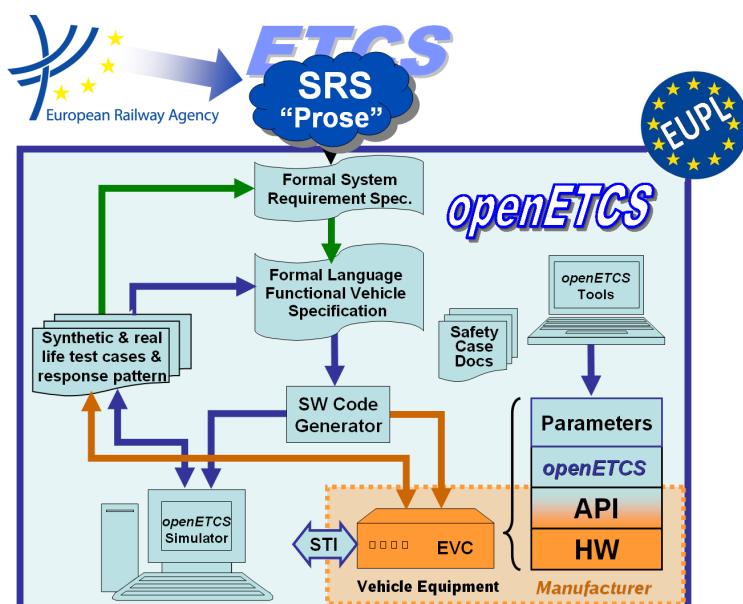


openETCS: Entwicklung und Implementierung des „Open Proof“-Konzepts für das europäische Signal- und Zugsicherungssystem ETCS

Schlussbericht des openETCS ITEA2 Projekts

Dr. Peter Mahlmann, Dr. Klaus-Rüdiger Hase, Bernd Hekele,
Abdelnasir Mohammed, Marc Behrens, Dr. Hardi Hungar,
Dr. Michael Jastram, Stefan Karg, Uwe Steinke, Jan Welte,
Dr. Christian König, Dr. Christian Stahl und Dr. Frank Golatowski,
Thorsten Schulz

März 2016



Funded by:



Federal Ministry
of Education
and Research



MINISTÈRE
DE L'ENSEIGNEMENT
SUPÉRIEUR
ET DE LA RECHERCHE



INVESTISSEMENTS
D'AVENIR



Région de
Bruxelles-
Capitale



GOBIERNO
DE ESPAÑA



MINISTERIO
DE INDUSTRIA,
ENERGÍA
Y TURISMO

Das diesem Bericht zugrunde liegende Vorhaben wurde mit Mitteln des Bundesministeriums für Bildung und Forschung unter den Förderkennzeichen 01IS12021A-L gefördert. Die Verantwortung für den Inhalt dieser Veröffentlichung liegt bei den Autoren.

openETCS Schlussbericht
März 2016

openETCS: Entwicklung und Implementierung des „Open Proof“-Konzepts für das europäische Signal- und Zugsicherungssystem ETCS

Schlussbericht des openETCS ITEA2 Projekts

Dr. Peter Mahlmann, Dr. Klaus-Rüdiger Hase, Bernd Hekele

Deutsche Bahn / DB Netz AG
Völckerstr. 5
80939 München

Abdelnasir Mohammed

AEBt Angewandte Eisenbahntechnik GmbH
Adam-Klein-Str. 26
90429 Nürnberg

Marc Behrens, Dr. Hardi Hungar

Deutsches Zentrum für Luft- und Raumfahrt e.V.
Lilienthalplatz 7
38108 Braunschweig

Dr. Michael Jastram

Formal Mind GmbH
Bagelstr. 114
40479 Düsseldorf

Stefan Karg

LEA Railergy
Am Mittleren Moos 48
86167 Augsburg

Uwe Steinke

Siemens AG
Ackerstr. 22
38126 Braunschweig

Jan Welte

Technische Universität Braunschweig
Institut für Verkehrssicherheit und Automatisierungstechnik
Herrmann-Blenk-Str. 42
38108 Braunschweig

Dr. Christian König, Dr. Christian Stahl

TWT GmbH
Ernstthalstr. 17
70565 Stuttgart

Dr. Frank Golatowski, Thorsten Schulz

Universität Rostock
Institut für angewandte Mikroelektronik und Datentechnik
Richard-Wagner-Str. 31
18199 Rostock-Warnemünde

Disclaimer: This work is licensed under the "openETCS Open License Terms" (oOLT) dual Licensing: European Union Public Licence (EUPL v.1.1+) AND Creative Commons Attribution-ShareAlike 3.0 – (cc by-sa 3.0)

THE WORK IS PROVIDED UNDER openETCS OPEN LICENSE TERMS (oOLT) WHICH IS A DUAL LICENSE AGREEMENT INCLUDING THE TERMS OF THE EUROPEAN UNION PUBLIC LICENSE (VERSION 1.1 OR ANY LATER VERSION) AND THE TERMS OF THE CREATIVE COMMONS PUBLIC LICENSE ("CCPL"). THE WORK IS PROTECTED BY COPYRIGHT AND/OR OTHER APPLICABLE LAW. ANY USE OF THE WORK OTHER THAN AS AUTHORIZED UNDER THIS OLT LICENSE OR COPYRIGHT LAW IS PROHIBITED.

BY EXERCISING ANY RIGHTS TO THE WORK PROVIDED HERE, YOU ACCEPT AND AGREE TO BE BOUND BY THE TERMS OF THIS LICENSE. TO THE EXTENT THIS LICENSE MAY BE CONSIDERED TO BE A CONTRACT, THE LICENSOR GRANTS YOU THE RIGHTS CONTAINED HERE IN CONSIDERATION OF YOUR ACCEPTANCE OF SUCH TERMS AND CONDITIONS.

<http://creativecommons.org/licenses/by-sa/3.0/>
<http://joinup.ec.europa.eu/software/page/eupl/licence-eupl>

Inhaltsverzeichnis

Abbildungsverzeichnis	vii
Tabellenverzeichnis.....	ix
I Kurzdarstellung	1
1 Aufgabenstellung	3
1.1 Ziele und angestrebte Ergebnisse	4
2 Voraussetzungen, unter denen das Vorhaben geführt wurde	7
3 Planung und Ablauf des Vorhabens	9
4 Wissenschaftlicher und technischer Stand	11
4.1 Domänen spezifische Sprachen für sicherheitskritische Softwareentwicklung	11
4.2 Open Source: Vorgehensmodelle, Geschäftsmodelle und Lizenzen.....	12
5 Zusammenarbeit mit anderen Stellen	13
5.1 Zusammenarbeit im internationalen ITEA Konsortium	13
5.1.1 Französische Projektpartner	13
5.1.2 Spanische Projektpartner	14
5.1.3 Belgische Projektpartner	15
5.1.4 Niederländische Projektpartner	15
5.1.5 Italienische Projektpartner	15
5.1.6 Britische Projektpartner.....	16
5.2 Zusammenarbeit mit externen Partnern	16
5.2.1 LEA Railergy	16
5.2.2 Bachleitner & Heugel Elektronik	16
5.2.3 MEN Mikro Elektronik	17
5.3 Zusammenarbeit mit anderen Projekten und Institutionen	17
5.3.1 Eclipse Foundation	17
5.3.2 PolarSys	17
5.3.3 VeTeSS	18
II Eingehende Darstellung	19
6 Verwendung der Zuwendung und des erzielten Ergebnisses im Einzelnen.....	21
6.1 Arbeitspaket 1 – Projektmanagement	21
6.2 Arbeitspaket 2 – Anforderungen an Open Proof	22
6.2.1 Stand der Wissenschaft und Technik.....	22
6.2.2 Definition der Methodik	24
6.2.3 Anforderungen	27
6.3 Arbeitspaket 7 – openETCS Tool Chain.....	28
6.3.1 openETCS Werkzeugplattform	29
6.3.2 Prototypen	34
6.3.3 Studien und Konzepte.....	35
6.3.4 Dienstleistungen.....	38

6.3.5 ERTMS Formal Specs	40
6.4 Arbeitspaket 3 – Modellierung – Codegenerierung	40
6.4.1 openETCS Traceability von Anforderungen	40
6.4.2 openETCS OBU Anforderungen	42
6.4.3 openETCS OBU Architektur	45
6.4.4 Generische openETCS API	48
6.4.5 Komponenten des ETCS Kernel Moduls F2	51
6.4.6 Modul F1 Receive Information from Trackside	67
6.4.7 Modul F6 DMI Controller	67
6.5 Arbeitspaket 4 – Validierungs- und Verifikationsstrategie	70
6.5.1 Überblick über Verifikation und Validierung	70
6.5.2 Verifikation und Validierung in der Planungsphase	71
6.5.3 Verifikation und Validierung in der Systementwicklungsphase	71
6.5.4 Verifikation und Validierung in der Softwareentwurfsphase	73
6.5.5 Verifikation und Validierung in der Software-Komponentenentwurfsphase	74
6.5.6 Verifikation und Validierung in der Software-Integrationsphase	75
6.5.7 Verifikation und Validierung in der Software-Validierungsphase	76
6.5.8 Ergebnisse der Verifikations- und Validierungsaktivitäten	76
6.5.9 Begutachtung der Ergebnisse	78
6.6 Arbeitspaket 5 – openETCS Demonstrator	79
6.6.1 nanoETCS Demonstrator	79
6.6.2 openETCS ERSA Demonstrator	82
6.6.3 openETCS GE Demonstrator	84
6.6.4 LEA/B&H Demonstrator	91
6.7 Arbeitspaket 6 – Veröffentlichung, Verwertung und Standardisierung	93
6.7.1 Veröffentlichungen	93
6.7.2 Markteinschätzung und Verwertung	95
6.7.3 Standardisierung	95
7 Wichtigste Positionen des zahlenmäßigen Nachweises	97
8 Notwendigkeit und Angemessenheit der geleisteten Arbeit	99
9 Voraussichtlicher Nutzen und Verwertbarkeit der Ergebnisse	101
9.1 Fahrzeughersteller	101
9.1.1 SWOT-Analyse	101
9.1.2 Quantifizierte Auswirkungen auf das Kundengeschäft	102
9.2 Eisenbahnunternehmen	104
9.2.1 SWOT Analyse	104
9.2.2 Quantifizierte Auswirkungen auf das Kundengeschäft	105
9.3 Dienstleister, Tool-Hersteller und industrielle Forschung	106
9.3.1 SWOT Analyse	107
9.3.2 Quantifizierte Auswirkungen auf das Kundengeschäft	109
9.4 Akademische Forschung	109
9.4.1 SWOT Analyse	110
9.4.2 Quantifizierte Auswirkungen	110
10 Fortschritte auf dem Gebiet des Vorhabens bei anderen Stellen	113
11 Erfolgte Veröffentlichungen der Ergebnisse	115
11.1 Dissertationen	115
11.2 Diplom- und Masterarbeiten	115
11.3 Schriftliche Veröffentlichungen	115

11.4 Präsentationen und Vorträge	117
Literaturverzeichnis	121

Abbildungsverzeichnis

Abbildung 1. Migration von mehr als 20 nationalen Zugsicherungssystemen zu ERTMS/ETCS	3
Abbildung 2. Struktur eines generischen Entwicklungsablaufs am Beispiel der EVC Entwicklung.	8
Abbildung 3. Struktur der Arbeitspakete.	9
Abbildung 4. Screenshot des openETCS Produkt-Backlogs.	22
Abbildung 5. openETCS System- und Softwareentwicklungphase.	25
Abbildung 6. openETCS Entwicklungs-Lebenszyklus.	26
Abbildung 7. openETCS mit dem Papyrus-Modell der openETCS OBU.	30
Abbildung 8. openETCS mit Subset-026, im nach ReqIF konvertierten Format.	31
Abbildung 9. Ein Fehlerbaum im Eclipse Safety Framework (ESF), welches in openETCS integriert wurde...	32
Abbildung 10. Das openETCS-Werkzeug mit zwei Papyrus-Diagrammen (links) und RT-Tester-Ergebnissen (rechts).	32
Abbildung 11. Die Traceability des Modells in ReqCycle.	33
Abbildung 12. Das openETCS-Modell in SCADE Suite.	34
Abbildung 13. Alternativen für die openETCS Werkzeugkette (aus [20]).	36
Abbildung 14. Risikominimierung durch die Verfolgung von drei verschiedenen Werkzeugansätzen (aus [20]).	37
Abbildung 15. Das Eclipse-Hilfesystem mit einer Seite der openETCS-Dokumentation.	37
Abbildung 16. Traceability im openETCS Architekturdesign.	43
Abbildung 17. Traceability im funktionalen Modell der openETCS OBU.	43
Abbildung 18. Umfang des openETCS OBU Modells gemäß ERA Subset-026, Kapitel 2.5.	46
Abbildung 19. Top-Level Architektur mit externen Schnittstellen E1 bis E10.	47
Abbildung 20. Systemarchitektur der openETCS OBU mit internen Schnittstellen I1 bis I10.	48
Abbildung 21. openETCS Referenz Hardware Architektur.	49
Abbildung 22. openETCS Referenz Software Architektur.	50
Abbildung 23. F2.1 Manage_TrackSideInformation_Integration SysML Diagramm.	52
Abbildung 24. F2.2 Manage_ETCS_Procedures SysML Diagramm.	54
Abbildung 25. F2.3 trainData SysML Diagramm.	54
Abbildung 26. F2.4 TrackAtlas SysML Diagramm.	55
Abbildung 27. F2.5 ManageLevelAndMode SysML Diagramm.	56
Abbildung 28. F2.6 calculateTrainPosition SysML Diagramm.	57
Abbildung 29. F2.7 SpeedSupervision_Integration SysML Diagramm.	58
Abbildung 30. SysML Diagramm der F2.7 SpeedSupervision_Integration Komponente.	59
Abbildung 31. Bremskurven zu den jeweiligen Bremspunkten.	60
Abbildung 32. Ableitung der W, I und FLOI Kurven in LimitLocations, für Ausgangsbeschaltung durch die Sub-Komponente Commands.	60
Abbildung 33. Nicht-lineares, abschnittsweises Modell einer Bremskurve und deren Einfluss auf den frühesten Brems-Eingriffspunkt.	61
Abbildung 34. F2.8 Provide_Position_Report SysML Diagramm.	62
Abbildung 35. F2.9 MoRC_HO SysML Diagramm.	63
Abbildung 36. F2.10 ManageDMIInput SysML Diagramm.	64
Abbildung 37. F2.11 ManageDMIOOutput SysML Diagramm.	65
Abbildung 38. F2.12 ManageTIUInput SysML Diagramm.	66
Abbildung 39. F2.13 ManageTIUOutput SysML Diagramm.	67
Abbildung 40. Struktur des Moduls F1 Receive Information from Trackside.	68
Abbildung 41. Struktur des Moduls F6 DMI Controller.	69
Abbildung 42. Benutzeroberfläche des openETCS DMI mit Planning Area (rechts).	69
Abbildung 43. CPN Modell der openETCS Prozedur „Start of Mission“ in der On-Board Unit.	72
Abbildung 44. Safety Requirements.	73

Abbildung 45. Übersicht der Generierung und Absicherung des Quellcodes bei OpenETCS.	76
Abbildung 46. nanoETCS ist eine Proof-of-Concept-Implementierung eines Teils des openETCS-Modells.	79
Abbildung 47. Stufen der modellbasierten Entwicklung bis zur Integration.	80
Abbildung 48. Die Hardware des nanoETCS-Zuges im ICE-Speisewagen.	81
Abbildung 49. Momentaufnahme aus dem nanoETCS-Demonstrationsvideo im Labor der Universität Rostock.	82
Abbildung 50. System Sicht des openETCS ERSA Demonstrators.	83
Abbildung 51. Simulatoren in der agilen Softwareentwicklung.	84
Abbildung 52. GE-Demonstrator Hardware in einem 19-Zoll Einschub.	85
Abbildung 53. DLR Software Architektur der integrierten Gesamtsystems auf einem oder mehreren Standard-PCs.	87
Abbildung 54. DLR Softwarearchitektur der Implementierung auf der GE-Plattform.	88
Abbildung 55. Software Implementierung des DMI im DLR Labor.	89
Abbildung 56. LEA/B&H Demonstrator	91
Abbildung 57. LEA/B&H Demonstrator: Systemkomponenten.	92
Abbildung 58. Funktionalitäten des LEA/B&H Demonstrators.	92
Abbildung 59. Entwicklungsplanung des LEA/B&H Demonstrators.	93
Abbildung 60. Überblick der Marktrollen der Projektpartner.	101
Abbildung 61. Ergebnisverwertung durch die Dienstleister, Tool-Hersteller und industrielle Forschungspartner.	106
Abbildung 62. Arten der Ergebnisverwertung durch die Dienstleister, Tool-Hersteller und industrielle Forschungspartner.	107
Abbildung 63. Ergebnisverwertung durch die akademischen Partner.	110

Tabellenverzeichnis

Tabelle 1. Auszug aus der Tabelle B4 in D2.2.....	24
Tabelle 2. Anzahl von Veröffentlichungen und Vorträgen während die Projektaufzeit.	93
Tabelle 3. Liste relevanter Standardisierungsorganisationen sowie der darin vertretenen Projektpartner.	96
Tabelle 4. Verwertbare Projektergebnisse der Fahrzeugherrsteller.	102
Tabelle 5. SWOT Analyse der Fahrzeugherrsteller.....	103
Tabelle 6. SWOT Analyse der Eisenbahnunternehmen.	105
Tabelle 7. SWOT Analyse der Dienstleister, Tool-Hersteller und industriellen Forschungspartner.	108
Tabelle 8. SWOT Analyse der akademischen Partner.	111
Tabelle 9. Dissertationen.	115
Tabelle 10. Diplom- und Masterarbeiten.....	115
Tabelle 11. Schriftliche Veröffentlichungen.....	115
Tabelle 12. Präsentationen und Vorträge.....	117

Teil I

Kurzdarstellung

1 Aufgabenstellung

Die europäischen Eisenbahnen haben sich in den letzten 150 Jahren getrennt voneinander innerhalb ihrer jeweiligen Landesgrenzen entwickelt. Das Resultat ist eine Vielzahl von unterschiedlichen Signal- und Zugsicherungssystemen, die den internationalen Zugverkehr behindern und erschweren (vgl. Abb. 1). Die Europäische Union hat beschlossen, diesen Mangel an Interoperabilität im europäischen Eisenbahnsektor zu beheben: Das European Train Control System (ETCS) als Teil des European Rail Traffic Management System (ERTMS), welches erstmals von der UIC¹ Anfang der 1990er Jahre vorgestellt wurde, wird die nationalen Signal- und Zugsicherungssysteme in Europa ablösen.

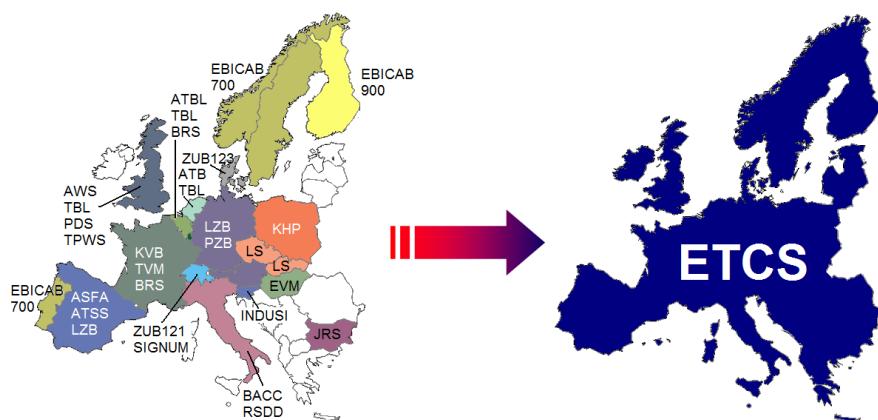


Abbildung 1. Migration von mehr als 20 nationalen Zugsicherungssystemen zu ERTMS/ETCS.

ETCS besteht aus Infrastrukturkomponenten sowie aus fahrzeugseitigen Ausrüstungskomponenten (On-Board Units, OBU). Die ETCS Anforderungsspezifikation (System Requirements Specification, SRS) wurde im Wesentlichen durch sechs namhafte Unternehmen aus dem Bereich der Signal- und Zugsicherungstechnik in enger Zusammenarbeit mit der Europäischen Kommission und den Eisenbahnunternehmen erstellt. Zur herstellerseitigen Koordination dieser Aktivitäten wurde die so genannte UNISIG (Union Industry of Signalling) gebildet², eine Arbeitsgruppe der UNIFE (Union des Industries Ferroviaires Européennes; Verband der europäischen Eisenbahnindustrie). Seitens der Bahnen hat sich die ERTMS Users Group (EUG) in der Rechtsform einer EEIG (European Economic Interest Group) etabliert. Daneben gibt es die EIM (European Infrastructure Manager) und die CER (Community of European Railways), welche sich in den Gestaltungsprozess einbringen. Beginnend mit der dritten Baseline der SRS wurde die Weiterentwicklung der Spezifikation durch die European Railway Agency (ERA) übernommen. Die Spezifikation ist öffentlich verfügbar und zudem seit dem Jahr 2015 verbindlich für neue Zugsicherungstechnik auf Infrastruktur- sowie Fahrzeugseite.

Man verspricht sich von ETCS die Überwindung nationaler Schranken, mehr Wettbewerb und insgesamt geringere Kosten für das System Bahn. Mit ETCS wird aber nicht nur das Ziel technischer Harmonisierung verfolgt, sondern es sollen dank moderner Informationstechnologien Steuerfunktionen und damit auch Sicherheitsverantwortung von der Strecke in das Fahrzeug ver-

¹Internationaler Eisenbahnverband (französisch *Union internationale de chemins de fer*, UIC).

²Gründungsmitglieder der UNISIG sind die Unternehmen Alstom, Ansaldo STS, Bombardier, Invensys, Siemens und Thales.

lagert werden, um dadurch die Streckenausrüstung zu vereinfachen und diese somit zuverlässiger und kostengünstiger zu gestalten.

Dies hat jedoch dazu geführt, dass die fahrzeugseitige ETCS Ausrüstung eine wesentlich höhere Komplexität aufweist als im Fall von konventionellen Zugsicherungssystemen. Hieraus resultieren erhebliche Kostensteigerungen bei der Entwicklung, Zulassung und Produktpflege für Eisenbahnbetreiber. Ferner fallen hohe Kosten für individuelle Anpassungen an, da es noch keinen umfassenden Standard für die relevanten Schnittstellen zwischen ETCS Bordgerät und Fahrzeug gibt. Zudem ist festzustellen, dass die funktionale Komplexität eine regelmäßige Softwarepflege erfordern wird.

1.1 Ziele und angestrebte Ergebnisse

Ziel des openETCS Projekts ist die Entwicklung einer integrierten Umgebung für die Modellierung, Entwicklung, Validierung sowie das Testen von Software unter Einsatz des neuesten Standes der Technik, um eine kosteneffiziente und zuverlässige Implementierung von ETCS zu ermöglichen. Diese Umgebung stellt somit eine ganzheitliche Werkzeugkette (Tool Chain) für den gesamten Entwicklungsprozess von ETCS On-Board Software dar. Die Werkzeugkette wird die formale Spezifikation und Verifikation eines ETCS Systems und seiner Anforderungen, die automatische und ETCS-spezifische Codeentwicklung und Validierung sowie die modellbasierte Entwicklung und Durchführung von Testszenarien ermöglichen. Im openETCS Projekt soll hierfür in sämtlichen Bereichen auf offene Standards gesetzt werden. Dies umfasst die Software- und Hardwarespezifikation, die Definition von Schnittstellen, die Entwicklung von Werkzeugen, Verifikations- und Validierungsprozesse sowie eingebettete Steuerungssoftware und induziert damit eine stärkere Kooperation für die gemeinsame Entwicklung und Pflege vorwettbewerblicher Produktkomponenten, da auch miteinander im Wettbewerb stehende Hersteller auf diese Weise ihre Kosten stärker senken können, als ohne eine solche „Co-Competition“.

Durch Verwendung dieser Technologien und vergleichbarer wirtschaftlicher Konzepte wird eine signifikante Kostenreduktion für das Endprodukt erwartet. Die Kosten der fahrzeugseitigen Komponenten sollen gleich oder sogar unter den Kosten von konventionellen hochwertigen Zugsicherungssystemen wie der Linienzugbeeinflussung (LZB) liegen, welche in Deutschland, Österreich und Spanien verwendet wird. Das Open Source Konzept in Verbindung mit formalen Methoden ermöglicht ein frei zugängliches, herstellerneutrales und „korrektes“ Referenzsystem, welches dazu beitragen wird, Probleme der Interoperabilität leichter und schneller zu beseitigen. Weiterhin wird das openETCS Konzept Hersteller, Eisenbahninfrastrukturbetreiber und Eisenbahnverkehrsunternehmen durch die Vermeidung von intensiven Betriebsversuchen unterstützen. Dies wird durch die Verlagerung der Verifikation und Validierung von der Strecke in Labore ermöglicht, wodurch wertvolle Ressourcen eingespart werden. Das openETCS Projekt wird somit die Migration nach ETCS beschleunigen und unterstützt damit die europaweite Umsetzung von ERTMS.

Die wesentlichen angestrebten Ergebnisse des Projekts lassen sich wie folgt zusammenfassen:

- Eine formale Referenz-Spezifikation und Implementierung sowie Referenz-Architektur für eine ETCS OBU.
- Eine offene Werkzeugkette für die Erstellung des ETCS Basis-Softwareproduktes.
- Das o. g. Basis-Produkt, das von den Herstellern in Geräte integriert werden kann.
- Eine integrierte Werkzeugkette für formale Spezifikation und Verifikationstechnologien.

- Eine zugehörige Methodik, welche die Konformität mit den relevanten Sicherheitsnormen gewährleistet.

2 Voraussetzungen, unter denen das Vorhaben geführt wurde

Der European Vital Computer (EVC) stellt den Kern eines ETCS On-Board Systems dar. Dieser sicherheitsrelevante Rechner implementiert die Funktionen der ETCS Anforderungsspezifikation (SRS, Subset-026 der ERA), um die Sicherheit von Bahnfahrten zu garantieren. Der EVC ist außerdem eine wesentliche Voraussetzung, um die Interoperabilität in der ETCS Vision umzusetzen. Dafür muss das On-Board System:

- sämtliche SRS Funktionen ausführen können und
- mehrere verschiedene Versionen der SRS ausführen können, um die Abwärtskompatibilität sicherzustellen (vgl. Subset-026, Kapitel 6).

Aufgrund dieser Eigenschaften haben sich die meisten Hersteller dafür entschieden, zunächst (jeder für sich) eine generische EVC Software zu entwickeln, um diese dann in verschiedenen Anwendungsprojekten mit spezifischer Konfiguration wiederverwenden zu können. Abbildung 2 zeigt die Struktur eines solchen generischen Entwicklungsablaufs. Dieses Entwicklungsmodell sorgt dafür, dass kontinuierlich Erfahrung in die Softwareentwicklung zurückfließt und eine stetige Verbesserung der Produktreife erwartet werden kann. Es ist jedoch hervorzuheben, dass bislang nur wenig Kooperation in diesem Softwareentwicklungsprozess zwischen den einzelnen Herstellern stattgefunden hat.

Die Entwicklung der EVC Software richtet sich nach den Methoden zur Softwareentwicklung der CENELEC EN 50128 [26] für Software des Sicherheits-Integritätslevels 4 (SIL4). Das von diesem Standard geforderte Maß an Zuverlässigkeit hat erheblichen Einfluss auf den Umfang der für einen Software Release erforderlichen Aktivitäten.

Die Softwareentwicklung im ETCS Bereich ist insbesondere geprägt von den folgenden Herausforderungen:

1. Die SRS wurde in natürlicher Sprache (Englisch) verfasst und birgt die Gefahr von Softwareentwicklern unterschiedlich interpretiert zu werden. Dies kann zu unterschiedlichem Verhalten der On-Board Systeme verschiedener Hersteller führen.
2. Die SRS muss weiterentwickelt werden, da sie bisher eine Systemspezifikation darstellt, die keine konkrete Zuordnung zwischen den streckenseitigen und fahrzeugseitigen Funktionen von ETCS Systemen vornimmt.
3. Da die Softwareentwicklung gegen Ende der 90er Jahre begann, basiert diese auf klassischen Verfahren der Softwareentwicklung in Hinblick auf Nachvollziehbarkeit der Spezifikationsebenen, unabhängigem Code Review und einheitlichen Tests. Diese Verfahren benötigen einen hohen Aufwand, wenn funktionale Veränderungen notwendig werden.
4. Die SRS wurde in den vergangenen zehn Jahren stetig überarbeitet und verbessert. Die stetig zunehmende Betriebserfahrung führt zu neuen Erkenntnissen und damit zu einer Weiterentwicklung der SRS. Die in Folge dessen immer wieder erforderlichen Anpassungsarbeiten an der SRS führen zu einer kostspieligen Produktpflege.

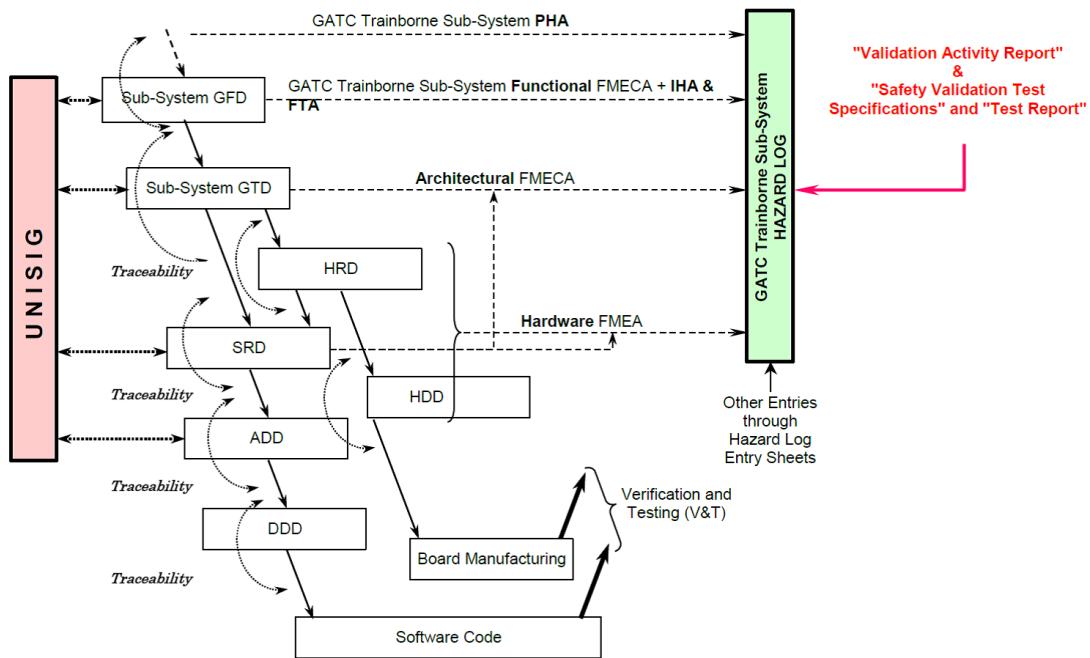


Abbildung 2. Struktur eines generischen Entwicklungsablaufs am Beispiel der EVC Entwicklung (Quelle: Alstom).

5. Die parallele und unabhängige Entwicklung von EVC Software durch mehrere Hersteller erfordert hohe Investitionskosten und verursacht somit teure Endprodukte.

Obwohl mehrere große europäische Anbieter mit erheblichem Know-how im Bereich der Zugsictherungstechnologien seit mehr als einem Jahrzehnt an einer gemeinsamen Systemanforderungsspezifikation arbeiten, wurde das Ziel der Interoperabilität bislang nicht erreicht, d. h. bis heute wurde keine ETCS On-Board für alle existierenden europäischen ETCS Strecken zugelassen. Die aktuellen Entwicklungen des ETCS müssen daher als „Work in Progress“ bezeichnet werden, die in naher Zukunft eine Vielzahl an erforderlichen Software Updates nach sich ziehen wird.

Die fehlende Standardisierung auf unterschiedlichen Stufen, unterschiedliche nationale Abnahmeprozeduren und verschiedene Funktionsweisen, in Kombination mit der Unterstützung von verschiedenen bestehenden Systemen sorgen für eine lange Übergangsphase und stellen einen wesentlichen Kostentreiber dar. Beinahe alle auf dem Markt befindlichen Produkte basieren auf unterschiedlichen proprietären Softwaredesigns. Daraus ergibt sich eine lebenslange Abhängigkeit zu dem ursprünglichen Hersteller, was wiederum in hohen Lebenszykluskosten für die Fahrzeughalter resultiert, aber auch zu Investitionsrisiken bei Ausfall einzelner Hersteller führt.

Entgegen der aktuellen Situation im Bahnsektor existieren in anderen Branchen mehrere Programme, bei denen Wettbewerber nicht nur eine gemeinsame funktionale Spezifikation verwenden, sondern gemeinsame Gestaltungsprinzipien und sogar gemeinsame Schnittstellen oder Hardware-/Software Designs verwenden (z. B. AUTOSAR) oder eine gemeinsame Entwicklungsumgebung verwenden (z. B. TOPCASED). Auf diese Weise werden die Entwicklungskosten geteilt und die Qualität des Gesamtprodukts verbessert.

3 Planung und Ablauf des Vorhabens

Um die genannten Ziele des openETCS Projekts zu erreichen, wurde das Projekt in größere Arbeitspakete (AP) aufgeteilt und weiter in Einzelaufgaben unterteilt. Diese Arbeitspaketstruktur sollte dazu dienen, den Projektsteuerungsaufwand durch ergebnisgetriebene, eng zusammenarbeitende Projektteams minimal zu halten. Die Arbeitspakte selbst lassen sich wie folgt in drei Kategorien einteilen (siehe Abb. 3):

- Projekt Governance und der Ergebnisverwertung, Veröffentlichung und Standardisierung (AP1, AP6),
- Architektur, Use-Case und Validierung Arbeitspakte (AP2, AP4), sowie
- Arbeitspakte der technischen Entwicklung und Demonstration (AP3, AP5 und AP7).

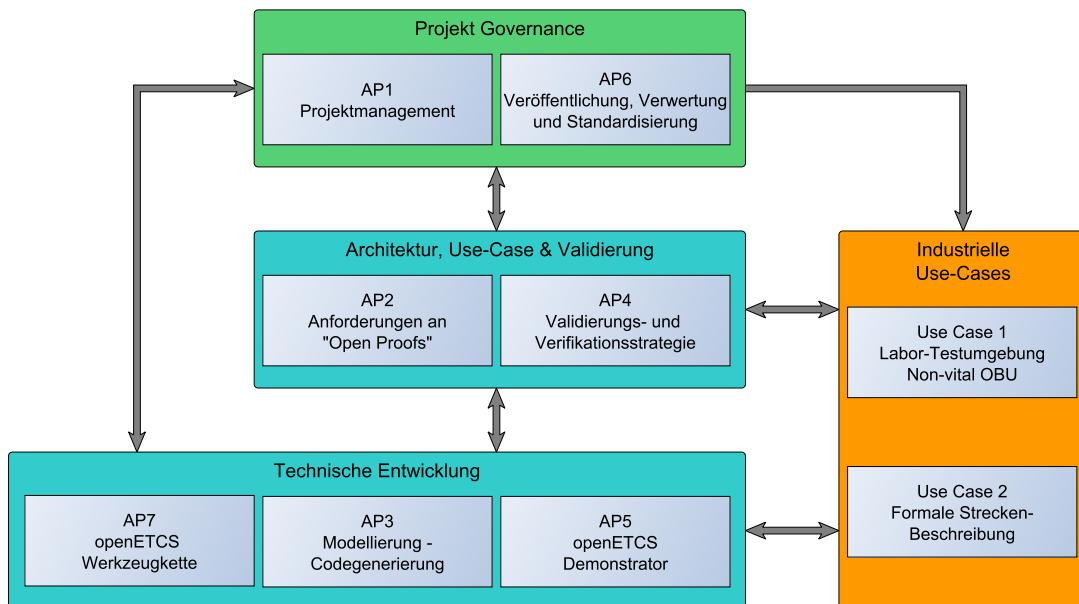


Abbildung 3. Struktur der Arbeitspakte.

Zudem wurden zwei industrielle Use-Cases (Fallbeispiele) definiert: Eine Labortestumgebung sowie eine formale Modellierung einer realen ETCS Strecke, welche typische Anwendungen repräsentieren soll. Im Folgenden werden die inhaltlichen Schwerpunkte der Arbeitspakte kurz dargestellt.

Arbeitspaket 1 – Projektmanagement Das Ziel dieses Arbeitspaket war, das Projekt einzurichten und zu steuern sowie eine Kommunikations- und Controlling-Infrastruktur durch ein gut definiertes Governance-Konzept aufzubauen. Darüber hinaus wurden Leitlinien für die Durchführung von Qualitätsmanagement definiert und überwacht. Schließlich regelte das Arbeitspaket die Pflege des geistigen Eigentums.

Arbeitspaket 2 – Anforderungen an Open Proof Das Ziel des Arbeitspaket war die Definition der Anforderungen an die mit der Modellierung, Verifikation und Validierung verbundenen Prozesse, um den Sicherheitsanforderungen des ERTMS Systems zu genügen.

Arbeitspaket 3 – Modellierung – Codegenerierung Das Arbeitspaket 3 erstellte ein formales Modell des ETCS OBU Systems sowie die ausführbare ETCS OBU Software. Die OBU Software wurde dabei mittels eines Code-Generators aus dem formalen Modell generiert.

Arbeitspaket 4 – Validierungs- und Verifikationsstrategie Dieses Arbeitspaket konzentrierte sich auf die Validierung und Verifikation (V&V) des in Arbeitspaket 3 erstellten formalen Modells. Die Arbeiten im Arbeitspaket 4 wurden parallel zur Modellierung durchgeführt, um eine Rückkopplung zum Modellierungsprozess zu generieren und somit die Qualität und Reife des Modells zu evaluieren. Voraussetzung für das erfolgreiche Arbeiten in Arbeitspaket 4 war die Verfügbarkeit einer offenen V&V Werkzeugkette. Hierzu wurden in der frühen Phase des Projektes zunächst Werkzeuge und Methoden evaluiert, ausgewählt und angepasst. Ein weiterer Schwerpunkt war die Koordination der Einbindung relevanter Sicherheitsanforderungen in den Modellierungsprozess (z. B. CENELEC EN 50128 [26]).

Arbeitspaket 5 – openETCS Demonstrator Das Arbeitspaket stellte die Mittel zur Verfügung, um die in Arbeitspaket 3 erstellte ETCS On-Board Unit mittels eines Demonstrators präsentieren zu können. Das Arbeitspaket arbeitete im Wesentlichen mit Arbeitspaket 2 zur Schnittstellenspezifikation und mit Arbeitspaket 3 zur Integration des im Projekt erstellten Quellcodes zusammen.

Arbeitspaket 6 – Veröffentlichung, Verwertung und Standardisierung Im Rahmen des Arbeitspakets wurden sämtliche Aktivitäten der Bereiche Verwertung, Verbreitung und Veröffentlichung, sowie Standardisierung organisiert und überwacht. Ziel war es, die Projektergebnisse von openETCS laufend zu präsentieren und zu verbreiten, die Verwertungspläne der Projektpartner stetig weiter zu entwickeln und die Marktchancen zu präzisieren, sowie die Standardisierung als Grundlage für die Interoperabilität von openETCS voranzutreiben.

Arbeitspaket 7 – openETCS Tool Chain Das Arbeitspaket stellte die auf formalen Methoden basierende Werkzeugkette bereit, die zur Modellierung, Entwicklung sowie Verifikation und Validierung des ETCS Systems erforderlich war. Die Werkzeugkette sollte die Systementwicklung auf allen Ebenen (von der ganzheitlichen Sichtweise auf das System bis zur Codegenerierung) ermöglichen. Insbesondere Modellierungssprache(n) und die Rückverfolgbarkeit (Traceability) standen dabei im Vordergrund.

Diese Struktur sollte das schnelle Erarbeiten guter Lösungen ermöglichen, aber auch die Überarbeitung erleichtern, da auf dem Weg zu professionell einsetzbaren Produkten und Werkzeugen mit mehreren Iterationsschritten zu rechnen war. In jedem der Arbeitspakte konnte weitgehend unabhängig an einer Lösung auf Basis bereits vorliegender Ergebnisse der anderen Arbeitspakte aus dem vorherigen Iterationsschritt gearbeitet werden. Im Projekt wurden dabei agile Methoden (Scrum) eingesetzt, die sich in der Wirtschaft als besonders produktiv erwiesen haben.

4 Wissenschaftlicher und technischer Stand

Der derzeitige technische Stand und die Voraussetzungen bezüglich der Entwicklung von EVC Software im Bahnsektor wurde bereits in Kapitel 2 dargestellt. Ergänzend wird im folgenden Kapitel 4.1 der wissenschaftliche und technische Stand bzgl. domänenspezifischer Sprachen im Bereich der Entwicklung sicherheitskritischer Software dargestellt und in Kapitel 4.2 wird auf die Besonderheiten von Open Source Projekten in Bezug auf Geschäftsmodelle eingegangen.

4.1 Domänenspezifische Sprachen für sicherheitskritische Softwareentwicklung

An Software für sicherheitskritische Systeme werden in der Regel in Form entsprechender Standards besondere Anforderungen gestellt. Im Allgemeinen fordern diese schon während der Entwicklungsphase bestimmte Dokumentationsverfahren und Entwicklungsmethoden. Deswegen können solche Standards auch als Software Lifecycle orientiert bezeichnet werden. Betrachtet man wiederum die typischen Entwicklungsverfahren bei Open Source Software (OSS), wird schnell klar, dass diese nicht ohne weiteres die Anforderungen eines Standards erfüllen können. In der Welt des OSS wird meist nur auf Basis des Source Code entwickelt, selten werden konkrete Dokumentationsstrategien benutzt. Eine Software Lifecycle orientierte Entwicklung scheint so ausgeschlossen.

Eine gute Möglichkeit dieses Problem zu lösen, ist die Entwicklung nicht mehr auf reiner Source Code Basis zu betreiben, sondern stattdessen Modelle zu verwenden. Wird die komplette Entwicklung basierend auf Modellen durchgeführt, wird das Ganze auch „Model Driven Software Development“ (MDSD) genannt.

MDSD ist dabei nicht auf konkrete Modellsprachen oder –Typen festgelegt. Eine weitverbreitete Modellsprache ist die Unified Modelling Language (UML), welche unterschiedliche Diagramm bzw. Modelltypen für die Entwicklung von objektorientierter Software bietet. Ihr größter Vorteil, nämlich dass diese universell ist, ist leider auch ihr größter Nachteil. D. h. mittels UML-Modellen lassen sich praktisch die meisten objektorientierten Software-Konstrukte modellieren, welche dabei aber wenig Abstraktion im Vergleich zum reinen Source Code bieten. Gerade aber Abstraktion vereinfacht den Entwicklungsprozess oder hilft dabei Fehler zu finden. Sind die verwendeten Modelle nicht wesentlich abstrakter als der entsprechende Source Code, ist der Nutzen für die Entwicklung eher gering und ihr Einsatz kaum zu rechtfertigen. Ein weiteres Problem ist in diesem Fall, dass selbst aus komplexen und aufwendigen UML-Modellen i.d.R. nicht der gesamte, benötigte Source Code erzeugt werden kann. Dieser Prozess wird auch Generierung genannt. Wenn es jedoch notwendig ist parallel zu den Modellen auch noch Source Code zu entwickeln, kann nicht mehr von MDSD gesprochen werden.

Eine Alternative zu UML oder generell zu jeder allgemeinen Modellsprache ist der Einsatz sogenannter domänenspezifischer Sprachen (Domain Specific Language, DSL). Wissenschaftlich schon seit geraumer Zeit von Interesse, propagiert dieses Konzept die Entwicklung einer Modellsprache für eine konkrete Anwendungsdomäne – hier z. B. Signaltechnik, ETCS.

Die Entwicklung einer ETCS Implementierung bzw. einer openETCS Implementierung für die On-Board Unit eines Triebfahrzeugs ist ein Paradebeispiel für den Einsatz einer DSL. Die konkreten Anforderungen an eine openETCS-Modellsprache können prinzipiell von den relevanten

Teilen der ETCS Spezifikation (SRS, Subset-026) abgeleitet werden. Idealerweise sollte eine Modellsprache für openETCS es ermöglichen, die komplette relevante Spezifikation geschlossen zu modellieren, ohne dass dabei Experten der Softwareentwicklung eingebunden werden müssen.

Um aus der modellierten Spezifikation ausführbare Software für die On-Board Unit zu erzeugen, werden Generatoren eingesetzt. Typischerweise generieren diese dann Source Code, welcher anschließend kompiliert werden kann. Diese Generatoren müssen natürlich zusätzlich zur Modellsprache entwickelt werden. Vorteilhaft gegenüber einer nur auf Source Code basierenden Entwicklung ist, dass eine DSL bzw. Generatoren nur einmal für eine Modellsprache entwickelt werden müssen und dann für jegliche konkreten Modelle in dieser Sprache verwendet werden können. Des Weiteren bietet der Einsatz einer DSL auch den Vorteil, dass so implizit weitere Artefakte zum (generierten) Source Code direkt durch die Entwicklung zur Verfügung stehen, welche für die Zertifizierung der Software für Standards eingesetzt werden können.

Eine neuer Ansatz ist deshalb anstelle von Open Source Software basierend auf reinem Source Code diese mittels offener Modelle zu entwickeln. Solche Software wird analog dazu Open Model Software (OMS) genannt. D. h. es wird nicht nur der generierte Source Code unter den Prinzipien von OSS veröffentlicht, sondern vor allem die Modellsprache, die Modelle und die Generatoren.

4.2 Open Source: Vorgehensmodelle, Geschäftsmodelle und Lizenzen

Im Unterschied zum bisherigen Vorgehen der Industrie versucht das openETCS Projekt einerseits durch Formalisierung unterschiedliche Interpretationen der Systemanforderungsspezifikation (SRS) zu vermeiden, aus der formalen Spezifikation über Codegeneratoren die ausführbare Software zu gewinnen und zudem sämtliche Dokumente, Werkzeuge, Softwareprodukte und Nachweise gemäß dem „Open Proofs“ Konzept unter eine Open Source Lizenz zu stellen und im Rahmen eines Open Source Projektes zu entwickeln. Diese Vorgehensweise ist für sicherheitsrelevante Systeme im Bahnbereich völlig neu.

Das Geschäftsmodell besteht nun darin, dass alle Komponenten, in denen sich Hersteller nicht differenzieren können, da sie funktionsidentisch sind oder aus Sicht des Nutzers keinen Gewinn bringen, wenn sie sich von Hersteller zu Hersteller unterscheiden, unter eine Open Source Lizenz gestellt werden und damit für den Sektor frei zugänglich gemacht werden. Der Nutzen dieser Vorgehensweise für die Hersteller besteht darin, dass diese gemeinsamen Komponenten dann wesentlich kostengünstiger hergestellt und langfristig gepflegt werden können, als wenn jeder Hersteller sich individuell um die Produktpflege kümmert. Ungeachtet dessen können die einzelnen Produkte der im Wettbewerb stehenden Hersteller sich in denjenigen Punkten unterscheiden, welche die Interoperabilität nicht beeinflussen, wie beispielsweise die Ausgestaltung der Hardware oder zusätzliche Ergänzungsfunktionen (wie z. B. Diagnosefunktionen). Ein weitere Differenzierung kann über die mit dem Produkt in Verbindung stehenden Dienstleistungen und deren Umfang oder Qualität erfolgen (Systemintegration, Update-Service, etc.).

5 Zusammenarbeit mit anderen Stellen

Das deutsche openETCS Konsortium war Teil des europäischen Gesamtkonsortiums innerhalb der ITEA2 Initiative. Die Leitung des europäischen Gesamtprojekts fand dabei durch die Deutsche Bahn / DB Netz AG statt. Insgesamt bestand das deutsche Konsortium aus den folgenden Partnern bzw. Institutionen:

- Angewandte Eisenbahntechnik GmbH (AEbt)
- Alstom Transport Deutschland GmbH
- Deutsches Zentrum für Luft- und Raumfahrt e.V. (DLR)
- Deutsche Bahn / DB Netz AG (Konsortialleitung)
- Eclipse Source / Innoopract Informationssysteme GmbH
- Formal Mind GmbH
- Fraunhofer-Gesellschaft (FOKUS, ESK)
- Siemens AG
- Technische Universität Braunschweig
- TWT GmbH Science & Innovation
- Universität Bremen
- Universität Rostock

Von den rund 167 Personenjahren im europäischen Gesamtprojekt entfielen rund 61 Personenjahre und somit ca. 37% des Gesamtaufwandes auf die deutschen Partner.

5.1 Zusammenarbeit im internationalen ITEA Konsortium

Im Projekt erfolgte eine intensive Zusammenarbeit zwischen dem deutschen Konsortium und den weiteren europäischen Partnern. Im Folgenden werde diese und ihre jeweilige Rolle im Konsortium kurz dargestellt.

5.1.1 Französische Projektpartner

Der französische Cluster war mit ca. 68 Personenjahren am Projekt beteiligt und hatte somit einen Anteil von rund 41% am Gesamtaufwand des ITEA Projekts. Die Projektpartner sind im Einzelnen:

All4tec Das Unternehmen hat sich auf Sicherheitsanforderungen und die Systemvalidierung spezialisiert. Schwerpunkt der Aktivitäten im Projekt lag somit im AP4 (Verifikations- und Validierungsstrategie) sowie im AP3 (Modellierung).

Alstom Alstom ist eines der führenden Unternehmen auf dem ETCS Markt und hat sein ETCS- und industrielles Know-how in das Projekt eingebracht. Der Schwerpunkt der Aktivitäten seitens Alstom lag im AP3 (Modellierung), welches durch Alstom geleitet wurde, sowie im AP2 (Anforderungen).

CEA CEA wurde im Projekt durch CEA LIST (Laboratory of Integration of Systems and Technologies) vertreten. Das Labor ist federführend bei der Entwicklung der Open Source

Modellierungsumgebung Papyrus, welches einen wesentlichen Bestandteil der openETCS Tool Chain darstellt. Der Schwerpunkt der Aktivitäten seitens CEA LIST lag entsprechend im AP7 (openETCS Tool Chain) und weiterhin im AP4 (Verifikations- und Validierungsstrategie).

ERSA ERSA ist ein mittelständiges Unternehmen, das sich auf die Entwicklung von Software Anwendungen im Bahnbereich spezialisiert hat. Zum Portfolio des Unternehmens zählen ETCS Simulations- und Trainings-Tools sowie Prüfstände für die Eisenbahnindustrie. Schwerpunkt der Aktivitäten seitens ERSA lag folglich im AP5 (openETCS Demonstrator), welches auch durch ERSA geleitet wurde.

INPT Das Institut National Polytechnique de Toulouse (INPT) ist ein Verbund dreier höherer Technischer Schulen („Grandes Ecoles“ – ENSAT, ENSEEIHT, ENSIACET). Der Schwerpunkt der Projektaktivität lag im AP4 (Verifikations- und Validierungsstrategie) sowie im AP3 (Modellierung).

Institut Télécom Das Institut Télécom setzt sich aus mehreren „Grandes Ecoles“ im Bereich der Informations- und Kommunikationstechnologien zusammen und verfügt über umfassende Erfahrung bei der Anwendung von Testmethoden und zugehörigen Software-Tools. Der Arbeitsschwerpunkt des Instituts konzentrierte sich im Wesentlichen auf AP4 (Verifikations- und Validierungsstrategie).

LAAS-CNRS LAAS, als Teil des Centre National de la Recherche Scientifique (CNRS), verfügt über langjährige Erfahrung in formalen Beschreibungstechniken für sicherheitskritische eingebettete Systeme, insbesondere im Hinblick auf ihre zeitlichen Aspekte. Die Beiträge von LAAS im Rahmen des openETCS Projektes basieren auf den Ergebnissen der Projekte TOPCASED und OPEES und die dabei entwickelten Modellierungs- und Verifikations- Tools. Arbeitsschwerpunkt war dementsprechend das AP7 (openETCS Tool Chain) sowie das AP2 (Anforderungen).

Mitsubishi Electric Mitsubishi Electric R&D Center Europe (MERCE) ist ein international tätiger Hersteller von Bahnfahrzeugen und Bahnausrüstungen. MERCE stellt hierbei das europäische Forschungs- und Entwicklungszentrum des Konzerns dar und besitzt umfassende Expertise im Bereich formaler Spezifikation und Verifikation von sicherheitskritischen Systemen. Arbeitsschwerpunkte von MERCE lagen im AP3 (Modellierung) und im AP2 (Anforderungen).

SNCF Der französische Bahnbetreiber SNCF ist ein führendes Unternehmen im Bereich Mobilität und Logistik. Im Stellwerksbereich verfügt SNCF über langjährige und positive Erfahrung beim Einsatz formaler Methoden bei der Spezifikation sicherheitsrelevanter Systeme sowie beim Sicherheitsnachweis und setzt hier u. a. Petri-Netze ein. im openETCS Projekt konzentrierte SNCF sein Engagement auf AP2 (Anforderungen) und leitete dieses Arbeitspaket.

Systerel Das Unternehmen Systerel ist spezialisiert auf die Entwicklung, Validierung und Evaluierung von sicherheitskritischen Echtzeitssystemen. Etwa 80% dieser Aktivitäten sind mit SIL4 oder SIL2 Sicherheitsniveau verbunden und rund ein Drittel davon werden mit formalen Methoden entwickelt und ausgewertet, vor allem mit der so genannten B-Methode. Systerel war wesentlich in AP3 (Modellierung), AP4 (Verifikations- und Validierungsstrategie), AP2 (Anforderungen) sowie AP7 (openETCS Tool Chain) involviert.

5.1.2 Spanische Projektpartner

Der spanische Cluster war mit ca. 17 Personenjahren am Projekt beteiligt und hatte somit einen Anteil von rund 10% am Gesamtaufwand des ITEA Projekts. Die spanischen Projektpartner sind im Einzelnen:

Innovalia Die Innovalia Association ist ein assoziiertes Forschungslabor, welches mit dem Auftrag zur Förderung der technologischen Innovation in der Wirtschaft gebildet wurde und insbesondere kleine und mittlere Unternehmen (KMU) unterstützt. Die Innovalia Association wurde von einer Gruppe technologieorientierter KMUs gegründet. Innovalias Hauptbetätigungsfelder sind u. a. Forschungs- und Entwicklungsprojekte, Innovationsmanagement, System-Design und die Einführung von neuen Technologien. Der Schwerpunkt der Projekttätigkeit lag im AP4 (Verifikations- und Validierungsstrategie) sowie im AP2 (Anforderungen).

Software Quality Systems S.A. Software Quality Systems S.A. (SQS) ist ein unabhängiges Testinstitut und spezialisiert auf die Validierung und Verifikation von Prozessen und Werkzeugen. SQS stellt unter anderem Testplattformen zur Validierung von funktionalen und nicht-funktionalen (Interoperabilität, Benutzerfreundlichkeit, Belastung, etc.) Anforderungen zur Verfügung. Im Projekt lagen die Arbeitsschwerpunkte von SQS in den folgenden Arbeitspaketen: AP1 (Projektmanagement, Quality Assurance), AP4 (Verifikations- und Validierungsstrategie) und AP3 (Modellierung).

5.1.3 Belgische Projektpartner

Der belgische Cluster war mit ca. 12 Personenjahren am Projekt beteiligt und hatte somit einen Anteil von rund 7% am Gesamtaufwand des ITEA Projekts. Die belgischen Projektpartner sind im Einzelnen:

Alstom Alstom Belgien ist eins der führenden Unternehmen auf den weltweiten ETCS Markt und hat insbesondere ETCS- und industrielles Know-how in das Projekt eingebracht. Die Schwerpunkte der Aktivitäten lagen im AP3 (Modellierung) und im AP2 (Anforderungen).

ERTMS-Solutions ERTMS-Solutions ist ein mittelständisches Unternehmen und bietet Produkte und Dienstleistungen rund um ERTMS / ETCS an. Im Projekt lag der Arbeitsschwerpunkt im AP3 (Modellierung) und AP7 (openETCS Tool Chain), hier insbesondere beim Open Source Software-Tool ERTMS Formal Specs.

5.1.4 Niederländische Projektpartner

Der niederländische Cluster war mit ca. 5 Personenjahren am Projekt beteiligt und hatte somit einen Anteil von rund 3% am Gesamtaufwand des ITEA Projekts. Die niederländischen Projektpartner sind im Einzelnen:

Nederlandse Spoorwegen Der Bahnbetreiber Nederlandse Spoorwegen (NS) ist aus der niederländischen Staatsbahn hervorgegangen. Arbeitsschwerpunkte von NS lagen im AP2 (Anforderungen) und AP3 (Modellierung). Zudem wurde durch NS die Streckenbeschreibung der ETCS Strecke Amsterdam Utrecht bereitgestellt, welche für den openETCS Demonstrator genutzt wurde.

Lloyd's Register Rail B.V. Lloyd's Register Rail Europe B.V. (LRRE) ist Teil der Transportsparte der Lloyd's Register Group. LRRE ist in Utrecht angesiedelt und bietet Dienstleistungen auf dem Gebiet des Eisenbahnwesens an. LRRE ist als „benannte Stelle“ („notified body“, NoBo) für die Zulassung von Eisenbahnfahrzeugen und Anlagen qualifiziert. Die Arbeitsschwerpunkte von LRRE lagen im AP3 (Modellierung).

5.1.5 Italienische Projektpartner

Der italienische Cluster war mit ca. einem Personenjahr am Projekt beteiligt und hatte somit einen Anteil von rund 1% am Gesamtaufwand des ITEA Projekts. Der Cluster bestand nur aus einem Unternehmen:

GE Transportation GE Transportation³ ist ein weltweit führender Hersteller von dieselelektrischen Lokomotiven. Ferner bietet GE Lösungen für OBUs, Kommunikations-, Zugsteuer- und Informationssysteme an. Im Projekt konzentrierten sich die Beiträge von GE Transportation auf AP5 (Demonstrator), wo unter anderem ETCS OBU Hardware für den openETCS Demonstrator bereitgestellt wurde.

5.1.6 Britische Projektpartner

Der britische Cluster war mit ca. 0,1 Personenjahren am Projekt beteiligt und hatte dementsprechend lediglich einen geringen Anteil am Gesamtaufwand des ITEA Projekts. Der Cluster bestand wiederum aus nur einem Unternehmen:

ATOC ATOC ist die Vereinigung von 24 Eisenbahnverkehrsunternehmen (EVU) in Großbritannien und stellt eine Plattform für die Kooperation dieser EVUs auf verschiedenen Gebieten dar, u.a. beim Engineering und der Ausrüstungsstrategie bei ETCS in Zusammenarbeit mit Network Rail, dem britischen Netzbetreiber. Schwerpunkt der Aktivitäten seitens ATOC lag im AP6 (Veröffentlichung, Verwertung und Standardisierung), insbesondere für Großbritannien.

5.2 Zusammenarbeit mit externen Partnern

Durch die offene, auf „Open Proofs“ ausgelegte Arbeitsweise in openETCS haben sich während der Laufzeit des Vorhabens mehrere Kooperationen mit externen Unternehmen ergeben. Diese Unternehmen sind durch die Öffentlichkeitsarbeit auf openETCS aufmerksam geworden und sehen im Projekt und ihrer Beteiligung eine Chance neue Produkte im Bereich der Zugsicherungstechnik zu platzieren. Im Folgenden werden diese Unternehmen und Ihre Beiträge zum Projekt, welche jeweils aus Eigenmitteln finanziert wurden, kurz dargestellt.

5.2.1 LEA Railergy

LEA Railergy ist ein in Augsburg ansässiges Kleinunternehmen, welches sich neben der Entwicklung von Software und Systemen zur intelligenten Steuerung von Energienetzen auch auf die Entwicklung von Software und Systemen im Bahnbereich spezialisiert hat.

Im Rahmen des Projekts war LEA Railergy zunächst als Unterauftragnehmer der DB Netz AG tätig, um im Arbeitspaket 3 die Modellierungsaktivitäten im SCADE Bereich zu unterstützen. Im Projektverlauf ergaben sich jedoch auch umfassende Aktivitäten im Arbeitspaket 5 (openETCS Demonstrator), die weitgehend eigenfinanziert waren. Hier entwickelte LEA Railergy zusammen mit dem Unternehmen Bachleitner & Heugel Elektronik (siehe nächster Abschnitt) einen weiteren openETCS Demonstrator. Für weitere Details bezüglich dieses Demonstrators verweisen wir auf Kapitel 6.6.4.

³Gegen Ende des Projektes wurde GE Transporation durch Alstom übernommen.

5.2.2 Bachleitner & Heugel Elektronik

Bachleitner & Heugel Elektronik ist ein mittelständisches Unternehmen, welches sich auf die Entwicklung und Fertigung von kundenspezifischen Anzeigegeräten und elektronischen Baugruppen mit dem Schwerpunkt Bahntechnik spezialisiert hat. Zu den Kunden zählen viele namhafte Großunternehmen aus dem Bereich der Zugsicherungstechnik bzw. dem Bahnbereich.

In der zweiten Projekthälfte entstand eine enge Kooperation zwischen Bachleitner & Heugel Elektronik und dem openETCS Modellierungsteam (Arbeitspaket 3). Ergebnis dieser Kooperation war der bereits im vorigen Abschnitt erwähnte openETCS Demonstrator (für Details siehe Kapitel 6.6.4). Dieser wird zurzeit zur Produktreife weiterentwickelt und dürfte dann das erste auf openETCS Ergebnissen basierende industrielle Produkt darstellen.

5.2.3 MEN Mikro Elektronik

MEN Mikro Elektronik ist ein international tätiges Unternehmen im Bereich der Elektronikentwicklung für industrielle Rechnersysteme und eingebettete Systeme mit Stammsitz in Nürnberg. Das Unternehmen besitzt umfassende Erfahrung im Bahnbereich und zählt die wichtigsten Ausrüster im Bereich der fahrzeugseitigen Sicherungssysteme zu seinen Kunden. Im Rahmen des openETCS Projekts wurde seitens MEN an einem openETCS Demonstrator gearbeitet, welcher das openETCS OBU Modell auf MEN eigener OBU Hardware ausführt.

5.3 Zusammenarbeit mit anderen Projekten und Institutionen

Während der Laufzeit des Vorhabens stand das openETCS Projekt im engen Kontakt und gegenseitigem Austausch mit den folgenden Projekten und Institutionen.

5.3.1 Eclipse Foundation

Die Eclipse Foundation⁴ ist eine nicht auf Gewinn orientierte Organisation mit der Aufgabe, die Eclipse Open Source Gemeinschaft, ihre Projekte und das zugehörige Ökosystem zu koordinieren. Da auch die openETCS Tool Chain auf Eclipse basiert, stand das Projekt im kontinuierlichem Austausch mit der Eclipse Foundation. Dies geschah zum einen über den openETCS Projektpartner Eclipse Source / Innoopract, welcher zu den „Strategic Members“ der Eclipse Foundation zählt. Weiterhin fanden insbesondere in der ersten Phase des Projektes mehrere Workshops und Meetings zum Austausch mit der Eclipse Foundation statt. Nicht zuletzt war das openETCS Projekt auf der Eclipse Con mit Vorträgen vertreten (vgl. Kapitel 11), wodurch der Austausch mit der Eclipse Community sichergestellt wurde.

5.3.2 PolarSys

PolarSys⁵ ist eine Eclipse Industry Working Group, in der sich Industrieunternehmen und Hersteller von Softwarewerkzeugen zusammengefunden haben, um gemeinsam an der Entwicklung von Open Source Werkzeugen für die Entwicklung eingebetteter Systeme zu arbeiten. Zu den Kernbereichen der Werkzeuge zählen Modellierung, Traceability, Konfigurationsmanagement, etc. Zu diesen Werkzeugen zählt auch Papyrus für die SysML und UML Modellierung. Die Weiterentwicklung von Papyrus wird dabei durch den französischen openETCS Partner CEA geleitet, so dass ein gegenseitiger und kontinuierlicher Austausch zwischen dem openETCS

⁴Siehe <https://eclipse.org>.

⁵Siehe <https://www.polarsys.org>.

Projekt und der PolarSys Working Group stattgefunden hat, von dem beide Seiten erheblich profitieren konnten.

5.3.3 VeTeSS

Ziel des Artemis Joint Undertaking (JU) VeTeSS⁶ (Verification and Testing to Support Functional Safety Standards) ist die Entwicklung neuer Prozesse, mit denen sicherheitskritische Systeme konform zu der ISO-Norm 26262 effizient entwickelt werden können. Die zunehmende Komplexität sicherheitskritischer Komponenten erhöht das Risiko für Entwurfsfehler im Entwicklungsprozess. Ein Ziel von VeTeSS ist die Entwicklung einheitlicher Methoden zum kontinuierlichen Testen und Verifizieren solcher Systeme während des Entwurfs.

In den Entwicklungsprozess integrierte Verifikation und Validierung zählt auch zu den Zielen des openETCS Projekts, so dass während der Projektlaufzeit ein gegenseitiger Austausch stattfand. Unter anderem wurde ein gemeinsamer Workshop im Rahmen der IEEE INDIN Konferenz durch openETCS und VeTeSS organisiert (vgl. Abschnitt 6.7.1.3).

⁶Siehe <http://vetess.eu/>.

Teil II

Eingehende Darstellung

6 Verwendung der Zuwendung und des erzielten Ergebnisses im Einzelnen

Die Verwendung der Zuwendung wird in diesem Kapitel für jedes der in Kapitel 3 beschriebenen Arbeitspakete des Projekts einzeln dargestellt.

6.1 Arbeitspaket 1 – Projektmanagement

Das Arbeitspaket fasst die Aktivitäten bzgl. Projektmanagement und Administration zusammen. Neben der Berichterstattung über den Projektfortschritt und der Steuerung von Ressourcen waren weitere wesentliche Aufgaben die Erarbeitung eines Plans zur Qualitätssicherung sowie das Management des geistigen Eigentums.

Da für die Entwicklung im Projekt agile Methoden eingesetzt werden sollten, wurden zu Beginn des Projekts mehrere Workshops über agile Entwicklung mit Scrum sowie über die im Projekt verwendete Arbeitsplattform GitHub⁷ für die Projektteilnehmer organisiert, um einen gemeinsamen Wissensstand diesbezüglich zu schaffen. Im Rahmen des Scrum-Prozesses führte jedes der technischen Arbeitspakete ein eigenes Backlog, d. h. eine priorisierte Liste von Produktanforderungen, welche dann gemäß der gemeinsam bestimmten Prioritäten in den einzelnen Iterationsschritten (Sprints) bearbeitet wurden.

Zur Koordination des Gesamtprojekts und der Abstimmung unter den Arbeitspaketen wurde seitens Arbeitspaket 1 ein so genannter Scrum of Scrum Prozess definiert und ein openETCS Produkt-Backlog gepflegt. Die zugehörige Prozessbeschreibung ist frei verfügbar [21] und das Produkt-Backlog selbst ist ebenfalls öffentlich einsehbar.⁸ Zur Abstimmung der Arbeitspakete wurde das Produkt-Backlog wöchentlich im Rahmen eines online Meetings gepflegt, weiterentwickelt und neu priorisiert („groomt“). Als Werkzeug wurde hierbei das Webbrowser basierte Werkzeug waffle.io⁹ eingesetzt, welches ein virtuelles Scrum-Board auf Basis des in GitHub integrierten Issue Trackers implementiert. Abbildung 4 zeigt einen Screenshot des durch waffle.io visualisierten openETCS Produkt-Backlogs. Die Pflege eines produktweiten Backlogs sowie die wöchentlichen Web-Meetings zum Grooming des selbigen haben sich im Projektverlauf als essentiell und äußerst hilfreich für das Management des Projekts und die Koordination der Arbeiten des europäischen Konsortiums herausgestellt.

Weiterhin wurde wie eingangs erwähnt ein umfangreicher Plan zur Qualitätssicherung (Quality Assurance Plan, QA-Plan) erstellt und veröffentlicht [40]. Der QA-Plan definiert die Prozesse, Methoden und Werkzeuge, um das Projekt den Open Source Prinzipien, ITEA Anforderungen und agilen Prozessen gerecht werden zu lassen. Nicht zuletzt stellen die im QA-Plan festgelegten Prozesse sicher, dass die zwei wesentlichen Projektergebnisse – die openETCS Werkzeugkette (vgl. Abschnitt 6.3) und die openETCS Referenz OBU (vgl. Abschnitt 6.4) – entsprechend der CENELEC EN 50128:2011 Anforderungen in die Entwicklung sicherheitskritischer Systeme integriert werden können.

⁷GitHub (<http://www.github.com>) ist ein webbasierter Hosting Service für Git Repositories und bietet neben verteilter Versionskontrolle und Source Code Management erweiterte Funktionen für die Zusammenarbeit und Kommunikation von Entwicklerteams.

⁸Siehe <https://waffle.io/openETCS/product-backlog>.

⁹Siehe <http://waffle.io>.

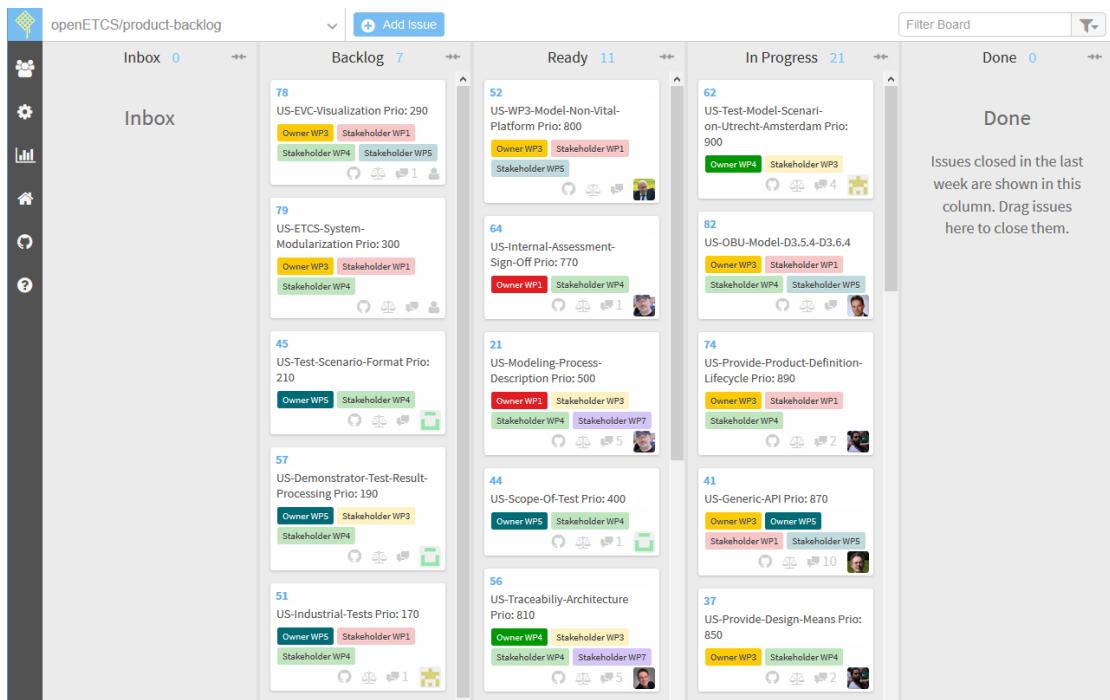


Abbildung 4. Screenshot des openETCS Produkt-Backlogs.

Nachdem die Entscheidung gefallen war das Closed Source Werkzeug SCADE für die funktionale Modellierung im Arbeitspaket 3 zu verwenden, wurden seitens Arbeitspaket 1 zudem mehrere Workshops zur Einarbeitung des Modellierungsteams in dieses Werkzeug organisiert. Zwei dieser Workshops fanden in München statt, ein weiterer in Rostock. Weiterhin wurde ein viertägiger Workshop zur Systemmodellierung mit Papyrus/SysML in der Nähe von Paris durch den französischen Projektpartner CEA organisiert.

6.2 Arbeitspaket 2 – Anforderungen an Open Proof

Das Arbeitspaket 2 hat den vorliegenden Stand der Wissenschaft und Technik für das openETCS Projekt analysiert und darauf aufbauend den Entwicklungsprozess und die zu verwendenden Methoden definiert. Daraus abgeleitet wurden Anforderungen an die Modellierung, die Programmierschnittstellen und die Werkzeuge formuliert. Dies beinhaltete Anforderungen an die Modellierungs-, Verifikations- und Nachweisprinzipien.

6.2.1 Stand der Wissenschaft und Technik

Der für das openETCS Projekt maßgebliche Stand der Wissenschaft und Technik wurde in den Ergebnisdokumenten D2.1 [43] und D2.2 [35] aufbereitet. Das erste Dokument ist ein Bericht bezüglich der bestehenden Methoden und der verwendeten Werkzeuge in der Eisenbahnsignaltechnik und in vergleichbaren Industriebereichen. Das zweite Dokument stellt die Anforderungen aus den relevanten CENELEC Standards für die openETCS Entwicklung und speziell die für das openETCS Projekt zentrale Open Proofs Methodik dar.

Die Betrachtung zu bestehenden Methoden und Werkzeugen basierte auf einer Aufarbeitung der aktuellen Literatur und auf 14 Interviews mit Experten aus dem Eisenbahnsektor und anderen verwandten Branchen hinsichtlich der von ihnen genutzten Ansätze und ihrer Erfahrungen. Die Experten stammten von verschiedenen an der Fahrzeug- und Infrastrukturentwicklung beteiligten

Unternehmen aus Belgien, Deutschland, Frankreich und der Schweiz und hatten Expertise in den unterschiedlichen Aspekten des Entwicklungszyklus.

Obwohl die Interviews in einer offenen Weise durchgeführt wurden und alle drei Bereiche (Beschreibungsmittel, Methoden und Werkzeuge) abdeckten, legten die Befragten bei ihren Antworten den Schwerpunkt auf Werkzeugeigenschaften. Dies ist leicht nachvollziehbar, da in der Praxis meist Methodik und Beschreibungsmittel durch die Verwendung eines Softwarewerkzeugs impliziert werden und daher für den Anwender nur begrenzt separierbar sind. Basierend auf der Auswertung der Interviews und der Literatur konnte aufgezeigt werden, dass bei der Softwareentwicklung in der Eisenbahnsignaltechnik bereits eine breite Palette von unterschiedlichen Beschreibungsmitteln, Methoden und Werkzeugen Anwendung findet, um die erforderlichen Entwicklungsschritte gemäß des Standards EN 50128:2011 durchzuführen. In der Regel ist mehr als ein Beschreibungsmittel erforderlich, um die unterschiedlichen Abstraktionsebenen über die Phasen des Entwicklungsprozesses zu erfassen.

Die Interviews haben gezeigt, dass formale, modellbasierte Entwicklungsmethoden bereits gängige Praxis für die niedrigeren Abstraktionsebenen der Softwarearchitektur und des Designs sind. Auf den höheren Abstraktionsebenen der Spezifikationen werden natürlichsprachliche oder semi-formale Beschreibungen verwendet, die den Einsatz von formalen Nachweisen nicht ermöglichen. Dabei existieren unterschiedliche Ansätze, um von textuellen Anforderungen, wie für ETCS vorhanden, zu ausführbarem Code zu gelangen. Dabei lag der Fokus des Berichts für openETCS auf Methoden, die auf Basis einer erstellten formalen Softwarespezifikation Code für die Zielplattform generieren, da ein solches Vorgehen von Beginn an Ziel des openETCS Projekts war. Ein solches Vorgehen erfordert einen Schritt, in dem die textuellen ETCS Systemanforderungen in Softwarespezifikationen überführt werden. Dies kann über semi-formale Modelle oder durch eine direkte Übersetzung geschehen. Dabei hat die Recherche zum Stand der Technik gezeigt, dass solche Ansätze in der Praxis nur in speziellen Fällen für begrenzte Anwendungen eingesetzt werden. In vielen Bereichen dominiert beim Design noch die manuelle Erstellung von Code. Die formalen Ansätze werden meist bei der Verifikation und Validierung angewendet. Dabei sind die Methoden, die für die Verifikation und Validierung gewählt werden, direkt abhängig von den genutzten primären Entwicklungsmethoden für die Formalisierung der System- und Software-Spezifikationen, der Erstellung von Software-Design und Architektur sowie der Quellencodeerzeugung. Abhängig von den Informationen, die auf den verschiedenen Abstraktionsebenen bereits vorliegen, kann eine formale Beschreibung erstellt werden, die auf bestimmte, vordefinierte Eigenschaften wie Konsistenz überprüft wird. Dabei existiert ein breites Spektrum an Beschreibungsmitteln, Methoden und Werkzeugen, die bereits in der Industrie für verschiedenste Nachweise Anwendung finden. Abhängig von den zu untersuchenden Eigenschaften und den vorliegenden Daten können formale Beweise, Modellchecking oder Tests angewendet werden.

In Bezug auf die Werkzeuge haben die Interviews gezeigt, dass bisher in der Industrie vorwiegend kommerzielle Werkzeuge für die Entwicklung sowie Verifikation und Validierung eingesetzt werden. Dennoch haben die meisten Interviewpartner bestätigt, dass das Konzept von Open Source Werkzeugen kein Hindernis darstellt. Entscheidende Faktoren sind hier jedoch Funktionsumfang und Stabilität, sowie ein geeignetes Servicemodell. Allgemein ist es dabei notwendig, dass die unterschiedlichen Funktionalitäten durch die Werkzeuge unterstützt werden und die Werkzeugkette dabei eine konsistente Zusammenarbeit und Datenaustausch ermöglicht. Hier haben alle Interviewpartner klar dargelegt, dass die Kombination von Werkzeugen in einer durchgängigen Werkzeugkette entscheidend für die effiziente Entwicklung ist. Dabei nutzen allerdings die meisten Industriepartner, bedingt durch ihre Rahmenbedingungen und vorangegangene Entwicklungen, sehr individuelle Werkzeugketten, die nicht direkt für das openETCS Projekt

übernommen werden konnten. Daneben existieren bereits mit TOPCASED und Rodin zwei Eclipse basierte Open Source Ansätze, welche jedoch aufgrund des fehlenden Funktionsumfangs und der Stabilität noch keine breite Anwendung in der Eisenbahnindustrie finden.

Insgesamt hat der Bericht zum Stand der Technik gezeigt, dass derzeit verschiedene Beschreibungsmittel, Methoden und Werkzeuge zur Softwareentwicklung in der Signalindustrie verwendet werden, diese jedoch noch nicht durchgängig den von openETCS angestrebten Grad an Formalisierung entsprechen. Auch existiert noch keine breite Erfahrung mit der Anwendung von Open Source Werkzeugen in der Eisenbahnindustrie. Entsprechend kam der Bericht zu dem Ergebnis, dass in den folgenden Projektschritten basierend auf einer formalen Entwicklungsmethodik eine konsistente Werkzeugkette aus vorhandenen Werkzeugen zusammengestellt werden muss, da noch keine industriell akzeptierte und umfassende Open Source Variante vorliegt.

Im Bericht zu den Anforderungen, die sich aus den relevanten Standards für das openETCS Projekt ergeben, wurden primär die Anforderungen aus der CENELEC Norm EN 50128:2011 für die Entwicklung von Software bei Eisenbahnanwendungen für die Ziele des openETCS Projekts aufbereitet. Wie im Kapitel 1 erwähnt wurde, ist ein übergeordnetes Ziel des openETCS Projekts, eine ganzheitliche Werkzeugkette (Tool Chain) für den gesamten Entwicklungsprozess zu entwickeln. Mit dieser Tool Chain soll die ETCS Software entwickelt werden. Diese zwei Artefakte (Tool Chain und ETCS Software inkl. Modell) sollten als FLOSS¹⁰, vorzugsweise unter der European Union Public License (EUPL), zur Verfügung gestellt. Wegen der Besonderheit des openETCS Projekts und unter Berücksichtigung der Projektziele sollte im Voraus geklärt werden, welche Anforderungen erfüllt werden müssen, um eine sicherheitsrelevante Entwicklung nach den europäischen CENELEC-Normen, insbesondere der CENELEC-Norm EN 50128:2011, nach dem Open Proof Prinzip zu ermöglichen.

Ein Teil des Arbeitspakets 2 soll diese Anforderungen für die verschiedenen Artefakte darstellen. In Tabelle 1 ist ein Auszug aus der Tabelle B4 in D2.2 [35] aufgeführt, um beispielhaft die in diesem Bericht aufbereiteten Grundvoraussetzungen für die openETCS *Open Proof* Entwicklung aufzuzeigen. In der Spalte „Requirements“ ist jeweils die Anforderungsnummer aus der Norm aufgelistet. Die Spalten „How the evidence shall be provided“ und „Documents to be created“ beschreiben dadurch die Art und Weise der Erstellung und wie der Nachweis im Projekt zu führen ist.

Tabelle 1. Auszug aus der Tabelle B4 in D2.2.

Requirements	How the evidence shall be provided	Documents to be created
5.3.2.1	Select a lifecycle model for the development of software and describe it in the openETCS quality assurance plan.	openETCS software lifecycle model.

Diese Anforderungen aus der EN 50128:2011 wurden für die zwei prinzipiellen openETCS Artefakte tabellarisch aufbereitet und bildeten somit die Grundlage für die weitere normenkonforme Entwicklung im openETCS Projekt.

¹⁰Abkürzung für „Free/Libre Open Source Software“ welches als hybrider Begriffe für Freie Software und Open Source Software genutzt wird, um das unterschiedliche Verständnis der beiden Richtungen zu überbrücken.

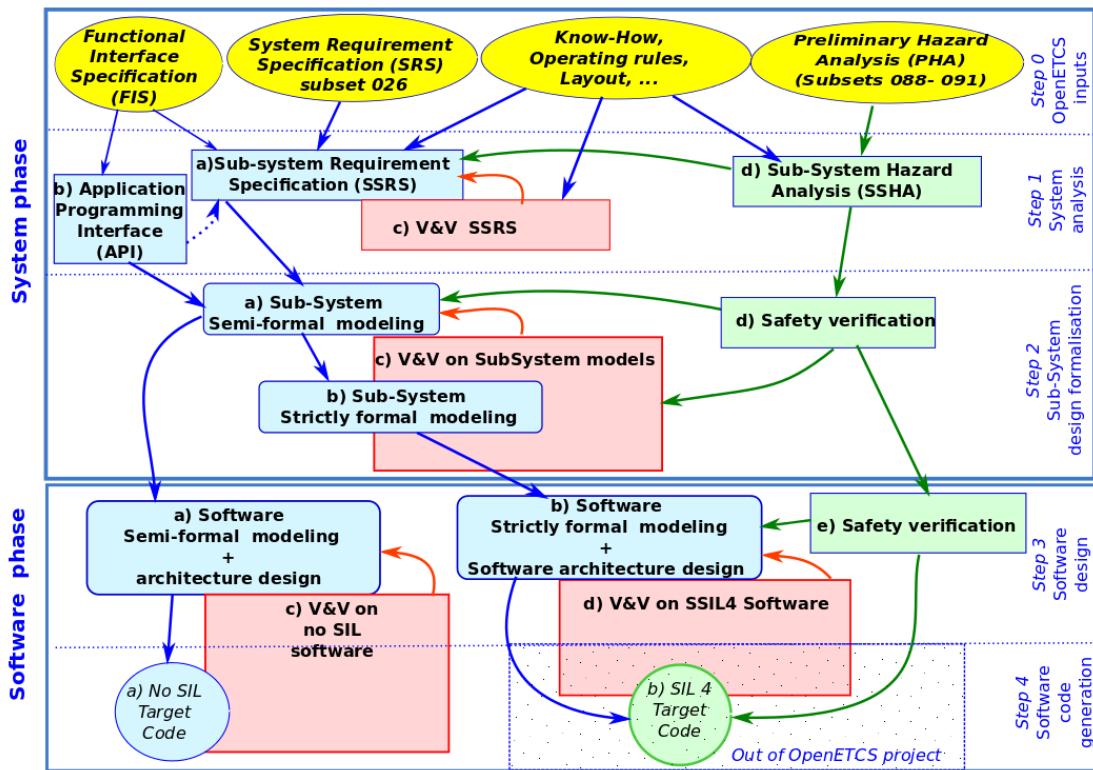


Abbildung 5. openETCS System- und Softwareentwicklungsphase.

6.2.2 Definition der Methodik

Aufbauend auf diesem Stand der Technik Analysen wurde durch AP2 in enger Diskussion mit den Projektpartnern ein Entwicklungsprozess herausgearbeitet, welcher zur Überführung der textuellen ETCS Spezifikationen in das openETCS EVC Modell und den ausführbaren Code angewendet wurde. Der Gesamtprozess und die verschiedenen Aspekte wurden in der initialen Version von D2.3 [33] ausgeführt und in den folgenden Ergänzungen des Dokuments überarbeitet und konkretisiert.

Dabei war zu berücksichtigen, dass sich die Entwicklung der EVC Software an den Entwicklungsprozess nach den CENELEC-Normen orientieren musste, um eine zukünftige industrielle Anwendung zu ermöglichen. Im Rahmen des AP2 wurde dieses Ziel in zwei Phasen realisiert. Als erstes wurde die Prozessdefinition der System- und Softwarephase in einzelne Schritte unterteilt. Abbildung 5 stellt diese Schritte dar.

Aufgrund der Komplexität des openETCS Projekts als Forschungs- und Entwicklungsprojekt war es notwendig, einige ursprünglich beabsichtigte Entwicklungsziele zurückzustellen. Zugleich sind jedoch auch neue Aspekte hinzugekommen. Im zweiten Schritt wurde der openETCS Entwicklungsprozess aktualisiert und erweitert. Das Ergebnis ist ein Entwicklungslebenszyklus entsprechend der CENELEC-Normen EN 50128:2011 und EN 50126:1999, der alle notwendigen Phasen für die Softwareentwicklung beinhaltet. Somit ist eine High-Level-Ansicht eines geeigneten Verfahrens für die Entwicklungsaktivitäten im openETCS Projekt definiert worden. Das Projektteam ist in der openETCS Entwicklung diesem Lebenszyklus gefolgt und die Entwicklungsergebnisse des Projekts sind Artefakte dieses Entwicklungsprozesses oder Teile solcher Artefakte.

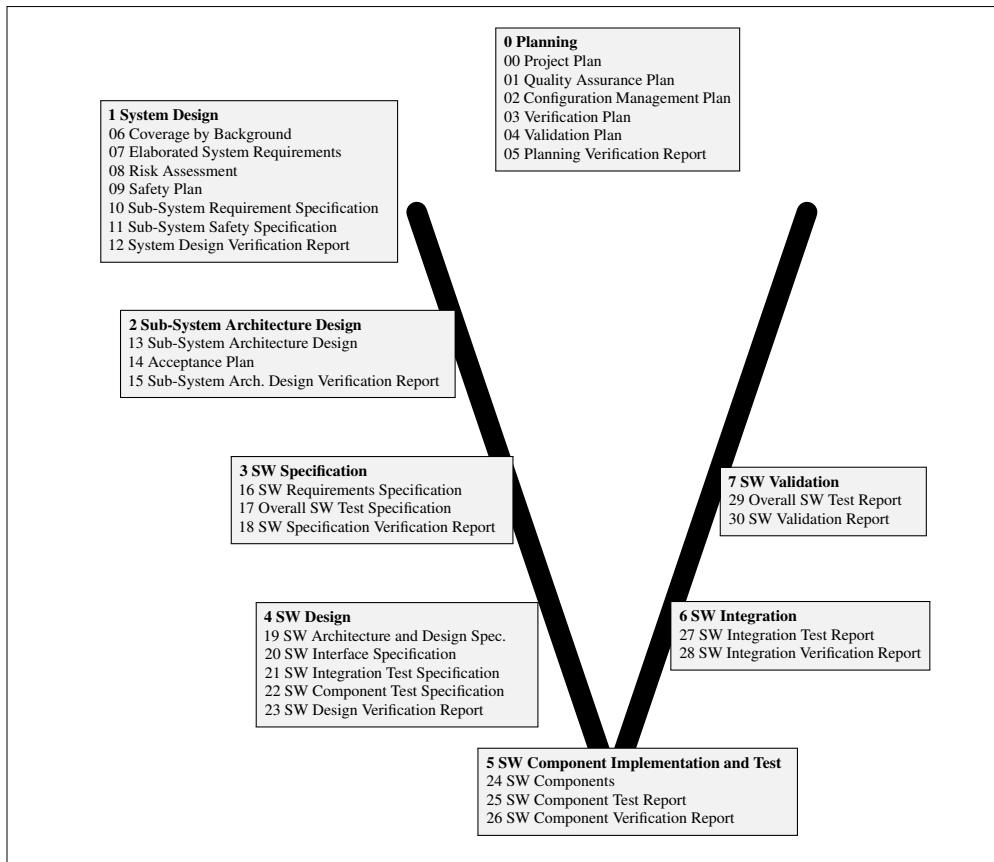


Abbildung 6. openETCS Entwicklungs-Lebenszyklus.

Abgerundet wurde diese Prozessbeschreibung in der Schlussphase des Projektes durch das Dokument D2.3a [18], die Dokumentation eines CENELEC konformen und gleichzeitig agilen Prozesses für die EVC Software Entwicklung. In diesem Dokument wurden die in der openETCS Softwareentwicklung gemachten Erfahrungen dokumentiert und für Folgeprojekte bereitgestellt.

Anschließend wurde durch AP2 in Abstimmung mit den Projektpartnern aus der Entwicklung und Verifikation und Validierung der Entwicklungsprozess weiter präzisiert und die konkreten Methodiken zur System- und Softwareformalisierung in den verschiedenen Entwicklungsphasen von openETCS eingegrenzt. Dazu wird im Dokument D2.4 [34] eine Definition der verschiedenen methodischen Konzepte und deren Zusammenwirken dargestellt. Dabei gibt das Dokument zunächst eine Einführung in formale Methoden und deren allgemeine Anwendung im openETCS Projekt. Im Weiteren erfolgt eine Erläuterung der Methoden, die auf Basis des ausführlichen Entscheidungsverfahrens aus den Beschreibungsmitteln, Methoden und Werkzeugen gemeinsam von allen Arbeitspaketen ausgewählt wurden. Dabei musste jedoch eine Vielzahl an Details über die Zeit ergänzt und überarbeitet werden, da erst mit Erfahrungen aus der Anwendung weitere Details spezifiziert werden konnten.

Generell werden im Dokument D2.4 die vier folgenden Schritte definiert für die in openETCS formale Ansätze herangezogen:

- Systemanalyse,
- Architekturmodellierung,
- Funktions- und Verhaltensmodellierung,
- Ausführbare Software.

Für diese Aktivitäten stellen die openETCS Datenmodelle die Basis der Arbeit dar, um die Entwicklungsartefakte zu repräsentieren und auszutauschen. Dazu wurden Benennungsregeln für Anforderungen, Typen, Variablen, Funktionen und Eigenschaften festgelegt. Außerdem wurde eine Methodik definiert, um die in ERA Subset-026 definierten ETCS Funktions- und Kommunikationsstrukturen in die formalen Modelle zu überführen und so in der Entwicklung durchgängig und konsistent nutzbar zu machen.

Für die Modellierung wurde im openETCS Projekt das Beschreibungsmittel SysML in Verbindung mit dem Werkzeug Papyrus für die Darstellung der übergeordneten Systemzusammenhänge genutzt. Dabei werden die textuellen ETCS Spezifikationen in einer strukturierten Vorgehensweise in Blockdefinitionsdiagramme, Paketdiagramm, Interne Blockdiagramme, Anforderungsdiagramme und Sequenzdiagramme überführt. So wurde das System analysiert und daraus ein Designentwurf abgeleitet. Basierend auf der dabei erstellten funktionalen Systemarchitektur konnten die Sicherheitsanalysen durchgeführt und für die Verifikation und Validierung zugehörige Testfälle in Sequenzdiagrammen erstellt werden.

Darauf aufbauend wurde von den Projektpartnern entschieden, dass die openETCS Entwicklungsmethodik eine Kombination von SysML und SCADE nutzt, um die Architektur der openETCS Fahrzeugeinheit zu entwerfen. Im Weiteren wurden synchrone Zustandsmaschinen und Datenflüsse genutzt, um in SCADE Funktionen und Verhalten in den unteren Abstraktionsebenen zu spezifizieren. Die Programmiersprache C wurde als Implementierungssprache für den ausführbaren Code gewählt, welcher unter Nutzung der Scade Code Generatoren aus den Modellen erzeugt wurde. Diese Methoden decken die CENELEC Anforderungen zum Erstellen einer SIL4 Software ab. Eine alternative Nutzung von Open Source Werkzeugen konnte in AP7 aufgrund der fehlenden Leistungsfähigkeit der verfügbaren Werkzeuge nicht ermöglicht werden, allerdings sind die dargestellten Methodiken und die verwendeten Datenformate offen, so dass eine zukünftige Anwendung von Open Source Werkzeugen möglich ist.

Parallel zur Definition des Entwicklungsprozesses und der anzuwendenden Methodiken wurden potentielle Werkzeug- und Methodenkombinationen, die in den Stand der Technik Analysen identifiziert wurden, auf Teile der ETCS Spezifikationen angewendet. Dabei wurde deren Eignung für den openETCS Entwicklungsprozess genauer evaluiert. Um dabei die verschiedenen Ansätze vergleichen zu können, wurde dafür in D2.5 [24] eine Gruppe an Anforderungen festgeschrieben. Der eigentliche Vergleich und dessen Auswertung wurde darauf aufbauend federführend von AP7 durchgeführt.

Als Basis des Vergleichs wurden aus den Subset-026 ETCS Systemspezifikationen einzelne Bereiche ausgewählt, welche die unterschiedlichen Aspekte des Systems repräsentieren. Diese wurden mit Prioritäten versehen und deren Formalisierbarkeit durch die verschiedenen Beschreibungsmittel, Methoden und Werkzeuge verglichen. Darüber hinaus wurden für die unterschiedlichen Bereiche Sicherheitsanforderungen auf Basis des Subset-091 [12] bestimmt, deren Sicherheitsnachweis mit geeigneten Methoden auf Basis der erstellten Modelle realisiert wurde. Diese Grundlage erlaubte in der Anfangsphase des Projekts eine gezielte Analyse der Formalisierungen von ETCS und darauf aufbauend eine Schrittweise Abstimmung von Methoden und Werkzeugen zwischen den Arbeitspaketen.

6.2.3 Anforderungen

Resultierend aus den CENELEC Standard Anforderungen und dem definierten openETCS Entwicklungsprozess wurde als abschließendes Ergebnis des AP2 eine Sammlung der Meta-Anforderungen für die grundlegenden Aktivitäten im Projekt erstellt. Dieses umfasst Prozess- und

Werkzeuganforderungen für Anforderungsmanagement, Modellierung, Werkzeuge und deren Verifikation und Validierung. Zur besseren Übersichtlichkeit wurden die ursprünglichen vier Dokumente D2.6 bis D 2.9 [1] von den AP2 Partnern zu einem Dokument mit Anforderungen für openETCS in den folgenden Bereichen zusammengefasst:

- Ausführbares Modell und Programmierschnittstelle (API),
- System und Architektur,
- Modelle,
- Sicherheit,
- Verifikation und Validierung,
- Sprachen und Formalisierung,
- Werkzeugkette,
- Demonstrator.

Damit bilden diese ergänzende Anforderungen für die SIL 4 Softwareentwicklung neben den umzusetzenden System- und Sicherheitsanforderungen aus den ETCS Spezifikationen.

Als detaillierte Erweiterung dieses Anforderungsdokuments wurde ein technischer Anhang zu den Anforderungen an die API [4] erarbeitet, welcher auf Basis der bereits produktiv in Fahrzeugen eingesetzten Systeme detaillierte technische Anforderungen für die herstellerunabhängige openETCS Schnittstelle definiert. Diese API Anforderungen beinhalten die betrieblichen und funktionalen Anforderungen an die Schnittstelle zwischen der openETCS Anwendungssoftware und der Umgebung, vorwiegend der Basissoftware, einschließlich der folgenden Aspekte:

- Eingabe-Routinen die von der openETCS Anwendung bereitgestellt werden müssen,
- Ein- und Ausgangsdatenflüsse,
- Funktionen der Basissoftware, die von der openETCS Anwendung aufgerufen werden müssen, sowie
- Bedingungen, die eingehalten werden müssen, um die Anwendungssoftware zu entwickeln und zu nutzen.

Damit dienten diese API Anforderungen als ergänzende Schnittstellenanforderungen zu den in den ETCS Systemspezifikationen vorgeschriebenen Schnittstellen und vervollständigten damit die Basis für das Design der openETCS Anwendungsschnittstellen in den nachfolgenden Entwicklungsphasen.

Im Allgemeinen wurden somit in AP2 die Vorbetrachtungen für die *Open Proof* Entwicklung durchgeführt und daraus ein anzuwendender openETCS Entwicklungsprozess mit geeigneten formalen Methoden definiert. Die daraus resultierenden Prozessanforderungen wurden in entsprechende Meta-Anforderungen für die unterschiedlichen Bereiche des Projekts konkretisiert. Dieser Prozess wurde in der Entwicklungsarbeit der anderen APs umgesetzt. Darüber hinaus wurden durch die Partner in AP2 für die nicht in den ETCS Spezifikationen abgedeckten Schnittstellen anzuwendende technische Anforderungen aufbereitet.

6.3 Arbeitspaket 7 – openETCS Tool Chain

Das Arbeitspaket 7 hatte zum Ziel, den anderen Arbeitspakete eine auf Open Source basierende Werkzeuglandschaft für die Modellierung zur Verfügung zu stellen.

Kurz nach Beginn des Projekts wurde das Arbeitspaket 3 formell in die beiden Teilbereiche „Modellierung“ und „Toolentwicklung“ (neues Arbeitspaket 7) getrennt. Damit wurde den unterschiedlichen Abläufen, sowie den wenig überlappenden mitwirkenden Parteien Rechnung getragen. Daher wurde als erstes Aufgabe die Arbeitspaketbeschreibung des neuen Arbeitspaket 7 erstellt. Das neue Arbeitspaket bestand aus vier Teilaufgaben:

Entscheidungsfindung Technologien (T7.1 und T7.2)

Zunächst sollte die beste, passende Technologie für die Modellierung identifiziert werden. Aus Zeitgründen wurde diese Aufgabe in zwei Aufgaben geteilt: Zunächst sollte das Werkzeug für die Kernaufgaben (Modellierung und V&V) identifiziert werden. Im zweiten Schritt sollten die Werkzeuge für die unterstützenden Aufgaben (Versionierung, Traceability, etc.) identifiziert werden. Das Ergebnis dieser Analyse ist in [20] und [32] dokumentiert.

Werkzeugentwicklung (T7.3)

Basierend auf den Anforderungen von Arbeitspaket 2 sollte aus den in T7.1 und T7.2 ausgewählten Technologien eine Werkzeugplattform entwickelt werden. Das Werkzeug wurde nach Scrum entwickelt und dementsprechend kontinuierlich während der Projektlaufzeit aktualisiert. Zum ersten Release des Werkzeugs wurde das Deliverable D7.4 veröffentlicht [5]. Die aktuelle Version des Werkzeugs kann von der openETCS-Webseite heruntergeladen werden [30].

Aufbau und Pflege eines Open Source Ökosystems (T7.4)

Begleitend sollte ein Open Source Ökosystem aufgebaut und den Anforderungen der Nutzer entsprechend weiterentwickelt werden. Das Ergebnis ist primär das bei GitHub gehosteten Repositories. Die Ergebnisse werden in Kapitel 6.3.4 detailliert vorgestellt und in [5] dokumentiert.

6.3.1 openETCS Werkzeugplattform

Die Technologien für die openETCS Werkzeugplattform wurden nach einer ausführlichen Analyse (Kapitel 6.3.3.1), basierend auf den Anforderungen von Arbeitspaket 2 (Kapitel 6.2) ausgewählt. Identifiziert wurde Eclipse als primäre Plattform (Kapitel 6.3.1.1), als auch eine Anzahl von bestehenden Plug-Ins, die im Folgenden vorgestellt werden.

Es wurden auch mögliche Alternativen identifiziert (Kapitel 6.3.3.1). Dabei ist insbesondere SCA-DE relevant, welches später auch für Teile der Modellierung eingesetzt wurde (Kapitel 6.3.1.11).

6.3.1.1 Eclipse als Plattform

Eclipse ist eine modulare „Rich Client Platform“. Eine typische Eclipse-Anwendung besteht aus hunderten von Plug-Ins. Daher musste zunächst eine Basisplattform ausgewählt werden. Dafür wurde die damals aktuelle Version „Kepler“ ausgewählt. Als Basiskonfiguration wurden die minimalen für Papyrus (Kapitel 6.3.1.2) erforderlichen Plug-Ins ausgewählt. Das automatische Bauen und Testen der Software (Continuous Integration, siehe Kapitel 6.3.1.8) wurde von Anfang an umgesetzt.

Anfang 2015 wurde die Software von Kepler zur nächsten Eclipse-Version, „Luna“, migriert (Kapitel 6.3.4.3).

Abbildung 7 zeigt das openETCS-Werkzeug mit einem Papyrus-SysML-Modell.

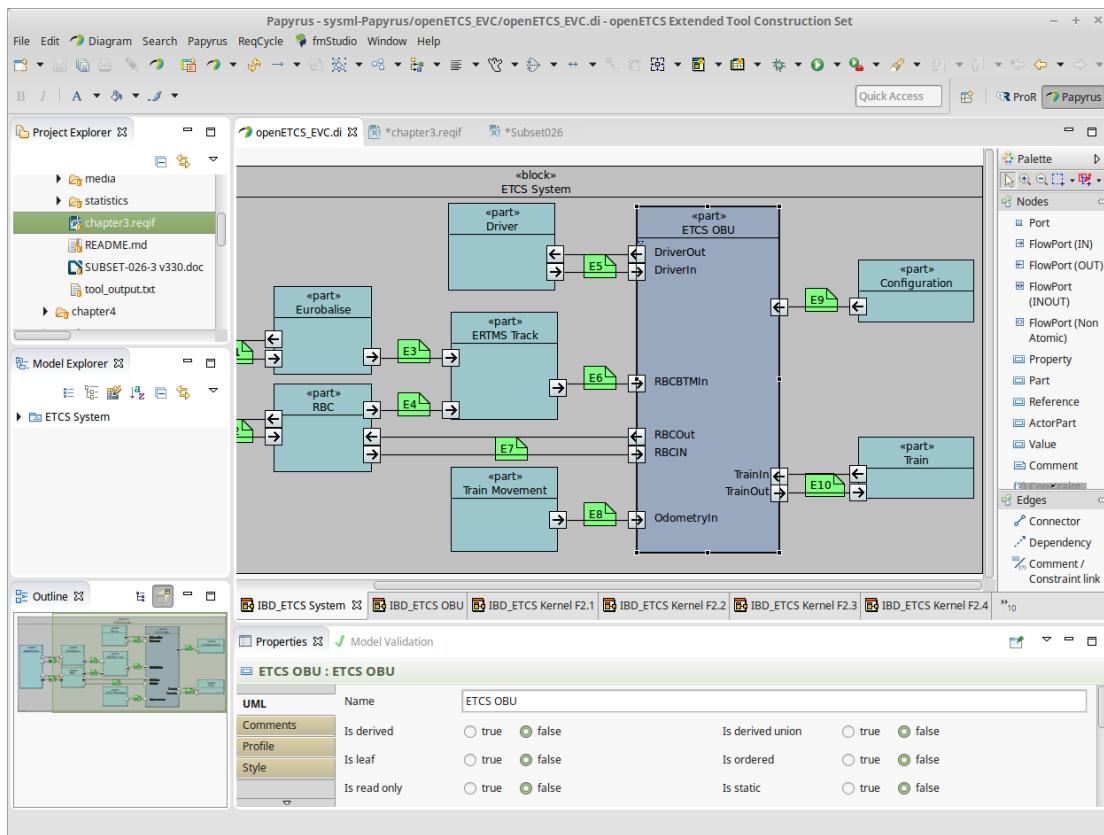


Abbildung 7. openETCS mit dem Papyrus-Modell der openETCS OBU.

6.3.1.2 Papyrus Integration

Papyrus ist die Eclipsekomponente für UML/SysML-Modellierung und damit das Kernstück für die Arbeiten in den Arbeitspaketen 3 und 4. In Abbildung 10 sind zwei Papyrus-Diagramme im Werkzeug zu sehen.

Papyrus war sofort für die Arbeiten der anderen Arbeitspakte einsatzbereit. Während des Einsatzes wurden kontinuierlich Anpassungen an die Bedürfnisse von Arbeitspaket 3 und 4 durchgeführt, bspw. die Anpassung des Erscheinungsbilds (Style Sheets). Während des Einsatzes wurden auch schwere Defizite identifiziert, die seitens Arbeitspaket 7 so gut wie möglich behoben wurden, wie bspw. die Performanz. Allerdings zeichnete sich schon bald ab, dass mit den verfügbaren Ressourcen nur begrenzt Einfluss genommen werden konnte.

Als Lösung dieser Probleme mit Papyrus wurde eine Migration zur Version „Luna“ von Eclipse vorgeschlagen und im ersten Quartal 2015 durchgeführt (Kapitel 6.3.4.3).

6.3.1.3 Requirements Modeling Framework Integration

Die Ausgangsbasis für die Modellierung waren textuelle Anforderungen, die in der Form von Word-Dokumenten vorlagen (ERA Subset-026). Diese sollten in einem offenen Format in das Eclipse Ökosystem überführt werden. Dazu wurde das Eclipse Requirements Modeling Framework (RMF) ausgewählt. Dieses ermöglicht das Arbeiten mit Anforderungen im ReqIF-Format. Damit war eine wichtige Anforderung an „Open Proofs“ erfüllt (offenes Werkzeug und offener Standard). Für die Überführung von Word nach ReqIF wurde ein eigenes Werkzeug entwickelt (vgl. Kapitel 6.3.1.9). Abbildung 8 zeigt einen Teil des konvertierten Subset-026 im RMF-Editor.

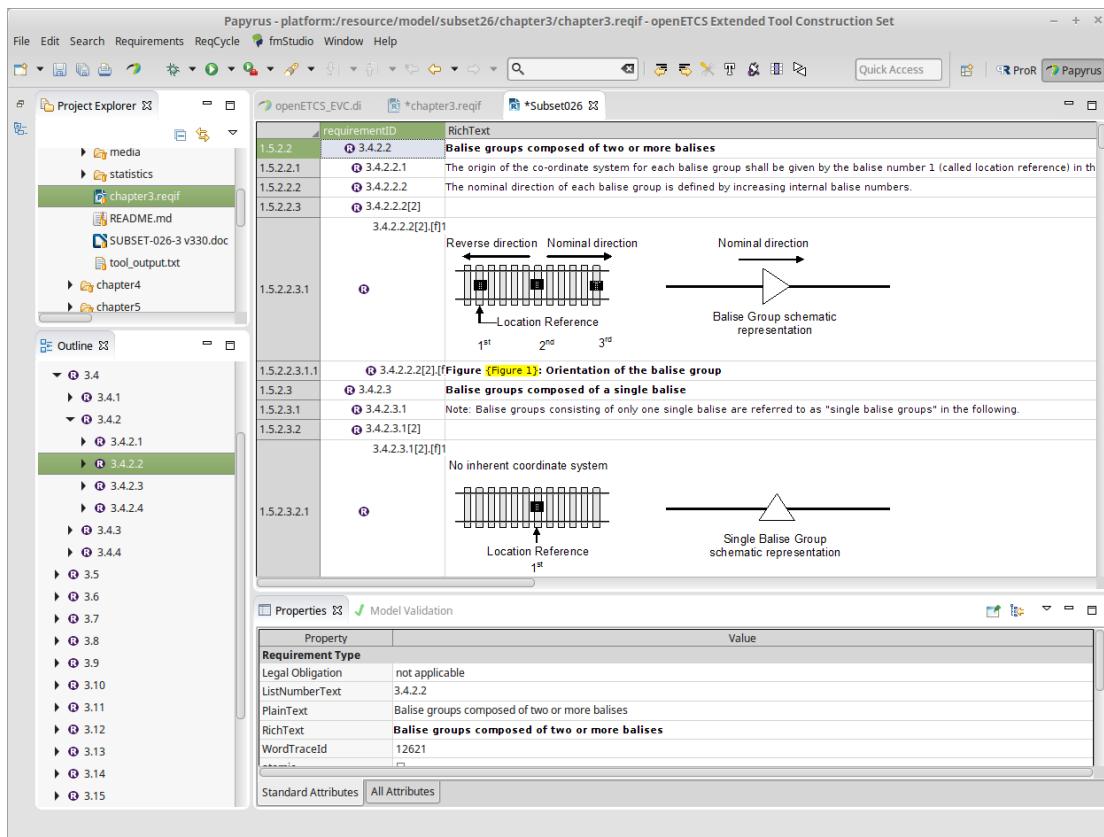


Abbildung 8. openETCS mit Subset-026, im nach ReqIF konvertierten Format.

RMF erfüllte schon zu Beginn des Projekts einen großen Teil der Anforderungen des Projekts. Während der Laufzeit des Projekts wurde RMF weiterentwickelt, um die Bedürfnisse der anderen Arbeitspakete optimal abdecken zu können. Dazu gehört eine erweiterte Suchfunktion, Validierung der Anforderungen und eine Prototypische Nachverfolgbarkeit zu Papyrus mit Änderungsmanagement (siehe Kapitel 6.3.2.1).

6.3.1.4 Eclipse Safety Framework Integration

Das Eclipse Safety Framework (ESF) ermöglicht die Fehleranalyse von Papyrus-Modellen mit Fault Trees und anderen Techniken. ESF wurde problemlos in die Werkzeugplatform integriert und ist in Abbildung 9 zu sehen.

6.3.1.5 RT-Tester Integration

RT-Tester ist ein serverbasiertes System für die Testautomatisierung, für das eine Eclipse-Integration entwickelt wurde. Da die eigentliche Testautomatisierung nach wie vor auf den Servern des Anbieters (Verified Systems International) durchgeführt wird, wurden die Anforderungen von Open Proofs nur zum Teil erfüllt. Das Werkzeug ist in Abbildung 10 zu sehen.

6.3.1.6 ReqCycle Integration

ReqCycle ist eine Eclipse-Komponente für Nachverfolgbarkeit und Teil des Eclipse PolarSys-Projekts. Diese Komponente wurde ebenfalls in die Werkzeugplattform integriert. Da dies relativ spät geschah (3. Quartal 2015), wurde diese Komponente nicht von den anderen Arbeitspaketen produktiv eingesetzt. Das Werkzeug ist in Abbildung 11 zu sehen.

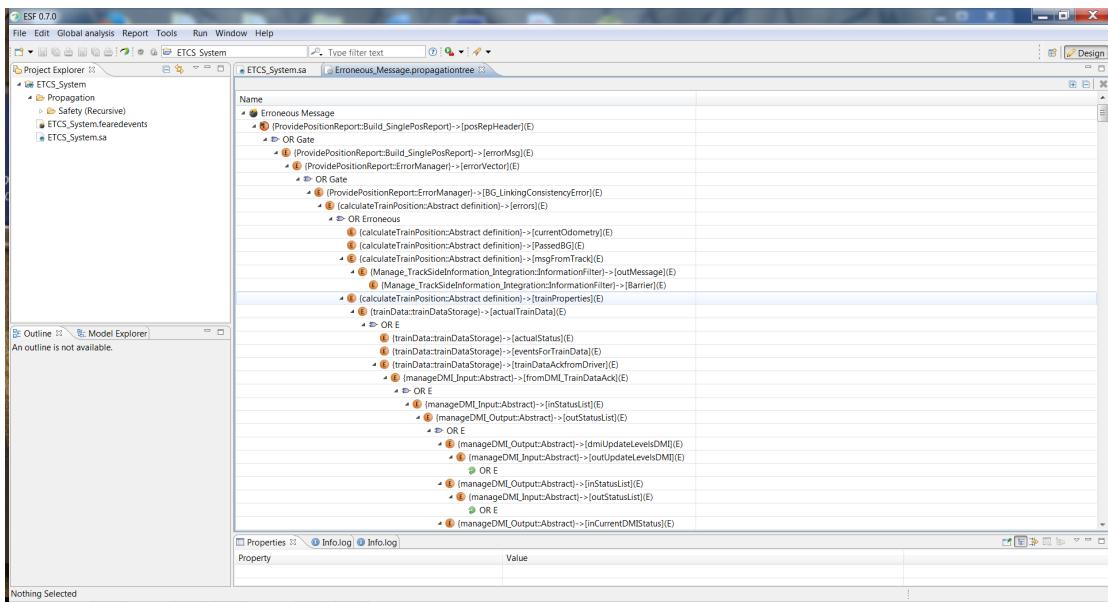


Abbildung 9. Ein Fehlerbaum im Eclipse Safety Framework (ESF), welches in openETCS integriert wurde.

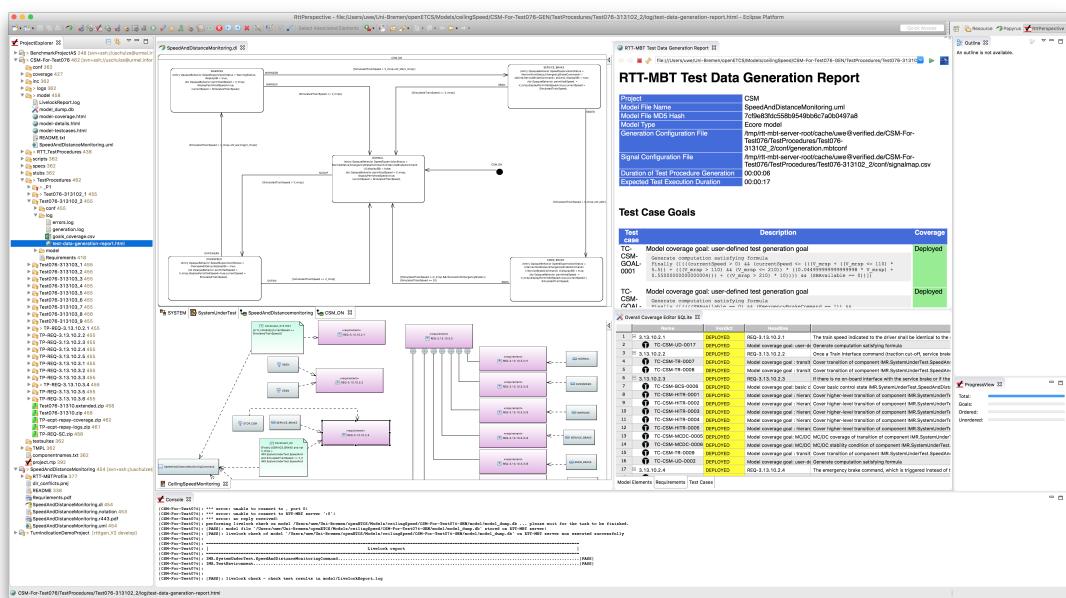


Abbildung 10. Das openETCS-Werkzeug mit zwei Papyrus-Diagrammen (links) und RT-Tester-Ergebnissen (rechts).

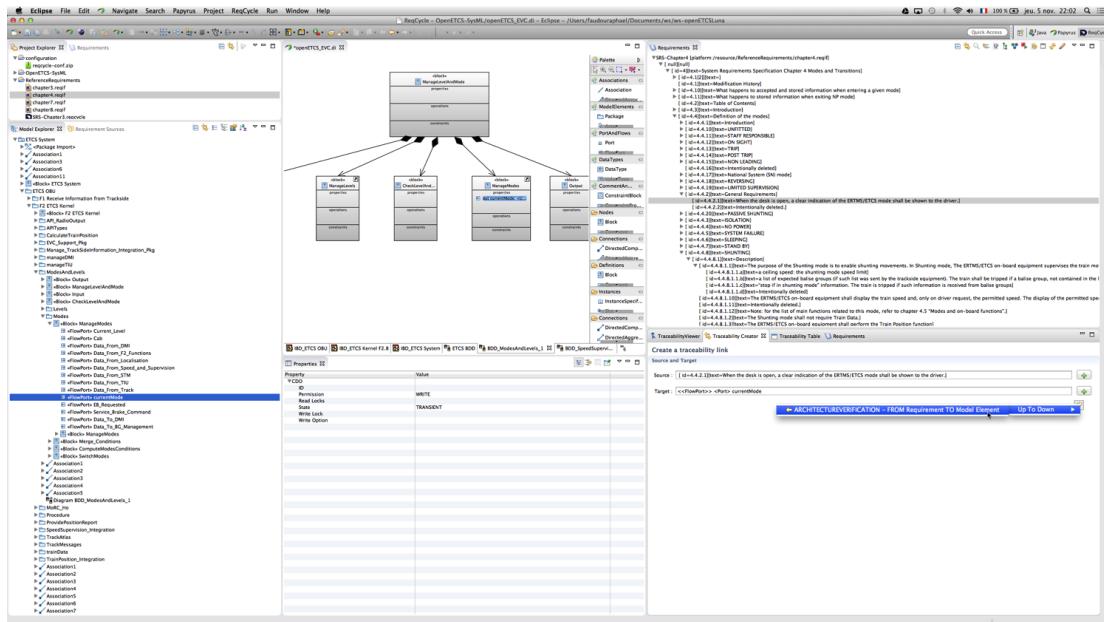


Abbildung 11. Die Traceability des Modells in ReqCycle. Rechts oben ist der Baum der Anforderungen zu sehen, rechts unten die Verlinkung zum SysML FlowPort.

6.3.1.7 Dokumentations-Infrastruktur

Eclipse hat ein eigenes, kontextsensitives Hilfesystem, welches für die Dokumentation genutzt wurde (siehe Abbildung 15). Dadurch kann die openETCS-Dokumentation direkt aus dem Werkzeug heraus über das Hilfe-Menü aufgerufen werden.

Die Inhalte wurden zentral erstellt. Neben der Eclipse-Hilfe gibt es noch andere Wege, um auf die Dokumentation zuzugreifen (siehe Kapitel 6.3.3.2).

6.3.1.8 Continuous Integration

Um die Reproduzierbarkeit zu gewährleisten, wurde die Software über ein kontinuierliches Buildsystem (Continuous Integration) kompiliert, konfiguriert und getestet. Dazu wird ein Server benötigt, wobei auf den kostenlosen Online-Dienst „Cloudbees“ zurückgegriffen wurde.

6.3.1.9 Word-nach-ReqIF Konvertierung

Für die Verarbeitung der relevanten Teile der ETCS-Spezifikation (ERA Subset-026) in Eclipse musste eine Konvertierung von Microsoft Word nach ReqIF vorgenommen werden. Dazu wurde ein Konverter erstellt. Die resultierenden Dateien wurden für die Verarbeitung in Eclipse RMF optimiert, können jedoch mit jedem ReqIF-kompatiblem Werkzeug verarbeitet werden.

6.3.1.10 Bitwalker Data Dictionary

Teile der ETCS Spezifikation (ERA Subset-026, Kapitel 7 und 8) konnten direkt nach XML konvertiert werden, wodurch die entsprechenden Datenstrukturen direkt verfügbar waren („Data Dictionary“). Dies wurde mit dem Werkzeug Bitwalker in XML umgewandelt und konnte von dort weiterverwendet werden, sowohl in Papyrus als auch in SCADE. Bitwalker ist ein von Hand in Python geschriebenes Werkzeug.

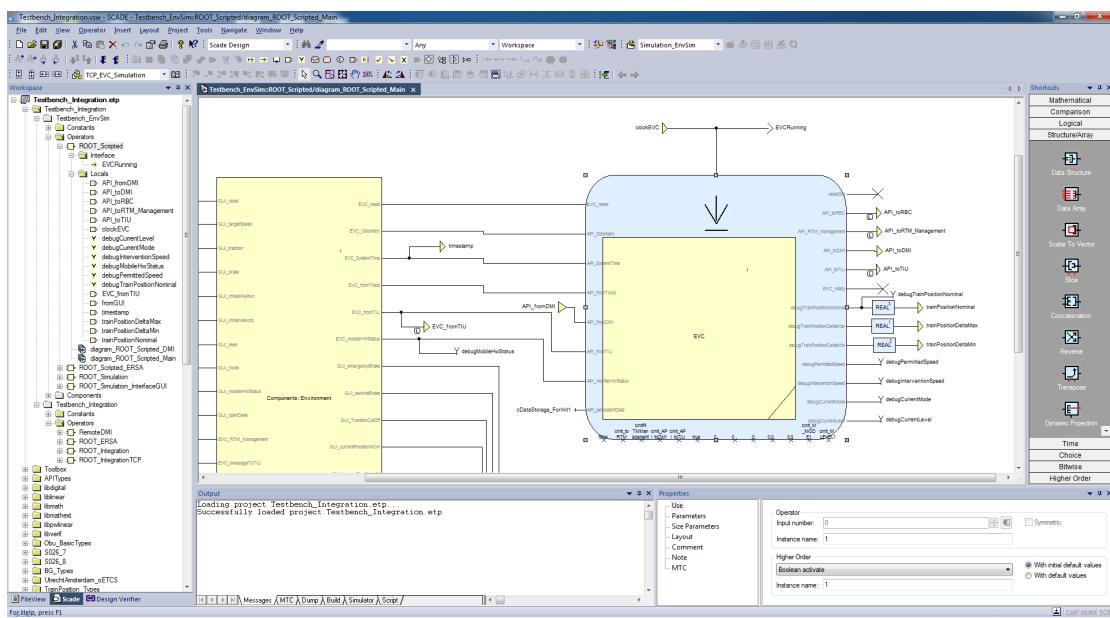


Abbildung 12. Das openETCS-Modell in SCADE Suite.

6.3.1.11 SCADE Suite und SCADE Systems

Die Arbeiten in Arbeitspaket 7 haben sich auf quelloffene Software beschränkt. Dementsprechend wurde keine Entwicklung an SCADE Suite oder SCADE System betrieben. Dennoch wurde die Interoperabilität insbesondere mit SCADE System aktiv unterstützt. SCADE Systems basiert auf Eclipse Papyrus, wodurch Modelle zwischen den beiden Werkzeugen ausgetauscht werden können. Dies wurde durch Tailoring und entsprechende Dokumentation sichergestellt. Weiterhin wurde prototypisch ein Validator entwickelt der prüfen kann, ob nur erlaubte Strukturen im Modell verwendet wurden (Kapitel 6.3.2.2). In Abbildung 12 ist ein Screenshot von SCADE Suite sehen.

6.3.2 Prototypen

Neben den in Kapitel 6.3.1 beschriebenen Teilen der Werkzeugplattform wurden eine Reihe von Prototypen entwickelt. Diese sind nicht in das Eclipse-basierte Werkzeug integriert. Der Quellcode ist jedoch im Projektrepository vorhanden.

6.3.2.1 RMF-SysML-Traceability

Ein wichtiges Thema für V&V-Aktivitäten sowie Änderungsmanagement ist die Nachverfolgbarkeit von Artefakten. Später wurde für diesen Zweck die Komponente ReqCycle integriert (Kapitel 6.3.1.6). Diese stand jedoch bei Beginn der Entwicklung noch nicht zur Verfügung. Daher wurde mit einer Traceability zwischen Anforderungen und SysML als „Proof of Concept“ experimentiert.

Das resultierende Plug-In ermöglicht eine Verlinkung zwischen Anforderungen und beliebigen SysML-Elementen per Drag & Drop. Weiterhin werden Änderungen von Elementen über ein entsprechendes Flag markiert.

Das Plug-In war für die anderen Arbeitspakte nützlich, um unterschiedliche Ansätze zur Traceability zu evaluieren. Es wurde nicht mehr weiterentwickelt, nachdem ReqCycle als weiter verbreitete Lösung in die Werkzeugplattform integriert wurde.

6.3.2.2 SysML Constraint Checker

Die gewählte Modellierungssprache SysML ist sehr umfangreich und es wurde frühzeitig beschlossen, nur einen Teil der Sprache zu nutzen. Das war auch notwendig, um die Interoperabilität mit SCADE System zu gewährleisten (Kapitel 6.3.1.11). Die Einschränkungen der Sprache wurden zunächst dokumentiert, allerdings war die automatische Prüfung gewünscht. Dies wurde prototypisch mit dem Eclipse Validation Framework umgesetzt. Diese Komponente wurde nicht in die Werkzeugkette übernommen, da die Ressourcen fehlten, die entsprechenden Validierungsregeln umzusetzen.

6.3.2.3 SysML nach SCADE Konvertierung

Nachdem klar war, dass das kommerzielle Werkzeug SCADE Suite von den Modellierern genutzt werden würde, wurde ein Prototyp entwickelt, der SysML automatisch in ein SCADE-Modell konvertierte. Dieser Konverter war aber nur ein „Proof of Concept“, der nicht alle Sprachfeatures der SysML abdeckte. Da dieser Ansatz nicht die Projektziele unterstützte, insbesondere „Open Proofs“, wurden diese Arbeiten eingestellt.

6.3.2.4 SysML nach B Konvertierung

Im Rahmen der Entscheidungsfindung [20] wurde auch die Modellierungssprache B in Erwägung gezogen. B wurde ebenfalls in Erwägung gezogen, um ein SysML-Modell weiter zu verfeinern. Dazu wurde ein prototypischer Konverter entwickelt. Nachdem die Entscheidung auf SCADE Suite für die Verfeinerung gefallen war, gab es keine Notwendigkeit mehr, diesen Konverter weiterzuentwickeln.

6.3.2.5 SysML nach C/Frama-C Konvertierung

Ein Ansatz für die Code-Generierung wäre die Konvertierung von SysML nach C (bspw. Frama-C) gewesen. Diese Technologien wurden im Rahmen der Entscheidungsfindung analysiert [32] und dementsprechend weiter untersucht. Allerdings wäre für diesen Ansatz ein sehr detailliertes SysML-Modell notwendig gewesen, welches mit Papyrus nicht realisiert werden konnte, da es nicht entsprechend skalierte. Auch hier wurden Arbeiten beendet, nachdem die Entscheidung für SCADE Suite gefallen war.

6.3.2.6 SysML nach System-C

System C ist eine weitere Sprache, die analysiert wurde [20] und für System Design und Codegenerierung eingesetzt werden kann. Ähnlich wie bei Frama-C wurde ein Prototyp entwickelt und nach der Entscheidung für SCADE Suite verworfen.

6.3.3 Studien und Konzepte

Neben den Deliverables wurden eine Reihe von Studien und Konzepten entwickelt, welche im Folgenden vorgestellt werden.

6.3.3.1 Primary und Secondary Tool Chain Analysis

Zu Beginn des Projekts musste eine Technologie ausgewählt werden, die für die Werkzeugkette eingesetzt werden sollte. Um den Entscheidungsprozess zu beschleunigen wurden die Werkzeuge in zwei Gruppen geteilt, Primary und Secondary Toolchain.

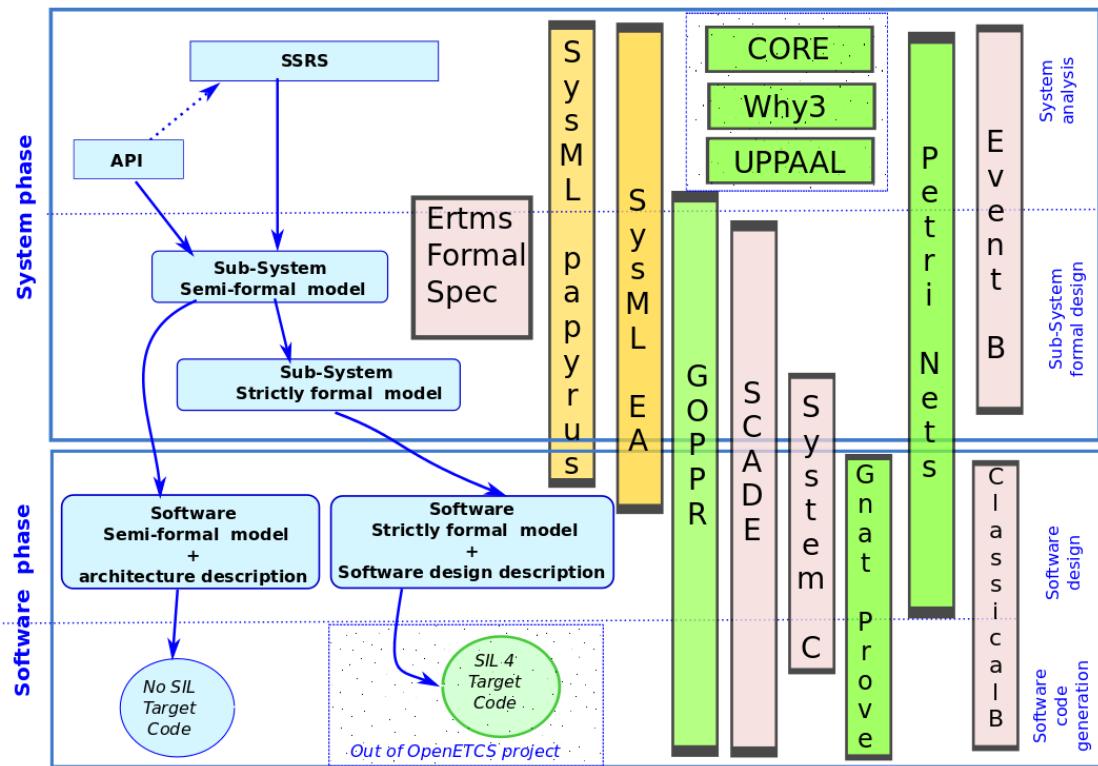


Abbildung 13. Alternativen für die openETCS Werkzeugkette (aus [20]).

Die „Primary Toolchain“ beinhaltet Werkzeuge, die für die Modellierungsarbeiten essenziell sind, insbesondere die Modellierungssprache selbst, sowie das Werkzeug, mit dem die Modelle bearbeitet werden können.

Die „Secondary Toolchain“ beinhaltet Werkzeuge, die eine Unterstützende Funktion haben (bspw. Versionierung) oder die nicht ausgewählt werden können, bevor die Modellierungssprache nicht bekannt ist (bspw. V&V).

Der Entscheidungsprozess war nicht leicht, denn keines der möglichen Werkzeuge erfüllte alle Anforderungen. Weiterhin gab es auch kein Werkzeug, das die gesamte Entwicklung abdecken würde, wie in Abbildung 13 zu sehen ist.

Zur Minimierung des Risikos, dass die gewählte Werkzeugkette nicht funktioniert, wurden drei Wege parallel verfolgt, wie in Abbildung 14 zu sehen: die linken drei Spalten repräsentieren (1) komplett Open Source, (2) SCADe und (3) ERTMS Formal Specs als drei Wege der Umsetzung, mit möglichen Migrationswegen zwischen den einzelnen Entwicklungsphasen. Dabei wurde der erste Weg primär von Arbeitspaket 7 verfolgt, da hier ja auch Entwicklungsarbeit notwendig war. Der zweite Weg wurde von Arbeitspaket 3 verfolgt, da dies den sofortigen Beginn der Arbeiten ermöglichte, und der dritte Weg von Arbeitspaket 4, die über die V&V-Aktivitäten auf dem für das Werkzeug bereits bestehenden Modell aufsetzen konnten. Dieser Ansatz hat sich bewährt und ermöglichte es, zu einem soliden Ergebnis zu kommen, wie in Kapitel 6.4 und Kapitel 6.5 beschrieben.

6.3.3.2 Handbuch

Für die Erstellung der Dokumentation wurde die das von GitHub bereitgestellte Wiki genutzt (siehe auch Kapitel 6.3.4.1). Dies ermöglichte allen Projektteilnehmern, durch das direkte bearbei-

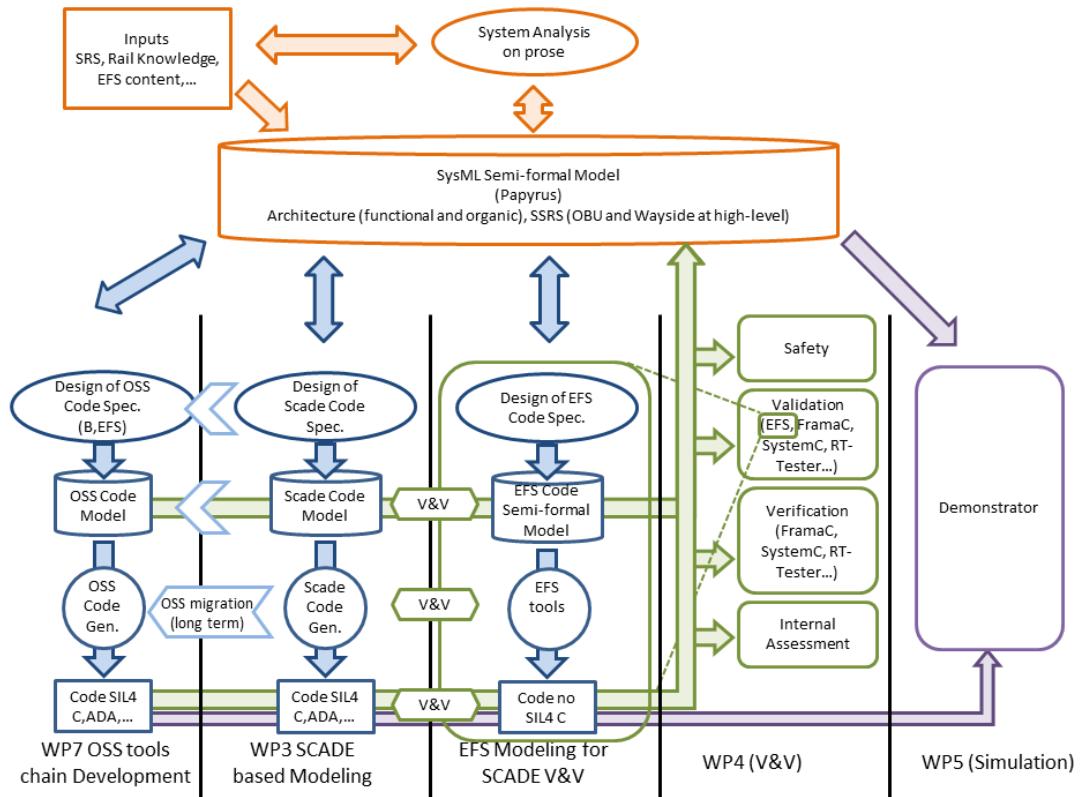


Abbildung 14. Risikominimierung durch die Verfolgung von drei verschiedenen Werkzeugansätzen (aus [20]).

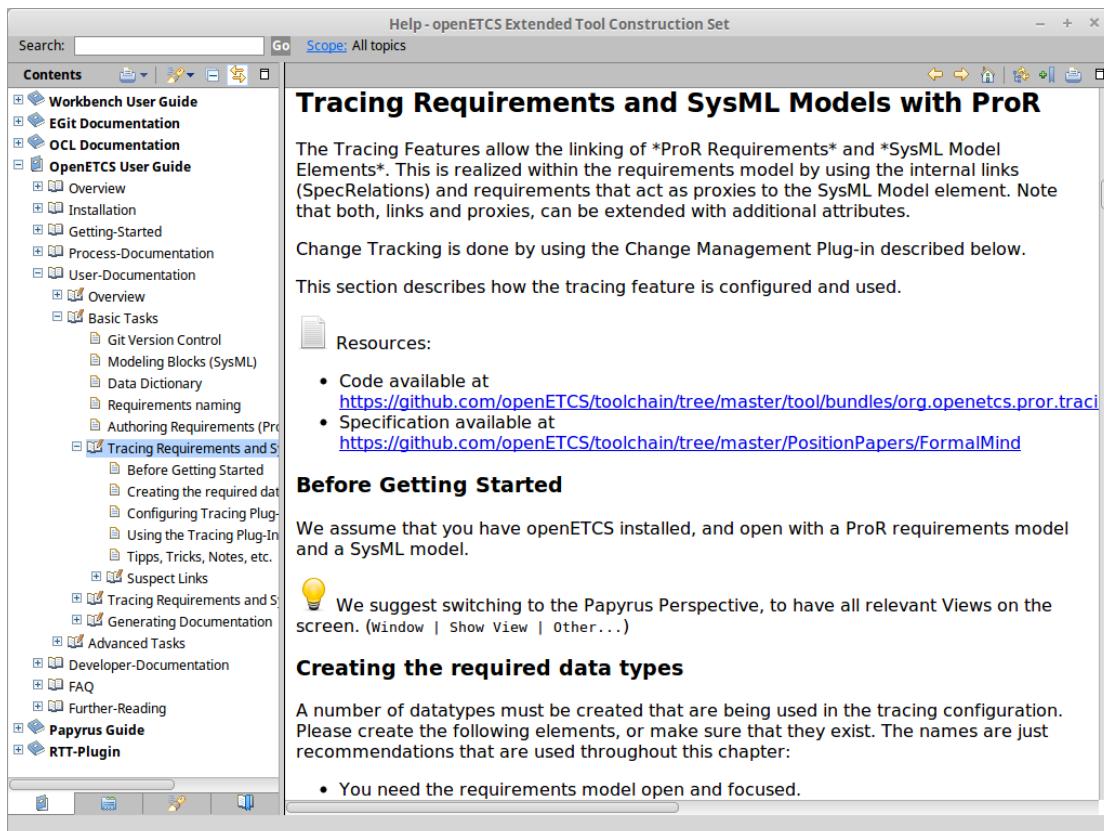


Abbildung 15. Das Eclipse-Hilfesystem mit einer Seite der openETCS-Dokumentation.

ten der Wikiseiten Inhalte hinzuzufügen oder zu korrigieren. Über das Wiki war das Handbuch auch sofort öffentlich im Web zugänglich. Das Handbuch besteht aus den folgenden Teilen:

- Overview
- Installation
- Getting Started
- Process Documentation
- User Documentation
- Developer Documentation
- FAQ
- Further Reading

Wie in Kapitel 6.3.1.7 erwähnt, wurden die Wikiseiten des Handbuchs auch vom CI-System verarbeitet. Dadurch wurden die selben Inhalte direkt im Werkzeug über Eclipse Help zugänglich gemacht (auch offline, wie in Abbildung 15 zu sehen).

Zuletzt wurden die Wikiseiten ein drittes Mal verarbeitet, um daraus ein PDF zu generieren. damit ist das Handbuch als eine einzige PDF-Datei verfügbar. Da diese auch vom CI-System generiert wird, ist sie entsprechend aktuell.

6.3.3.3 Traceability Architecture Roadmap

Wie in Kapitel 6.3.1.6 erwähnt, konnte ReqCycle erst sehr spät in die Werkzeugkette integriert werden. Dadurch konnte ReqCycle auch von den anderen Arbeitspaketen nicht genutzt werden. Da Nachverfolgbarkeit ein zentraler Aspekt für eine Systementwicklung ist, haben wir eine Roadmap für eine durchgängige Nachverfolgbarkeit entwickelt. Ein etwaiges Nachfolgeprojekt kann direkt auf dieser Arbeit aufsetzen.

6.3.3.4 Migration zu einer offenen SCADE-Alternative

Die Ziele bezüglich „Open Proofs“ wurden nicht vollständig erreicht, da mit SCADE Suite ein nicht-offenes Werkzeug in einer zentralen Rolle in die Werkzeugkette aufgenommen wurde. Für ein mögliches Nachfolgeprojekt wäre ein wichtiges Ziel, das Modell zu einem offenen Werkzeug, oder zumindest einer offenen Modellierungssprache zu migrieren. Dazu wurde ein Konzept erstellt, dass die Optionen analysiert und eine Empfehlung ausspricht [38].

6.3.3.5 Testplan

Da die Werkzeugkette für die Entwicklung sicherheitskritischer Software eingesetzt werden soll, muss sie nach EN50128 getestet werden. Wie dies erreicht werden kann ist im Testplan [23] dokumentiert. Der Testplan stellt entsprechende Templates zur Verfügung. Die Durchführung der Tests selbst ist nicht im Arbeitsbereich des Projektes. Diese müssen von Unternehmen durchgeführt werden, die die Ergebnisse von openETCS weiterverwerten.

6.3.3.6 Qualification-Plan

Ähnlich wie der Testplan (Kapitel 6.3.3.5) wurde eine Prozessbeschreibung für eine Werkzeugqualifikation erstellt [7], die für eine Konformität mit EN50128 erforderlich ist. Auch hier wird erwartet, dass diese von verwertenden Unternehmen durchgeführt wird.

6.3.4 Dienstleistungen

Im Rahmen von Arbeitspaket 7 wurden weiterhin eine Anzahl von Dienstleistungen durchgeführt, die im Folgenden beschrieben werden.

6.3.4.1 Ecosystem

Im Rahmen dieses Arbeitspakets wurde das Ökosystem aufgesetzt und den Anforderungen der Nutzer entsprechend konfiguriert. Diese Arbeiten begleiteten das Projekt von Anfang bis Ende und sind in einem eigenen Deliverable dokumentiert [19].

Schon frühzeitig wurde GitHub als Kommunikationsplattform identifiziert. Den Anforderungen von Arbeitspaket 1 entsprechend wurde diese genutzt, um mit öffentlichen und privaten Repositories Ablageorte für die Arbeitdateien der Projektteilnehmer zur Verfügung zu stellen. Große Teile der Arbeit wurden in von GitHub zur Verfügung gestellten Wikis durchgeführt. Hier unterstützte Arbeitspaket 7 mit Anpassungen und Templates. GitHub wurde ebenfalls für die offizielle Projektwebseite (openetcs.org) genutzt.

Für die Kommunikation wurden kostenlose Anwendungen von Google eingesetzt: Zum einen wurden die Mailinglisten mit Google Groups verwaltet. Zum anderen wurden alle Termine in Google Calendar eingetragen. Arbeitspaket 7 sorgte für die entsprechende Anpassung und Integration. Bspw. wurden die Termine aus dem Kalender automatisch auf der Webseite angezeigt.

Im Rahmen der Bedürfnisse der Projektteilnehmer wurden eine Anzahl von Erweiterungen des Ökosystems nach Bedarf vorgenommen. Bereits angesprochen wurde Continuous Integration (Kapitel 6.3.1.8).

Für das gemeinsame Arbeiten an Texten wurde Latex ausgewählt, ein Textsystem, das sich hervorragend für die Arbeit im Team eignet. Arbeitspaket 7 erstellte hierfür, den Anforderungen von Arbeitspaket 1 entsprechend, die Infrastruktur, einschließlich der Erstellung von Vorlagen, Durchführung von Schulungen und Werkzeugempfehlungen.

Relativ spät im Projekt wurde beschlossen, die Arbeit aller Arbeitspakete über ein „Scrum of Scrums“ zu koordinieren. Dazu wählte Arbeitspaket 7, ebenfalls den Projektanforderungen entsprechend, die Online-Anwendung „Waffle“ aus, und konfigurierte diese für den Einsatz als Backlog.

6.3.4.2 CI für die Modellierung

Sowohl das Modellierungsteam (Arbeitspaket 3) als auch das V&V-Team (Arbeitspaket 4) mussten sicherstellen, dass deren Artefakte bestimmte Qualitätsanforderungen erfüllen. Dazu haben diese Teams entsprechende Testskripte erstellt. Um diese automatisch bei Veränderungen im Repository auszuführen, wurde eine entsprechende Schnittstelle zum CI-System Jenkins geschaffen, welches bereits für das Werkzeug eingesetzt wurde (vgl. Kapitel 6.3.1.8).

6.3.4.3 Migration nach Luna

Als die Eclipse-basierte Werkzeugentwicklung begann, war Eclipse Kepler die aktuellste Version (siehe auch Kapitel 6.3.1.1). Neue Versionen von Eclipse werden im Jahresrhythmus herausgegeben, jedoch war eine Migration zu einer neueren Version nicht geplant, da Stabilität einen hohen Stellenwert hatte. Es stellte sich jedoch heraus, dass die Luna Version von Papyrus (Kapi-

tel 6.3.1.2) für die Modellierer essentielle Neuerungen mitbrachte. Daher wurde beschlossen, zu Eclipse Luna zu migrieren.

Diese Migration war nicht trivial und fand über mehrere Monate statt. Insbesondere konnten einmal nach Luna konvertierte SysML-Modelle nicht mehr von der Kepler-Version geöffnet werden. Daher wurde die neue Version ausführlich getestet, bevor endgültig migriert wurde. Durch die sorgfältige Vorbereitung konnte die Migration schließlich ohne Probleme durchgeführt werden.

6.3.5 ERTMS Formal Specs

Neben der Eclipse-basierten Werkzeugplattform wurde noch das Werkzeug „ERTMS Formal Specs“ weiterentwickelt worauf bereits in Kapitel 6.3.3.1 hingewiesen wurde. Für dieses Werkzeug wurde eine eigene domänenspezifische Sprache (DSL) entwickelt, die auf die ETCS-Spezifikation abgestimmt war.

Da die Sprache nicht strikt formal war, wurde das Werkzeug nicht für die Modellierung, also Arbeitspaket 3, empfohlen. Allerdings hatte es durchaus Potential für V&V-Aktivitäten (Arbeitspaket 4).

Letzten Endes wurde ERTMS Formal Specs nicht von Arbeitspaket 4 genutzt. Statt dessen wurde es weitgehendst unabhängig von openETCS weiterentwickelt. Die Ergebnisse sind in [10] dokumentiert.

6.4 Arbeitspaket 3 – Modellierung – Codegenerierung

Primäres Ziel von Arbeitspaket 3 war die Erstellung eines formalen Modells der ETCS OBU gemäß der Spezifikation in ERA Subset-026 [11], Version 3.3.0. Das formale Modell stellt zum einen eine Formalisierung der in natürlicher Sprache verfassten Spezifikation dar und erlaubt zum anderen mit Hilfe der Werkzeugkette die Generierung der zugehörigen Software. Weiterhin wurde eine generische openETCS Referenz Schnittstelle (Application Programming Interface, API) definiert.

Die Entwicklung bzw. Formalisierung erfolgte dabei Scrum-basiert in Sprints und führte zu vier Releases im Projektzeitraum. Für die Modellierung auf Systemebene wurde SCADE System/Papyrus eingesetzt und die funktionale Modellierung wurde mit SCADE Suite durchgeführt. Die Entscheidung mit SCADE Suite Closed Source Software für die Modellierung zu verwenden wurde getroffen nachdem sich im Rahmen der Evaluation verfügbarer Open Source Werkzeuge herausgestellt hatte, dass deren Reife zur Zeit noch nicht den Anforderungen des Projektes genügt (vgl. Kapitel 6.3). In den folgenden Abschnitten werden Aufbau und Struktur der openETCS OBU dargestellt (Kapitel 6.4.3 und 6.4.5). Kapitel 6.4.4 gibt einen Überblick über die openETCS API. Eine detaillierte Beschreibung der Ergebnisse des Arbeitspakets liegt mit den folgenden öffentlichen Projekt Deliverables vor:

- Das openETCS SysML Modell (SCADE System/Papyrus) [29].
- Das zugehörige funktionale Modell (SCADE Suite) der openETCS OBU [28].
- Die Dokumentation der System Architektur, D3.5.4 [22].
- Die Dokumentation des funktionalen Modells, D3.6.4 [16].
- Die Dokumentation der generischen openETCS API [17].

6.4.1 openETCS Traceability von Anforderungen

Die Notwendigkeit von Traceability (Rückverfolgbarkeit) resultiert im openETCS Projekt aus drei wesentlichen Projektzielen:

1. openETCS soll zeigen, wie CENELEC 50128 konforme Entwicklung mit Open Proof Verfahren und agiler Entwicklungsumgebung funktionieren kann.
2. Das openETCS Entwicklungsergebnis, das formale openETCS OBU Model, muss als SRS Referenz zu seinen Ursprüngen zurückweisen können.
3. Das openETCS Model muss im Rahmen der Weiterentwicklung der ETCS Standardisierung selbst rückverfolgbar weiterentwickelt werden.

Obwohl Traceability auch in der industriellen Entwicklung von Systemen eine zentrale Anforderung durch Kunden und Standards ist, gibt es doch nur wenige funktionierende Lösungen in komplexeren Entwicklungsprojekten. Aus den bekannten Open Source Projekten konnten daher keine bestehenden Lösungen übernommen werden. Erschwerend kommt hinzu, dass das ETCS System Requirements Spezifikation (ERA Subset-026) nicht in allen Bereichen hinreichend referenzierbar ist. Das Dokument enthält neben textuellen Komponenten auch grafische und tabellarische Anteile, die eine eindeutige Referenzierung von technischen Anforderungen in diesem Dokument nicht zulassen.

Entsprechend wurde Traceability als Thema in alle Teile des openETCS Projektes übernommen. In den folgenden Abschnitten [6.4.1.1](#) bis [6.4.1.4](#) wird auf die Umsetzung von Traceability in den relevanten Arbeitspaketen eingegangen.

6.4.1.1 Traceability im Arbeitspaket 2: Anforderungen

Die Anforderungen an Traceability im openETCS Projekt wurden seitens Arbeitspaket 2 in den openETCS Requirements dokumentiert und verfeinert. Dabei wurde frühzeitig erkannt, dass diese Anforderungen in zwei prinzipielle Richtungen gehen:

1. Anforderungen an die openETCS Tool Chain.
2. Anforderungen an das openETCS OBU Modell und die Modellierungsprozesse.

6.4.1.2 Traceability in der openETCS Tool Chain

Die Anforderungen bzgl. Traceability und Requirementsmanagement an die openETCS Tool Chain gliedern sich in die folgenden vier Teilaufgaben:

1. Konvertierung der SRS (ERA Subset-026) in ein Referenzierbares Format

Als referenzierbares Format wurde frühzeitig ReqIF identifiziert. ReqIF wird sowohl in der beteiligten Industrie (IBM Rational DOORS) als auch in der Open Source Welt (ProR in Eclipse) hinreichend unterstützt und für die Speicherung von Anforderungen eingesetzt.

Für die Konvertierung der SRS in das ReqIF Format existierten bislang lediglich unvollständige Ansätze. Daher wurde ein neues Konvertierungswerkzeug als Teil der openETCS Tool Chain entwickelt, um die SRS vom vorliegenden Microsoft Word Format nach ReqIF zu konvertieren. Das dabei entstehende ReqIF Dokument kann von den Entwicklern mit Hilfe des Werkzeugs ProR genutzt werden.

2. Verknüpfung der Anforderungen mit der Architektur

Die openETCS Architektur sollte durch SysML Diagramme dargestellt werden. Als Werkzeug wurde hierbei Papyrus gewählt. Daher bestand die zweite Anforderung an die openETCS Tool Chain darin, die Verknüpfung von SysML-Elementen (Papyrus) mit ReqIF-Elementen (ProR) auf einfache Art und Weise zu ermöglichen. Entsprechend wurde von Arbeitspaket 7 eine ProR Erweiterung für die openETCS Tool Chain entwickelt.

3. Verknüpfung der Anforderungen mit dem formalen Modell

Durch die Auswahl des Closed Source Werkzeugs SCADE des Herstellers Esterel für die funktionale Modellierung im Arbeitspaket 3 und ReqIF als Requirementsformat, lag die Verwendung des in SCADE Suite integrierten Werkzeugs Reqify für die Umsetzung der Traceability in der funktionalen Modellierung nahe. Reqify ist ebenfalls ein Closed Source Werkzeug.

Durch diese Entscheidung beschränkte sich der Entwicklungsaufwand in der openETCS Tool Chain auf die Dokumentation der Prozesse für die gewählte Lösung.

4. Die Open Proofs Vision

Die Wahl eines Closed Source Werkzeugs ist im Hinblick auf den angestrebten Open Proofs Ansatz an dieser Stelle nicht optimal. Als Baustein des Migrationspfads zu einer vollständig offenen Tool Chain wurde daher gegen Projektende die Eclipse-basierte Komponente ReqCycle für das Requirementsmanagement integriert. ReqCycle konnte jedoch wegen der späten Verfügbarkeit nicht mehr produktiv im Rahmen des openETCS ITEA Projektes eingesetzt werden.

6.4.1.3 Traceability in der Modellierung

Die in der Tool Chain bereitgestellten Lösungen wurden vom Modellierungssteam in Arbeitspaket 3 eingesetzt. Dabei zeichnete sich die Papyrus Lösung durch einfache Nutzbarkeit und hohen Nutzen zugleich aus. Der Entwickler sieht während der Modellierung Requirements und Modell nebeneinander angeordnet in der Benutzeroberfläche. Verknüpfungen zwischen Requirements und Architektur erfolgen durch einfaches Drag & Drop vom Papyrus Modell in das daneben liegende Requirements Feld (siehe Abbildung 16). Dank des neuen Formates können Zellen der SRS sehr feingranular gewählt werden.

Eine automatische Summenfunktion, die angibt welche Teile der Anforderungen bereits abgedeckt sind, kann mit dem Wechsel auf ReqCycle erreicht werden kann.

Die Lösung für das formale openETCS OBU Modell (siehe Abbildung 17) erlaubt Vollständigkeitsaussagen und Abdeckungsaussagen. Im Rahmen des Projektes waren diese Ergebnisse aber nur eingeschränkt nutzbar, da nicht alle Projektpartner Reqify Lizenzen besaßen und das Verfahren somit nicht in Breite angewendet werden konnte.

6.4.1.4 Traceability in Verifikation und Validierung

Traceability ist unverzichtbar in der Verifikation. Ohne die Verknüpfung von Anforderungen und Implementierung ist die Aussage, ob bestimmte Anforderungen im Modell umgesetzt werden nicht möglich. Damit bildet ein solches Verfahren die Grundlage für jegliche Tätigkeiten in der Verifikation.

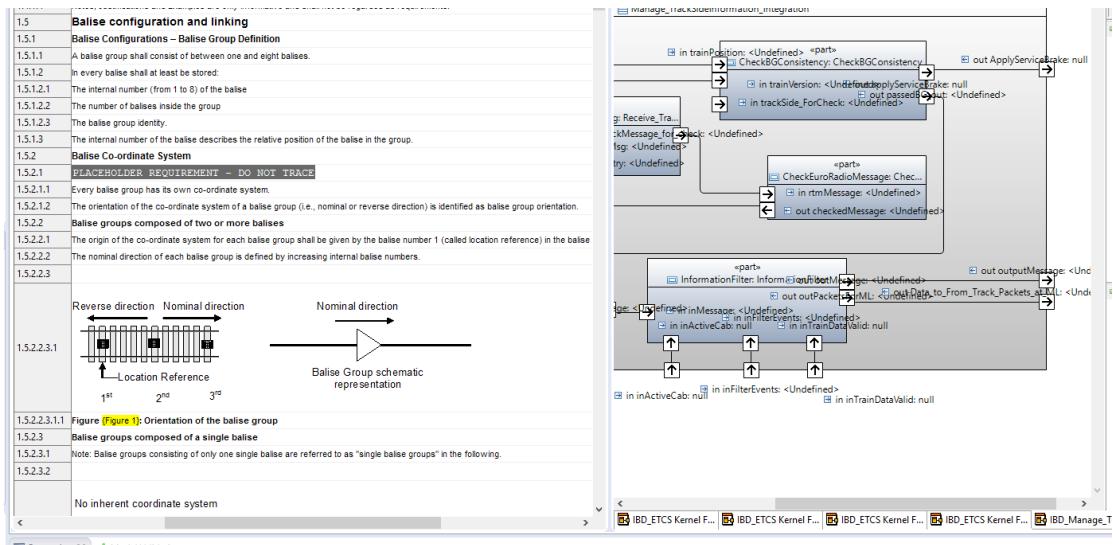


Abbildung 16. Traceability im openETCS Architekturdesign: Links ERA Subset-026 im ProR Viewer (RealIE), rechts das dazu gehörige Papyrus Modell (SysML).

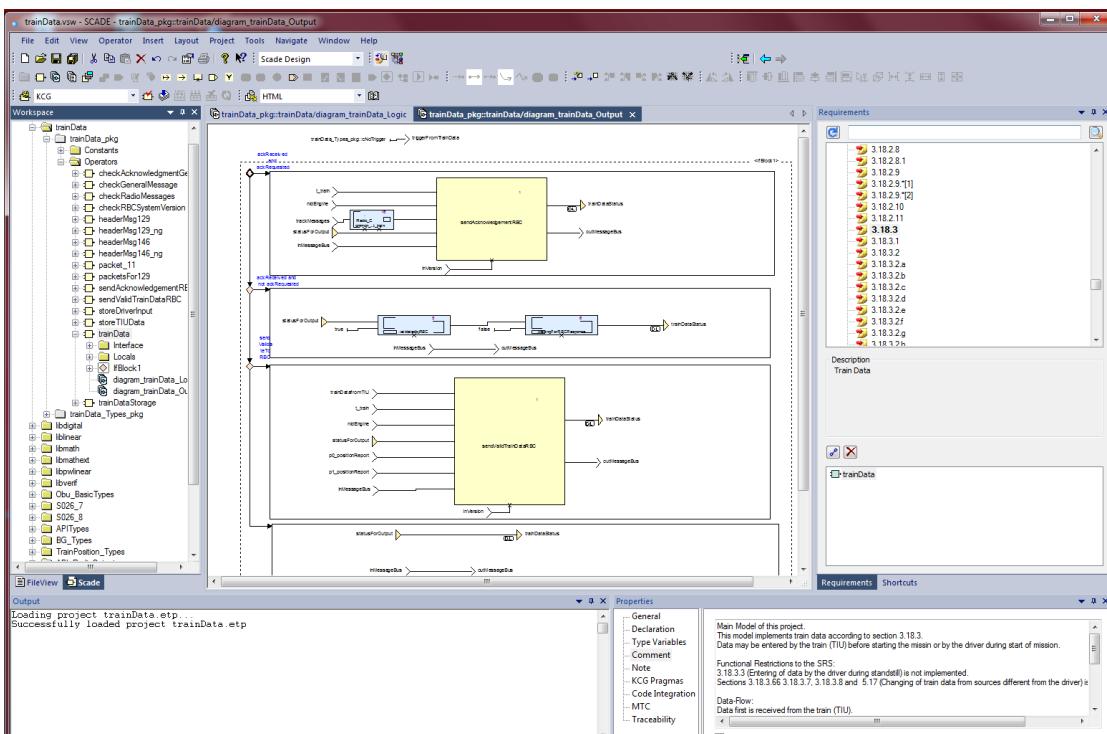


Abbildung 17. Traceability im funktionalen Modell der openETCS OBU. Links das SCADE Model, rechts die dazugehörige Liste der Anforderungen.

6.4.2 openETCS OBU Anforderungen

Ziel des openETCS Projekts war nicht die vollständige Modellierung der in ERA Subset-026 beschriebenen Spezifikation, da die hierfür erforderlichen Ressourcen die im ITEA Projekt verfügbaren Ressourcen deutlich überschritten hätten. Stattdessen war es Ziel des Projekts, ein ETCS OBU Modell zu erstellen, welches die vorgegebenen Betriebsabläufe und die Anforderungen einer realen und repräsentativen Strecke implementiert. Anhand dieses Beispiels sollten die eingesetzten Methoden und Prozesse erprobt werden („Proof of Concept“) und ein erstes Kernmodell entstehen.

Als Referenzstrecke wurde die niederländische ETCS Pilot Strecke Amsterdam-Utrecht ausgewählt. Auf Basis des vom Projektpartner Nederlandse Spoorwegen (NS) bereitgestellten Streckenplans – sowie mit Hilfe der Aufnahmen der Juridical Recording Unit (JRU) einer realen Fahrt eines ICE der DB über diese Strecke – wurde die Strecke für die openETCS Simulationsumgebung als Test- und Demonstrationsstrecke implementiert. Details zu dieser Methodik werden in Abschnitt 6.6.2 näher beschrieben.

Die aus diesem Anwendungsszenario, d. h. einer Fahrt auf dieser Strecke mit dem openETCS OBU Modell, resultierenden Anforderungen wurden analysiert und in 12 User Stories umgesetzt. Wie sämtliche Projektergebnisse sind die dokumentierten User Stories öffentlich und frei verfügbar [41]. Ergänzt wurden die User Stories durch vier Testfälle, die bei Abnahmetests für Fahrzeuge auf dieser Strecke typischerweise verlangt werden.

Aus den User Stories ergeben sich die technischen Anforderungen, über die im Folgenden ein kurzer Überblick gegeben wird. Die Beschreibung an dieser Stelle ist nicht vollständig und dient lediglich dazu, einen Überblick über die in der openETCS OBU vorhandene Funktionalität zu geben.

Bahnhof Amsterdam: Start of Mission

Das System einschließlich des EVC wird gestartet.

Der Triebfahrzeugführer öffnet den Führerstand. Anschließend wird an der Bedienschnittstelle (Driver Machine Interface, DMI) das Aufrüsten des Triebfahrzeugs durch den Triebfahrzeugführer durchgeführt. Das Fahrzeug unterstützt zu diesem Zeitpunkt und an diesem Ort auf der Strecke lediglich das nationale Zugsicherungssystem (Level NTC und Mode National System). Eine Verbindung zum RBC kann aufgrund fehlender Informationen in diesem Zustand noch nicht aufgebaut werden.

Beginn der Fahrt Das Fahrzeug fährt unter den Regeln des Nationalen Systems an. Die ersten Balisen werden passiert. Dabei werden über die Balise Transmission Module (BTM) des Fahrzeuges Inhalte gelesen und an den EVC übermittelt. Die empfangene Information der Balisen ermöglicht den Verbindungsauflauf zum Radio Block Center (RBC).

Annäherung an den Level 2 Bereich Das Fahrzeug sendet regelmäßige Positionsberichte an das Radio Block Center (RBC). Das RBC erkennt Anhand dieser Meldungen, dass sich das Fahrzeug nun dem Bereich nähert, in dem Fahrzeuge unter Überwachung durch das RBC fahren können (Level 2 und Mode Full Supervision). Das Fahrzeug erhält nun zunächst über die Strecke (mittels Balisen), die Aufforderung in Level 2 zu wechseln und anschließend über das RBC Informationen über die voraus liegende Strecke und die Streckenfreigabe für diesen Streckenabschnitt (Movement Authority, MA).

Wechsel in den Level 2 Bereich Etwa 200 Meter vor erreichen des ETCS Level 2 Bereiches sieht der Triebfahrzeugführer am DMI den anstehenden Level-Wechsel. Der Wechsel selbst muss vom Triebfahrzeugführer quittiert werden. Nach dem Wechsel in ETCS Level 2 fährt das Fahrzeug unter stetiger Überwachung der Geschwindigkeit durch das RBC. Die voraus-

liegenden Streckenabschnitte werden für den Fahrer grafisch aufbereitet und als so genannte „Planning Area“ dargestellt.

Fahren im Level 2 Bereich Das Fahrzeug kontrolliert selbstständig die zulässige Geschwindigkeit anhand der Länge des freien Streckenabschnittes und anderer Grenzparameter. Dazu werden regelmäßig Positionsberichte an das RBC gesendet, welche vom RBC benutzt werden um die Streckenfreigabe (Movement Authority) zu verlängern.

Annäherung an das Ziel Das Fahrzeug nähert sich nun dem Bahnhof Utrecht und damit dem Ende der Level 2 Strecke. Der Wechsel in das nationale Zugsicherungssystem (Level NTC und Mode National System) erfolgt wiederum aufgrund von Daten, welche über Balisen von der Strecke empfangen werden. Nach Wechsel auf Level NTC beendet das RBC die Radioverbindung.

Bahnhof Utrecht: End of Mission Der Triebfahrzeugführer bringt das Fahrzeug im Bahnhof zum Stehen und beendet die Fahrt. Das Fahrzeug wechselt in den Mode Standby.

Bei den weiteren Betriebsszenarien handelt es sich um folgende Vorgaben:

Mode Wechsel durch das RBC Hintergrund dieses Szenarios ist das Annähern an einen Gefahrenbereich, hier ein beschränkter Bahnübergang. In diesem Bereich muss der Triebfahrzeugführer entsprechend der Betriebsregeln auf Sicht und mit maximal 40 km/h fahren.

Das Fahrzeug fährt mit erteilter Streckenfreigabe zunächst im Mode „Full Supervision“. Für einen Teilabschnitt der Strecke muss das Fahrzeug „Auf Sicht“ (Mode „On Sight“) und mit reduzierter Geschwindigkeit fahren. Dieser Streckenabschnitt wird durch das RBC angekündigt. Der neue Mode wird dem Triebfahrzeugführer am DMI angezeigt und muss durch ihn quittiert werden. Das streckenseitige RBC wird über den neuen Mode und die Position durch den so genannten „Position Report“ informiert.

Zur Beendigung der On Sight Strecke sendet das RBC eine neue Streckenfreigabe mit Mode Full Supervision und erhöhter Geschwindigkeit. Das Fahrzeug kann seine Fahrt wieder mit erhöhter Geschwindigkeit entsprechend der neuen Streckenfreigabe fortsetzen.

Rücknahme einer zuvor erteilten Streckenfreigabe Hintergrund dieser und der darauf folgenden User Stories ist eine betriebliche Änderung des Streckenplans durch das Strecken Management. Dabei werden sicherheitskritische Unterschiede bei den Haltepunkten des Fahrzeuges betrachtet.

Das Fahrzeug fährt mit erteilter Streckenfreigabe im Mode Full Supervision. Der Fahrdienstleiter im Stellwerk nimmt diese Freigabe zurück, das betroffene Signal wechselt auf Rot. Das Fahrzeug reagiert und kann in diesem Szenario regelkonform mit der normalen Betriebsbremse vor dem Signal stehen bleiben.

Zwangsbremse durch die Rücknahme einer zuvor erteilten Streckenfreigabe Hier befindet sich das Fahrzeug mit erteilter Streckenfreigabe im Mode Full Supervision. Der Fahrdienstleiter nimmt diese Freigabe zurück, das betroffene Signal wechselt auf Rot. Das Fahrzeug reagiert und benötigt die Zwangsbremse um vor dem Signal zum Stehen zu kommen. In dieser Situation darf das Fahrzeug nicht mehr ohne Abstimmung mit dem RBC weiterfahren, d. h. das RBC muss eine neue Streckenfreigabe für die Weiterfahrt senden.

Überfahrenes eines Signals durch die Rücknahme einer zuvor erteilten Streckenfreigabe

Dieses Szenario ist ähnlich zum vorhergehenden. Das Fahrzeug kommt aber nicht mehr vor dem Ende der erlaubten Strecke zum Stehen und muss in Abstimmung mit dem RBC um einige Meter zurückgesetzt werden.

Mit Ausnahme des letzten Szenarios werden alle genannten Anforderungen bzw. Szenarien durch die openETCS OBU unterstützt. Das letzte Szenario konnte aufgrund der bereits fortgeschrittenen Projektlaufzeit nicht mehr vollständig realisiert werden.

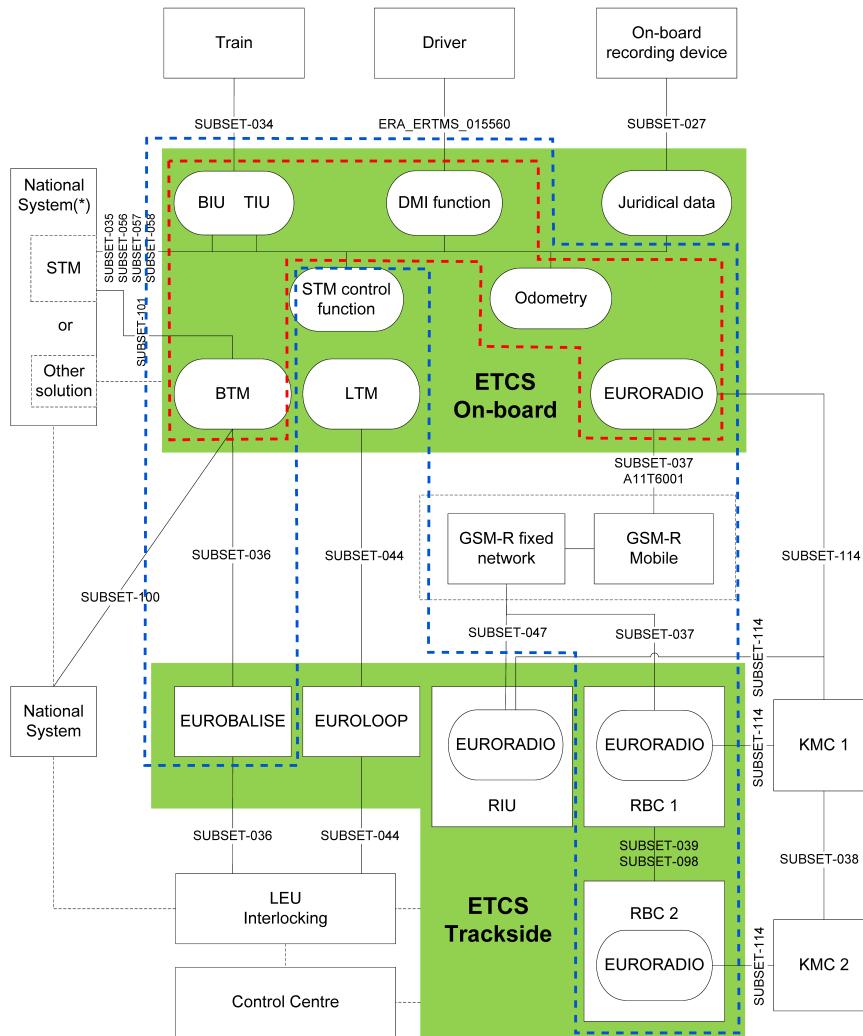


Abbildung 18. Umfang des openETCS OBU Modells gemäß ERA Subset-026, Kapitel 2.5.

6.4.3 openETCS OBU Architektur

Die Systemarchitektur der openETCS OBU basiert auf der im ERA Subset-026, Kapitel 2.5, definierten Systemstruktur (vgl. Abb. 18). Die blau gestrichelte Linie in Abb. 18 markiert die im openETCS Projekt berücksichtigten funktionalen Komponenten und die rot gestrichelte Linie markiert die für das Projekt relevanten Komponenten auf On-Board Seite. Ausgewählte Teile der Infrastrukturausrüstung (Eurobalise, Euroradio) wurden modelliert, um eine im Modell integrierte Testumgebung zu erhalten.

6.4.3.1 Top-Level Architektur und externe Schnittstellen

In Abbildung 19 wird die openETCS OBU Top-Level Architektur mit den externen Schnittstellen E1 bis E10 dargestellt. Die externen Schnittstellen dienen der Kommunikation zwischen der openETCS OBU (vgl. rot-gestrichelte Linie in Abb. 18) und externen Systemen wie Eurobalisen, Fahrzeugsystemen, dem Triebfahrzeugführer, etc. Im Folgenden geben wir einen kurze Beschreibung der externen Schnittstellen:

E1 Eingangs- und Ausgangsschnittstelle zwischen Stellwerk (Interlocking) und Eurobalise.

Diese Schnittstelle ist lediglich für gesteuerte Eurobalisen relevant.

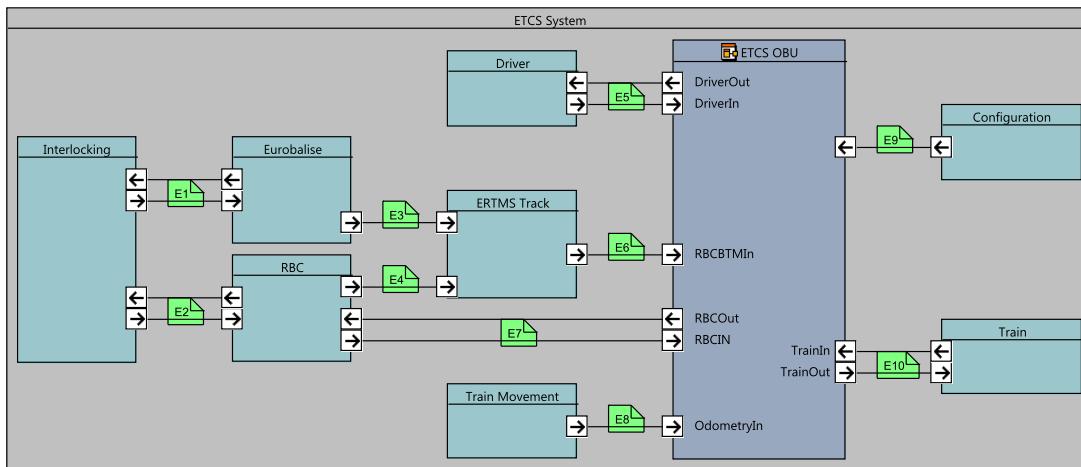


Abbildung 19. Top-Level Architektur mit externen Schnittstellen E1 bis E10.

- E2** Eingangs- und Ausgangsschnittstelle zwischen Stellwerk and Radio Block Center (RBC).
- E3** Eingangsschnittstelle von der Eurobalise zum ERTMS Track Modul.
- E4** Eingangsschnittstelle vom RBC zum ERTMS Track Modul.
- E5** Eingangs- und Ausgangsschnittstelle zwischen Triebfahrzeugführer und der ETCS OBU. Die Schnittstelle dient der Interaktion zwischen Triebfahrzeugführer und dem Bedienbildschirm (Driver Machine Interface, DMI) des Fahrzeugs.
- E6** Diese Eingangsschnittstelle vom ERTMS Track Modul zur ETCS OBU ist eine Verbund-schnittstelle und kombiniert die Schnittstellen E3 sowie E4 zum Empfangen von Strecken-nachrichten (von Eurobalisen oder dem RBC) an die ETCS OBU.
- E7** Eingangs- und Ausgangsschnittstelle zwischen RBC und ETCS OBU. Die Schnittstelle dient dem Management der Funkverbindungen mit dem RBC (dem so genannten Session Management) sowie dem Senden von Nachrichten von der ETCS OBU an das RBC. Es ist zu beachten, dass die ETCS OBU RBC Nachrichten über Schnittstelle E6 empfängt.
- E8** Eingangsschnittstelle vom Odometrie Modul zur ETCS OBU.
- E9** Eingangsschnittstelle zur ETCS OBU um Konfigurationsdaten wie Systemwerte, Fahrzeugg-konfiguration, etc. zu lesen.
- E10** Eingangs- und Ausgangsschnittstelle zwischen der ETCS OBU und dem Fahrzeug. Die Schnittstelle dient der Interaktion zwischen Fahrzeug und der ETCS OBU, d. h. Bremssteue-rung, Traktionskontrolle, Türbedienung, etc.

6.4.3.2 Funktionale Aufteilung der ETCS OBU

In Abb. 20 wird die funktionale Aufteilung der ETCS OBU Komponente aus Abb. 19 dargestellt. Die ETCS OBU Komponente selbst besteht aus den folgenden acht funktionalen Modulen:

- F1 Receive Information from Trackside** Das Modul ist verantwortlich für den Empfang von streckenseitigen Nachrichten (RBC und Eurobalise) und leitet diese weiter zum Modul F2 ETCS Kernel.
- F2 ETCS Kernel** Dieses Modul stellt die Kernkomponente der openETCS OBU mit umfangrei-cher Funktionalität dar. Eine detaillierte Beschreibung wird in den Abschnitten 6.4.5.1 bis 6.4.5.13 dieses Schlussberichts gegeben.
- F3 Measure Train Movement** Dieses Modul versorgt das Modul F2 ETCS Kernel mit Odome-trie Daten.

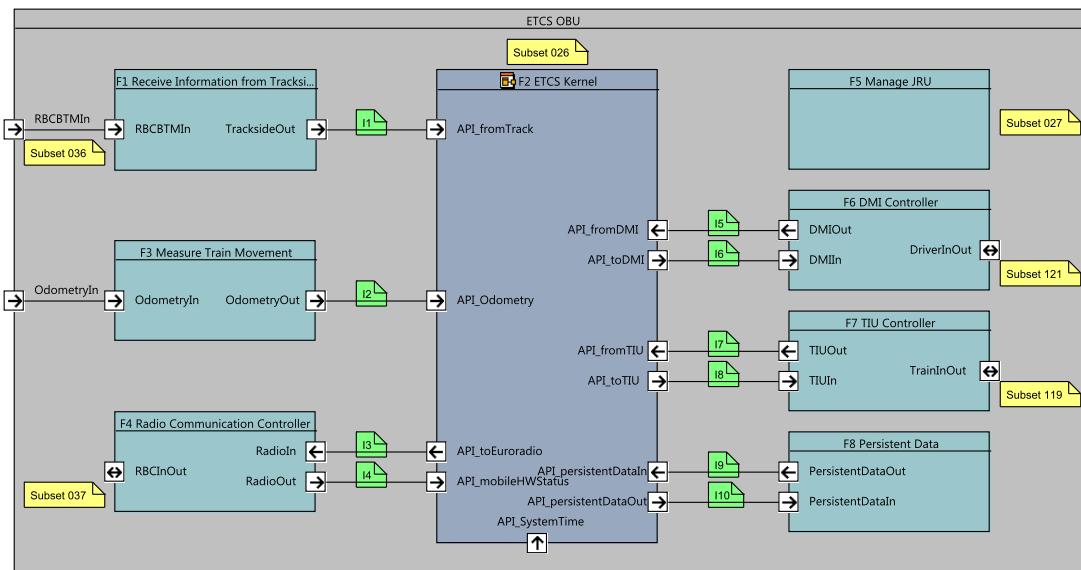


Abbildung 20. Systemarchitektur der openETCS OBU mit internen Schnittstellen I1 bis I10.

F4 Radio Communication Controller Das Modul steuert das Mobilfunkmodul (Radio Transmission Modules, RTM). Das Modul unterstützt bis zu zwei Mobilfunkmodule. Jedes der Mobilfunkmodule ist in der Lage einen eigenen Kommunikationskanal zwischen RBC und EVC aufzubauen.

F5 Manage JRU Bei dem Modul handelt es sich um die Aufzeichnungseinheit für juristische Daten. Es ist derzeit nicht im Umfang der openETCS OBU enthalten.

F6 DMI Controller Das Modul implementiert die Steuerelemente des Bediendisplays des Fahrzeugs.

F7 TIU Controller Das stellt die Verbindung des EVC mit diversen Fahrzeugkomponenten dar und ist unter anderem für die Ansteuerung der Bremsen zuständig.

F8 Persistent Data Das Modul versorgt den ETCS Kernel mit persistenten Daten des Fahrzeugs (Zuglänge, nationale Parameter, etc.).

Die genannten Module interagieren mittels der internen Schnittstellen I1 bis I10 (vgl. Abb. 20). Eine detaillierte Beschreibung des ETCS Kernel Moduls F2 folgt in Abschnitt 6.4.5. Der folgende Abschnitt beschreibt zunächst das openETCS Application Programming Interface (API).

6.4.4 Generische openETCS API

Durch das Ziel mit openETCS eine ETCS OBU Referenzeinheit zu entwickeln, entsteht die Notwendigkeit, die Eigenschaften für das System in seiner Anwendung genauer zu spezifizieren. Zu diesem Zweck wird das openETCS Application Programming Interface (API) als offene Referenz spezifiziert. Als Ausgangspunkt diente das im Betrieb bewährte API des Projektpartners Alstom, das als openETCS Arbeitsergebnis im GitHub Modelling Repository des Projekts veröffentlicht wurde.¹¹ Bei der Entwicklung der openETCS API wurde besonderes Augenmerk darauf gerichtet, dass diese flexibel gehalten wird, um für andere Hersteller adaptierbar zu sein. Die daraus abgeleitete Software Architektur und die Einbindung des openETCS OBU Modells wird in den folgenden Abschnitten beschrieben.

¹¹ Die API Dokumentation ist verfügbar unter https://github.com/openETCS/modeling/blob/master/openETCS%20ArchitectureAndDesign/D3.5.4%20_API/OETCS_API%20Requirements_v1.4.pdf.

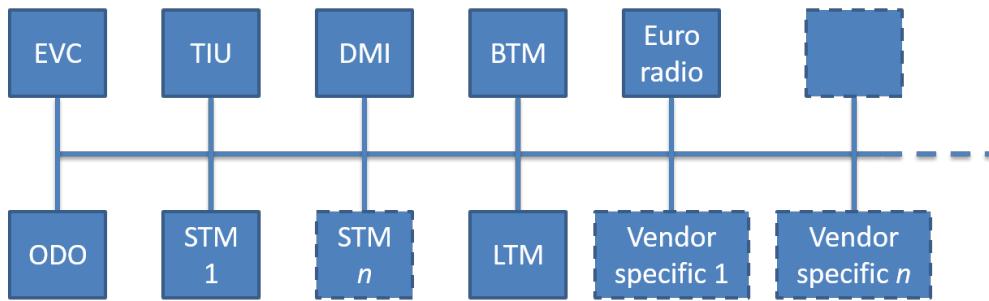


Abbildung 21. openETCS Referenz Hardware Architektur.

6.4.4.1 openETCS Hardware Architektur

Ausgangspunkt für die API Definition ist die Referenz Hardware (HW) Architektur, wie in Abbildung 21 dargestellt. Die Referenz Hardware Architektur ist konform mit ERA Subset-026, Kapitel 2 und definiert die logischen Hardwareeinheiten, die für das ETCS OBU System betrachtet werden müssen.

Die Kommunikation zwischen den Hardwareeinheiten erfolgt mittels asynchroner Meldungen. Typischerweise handelt es sich hier um ein Bussystem mit hoher Durchsatzrate. Im openETCS Demonstrator wird eine TCP/IP Verbindung für die Kommunikation eingesetzt, da diese technisch einfach auf verschiedenen Plattformen realisiert werden kann und die Performanz des Modells nicht der primäre Fokus im Rahmen des Projekts war. Die mögliche Trennung der einzelnen Bausteine auf unterschiedliche Plattformen ist dagegen ein wesentlicher Aspekt der Architektur, welcher von Anbeginn an verfolgt wurde.

6.4.4.2 openETCS Software Architektur

Die Software Architektur wird in Abbildung 22 dargestellt und setzt sich aus den folgenden vier Komponenten zusammen:

openETCS executable model Hierbei handelt es sich um das funktionale Modell des ETCS OBU Kerns, wie in Kapitel 6.4.5 beschrieben. Das „executable model“, d. h. der ausführbare Code, wird dabei aus dem funktionalen openETCS Modell generiert.

openETCS model run-time system Der ausführbare OBU Code muss in ein Laufzeitsystem eingebunden werden. In diesem Laufzeitsystem wird ein Takt und die Zykluszeit definiert sowie die Kommunikation mit anderen Einheiten über Buffer ermöglicht. Als weitere Funktionen werden die Kodierung und Dekodierung von Meldungen sowie ein Scheduling der unterschiedlichen Meldungsflüsse benötigt, um die Priorität von Meldungen berücksichtigen zu können.

Hersteller spezifische API Adapter Diese Adapter bilden die spezifischen Hersteller Schnittstellen auf das generische API ab. Für die Implementierung im openETCS Projekt ist hier insbesondere ein Adapter für den ERSA Simulator relevant, welcher im Arbeitspaket 5 für den openETCS Demonstrator realisiert wurde.

Hersteller spezifische Funktionen Dies sind weitere Elemente des Systems wie Bussysteme oder andere Hardwareeinheiten, die nicht durch openETCS direkt definiert werden.

Aus dieser Architektur ergeben sich die folgenden drei Schnittstellen:

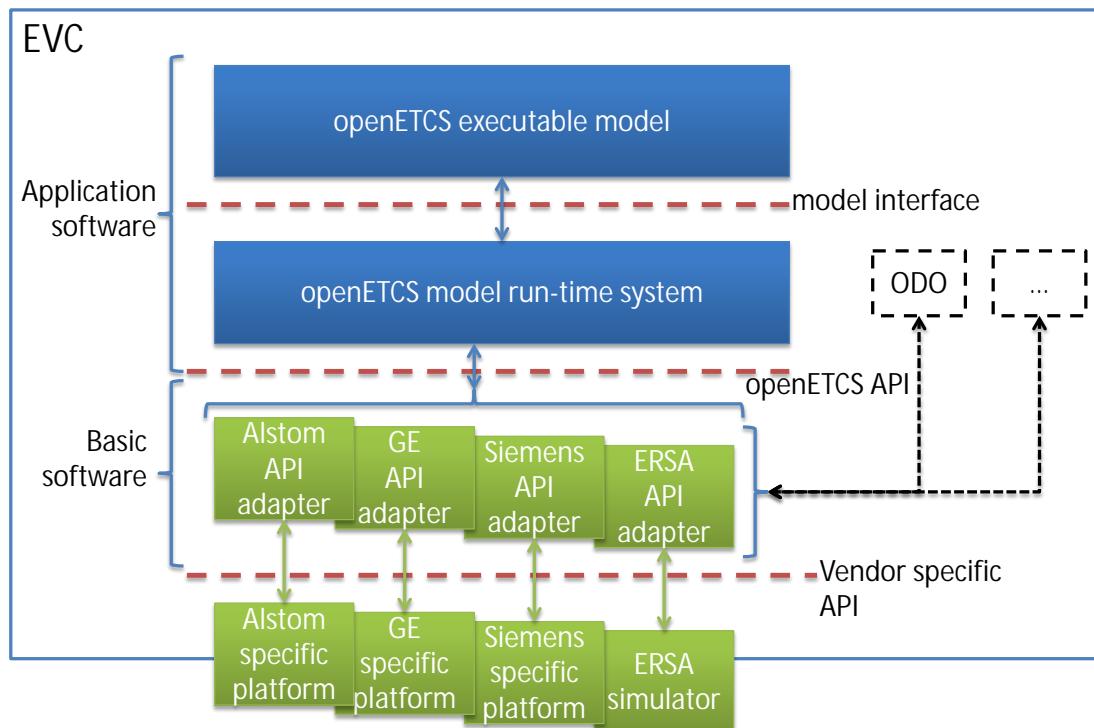


Abbildung 22. openETCS Referenz Software Architektur.

Model Interface Das Model Interface ist die Schnittstelle zwischen der openETCS OBU und der Laufzeitumgebung.

openETCS API Dies ist die generische Schnittstelle zu den herstellerspezifischen Adapters.

Vendor specific API Hierbei handelt es sich um herstellerspezifische Schnittstellen zu den jeweiligen Plattformen der Hersteller.

6.4.4.3 openETCS Meldungen

Zu den im Laufzeitsystem enthaltenen Funktionen zählt die Kodierung von Nachrichten vom EVC zum umgebenden System und die Dekodierung von Nachrichten vom umgebenden System zum EVC. Die System Requirements Spezifikation (SRS) definiert diese Nachrichten in Kapitel 7 und 8 des Subset-026 in kodierter Form, d. h. Inhalte der Nachrichten werden über Bit-Positionen in der Nachricht definiert. Zur Verarbeitung innerhalb des openETCS OBU Modells sind diese Strukturen weder praktikabel noch performant. Daher wurden diese Funktionen in das Laufzeitsystem gelegt.

Abbildung 45 auf Seite 76 zeigt schematisch den Verarbeitungsablauf der Meldungskodierung von der Spezifikation bis zum ausführbaren C-Code. Die wesentlichen Komponenten für die Meldungskodierung gemäß Abbildung 45 sind die Folgenden:

ETCS Spezifikation Relevant ist hier die Definition der Meldungen und Pakete aus ERA Subset-026, Kapitel 7 und 8. Diese beiden Kapitel entsprechen soweit einer formalen Spezifikation, dass der Einsatz eines Werkzeugs zur Code-Erstellung möglich ist. In einem ersten Schritt entsteht aus der Spezifikation eine XML Beschreibung der Datentypen, das so genannte openETCS Data Dictionary.

openETCS Data Dictionary Das Data Dictionary im XML Format enthält die vollständige Information über den Aufbau der standardisierten Schnittstellen des ETCS und stellt die Basis

für die nächsten Schritte der Code Generierung dar. Einerseits wird aus den XML Daten ein SCADE Modell erzeugt, um sicherzustellen, dass im EVC Modell die korrekten Datentypen verarbeitet werden. Parallel zu diesem Modell wird auch der zur Meldungsverarbeitung benötigte C-Code generiert.

C-Code Der C-Code bearbeitet in der untersten Ebene den Bit-Stream, welcher über die Meldungsschnittstelle von der Strecke empfangen wird. Die Informationen werden hier wieder zu logischen Paketen, Telegrammen und Meldungen zusammengesetzt. Soweit sinnvoll werden in diesem Schritt auch alle notwendigen strukturellen Prüfungen der Meldungsinhalte vorgenommen. Der C-Code für die Telegramme arbeitet in einem Softwarepaket zusammen mit dem C-Code, der direkt aus dem funktionalen openETCS Modell erzeugt wird. Die Ausgabe der Dekodierung erfolgt im Format des Meldungsbusses.

Meldungsbus Der Meldungsbus selbst ist eine Bibliothek und wurde in der Modellkomponente trackMessages implementiert. Er ist als Teil des openETCS Modells in Abbildung 45 zu sehen. Der daraus generierte C-Code bildet die Integer Stream Funktionen. Die Bibliothek stellt den EVC Anwendungen die Meldungen über komfortable und effiziente Zugriffsoperatoren zur Verfügung.

6.4.5 Komponenten des ETCS Kernel Moduls F2

Nach der Beschreibung der openETCS Systemarchitektur (vgl. Abschnitt 6.4.3) sowie des Application Programming Interface im vorigen Abschnitt, wenden wir uns nun der zentralen Komponente der openETCS Architektur zu. Dies ist das ETCS Kernel Modul F2, welches wiederum aus den 13 funktionalen Komponenten F2.1 bis F2.13 besteht:

- F2.1 Manage_TrackSideInformation_Integration
- F2.2 Manage_ETCS_Procedures
- F2.3 trainData
- F2.4 TrackAtlas
- F2.5 ManageLevelAndMode
- F2.6 calculateTrainPosition
- F2.7 SpeedSupervision_Integration
- F2.8 Provide_Position_Report
- F2.9 MoRC_HO
- F2.10 manageDMI_input
- F2.11 manageDMI_output
- F2.12 manage TIU_input
- F2.13 manage TIU_output

Aufgrund der Komplexität des Moduls verzichten wir an dieser Stelle auf eine grafische Gesamtübersicht des ETCS Kernel Moduls F2. Stattdessen wird in den Abbildungen 23 bis 39 jede der 13 Komponenten einzeln im Kontext mit den jeweils für diese relevanten anderen Komponenten in F2 dargestellt. Die Verbindungen zwischen den Komponenten entsprechen dem Datenfluss, wie er im openETCS OBU Modell realisiert wurde. Das funktionale Modell ist auf eine Zykluszeit von 50 Millisekunden ausgelegt. In jedem Zyklus werden je Komponente zunächst die Eingänge gelesen, die Daten entsprechend der definierten Funktionalität verarbeitet und schließlich die verarbeiteten Daten an den Ausgängen der Komponente bereitgestellt.

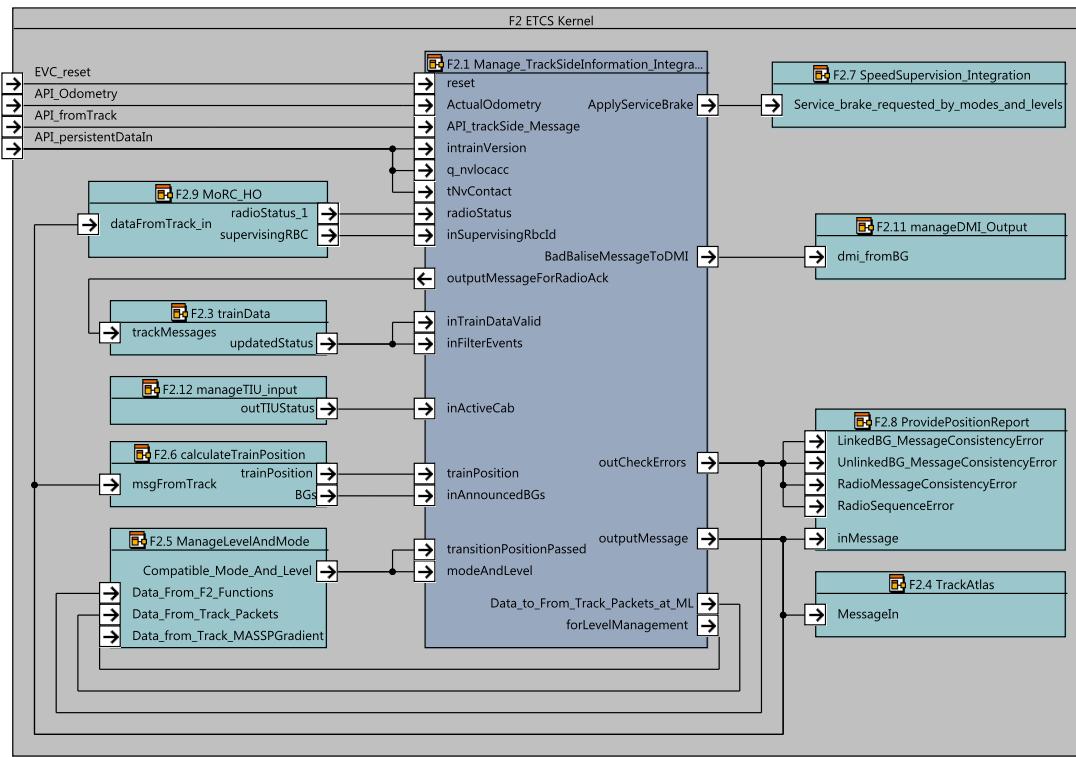


Abbildung 23. F2 ETCS Kernel SysML Diagramm mit Fokus auf die Komponente F2.1 Manage_TrackSideInformation_Integration.

Das F2 ETCS Kernel Modul bildet den Integrationsrahmen für die Sub-Komponenten des EVC und wird zyklisch aufgerufen. In jedem Zyklus werden zunächst die aktualisierten Eingaben verarbeitet, die Sub-Komponenten in fester Reihenfolge ausgeführt und schließlich die Ausgabe aufbereitet.

Eine besondere Herausforderung bei der Modellierung waren die komplexen Datenstrukturen, die für die Modellierung von Telegrammen der Eurobalisen sowie den Radio Nachrichten erforderlich waren. Diese im ERA Subset-026, Kapitel 7 und 8 definierten Datenstrukturen sind, nach Standard, in ihrer Zusammensetzung und Länge variabel ausgelegt. Datenstrukturen variabler Größe stehen jedoch im Widerspruch zu den Sicherheitsanforderungen an den OBU Code. Das openETCS OBU Modell löst dies mithilfe der Track-Messages Komponente, welche einen Datenbus für ankommende und abgehende Meldungen zwischen Strecke und EVC bereitstellt.

In den folgenden Abschnitten werden die 13 Sub-Komponenten des ETCS Kernel Moduls F2 beschrieben.

6.4.5.1 F2.1 Manage_TrackSideInformation_Integration

Diese Komponente implementiert die in ERA Subset-026, Kapitel 3.4 spezifizierten Funktionen zum Empfang von streckenseitigen Nachrichten. Das Fahrzeug erhält dabei über zwei Wege Informationen von der Streckenseite: Über die im Gleis verbauten Eurobalisen sowie über Radio Meldungen vom RBC. Die Eurobalisen dienen zum einen der Positionsbestimmung des Fahrzeugs im Koordinatensystem der Strecke. Zusätzlich erhält das Fahrzeug über die in den Eurobalisen gespeicherten Telegramme Steuerungsinformationen. Die Radio Meldungen dienen im Wesentlichen der Steuerung des Fahrzeugs durch Übermittlung von Informationen wie z. B. der Streckenfreigaben.

Im der Manage_TrackSideInformation_Integration Komponente werden sowohl Telegramme von Eurobalisen als auch Radio Meldungen von den empfangenden Einheiten (BTM und RTM) in das OBU Modell eingespeist.

Für die Schnittstelle zu Eurobalisen werden dabei folgende Funktionen bereitgestellt:

1. Kombination einzelner Balisen-Telegramme zu Balisengruppen Meldungen.
2. Erkennen von Duplikaten von Balisen-Telegrammen.
3. Erkennen von fehlenden oder falsch angeordneten Balisen.
4. Überprüfung der Konsistenz von Balisen-Telegrammen in Bezug auf die übermittelten Pakete.
5. Überprüfung der Position von Balisengruppen in Bezug auf die vorgegebene Strecke.

Erkannte Fehler werden zur weiteren Behandlung an andere Komponenten des OBU Modells weitergegeben. Reaktionen können z. B. eine Anzeige über den Fehler an den Triebfahrzeugführer oder die Einleitung einer Notbremsung sein.

Für die Radio Schnittstelle werden folgende Funktionen bereitgestellt:

1. Erkennen von fehlenden oder falsch angeordneten Radio Meldungen.
2. Überprüfung der Konsistenz von Radio Meldungen in Bezug auf die übermittelten Pakete.
3. Überprüfung der Radio Kommunikation auf Abbruch/Unterbrechung (Time-Out).

Für beide Kommunikationswege wird zudem sichergestellt, dass die übermittelten Informationen für die gefahrene Richtung des Triebfahrzeugs und für die aktuelle Betriebsart (den ETCS Mode) gültig sind bzw. nach einem Level- oder Betriebsart-Wechsel gültig werden können (vgl. ERA Subset-026, Kapitel 4.8 und 4.9). Im Ausgang des Modells werden die empfangenen Informationen in Form eines Busses an für die weiteren Komponenten des ETCS OBU Kernels F2 bereitgestellt.

6.4.5.2 F2.2 Manage_ETCS_Procedures

Die ETCS Prozeduren werden in Kapitel 5 von ERA Subset-026 spezifiziert. Sie beschreiben die dynamischen Abläufe zwischen Systemkomponenten und sind eine Voraussetzung für das Zusammenspiel der Komponenten von ETCS Systemen verschiedener Hersteller. Ohne die genaue Beachtung der in den Prozeduren festgeschriebenen Abläufe und der damit festgelegten Schnittstellen sind Systeme wie das EVC, das DMI oder das RBC nicht in der Lage miteinander zu kommunizieren.

Für das operative Szenario im openETCS Projekt (befahren der ETCS Strecke Amsterdam Utrecht) sind lediglich die Prozeduren „Start of Mission“ (Aufrüsten) und „End of Mission“ (Ende der Fahrt) von Bedeutung. Diese wurden für die auf der Strecke realisierten ETCS Level NTC und Level 2 in dieser Komponente implementiert.

6.4.5.3 F2.3 trainData

Für die sichere Steuerung des Fahrzeugs benötigt die Leitstelle charakteristische Information über das Fahrzeug, wie zum Beispiel die Länge des Zuges oder die maximale Geschwindigkeit. Diese Daten und deren sichere Handhabung ist unter dem Begriff Train Data in ERA Subset-026,

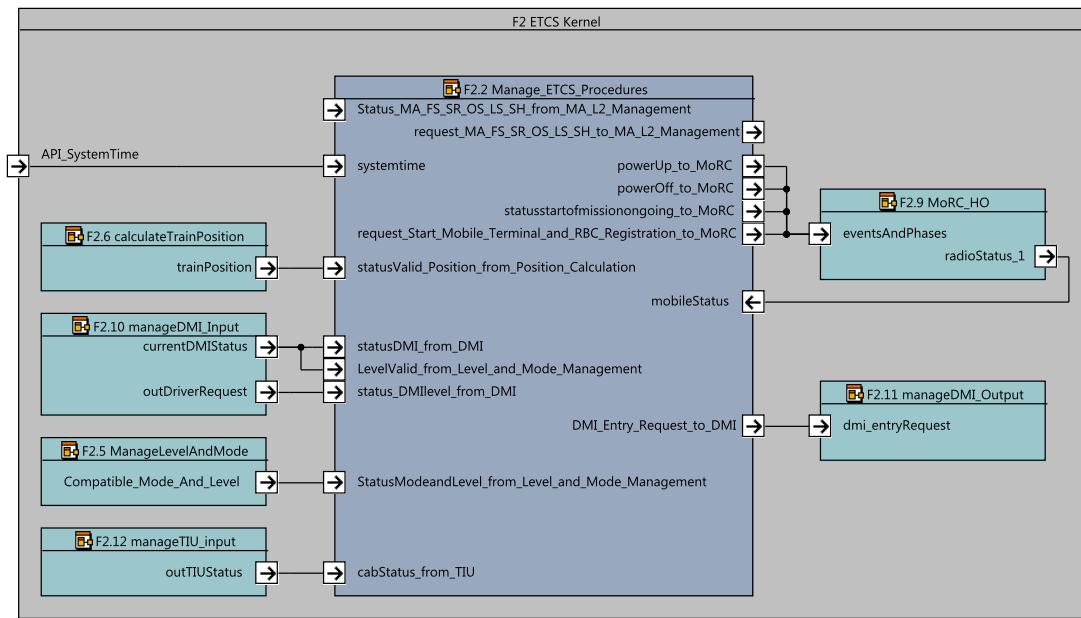


Abbildung 24. F2 ETCS Kernel SysML Diagramm mit Fokus auf die Komponente F2.2 Manage_ETCS_Procedures.

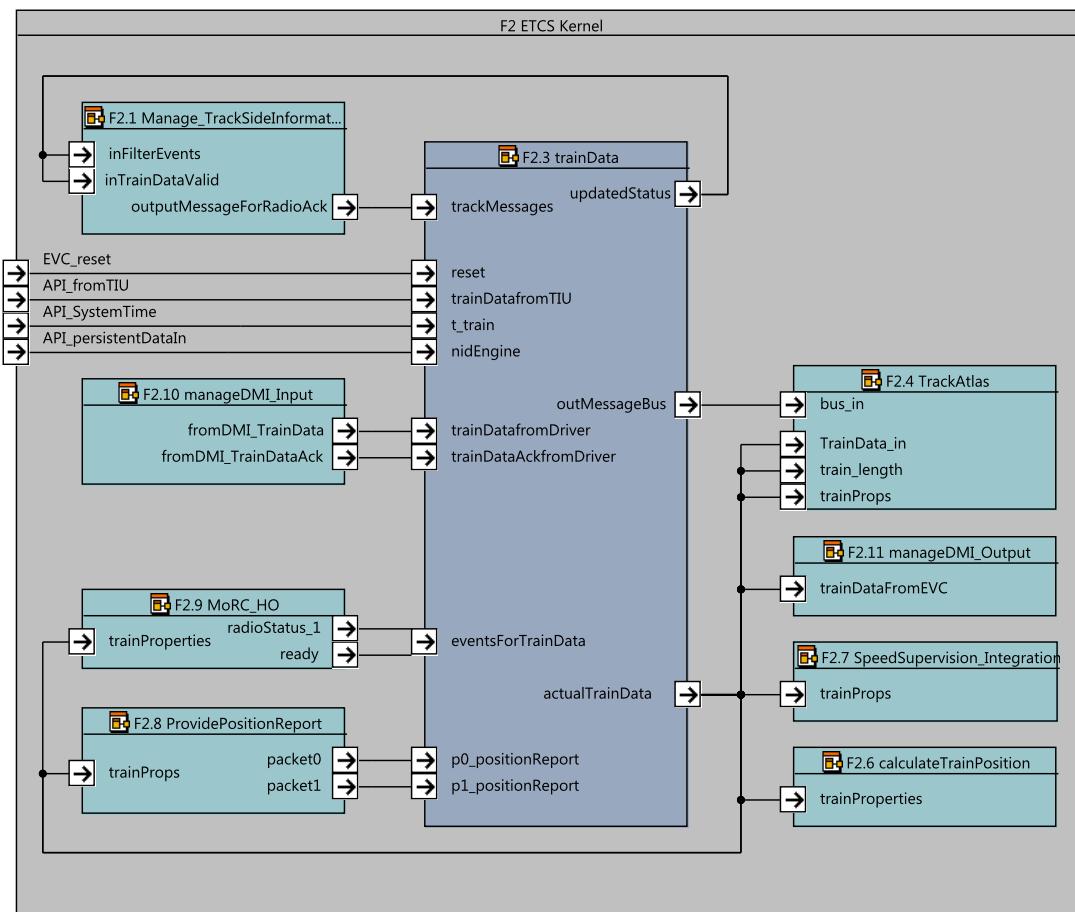


Abbildung 25. F2 ETCS Kernel SysML Diagramm mit Fokus auf die Komponente F2.3 trainData.

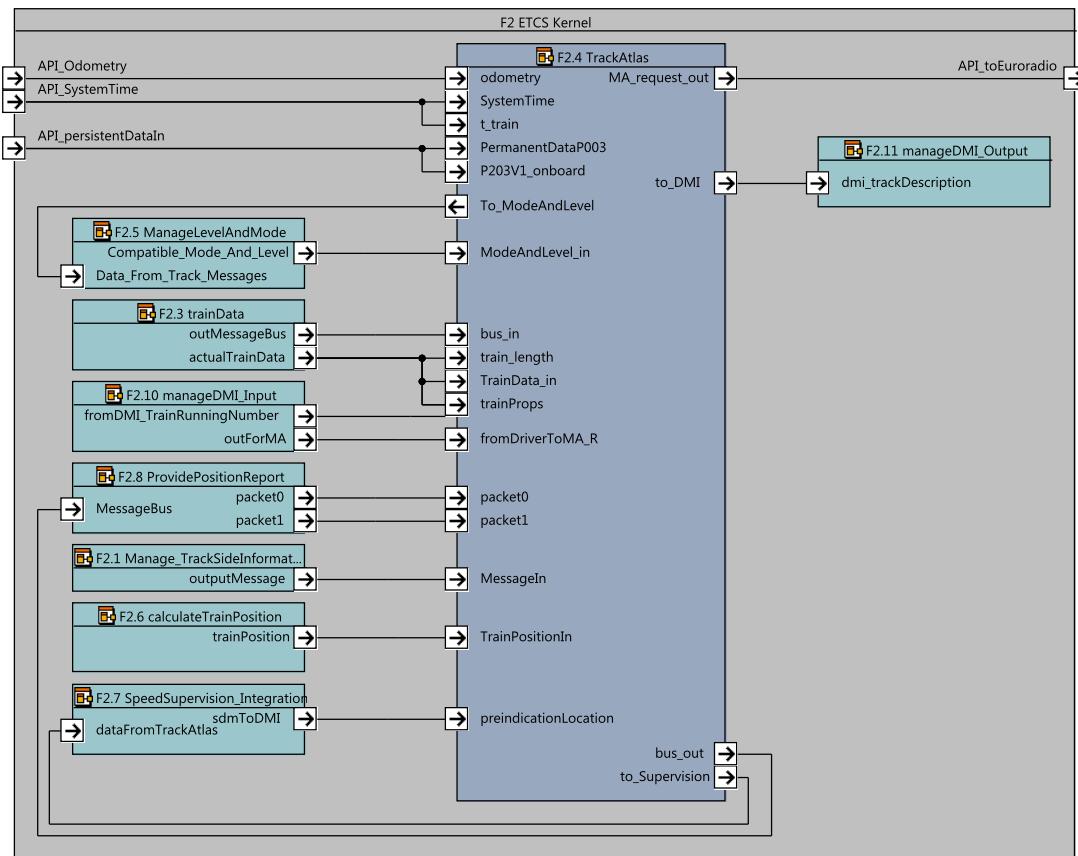


Abbildung 26. F2 ETCS Kernel SysML Diagramm mit Fokus auf die Komponente F2.4 TrackAtlas.

Kapitel 3.18.3 definiert. Die Komponente `trainData` (siehe Abb. 25) stellt diesbezüglich die folgenden Funktionen bereit:

- Initialisierung.
- Erstes Setzen der Daten anhand der Systemwerte des Fahrzeugs über die Train Interface Unit (TIU).
- Anschließende Eingabe und Validierung von Daten durch den Triebfahrzeugführer mittels DMI.
- Bei RBC-Anmeldung/Wechsel Validierung der Daten durch das RBC.
- Im laufenden Betrieb Bereitstellung gültiger Daten für das EVC System.

6.4.5.4 F2.4 TrackAtlas

Ein wichtiger Baustein des ETCS Kernels ist die Verwaltung einer internen Repräsentation bzw. Darstellung der Strecke. Die openETCS OBU setzt dies mithilfe der TrackAtlas Komponente um (siehe Abb. 26). Die interne Streckenrepräsentation des TrackAtlas bildet u. a. folgende Streckendaten ab:

- Physikalischen Eigenschaften der Strecke, wie die Steigung oder Neigung.
- Geschwindigkeitsprofile der Strecke und andere Geschwindigkeitsbeschränkungen.
- Die aktuelle Streckenfreigabe (Movement Authority, MA).

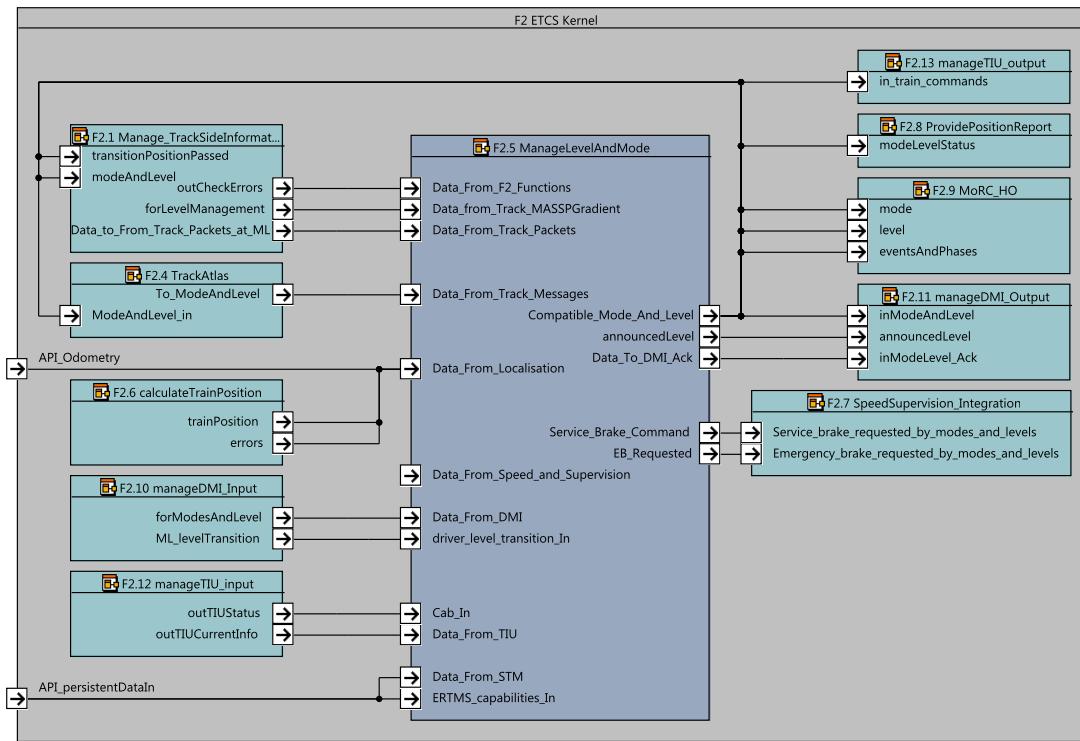


Abbildung 27. F2 ETCS Kernel SysML Diagramm mit Fokus auf die Komponente F2.5 ManageLevelAndMode.

Der TrackAtlas stellt die aktuellen Daten in optimierter Form für die Geschwindigkeitsüberwachung, die Anzeige beim Fahrer sowie Mode und Level Wechsel zur Verfügung. Unterstützt wird der TrackAtlas dabei durch die genaue Fahrzeugposition, welche durch die CalculateTrainPosition Komponente F2.6 bereitgestellt wird.

6.4.5.5 F2.5 ManageLevelAndMode

Die Komponente F2.5 ManageLevelAndMode (siehe Abb. 27) übernimmt, wie der Name schon andeutet, das Management von ETCS Level und Mode innerhalb des OBU Kernels. Die ETCS Ausbaustufe, das so genannte Level, bestimmt u. a. die Kommunikationsmöglichkeiten zwischen Streckenseite und Fahrzeug. Die Kommunikation kann dabei punktförmig (mittels Signalen in ETCS Level 1) oder kontinuierlich sein (mittels Radiokommunikation in ETCS Level 2). Die verschiedenen Ausbaustufen der Kommunikation zwischen Triebfahrzeug und Strecke sind in ERA Subset-026, Kapitel 2 definiert. Der ETCS Mode stellt die Betriebsart des Fahrzeugs dar. Zu den möglichen Betriebsarten gehören u. a. „Rangieren“ (Shunting) oder „ETCS voll überwachte Zugfahrt“ (Full Supervision).

Das Mode und Level Management implementiert die Übergänge zwischen den verschiedenen Leveln und Betriebsarten in Abhängigkeit von anliegenden Triggern und setzt insbesondere die in ERA Subset-026, Kapitel 4 beschriebene Wechselwirkung zwischen Mode und Level um. Im openETCS OBU Modell wurde die gesamte Spezifikation bzgl. Mode und Level implementiert, obgleich in den gewählten Szenarien lediglich ETCS Level NTC und Level 2 relevant sind und die Zahl der auf der Strecke Amsterdam Utrecht erforderlichen Betriebsarten gering ist.

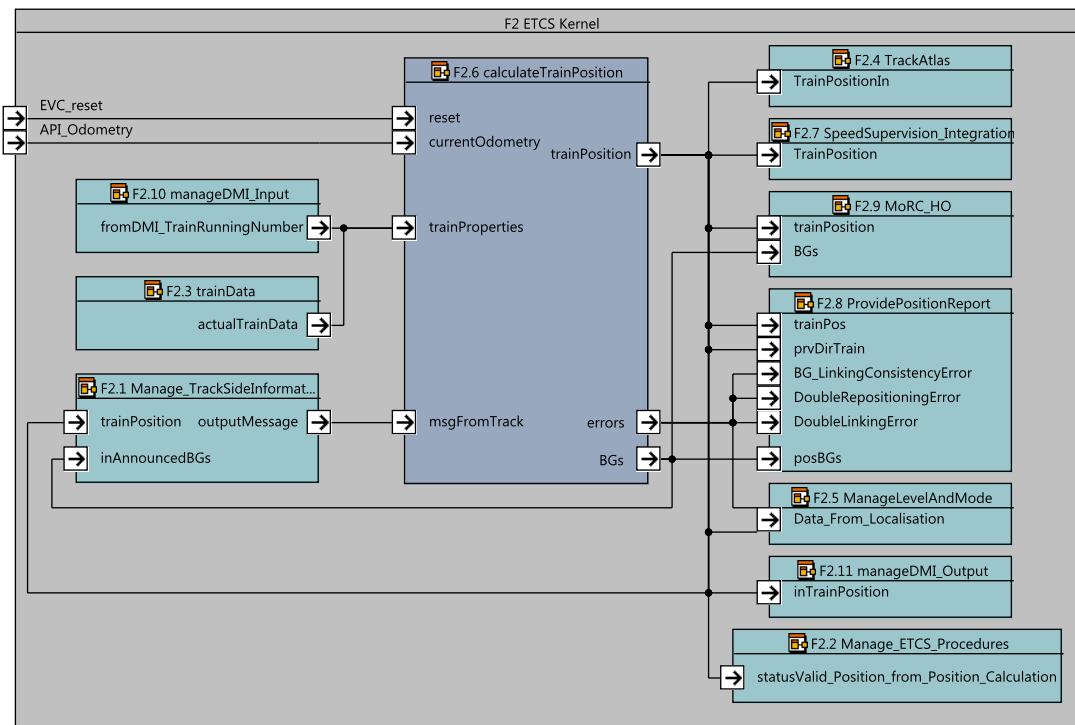


Abbildung 28. F2 ETCS Kernel SysML Diagramm mit Fokus auf die Komponente F2.6 calculateTrainPosition.

6.4.5.6 F2.6 calculateTrainPosition

Die Komponente calculateTrainPosition stellt eine Implementierung von ERA Subset-026, Kapitel 3.6, (Location Principles, Train Position and Train Orientation) und in Teilen von Subset-026, Kap. 3.4.2 (Balise Coordinate System) und 3.16.2.3 (Linking Consistency) dar.

calculateTrainPosition verwertet Informationen aus der Odometrie, dem Balisenempfangskanal des Fahrzeugs und die von den Balisen und via Radio empfangenen Daten und erzeugt daraus ein fahrzeugeigenes eindimensionales Koordinatensystem, positioniert die bereits überfahrenen und angekündigte Balisengruppen darin und bestimmt zu jedem Zeitpunkt die aktuelle Fahrzeugposition. Zusätzlich überwacht es während der Fahrt ständig, ob angekündigte Balisengruppen innerhalb ihres Erwartungsfensters gefunden wurden oder vermisst werden.

Die Komponente calculateTrainPosition bildet die Basis für alle ortsbezogenen Daten des Fahrzeugs wie Geschwindigkeits- und Gradientprofile, Bremskurven und die Kommunikation zwischen Fahrzeug und Radio Block Center (RBC) auf Streckenseite. Die wesentlichen, durch diese Komponente allen anderen Komponenten der OBU zur Verfügung gestellten Informationen sind:

- Die aktuelle Fahrzeugposition mit Toleranzbereich und Attributen.
- Die aktuelle und vorletzte LRBG (Last Recently Used Balise Group).
- Eine nach ihrem Ort sortierte und mit ihrem Ort samt Toleranzbereich versehene Liste aller bekannten überfahrenen und angekündigte Balisengruppen.
- Fehlerinformationen, insbesondere erkannte Linking Consistency Errors.

Neben calculateTrainPosition wurde eine Bibliothek „BasicLocationFunctions“ für ortsbezogene Berechnungen in allen anderen Komponenten des OBU Gesamtmodells erstellt.

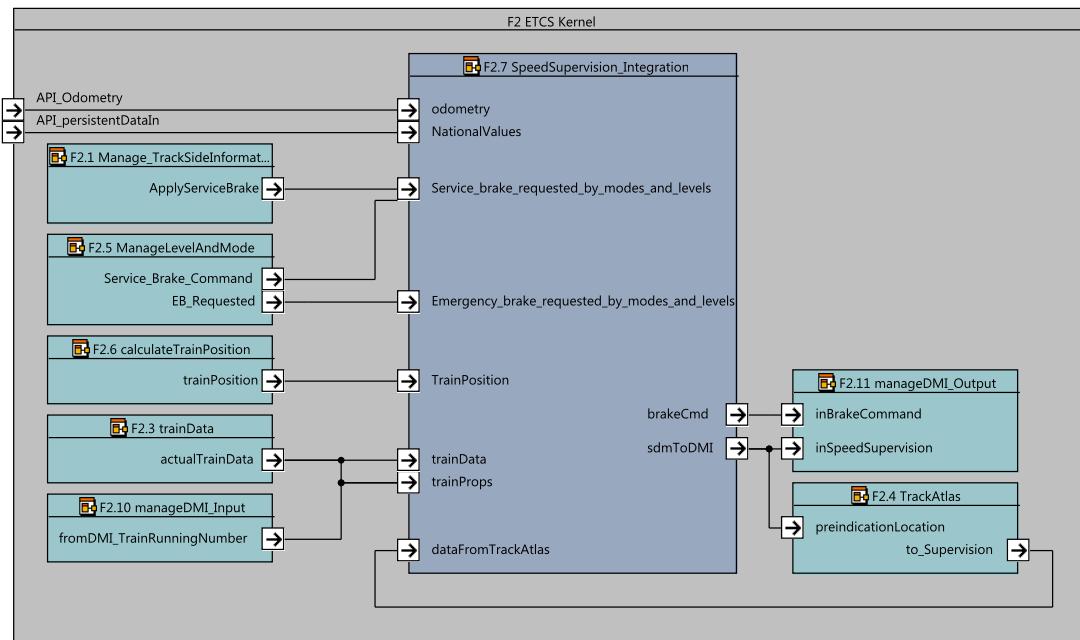


Abbildung 29. F2 ETCS Kernel SysML Diagramm mit Fokus auf die Komponente F2.7 SpeedSupervision_Integration.

Der Modellierung vorgeschaltet war eine eingehende Analyse der geforderten Ortung des Fahrzeugs. Aus dem Anforderungsdokument Subset-026, Kapitel 3.6, wurde eine mathematische Berechnungsvorschrift der Ortungsfunktion „DetermineTrainLocationProcedures“ abgeleitet.¹² Diese wurde gegen einen zweiten, diversitären Ansatz geprüft und einer Verifikation unterzogen.¹³ Die Verifikation des Modells erfolgte dabei implizit zusammen mit anderen ortsbezogenen Funktionen, da calculateTrainPosition die unabdingbare Basis für diese darstellte.

6.4.5.7 F2.7 SpeedSupervision_Integration

Die Komponente F2.7 SpeedSupervision_Integration stellt die Umsetzung der so genannten Bremskurven dar und ist eine Implementierung des Subset-026, Kapitel 3.13 (Speed and distance monitoring), einschließlich sämtlicher Unterkapitel. Das so genannte Lambda-Train Bremsmodell (bzw. Conversion Modell, Berechnung nach Bremsprozenten) wurde vollumfänglich umgesetzt. Das fahrzeugspezifische Gamma-Train-Bremsmodell wurde mangels charakteristischer Fahrzeug- und Streckendaten hingegen nicht implementiert.

Das Speed and Distance Monitoring hat die Aufgabe im Full- bzw. Limited Supervision Modus (in kleinerem Umfang auch in anderen Modi) die Ist-Geschwindigkeit mit den zulässigen Geschwindigkeiten der Strecke und des Fahrzeugs zu vergleichen. Hinzu kommt die Überwachung von Bereichen mit Geschwindigkeitsbeschränkungen und das Ende der freigegebenen Strecke. Damit sich das Fahrzeug stets im zulässigen Geschwindigkeits-Orts-Bereich aufhält, werden anhand der Fahrzeugcharakteristik Bremsmodelle berechnet (so genannte Bremskurven), um einen rechtzeitigen Bremseingriff zu ermöglichen.

¹² Verfügbar unter https://github.com/openETCS/SRS-Analysis/blob/master/System%20Analysis/WorkingRepository/Group4/SUBSET_26_3-6/DetermineTrainLocationProcedures.pdf.

¹³ Die Dokumentation der Verifikation dieser Komponente ist verfügbar unter https://github.com/openETCS/validation/blob/master/MinutesOfMeeting/141028_Verification_Closing_Session_Train_Positioning/A-D-VerfRprt-Train-Position-141028-V01.pdf und <https://github.com/openETCS/modeling/blob/master/openETCS%20ArchitectureAndDesign/Work%20Groups/Group%204/CalculateTrainPosition/Calculations/MatchingTrainPositionCalculationAlgorithms.pdf>.

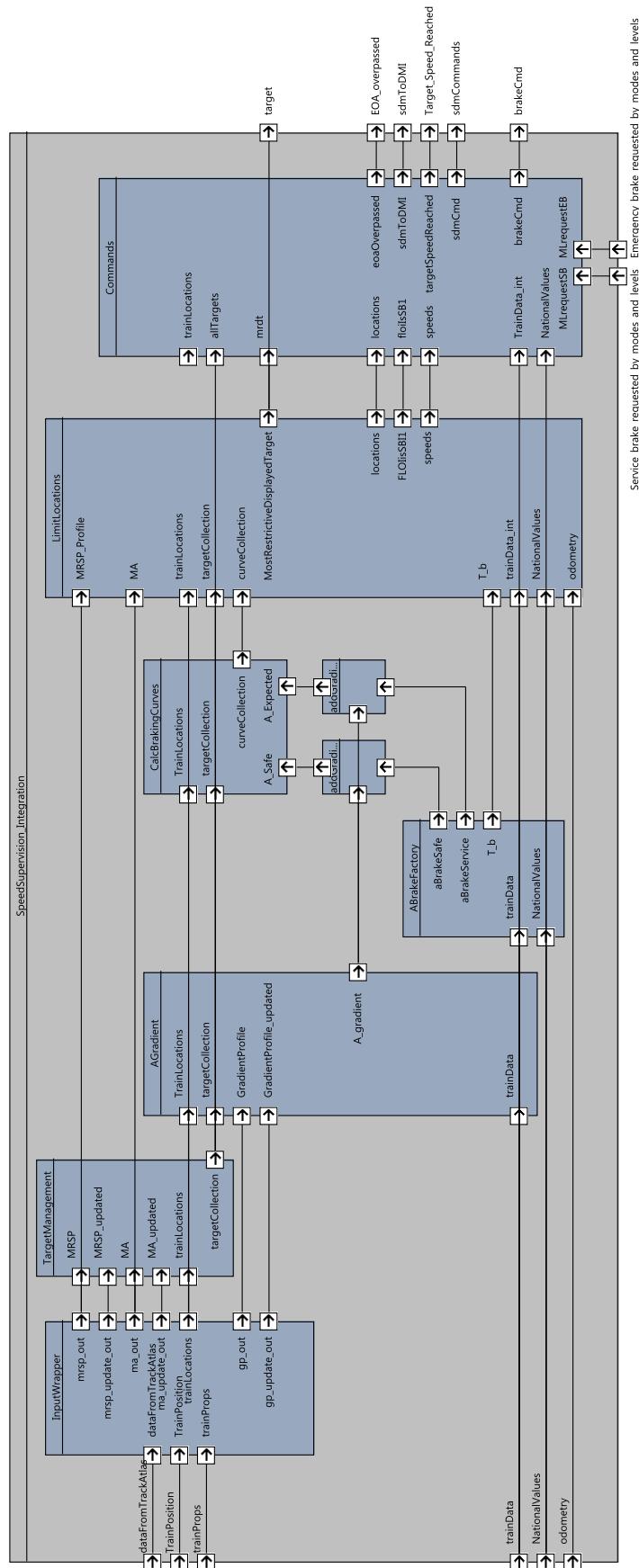


Abbildung 30. SysML Diagramm der F2.7 SpeedSupervision_Integration Komponente.

Die essentiellen Ausgaben der Komponente sind zum einen die Bremseingriffsbefehle und zum anderen die Ausgabe der aktuell zulässigen Geschwindigkeit und Entfernung zum nächsten Bremspunkt über den Bildschirm an den Triebfahrzeugführer. Die Eingangsdaten akkumulieren fahrzeugspezifische Daten, laufende Streckendaten und die Odometrie (siehe Abb. 30).

Im InputWrapper werden einige Signale konditioniert um flexibel mit den iterativen Entwicklungszyklen von openETCS umgehen zu können. Das TargetManagement entfernt passierte Bremspunkte (Geschwindigkeitsreduktionen) und wandelt das Ende der freien Strecke zu einem Bremspunkt. Parallel wird das Bremsmodell anhand der Zugcharakteristika generiert und anschließend mit dem Streckengradienten beaufschlagt. Dieses Zug-Strecken-Modell faltet die Subkomponente CalcBrakingCurves über die Bremspunkte, so dass die Bremskurven entstehen (siehe Abb. 31).

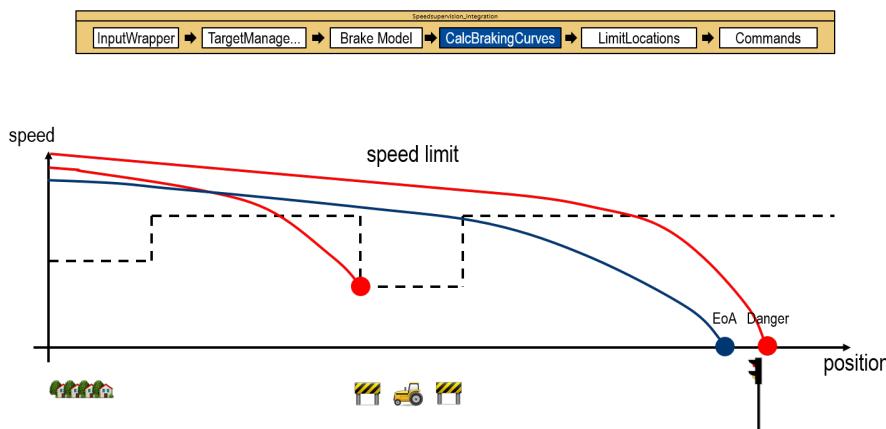


Abbildung 31. Bremskurven zu den jeweiligen Bremspunkten.

Folgend leitet die Subkomponente LimitLocations die verschiedenen Schranken ab (Hinweis, Warnung, Überschreitung usw.). Entsprechend des Vorzustands generiert am Ende die Komponente Commands Bremsbefehle oder die Aufhebung eines Bremseingriffs (siehe Abb. 32).

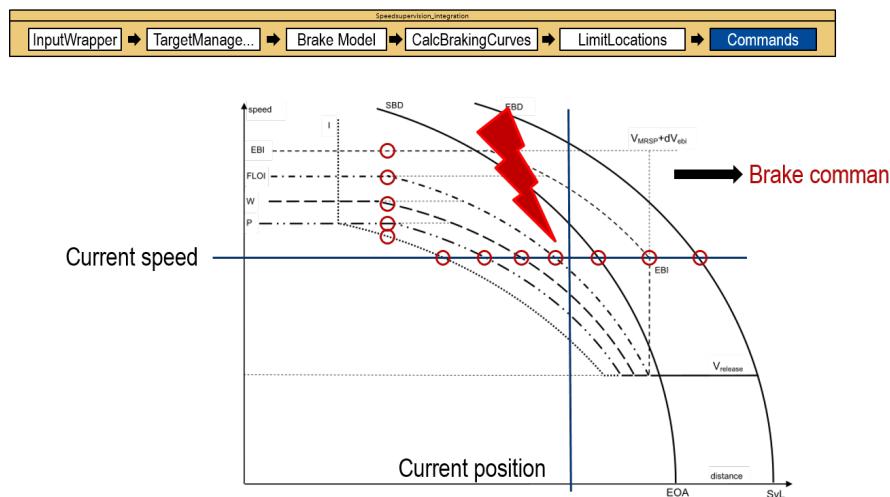


Abbildung 32. Ableitung der W, I und FLOI Kurven in LimitLocations, für Ausgangsbeschaltung durch die Sub-Komponente Commands.

Die Besonderheit dieser Art der Berechnung ist die Abschnittsweise Generierung der Bremskurve (siehe Abb. 33). Übertrieben dargestellt führen bestimmte Streckenparameter dazu, dass eine

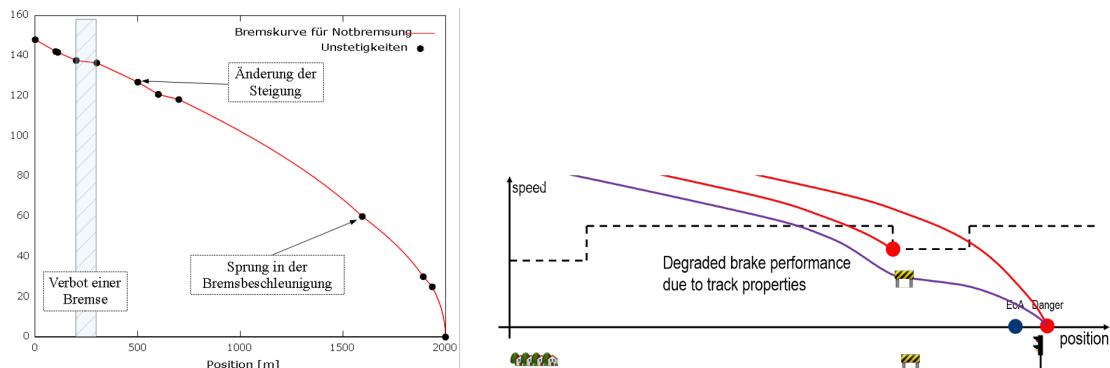


Abbildung 33. Nicht-lineares, abschnittsweises Modell einer Bremskurve und deren Einfluss auf den frühesten Brems-Eingriffspunkt.

Bremsung deutlich früher eingeleitet werden muss und eine vorhandene Streckengeschwindigkeitsbeschränkung nicht gesondert zur Geltung kommt, da sie bereits außerhalb des zulässigen Bereichs liegt (siehe Abb. 33).

In Systemen mit hohen Sicherheitsanforderungen wird möglichst auf dynamische Speicherbelegung verzichtet, da sie in der Regel nicht deterministisch ist. Statt dessen werden auch große Variablen statisch im Stack angelegt. Aufgrund der hohen potentiellen Variabilität der Strecken- und Zugparameter führt dies besonders in der Komponente F2.7 SpeedSupervision_Integration zu einem hohen Speicherbedarf. Einer Optimierung entgegen dieser Problematik ist in einer späteren Iteration nachzugehen. Hierfür müssen zunächst die Mindestanforderungen an eine Zielplattform festgelegt werden.

Neben dem finalen SCADE Modell der Komponente wurden Implementierungen auch in System-C und Matlab umgesetzt. Hintergründe waren hier unter anderem die diversitäre Programmierung und Konzepte zur frühzeitigen Plattformanforderungsanalyse. Diese Ansätze wurden jedoch zugunsten eines größeren funktionalen Umfangs des SCADE Modells nicht weiter vertieft.

6.4.5.8 F2.8 Provide_Position_Report

Die Komponente F2.8 Provide_Position_Report (vgl. Abb. 34) modelliert das Erstellen eines Position-Reports nach den in ERA Subset-026, Kapitel 3.6.5 (Provide Position Report) beschriebenen Anforderungen. Der Position-Report wird als Message 132 an das RBC verschickt. Angestoßen wird das Erstellen eines Positionreports, wenn

1. mindestens einer der Trigger aus dem Position-Report-Parametern (Paket 58) wahr wird oder
2. mindestens ein Ereignis, dass das Senden eines Position-Reports aktiviert, eintritt.

Das Modell verarbeitet die Position-Report Parameter sowie Ereignisse der Strecke, um mögliche Trigger zum Erstellen des Position-Reports zu identifizieren. Falls ein Position-Report erstellt werden soll, wird aus den benötigten Daten die Nachricht 132 erstellt.

Im Modell wird der Position-Report in jedem Takt neu erstellt, so dass er anderen Komponenten jederzeit zur Verfügung steht. Dies ist notwendig, da die Hauptinformation des Position-Reports (d. h. Paket 0 bzw. 1) auch Bestandteil anderer Nachrichten ist, die von anderen Komponenten erstellt werden.

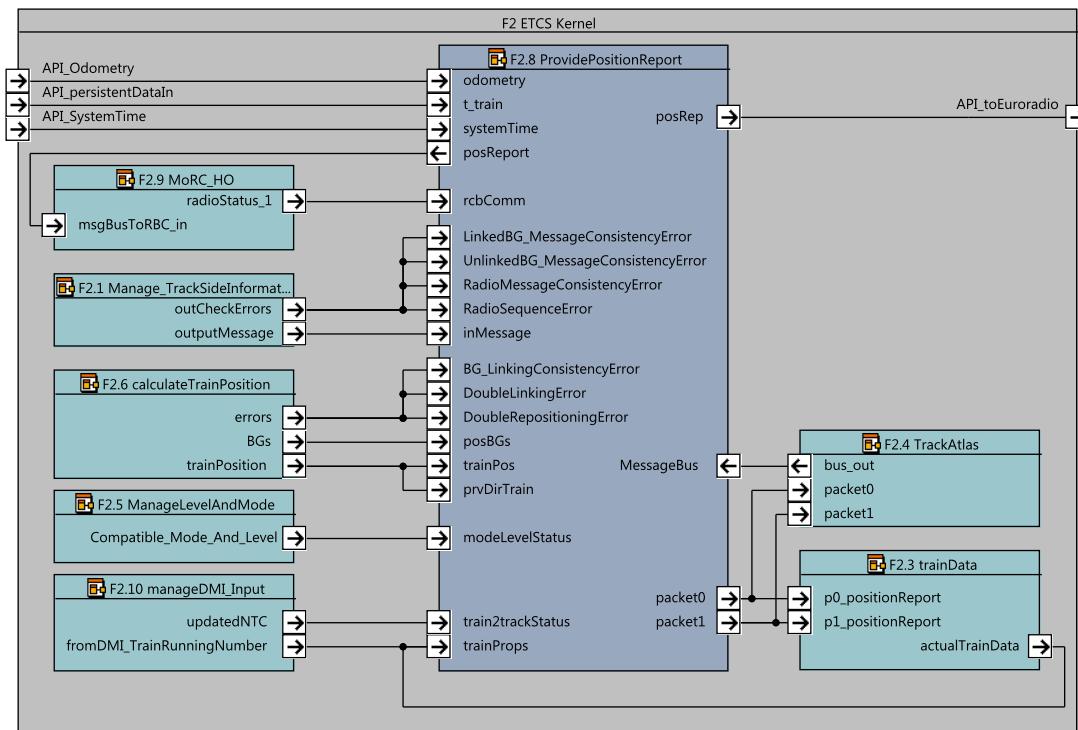


Abbildung 34. F2 ETCS Kernel SysML Diagramm mit Fokus auf die Komponente F2.8 Provide_Position_Report.

Provide_Position_Report speichert die Liste der acht letzten Balisengruppen, für die ein Position-Report verschickt wurde. Als Reaktion auf einen Position-Report sendet das RBC die Position der Balisengruppe, bezüglich der, der Position-Report geschickt wurde. Provide_Position_Report aktualisiert mit dieser Information die Position der entsprechenden Balisengruppe in der Liste.

Das SysML Diagramm in Abbildung 34 zeigt wie sich die Komponente Provide_Position_Report in den ETCS Kernel des openETCS OBU Modells einbettet. Neben Odometriedaten und der Fahrzeugposition erhält die Komponente auch sämtliche Fehler als Eingangsdaten. Diese Fehler werden als Teil des Position-Reports an das RBC übermittelt.

6.4.5.9 F2.9 MoRC_HO

Die Komponente F2.9 MoRC_HO implementiert das Radio Session Management (Management of Radio Communication, MoRC) sowie den Wechsel (RBC/RBC Hand Over, HO) des steuernden Radio Block Center (RBC) von einem übergebenden RBC auf ein übernehmendes RBC auf Fahrzeugseite. Die realisierte Funktionalität umfasst ERA Subset-026, Kap. 3.5 (Management of Radio Communication), Kap. 3.15 und 5.15 (RBC/RBC Handover).

Das Session Management (MoRC) wurde als erste Komponente im Projekt zu einem sehr frühen Zeitpunkt im Zuge der Untersuchungen zur openETCS Tool Chain erstellt, um die Eignung von Werkzeug und Modellierungssprache für die Zwecke von openETCS zu zeigen. Das Session Management zählt zu den schwer verständlich spezifizierten Funktionen; es wurde gerade deshalb ausgewählt, um zu prüfen, ob diese Art der Modellierung (Rapid Prototyping) geeignet ist, die textuelle Spezifikation in eine formale umzusetzen.

Nachdem beide Fragen positiv beantwortet werden konnten und SCADE im Projekt als Implementierungssprache ausgewählt worden war, wurde das vorhandene MoRC-Modell überarbeitet

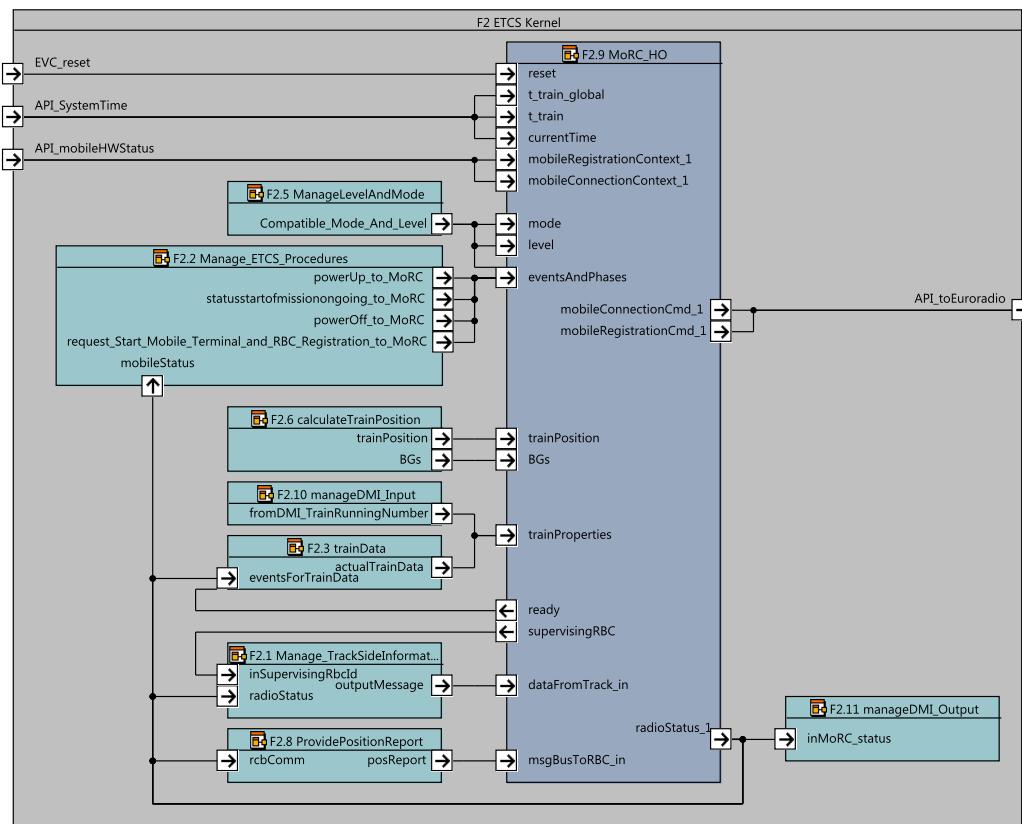


Abbildung 35. F2 ETCS Kernel SysML Diagramm mit Fokus auf die Komponente F2.9 MoRC_HO.

und in das Gesamtmodell des EVC integriert. Auf Basis des am Rapid Prototype erworbenen Verständnisses wurde die Kommunikationssteuerung im Rahmen der Überarbeitung in drei Schichten aufgeteilt:

- Registrierung im Radio Netzwerk
- Radio-Verbindung
- Radio Session

Da das Management of Radio Communication (MoRC) eng mit dem RBC/RBC-Handover-Prozess verknüpft ist, wurde diese Funktion hinzugefügt. Beides – RBC/RBC Handover und zwei Instanzen von Management of Radio Communication – befindet sich in einer einhüllenden Komponente MoRC_HO mit der folgenden Funktionalität:

- Steuerung von bis zu zwei Mobile Modems.
- Registrierung der Mobile Modems im Radio Netzwerk.
- Verbindungsauflaufbau und -abbau zu ein oder zwei RBCs.
- Radio Session Auf- und -abbau für bis zu zwei RBCs.
- Steuerung des Übergangs (Handover) zwischen zwei RBCs.
- Umschaltung des EVC-Radio-Ausgangsdatenstroms zwischen zwei RBCs.
- Unterstützung der Umschaltung des EVC-Radio-Eingangsdatenstroms zwischen zwei RBCs.

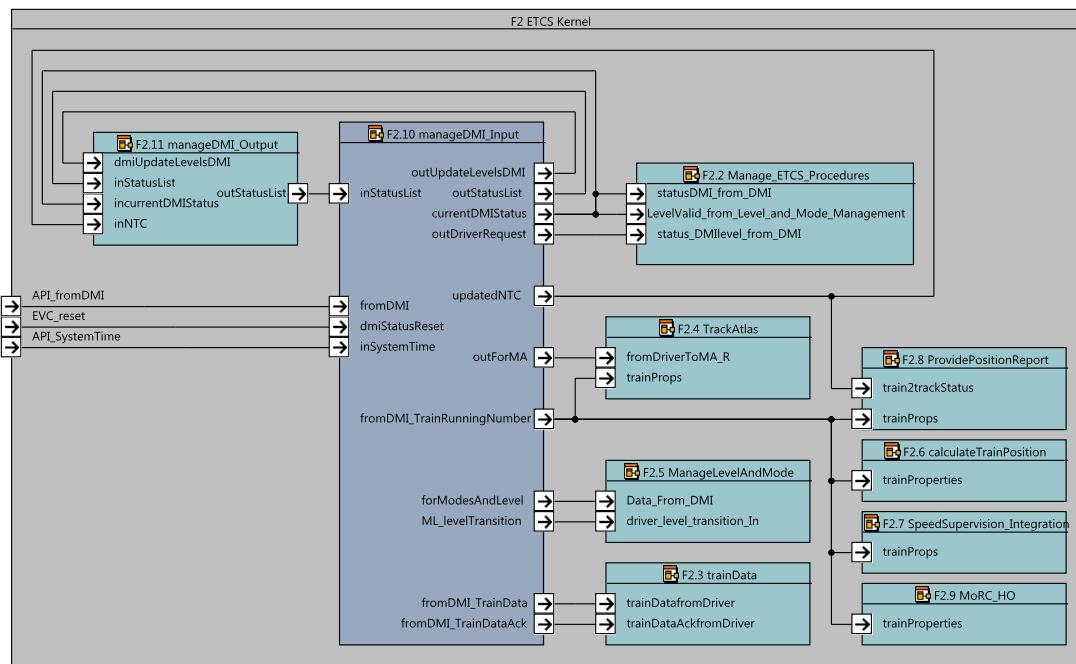


Abbildung 36. F2 ETCS Kernel SysML Diagramm mit Fokus auf die Komponente F2.10 ManageDMIInput.

6.4.5.10 F2.10 manageDMI_input

Über das Driver Machine Interface (DMI) erhält der Triebfahrzeugführer Einsicht in die Funktionen des EVC. Das Display des DMI zeigt dem Triebfahrzeugführer wichtige Parameter, wie gefahrene Geschwindigkeit und Informationen über die voraus liegende Strecke. Zudem muss der Triebfahrzeugführer über das DMI Eingaben an den EVC übermitteln, wie z. B. die Bestätigung der Fahrzeugdaten während der Start-of-Mission Prozedur.

Die Schnittstelle zwischen DMI und EVC ist zur Zeit noch nicht standardisiert. Im Rahmen des openETCS Projekts wurde daher die in Arbeitspaket 5 (openETCS Demonstrator) dokumentierte Schnittstelle des Projektpartners ERSA verwendet.

Um eine bessere Testbarkeit des Gesamtmodells zu erreichen, wurde im Projekt frühzeitig begonnen, ein eigenes DMI als Software Anwendung zu entwickeln. Dieses openETCS DMI ist über die EVC DMI Input und Output Management Funktionen mit dem EVC verbunden. Die Schnittstelle basiert auf Paketen, welche in jedem Zyklus bei Bedarf zwischen DMI Manager und DMI ausgetauscht werden. Die beiden Systeme sind durch eine TCP/IP Verbindung gekoppelt.

Die DMI Manager wurden so entwickelt, dass ein Wechsel auf das Protokoll eines anderen DMI Herstellers – oder etwa eines später verfügbaren offiziellen ERA Standards – leicht umzusetzen ist. In der DMI zu EVC Schnittstelle stellt der DMI-Manager folgende Funktionen bereit:

- Initialisieren der Zustände und Daten bei Systemstart und auf Anforderung (z. B. bei Level Wechsel).
- Synchronisierung der (DMI-) Ein- und Ausgaben bei der Start-of-Mission Prozedur.
- Abbildung der DMI Eingaben auf Trigger für andere EVC Funktionen. Die wichtigste Schnittstelle ist dabei das Mode- und Level Management, da wesentliche Änderungen bzgl. Mode oder Level durch den Triebfahrzeugführer bestätigt werden müssen.

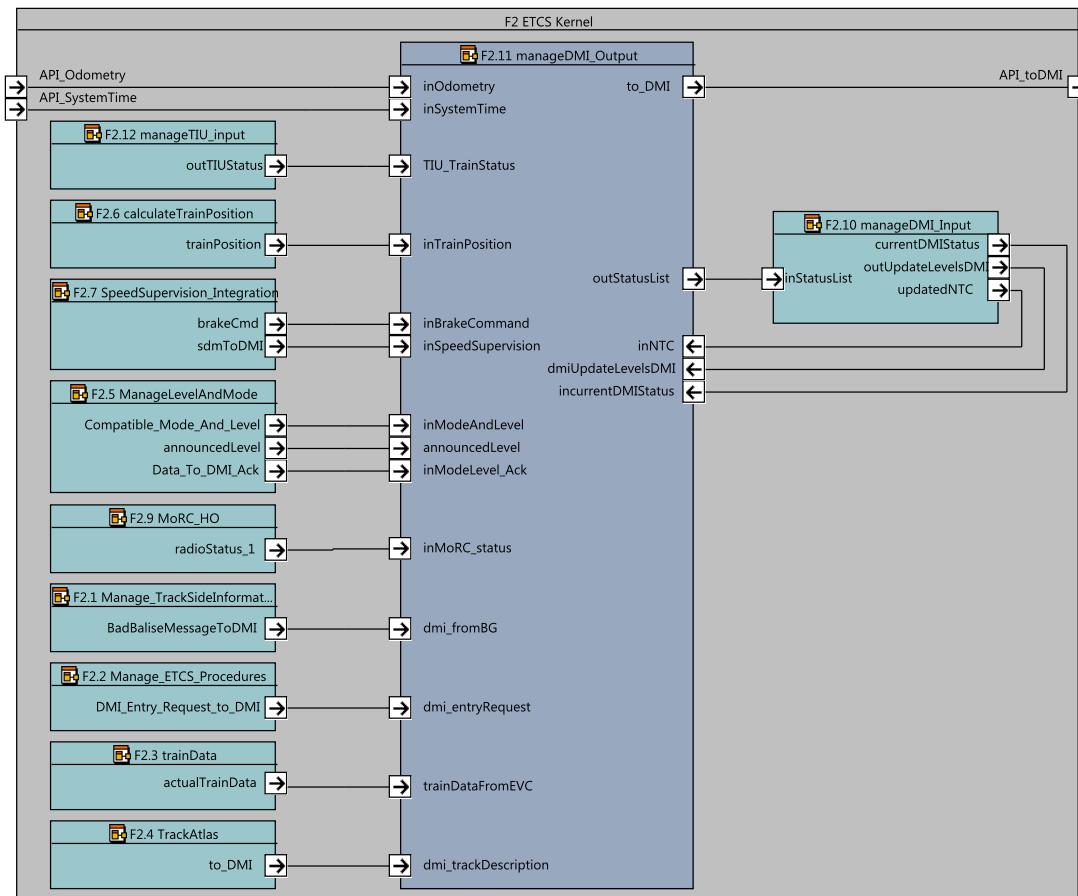


Abbildung 37. F2 ETCS Kernel SysML Diagramm mit Fokus auf die Komponente F2.11 ManageDMIOutput.

6.4.5.11 F2.11 manageDMI_output

Analog zu dem DMI Input Manger ist der Output Manger für die Abstraktion der Schnittstelle vom EVC zum DMI zuständig. Die Schnittstelle basiert ebenfalls auf Paketen, die, soweit notwendig, in jedem Zyklus vom EVC zum DMI gesendet werden können und stellt folgende Funktionen bereit:

- Initierung des Versionsabgleich zwischen EVC und DMI bei Systemstart.
- Senden des regelmäßigen Statusberichtes zum DMI. Dieser Bericht wird spätestens alle 300 Zyklen gesendet oder bei Änderung von wesentlichen Daten, die unmittelbar am DMI angezeigt werden müssen. Beispiele dafür ist die Anzeige von Mode und Level nach Wechsel oder die Anzeige der Notbremse.

Für den Triebfahrzeugführer spielen zdem Textmeldungen eine wichtige Rolle. Diese Meldungen werden dem Triebfahrzeugführer am Display angezeigt und können bei Bedarf quittiert werden. Die Spezifikation kennt sowohl standardisierte Texte als auch die Möglichkeit, freie Texte anzuzeigen. Ein Beispiel für Textmeldungen sind wiederum Mode- und Level-Wechsel, die an den Triebfahrzeugführer kommuniziert werden und von diesem quittiert werden müssen. Textmeldungen werden in den DMI-Managern durch folgende Funktionen unterstützt:

- Speichern von Textmeldungs-Anforderungen.
- Sicherstellung, dass Meldungen nur einmalig am DMI angezeigt werden.

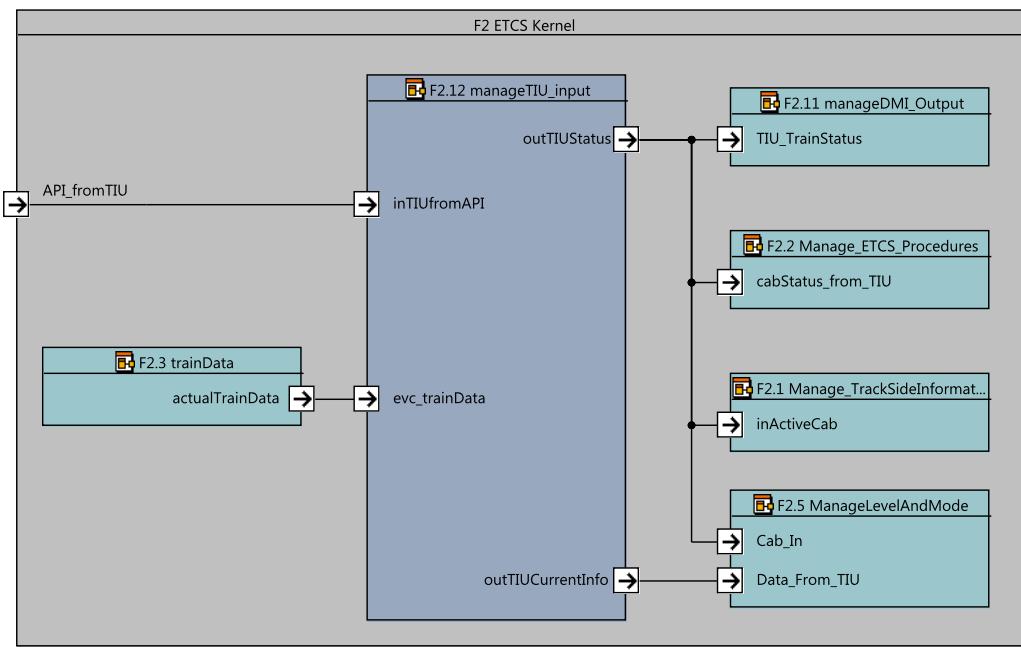


Abbildung 38. F2 ETCS Kernel SysML Diagramm mit Fokus auf die Komponente F2.12 ManageTIUInput.

- Zuordnung von Quittierungen vom DMI zu den EVC-Funktionen und Triggern von Reaktionen auf Antworten vom DMI.
- Überwachung der Antwort vom DMI.

6.4.5.12 F2.12 manage TIU_input

Die ETCS Train Interface Unit (TIU) definiert die Schnittstelle zwischen dem Fahrzeug und der OBU. Die Implementierung der Schnittstelle im openETCS OBU Modell folgt der Spezifikation in ERA Subset-034 (Train Interface FIS). An Stellen, an denen diese Lücken aufwies, wurde die Umsetzung aus einer Implementierung des Herstellers Alstom übernommen.

Die TIU ermöglicht sowohl Aktionen des Triebfahrzeugführers als auch Steueraktionen durch den EVC, wie zum Beispiel:

- Öffnen und Schließen des Führerstandes und des Fahrpultes.
- Auswahl der aktiven Kabine durch den Triebfahrzeugführer.
- Bremsfunktionen.
- Fahrtrichtungs-Wechsel.
- Antrieb.
- Steuern des Pantographen.

Die für den EVC relevanten Informationen dieser Schnittstelle werden über den TIU_Input_Manager von der TIU zum EVC gesendet. Die Schnittstelle ist wiederum über Pakete definiert. Daten können in jedem Zyklus geändert werden. Die Komponente TIU_Input_Manager stellt dabei folgende Funktionen zur Verfügung:

- Initialisierung der Daten bei Start des Systems.
- Aktualisierung von EVC-Daten nach Änderungen durch die TIU.

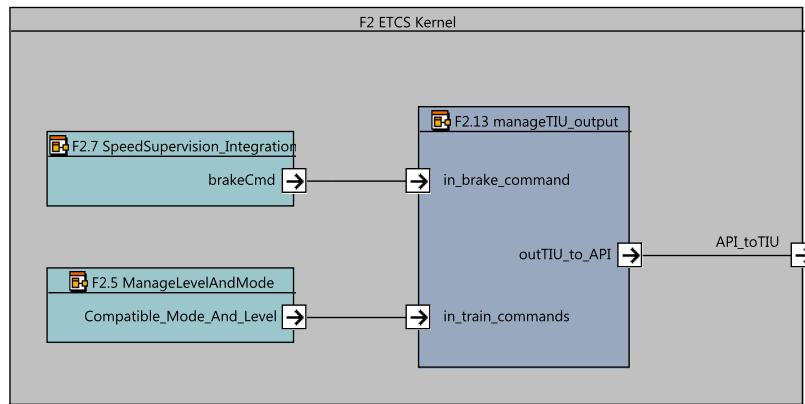


Abbildung 39. F2 ETCS Kernel SysMLDiagramm mit Fokus auf die Komponente F2.13 ManageTIUOutput.

- Bereitstellung einer Kopie von Daten für andere OBU Komponenten.
- Abstraktion der EVC internen Darstellung von Informationen an der Schnittstelle.

Die Schnittstelle zwischen EVC und TIU ist im openETCS System Fahrzeug als TCP/IP Verbindung realisiert worden. Dies geschah wohl wissend, dass diese Lösung für einen realen Betrieb mit Sicherheitsanforderungen an die Performanz-Grenzen stoßen kann.

6.4.5.13 F2.13 manage TIU_output

Auf der Ausgabeseite zur TIU werden die Steuerkommandos des EVC als Trigger entgegengenommen und in ein TIU Kommando abgebildet. Die Ausgabe ist wiederum in Paketen organisiert, die immer dann gesendet werden, wenn sich der Wert eines Parameters geändert hat. TIU Input und Output sind funktional eng miteinander abgestimmte Schnittstellen. Die wichtigsten Funktionen in unserem Anwendungsfall sind hierbei diejenigen zur Steuerung des Bremssystems und des Antriebs:

- Kommando zur Steuerung der Betriebsbremse.
- Kommando zur Steuerung der Zwangsbremse.
- Kommando zum Abschalten des Antriebs.

6.4.6 Modul F1 Receive Information from Trackside

Um die im Arbeitspaket 3 entwickelte ETCS On-Board Unit testen und präsentieren zu können, war neben der Entwicklung des OBU Modells die Modellierung der Strecke Utrecht Amsterdam ein weiterer Schwerpunkt. Das Streckenmodell selbst bildet die Balisen und zugehörige Telegramme punktgenau ab und wird ergänzt durch das Radio Block Center (RBC) Modell. Dieses RBC simuliert das RBC der Strecke Utrecht-Amsterdam.

Die Struktur des RBC ist im Modell in Abbildung 40 dargestellt. Auch beim RBC wurde der bereits zuvor erwähnte openETCS Meldungsbus für die Kommunikation eingesetzt. Die Schnittstelle zwischen RBC und EVC im openETCS System wurde als TCP/IP Schnittstelle realisiert. Die im realen ETCS System vorhandenen Verzögerungen über die Luftschnittstelle wurden durch künstliche Wartezyklen nachgebildet.

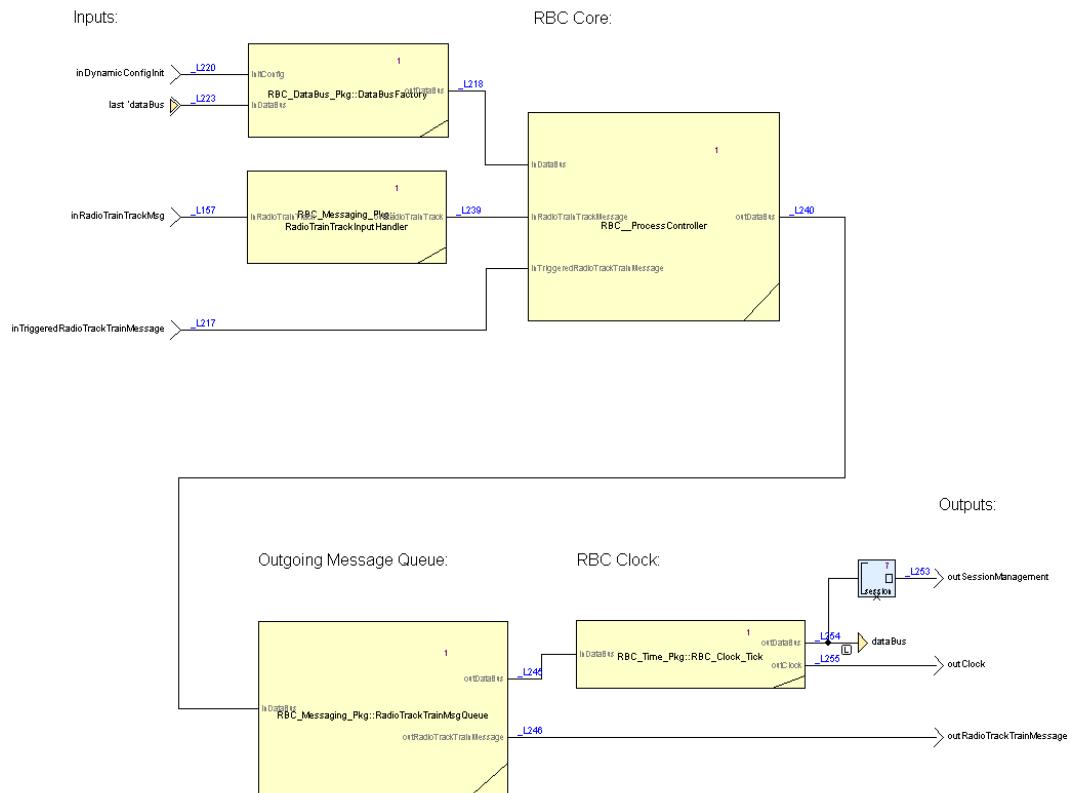


Abbildung 40. Struktur des Moduls F1 Receive Information from Trackside.

6.4.7 Modul F6 DMI Controller

Ebenso wie das Streckenmodell ist die Schnittstelle zum Triebfahrzeugführer (DMI) für das Testen und die Präsentation des EVC erforderlich. Wie bereits erwähnt ist diese Schnittstelle im Gegensatz zu den meisten anderen ETCS Schnittstellen nicht standardisiert. Somit war die Verwendung eines bestehenden industriellen DMI im openETCS Projekt nicht möglich und es wurde im Rahmen des Projekts ein eigenes DMI Modul entwickelt.

Für die Entwicklung des openETCS DMI wurde das Werkzeug SCADE Display eingesetzt. Das Modul F6 selbst besteht aus dem DMI Controller (siehe Abbildung 41) und dem DMI Display. Letztere Komponente wurde dem openETCS Projekt vom SCADE Hersteller Esterel als Modell zur Verfügung gestellt.

Der DMI Controller implementiert auf der einen Seite die Schnittstelle zum EVC. Daraus leitet der Controller ein Abbild des Fahrzeugs mit den wesentlichen Informationen für die Darstellung der voraus liegenden Strecke (die so genannte Planning Area) und weiteren sicherheitskritischen Parametern (wie der Geschwindigkeit) ab. Diese Daten werden in aufbereiteter Form zur Ansteuerung der Display Komponente verwendet.

Die Display Komponente realisiert neben der Darstellung der grafischen Oberfläche auch die Entgegennahme von Eingaben durch den Triebfahrzeugführers. Abbildung 42 zeigt die zugehörige DMI Benutzeroberfläche.

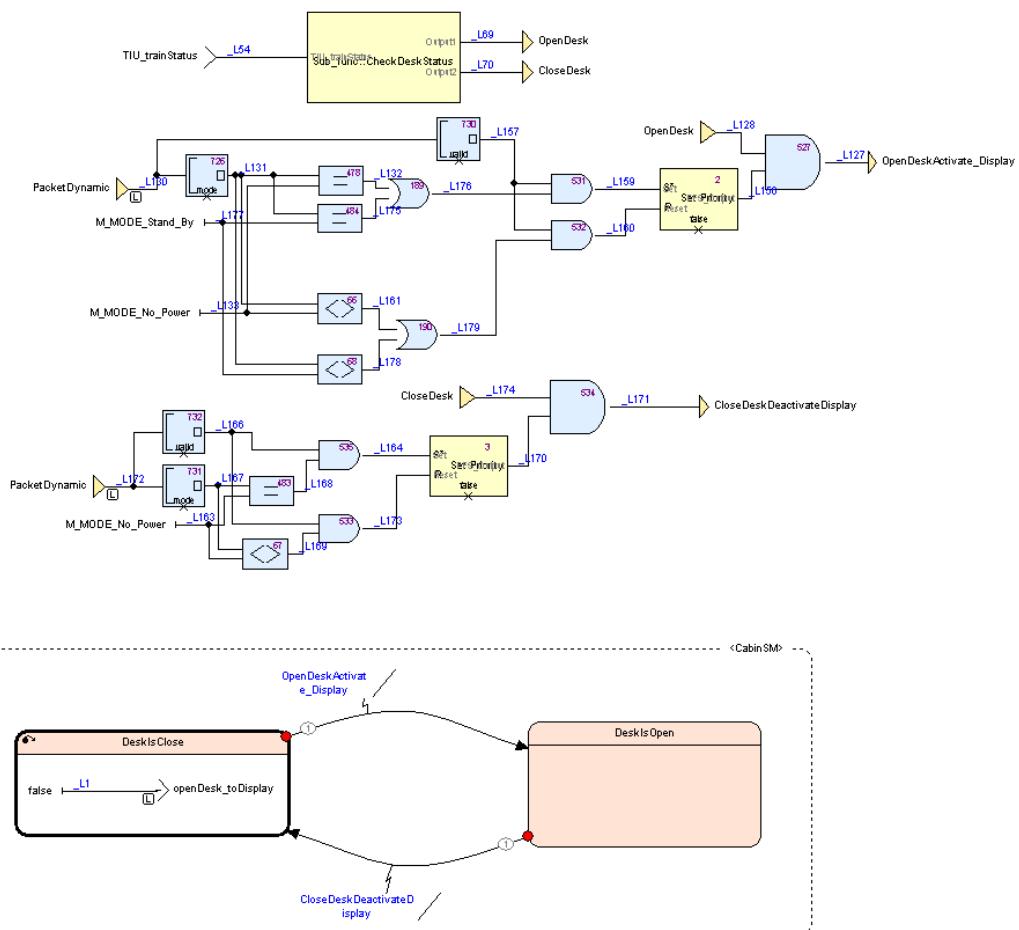


Abbildung 41. Struktur des Moduls F6 DMI Controller.



Abbildung 42. Benutzeroberfläche des openETCS DMI mit Planning Area (rechts).

6.5 Arbeitspaket 4 – Validierungs- und Verifikationsstrategie

6.5.1 Überblick über Verifikation und Validierung

Ziel der Arbeiten

Das Arbeitspaket 4 beschäftigte sich mit der Thematik der Verifikation und Validierung (V&V). Ziel war es, geeignete Mittel dafür bereitzustellen und konkret auf Entwicklungsergebnisse anzuwenden.

Verifikation

Im Sprachgebrauch der Bahnsystementwicklung bezeichnet *Verifikation* die Überprüfung dessen, was in einem Entwicklungsschritt produziert wurde. Gemäß der CENELEC 50128:2011 [26] betrifft dies neben Formalitäten und Nachverfolgbarkeit in der Regel Eigenschaften wie Korrektheit, Vollständigkeit und Konsistenz der Ergebnisse. Ein Beispiel einer Verifikationsaufgabe ist die Kontrolle, dass die Definition von SW-Architektur und -Design, ein Resultat der Phase 4 nach dem Entwicklungsprozess [18], alle Anforderungen, die in Phase 3 für die SW definiert wurden, Komponenten der SW zuweist (Vollständigkeit).

Ausgangspunkt der Entwicklung und somit auch Eingangsdokument für die Verifikation ist die ETCS System Requirements Specification (SRS) [11]. Die Anforderungen der SRS wurden unter verschiedenen Aspekten formalisiert und sind in die ausgearbeiteten Systemanforderungen eingeflossen. Zentrale Beschreibung für die Aufteilung der Anforderungen auf Module der SW ist die openETCS Architektur und Design Spezifikation [22]. Dass die Aufteilung selber in einem agilen Prozess geschah, der weitgehend von den Entwicklern der Module getrieben wurde, ist für die Verifikation von geringer Bedeutung – entscheidend ist der finale Stand mit den dokumentierten Bezügen der zugeteilten Anforderungen aus der SRS. Dagegen wurden dann die SCADE-Modelle [28] geprüft.

Validierung

Ergänzend zur Verifikation hat die *Validierung* bei der Bahnsystementwicklung zum Ziel, die Übereinstimmung des Endresultates der Entwicklung mit den Nutzeranforderungen zu zeigen. Nur im unrealistischen Idealfall könnte man dies aus der Überprüfung der einzelnen Schritte in der Verifikation folgern. Eine umfassende Überprüfung der Gesamtfunktionalität am Ende ist in der Praxis unverzichtbar.

Wenn eine modellbasierte Entwicklung, wie im Projekt openETCS, ausführbare Zwischenresultate erzeugt, lassen sich auch schon vorzeitig Validierungen durchführen. Diese ersetzen nicht die abschließenden Überprüfungen, helfen aber, Schwächen in der Entwicklung frühzeitig zu erkennen und den Korrekturaufwand zu reduzieren. Solche Vorab-Validierungen ließen sich z. B. mit den SCADE-Modellen durchführen.

Die finale Validierung in openETCS setzte auf den Schnittstellenspezifikationen der Technical Specification for Interoperability – Control Command and Signaling (TSI-CCS) [27]. Diese Spezifikationen liegen als Functional Interface Specification (FIS) oder Form Fit Function Interface Specification (FFFIS) vor. Die Herausforderung bei der Validierung in openETCS bestand darin, den openETCS EVC so nah wie möglich an einer bereits spezifizierten ETCS-Schnittstelle zu validieren. Hierzu wurde vom DLR-Labor die Hardwareebene der Balisenschnittstelle im openETCS EVC nach Subset-036 mit einer Simulation ersetzt (siehe Abbildung 54 auf Seite 88).

Über diese Schnittstelle wurde der openETCS EVC mit den bitcodierten Balisennachrichten beaufschlagt. Bei dieser Beaufschlagung wurden die Reaktionen des openETCS EVC manuell gegen die Akzeptanztests [41] verglichen.

Testszenarien

Als Anwendungsfall für die Validierung und einige Verifikationen wurde die Strecke Amsterdam-Utrecht ausgewählt. Hier wurden zugrunde liegende betriebliche Abläufe verwendet, um die Funktionalität in Use Cases zusammen zu fassen und Testszenarien zu erstellen (vergleiche Kapitel 6.4.2).

Darstellung der Aktivitäten

Die Darstellung der Arbeiten orientiert sich an ihrer Einordnung in den Entwurfsprozess nach [18]. Bei den einzelnen Aktivitäten in den Phasen sind in den Überschriften die Titel des SRS-Abschnittes nach [11] angegeben, auf die sich die jeweiligen Anforderungen zurückverfolgen lassen (z. B.: „Management der Radiokommunikation“).

Die Aktivität muss dabei nicht immer hauptsächlich darauf abgezielt haben, ein Entwicklungsartefakt zu verifizieren oder validieren. Es kann auch das Hauptinteresse darin bestanden haben, die Anwendbarkeit einer Methode oder eines Werkzeugs zu evaluieren.

Referenzen auf den openETCS Entwicklungsprozess sind ggf. als Liste von Kapitelnummern in Klammern angegeben (z. B.: (3.4.3, 3.4.4, (3.4.5) [18])).

6.5.2 Verifikation und Validierung in der Planungsphase

Der Planungsphase sind im Prinzip diejenigen Aktivitäten zuzuordnen, die sich mit der Auswahl der Werkzeuge und Methoden für V&V beschäftigen. Als Hauptkriterien der Evaluierung wurden dabei

1. die Eignung für die Verwendung bei der Entwicklung eines SIL-4-Systems (Safety Integration Level 4 nach der CENELEC),
2. der Offenheitsgrad im Sinne von Open Proofs und
3. die Nachhaltigkeit mit der Möglichkeit die Arbeiten wieder aufzunehmen

herangezogen.

Es wurden eine ganze Reihe von Ansätzen, Methoden und Werkzeugen erprobt. Diese Diversität reflektiert zum einen die inhärente Varianz der Verifikations- und Validierungsaufgaben und ist zum anderen dem Forschungscharakter des Projektes zuzurechnen, das explizit auch neue Wege explorieren sollte. Während ein wesentlicher Teil der Planungsphase mit der Wahl der SCADe Entwicklungsumgebung [20, 32] im Jahr 2014 abgeschlossen wurde, ist die Weiterentwicklung der Ansätze zur Verifikation, von dieser Entscheidung beeinflusst, fortgeführt worden. Auf eine zeitliche Trennung der Entwicklungsphasen gemäß dem Wasserfallmodell [26] ist zugunsten der agilen Entwicklung verzichtet worden.

Neben diesen Auswahlaktivitäten wurden in dieser Phase Planungsdokumente durch klassische Reviews verifiziert.

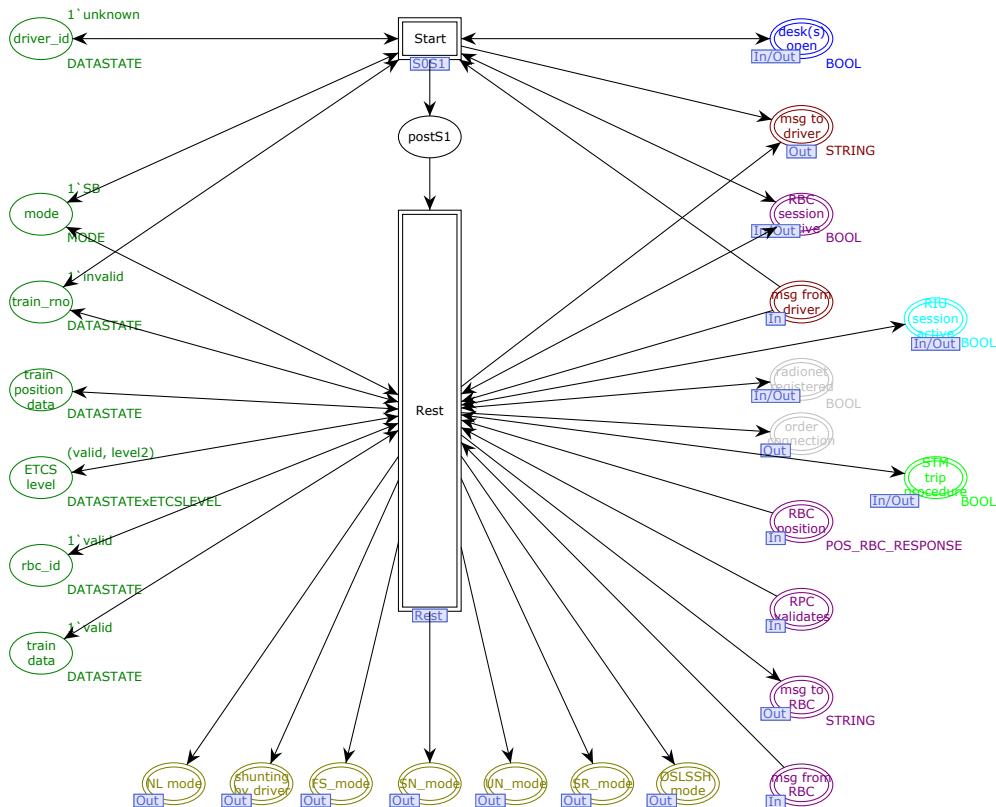


Abbildung 43. CPN Modell der openETCS Prozedur „Start of Mission“ in der On-Board Unit.

6.5.3 Verifikation und Validierung in der Systementwicklungsphase

6.5.3.1 Verifikation von „ETCS-Prozeduren“

Im Rahmen der Ausarbeitung der Spezifikation wurde der Kontrollfluss, der in den ETCS-Prozeduren der SRS (vgl. [11, Kap. 5]) beschrieben ist, mit Coloured Petri Nets (CPN) modelliert (vgl. Abb. 43). Bei dieser Formalisierung der SRS sind einige Inkonsistenzen entdeckt worden, die über die Subset-076 Arbeitsgruppe bei der Europäische Eisenbahnagentur (ERA) eingebracht wurden [3, 37]. CPN haben sich als geeignetes Hilfsmittel zur Formalisierung des Kontrollflusses gezeigt, womit die Präzision und Konsistenz der Spezifikation erhöht werden kann.

6.5.3.2 Verifikation von „Radiokommunikationsmanagement“

Im Rahmen der Ausarbeitung der Spezifikation ist ein vollständig formales Modell vom Management der Radiokommunikation (vgl. [11, Kap. 3.5]) nach der B-Methode in Event-B entstanden. Die Umsetzung der Anforderungen im Modell ist durch Annotationen nachverfolgbar. Das Modell kann zur Validierung oder Verifikation animiert werden. Ausgewählte Sicherheitsanforderungen (vgl. Abb. 44) sind in Prädikatenlogik formalisiert und deren Eigenschaften an dem formalen Modell bewiesen worden [3].

Die Arbeiten zeigen an dem gewählten Beispiel, dass eine Formalisierung der SRS nach der B-Methode möglich ist. Viele der damit zusammenhängenden prozessrelevanten Tätigkeiten, insbesondere formale Verifikation von Sicherheitseigenschaften, werden bereits weitgehend durch Werkzeuge unterstützt. Einige, jedoch nicht alle der Werkzeuge sind Open Source. Zum Teil sind die Werkzeuge bereits ausreichend qualifiziert.

	Name	Description	Source	Target	Link
1	REQ_FMEA_ID_001	The Mobile Terminal shall be safely registered to a Radio Network. Link to general function.		0 > 1 > 6 3.5.6 3.5.6.1 3.5.6.3 3.5.6.5 3.5.6.6 3.5.6.7	
2	REQ_FMEA_ID_002	The driver shall be safely informed of the state of the radio communication (resulting of the different steps: registration of the Mobile Terminal to the Radio Network, establishment of the communication, end of communication). Link to general function.		0 > 1 > 4 3.5.7.2 3.5.7.1 3.5.7 b) if none of its ...	
3	REQ_FMEA_ID_003	If a communication through a Radio Network is active, registration to another Radio Network mustn't be performed.		0 > 1 > 1 3.5.5.6	
4	REQ_FMEA_ID_004	A safety protocol shall be used to performed communication between the Mobile Terminal and the Radio Network.		0 > 1 > 2 3.5.1.1 3.5.2.2	
5	REQ_FMEA_ID_005	If a communication with trackside equipment is active, set-up of safe radio connection with another trackside equipment mustn't beformed. Exception in case of handover with RBC.		0 > 1 > 4 3.5.3.5.2 inv6 (m6_hand_over_RBC) inv7 (m6_hand_over_RBC) inv8 (m6_hand_over_RBC)	
6	REQ_FMEA_ID_006	Communication session with trackside equipment shall be safely established.		0 > 1 > 6 3.5.3.8 3.5.3.7 3.5.3.9	

Abbildung 44. Safety Requirements.

6.5.3.3 Verifikation der „Fahrerlaubnis in Level 2“

Im Rahmen der Ausarbeitung der Spezifikation ist ein formales Modell zur Beschreibung des Mechanismus von Fahrerlaubnisvergaben (vgl. [11, Kap. 3.8]) als zeitbasierter Zustandsautomat in der IF-Sprache [3] erstellt worden (IF: Intermediate Format, in dem verschiedene Modellierungssprachen dargestellt werden können). Es wurde nachgewiesen, dass der Zug innerhalb der von der Fahrerlaubnis gesetzten Grenzen bleibt. Als Nebenprodukt der Formalisierung konnten Mehrdeutigkeiten der Spezifikation in ERA Subset-026 [11] aufgezeigt werden. Zur Validierung wurden Tests generiert und diese auf das Modell und daraus generierte Mutanten in einer JAVA-Simulationsumgebung angewendet. Da die Modellierung nur einen kleinen Ausschnitt des Systemverhaltens auf einem hohen Abstraktionsniveau erfasst, ist das Resultat als prinzipieller Machbarkeitsnachweis entsprechender formaler Analysen anzusehen.

6.5.4 Verifikation und Validierung in der Softwareentwurfsphase

6.5.4.1 Verifikation der Prozedur „Auf Sicht“

Dem Ziel „Open Proofs“ kam bei openETCS eine besondere Bedeutung zu. Zwar ist mit der Entscheidung für SCADE eine proprietäre, nicht quelloffene und nicht gebührenfreie Software als Entwicklungsumgebung gewählt worden. Daneben sind jedoch auch mehr den Anforderungen des Open Proofs Gedankens entsprechende Ansätze verfolgt worden. Hier wurde die Prozedur der Betriebsart „Auf Sicht“ (onsight) (vgl. [11, Kap. 5.9]) mit Classical B auf eine Weise modelliert, so dass daraus ausführbarer C-Code generiert werden konnte. Für das Modell wurden Konsistenzegenschaften (Typ- und Wertebedingungen) sowie einige weitere Invarianten formal bewiesen. Aus dem Modell wurde ein Testsatz abgeleitet, mit dem andere Implementationen der Prozedur (z. B. manuell erstellte oder Modifikationen) auf Übereinstimmung mit dem Modell überprüft werden können.

Die Resultate zeigen die funktionale Eignung des Ansatzes für die modellbasierte Entwicklung des EVC Codes. Allerdings ist die genutzte Software Atelier B nur zum Teil quelloffen. Beispielsweise trifft dies auf einige Beweiser zu, siehe auch [3].

6.5.4.2 Modellbasierte Testgenerierung bei der „Geschwindigkeitsobergrenze“

Der Zug wird in der Betriebsart Full Supervision (FS, zu Deutsch: Vollüberwachung), sofern er noch weit genug von dem Ende seiner Fahrerlaubnis entfernt ist, auf die Einhaltung einer Geschwindigkeitsobergrenze (vgl. [11, Kap. 3.13.9.2]) überwacht.

Es wurde ein Testmodell in SysML entwickelt, das die entsprechenden Anforderungen umsetzt. Dies dient als Referenzverhalten. Mithilfe des Werkzeugs RT-Tester wurden daraus ein Testsatz generiert, welcher die Übereinstimmung einer Implementierung mit dem Verhalten des Testmodells überprüft. Dabei wurde eine neue Methode eingesetzt, die auf einer fehlermodellbezogenen Äquivalenzklasseneinteilung beruht. Es zeigte sich an Mutantentests, dass der Testsatz eine sehr hohe Fehleraufdeckungsquote erreicht. Zum Vergleich wurde über eine LTL-Codierung, ebenfalls mit RT-Tester, ein Testsatz generiert, der alle Testfälle nach Subset-076 enthält. Dieser Testsatz erreicht nur eine erheblich geringere Aufdeckungsquote von 62%. Während der erste Testsatz zu viele Tests für einen Test am realen System enthält, wäre mit seiner Hilfe eine Ergänzung der Tests nach Subset-076 möglich, um die Aufdeckungsquote zu erhöhen. Damit wurde gezeigt, dass die angewendete Testerzeugungsmethode eine Systematisierung und wesentliche Verbesserung gegenüber herkömmlichen Verfahren erreichen würde [6].

6.5.4.3 Modellbasiertes Tests für das „Bremsen auf Zieldistanz“

Um die durch die Fahrerlaubnis gesetzte Zieldistanz (vgl. [11, Kap. 3.13.9.3]) nicht zu überschreiten, wird der Zug, wenn er sich diesem Zielpunkt nähert, auf die Einhaltung einer vorgegebenen Geschwindigkeitskurve vor dem Zielpunkt überwacht.

Ziel der Verifikation war die modellbasierte Generierung von Tests für dieser Funktion. Die besondere Herausforderung für die Testgenerierung besteht in der hybriden Natur der Funktion, da neben diskreten Variablen reelle Werte eine wesentliche Rolle spielen. Der Lösungsansatz kombinierte SysML-Modelle der zeitdiskreten Verhaltensanteile auf neuartige Weise mit parametrischen Bedingungen, die das zeitkontinuierliche, physikalische Bewegungsverhalten beschreiben. Mit dem Werkzeug RT-Tester wurden Tests erzeugt, die das so modellierte Verhalten systematisch überdecken, wobei die Testdaten auch physikalisch sinnvolle Werte enthalten [3].

6.5.4.4 Code Review am Beispiel des SCADE Pakets „trainData“

Nicht alle Verifikationen konnten formalisiert oder automatisiert werden. Ein Code-Review von Teilen der *openETCS Architektur und Designspezifikation* [22] gegen die Spezifikation ist manuell durchgeführt worden. Hierzu wurden die Aufteilung der ETCS-Funktionen, die Schnittstellen der Module und die Umsetzung in SCADE gegen die Anforderungen der Spezifikation [11] verifiziert. Das Ergebnis der Verifikation [8] ist als Abdeckungsanalyse und verfeinerte Funktionsaufteilung in das Designdokument zurückgeflossen.

6.5.5 Verifikation und Validierung in der Software-Komponentenentwurfsphase

6.5.5.1 Verifikation und Validierung der Codierungsregeln

Um die Einhaltung von Codierungsregeln zu prüfen, wie sie für sicherheitskritische elektronische Bahnanwendungen vorgeschrieben sind, wurde manuell für das DataDictionary¹⁴ geschriebener C-Programmcode einer statischen Codeanalyse unterzogen. Relevante Codierungsregeln sind

¹⁴In dem DataDictionary ist die Schicht des Wrappers umgesetzt, welche die bitkodierten Air-Gap Nachrichten in Variablen umwandelt, die vom SCADE Modell gelesen werden können und umgekehrt

in der MISRA-C:2004 [25] und der Mü 8004 [9] aufgeführt. Ein für openETCS adäquater Satz von Codierungsregeln wurde aus der MISRA nach Vergleich mit der Mü 8004 aufgestellt. Diese Regeln waren dann Referenz für die Codeanalyse.

6.5.5.2 Verifikation des Codes für das „Bremsen auf Zieldistanz“

Zur Absicherung der openETCS OBU Komponente F2.7 SpeedSupervision_Integration (siehe Kapitel 6.4.5.7) ist eine separate Testsuite mit dem SCADE Suite Advanced Modeler erstellt worden. Mit benutzerdefinierten Szenarien wurden die Odometrie, Zugdaten und Streckeninformationen nachgebildet und in Vorbereitung der Integration die funktionale Korrektheit durch die Durchführung manueller Tests abgesichert.

6.5.5.3 Verifikation von „Mode und Level Management“

Die Betriebsarten, in denen sich eine ETCS OBU befindet, geben Aufschluss darüber, welche Funktionen aktiviert sind. Zur Absicherung der Funktionalität der openETCS OBU Komponente F2.5 ManageLevelAndMode (siehe Kapitel 6.4.5.5) ist ein separates SCADE-Umgebungsmodell für die Funktion „Mode und Level Management“ (vgl. [11, Kap. 4.5]) erstellt worden. Für dieses Umgebungsmodell sind Szenarien zur Absicherung der Funktionalität und kontinuierliche Prüfungen als Vergleich zwischen Modul-Eingang und Modul-Ausgang modelliert worden. Zugesicherte Eigenschaften des Moduls wurden per Modellprüfung überprüft und eine formale Sicherheitsverifikation durchgeführt [3].

6.5.5.4 Verifikation der Codierung und Decodierung von Datenpaketen

Funktionen, mit denen Datenpakete zum Informationsaustausch zwischen EVC und Streckenkomponenten codiert und decodiert werden, wurden mit Verfahren zur strikten, formalen Verifikation im Sinn der mathematischen Logik verifiziert. Dafür wurden die Funktionen um zu prüfende Vor- und Nachbedingungen in Form von ACSL-Kontrakten [2] ergänzt. Diese so formalisierten Eigenschaften wurden in einer statische Quellcode-Analyse [14] mit FRAMA-C [13] formal verifiziert, siehe Abbildung 45. Im Zuge der Aktivität traten einige Probleme der FRAMA-C Werkzeuge zutage, die an den Entwicklungspartner CEA LIST gemeldet und bereits zum Teil behoben wurden.

Diese Aktivität ist mit den Zielen von Open Proof kompatibel, da es sich bei den FRAMA-C-Werkzeugen weitgehend um quelloffene Software handelt. Eine genauere Darstellung zur Qualifikation der Werkzeuge und der Offenheit der Verfahren findet sich in der Beschreibung zu Arbeitspaket 7 (vgl. Kapitel 6.3.3.1).

6.5.6 Verifikation und Validierung in der Software-Integrationsphase

6.5.6.1 Verifikation und Validierung der openETCS OBU Komponenten

Um die Funktionalität des mit SCADE entwickelten openETCS OBU-Modells zu überprüfen, wurde die Strecke Amsterdam-Utrecht als Referenz gewählt. Es wurden 12 User-Stories als Prüfläufe zusammengestellt [41] und in Sequenzdiagrammen formalisiert. Die Sequenzdiagramme bilden das Verhalten des EVCs und der Umgebung an den Schnittstellen des EVC ab. Diese Sequenzdiagramme sind als Streckenrepräsentation im SCADE Suite Advanced Modeler als Umgebungsmodell modelliert worden. Die Tests mit diesem Umgebungsmodell dienten als Software-Integrationstests und haben das Zusammenspiel der einzelnen openETCS OBU-Module, sowie die funktionale Korrektheit des integrierten Systems im Zusammenspiel mit dem DMI und

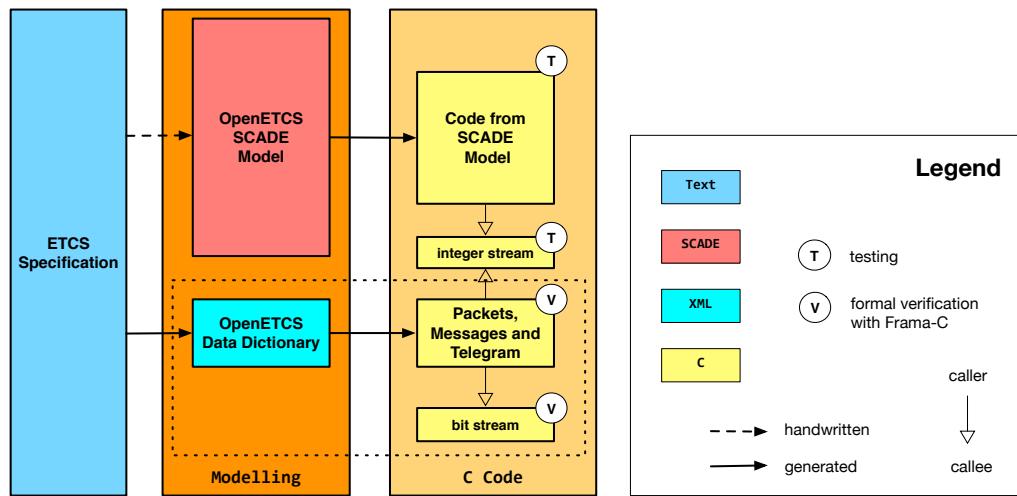


Abbildung 45. Übersicht der Generierung und Absicherung des Quellcodes bei OpenETCS.

den Aufzeichnungen innerhalb des EVC manuell abgesichert. Die Validierung wurde im SCADE Simulator und auf dem unter Win32 kompilierten Quellcode durch das Abfahren der Strecke Amsterdam-Utrecht und der Kontrolle auf dem DMI durchgeführt.

6.5.6.2 Kontinuierliche Verifikation der „Ortsgültigkeit von Balisennachrichten“

Im Rahmen der Validierung der kontinuierlichen Integrationsplattform wurde eine Verifikation der Gültigkeitsbewertung von Balisennachrichten (vgl. [11, Kap. 3.6.2.1]) durchgeführt. Die Formalisierung der Tests und Checks erfolgte mit SCADE. Die Einbettung der Integrationsumgebung erfolgte in der DLR-Laborumgebung [36]. Zur kontinuierlichen Verifikation wurden in einem Zeitabstand von 45 Minuten jeweils die von Arbeitspaket 7 zur Verfügung gestellten Änderungen des Modeling Repository ausgewertet und die Gültigkeit der Änderungen des Design Artefakts im Vergleich zur Testschnittstelle ausgewertet, das EVC Modell kompiliert und die Tests automatisch ausgeführt und bewertet. Die Auswertung der kontinuierlichen Integration und der ausgeführten Tests wurde per E-Mail verschickt, so dass nach jeder Änderung am Design Repository 90 Minuten später die Ergebnisse der kontinuierlichen Integration und abgeschlossenen Tests vorlagen.

6.5.7 Verifikation und Validierung in der Software-Validierungsphase

Zur Validierung des EVC-Modells ist der von SCADE generierte Quellcode auf eine SIL-4 fähigen Hardware von Alstom, ehemals General Electric, portiert worden und gegen die vom DLR getestete Streckenbeschreibung im Subset-076 Format geprüft worden. Die Realisierung dieser Schritte wird im Kapitel 6.6.3 näher beschrieben.

6.5.8 Ergebnisse der Verifikations- und Validierungsaktivitäten

Im Rahmen der Verifikation und Validierung wurden Ergebnisse verschiedener Entwicklungsphasen und Beschreibungsformen untersucht. In Folge dieser heterogenen Ausgangssituation mussten auch verschiedene, mehr oder weniger formalisier- und automatisierbare Verfahren angewendet werden.

Die Verifikations- und Validierungsaktivitäten mussten sich auf die eingeschränkte Verfügbarkeit der Spezifikationen und Verifikationsobjekte einstellen. Als Folge konnten einige zum Projekten-de vorliegende Entwicklungsartefakte nicht mehr untersucht werden. In diesem Berichtsabschnitt werden die Ergebnisse der verschiedenen Ansätze und Evaluationen zusammengestellt.

6.5.8.1 Testgenerierung

Diese Technik (3.4.3, 3.4.4, 3.4.5 [18]) hat, als Teil der modellbasierten Entwicklung, hohes Potential für die Nutzung zur Generierung von Testsätzen und deren Ausführung gezeigt (siehe Kap. 6.5.4.2, 6.5.5.2). Das verwendete Werkzeug (vgl. Abb. 10) hat eine quelloffene Benutzerschnittstelle und eine proprietäre, serverbasierte Verarbeitungskomponente. Eine Verifikation des gesamten EVC-Modells von openETCS sowie eine T2-Qualifikation des Werkzeuges wurden als machbar eingeschätzt, jedoch aufgrund des geringen Zeitbudgets, das im Projekt dafür zur Verfügung gestanden hätte, nicht weiter verfolgt.

6.5.8.2 Modell-Prüfungen und Modell-Verifikationen

Unter Verwendung geeigneter Codegeneratoren ist es möglich, aus hinreichend präzisen Modellen Implementierungen zu generieren. Wenn der Codegenerator verlässlich ist, kann die Verifikation des Codes auf die Modellebene verlagert werden. Dann sind auf Modellebene entsprechende Codierungsregeln (Referenz 3.5.5 in [18]) zu überprüfen, und die funktionale Korrektheit des Modells ist nachzuweisen. Damit wird aus dem traditionellen V-Modell [26] das angestrebte „Y“: Der Code selber braucht nicht explizit betrachtet zu werden.

Da der SCADE-Codegenerator qualifiziert ist, trifft dies auf die SCADE-Modelle in openETCS zu. Die funktionale Korrektheit auf Modellebene nachzuweisen, kann, wie in openETCS zum Teil geschehen, durch Tests erfolgen (Kap. 6.5.6.1, 6.5.5.3). Die Überdeckungsbegriffe können sich dann an der Modell- und nicht der Codestruktur orientieren. Ähnlich wäre eine umfassende Anwendung der Codegenerierung aus B-Modellen (Kap. 6.5.4.1) einzuschätzen.

Da die Nachweisführung für die Korrektheit eines Codegenerators sehr aufwändig ist, ist jedoch mittelfristig kaum mit einer freien Verfügbarkeit (Open Source) derartiger Werkzeuge zu rechnen.

6.5.8.3 Strikte formale Verifikation

Verifikation im Sinne der mathematischen oder rechnergestützten Logik bedeutet, einen Beweis vollständig formal im konsistenten System zu konstruieren. Dies ist die anspruchsvollste Form der Verifikation. Sie setzt Formalisierungen der zu prüfenden Eigenschaften und eine mathematisch präzise Semantik des zu prüfenden Systems voraus. Demzufolge ist strikte Verifikation nicht universell auf alle Entwicklungsschritte gleichermaßen anwendbar. Auch der Aufwand, der z. B. schon mit dem manuellen Erstellen der Prüfbedingungen verbunden ist, erlaubt derzeit keine allumfassende Verwendung dieser Verfahrens.

Prinzipiell ist es jedoch möglich, auch komplett Implementierungen, also Programme einer gängigen Programmiersprache, formal zu verifizieren. Wie auch durch die Anwendung von FRAMA-C im Projekt openETCS demonstriert (siehe Kap. 6.5.5.4), lässt sich dies heutzutage mit vorhandener Werkzeugunterstützung realisieren.

6.5.8.4 Reviews und Teilformalisierungen

Auf Reviews (3.5.5 [18]), also die klassische Begutachtung durch Fachleute, kann trotz der benannten formalen und automatisierten Verfahren nicht immer verzichtet werden. Sie sind insbesondere dann alternativlos, wenn Ein- oder Ausgaben von Entwicklungsschritten nicht formal vorliegen. Um diese Lücken zu schließen, wäre eine stringente Entwicklungsstruktur mit präzise definierten und kontrollierten Artefaktypen notwendig. Die Erfahrung im Projekt hat gezeigt, dass es zumindest sehr schwierig ist, dies in der Praxis umzusetzen.

Reviews sind in ihrer Aussagekraft nur schwer einschätzbar, da die Ergebnisse maßgeblich von der Art und Komplexität des zu begutachtenden Systems sowie den Sachkenntnissen und Erfahrungen des Begutachtenden abhängt. Im Laufe der Arbeiten von openETCS zeigte sich, dass das Subset-026 selbst ein treffendes Beispiel ist. Durch partielle Modellierung und Analyse (3.1.2, 3.1.1 [18], siehe Kap. 6.5.3.1) konnten Inkonsistenzen in der informellen SRS aufgedeckt werden, die im Rahmen der Reviews vor deren Veröffentlichung nicht gefunden wurden. Obwohl diese Formalisierung der Anforderungen nicht im strengen Sinne komplett war, hat sie sich als tauglich im Sinne der CENELEC [26] für frühe Schritte der Entwicklung herausgestellt.

6.5.8.5 Zusammenfassung

Zusammenfassend lässt sich sagen, dass:

1. Verschiedene Verfahren und Ansätze zur Verifikation beziehungsweise Validierung auf die Produkte aus den unterschiedlichen Entwicklungsphasen angewandt und die Ergebnisse bewertet wurden. Insbesondere die modellbasierte Generierung von Tests und die Verlagerung der Codeverifikation auf die Modellebene sind Ansätze mit hohem Potential für Verbesserungen des Entwurfsprozesses in der Praxis. In der Durchgängigkeit der Entwicklungsschritte sind offene Lücken identifiziert worden, die es zu füllen gilt.
2. Validierungsschritte im Sinne der CENELEC durchgeführt werden können, obwohl der zurückverfolgbare Übergang (Traceability) der Artefakte zwischen den Werkzeugen größtenteils noch manuell erfolgen muss. Um die Nachverfolgbarkeit dieses Übergangs automatisiert nachweisen zu können, würden eine komplette Werkzeugkette und Formalismen von der Systemspezifikation bis hin zum Quellcode benötigt.
3. Die klassische Verifikation (3.5.2 [18]) als nützlicher Schritt in dem Entwicklungsprozess voraussichtlich nicht so leicht gänzlich zu ersetzen ist.
4. 80% der für die Strecke Amsterdam-Utrecht vorgeschriebenen betrieblichen Szenarien am Modell geprüft worden sind, siehe Kapitel 6.5.6.1.

6.5.9 Begutachtung der Ergebnisse

Zur Bewertung der Konformität nach CENELEC EN50128:2011 [26] der Gesamtergebnisse des Projektes wurden Stichproben der Projektdokumente für eine unabhängige Begutachtung durch Experten ausgewählt. In der Begutachtung wurde die Konformität von

- der herstellerunabhängigen generischen openETCS EVC Software und
- dem Softwareentwicklungsprozess

beurteilt. Zur Kommunikation der Ergebnisse wurde ein Assessment-Workshop in Nürnberg abgehalten. Die Ergebnisse sind in dem Bericht „Independent Assessment according to the Standard EN50128:2011“ [42] zusammengefasst.

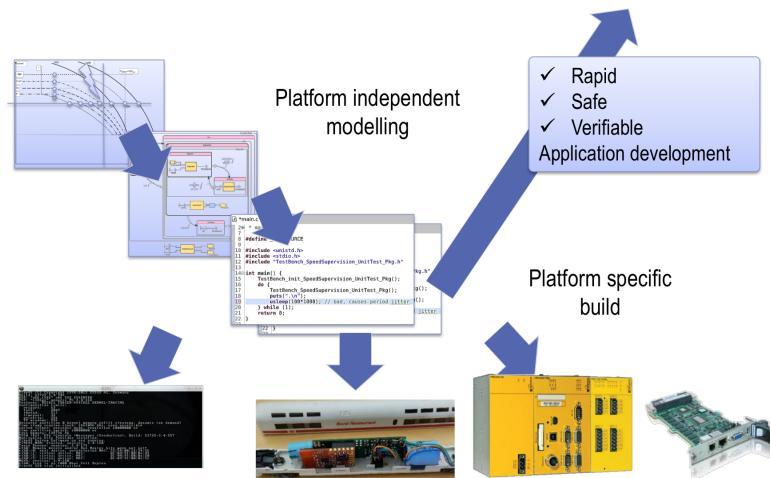


Abbildung 46. nanoETCS ist eine Proof-of-Concept-Implementierung eines Teils des openETCS-Modells.

Die Verwendung formaler Verfahren und der Einsatz von Modellierungen wird im Standard für SIL-4-Entwicklungen „dringend empfohlen“ (highly recommended, HR). Wesentliche Teile des Entwurfsprozesses wurden normkonform durchgeführt, bzw. es wurde demonstriert, dass die eingesetzten Methoden und Techniken für eine normkonforme Entwicklung geeignet wären. Allerdings wäre oft noch zusätzliche Arbeit erforderlich, um die jeweiligen Anforderungen des CENELEC-Standards zu 100% zu erfüllen.

6.6 Arbeitspaket 5 – openETCS Demonstrator

Wesentliches Ziel des Arbeitspakets war die Erstellung von mehreren Plattformen für die Demonstration der in Arbeitspaket 3 erstellten openETCS On-Board Unit. Insgesamt sind im Projektverlauf vier – und damit wesentlich mehr als ursprünglich geplant – solcher Demonstratoren entstanden. Dies sind im Einzelnen:

- Der so genannte nanoETCS Demonstrator, welcher das in Arbeitspaket 3 entwickelte Speed and Distance Monitoring Modell und Bremskurven Modelle mit Hilfe einer mikroprozessor gesteuerten Modelleisenbahn demonstriert.
- Ein maßgeblich vom französischen Projektpartner ERSA entwickelter Demonstrator, im Folgenden als openETCS ERSA Demonstrator bezeichnet, welcher das openETCS OBU Modell in einer industriellen Testumgebung ausführt.
- Ein Demonstrator, welcher den generierten openETCS OBU Code auf einer GE bzw. Alstom On-board Unit Hardware ausführt (im Folgenden als openETCS GE Demonstrator bezeichnet).
- Ein von den externen Partnern LEA Railergy und Bachleitner & Heugel entwickelter Demonstrator (fortan als LEA/B&H Demonstrator bezeichnet), der u. a. das openETCS OBU Modell in einem portablen Koffer integriert und zu einem industriellen Produkt zur Validierung von OBUs und ETCS Strecken weiterentwickelt werden soll.

Die entwickelten Demonstratoren werden in den folgenden Unterabschnitten näher beschrieben.

6.6.1 nanoETCS Demonstrator

Der nanoETCS Demonstrator (vgl. Abbildung 46) ist aus den folgenden zwei Anforderungen entstanden:

1. Frühzeitige Code-Generierung auf Basis des openETCS OBU Modells für reale Hardware, um mögliche Flaschenhälse früher erkennen zu können.
2. Erstellung einer einfachen, greifbaren Darstellung des Bremsverhaltens eines Zuges als Messe Demonstrator (vgl. Kapitel 6.7.1.3, ITEA & Artemis Co-Summit 2015).

6.6.1.1 Modellbasierte Entwicklung

Die modellbasierte Systementwicklung im V-Modell bringt entscheidende Vorteile mit sich. So ist besonders die frühzeitige Verifikation der oberen Entwicklungsschritte möglich: Anforderungsspezifikation, Formalisierung und Modellierung. Diese Schritte sind, zumindest anfangs, unabhängig von der Zielhardware. Die aus der Spezifikation heraus formalisierten Modelle können je nach Integrationstiefe der geplanten Tests bzw. Verifikation für mehr oder weniger spezifische Hardware Code generieren (siehe Abbildung 47).

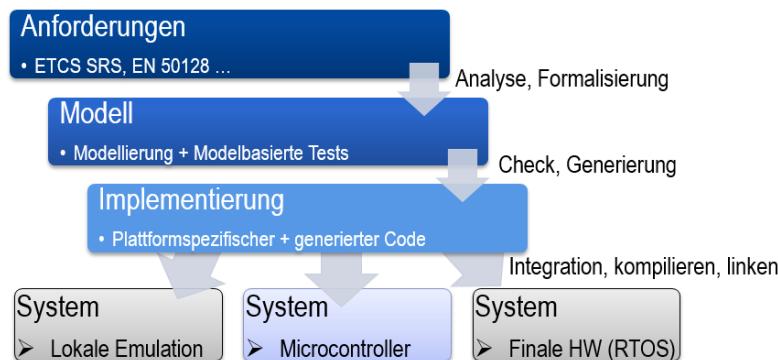


Abbildung 47. Stufen der modellbasierten Entwicklung bis zur Integration.

Die Modellierungsprache SCADE ist plattformunabhängig. Beim Generieren von C-Code ist der Overhead sehr gering und es fallen viel mehr die Restriktionen durch Plattformunabhängigkeit und Determinismus ins Gewicht. Hardwarezugriff und externe Datenkommunikation müssen deshalb in einer separaten Plattform-Abstraktionsschicht programmiert werden. Um den Test- und Verifikationsaufwand hier gering zu halten, empfiehlt es sich dedizierte Hardware und Pre-Zertifizierte Betriebssysteme mit abzuwagen. So kann die Abstraktionsschicht klein gehalten werden.

6.6.1.2 Die Hardware

Aus einem früheren Projekt heraus existiert am Institut für Angewandte Mikroelektronik und Datentechnik der Universität Rostock eine transportable Modelleisenbahn, ein Eisenbahnkoffer. Trotz des minimalistischen Modellmaßstabs 1:160 der Größe N (daher der Name *nanoETCS*) beträgt die maßstäbliche Länge einer Rundung nur etwa 0,5 km. Im Gegenzug sind die Modelle unproportional übermotorisiert, so dass Bremsen und Beschleunigen künstlicher Reduktion per Simulation bedarf. Des Weiteren werden Odometriedaten benötigt, die, mangels Messmöglichkeit, auch simuliert werden müssen.

Für diese Simulation wurde die Modell Lokomotive mit einer eigens entwickelten Mikrocontrollerschaltung mit Motoransteuerung erweitert, auf dem die Fahrsimulation ausgeführt wird (siehe Abb. 48). Für die Fahrsteuerung wurde entgegen klassischer digitaler Modellbahnsteuerungen, eine kleine Android-App entworfen, die mit der Modell Lokomotive per Bluetooth-Low-Energy kommuniziert. Die Android-App liefert ausschließlich den Fahrsollwert für die Beschleunigung und zeigt die empfangenen Ist-Werte der Simulation an: zurückgelegte Distanz und Geschwindigkeit.



Abbildung 48. Die Hardware des nanoETCS-Zuges im ICE-Speisewagen.

Auf Grund fehlender „Referenzbalisen“ oder präziser Odometrie und nicht-linearer Anomalien im Antrieb ist der Drift der Simulation erheblich und somit nur für One-Shot-Demonstrationen geeignet.

Der verwendete Mikrocontroller ist ein ATMEAL Xmega32e5, getaktet mit 32 MHz. Es kann maximal 36 kB Programmspeicher („flash“) genutzt werden, sowie bis zu 4 kB RAM für den Stack. Im Controller wird zusätzlich die komplexe PWM-Modulationseinheit WeX (Waveform Extension) genutzt, mit der alle Transistoren der Motor-H-Brücke direkt angesteuert werden können. Um Spannungsschwankungen zu kompensieren, misst der Controller kontinuierlich die Gleisspannung und skaliert entsprechend das Motorsteuersignal.

6.6.1.3 Die Software

Die Mikrocontroller Software besteht grundsätzlich aus der klassischen Initialisierung und dem interrupt-aktivierten Datentransfer mit dem Bluetooth-Modul. Der C-Code wurde mittels SCADE aus dem openETCS OBU Modell generiert. Dabei wird in der Regel jeder Operator des Modells zu einer C-Funktion. Der Einsprungpunkt von der Plattform-Abstraktionsschicht ist der Root-Operator. Dieser wird über einen Hardware-Timer getriggert, alle 100 ms aufgerufen, so dass der Modell-Code alle 100 ms ausgeführt wird. Zusätzlich müssen noch plattformspezifische Operatoren implementiert werden. Hier ist das die Wurzelfunktion `sqrt()` und die Array-Kopier-Funktion `memmove()`, die lediglich mit den Funktionen der AVR-libc verlinkt werden müssen.

Die erste Version des nanoETCS-Demonstrators lief im Frühjahr 2015 mit einem stark reduzierten Modell der ETCS Kernel Komponente F2.7 speedSupervision_Integration (vgl. Kapitel 6.4.5.7). So wurde eine einfache Bremskurve für das Ende der freien Strecke generiert, überwacht und falls nötig eine Zwangsbremsung eingeleitet. Abbildung 49 zeigt eine Momentaufnahme des zugehörigen Demonstrationsvideos.

6.6.1.4 Ergebnisse

Das Ausführen von SCADE generierten Modell-Code auch auf kleinen Mikrocontrollern ist ohne Weiteres möglich. Jedoch ist besonders das statische Anlegen großer Datenstrukturen ein Problem.

Der volle Code-Umfang der ETCS Kernel Komponente F2.7 speedSupervision_Integration (Stand Sept. 2015) würde einen Programmspeicher von knapp 100 kB benötigen. Generiert mit



Abbildung 49. Momentaufnahme aus dem nanoETCS-Demonstrationsvideo im Labor der Universität Rostock.

den Standard-Parametern für die Maximalgröße der Streckendaten werden statisch etwa 2 MB Arbeitsspeicher benötigt. Dies liegt Dimensionen über den Möglichkeiten kleiner Mikrocontroller, vor allem denen der nanoETCS-Hardware.

Bei der modellbasierten Entwicklung spielen Hardware Ressourcen zunächst keine Rolle. Auch wenn es einfach wäre, zur nächstgrößeren Hardware zu greifen, vervielfachen sich auch Laufzeit und Fehleranfälligkeit. Gerade Komponenten deren Dimensionierungsparameter mehr als linear in Laufzeit oder Speicherbedarf eingehen, sollten daher frühzeitig für eine potentielle Hardware analysiert werden.

Beispielhaft könnte ein trivialer Initialisierungsvektor einer großen Vektorstruktur statisch in vollem Umfang angelegt werden oder nur als einzelner Eintrag, der über eine einfache Iteration allen Vektorelementen zugewiesen wird. Im Trivialfall wäre die Laufzeit vergleichbar, der Initialisierungsvektor aber nur $1/n$ groß. Diese Optimierung könnte der Code-Generator oder der Compiler je nach zulässiger Optimierungsstufe sogar selbstständig vornehmen. Da das in diesem Demonstrator nicht der Fall war, wurde dies bei der Modellierung von der ETCS Kernel Komponente F2.7 speedSupervision_Integration (vgl. Kapitel 6.4.5.7) berücksichtigt.

6.6.2 openETCS ERSA Demonstrator

Der openETCS Projektpartner ERSA (vgl. Abschnitt 5.1.1) ist spezialisiert auf die Entwicklung von ERTMS/ETCS-Lösungen im Bereich von Simulations-Werkzeugen für Forschung, Präsentation, Test und Ausbildung, jedoch auch für sicherheitsrelevante Produkte für fahrzeug- und streckenseitige Anwendungen. In das openETCS Projekt eingeflossen ist ERSA's ERTMS/ETCS-Betriebssimulator, welcher eine Echtzeitsimulation darstellt, die veranschaulicht, wie Züge auf Strecken unter ERTMS/ETCS-Überwachung gefahren werden können. Auf Basis dieses Betriebssimulators ist der so genannte „openETCS ERSA Demonstrator“ entstanden. Der prinzipielle Aufbau dieses Demonstrators ist in Abbildung 50 dargestellt. Die einzelnen Komponenten werden im Folgenden erläutert:

OPEN EVC Diese Komponente stellt den funktionalen Kern der ETCS OBU dar. Der Demonstrator erlaubt hier den alternativen Einsatz zweier unterschiedlicher Softwarepakete:

1. Eine von ERSA entwickelte ETCS OBU Software, welche als Open Source verfügbar ist.
2. Den aus dem im Arbeitspaket 3 entwickelten formalen Modell generierten Code der openETCS OBU.

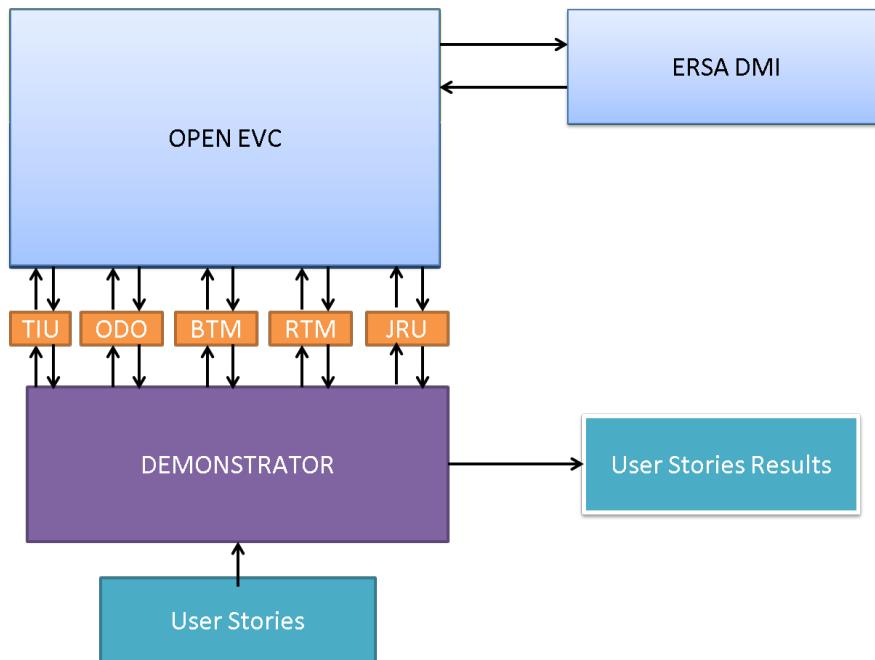


Abbildung 50. System Sicht des openETCS ERSA Demonstrators.

Demonstrator Die Demonstrator Komponente besteht aus einer Reihe von ERSA vertriebenen Produkten, um die Simulationsumgebung für die ETCS OBU zu schaffen:

- Dem Strecken-Editor und das damit erstellte Streckenmodell Amsterdam Utrecht.
- Die 3D Simulationsumgebung zum realitätsnahen Fahren eines Triebfahrzeugs.
- Dem Szenarien-Editor zum Erstellen von Fahrszenarien mit mehreren gleichzeitig verkehrenden Fahrzeugen auf der Strecke.

ETCS System Schnittstellen Entsprechend ERA Subset-026, Kapitel 2, werden im ETCS System die Schnittstellen des Systems spezifiziert. Die Schnittstellen wurden standardkonform zwischen Demonstrator und EVC Kern realisiert.

ERSA DMI Die Bedienoberfläche für den Triebfahrzeugführer, hier in einer ERSA eigenen Implementierung.

User Stories User Stories beschreiben aus Sicht der Kunden (i.d.R. Eisenbahnverkehrsunternehmen) das Verhalten des Systems. Die openETCS User Stories wurden hierbei an die nationalen Regeln für die Abnahme von ETCS Systemen angelehnt. Im hier betrachteten Simulationssystem werden User Stories durch Szenarien implementiert, über den Demonstrator an das System übergeben und in ihrer Reaktion aufgezeichnet und mit der erwarteten Reaktion verglichen. Daraus ergeben sich die User Story Results.

User Story Results Dies sind die gerade erwähnten Ergebnisse der simulierten Fahrt. Das sind neben der Beobachtung am DMI auch andere Parameter, wie zu jedem Zeitpunkt der Simulation aktuelle Bremskurven, ETCS Mode und Level, Bremszustände und vieles mehr.

Der openETCS ERSA Demonstrator bzw. das ERSA Simulationssystem bietet, wie in den folgenden Absätzen dargestellt, reichhaltige Anwendungsfälle im openETCS Projekt.

Agile Softwareentwicklung Der Simulator ist zentraler Bestandteil der agilen Softwareentwicklung im openETCS Projekt (vgl. Abbildung 51). Das klassische V-Modell wurde in kurzen Sprint Zyklen realisiert. Notwendige Voraussetzung dafür ist ein hoher Grad an Testautomatisierung und eine hohe Testeffizienz, welche durch Simulatoren unterstützt wird. Zunächst werden

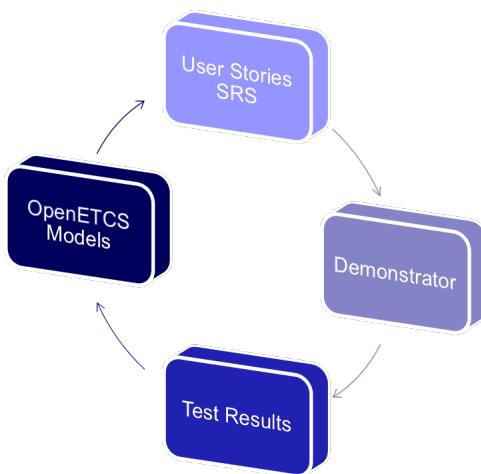


Abbildung 51. Simulatoren in der agilen Softwareentwicklung.

1. User Stories formal nach der ETCS Spezifikation und dem Betriebshandbuch als Demonstrator Szenario spezifiziert. Das geschieht in der Regel mit dem ERSA EVC.
2. Die Szenarien werden am Demonstrator mit openETCS OBU getestet,
3. Testergebnisse ausgewertet
4. und das Ergebnis fließt schließlich in das Modell zurück.

Rapid Prototyping Change Requests (Änderungsanträge) bzgl. der ETCS Spezifikation von Betreiberseite sind keine Ausnahme und stellen einen wesentlichen Baustein zur Erhöhung des Reifegrades der Spezifikation dar. Die vollständigen Auswirkungen von solchen Change Requests lassen sich aus dem geänderten Text- bzw. Spezifikationsvorschlag nicht immer direkt und vollumfänglich überblicken. Mithilfe des Simulationssystems können vorgeschlagene Änderungen einfach modelliert und über Szenarien-Editoren visualisiert werden. Dies hilft etwaige Nebeneffekte bereits vor der Änderung des Standards zu erkennen und die richtige Entscheidung zu treffen.

Schulung Simulatoren sind ein ideales Mittel zur Schulung von Mitarbeitern im Eisenbahnenwesen. Doch die Simulatoren werden in der Regel erst mit Auslieferung neuer Fahrzeuge an die neuen Standards angepasst, also vergleichsweise spät. Der openETCS Demonstrator steht frühzeitig und in Bezug auf kurzfristige Änderungen auch flexibel für Schulungen zur Verfügung. Aktualisierungen können parallel zur Entwicklung des Standards in die Schulung einfließen.

Einfache Portierung auf Zielplattformen Ein Beispiel für eine Portierung wird im nächsten Abschnitt dieses Berichtes beschrieben.

Konformitätstest Mittels ETCS Konformitätstests wird die Konformität von ETCS Ausrüstung zum entsprechenden funktionalen Teststandard überprüft. Solche Tests sind Voraussetzung für die Zulassung von ETCS Ausrüstung und werden unter anderem durch das DLR Labor durchgeführt. Die Simulationsumgebung bildet eine Basis für die Durchführung von Konformitätstests. Diese Art von Tests wurden im Projekt nicht mehr durchgeführt, sind aber zukünftig möglich.

6.6.3 openETCS GE Demonstrator

6.6.3.1 Die GE-Demonstrationsplattform

Zur Demonstration des openETCS European Vital Computer (EVC) wurde von General Electric Transportation (GE) die Hardware Plattform und die Entwicklungsumgebung für ein eigenentwi-

ckeltes ETCS On-board Gerät, siehe Abbildung 52, zur Verfügung gestellt. Die Schnittstellen zur Integration des Demonstrators, kurz „GE Demonstrator“ genannt, wurden zwischen GE und dem DLR Labor [36] abgestimmt und im DLR Labor auf der Hardware Plattform unter Verwendung der mit SCADE generierten openETCS EVC Quellcodes implementiert. Während der Projektlaufzeit ist die GE Transportation in Alstom aufgegangen, so dass die Plattform, die einen GE-Aufdruck vorweist, jetzt zum Konzern Alstom gehört.



Abbildung 52. GE-Demonstrator Hardware in einem 19-Zoll Einschub.

Das Zielsystem der Implementierung besteht aus der von Alstom gestellten Echtzeitplattform, auf dem das openETCS-Modell implementiert ist, und einer PC-Simulationsumgebung, auf der das DMI visualisiert wird und mit der die fahrzeugseitigen und streckenseitigen Komponenten simuliert werden.

Bei der Echtzeitplattform handelt es sich um ein redundantes, zweikanaliges System. Von GE wird ein proprietäres Betriebssystem mit dem Namen „OS4“ zur Verfügung gestellt. Dieses beinhaltet grundlegende Kommunikations- und Sicherheitsfunktionen. Kern ist ein Scheduler, der die Ausführungszeiten der einzelnen Threads auf dem System überwacht und, falls eine Obergrenze für die Ausführungszeit eines Threads überschritten wird, einen Fehler meldet und das System anhält.

In der verwendeten Konfiguration verfügt die Hardwareplattform über zwei redundante Central Processing Unit (CPU) Module. Auf beiden wird der integrierte openETCS EVC parallel ausgeführt. Die nachgeschalteten Vergleicher sichern durch Vergleich der Ausgaben beider Module das System gegen das Auftreten nicht funktionaler Fehler ab. Außerdem verfügt die Plattform über ein weiteres CPU Modul, das u. a. für die interne und externe Plattformkommunikation verantwortlich ist. Ein dediziertes Diagnosemodul erlaubt u. a. die Auswertung von Log-Nachrichten.

6.6.3.2 Demonstration der Software Integration

Für erste Funktionstests wurde das openETCS EVC auf einer leistungsfähigen PC-Plattform integriert. Grundlage bildete das vom Arbeitspaket 3 entwickelte EVC Modell in SCADE, aus dem unter Verwendung des SCADE Suite Code Generators (KCG) eine Implementierung abgeleitet wurde.

Eine korrekte Implementierung kann nur garantiert werden, wenn die Generierung des Quellcodes automatisch mit einem Code Generator erstellt wird, dessen Korrektheit ebenso nachgewiesen ist.

Esterel stellt mit dem KCG einen leistungsstarken und nach EN50128 T3 qualifizierten Quellcode Generator zur Verfügung, der für die Demonstration in der Version 6.4 genutzt wird.

6.6.3.3 Generierung aus Modell

KCG generiert die ANSI C Realisierung des hierarchischen SCADE Modells des EVC. Alle notwendigen Datenstrukturen, die durch generierte Funktionen beeinflusst werden, werden global allokiert. Eine Auswertung der Implementierung wird durch den Aufruf der einzelnen Funktion getriggert. Das führt zu einer kompletten Ausführung von jeder Komponente im hierarchischen, originalen EVC Modell. Aus diesem Grund ist das Zeitverhalten der Implementierung auf die Ausführung dieses Triggers berechenbar. Während die Relationen zu physikalischer Zeit durch die Einführung von Verzögerungen und Schleifen beeinflusst werden können, muss stets darauf geachtet werden, dass die kleinste zeitliche Auflösung der Worst-Case Execution Time (WCET) des gesamten Modells entspricht.

Der vom KCG generierte Quellcode ist komplett statisch und hardware- bzw. plattformunabhängig. Für die Implementierung auf einer spezifischen Plattform ist lediglich ein Wrapper zum Triggern des Modells notwendig. Folglich kann der vom Modell abgeleitete Quellcode von SCADE plattformunabhängig kompiliert werden. Selbst die Realisierung auf einem Echtzeit-Betriebssystem ist unter Berücksichtigung der konkreten Speicherverwaltung und der CPU-Scheduling-Strategie möglich.

Da das openETCS Modell neben SCADE Suite die Komponenten SCADE Display und SCADE Rapid Prototyper nutzt, ist die Implementierung des graphischen Subsystems auf einer geeigneten Zielplattform notwendig. Die Implementierung des Systeminputs, wie z. B. der Mauseingaben oder der grafischen Ausgaben sind plattformabhängig. SCADE Suite KCG unterstützt die plattformunabhängige Modellierung der Oberflächen in OpenGL [31], benötigt für die Nutzung jedoch entsprechende Systembibliotheken. Diese stehen derzeit für die Plattformen Windows/PC, Apple iOS und Android zur Verfügung. Demzufolge ist die Implementierung von openETCS für diesen Teil des Gesamtsystems plattformabhängig.

6.6.3.4 Systemarchitektur

Aus mehreren Gründen wurde das Gesamtsystem in zwei Komponenten geteilt: nämlich das EVC und das DMI. Einer der Gründe ist die logische Trennung, die bereits in der System Requirements Specification (SRS, [11]) eingeführt wurde und sich entsprechend auch im Modell wiederfindet. Einen zweiten Grund lieferte die geplante Umsetzung. Wegen der Plattformanforderungen des generierten DMI-Codes muss dieser zwangsläufig auf einer Konsumentenplattform – im Falle des Demonstrators auf einem handelsüblichen PC mit dem Betriebssystem Windows – ausgeführt werden. Das EVC hingegen ist plattformunabhängig und kann neben der Konsumentenplattform auch auf eine sicherheitskritische Plattform, bspw. die GE-Echtzeitplattform, portiert werden. Durch die logische Trennung von EVC und DMI wird die Umsetzung eines derart verteilten Systems vereinfacht.

Das SCADE EVC, beinhaltet alle modellierten Funktionalitäten der SRS innerhalb des openETCS-EVC Kernels, siehe Abbildung 53. Das EVC stellt neben der Synchronisierungsschnittstelle mit dem DMI zusätzliche Schnittstellen für die externe Zugkomponenten, wie die Train Interface Unit (TIU), den Wegstreckenzähler (Odometrie, ODO), das Balisen-Übertragungsmodul (BTM) und das Radio-Übertragungsmodul (RTM), zur Verfügung. Diese Komponenten können physisch, bspw. die entsprechenden Komponenten eines realen Zuges, oder logisch, bspw. durch eine Simulationsumgebung, realisiert werden.

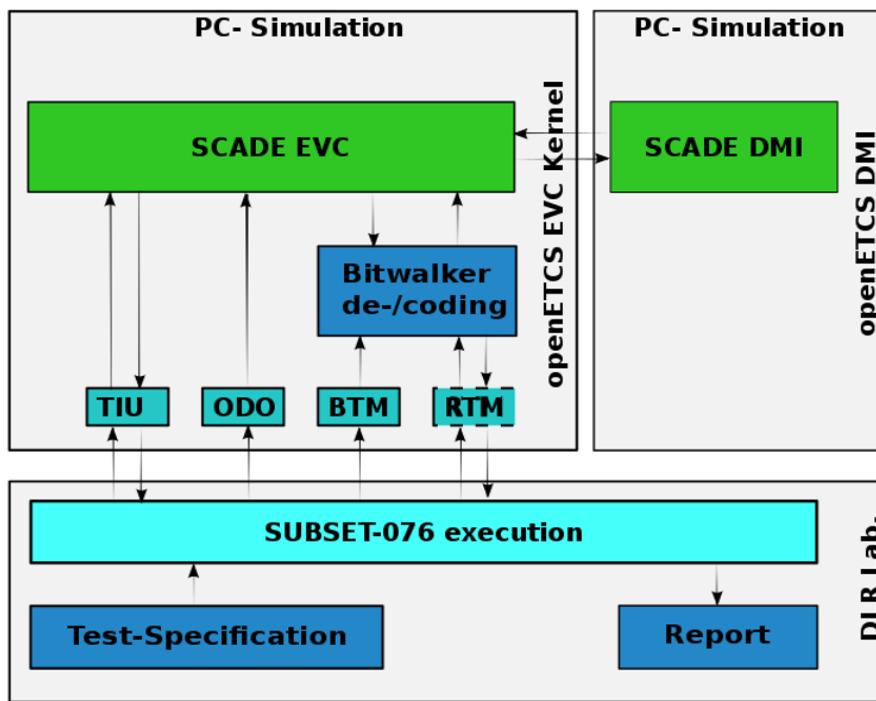


Abbildung 53. DLR Software Architektur der integrierten Gesamtsystems auf einem oder mehreren Standard-PCs.

Um die Kommunikation für die verschiedenen Schnittstellen möglich plattformübergreifend realisieren zu können, wurde TCP/IP als Kommunikationsprotokoll ausgewählt. Hierfür existieren Implementierungen für nahezu jede Plattform. Die physikalische Übertragung, bspw. über Ethernet, wird ebenso von den meisten Plattformen angeboten.

Als zentrale Kommunikationseinheit stellt das EVC Server-Ports für jede Schnittstelle zu den anderen Teilkomponenten zur Verfügung. Abbildung 53 illustriert die Software Systemarchitektur, wie sie im openETCS-Demonstrator realisiert wurde. Die externen Komponenten, wie TIU, ODO oder andere, wurden in dieser Konfiguration durch die Laborsoftware des RailSiTe® simuliert.

Da BTM- und RTM-Nachrichten im openETCS-Modell nur durch die externe Komponente Bitwalker kodiert und dekodiert werden können, musste die Implementierung dieser Komponente zwischen den Datenfluss vom RailSiTe und der EVC-Implementierung integriert werden.

Das zweite Teilsystem ist das SCADE DMI, siehe Abbildung 53, in welchem die grafische Oberfläche, die Interaktion mit dem Triebfahrzeugführer und einige grundlegende Funktionalitäten realisiert sind. Die komplette Implementierung – selbst für das native OpenGL-Teilsystem – wird, abgeleitet vom openETCS EVC Modell, ebenfalls durch KCG erstellt.

Dieses Teilsystem beinhaltet keine relevanten Sicherheitsmerkmale und ist für eine Win32-basierte Plattform generiert, für die OpenGL Unterstützung in Form von Bibliotheken durch SCADE bereitgestellt wird.

Das EVC und das DMI kommunizieren und synchronisieren über einen TCP Socket. Die entsprechenden Zugriffe auf die benötigten Datenstrukturen sind bereits im Modell vorgesehen.

6.6.3.5 Wrapper

Der Aufruf der EVC- sowie der DMI-Implementierung wird durch einen plattformabhängigen Wrapper (eine Hüllenfunktion) realisiert. In dieser Funktion sind die Zugriffe auf die einzelnen Kommunikationskanäle und sowie die Verwaltung sämtlicher Datenstrukturen und Ressourcen implementiert. Der Wrapper des EVC und des DMI hängen jeweils von der gewählten Zielplattform ab und wurden manuell implementiert. Jedoch ist die zugrunde liegende Struktur identisch, da in beiden Teilsystemen die gleichen Aufgaben zu handhaben sind.

Für jeden Kommunikationskanal wird ein einzelner Thread oder Prozess (in Abhängigkeit des zugrundeliegenden Betriebssystems) erstellt, um so den Informationsaustausch zwischen den Schnittstellen und dem EVC vom Zustand der einzelnen Sockets zu entkoppeln. Alle empfangenen Informationen eines Kanals werden in einer Hauptschleife des Wrappers behandelt. Die Informationen werden geprüft und gemäß der Datenstrukturen der API aktualisiert.

6.6.3.6 Portierung auf GE-Plattform

Nachdem die Funktionalität des integrierten Gesamtsystems (siehe Abbildung 53) geprüft wurde, galt es, die Implementierung des openETCS EVC auf die GE Plattform durchzuführen. Auch in diesem Fall wird das DMI auf einem Standard PC ausgeführt. Das resultierende System ist in Abbildung 54 dargestellt.

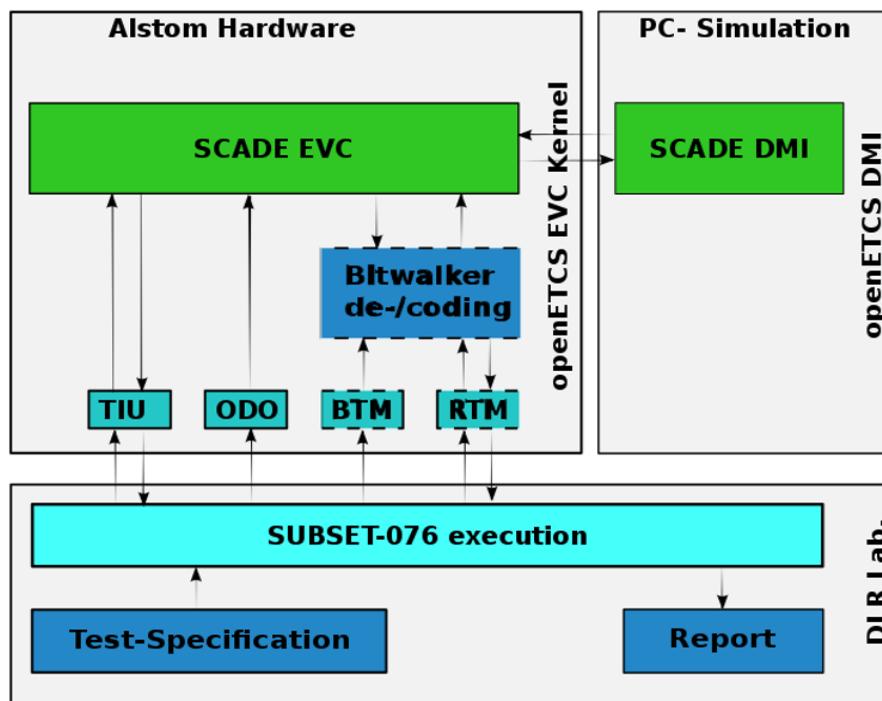


Abbildung 54. DLR Softwarearchitektur der Implementierung auf der GE-Plattform.

Während der generierte openETCS EVC Kern plattformunabhängig ist, musste der Wrapper auf die neue Plattform und vor allem auf das Echtzeitbetriebssystem angepasst werden, was insbesondere Änderungen bei der Nebenläufigkeit und dem Speichermanagement nach sich zog.

OS4 sieht vor, dass nebenläufige Threads als einzelne Prozesse – also komplett unabhängige, ausführbare Programme – realisiert werden sollten. Hierzu war es nötig, sämtliche Kommunikationsthreads aus dem Wrapper herauszulösen und in unabhängige Programme auszulagern. Die Kommunikation zwischen diesen und dem Wrapper erfolgt mittels Shared Memory.

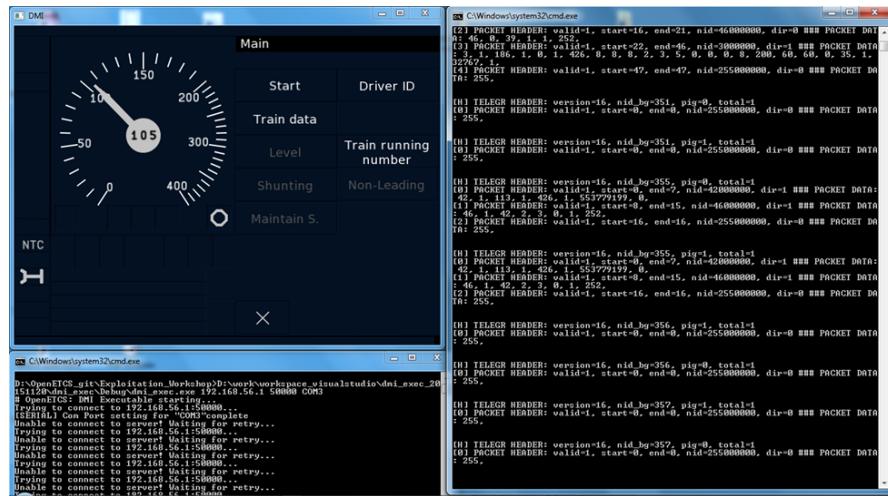


Abbildung 55. Software Implementierung des DMI im DLR Labor.

Auch in dieser Variante gelang eine vollständige Integration des Systems. Probleme bereitete jedoch die Synchronisation von EVC und DMI. Hierzu werden rund 5 kByte vom EVC zum DMI und ca. 1,5 kByte vom DMI zum EVC übertragen. In der zur Verfügung stehenden Konfiguration des Betriebssystems verfügt der TCP-Stack jedoch nur über einen Puffer von etwa 1 kByte Größe, so dass die Übertragung eines einzelnen Synchronisationsvorganges auf mehrere Durchläufe ausgeweitet werden muss. Durch das Zusammenspiel mit dem Scheduler benötigt ein kompletter Durchlauf des EVC – bestehend aus der Modellausführung und der Synchronisierung mit dem DMI – in etwa 750 ms. Davon werden über 700 ms für die Synchronisation benötigt.

Damit ist eine Realzeitausführung unmöglich. In Folge dessen, ist die Kopplung an das DLR-Labor (RailSiTe) auch nicht praktikabel, da dieses die Simulation in Realzeit durchführt.

Um dennoch eine erste Validierung auf dem per se lauffähig integrierten System vornehmen zu können, wurde die Streckensimulation, die bereits Teil des SCADE-Modells ist, mit im EVC – und damit auch auf der Echtzeitplattform – integriert. Prototypische Eingaben für den Simulationskern, bspw. die TIU-Eingaben, werden weiterhin über eine TCP-Verbindung von einem externen System, in diesem Falle einer kleinen GUI auf dem Standard-PC, empfangen.

Durch das monolithische System werden Systemzeiten der Simulation und des EVC implizit gekoppelt, so dass eine Validierung auf Basis der simulierten Strecke Amsterdam-Utrecht möglich ist, wenn auch deutlich langsamer. Ein Überblick über die verwendeten Teilprogramme ist in Abbildung 55 zu sehen.

6.6.3.7 Verifikation

Die grundlegende Annahme der modellgetriebenen Quellcodegenerierung ist die Korrektheit des Codegenerators. Diese Annahme ist für den qualifizierten KCG durch Esterel angegeben, und muss innerhalb der Anwendungsumgebung des Kunden durch die korrekte Integration und Absicherung der Werkzeugkette sichergestellt werden.

Für das sicherheitsrelevante System ist neben der Qualifizierung des Quellcode-Generators die Qualifizierung des plattformabhängigen Kompilierers ebenso wichtig. Die Qualifizierung des Kompilierers wurde im openETCS Projekt nicht weiter verfolgt. Es existieren hierzu kommerzielle Produkte.

Die Absicherung der Implementierung gliedert sich je nach Herkunft des Quellcodes auf:

- Für den mit dem qualifizierten Generator erstellten Quellcode gilt, dass die funktionale Korrektheit zu einem Teil auf Modellebene abgesichert werden kann (siehe Abschnitt 6.5.6). Zum anderen Teil werden aus dem generierten Quellcode Äquivalenzklassen abgeleitet und auf Quellcodeebene abgesichert (siehe Abschnitt 6.5.5).
- Für den außerhalb des qualifizierten Generator erstellten Quellcode, sei es händisch geschriebener Quellcode oder mit einem nicht nach CENELEC [26] T3 qualifizierten Generator erstellter Code, muss der komplette Quellcode nach der Implementierung auf Korrektheit geprüft werden.

Der manuell implementierte Wrapper und die socket-basierte Kommunikation müssen somit auf korrekte Funktionalität geprüft werden. Dies ist ebenfalls für das Gastbetriebssystem notwendig. Für den openETCS Demonstrator wurde diese Prüfung an dem EVC Teilsystem durchgeführt.

Da das DMI auf einem Commercial Off-The-Shelf (COTS) System ausgeführt wurde, wird dieser Teil als Konzeptnachweis gewertet.

Der EVC-Wrapper beeinflusst die generierte EVC-Implementierung nicht funktional. Sie stellt nur Eingangsparameter für die EVC-Implementierung zur Verfügung und ruft diese dementsprechend auf.

Zwischen dem EVC und dem DMI werden die kompletten Datenstrukturen zur Synchronisierung ausgetauscht. Die übertragenen Informationen werden vom Wrapper weder interpretiert noch manipuliert. Dementsprechend ist die korrekte Reihenfolge und Adressierung der übertragenen Funktion erfüllt - insofern beide Teilsysteme gleichen Paradigmen (Datentypen, Byte-Ordering, Serialisierung) folgen. Alle anderen Schnittstellen übertagen konkrete physikalische Werte, die auf Konsistenz geprüft werden müssen.

6.6.3.8 Validierung

Zur Validierung der Implementierung wurde die als Akzeptanzkriterium festgelegte ETCS Strecke Amsterdam-Utrecht in das SUBSET-076 Format übertragen, so dass diese im DLR Labor RailSiTe ausgeführt werden kann. Diese Testspezifikation wurde beim DLR erstellt und als Simulation gegen die erste Variante des integrierten Gesamtsystems (siehe Abbildung 53) getestet. Die Demonstration der Integration wurde auf der Test4Rail im Oktober 2015 in Braunschweig vorgestellt [39].

Auch mit dem GE-Demonstrator konnten erste Validierungen durchgeführt werden – hier auf der Basis der integrierten Streckensimulation. Die Hardwareintegration und -validierung wurde auf dem openETCS Abschlussreview im Dezember 2015 in München vorgestellt. Das DLR zeigte damit die erste funktionierende Implementierung des openETCS EVC auf einer eingebetteten Echtzeitplattform.

Ein Vergleich beider Validierungsergebnisse war bisher noch nicht möglich, da zwar beide Implementierungen auf Basis der gleichen Strecke validiert wurden, die Fahrt jedoch wegen der verschiedenen Bewegungssimulationen nicht identisch war.

6.6.3.9 Erkenntnisse der Hardware/Software Integration

Die Integration hat gezeigt, dass mit der gewählten Entwicklungsumgebung eine ETCS On-Board Einheit als PC-Simulation mit einer praxisüblichen Zykluszeit von unter 10ms realisiert werden kann. Weiter hat die Integration gezeigt, dass der generierte Quellcode selbst ohne Veränderung auf einem eingebetteten System lauffähig ist. Zykluszeiten des EVC von typischerweise unter 10ms konnten ebenfalls auf dem eingebetteten System erreicht werden. Es zeigte sich jedoch, dass die Kommunikation zwischen dem EVC und dem DMI problematisch ist. Wegen nicht abgestimmter Puffergrößen zwischen der Modellimplementierung und der TCP/IP-Bibliothek des Betriebssystems benötigt die Synchronisierung von EVC und DMI im Schnitt rund 700ms. Da diese Synchronisation nach jeder Modellauswertung durchgeführt werden muss, ist eine nahezu Realzeitausführung auf der Plattform derzeit nicht möglich.

Verbesserungen an der generischen openETCS-Schnittstellenmodellierung, vor allem an der Schnittstelle zwischen EVC und DMI, die zum Projektende noch den kompletten internen Zustandsraum mit jedem Zyklus übertragen hat und aufgrund der Abhängigkeiten zwischen den beiden Zustandssystemen ebenfalls komplett übertragen werden musste, sind bereits im openETCS-EVC Modell veranlagt und als nächster Implementierungsschritt zu empfehlen. Mit einer deutlich reduzierten Zykluszeit eines gesamten Wrapper-Durchlaufs (bestehend aus der Modellauswertung und der Kommunikation mit dem DMI bzw. den Fahrzeugschnittstellen) ist auch eine zielgerichtete Validierung des Gesamtsystems und ein Vergleich mit der PC-Integrationsversion möglich.

Prinzipiell gilt für das gesamte, integrierte Modell, dass es bisher ausschließlich mit den betrieblichen Tests auf der Strecke Amsterdam-Utrecht validiert wurde.

6.6.4 LEA/B&H Demonstrator

Durch eine Kooperation der Unternehmen LEA Railergy und Bachleitner & Heugel Elektronik OHG wurde ein gemeinsamer Demonstrator im Rahmen des Arbeitspakets 5 entwickelt, der in Abbildung 56 dargestellt ist.



Abbildung 56. LEA/B&H Demonstrator (vorgestellt im Rahmen des openETCS Abschluss-Reviews).

Der Demonstrator ist der Prototyp einer auf den Ergebnissen des openETCS Projekts basierenden kommerziellen Produktlinie. Die Produktlinie soll Lösungen für die folgenden Aufgabenstellungen bieten:

1. Validierung von ETCS Strecken gegen eine Referenz OBU
2. Validierung von OBUs auf realen Strecken
3. Validierung von OBUs auf simulierten Strecken
4. Datenerfassung und -aufbereitung für Zero-Onsite-Testing
5. Interaktive Simulation

Abbildung 57 stellt die Komponenten des Demonstrators dar. Die dargestellten Komponenten sind dynamisch koppelbar und können dadurch leicht auf verschiedene Hardwaresysteme verteilt werden. Eine von vielen möglichen Realisierungen dieser Koppelung stellt der im Rahmen des finalen openETCS Reviews vorgestellte portable Demonstrator dar.

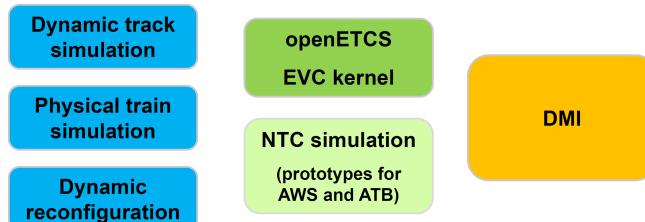


Abbildung 57. LEA/B&H Demonstrator: Systemkomponenten.

In der Größe eines Reisekoffers vereint der Prototyp alle in Abbildung 57 dargestellten Komponenten zu einem Gesamtsystem, welches es ermöglicht, ETCS Strecken zu simulieren. Die selbstentwickelte, aus einem massiven Aluminiumblock gefräste Kofferschale bietet den Komponenten Schutz auch in rauen Umgebungen. Als Bedien- und Anzeigeeinheit (DMI) kommt ein reales Führerstandgerät zum Einsatz, welches in der Lage ist, sowohl mit der openETCS DMI Software als auch mit kommerziellen Implementierungen und ihren Schnittstellen zu arbeiten. Durch einen Fahr-/Bremshebel und eine Sicherheitsfahrschaltung (SIFA) lässt sich der Zug realitätsnah beschleunigen und abbremsen.

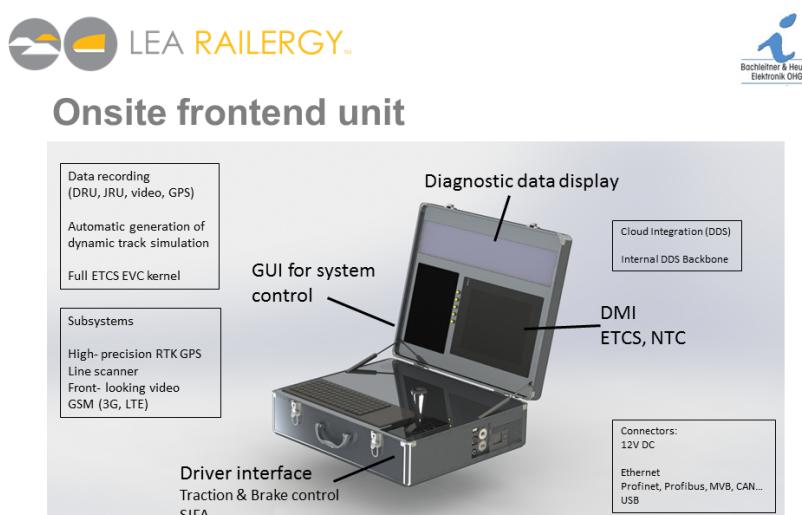


Abbildung 58. Funktionalitäten des LEA/B&H Demonstrators.

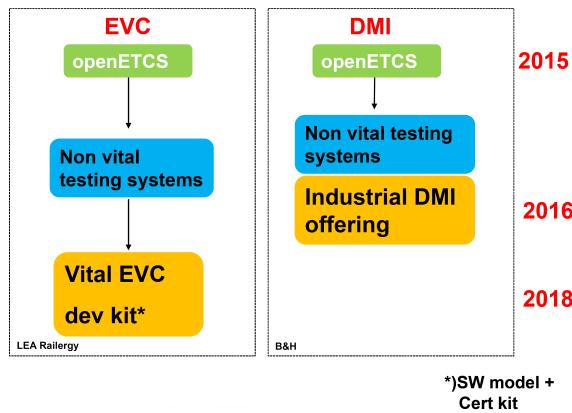


Abbildung 59. Entwicklungsplanung des LEA/B&H Demonstrators.

Abbildung 59 stellt den Zeitplan der Entwicklung vom Prototyp zum Produkt dar. Beginnend ab Ende des openETCS-Projekts werden die Funktionalitäten der im openETCS Projekt entstandenen Artefakte openETCS OBU und DMI erweitert und vervollständigt. Für die DMI-Software ist im Laufe des Jahres 2016 mit einer Veröffentlichung einer industriell verwendbaren Version zu rechnen. Der EVC wird ab 2018 als Softwaremodell mit zugehörigem Zertifizierungskit zur Verfügung stehen.

6.7 Arbeitspaket 6 – Veröffentlichung, Verwertung und Standardisierung

Ziele von Arbeitspaket 6 waren die Koordination der Öffentlichkeitsarbeit, der Verwertung der Projektergebnisse sowie der Transfer von Projektergebnissen in relevante technische Standards. Dabei war eine hohe Sichtbarkeit von openETCS in der fachlichen und allgemeinen Öffentlichkeit ein wichtiges Ergebnis, welches sich in den zahlreichen Veröffentlichungen und Präsentationen niederschlägt. Daneben war die gezielte Nutzung der Projektergebnisse in Form von Folgeaktivitäten (z. B. im Rahmen von Produktentwicklung oder Kundenprojekten) und gewonnener Expertise ein wichtiges Ergebnis dieses Arbeitspakets. Im Folgenden werden die Teilaufgaben im Einzelnen beschrieben.

6.7.1 Veröffentlichungen

Die Veröffentlichung der Projektergebnisse half dem Projekt die gewünschte Aufmerksamkeit seitens Industrie, Gesellschaft sowie der akademischen Welt zuteil werden zu lassen und geschah im Projekt über diverse Kanäle. Für eine wissenschaftlich-technische Verbreitung der Projektergebnisse wurden zahlreiche Vorträge gehalten und Artikel veröffentlicht (vgl. Tabelle 2). Eine detaillierte Liste der Projektpublikationen befindet sich in Kapitel 11. Die folgenden Abschnitte stellen die weiteren zur Verbreitung der Projektergebnisse genutzten Kanäle dar.

Tabelle 2. Anzahl von Veröffentlichungen und Vorträgen während die Projektlaufzeit.

Jahr	Veröffentlichungen	Vorträge
2012	3	3
2013	9	7
2014	20	25
2015	9	15

6.7.1.1 openETCS Webseite

Die öffentliche Webseite des Projekts ist unter <http://openetcs.org> verfügbar. Neben einer Kurzdarstellung des Projekts, der Ziele und der Projektpartner enthält die Webseite Links zur openETCS Arbeitsplattform auf GitHub¹⁵ und erlaubt somit einfachen Zugriff auf die im Projekt entstandenen Deliverables, Modelle, Software, sämtliche Protokolle der wöchentlichen Projektmeetings, etc. Weiterhin informiert ein Kalender über anstehende Veranstaltungen und Aktivitäten und erlaubte somit einen einfachen Projekteinstieg für Parteien, die bislang nicht im Projekt involviert waren.

6.7.1.2 Twitter und YouTube

Im Rahmen der Social-Media Aktivitäten wurde ein openETCS Twitter Kanal eingerichtet und ist unter <https://twitter.com/openetcs> erreichbar. Über den Kanal wurde regelmäßig über Projektneuigkeiten, Veranstaltungen, Fertigstellung von Deliverables und Meilensteinen, etc. berichtet.

Weiterhin wurde ein öffentlicher openETCS YouTube Kanal eingerichtet, welcher unter <https://www.youtube.com/user/openetcs> verfügbar ist. Der YouTube Kanal bietet insbesondere Zugriff auf Aufnahmen von online Trainings für Entwickler der openETCS Werkzeugkette. Die Trainings behandeln Themen wie das Arbeiten mit Git, SysML Model Checks und einen Leitfaden zum Einrichten von Eclipse und das Generieren/Übersetzen der openETCS Tool Chain für neue Entwickler.

6.7.1.3 Messen und Veranstaltungen

Messen und Veranstaltungen spielten eine große Rolle im Verlauf des Vorhabens. Das Projekt war, wie im Folgenden chronologisch dargestellt, auf zahlreichen Messen und Veranstaltungen vertreten und hat weiterhin eigene öffentliche Workshops organisiert.

ITEA & ARTEMIS Co-Summit 2012 „Sharing a Vision for ICT Innovation“

Auf diesem ITEA & ARTEMIS Co-Summit (30.-31. Oktober 2012 in Paris, Frankreich) wurde openETCS einem breiten Publikum aus Wissenschaft, Industrie, öffentlichen Institutionen und Presse vorgestellt.

ITEA & ARTEMIS Co-Summit 2013 „Software innovation: Boosting high-tech employment and industry“

Auch dieser ITEA & ARTEMIS Co-Summit (4.-5. Januar 2013 in Stockholm, Schweden) wurde genutzt um openETCS vor Experten aus Industrie, Wissenschaft und öffentlichen Institutionen zu präsentieren.

Workshop „Ensuring Safety of Industrial Critical Systems (ESICS)“ (IEEE INDIN 2013)

Auf der IEEE INDIN 2013 Konferenz (29.-31. Juli 2013 in Bochum, Deutschland) wurde ein öffentlicher Workshop mit dem Titel „Ensuring Safety of Industrial Critical Systems“ zusammen mit Partnern aus dem ARTEMIS Projekt VeTeSS organisiert. Neben Projektpartnern haben hier auch externe Teilnehmer (wie z. B. vom CERN) Beiträge präsentiert.

¹⁵Siehe <http://github.com/openETCS>.

InnoTrans 2014

Die InnoTrans 2014 (23.-26. September 2014 in Berlin, Deutschland), die weltweit größten Messe für Bahnverkehr, wurde vom Projekt als Plattform genutzt um die Projektergebnisse dem Fachpublikum zu präsentieren. Insbesondere folgende Aktivitäten wurden im Rahmen der InnoTrans durch das openETCS Konsortium organisiert:

- Eine „openETCS Session“ wurde in der „Speaker’s Corner“ der InnoTrans abgehalten, bei der drei Präsentationen von Projektpartnern (Deutsche Bahn, Nederlandse Spoorwegen und TWT) sowie ein eingeladener Beitrag von Ralph Müller (Eclipse Europe) gehalten wurden. Thematischer Fokus war die Nutzung des Open Source Konzepts in openETCS im speziellen sowie in der Entwicklung sicherheitskritischer Systeme im Allgemeinen.
- Der „openETCS Mid-Term Workshop“ wurde als öffentliche Veranstaltung im Rahmen der InnoTrans 2014 organisiert. Im Rahmen des Workshops haben Projektpartner Vorträge zu openETCS relevanten Themen von Standardisierungsaspekten bis zu Verifikationstechniken präsentiert und diese mit dem Publikum diskutiert.
- Weiterhin wurden openETCS Ergebnisse auf den Messeständen der Projektpartner präsentiert.

ITEA & ARTEMIS Co-Summit 2015 „Smart Industry: impact of software innovation“

Der Projektfortschritt und bisherige Ergebnisse wurden auf diesem Co-Summit (10.-11. März 2015 in Berlin, Deutschland) einem breiten Publikum präsentiert. Dies umfasste u.a. einen openETCS Demonstrator mit einer Mikrocontroller gesteuerten Modelleisenbahn. Die auf dem Mikrocontroller eingesetzte Software entstammt dabei aus dem in Arbeitspaket 3 entwickelten formalen Modell für Bremskurven. Eine detaillierte Beschreibung des Demonstrators findet sich in Kapitel 6.6.1.

Final openETCS Workshop

Am 17. Dezember 2015 wurde in München, Deutschland, ein finaler openETCS Workshop im Rahmen des Projektabschluss durchgeführt. Der Workshop wurde im Format einer „Unconference“ durchgeführt und das Programm bestand somit aus ad-hoc Vorträgen und Diskussionen der Teilnehmer, über die zu Beginn des Workshops abgestimmt wurde.¹⁶ An dem Workshop nahmen zahlreiche projekexterne Teilnehmer aus ganz Europa teil.

6.7.2 Markteinschätzung und Verwertung

Die Markteinschätzung und Verwertung wurde insbesondere im Rahmen einer „SWOT“ Analyse durchgeführt, in der je Partner jeweils die Stärken, Schwächen, Chancen und Gefahren des Projektes beleuchtet wurden. Diese Analysen wurden von den Projektpartnern auf einem zweitägigem Exploitation Workshop im November 2015 innerhalb des Projektes präsentiert. Dabei zeigt die Analyse deutlich, dass die Stärken und Chancen von openETCS gegenüber den Schwächen und Gefahren überwiegen. Eine quantitative Abschätzung der Auswirkungen von openETCS auf die Geschäftstätigkeit der Partner zeichnet ein positives Bild der Verwertungsmöglichkeiten. Ausführliche Beschreibungen des voraussichtlichen Nutzens und der Verwertbarkeit finden sich in Kapitel 9.

¹⁶Für weitere Informationen über das Format einer Unconference verweisen wir auf <https://en.wikipedia.org/wiki/Unconference>.

6.7.3 Standardisierung

Im Hinblick auf einen Transfer der Projektergebnisse ist es wichtig zu erwähnen, dass zahlreiche openETCS Projektpartner in einschlägigen Standardisierungsorganisationen vertreten sind und die Projektergebnisse dort einfließen lassen werden. Ein Überblick über die relevanten Standardisierungsorganisationen und die darin vertretenen Projektpartner ist in Tabelle 3 zu finden.

Tabelle 3. Liste relevanter Standardisierungsorganisationen sowie der darin vertretenen Projektpartner

Organisation	Kurzbeschreibung	vertretene Projektpartner
UIC	Internationaler Verband von Eisenbahnunternehmen	ATO, DB, NS, SNCF
CER	Vereinigung von europäischen Eisenbahnunternehmen und Infrastrukturbetrieben	ATO, DB, NS, SNCF
CENELEC CLC/SC 9XA	Zuständig für die EN 50128 und EN 50129	Siemens
DKE/AK 351.3	Deutsche Spiegelgruppe zur CLC/SC 9XA	TU-BS
UNISIG	Arbeitsgruppe der UNIFE zu ERTMS/ETCS Themen	Alstom, Siemens
EUG	ERTMS Anwendergruppe	ATO, DB, NS, SNCF
EuroSpec	Europäische Spezifikationen	ATO, DB, SNCF, NS
GMA techn. Kommittee 7.62	Automatisierung für Bahnverkehrssysteme	TU-BS
ProStep iViP	Vereinigung mit Fokus auf Lösungen zur virtuellen Produktentwicklung und deren Standardisierung, u. a. verantwortlich für den ReqIF Standard	Formal Mind, TWT
Subset-076 Arbeitsgruppe	Verantwortlich für die Pflege und Weiterentwicklung von Subset-076, d. h. der Testspezifikation für Subset-026	DLR

7 Wichtigste Positionen des zahlenmäßigen Nachweises

Die wesentlichen Kosten im openETCS Projekt waren die Personalkosten. Diese Aufwände summieren sich für die deutschen openETCS Partner auf rund 700 Personenmonate. Der zahlenmäßige Nachweis der entstandenen Kosten wird von den Projektpartnern einzeln im Rahmen der Erfolgskontrollberichte dargestellt.

8 Notwendigkeit und Angemessenheit der geleisteten Arbeit

Das openETCS Projekt adressiert viele wichtige Fragen, die den Lebenszyklus sicherheitsrelevanter eingebetteter Software von Beginn an bis zu ihrer langjährigen Pflege betreffen. Das grundlegende Konzept ist, Open Source Prinzipien auf den kompletten Lebenszyklus, beginnend mit der Spezifikation über Entwicklung, Verifikation und Validierung bis zum fertigen Software Produkt inklusive aller Werkzeuge und Sicherheitsnachweisdokumente anzuwenden. Dieses „Open Proofs“ Prinzip konnte im openETCS Projekt erstmalig in der Industrie im Allgemeinen und auf das europäische Zugsicherungssystem ETCS im Besonderen angesetzt werden, da die wichtigsten Voraussetzungen dafür bereits erfüllt waren: die Systemspezifikation SRS (ERA Subset-026) und die Testfallspezifikation (ERA Subset-076) waren bereits „Public Domain“. Überdies wurden anspruchsvolle Open Source Werkzeugketten wie TOPCASED bereits erfolgreich für sicherheitsrelevante Avionics-Funktionen eingesetzt und bilden ein unterstützendes Umfeld für Open Source Methoden.

Das openETCS Projekt und die erzielten Ergebnisse werden insbesondere:

- die starke Position der europäischen Bahnindustrie weiter verbessern,
- die Interoperabilität verschiedener ETCS Realisierungen erhöhen,
- helfen, die langfristigen Wartungskosten der ETCS Software zu verringern und
- die Qualität der ETCS Software mit Hilfe durchgängiger Verifikations- und Validierungsmethoden, die bereits in anderen Anwendungsbereichen erfolgreich waren, erhöhen.

Um diese Ziele gemeinsam zu erreichen hat sich ein Konsortium aus Großunternehmen mit umfassender Erfahrung im Bahnsektor (Alstom, Deutsche Bahn, Siemens), mittelständischen Innovationsträgern (AEbt, Innooprac / Eclipse Source, Formal Mind, TWT) sowie namhaften Forschungseinrichtungen (DLR, Fraunhofer-Gesellschaft) und Universitäten (Technisches Universität Braunschweig, Universität Bremen, Universität Rostock) zusammengefunden. Dieses Konsortium zeichnet sich durch eine Balance zwischen den verschiedenen Arten von Beteiligten aus dem Bereich der Bahntechnik aus und repräsentiert die gesamte Wertschöpfungskette mit Forschungsinstituten, Herstellern, Dienstleistern, Prüfinstituten und Endnutzern. Das im Projekt umgesetzte Open Proofs Konzept erlaubt allen anderen Marktteilnehmern Zugriff auf sämtliche Ergebnisse und stellt einen wesentlichen Multiplikator sowohl für die Verwertung als auch für den volkswirtschaftlichen Nutzen dar.

Die Notwendigkeit der Förderung dieses Vorhabens ergibt sich insbesondere aus den im Folgenden aufgeführten Gründen:

- Das Projekt weist einen hohen Grad an Komplexität und Innovation auf. Hieraus begründet sich ein Risiko, die Projektergebnisse auf dem Markt durchzusetzen, obgleich es von wichtigen Teilnehmern im Bahnsektor unterstützt wird.
- Das Projekt ermöglicht die Bildung einer neuartigen und vollständigen Wertschöpfungskette, für die verschiedene Arten von Kooperationspartnern erforderlich sind und von denen keiner für sich allein in der Lage ist, eine solche Innovation durchzusetzen.

- Es gibt ein hohes Standardisierungspotential unter europäischer Führung.
- Insbesondere Open Source Konzepte können nur dann erfolgreich sein, wenn zwei wesentliche Voraussetzungen erfüllt sind:
 1. Die Open Source Software (OSS) muss einen hinreichend großen Wert repräsentieren, so dass es für einen potentiellen Anwender ökonomischer ist, sich der OSS Initiative anzuschließen, als selbst eine vergleichbare Software zu entwickeln und diese eigenständig zu vermarkten.
 2. Das zugehörige Ökosystem muss hinreichend leistungsfähig sein, so dass eine gemeinsame Weiterentwicklung ökonomischer ist, als die Software selbst weiter zu entwickeln.

Um die Voraussetzungen für diese beiden Erfolgsfaktoren zu erfüllen, war die Förderung des Verbundvorhabens im ITEA Programm erforderlich und ein wesentlicher Schlüssel um die finanziellen und organisatorischen Rahmenbedingungen zu schaffen.

Alle im Projektantrag sowie dem Arbeitsplan formulierten Aufgaben und Ziele wurden durch das Konsortium erfolgreich bearbeitet. Wo möglich wurde auf existierenden Lösungen und Vorarbeiten aufgesetzt und etablierte Konzepte aufgegriffen und weiterentwickelt. Sämtliche Projektergebnisse stehen der Öffentlichkeit über die Projektwebsite und die GitHub Repositories des Projekts zur Verfügung.¹⁷

¹⁷Siehe <http://openETCS.org> und <https://github.com/openETCS>.

9 Voraussichtlicher Nutzen und Verwertbarkeit der Ergebnisse

Die Betrachtungen zur Verwertbarkeit der Ergebnisse sind im Folgenden dargestellt. Die Projektteilnehmer werden dabei gruppiert nach den jeweiligen Marktrollen (Fahrzeugherrsteller, Eisenbahnunternehmen, Dienstleister, akademische Forschung) bzw. der Position in der Wertschöpfungskette betrachtet (vgl. Abbildung 60).

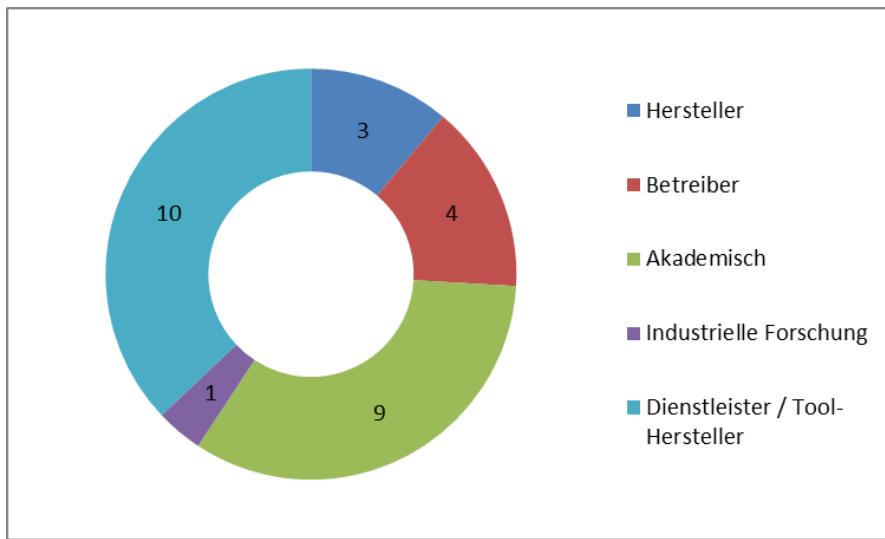


Abbildung 60. Überblick der Marktrollen der Projektpartner. Die Zahlen im Diagramm geben die Anzahl der Partner in der jeweiligen Marktrolle an.

9.1 Fahrzeugherrsteller

Im openETCS Konsortium sind die Fahrzeugherrsteller Alstom und Siemens vertreten. Nachdem der Geschäftsbereich Bahnverkehr und Signalechnik von General Electric (GE) durch Alstom übernommen wurde, sind die diesbezüglichen Betrachtungen mit denen von Alstom zusammengefasst. Beide Hersteller (Alstom und Siemens) planen die openETCS Tool Chain, das entstandene Referenzmodell sowie die Methodik, Werkzeuge und die im Projekt gesammelte Expertise zu verwerten. Darüber hinaus plant Alstom insbesondere die API und die Architektur, die im Rahmen von openETCS entwickelt wurde, zu verwerten. Die einzelnen verwertungsrelevanten Ergebnisse der Fahrzeugherrsteller sind in Tabelle 4 zusammengefasst.

Die Verwertung der gewonnenen Expertise und Ergebnisse, wie beispielsweise der Tool Chain, der openETCS API und der Architektur wird von den Fahrzeugherrstellern hauptsächlich im Rahmen der internen Systementwicklung, Beratung und Validierung der eigenen Produkte stattfinden. Darüber hinaus plant Alstom eine Verwertung durch eine verbesserte Wartung der ETCS On-Board Ausrüstung.

Tabelle 4. Verwertbare Projektergebnisse der Fahrzeughersteller.

Organisation	Verwertbare Ergebnisse
Alstom	openETCS API, Architektur, Tool Chain, Referenzmodell, ETCS Expertise, insbesondere in Bezug auf Formalisierung und Modellierung
Siemens	Tool Chain, Referenzmodell, Expertise insbesondere in Bezug auf Methodik und Werkzeuge

9.1.1 SWOT-Analyse

Von beiden Partnern wurde eine „SWOT“ Analyse¹⁸ durchgeführt. Die wichtigsten Ergebnisse dieser Analyse werden in Tabelle 5 aufgeführt.

Während das openETCS Projekt zahlreiche Stärken und Chancen bietet, wie beispielsweise reduzierte Entwicklungskosten, verbesserte Interoperabilität (was wiederum zu einem größeren Marktzugang und niedrigeren Kosten für Homologation führt) oder potentielle Kooperationen, so birgt das Projekt auch Risiken. Diese bestehen vorrangig in einem möglicherweise verschärften Wettbewerb zwischen den Fahrzeugherstellern (was andererseits für die Bahnbetreiber vorteilhaft ist), einem bislang nicht erprobtem Geschäftsmodell, das auf Open Source Software basiert sowie zusätzlichen Aufwänden die notwendig sind, um die Ergebnisse (vor allem die Softwarewerkzeuge und das Referenzmodell) zur Industriereife zu bringen. Die Nutzung des Closed Source Modellierungswerkzeugs SCADE wird von einem der Hersteller als Gefahr bewertet, da nicht sichergestellt ist, dass eine langfristige Unterstützung durch den Werkzeughersteller (Esterel Technologies / Ansys) gewährleistet ist.

9.1.2 Quantifizierte Auswirkungen auf das Kundengeschäft

Seitens Alstom wurden Zahlen bereitgestellt, die einen signifikanten Zuwachs der messbaren Ergebnisse vorhersagen. Diese Zahlen beinhalten die von GE übernommenen Geschäftsbereiche:

- Eine Senkung der Kosten für Einkauf, Tests und Inbetriebnahme von ETCS Fahrzeugen im Umfang von 10 - 20% wird ab 2017 für Software und ab 2020 für Software und Hardware von Alstom und GE erwartet.
- Ab dem Jahr 2020 können die Softwarewartungskosten für ETCS OBU Funktionen, die durch openETCS abgedeckt werden, um bis zu 50% gesenkt werden.
- Alstom plant Nachfolgeaktivitäten von 2016 bis 2022 im Rahmen von Shift²Rail. Durch diese Aktivitäten werden weitere Einsparungen von 10 - 20% erwartet.

¹⁸Aus dem englischen; Abkürzung für Strengths – Weaknesses – Opportunities – Threats (zu deutsch: Stärken – Schwächen – Chancen – Gefahren).

Tabelle 5. SWOT Analyse der Fahrzeughersteller.

Stärken	<ul style="list-style-type: none"> • Verringerte Kosten für Systementwicklung durch Open Source Komponenten • Verbesserte Interoperabilität von ETCS Fahrzeugen • Verbesserte Wartbarkeit der ETCS On-Board Ausrüstung • Reduzierte Produkteinführungszeit bei Hardware-Änderungen • Erhöhte Flexibilität bezüglich unterschiedlicher Hardware-Plattformen
Schwächen	<ul style="list-style-type: none"> • Open Source Geschäftsmodell muss sich noch als tragfähig herausstellen • Möglicherweise erhöhter Wettbewerb
Chancen	<ul style="list-style-type: none"> • Kooperationspartner für Systementwicklung konnten gefunden werden • Unabhängig zertifizierte API für die Hardware-Plattform ist möglich • Verbesserte Validierungsstrategie aufgrund der Ergebnisse aus Arbeitspaket 4
Gefahren	<ul style="list-style-type: none"> • Die Ergebnisse aus openETCS benötigen weitere Aufwände um Industriereife zu erlangen • Lebensdauer von Closed Source Werkzeugen (SCADE) abhängig von Dritten • Mögliche neue Wettbewerber • Das Referenzmodell deckt ERA Subset-026 noch nicht vollständig ab • Das openETCS Ökosystem muss weiter wachsen, um die volle Wirkung am Markt zu entfalten

9.2 Eisenbahnunternehmen

Das openETCS Konsortium beinhaltet vier Eisenbahnunternehmen: Deutsche Bahn (DB), Nederlandse Spoorwegen (NS), ATOC und SNCF. Die Informationen in diesem Abschnitt beziehen sich hauptsächlich auf die Verwertung durch DB, NS und teilweise ATOC. Die verwertbaren Ergebnisse sind in allen drei Fällen sehr ähnlich und im Folgenden dargestellt.

openETCS Prinzipien („Open Proofs“) Für Eisenbahnunternehmen ist dies einer der wichtigsten Punkte des Projektes. Die openETCS Prinzipien werden bereits von mehreren der Eisenbahnunternehmen im Zusammenhang mit Fahrzeugausschreibungen und langfristigen Wartungsverträgen angewandt, beispielsweise indem Hersteller vertraglich verpflichtet werden den zu den ETCS On-Board Units gehörigen Software-Quellcode offenzulegen. Dieser Ansatz wird bereits erfolgreich bei der Deutschen Bahn sowie bei Nederlandse Spoorwegen angewendet.

Der Ansatz ist natürlich nicht auf ETCS Systeme beschränkt, sondern kann auch auf andere Projekte angewandt werden. Konkret erwägt der Projektpartner Nederlandse Spoorwegen die Prinzipien auch in anderen Projekten wie gSSD¹⁹, ORBIT²⁰ oder STM²¹ einzusetzen.

Herstellerunabhängige Referenzeinheiten Langfristig ist es Ziel der Eisenbahnunternehmen, zeit- und kostenintensive Testfahrten (welche möglicherweise auf stark ausgelasteten Strecken stattfinden müssen) von der Schiene in das Labor zu verlegen. Weiterhin werden die Bahnbetreiber in die Lage versetzt, die Standardkonformität von ETCS Ausrüstungen zu validieren.

Methoden- und Werkzeugexpertise, openETCS Tool Chain Die erworbene Expertise in den Bereichen Formalisierung, Modellierung und ERTMS Systementwurf in Kombination mit der openETCS Werkzeugkette bringt den Eisenbahnunternehmen Vorteile bei der Beschaffung von Fahrzeugen und bei deren Um- bzw. Nachrüstung. Die Bahnbetreiber werden dadurch in die Lage versetzt, die Systementwicklung selbst oder in Zusammenarbeit mit Partnern durchzuführen. Sie erhalten dadurch mehr Kontrolle über den Prozess und vermeiden die Abhängigkeit von einzelnen Lieferanten. Darüber hinaus kann der Formalisierungsansatz ebenso für andere sicherheitsrelevante Systeme, wie z. B. im Stellwerksbereich, eingesetzt werden.

9.2.1 SWOT Analyse

Eine SWOT Analyse wurde von DB und NS durchgeführt, die Ergebnisse dieser Analyse sind in Tabelle 6 dargestellt. Aus Sicht der Bahnbetreiber bietet der Ansatz des Projektes lediglich wenige Schwächen aber zahlreiche Stärken und Chancen. Der größte Vorteil des openETCS Ansatzes ist sicherlich die verbesserte Interoperabilität von Fahrzeugen, die ETCS einsetzen – schließlich gibt es derzeitig immer noch kein mit ETCS ausgerüstetes Fahrzeug, welches sämtliche existierenden ETCS Strecken in Europa befahren kann, da die einzelnen Implementierungen nicht zu hundert Prozent kompatibel untereinander sind.

Daneben sind wichtige aus openETCS resultierende Vorteile für die Eisenbahnunternehmen die Möglichkeit für Eigenentwicklungen und langfristige herstellerunabhängige Wartung, wodurch auf längere Sicht niedrigere Kosten beim Einkauf und für die Wartung erwartet werden. Die Schwächen der Fahrzeugherrsteller sind somit Chancen für die Bahnbetreiber. Insbesondere in Hinblick auf Wartung wird ein verstärkter Wettbewerb erwartet. Zusätzliche Chancen ergeben

¹⁹gSSD beschäftigt sich mit der Stromüberwachung an Schnittstellen und der Energieverbrauchsmessung.

²⁰ORBIT ist ein Fahrerinformationssystem.

²¹„Specific Transmission Module“ (STM) sind erforderlich um ein ETCS Fahrzeuge auch auf Strecken mit nationalen Zugsicherungssystemen einsetzen zu können.

Tabelle 6. SWOT Analyse der Eisenbahnunternehmen.

Stärken	<ul style="list-style-type: none"> • Verbesserte Interoperabilität von ETCS Fahrzeugen • Potential für Eigenentwicklungen • Unabhängigkeit bei der langfristigen Wartung von ETCS OBU Systemen • Vorteile bei der Ausschreibungsspezifikation • Reduzierte Entwicklungskosten aufgrund von Open Source
Schwächen	<ul style="list-style-type: none"> • Notwendige Zeitspanne bis Ergebnisse erzielt werden
Chancen	<ul style="list-style-type: none"> • Ansatz ist erweiterbar und universell einsetzbar • Verstärkter Wettbewerb unter den ETCS Anbietern • Konzept der „Open Proofs“ führt zu erhöhter Sicherheit • Open Source Ansatz kann die Innovation im Sektor der Zugsiccherungstechnik EU-weit vorantreiben
Gefahren	<ul style="list-style-type: none"> • Politisch motivierte Entscheidungsfindung bei ERTMS Projekten • Keine vollständige Unterstützung des Ansatzes durch die Industrie • Das OBU Referenzmodell deckt noch nicht 100% der Spezifikation in ERA Subset-026 ab • Das openETCS Ökosystem muss weiter wachsen, um das volle Potential zu entfalten • Knappe Ressourcen

sich aus der Anwendbarkeit des „Open Proof“ Ansatzes auf andere Systeme, beispielsweise im Infrastrukturbereich.

Die höchsten Risiken werden auf Seiten der zum Teil politischen motivierten Entscheidungsfindung sowie in möglicherweise mangelnder Unterstützung durch die Industrie gesehen. So ist das openETCS Projekt immer wieder auf den Widerstand einzelner Unternehmen getroffen, die ihre bisherigen Geschäftsmodelle gefährdet sehen und daher die breite Anwendung des „Open Proof“ Ansatzes nicht unterstützen.

9.2.2 Quantifizierte Auswirkungen auf das Kundengeschäft

Die erwarteten Auswirkungen wurden seitens der Deutschen Bahn wie folgt quantifiziert:

- Die Deutsche Bahn erwartet verringerte Kosten für Einkauf und Installation von ETCS Fahrzeugen. Von aktuell 350 kEUR pro ETCS OBU wird ein Reduktion der Kosten auf etwa

100 kEUR bis zum Jahr 2020 erwartet. Ab 2017 wird zudem mit einer Kostenreduktion bei der Software gerechnet.

- Bzgl. der Softwarewartung wird ab dem Jahr 2020 mit einer Reduktion der Kosten von derzeit mehr als 4000 EUR je ETCS OBU auf weniger als 1000 EUR gerechnet.

9.3 Dienstleister, Tool-Hersteller und industrielle Forschung

Die Gruppe der Dienstleister und Tool-Hersteller – in der Mehrzahl KMU – nimmt im openETCS Konsortium eine wichtige Rolle ein. Der Einfachheit halber wird das Mitsubishi R&D Centre Europe (MERCE), ein industrieller Forschungspartner, ebenfalls zu dieser Gruppe gezählt, was zu insgesamt 12 Unternehmen aus dem europäischen Konsortium führt, die in diesem Abschnitt betrachtet werden. Die drei wichtigsten verwertbaren Projektergebnisse sind im Folgenden dargestellt.

openETCS Tool Chain 10 von 12 Unternehmen dieser Gruppe planen die Integration der openETCS Tool Chain in ihrer Geschäftsplanung, entweder als Basis für Dienstleistungen oder Support, um existierende Produkte zu verbessern oder neue zu entwickeln.

Expertise in Methodik und Tools Die Expertise bezieht sich hierbei auf die Entwicklung sowie die Verifikation und Validierung. 9 von 12 Unternehmen werden die gewonnene Expertise in ihrer Geschäftstätigkeit nutzen. Diese umfasst Dienstleistungen für die Bahnindustrie, Beratung, Produktentwicklung oder domänenübergreifende Anwendung des gewonnenen Wissens.

Domänenwissen 6 von 12 Unternehmen haben signifikante Expertise im Bereich des Bahnsektors im Allgemeinen und ETCS im Speziellen aufgebaut. Dies wird ihnen erlauben ihre jeweilige Marktposition zu stärken und eröffnet neue Märkte. Einige der Partner – wie beispielsweise TWT – waren bislang nicht im Bahnsektor aktiv.

Ein Überblick der verwertbaren Projektergebnisse der Dienstleister ist in Abbildung 61 dargestellt.

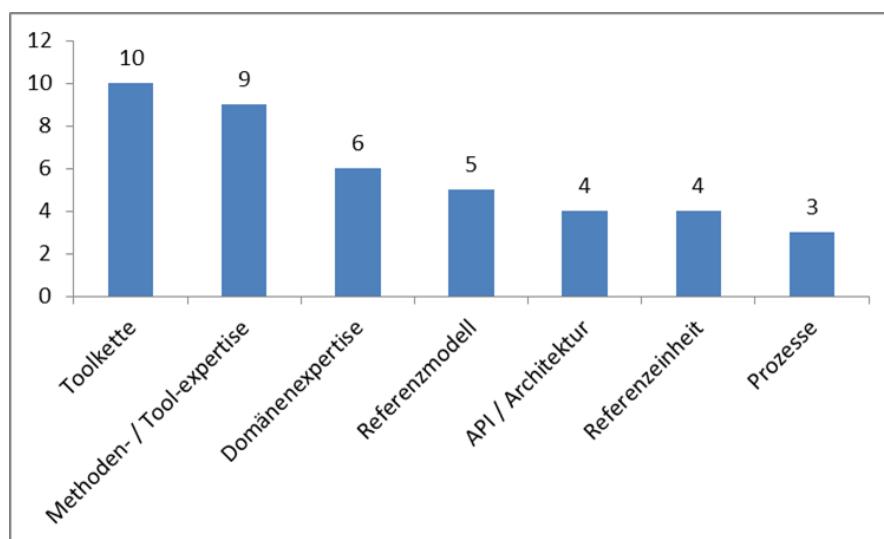


Abbildung 61. Ergebnisverwertung durch die Dienstleister, Tool-Hersteller und industrielle Forschungspartner. Ziffern im Diagramm repräsentieren die Anzahl der Partner, die das jeweilige Ergebnis verwerten werden.

Ergebnisverwertung jenseits des Bahnsektors

Ein weiterer wichtiger Aspekt ist die Verwertung der openETCS Methoden und Tools in anderen Domänen. 5 von 12 Unternehmen aus dem Konsortium haben derartige Pläne. Dabei sind insbesondere der Automobil- und der Luftfahrt-Sektor im Fokus, da diese vergleichbare Anforderungen und sicherheitsrelevante Anwendungen haben. Zum openETCS Konsortium zählen mehrere Unternehmen, die traditionell sehr aktiv in diesen anderen Branchen sind, wie beispielsweise Formal Mind und TWT. Daher wird von einer domänenübergreifenden Anwendung der Projektergebnisse bereits in den nächsten 1 bis 3 Jahren gerechnet. Dies betrifft neue sowie bestehende Kundenprojekte.

Arten der Ergebnisverwertung

Dienstleister, Tool-Hersteller und Forschungsunternehmen haben unterschiedliche Vorstellungen bzgl. der Verwertung der Projektergebnisse. Die häufigste Art ist die Produktentwicklung (8 von 12 Unternehmen), was sowohl die Verbesserung von bestehenden Produkten oder Tools als auch die Neuentwicklung von Produkten oder Dienstleistungen umfasst. Dies betrifft den Bahnsektor (7 von 12 Unternehmen) sowie andere Industriesektoren (5 von 12 Unternehmen).

Ein Gesamtüberblick der jeweiligen Verwertungsarten dieser Gruppe ist in Abbildung 62 dargestellt. Zu den genannten Verwertungspunkten zählen domänenübergreifende Dienstleistungen, Wartung, Beratung oder Dienstleistungen im Bereich Verifikation und Validierung.

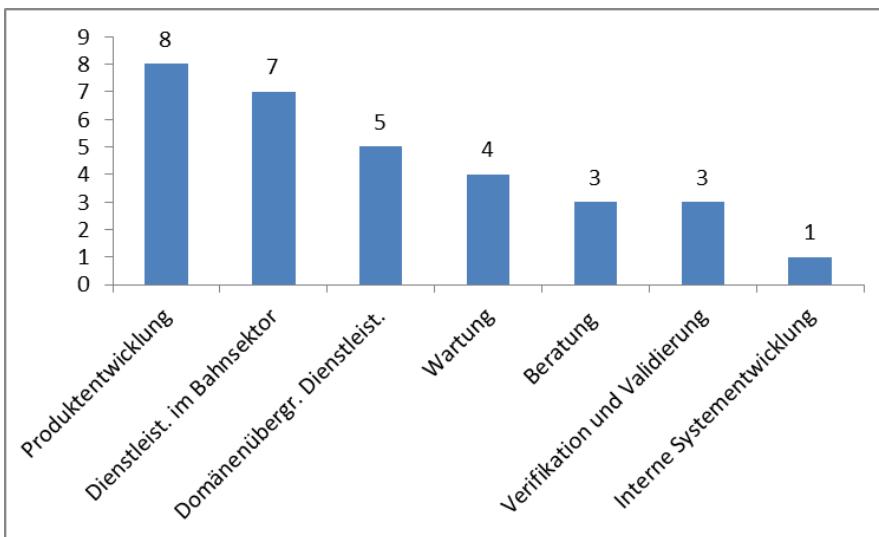


Abbildung 62. Arten der Ergebnisverwertung durch die Dienstleister, Tool-Hersteller und industrielle Forschungspartner. Ziffern im Diagramm repräsentieren die Anzahl der Partner, die das jeweilige Ergebnis verwertern werden.

Kooperationen zwischen mehreren KMU oder zwischen KMU und Großunternehmen wie Fahrzeugherstellern oder Bahnbetreibern bringen in der Regel erhebliche Vorteile für alle Beteiligten. Diesbezüglich stellt das openETCS Projekt einen hervorragenden Ausgangspunkt dar, da viele der Partner bereits gemeinsam Projektthemen bearbeitet haben. Zusätzlich zu den bereits beschriebenen Dienstleistungs- und Produktentwicklungen beinhalten solche Kooperationen auch weitere Forschungsprojekte, wie sie von mehreren Partnern geplant werden.

9.3.1 SWOT Analyse

Von den 12 Unternehmen in dieser Gruppe haben 9 eine SWOT Analyse durchgeführt. Die Ergebnisse sind in Tabelle 7 zusammengefasst.

Tabelle 7. SWOT Analyse der Dienstleister, Tool-Hersteller und industriellen Forschungspartner.

Stärken	<ul style="list-style-type: none"> • Verbesserte Dienstleistungen und erweitertes Dienstleistungsportfolio • Erworbenes Domänen- und Methodenwissen stellt einen Vorteil gegenüber Wettbewerbern dar • Open Source Ansatz ermöglicht breite Nutzung • Verbesserte Produkte • Senkung der Entwicklungskosten
Schwächen	<ul style="list-style-type: none"> • Open Source Geschäftsmodell muss sich als tragfähig herausstellen • Möglicherweise verschärfter Wettbewerb
Chancen	<ul style="list-style-type: none"> • Erschließung neuer Märkte • Schaffung neuer Dienstleistungen und Produkte • Verbesserte Sichtbarkeit auf dem Markt durch Open Source • Domänenübergreifender Einsatz der Ergebnisse
Gefahren	<ul style="list-style-type: none"> • Aufwand und Kosten zur Entwicklung von industriereifen Lösungen • Schwierigkeiten bei der Durchdringung des fragmentierten Marktes des Eisenbahnsektors • Proprietäre Werkzeuge (SCADE) mit hohen Lizenzkosten in der openETCS Tool Chain • Potentielle Risiken bei der Sicherstellung der geistigen Eigentumsrechte (IP) • Das openETCS Ökosystem muss weiter wachsen, um den vollen Nutzen am Markt zu entfalten • Die Kundenbasis ist zur Zeit noch klein

Es wird erwartet, dass der openETCS Ansatz zur Verbesserung existierender Produkte und Dienstleistungen sowie zur Schaffung neuer Produkte und Dienstleistungen führen wird. Der Open Source Ansatz wird von den Projektpartnern in dieser Gruppe als essentielle Stärke angesehen. Die Integration von Closed Source Tools, d. h. SCADE, in den Kern des openETCS Entwicklungsprozesses wird als Gefahr eingeordnet. Einige Partner erwarten eine mögliche Verschärfung des Wettbewerbes aufgrund des Open Source Ansatzes, da Wettbewerber die frei verfügbaren Projektergebnisse nutzen und in ihre kommerziellen Produkte integrieren können.

Zusätzliche Gefahren bergen die Aufwände und Kosten, die für eine Weiterentwicklung bis zu industriereifen Lösungen erforderlich sind, sowie möglichen Schwierigkeiten im Zugang zum Markt des Eisenbahnsektors. Der letzte Punkt trifft insbesondere auf Unternehmen zu, die im Bahnsektor bislang nicht aktiv sind. Gleichzeitig stellt dies natürlich auch eine Möglichkeit zur Expansion in eben diesen Sektor dar.

9.3.2 Quantifizierte Auswirkungen auf das Kundengeschäft

Daten zu den erwarteten Auswirkungen auf das Kundengeschäft wurden von 7 der Unternehmen in dieser Gruppe bereitgestellt. Im Folgenden sind die wesentlichen Ergebnisse aufgeführt.

- Die meisten Unternehmen erwarten bis zum Jahr 2020 mindestens 10% ihres Geschäftsvolumens im Bahnverkehrsmarkt aufgrund der in openETCS erzielten Ergebnisse. Zwei Unternehmen erwarten sogar mehr als 20%.
- Folgeprojekte im Forschungsbereich werden von mindestens drei Unternehmen geplant.
- Formal Mind und TWT, die zur Zeit beide nicht im Bahnverkehrssektor aktiv sind, erwarten einen Zuwachs des Auftragsvolumens von jeweils 10% und 25%, ausschließlich auf Basis der openETCS Ergebnisse.
- ALL4TEC erwartet ein Wachstum von 40% in den Projekten zur modellbasierten Sicherheitsanalyse (über mehrere Domänen hinweg) zwischen 2016 und 2020. Dies liegt auch an der verbesserten Marktsichtbarkeit aufgrund der Open Source Lizenz der Tools.
- Aufgrund der reduzierten Kosten wird von ERSA bis 2018 eine Erhöhung der Margen von 5% und eine Steigerung des Umsatzes von 15% erwartet. Ab 2018 wird durch openETCS basierte Produkte und Dienstleistungen eine weitere Steigerung des Umsatzes um 5% erwartet.
- Auf Basis der openETCS Ergebnisse planen die Unternehmen ALL4TEC, Formal Mind und TWT vor dem Jahr 2020 je ein neues Produkt entwickeln.
- MERCE hat zwei neue Prototypen entwickelt, die in internen, domänenübergreifenden Projekten eingesetzt werden.
- TWT erwartet zwischen 2016 und 2020 ein Volumen von domänenübergreifenden Projekten im Umfang von 230 - 420 kEUR.
- LEA Railergy zielt bis zum Jahr 2020 auf einen Marktanteil von 60% bei ETCS On-Board Einheiten zur Streckenvalidierung und -prüfung sowie Cloud-basierter Streckensimulation. Weiterhin wird seitens LEA Railergy erwartet, dass das openETCS relevante geistige Eigentum in den kommenden 10 Jahren einen Anteil von 25% im Gesamtwert der ETCS Systeme einnehmen wird.
- AEbt erwartet bis 2020 einen Anteil der openETCS basierten Projekte von 5 bis 10% des Gesamtvolumens und hat im Zusammenhang mit openETCS basierten Produkten bereits erste Anfragen zu Beratungsprojekten erhalten.

9.4 Akademische Forschung

Ein Überblick über die durch akademische Partner verwerteten Projektergebnisse wird in Abbildung 63 gegeben. Der Schwerpunkt liegt stark auf der erworbenen Tool- und Methodenkompetenz. Sieben von neun akademischen Partnern haben insbesondere die Verwertung der Methoden zur Validierung und Verifikation geplant.

Die Schwerpunkte für die Verwertung durch die akademischen Partner liegen in der Lehre, Veröffentlichungen, der Weiterentwicklung von Technologien und Sicherheit sowie weitergehender Forschung. Einige der akademischen Partner und Forschungseinrichtungen entwickeln Produkte

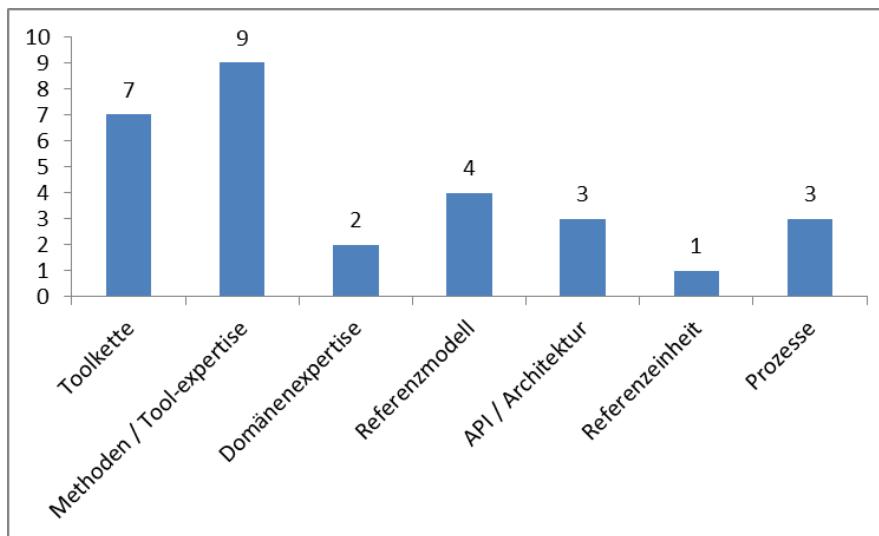


Abbildung 63. Ergebnisverwertung durch die akademischen Partner. Ziffern im Diagramm repräsentieren die Anzahl der Partner, die das jeweilige Ergebnis verwerten werden.

(z. B. Fraunhofer oder die Universität Bremen) in Form von Software Werkzeugen die entweder als Open Source oder als kommerzielles Closed Source Produkt (RT Tester) vertrieben werden. Diese Tools sind in die openETCS Tool Chain integriert und sind nicht auf die Domäne des Bahnverkehrs beschränkt. Die beteiligten Institute der Fraunhofer-Gesellschaft bieten industrielle Dienstleistungen an, die auf den Bahnsektor bzw. die ETCS Systementwicklung und die Modellpflege erweitert werden sollen. Die Universität Rostock zieht derartige Dienstleistungen ebenfalls in Betracht.

9.4.1 SWOT Analyse

Die Fraunhofer-Gesellschaft, die TU Braunschweig sowie die Universität Rostock haben eine SWOT Analyse durchgeführt. Die Ergebnisse dieser Analyse werden in Tabelle 8 dargestellt.

Während innerhalb dieser Gruppe keine allgemeinen Schwächen ausgemacht wurden existieren diverse Gefahren für die Verwertung durch die akademischen Partner. Ein Aspekt dabei ist die Mitarbeiterfluktuation aufgrund der projektbezogenen Stellen und den damit verbundenen Herausforderungen beim Wissenstransfer, sowie der generelle Mangel an Ressourcen. Weiterhin muss zusätzliche Expertise aufgebaut werden, um auf Basis der openETCS Ergebnisse neue Forschungsprojekte oder Dienstleistungen initiieren zu können. Sowohl die akademischen als auch die industriellen Partner betrachten die Einbindung kommerzieller Software in die openETCS Tool Chain als eine Gefahr.

Allerdings überwiegen die positiven Auswirkungen die Risiken deutlich, da das Projekt Möglichkeiten für domänenübergreifenden Wissenstransfer und neue Kooperationsmöglichkeiten im Rahmen von Forschungsprojekten oder Industrieaufträgen bietet.

9.4.2 Quantifizierte Auswirkungen

Die Fraunhofer-Gesellschaft, die TU Braunschweig sowie die Universitäten Bremen und Rostock haben die folgenden quantifizierten Perspektiven bereitgestellt:

- Die drei akademischen Partner TU Braunschweig, Universität Bremen und Universität Rostock planen Nachfolgeaktivitäten im Eisenbahnsektor in einem Umfang von 300 bis 400 kEUR.

Tabelle 8. SWOT Analyse der akademischen Partner.

Stärken	<ul style="list-style-type: none"> • Veröffentlichbare Resultate • Verbesserte Dienstleistungen und erweitertes Dienstleistungsportfolio • Breite Nutzbarkeit der Ergebnisse durch Open Source • Signifikante domänenspezifische Expertise erarbeitet
Schwächen	<ul style="list-style-type: none"> • Keine allgemeinen Schwächen identifiziert
Chancen	<ul style="list-style-type: none"> • Domänenübergreifender Wissenstransfer • Nachfolgende Industriekooperationen • Neue Forschungsaktivitäten • Zugang zum Bahnverkehrsmarkt
Gefahren	<ul style="list-style-type: none"> • Fluktuation der Mitarbeiter und mangelnde Ressourcen • Notwendigkeit neue Expertise aufzubauen • Schwierigkeiten im Zugang zum fragmentierten Bahnverkehrsmarkt • Proprietäre und teure Software (SCADE) in der Tool Chain • Industrielle Partner für Weiterentwicklung notwendig
<ul style="list-style-type: none"> • Die Universität Rostock erwartet im Jahr 2016 den Beginn einer Kollaboration mit einem Industriepartner zum Thema sicherer ETCS Plattformen. Drei Veröffentlichungen und eine Messeteilnahme auf Basis der openETCS Ergebnisse werden 2016 erwartet. Der Abschluss zweier Doktorarbeiten, die auf dem openETCS Projekt basieren, wird bis 2019 erwartet. • Die Universität Bremen kollaboriert mit einem Spin-Off Unternehmen um Teile der openETCS Tool Chain zu verwerten und das kommerzielle Produkt RT-Tester weiter auszubauen. 	

10 Fortschritte auf dem Gebiet des Vorhabens bei anderen Stellen

Wie in Kapitel 5 erläutert fand während der Laufzeit des Vorhabens ein Austausch mit thematisch verwandten Projekten statt. Zudem war das openETCS Projekt während der Projektlaufzeit auf den einschlägigen Messen und Kongressen vertreten, so dass Fortschritte bei anderen Stellen jederzeit beobachtet wurden.

Während der Projektlaufzeit wurde insbesondere mit Shift²Rail eine neue institutionelle „Public Private Partnership“ ins Leben gerufen.²² Shift²Rail soll durch Forschungs- und Innovationsaktivitäten die Wettbewerbsfähigkeit der europäischen Bahnindustrie fördern sowie einen Beitrag zur Schaffung eines einheitlichen europäischen Raums des Schienenverkehrs leisten. Somit werden hier prinzipiell gleiche Ziele wie im openETCS Projekt verfolgt. Jedoch wurden in Shift²Rail bislang keine Teilvergaben initiiert, welche sich mit der ETCS On-Board Unit befassen und sich somit mit dem openETCS Kernthema überschneiden würden.

²²Siehe <http://www.shift2rail.org/>.

11 Erfolgte Veröffentlichungen der Ergebnisse

11.1 Dissertationen

Tabelle 9. Dissertationen.

Autor	Titel	Hochschule, Datum
Feuser, Johannes	Open Source Software for Train Control Applications and its Architectural Implications	Universität Bremen, Jun 13
Helge Löding	Model-based Scenario Testing and Model Checking With Applications in the Railway Domain	Universität Bremen, Jul 14

11.2 Diplom- und Masterarbeiten

Tabelle 10. Diplom- und Masterarbeiten.

Autor	Titel	Hochschule, Datum
Benjamin Beichler	Ein simulationsbasierter Ansatz zur frühzeitigen Bewertung von Entwurfsalternativen	Universität Rostock, 2013
Gadireddi Venugopal	Modeling Safety-Critical Systems in the Railway Sector with SCADE	Universität Rostock, Feb 14
Jafar Nassar	Modellierung sicherheitskritischer Systeme im Eisenbahnsektor mit SysML	Universität Rostock, Jun 14
Giovanni Trotta	Model Driven Engineering of railway control system with the openETCS process	University of Napoli, 2014
Moritz Dorka	On the domain-specific formalization of requirement specifications – a case study of ETCS	TU Dresden, Jun 15
Alexander Probst	Modeling a safety-critical system for ensuring the interoperability in the european railway traffic	Universität Augsburg, 2015
Stefan Karg	Evaluation of the model-driven development approach in the openETCS project	Universität Ulm, Nov 15

11.3 Schriftliche Veröffentlichungen

Tabelle 11. Schriftliche Veröffentlichungen.

Autoren	Titel	Rahmen	Ort, Datum
Johannes Feuser, Jan Peleska	Model Based Development and Tests for openETCS Applications - A Comprehensive Tool Chain	FORMS/FORMAT 2012 (conference)	Braunschweig Dez 12

Fortsetzung von Tabelle 11

Autoren	Titel	Rahmen	Ort, Datum
Johannes Feuser, Jan Peleska	Dependability in Open Proof Software with Hardware Virtualization	Science of Computer Programming	Dez 12
Hardi Hungar, Marc Behrens	Opening up the Verification and Validation of Safety-Critical Software	ZeMoSS – Zertifizierung und modellgetriebene Entwicklung sicherer Software	Aachen Feb 13
Klaus-Rüdiger Hase	openETCS: Ein internationales ITEA2-Projekt begleitet den Wandel zu mehr Transparenz	41th Symposium „Moderne Schienenfahrzeuge“	Graz Apr 13
Stefan Gulan, Sven Johr, Roberto Kretschmer, Stefan Rieger, Michael Ditze	Graphical Modelling meets Formal Methods	INDIN 2013	Bochum Jul 13
Jens Gerlach, Virgile Prevosto, Jochen Burghardt, Kerstin Hartig, Kim Völlinger, Hans Pohl	Formal Specification and Automated Verification of Railway Software with Frama-C	INDIN 2013	Bochum Jul 13
Dirk Spiegel, René Sebastian Hosse, Jan Welte, Eckehard Schnieder	Integration of Petri Nets into STAMP-/CAST on the example of Wenzhou 7.23 accident	IFAC Workshop on Advances in Control and Automation Theory for Transportation Applications	Istanbul Sep 13
Jan Welte and Hansjörg Manz, Eckehard Schnieder	Survey of formal model-based development of safety-critical software for railway applications	World Congress on Railway Research (WCRR 2013)	Sydney Nov 13
Benjamin Beichler, Alexander Nitsch, Frank Golatowski, Christian Haubelt	Ein abstraktes SystemC-Modell zur Analyse und Leistungsabschätzung des europäischen Zugsicherungssystems ETCS	cpsDATA 2014	Stuttgart Sep 14
Klaus-Rüdiger Hase	An openETCS@ITEA2 project update	European Railway Review	Apr 14
Jan Peleska, Anne E. Haxthausen, and Ralf Pinger	Applied Bounded Model Checking for Interlocking System Designs	SEFEM Workshops 2013	Madrid Sep 13
Linh Vu Hong, Anne E Haxthausen and Jan Peleska	A Domain-Specific Language for Railway Interlocking Systems. (Best paper award)	FORMS/FORMAT 2014	Braunschweig Sep 14
Jan Peleska, Cécile Braunstein, Anne E. Haxthausen, Wen-ling Huang, Linh Vu Hong, and Uwe Schulze	Complete Model-Based Equivalence Class Testing for the ETCS Ceiling Speed Monitor	ICFEM 2014	Luxemburg Nov 14
Benjamin Beichler, Thorsten Schulz, Frank Golatowski, Christian Haubelt	A Parametric Dataflow Model for the Speed and Distance Monitoring in Novel Train Control Systems	CyPhy 2015	Amsterdam Oct 15
Anne E. Haxthausen and Jan Peleska	Model Checking and Model-Based Testing in the Railway Domain	Workshop	Bremen Jun 15

Fortsetzung von Tabelle 11

Autoren	Titel	Rahmen	Ort, Datum
Felix Hübner, Wen-ling Huang, and Jan Peleska	Experimental Evaluation of a Novel Equivalence Class Partition Testing Strategy	TAP 2015	Aquila Jul 15
Ayşe Yurdakul, Jan Welte, and Eckehard Schnieder	Construction of a Consistent openETCS Tool Development Terminology by Modeling	FORMS/FORMAT 2014	Braunschweig Sep 14
Alexander Nitsch, Benjamin Beichler, Frank Golatowski, Christian Haubelt	Model-based Systems Engineering with Matlab/Simulink in the Railway Sector	MBMV 2015	Chemnitz Mrz 15
Linh Vu Hong, Anne E Haxthausen and Jan Peleska	Formal Modeling and Verification of Interlocking Systems Featuring Sequential Release	FTSCS 2014	Luxembourg Nov 14
Michael Jastram	How the ReqIF Standard for Requirements Exchange Disrupts the Tool Market	Requirements Engineering Magazine	Mrz 14
Linh Vu, Anne Elisabeth Haxthausen, and Jan Peleska	Formal Modeling and Verification of Interlocking Systems Featuring Sequential Release	Science of Computer Programming	Mai 15
Jan Peleska, Cecile Braunestein, Uwe Schulze, Felix Hübner, Wen-ling Huang, Anne E. Haxthausen, Linh Vu Hong	A SysML Test Model and Test Suite for the ETCS Ceiling Speed Monitor	Tech. Report Uni-Bremen	Bremen Jul 15
Michael Jastram	openETCS – Eclipse in the Rail Domain	EclipseCon Europe	Ludwigsburg Nov 15
Michael Jastram	Solide Anforderungen dank ReqIF im europäischen Schienenverkehr	Tag des Systems Engineering (TdSE)	Ulm Nov 15
Thorsten Schulz, Frank Golatowski, Dirk Timmermann	ecure Privacy Preserving Information Beacons for Public Transportation Systems	Pervasive Computing and Communication Workshops (PerCom Workshops), IEEE International Conference	Sydney Mär 16
Klaus-Rüdiger Hase, Peter Mahlmann, Bernd Hekele, Michael Jastram, Jakob Gärtner, Stefan Karg	openETCS: Modellbasiert, agil und open Source; Ergebnisse aus dem ITEA2-Förderprojekt	43. Schienenfahrzeugtagung, Sonderheft ZEV	Graz Apr 16

11.4 Präsentationen und Vorträge

Tabelle 12. Präsentationen und Vorträge.

Autoren	Titel	Veranstaltung	Ort, Datum
Klaus-Rüdiger Hase, Jean Koulischer	openETCS: Open Source Prinzipien für das Europäische Zugsicherungssystem	Tagung Moderne Schienenfahrzeuge, ZEVrail	Graz 11
Klaus-Rüdiger Hase	openETCS	12th Signal+Draht Congress	Fulda Nov 12

Fortsetzung von Tabelle 12

Autoren	Titel	Rahmen	Ort, Datum
Klaus-Rüdiger Hase	Open Proofs Method within the European Train Control System	UIC ERTMS World Congress	Sweden Jun 12
Klaus-Rüdiger Hase	openETCS: Applying „Open Proofs“ to the European Train Control System	UIC Highspeed Congress	Philadelphia Jul 12
Marc Behrens	Early Verification of Concepts on the Example of openETCS	TEST4RAIL Symposium	Braunschweig Okt 13
Klaus-Rüdiger Hase	Den Vertrauenskrisen unserer Gesellschaft entgegenwirken! Ein möglicher Ausweg mit „open Proofs“ am Beispiel openETCS	safe.tech 2013	München Apr 13
Klaus-Rüdiger Hase	openETCS: Unsere Antwort auf die Vertrauenskrisen unserer Zeit	BITCOM OSS Forum	Berlin Jun 13
Klaus-Rüdiger Hase	Mit „open Proofs“ das Vertrauen in komplexe sicherheitskritische Systeme stärken: Am Beispiel des internationalen ITEA2-Projektes openETCS	DLR-Verkehrs-Kolloquium	Braunschweig Aug 13
Klaus-Rüdiger Hase	openETCS-ITEA2 an Overview	POLARSYS Workshop	Ludwigsburg Okt 13
Klaus-Rüdiger Hase	openETCS-ITEA2: An Open Source Software ETCS Project is making Progress	2013 International Rail Summit Berlin	Berlin Nov 13
Klaus-Rüdiger Hase	openETCS-ITEA2: Open Proofs becomes Reality in Railway Safety Application	ITEA2/ARTEMIS Co-Summit 2013	Stockholm Dez 13
Klaus-Rüdiger Hase	openETCS-ITEA2: An Open Source Software ETCS Project is making Progress	UIC ERTMS Workshop	Paris Jan 14
Klaus-Rüdiger Hase	openETCS: Erste Implementation für das openProofs-Konzept	AAET 2014 Braunschweig	Braunschweig Feb 14
Klaus-Rüdiger Hase	openETCS: First Implementation of the openProofs Approach	OpenUp-Camp	Nürnberg Feb 14
Klaus-Rüdiger Hase, Pierre Gaufillet	openETCS Tools going PolarSys Learning from the Aviation Sector	Smart Rail Conference	Amsterdam Feb 14
Klaus-Rüdiger Hase	Developing innovation and interchangeability with openETCS	Smart Rail Conference	Amsterdam Feb 14
Klaus-Rüdiger Hase	openETCS: European Train Control Crossing Borders through joint Software	embedded World, ITEA3 Kick-off	Nürnberg Feb 14
Linh Vu Hong, Anne E Haxthausen and Jan Peleska	Formal Verification of the Danish Railway Interlocking Systems. Admitted for presentation as „Ongoing research paper“, presented at AVOCS 2014	AVOCS2014	Enschede Sep 14
Uwe Steinke	SCADE in openETCS – the open source approach of ETCS, the European Train Control System standard	SCADE Academic Community Conference 2014	Berlin Dez 14

Fortsetzung von Tabelle 12

Autoren	Titel	Rahmen	Ort, Datum
Jan Welte	Agile Methods meet Safety	openETCS Workshop InnoTrans 2014	Berlin Sep 14
Jos Holtzer	Catapulting the railway industry into the 21st century	InnoTrans 2014 / Speakers' Corner	Berlin Sep 14
Klaus-Rüdiger Hase	openETCS: An Idea becomes Reality	InnoTrans 2014 / Speakers' Corner	Berlin Sep 14
Stefan Rieger	openETCS: Modelling and Formalisation for Safety and Interoperability	openETCS Workshop InnoTrans 2014	Berlin Sep 14
Stefan Rieger	Towards Interoperable Standards – The openETCS Approach	InnoTrans 2014 / Speakers' Corner	Berlin Sep 14
Alexander Nitsch	Towards model-based design for ETCS speed and distance monitoring	InnoTrans 2014 / Speakers' Corner	Berlin Sep 14
Christian Stahl	Petrinetze im industriellen Einsatz – Modellierung und Analyse des europäischen Zugsicherungssystems	Invited Talk at University of Augsburg	Augsburg Jan 15
Stefan Rieger	Formal Modeling and Analysis in Industry – Exemplary Application to the European Train Control System (ETCS)	DCON 2015	Rolduc Mrz 15
Jan Welte	Development of a railway application	The SCADE Academic Program Webinar	Apr 15
Klaus-Rüdiger Hase, Stefan Rieger	Mit Open Source zu einem einheitlichen Eisenbahnleitsystem in Europa	TWT Forum 2015	München Jun 15
Jan Welte	openETCS – Development of an open ETCS On-board reference model	InnoRail 2015	Budapest Okt 15
Frank Golatowski, Thorsten Schulz	Modeling Speed and Distance Monitoring of the European Standard for Train Control Systems ETCS	SCADE Academic Conference	Erlangen Sep 15
Tony Crabtree	Lessons learned from DB's openETCS initiative, available to CBTC users through an open source approach	3rd Annual Communication Based Train Control (CBTC)	London Mrz 16
Mairamou Haman Adjji, Jakob Gärtner	Skalierbarkeit von Simulation basierend auf formalen Modellen – Erfahrungen aus dem Bahnbereich	Modellierung 2016, Karlsruher Institut für Technologie	Karlsruhe Mrz 16
Klaus-Rüdiger Hase, Peter Mahlmann, Bernd Hekele, Michael Jastram, Jakob Gärtner, Stefan Karg	openETCS: Modellbasiert, agil und open Source; Ergebnisse aus dem ITEA2-Förderprojekt	43. Tagung Moderne Schienenfahrzeuge	Graz Apr 16

Literaturverzeichnis

- [1] Baro, Sylvain und Jan Welte: *WP2/D2.6-9 Requirements for openETCS*. Technischer Bericht, May 2013. https://github.com/openETCS/requirements/blob/master/D2.6-9/D2_6-9.pdf.
- [2] Baudin, Patrick und et al.: *ANSI/ISO C Spezification Language*. Technischer Bericht, CEA und INRIA, 2015. <http://frama-c.com/download.html>.
- [3] Behrens, Marc, Ana Cavalli, João Santos, Huu Nghia Nguyen, Stefan Rieger, Cécile Braunstein, Uwe Steinke, Benoît Lucet, Matthias Gudemann, Brice Gombault, Marielle Petit-Doche, Alexander Nitsch, Benjamin Beichler, Silvano Dal Zilio, Ning Ge und Marc Pantel: *D4.3.1 V&V report on the applicability of the V&V approach to the formal abstract model*. Technischer Bericht, 2015. <https://github.com/openETCS/validation/raw/master/Reports/D4.3/D4.3.1-Final-VV-report-on-model/D4.3.1.pdf>.
- [4] Boverie, Nicolas: *API Requirements for OpenETCS*. Technischer Bericht, September 2014. https://github.com/openETCS/requirements/blob/master/D2.7-Technical_Appendix/OETCS_API%20Requirements_v1.4.pdf.
- [5] Braunstein, Cecile: *D7.4: openETCS: openETCS Extended Tool Construction Set*. Technischer Bericht, 2013. <https://github.com/openETCS/toolchain/blob/master/Deliverables/D7.4.pdf>.
- [6] Braunstein, Cécile, Weng Ilng Huang, Felix Hübner und Jan Peleska: *A SysML Test Model and Test Suite for the ETCS Ceiling Speed Monitor*. Technischer Bericht, Universität Bremen, 2015. <https://github.com/openETCS/validation/blob/master/VnVUserStories/VnVUserStoryUniBremen/04-Results/ceiling-speed-monitoring-test-suites/main.pdf>.
- [7] Braunstein, Cecile, Jan Peleska, Stefan Rieger und Izaskun de la Torre: *D7.3: Toolchain Qualification Process Description*. Technischer Bericht, 2014. <https://github.com/openETCS/toolchain/blob/master/Deliverables/D7.3.pdf>.
- [8] DLR: *openETCS Architecture Verification*, 2015. <https://github.com/openETCS/validation/tree/master/VnVUserStories/VnVUserStoryDLR/05-Work/ArchitectureVerification>.
- [9] EBA: *Technische Grundsätze für die Zulassung von Sicherungsanlagen*. EBA, August 2003.
- [10] ERTMS Formal Specs GitHub Repository. <https://github.com/ERTMSSolutions/ERTMSFormalSpecs>.
- [11] ERA: *System Requirements Specification*, SUBSET-026, v3.3.0 Auflage, March 2012.
- [12] ERA: *Safety Requirements for the Technical Interoperability of ETCS in Levels 1 & 2*, SUBSET-091, v3.3.0 Auflage, May 2014.
- [13] Framework for Modular Analysis of C programs, 2015. <http://frama-c.com/>.

- [14] Gerlach, J., J. Burghardt, T. Lapawcyk, M. Behrens, H. Hungar und J. Peleska: *Final Report on Validation and Verification Report on Implementation/Code*. Technischer Bericht, Fraunhofer, DLR, Universität Bremen, 2016. <https://github.com/openETCS/validation/raw/master/Reports/D4.3/D4.3.2-Final-VV-report-on-code/D4.3.2.pdf>.
- [15] Gerlach, Jens und Izaskun de la Torre: *Preliminary Validation and Verification Report on Implementation/Code*. Technischer Bericht, 2015. <https://github.com/openETCS/validation/raw/master/Reports/D4.2/D4.2.2-VV-Implementation/D4.2.2.pdf>.
- [16] Hekele, Bernd und Stefan Karg: *D3.6.4: openETCS EVC Design Documentation*. Technischer Bericht, Deutsche Bahn, 2015. <https://github.com/openETCS/modeling/blob/master/deliverables/D3.6.4.docx>.
- [17] Hekele, Bernd und David Mentré: *openETCS API Documentation*. Technischer Bericht, Deutsche Bahn, 2015. https://github.com/openETCS/modeling/blob/master/openETCS%20ArchitectureAndDesign/D3.5.4%20_API/D3.5.4-API.pdf.
- [18] Hungar, Hardi: *Definition of the openETCS Development Process*. Technischer Bericht, 2015. https://github.com/openETCS/requirements/raw/master/D2.3/D2_3a_03.pdf.
- [19] Jastram, Michael: *D7.5: Ecosystem Artifacts*. Technischer Bericht, 2015. <https://github.com/openETCS/toolchain/blob/master/Deliverables/D7.5.pdf>.
- [20] Jastram, Michael und Marielle Petit-Doche: *D7.1: Report on the Final Choice of the Primary Toolchain*. Technischer Bericht, 2014. <https://github.com/openETCS/toolchain/blob/master/Deliverables/D7.1.pdf>.
- [21] Mahlmann, Peter: *The openETCS Scrum-of-Scrum Process*, 2014. <https://github.com/openETCS/product-backlog/wiki/Scrum-of-scrum-process>.
- [22] Mahlmann, Peter, Bernd Hekele, Peyman Farhangi, Uwe Steinke, Christian Stahl, Jakob Gärtner, Thorsten Schulz, Marielle Petit-Doche und Alexander Stante: *D3.5.4: openETCS Architecture and Design Specification*. Technischer Bericht, Deutsche Bahn, 2015. <https://github.com/openETCS/modeling/blob/master/deliverables/D3.5.4.pdf>.
- [23] Martija, Amaia und Izaskun de la Torre: *Tool Chain Test Plan*. Technischer Bericht, 2014. https://github.com/openETCS/toolchain/blob/master/T7.3/TestPlan/OpenETCS_Toolchain_TestPlan.pdf.
- [24] Mentre, David, Stanislas Pinte und Guillaume Pottier: *D2.5 A Subset of Requirements for Benchmarking of Tools*. Technischer Bericht, April 2013. <https://github.com/openETCS/requirements/blob/master/D2.5/D2.5SubsetofRequirementsforBenchmarkingofTools.pdf>.
- [25] MISRA: *Guidelines for the use of the C language in critical systems*. MISRA, Oktober 2004, ISBN 978-0-9524156-2-6.
- [26] Normalisation Electrotechnique, Comité Européen de: *Railway applications - Communication, signalling and processing systems - Software for railway control and protection systems*, EN 50128. EUROPEAN STANDARD, Juni 2011.
- [27] Official Journal of the European Union: *Commission Decision of 6 November 2012 amending Decision 2012/88/EU on the technical specifications for interoperability relating to the control-command and signalling subsystems of the trans-European rail system*

- (notified under document C(2012) 7325). Technischer Bericht, European Commission, 2012. <http://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=OJ:L:2012:311:FULL&from=EN>.
- [28] openETCS SCADE model, 2014. <https://github.com/openETCS/modeling/tree/master/model/Scade/System>.
 - [29] openETCS SysML model, 2015. https://github.com/openETCS/modeling/tree/master/model/sysml/openETCS_EVC.
 - [30] openETCS Webseite. <http://openetcs.org>.
 - [31] Open Graphics Library Webseite. <https://www.opengl.org/>.
 - [32] Petit-Doche, Marielle: D7.2: Report on all aspects of secondary tooling. Technischer Bericht, 2014. <https://github.com/openETCS/toolchain/blob/master/Deliverables/D7.2.pdf>.
 - [33] Petit-Doche, Marielle und Matthias Gudemann: D2.3 openETCS Process. Technischer Bericht, June 2013. https://github.com/openETCS/requirements/blob/master/D2.3/D2_3.pdf.
 - [34] Petit-Doche, Marielle, David Mentre und Matthias Gudemann: D2.4 openETCS Methods. Technischer Bericht, January 2014. https://github.com/openETCS/requirements/blob/master/D2.4/D2_4.pdf.
 - [35] Pokam, Merlin und Norbert Schäfer: D2.2 Report on CENELEC standards. Technischer Bericht, April 2013. <https://github.com/openETCS/requirements/blob/master/D2.2/D2.2.pdf>.
 - [36] Railway Simulation and Testing ® Laboratory. <https://www.railsite.de/>.
 - [37] Rieger, Stefan und Marc Behrens: ETCS Specification Findings. Technischer Bericht, TWT, DLR, 2015. <https://github.com/openETCS/validation/raw/master/VnVUserStories/ModelVerificationTWT/04-Results/SpecificationFindings/SpecificationFindings.pdf>.
 - [38] Silvano Dal Zilio, Marc Behrens, Raphaël Faudou, David Mentré und Xavier Zeitoun: Migration from Scade to an Open Alternative. Technischer Bericht, 2015. https://github.com/openETCS/toolchain/blob/master/T7.3/MigrationScade2OpenAlternative/OpenETCS_MigrationScade2OpenAlternative.pdf.
 - [39] Symposium Test4Rail on Testing of safety-critical, software-based railway systems, Oktober 2015. <https://www.test4rail.org/>.
 - [40] Torre, Izaskun de la: openETCS Quality Assurance Plan, 2013. https://github.com/openETCS/governance/blob/master/QA%20Plan/D1.3.1_QA_Plan.pdf.
 - [41] openETCS User Stories. <https://github.com/openETCS/modeling/tree/master/User%20Stories>.
 - [42] Vallée, Frédérique und Norbert Schäfer: Independant Assessment according to the standard EN 50128:2011. Technischer Bericht, All4tec and AEbt, 2016. https://github.com/openETCS/internal-assessment/blob/master/Assessment_Report/Assessment_Report.pdf.

- [43] Welte, Jan und Hansjörg Manz: *D2.1 Report on existing methodologies*. Technischer Bericht, January 2013. https://github.com/openETCS/requirements/blob/master/D2.1/dkn_2012-12-23_Report-WP2-T2-1-1_23_jw.pdf.