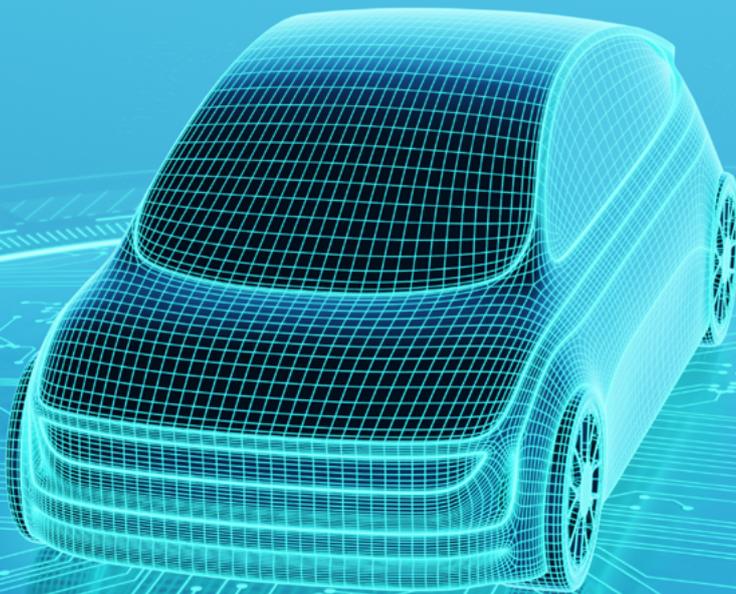


POSITIONSPAPIER: CHANCEN DURCH STANDARDISIERUNG UND OPEN-SOURCE-SOFTWARE IM MOBILITÄTSSEKTOR

KARLSRUHE UND STUTTGART | FEBRUAR 2024

AUTOR

Hussem Guissouma
Forschungsfeldkoordinator des
InnovationsCampus Mobilität der Zukunft



POSITIONSPAPIER: CHANCEN DURCH STANDARDISIERUNG UND OPEN-SOURCE-SOFTWARE IM MOBILITÄTSSEKTOR

Autor: Housseem Guissouma, Forschungsfeldkoordinator des InnovationsCampus Mobilität der Zukunft (Forschungsfeld „Software-System-Architectures“)

Abstract

Die Automobilbranche erlebt eine radikale Veränderung durch die wachsende Anzahl von Softwarefunktionen und -Services in den Mobilitätssystemen (bis zu 100 Mio. Codezeilen und 40.000 Signale in einem Oberklassefahrzeug). Trotz ihrer Vorteile, wie der Ermöglichung kontinuierlicher Systemverbesserungen, können die verteilten Softwarekomponenten bei fehlerhafter Interaktion oder bei unvorhergesehenen Situationen die Sicherheit der Fahrzeuge und ihrer Umgebung gefährden. Um das zu verhindern und die Entwicklungsaufwände beherrschbar zu erhalten, müssen die bestehenden **Standards für Elektrik/Elektronik-Architekturen** konsolidiert und weiterentwickelt werden. Free and **Open-Source-Software (FOSS)** (z.B. das Linux-Betriebssystem), spielt bereits heute eine wichtige Rolle bei der Implementierung von Fahrzeugfunktionen und kann an dieser Stelle einen Beitrag dazu leisten. Es ist jedoch fraglich, ob FOSS in der Bilanz ein Fluch oder ein Segen für die standardisierte Entwicklung des **Software-Defined Vehicle (SDV)** darstellt.

Die Themen **Standardisierung** und **FOSS** im Mobilitätssektor wurden im Rahmen eines Workshops für Standardisierte Automotive Software (SAS) von renommierten **Experten aus Forschung und Industrie** intensiv diskutiert. Die Veranstaltung wurde vom InnovationsCampus Mobilität der Zukunft (ICM) (s. Anhang) am 30.11.2023 im Monbachtal organisiert. Ausgehend aus den Ergebnissen des Workshops wurden Empfehlungen für notwendige Maßnahmen abgeleitet. Die Empfehlungen umfassen im Wesentlichen eine stärkere Förderung von Forschungs- und Standardisierungsarbeiten in den softwarebezogenen Themen (z.B. Schnittstellenspezifikation und Datensicherheit) sowie eine Harmonisierung der bestehenden Gremien und FOSS-Aktivitäten auf nationaler und internationaler Ebene.

Inhaltsverzeichnis

1	Einführung.....	3
2	Standardisierte Softwarefunktionen im Automobil.....	4
2.1	Software-bezogene Standards, Normen und Regulierungen.....	4
2.2	Standardisierte Anforderungen an Softwarekomponenten.....	6
2.3	Anforderungen an standardisierte Testverfahren.....	7
3	Open-Source-Software im SDV.....	7
3.1	Wo wird FOSS in der Automobilindustrie verwendet?.....	7
3.2	Open-Source-Anwendungsmodelle.....	8
3.3	Vorteile und Herausforderungen von FOSS.....	8
4	Empfehlungen für Handlungsmaßnahmen.....	9
4.1	Notwendige technische Entwicklungen.....	10
4.2	Kooperations- und Gremienarbeit.....	10
5	Zusammenfassung.....	11
	Danksagung.....	12
	Literaturverzeichnis.....	13
	Anhang - Über den InnovationsCampus Mobilität der Zukunft.....	14

1 Einführung

In der Elektrik/Elektronik (E/E)-Architektur eines Oberklassefahrzeuges befinden sich heutzutage bis zu 150 Steuergeräte, Electronic Control Units (ECUs) genannt, und mehr als 100 Mio. Codezeilen [1]. Auf jedem dieser Steuergeräte werden unterschiedliche Softwarekomponenten und -module ausgeführt, die im Verbund zahlreiche Fahrzeugfunktionen realisieren. Während sich der Trend der E/E-Architekturen in den letzten fünf Jahren von einer starken Verteilung hin zu einer Domänen- und Zonenzentralisierung verschiebt (s. Abbildung 1) nimmt die Gesamtanzahl an Softwarekomponenten und ihr Umfang (in Codezeilen) weiter zu [2].

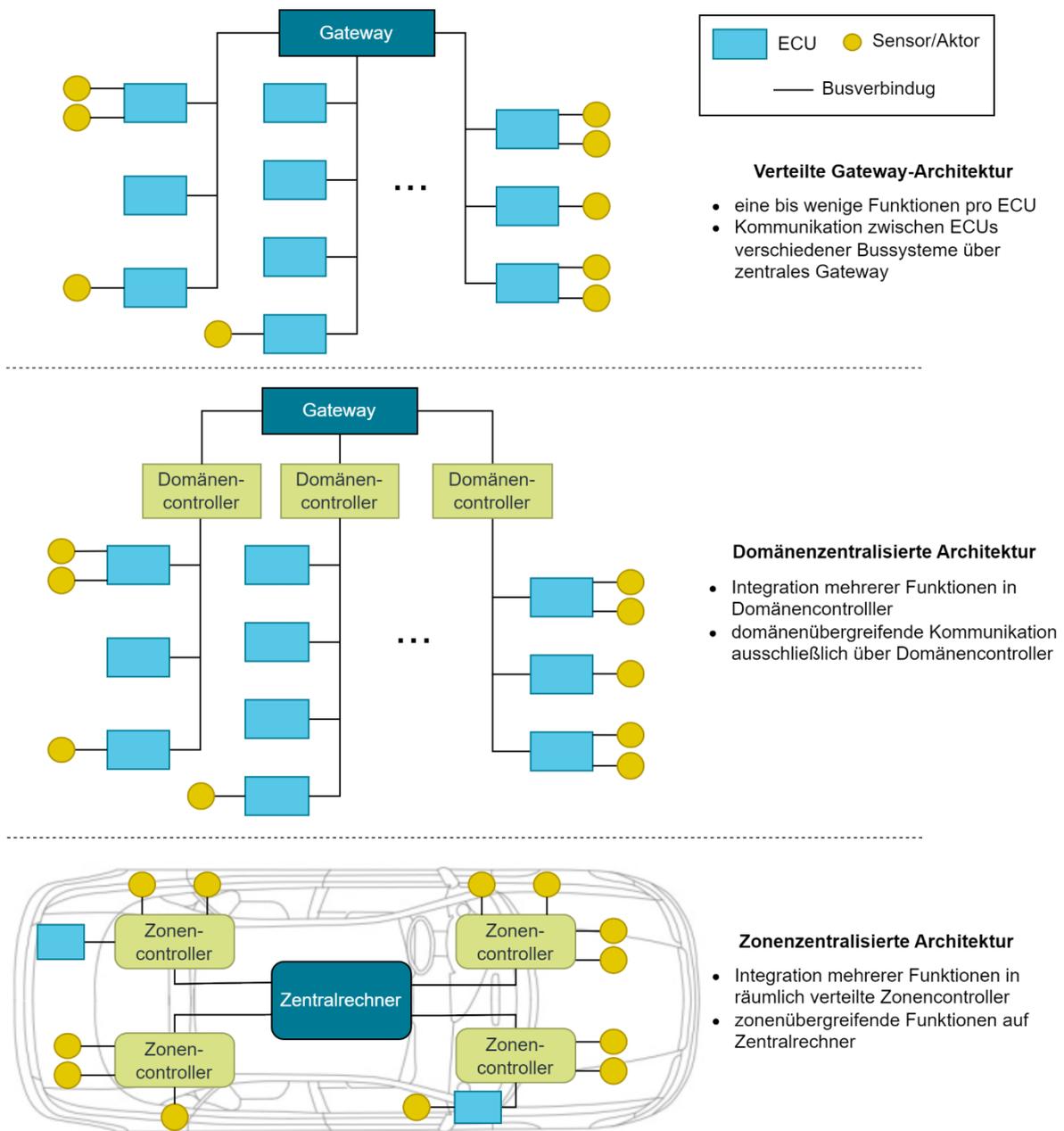


Abbildung 1 Evolution der E/E-Architekturen von einer verteilten Gateway-Architektur zu einer Domänen- und Zonenorientierten Zentralisierung (nach [3])

Die zusammenhängenden kundenerlebbaren Fahrzeugfunktionen (*Features*) werden verschiedenen Domänen zugeordnet. Typische Domänen sind Chassis (Fahrwerk), passive und aktive Sicherheit, Fahrassistenz (ADAS) und Infotainment.

Heutzutage ist die Software eines Fahrzeuges nicht mehr auf die klassischen Domänen begrenzt. Sie ist vielmehr mit unterschiedlichen Services, die in der Edge¹, z.B. intelligente Infrastruktursensoren, und in Backend-Cloudsystemen beheimatet sind, verbunden und bietet den Nutzern ein ganzheitliches digitales Erlebnis.

„Analog zur historischen Entwicklung von Mobiltelefonen zum Smartphone sind 80 Prozent der Softwarefunktionen eines modernen Fahrzeugs keine klassische Fahrfunktionen mehr. Der große Unterschied liegt jedoch darin, dass Fahrzeuge für alle Verkehrsteilnehmer sicher sein müssen.“

Dr. Christian Müller, ZF Group

Eine Chance, die Entwicklung des softwaredefinierten Fahrzeuges zu beschleunigen, ist die Harmonisierung und stärkere Förderung (durch Politik und Industrie) von Standardisierungs- und Free and Open-Source-Software (FOSS)-Initiativen im Automobilbereich.

2 Standardisierte Softwarefunktionen im Automobil

Eine erfolgreiche Standardisierung der Software setzt die Vereinheitlichung der darunterliegenden Hardware (eingebettete elektronische Systeme) voraus. Diese muss mit standardisierten elektronischen Komponenten (Prozessoren, Speicher, etc.) und den notwendigen Kommunikationsschnittstellen (drahtgebunden und drahtlos) ausgestattet sein. Die Hardware-Standardisierung ist jedoch nicht im Fokus dieses Papiers und wird deshalb im Folgenden nicht erörtert.

2.1 Software-bezogene Standards, Normen und Regulierungen

Definition 1 – Standard, Norm und Regulierung

Während Standards auf der Einigung von Einzelgruppen ohne Einbeziehung der Öffentlichkeit beruhen, werden Normen von allen interessierten Kreisen gemeinsam erarbeitet und im Konsens eines öffentlichen Einspruchsverfahrens verabschiedet. Somit ist jede Norm auch ein Standard, aber das Gegenteil ist nicht valide.

Regulierungen wohingegen sind verpflichtend und müssen, z.B. bei Homologationsprozessen, eingehalten werden. [4]

Standards, Normen und Regulierungen mit Softwarebezug sind im Automobilbereich vertreten. Tabelle 1 gibt Beispiele dafür.

¹ Beim Edge Computing werden die Daten näher an der Quelle der Datenerzeugung verarbeitet, anstatt sich ausschließlich auf einen zentralen Cloud-Server zu verlassen.

Tabella 1 Beispiele von Standards, Normen und Regulierungen mit Softwarebezug im Automobilbereich

Standards	Normen	Regulierung
AUTOSAR	ISO 26262	UNECE R155
V-Modell (Vorgehensmodell)	ISO/PAS 21448 (als SOTIF bekannt)	UNECE R156
Scrum (Vorgehensmodell)	ISO 11898-2 (High-speed-CAN)	EU „Cyber Resilience Act“
TISAX (Trusted Information Security Assessment eX-change)		

Weitverbreitete Vorgehensmodelle, wie das V-Modell oder Scrum, sind als Standards zu betrachten, da diese von *Original Equipment Manufacturers* (OEMs) und Zulieferern nach Bedarf angepasst werden können. Andere Standards beschäftigen sich mit spezifischen Qualitätsaspekten, wie TISAX [5], das sich auf die Datensicherheit (Security) konzentriert.

Zu den Normen zählen die ISO 26262 und ISO 21448, die ein Regelwerk für die Entwicklung und Validierung von sicherheitskritischen Funktionen definieren. Andere Normen fassen international anerkannte technische Spezifikationen zusammen, wie bspw. ISO 11898-2, die die technischen Aspekte des CAN-Buses dokumentiert.

Die UNECE R155 und R156 über den Umgang mit Risiken der Cybersicherheit und Over-The-Air (OTA)-Updates, die seit dem Jahr 2022 für neuzugelassene Fahrzeuge wirken, sind Beispiele für Regulierungen. Sie sind für alle Mitgliedsländer der vereinigten Nationen verpflichtend. Ein anderes Beispiel ist das aktuell auf EU-Ebene in finaler Verabschiedungsphase „*Cyber Resilience Act*“ [6].

Nach Definition 1 gilt die AUTOSAR-Softwarearchitektur als Standard. AUTOSAR (AUTomotive Open System Architecture) ist ein Konsortium aus verschiedenen Automobilherstellern und -Zulieferern mit dem Ziel, die Software von ECUs für die Senkung der Herstellungskosten und Erhöhung der Kompatibilität zu standardisieren [7]. Dies ist dadurch gelungen, dass die Applikationssoftware möglichst unabhängig von der darunterliegenden Hardware spezifiziert wurde. Dies gilt insbesondere für die Classic-Version, die sich auf ECUs mit signalorientierter² Kommunikation konzentriert. Dabei wird die Software in drei Schichten von der hardwarenahen Basissoftware bis zur abstrahierten Applikationsschicht unterteilt (s. Abbildung 2).

OTA-Updates, Serviceorientierung und Cloud-Integration haben Einzug in das Fahrzeug gehalten und sind im Software-Defined Vehicle (SDV) der Zukunft nicht mehr wegzudenken. Das zukünftige Fahrzeug muss also *End-to-End*, mit einer Verankerung der Edge- und Cloud-Komponenten, betrachtet werden. Mit AUTOSAR Adaptive gibt es bereits erste Versuche, die

² Bei der signalorientierten Kommunikation basiert der Informationsaustausch zwischen Softwarekomponenten auf physikalischen Signalen (z.B. CAN-Botschaften), die von einem Sendersteuergerät zu einem oder mehreren Empfängern gesendet werden. Im Gegensatz dazu stehen bei Serviceorientierten Architekturen Dienste im Mittelpunkt, die von den Softwarekomponenten angeboten und genutzt werden können.

beschriebenen Herausforderungen mit Hilfe der SOME/IP-Middleware und anderen standardisierten Services zu bewältigen.

„AUTOSAR standardisiert seit 20 Jahren erfolgreich die Software-Architektur für klassische Steuergeräte (Classic Platform). Um im Kontext SDV eine vergleichbare Rolle einzunehmen (Adaptive Platform), muss sich AUTOSAR weiterentwickeln und teilweise neu erfinden.“

Dr. Marc Weber, Vector Informatik

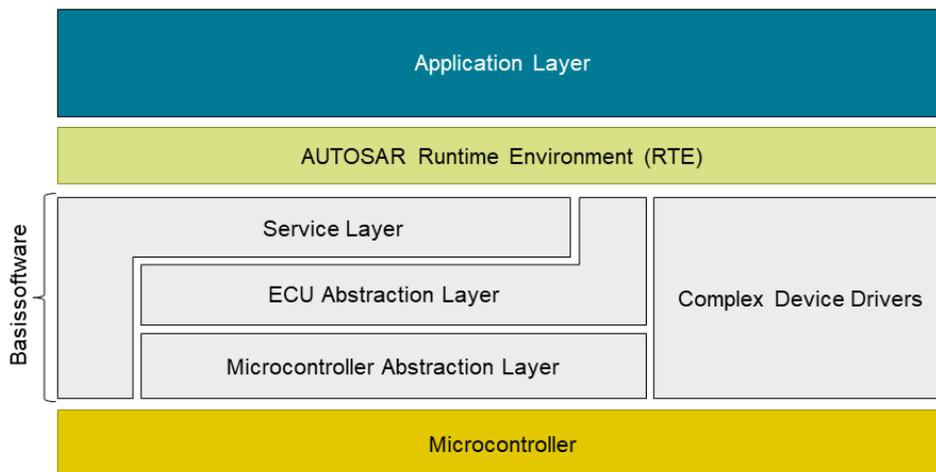


Abbildung 2 Schichten der AUTOSAR-Architektur (Classic-Plattform)

2.2 Standardisierte Anforderungen an Softwarekomponenten

Bestehende Austauschformate, wie der XML-basierte ReqIF-Standard, erfassen menschenlesbare Anforderungen samt zugehörige Metadaten. Sie stellen also eine Austauschgrundlage dar, ermöglichen aber keine automatisierten Analysen (z.B. systematische Verifikation der Architekturkorrektheit).

Standardisierte Anforderungen unterscheiden zwischen Annahmen an die Umgebung der Komponenten und Garantien, die die Komponenten unter ihren spezifizierten Betriebsbedingungen gewährleisten. In diesen Anforderungen müssen neben den funktionalen Aspekten alle anderen relevanten Faktoren, wie Echtzeitverhalten und Ressourcenverbrauch, erfasst werden. Die erarbeiteten Eigenschaften eines standardisierten Formats für maschinenlesbare Anforderungen können in den folgenden Punkten zusammengefasst werden:

- Minimalforderung: statische Schnittstellenbeschreibung, welche zumindest die vorhandenen Eingangs- und Ausgangsvariablen mit ihren Typen beschreiben sollte.³
- Quality-of-Service der Kommunikation mit der Softwarekomponente, z.B. Jitter, Verzögerung und Bandbreite. Damit einher geht die Deployment-Spezifikation, welche den Verbrauch von Ressourcen (z.B. CPU, Speicher, Netzwerk) beschreibt, ebenso wie die Abhängigkeiten und Erreichbarkeiten von anderen Komponenten.

³ In der Praxis ist selbst dies aktuell oft nicht gegeben.

- Betriebssicherheit (funktionale Korrektheit): für allgemeines funktionelles Verhalten gibt es eine breite Spannweite von Formalismen, die sich für unterschiedliche Softwaredesignprinzipien eignen. Für zustandsbehaftete Systeme kann zum Beispiel das Zustandsdiagramm ein geeignetes Werkzeug sein.
- Informationssicherheit: Softwarekomponenten können spezifizieren, welche Schutzziele durch welche Maßnahmen erreicht werden. Zum Beispiel kann Verschlüsselung von Kommunikation einen Angriffsvektor auf die Vertraulichkeit aushebeln.

Eigenschaften wie die Absicherung auf ASIL-D-Niveau können nur unter Betrachtung von Kompositionen von Komponenten (s.g. Wirkketten) beurteilt werden und bedürfen daher einer Spezifikation und Prüfung der Komposition [8].

2.3 Anforderungen an standardisierte Testverfahren

Da einige Automobilfunktionen, wie der Notbremsassistent, direkte oder indirekte Auswirkungen auf das sicherheitskritische Verhalten des Fahrzeuges haben, müssen sie vor ihrer Freigabe ausführlich getestet werden. Die kurzen Updatezyklen und der existierende Variantenreichtum im Feld, als Beispiel 425 Milliarden Varianten des Bordnetzes allein beim VW Golf 7 [9], stellen wesentliche Herausforderungen für diesen Testprozess dar.

Derzeit werden die Testfälle auf Modul- oder Komponentenebene (Unit Tests), im Verbund (Integrationstests) und auf Systemebene in Form von Szenarien durchgeführt. Die Auswahl von Testfällen und Konfigurationen der Testsysteme erfolgt meist noch mit Hilfe von Brute-Force-Methoden. Simulationsbasiertes Testen wird als unterstützende Methode verwendet. Jedoch muss die Funktionsweise zur Homologation auf realen Teststrecken nachgewiesen werden.

Für die Etablierung von standardisierten Testabläufen müssen die Rollen, die im Testprozess involviert sind, klar definiert werden. Außerdem müssen einheitliche Austauschformate für Testfälle und -szenarien in diesen Prozess integriert werden. Eine weitere Anforderung ist die Anwendung von Methoden und Technologien aus der Forschung, wie formaler Verifikation, KI-Verfahren und delta-basierte Tests, im industriellen Umfeld. Letzteres ist besonders für die inkrementelle Verifikation und Validierung von Updates notwendig [10]. Für das Testverfahren sind neben den realen Systemen auch digitale Zwillinge erforderlich. Diese stellen eine virtuelle Repräsentation der physikalischen Assets der verwalteten Systeme dar [11]. Schließlich sollen die Testprozesse durch Integration in CI/CD-Pipelines⁴ automatisiert werden.

3 Open-Source-Software im SDV

FOSS garantiert, neben dem Zugriff auf den Quellcode der Software, auch dessen kostenfreien Verwendung. Linux, Python, Mozilla Firefox, Chromium, Docker, Kubernetes und ROS sind nur einige Beispiele.

3.1 Wo wird FOSS in der Automobilindustrie verwendet?

Eine aktuelle Studie von Zhang u. a. [12] konnte empirisch ermitteln, dass mindestens 16% aller Binärdateien der eingebetteten Steuergerätesoftware in kommerziellen Fahrzeugen aus Open-

⁴ Eine CI/CD-Pipeline ist eine automatisierte Prozesskette in der Softwareentwicklung, die Continuous Integration (regelmäßiges Zusammenführen von Code) und Continuous Delivery (automatisiertes Bereitstellen und Testen) umfasst, um die Effizienz und Qualität der Softwareentwicklung zu steigern.

Source-Komponenten besteht. Dies ist kein überraschendes Ergebnis, da zum Beispiel die meisten Betriebssysteme moderner Steuergeräte und High Performance Computers (HPCs) Linux-basiert sind. Auch zahlreiche frei verfügbare Bibliotheken, wie bspw. OpenSSL für die verschlüsselte SSH-Kommunikation, finden Anwendung im Automobilbereich. Darüber hinaus bieten FOSS-Projekte, wie Apollo⁵ oder Autoware⁶, Sammlungen von prototypischen Implementierungen der gesamten Funktionskette des automatisierten Fahrens: von der Perzeption bis zur Aktorsteuerung.

Der Nutzen von Open-Source ist nicht nur auf die tatsächlich eingebettete Software begrenzt. Er deckt zudem verschiedene Werkzeuge ab, die entlang der Entwicklungs- und Wartungsprozesse verwendet werden. So gehören verschiedene Open-Source-Frameworks für die Erstellung von Algorithmen des maschinellen Lernens, z.B. TensorFlow und PyTorch, zum Stand der Technik im Bereich der Perzeptionsfunktionen. Während der Implementierungsphase werden Werkzeuge, wie die Compiler GCC und LLVM-Clang, intensiv verwendet. Für die Verteilung und Orchestrierung der fertig implementierten Softwarekomponenten sind FOSS-Werkzeuge, wie Docker und Kubernetes, vom Prozess des Lebenszyklusmanagement nicht mehr wegzudenken.

Der Open-Source-Gedanke betrifft außerdem die gesammelten Sensor- und Kontextdaten, die vor allem für die Entwicklung von KI-Funktionen oder die Entwicklung von digitalen Zwillingen eine entscheidende Rolle spielen. Beispiele hierfür sind KITTI oder Waymo Open Dataset.

3.2 Open-Source-Anwendungsmodelle

Im Kontext von Mobilitätssystemen gibt es zwei Hauptkooperationsmodelle für Open-Source-Projekte. Auf der einen Seite können die involvierten interessierten Gruppen lediglich gemeinsame Spezifikationsdokumente erarbeiten und veröffentlichen, ohne eine bestimmte Implementierung vorauszusetzen. Dieses Modell wird *Collaboration on Specification/Standards* genannt und wird beispielweise von AUTOSAR verfolgt.

Auf der anderen Seite gibt es das Modell *Collaboration on Implementation*, das dem klassischen FOSS-Modell aus dem IT-Bereich entspricht. Hier wird der Quellcode gemeinsam von verschiedenen Beitragenden entwickelt und auf Plattformen wie GitHub öffentlich unter einer vordefinierten Lizenz zur Verfügung gestellt. Die Arbeitsgruppe *Eclipse SDV*⁷, als Beispiel, verfolgt dieses Modell, das die „Code-First“-Mentalität in den Vordergrund stellt. Ein Vorteil dieses Modells gegenüber des *Collaboration on Specification* ist die schnellere Time-to-Market.

3.3 Vorteile und Herausforderungen von FOSS

Tabelle 2 zeigt eine Übersicht von Vorteilen und Herausforderungen, die durch FOSS-Komponenten in der Automobilindustrie entstehen.

⁵ <https://developer.apollo.auto/>

⁶ <https://autoware.org/>

⁷ <https://sdv.eclipse.org/>

Tabelle 2 Vorteile und Herausforderungen von FOSS im Automobilbereich

Bereich/Aspekt	Vorteile/Chancen von FOSS	Herausforderungen von FOSS
Wettbewerbs-differenzierung	Einige Standard-Softwarekomponenten im Fahrzeug, wie z.B. Kommunikationsdienste oder Aktorsteuerung, sind in der SDV-Zeit nicht mehr wettbewerbs-differenzierend. Diese können durch FOSS stärker vereinheitlicht werden.	Die angestrebten Geschäftsmodelle müssen im Vorfeld genau definiert werden, damit die Wettbewerbsdifferenzierung zwischen den OEMs und Zulieferern erhalten bleibt.
	Kooperation als Wettbewerbsvorteil	
Innovations-potentiale	Open-Source-Projekte nutzen in der Regel die neusten Ansätze und Innovationen aus dem Stand der Wissenschaft und Technik.	Die innovativen Ansätze der FOSS-Projekte (z.B. Formate der Datensätze) sollen vordefinierten Standards entsprechen, um potentielle Inkonsistenzen und Unsicherheiten zu minimieren.
Effiziente Entwicklung	FOSS kann die agile Entwicklung von Prototypen und die Verbreitung der Software-Assets in der Forschungslandschaft fördern.	
Lizensierung und Haftung		Lizenzen können Regeln enthalten, die eine Anwendung in kommerziellen Produkten ausschließt.
	Entwicklung von Add-On-Modellen (Weiterentwicklung von FOSS-Komponenten als kommerzielles Produkt) minimiert die Risiken bei der Haftungsfrage.	Die Anwendung von FOSS nimmt die Haftungsverantwortlichkeit vom Anwender (OEM, Zulieferer) nicht ab.
Wartung und Weiterentwicklung	Größerer Kreis von Beitragenden kann die Wartungsprozesse beschleunigen.	Anwender (z.B. OEM oder Zulieferer) sollen beitragen. Hierfür ist ein Prozess auf Organisationsebene notwendig, um die Strategie und Rahmenbedingungen der Beiträge festzulegen.
Qualität (Safety und Security)	Offenlegung der Logik der Datenverarbeitung → transparenter Datenschutz	Angreifer, die Zugriff zum Quellcode haben, können die Sicherheitslücken in der Programmlogik einfacher nutzen.

4 Empfehlungen für Handlungsmaßnahmen

Die folgenden Empfehlungen sollen dazu beitragen, die Wettbewerbsfähigkeit und Unabhängigkeit der europäischen Automobilindustrie, insbesondere gegenüber amerikanischen und chinesischen Technologieanbietern, zu erhöhen. Sie sollen vor allem in Deutschland und Europa wirken.

4.1 Notwendige technische Entwicklungen

Die Standardisierung der Softwareschnittstellen mit Berücksichtigung der neuen E/E-Architekturen (Cloud-Anbindung, Serviceorientierung und HPC) wird benötigt, um externe Services als neues Geschäftsmodell einzubinden (analog zum „App-Store“ in Smartphones) [2]. Eine Herausforderung, die hierbei von Grund auf berücksichtigt werden muss, ist die Minimierung der Risiken durch Cybersicherheit in der offenen Softwarearchitektur. Hierfür ist eine Weiterentwicklung der bestehenden Standards, wie AUTOSAR, notwendig.

Der Fokus der zukünftigen Forschung muss auf der Entwicklung von sicherheitskritischen und automatisierten Fahrfunktionen liegen, denn die bestehenden Ansätze aus der IT-Welt sind für Komfort- und Infotainment-Funktionen übertragbar.

Zusätzlich zu den beiden oben genannten Maßnahmen müssen die folgenden Themen stärker erforscht und im industriellen Umfeld erprobt werden:

- Standardisierung der Entwicklung und Wartung von digitalen Zwillingen
- Homogenisierung der kritischen CI/CD-Prozesse
- Gewährleistung der Datenschutz-Anforderungen auf Fahrzeug und auf Digital-Twin-, wie von der Regulierung UNECE R155 gefordert
- Methoden und Prozesse für das Management und die Validierung von sicherheitskritischen OTA-Updates, wie von der Regulierung UNECE R156 gefordert
- Aussagekräftige Softwarebeschreibungen, die sich für maschinelle Analysen eignen.

Bei all diesen Forschungsbedarfen können Open-Source-Lösungen bei der schnellen prototypischen Umsetzung und bei unkomplizierten Kooperationsmodellen unterstützen.

4.2 Kooperations- und Gremienarbeit

Es gibt sowohl auf nationaler als auch auf internationaler Ebene bereits eine Vielzahl von Gremien und Kooperationsgruppen im SDV-Bereich (s. Abbildung 3).

Die existierenden Gremien müssen konsolidiert und fokussiert werden. Hierbei sollen die Forschungs- und Entwicklungsschwerpunkte auf die innovativen Softwarethemen gelegt werden. Auch die europäischen Unternehmen müssen stärker kooperieren. Open-Source-Initiativen können die Zusammenarbeit an dieser Stelle vereinfachen. Diese benötigen allerdings rechtliche Rahmenbedingungen für die problemfreie Entwicklung und Anwendung von Open-Source-Komponenten.

„Europäische Unternehmen müssen stärker kooperieren, um vor allem die Integrationsaufwände zu reduzieren.“

Thorge Erichsen, Mercedes-Benz

Die bestehenden FOSS-Initiativen, wie die Arbeitsgruppe Eclipse SDV, sollen auf europäischer Ebene sowohl von den Industrieunternehmen als auch von der Politik unterstützt werden. Geschlossene Kooperationsprojekte, wie das Betriebssystem MBOS [13] von Mercedes sollten auf einen größeren Kooperationskreis ausgeweitet werden. Gleichzeitig sollen die neuen Zusammenarbeitsmodelle zwischen einem oder mehreren OEMs und anerkannten IT-Unternehmen

weiter ausgebaut werden. Diese sollen es jedoch ermöglichen, dass beide Seiten technologisch von ihren gegenseitigen Erfahrungen profitieren.

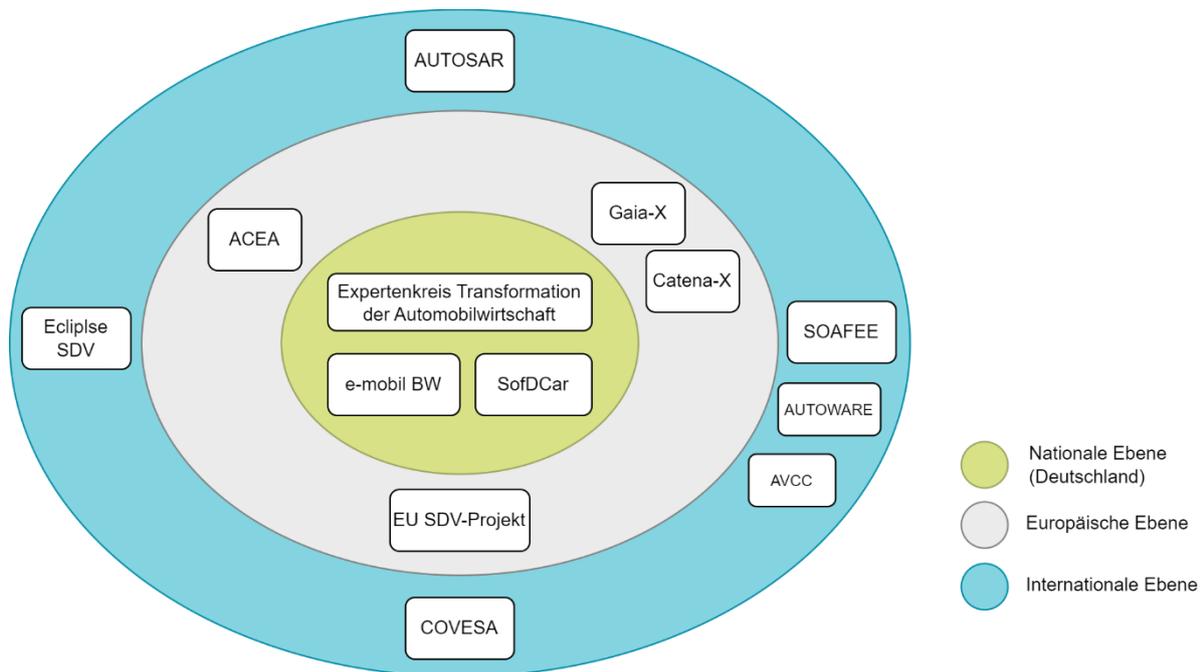


Abbildung 3 Übersicht über bekannte Gremien- und Kooperationsgruppen im SDV-Bereich

Für jede Gremiengruppe braucht man mindestens eine Partei, die die Rolle einer „Lokomotive“ annimmt und das Gesamtbild skizziert und strukturiert. Idealerweise sollten das Industriepartner sein, wie zum Beispiel OEMs.

Die identifizierten Forschungsbedarfe (s. Abschnitt 4.1) sollen von der Politik und Industrie unterstützt und vorangetrieben werden. Insbesondere ist es zu empfehlen, Startup-Initiativen im Bereich der Automotive-IT stärker zu fördern.

Eine weitere wichtige Maßnahme ist die Stärkung aktuellen Kooperationen zwischen Universitäten und Industrie. Die Themen der Automotive-IT müssen in den Ingenieurstudiengängen verankert und die Branche attraktiviert werden. Auf der anderen Seite müssen die bürokratischen Hürden bei gemeinsamen Forschungsprojekten reduziert werden, damit die Forschungsergebnisse anhand von realen und aktuellen Daten aus der Industrie validiert werden können.

„Die Zusammenarbeit mit Hochschulen in Forschungsprojekten muss durch weniger Bürokratie und einen Austausch von realen Daten und Erfahrungen verbessert werden.“

Prof. Dr. Eric Sax, Karlsruher Institut für Technologie

5 Zusammenfassung

Der Wandel von klassischen mechatronischen Systemen hin zu software-definierten Fahrzeugen stellt die deutsche und europäische Industrie vor eine starke Konkurrenz mit neuen Playern. Diese sind hauptsächlich der schnell wachsende asiatische Markt, angeführt von chinesischen Automobilunternehmen, und etablierten IT-Unternehmen aus Nordamerika. Um mithalten zu können, spielen die Standardisierung der Automobilsoftware und die Förderung und

strategische Nutzung von Open-Source-Lösungen eine Schlüsselrolle. Dies trägt wiederum zur Sicherung von Arbeitsplätzen in Europa bei.

„Moving up the value chain funktioniert immer. Man muss aber die richtigen Prioritäten für die Zukunft setzen.“

Prof. Dr. Michael Weyrich, Universität Stuttgart

Auch wenn Open-Source-Software bei der Entwicklung von Datensätzen und Algorithmen des automatisierten Fahrens eine vielversprechende Option ist, kann sie auf keinem Fall eine Allzwecklösung werden. Sie muss mit anderen Kooperationsmodellen für Standardisierungsarbeiten harmonisiert werden. Auch andere verwandte Modelle, wie Inner Source⁸ sollen als Teil der Gesamtlösung fungieren. Am Ende müssen für einen erfolgreichen Einsatz die genauen Ziele hinter der Anwendung der OSS-Komponenten definiert und gleichzeitig die Vorteile und Risiken abgewogen werden.

Für die Bewältigung der in diesem Papier diskutierten Herausforderungen und somit zum Erhalt der führenden Position des Wirtschaftsstandortes Deutschland im Mobilitätssektor beizutragen, sollen die beschriebenen Maßnahmen:

1. Förderung von Forschungs- und Standardisierungsarbeiten für die in Abschnitt 4.1 identifizierten technischen Fragestellungen im Bereich der Automotive-IT
2. Konsolidierung und Harmonisierung der bestehenden Standardisierungs- und FOSS-Gremienarbeiten (national, europa- und weltweit)
3. Unterstützung der Kooperation zwischen Industrie- und Hochschuleinrichtungen im Bereich der Automotive-IT und Förderung von Transferaktivitäten

gemeinsam von Politik, Industrie und Wissenschaft in Form von strategischen Plänen (kurz- und langfristig) umgesetzt werden. Dies soll dabei helfen, die kontinuierlich evolvierenden Normen und Regulierungen (z.B. UNECE R155 und R156) einzuhalten und gleichzeitig eine agile Entwicklung innovativer Services zu ermöglichen.

Danksagung

Der Autor bedankt sich bei den Partnern des InnovationsCampus Mobilität der Zukunft (ICM), dem Karlsruher Institut für Technologie (KIT) und der Universität Stuttgart für die Organisation des SAS-Workshops, in dem die Inhalte dieses Papiers entstanden sind.

Zudem möchte sich der Autor bei den Vortragenden des Workshops Herrn Thorge Erichsen, Herrn Dr. Christian Müller, Herrn Prof. Dr. Eric Sax, Herrn Dr. Marc Weber, Herrn Prof. Dr. Michael Weyrich, den Session-Leitern Herrn Tobias Pett, Herrn Schindewolf, Herrn Dr. Matthias Vollat und Herrn Dr. Alexander Weigl sowie Herrn Daniel Grimm für sein inhaltliches Feedback zu diesem Papier und Frau Teresa Mittner für die Erstellung des Titelblatts bedanken.

⁸ Anwendung von Open-Source-Praktiken innerhalb eines Unternehmens

Literaturverzeichnis

- [1] M. Staron, *Automotive Software Architectures: An Introduction*, Cham: Springer International Publishing, 2021.
- [2] Expertenkreis Transformation der Automobilwirtschaft (ETA), Vorsitzende: Prof. Dr.-Ing. Ina Schaefer und Prof. Dr. Dr. h.c. Monika Schnitzer, „Vom Hardware-Produkt zum Software-Defined Vehicle, Durch Open-Source-Softwareentwicklung den Automobilstandort Europa stärken,“ <https://expertenkreis-automobilwirtschaft.de/>, 2023.
- [3] S. Brunner, J. Roder, M. Kucera und T. Waas, „Automotive E/E-architecture enhancements by usage of ethernet TSN,“ in *2017 13th Workshop on Intelligent Solutions in Embedded Systems (WISES)*, Hamburg, Germany, 2017.
- [4] H. Möbus und M. Möbus, „Norm oder Standard – was ist der Unterschied?,“ *Elektro Praktiker*, März 2008.
- [5] F. Gleich, „TISAX-Teilnehmerhandbuch, Wie Sie die TISAX-Prüfung bestehen und ihr Prüfergebnis mit Ihrem Partner teilen,“ ENX Association, 2023.
- [6] O. Hanke, „Cyber Resilience Act: Neue Sicherheitsstandards für digitale Produkte,“ 18 12 2023. [Online]. Available: <https://www.silicon.de/41711141/cyber-resilience-act-neue-sicherheitsstandards-fuer-digitale-produkte>. [Zugriff am 15 Januar 2024].
- [7] AUTOSAR-Konsortium, „AUTOSAR - AUTOMOTIVE OPEN SYSTEM ARCHITECTURE,“ 2023. [Online]. Available: <https://www.autosar.org/>. [Zugriff am 15 Januar 2024].
- [8] H. Guissouma, C. P. Hohl, F. Lesniak, M. Schindewolf, J. Becker und E. Sax, „Lifecycle Management of Automotive Safety-Critical Over the Air Updates: A Systems Approach,“ *IEEE Access*, 2022.
- [9] L. Braun, „Modellbasierte Design-Space-Exploration nicht-funktionaler Auslegungskriterien des Fahrzeugenergiebordnetzes,“ Karlsruher Institut für Technologie, 2018.
- [10] Convide-Konsortium, *SFB 1608 Convide - Konsistenz in der sichtenbasierten Entwicklung Cyber-Physikalischer Systeme*, Karlsruher Institut für Technologie, 2023.
- [11] B. Ashtari Talkhestani, T. Jung, B. Lindemann, N. Sahlab, N. Jazi, W. Schloegl und M. Weyrich, „An architecture of an Intelligent Digital Twin in a Cyber-Physical Production System,“ *at - Automatisierungstechnik*, pp. 762-782, 2019.
- [12] Y. Zhang und Y. M. C. Y. L. G. Z. Ning, „Empirical Study for Open Source Libraries in Automotive Software Systems,“ *IEEE Access*, 2023.

[13] Mercedes-Benz Group AG, „MB.OS ist das "Next Big Thing". Interview mit Dr. Michael Hafner,“ [Online]. Available: <https://group.mercedes-benz.com/karriere/ueberuns/mercedes-benz-operating-system/michael-hafner.html>. [Zugriff am 29 Januar 2024].

Anhang - Über den InnovationsCampus Mobilität der Zukunft

Die Mobilität und die Produktion der Zukunft sind nachhaltig, effizient und kommen aus Baden-Württemberg. Voraussetzung hierfür sind neue bahnbrechende Technologien – von innovativen Fahrzeugantrieben bis zu wandlungsfähigen Produktionsverfahren. Das Ziel des InnovationsCampus Mobilität der Zukunft (ICM)⁹, der vom Land Baden-Württemberg gefördert ist, ist es, diesen Wandel zu gestalten.

Im ICM bündeln die Universität Stuttgart und das Karlsruher Institut für Technologie (KIT) ihre Kompetenzen in Forschung und Innovation, um gemeinsam schnell und flexibel neue Technologien zu entwickeln, neue Ansätze zu erproben und die Basis für Sprunginnovationen zu schaffen. Der ICM ist mit über 300 Forschenden in über 160 Forschungsprojekten und 50 Forschungsinstituten eine der größten Initiativen zur Mobilität und Produktion der Zukunft in Deutschland.

Eins der drei Forschungsfelder des ICM beschäftigt sich mit der Software und den Architekturen moderner und zukünftiger Mobilitätssysteme sowie den für ihre Herstellung notwendigen Produktionssystemen. Das Konzept der softwaredefinierten Fahrzeuge (SDV) ist in diesem Zusammenhang von entscheidender Bedeutung.

⁹ <https://www.icm-bw.de/>