Testfallgenerierungsgestützte Validierung und Verifikation von EULYNX-Spezifikationen

Validation and verification of EULYNX specifications supported by test case generation

Daniel Schwencke

Aktuell spezifizieren Initiativen wie EULYNX und RCA Schnittstellen und Teilsysteme künftiger Leit- und Sicherungstechnikarchitekturen. Da eine breite Nutzung der Spezifikationen als Standards beabsichtigt ist, ist eine hohe Qualität unabdingbar. Dieser Beitrag stellt eine Fallstudie vor, in der Aktivitäten zur Validierung und Verifikation der EULYNX-Weichenspezifikation auf Basis der Inspektion automatisch aus einem Modell des spezifizierten Verhaltens generierter Testfälle durchgeführt wurden. Die Methode, die die bereits modellbasierte Spezifikation ausnutzt, erwies sich als effizient und konnte verschiedene Probleme der Spezifikation ermitteln.

1 Einleitung

1.1 Die Bedeutung der Qualität von Systemspezifikationen

Die Qualität der Anforderungen ist von großer Bedeutung für eine kosteneffiziente und termingerechte Systementwicklung sowie die resultierende Produktqualität. Beispielsweise ist seit langem bekannt, dass in Softwareprojekten die Kosten von während des Betriebs gefundenen Fehlern 100-mal so groß wie von in der Anforderungsphase gefundenen Fehlern sein können [1]. Um solche Situationen zu vermeiden oder zu entschärfen, werden verschiedene Eigenschaften von Anforderungen verlangt und Entwicklungsprozesse genutzt; vor allem jedoch müssen Spezifikationen die beabsichtigte Funktionalität korrekt beschreiben. Zugleich ist bekannt, dass Anforderungsermittlung keine einfache Aufgabe ist und Spezifikationen durch Erfahrungsrückfluss reifen.

Aktuell spezifizieren Initiativen wie EULYNX [2] und Reference CCS Architecture (RCA) [3] Schnittstellen und Teilsysteme künftiger Architekturen für Leit- und Sicherungstechnik (LST). Aus einer Reihe von Gründen ist für sie die Erstellung von Spezifikationen hoher Qualität besonders bedeutsam und zugleich herausfordernd:

- Die Schnittstellen und Teilsysteme sind neu oder lagen zuvor im Verantwortungsbereich von Systemherstellern,
- Ziel der Spezifikationen ist, verschiedene europäische LST-Funktionalitäten zusammenzuführen,
- letztlich ist eine breite Nutzung als interoperable Standards beabsichtigt und
- bei der Spezifikation werden neue Vorgehensweisen verwendet.

In dieser Situation sind – neben definierten Spezifikationsprozessen – effektive und effiziente Methoden zur frühzeitigen Validierung und Verifikation (V&V) des spezifizierten funktionalen Verhaltens notwendig. Dieser Beitrag stellt eine Fallstudie vor, in der

Initiatives such as EULYNX and RCA are currently specifying the interfaces and subsystems for the future CCS system architectures. A high degree of quality is essential, as the specifications are intended to become widely used standards. This article reports on a case study that has performed validation and verification activities on the EULYNX point specification based on the inspection of test cases that have been automatically generated from a model of the specified behaviour. The method, which utilises the already model-based specification, has proved to be efficient and has been able to detect several specification issues.

1 Introduction

1.1 The importance of high-quality system specifications

The requirement quality is of utmost importance for cost-efficient and timely system development and the resulting product quality. For example, it has long been known that the cost of any errors in software projects found during operations may be 100 times higher than those found during the requirement phase [1]. Various requirement characteristics have been demanded and many development processes have been used in order to avoid or mitigate any such situations. Above all, however, all the specifications need to describe the intended functionality correctly. At the same time, it is also known that the determination of the requirements is a difficult task and that specifications mature through experiential feedback.

Initiatives such as EULYNX [2] and Reference CCS Architecture (RCA) [3] are currently specifying the interfaces and subsystems for the future command, control and signalling (CCS) system architectures. The production of high-quality specifications for them is both especially important and challenging for a number of reasons:

- the interfaces and subsystems are new or have been previously dealt with internally by the system suppliers;
- the specifications aim to combine different European CCS systems functionalities;
- they are eventually intended to become interoperable standards for large-scale use; and
- new specification approaches are applied.

In this situation, effective and efficient methods for the early validation and verification (V&V) of the specified functional behaviour are necessary in addition to well-defined specification processes. This article reports on a case study that has applied a method based on automated test case generation (TCG) for the V&V of a EULYNX specification.

für V&V einer EULYNX-Spezifikation eine Methode angewandt wurde, die auf automatisierter Testfallgenerierung (TFG) beruht.

1.2 Der Wert von generiertem explizitem Systemverhalten

Bei der Systemspezifikation ist es üblich, mehrere Anwendungsfälle zu definieren, bevor das Verhalten detailliert spezifiziert wird, und zuletzt zu verifizieren, dass das spezifizierte Verhalten diese Anwendungsfälle abdeckt. Dieses Vorgehen hat sich als effektiv zur Berücksichtigung der Hauptanwendungsfälle in Spezifikationen erwiesen, jedoch decken (a) die Anwendungsfälle nicht das gesamte zulässige Verhalten ab, können (b) Spezifikationsfehler bei der Detaillierung ursprünglich beabsichtigten Verhaltens entstehen und ist es möglich, dass (c) die Spezifikation zusätzliches unbeabsichtigtes Verhalten erlaubt. Aus diesen Gründen ist die Verifikation einer Spezifikation gegen Anwendungsfälle normalerweise nicht ausreichend; das gesamte spezifizierte Verhalten muss validiert und verifiziert werden. Neben einem direkten manuellen Review, bei dem es schwierig sein kann, komplexe Abhängigkeiten zu durchdringen, ist eine andere Möglichkeit, ein Verhaltensmodell aus der Spezifikation zu erzeugen und einen Testfallgenerator darauf anzuwenden. Dieser produziert eine – weniger komplexe – Testsuite, die dennoch das durch die Spezifikation erlaubte Verhalten in definierter Weise abdeckt. Solch eine Testsuite wird in der Regel beabsichtigtes, fehlerhaftes und unbeabsichtigtes Verhalten zugleich beinhalten, da weder Modell noch Testfallgenerator diese Begrifflichkeiten kennen. In der Tat bestätigen frühere Erfahrungen den Wert solch generierter Testsuiten für V&V von Spezifikationen [4]: *„Die […] generierten Testfälle lieferten […] wertvolle Rückmeldungen […] bzgl. nicht bedachter Szenarien, z. B. zu Wiederholungen exakt gleicher Nachrichten, Szenarien, in denen keine Reaktion des Systems erfolgen soll oder einer unvollständig spezifizierten Kombinatorik von Funktionalitäten.*“ Während der Schwerpunkt in [4] auf der Untersuchung der allgemeinen Anwendbarkeit der TFG für signaltechnische Systeme lag, knüpft die Fallstudie, über die der vorliegende Beitrag berichtet, an die zitierte Beobachtung an und wendet modellbasierte Testfallgenerierung und -inspektion – zum Zweck der V&V einer realen Systemspezifikation – tatsächlich an. Validierung bezieht sich dabei auf (die Abwesenheit) unbeabsichtigten Verhaltens; Verifikation auf während des Spezifikationsprozesses entstandene Fehler.

1.3 Aufbau des Beitrags

Das folgende Kapitel 2 stellt den Kontext der Fallstudie dar, indem es die EULYNX-Initiative und modellbasiertes Testen sowie ihr Zusammenspiel in der Fallstudie einführt. Ein Überblick und Details der verschiedenen Phasen des in der Fallstudie angewandten V&V-Prozesses werden in Kapitel 3 präsentiert. Kapitel 4 beschreibt die V&V-Ergebnisse. Schließlich wird in Kapitel 5 die Fallstudie diskutiert und ein Ausblick gegeben.

2 Kontext der Fallstudie

2.1 EULYNX

EULYNX ist eine europäische Initiative von aktuell 13 Eisenbahninfrastrukturmanagern (IM), die die Definition und Standardisierung von Stellwerksschnittstellen anstrebt. Eine Reihe von Schnittstellenspezifikationen wurden zusammen mit weiteren Dokumenten auf der EULYNX-Webseite [2] veröffentlicht, sodass sie in Ausschreibungen verwendet werden können. Ein wichtiges Ziel ist es, eine Reduktion der Lebenszykluskosten für künftige digitale signaltechnische Systeme zu erreichen und zugleich Innovation durch austauschbare Teilsysteme zu fördern.

Die Spezifikationen beinhalten eine detaillierte Beschreibung des an den Schnittstellen sichtbaren Systemverhaltens. Dies bringt die Notwendigkeit der Argumentation von Korrektheit und Security-Leveln

1.2 The value of generated explicit behaviours

Several use cases are usually defined during the system specification before the detailed behaviour is specified and the use cases are subsequently verified to ensure that they are covered by the specified behaviour. This procedure has proven effective for accommodating the main use case scenarios in a specification, but for all that (a) the use cases do not cover all the allowed behaviours, (b) the specification may introduce errors when detailing the originally intended behaviour and (c) the specification may introduce further undesired behaviour. For these reasons, the verification of a specification against use cases is usually not sufficient. The whole specified behaviour needs to be subjected to V&V. In addition to a direct manual review, which can have a hard time comprehending any complex dependencies, it is also possible to create a behavioural model from the specification and to apply a test case generator to it. This produces a less complex test suite that still covers the behaviours allowed by the specification in some defined sense. Such a test suite will usually contain intended, erroneous and unintended behaviours at the same time, because the model and the test case generator are unaware of these notions.

In fact, previous experience has confirmed the value of such generated test suites for the V&V of specifications [4]: “*The test cases generated […] provided valuable feedback. […] this involved non-anticipated scenarios, i.e. scenarios showing repetitions […], scenarios in which no system reaction should occur or scenarios uncovering the incompletely specified combinatorics of functionalities were able to be detected.*” While the focus of [4] was to investigate the feasibility of TCG for signalling systems in general, the case study reported in this article follows up on the cited observations and actually applies model-based test case generation and inspection for the purpose of the V&V of a real system specification. In this case, the validation refers to (the absence of) unintended behaviour and the verification of any errors introduced during the specification process.

1.3 The article’s structure

Chapter 2 provides the context for the case study by introducing the EULYNX initiative and model-based testing, as well as the way that both have come together in the case study. Chapter 3 presents an overview and details of the different phases of the V&V process applied in the case study. Chapter 4 describes the results of the V&V activity. Finally, the case study is discussed and an outlook is given in Chapter 5.

2 The context for the case study

2.1 EULYNX

EULYNX is a European initiative of currently 13 railway infrastructure managers (IM) that aims to define and standardise interlocking interfaces. Several interface specifications have been published along with the related documentation on the EULYNX website [2], so that they can be used in tenders. An important goal is to achieve reduced lifecycle costs for future digital signalling systems, while fostering innovation through exchangeable subsystems.

The specifications include a detailed description of the system behaviour that is visible at the interfaces. This entails the need to argue about the correctness and security levels and to perform a safety evaluation of the specified behaviour.

ebenso mit sich wie die einer Sicherheitsbewertung des spezifizierten Verhaltens. EULYNX verwendet einen strukturierten modellbasierten Systems-Engineering (MBSE)- Ansatz, der sich auf die Systems Modeling Language (SysML) stützt, um die Erstellung von korrekten, vollständigen und konsistenten Spezifikationen zu fördern und Automatisierung zu ermöglichen. Der EULYNX-Modellierungsstandard [5] beschreibt den Spezifikationsprozess, der zunächst für beide Seiten der Schnittstelle eine Struktur aus logischen Komponenten erstellt (unter Nutzung von SysML-Blockdefinitionsdiagrammen und internen Blockdiagrammen). Das Verhalten dieser Komponenten wird dann schrittweise von Anwendungsfällen (Anwendungsfalldiagramme) über Szenarien (Sequenzdiagramme) bis hin zu detailliertem ausführbarem Verhalten (Zustandsdiagramme) detailliert.

2.2 Testfallgenerierung für V&V von Spezifikationen

Modellbasiertes Testen (MBT) bezeichnet die Nutzung von Modellen als Teil von Testprozessen mit dem Ziel, die Generierung von Testartefakten zu automatisieren. Insbesondere werden TFG-Werkzeuge auf Modelle angewandt, um eine (teilweise) Automatisierung des Testdesigns zu erreichen. Neben möglichen Einsparungen bei Zeit und Kosten durch die Automatisierung liegen Vorteile in einer erhöhten Wiederverwendbarkeit und Wartbarkeit von Testfällen aufgrund des modellzentrierten Ansatzes, einer besseren Testfallqualität durch die Präzision und automatisierte Überprüfbarkeit von Modellen sowie einer nachvollziehbareren Testauswahl, da modellbezogene Abdeckungskriterien von den Generierungswerkzeugen verwendet werden; siehe auch [6].

Verschiedene MBT-Ansätze unterscheiden sich signifikant in den verwendeten Modellen und Werkzeugen; in [7] findet sich eine Taxonomie. Die in diesem Beitrag vorgestellte Fallstudie verwendet einen systemmodellbasierten Ansatz, d. h. das Verhalten des betrachteten Systems (und seiner Umgebung) wird modelliert; in diesem Fall durch untereinander verbundene Zustandsautomaten und unter Nutzung von SysML. Aus dem Modell werden eine Testarchitektur und schließlich Testfälle mithilfe formaler Techniken wie symbolischer Ausführung und Constraint Solving generiert, die die Zustände und Transitionen des Modells abdecken.

Während MBT in den frühen 2000er Jahren hauptsächlich Forschungsgegenstand war, wurde es im vergangenen Jahrzehnt mehr and mehr durch die Industrie übernommen. Meistens wird es für funktionales Testen im Rahmen von Integrations-, System- oder Akzeptanztests angewandt [8]. Dies schließt normalerweise die Ausführung von generierten Testfällen auf einem Produkt ein. Bezüglich des letztgenannten Aspekts unterscheidet sich der MBT-Anwendungsfall dieses Beitrags vom Standard: Die generierten Testfälle werden lediglich für eine manuelle Inspektion verwendet, ohne sie auszuführen. Entsprechend besteht das Ziel nicht in V&V eines Produktes, sondern einer Spezifikation, indem eine Menge von Verhaltensweisen (durch die Testfälle gegeben) einem Review unterzogen wird, die mit der Spezifikation übereinstimmt und sie in definierter Weise abdeckt.

2.3 V&V der EULYNX-Weichenspezifikation

Die Anwendung von MBT auf EULYNX-Spezifikationen erscheint aus mehreren Gründen besonders vorteilhaft. Erstens können die SysML-Zustandsautomaten aus den Spezifikationen leicht als Grundlage eines Verhaltensmodells verwendet werden, aus dem Testfälle generiert werden können. Tatsächlich basieren viele TFG-Werkzeuge auf Zustandsautomaten und auf UML / SysML (UML = Unified Modeling Language), sodass – zusätzlich zur automatisierten Generierung von Testfällen aus dem Modell – auch die Modellerstellung sehr effizient wird. Zweitens wird die Abdeckung der Anforderungen besonders einfach, da in den EULYNX-Spezifikationen die Transitionen der Zustandsauto-

EULYNX uses a structured model-based systems engineering (MBSE) approach employing the Systems Modelling Language (SysML) to support the creation of correct, complete and consistent specifications and to facilitate automation. The EULYNX Modelling Standard [5] describes the specification process that establishes a structure based on logical components (using SysML block definition diagrams and internal block diagrams) on both sides of an interface. The behaviour of such components is then detailed stepwise from use cases (use case diagrams) via scenarios (sequence diagrams) to detailed executable behaviour (state machine diagrams).

2.2 Test case generation for the V&V of specifications

Model-based testing (MBT) refers to the use of models as part of testing processes in order to automate the generation of test artefacts. TCG tools are particularly applied to models in order to achieve the (partial) automation of the test design. In addition to the possible time and cost savings achieved through automation, the advantages also include the improved reuse and maintainability of the test cases due to the model-centric approach, better test case quality through precision and automated model checks and more comprehensible test selection, as model-related coverage criteria are used by the generation tools; also see [6].

Significant differences exist among the MBT approaches regarding the models and tools used; [7] provides a taxonomy. The case study presented in this article follows a system model based approach, i.e. the behaviour of the system under consideration (and of its environment) has been modelled – in this particular case by interconnected state machines using SysML. A test architecture and then the test cases covering the model's states and transitions are generated from the model based on formal techniques such as symbolic execution and constraint solving.

While MBT was mainly a research topic in the early 2000s, it has been increasingly adopted by the industry over the last decade. It is most often applied to functional testing during integration, system or acceptance tests [8]. This usually includes the execution of the generated test cases on a product. With regard to this latter aspect, the MBT use case presented in this article differs from the standard: the generated test cases are simply used for manual inspections without any execution. Accordingly, the goal is not the V&V of a product, but of a specification by reviewing a set of behaviours (given by the test cases) that conform to the specification and also cover it in a defined sense.

2.3 V&V of the EULYNX point specification

There are several reasons why the application of MBT to EULYNX specifications appears to be particularly beneficial. Firstly, the SysML state machines provided in the specifications can easily form the basis of a behavioural model, from which test cases can be generated. In fact, many TCG tools are based on state machines and on the UML / SysML (UML = Unified Modelling Language), so that – in addition to the automated generation of test cases from the model – the model creation also becomes very efficient. Secondly, requirement coverage becomes particularly easy, as the state machine transitions are considered to constitute requirements in the EULYNX specifications. The TCG can produce test cases according to the transition coverage goal and provide information about the model elements covered for them. The tracing of re-

maten als Anforderungen verstanden werden. Die TFG kann Testfälle mit dem Ziel einer Transitionsüberdeckung erzeugen und mit ihnen Informationen zu abgedeckten Modellelementen bereitstellen. Zusammen mit der Ähnlichkeit der Zustandsautomaten aus Spezifikation und Testmodell wird eine Nachverfolgung der Anforderungen in beiden Richtungen zwischen Spezifikation und Testfällen leicht machbar. Schließlich enthalten die EULYNX-Spezifikationen keine expliziten erwünschten Systemeigenschaften, wie sie z. B. für formale Verifikation nötig sind. Testen benötigt dies nicht; es kann auf der Grundlage einer operativen Beschreibung des Systemverhaltens erfolgen. Dies kann von Vorteil sein, da IM traditionell in betrieblichen Abläufen denken, deren historische Hintergründe unbekannt oder geistiges Eigentum des IM sein können.

Die für die Fallstudie gewählte Spezifikation ist die EULYNX-Anforderungsspezifikation für das Teilsystem Weiche [9]. Sie gehört zu denjenigen EULYNX-Spezifikationen, die nicht nur die Schnittstelle zwischen Stellwerk und verbundenem Teilsystem spezifizieren, sondern auch die Spezifikation des Teilsystems selbst (hier: des Weichencontrollers) umfassen. Dies ist bedeutsam, da die Generierung von Testfällen für das reine Schnittstellenverhalten auf Anwendungsebene auf generisches Verbindungsmanagement und die Weiterleitung von Nachrichten begrenzt wäre. Ein Grund für die Wahl der Weichenspezifikation war, dass sie zu dieser Zeit vollständiger als andere EULYNX-Teilsysteme war. Die Weichenspezifikation definiert drei logische Komponenten für die spezifische Schnittstelle und das spezifische Teilsystem und stellt ihr Verhalten als Zustandsautomaten zur Verfügung; zusätzliche Zustandsautomaten für generische Schnittstellenkomponenten, z. B.

quirements back and forth between the specification and the test cases becomes simple due to the similarity of the state machines from the specification and the test model. Finally, the EULYNX specifications do not include any explicitly desired system properties as required for formal verification, for example. Testing does not require this, as it can be based on an operating description of the system behaviour. This can be an advantage, because IM traditionally think in operating procedures, all of whose historical rationales may not be known or may be the intellectual property of the IM.

The specification chosen for the present case study is the EULYNX requirement specification for the point subsystem [9]. It is one of those EULYNX specifications that do not only specify the interface between the interlocking and the adjacent subsystem, but also include the subsystem specification (here, the point controller). This is important because the generation of test cases for the pure interface behaviour at the application level would be limited to generic connection management and message relaying behaviour. One reason to choose the point specification was that it was considered to be more complete than the other EULYNX subsystems at that time. The point specification defines three logical components for the specific interface and subsystem and provides their state machine behaviour; additional state machines for generic interface components dealing with connection management, for example, are provided in a separate specification [10].

The tool used for TCG in the present case study is the Rhapsody Automatic Test Generation (ATG) add-on [11] to the

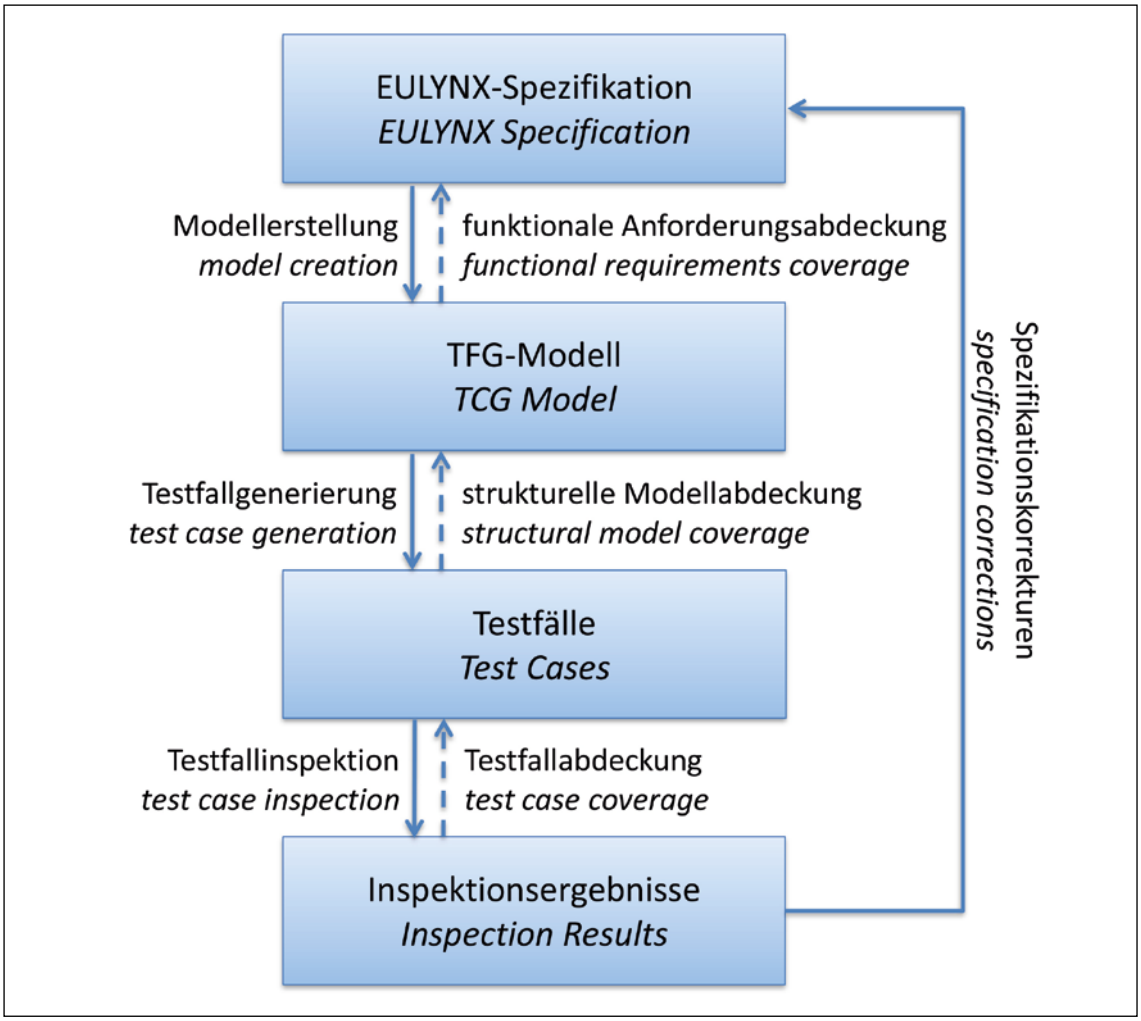

Bild 1: Überblick über den für die testfallinspektionsbasierte V&V der EULYNX-Weichenspezifikation angewandten Prozess
Fig. 1: An overview of the process applied to the test case based V&V of the EULYNX point specification

für das Verbindungsmanagement, finden sich in einer separaten Spezifikation [10].

Das für die Fallstudie genutzte TFG-Werkzeug ist das Automatic Test Generation (ATG) Add-on [11] des UML/SysML-Werkzeugs Rhapsody [12] von IBM. ATG wiederum baut auf dem TestConductor Add-on [11] auf. Gemeinsam bilden Rhapsody und die beiden Add-ons eine eng integrierte Umgebung, die den gesamten MBT-Prozess abdeckt. Insbesondere wurde die Erstellung des TFG-Modells und die Inspektion der generierten Testfälle in Rhapsody durchgeführt.

3 V&V-Prozess

3.1 Überblick

Die grundlegenden zur V&V der EULYNX Weichenspezifikation angewandten Prozessschritte sind in Bild 1 als durchgezogene Pfeile dargestellt. Nach dem Studium der relevanten EULYNX-Dokumente [9, 10] wurde aus diesen ein TFG-Modell erstellt (Details in Abschnitt 3.2 weiter unten). Als nächstes wurde die automatische TFG auf dem Modell durchgeführt, um so eine Menge an Testfällen zu erhalten (in Abschnitt 3.3 beschrieben). Die Testfälle wurden auf jegliches unerwartete Verhalten hin inspiziert (siehe Abschnitt 3.4). Schließlich gaben die Inspektionsergebnisse Anlass zu Spezifikationskorrekturen. Auch wenn die eigentliche Durchführung der Korrekturen nicht Teil der Fallstudie war (sie müssen durch das für die Spezifikation zuständige EULYNX-Cluster erfolgten), war es für diesen letzten Schritt wichtig, die Inspektionsergebnisse auf die Spezifikation zurückbeziehen zu können, was durch die gestrichelten Pfeile in Bild 1 dargestellt ist und in Abschnitt 3.5 weiter diskutiert wird. Es ist möglich, den gesamten Prozess zu iterieren, bis die Inspektion keine weiteren Probleme mehr aufdeckt, was jedoch nicht Teil der Fallstudie war.

3.2 Modellerstellung

Es wurde entschieden, als System Under Test (SUT) die drei weichenspezifischen logischen Komponenten (vgl. Abschnitt 2.3) zu verwenden, da sie ein interessanteres V&V-Zielobjekt darstellten als die generischen Komponenten, die bereits durch den Einsatz in den verschiedenen EULYNX-Schnittstellen gereift waren. Außerdem wurden Kon-

UML/SysML tool [12] by IBM. ATG in turn is based on the TestConductor add-on [11]. Together, Rhapsody and the two add-ons form a tightly integrated environment which covers the whole MBT process. In particular, the TCG model creation and the inspection of the generated test cases were performed in Rhapsody.

3 The V&V process

3.1 Overview

The basic process steps performed for the V&V of the EULYNX point specification are depicted in fig. 1 by the solid line arrows. After studying the relevant EULYNX documents [9, 10], a model for the TCG was created from them (details in Section 3.2 below). Next, the automatic TCG was applied to the model in order to obtain a number of test cases (described in Section 3.3). The test cases were then inspected for any kind of unexpected behaviour (see Section 3.4). Finally, the inspection results gave rise to specification corrections. Even though the implementation of the actual corrections was not part of the case study (they have to be performed by the EULYNX cluster in charge of the specification), it was important for this last step to be able to trace the inspection results back to the original specification, which is indicated by the dashed line arrow in fig. 1 and further discussed in Section 3.5. It is possible to iterate the whole process until the inspection reveals no further issues, but this was not part of the case study.

3.2 Model creation

A decision was made to use the three point-specific logical components (cf. Section 2.3) as the system under test (SUT), as this seemed a more interesting V&V target than the generic ones which had already matured through use in different EULYNX interfaces. Moreover, the configuration parameters, e.g. for the selection of any country-specific functions, were identified and two system configurations were chosen that allowed almost all the specified behaviour to be covered.

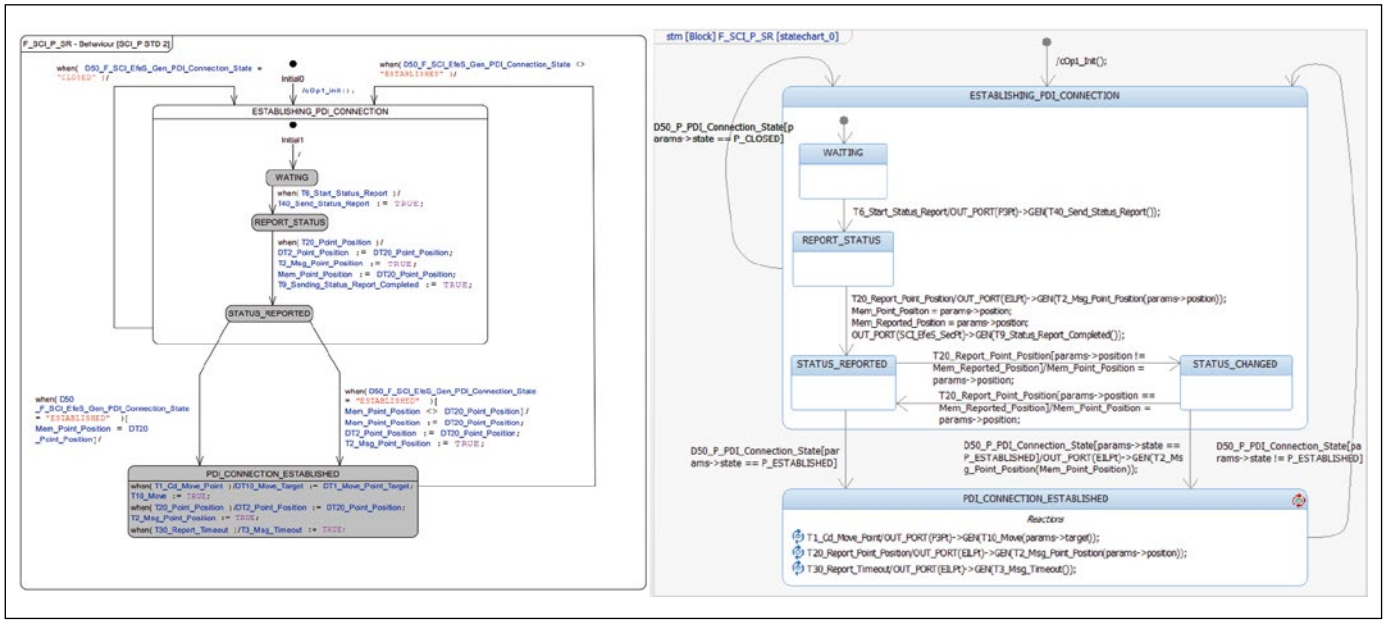

Bild 2: Ein ursprünglicher Zustandsautomat aus der Spezifikation [9] (links) und der übertragene und leicht angepasste Zustandsautomat aus dem TFG-Modell (rechts)

Fig. 2: An original state machine from the specification (left) and the redrawn and slightly adapted state machine from the TCG model (right)

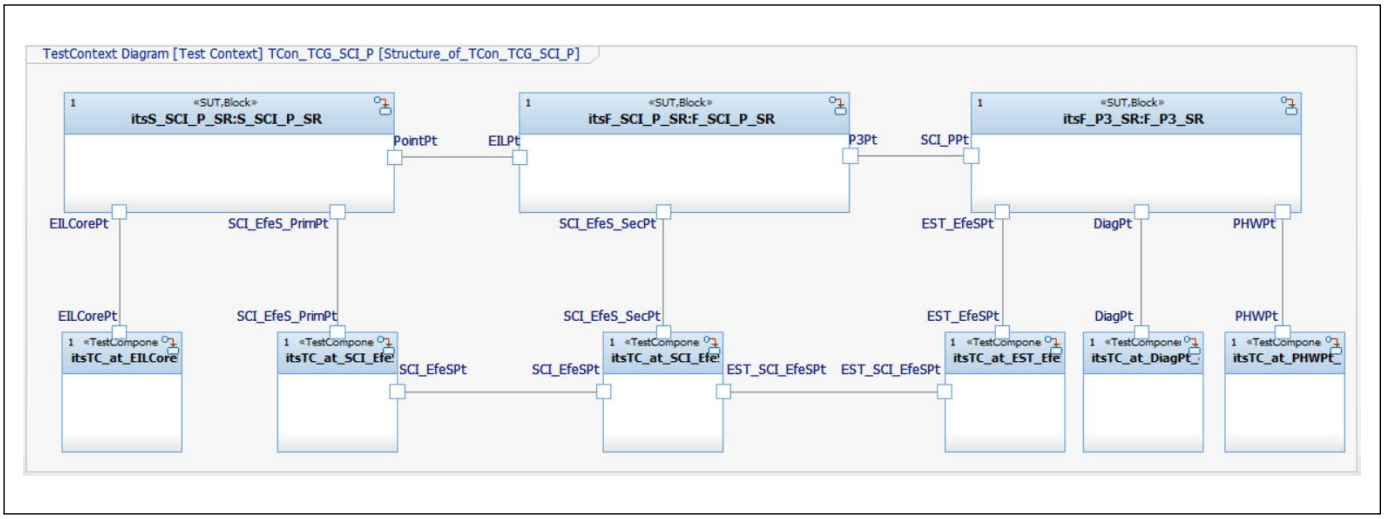

Bild 3: Testarchitektur bestehend aus drei SUT-Komponenteninstanzen (im oberen Teil), sechs Testkomponenteninstanzen (im unteren Teil) und ihren Verbindungen untereinander

Fig. 3: The test architecture, including three SUT component instances (in the upper part), six test component instances (in the lower part) and their interconnections

figurationsparameter, z. B. für die Auswahl länderspezifischer Funktionen, identifiziert und zwei Systemkonfigurationen gewählt, mit denen nahezu das gesamte spezifizierte Verhalten abgedeckt werden konnte. Da sowohl die Spezifikation als auch das TFG-Modell auf SysML basieren, war die Modellerstellung besonders einfach. In weiten Teilen lief dies auf ein Nachzeichnen der logischen Komponentenstruktur und des Zustandsautomaten-Verhaltens in Rhapsody hinaus (Bild 2). Einige Situationen wie die Verwendung nutzerdefinierter SysML-Ports in der Spezifikation, eine weniger allgemeine von Rhapsody unterstützte Transitionslabel-Syntax und ein inkorrektes Transitionslabel in der Spezifikation benötigten individuelle Modellierungslösungen – alle konnten jedoch unter Nutzung der in Rhapsody verfügbaren Modellelemente behandelt werden.

Neben dem SUT-Verhalten kann auch Umgebungsverhalten in SysML modelliert und durch den Testfallgenerator berücksichtigt werden. Die Testkomponentenstruktur, in der diese Modellierung erfolgt, kann weitgehend automatisch durch TestConductor generiert werden (Bild 3). Entsprechend dem gewählten SUT (SUT-Komponenteninstanzen im oberen Teil des Bildes) bestand die Umgebung (Komponenteninstanzen im unteren Teil des Bildes, von links nach rechts) aus einem Stellwerkskern, einem unterliegenden Kommunikationssystem (je eine Komponenteninstanz auf jeder Seite der Schnittstelle), einer Weichen-Startsteuerung, einem Weichen-Diagnosespeicher und den Weichenantrieben (unabhängig von ihrer Anzahl durch eine einzige Komponenteninstanz repräsentiert). Der größte Teil der Umgebungsmodellierung erfolgte auf Basis vereinfachter Versionen der generischen EULYNX-Zustandsautomaten der an die SUT-Komponenten angrenzenden Komponenten. Auf diese Weise wurden Umgebungseigenschaften wie der symmetrische Verbindungsabbau zwischen Stellwerk und Weichencontroller realisiert. Andere Eigenschaften wie die Erhaltung der Nachrichtenfolge der Kommunikation zwischen den Schnittstellenseiten benötigten keine explizite Modellierung (implizit durch das Rhapsody Event-Scheduling).

Die Konfigurationsdaten für die gewählten Konfigurationen wurden dem Model in Form einer initialen Konfigurationsprozedur, d. h. einer SysML-Operation, hinzugefügt.

Zur Validierung des Modells wurde Rhapsody für die Generierung von ausführbarem C++-Code hieraus und für die animierte interaktive Modellsimulation zweier grundlegender Ereignisabfolgen verwen-

The model creation was particularly easy, as both the specification and the TCG model are based on SysML. In large parts, this amounted to the redrawing of the logical component structure and state machine behaviour in Rhapsody (fig. 2). However, the use of customised SysML ports in the specification, a less general transition label syntax supported by Rhapsody and an ill-formed transition label in the specification required individual modelling solutions, but all of this was able to be handled using the model elements that are available in Rhapsody.

In addition to the SUT behaviour, the environmental behaviour can also be modelled in SysML and taken into account by the test generator. The test component structure in which this is realised can mostly be created automatically by TestConductor (fig. 3). The environment (the component instances in the lower part of the figure from left to right) consisted of an interlocking core, an underlying communication system (one component instance on each interface side), a point start-up control, a point diagnosis memory and the point machines (represented by a single component instance regardless of their number) in line with the scope of the chosen SUT (the SUT component instances in the upper part of the figure). Most of the environment modelling was performed on the basis of simplified versions of the generic EULYNX state machines for the components adjacent to the SUT components. Environmental properties such as the symmetric connection termination between the interlocking and the point controller were realised in this way. Other properties, such as the preservation of the message order in the communication between the interface sides, did not require any explicit modelling (implicit through the Rhapsody event scheduling).

The configuration data for the chosen configurations was added to the model in the form of an initial configuration procedure, i.e. a SysML operation.

Rhapsody was used to generate an executable C++ code from the model and to execute several basic event sequences using animated interactive model simulation in order to validate the model. They behaved as expected, thus confirming the general usefulness of the model.

det. Diese verhielt sich wie erwartet und bestätigte so die allgemeine Brauchbarkeit des Modells.

3.3 Testfallgenerierung

Die Generierung von Testfällen mit ATG erfolgt im Wesentlichen per Mausklick. In der Fallstudie wurde sie erst ausgeführt, nachdem das gesamte Modell fertiggestellt war und nach kleineren Modellkorrekturen wiederholt. ATG benötigt als Eingabe ein ausführbares TFG-Modell. Für die Versuche wurde strukturelle Modellabdeckung (Zustände, Transitionen und Operationen) als Abdeckungsziel gewählt. Die Testgenerierungsprozedur versucht eine 100 %-Abdeckung zu erreichen, aus verschiedenen Gründen ist jedoch eine Terminierung mit einer Menge von Testfällen mit geringerer Abdeckung möglich. Über Schnittstelleneinstellungen lässt sich bestimmen, welche SUT-Eingaben generiert werden können und welche Nachrichten zwischen den Komponenteninstanzen der Testarchitektur in den Testfällen aufgezeichnet werden. Für weitere Details siehe [11].

Als zu generierende Eingaben wurden die Kommandos des Stellwerkskerns zum Stellen der Weiche sowie die Rückmeldungen des Weichenantriebs zu Position und Aktivierungsstatus ebenso erlaubt wie der Status der Kommunikationsverbindung und des Weichenstarts aus den generischen EULYNX-Komponenten. Als aufzuzeichnende Nachrichten wurden alle Nachrichten zu, von und zwischen den drei SUT-Komponenteninstanzen gewählt.

Mit diesen Einstellungen und der finalen Modellversion (inklusive Korrekturen nach der Inspektion von aus vorläufigen Versionen generierten Testfällen, vgl. Abschnitt 3.4) wurden 28 Testfälle für jede der beiden Konfigurationen erzeugt, die jeweils zwischen drei und mehr als 40 aufgezeichnete Nachrichten enthielten. Für die erste Konfiguration wurden 78 % Abdeckung in 23 Sekunden erreicht, für die zweite waren es 87 % in 36 Sekunden. Die erzeugten Testfälle wurden nach Rhapsody exportiert, wo sie wie in Bild 4 gezeigt als SysML-Sequenzdiagramme betrachtet werden können.

Nach der Generierung sollten nicht abgedeckte Modellelemente (der übrige Prozentsatz) überprüft werden. Für beide Konfigurationen waren einige der nicht abgedeckten Elemente unerreichbar, da die zugehörige Funktionalität nicht Teil der entsprechenden Konfiguration war. Darüber hinaus wurden ein paar unerreichbare Elemente durch die im Umgebungsmodell verwendeten vereinfachten Zustandsautomaten aus dem generischen Teil von EULYNX verursacht. Unter den nicht abgedeckten Elementen schien kein erreichbares zu sein, was bedeutet, dass der TFG-Algorithmus Testfälle geliefert hat, die das TFG-Modell im maximal möglichen Umfang abdecken.

3.4 Testfallinspektion

Die generierten Testfälle wurden inspiziert, d. h. ihre Nachrichtenabfolgen wurden systematisch daraufhin untersucht, ob sie erwartetes / gewünschtes funktionales Verhalten zeigen. „Unerwartet / unerwünscht" bezieht sich dabei z. B. auf fehlende oder zusätzliche Nachrichten, inkorrekte Nachrichtenparameter, eine falsche Nachrichtenreihenfolge, falsche Nachrichtensender oder -empfänger oder eine inkonsistente oder ungültige Abfolge von Ereignissen. Als Inspektionsergebnis wurden über zehn Probleme identifiziert. Die Hälfte von ihnen stellte sich als Modellierungsfehler heraus; wann immer solche Fehler gefunden wurden, wurde das TFG-Modell unverzüglich korrigiert, wurden die Testfälle neu generiert und wurde die Inspektion neu begonnen. Die verbleibenden Fehler wurden als mögliche Spezifikationsprobleme (Details in Abschnitt 4 weiter unten) identifiziert; sie wurden in einer Liste gesammelt, die eine textuelle Beschreibung sowie Verweise auf Testfall und Spezifikation beinhaltet. Schließlich wurde die Liste um weitere im Verlauf der Fallstudie gefundene Spezifikationsprobleme (vgl. Abschnitt 4 weiter unten) ergänzt und das für die Weichenspe-

3.3 Test case generation

Test case generation using ATG is essentially a push-button activity. It was only applied in the case study after the whole model had been completed and it was iterated after some smaller model corrections had been made. ATG requires an executable TCG model as its input. Structural model coverage (states, transitions and operations) was set as the coverage goal for the experiments. The test generation procedure tries to achieve 100 % coverage, but may terminate with a set of test cases achieving a lower coverage for various reasons. Interface settings determine which SUT inputs may be generated and which messages between the test architecture's component instances will be recorded in the test cases. For more details, see [11].

Interlocking core commands to move the points and point machine feedback on the position and activation status were used as the inputs to be generated, but communication connection statuses and start-up statuses from generic EULYNX components were also allowed. All the messages to, from and between the three SUT component instances were chosen as the messages to be recorded.

The use of those settings and the final model version (including the corrections after inspecting the test cases generated from the preliminary versions, cf. Section 3.4) resulted in 28 test cases for each of the two configurations with the number of recorded messages between three and more than 40 per test case. 78 % coverage was reached in 23 seconds for the first configuration and 87 % was reached in 36 seconds for the second configuration. The resulting test cases were exported to Rhapsody, where they can be displayed as SysML sequence diagrams as shown in fig. 4.

The non-covered model elements (the remaining percentage) needed to be checked after generation. Some non-covered elements were unreachable for each configuration, because the associated functionality was not part of the given configuration. Furthermore, the simplified state machines of the generic EULYNX parts used for the environment model caused several elements to be unreachable. There seemed to be no reachable elements among the non-covered ones, which means that the TCG algorithm delivered test cases that cover the TCG model to the maximum possible extent.

3.4 Test case inspection

The generated test cases were inspected, i.e. their message sequences were systematically reviewed to see whether they exhibited the expected / desired functional behaviour. "Unexpected / undesired" refers, for example, to missing or extra messages, incorrect message parameters, the wrong message order, the wrong message source or target or a broader course of events that is inconsistent or invalid. More than ten issues were identified as inspection results. Half of them turned out to be modelling errors; once found, the TCG model was immediately corrected, the test cases were regenerated and the inspection was restarted. The others were identified as potential specification issues (see Section 4 below for more details); they were collated into a list, including a textual description and references to the test case and the specification. Finally, this list was completed with further specification issues found during the case study (cf. Section 4 below) and reported to the EULYNX cluster in charge of the point specification. Much less time (about a fifth) was spent on the inspection phase (including any model corrections) compared to the model creation.

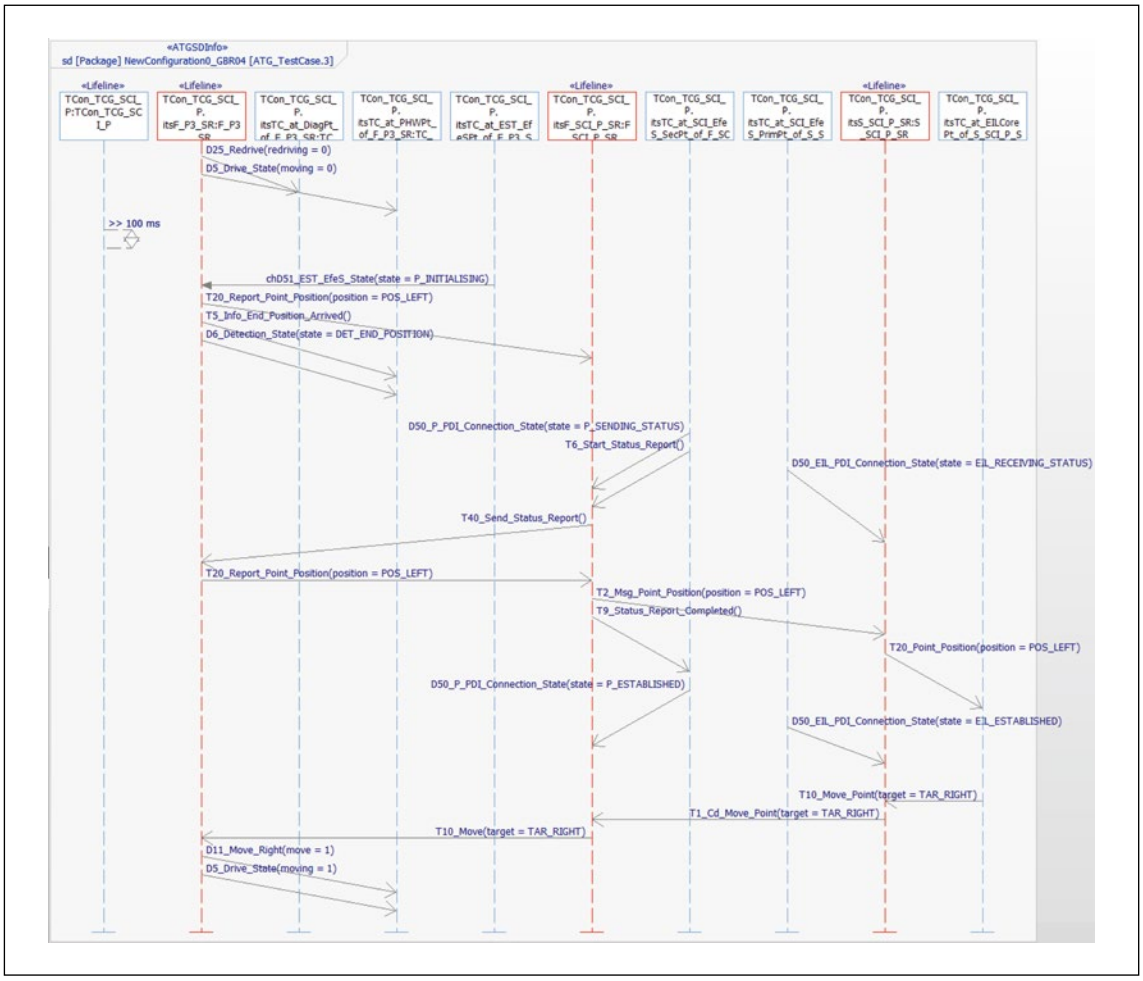

Bild 4: Aus dem TFG-Modell generierter Testfall als SysML-Sequenzdiagramm
Fig. 4: The test case generated from the TCG model as a SysML sequence diagram

zifikation zuständige EULYNX-Cluster informiert. Im Vergleich zur Modellerstellung war der zeitliche Aufwand für die Inspektionsphase (inkl. Modellkorrekturen) mit einem Fünftel deutlich geringer.

3.5 Nachverfolgbarkeit zwischen Spezifikation und Inspektionsergebnissen

Für jegliche V&V ist es wichtig, Nachverfolgbarkeit herzustellen; sowohl rückwärts für Debugging-Zwecke als auch vorwärts, um Aussagen zur Vollständigkeit der V&V treffen zu können.

Mit Blick auf die Beziehung zwischen Spezifikation und TFG-Modell stehen aufgrund des Modellerstellungsansatzes (vgl. Abschnitt 3.2) die SysML-Blöcke, Zustände und Transitionen, die in beiden verwendet wurden, größtenteils in einem Eins-zu-eins-Verhältnis, inklusive ihrer Benennung. Insbesondere stellen in EULYNX-Spezifikationen Transitionen (bzw. das durch sie modellierte Verhalten) verpflichtende Anforderungen dar, sodass letztere mit TFG-Modelltransitionen übereinstimmen. Die Zustandsautomaten der drei logischen SUT-Komponenten wurden vollständig im TFG-Modell modelliert, woraus die funktionale Anforderungsabdeckung durch das TFG-Modell folgt (vgl. den obersten gestrichelten Pfeil in Bild 1). Für Konstrukte, die nicht in einem Eins-zu-eins-Verhältnis stehen, wurde die von der Spezifikation zu TFG-Modell erfolgte Übersetzung textuell dokumentiert; unter Einbeziehung des Modellkontextes, in dem sie genutzt werden, ist es auch in jedem dieser Fälle möglich, einen eindeutigen Zusammenhang herzustellen.

Was die Beziehung zwischen TFG-Modell und den daraus generierten Testfällen betrifft, werden durch ATG die durch einen Testfall abgedeckten Modellelemente zu diesem Testfall hinzugefügt. Diese

3.5 Traceability between the specification and inspection results

It is important for any V&V activity to establish traceability, including backwards traceability for debugging purposes and forwards traceability in order to be able to make the point about the completeness of the V&V.

With a view to the relation between the specification and the TCG model based on the model creation approach (cf. Section 3.2), the SysML blocks, states and transitions used in both of them are mostly in a one-to-one correspondence, including their names. Transitions in EULYNX specifications (or the behaviours modelled via them) are particularly considered to constitute mandatory requirements, so that the latter correspond to the TCG model transitions. The state machines for the three logical SUT components were modelled completely in the TCG model, which established the functional requirement coverage by the TCG model (cf. the upper dashed arrow in fig. 1). The translation from the specification to the TCG model for constructs that are not in a one-to-one correspondence was documented textually; it is also always possible to establish unique correspondences in those cases along with the model context in which they are used.

With regard to the relations between the TCG model and the test cases generated from it, the model elements covered by some test cases are attached to that test case by ATG. Moreover, this assignment can be verified and broken down to a particular position in the test case's message sequence by simulating the test cases on the model. A covering test case was particularly gener-

Zuordnung kann außerdem verifiziert und auf eine bestimmte Position in der Nachrichtenabfolge des Testfalls heruntergebrochen werden, indem dieser auf dem Modell simuliert wird. Insbesondere wurde für alle Modelltransitionen außer denjenigen im Zusammenhang mit dem "Not-Initialized"-Status (da das Umgebungsmodell eine Rückkehr zum Not-Initialized-Status des Weichencontrollers nicht unterstützt) und denjenigen mit Bezug zu Weichenpositionsänderungen während des Verbindungsaufbaus (aufgrund des TFG-Algorithmus) ein Testfall generiert, der sie abdeckt. Dies bedeutet, dass eine nahezu vollständige strukturelle Modellabdeckung erreicht wurde (vgl. den mittleren gestrichelten Pfeil in Bild 1).
Zwischen Testfällen und Inspektionsergebnissen existiert logischerweise keine streng formale Beziehung: Die Inspektion erfolgt manuell, und es kann nicht alles unerwünschte Verhalten in den Testfällen entdeckt werden, da "unerwünschtes Verhalten" nicht formal definiert werden kann. Dennoch beziehen sich gefundene Probleme natürlicherweise auf den inspizierten Testfall und eine Position in seiner Nachrichtenabfolge, und es wurden die Nachrichtenabfolgen aller generierten Testfälle vollständig untersucht, was als eine gewisse Testfallabdeckung aufgefasst werden kann (vgl. den unteren gestrichelten Pfeil in Bild 1).
Fasst man die obigen Tatsachen zusammen und nimmt an, dass kein unerwünschtes Verhalten durch den Reviewer übersehen wurde, gilt für nahezu jede der etwa 50 transitionsbasierten Anforderungen, dass entweder ein Problem im Zusammenhang mit der Anforderung gefunden wurde, das zeigt, dass sie nicht wie gewünscht funktioniert, oder dass man weiß, dass sie durch mindestens ein beabsichtigtes (durch einen Testfall gegebenes) Systemverhalten erfüllt wird.

4 V&V-Ergebnisse

Während der Fokus der Fallstudie auf der Nutzung der Testfallinspektion zum Auffinden von Spezifikationsproblemen lag, wurden fast 20 weitere Probleme in früheren Prozessschritten gefunden; die meisten von ihnen bereits während der Analyse der EULYNX-Weichenspezifikation für die Erstellung des TFG-Modells. Einzelne Probleme wurden jeweils während des Modell-Reviews, der Kompilierung des Modellcodes sowie der TFG-Modellabdeckungsüberprüfung gefunden. Diese Probleme wurden dann meist im Modell behoben (durch das Treffen von Annahmen oder die Korrektur von offensichtlichen Spezifikationsfehlern), sodass sie die folgenden Schritte nicht mehr beeinflussen konnten.
Die gefundenen Probleme können in die folgenden Kategorien eingeteilt werden:

- Spezifikationslücken (fehlende Information). Beispiele sind die größtenteils unspezifizierte Kommunikation zwischen logischen Komponenten, fehlende Aussagen zur Ereignisverarbeitung und zu offenen Ports.
- Sprachliche und Begriffsprobleme. Beispiele sind undefinierte oder falsch benutzte Begriffe in textuellen Teilen der Spezifikation oder eine uneindeutige oder inkonsistente Benennung von Ports.
- Datentypfehler. Ein Beispiel ist die Nutzung von Daten in einem Kontext, der einen anderen Datentyp verlangt.
- Überzählige Elemente. Beispiele sind ungenutzte Ports und Variablen sowie redundante Zuordnungen von Variablenwerten.
- Möglicherweise problematisches Verhalten. Zum Beispiel, wenn die Ausführung einer Transition von der Interpretation von Change-Ereignissen abhängt (deren Effekte nicht vollständig durch SysML definiert sind).

Trotz der Behebung der meisten der soeben diskutierten Probleme im TFG-Modell wurden weitere Probleme (der Kategorie möglicherweise problematischen Verhaltens) während der Testfallinspektion ge-

ated for all the model transitions with the exception of those related to the "not initialised" status (due to the environment model which did not support returning to the point controller's non-initialised status) and one related to a point position change during the establishment of the connection (due to the TCG algorithm), meaning that near complete structural model coverage was achieved (cf. the middle dashed arrow in fig. 1).
Of course, there is no strong formal relation between the test cases and the inspection results. Inspection is a manual task and may not detect all the undesired behaviour shown by the test cases, as the "undesired behaviour" cannot be formally defined. However, the discovered issues are naturally connected to the inspected test case and a position in its message sequence and the complete message sequences of all the generated test cases were reviewed, amounting to a certain degree of test case coverage (cf. the lower dashed arrow in fig. 1).
If one summarises the above facts and assumes that no undesired behaviour has been missed by the reviewer, it holds for nearly all of the approximately 50 transition-based requirements that either an issue has been found in connection with the requirement that shows that it does not work as desired or it will be apparent that the requirement has been fulfilled by at least one intended system behaviour (given by a test case).

4 The V&V results

While the case study focussed on the use of test case inspection for finding specification issues, nearly 20 additional issues were also discovered in the earlier process steps. Most of them were found during the analysis of the EULYNX point specification for the TCG model creation. Single issues were found during the model review, the model code compilation and the TCG model coverage check. Such issues were then mostly dealt with in the model (by making assumptions and correcting any obvious specification errors) so that they could no longer affect the following steps.
The ascertained issues can be structured in the following categories:

- incompleteness of the specification (missing information). Examples include the mostly unspecified communication between the logical components and the missing specification of event handling and open ports.
- wording / naming issues. Examples include undefined or wrongly used terms in the textual parts of the specification or the equivocal or inconsistent naming of ports.
- typing errors. Just such an example involves the use of data within a context that requires a different data type.
- redundant elements. Examples include unused ports and variables as well as redundant variable assignments.
- potential behavioural issues. For example, whether or not a transition is taken depends on the interpretation of the change events (whose effects are not completely defined by SysML).

Despite resolving most of the aforementioned issues in the TCG model, still further issues (belonging to the category of potential behavioural issues) were found during the test case inspection as reported in Section 3.4. Examples include nondeterministic state machine behaviour, non-uniform behaviour under very similar conditions and conditions under which the initialisation gets stuck or performs infinite trials.
It should be noted that a few of the found issues may not exhibit specification omissions or errors in the strictest sense or may even be intentional from the point of view of the EULYNX

funden, wie in Abschnitt 3.4 berichtet. Beispiele sind nichtdeterministisches Verhalten von Zustandsautomaten, ungleiches Verhalten unter gleichartigen Bedingungen und Bedingungen, unter denen die Initialisierung stecken bleibt oder in eine Endlosschleife gerät.
Es sollte angemerkt werden, dass einige wenige der gefundenen Probleme keine Spezifikationslücken oder -fehler im engeren Sinne darstellen könnten oder aus Sicht des EULYNX-Clusters sogar beabsichtigt sein könnten, weil sie das Ergebnis des Verständnisses und der Beurteilung eines Reviewers sind. Allerdings wurden viele der an das EULYNX-Weichencluster berichteten Probleme in einer neuen Entwurfsversion [13] der Spezifikation behoben.

5 Diskussion und Ausblick

Die Fallstudie hat erfolgreich Generierung und Inspektion von Testfällen zur V&V einer EULYNX-Spezifikation angewandt und sowohl die Machbarkeit des Ansatzes als auch seinen Nutzen gezeigt:

- Mehrere Probleme des spezifizierten Verhaltens wurden während der Testfallinspektion aufgedeckt, obwohl die Spezifikation nicht besonders komplex ist (drei Zustandsautomaten mit rund 50 Transitionen).
- Viele weitere Spezifikationsprobleme verschiedener Kategorien wurden während des Studiums der Spezifikation für die Modellerstellung und während weiterer Schritte des V&V-Prozesses gefunden.
- Der Aufwand der Fallstudie (etwa 65 Arbeitsstunden, SysML-Kenntnisse und Erfahrung mit dem Werkzeug Rhapsody inkl. TFG waren bereits vorhanden) war dank der einfachen Erstellung eines TFG-Modells aus der bereits modellbasierten Spezifikation und der automatischen Testgenerierung recht gering.

In parallelen Fallstudien anderer Partner im selben Projekt wurde formale Verifikation (FV) auf eine Obermenge des in der hier vorgestellten Fallstudie untersuchten Verhaltens angewandt, siehe [14]. Verglichen mit der TFG-basierten V&V bieten sie sogar erschöpfende Verifikation von Eigenschaften und "automatisieren die Inspektion". Trotzdem kann TFG-basierte V&V als ein intuitiverer Ansatz (der keine Auseinandersetzung mit formalen Eigenschaften benötigt) attraktiv sein, der gleichwohl effektiv ist und sich auf einen strukturierten Prozess stützt. Die durch die beiden Ansätze entdeckten Probleme waren teils unterschiedlich und teils dieselben, und beide haben offensichtlich zwischenzeitlich zu zahlreichen Verbesserungen der Spezifikation geführt.
Aufgrund derselben Art der Spezifikation kann davon ausgegangen werden, dass der in der Fallstudie gewählte Ansatz genauso gut auf die anderen EULYNX-Teilsystemspezifikationen anwendbar ist. Derzeit wird untersucht, inwieweit dies auch auf den erweiterten Rahmen der durch die RCA-Initiative [3] behandelten LST-Systeme, Schnittstellen und Verhalten zutrifft. Ein anderer logischer nächster Schritt wäre, ein reifes TFG-Modell anzustreben und die generierten Testfälle als Standard-Testsuite zu verwenden, um durch Ausführung der Tests die Konformität von tatsächlichen Implementierungen zur EULYNX-Spezifikation zu verifizieren. ■

Danksagung

Die Arbeiten wurden im Rahmen des X2Rail-2-Projekts durchgeführt. Das Projekt wurde durch das Shift2Rail Joint Undertaking (JU) unter dem Horizon 2020 Forschungs- und Innovationsprogramm der europäischen Union unter der Zuwendungsvereinbarung Nr. 777465 gefördert.

cluster, because they are the result of the reviewer's understanding and judgement. However, many of the issues reported to the EULYNX point cluster have been resolved in the new draft version [13] of the specification.

5 Discussion and outlook

The case study has successfully applied the generation and inspection of test cases for the V&V of a EULYNX specification, proving the approach to be both feasible and worthwhile:

- behavioural issues were found during the test case inspection, although the specification is not overly complex (three state machines comprising about 50 transitions).
- many more specification issues were found in different categories during the specification review for the model creation and during further steps in the V&V process.
- the effort spent on the case study (about 65 hours of work; the knowledge of SysML and experience with the Rhapsody tool, including TCG, were pre-existing) was quite low due to the easy creation of a TCG model from the already model-based specification and the automated test generation.

Formal verification (FV) approaches have been applied to a superset of the behaviours investigated in the present case study in parallel case studies performed by other partners in the same project: see [14]. Compared to the TCG-based V&V, they have even provided exhaustive verification of the properties and "inspection automation". Still TCG-based V&V can be appealing as a more intuitive approach (it does not require any examination of formal properties) which is nevertheless effective and based on a structured process. Both approaches have discovered issues that were partly dissimilar and partly the same and both have obviously led to numerous improvements to the specification.
It can be expected that the approach taken in the case study will work well when applied to other EULYNX subsystem specifications, since they follow the same specification approach. An investigation is currently underway into the extent that this also holds true for the wider framework of the CCS systems, interfaces and behaviours covered by the RCA initiative [3]. Another natural next step would be to strive towards a mature TCG model and to use the generated test cases as a standard test suite to verify the conformity of actual implementations to the EULYNX specification by executing the tests. ■

Acknowledgement

This work has been conducted within the X2Rail-2 project. This project has received funding from the Shift2Rail Joint Undertaking (JU) under the European Union's Horizon 2020 research and innovation program under grant agreement No 777465.

LITERATUR | LITERATURE
[1] Boehm, B. W.: Software Engineering Economics. Prentice-Hall, 1981
[2] EULYNX Website, https://eulynx.eu, accessed on 15/02/2022 4:32 pm
[3] Reference CCS Architecture Workgroup on the ERTMS Users Group Website, https://ertms.be/workgroups/ccs_architecture, accessed on 02/03/2022 12:30 am
[4] Schwencke, D.: Model-based Test Case Generation on the Example of an RBC Function, SIGNAL+DRAHT 1/2020
[5] EULYNX: Modelling Standard. Eu.Doc.30, baseline 3.1, Dec 13, 2019
[6] ISTQB: ISTQB® Foundation Level Certified Model-Based Tester Syllabus, Version 2015
[7] Utting, M.; Pretschner, A.; Legeard, B.: A taxonomy of model-based testing approaches. Softw. Test. Verif. Reliab., vol. 22, 2012
[8] Kramer, A.; Legeard, B.: 2019 Model-based Testing User Survey: Results, 2020
[9] EULYNX: Requirements specification for subsystem Point. Eu.Doc.36, baseline 3.1, Jun 19, 2020
[10] EULYNX: Generic interface and subsystem requirements. Eu.Doc.20, baseline 3.1, Nov 27, 2019
[11] TestConductor and ATG Web documentation on the IBM website, https://www.ibm.com/docs/en/rhapsody/8.4.0?topic=dm-model-based-testing-testconductor-automatic-test-generation-atg, accessed on 02/03/2022 12:10 am
[12] IBM Engineering Systems Design Rhapsody Website, https://www.ibm.com/de-de/products/uml-tools, accessed on 02/03/2022 12:07 am
[13] EULYNX: Requirements specification for subsystem Point. Eu.Doc.36, baseline 3.4 draft, Feb 9, 2022
[14] Borälv, A. et al.: Holistic Study of Formal Methods and Standardization in Specification, Development, Verification and Validation of Railway Signalling System Software. Accepted for WCRR 2022

AUTOR | AUTHOR

Dr. Daniel Schwencke
Wissenschaftlicher Mitarbeiter Verifikations- und Validierungsmethoden / *Scientific staff Verification and Validation Methods*
Deutsches Zentrum für Luft- und Raumfahrt e.V. / *German Aerospace Center*
Institut für Verkehrssystemtechnik / *Institute of Transportation Systems*
Anschrift / *Address*: Lilienthalplatz 7, D-38108 Braunschweig
E-Mail: daniel.schwencke@dlr.de