# 1 Einleitung

Ersetzen von Testfahrten durch Simulationen ermöglicht der Automobilindustrie kürzere Produktzyklen zu erreichen. Während in den 1980er Jahren die durchschnittliche Generationsfolge noch etwa 50 Monate betrug, so sind heute Zyklen von 36 Monaten die Regel. Vor allem im konstruktiven Bereich leisten Simulation heute einen entscheidenden Beitrag zur Verkürzung der Entwicklungszyklen, da sich durch Simulationen die Abhängigkeit gegenüber realen Systemen reduzieren lässt.

Neben dem Einsatz in der konstruktiven Fahrzeugentwicklung gewinnen diese Verfahren auch für softwarebasierte Fahrzeugapplikationen zusehends an Bedeutung. Fahrzeugapplikationen teilen sich ein in Fahrerassistenzsysteme und in autonome Systeme. Fahrerassistenzsysteme analysieren die aktuelle Fahrsituation und warnen den Fahrer vor kritischen Situationen oder greifen aktiv in die Fahrzeugführung ein (vgl. Wallentowitz et al. [2009]; Wiener [1994]). Autonome Systeme übernehmen Aufgaben der Fahrzeugführung ohne Zutun des Fahrers. In den zukünftigen Fahrzeuggenerationen ist mit einer Zunahme dieser Funktionen zu rechnen, was sich auf Komfort-, Sicherheits- und Effizienzanforderungen von Markt und Politik zurückführen lässt.

## 1.1 Vernetzte Fahrzeugapplikationen

Dieser Trend spiegelt sich in steigenden Kosten für die Entwicklung von Elektronik- und Softwarekomponenten wieder. Bezogen auf die gesamten Entwicklungsaufwände stieg nach Broy [2006]; Abloeser [2010] der Anteil an Elektronik- und Softwareaufwänden von weniger als 16 % im Jahr 1990, auf derzeit etwa 40 %. Entsprechend einer Studie von Bechmann et al. [2011]

ist mit einer weiteren Zunahme zu rechnen, da sich der Fokus zukünftiger Systeme auf die Vernetzung mit der Umgebung verlagert. Aufgrund der nicht deterministischen Umgebung bringt die Vernetzung eine Komplexitätssteigerung bei der Entwicklung der Fahrzeugapplikationen mit sich. Grundlage dafür bilden Technologien im Bereich der maschinellen Umfeldwahrnehmung, sowie die Möglichkeiten der mobilen Datenkommunikation. Diese Technologien stellen die Grundlage für Applikationen im Bereich hoch-automatisiertes Fahren sowie für das autonome Fahren[1]. Die dadurch zur Verfügung stehenden Informationen unterscheiden moderne Systeme grundlegend von den ersten elektronischen Fahrzeugapplikationen. Diese verarbeiteten ausschließlich Messgrößen von Fahrzeugsensoren, wie beispielsweise das 1978 erstmalig in einem Serienfahrzeug eingesetzte Anti Blockiersystem (ABS). Aktuelle Fahrzeugapplikationen erfassen das Fahrzeugumfeld und sind mit der Umgebung vernetzt. Dies führt zu einer gegenseitigen Beeinflussung von Systemen über die Fahrzeugsystemgrenze hinaus. Die Informationen für diese Systeme stammen von autozentrierten Sensoren (On-Board), welche das Umfeld im Sichtbereich der Sensoren maschinell erfassen, sowie von der instrumentierten Umgebung des Fahrzeugs. Die instrumentierte Umgebung erfasst Zustände mittels Sensoren, die sich außerhalb des eigenen Fahrzeugs befinden (Off-Board). Die Erfassung des Umfelds, sowie die Kommunikation mit der Umgebung führen in beiden Fällen zu einer Abhängigkeit gegenüber nicht deterministischen Informationen. Neben den On-Board Sensoren des Fahrzeugs beeinflussen nun auch der Fahrer, sowie Off-Board Sensoren der Umgebung das Applikationsverhalten.

Wie in Abbildung 1.1 skizziert, führen die vernetzten Applikationen zu Abhängigkeiten zwischen den Komponenten Fahrer, Fahrzeug und Umfeld. Diese Komponenten müssen während der Entwicklung zur Verfügung stehen.

1 Hoch-automatisiertes Fahren bezeichnet einen Zwischenschritt zwischen dem heutigen assistierten Fahren, bei dem der Fahrer durch zahlreiche Assistenzsysteme bei der Fahraufgabe unterstützt wird, und dem autonomen Fahren, wo das Fahrzeug selbsttätig und ohne Einwirkung des Fahrers fährt.

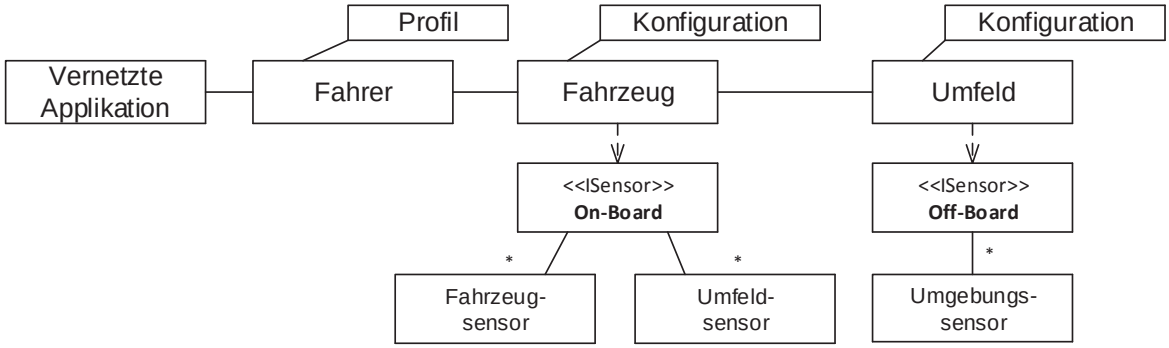



**Abb. 1.1:** Abhängigkeiten zwischen vernetzten Applikationen

## 1.2 Problemstellung

Aufgrund der Abhängigkeiten die vernetzte Applikationen mit der Umgebung, dem Fahrer und dem Fahrzeug aufweisen, erfolgt die Entwicklung dieser Applikationen in weiten Teilen unter Verwendung realer Fahrzeugprototypen. Die Modifizierung der Fahrzeuge, die Ausstattung der Infrastruktur sowie die Koordinierung der Testfälle erfordern hohe finanzielle und zeitliche Aufwände.

Diese Situation zeigt sich anhand aktueller Forschungsprojekte wie SimTD [Stübing et al., 2010] oder Travolution [Pudenz, 2013]. Die Projekte beschäftigen sich mit der Vernetzung von Fahrzeug und Infrastruktur. SimTD[2] untersucht das Potential vernetzter Fahrzeug- und Infrastruktursysteme zur Erhöhung der Verkehrssicherheit. Für SimTD wurden 400 Fahrzeuge modifiziert und mehr als 100 Roadside Units (intelligente Infrastruktur) installiert. Das Forschungsprojekt Travolution untersucht die Optimierung des Verkehrsfluss durch Vernetzung von Fahrzeug und Infrastruktur. Dazu war es erforderlich zu Projektbeginn 50 Ampelanlagen mit einer Sendevorrichtung zur Übertragung der Ampelphasen auszustatten.

Die Beobachtungen aus diesen Fallstudien lassen sich auf die Serienentwicklung von Fahrzeugapplikationen übertragen. Es ist zudem absehbar, dass die Aufwände für eine Serienentwicklung höher ausfallen, um die erforderliche robuste Auslegung zu gewährleisten.

2 SimTD: Sichere intelligente Mobilität Testfeld Deutschland

## 1.3 Beitrag

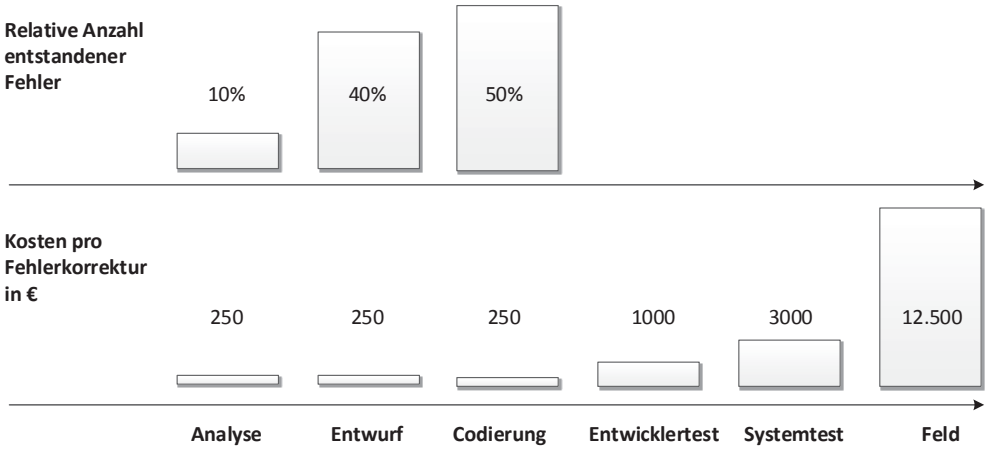



**Abb. 1.2:** Auswirkung der Fehlerkorrekturkosten [Liggesmeyer, 2009]

Zielsetzung der vorliegenden Arbeit ist die Reduzierung von zeitlichen und finanziellen Aufwänden während der Entwicklung von Fahrzeugapplikationen, durch Verlagerung aufwändiger Testszenarien in die virtuelle Welt. Reale Komponenten werden dazu durch Simulationsmodelle ersetzt. Durch Kombination von realen Komponenten und Simulationsmodellen wird unabhängig von der Entwicklungsphase ein Gesamtsystem konfiguriert, das im Verhalten dem realen System entspricht. Das Gesamtsystem berücksichtigt die erforderlichen Applikationen, das Fahrzeug, den Fahrer und die Umgebung des Fahrzeugs. Umfeldbasierte Applikationen lassen sich damit bereits während des Entwurfs unabhängig von den umgebenden Systemen testen. Dies führt entsprechend Liggesmeyer [2009] zu einer Kosten-/Nutzenoptimierung (Abb. 1.2). Besonders wenn das Originalsystem nicht deterministische Informationen verarbeitet oder komplexe Szenarien erfordert ist dies von Bedeutung. Diese Situationen finden sich beispielsweise bei Car-2-Car Funktionen aufgrund der Beteiligung zahlreicher Fahrzeuge vor. Für sicherheitskritische Systeme bei denen Mensch oder Fahrzeug gefährlichen Zustände ausgesetzt sind, lassen sich durch die Verlagerung von Aufwänden in die Simulation Risiken reduzieren.

Im Rahmen dieser Dissertation wurde dazu ein Framework entwickelt, das die umgebenden Systeme einer Applikation hardwareunabhängig zur Verfügung stellt. Dieses Framework wird im weiteren Verlauf als VU-Framework bezeichnet. Das VU-Framework bildet die Umgebung der zu entwickelnden Applikation aus einer Kombination von realen und virtuellen Komponenten ab, wodurch ein hybrides Gesamtsystem entsteht. Das VU-Framework abstrahiert die zur Entwicklung von elektronischen Fahrzeugapplikationen relevanten Komponenten Fahrzeug, Fahrer und Umfeld. Dadurch lässt sich nicht vorhandene reale Hardware durch eine entsprechende virtuelle Komponente ersetzen. Um dies zu gewährleisten sind den realen Komponenten im VU-Framework virtuelle Komponenten in Form von Simulationsmodellen oder Softwarekomponenten gegenübergestellt. Die Abbildung 1.3 zeigt das Prinzip auf oberster Ebene.

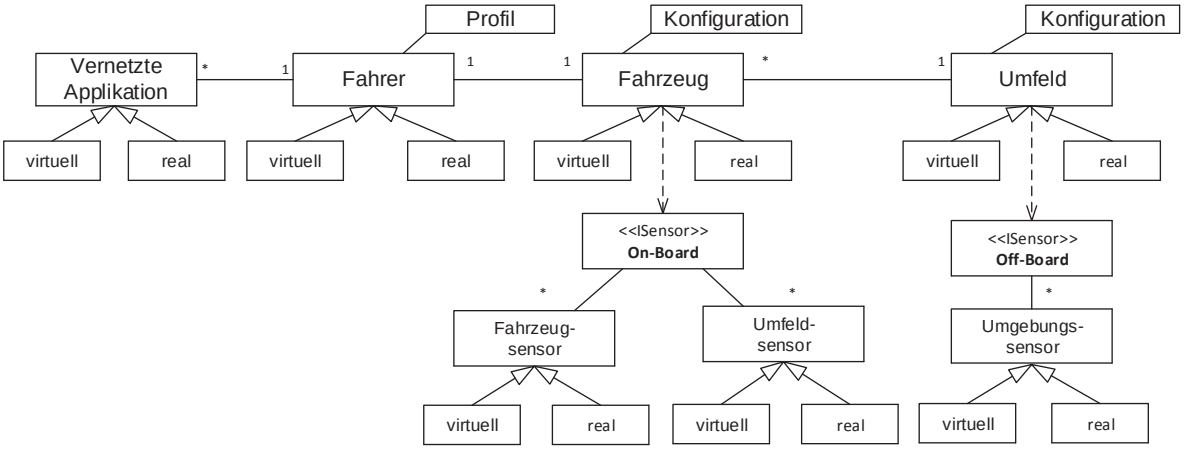



**Abb. 1.3:** Framework kombiniert reale und virtuelle Komponenten zu einem Gesamtsystem

Durch das VU-Framework ist der Entwickler in der Lage, das benötigte Gesamtsystem zur Applikationsentwicklung aus realen, virtuellen oder einem Mix von realen und virtuellen Komponenten zu konfigurieren. Die Granularität soll dabei nur durch die virtuellen Komponenten beschränkt sein. Aufgrund der Abstraktion benötigt er dazu kein Detailwissen über die einzelnen Komponenten. Die Aufgabe des VU-Frameworks ist die Bereitstellung von Schnittstellen

und Mechanismen, um aus existierenden Simulationsmodellen und realen Komponenten ein Gesamtsystemverhalten zu konfigurieren. Die Abstraktion der Komponenten Fahrer, Fahrzeug und Umfeld durch das VU-Framework, sowie die Möglichkeit aus virtuellen und realen Komponenten ein Gesamtsystem zu konfigurieren ermöglichen:

1. Darstellung einer Probefahrt unabhängig von der Verfügbarkeit von Hardwarekomponenten.

2. Die Wiederverwendung von Simulationsmodellen in verschiedenen Projekten und in verschiedenen Entwicklungsphasen.

3. Das Testen und Entwickeln von Systemen, die Informationen von der Fahrzeugumgebung benötigen.

Aufgebaut ist die Arbeit wie folgt. Unter Kapitel 2 werden erforderliche Grundlagen zu Systemen im Automobil vermittelt. Im Kapitel 3 sind die erforderlichen Bausteine, sowie die Motivation für eine Verlagerung von realen Probefahrten in die Simulation aufgeführt. Das Kapitel 4 fasst die Anforderungen zusammen, die zur Realisierung des VU-Framework erforderlich sind. Aus den Anforderungen leitet sich das VU-Framework Konzept ab. Die Überführung der Anforderungen in eine Referenzimplementierung erfolgt in Kapitel 5. Dabei werden den realen Komponenten Fahrer-, Fahrzeug und Umfeld virtuelle Komponenten auf Basis existierender Modelle gegenübergestellt und prototypisch in das VU-Framework integriert. Die Evaluierung der Referenzimplementierung erfolgt in Kapitel 6 anhand konkreter Beispiele. Dazu werden verschiedenen Fallstudien durchgeführt, die die Verwendbarkeit des VU-Framework in den unterschiedlichen Entwicklungsphasen untersuchen.

Aus realen und virtuellen Komponenten wird für die Durchführung jeder Fallstudien das jeweils erforderliche Gesamtsystem konfiguriert und mittels definierten Testszenarien evaluiert. Die Konzeptionierung und Evaluierung des VU-Framework basiert auf den Prinzipien der formativen Evaluation. Die Fallstudien umfassen die virtuelle Auslegung eines elektrischen Energiespeichers für Elektrofahrzeuge (Abschnitt 6.1.1), den Vergleich von realen und virtuellen Fahrzeugen (Abschnitt 6.1.2), einer rein virtuellen Konzeptentwicklung (Abschnitt 6.1.3), sowie der Verlagerung von Testfahrten in die Simulation

im Rahmen der Entwicklung eines Fahrerassistenzsystems (Abschnitt 6.2.1). Zusätzlich zu den Fallstudien wird das Framework im produktiven Einsatz zum Testen einer umfeldbasierten adaptiven Lichtsteuerung eingesetzt (Abschnitt 6.2.2). In einem Fahrsimulator ermöglicht das VU-Framework die Anbindung realer Fahrzeugapplikationen zur Demonstration für Kunden (Abschnitt 6.3).

# 2 Terminologie

Dieses Kapitel befasst sich mit den Grundlagen und Begrifflichkeiten im Zusammenhang von Fahrzeugapplikationen. Im Abschnitt 2.1 wird ein Überblick über die im Automobil eingesetzten Systeme vermittelt. Die zur Entwicklung der Fahrzeugapplikationen verwendeten Prozesse und Simulationsmethoden sind in Abschnitt 2.2 aufgeführt. Ein Ausblick auf zukünftige Systeme und die daraus resultierenden Herausforderung beschreibt Abschnitt 2.3.

## 2.1 System im Automobilkontext

Ein System bezeichnet eine Einheit wechselseitig interagierender Teile, das durch die Abgrenzung zu seiner Umgebung gekennzeichnet ist. Das Deutsche Institut für Normung (DIN) definiert Systeme anhand der Norm 19226:

**Definition: 2.1**

*Ein System grenzt sich von seiner Umgebung ab und erzeugt aus den Eingangsparametern ein definiertes Ausgangsverhalten.*

Dabei lassen sich deterministische Systeme, bei denen die Vergangenheit eindeutig die Zukunft festlegt, und nicht deterministische Systeme, bei denen die Zukunft zufällig ist unterscheiden. Diese allgemeine Definition wird für die vorliegende Arbeit auf technische Vorgänge beschränkt, was nach Simon

[1996] im Kontext der Systemtheorie zu einer Einschränkung auf künstliche Systeme führt. Im weiteren Verlauf ist der Begriff System daher gleichbedeutend mit dem Terminus künstliches System.

### 2.1.1 Wirklichkeit und Modell

Beschreibt ein System einen räumlich abgegrenzten Teil der Wirklichkeit, spricht man von einem konkreten System. Wird das reale Verhalten durch Algorithmen und Gleichungen beschrieben, bezeichnet man es als mathematisches bzw. nach Goos u. Zimmermann [2006] als Informatik System. Dies lässt sich exemplarisch anhand eines Fahrzeugs verdeutlichen:

**Beispiel:** Ein Fahrzeug ist real existent und daher ein konkretes System. Wird das Verhalten des Fahrzeugs durch Algorithmen und Gleichungen beschrieben, spricht man entsprechend Vogel et al. [2009] von einem mathematischen System.

Mathematische Systeme bilden die Realität ab und helfen die Zusammenhänge und Funktionsweise konkreter Systeme zu erklären. Gegenstände bzw. Systeme lassen sich somit auf verschiedenen Ebenen darstellen. In der Wirklichkeit als konkretes System bzw. als Abbild der Wirklichkeit durch ein Modell. Abbildung 2.1 skizziert die Zusammenhänge.

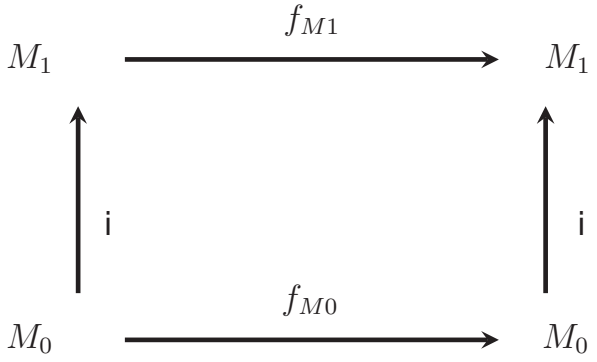



**Abb. 2.1:** Beziehung zwischen Wirklichkeit und Modell nach Goos u. Zimmermann [2006]

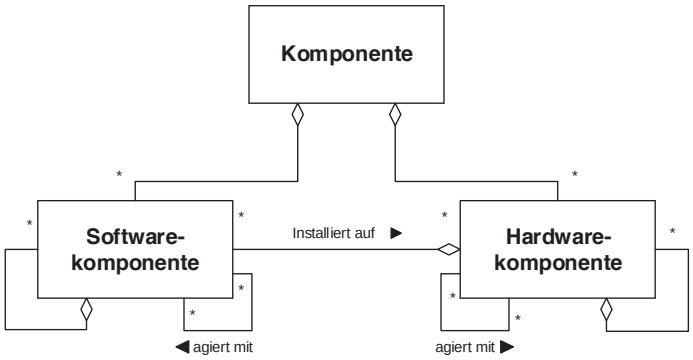
Ist $i$ die Abbildung, die einem konkreten System $M_0$ ein anwendungsspezifisches Modell $M_1$ zuordnet und $f_{M1}$ eine beliebige Beziehung des Modells, so existiert auch eine korrespondierende reale Beziehung $f_{M0}$. Modelle $M_1$ bilden im Verlauf dieser Arbeit die Grundlage zur Simulation konkreter Systeme $M_0$. Real existente Zusammenhänge lassen sich dadurch in die Modellebene überführen und unabhängig von realen Komponenten evaluieren. Als Komponente wird in diesem Zusammenhang eine eigenständige Applikationseinheit bezeichnet, die einen klar definierten und abgegrenzten Funktionsinhalt repräsentiert.

Die Benennung der Ebenen stützt sich auf das Meta Object Facility (MOF) nach OMG [2006] und beschreibt eine spezielle Metadaten-Architektur, die die Voraussetzung für eine modellbasierte Softwareentwicklung bildet. Diese Vorgehensweise findet sich u.a. bei der Realisierung von Fahrzeugapplikationen wieder. Zusätzlich zu der konkreten Ebene $M_0$ und der Modellschicht $M_1$ definiert das Meta Object Facility die Meta-Modell Ebene $M_2$ sowie die Meta-Meta-Modell Ebene $M_3$. Dabei definiert die Meta-Modell Ebene $M_2$ nach Rechenberg [1997] die Konstruktionsregeln für die Erstellung des Modellsystems. Die Meta-Meta-Modell Ebene $M_3$ kennzeichnet eine abstrakte Ebene zur Definition der Meta-Modell Ebene. Diese Abstraktion stellt Möglichkeiten zur Verfügung, zwischen unterschiedlichen Metamodellen Beziehungen aufzubauen, um eine gemeinsame Basis für die Metamodellgenerierung zu schaffen.

### 2.1.2 Eingebettete Systeme

Im Automobilbereich besteht heute ein konkretes System der $M_0$ Ebene in der Regel aus einem Verbund von Steuergerät (Electronic Control Unit (ECU)), Software, sowie Sensoren und Aktuatoren [Marwedel, 2007]. Mit Sensoren wird die aktuelle Situation erfasst, aus der Algorithmen Aktionen ableiten. Die Aktionen werden mittels Aktuatoren in physikalische Größen gewandelt. Ist ein derartiges System in ein größeres System (z.B. Fahrzeug) eingebettet, wird dieses als eingebettetes System bezeichnet (vgl. Zöbel u. Albrecht [1995]; Wörn u. Brinkschulte [2005]). Abbildung 2.2 a) zeigt ein exemplarisches Steuergeräte eines eingebetteten Systems im Automobilbereich. Den Zusammenhang von Software- und Hardwarekomponenten eines eingebetteten Systems skizziert

**(a)** Beispiel eines Steuergeräts (ECU)




**(b)** Komponenten eines eingebetteten Systems

**Abb. 2.2:** Aufbau eines konkreten Systems ($M_0$) aus Hardware und Softwarekomponenten

die Abbildung 2.2 b). Der Übergang vom anwendungsspezifischen Modell $M_1$ in ein konkretes System der Ebene $M_0$ ist in dieser Arbeit durch die Überführung des Modells bzw. der Softwarekomponente in die Zielhardware definiert.

Eingebettete Systeme stellen eine Funktionalität für den Nutzer bereit und werden als Applikation bezeichnet (vgl. Bauer et al. [2002]). Der Wirkbereich von eingebetteten Systemen ist auf die Systemgrenze beschränkt. Im Rahmen dieser Arbeit ist die Systemgrenze das Fahrzeug. Durch die Etablierung mobiler Datenkommunikation sowie der maschinellen Umfelderfassung im Fahrzeug, erweitert sich der Wirkbereich von Applikationen über die Systemgrenze hinaus.

### 2.1.3 Fahrzeugvernetzung

Nach Broy [2006] verfügt ein modernes Oberklassefahrzeug über mindestens 70 Steuergeräte, die mit mehr als fünf verschiedenen Bussystemen vernetzt sind. Mit Ausnahme der Schnittstellen sind die Algorithmen der Steuergeräte unabhängig vom restlichen System, wodurch sich eine komponentenorientierte Architektur ergibt. Die Komponentenorientierung entspricht dem Prinzip der losen Kopplung (vgl. Sametinger [1997]), was zu einer Verringerung der Ab-

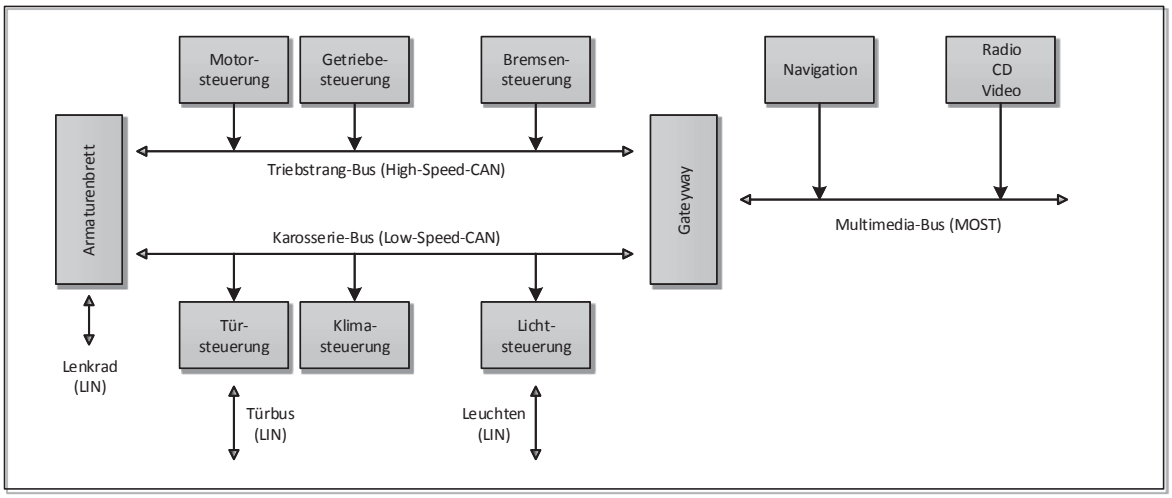



**Abb. 2.3:** Bus-Kommunikation im Fahrzeug [Zimmermann u. Schmidgall, 2011]

hängigkeiten innerhalb des Systems führt, die Toleranz gegenüber dem Ausfall eines Steuergeräts erhöht, die Wartbarkeit erhöht und die Wiederverwendung einzelner Komponenten ermöglicht. Jede Komponente lässt sich ersetzen oder aktualisieren, ohne die restlichen Steuergeräte anzupassen.

Die Bussysteme im Fahrzeug sind hinsichtlich Kosten und Performance optimiert, wodurch in einem Fahrzeug verschiedene Kommunikationsmethoden zum Einsatz kommen. Durch Vernetzung werden Funktionen auf verschiedene Rechner verteilt und die Qualität der Daten erhöht, indem eine Plausibilisierung zwischen verschiedenen Sensoren erfolgt (vgl. Marscholik u. Subke [2007]; Darms [2009]). Sensorinformationen unterschiedlicher Systeme werden fusioniert, um die Güte der Messergebnisse zu verbessern (vgl. Haselhoff et al. [2007]; Bertozzi et al. [2008]). Die Kommunikation über Bussysteme führt zu definierten Schnittstellen zwischen den verschiedenen Steuergeräten. Dadurch lässt sich die Entwicklung der Applikationen verteilen und die Steuergeräte in verschiedenen Fahrzeugtypen wiederverwenden.

In der Abbildung 2.3 ist eine exemplarische Buskonfiguration skizziert, wie sie derzeit in Mittelklassenfahrzeugen anzutreffen ist. Die Teilnehmer sind entsprechend der jeweiligen Kommunikationsmethode entweder in einem Linien-, Stern, oder Ringnetz angeordnet (Abb. 2.4).

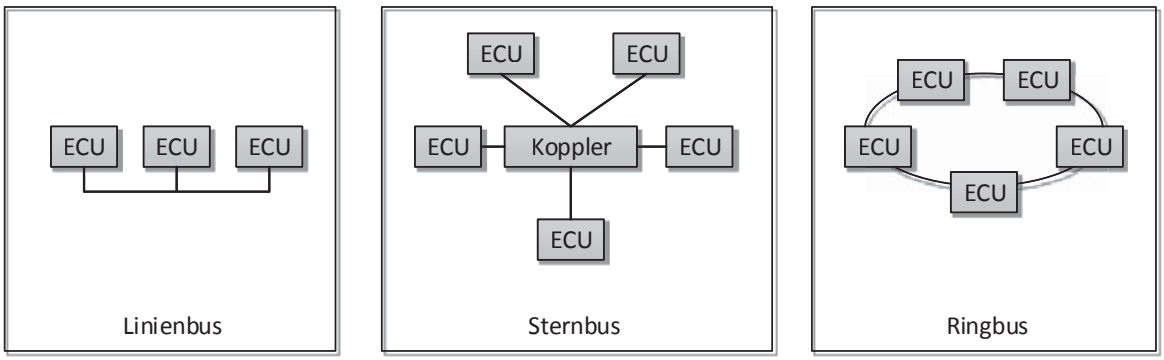



**Abb. 2.4:** Eingesetzte Bus-Typologien im Fahrzeug [Zimmermann u. Schmidgall, 2011]

### Controller Area Network (CAN)

Das Controller Area Network (CAN) ist ein asynchrones, serielles Bussystem, bei dem die Teilnehmer linienförmig angeordnet sind. Die Teilnehmerpriorität ist durch einen Identifier festgelegt. Mittels bitweiser Arbitrierung wird verlustfrei das Senderecht vergeben. Beim CAN-Bus unterscheidet man zwischen dem Highspeed-Bus, bei dem die maximale Datenübertragungsrate 1 Mbit/s beträgt und dem Lowspeed-Bus mit 125 kBit/s.

### Local Interconnect Network (LIN)

Das Local Interconnect Network (LIN) ist ein serieller Master-Slave Bus zur Kommunikation zwischen intelligenten Sensoren und Aktoren im Fahrzeug. In der Regel wird der LIN-Bus als kostengünstiger Sub-Bus eines CAN-Bus eingesetzt. Der LIN-Bus ist als Eindrahtbus für geringe Datenraten (max. 20 kBit/s) ausgelegt (vgl. ISO 17987). Typische Anwendungsbereiche des LIN-Buses sind Tür- bzw. Sitzsteuerung.

### Flexray

Die Flexray Kommunikation basiert auf einer seriellen, deterministischen Kommunikation und wurde zur Bewältigung der steigenden Anforderungen hinsichtlich Datenrate, Ausfallsicherheit und Latenzzeiten konzipiert. Das Bussystem ist sternförmig aufgebaut und erreicht eine Bruttodatenrate von bis zu 10 MBit/s. Neben den Datenraten stellen die garantierten Latenzzeiten des Flexray Protokolls einen wesentlichen Vorteil dar. Vor allem im Hinblick auf X-by-wire Systeme[1] ist Flexray von Bedeutung.

### Media Oriented Systems Transport (MOST)

Der Media Oriented Systems Transport (MOST) Bus ist ein spezialisierter Ringbus, der zur Übertragung von Audio-, Video-, Sprach- und Datensignalen im Fahrzeug eingesetzt wird. Er stellt einen synchronen sowie einen asynchronen Datenkanal zur Verfügung.

### Zentrale Steuergeräte

Bei zukünftigen Fahrzeuggenerationen wird ein Rückgang der Steuergeräteanzahl zu beobachten sein, indem man verschiedene Applikationen auf ein Steuergerät zusammenfasst. Ein Beispiel dafür ist das zentrale Fahrerassistenzsteuergerät (zFAS) von Audi [Simonite, 2014]. Grundlage dafür bildet das AUTomotive Open System ARchitecture (AUTOSAR) Projekt, das einen einheitlichen Standard für die Entwicklung von elektronischen Fahrzeugapplikationen verfolgt. AUTOSAR definiert eine universelle Plattform Architektur, die es ermöglicht Softwarekomponenten hardwareunabhängig zu entwickeln und wieder zu verwenden.

1 X-by-wire Systeme ersetzen mechanischen Kraftfluss. Beispielsweise wird bei Steer-by-wire die Informationen des Lenkradwinkels über einen Datenbus zu einem Aktuator übermittelt, wodurch sich die Lenkstange ersetzten lässt.