives the rolling stock correction factor K_{dry}(C,V,EBCL) for dry rails while considering the stochastic deviations. This factor indicates the ratio of the safe to the nominal emergency brake deceleration. Technically, the tool is implemented as a Python script via a Docker container (fig. 1).

The following assumptions were made given that the "Gamma-sim" tool calculates the parameters for train braking performance:

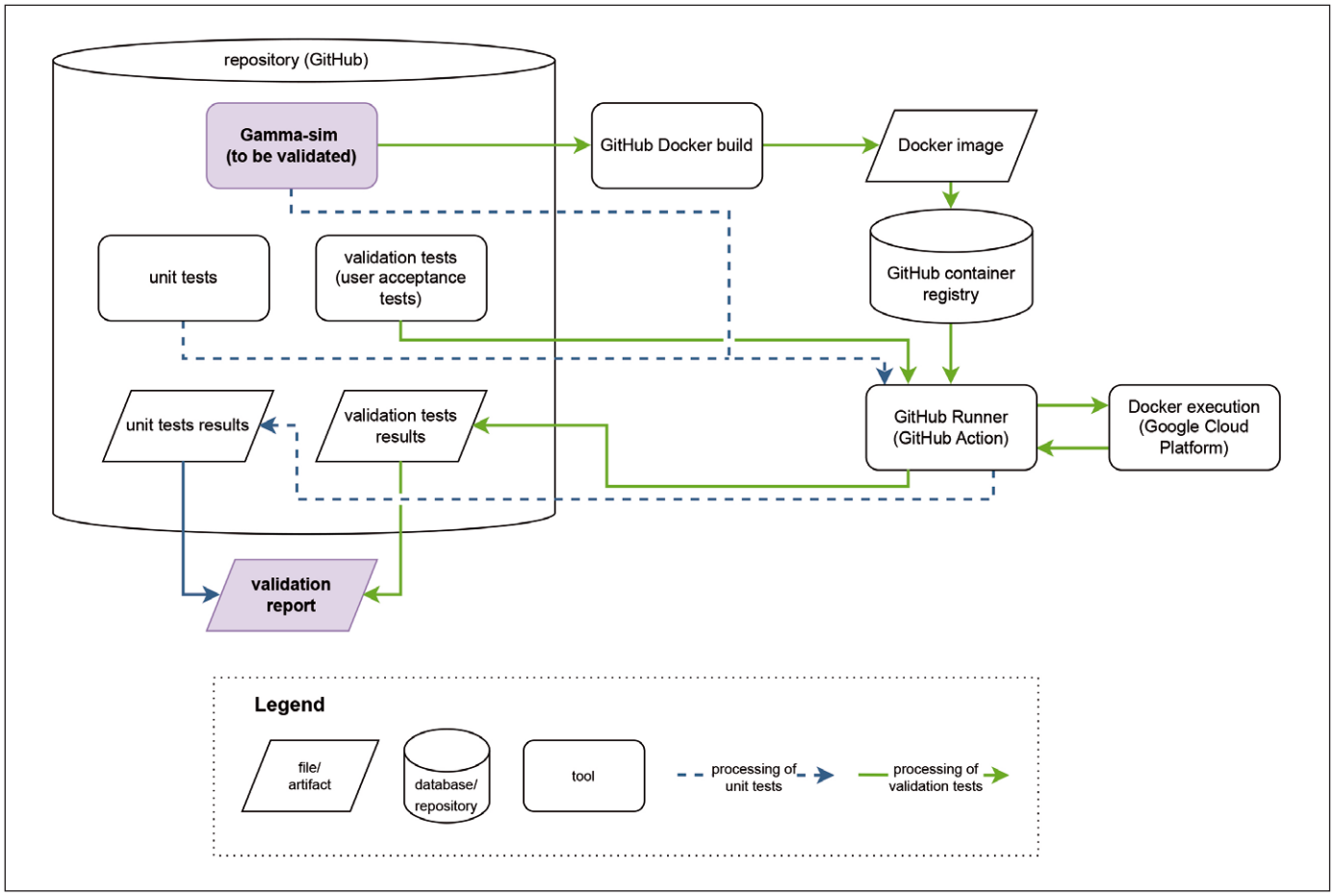

Bild 1: Validierungsaufbau und Werkzeugkette für das Tool „Gamma-sim"
Fig. 1: The validation setup and toolchain for the "Gamma-sim" tool

bremsverzögerung an. Technisch ist das Tool als Python-Skript in einem Docker-Container umgesetzt (Bild 1).
Da das Tool „Gamma-sim" Parameter für das Bremsvermögen eines Zuges berechnet, wurde von folgenden Annahmen ausgegangen:

- Das Werkzeug hat einen direkten Einfluss auf das Endprodukt (Bremskurvenparameter des Zuges) und wird daher gemäß EN 50716 als Werkzeugklasse T3 eingestuft.
- Das Werkzeug wird im Entwicklungsprozess für eine Funktion mit SIL 1 oder höher eingesetzt.

Für die daher notwendige Qualifizierung wurde die Methode der Werkzeugvalidierung gewählt, da sich alternative Methoden der EN 50716 als nicht geeignet erwiesen:

- Eine Chronik erfolgreicher Verwendungen war nicht vorhanden für diese Neuentwicklung.
- Tooldiversität war wegen der stochastischen Natur der Ergebnisse der Monte-Carlo-Simulation nicht mit vertretbarem Aufwand nutzbar.
- Eine Konformität des Entwicklungsprozesses des Werkzeugs mit einem bestimmten SIL erschien im Hinblick auf das Verhältnis von Aufwand und Nutzen nicht zielführend.
- Verifizierung des Werkzeug-Outputs war durch die Randomisierung ebenfalls nicht praktikabel.

3.1.2 Umsetzung der Validierung

Die Validierung von „Gamma-sim" wurde in mehreren Schritten durchgeführt (Bild 2), um die normativen Anforderungen effizient zu erfüllen:

- The tool has a direct influence on the final product (the train's braking curve parameters) and is therefore classified as a T3 tool according to EN 50716.
- The tool is used in the development process for a function with SIL 1 or higher.

Since qualification is mandatory for this tool, the tool validation method was chosen, as the alternative methods from EN 50716 proved to be unsuitable:

- A history of successful use was not available for this newly developed tool.
- Due to the stochastic nature of the Monte Carlo simulation results, tool diversity could not be implemented with reasonable effort.
- Compliance of the tool's own development process with a specific SIL was not deemed cost-effective.
- Verification of the tool output was also not considered practical due to the randomisation.

3.1.2 Performing the validation

The validation of "Gamma-sim" was carried out in several steps so as to efficiently meet the normative requirements (fig. 2):

1. Creating the basic documentation: First, a requirements specification for the tool and a software development plan were created. This established a traceable basis for all further activities.

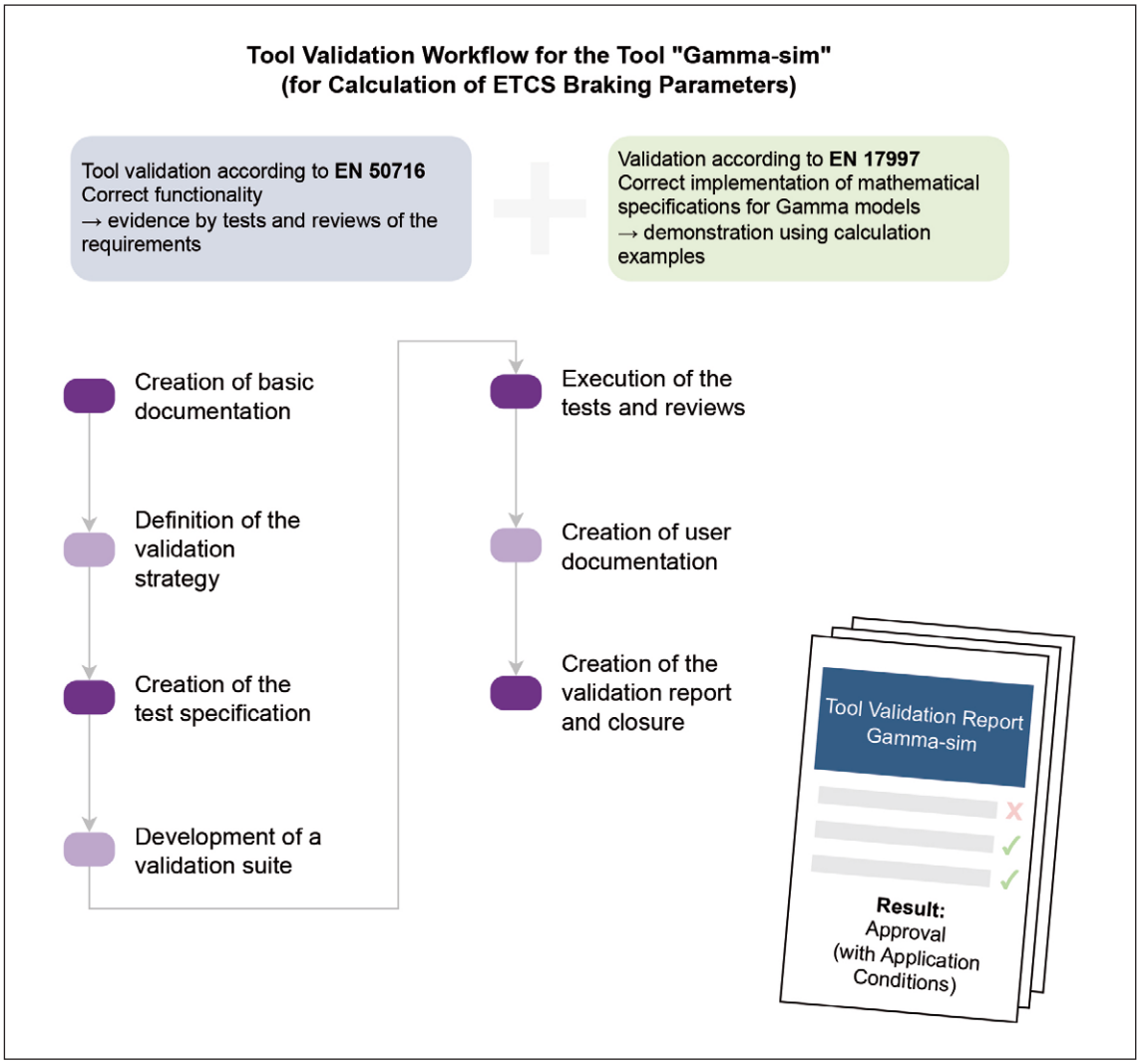

Bild 2: Ablauf der Werkzeugvalidierung für das Tool „Gamma-sim"
Fig. 2: The steps in the tool validation for "Gamma-sim"

1. Erstellung der grundlegenden Dokumentation: Zunächst wurden eine Anforderungsspezifikation für das Werkzeug und ein Softwareentwicklungsplan erstellt. Dies schuf eine nachvollziehbare Grundlage für alle weiteren Aktivitäten.
2. Festlegung der Validierungsstrategie: Als Nachweismethode wurde eine Kombination von Tests und Reviews gewählt. Dabei wurden zwei Aspekte berücksichtigt:
   - Validierung nach EN 50716 (Korrekte Funktionalität des Werkzeugs): Der Nachweis erfolgt durch Tests und Reviews.
   - Validierung nach EN 17997 (Korrekte Umsetzung der mathematischen Vorgaben für Gamma-Modelle): Der Nachweis erfolgt anhand von Rechenbeispielen für Zugkonfigurationen. Die einzuhaltende Validierungsprozedur ist in Kapitel 10 der EN 17997 beschrieben.
3. Erstellung der Testspezifikation: Für jede Anforderung wurde die Nachweismethode (Test / Review) festgelegt, mit Präferenz für Tests. Nicht testbare Anforderungen wurden per Code-Review geprüft. Aus den Anforderungen wurden Testfälle und Review-Vorgaben abgeleitet und in einer Testspezifikation zusammengefasst. Die Testfälle wurden so konzipiert, dass Positiv- und Negativtests durchgeführt wurden.
4. Entwicklung einer Validierungssuite (Sammlung von implementierten Testfällen gemäß Testspezifikation inkl. Testumgebung): Um die Tests effizient und wiederholbar zu gestalten, wurde eine weitgehend automatisierte Validierungssuite entwickelt. Diese umfasst die Testfälle selbst und eine Umgebung für die Ausführung und Auswertung von Tests. Die Validierungssuite wurde in den bestehenden Workflow zur Nutzung des Tools eingefügt (Bild 1).
5. Durchführung der Prüfungen: Tests und Reviews wurden durchgeführt und in einem Testbericht dokumentiert.
6. Erstellung der Nutzerdokumentation: Parallel zu den Prüfaktivitäten wurden ein Handbuch und ein Release-Notes-Dokument erstellt. Aus den in Schritt 5 gefundenen Abweichungen oder Auffälligkeiten wurden Anwendungsbedingungen für einen sicheren Einsatz des Werkzeugs abgeleitet und in der Nutzerdokumentation erfasst. Das Release-Notes-Dokument gibt u. a. die Werkzeugversion an und referenziert alle zugehörigen Dokumente.
7. Erstellung des Validierungsberichts und Abschluss der Validierung: Nach erfolgreicher Durchführung aller Tests und Reviews wurden die Ergebnisse im Werkzeugvalidierungsbericht zusammengefasst. Zusätzlich zu den Tests / Reviews wurden folgende Aspekte bewertet:
   - Eignung der Dokumentation
   - Erfüllung der Anforderungen
   - Einhaltung des Softwareentwicklungsplans
   - Einhaltung der Normvorgaben (EN 50716 und EN 17997)

Der Bericht dient für die validierte Toolversion als zentraler Nachweis, dass das Werkzeug für den vorgesehenen Einsatzzweck geeignet ist. Das Ergebnis war eine Freigabe des Tools inkl. Anwendungsbedingungen.

Alle Dokumente wurden als Teil des Release-Packages für das Werkzeug abgelegt, um das Ergebnis der Werkzeugvalidierung einfach zugänglich zu machen. Dadurch ist auch auf den ersten Blick sichtbar, dass es sich um eine validierte Werkzeugversion handelt.

3.2 Validierung einer Werkzeugkette aus Modellierungstool, Codegenerator und Compiler

Ein weiteres typisches Szenario ist der Einsatz einer Kette von kommerziell erworbenen Werkzeugen („Commercial Off-The-Shelf" / COTS) beispielsweise für die modellbasierte Entwicklung.

2. Defining the validation strategy: A combination of tests and reviews was chosen as the method for providing evidence. Two aspects were considered:
   - Validation according to EN 50716 (the correct functionality of the tool): the evidence is provided by tests and reviews.
   - Validation according to EN 17997 (the correct implementation of the mathematical specifications for gamma models): the evidence is provided using calculation examples for train configurations. The validation procedure to be followed is described in Chapter 10 of EN 17997.
3. Creating the test specification: The verification method (test / review) was defined for each requirement, with a preference for tests. Non-testable requirements were checked by code review. The test cases and review guidelines were derived from the requirements and summarised in a test specification. The test cases were designed to include positive and negative tests.
4. Developing a validation suite (a set of implemented test cases according to the test specification, including the test environment): A largely automated validation suite was developed so as to make the tests efficient and repeatable. This included the test cases themselves and an environment for executing and evaluating tests. The validation suite was integrated into the existing tool workflow (fig. 1).
5. Executing the checks: Tests and reviews were carried out and documented in a test report.
6. Creating the user documentation: A manual and release notes document were created in parallel with the testing activities. The application conditions for the safe usage of the tool were derived and documented from the findings in Step 5. The release notes document specifies the tool version and references all the associated documents.
7. Creating the validation report and completing the validation: Once all the tests and reviews had been successfully completed, the results were summarised in the tool validation report. The following aspects were evaluated in addition to the tests / reviews:
   - the suitability of the documentation
   - the fulfilment of the requirements
   - the compliance with the software development plan
   - the compliance with the standard specifications (EN 50716 and EN 17997)

The tool was finally approved (together with a set of application conditions to be considered whenever the tool is used). The report serves as the central evidence that the validated tool version is suitable for its intended purpose.

All the documents were stored as part of the tool's release package in order to make the results of the tool validation easily accessible. This also makes it immediately apparent that it is a validated tool version.

3.2 Validating a toolchain comprising a modelling tool, a code generator and compilers

Another typical scenario is the use of a chain of commercial off-the-shelf (COTS) tools, for example for model-based development. This anonymised example illustrates the validation of such a toolchain.

3.2.1 The toolchain and its use

A model-based approach was adopted within the context of developing safety-related system functions (SIL 1 or higher).

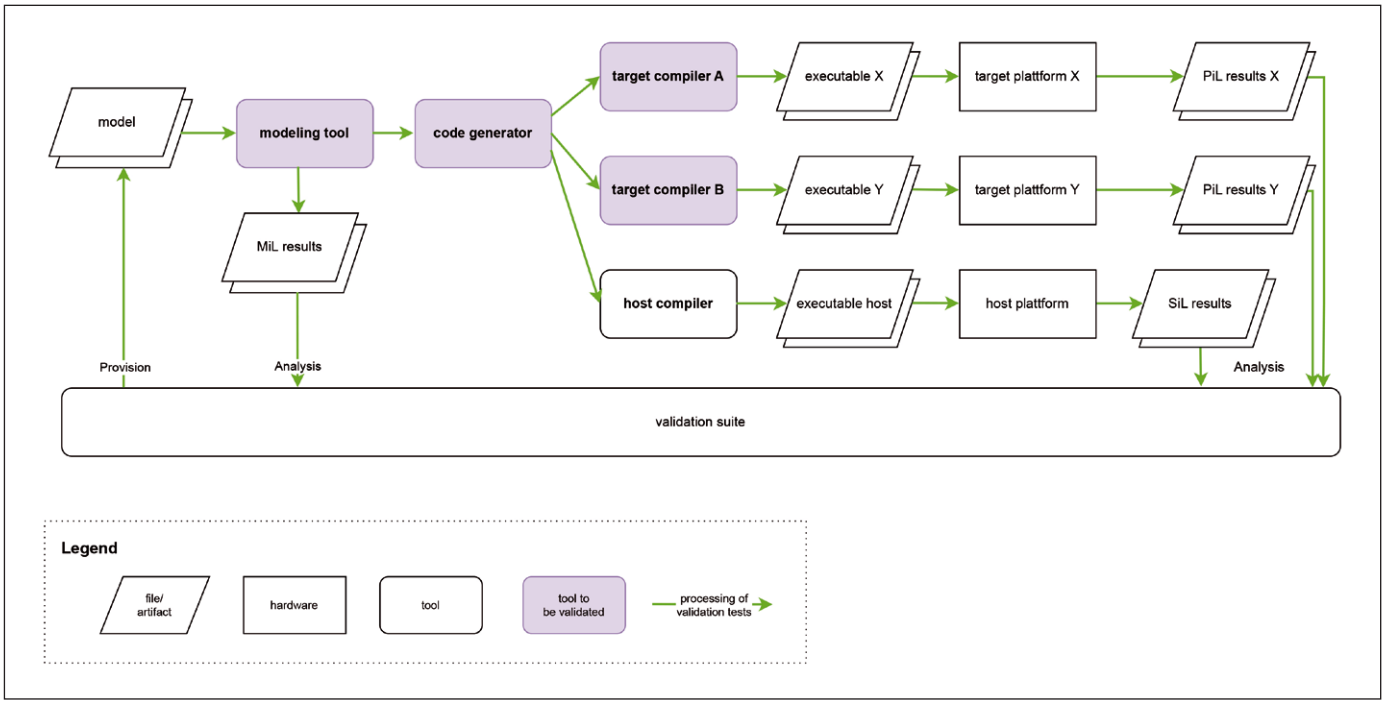

Bild 3: Validierungsaufbau und Werkzeugkette mit zu validierenden Tools
Fig. 3: The validation setup and toolchain with tools to be validated

Dieses anonymisierte Beispiel illustriert die Validierung einer solchen Toolkette.

3.2.1 Werkzeuge und Anwendungsfall

Im Rahmen der Entwicklung von sicherheitsrelevanten Systemfunktionen (SIL 1 oder höher) wurde auf eine modellbasierte Vorgehensweise gesetzt.
Dabei kam eine Werkzeugkette zum Einsatz, die aus einem kommerziellen Modellierungstool, einem darauf abgestimmten Codegenerator und verschiedenen Compilern bestand (Bild 3). Der Prozess sah vor, dass aus den in der Entwicklungsumgebung erstellten Modellen C/C++-Code generiert wurde. Dieser Code wurde anschließend für mehrere unterschiedliche Zielplattformen für Steuergeräte kompiliert, um ihn dort bereitzustellen und auszuführen. Da sowohl der Codegenerator als auch die Compiler direkten Einfluss auf den ausführbaren Code hatten und das Modellierungstool die Grundlage für die Codegenerierung bildete, war eine Qualifizierung zwingend erforderlich.

3.2.2 Umsetzung der Validierung

Für die Qualifizierung wurde die Methode der Werkzeugvalidierung gewählt. In der Umsetzung waren insbesondere eine detailliert ausgearbeitete Validierungsstrategie und ein hoher Automatisierungsgrad von zentraler Bedeutung.
Die Validierungsstrategie zielte darauf ab, die Korrektheit der gesamten Kette – vom Modell bis zum ausführbaren Code auf der Zielhardware – nachzuweisen. Um die „kombinatorische Explosion" möglicher Testfälle zu vermeiden, wurden geeignete und explizit beschriebene Heuristiken angewendet. Der Fokus lag auf folgenden Aspekten:

- Mathematische Operationen: Prüfung aller relevanten verwendeten mathematischen Funktionen und Operationen unter Berücksichtigung der möglichen Anzahl der Operanden.
- Datentypen: Abdeckung relevanter Kombinationen von Datentypen.

A toolchain consisting of a commercial modelling tool, a corresponding code generator and various compilers was used (fig. 3). The process involved generating C/C++ code from models created in the development environment. This code was then compiled for several different control-unit target platforms on which it was deployed and executed.
The qualification was mandatory, as both the code generator and the compilers had a direct influence on the executable code and the modelling tool formed the basis for code generation.

3.2.2 Performing the validation

The tool validation method was chosen for the qualification. A particularly detailed validation strategy and a high degree of automation were of central importance for executing the validation efficiently.
The validation strategy aimed to prove the correctness of the entire chain, from the model to the executable code on the target hardware. Suitable and explicitly described heuristics were applied so as to avoid a "combinatorial explosion" of the possible test cases. The focus was on the following aspects:

- mathematical operations: testing all the relevant mathematical functions and operations, considering the potential number of operands.
- data types: coverage of the relevant data type combinations.
- experience-based test cases: including test cases based on experience, such as special cases in fixed-point arithmetic.
- comparisons at different levels: the systematic comparison of the results at various levels, i.e. MiL (Model-in-the-Loop), SiL (Software-in-the-Loop) and PiL (Processor-in-the-Loop), in order to precisely locate any deviations.

A largely automated validation suite was developed so as to make the validation efficient and repeatable. This suite handled the central tasks in the validation process:

- the automatic generation of most of the test cases (models) based on a formal specification.

- Erfahrungsbasierte Testfälle: Einbeziehung von Testfällen, die auf Erfahrungen beruhen, wie beispielsweise Sonderfälle bei der Fixpunkt-Arithmetik.
- Vergleich auf verschiedenen Ebenen: Systematischer Vergleich der Ergebnisse auf verschiedenen Ebenen: MiL (Model-in-the-Loop), SiL (Software-in-the-Loop) und PiL (Processor-in-the-Loop), um Abweichungen präzise lokalisieren zu können.

Um die Validierung effizient und wiederholbar zu gestalten, wurde eine weitgehend automatisierte Validierungssuite entwickelt. Diese Suite übernahm zentrale Aufgaben des Validierungsprozesses:

- Automatische Generierung eines Großteils der Testfälle (Modelle) aus einer formalen Spezifikation.
- Automatisierte Ausführung der Tests über die gesamte Werkzeugkette, inklusive Codegenerierung, Kompilierung und – für PiL-Tests – Flashen auf den Zielprozessor, Testlauf sowie Erfassung der Ergebnisse.
- Intelligenter Vergleich der Ergebnisse der verschiedenen Ebenen bzw. Plattformen.
- Zusammenfassende Darstellung der Testergebnisse und Unterstützung der Fehleranalyse durch Filterfunktionen.

Die Durchführung systematischer Tests führte zur Aufdeckung von Abweichungen. Diese wurden analysiert, um das verursachende Werkzeug (Modellierungstool, Codegenerator oder Compiler) zu identifizieren. Die gefundenen Fehler wurden an die jeweiligen Hersteller gemeldet und führten zu einer deutlichen Verbesserung der Werkzeuge. Sogar in den als sehr reif eingeschätzten Compilern wurden Fehler aufgedeckt.

Alle Ergebnisse, die Analysen der Abweichungen sowie die daraus abgeleiteten Anwendungsbedingungen für den sicheren Werkzeugeinsatz wurden in einem detaillierten Werkzeugvalidierungsbericht dokumentiert. Dieser Bericht enthielt eine abschließende Freigabeempfehlung. Durch die Validierung wurde somit der Einsatz der gesamten Werkzeugkette abgesichert.

4 Good practices

Um die Werkzeugvalidierung effizient und zielgerichtet zu gestalten, haben sich in der Praxis einige Vorgehensweisen bewährt. Die folgenden Empfehlungen können dabei helfen, den Aufwand moderat zu halten und gleichzeitig die Qualität und Sicherheit zu maximieren.

4.1 Bewährte Dokumentenvorlagen nutzen

Die Verwendung von erprobten Dokumentenvorlagen für den Werkzeugvalidierungsplan und den Werkzeugvalidierungsbericht ist ein entscheidender Effizienzfaktor. Solche Vorlagen fungieren als Checkliste und stellen sicher, dass alle normativ geforderten und für die Nachweisführung relevanten Aspekte berücksichtigt werden. Eine klare Trennung zwischen dem Validierungsplan, der die Strategie und das Vorgehen festlegt, und den Validierungsberichten, welche die Durchführung der Werkzeugvalidierung und deren Ergebnisse dokumentieren, hat sich ebenfalls als vorteilhaft erwiesen.

4.2 Eine durchdachte Validierungsstrategie entwickeln

Der größte Hebel zur Effizienzsteigerung liegt in einer sorgfältig ausgearbeiteten Validierungsstrategie. Anstatt zu versuchen, jede erdenkliche Funktion zu testen, sollte der Fokus auf den tatsächlich im Projekt genutzten und relevanten Funktionen liegen. Dabei muss eine bewusste Entscheidung getroffen werden, welche Anforderungen durch Tests und welche durch Reviews nachgewiesen werden. Um eine „kombinatorische Explosion" bei der Anzahl der Testfälle zu vermeiden, ist es essenziell, nachvollziehbare

- the automated execution of the tests across the entire toolchain, including code generation, compilation, and – for PiL tests – flashing to the target processor, test execution and result collection.
- smart comparisons of the results from the different levels or platforms.
- a summarised presentation of the test results and support for error analysis through appropriate filtering.

The execution of the systematic tests led to the discovery of deviations. These were analysed to identify the causative tool (modelling tool, code generator or compiler). The identified errors were reported to the respective manufacturers and led to a significant improvement in the tools. Errors were even discovered in some compilers that were considered very mature.

All the results, the analyses of the deviations and the derived application conditions for the safe application of the tools were documented in a detailed tool validation report. This report included a final release recommendation. As such, the validation safeguarded the use of the entire toolchain.

4 Good practices

Some procedures have proven effective in practice so as to make tool validation efficient and targeted. The following recommendations can help to keep the effort moderate while maximising quality and safety.

4.1 Use proven document templates

The use of proven document templates for the tool validation plan and the tool validation report is a crucial efficiency factor. Such templates serve as a checklist and ensure that all the aspects that are required by the standard and are relevant for providing evidence have been considered. A clear separation between the validation plan, which defines the strategy and procedure, and the validation reports, which document the execution of the tool validation and its results, has also proven to be advantageous.

4.2 Develop a carefully crafted validation strategy

The greatest lever for increasing efficiency lies in a carefully crafted validation strategy. Instead of trying to test every conceivable function, the focus should be on the functions that are actually used and are relevant in the project. A conscious decision must be made as to which requirements are to be verified by tests and which by reviews. It is essential to apply comprehensible and clearly documented heuristics to the test case selection so as to avoid a "combinatorial explosion" in the number of test cases.

4.3 Automate the validation using a validation suite

The development of an automated validation suite is recommended for all the tests that can be performed automatically. Such a suite not only increases the efficiency of the initial validation, but is also invaluable when tool updates are pending and re-validations must be performed. Furthermore, synergies can arise by reusing parts of the validation suite for other assurance activities in the development process.

4.4 Create a basis for validation in the case of insufficient manufacturer documentation

Sometimes the specifications or manual provided by the tool manufacturer constitute an insufficient basis for validation. In

und klar dokumentierte Heuristiken für die Testfallauswahl anzuwenden.

4.3 Validierung durch eine Validierungssuite automatisieren

Für alle maschinell durchführbaren Tests empfiehlt sich die Entwicklung einer automatisierten Validierungssuite. Eine solche Suite steigert nicht nur die Effizienz der erstmaligen Validierung, sondern ist besonders dann von unschätzbarem Wert, wenn Werkzeug-Updates anstehen und Re-Validierungen durchgeführt werden müssen. Darüber hinaus können sich Synergien ergeben, indem Teile der Validierungssuite auch für andere Absicherungsaktivitäten im Entwicklungsprozess wiederverwendet werden.

4.4 Validierungsgrundlagen bei unzureichender Herstellerdokumentation schaffen

Manchmal ist die vom Hersteller bereitgestellte Spezifikation oder das Handbuch eines Werkzeugs ungenügend, um eine Validierung darauf aufzubauen. In solchen Fällen ist es notwendig, diese Grundlage selbst zu schaffen, indem eine eigene Beschreibung der relevanten Funktionen und ihres erwarteten Verhaltens erstellt wird. Dieses Dokument dient dann als Referenz für die Test- und Review-Aktivitäten bei der Werkzeugvalidierung.

4.5 Gestaltungsmöglichkeiten bei Eigenentwicklungen nutzen

Wird ein Werkzeug eigens entwickelt, bieten sich zusätzliche Möglichkeiten, die Validierung von Beginn an zu unterstützen und die Werkzeugqualität zu steigern:

- Geeignetes Entwicklungsvorgehen: Die Anwendung eines strukturierten Entwicklungsprozesses ist fundamental. Dazu gehören das Formulieren von Anforderungen, die Umsetzung einer modularen Software-Struktur mit klaren Schnittstellen (Inputs / Outputs) und die Möglichkeit einer Ansteuerung des Tools über die Kommandozeile. Mindestens sollten Tests des Werkzeugs vorgesehen werden, welche die Hauptfunktionen prüfen und deren erfolgreiche Durchführung eine Voraussetzung für die Freigabe einer Werkzeugversion ist (Releasetests).
- Frühe Erprobung des Handbuchs: Das Benutzerhandbuch sollte frühzeitig von Personen erprobt werden, die nicht an der Entwicklung beteiligt waren. Dies entspricht einem frühen „User-Acceptance-Test“ und hilft, die Verständlichkeit und Anwendbarkeit zu verbessern.
- Frühzeitige Einbindung des Werkzeugvalidierers: Der für die Validierung verantwortliche Experte sollte bereits während der Werkzeugentwicklung einbezogen werden. Durch parallel laufende Validierungsaktivitäten kann die Qualität des Werkzeugs signifikant verbessert werden.

5 Fazit

Zusammenfassend lässt sich festhalten:

- Werkzeugvalidierung ist in einigen Fällen die normkonforme Qualifizierungsmethode, die ökonomisch, fachlich und personell am sinnvollsten ist.
- Die Methode ist flexibel einsetzbar, sowohl für Einzelwerkzeuge (selbst entwickelte oder zugekaufte Tools) als auch für komplexe, heterogene Werkzeugketten.
- Durch Verwendung bewährter Dokumentenvorlagen und Auswahl einer geeigneten Validierungsstrategie lässt sich der Aufwand einer Werkzeugvalidierung moderat halten.
- In einigen Fällen ist es möglich, Synergien dadurch zu nutzen, dass ein Teil der Maßnahmen auch zur generellen Qualitätssi-

such cases, it is necessary to create this basis by preparing a separate description of the relevant functions and their expected behaviour. This document then serves as a reference for the test and review activities during tool validation.

4.5 Leverage the opportunities in in-house tool development

If a tool is developed in-house, there are additional opportunities to support the validation from the outset and to increase the tool's quality:

- a suitable development process: the application of a structured development process is fundamental. This includes formulating requirements, implementing a modular software structure with clear interfaces (inputs / outputs) and the ability to control the tool via a command-line interface. At a minimum, tool tests should be planned to check the main functions and their successful execution should be a prerequisite for the release of a tool version (release tests).
- early testing of manual: the user manual should be tested early by individuals who were not involved in the development. This corresponds to an early “User Acceptance Test” and helps to improve clarity and usability.
- early tool validator involvement: the expert responsible for the validation should be involved during the tool's development. The quality of the tool can be significantly improved through parallel validation activities.

5 Conclusion

To summarise:

- In some cases, tool validation is the standard-compliant qualification method that makes the most sense economically, technically and from a staffing perspective.
- The method can be used flexibly both for individual tools (self-developed or commercial tools) and for complex, heterogeneous toolchains.
- The effort can be kept moderate by using proven document templates and selecting a suitable validation strategy.
- In some cases, it is possible to leverage synergies by using some of the measures for general quality assurance as well (e.g. when testing the railway application under development).
- Tool validation provides evidence of the correct functionality of the used tools, which leads to high application efficiency through the demonstrated inherent reliability of the tools, as, for example, the complex verification of the tool output with each individual use can be omitted.

Thus, tool validation makes a sustainable contribution to increasing safety and efficiency in the development and use of systems for the railway sector. ■

cherung dienen kann (z. B. beim Testen der zu entwickelnden Bahnanwendung).
- Werkzeugvalidierung stellt die korrekte Funktionalität der eingesetzten Werkzeuge sicher und führt durch die nachgewiesene Zuverlässigkeit der Tools zu einer hohen Anwendungseffizienz, da beispielsweise auf eine aufwendige Verifizierung des Werkzeug-Outputs bei jeder einzelnen Nutzung verzichtet werden kann.

Damit leistet die Werkzeugvalidierung einen nachhaltigen Beitrag zur Steigerung von Sicherheit und Effizienz in der Entwicklung und dem Einsatz von Systemen für den Bahnbereich. ■

AUTOREN | AUTHORS

Dr. Katharina Prochazka
Systemingenieurin / *System Engineer*
quattron Austria GmbH
Anschrift / *Address*: Marxergasse 1b, A-1030 Wien
E-Mail: katharina.prochazka@quattron.com

Dr. David Seider
Software Consultant
quattron GmbH
Anschrift / *Address*: Herzog-Wilhelm-Straße 19, D-80331 München
E-Mail: david.seider@quattron.com

Dr.-Ing. Benedikt Wenzel
Partner, Leitung quattron engineering / *Partner, Head of quattron engineering*
quattron GmbH
Anschrift / *Address*: Unter den Linden 21, D-10117 Berlin
E-Mail: benedikt.wenzel@quattron.com

LITERATUR | LITERATURE

[1] Prochazka, K., Seider, D., Wenzel, B.: „Normkonforme Qualifizierung unterstützender Werkzeuge – zielgerichtet und effizient", Eisenbahntechnische Rundschau 6/2026
[2] DIN EN 50129:2019 Bahnanwendungen – Telekommunikationstechnik, Signaltechnik und Datenverarbeitungssysteme – Sicherheitsbezogene elektronische Systeme für Signaltechnik; Deutsche Fassung EN 50129:2018 + AC:2019
[3] DIN EN 50716:2024 Bahnanwendungen – Anforderungen für die Softwareentwicklung; Deutsche Fassung EN 50716:2023 inkl. Berichtigung 1
[4] DIN EN 50128:2012 Bahnanwendungen – Telekommunikationstechnik, Signaltechnik und Datenverarbeitungssysteme – Software für Eisenbahnsteuerungs- und Überwachungssysteme; Deutsche Fassung EN 50128:2011 + DIN EN 50128/A1:2020 + DIN EN 50128/A2:2020
[5] DIN EN 50657:2017 Bahnanwendungen – Anwendungen für Schienenfahrzeuge – Software auf Schienenfahrzeugen; Deutsche Fassung EN 50657:2017 + DIN EN 50657/A1:2024
[6] Projekt Bremssimulation als Werkzeug der Klasse T3 für SIL > 0 nach EN 50716, https://www.quattron.com/referenz/bremssimulation-als-werkzeug-der-klasse-t3-fuer-sil-0-nach-en-50716/, 05.08.2026 um 13:37 Uhr
[7] DIN EN 17997:2025 Bahnanwendungen – Bremsen – Bestimmung der ETCS-Bremskurvenparameter für Gamma-Züge; Deutsche Fassung EN 17997:2025