Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Masterarbeit

Jan Depke

Inkrementelle Konsolidierung automobiler Bussysteme auf Basis eines Realtime Ethernet Backbones

Fakultät Technik und Informatik
Studiendepartment Informatik

Faculty of Engineering and Computer Science
Department of Computer Science

Jan Depke

Inkrementelle Konsolidierung automobiler Bussysteme auf Basis eines Realtime Ethernet Backbones

Masterarbeit eingereicht im Rahmen der Masterprüfung

im Studiengang Master of Science Informatik
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer: Prof. Dr. Franz Korf
Zweitgutachter: Prof. Dr. Wolfgang Fohl

Eingereicht am: 24. März 2015

Jan Depke

Thema der Arbeit

Inkrementelle Konsolidierung automobiler Bussysteme auf Basis eines Realtime Ethernet Backbones

Stichworte

Echtzeit-Ethernet, Bussysteme, Kommunikationsgateway, Migration, Schnittstellenbschreibungssprache, verteilter CAN Bus, automotive Anwendungen, CAN, FlexRay, MOST

Kurzzusammenfassung

Echtzeit-Ethernet ist vielversprechender Kandidat zur Verbesserung der Kommunikation in zukünftigen Automobilen. Es bietet hohe Bandbreite, geringen Jitter, und die physikalische Schicht kann mit den Anforderungen der Automobilindustrie wachsen. Eine Migrationsstragie unter Weiterverwendung der Errungenschaften der letzten Jahrzehnte im Bereich der Feldbusse ist nötig, um einen sanften Übergang von aktuellen Feldbusarchitekturen zum Echtzeit-Ethernet zu ermöglichen.
Diese Arbeit untersucht die Auswirkungen einer inkrementellen Migration von Controller Area Network (CAN)-Bussen hin zum Echtzeit-Ethernet. Eine Migrationsstrategie wird aus Restriktionen technologischer Standards und Entwurfsmustern entwickelt, welche in Richtlinien für Machbarkeitsuntersuchungen überführt werden. Eine flexible Schnittstellenbeschreibungssprache wird entwickelt, implementiert und zur Übersetzung zwischen etablierten Protokollen und Echtzeit-Ethernet in einem Kommunikationsgateway genutzt. Der Einsatz in einem Prototypenfahrzeug liefert realistische Kenndaten um die Machbarkeit der Migration unter Beibehaltung bestehender CAN-Anwendungen zu zeigen.

Title of the paper

Incremental consolidation of in-vehicular buses based on a realtime ethernet backbone

Keywords

Realtime Ethernet, field bus, communication gateway, migration, interface description language, distributed CAN bus, automotive applications, CAN, FlexRay, MOST

Abstract

Realtime Ethernet is a promising candidate to improve automotive communication networks in upcoming cars. It provides high bandwidth and low jitter and can be adapted using a suitable physical layer to comply with the requirements of the automotive industry. To support a smooth transition from today's heterogeneous fieldbus dominated communication architecture towards a Realtime Ethernet backbone design, a migration strategy is demanded, preserving the development and knowledge about fieldbus technologies acquired over the last decades.
This thesis investigates the impact of the step-wise transition from the controller area network technology towards Realtime Ethernet. A migration strategy is contributed by evaluating compatibility restrictions arising from formal technology standards and development patterns. These restrictions are refined into guidelines allowing to perform feasibility checks for individual use-cases. An interface description language covering relevant aspects of the translation between legacy and Realtime Ethernet protocols is implemented in a drop-in gateway architecture. The results obtained from the prototype implementation using realistic hardware resources show the feasibility of transparently migrating parts of CAN buses.

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Einleitung

In den letzten 20 Jahren hat sich die Nutzung elektronischer Systeme in jeder neuen Automobilgeneration verstärkt. Aktuelle Automobile beherbergen ein verteiltes System aus mehr als 70 Steuergeräten/Electronic Control Unit (ECU), welche untereinander ca. 2500 Nachrichten über ein zentrales Kommunikationsgateway austauschen.

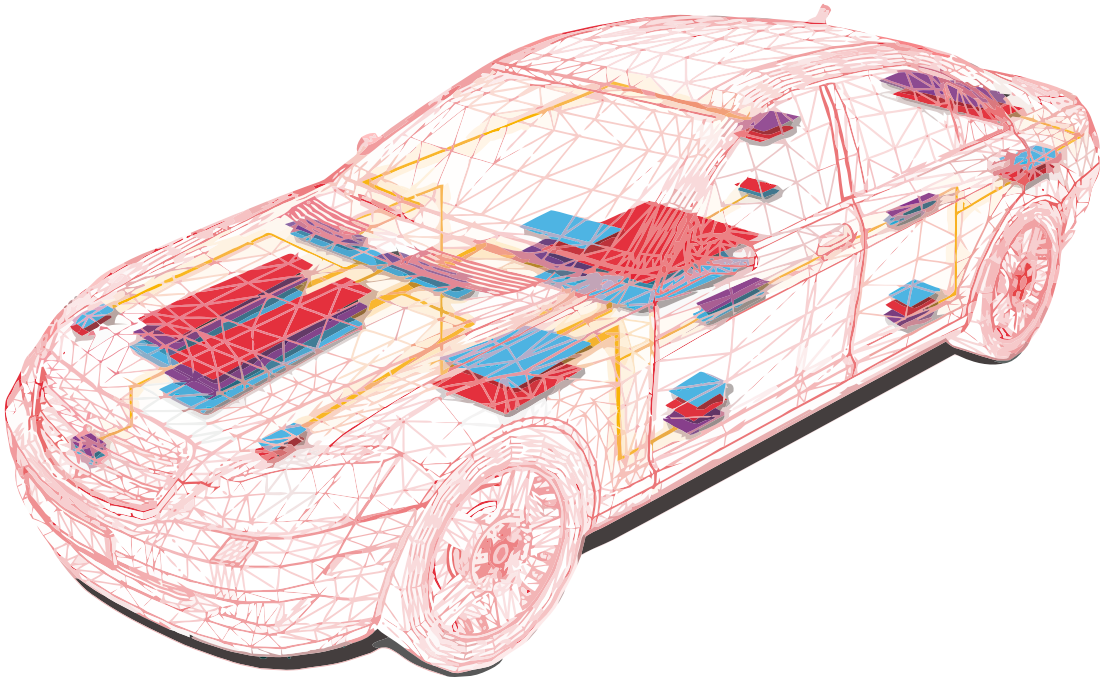

Abbildung 1.1: Symbolbild vernetzter elektronischer Systeme in einem aktuellen Automobil, Quelle: Core Arbeitsgruppe

In den kommenden Jahren werden Anzahl und Leistungsfähigkeit elektronischer Systeme mit verstärkter Verbreitung von drive-by-wire, Vehicle-to-Vehicle (V2V) und Vehicle-to-Roadside (V2R) weiterhin ansteigen. Mit zunehmend komplexeren Aufgabenstellungen automobiler elektronischer Systeme, wie z.B. der automatischen Reaktion auf Verkehrshindernisse, welche auf Basis von Kamera- und Light Detection and Ranging (LIDAR)-Daten erkannt werden, müssen neue Netzwerktechnologien im Automobil zum Einsatz kommen. Diese Netzwerktechnologien

haben den zukünftigen Anforderungen an Übertragungsbandbreite und deterministischer Echtzeitkommunikation zu genügen.
Ethernet wird momentan verstärkt in Automobile der Oberklasse für nicht sicherheitsrelevante, bandbreitenintensive Multimedia- und Komfortanwendungen integriert. Prognosen deuten darauf hin, dass im Jahr 2020 40% der neu produzierten Automobile anteilig Ethernet verwenden werden (ABI Research (2014)). Diese Entwicklung begünstigt einen später folgenden Einsatz von RTE für sicherheitskritische Anwendungen mit hohem Bandbreitenbedarf auf Basis deterministischer Echtzeitkommunikation.
Die Core Arbeitsgruppe der Hochschule für Angewandte Wissenschaften Hamburg (HAW) erforscht switch-gestütztes RTE hinsichtlich der technisch sinnvollen und flexiblen Einsetzbarkeit als Kommunikations-Backbone in Automobilen. Die Verwendung von RTE für ein Backbone-Netzwerk als Vorgänger des perspektivisch langfristigen Ziels einer homogenen Ethernet-Kommunikationsinfrastruktur im Automobil wird durch die Core Arbeitsgruppe anhand von Simulationen und der Entwicklung technischer Prototypen vorangetrieben.
Die Planung und Entwicklung heterogener automobiler Bussysteme wird mit jeder zusätzlichen, neuen Netzwerktechnologie komplexer und kostenintensiver. Einzelne Kostenfaktoren und Bestandteile der Komplexität sind der geographisch dezentrale Entwicklungsprozess durch unterschiedliche Dienstleister, der Organisationsaufwand bezüglich technischer Implementierungsvorgaben, die Entwicklung von auf neuen Netzwerktechnologien basierenden Steuergeräten, sowie die Integration der neuen Netzwerktechnologie in das bestehende Automobilbussystem. Die Migration in Richtung einer homogenen, auf RTE basierenden Netzwerkinfrastruktur ist daher besonders attraktiv, um die sich dezentral bildenden Expertengruppen für einzelne, teilweise exotische Technologien wieder zusammenzuführen. Allerdings kann eine soche Veränderung nicht in einem einzelnen Entwicklungsschritt vorgenommen werden. Aus diesen Gründen ist es notwendig, eine flexible Migrationsstrategie zu entwickeln, welche den organisatorischen Aufwand zur Synchronisation zwischen unterschiedlichen Dienstleistern mit unterschiedlichen Zuständigkeitsbereichen minimiert, die Integration von neuen Steuergeräten und Netzwerktechnologien durch weitestgehend unabhängige Entwicklerteams erlaubt und gleichzeitig in der Übergangsphase hin zu homogenen Netzinfrastrukturen im Automobil die Weiterverwendung von bestehendem Wissen, vorhandenen Arbeitskräften, Technik und Software maximal begünstigt.

1.1 Problemstellung und Zielsetzung

Das Ziel dieser Arbeit ist die Unterstützung der Migration von Automobilbussystemen hin zu einem RTE-basierten Backbone-Netzwerk durch geeignete Strategien zur Vereinfachung, Kostenreduzierung und Flexibilitätssteigerung der Migration. Hauptaugenmerk liegt auf der Anbindung bestehender CAN-Bussysteme an den RTE Backbone. Insbesondere soll hierbei sichergestellt werden, dass die bestehenden Bussysteme schrittweise, gegebenenfalls nur anteilig, unter transparenter Weiterverwendung der CAN-Anwendungen migriert werden können. Darüber hinaus wird überprüft, ob die vorgestellte Migrationsstrategie auf weitere etablierte Automobilbussysteme wie FlexRay und MOST übertragbar ist. Im Rahmen der Entwicklung dieser Migrationsstrategie müssen unterschiedliche Themenkomplexe diskutiert und erarbeitet werden:

- Standardkonforme, transparente Aufteilung bestehender CAN-Bussysteme
- Sicherstellung der Weiterverwendbarkeit von CAN-Anwendungen
- Optionen zur Steigerung der Netzwerkeffizienz
- Varianten der Anbindung bestehender Bussysteme an einen RTE Backbone
- Flexible Konfigurationsmöglichkeiten für unterschiedliche Anwendungsfälle
- Evaluierung der Qualität des vorgestellten Ansatzes
- Übertragbarkeit auf weitere Bussysteme

1.2 Struktur dieser Arbeit

Diese Arbeit ist so strukturiert, dass zuerst in Kapitel 2 die Grundlagen automobiler Bussysteme und RTE vorgestellt werden, gefolgt von der Präsentation der einzelnen Aspekte der Problemstellung und der Vorstellung verwandter Arbeiten in Kapitel 3. Kapitel 3 beschreibt dabei detailliert die etablierten Vorgehensweisen des Entwurfs und der Integration automobiler Systeme sowie die aus der Verbindung von Netzwerken entstehenden Probleme und Notwendigkeiten. In Kapitel 4 werden die mit der Erfüllung der Zielsetzung und Behandlung der Problemstellung verbundenen Anforderungen entwickelt und dokumentiert. Deren Erfüllung wird durch die unterschiedlichen Teile des in Kapitel 5 präsentierten Konzeptes sichergestellt. Kapitel 6 wertet die Implementierung in Hinblick auf Qualitätssicherung, Praxistauglichkeit, Zeitverhalten und in Hinblick auf das Potential für zukünftige Anwendungsbereiche aus. Abgeschlossen wird diese Arbeit durch den Versuch der Verallgemeinerung des Lösungsansatzes durch Übertragung auf nicht explizit erfasste Anwendungsfälle in Kapitel 7 und ein Fazit sowie einen Ausblick auf zukünftige Weiterentwicklung in Kapitel 8.

2 Grundlagen

2.1 Etablierte automobile Bussysteme

Die in der Automobilindustrie üblicherweise verwendeten Bussysteme erfüllen jeweils unterschiedliche Aufgaben und sind unterschiedlich stark verbreitet. Primär unterscheiden sich die Bussysteme in

- Bandbreite, Protokoll, unterstützten Topologien,
- Verfügbarkeit für Automobilanwendungen zertifizierter Steuergeräte,
- Sendeparadigma (ereignisgesteuert, zeitgesteuert, Hybrid)
- Kostenfaktor der Anschaffung, Implementierung und Produktpflege.

Prinzipiell muss die Entscheidung, welche Bussysteme an einen RTE Backbone angebunden werden sollen, zumindest von den Faktoren „Zukunftsrelevanz“ und „Mehrwert“ getrieben sein. Die Zukunftsrelevanz eines Bussystems lässt sich anhand der Entwicklung der Steuergeräteverkaufszahlen approximieren. Unter dem Mehrwert ist zu verstehen, ob ein Bussystem technisch sinnvoll und kostenneutral durch native RTE-Steuergeräte ersetzt werden könnte oder nicht.

2.1.1 Local Interconnect Network

Das im Jahr 1998 entwickelte Local Interconnect Network ist eine Bustechnologie für einfache Mess-/Steuerungsaufgaben. Die typische LIN-Topologie besteht aus einem Master-Controller und einem oder mehreren Slave-Controller(n). Ausschließlich der Master kann einen Kommunikationsvorgang, bestehend aus einer Master-Anfrage und einer Slave-Antwort, initiieren. Daher eignen sich LIN-Controller primär für Steuerungsaufgaben, während im Fall von Messaufgaben ein regelmäßiges Abfragen (polling) der Messknoten notwendig ist. Die nur sehr geringe Bandbreitenkapazität von bis zu 20 $^{KBit}/_{s}$ schränkt den Anwendungsbereich von LIN-Controllern zusätzlich ein. Aufgrund der nur sehr niedrigen Anschaffungskosten und des einfach zu verwendenden Kommunikationsprotokolls werden LIN-Controller trotz ihrer eingeschränkten Fähigkeiten auch heute noch eingesetzt.

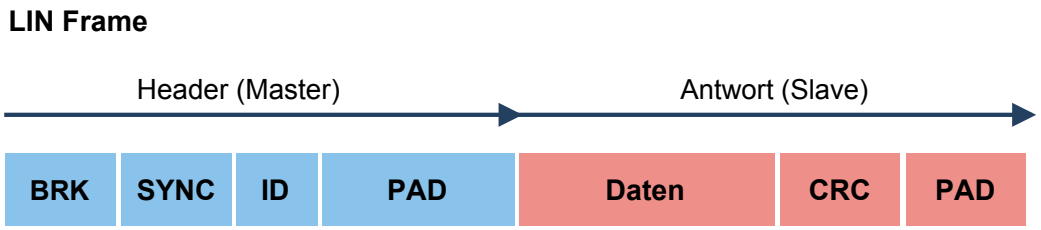

Abbildung 2.1: Veranschaulichung des Master-Slave-Konzepts in LIN Frames mit Header/Anfrage-Segment und Antwort-Segment.

2.1.2 Controller Area Network

Seit Beginn der Entwicklung des Controller Area Network (CAN) im Jahr 1983 und der erstmaligen Verwendung in Fahrzeugen der Oberklasse von Mercedes Benz im Jahr 1992 (Voss, 2005) sind die Verkaufzahlen von CAN-fähigen Microcontrollern von 50 Millionen Stück (1999) auf über 340 Millionen Stück (2003) angestiegen und werden für das Jahr 2014 auf 1 Milliarde Stück prognostiziert (CAN in Automation, 2014) (Davis u. a., 2007). CAN ist der International Organization for Standardization standardisiert (ISO 11898 1, 2003) und definiert ein Shared Medium, Multi Master Bussystem mit differentiellen Pegeln. CAN verwendet Carrier Sense Multiple Access/non destructive arbitration (CSMA/NDA) zum Kollisionsmanagement und erlaubt die Übertragung von Frames der Längen 0 bytes bis 8 bytes bei Bandbreiten von bis zu 1 $^{MBit}/_{s}$. Die empfängerselektive Nachrichtenübertragung, Nachrichtenpriorisierung und Busarbitrierung basieren auf einer entweder 11 Bit oder 29 Bit langen Nachrichten-ID. Zur Busarbitrierung werden die Bits der Nachrichten-ID schrittweise dominant (0) bzw. rezessiv (1) auf die physikalische Busleitung geschrieben und der Zustand der Busleitung geprüft. Sobald eine Nachricht höherer Prioriät bzw niedrigerer Nachrichten-ID auf den Bus geschrieben wird, werden deren dominante Bits die rezessiven Bits einer niedriger priorisierten Nachricht verdrängen und dem Sender der verdrängten Bits signalisieren, seine Übertragung einzustellen.

Die Kodierung der Bits auf der Busleitung erfolgt entsprechend des Non-Return-to-Zero (NRZ)-Ansatzes, d.h. bei der Übertragung einer langen Sequenz identischer Bits liegt ebenso lange eine identische Spannung auf der Busleitung. Dieser Umstand, das Fehlen von Pegelwechseln zwischen einzelnen Bits, macht es einem CAN-Controller sehr schwer, einzelne Bits einer solchen langen Sequenz identischer Bits zu erkennen. Um dieses Problem zu lösen kann entweder ein separates Taktsignal zur Verfügung gestellt werden oder aber eine geschickte Kodierung gewählt werden. Das CAN-Protokoll folgt letzterm Ansatz und implementiert das

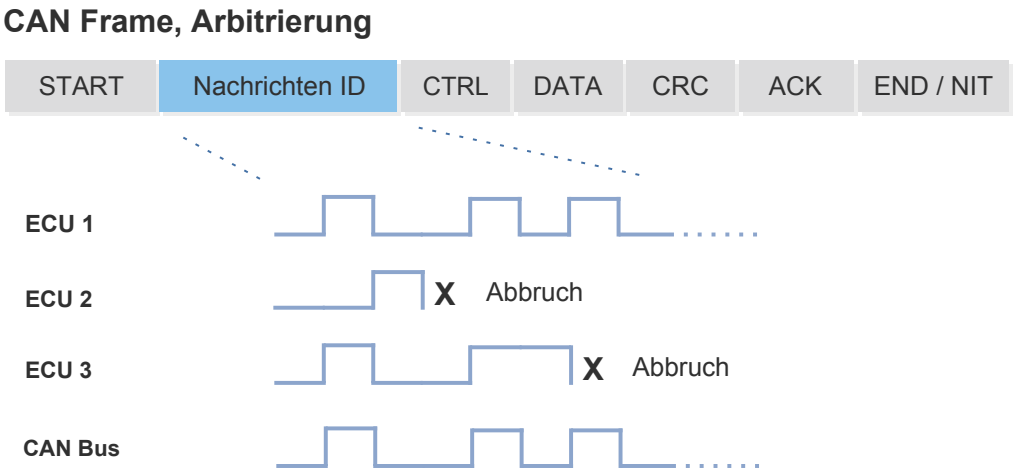

Abbildung 2.2: Veranschaulichung der CAN-Arbitrierung zwischen 3 ECU mit fiktiven Nachrichten-IDs anhand dominanter und rezessiver Bits, ECU 1 dominiert.

sogenannte Bit Stuffing. Beim Bit Stuffing wird nach einer Sequenz von fünf identischen Bits ein komplementäres Bit eingefügt, also ein Pegelwechsel erzwungen. Nachteil dieser Lösung ist, dass somit CAN-Nachrichten gleicher Nutzlastlänge nicht gleich lang [Bit] sein müssen.

2.1.3 Media Oriented Systems Transport (MOST)

Das MOST-Bussystem wurde ursprünglich aufgrund des steigenden Bandbreitenbedarfs im Zusammenhang mit der verstärkten Integration von Multimediaanwendungen in das Automobil entwickelt. MOST spezifiziert synchrone Kommunikation in unterschiedlichen Geschwindigkeitsklassen, MOST25 (25 $^{MBit}/_{s}$), MOST50 (50 $^{MBit}/_{s}$), MOST150 (150 $^{MBit}/_{s}$) auf Lichtwellenleitern bzw. elektrischen Leitungen. In der Automobilindustrie werden primär Ringtopologien oder Doppelringtopologien (Redundanz) verwendet, spezifiziert sind aber ebenso Stern- bzw. durch Hub oder Switch gebildete Topologien. Das synchrone Verhalten wird durch einen Zeitmaster realisiert, der periodische Synchronisationszeitstempel an die bis zu 64 angeschlossenen Busteilnehmer versendet. Die Busteilnehmer erhalten via Time Division Multiplex (TDM) Zugriff auf das Medium.

2.1.4 FlexRay

FlexRay ist ein Automobilbussystem für Anwendungen durchschnittlicher Bandbreitenanforderungen und hohen Anforderungen an Determinismus und Fehlertoleranz und wurde im Jahr 2005 vom FlexRay Konsortium für den Produktiveinsatz freigegeben. FlexRay erlaubt ereignisgesteuerte oder zeitgesteuerte Kommunikation bei Bandbreiten bis zu 10 $^{MBit}/_{s}$ und kann als Weiterentwicklung des ByteFlight-Bussystems von BMW angesehen werden. FlexRay verwendet ein Time Division Multiple Access (TDMA)-Verfahren, in welchem Steuergeräten bzw. Nachrichten einzelne Zeitschlitze des statischen Segments des Nachrichten-Schedule zugeordnet werden, innerhalb derer die Steuergeräte exklusiven Zugriff auf das physikalische Netzwerk haben. Die für TDMA-Verfahren notwendige Synchronisation auf eine globale Zeit wird anhand spezieller Synchronisations-Nachrichtenframes abgewickelt, welche sich von regulären Nachrichten im Wert des Sync Frame Indikator Feldes des FlexRay-Protokolls unterscheiden. Um eine ereignisgesteuerte Nachrichtenübermittlung zu ermöglichen, wird in jedem TDMA-Schedulezyklus ein sogenannter dynamischer Teil definiert, in welchem der Sendezugriff dynamisch ausgehandelt wird. Das Aushandeln des dynamischen Zugriffs erfolgt dadurch, dass jedes Steuergerät in einem spezifischen Anteil des dynamischen Segments (Minislot) auf einem freien physikalischen Bus eine Übertragung beginnen darf. Da die Minislots für eine vollständige Datenübertragung zeitlich zu kurz definiert sind, wird der Minislot zur Übertragung erweitert und verdrängt damit die Übertragungsmöglichkeit der Steuergeräte bzw. Nachrichten auf überlappenden Minislots.

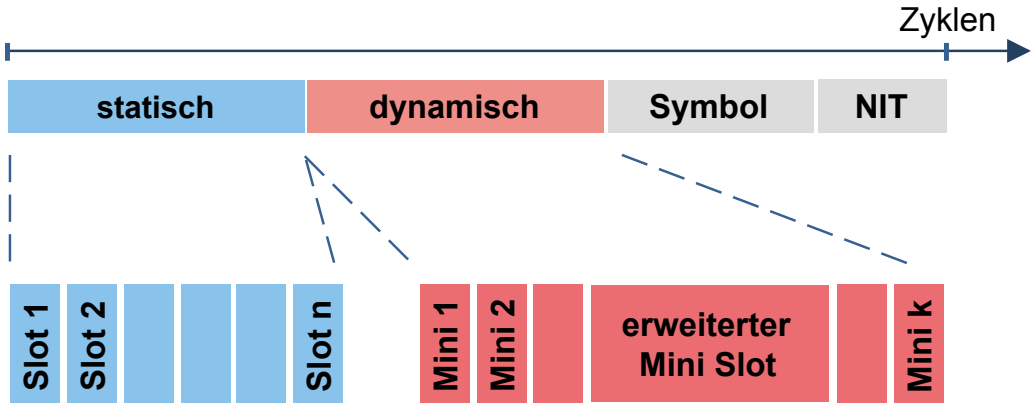

Abbildung 2.3: FlexRay Zyklus mit statischem und dynamischem Segment und einem erweitertem Minislot.

2.2 RTE als automobiles Kommunikationsnetzwerk

Das gemeinhin als Ethernet bekannte Netzwerkprotokoll nach IEEE 802.3 (2012) unterstützt ausschließlich das asynchrone Versenden von Nachrichten und bietet keine Mechanismen zur Priorisierung dieser Nachrichten an. Aus diesem Grund könnten unter Verwendung des Ethernetprotokolls einzelne Sender mit bandbreitenintensiven Anwendungen wie z.B. einem Video-Stream eine Infrastrukturkomponente wie z.B. einen Switch so überlasten, dass Nachrichten anderer Sender vernachlässigt werden. Die im Rahmen der Quality of Service (QoS)-Erweiterungen des IEEE 802.1 Standards durch IEEE 802.1 (2011) eingeführten Class of Service (CoS) sind aufgrund der geringen Auflösung der Prioritätsklassen (3 Bit) nicht dazu geeignet, konkurrierendes Verhalten von Nachrichten auszuschließen, da bei hoher Systemkomplexität eine Mehrfachverwendung der CoS durch unterschiedliche Nachrichten auftreten würde. Dieses nichtdeterministische Verhalten eignet sich nicht für sicherheits- und zeitkritische Anwendungen.
Hartes Echtzeitverhalten lässt sich in Kommunikationsnetzwerken nur anhand von Protokollen erreichen, die ein deterministisches, vorhersagbares Verhalten aufweisen. Um ein deterministisches Verhalten zu realisieren, können unterschiedliche Maßnahmen ergriffen werden, die aber alle darauf ausgelegt sind entweder das Sendeverhalten der Hosts zu kontrollieren. Eine abgeschwächte Variante harten Echtzeitverhaltens kann durch eine Kontrolle der maximalen, gleichzeitig verwendeten Bandbreite erreicht werden. Unterschiedliche Ansätze zur Kontrolle des Sendeverhaltens finden sich in Token-Netzwerken und TDMA-Netzwerken.
Token-(Ring)-Netzwerke gehören zu den Urgesteinen der Netzwerktechnik und werden dadurch realisiert, dass die Netzwerkteilnehmer untereinander eine Kennzeichnung zur Sendeerlaubnis austauschen. Zu jedem Zeitpunkt kann nur ein Sender diese Kennzeichnung tragen und wird diese Kennzeichnung nach dem Sendevorgang an einen anderen Netzwerkteilnehmer weiterreichen. Schwachpunkte dieses Mechanismus liegen in der für die Übertragung der Kennzeichen genutzten Ressourcen, der Anzahl der Netzteilnehmer und der Fehlerverwaltung. So ist die maximale Sendehäufigkeit eines Netzwerkteilnehmers sowohl von der Anzahl der Netzteilnehmer als auch von den durch die Kennzeichenübertragung bedingten Sendepausen abhängig. Die Fehlerverwaltung wird in dem Moment zeitaufwendig, sobald ein Netzwerkteilnehmer im Besitz der Kennzeichnung fehlerhaft ist und er somit die Kennzeichnung nicht oder auf unvorhergesehene Art und Weise weitergibt. Alternative Implementierungen mit identischen Schwachstellen verwenden die Kennzeichnungsnachrichten selbst als Hülle für die auszutauschenden Daten.

Ein konservativer Ansatz zur Kontrolle des Sendeverhaltens der Netzteilnehmer ist in TDMA-Protokollen realisiert. In diesen Protokollen verfügen die Netzteilnehmer und Infrastrukturkomponenten über eine synchronisierte, globale Zeit und vordefinierte, kollisionsfreie Sendezeitpunkte. Vorteile der TDMA-Protokolle ist die deterministische und zeitdeterministische Übertragung der Kommunikationsdaten ohne die Gefahr der ungeplanten Verzögerung von Nachrichten durch hohes Nachrichtenaufkommen dritter Netzteilnehmer. Diese Vorteile werden durch einen erhöhten Aufwand im Zuge der Planung der Sendezeitpunkte und durch einen Protokoll-Overhead im Zuge der Synchronisation der globalen Zeit unter den Netzteilnehmern erkauft.
Die Erfüllung einer weniger harten Echtzeitanforderung ist durch die Kontrolle der maximalen, gleichzeitig verwendeten Bandbreite durch eine die Netzteilnehmer überwachende Infrastrukturkomponente möglich. Hierbei wird jedem Netzteilnehmer ein Anteil an der Gesamtkapazität des Netzwerkes zugeteilt, welche dieser ausnutzen darf, aber nicht muss. Die Überwachung der von einem Netzteilnehmer genutzten Bandbreite kann entweder aufwändig dynamisch durch Regelkreise oder vereinfacht und bevorzugt durch Verwendung eines Bandwidth Allocation Gap (BAG)-Verfahrens realisiert werden. BAG-Verfahren definieren eine statische Verarbeitungspause zwischen aufeinanderfolgenden Nachrichten eines Senders, welche sich an der maximalen Nachrichtengröße des Protokolls und der Bandbreite der physikalischen Schicht orientiert. Somit ist die maximale Datenrate eines Senders in einem frei wählbaren Zeitintervall auf Basis der Nachrichtengröße, der Sendepausen und der Netzwerkbandbreite vorhersagbar. Der Nachteil dieser Form der Datenflusskontrolle findet sich im Echtzeitverhalten von Burst-Nachrichten: Prinzipiell akkumulieren sich Latenzen aufgrund der Sendepausen in Richtung der letzten Nachrichten eines Sende-Bursts. Obwohl eine globale Zeitsynchronisation im Gegensatz zu TDMA-Protokollen nicht notwendig ist, muss dennoch eine Ermittlung und Planung des Sendeverhaltens zur Entwurfszeit durchgeführt werden, um die BAG der einzelnen Nachrichten in Hinblick auf Netzwerkkapazität und Anzahl der Kommunikationsteilnehmer zu entwickeln.
Im Kontext dieser Thesis wird ein RTE verwendet und betrachtet. RTE ist ein Gattungsbegriff zur Klassifizierung von Erweiterungen und Anpassungen des Ethernet-Protokolls, welche unterschiedliche Implementierungen einer zeitdeterministischen Nachrichtenübertragung erreichen. Die Norm IEC 61784 definiert ca. 10 unterschiedliche RTE-Protokolle und die entsprechenden, proprietären und geschützten Implementierung in den Communication Profile Families (CPF) (Felser (2005)). Die experimentell gewonnenen Ergebnisse dieser Arbeit basieren auf TTEthernet entsprechend AS6802 (SAE International (2011)), einer natürlichen Erweiterung des IEEE 802.3 Standards (IEEE 802.3 (2012)), können aber durchaus auf andere

RTE-Implementierungen und deren spezifische Unterschiede übertragen werden. TTEthernet erfordert spezielle Infrastrukturkomponenten (z.B. Switch), welche die Mechanismen zur Umsetzung der Übertragungsverfahren realisieren, und erfordert Netzwerkteilnehmer, die einen Protokollstacks mit einer proprietären API verwenden.
TTEthernet definiert drei unterschiedliche Nachrichtenklassen, welche sich die vorgestellten Kontrollmechanismen zur Erlangung von Echtzeitverhalten zu Nutze machen: Die einem TDMA-Ansatz folgende Time Triggered (TT) Nachrichtenklasse, die bandbreitenlimitierende Rate Constraint (RC) Klasse und die Best Effort (BE) Klasse, welche IEEE 802.3 (2012) entspricht. In der TT-Nachrichtenklasse wird die Ziel-Media Access Control (MAC)-Adresse im Ethernet-Nachrichtenheader durch Felder für einen Critical Traffic (CT) Indikator und die Critical Traffic Identifier (CTID) ersetzt. Anhand des CTID-Feldes werden unterschiedliche Nachrichtenprioritäten mit einer faktischen Auflösung von 12 Bit realisiert. Die RC-Klasse bedient sich konfigurierbarer BAG entsprechend des AFDX-Protokolls nach ARINC 664 1 (2002) und ist der Übertragung von TT-Nachrichten untergeordnet. Für klassische, nichtdeterministische Datenübertragung kann die BE-Klasse gewählt werden. Insgesamt ergibt sich eine Priorisierung der Nachrichten der unterschiedlichen Nachrichtenklassen entsprechend Abbildung 2.4.

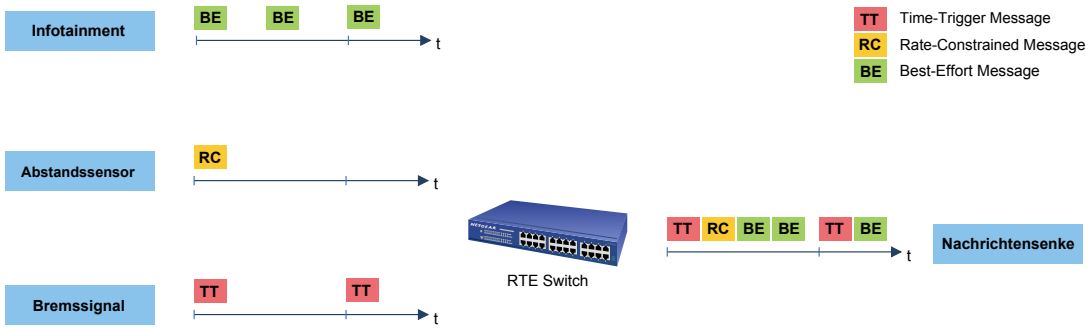

Abbildung 2.4: Nachrichtenklassen und ihr Zusammenspiel auf einer exemplarischen Übertragungsstrecke hin zu einer Nachrichtensenke.

Diese Implementierung hat unterschiedliche Konsequenzen: Zum einen muss für die Verwendung des TDMA-Ansatzes eine globale, synchronisierte Zeitbasis zur Verfügung stehen, zum anderen muss die in TT-Nachrichtenheadern ersetzte Ziel-MAC-Adresse kompensiert werden. Die globale Zeitsynchronisation basiert auf dem Austausch von Synchronisationsnachrichten in Protocol Control Frames (PCF), welche über einen spezifischen, von TT-Kommunikation unterschiedlichen Wert im Type-Protokollfeld des Ethernet-Nachrichtenheaders erkennbar sind. Der Synchronisationsmechanismus folgt einer Hierarchie aus einem oder mehreren

Synchronization Master (SM), keinem oder einem Compression Master (CM) und einem oder mehreren Synchronization Client (SC). Mehrere SM übermitteln ihre Ansicht der globalen Zeit an den CM, welcher aus den unterschiedlichen Zeiten eine gemeinsame, globale Zeit errechnet. Sobald nur ein SM existiert, kann auf die Bildung der gemeinsamen Zeit durch einen CM verzichtet werden. Die globale Zeit wird anschließend vom CM bzw. alleinigen SM an die unterschiedlichen SC weitergeleitet, welche ihre Systemzeit an die globale Zeit anpassen.
Die Kompensierung der im Nachrichtenheader ersetzten Ziel-MAC-Adressen erfolgt durch statische, vorkonfigurierte Nachrichtenrouten. Hierbei wird jede CTID mit genau einem entsprechend ARINC 664 1 (2002) definierten Virtual Link Identifier (VLID) verknüpft. Jeder VLID ist wiederum zur Entwurfszeit einer Route von einem Sender zu einem oder mehreren Empfängern zuzuordnen. Über diese Assoziationskette kann die Netzwerkinfrasturktur aus einer im Nachrichtenheader enthaltenen CTID ebenso die Routinginformationen für die empfangene Nachricht ableiten.

3 Problemstellung und verwandte Arbeiten

Diese Arbeit zeigt eine in einem Prototypenfahrzeug implementierte Strategie zur Anbindung von herkömmlichen Automobilbussystemen an ein RTE Backbone Netzwerk. Diese Strategie beinhaltet die Lösung der informationstechnischen Problemstellung, die Einbindung von Real-time Ethernet in bestehende Netzwerkinfrastrukturen, und berücksichtigt dabei wirtschaftliche und industrielle Erfordernisse. Den wirtschaftlichen und industriellen Erfordernissen wird dadurch Rechnung getragen, dass die Umsetzung der Strategie unaufwendig, unter Weiterverwendbarkeit von weiten Teilen von Hardware, Software und Fachwissen im speziellen Entwicklungsablauf der Automobilindustrie umsetzbar ist.

3.1 Entwurf und Integration automobiler Systeme

Die von dieser Arbeit berührten Bereiche des Entwurfs- und der Integration automobiler Systeme sind

- Bordnetz und
- Software.

Die Verwendung eines RTE Backbone zur Automobilvernetzung greift in das Systemdesign und die Systemarchitektur des Bordnetzes ein. Sobald dieser Eingriff im Entwicklungsprozess lokalisiert ist, können Maßnahmen zur Optimierung der Migration hin zu einem RTE Backbone ergriffen werden. Die an den RTE Backbone angebundenen Steuergeräte und die Schnittstelle zwischen RTE Backbone und dritten Bussystemen haben direkten Einfluss auf den Softwareentwicklungsprozess für weiterzuverwendende Altsteuergeräte. Es ist notwendig, dieses Zusammenspiel zu untersuchen, um eine möglichst konfliktarme Migration zu ermöglichen.

3.1.1 Bordnetz

Automobile Bordnetze und Steuergeräte-Schnittstellen werden üblicherweise auf Basis des OSI-Schichtmodells betrachtet. Hierbei werden komplexe Kommunikationsstrukturen in Bus-, Ring-, oder Stern-Topologie zusammen mit Aktoren und Sensoren zu einem Gesamtsystem

modelliert. Dazu wird aus einem Systemdesign eine Systemarchitektur abgeleitet, welche über den kompletten Produktlebenszyklus verwaltet werden muss.

3.1.1.1 Systemdesign und Systemarchitektur

In der Systemdesignphase werden auf Basis funktionaler und nicht-funktionaler Anforderungen die unterschiedlichen Eigenschaften der Hardware und Software festgelegt. Zu diesen Eigenschaften gehören neben Einsatzbedingungen, Qualitätsmerkmalen und Komponentenkompatibilität ebenso konkrete technologische Eigenschaften wie Vorgaben zu Steuergerätetechnologie und physikalische Buseigenschaften. Die aus dem Systemdesign resultierenden Vorgaben gelten für die Systemarchitekturphase, in welcher die Struktur des Bordnetzsystems inklusive einzelner Systemelemente beschrieben wird. Die Beschreibung umfasst neben der Spezifikation der Systemschnittstellen, der Daten- und Softwarearchitektur und des Datenflusses unter anderem auch Zuordnung abstrakter Funktionalität auf konkrete Systemelemente wie z.B. Steuergeräte. Aus Sicht der physikalischen Schicht werden durch die Systemarchitektur die Daten- und Energieleitungsanordnung und die einzelnen elektrischen und elektronischen Systeme definiert. Eine abschließend validierte und optimierte Systemarchitektur erlaubt letztendlich die Erstellung einer Konfiguration. Die Konfiguration besteht aus der Kommunikationsmatrix eines oder mehrerer Bussysteme, den Konfigurationsdaten von Kommunikationsgateways und Prozessdefinitionen für das Änderungsmanagement. Sie ermöglicht außerdem die Bereitstellung herstellerspezifischer Steuergerätesoftwaremodule durch Zulieferer und Dienstleister (vergl. Wallentowitz und Reif (2011)).

3.1.1.2 Einordnung

Grundlegend gibt es zwei unterschiedliche Ansätze zur Migration hin zu einem RTE-gestütztem Bordnetz. Zum einen kann das vollständig neue Design eines Bordnetzes inklusive aller Systeme auf Basis von RTE initiiert werden, zum anderen kann das Bordnetz unter Weiterverwendung bestehender Systeme sinnvoll durch RTE unterstützt werden. Da ein vollständig neues Design das Durchlaufen des Produktentstehungsprozesses für sämtliche verwendeten Systeme erfordern würde, ist ein derartiger Ansatz aus Kosten- und Aufwandsgründen vorerst nicht realistisch. Eine Möglichkeit zur sinnvollen Unterstützung bestehender Bordnetze durch RTE besteht in der Verwendung als Backbone für Kommunikation zwischen den Systemen. Dadurch wird ein hoher Grad an Wiederverwertbarkeit der existierenden Systeme unter optimaler Ausnutzung der Vorzüge des RTE, wie z.B. Bandbreitenkapazität, zeitlicher Determinismus und Austauschbarkeit der physikalischen Schicht erreicht. Insbesondere die Wiederverwertbarkeit existierender Systeme stellt einen Schlüsselpunkt für eine kosteneffiziente Migration dar. Es

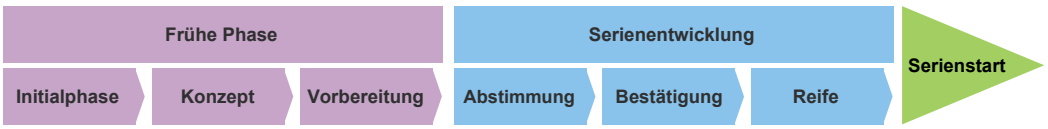

Abbildung 3.1: Entstehungsprozess eines automobilen Systems von Initialphase bis Serienstart nach Wallentowitz und Reif (2011)

können neben Hardware und Steuergerätesoftware ebenfalls Tools, Anwendungsentwicklerwissen und firmenspezifische Erfahrungen übernommen werden.

3.1.2 Software

Der Anteil an Software am Gesamtbestand der Funktionen eines Automobils und die Bedeutung softwarebasierter Funktionalität als Element von Herstelleralleinstellungsmerkmalen hat in den letzten Jahren deutlich zugenommen. Parallel dazu hat sich die Entwicklung von Steuergerätesoftware von einer nebenläufig zur Hardwareentwicklung durchgeführten Notwendigkeit zu einer höher organisierten, teilweise im Fokus stehenden Aufgabe weiterentwickelt.

3.1.2.1 Entwicklungsprozess

Software- und Hardwareentwicklung werden in der Automobilindustrie weitestgehend entsprechend des V-Modells durchgeführt. Das V-Modell definiert hierarchische Prozessebenen, bestehend aus sich verfeinernden Paaren von Anforderung und Validierung bzw. Spezifikation und Verifikation. Der namensgebende V-Charakter entsteht aus dem Ansatz, die Prozessebenenhierarchie von den Anforderungen bis zur Implementierung zu durchlaufen um daraufhin die Implementierung bis hin zum Abnahmetest zu überprüfen (vergleiche Abbildung 3.2). Das V-Modell kann auf unterschiedlichen Abstraktionsebenen angewandt werden. So kann ein einzelnes Steuergerät ebenso wie ein System, bestehend aus diversen Steuergeräten oder anderen Komponenten, entsprechend des V-Modells entwickelt werden. Die verteilte Entwicklung in der Automobilindustrie durch Hersteller und Zulieferer erfordert wohldefinierte Übergabepunkte und Qualitätssicherungsmethoden zwischen den beteiligten Parteien. Üblicherweise ist diese Schnittstelle auf Ebene der Software- und Hardwarespezifikation und des Steuergerätetests angesiedelt.

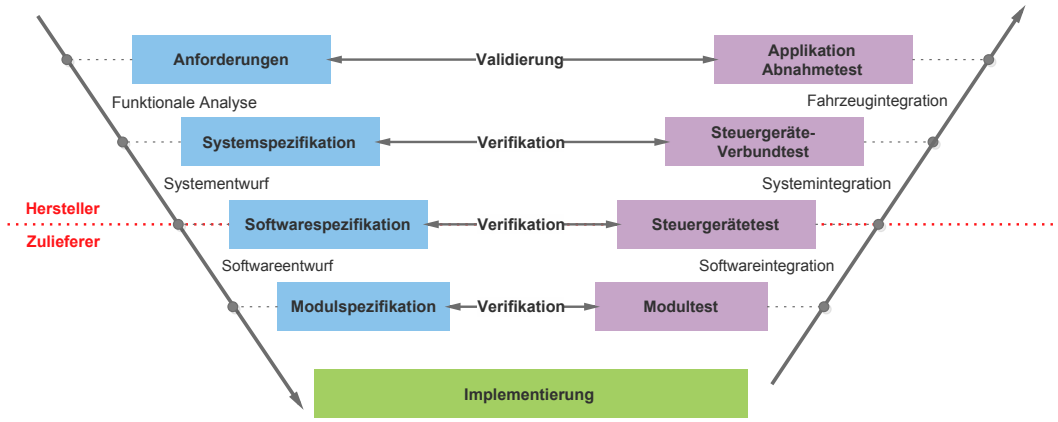

Abbildung 3.2: Automobiler Entwicklungsprozess entsprechend des V-Modells nach Wallentowitz und Reif (2011)

3.1.2.2 Einordnung mit Fokus auf Wiederverwendbarkeit

Der Entwurf und Einsatz eines RTE Backbone zur Vernetzung mehrerer Steuergeräte und automobiler Bussysteme ist auf diversen Hierarchieebenen des V-Modells denkbar. Die Auswahl der Hierarchieebene kann allerdings entscheidenden Einfluss auf Entwicklungsaufwand, -dauer und -kosten haben. So bedeutet jede Überschreitung der Zuständigkeitsgrenze zwischen Hersteller und Zulieferer einen hohen Aufwand zur formalen Absicherung mit Blick auf die Spezifikationen und Qualitätssicherung. Darüber hinaus wird anhand des V-Modells die Bedeutung der Software- und Hardwarewiederverwendbarkeit deutlich. Eine Veränderung der Systemspezifikation (Nutzung eines RTE Backbone) führt bei Wiederverwendung existierender Steuergeräte nur für neue Komponenten und Systeme zu einem vollständigen Durchlaufen des V-Modells. Die existierenden Steuergeräte sind bereits im Sinne der Software- und Hardwarespezifikation und der entsprechenden Steuergerätetests verifiziert. Es besteht also die Herausforderung, eine Systemspezifikation durch Entwurf, Implementierung und Tests RTE-orientierter Komponenten dahingehend zu verändern, dass der Steuergeräteverbundtest transparent bezüglich der RTE-Komponenten denselben Anforderungen an den Verbund genügt wie zuvor.

3.1.2.3 Tests

Während davon auszugehen ist, dass wiederverwendete Steuergeräte bereits beim Zulieferer bezüglich der Einhaltung der entsprechenden Spezifikation getestet wurden, müssen die RTE-

Komponenten noch getestet werden. Hierzu kommen unterschiedliche Methoden in Frage, die unterschiedliche Aspekte der Komponenten betrachten. Bei Hardware in the Loop (HIL)-Simulation mit Restbussimulation liegt der Betrachtungsschwerpunkt auf der Verifikation des Wohlverhaltens der Komponenten in Zusammenhang mit der Außenwelt, konkret Fahrzeug, anderen Steuergeräten, Sensoren und Aktoren. HIL-Simulationen können flexibel während des gesamten Entwicklungsprozesses genutzt werden. In funktionalen Tests (Black Box Tests) wird stattdessen ein Steuergerät mit einer Auswahl an Stimuli-Signalen betrieben, um das Steuergeräteausgabeverhalten mit den aus der Spezifikation abgeleiteten Erwartungswerten zu vergleichen. Funktionale Tests werden üblicherweise durch strukturelle Tests ergänzt. Strukturelle Tests erfordern Kenntnisse bezüglich der Implementierungsdetails einer Komponente und ermöglichen das gezielte Auslösen aller Ausführungspfade der Software. Ziel der strukturellen Tests ist es, Randfälle der Ausführungspfade zu testen und potentiell ungenutzte Programmcodebestandteile zu identifizieren. Im Rahmen dieser Arbeit werden funktionale Tests und strukturelle Tests zur Verifikation durchgeführt.

3.2 Verbindung von Netzwerken, klassisch und automobil

Um Netzwerke miteinander zu verbinden, bietet die nicht-automobile Netzwerktechnik unterschiedliche Komponenten mit unterschiedlichen Fähigkeiten für spezielle Anwendungsszenarien. Diese Komponenten lassen sich in folgende Gruppen einordnen:

- Bridge
- Switch
- Router
- Gateway

Es gibt unterschiedliche Ansätze zur Abgrenzung dieser Gruppen gegeneinander. Während eine Abgrenzung allein über die Implementierungsschicht entsprechend des Open Systems Interconnection (OSI)-Schichtmodells zeitweilig populär war, wurde dieses Abgrenzungscharakteristikum durch die Implementierung von Switches auf Basis von OSI-Schicht 2 und OSI-Schicht 3 aufgeweicht. Im Rahmen dieser Arbeit werden die Gruppen anhand ihres Funktionsumfanges und der Funktionseigenschaften und der verwendeten OSI-Schicht abgegrenzt, wobei auch hier aufgrund der Implementierungsvielfalt Überschneidungen möglich sind. Bridges verbinden im üblichen Sinne zwei unterschiedliche Netzwerksegmente. Diese Verbindung wird durch das Weiterleiten von Frames aus dem Quellsegment in das Zielsegment erreicht. Switche verhalten sich wie multi-port Bridges. Das Unterscheidungscharakteristikum auf

dieser Ebene ist die Anzahl der verbundenen Netzwerksegmente. Allerdings können Switche die Bridgefunktionalität um die Möglichkeit, Frames anhand der OSI-Schicht 2 MAC-Adresse zu filtern, erweitern. Also wird eine selektive Frameweiterleitung ermöglicht. Router hingegen untersuchen die eingehenden Frames auf Ebene der OSI-Schicht 3 z.B. anhand einer Internet Protocol (IP)-Adresse und ermöglichen analog zu den Switches eine selektive Frameweiterleitung. Router können zusätzlich ebenfalls Sicherheitsfunktionalität (Firewall) und Network Address Translation (NAT)- bzw. Port Address Translation (PAT)-Funktionen anbieten.
Die Gruppe der Gateways ist dadurch gekennzeichnet, dass Gateways Netzwerksegmente und Netzwerke auf Basis unterschiedlicher Netzwerkprotokolle verbinden. Dazu werden bei Bedarf Protokollinformationen eingehender Nachrichten entfernt und Protokollinformationen entsprechend des Zielnetzwerkes hinzugefügt. In der nicht-automobilen Netzwerktechnik werden Gateways auf den OSI-Schichten 4 bis 7 angeordnet.

In der automobilen Netzwerktechnik werden diverse, teilweise unterschiedliche, Bussysteme durch ein zentrales Gateway verbunden (vergleiche Abbildung 3.3). Die Klassifizierung als Gateway ist hauptsächlich durch das Charakteristikum der Verwendung unterschiedlicher Netzwerkprotokolle gegeben.
Die von automobilen Gateways bereitgestellte Funktionalität beschränkt sich nicht auf die Protokollübersetzung, sondern besteht ebenso so aus

- Nachrichtenfilterung
- Nachrichtenzuordnung
- Nachrichtenübersetzung und der
- Aufbereitung von Signalen.

Unter Nachrichtenfilterung ist hierbei das selektive Übertragen einzelner Frames anhand eines Nachrichtenidentifikationskriteriums wie z.B. einer CAN-Nachrichten-ID oder einer Ethernet-MAC-Adresse zu verstehen. Die Nachrichtenfilterung ist hierbei ein wesentliches Merkmal zur Buslastreduzierung im Gegensatz zum nicht-bedingten flooding. Eine Nachrichtenzuordnung findet in automobilen Gateways entweder im Zuge einer Protokollübersetzung oder aber zur Auflösung von Konflikten bei Übertragungen zwischen identischen Netzwerkprotokollen statt. So kann eine eingehende CAN-Nachricht mit einer bestimmten ID entweder an z.B. eine vorkonfigurierte Ethernet-MAC-Adresse weitergeleitet werden oder aber auf eine abweichende, vorkonfigurierte CAN-Nachrichten-ID abgebildet werden. Letzterer Fall kann dann notwendig sein, wenn die Quell-Nachrichten-ID im Zielbus durch eine semantisch unterschiedliche Nachricht verwendet wird. Die Aufbereitung von Signalen fasst sämtliche Funktionen zusammen, welche auf Basis eines Nachrichtenidentifikationskriteriums die eingehende Nachricht

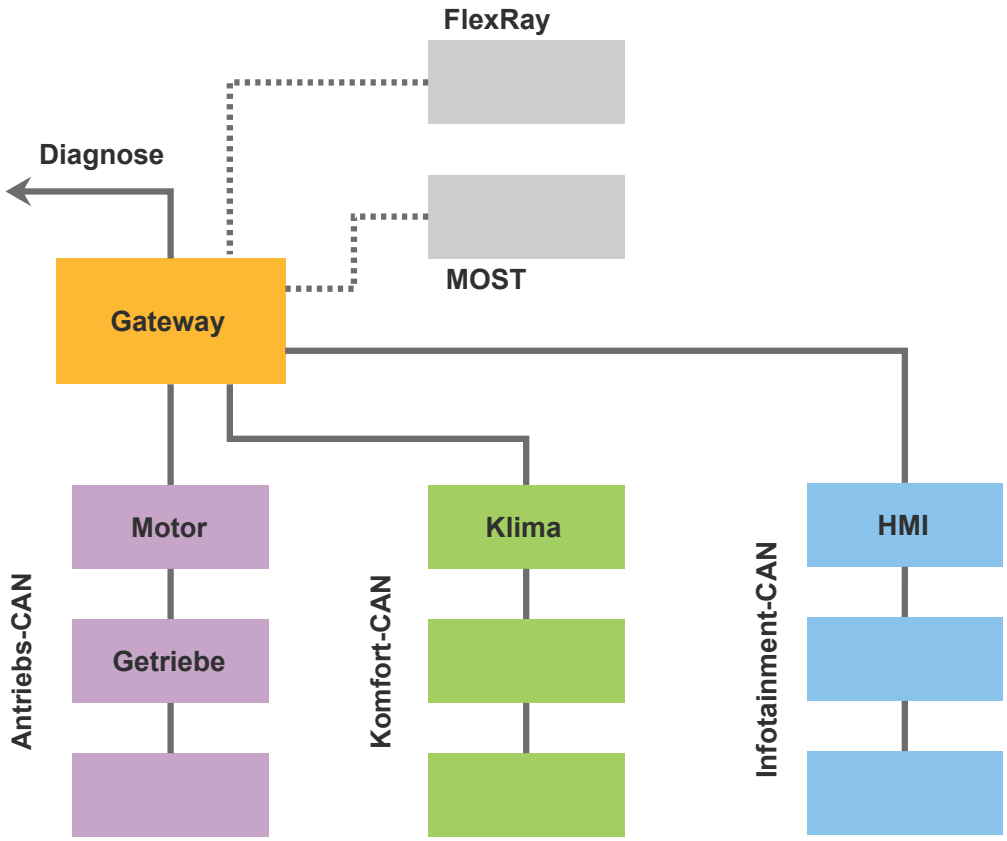

Abbildung 3.3: Fahrzeugnetz mit zentralem Gateway, abstrahiert von einem Golf V, erweitert um potentielle MOST- und FlexRay-Strecken.

vollständig oder anteilig weiterverarbeiten und das Ergebnis dieser Verarbeitung an Stelle der ursprünglichen Nachricht in das Zielbussystem weiterleiten.

3.3 Verwandte Arbeiten

In der vernetzten Welt wurden die Verbindungen und die Verbindbarkeit unterschiedlicher Netzwerksegmente bereits früh wissenschaftlich untersucht. Prinzipiell müssen dabei zwischen drei Arten von Netzwerk(segment)verbindungen unterschieden werden (Benhamou und Estrin (1983)):

- Verbindung identischer Netzwerktypen
- Verbindung von Netzwerken kompatibler Protokolle
- Verbindung inkompatibler Netzwerke

Der im Rahmen dieser Arbeit interessante Bereich der Unterscheidung liegt zwischen den Netzwerken kompatibler Protokolle und den inkompatiblen Netzwerken. Netzwerke kompatibler Protokolle sind dadurch gekennzeichnet, dass sie auf den OSI-Schichten ab inklusive Schicht 3 als homogen zu interpretieren sind. Inkompatible Netzwerke verhalten sich invers dazu und werden durch Gateways in der Anwendungsschicht realisiert. Besondere Herausforderungen der Verbindung inkompatibler Netzwerke ergeben sich aus dem OSI-Schichtmodell, welches die Übertragungszuverlässigkeit auf Ebene der Transportschicht ansiedelt. Daher müssen in einem Gateway auf Anwendungsebene Maßnahmen getroffen werden, um Paketverlust oder -duplikation zu vermeiden, fehlerbedingte Paketinhaltsveränderungen auszuschließen und QoS-Eigenschaften zu wahren.

3.3.1 Gatewayentwurf

Die Verbindung von homogenen oder protokoll-kompatiblen Netzwerken kann entsprechend einer universellen Schematik erfolgen (Bochmann und Mondain-Monval (1990)). Wie von Bochmann und Mondain-Monval gezeigt, kann eine Analyse der auf den einzelnen OSI-Schichten angebotenen und genutzten Service Primitives dazu genutzt werden, innerhalb eines Gateways die Kommunikationsschnittstellen zu verbinden. Eine erfolgreiche Verbindung definiert Pfade zwischen den Service Primitives der Kommunikationsschnittstellen und bildet dabei den Funktionsumfang der Protokolle umkehrbar auf die Service Primitives der jeweiligen Zielschnittstelle ab. Ein derartiger Ansatz erfordert vollständigen Zugriff auf sämtliche OSI-Schichten kann die Neuentwicklung diverser Komponenten wie z.B. eines IP-Stack erfordern. Wesentliche Voraussetzung für eine erfolgreiche Verbindung ist hierbei die theoretische Abbildbarkeit aller

Service Primitives auf Service Primitives der Zielschnittstelle, die aber nur bei homogenen Netzwerken angenommen werden kann. Obwohl sich somit das Konzept der Gatewayentwicklung nach Bochmann und Mondain-Monval nicht auf die heterogenen Automobilbussysteme übertragen lässt, ist die Betrachtung der Kommunikationsschnittstellen aus der Perspektive der Service Primitives ein hilfreiches Instrument, um mögliche Konflikte in der Verbindung inhomogener Netzwerke zu entdecken und die Verbindbarkeit dieser Netzwerke zu bewerten.

3.3.2 Bekannte CAN-Gateways

Schmidt u. a. entwickelten und untersuchten ein FlexRay/CAN-Gateway hinsichtlich der Nachrichtenzuordnung zwischen den Protokollen und der WCRT der durch das Gateway vermittelten Signale (Schmidt u. a. (2010)). Als Nachrichtenzuordnungsverfahren wählten die Autoren von CAN in Richtung FlexRay eine 1:1-Zuordnung von CAN Nachrichten IDs auf FlexRay-Nachrichten, wobei periodische CAN Nachrichten auf periodische FlexRay-Nachrichten abgebildet wurden. In Senderichtung des CAN-Busses lösen die Autoren für FlexRay-Nutzlasten, welche den Nutzlast einer CAN-Nachricht überschreiten würden, gegebenenfalls mehrere CAN-Nachrichten aus, was ebenso dem Ansatz aus Seo u. a. (2006) entspricht. Die WCRT-Berechnung basiert auf der Annahme einer Akkumulation der FlexRay-, CAN- und Gateway-Latenzen ohne weitere Abhängigkeiten. Daher ist es den Autoren möglich, einen Gesamtschedule für CAN und FlexRay sukzessive aus einem CAN-Schedule und einem auf diesen Schedule aufbauenden FlexRay-Schedule zu gewinnen.
Da eine Anforderung an diese Masterarbeit darin besteht, die Weiterverwendung bestehender CAN-Anwendungen zu begünstigen, ist im Kontrast zu Schmidt u. a. (2010) nicht davon auszugehen, dass neue Nachrichten in bestehende CAN-Schedules eingearbeitet werden müssen. Es besteht vielmehr die Herausforderung, jegliche Form unter Umständen manipulierender Nachrichtenübermittlung durch existierende Gateways flexibel erfassen und nachbilden zu können. Grundlegende Prinzipien wie die der Nachrichtenzuordnung werden hinsichtlich der Bandbreitennutzung optimiert, da eine 1:1-Zuordnung von CAN-PDU auf RTE-PDU aufgrund der hohen minimalen RTE-Framegröße zu einem signifikanten Overhead führen würden.
In Scharbarg u. a. (2005) untersuchten Scharbarg u. a. die Einhaltung von Zeitgarantien in der Kommunikation unterschiedlicher CAN-Busse. Die Autoren erstellten eine Topologie mehrerer über ein FTT-Ethernet-Derivat (Pedreiras u. a. (2005)) verbundener CAN-Busse und stellten eine inakzeptabel hohe Anzahl an Verletzungen der Zeitgarantien der CAN-Nachrichten fest. Die Untersuchung der Autoren führt diese Zeitgarantieverletzung ausschließlich auf Ethernet-Kollisionen innerhalb der Non-Switched Ethernet Topologie zurück. Im Kontext eines automobilen RTE-Backbone treten derartige Kollisionen aufgrund der Switch-gestützten

Vollduplexübertragung nicht auf. Es verbleibt, die von den Autoren ebenfalls diskutierte, aber von der physikalischen Schicht und der spezifischen Topologie stark abhängige Latenz eines Backbone-Netzwerkes zu betrachten.

3.3.3 ETAS EDE

In Kooperation der ETAS GmbH und der Robert Bosch GmbH wurde mit der ETAS Data Engine (EDE) im Jahr 2013 eine hardwareseitige Lösung zur konfigurierbaren Verarbeitung von Datenpaketen außerhalb des klassischen CPU-Rahmens angekündigt. Das Konzept der EDE sieht vor, dass ein Intellectual Property Core (IP-Core) parallel zur klassischen CPU in einem System on Chip (SoC) implementiert wird, wobei der IP-Core den Datentransfer zwischen den Kommunikationsschnittstellen des SoC übernimmt (ETAS GmbH (2013)). Ersatzweise soll die EDE ebenfalls als Coprozessor bezogen werden können und dann via PCIe oder xMII an die eigentliche CPU angebunden werden (Hogenmüller und Triess (2013)). In dieser Aufgabenteilung wird die CPU nur mit der Verwaltung und Konfiguration des IP-Core belastet. Die Vorzüge dieses Konzeptes liegen in Abhängigkeit von der Güte des IP-Core in äußerst niedrigen und planbaren Latenzen des on-chip Datentransfers. Nachteil einer derartigen Lösung sind die Bindung an Lizenznehmer des IP-Core und deren Hardware sowie die Bindung an den konkreten Funktionsumfang der Data Engine. In der Evaluationsphase der Migration in Richtung RTE-gestützter Automobilbussysteme erschwert die erste Bindung eine Untersuchung des Wiederverwendbarkeitspotentials existierender Steuergeräte. Des weiteren kann die Bindung an den Funktionsumfang gegebenenfalls die Umsetzung notwendiger Datenverarbeitungs- bzw. Datenmanipulationsmechanismen verhindern. In der Evaluationsphase kann die Flexibilität einer Softwarelösung die Vorzüge einer Hardwarelösung relativieren. In der Serienfertigung hingegen ist eine Hardwarelösung problemlos und vorteilhaft einzusetzen.

4 Anforderungen

Im vorangegangenen Kapitel wurde die Zielsetzung dieser Arbeit in das Umfeld des Entwurfs und der Integration automobiler Systeme eingeordnet und grundlegende Eigenschaften und Methoden zur Entwicklung von Kommunikationsgateways vorgestellt. Dieses Kapitel definiert die Anforderungen an die Vorgehensweise zur Anbindung etablierter Bussysteme an einen RTE Backbone sowie Anforderungen an deren Implementierung.

4.1 Entwicklung der Anforderungen

Die Anforderungsanalyse ergibt zu Beginn der Planungsphase in der Entwicklung eines Produktes eine Menge wohldefinierter funktionaler und nicht-funktionaler Anforderungen, die strukturiert zusammen mit Anforderungsbewertungen und Metriken zur Überprüfung der Anforderungserfüllung dokumentiert werden. Bei der Entwicklung von Systemen auf Basis neuartiger Technologien sind im Kontrast dazu viele Produkteigenschaften und mögliche Anwendungsfälle in der Planungsphase noch nicht bekannt, so dass eine Fortschreibung der Anforderungen mit Zunahme von Wissen und Erfahrung bezüglich der neuartigen Technologie notwendig wird.
Die Migration hin zu einer Nutzung eines RTE Backbone beinhaltet neben der Verwendung einer neuartigen Technologie ebenso den Bruch mit dem in der Automobilindustrie etablierten Konzept des zentralen Kommunikationsgateways und den Umstieg auf eine dezentrale Kommunikationsstruktur. Die deutlichsten Unschärfen bezüglich der letztendlichen Produkteigenschaften und Anwendungsfälle finden sich hierbei darin, mit welchen etablierten Bussystemen der RTE Backbone interagieren wird und ob und wie der RTE Backbone Teilbereiche der bestehenden Netzinfrastruktur oder die gesamte Netzinfrastruktur ersetzen wird.
Aus diesem Grund muss ein Lösungsansatz für die Migration hin zur Nutzung des RTE Backbone nicht nur geerbte Anforderungen an ein zentrales Kommunikationsgateway, sondern darüber hinaus ebenso allgemeine Anforderungen an effiziente Netzinfrastrukturen erfüllen. Aus der vorgenannten Unschärfe ergibt sich die Notwendigkeit, Anforderungen bezüglich einer generischen und flexiblen Unterstützung unterschiedlicher Topologievarianten auf Basis diverser Bustechnologien zu definieren.

4.2 Transparenz bezüglich bestehender CAN Anwendungen

Aufgrund des in 3.1 beschriebenen Entwicklungsprozesses unter Inanspruchnahme von externen Dienstleistern und aufgrund des mit 30% hohen Anteils der Entwicklungskosten an den Automobilherstellungskosten (siehe Mercer Management Consulting (2001)), muss im Rahmen dieser Arbeit sichergestellt sein, dass die entwickelte Lösung keine Änderung der Hardware oder Software notwendig macht, die bisher in den zu migrierenden Automobilnetzwerken verwendet werden.
In Bezug auf die Weiterverwendbarkeit der Software, speziell CAN-Anwendungen, ist sicherzustellen, dass die Nachrichteninhalte (PDU) bei Verwendung eines RTE Backbone unverändert an die existierenden Steuergeräte übermittelt werden. Die eigentliche Herausforderung liegt darin, die protokollspezifischen Metadaten, welche nicht Bestandteil einer CAN-Nachricht sind, korrekt zu übertragen. Die Korrektheit der Übertragung kann mit unterschiedlicher Schärfe definiert werden. Zum einen kann gefordert werden, dass es einem externen Betrachter nicht möglich ist, eine über einen RTE Backbone übermittelte Nachricht von ihrem herkömmlichen Äquivalent zu unterscheiden. Zum anderen kann gefordert werden, dass eine CAN-Anwendung nicht in der Lage ist, diese Unterscheidung vorzunehmen. Weiterhin kann die Anforderung der korrekten Übertragung von protokollspezifischen Metadaten und PDU auch dadurch als erfüllt gelten, dass die verarbeitende CAN-Anwendung trotz der Nutzung eines RTE-Gateways ein zur herkömmlichen Übertragung identisches Eingabe-/Ausgabe-Verhalten aufweist. Dieser letzte Fall unterscheidet sich genau dann von der Identität aus Sicht der CAN-Anwendung, wenn eine CAN-Anwendung nicht sämtliche zur Verfügung stehenden, protokollspezifischen Metadaten auswertet.
Im Rahmen dieser Arbeit gilt die Weiterverwendbarkeit von CAN-Anwendung dann als gegeben, wenn die Nutzung eines RTE Backbone das Eingabe-/Ausgabe-Verhalten nicht beeinflusst. Die Erfüllung dieser Anforderung kann auf unterschiedliche Arten und Weisen überprüft werden. Optimal wäre im industriellen Kontext die Anwendung sämtlicher ursprünglicher Testsuites, die das CAN-Anwendungsverhalten abdecken. Sobald derartige Testsuites nicht zur Verfügung stehen, kann eine genaue Kenntnis der CAN-Anwendung in Zusammenhang mit den RTE Backbone gestützten Eingabedaten dazu genutzt werden, das Ausgabeverhalten vorherzusagen. Um derartige Prognosen zu ermöglichen und zu erleichtern, wird im Rahmen dieser Arbeit dokumentiert, inwiefern die Nutzung eines RTE Backbones die Eingabedaten an eine CAN-Anwendung beeinflußt.

4.3 Schrittweise Migration, verteilter CAN Bus

Die in der Automobilindustrie etablierten (Feld-) Bustechnologien erlauben den angeschlossenen Steuergeräten, Aktoren und Sensoren einen gemeinsamen Zugriff auf das physikalische Medium. Aufgrund der gemeinsamen Zugriffsmöglichkeit besteht die Notwendigkeit, Methoden zum Ausschluß des gleichzeitigen Sendens zu verwenden. In Local Interconnect Network (LIN)-Bussystemen ist der gleichzeitige Zugriff dadurch ausgeschlossen, dass nur ein dediziertes Mastergerät eine Datenübertragung initiieren darf und der Master daher die Kollisionsfreiheit sicherstellen kann. In CAN-Bussystemen kommt das in 2.1.2 beschriebene CSMA/NDA-Verfahren zum Einsatz, welches neben der Kollisionsfreiheit zusätzlich auch die korrekte Priorisierung von Nachrichten sicherstellt. Das TDMA-Verfahren findet in zeitgesteuerter Kommunikation wie zum Beispiel bei FlexRay und RTE Anwendung und reserviert in einem sich wiederholenden Zeitplan spezifische Sendezeitpunkte für Steuergeräte, Nachrichtenklassen oder individuelle Nachrichten.
In automobilen Netzwerken werden zusätzlich sogenannte Kollisionsdomänen definiert, in denen ein physikalischer Bus Steuergeräte, Sensoren und Aktoren einer aufgrund unterschiedlicher Kriterien ausgewählter Gruppe zusammenfasst. Diese Kriterien umfassen

- Abgegrenztheit zu dritten physikalischen Bussen (Private CAN),
- sicherheitsbedingte Partitionierung (z.B. Powertrain CAN) und,
- funktionsabhängige Partitionierung (z.B. Chassis Bus, Multimedia Bus).

Aus dieser Definition der Kollisionsdomänen ergeben sich zwei für diese Arbeit relevante Konsequenzen. Zum einen ergibt sich aus der funktionsabhängigen Partitionierung aufgrund des systembedingt geringeren Kommunikationsaufkommens zwischen den Kollisionsdomänen die Möglichkeit, ein zentrales Kommunikationsgateway sinnvoll einzusetzen. Hier muss bei Verwendung eines RTE Backbone ersatzweise eine Architektur mit mehreren Gateways an den Übergabepunkten zwischen RTE Backbone und lokalen Bussystemen eingesetzt werden. Die zweite Konsequenz basiert auf dem Umstand, dass die Kollisionsdomänen nicht zwangsweise einer lokalen Häufung in der Positionierung der Steuergeräte im Automobil entsprechen, sondern dass eine Kollisionsdomäne sich durchaus über das gesamte Automobil erstrecken kann. Da ein parallel zum RTE Backbone verlegter Kabelstrang einer Kollisionsdomäne als konzeptionell und wirtschaftlich nachteilig anzusehen ist, und da nicht notwendigerweise eine vollständige Kollisionsdomäne Richtung Nutzung des RTE Backbone migriert wird, muss eine Möglichkeit geschaffen werden, Gruppen von mindestens einem Steuergerät aus der physikalischen Busleitung einer bestehenden Kollisionsdomäne zu entnehmen und über das RTE Backbone Netzwerk mit dem verbleibenden Rest der Kollisionsdomäne zu verbinden.

In Kombination mit der Anforderung der Transparenz bezüglich etablierter CAN-Anwendungen (siehe 4.2) entsteht die Anforderung, mehrere physikalische CAN-Teilbusse über ein RTE Backbone Netzwerk zu einem virtuellen CAN-Bus zusammenzufügen, welcher durch die Steuergeräteanwendungen nicht von einem physikalisch geschlossenem CAN-Bus unterscheidbar ist (im Folgenden: verteilter CAN-Bus). Diese Vorgehensweise erlaubt die sukzessive Migration einzelner Steuergeräte beliebiger Kollisionsdomänen hin zu RTE.
Bestimmte Eigenschaften eines Kommunikationsprotokolls und der Nachrichtenübertragung können allerdings darauf basieren, dass der Kommunikation ein geschlossener physikalischer Bus zu Grunde liegt. Insbesondere auf dem elektrischen Zustand der physikalischen Leitung basierende Statusvariablen - zum Beispiel link detection - als auch logische Konsequenzen aus den Medienzugriffsverfahren wie die temporale Ordnung von Nachrichten sind von einer Verteilung eines Bussystems betroffen.
Da es nicht auszuschließen ist, dass einzelne Eigenschaften eines Kommunikationsprotokolls und der Nachrichtenübertragung durch die Verteilung verletzt werden oder eine semantische Neudefinition erfahren, gilt es, die beteiligten Bussysteme hinsichtlich ihrer Verteilbarkeit zu untersuchen, die Auswirkungen der Verteilung festzustellen und gegebenenfalls die aus der Verteilung resultierenden Einschränkungen zu dokumentieren. Ziel dieser Vorgehensweise ist es, nicht nur einen positiven Anwendungsfall für eine Verteilbarkeit zu ermitteln, sondern eine detaillierte Bewertungsgrundlage zu schaffen, anhand derer die Relevanz der Auswirkungen einer Bus-Verteilung ersichtlich wird.

4.4 Konfigurierbarkeit

Die Implementierung im Rahmen dieser Ausarbeitung muss unterschiedlichen als auch anwendungsfallspezifischen Anforderungen genügen. Aus den anwendungsfallspezifischen Anforderungen ensteht der Bedarf an einer flexiblen Konfigurationsmöglichkeit der Implementierung, welche die spezifischen Anwendungsfälle generisch abdeckt. Solche spezifischen, vom Anwendungsfall abhängigen Anforderungen sind insbesondere in den Anforderungen an Bandbreiteneffizienz, Kollisionsfreiheit und flexiblen PDU-Transfer enthalten.

4.4.1 Bandbreiteneffizienz, Übertragungsmodus und Nachrichtenbündelung

Eine effiziente Nutzung der zur Verfügung stehenden RTE Backbone Bandbreite ist im wesentlichen vom implementierten Übertragungsmodus, einer geeigneten, selektiven Datenübertra-

gung und dem durch die Übertragung entstehenden Protokolloverhead abhängig. Diese drei Einflussfaktoren müssen entweder einmalig oder anwendungsfallabhängig optimiert werden. Der Übertragungsmodus legt fest, auf Basis welchen Verfahrens Nachrichten aus einem an den RTE Backbone angeschlossenen Bussystem (z.b. CAN-Bus) über den RTE Backbone an ein weiteres, an den RTE Backbone angeschlossenes Bussystem übertragen werden. Als Übertragungsmodi kommen eine bitweise Übertragung der Pegel der physikalischen Quellbusleitung oder eine nachrichtenbasierte Übertragung auf höheren OSI-Schichten in Frage.
Die bitweise Übertragung entsprechend der physikalischen Schicht wird in der Machbarkeit durch das Bit Timing des Quell- und Zielbusses in Verbindung mit dem RTE-Jitter begrenzt. Für CAN-Busse folgt unter der Vernachlässigung der uneinheitlichen Sample Points (Richards (2001), Philips Semiconductors (1997)) unterschiedlicher CAN-Steuergeräte, dass das Bit Timing nur durch die Baudrate des CAN-Bus bestimmt wird. Auf einer Übertragungsstrecke CAN1->RTE->CAN2 muss jedes Quellbit innerhalb des Bit Timings des Zielbusses übertragen werden. Daraus folgt, dass das CAN-Bit Timing von z.B. $1\mu s$ bei 1 $^{\text{MBaud}}/_{\text{s}}$ größer als der RTE-Jitter sein müsste. Diese Bedingung ist aufgrund des RTE-Jitters im Millisekundenbereich nicht erfüllbar. Ein hinreichend kleiner RTE-Jitter würde die bitweise Übertragung aufgrund der minimalen Framegröße einer RTE-Nachricht von 64 Byte und somit des Einfügens von über 45 padding bytes pro Quellbus-Bit ebenfalls nicht sinnvoll umsetzbar erscheinen lassen.
Bei einer nachrichtenorientierten Übertragung über den RTE Backbone erhält auf einer CAN1->RTE->Zielbus Strecke die maximale CAN-Nachrichtenfrequenz von rund 18 $^{\text{frames}}/_{\text{s}}$ bei 1 $^{\text{MBaud}}/_{\text{s}}$ Gewicht und würde zum Versand von rund 18 RTE-frames pro Sekunde mit minimaler Größe von 64 Byte führen. Dieser Ansatz ist der bitweisen Übertragung in Hinsicht auf die effiziente Bandbreitennutzung deutlich überlegen.
Die nachrichtenorientierte Übertragung kann wiederum in Abhängigkeit zum Anwendungsfall durch zwei Methoden sinnvoll optimiert werden, durch selektive Übertragung über den RTE Backbone und durch Verbesserung der Auslastung von RTE-PDU durch zusammengefasste Übertragung mehrerer Quellnachrichten in einem RTE-PDU. Aus der Anforderung, Nachrichten selektiv übertragen zu können, ergibt sich die Notwendigkeit, einen konfigurierbaren Nachrichtenfilter basierend auf konfigurierbaren Nachrichtenidentifikationskriterien zu implementieren. Die Anforderung, die Anzahl an Padding Bytes in RTE-PDU durch Nachrichtenbündelung zu verringern, muss über eine anwendungsfallabhängige Konfiguration realisiert werden, die sich neutral bezüglich der Zeitgarantien der zu übermittelnden Nachrichten verhält.

4.4.2 Nachrichtenzuordnung

Aus den Darstellungen unter 3.2 resultiert die Anforderung, in Kommunikationsgateways eine Zuordnung zwischen Nachrichten potentiell unterschiedlicher Protokolle anhand eines Identifikationskriteriums für Nachrichten vorzunehmen. Die Nachrichtenzuordnung übernimmt hierbei die Kollisionsauflösung bei mehrfacher Verwendung eines Identifikationskriteriums in unterschiedlichen Bussystemen und erlaubt die eindeutige Definition von Regeln für den PDU-Transfer zwischen den Bussystemen.
Die etablierten Entwicklungsmethoden in der Automobilindustrie auf Basis eines zentralen Kommunikationsgateways setzen auf ab der Planungsphase bekannte, über das gesamte Bordnetz eindeutige Nachrichtenidentifikationsmerkmale, zum Beispiel CAN-Nachrichten-IDs. Die frühe Festlegung von global eindeutigen CAN-Nachrichten-IDs ermöglicht die kollisionsfreie, hart kodierte Implementierung von PDU-Transfers zwischen verbundenen CAN-Bussen.
Die sukzessive Migration hin zur Verwendung eines RTE Backbone erschwert das harte Kodieren von PDU-Transferregeln. Unter Umständen führt sie neue Nachrichten ein, die Kollisionen der Nachrichtenidentifikationsmerkmale in Bezug auf das bestehende System verursachen und neue PDU-Transferregeln erfordern. Zumindest durch das RTE Backbone Netzwerk wird ein neues Bussystem mit spezifischen PDU-Transferregeln und Nachrichtenidentifikationsmerkmalen in das Bordnetz eingebracht. Zusätzlich ist es möglich, durch die Verteilung von CAN-Bussen oder das Hinzufügen von neuen Bussen mit dem etablierten Konzept der global eindeutigen Identifikationsmerkmale von Nachrichten zu brechen.
Daher muss ein Kommunikationsgateway im Rahmen dieser Arbeit dahingehend konfigurierbar sein, dass sämtliche Nachrichten auf dem RTE Backbone eindeutig bezüglich des Quell- und Zielbusses und des Nachrichtentypus identifizierbar sind. Für jede Kombination von Quellbus, Zielbus und Nachrichtentypus muss eine einzigartige Verarbeitungsvorschrift definiert werden können.

4.4.3 Flexibler, erweiterter PDU Transfer

Eine wesentliche Aufgabe des Kommunikationsgateways ist der PDU-Transfer zwischen eingehenden und ausgehenden Frames potentiell unterschiedlicher Protokolle. Dieser PDU-Transfer kann unterschiedliche Komplexitätsgrade aufweisen, wie z.B. den vollständigen Transfer aller PDU-Bytes oder den Transfer einzelner Signale einer PDU. Dieses Verhalten wird von den Gateway Requirements nach AUTOSAR Development Partnership (2008) gefordert und bildet den Grundstein der Anforderungen an das in dieser Arbeit implementierte Kommunikationsga-

teway. Darüber hinaus besteht ein Interesse daran, die ausgehenden PDU mit weiteren Daten anreichern zu können. Zu diesen Daten zählen

- protokollspezifische Daten ohne Eingangs-PDU-Bezug,
- vorverarbeitete Eingangs-PDU-Bestandteile und
- beliebige gateway-generierte Daten wie z.B. Zeitstempel.

Das Kommunikationsgateway muss bezüglich der unterschiedlichen Ausprägungen eines solchen, erweiterten PDU-Transfers für einzelne Nachrichten konfigurierbar sein.

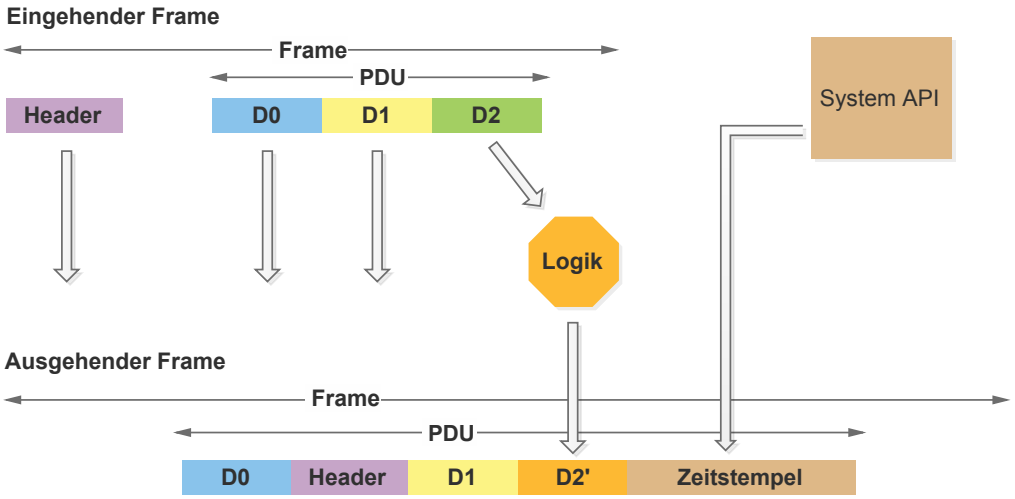

Abbildung 4.1: Varianten des PDU-Transfers aus unterschiedlichen Datenquellen.

4.4.4 Inhaltsabhängige Frameverarbeitung

Zusätzlich zu den für die selektive Datenübertragung durch das Kommunikationsgateway genutzten Nachrichtenidentifikationsmerkmalen soll eine vom Inhalt einer Nachricht abhängige Behandlung der Nachrichten konfigurierbar sein. Dies betrifft zum einen die inhaltsabhängige, selektive Übertragung von Nachrichten und zum anderen die inhaltsabhängige Anwendung eines Konfigurationssatzes betreffend des PDU-Transfer. Analog zu den Konfigurationsparametern des erweiterten PDU-Transfer sollen Konfigurationsparameter zur Identifikation einzelner Bestandteile eingehender Nachrichten existieren, um eine Auswertung auswählbarer Vergleichsoperationen und Vergleichswerte in Bezug auf die Nachrichtenbestandteile zu ermöglichen. Dieser Mechanismus ist außer zur inhaltsabhängigen Nachrichtenverarbeitung ebenfalls zur Prüfung und Durchsetzung von Wertebereichen ausgewählter Signale geeignet.

4.5 Zeitgarantien und Worst Case Response Time

Automobile Bussysteme sind teilweise als zeitkritische Systeme einzustufen. Während Latenzausreißer in der Übertragung eines Multimediainhaltes keine schwerwiegenden Folgen haben, sind die Anforderungen an das Echtzeitverhalten von zum Beispiel Powertrain-CAN deutlich höher. Die Berechnung der schärfsten einhaltbaren Zeitgarantien in ereignisgesteuerten Bussystemen wie dem CAN-Bus wird dadurch begünstigt, dass in automobilen Bussystemen üblicherweise zyklisch definierte Nachrichten versendet werden. Somit ist es möglich, in Kenntnis der zur Verfügung stehenden Bandbreite, der Nachrichtenlängen und der Nachrichtenfrequenzen die WCRT von komplexen Systemen aus tausenden zyklischen Nachrichten zu berechnen. Die Berechnung der CAN-WCRT nach dem im Jahr 1994 vorgestellten Modell nach Tindell und Burns (1994), Tindell u. a. (1994a) und Tindell u. a. (1994b) wurde ab 1995 durch die Volvo Car Corporation und Volcano Communications Technologies produktiv eingesetzt. Das Modell wurde in Davis u. a. (2007) dahingehend verbessert, dass sporadisch auftretende Interferenzen aus längeren Nachrichtensequenzen in die WCRT-Berechnung einbezogen werden und kann seitdem als hinreichend bestätigt gelten.

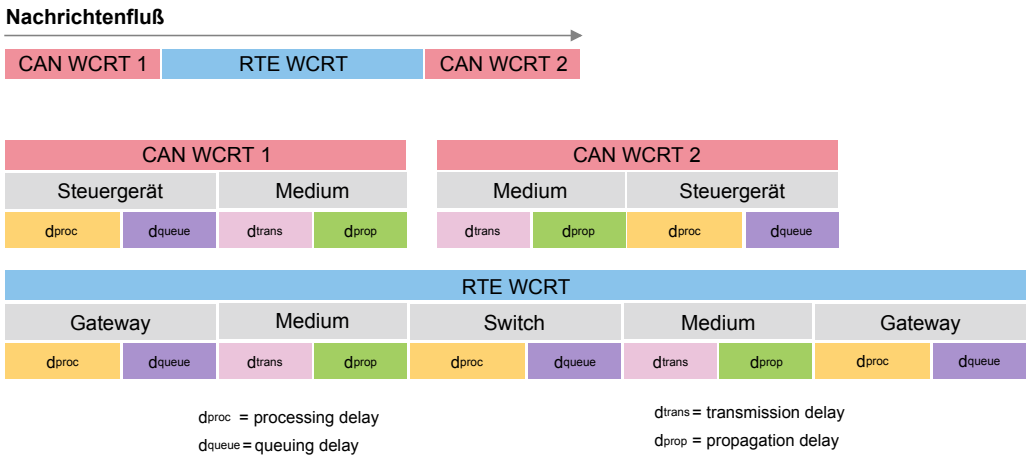

Abbildung 4.2: Bestandteile der WCRT einer Übertragung von CAN-Steuergerät zu CAN-Steuergerät über einen RTE Backbone.

Die für CAN-Nachrichten im automobilen Bordnetz definierten Zeitgarantien in Form von Deadlines für die längste, noch akzeptable Nachrichtenlatenz gehen entweder von der Übermittlung innerhalb eines geschlossenen CAN-Bus oder aber von der Übermittlung zwischen zwei per zentralem Kommunikationsgateway verbundenen CAN-Bussen aus. Die diesen Deadlines

zugrundeliegende WCRT-Berechnung verliert ihre Gültigkeit durch die zusätzliche Übertragung der Nachricht über einen RTE Backbone. Darüber hinaus führt ein verteilter CAN-Bus nicht nur eine zusätzliche Latenz durch Nutzung des RTE-Backbone ein, sondern ebenfalls eine zweite CAN-basierte WCRT aufgrund der physikalischen Aufteilung des verteilten CAN-Bus. So kann eine vormals für einen CAN-Bus berechnete WCRT bei Verwendung eines verteilten CAN-Bus durch die einzelnen Delays entsprechend Abbildung 4.2 dargestellt werden.
Die Berechenbarkeit der WCRT bleibt auch unter Verwendung eines RTE Backbone gegeben und kann als Akkumulation unterschiedlicher CAN- bzw. RTE-Anteile definiert werden. Das unter 4.2 beschriebene Kritierium der Transparenz bezüglich bestehender CAN-Anwendungen macht eine fallabhängige, erneute Berechnung der WCRT notwendig und kann gegebenenfalls zu einem Bruch der Zeitgarantien führen. Da die Berechnung der auf Infrastrukturkomponenten und Medium basierenden Anteile der RTE-WCRT keine neue Herausforderung darstellt, besteht primär ein Interesse daran, die Anteile der WCRT zu untersuchen, die durch das Kommunikationsgateway verursacht werden (processing delay, queuing delay, siehe Abbildung 4.2). Insbesondere das processing delay wird durch die Funkionalität des Gateways entsprechend der Anforderungen aus den Kapiteln 4.3, 4.4.3 und 4.4.4 massiv beeinflußt. Die Auswirkung der einzelnen Funktionalitäten sind entsprechend zu ermitteln und zu dokumentieren.
Ein weiterer Einflussfaktor auf die WCRT einer Nachricht findet sich in der unter Kapitel 4.4.1 geforderten Nachrichtenbündelung. Die gemeinsame Übertragung mehrerer Nachrichten auf dem RTE Backbone führt zwangsweise zu einer Verzögerung der zuerst im Kommunikationsgateway eingegangenen Nachrichten eines solchen Nachrichtenbündels. Die Auswirkung dieser Verzögerung kann entweder durch das exklusive Bündeln wenig zeitkritischer Nachrichten minimiert werden oder aber durch eine Untersuchung der konkreten Implementierung des Nachrichtenbündelungsalgorithmus fallabhängig untersucht werden.

5 Konzept

Im vorangegangenen Kapitel wurden die Anforderungen an die Vorgehensweise zur Anbindung etablierter Bussysteme an einen RTE Backbone sowie Anforderungen an deren Implementierung definiert. Dieses Kapitel stellt konzeptionelle Ansätze zur Lösung von Problemstellungen vor, die aus den Anforderungen und Vorgehensweisen erwachsen. Abschließend werden in diesem Kapitel die konzeptionellen Ansätze im Kontext der Softwarearchitektur des Kommunikationsgateways vorgestellt.

5.1 Verteilte CAN Bustopologie

Aus der Anforderung der Transparenz bezüglich etablierter CAN-Anwendungen (siehe 4.2) und der Anforderung, einzelne CAN-Steuergeräte beliebiger Kollisionsdomänen sukzessiv in Richtung RTE zu migrieren (siehe 4.3), ist die Erfordernis abgeleitet, mehrere physikalische CAN-Teilbusse über ein RTE Backbone Netzwerk zu einem virtuellen CAN-Bus zusammenzufügen, welcher durch die Steuergeräteanwendungen nicht von einem physikalisch geschlossenem CAN-Bus unterscheidbar ist (verteilter CAN-Bus).

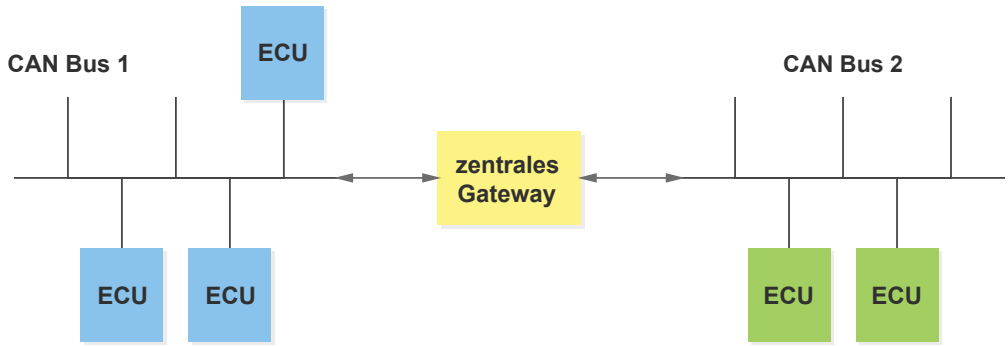

Abbildung 5.1: CAN Busse 1 und 2, verbunden über zentrales Gateway.

In einem verteilten CAN-Bus wird mit dem in der Automobilindustrie vorherrschenden Konzept des zentralen Kommunikationsgateways (siehe Abbildung 5.2) gebrochen.

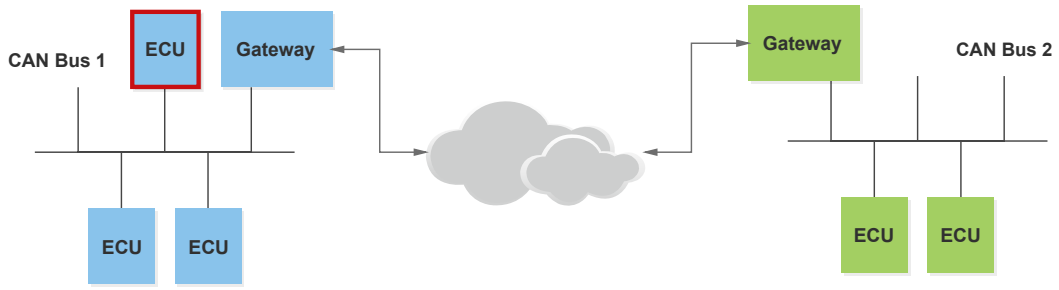

Abbildung 5.2: CAN Busse 1 und 2, verbunden über dezentrale Gateways und RTE Backbone. Die rot markierte ECU soll in einen separaten Teilbus verschoben werden.

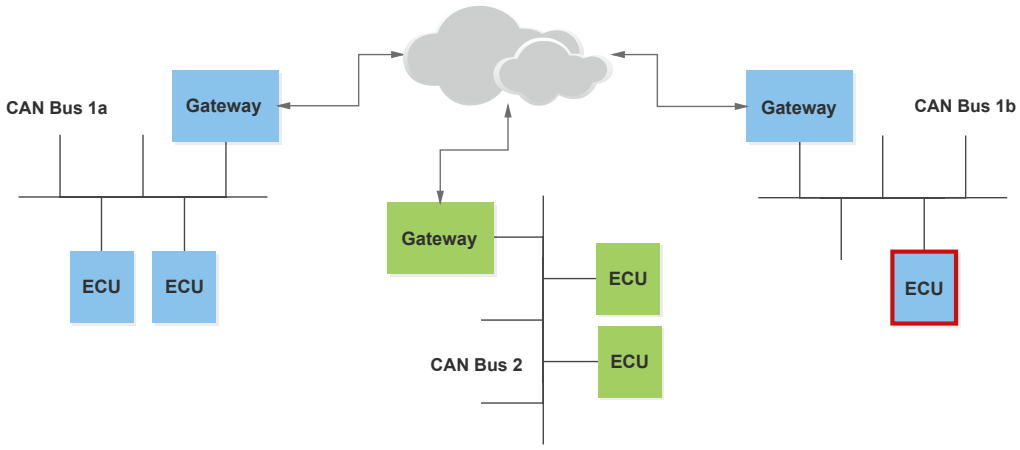

Abbildung 5.3: CAN Bus 1 verteilt in Teilbusse 1a, 1b, unverteilter CAN Bus 2, verbunden über RTE Backbone, rot markierte ECU entsprechend Abbildung 5.2.

Ein CAN-Bus besteht aus mindestens zwei CAN-Steuergeräten, den Busteilnehmern. Die Gateways in Abbildung 5.3 sind jeweils vollwertige Busteilnehmer des RTE Backbone, also auch der CAN-Busse. Im Zuge der Migration in Richtung RTE auf Basis eines verteilten CAN-Bus bedarf es also mindestens eines zu migrierenden Steuergeräts und eines Gateways, um einen funktionierenden CAN-Teilbus zu bilden.

5.2 Restriktionen verteilter CAN Bustopologien

Die Verteilung eines CAN-Bus ist nicht im CAN-International Organization for Standardization (ISO)-Standard (ISO 11898 1 (2003)) definiert, soll aber dennoch unter Verwendung von CAN-Steuergeräten umgesetzt werden, welche strikt entsprechend des CAN-ISO-Standards implementiert wurden und deren CAN-Anwendungen nicht für den Betrieb im verteilten CAN-Bus modifiziert worden sind. Es ist nicht sofort überschaubar, ob die mit der Verteilung einhergehenden Konsequenzen wie z.B. der physikalischen Trennung der CAN-Teilbusse frei von Auswirkungen auf die Definitionen des CAN-ISO-Standards sind. Ziel dieses Abschnittes ist es zu ermitteln, ob und unter welchen Einschränkungen sich beliebige CAN-Busse wie beschrieben verteilen lassen.
Um mögliche, aus der CAN-Verteilung entstehende Einschränkungen zu ermitteln, wird zunächst der Erfolgsfall für die Verteilung abstrakt definiert. Aus dieser Definition werden daraufhin sukzessiv in Verfeinerungen einzelne Abstraktionsebenen ermittelt, auf denen eine Untersuchung der erfolgreichen Verteilbarkeit eines CAN-Bus notwendig ist. Aus der Untersuchung entsprechend dieser Abstraktionsebenen kann eine Menge an Bedingungen für die erfolgreiche CAN-Verteilung gewonnen werden, im Umkehrschluss lassen sich daraus Einschränkungen der Verteilung ermitteln.
Eine CAN-Verteilung ist dann erfolgreich, wenn das Verhalten der CAN-Anwendungen im verteilten CAN-Bus identisch zum Verhalten im nicht-verteilten CAN-Bus ist. Die Validierung der Definition erfordert also die Betrachtung von CAN-Steuergeräten auf Anwendungsebene. In Unkenntnis der konkreten CAN-Anwendung wird ein Steuergerät exklusiv auf OSI-Schicht 7 (Anwendungsschicht) als Black Box betrachtet. Für die Black Box wird angenommen, dass sich ihr Ausgabeverhalten bei gleichbleibenden Eingangsdaten/Stimuli nicht verändert. Für diese Black Box-Sicht kann die Definition des Erfolgsfalls so verfeinert werden, dass im verteilten wie im nicht-verteilten CAN-Bus die Eingangsdaten der Anwendung/Black Box identisch sein müssen.
Es gibt zwei Arten von Eingangsdaten einer CAN-Anwendung: zum einen die Eingangsdaten, welche durch den CAN-Protokollstack unterhalb der Anwendungsschicht generiert werden und zum anderen die Eingangsdaten, welche den Systemzustand des CAN-Steuergerätes widerspiegeln. Die vom Protokollstack generierten Eingangsdaten der CAN-Anwendung enthalten üblicherweise Daten wie z.B. den CAN-Nachrichteninhalt und die Nachrichten-ID - nicht aber Daten, welche nur für die Funktionalität der OSI-Schichten unterhalb der Anwendungsschicht benötigt werden. Da der Systemzustand des CAN-Steuergerätes nicht detailliert vorhersagbar ist, fließt in die folgenden Betrachtungen nur die Größe „Zeit“ als Teil des Systemzustandes ein.

Zusammengefasst lässt sich die Definition des Erfolgsfalles der CAN-Verteilung somit insoweit verfeinern, als dass der CAN-Protokollstack unterhalb der Anwendungsschicht unabhängig von der Größe „Zeit“ im verteilten wie nicht-verteilten Einsatz identische Daten generiert und an die Anwendungsschicht weiterleitet.
Der Protokollstack unterhalb der Anwendungsschicht kann nur dann identische Daten für die Anwendungsschicht generieren, wenn die Eigenschaften und Mechanismen des CAN-Protokolls durch die Verteilung des CAN-Bus nicht verändert werden oder aber für die Verteilung des CAN-Bus irrelevant sind. Eigenschaften sind hierbei z.B. ein gesetztes Acknowledge-Bit in einem CAN-Frame. Ein Mechanismus ist z.B. der Arbitrierungsvorgang zur Medienzugriffssteuerung. Um die Eigenschaften und Mechanismen des CAN-Protokolls zu untersuchen, sind die Definitionen der sogenannten Service Primitives hilfreich. Die Service Primitives definieren dabei für jede OSI-Schicht des Protokollstacks die Funktionen und Daten, welche die Schicht den unmittelbar an sie anschließenden Schichten zur Verfügung stellt. Die Eigenschaften und Mechanismen des CAN-Protokolls werden durch das Zusammenspiel unterschiedlicher Service Primitives gebildet.
Somit sind drei unterschiedliche Abstraktionsebenen identifiziert, auf denen die Verteilbarkeit eines CAN-Bus untersucht werden muss: Die Black Box-Sicht auf Anwendungsebene, die Sicht mit Fokus auf die Eigenschaften und Mechanismen des CAN-Protokolls sowie die Sicht auf die Service Primitives des CAN-ISO-Standards. Diese Untersuchung erfordert, dass die wesentlichen Merkmale eines verteilten CAN-Bus im Gegensatz zu einem nicht-verteilten CAN-Bus bekannt sind. Diese Merkmale bestehen in der physikalischen Trennung der verteilten CAN-Teilbusse sowie der Relevanz der „Zeit“ als Systemzustand für den verteilten CAN-Bus.
In den folgenden Abschnitten wird zuerst die Relevanz der Zeit und der physikalischen Bustrennung für die Untersuchung der Verteilbarkeit dargestellt. Darauf aufbauend werden Service Primitives, Protokolleigenschaften und Anwendungsverhalten untersucht. Abschließend werden die Restriktionen bezüglich der Verteilbarkeit auf eine Prüfliste reduziert, welche schnelle Bewertung der Verteilbarkeit eines CAN-Bus zulässt.

5.2.1 Zeitverhalten

Die Zeit wirkt sich auf eine verteilte CAN-Infrastruktur an zwei wesentlichen Punkten aus. Zum einen wird eine zusätzliche Übertragungslatenz in den verteilten CAN-Bus eingebracht, entsprechend der Zeit, die eine CAN-Nachricht im RTE Backbone und in den Gateways verbringt. Zum anderen wird die Zeit indirekt in Form der globalen, zeitlichen Ordnung der Nachrichten für eine verteilte CAN-Infrastruktur relevant.

Im Rahmen der Verteilbarkeitsuntersuchung muss der Übertragungslatenz für jede Protokolleigenschaft oder jedes angestrebte Anwendungsverhalten einzeln Rechnung getragen werden. In einem herkömmlichen CAN-Bus kann eine globale, zeitliche Nachrichtenordnung angenommen werden. Diese Annahme liegt in der Shared Medium Charakteristik des CAN-Bus begründet: Zu jedem Zeitpunkt kann maximal eine Nachricht auf dem physikalischen CAN-Bus liegen und diese Nachricht kann von allen Empfängern umgehend ausgewertet werden. In einem verteilten CAN-Bus kann hingegen zu jedem Zeitpunkt pro Teilbus je eine, mit den übrigen konkurrierende, Nachricht existieren. Die zeitliche Ordnung der Nachrichten ist somit aus Sicht einzelner Controller nicht mehr gewährleistet.

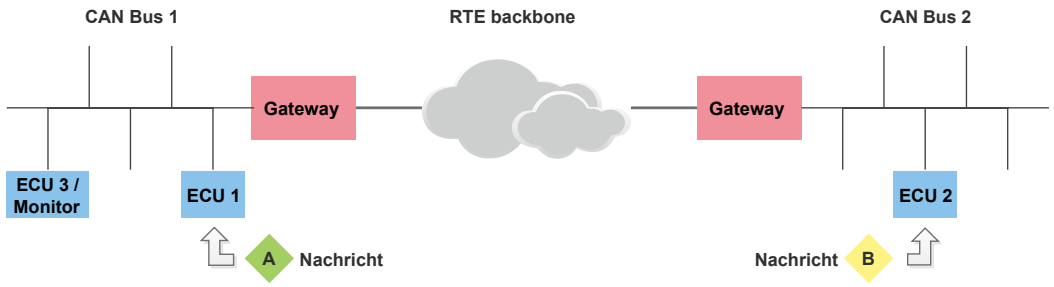

Abbildung 5.4: ECU 1 und ECU 2 senden Nachrichten an ECU 3, Nachricht B von ECU 2 unterliegt der Latenz des RTE Backbones und des Gateways.

Abbildung 5.4 zeigt einen abstrahierten, verteilten CAN-Bus mit 2 Teilbussen und einen bzw. zwei Controllern pro Teilbus. Die Betrachtung der zeitlichen Ordnung findet aus Sicht des Monitor-Controllers „ECU 3“ statt. Wenn der Controller „ECU 2“ eine Nachricht „B“ an die im fremden Teilbus existierende „ECU 3“ sendet, ist diese für „ECU 3“ erst dann sichtbar, sobald das RTE Backbone und zwei Gateways von der Nachricht durchlaufen wurden und diese auf den Teilbus der „ECU 3“ übertragen wurde.
Wenn eine spätere Nachricht „A“ von „ECU 1“ an „ECU 3“ gesendet wird, kann diese umgehend von „ECU 3“ ausgewertet werden. Somit ist die globale, zeitliche Ordnung nicht gewährleistet, wenn Nachricht „A“ gesendet wird, solange Nachricht „B“ noch nicht im Teilbus der „ECU 3“ bekannt ist (siehe Abbildung 5.5).

5.2.2 Auswirkung von physikalischer Trennung und Übertragungsmodus

Durch die Verteilung des CAN-Bus wird die physikalische Busleitung getrennt und die Lücke durch Gateways und RTE Backbone überbrückt. Die Gateways übertragen dazu nur die CAN-Nachrichten unter Auslassung aller für Nachrichten irrelevanten, auf dem Bus übertragenen

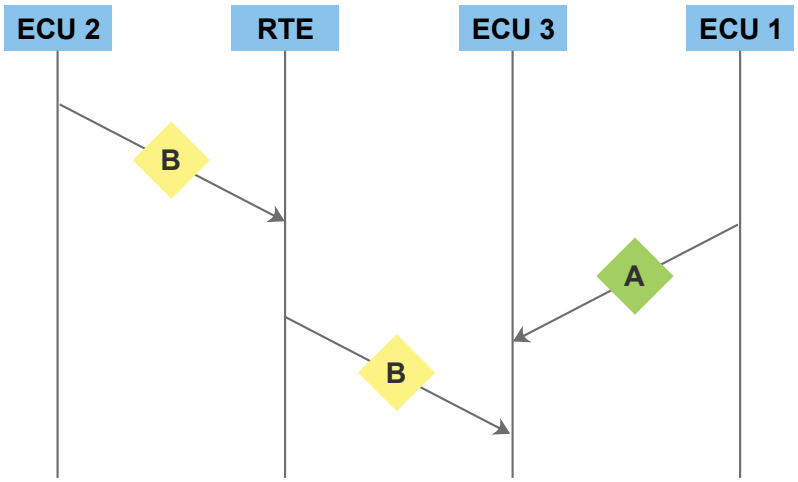

Abbildung 5.5: Bruch der zeitlichen Ordnung der Nachrichten A und B aus Sicht der ECU 3 analog zu Abbildung 5.4.

Informationen. Die physikalische Trennung und dieser selektive Übertragungsmodus haben unterschiedliche Auswirkungen.
Das CAN-Protokoll macht sich eine nicht getrennte Busleitung zu Nutze, indem z.B. ein CAN-Steuergerät in die laufende Übertragung eines anderen Steuergeräts hineinschreiben kann, um eine Empfangsbestätigung anzuzeigen. Dies ist bei einer physikalischen Trennung nicht möglich. Darüber hinaus folgt aus dem Übertragungsmodus, dass theoretisch einzelne Eigenschaften und Mechanismen des CAN-Protokolls nicht teilbusübergreifend weiterverwendet werden können, da die Eigenschaften und Mechanismen auf Informationen angewiesen sind, die nicht zwischen den Teilbussen übertragenen.
Um die Verteilbarkeit eines CAN-Bus zu bewerten, müssen also die Eigenschaften und Mechanismen des CAN-Protokolls darauf untersucht werden, ob sie anfällig für die genannten Auswirkungen der physikalischen Trennung und des Übertragungsmodus sind.

5.2.3 Protokolleigenschaften

Voraussetzung für die transparente Verteilung des CAN-Bus ist, dass die Verteilung keine seiner Protokolleigenschaften verändert. Protokolleigenschaften umfassen in dieser Untersuchung Mechanismen, Eigenschaften und die Service Primitives der einzelnen OSI-Schichten nach CAN-ISO-Standard (ISO 11898 1 (2003)).
Die Verteilbarkeit der Protokolleigenschaften wird daran gemessen,

- ob sich die Service Primitives bei verteilter und nicht-verteilter Kommunikation identisch verhalten,
- ob der Aussagegehalt der Service Primitives für die Anwendungslogik unverändert bleibt und
- ob alle Protokoll-Features (ACK, Data Frame, etc.) in verteilter Kommunikation verwendbar sind,
- ob die Unterbrechung der physikalischen Schicht CAN-Anwendungen nicht stört.

5.2.3.1 Service Primitives

Service Primitives definieren die Funktionen und Dienste, die eine OSI-Schicht den benachbarten OSI-Schichten anbietet und sind wesentlicher Bestandteil des CAN-ISO-Standards. Die Untersuchung der Verträglichkeit der Service Primitives mit einem verteilten CAN-Bus gibt Aufschluss über die Kompatibilität abstrakterer Protokoll-Mechanismen (Arbitrierung, Acknowledge, etc.) zum verteilten CAN-Bus.
Im CAN-ISO-Standard wird in OSI-Schichten zwischen drei Typen von Service Primitives unterschieden, „request“-, „confirm“- und „indication“-Service Primitives. Für jede OSI-Schicht gilt dabei: „request“-Service Primitives sind Dienste, die eine tiefer liegende Schicht anfordern kann, „confirm“-Service Primitives bestätigen vorangegangene „request“-Service Primitive-Nutzungen und „indication“ Service Primitives zeigen höher liegenden Schichten einen Dateneingang an.
Die korrekte Funktion von Service Primitives setzt ausschließlich voraus, dass auf der physikalischen Busleitung Pegel entsprechend des CAN-Protokolls vorliegen. Somit ist sichergestellt, dass auch in einem verteilten CAN-Bus die durch die Service Primitives einer OSI-Schicht bereitgestellten Funktionen und Daten von den umliegenden OSI-Schichten genutzt werden können. Der einzige Problemfall ergibt sich aus den inhaltlichen Rückschlüssen, die per CAN-ISO-Standard aus der Auslösung bestimmter Service Primitives gezogen werden können. So beschreibt der „confirm“ Typus Erfolg, Mißerfolg oder teilweisen Erfolg der vorherigen Nutzung einer „request“ Service Primitive, genauer: Nach ISO 11898 1 (2003) bestätigt „confirm“ teilweise, aber nicht notwendigerweise, eine Aktion der entsprechenden OSI-RM-Schicht des entfernten Busteilnehmers. Dieser Aussagegehalt bezüglich eines entfernten Busteilnehmers innerhalb einer Service Primitive von Typ „confirm“ ist in einem verteilten CAN-Bus kritisch. Insgesamt zwei „confirm“ Service Primitives („MAC Data confirm“ und „MAC Remote confirm“) machen derartige Aussagen und müssen genauer untersucht werden.
Die „MAC Data confirm“ Service Primitive bestätigt, dass der Empfänger eine „MAC Data indication“ in seiner MAC-Schicht als Folge eines vorher aus einem Data Frame gelesenen

„MAC Data request“ ausgelöst hat. Dieser Mechanismus zeigt einem Sender an, dass der Empfänger die Nachricht erhalten hat und das ACK-Bit in der Nachricht überschrieben hat. Analog dazu bestätigt die „MAC Remote confirm“ Service Primitive dasselbe für Remote Frames. Unter passenden Umständen (selektive Bestätigung oder nur 2 Busteilnehmer) kann somit „MAC Data confirm“ ebenfalls als Identifikationskriterium für einen Empfänger genutzt werden. In einem verteilten CAN-Bus ist der vermeintliche Empfänger aber potentiell tatsächlich das Gateway in Richtung des RTE Backbone. Das Identifikationskritierium ist somit ungültig.
Die Nutzung von Protokoll-Features, über die eine CAN-Anwendung Zugriff auf dieses Identifikationskriterium erhalten könnte, ist in die Prüfliste zur Verteilbarkeit eines CAN-Bus aufzunehmen.

5.2.3.2 Arbitrierung

Während der in Abbildung 2.2 dargestellten Arbitrierung werden die Nachrichten-IDs von sendewilligen CAN-Controllern dominant (0) bzw. rezessiv (1) ausgetaktet. Gleichzeitig wird der Pegel der Busleitung überprüft um zu festzustellen, ob Dritte die rezessiv gesendeten ID-Bits überschrieben haben. Wenn die gesendete Nachrichten-ID der gelesenen Nachrichten-ID entspricht, darf die eigentliche Nachricht gesendet werden. Dieser Mechanismus kann aufgrund der unterbrochenen physikalischen Busverbindung in einem verteilten CAN-Bus nicht genutzt werden (Schreiben in laufende Übertragung einer Nachricht). Allerdings ist die Arbitrierung nur für Medienzugriffskontrolle und Nachrichtenpriorisierung verantwortlich. Daher wird die Arbitrierung im Sinne der CAN-Verteilung irrelevant, da es zwischen zwei verteilten CAN-Teilbussen keine Notwendigkeit einer gemeinsamen Medienzugriffskontrolle gibt und das Gateway die Nachrichtenpriorisierung in jede Übertragungsrichtung intakt halten kann.

5.2.3.3 Acknowledgement

Der CAN-ISO-Standard (ISO 11898 1 (2003)) definiert auf der MAC-Schicht Empfangsbestätigungen für Data Frames und Remote Frames. Dies ist dadurch realisiert, dass während der Nachrichtenübertragung positive Empfangsbestätigungen als dominantes Bit in ein rezessiv gesendetes Feld der Nachricht geschrieben werden. Dieser Mechanismus kann aufgrund der unterbrochenen physikalischen Busverbindung in einem verteilten CAN-Bus nicht genutzt werden (Schreiben in laufende Übertragung einer Nachricht). Da allerdings das Gateway als regulärer CAN-Controller stellvertretend für den durch die physikalische Trennung verdeckten Empfänger seinerseits die Empfangsbestätigungen schreibt, wird dieses Protokoll-Feature im Sinne der CAN-Spezifikation nicht verletzt. Es verbleiben die in Abschnitt 5.2.3.1 diskutierten

Konsequenzen aus den Service Primitives für die Nutzung der Empfangsbestätigungen als Identifikationskritierium.

5.2.3.4 Data Frames

Data Frames werden von CAN-Controllern genutzt, um Nachrichten zu versenden. Diese Nachrichten werden in einem verteilten CAN-Bus von den Gateways auf Anwendungsebene abgefangen und transparent über den RTE Backbone propagiert. Da während der Übertragung der Data Frames die Protokoll-Features „Arbitrierung“ und„Acknowledgement“ genutzt werden, unterliegen die Empfangsbestätigungen in Data Frames auch deren Nutzungbeschränkungen. Konkret dürfen CAN-Anwendungen aus der Data Frame Empfangsbestätigung nicht auf die Empfängeridentität schließen (siehe 5.2.3.1, „MAC Data confirm“).

5.2.3.5 Remote Frames

Remote Frames sind spezifiziert, damit CAN-Anwendungen andere CAN-Anwendungen zur Übermittlung einer Nachricht auffordern können. Dazu wird ein Remote Frame mit der gewünschten Nachrichten-ID ohne Nutzlast gesendet, woraufhin der zuständige Empfänger mit der gewünschten Nachricht zu antworten hat. Analog zum Data Frame ist der Remote Frame aufgrund der Nutzung identischer Protokoll-Features ebenfalls den Einschränkungen des Data Frame unterworfen. Diese Einschränkungen werden faktisch dadurch relativiert, dass die Verwendung von Remote Frames in Automobilanwendungen vermieden wird. Es müssen allerdings immer potentiell anwendungsseitig implementierte Antwortzeitgarantien mit Blick auf die Kompatibilität zu den RTE Backbone Übertragungslatenzen und Gatewaylaufzeiten überprüft und gegebenenfalls angepaßt werden.

5.2.3.6 Overload Frames

Entsprechend der CAN-Spezifikation können die Logical Link Control (LLC)- und MAC-Schichten der CAN-Conroller das Versenden von Overload Frames auslösen um eine Überlastsituation des Controllers aufzulösen. Die Overload Frames werden in den Sendepausen zwischen CAN-Nachrichten gesendet und erweitern diese somit. Dadurch wird die CAN-Nachrichtenfrequenz verringert, um den Sender der Overload Frames zu entlasten. Die Umsetzung dieses Mechanismus erfordert das genau abgestimmte Versenden des Overload Frames in der Sendepause zwischen CAN-Nachrichten (Interframe Space) und kann somit aufgrund der unterbrochenen physikalischen Busverbindung in einem verteilten CAN-Bus nicht genutzt werden (Schreiben in laufende Übertragung einer Nachricht). Der Effekt dieser Einschrän-

kung wird dadurch relativiert, dass Gateways mit wohldimensionierten Nachrichtenpuffern eine vorübergehende Reduzierung der Nachrichtenfrequenz auf einem CAN-Bus problemlos kompensieren können.

5.2.3.7 Error Frames

Sobald ein beliebiger CAN-Controller einen Busfehler entdeckt, generiert er einen Error Frame und sendet ihn in die laufende, fehlerhafte Übertragung. Sobald der Busfehler nicht mehr vorliegt, wird versucht, die fehlgeschlagene Kommunikation zu wiederholen. Dieser Mechanismus kann aufgrund der unterbrochenen physikalischen Busverbindung in einem verteilten CAN-Bus nicht genutzt werden. Da allerdings in einem verteilen CAN-Bus die Busfehler der Teilbusse keine direkte Auswirkung aufeinander haben, kann diese Restriktion als irrelevant im Sinne der CAN-Verteilung angesehen werden.

5.2.4 Anwendungsverhalten

Die Verteilbarkeit eines CAN-Bus wird in zwei Aspekten durch die CAN-Anwendungen berührt. Zum einen nutzt die CAN-Anwendung Protokolleigenschaften, welche potentiell in einem verteilten CAN nicht mehr nutzbar sein könnten. Darüber hinaus bedienen sich Anwendungen klassischer Entwurfsmuster, welche durch die CAN-Verteilung invalidiert werden können. Die abgeschlossene Untersuchung der Protokolleigenschaften (Abschnitt 5.2.3) muss daher um eine Untersuchung gängiger Entwurfsmuster erweitert werden. Derartige Entwurfsmuster können vom Entwickler explizit durch eigene Implementierung oder implizit durch Nutzung der API eines „Application Layer Framework“ verwendet werden. Somit erfordert die Überprüfung des Anwendungsverhaltens detaillierte Kenntnis des anwendungsfallabhängigen Software-Stacks.

5.2.4.1 Zeitliche Synchronisierung

Die zeitliche Synchronisierung von CAN-Anwendungen basiert üblicherweise auf durch Referenznachrichten ausgelöster Uhrensynchronisation. Die Synchronisation auf Referenznachrichten kann nur dann eine stabile Zeitsynchronisation zur Folge haben, wenn die Übertragungslatenzen der Referenznachrichten ermittelbar und konstant sind. In einem verteilten CAN-Bus würde die Übermittlung von Referenznachrichten über den RTE Backbone die Übertragungslatenz entsprechend der Latenz und des Jitters des RTE Backbone verändern und somit die Synchronisationsgenauigkeit drastisch verringern. Zeitsynchronisationsverfahren, die auf dem Austausch von Referenznachrichten basieren - einschließlich TTCAN (ISO 11898 4

(2004)) - dürfen somit nicht über einen RTE Backbone verteilt werden. Relativierend gilt, dass TTCAN in automobilen Anwendungen üblicherweise nicht zum Einsatz kommt.

5.2.4.2 Timed Acknowledgement

„Timed Acknowledgement“ wird als Verfahren eingesetzt, um erfolgreiche Übertragungen, Nachrichtenverlust oder Übertragungslatenzen zu detektieren. Dabei ist der Empfänger einer Nachricht dazu verpflichtet, dem Sender den Erhalt der Nachricht binnen einer vordefinierten Zeitspanne zu bestätigen. Sobald der Sender innerhalb der vereinbarten Zeitspanne keine Empfangsbestätigung erhalten hat, stuft er die Originalnachricht als verloren ein und behandelt diesen Fehler auf Anwendungsebene. Der RTE Backbone kann Übertragungslatenzen einführen, die die Einhaltung der Zeitlimits für erfolgreiche Empfangsbestätigungen verhindern. Daher müssen die Zeitlimits für erfolgreiche Empfangsbestätigungen in einem verteilten CAN in Abhängigkeit von den RTE Zeitgarantien überprüft und gegebenenfalls angepasst werden.

5.2.4.3 Prüfsummen

Prüfsummen werden für die Erkennung und Korrektur von Übertragungsfehlern genutzt. Das CAN-Protokoll definiert ein CRC-Feld, welches eine von der CAN-Verteilung nicht betroffene Prüfsumme enthält. Diese Prüfsumme wird von den Gateways als Stellvertreter für den nicht im selben Teilbus existierenden Empfänger ebenfalls berechnet und ausgewertet. Auf Anwendungsebene erstellte und ausgewertete Prüfsummen über Bestandteile der Nachrichten können ebenfalls in einem verteilten CAN-Bus genutzt werden. Hierzu ist allerdings erforderlich, dass das Gateway neu berechnete Prüfsummen einfügen kann, um Manipulationen an den übertragenen Nachrichten (z.B. Endianess Wandlung) zu validieren. Darüber hinaus müssen ebenso Mechanismen überprüft werden, welche anwendungsseitig Prüfsummenfehler behandeln und z.B. eine erneute Übertragung der fehlerhaften Nachricht auslösen. Solange die Prüfsummenfehlerbenachrichtigungen auf dem Versenden regulärer CAN-Nachrichten beruhen, können diese ebenfalls von den Gateways weitergeleitet werden und somit vom Empfänger der Prüfsummenfehlerbenachrichtigung behandelt werden.

5.2.4.4 Sequenznummern

Sequenznummern sind ein Mechanismus, der in paketbasierter Kommunikation eingesetzt wird, um Nachrichtenordnung, Nachrichtenwiederholungen und Nachrichtenverlust in der Kommunikation zweier Teilnehmer zu erkennen. Sequenznummern entsprechen der zeitlichen Ordnung der Nachrichten aus Sicht des Senders. Somit können Kommunikationsteilnehmer ihre

Sende-/Empfangsstatistik vergleichen. Da Sequenznummern stark auf einer intakten zeitlichen Ordnung der Nachrichten basiert, können Sequenznummern entsprechend der Konsequenzen aus Abschnitt 5.2.1 nur innerhalb eines Teilbusses verwendet werden.

5.2.5 Prüfliste

Die hier präsentierte Liste erleichtert die Überprüfung der transparenten Verteilbarkeit eines bestehenden CAN-Bus und der aktuellen CAN-Anwendungen. Die tabellarische Auflistung fasst das Ergebnis der Untersuchung der Restriktionen aus der Verteilung eines CAN-Bus des Abschnitt 5.2 zusammen. Die Untersuchung basiert auf der Analyse der CAN-Verteilung auf den Abstraktionsebenen der Eigenschaften und Mechanismen des CAN-Protokolls, der Service Primitives, sowie gängiger Anwendungsentwurfsmuster. Die einzelnen Abstraktionsebenen wurden aus einer schrittweisen Verfeinerung der Definition einer erfolgreichen CAN-Verteilung gewonnen und berücksichtigen alle Aspekte eines CAN-Protokollstacks bis einschließlich der Anwendungsschicht. Die unterschiedlichen Merkmale dieser Abstraktionsebenen wurden hinsichtlich ihrer syntaktischen und semantischen Kompatibilität zum verteilten CAN-Bus untersucht. Alle untersuchten Merkmale sind im Folgenden zusammen mit Indikatoren ihrer Verwendbarkeit innerhalb eines Teilbusses oder zwischen verteilten Teilbussen aufgelistet.

Tabelle 5.1: Restriktionen der CAN-Verteilung

Feature	**lokaler Teil**	**verteilter Teil**
Arbitrierung	OK	NA
Acknowledge	OK	OK / ACK Semantik[1]
Data Frame	OK	OK / ACK Semantik[1]
Overload Frame	OK	OK / ACK Semantik[1]
Remote Frame	OK	OK / Probleme möglich[1]
Error Frame	OK	NA
Zeitsynchronisation	OK	NA
Timed Acknowledge	OK	Probleme möglich[1]
Prüfsummen	OK	Probleme möglich[1]
Sequenznummern	OK	Probleme möglich[1]
Zeitliche Nachrichtenordnung	OK	NA

[1] Probleme werden zusammen mit den Features detailliert dargestellt.

5.3 Interface Description Language

Eine wichtige Aufgabe des im Rahmen dieser Arbeit entwickelten Kommunikationsgateways ist die Übersetzung von Nachrichten zwischen unterschiedlichen Protokollen, wie z.B. CAN und RTE. In Abschnitt 4.4.3 wurde die Anforderung vorgestellt, dass jede Nachricht auf dem RTE Backbone eindeutig bezüglich Quell- und Zielbus identifizierbar sein muss und dass für jede Kombination aus Quellbus, Zielbus und Nachricht eine einzigartige Transformationsvorschrift zur Protokollübersetzung zwischen Quell- und Zielbus definiert werden kann.
Aus der Anforderung sind zwei Herausforderungen ableitbar. Aus Sicht der Softwareentwicklung für das Gateway muss die Protokollübersetzung durch einen frei konfigurierbaren Algorithmus implementiert werden. Aus Sicht des Entwicklungsprozesses muss die Konfiguration der Gatewayanwendung toolgestützt von beliebigen Projektbeteiligten erzeugt werden können.
Die toolgestützte Generierung einer Anwendungskonfiguration erfordert, dass die Konfigurationsparameter in einer bekannten Notation digital vorliegen, um dann von einem Compiler entsprechend der Grammatik in die Anwendungskonfiguration umgewandelt zu werden. Durch diese Vorgehensweise kann eine wenigen Shareholdern bekannte, von Programmiersprache und Hardwareplattform abhängige, Anwendungskonfiguration von ihrer allgemeinverständlichen, einem Industriestandard folgenden, Notation entkoppelt werden.
Obwohl geeignete Kandidaten eines Notationsstandards wie z.B. das Fieldbus Exchange Format (FIBEX) existieren, kann an dieser Stelle auf die Festlegung auf einen Notationsstandard aufgrund der unterschiedlichen Präferenzen der Automobilhersteller verzichtet werden. Diese Arbeit legt den Schwerpunkt auf die Entwicklung von Transformationsvorschriften und deren Grammatik zur Beschreibung von Protokollübersetzungen sowie deren Implementierung in der Gatewayanwendung. Durch die Grammatik müssen Nachrichtenelemente unterschiedlicher Protokolle ebenso wie Operationen auf eingehende und ausgehende Daten beschrieben werden.
Grammatikgestützte Beschreibungssprachen sind in den Computerwissenschaften hinlänglich bekannt. So existieren mit Konzepten wie IDL und Implementierung wie der Abstract Syntax Notation One (ASN.1) Information Object Set (IOS) Beschreibungssprachen für Objekte und Operationen. In ASN.1 wird die Grammatik in Information Object Class (IOC) und Operationen in Information Object (IO) definiert. Generell werden zur Beschreibung von IDL-Grammatiken häufig Regeln in Backus-Naur Form (BNF) verwendet.
Im Folgenden werden die Objekte und Operationen der IDL-Grammatik hergeleitet und die Grammatik entwickelt, formal definiert und in Augmented Backus-Naur Form (ABNF) (Crocker und Overell (1997)) dargestellt.

5.3.1 Herleitung der Grammatik

Die zu entwickelnde Grammatik der IDL muss dazu geeignet sein, alle für eine Protokollübersetzung notwendigen Informationen auszudrücken. Notwendig sind zur Übersetzung zwischen Protokollen zumindest Informationen bezüglich der Syntax/Struktur der eingehenden und ausgegebenen Nachricht und zusätzlich alle Daten, mit denen die Operationen des Übersetzungsalgorithmus parametrisiert werden.
Es sind zwei unterschiedliche Ansätze bezüglich der Formulierung des Beschreibungsumfanges der IDL denkbar. Die Ansätze unterscheiden sich darin, ob die Übersetzungsoperationen (z.B. copy) in der Grammatik explizit ausgedrückt werden müssen oder implizit in der Beschreibung der eingehenden und ausgehenden Daten enthalten sind. Explizit enthaltene (Sequenzen von) Operationen würden zwar eine genauere Kontrolle des Übersetzungsalgorithmus erlauben, brächten aber zusätzliche unerwünschte Komplexität und Dynamik in die IDL und den Übersetzungsalgorithmus ein.
In dieser Arbeit wird der Ansatz umgesetzt, zwei Nachrichtendarstellungen und implizierte Übersetzungsoperationen zu beschreiben. Die erste Darstellung beschreibt die eingehende Nachricht mit allen relevanten Merkmalen, die zweite Darstellung definiert den Soll-Zustand, die ausgehende Nachricht, ebenfalls mit allen relevanten Merkmalen. Die implizierten Operationen werden aus der Struktur und Ordnung der Nachrichtendarstellungen geschlossen.
Die Entwicklung der Grammatik wird im Folgenden analog zu diesem Ansatz wie folgt unterteilt:

- Formale Definition von Eingabe, Ausgabe und Übersetzung,
- Entwicklung der grundlegenden Elemente einer Nachrichtendarstellung,
- Entwicklung der Operationen und Nachrichtendarstellungen.

5.3.2 Eingabe, Ausgabe und Transformationsvorschrift

Eine anhand der Grammatik ausgedrückte Regel zur Protokollübersetzung (im Folgenden: „Transformationsvorschrift“) wird auf Daten angewendet, deren Inhalt und Struktur vom Systemkontext abhängig ist. Daher muss nicht nur formal definiert werden, was eine gültige Anwendung einer Transformationsvorschrift ausmacht. Es muss für den Entwurf der Grammatik ebenso untersucht werden, auf welche Daten der Übersetzungsalgorithmus Zugriff hat und was für Daten zu generieren sind.
Im Folgenden werden daher die Eingabe- und Ausgabedaten im Systemkontext betrachtet, um darauf aufbauend den Vorgang der Anwendung von Transformationsvorschriften (im Folgenden: „Transformation“) formal zu definieren.

5.3.2.1 Eingabe- und Ausgabedaten

Eine Transformationsvorschrift wird im Vorgang der Transformation auf die Eingabedaten angewendet und soll dadurch spezifische Ausgabedaten produzieren. Aus der Praxis kann geschlossen werden, dass eine Transformation unter bestimmten Voraussetzungen aus Eingabedaten keine Ausgabedaten produzieren soll - das ist z.B. dann der Fall, wenn aufgrund eines Fehlers in den Eingabedaten keine sinnvolle Ausgabe möglich ist. Somit werden Eingabe- und Ausgabedaten wie folgt definiert:

Definition 5.3.1: Daten

$\mathbb{E}$ = Menge der eingehenden Daten,
$\mathbb{A} = \{a_+\} \cup \{\bot\}$ = Menge der ausgehenden Daten mit
a_+ = ausgehende Daten,
$\bot$= ausgehenden Daten im Fehlerfall

In Abhängigkeit vom Systemkontext sind eingehende und ausgehende Daten unterschiedlich ausgeprägt. Aus Sichtweise der OSI-Schichten stehen z.B. an Eingabedaten natürlich nur die Daten zur Verfügung, welche für die OSI-Schicht sichtbar sind, auf der die Transformation implementiert ist. Umgekehrt muss eine Transformation Ausgabedaten produzieren, welche auf der OSI-Schicht sinnvoll weiterverarbeitet werden können. Das in dieser Arbeit vorgestellte Gateway ist ein Gateway auf Anwendungsschicht (OSI-Schicht 7). Daraus kann geschlossen werden, dass der Transformation eine Datenstruktur als Eingabedaten zur Verfügung gestellt werden kann, die eingehende Nachrichten beschreibt und neben der PDU noch ein Nachrichtenidentifikationskriterium enthält. Ausgangsseitig muss eine Datenstruktur produziert werden, welche alle für die Buscontroller-API auf OSI-Schicht 7 notwendigen Informationen enthält, welche ebenfalls aus PDU und Nachrichtenidentifikationskriterium bestehen.

5.3.2.2 Transformation

Eine Transformation, d.h. das Generieren von Ausgabedaten in Abhängigkeit von Eingabedaten und einer Transformationsvorschrift, baut darauf auf, dass Eingabedaten und Transformationsvorschriften eindeutig identifizierbar und miteinander verknüpfbar sind. Da die Eingabedatenstruktur nach Abschnitt 5.3.2.1 neben der PDU ebenfalls ein Nachrichtenidentifikationskriterium enthält, kann die Verknüpfung von Transformationsvorschrift und Eingabedaten leicht modelliert werden. Aufbauend auf Definition 5.3.1 kann somit die formale Definition einer

Transformation aufgestellt werden:

Definition 5.3.2: Transformation

Es gebe
$T =$ Transformation,
$\mathbb{T} =$ Menge der Transformationsvorschriften,
$\mathbb{E}, \mathbb{A}$ aus Definition 5.3.1.

Die Parserfunktion

$$p : \mathbb{E} \mapsto \mathbb{T}^n, n \geq 1 \tag{5.1}$$

ordnet jedem $e \in \mathbb{E}$ ein n-Tupel $(t_i)_{i \in \{1..n\}}$ an Transformationsvorschriften zu.

Für alle Transformationsvorschriften t_i des n-Tupels ist eine Transformation wie folgt definiert:

$$T : (e, t_i) \mapsto \mathbb{A} \tag{5.2}$$

Nachdem Eingabe- und Ausgabedaten sowie Transformationen formal definiert sind, muss eine Nachrichtendarstellung gefunden werden, so dass Formel 5.2 gelten kann. Vorbereitend dazu werden im folgenden Abschnitt die grundlegenden Elemente der Nachrichtendarstellung vorgestellt.

5.3.3 Grundlegende Elemente einer Nachrichtendarstellung

Eine Transformationsvorschrift auf Basis der zu entwickelnden Grammatik beinhaltet in ihrer Nachrichtendarstellung eingehender und ausgehender Nachrichten implizit alle Operationen und Operationsparameter des Übersetzungsalgorithmus. Als grundlegende Teilmenge der Elemente der Grammatik müssen die Datentypen der Operationsparameter ermittelt werden. Dazu ist es sinnvoll, die Operationen zur Protokollübersetzung generisch zu formulieren, um nicht die Anzahl möglicher Implementierungen des Übersetzungsalgorithmus einzuschränken. Der Übersetzungsalgorithmus kann als steuerbarer Kopieralgorithmus verstanden werden, der Daten in Kopieroperationen aus unterschiedlichen Datenquellen (eingehende Nachricht, Systeminformationen, etc.) an bestimmte Stellen der PDU der ausgehenden Nachricht kopiert. Eine Kopieroperation wiederum ist in unterschiedliche Verarbeitungsschritte unterteilbar. Ein identifizierender Verarbeitungsschritt markiert Datenquellen und Teile von Datenquellen

mit einem im Folgenden „Referenz“ genannten eindeutigen Bezeichner, ein schreibender Verarbeitungsschritt kopiert referenzierte Daten in ausgehende Nachrichten.
Die Steuerung des Übersetzungsalgorithmus wird über Operationen zur Flusskontrolle - hier: Evaluierung des Wahrheitswertes einer Aussage - realisiert. Entsprechend des Wahrheitswertes der Aussage wird die komplette Transformationsvorschrift als „anwendbar“ oder „ungültig“ eingestuft. Die Verarbeitungsschritte von Flusskontrolloperationen beinhalten wie zuvor einen identifizierenden Verarbeitungsschritt und zusätzlich einen evaluierenden Verarbeitungsschritt.
Aus den in den einzelnen Verarbeitungsschritten der Kopier- und Steueroperationen erforderlichen Operationsparametern lassen sich die grundlegenden Elemente der Grammatik ermitteln.

5.3.3.1 Identifizierende Verarbeitungsschritte

Ziel eines identifizierenden Verarbeitungsschrittes ist es, eine Referenz auf einen Teil der eingehenden PDU zu erzeugen und für eine Verwendung durch schreibende oder evaluierende Verarbeitungsschritte bereitzustellen. Alle identifizierenden Verarbeitungsschritte maskieren hierbei die gewünschte Information innerhalb der Datenquelle. Daraufhin ist es möglich, die isolierten Informationen mit einem eindeutigen Bezeichner (Referenz) zu versehen, welcher in der Transformationsvorschrift verwendet werden kann.
Um einen identifizierenden Verarbeitungsschritt erfolgreich ausführen zu können, müssen also dessen Parameter die Datenquelle hinreichend genau beschreiben können. Um die Parameter zu ermitteln, werden im Folgenden die zwei unterschiedlichen Datenquellen „PDU“ und „Nachrichtenidentifikationskriterium“ separat betrachtet.
Um die Referenzen auf PDU-Teile formal und gleichzeitig durch einen Algorithmus auswerten und durch einen Datentypen beschreiben zu können, müssen die PDUs im Kontext der Anwendung betrachtet werden. Auf Anwendungsebene liefern und fordern IO-APIs die PDUs eingehender und ausgehender Nachrichten in Form von zusammenhängenden Speicherbereichen (siehe Abschnitt 5.3.2.1). Dieser Speicherbereich kann in der Anwendung z.B. durch eine komplexe Datenstruktur oder durch einen Pointer und die Speicherbereichslänge repräsentiert werden. In jedem Fall ist der Anwendung die Startadresse des Speicherbereichs der PDU bekannt.
Um wiederum einen beliebig positionierten Speicherbereich beliebiger Länge innerhalb der PDU zu maskieren, müssen nur der Offset zur Startadresse der PDU und die Länge des zu maskierenden Bereichs bekannt sein.
Eine Datenstruktur, die eine Maskierung über Offset und Länge eines PDU-Abschnittes beschreibt und mit einem eindeutigen Bezeichner(Referenz) verküpft, erfüllt die Anforderungen

an die Parameter identifizierender Verarbeitungsschritte. Eine Menge dieser Datenstrukture reicht aus, um beliebig viele Speicherbereiche innerhalb einer PDU zu beschreiben. Die Parameter identifizierender Verarbeitungsschritte werden wie folgt definiert:

Definition 5.3.3: Bytebereich

$$\mathbb{O} = \mathbb{N} \times \mathbb{N} \times \mathbb{N} \tag{5.3}$$

mit
$\mathbb{N}$ = Menge der natürlichen Zahlen,
$\mathbb{O}$ = Menge der Bytemasken.

Ein Tupel $(n_0, n_1, n_2) \in \mathbb{N}^3$ beschreibt einen Teil einer eingehenden PDU durch seinen Offset zur Startspeicheradresse der PDU n_1 und seine Länge in Bytes n_2 und verknüpft diesen Teil mit einem eindeutigen Bezeichner n_0.

5.3.3.2 Evaluierende Verarbeitungsschritte

Ziel von evaluierenden Verarbeitungsschritten ist es,

- eine vom Inhalt eines PDU-Teil abhängige Behandlung auszulösen oder
- Wertebereichsvalidierungen von PDU-Teilen durchzuführen.

Alle evaluierenden Verarbeitungsschritte zur Transformationssteuerung werten Bedingungen für die Anwendbarkeit einer Transformationsvorschrift aus. Bei Nichterfüllung der Bedingung wird die weitere Verarbeitung der Transformationsvorschrift abgebrochen. Als Bedingung sind in diesem Zusammenhang einfache, auf die Wahrheitswerte „true“, „false“ abbildbare Ausdrücke festgelegt. Die Ausdrücke sollen dabei der Form *(Referenz, Operator, Argument)* folgen.
Durch die Verknüpfung mehrerer Bedingungen in einer Transformationsvorschrift können mit einfachen Bedingungen komplexe Abhängigkeiten modelliert werden. So ist zum Beispiel eine Wertebereichsvalidierung eines PDU-Teils dadurch möglich, dass der PDU-Teil, der von der im identifizierenden Verarbeitungsschritt generierten Referenz beschrieben wird, mit „größer gleich“ und „kleiner gleich“ Operatoren auf Einhaltung seiner Grenzwerte überprüft wird. Die Nichteinhaltung der Grenzen wäre somit Abbruchkriterium für die Transformation. Für den weiteren Verlauf werden die Bedingungen wie folgt formal definiert:

Definition 5.3.4: Bedingungen

$$\mathbb{COND} = \mathbb{N} \times \mathbb{OP} \times \mathbb{OP}_{arg} \tag{5.4}$$

mit
$\mathbb{N}$ = über eindeutigen Bezeichner referenzierter Bytebereich $\in \mathbb{O}$,
$\mathbb{OP}$ = Menge der zugelassenen Operatoren { ==, !=, <, >, <=, >= },
$\mathbb{OP}_{arg} \subseteq \mathbb{N}$ = Menge der Vergleichswerte für die Operation.

Bedingungen haben immer den Zahlenwert eines über seinen eindeutigen Bezeichner referenzierten Bytebereichs $\in \mathbb{O}$ als Argument und vergleichen diesen entsprechend des Operators mit dem zweiten Argument $\in \mathbb{OP}_{arg}$.

5.3.3.3 Schreibende Verarbeitungsschritte

Um in einer Transformation gültige Ausgangsdaten zu generieren, wird in schreibenden Verarbeitungsschritten eine Ausgangsdatenstruktur sukzessive mit den gewünschten Werten gefüllt. Ziel eines solchen Verarbeitungsschrittes ist es, Ausgangs-PDU-Elemente in die Ausgangsdaten einzufügen, wobei die Menge der Ausgangs-PDU-Elemente aus

- referenzierten Eingangs-PDU-Bestandteilen,
- Ergebnissen von Manipulationen von referenzierten Eingangs-PDU-Bestandteilen und
- referenzierten Systeminformationen

gebildet wird.
Die notwendigen Parameter eines schreibenden Verarbeitungssschrittes können aus diesen unterschiedlichen Arten von PDU-Elementen abgeleitet werden.
Für das Schreiben eines referenzierten Eingangs-PDU-Bestandteils wird neben der formalen Beschreibung des Schreibziels ebenfalls eine durch einen identifizierenden Verarbeitungsschritt generierte Referenz benötigt. Die formale Beschreibung des Schreibziels und des Parameters der Eingangs-PDU-Referenz können beide durch das grundlegende Grammatikelement „Bytebereich“ ausgedrückt werden.
Ergebnisse von Manipulationen an referenzierten Eingangs-PDU-Bestandteilen als auch Systeminformationen können nicht durch das Grammatikelement „Bytebereich“ vollständig beschrieben werden. Das Ermitteln von Manipulationsergebnissen oder Systeminformationen kann in einem Übersetzungsalgorithmus als (Software-) Funktionsaufruf und Verwendung seines Ergebnisses dargestellt werden.

Ein Funktionsaufruf lässt sich durch den Funktionsbezeichner und die Funktionsargumente darstellen. Als Funktionsargument wird eine Sequenz natürlicher Zahlen definiert, welche je nach Funktionsaufruf als Referenzen auf Eingangs-PDU-Bestandteile oder als einfache Zahlenwerte zu interpretieren sind. Solange jede Funktion einen Bytebereich zum Rückgabewert hat, kann das Funktionsergebnis problemlos als Datenquelle einer Kopieroperation in Richtung eines Ausgangs-PDU-Elements verwendet werden.
In der Grammatik sind „Funktionen“ wie folgt definiert:

Definition 5.3.5: Funktionen

$$\mathbb{F} = \mathbb{FN} \times \{\langle a_i \rangle_{i \in \mathbb{N}, a \in \mathbb{ARG}}\} \times \mathbb{O} \tag{5.5}$$

mit
$\mathbb{FN}$ = Menge der Funktionsbezeichner,
$\mathbb{ARG}$ = Menge der Funktionsargumente,
$\mathbb{O}$ = Menge der Bytebereiche, hier: Rückgabewert,
$\mathbb{F}$ = Menge der Funktionen.

Definition 5.3.6: Funktionsbezeichner und Funktionsargumente

$$\mathbb{FN} = \Sigma^n \tag{5.6}$$

$$\mathbb{ARG} = \mathbb{N}^i \times \mathbb{F} \tag{5.7}$$

mit
Σ = das für Funktionsbezeichner genutzte Alphabet,
$i, n \in \mathbb{N}$,
i, n sinnvoll groß,
$\mathbb{FN}$ = Menge der Funktionsbezeichner,
$r \in \mathbb{N}^i$ = Referenz auf Eingangs-PDU-Bestandteil,
$f \in \mathbb{F}$ = Ergebnis eines vorherigen Funktionsaufrufes,
$\mathbb{ARG}$ = Menge der Funktionsargumente.

5.3.4 Entwicklung der Grammatik

Als Grundlage der Entwicklung der Grammatik zur Beschreibung von Protokollübersetzungen wurden in den vergangenen Abschnitten die grundlegenden Elemente der Grammatik aus den

Verarbeitungsschritten einer Übersetzung ermittelt und formal definiert. Anhand dieser Elemente ist es möglich, einzelne Bestandteile eingehender und ausgehender PDU zu beschreiben. Um eine PDU vollständig zu beschreiben, müssen diese Elemente allerdings einer Ordnung unterliegen.
Eine Transformationsvorschrift soll aus zwei Nachrichtendarstellungen für eingehende und ausgehende PDU (im Folgenden: Eingangs- und Ausgangsdaten) bestehen, welche die notwendigen Kopieroperationen implizit beschreiben. Sobald ausgehende PDU als Sequenz von Referenzen auf Bytebereiche und Funktionen beschrieben werden, kann anhand der Reihenfolge der Referenzen in der Sequenz eine Abfolge von Kopieroperationen der referenzierten Inhalte in Richtung der Ausgangsdaten geschlossen werden.
Somit erfüllen die zwei Nachrichtendarstellungen unterschiedliche Zwecke, obwohl sie auf der gleichen Menge an grundlegenden Elementen der Grammatik aufbauen: Die Darstellung der Eingangsdaten definiert die Referenzen auf Eingangs-PDU-Bestandteile, die Darstellung der Ausgangsdaten definiert Struktur, Inhalt und Erzeugungsvorschrift der Ausgangs-PDU-Bestandteile. Die Nachrichtendarstellungen sind wie folgt definiert:

Definition 5.3.7: Eingangsdaten

$$\mathbb{IN} = \mathbb{N} \times \{\langle p_i \rangle_{i \in \mathbb{N}, p \in \mathbb{O}}\} \tag{5.8}$$

mit
$b \in \mathbb{N}$ = eindeutiger Bezeichner der Nachrichtendarstellung,
$\mathbb{O}$ = Menge der Bytebereiche,
$\mathbb{IN}$ = Menge der Eingangsdaten.

Der wesentliche formale Unterschied zwischen den Ausgangsdaten und Eingangsdaten besteht darin, dass die in den Ausgangsdaten formulierte Sequenz von Elementen im Gegensatz zu den Eingangsdaten ebenfalls Funktionen beinhalten darf.

Definition 5.3.8: Ausgangsdaten

$$\mathbb{OUT} = \mathbb{N} \times \{\langle p_i \rangle_{i \in \mathbb{N}, p \in \mathbb{X}}\} \tag{5.9}$$

mit
$b \in \mathbb{N}$ = eindeutiger Bezeichner der Nachrichtendarstellung,
$\mathbb{X} = \mathbb{O} \cup \mathbb{F}$ Menge der Bytebereiche und Funktionen
$\mathbb{OUT}$ = Menge der Ausgangsdaten.

Eine Übertragung zwischen z.B. zwei verteilten CAN-Bussen unter Nutzung des RTE Backbone lässt sich aus Sicht der Protokollübersetzung in zwei Schritte einteilen: Im ersten Schritt werden die CAN-Nachrichten in Richtung des RTE-Protokolls transformiert und im zweiten Schritt, sobald die Nachricht über den RTE Backbone übertragen wurde, aus dem RTE-Protokoll zurück in das CAN-Protokoll. Die Transformationsvorschriften dieser zwei Übersetzungen sind voneinander abhängig, d.h. die Nachrichtendarstellung der Ausgangsdaten der Übersetzung in Richtung RTE muss kompatibel zur Darstellung der Eingangsdaten der Rückübersetzung sein. Um diese Abhängigkeit aufzulösen, müssen Nachrichten auf dem RTE Backbone eindeutig identifzierbar sein. Eine artverwandte, zusätzliche Anforderung an die Identifizierbarkeit von RTE Nachrichten begründet sich in dem Wunsch, mehrere (CAN-)Nachrichten in einer RTE-Nachricht übermitteln zu wollen. Unter derartigen Anforderungen muss nicht nur eine RTE-Nachricht eindeutig identifizierbar sein. Jede in einer RTE-PDU übertragene (CAN-)Nachricht muss mit zusätzlichen Verwaltungsinformationen versehen werden, aus denen die Transformationsvorschrift zur Übersetzung aus dem RTE-Protokoll in das Zielprotokoll ermittelbar ist. Üblicherweise werden Information dieses Typs anhand eines Transportprotokolls kommuniziert.
Da es nicht immer möglich ist, die im Transportprotokoll kodierten Informationen aus den Eingangsdaten einer Transformationsvorschrift zu gewinnen, reichen Elemente des Typs „Bytebereich“ nicht aus, um das Transportprotokoll zu generieren. Da die Datenquelle des Transportprotokolls auch aus Systeminformationen oder in der Gatewayanwendung vorgehaltenen Variablen bestehen kann, wird im Folgenden hilfsweise angenommen, dass dem Übersetzungsalgorithmus eine Funktion bekannt ist, welche das Transportprotokoll generiert und definiert:

Definition 5.3.9: Transportprotokoll

$$\mathbb{TP} \subseteq \mathbb{F} \tag{5.10}$$

mit
$f \in \mathbb{F}$ = Generatorfunktion des Transportprotokolls,
$\mathbb{TP}$ = Menge der Transportprotokolle.

Auf Basis der aufgestellten Definitionen von Eingangsdaten, Ausgangsdaten, Bedingungen und Transportprotokollen kann die Transformationsvorschrift gebildet werden:

Definition 5.3.10: Transformationsvorschrift

$$\mathbb{T} = \mathbb{N} \times \mathbb{IN} \times \mathbb{COND}^* \times \mathbb{OUT} \times \mathbb{TP}^* \tag{5.11}$$

mit
$\mathbb{N}$ = Menge der eindeutigen Bezeichner von Transformationsvorschriften,
$\mathbb{IN}$ = Menge der Eingangsdaten,
$\mathbb{OUT}$ = Menge der Ausgangsdaten,
$\mathbb{COND}$ = Menge der Bedingungen für Gültigkeit der Transformationsvorschrift,
$\mathbb{TP}$ = Menge der Transportprotokolle $\subseteq$ Funktionen,
$\mathbb{T}$ = Menge der Transformationsvorschriften.

5.4 Nachrichtenbündelung

Das im Rahmen dieser Arbeit vorgestellte Kommunikationsgateway verbindet die CAN-Busse eines Automobils mit dem RTE Backbone und übertragt CAN-Nachrichten auf dem RTE Backbone zwischen den CAN-Bussen. Da zusätzlich zum Datenverkehr zwischen den automobilen CAN-Bussen weitere Datenquellen und -senken - wie z.B. Rückfahrkamera oder LIDAR - eine teilweise erhebliche und ungleichmäßige Auslastung des RTE Backbone verursachen können, müssen Maßnahmen zur Optimierung der Busauslastung getroffen werden.
Eine Optimierung der Busauslastung kann unterschiedliche Ziele haben, von der Verhinderung von Überlastsituationen über die Gewährleistung von Datenraten bis hin zu Schaffung von Bandbreitenreserven. Um eine Optimierung der Busauslastung planen und vornehmen zu können, muss abgegrenzt werden, was überhaupt im Kontext dieser Arbeit optimiert werden kann. Das in dieser Arbeit vorgestellte Kommunikationsgateway verbindet CAN-Busse über ein RTE Backbone Netzwerk unter Weiterverwendung existierender CAN-Anwendungen miteinander. Mit der fehlenden Kontrolle über die CAN-Anwendungen fehlt ebenso die Möglichkeit zur Steuerung der Nachrichtenfrequenz und des Nachrichteninhalts der CAN-Kommunikation. Die Möglichkeiten der Einflussnahme beschränken sich zum einen auf die Kommunikation des RTE Backbone und zum anderen auf die Bestandteile dieser Kommunikation, welche durch das Kommunikationsgateway verarbeitet werden. Im Rahmen dieser Arbeit wird daher unter der Optimierung der Busauslastung das Bestreben verstanden, die für die Übertragung von CAN-Nachrichten auf dem RTE Backbone benötigte Bandbreite zu verringern.
Unter der Annahme, dass die über das RTE Backbone Netzwerk zu übertragenen CAN-Nachrichten keiner Optimierung wie z.B. Kompression der Daten bedürfen, kann nur die

Nutzung des RTE-Kommunikationsprotokolls optimiert werden, um den sogenannten Protokolloverhead zu minimieren. Der Protokolloverhead einer RTE-Nachricht setzt sich aus dem Nachrichtenheader von 34 Bytes, der Prüfsumme von 4 Bytes und den Padding Bytes zusammen. Padding Bytes füllen eine RTE-Nachricht mit kleiner Nutzlast gegebenenfalls auf die minimale RTE-Nachrichtennutzlast von 46 Bytes auf. Sie verbrauchen also Bandbreite, ohne relevante Informationen zu tragen. Ein erster Optimierungsansatz würde darauf abzielen, RTE-Nachrichten mit einer minimalen Nachrichtennutzlast von 46 Bytes zu generieren.
Eine Erweiterung dieses Optimierungsansatzes versucht nicht nur die Padding Bytes des Protokolloverheads zu minimieren, sondern ebenfalls den Anteil der Protokolldaten an der Bandbreitennutzung zu verringern. An einer minimal kleinen RTE-Nachricht mit 46 Bytes Nutzlast und 38 Bytes Protokolloverhead hat der Protokolloverhead den Anteil von 45%, an einer maximal großen RTE-Nachricht mit 1500 Bytes Nutzlast hat der Protokolloverhead von 38 Bytes hingegen nur den Anteil von ca. 2,5%. Zur Optimierung der Busauslastung gilt es also Wege zu finden, möglichst maximal große RTE-Nachrichten zu versenden.
Die minimale RTE-Nutzlastgrösse von 46 Bytes kann selbst durch die Übertragung eines CAN-Nachrichtenframes maximaler Größe (129 Bits) nicht erreicht werden. Zur Optimierung der Busauslastung müssen daher Möglichkeiten und Konsequenzen einer gebündelten Übertragung mehrerer CAN-Nachrichten in einer RTE-Nachricht untersucht werden.
In den folgenden Abschnitten wird die Nachrichtenbündelung in Bezug zum konkreten Anwendungsfall gestellt und eine geeignete Implementierung eines Puffers für Nachrichtenbündel vorgestellt. Abschließend wird aus einer Verallgemeinerung der Nachrichtenbündelung für die Übertragung von Nachrichten anderer Busse außer CAN eine Verfeinerung der Implementierung entwickelt.

5.4.1 Einordnung und Konsequenzen der Nachrichtenbündelung

Die gebündelte Übertragung mehrerer CAN-Nachrichten in einer RTE-Nachricht bedingt, dass zuerst zum Nachrichtenbündel hinzugefügte CAN-Nachrichten im Kommunikationsgateway zwischengespeichert werden, bis ein Sendekriterium wie z.B. die Nachrichtenbündelgrösse erfüllt ist. Diese Zwischenspeicherung von CAN-Nachrichten erhöht die Übertragungslatenz der Nachrichten um die Dauer der Zwischenspeicherung.
In automobilen CAN-Bussen werden unterschiedliche Nachrichtentypen kommuniziert, zeitkritische Nachrichten und Nachrichten ohne jede Zeitgarantien. Für zeitkritische CAN-Nachrichten wird zur Entwurfszeit eine maximale Übertragungslatenz definiert, welche rechnerisch validierbar ist (siehe Abschnitt 4.5). Nachrichten ohne Zeitgarantien können zum Zweck der Nachrichtenbündelung beliebig lange im Kommunikationsgateway zwischengespeichert wer-

den, zeitkritische Nachrichten dürfen aber trotz Nachrichtenbündelung ihre maximale Übertragungslatenz nicht überschreiten.
Um dieses Verhalten sicherzustellen, können unterschiedliche Implementierungen einer Nachrichtenbündelung gewählt werden. Entweder es wird auf die Bündelung zeitkritischer Nachrichten komplett verzichtet, oder das Kommunikationsgateway stellt sicher, dass ein Nachrichtenbündel auch bei nicht-optimaler Auslastung der zur Verfügung stehenden RTE-Nachrichtennutzlast versendet wird, sobald dies für die Einhaltung der maximalen Übertragungslatenz einer CAN-Nachricht des Nachrichtenbündels erforderlich ist.

5.4.2 Implementierung eines Nachrichtenpuffers

Die Aufgabenstellung an einen Nachrichtenpuffer im Kommunikationsgateway ist die Speicherung von Nachrichten und die Bereitstellung von Informationen bezüglich der maximalen Übertragungslatenz der einzelnen gespeicherten CAN-Nachrichten. Eine erste Implementierungsmöglichkeit ist die Bereitstellung eines einzelnen Nachrichtenpuffers für alle CAN-Nachrichten, welche aufgrund ihrer schwachen Anforderungen an die maximale Übertragungslatenz problemlos bündelbar und tolerant bezüglich zusätzlicher Übertragungslatenz sind.
Die Datenstruktur eines derartigen Nachrichtenpuffers würde dann regelmäßig und iterativ dahingehend überprüft, ob eine der gepufferten CAN-Nachrichten versendet werden muss, um deren maximale Übertragungslatenz einzuhalten. Sobald eine oder mehrere zu versendende Nachrichten im Nachrichtenpuffer vorliegen, werden diese aus dem Nachrichtenpuffer entfernt und auf dem RTE Backbone versendet. Dieser Ansatz beinhaltet das Risiko, dass der Speicherbereich des Nachrichtenpuffers aufgrund der regelmäßigen Lese- und Schreiboperationen fragmentiert wird. Dies wäre z.B. dann der Fall, wenn mittig aus dem Speicherbereich eine so kleine Nachricht entfernt wird, dass der freigewordene Speicher nicht durch größere, neu hinzugefügte Nachrichten wiederverwendet werden kann. Darüber hinaus macht eine solche Fragmentierung des Nachrichtenpuffers erforderlich, dass Informationen bezüglich der Reihenfolge des Nachrichteneinganges gespeichert werden, damit die Eingangsreihenfolge der zu bündelnden Nachrichten auf dem RTE Backbone erhalten bleibt.
Um die Fragmentierung des Nachrichtenbündelpuffers zu kompensieren, kann ein Algorithmus entsprechend einer Memory Management Unit (MMU) implementiert werden, der die Fragmentierung des Nachrichtenbündelpuffers auflöst. Die Nutzung oder Emulation einer MMU fügt einer Anwendung allerdings ein nicht-deterministisches Laufzeitverhalten hinzu, was in Echtzeitanwendungen zu vermeiden ist. Dies erfolgt aufgrund der für die dynamische Speicherverwaltung notwendigen Rechenzeit und des schwer vorhersagbaren Fragmentierungsgrades

des Speichers. Daher wird im folgenden eine Implementierung eines Nachrichtenbündelpuffers vorgestellt, der nicht anfällig für eine Fragmentierung des Puffers ist und die Eingangsreihenfolge der zu bündelnden Nachrichten implizit bewahrt.
Die in dieser Arbeit gewählte Implementierung bedient sich mehrerer, zur Entwurfszeit definierter Speichergruppen für zu bündelnde CAN-Nachrichten. Für jede Speichergruppe ist eine maximale Lebenszeit definiert, zu deren Ende alle bis dahin in der Speichergruppe angesammelten CAN-Nachrichten versendet werden. Für jede CAN-Nachricht ist definiert, ob und in welcher Speichergruppe die Nachricht bündelbar ist.
Sobald die erste CAN-Nachricht zu einer Speichergruppe hinzugefügt wird, wird die Lebenszeit entsprechend der Speichergruppendefinition gesetzt. Ein Echtzeittask der Anwendung des Kommunikationsgateways aktualisiert fortlaufend die Lebenszeiten aller Speichergruppen und taktet gegebenenfalls das enthaltene Nachrichtenbündel in Richtung RTE Backbone aus (vergleiche Abbildung 5.6). Durch diese Implementierung und die Zuordnung von Nachrichten zu Speichergruppen mit begrenzten Lebenszeiten ist sichergestellt, dass zur Entwurfszeit bereits die maximale, zusätzliche Übertragungslatenz einer Nachricht aufgrund der Nachrichtenbündelung bekannt ist.

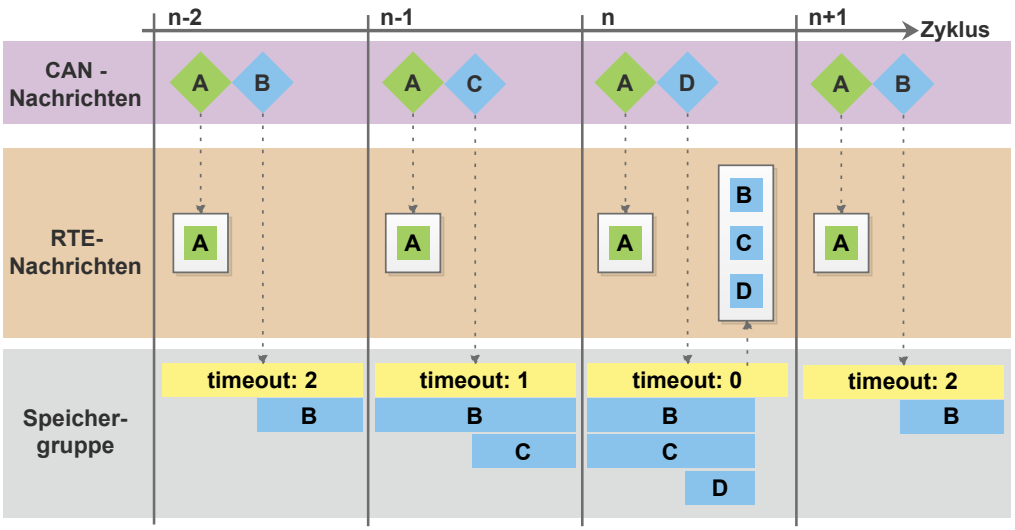

Abbildung 5.6: Nachrichtenbündelung der zeitkritischen Nachrichten „A“ und der schwachzeitkritischen Nachrichten „B“, „C“, „D“ aus Sicht der verbundenen Busse und der internen Speicherhaltung für Nachrichten im Kommunikationsgateway. Die Nachricht „A“ darf nicht verzögert werden, die Nachrichten „B“, „C“, „D“ dürfen je maximal 2 Schedulerzyklen verzögert werden.

5.4.3 Transportprotokoll

Die Anwendung des Kommunikationsgateways muss via RTE Backbone empfangene Nachrichtenbündel entpacken und weiterverarbeiten können. Dazu ist es notwendig, dass jedes Nachrichtenbündel Strukturinformationen beinhaltet, um die Identifikation einzelner Nachrichten des Bündels zu ermöglichen. Zu diesem Zweck wird ein Transportprotokoll für zu übertragene CAN-Nachrichten innerhalb eines Nachrichtenbündels eingesetzt. Der Vollständigkeit halber werden hier im Transportprotokoll nicht nur für die Entbündelung notwendige Informationen repräsentiert, sondern ebenfalls die Informationen, die für eine Nachrichtenübersetzung entsprechend der IDL aus Abschnitt 5.3 benötigt werden.

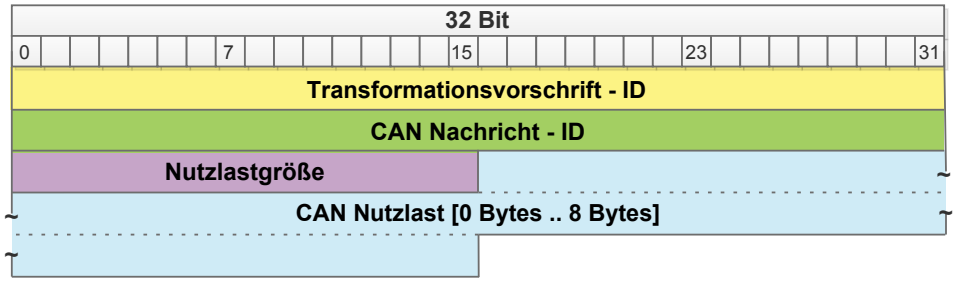

Abbildung 5.7: Transportprotokoll zur Repräsentation einer CAN-Nachricht innerhalb eines Nachrichtenbündels.

Ein Nachrichtenbündel besteht im Sinne dieses Transportprotokolls aus Blöcken von 10 Bytes bis 18 Bytes, welche jeweils eine CAN-Nachricht repräsentieren. Im Folgenden werden diese Blöcke als TPU bezeichnet.

5.4.4 Fragmentierung gebündelter Nachrichten

Das Versenden von Nachrichtenbündeln aus TPU entsprechend des Transportprotokolls aus Abschnitt 5.4.3 lässt sich naiv so implementieren, dass eine RTE-Nachricht solange mit TPU befüllt wird, bis das nächste TPU nicht mehr vollständig in die RTE-Nachricht passt. In einem solchen Ansatz besteht die Gefahr, dass bis zu 17 Bytes der maximalen Nutzlast einer RTE-Nachricht nicht verwendet werden. Verallgemeinert wird potentiell die „maximale Größe [Bytes]“ - 1 einer TPU verschwendet.
Da die geringe Nutzlastgröße von CAN-Nachrichten die maximale Größe des entsprechenden TPU definiert, ist die Verschwendung der Nutzlastkapazität einer RTE-Nachricht somit bei der Übertragung von CAN-Nachrichten wenig gravierend. Der Versuch einer Verallge-

meinerung der Nachrichtenbündelung zur Übertragung von Nachrichten eines unbekannten, dritten Bussystems zeigt aber schnell die Schwäche dieses Ansatzes: Sobald Nachrichten eines hypothetischen Bussystems mit einer TPU-Größe von knapp mehr als der Hälfte (751 Bytes) der maximalen Nutzlastkapazität von 1500 Bytes einer RTE-Nachricht übertragen werden sollen, schlägt die Nachrichtenbündelung fehl - es wird maximal eine dieser hypothetischen Nachrichten pro RTE-Nachricht übertragen.
Um diesen Missstand zu beheben, können die TPU fragmentiert werden. Diese Fragmentierung erlaubt es, verbleibende freie Nutzlastkapazität einer RTE-Nachricht anteilig mit einer TPU aufzufüllen und den Rest der TPU in einer weiteren RTE-Nachricht zu übertragen (siehe Abbildung 5.9).

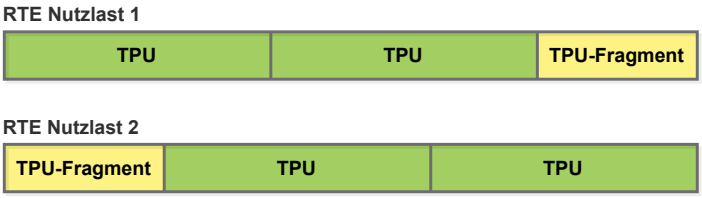

Abbildung 5.8: Fragmentierung einer TPU zur optimalen Ausnutzung einer RTE-Nachricht bei der Übertragung mehrerer TPU in mehreren RTE-Nachrichten.

5.4.5 Protokoll für fragmentierte TPU

Bei der Verwendung fragmentierter TPU in der Übertragung über den RTE Backbone kann eine empfangene RTE-Nachricht durch das Kommunikationsgateway nicht mehr ohne Weiteres entpackt und weiterverarbeitet werden. Entgegen des Ansatzes aus Abschnitt 5.4.3 kann nicht mehr davon ausgegangen werden, dass eine RTE-Nutzlast mit einer vollständigen TPU beginnt. Abhilfe kann durch ein weiteres Transportprotokoll geschaffen werden, welches sogenannte Bundle Protocol Unit (BPU) definiert. Das Transportprotokoll der BPU versieht jede RTE-Nachricht mit zusammengehörigen TPU-Fragmenten mit einem eindeutigen Bezeichner für das Nachrichtenbündel und Informationen über die Reihenfolge der BPU (siehe Abbildung 5.9).
Analog zu den Speichergruppen für zu bündelnde Nachrichten werden eingehende BPU solange in einem Defragmentierungsspeicher abgelegt, bis alle BPU eines Nachrichtenbündels empfangen worden sind. Daraufhin kann ein Echtzeittask des Kommunikationsgateways die einzelnen TPU aus den BPU entpacken und weiterverarbeiten.

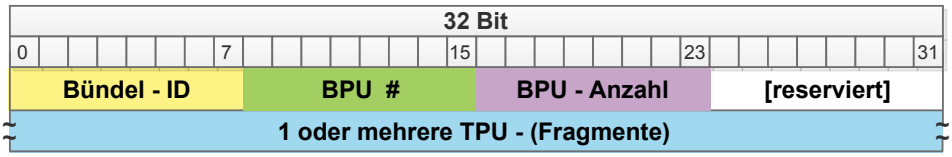

Abbildung 5.9: Protokoll für die Übertragung von BPU, welche fragmentierte TPU eines Nachrichtenbündels beinhalten.

5.5 Softwarearchitektur Gateway

In der Fachliteratur wird bis heute lebhaft diskutiert, was eine passende Definition für den Begriff „Softwarearchitektur“ sei. Anschauliches Beispiel für die Definitionsvielfalt ist das Resultat der Bemühung des Software Engineering Institute (SEI) der Carnegie Mellon University, Pittsburgh, Pennsylvania: Anstelle einer allgemeingültigen Definition bietet das SEI ca. 100 unterschiedliche Definitionen, welche sich darin unterscheiden, aus welcher Epoche der Computerwissenschaften sie stammen, aus welcher Perspektive sie die Fragestellung betrachten und wie oft sie in Fachliteratur veröffentlicht wurden [Software Engineering Institute (2007)]. Im Rahmen dieser Arbeit wird nicht versucht, einer beliebten Definition oder gar allen Definitionen zu entsprechen. Vielmehr legt dieser Abschnitt den Fokus auf ausgewählte Verwendungszwecke der unter dem Schlagwort „Software Architektur“ präsentierten Informationen, namentlich

- Beschreibung der Struktur, Komponenten, Schnittstellen und Abhängigkeiten,
- Beschreibung des Datenflusses innerhalb der Software,
- Beschreibung grundlegender Designentscheidungen,
- Kommunikation der Software mit anderen Entwicklern,
- Kommunikation der Software mit Stakeholdern.

Durch die Beschreibung der Softwarearchitektur werden somit implizit auch ausgewählte Aspekte der Softwarequalität sichtbar, wie z.B.

- Funktionalität der Software,
- Modularität der Software,
- Modifizierbarkeit der Software,
- Wartbarkeit der Software,
- Effizienz in Form von Zeitverhalten.

5.5.1 UML Komponentendiagramm

Das UML Komponentendiagramm zum Kommunikationsgateway visualisiert einzelne Komponenten der Software, die Schnittstellen der Komponenten und die durch die Schnittstellen modellierten Abhängigkeiten zwischen den Komponenten.

5.5.2 Black Box Modell

Sobald das Kommunikationsgateway als Black Box angesehen wird, das bidirektional ausschließlich mit den Buscontroller-APIs des Betriebssystems kommuniziert, bietet das Komponentendiagramm einen Einblick in diese Black Box. Die Anbindung an das Betriebssystem ist im Komponentendiagramm über die Schnittstelle „IRawMessage“ dargestellt, welche geeignet ist, die Parameter der Buscontroller-API-Funktionen aufzunehmen oder bereitzustellen.
Erwähnenswert ist hierbei, dass das Versenden einer RTE-Nachricht nicht explizit innerhalb der Black Box ausgelöst wird. Innerhalb der Black Box werden nur RTE-Nachrichten repräsentierende Speicherbereiche reserviert und befüllt, deren Versand in einem separaten Echtzeit-Task zyklisch ausgelöst wird.
Intuitiver erfolgt hingegen der Versand von CAN-Nachrichten, welche auf demselben Weg generiert werden, deren Versand aber explizit innerhalb der Black Box ausgelöst wird. Der Versand von CAN-Nachrichten ist allerdings aufgrund der Implementierung des CAN-Controllers ebenfalls vom Programmfluss der Black Box entkoppelt, da die CAN-Controller-API in Senderichtung nicht blockierend ist, d.h. nicht auf das tatsächliche Austakten der Nachricht wartet.

5.5.3 Erstellung von Jobs bei Nachrichteneingang

Ein Nachrichteneingang löst im Kommunikationsgateway einen Interrupt Request (IRQ) aus, für den die Anwendungssoftware eine ISR definieren muss. In einem Echtzeitsystem ist es nicht ratsam, rechenzeitintensive oder in der Rechenzeit schwankende Operationen innerhalb einer ISR zu implementieren. Im schlechtesten Fall unterbrechen diese ISR die softwareseitige Implementierung des Echtzeitverhaltens. Daher wird die Nachrichtenverarbeitung vom Nachrichtenempfang durch eine Queue mit FIFO-Charakteristik entkoppelt. Diese Queue speichert eingehende Nachrichten und Verarbeitungsvorschriften entsprechend des „IJob“-Interface für eine entkoppelte Verarbeitung in einem anderen Echtzeittask.
Die zu einer eingehenden Nachricht gehörigen Verarbeitungsvorschriften werden durch die Mapping-Komponente bereitgestellt, welche anhand eines Nachrichtenidentifikationskriteriums alle dieser Nachricht zugeordneten Verarbeitungsvorschriften ermittelt. Nur wenn mindestens eine Verarbeitungsvorschrift gefunden wurde, ist die entsprechende eingehende

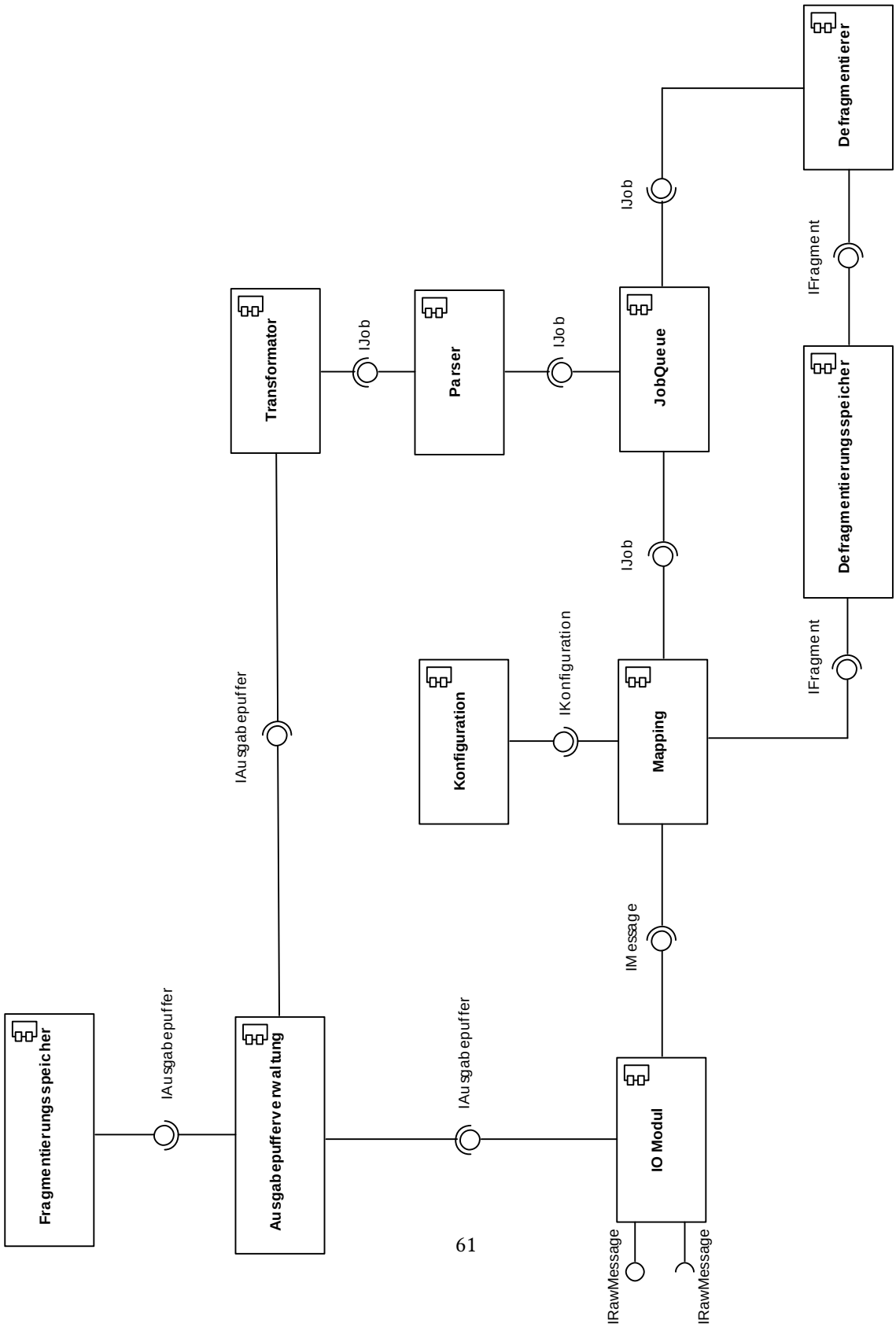

Abbildung 5.10: UML Komponentendiagramm der Gatewayanwendung.

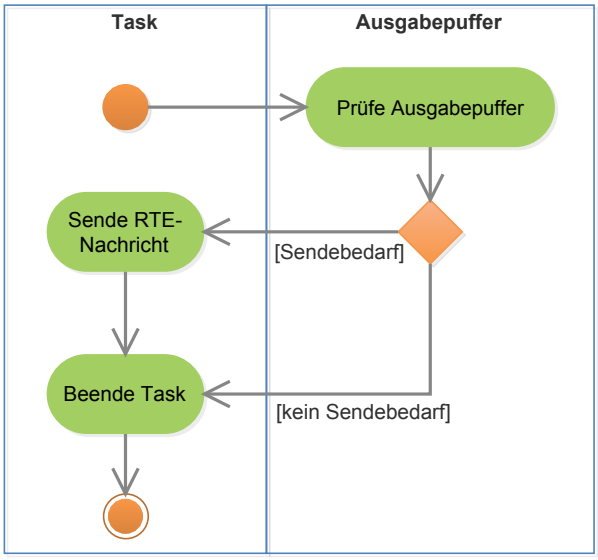

Abbildung 5.11: Aktivitätsdiagramm: Verhalten des Echtzeittasks zum Versand von RTE-Nachrichten. Der Ausgabepuffer kann als FIFO-Speicher verstanden werden, welcher alle noch nicht übertragenen RTE-Nachrichten enthält.

Nachricht durch das Kommunikationsgateway zu verarbeiten - andernfalls wird die Nachricht ignoriert.
Analog zum Empfang von CAN-Nachrichten ist die ISR zum Empfang von RTE-Nachrichten implementiert. Allerdings wird zusätzlich zur Nachrichtenzuordnung ebenfalls überprüft, ob die RTE-Nachricht Teil eines Nachrichtenbündels entsprechend Abschnitt 5.4 ist. Teile von Nachrichtenbündeln werden nicht direkt als Jobs in der JobQueue abgelegt, sondern in einem dedizierten Speicherbereich bis zum Eingang aller Teile eines Nachrichtenbündels zwischengespeichert.
Der Speicher für fragmentierte Nachrichten, also für Teile von Nachrichtenbündeln, wird von einem Echtzeittask regelmäßig darauf untersucht, ob ein Nachrichtenbündel vollständig empfangen wurde. Sobald ein vollständiges Nachrichtenbündel erkannt wird, werden die Verarbeitungsvorschriften und Nachrichteninhalte entsprechend des „IJob“-Interface in der JobQueue zur weiteren Verarbeitung abgelegt (siehe Abbildung 5.14).
Die JobQueue wird also mit Aufträgen aus drei unterschiedlichen Quellen befüllt, wobei die Quellen die CAN-ISR, die RTE-ISR oder der Echtzeit-Task zur Defragmentierung von Nachrichtenbündeln sein können.

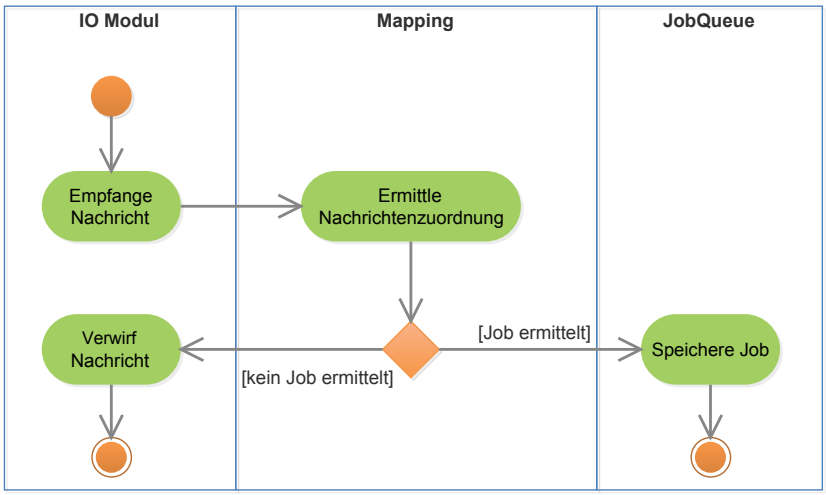

Abbildung 5.12: Aktivitätsdiagramm: Verhalten der im IO Modul implementierten ISR zur Behandlung eingehender CAN-Nachrichten; ein Job wird gespeichert, wenn die CAN-Nachricht einer Verarbeitungsvorschrift zugeordnet ist.

5.5.4 Verarbeitung der Jobs aus der JobQueue

Die Verarbeitung eines in der JobQueue gespeicherten Jobs ist in einem Echtzeittask implementiert, der sich unterschiedlicher Komponenten zur Erfüllung seiner Teilaufgaben bedient. Ein Job entsprechend des „IJob“-Interface wird der JobQueue entnommen und an die Parser-Komponenten weitergeleitet.
Die Aufgaben der Parser-Komponente bestehen zum einen daraus, die einzelnen Byteberei-che zu identifizieren (siehe Definition 5.3.3) und mit Referenzen zu versehen (siehe Abschnitt 5.3.3.1) und zum anderen daraus, die in einer Verarbeitungsvorschrift enthaltenen Bedingungen zu testen (siehe Definition 5.3.4). Anhand der Anzahl validierter Bedingungen bewertet die Parser-Komponente die Verarbeitungsvorschriften, um dadurch die bestmögliche passende Verarbeitungsvorschrift entsprechend der Gatewaykonfiguration zu ermitteln. Sollte die Parser-Komponente nicht in der Lage sein, eine Verarbeitungsvorschrift mit gültigen Bedingungen zu ermitteln, wird der Job verworfen und der Echtzeittask beendet. Sobald die Parser-Komponente jedoch eine Verarbeitungsvorschrift identifiziert hat, eliminiert sie die unterlegenen Verarbeitungsvorschriften aus dem Job und leitet diesen Job an die Transformationskomponente weiter.
Die Transformationskomponente hat primär die Aufgabe, eine Protokollübersetzung auf empfangene Nachrichten anzuwenden, um die auszugebenden Nachrichten zu generieren. Um

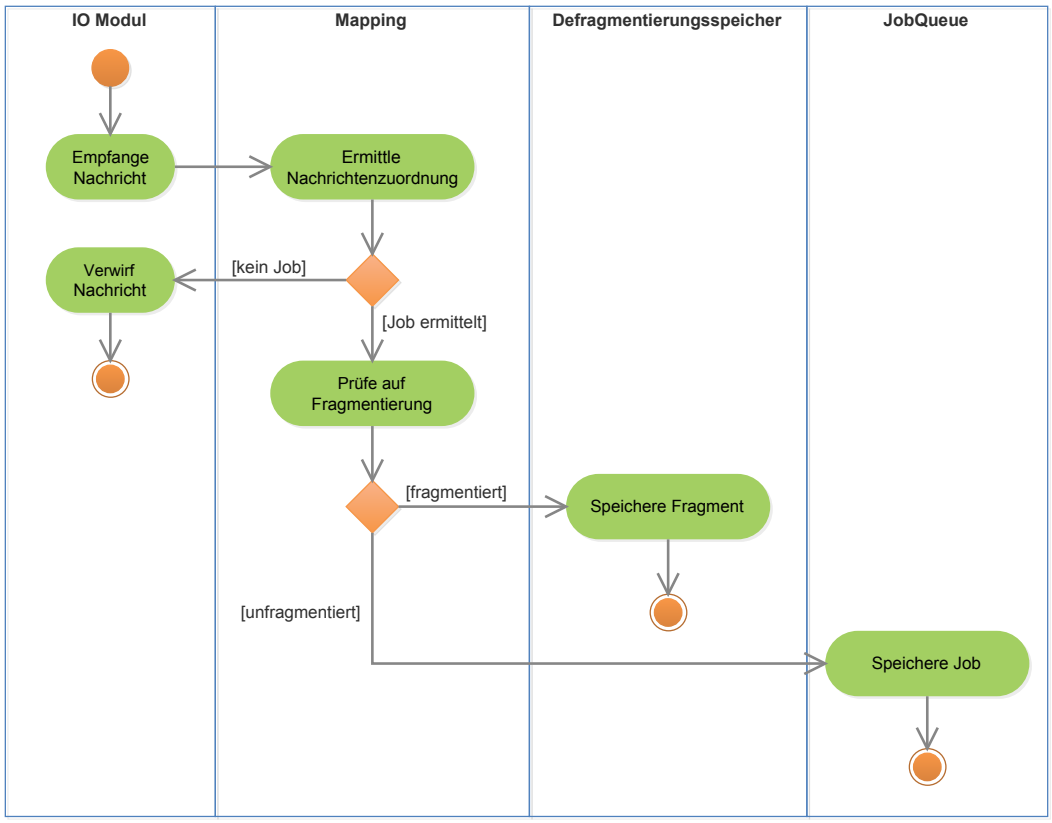

Abbildung 5.13: Aktivitätsdiagramm: Verhalten der im IO Modul implementierten ISR zur Behandlung eingehender RTE-Nachrichten; Teile von Nachrichtenbündeln werden zwischengespeichert, andere RTE-Nachrichten werden als Jobs in der JobQueue gespeichert.

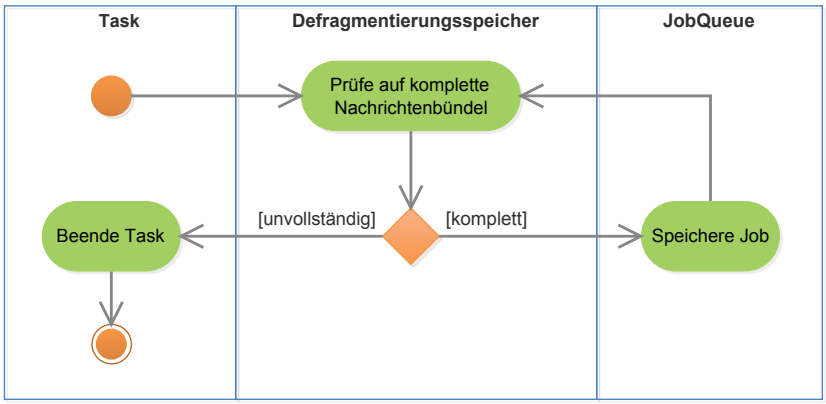

Abbildung 5.14: Aktivitätsdiagramm: Verhalten des Echtzeittasks zur Erkennung vollständiger Nachrichtenbündel. Aufgabe ist das Generieren und Speichern von Aufträgen entsprechend des „IJob“-Interface.

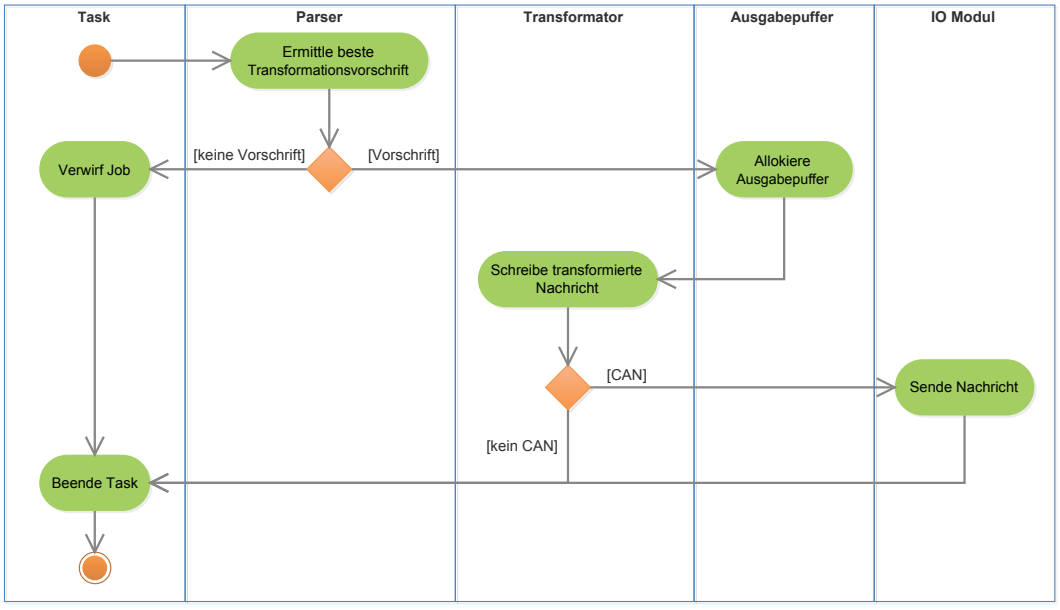

Abbildung 5.15: Aktivitätsdiagramm: Echtzeittask zur Verarbeitung von Jobs entsprechend des „IJob“-Interface, die aus der JobQueue entnommen wurden.

diese Aufgabe erfüllen zu können, müssen allerdings weitere Komponenten verwendet werden. Sobald die Transformationskomponente errechnet hat, welche Größe [Bytes] die auszugebende Nachricht haben wird, bedient sich die Transformationskomponente der Ausgabepufferverwaltung, um einen Speicherbereich für die zu schreibende Nachricht zu allozieren.
Was für einen Speicherbereich die Ausgabepufferverwaltung alloziert, ist davon abhängig, was für ein Ausgabeziel für die auszugebende Nachricht durch die Verarbeitungsvorschrift konfiguriert ist: Die Ausgabepufferverwaltung fragt entweder die Betriebssystem-API nach einem Speicherbereich für ausgehende RTE-Nachrichten, oder die Fragmentierungsspeicher-Komponente nach einem Speicherbereich innerhalb einer Speichergruppe für gebündelt zu übertragende Nachrichten (siehe Abschnitt 5.4.2), oder die Ausgabepufferverwaltung alloziert über das IO Modul eine Datenstruktur entsprechend der Betriebssystem-API zum Versand von CAN-Nachrichten.
In Kenntnis des Ausgabespeicherbereichs kann die Transformationskomponente die Verarbeitungsvorschrift zur Protokollübersetzung abarbeiten. Die Transformationskomponente schreibt nun - wieder in Abhängigkeit des Ausgabeziels - unterschiedliche Daten in den Ausgabespeicherbereich:
Für auszugebende CAN-Nachrichten nimmt die Transformationskomponente die Protokollübersetzung zur Generierung der CAN-Nutzlast vor und fügt der CAN-Nachrichtendatenstruktur Verwaltungsinformationen wie die Nutzlastlänge und die CAN-Nachrichten-ID hinzu. Für auszugebende RTE-Nachrichten nimmt die Transformationkomponente eine Fallunterscheidung vor, welche eine unterschiedliche Behandlung von gebündelt und nicht-gebündelt zu übertragenden RTE-Nachrichten auslöst. Für nicht gebündelt zu übertragende Nachrichten schreibt die Transformationskomponente einen passenden Protokoll-Header für BPU und TPU (siehe Abschnitt 5.4) sowie die Nachrichtennutzlast in einen vom Betriebssystem verwalteten Speicherbereich. Die Behandlung bebündelt zu übertragender Nachrichten weicht davon insoweit ab, als dass nur der Protokoll-Header für TPU sowie die Nachrichtennutzlast in einen von der Gatewayanwendung selbst kontrollierten Speicherbereich geschrieben wird, den Fragmentierungsspeicher.
Abschließend überprüft die Transformationkomponente, ob das Ausgabemedium ein CAN-Bus ist. In diesem Fall wird von der Transformationskomponente die Komponente „IO Modul“ beauftragt, das nicht-blockierende Senden der CAN-Nachricht auszulösen. Der zeitgesteuerte Versand von RTE-Nachrichten wird hingegen nicht explizit von der Transformationskomponente ausgelöst. Diese Aufgabe übernimmt stattdessen der in Abbildung 5.11 beschriebene Echtzeittask.

5.5.5 Echtzeittask zur Verwaltung von Nachrichtenbündel-Speichergruppen

Im vorherigen Abschnitt wurde die besondere Behandlung von gebündelt zu übertragenen Nachrichten durch die Transformationskomponente vorgestellt. Die besondere Behandlung manifestiert sich darin, dass die Transformationskomponente eine Nachrichtenbündel-Speichergruppe mit Sequenzen aus TPU-Protokollheader und Nachrichtennutzlast füllt.
Der eigentliche Versand gebündelter Nachrichten wird durch einen dedizierten Echtzeittask angestoßen, der die Nachrichtenbündel-Speichergruppen entsprechend des in Abschnitt 5.4 vorgestellten Konzeptes verwaltet. Hierzu überprüft der Echtzeittask die unterschiedlichen Speichergruppen auf das Ablaufen eines timeouts, also auf die Notwendigkeit, alle bis dahin in der Speichergruppe aufgelaufenen Nachrichten zu versenden.

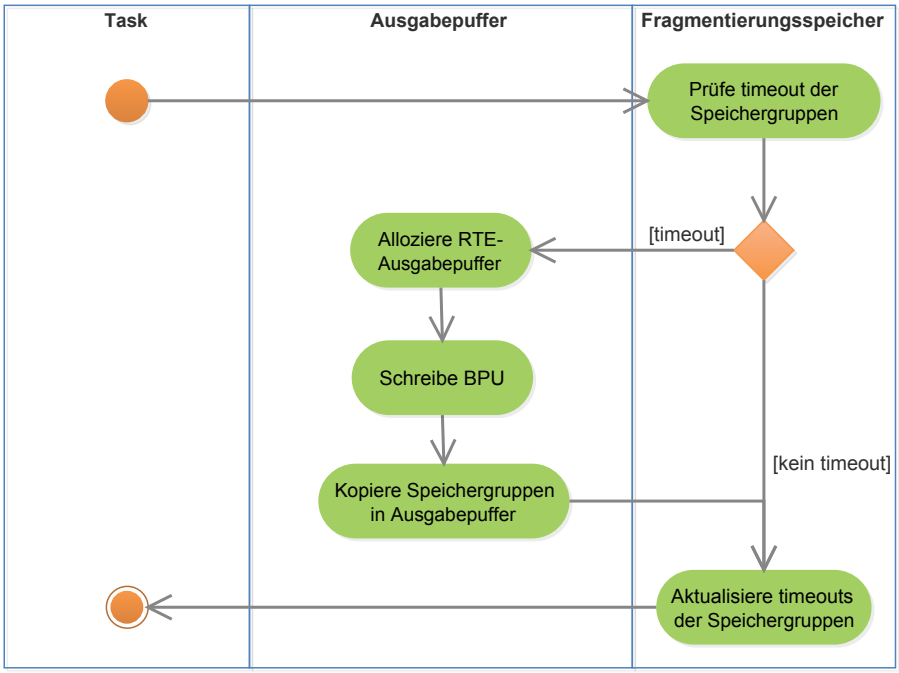

Abbildung 5.16: Aktivitätsdiagramm: Echtzeittask zur Überprüfung und Pflege von Nachrichtenbündel-Speichergruppen, sowie zum Initiieren des Versands gebündelter Nachrichten.

Sobald eine Speichergruppe versendet werden soll, muss der Echtzeittask analog zur Transformationskomponente einen Speicherbereich von der Ausgabepufferverwaltung anfordern. Dieser Speicherbereich stammt in diesem Fall allerdings nicht aus dem durch die Gatewayanwendung selbst verwalteten Fragmentierungsspeicher, sondern wird über die Betriebssystem-API

zum Versand einer RTE-Nachricht angefordert. In Abhängigkeit der Größe des zu versendenden Nachrichtenbündels wird somit vom Betriebssystem ein Speicherbereich alloziert, der ggf. der Größe mehrerer RTE-Nachrichten entspricht.
In diesen vom Betriebssystem allozierten Speicherbereich schreibt der Echtzeittask zuerst die BPU-Protokollheader der einzelnen RTE-Nachrichten an ihre jeweiligen Offsets. Daraufhin kopiert der Echtzeittask die Daten aus den Speichergruppen als Nutzlast in den allozierten Speicherbereich, der die RTE-Nachrichten aufnimmt. Abschließend aktualisiert der Echtzeittask die Timeouts der Speichergruppen, indem er entweder abgelaufene Timeouts zurücksetzt oder nicht abgelaufene Timeouts dekrementiert.

6 Auswertung

Die Auswertung des im Rahmen dieser Arbeit entwickelten Kommunikationsgateways berührt unterschiedliche Aspekte der Softwareentwicklung. Die Bewertung wird jeweils aus der Sicht der Qualitätssicherung, der Praxistauglichkeit und der Evaluation des Potentials für zukünftige Anwendungsfälle durchgeführt.

6.1 Qualitätssicherung

Die Untersuchung der Qualität eines Softwaresystems orientiert sich immer an einem Modell, das Merkmale und Indikatoren für die Qualität definiert. Dabei ist zwischen Modellen zur Bewertung der Softwarequalität wie z.B. dem ISO Standard 25000 und Modellen zur Bewertung des Entwicklungsprozesses zu unterscheiden.

6.1.1 Prozessqualität

Die Prozessqualität umfasst die Qualitätsanforderungen, die nicht mit einem Produkt, System oder der Software verbunden sind und nur durch eine Untersuchung der Durchführung eines Projektes verifizierbar sind. Beispiele für derartige Qualitätsanforderungen sind die Bestimmung bzw. die Entscheidung für ein bestimmtes Modell des Entwicklungsprozesses, Definitionen einer Anforderung an die Codeabdeckung durch Testfälle oder die Wahl einer Entwicklungsmethode wie z.B. agile Softwareentwicklung. Dabei ist die Zuordnung der Codeabdeckung durch Testfälle zur Prozessqualität nicht komplett eindeutig: Während die Forderung einer hohen Codeabdeckung der Prozessqualität zuzuschlagen ist, ist die Umsetzung und Erreichung einer hohen Codeabdeckung eine Frage der Softwarequalität.
Für die Entwicklung des Kommunikationsgateways wurde ein Entwicklungsprozess entsprechend des V-Modells (siehe 3.1.2) gewählt. Durch das V-Modell sind hierarchische Prozessebenen und für jede Prozessebene ein Paar aus Anforderung und Validierung bzw. Spezifikation und Verifikation definiert. Da Anforderungen teilweise vage oder nur implizit definiert sind, ist es kaum möglich, diese automatisiert und toolgestützt zu validieren. Im Rahmen dieser Arbeit sind daher Vorgehensweisen festgelegt und eingehalten worden, die sich primär auf die

Softwarequalität auswirken: Von Beginn der Entwicklung an sollen parallel zur Entwicklung funktionale Tests (Black Box Tests) für die entwickelten Softwarekomponenten definiert werden, welche nach positivem Durchlaufen dieser Tests um strukturelle Tests erweitert werden. Strukturelle Tests erfolgen in detaillierter Kenntnis der Implementierung und dienen dazu, durch das gezielte Auslösen exotischer Ausführungspfade in der Software die Codeabdeckung zu erhöhen und gegebenenfalls nicht erreichbare Ausführungspfade erkennbar zu machen. Aufgrund der in Abschnitt 6.1.2 genannten Besonderheiten der Entwicklung im Vergleich zur herkömmlichen Entwicklung von Desktopanwendungen, wird zur Durchführung der strukturellen Tests ebenfalls auf die von Black Box Tests bekannte Stimulierung durch externe Signale zurückgegriffen.

6.1.2 Softwarequalität

Die Implementierung von Software für eingebette Systeme unterliegt einigen Einschränkungen verglichen mit der herkömmlichen Entwicklung für Desktop Systeme. Architekturunterschiede zwischen Ziel- und Entwicklungsplattform müssen kompensiert werden, in diesem Fall die Entwicklung für die ARMv5-Architektur auf einem x86-System. Der gängige Ansatz ist das Cross Platform Development, welches sich eine vollständige Compiler-Toolchain für das Zielsystem auf dem Entwicklungssystem zu Nutze macht.
Mit einer passenden Toolchain ist es möglich, Binärdateien für das Zielsystem auf dem Entwicklungssystem zu erstellen. In diesem Ansatz fehlt allerdings ein Software-Emulator des Zielsystems, welcher das Ausführen und Testen des Programms auf dem Entwicklungssystem erlauben würde. Zu diesem Zweck kann auf die meisten der Zielsysteme über eine JTAG-Schnittstelle zugegriffen werden, über welche die Kontrolle der CPU-Register und die Kontrolle über den Programmablauf übernommen werden kann. Im Idealfall kommuniziert der Softwaredebugger des Entwicklungssystems über die JTAG-Schnittstelle mit dem Zielsystem.
Da der verfügbare Softwaredebugger für das Zielsystem allerdings von einer proprietären Entwicklungsumgebung und einem proprietären Zielsystem-Betriebssystem abhängig ist, konnte dieser nicht in Zusammenhang mit der Eigenentwicklung eines Betriebssystems der Core-Arbeitsgruppe verwendet werden. Diese Eigenentwicklung eines Betriebssystems implementiert darüber hinaus keinen Server für Remote-Protokolle gängiger Softwaredebugger wie z.B. gdb, so dass effektiv kein Softwaredebugger genutzt werden konnte. Um dieses Problem zu umgehen, wurde der Implementierungsprozess aufgeteilt:

Die erste Phase der Implementierung erfolgte nach der Methode des Test Driven Development auf dem Entwicklungssystem. Für jede Softwarekomponente wurden Testfälle definiert,

welche den Normalbetrieb, Randfälle und Fehlerfälle abdecken. Die Fehlerbehandlung wurde dahingehend vereinheitlicht, dass jede Softwarefunktion einen Pointer auf eine Fehlervariable übergeben bekommt, welche nach dem Aufruf einer Funktion auf komponentenübergreifend eindeutige Fehlercodes überprüft werden muss. Für diese eindeutigen Fehlercodes konnte somit überprüft werden, ob für sie ausreichend viele positive und negative Testfälle definiert wurden. Aus dem Konzept der Softwarekomponenten und den zu bedienenden Testfällen konnte daraufhin eine passende Implementierung abgeleitet werden.
Um die Kompatibilität mit dem Zielsystem vorab sicherzustellen, wurden auf dem Entwicklungssystem ausschließlich Softwarebibliotheken verwendet, welche ebenfalls auf dem Zielsystem vorhanden sind. So wurde z.B. auf dem Entwicklungssystem auf die Verwendung von Funktionen zur dynamischen Allokierung von Speicherbereichen verzichtet, da auf dem Zielsystem die MMU deaktiviert ist.
Um die hardwareseitigen Unterschiede zwischen Entwicklungs- und Zielplattform zu kompensieren, wurden zielsystemspezifische Komponenten auf dem Entwicklungssystem durch Mock-Ups ersetzt, welche ein identisches Black Box Verhalten aufweisen. Beispiele für derartige Mock-Ups sind Ersetzungen von Feldbuscontroller-APIs, Timern und ähnlichen Bestandteilen.

In der zweiten Phase der Entwicklung wurden zuerst die Mock-Ups auf dem Zielsystem in Form ihrer jeweiligen, realen Entsprechung implementiert und auf ein identisches Black Box Verhalten getestet. Daraus konnte in erster Näherung geschlossen werden, dass Implementierungen von Mock-Ups, welche selbst dritte Komponenten stimulieren, nicht zu einem vom Entwicklungssystem abweichenden Systemverhalten führen sollten.
Danach wurden die einzelnen Softwarekomponenten in das Zielsystem integriert. Hierbei wurde jede Komponente mit Eingangsdaten (aus Black Box Sicht) stimuliert, um die vorab entwickelten Testfälle zu bedienen. Dies entspricht aus Sicht des V-Modells den (Software-) Modultests. Das Problem dieser Vorgehensweise ist, dass einige Fehlerklassen auf dem Zielsystem nicht abdeckbar sind: So können Fehler, die auf Hardwaredefekten oder unspezifiziertem Verhalten basieren, nur schwer auf einem intakten Zielsystem ausgelöst werden. Da diese Fehlerklassen allerdings auf dem Entwicklungssystem bereits ausgelöst und deren Behandlung getestet wurde, relativiert sich die Schwere dieser Testlücke auf dem Zielsystem.
Die einzelnen Softwarekomponenten wurden analog zum Datenfluss in der Gatewayanwendung sukzessive implementiert. Dies erlaubte, parallel dazu die Softwareintegration durch Steuergerätetests entsprechend des V-Modells zu testen. Nach Abschluss der Steuergerätetests konnte das Zusammenspiel mehrerer Gateways im Steuergeräteverbundtest überprüft werden.

6.2 Einsatz im Prototypenfahrzeug

Im Rahmen des vom Bundesministerium für Bildung und Forschung unterstützten RECBAR-Projektes erhielt die Core-Arbeitsgruppe der HAW Hamburg Zugriff auf ein Prototypenfahrzeug (VW Golf 7). Das Fahrzeug wurde von der Core-Arbeitsgruppe in Zusammenarbeit mit der IAV GmbH, Berlin mit einer RTE-Netzinfrastruktur und für die Erprobung von RTE im Automobil geeigneter Hardware ausgerüstet. Zu der Erprobungshardware gehören Sensorik wie LIDAR-Scanner und Kameras, Embedded Systeme zur Anbindung der Sensorik an RTE, datenverarbeitende Computer und drei der in dieser Arbeit vorgestellten Kommunikationsgateways. Die RTE-Netzinfrastruktur besteht aus drei miteinander verbundenen RTE-Switches, also drei gekoppelten RTE-Sterntopologien.

Der für diese Arbeit relevante Ausschnitt der Topologie des Prototypenfahrzeugs ist Abbildung 6.1 zu entnehmen: Drei der sieben fahrzeuginternen CAN-Busse sind über in Schreibrichtung blockierende CAN-Dioden mit je einem Kommunikationsgateway verbunden. Aufgrund der Verwendung der CAN-Dioden ist garantiert, dass die Kommunikation der fahrzeuginternen CAN-Busse physikalisch gegen Störung und Manipulationen abgesichert ist.

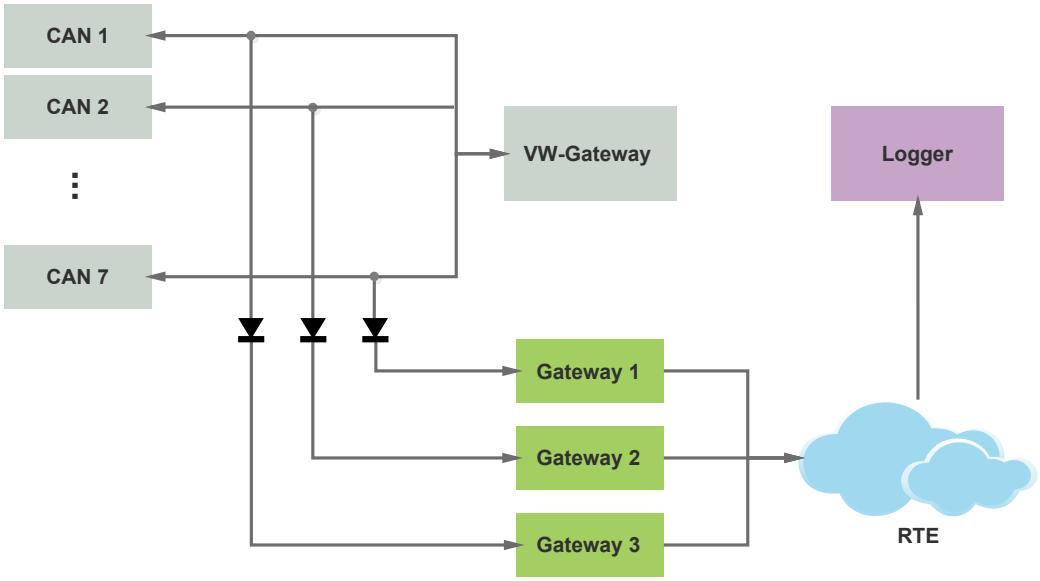

Abbildung 6.1: Topologiediagramm: Anbindung von drei der sieben CAN-Busse des VW Golf 7 über CAN-Dioden an drei Gateways. Zweck ist die Weiterleitung aller CAN-Nachrichten der drei Busse über RTE an einen Logging-PC.

Die Kommunikationsgateways haben die Aufgabe, alle empfangenen CAN-Nachrichten gebündelt über RTE an einen Logging-PC weiterzuleiten, der die Nachrichten in einer Datenbank zur späteren Analyse speichert. Zeitliche Analysen werden dadurch ermöglicht, dass jede CAN-Nachricht innerhalb der CAN-ISR des Kommunikationsgateways mit einem global gültigen Zeitstempel angereichert wird. Durch diesen Empfangszeitstempel können die Nachrichten aller CAN-Busse innerhalb der Datenbank in einen zeitlichen Kontext gesetzt werden.

6.3 Evaluation

Dieser Abschnitt ist der Auswertung und Bewertung des zeitlichen Verhaltens des Kommunikationsgateways gewidmet. Die Auswertung hat unterschiedliche Ziele: Zum einen legt die Untersuchung des zeitlichen Verhaltens des Kommunikationsgateways zeitlich kritische Pfade innerhalb der Software offen, zum anderen werden Verbesserungsmöglichkeiten bezüglich der Implementierung und Konfiguration ersichtlich.
Die Bewertung des zeitlichen Verhaltens ist allerdings nur in Kenntnis des zeitlichen Verhaltens des zentralen VW-Kommunikationsgateways möglich. Nur durch einen Vergleich der Eigenschaften beider Gateways ist die Abschätzung, ob das in dieser Arbeit entwickelte Kommunikationsgateway das VW-Gateway ersetzen könnte, sinnvoll möglich.
In den folgenden Abschnitten wird zuerst das Zeitverhalten des im Rahmen dieser Arbeit entwickelten Kommunikationsgateways ermittelt, um dieses Zeitverhalten dann mit dem errechneten und gemessenen Zeitverhalten des VW-Gateways zu vergleichen.

6.3.1 Zeitverhalten des eigenen Gateways

Jede Softwarekomponente des Kommunikationsgateways übernimmt eine gegen andere Komponenten abgegrenzte Teilaufgabe der Gatewaylogik und ist aufgrund der modularen Architektur leicht gegen alternative Implementierungen ausgetauschbar. Daher ist es sinnvoll, eine zeitliche Untersuchung des Kommunikationsgateways auf der Ebene der Softwarekomponenten durchzuführen.
Das zeitliche Verhalten kann durch weitere Informationen bezüglich der verwendeten Hardware in einen Kontext gesetzt werden. Als Hardwareplattform für alle Zeitmessungen wurde ein Embedded System auf Basis eines 32-Bit ARM 926EJ-S CPU verwendet. Die CPU wird bei 200 MHz mit einem von der Core-Arbeitsgruppe entwickelten Betriebssystem, deaktivierter MMU und deaktiviertem Data Cache betrieben.

6.3.1.1 Laufzeit von ISR und Mapping

Innerhalb der CAN- und RTE-ISR wird geprüft, ob eine eingehende Nachricht vom Kommunikationsgateway behandelt werden muss, um gegebenenfalls einen Job entsprechend des „IJob“-Interfaces in der JobQueue abzulegen. Dazu wird über die Menge an Verarbeitungsvorschriften iteriert, um eine oder mehrere Verarbeitungsvorschriften zu finden, deren Nachrichtenidentifikationskriterium zu der eingehenden Nachricht passt.
Die per Oszilloskop ausgemessene Laufzeit der ISR inklusive Mapping bei variierender Anzahl von Verarbeitungsvorschriften ist in Abbildung 6.2 dargestellt.

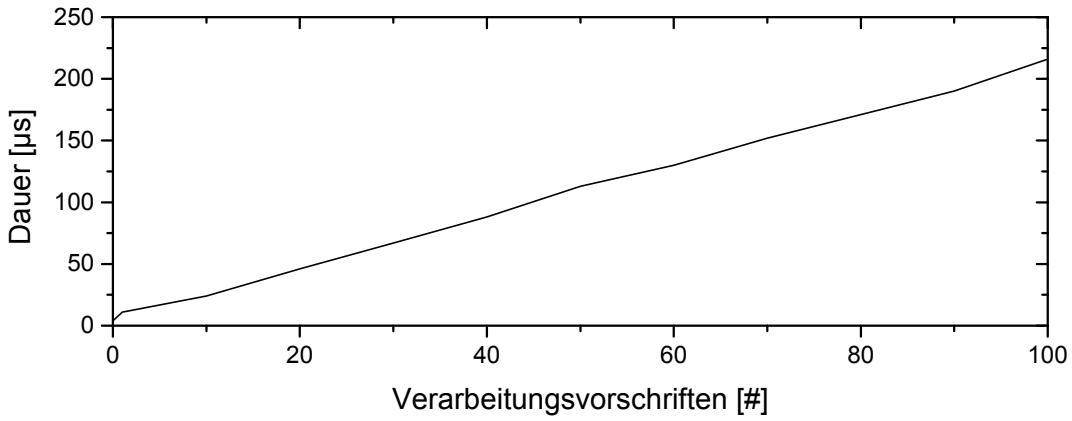

Abbildung 6.2: Laufzeitverhalten: Veränderlichkeit der Laufzeit der ISR inklusive Mapping in Relation zur Anzahl der zu überprüfenden Verarbeitungsvorschriften bei Speicherung in einem flachem Array.

Das Laufzeitverhalten des Mappings wird also maßgeblich durch die Anzahl der konfigurierten Verarbeitungsvorschriften und die zur Speicherung der Verarbeitungsvorschriften verwendete Datenstruktur beeinflusst. So ist die Iteration über ein Array von Verarbeitungsvorschriften mit einem Aufwand von O(n) für große n suboptimal: Prinzipiell ist es in Echtzeitsystemen wünschenswert, die Laufzeit von ISR möglichst klein zu halten, damit die durch die ISR unterbrochenen Echtzeittasks mit der im Schedule reservierten Zeit auskommen können.
Eine vielversprechende Alternative für größere Anzahlen an Verarbeitungsvorschriften sind binäre Suchbäume. Eine Eigenschaft der Konfiguration, namentlich der Umstand, dass sich die Menge der Verarbeitungsvorschriften zur Laufzeit nicht verändert, führt dazu, dass der Aufwand von Lösch- und Einfügeoperationen eines binären Suchbaums nicht beachtet werden muss. Die in binären Suchbäumen verwendeten Schlüssel (Knoten) können durch das Nachrichtenidentifikationskriterium der Verarbeitungsvorschriften gebildet werden: Die notwendige

totale Quasiordnung - also die durchgängige Vergleichbarkeit zweier Schlüssel durch „größer gleich“- bzw. „kleiner gleich“-Operationen - wäre z.B. für das Nachrichtenidentifkationskriterium „CAN-Nachrichten-ID“ leicht zu bilden.
Durch geschickte Wahl und Anordnung von Wurzel und Schlüsseln des binären Suchbaums kann ein verbesserter binärer Suchbaum erstellt werden, dessen Zweige eine nur minimal voneinander abweichenden Abstand zur Wurzel aufweisen. In einem solchen höhenbalancierten Suchbaum würde sich der Suchaufwand auf O(h) reduzieren, mit h gleich der Höhe des binären Suchbaums.
Unter der Voraussetzung, dass zur Entwurfszeit die Zugriffswahrscheinlichkeiten auf einzelne Verarbeitungsvorschriften aus der Kommunikationsmatrix abgeleitet werden können, kann versucht werden, einen gewichtsbalancierten Suchbaum zu implementieren. Der Aufwand zur Suche innerhalb eines gewichtsbalancierten binären Suchbaums variiert zwischen O(log n) und O(n), wobei der schlechtere Fall O(n) nur bei besonders ungünstiger Verteilung der Zugriffswahrscheinlichkeit erreicht wird. Abbildung 6.3 stellt den erwarteten zeitlichen Aufwand unter Verwendung eines binären Suchbaums mit O(log n) in Bezug zum gemessenen zeitlichen Aufwand unter Verwendung eines flachen Arrays.

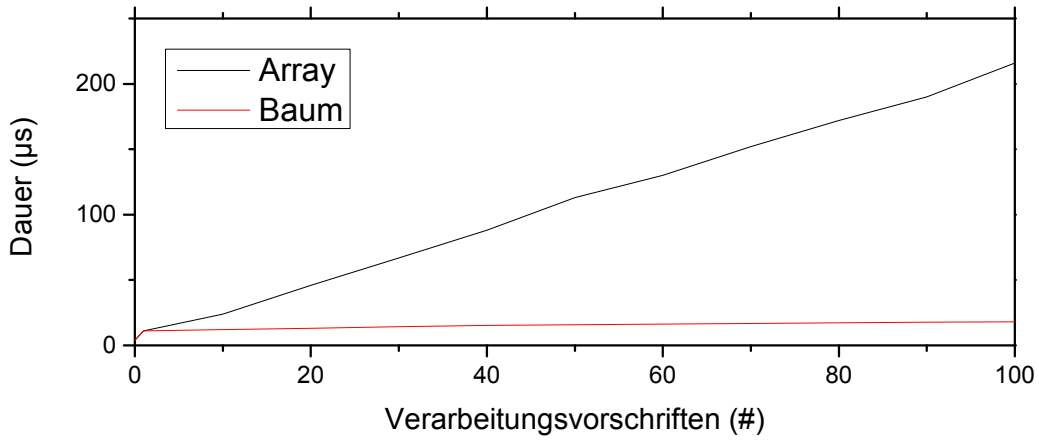

Abbildung 6.3: Laufzeitverhalten: Veränderlichkeit der Laufzeit der ISR inklusive Mapping in Relation zur Anzahl der zu überprüfenden Verarbeitungsvorschriften bei Speicherung in einem flachem Array bzw binären Suchbaum.

Letztendlich ist somit die konkrete Implementierung der Datenstruktur zur Speicherung der Verarbeitungsvorschriften anwendungsfallabhängig und sollte somit erst nach sorgfältiger Untersuchung gewählt werden.

6.3.1.2 Laufzeit des Parsers

Die Parserkomponente des Kommunikationsgateways analysiert eine, in einem Job entsprechend des „IJob“-Interface gespeicherte, eingegangene Nachricht, um aus der Menge der anwendbaren Verarbeitungsvorschriften die Bestmögliche zu identifizieren und die dazugehörigen Bytebereiche für die weitere Verarbeitung bereitzustellen. Die Identifikation der bestmöglichen Verarbeitungsvorschrift basiert auf der Auswertung der potentiell in der Verarbeitungsvorschrift enthaltenen Bedingungen.
Somit müssen für jede anwendbare Verarbeitungsvorschrift die durch Bedingungen auszuwertenden Bytebereiche bereitgestellt werden und daraufhin die Bedingungen validiert werden. Daraus ergeben sich drei, jeweils in die Laufzeit eingehende, Charakteristika, welche die Gesamtlaufzeit des Parsers beeinflussen: die Anzahl an Verarbeitungsvorschriften, die Anzahl der in den Verarbeitungsvorschriften definierten Bytebereiche und die Anzahl der auszuwertenden Bedingungen.
Um die Laufzeit des Parsers zu messen, wurde das Kommunikationsgateway anhand von CAN-Nachrichten mit 8 Byte Nutzlast stimuliert. Die dazugehörige Konfiguration definiert Permutationen von 1 bis 8 Bytebereichen, 1 bis 8 Bedingungen und 1 bis 8 Verarbeitungsvorschriften. Die Abdeckung unterschiedlicher Anzahlen von Bedingungen ist für Bedingungsanzahlen größer 4 hochgerechnet, da die Überprüfung einer Bedingung ein nicht-schwankendes, für alle Bedingungen konstantes Laufzeitverhalten aufweist. Abbildung 6.4 visualisiert den linearen Einfluß der die Laufzeit maßgeblich beeinflussenden Charakteristika.

6.3.1.3 Laufzeit der Ausgabepufferverwaltung

Es ist die Aufgabe der Ausgabepufferverwaltung, einen Pointer auf einen Speicherbereich zurückzugeben, in den eine ausgehende Nachricht geschrieben werden soll. Diese Aufgabenstellung ist für zwei von drei Anwendungsfällen trivial: Der Speicherbereich für ausgehende CAN-Nachrichten liegt auf dem Stack der Ausgabefunktion und läßt sich somit leicht referenzieren. Der Speicherbereich innerhalb des Fragmentierungsspeichers für in Richtung RTE gebündelt auszugebende Nachrichten ist statisch alloziert und global durch Wahl der Indizes eines multidimensionalen Arrays ebenfalls leicht zu ermitteln.
Der im Sinne des Laufzeitverhaltens kritische Pfad ist das Ermitteln eines Speicherbereichs für RTE-Nachrichten, die durch den Echtzeitsendetask verschickt werden sollen. Die Betriebssystem-API bietet Funktionen an, welche diese Aufgabe übernehmen und verdeckt dadurch den damit verbundenen, nicht unerheblichen Aufwand:

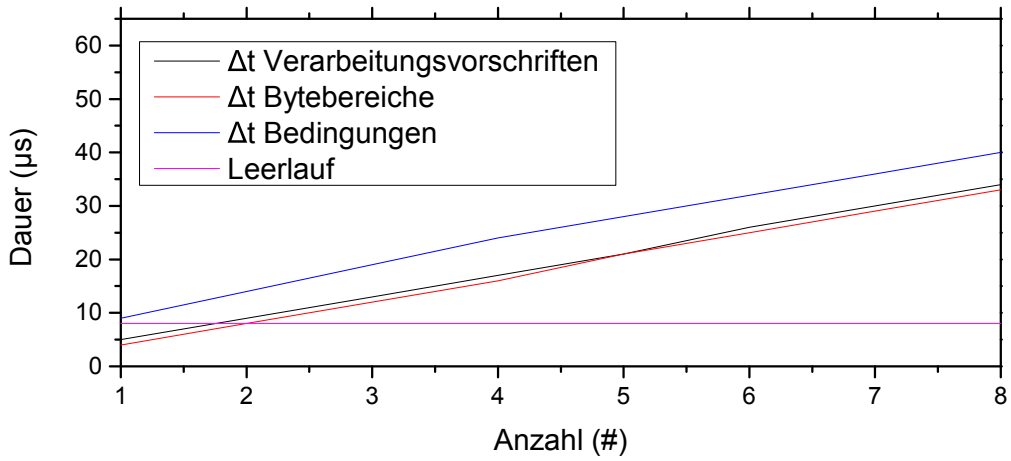

Abbildung 6.4: Laufzeitverhalten: Lineare Veränderlichkeit der Laufzeit des Parsers, jeweils durch die Einflußgrößen „Anzahl an Verarbeitungsvorschriften“, „Anzahl definierter Bytebereiche“ und „Anzahl an Bedingungen“. Die in jedem Fall vorhandene Laufzeit wird durch „Leerlauf“ repräsentiert.

Neben dem RTE-PHY verwendet die ARM-CPU konfigurierbare Hardware-MAC-Controller, die unterschiedlichen Bussen zugeordnet werden können. Diese MAC-Controller verwenden internen ARM-Speicher, der nicht von der Anwendung kontrolliert wird und über hardwaregestützte Pointer-FIFOs mit dem Betriebssystem verbunden ist. Dabei existieren unterschiedliche FIFOs zur Verwaltung von leerem Speicherbereich, angefordertem Speicherbereich, verwendetem Speicherbereich und Speicherbereich zur Verwaltung von Sendebestätigungen. Während der MAC-Controller per Direct Memory Access (DMA) auf den internen ARM-Speicher zugreifen kann, muss die Anwendung bzw. das Betriebssystem für jede Speicherverwaltungsoperation den Umweg über FIFOs und MAC-Controller gehen.
Die aus 200 Testfällen ermittelte Schwankungsbandbreite der Dauer, um über einen Betriebssystem-API-Aufruf einen Speicherbereich für RTE-Nachrichten zu erhalten, beträgt ca. 5 µs im Intervall von 80 µs bis 85 µs. Die Testfälle sind in Abbildung 6.5 dargestellt. Die isolierte Betrachtung der Dauer dieser Betriebssystem-API-Aufrufe dient dem Zweck, diese Dauer aus weiteren Messungen herauszurechnen und um das Laufzeitverhalten genauer betrachten zu können.

6.3.1.4 Laufzeit der Übersetzung

Der Übersetzungsvorgang, also die Anwendung der in der Verarbeitungsvorschrift enthaltenen Übersetzungsvorschrift entsprechend der IDL, besteht aus unterschiedlichen Einzelschritten, wie z.B.: Ermittlung der Größe der auszugebenden Nachricht, Akquise des Speicherbereiches

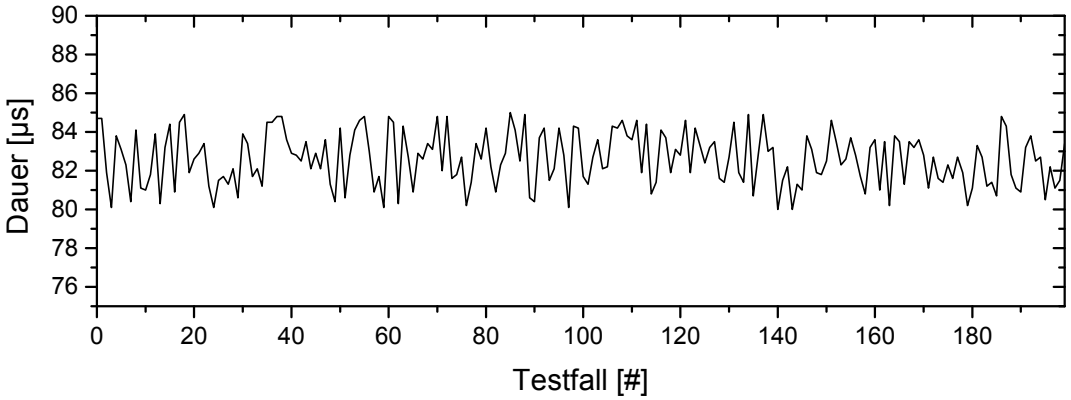

Abbildung 6.5: Laufzeitverhalten: Dauer eines Betriebssystem-API-Aufrufs, um den Speicherbereich für eine ausgehende RTE-Nachricht zu erhalten.

der auszugebenden Nachricht, Einfügen der Header des Transportprotokolls (falls notwendig), Einfügen der Header des Bündelungsprotokolls (falls notwendig), Kopieren von Teilen der zu verarbeitenden Nachricht und Ausführen von in der IDL konfigurierten Funktionen.
Die Übersetzung ist zwangsweise mit der in Abschnitt 6.3.1.3 beschriebenen Ausgabepufferverwaltung verknüpft, welche die Akquise des Speicherbereichs für die auszugebende Nachricht übernimmt. Die hohe zeitliche Belastung für die Akquise eines Speicherbereiches für auszugebende RTE-Nachrichten führt auch im Zuge der Auswertung der Übersetzung zu einer Fallunterscheidung: Für gebündelt auszugebende Nachrichten entfällt der Overhead der Ausgabepufferverwaltung, da derartige Nachrichten in den Fragmentierungsspeicher geschrieben werden. Nicht gebündelt zu übertragende Nachrichten unterliegen also einem zeitlichen Malus in der Verarbeitungsdauer.
Die Dauer der Übersetzung wird von der durch die Ausgabepufferverwaltung verbrauchten Zeit und der Größe [Bytes] der zu kopierenden Daten der eingegangenen Nachricht bestimmt. In eine RTE-Nachricht können theoretisch zwischen 0 Bytes und 1500 Bytes kopiert werden. Daher wurde die Dauer der Übersetzung in Abhängigkeit von dieser zu kopierenden Datenmenge gemessen. In Abbildung 6.6 wird die Verarbeitungszeit von nicht gebündelt zu übertragenden Nachrichten dargestellt. Die potentielle Zeiteinsparung durch Umgehung der Betriebssystem-API-Aufrufe zur Speicherbereichsakquise bei gebündelter Nachrichtenübertragung ist durch ein 82 µs breites Band unterhalb der Messkurve verdeutlicht. Diese 82 µs sind der Durchschnittswert der in Abbildung 6.5 dargestellten Laufzeit der Ausgabepufferverwaltung.

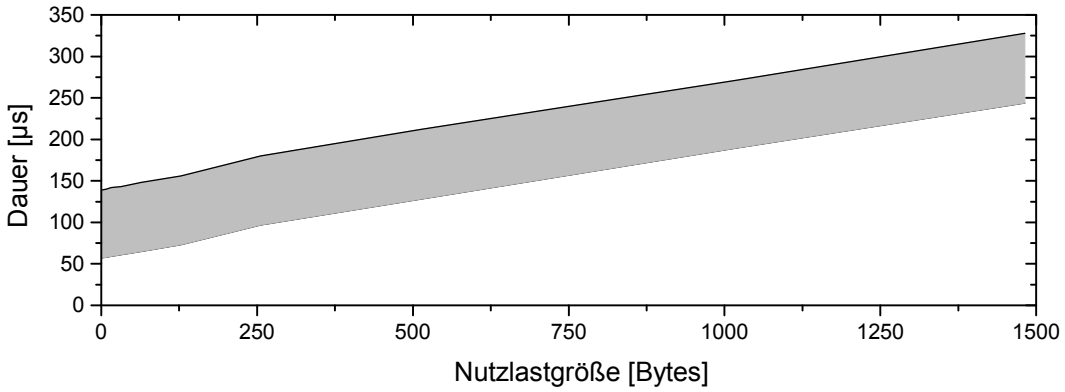

Abbildung 6.6: Laufzeitverhalten: Dauer der Übersetzung nicht gebündelt zu übertragender Nachrichten entsprechend der IDL - inklusive zeitlichem Overhead durch Ausgabepufferverwaltung. Das grau markierte Band unterhalb der Messkurve repräsentiert den zeitlichen Overhead der Ausgabepufferverwaltung.

6.3.1.5 Laufzeit der Gatewaylogik

Die Laufzeit der gesamten Gatewaylogik besteht aus den Laufzeiten von ISR, Mapping und Job-Task. Der Job-Task setzt sich wiederum aus den Laufzeiten des Parsers, der Ausgabepufferverwaltung und der Übersetzung zusammen. Die Summe dieser Einzellaufzeiten definiert die maximale, theoretisch zu verarbeitende CAN-Nachrichtenfrequenz.
Die Laufzeit muss bekannt sein, da diese Laufzeit wesentlich für das Erstellen eines Task-Schedules innerhalb des Gateways ist: Für jede zu verarbeitende Nachricht muss ein ausreichend großes Zeitfenster im Schedule vorhanden sein.
Die Laufzeit des Job-Tasks kann mit den bisher vorgestellten Daten für einen bestimmten Anwendungsfall gemessen werden. Zur Veranschaulichung wird folgender Anwendungsfall gewählt:

- Es gehen nur CAN-Nachrichten mit 8 Byte Nutzlast ein.
- Nur CAN-Nachrichten mit der ID 0xFF sollen über RTE weitergeleitet werden.
- Es soll die komplette Nutzlast unmanipuliert übertragen werden.
- Es werden Protokoll-Header für TPU eingefügt.
- Es findet keine Nachrichtenbündelung statt.
- Es wurde nur eine Verarbeitungsvorschrift konfiguriert.

Aufgrund der zeitlichen Unabhängigkeit des IRQ und des Starts des Job-Tasks ist es schwierig, diese ISR und Job-Task gemeinsam auszumessen. Daher wurde zur Messung der Laufzeit der gesamten Gatewaylogik ein Task-Schedule erstellt, der einen Nachrichtengenerator-Task enthält. Dieser Nachrichtengenerator-Task hat ein zur CAN-ISR identisches Verhalten und wird unter Reservierung einer pessimistischen Laufzeit in den Task-Schedule eingefügt. Aufgrund der zeitlichen Ordnung innerhalb des Task-Schedules können Nachrichtengenerator- und Job-Task einzeln zeitlich erfasst werden, um die Gesamtlaufzeit zu ermitteln.

Das in Abbildung 6.7 präsentierte Messergebnis zeigt eine nahezu konstante Laufzeit der Gatewaylogik im konkreten Anwendungsfall auf, welche nur durch gelegentliche Ausreißer aufgrund verfehlter Zugriffe auf den Instruction Cache getrübt wird. Der reguläre, nicht vom Instruction Cache beeinflußte Jitter bewegt sich in einem Intervall von 2 µs um die Laufzeit von 216 µs. Der vom Instruction Cache verursachte Jitter erhöht die Worst Case Laufzeit der Gatewaylogik auf 235 µs. Es läßt sich nicht vorhersagen, welche Auswirkungen die Nutzung einer anderen Hardwareplattform auf die Anzahl und Intensität der zeitlichen Ausreißer hat.

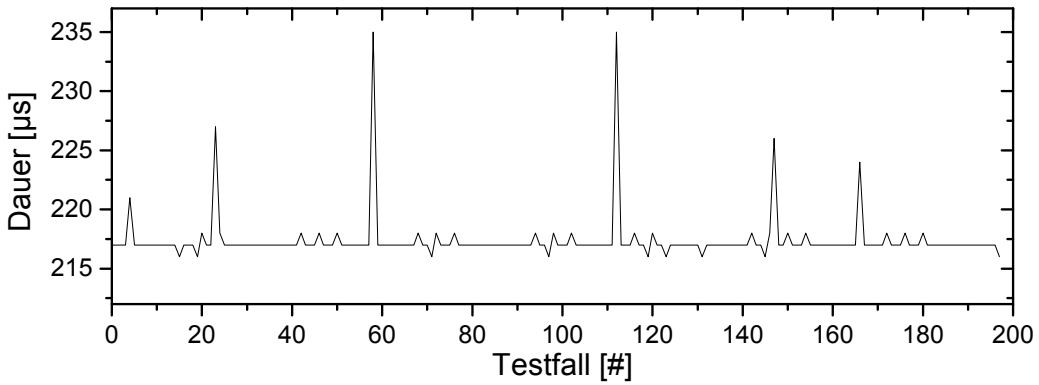

Abbildung 6.7: Laufzeitverhalten: Jitter der Gatewaylogik im Anwendungsfall entsprechend der Beschreibung in Abschnitt 6.3.1.5.

6.3.1.6 Zeitliche Stabilität der Gatewaykommunikation

Die zeitliche Stabilität der Übertragung von CAN-Nachrichten über den RTE Backbone hat ein besonderes Gewicht in der Echtzeitübertragung zeitkritischer CAN-Nachrichten: Eine zu hohe Schwankungsbreite der gesamten Übertragungslatenz würde erfordern, dass die Zeitgarantien der CAN-Nachrichten besonders schwach formuliert werden müssten. Ab einer gewissen Schwankungsbreite ginge der Echtzeitcharakter der Nachrichtenübertragung verloren.

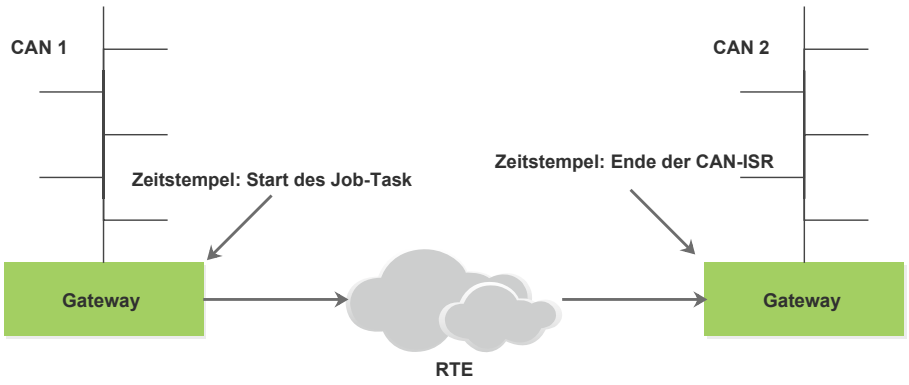

Abbildung 6.8: Versuchsaufbau: Die Übertragungslatenz einer CAN-Nachricht mit 8 Byte Nutzlast wird in 200 Durchläufen gemessen. Der Jitter der Zeitdifferenz zwischen den Zeitstempeln veranschaulicht die Stabilität der Übertragung.

Zur Messung des Jitters der Übertragungslatenz zwischen zwei Kommunikationsgateways wurde der in Abbildung 6.8 gezeigte Versuchsaufbau verwendet. Die Zeitpunkte, zu denen Start und Ende der Übertragungslatenz gemessen werden, sind so gewählt, dass alle Komponenten der Gatewayanwendung durchlaufen werden und dass die Wahl eines geschickten oder suboptimalen Task-Schedules keinen Einfluss auf den Jitter hat: Ein Start der Messung in der Quell-Gateway-ISR würde z.B. den zeitlichen Abstand zwischen CAN-IRQ und Start des Job-Task mit in den Jitter einfließen lassen.

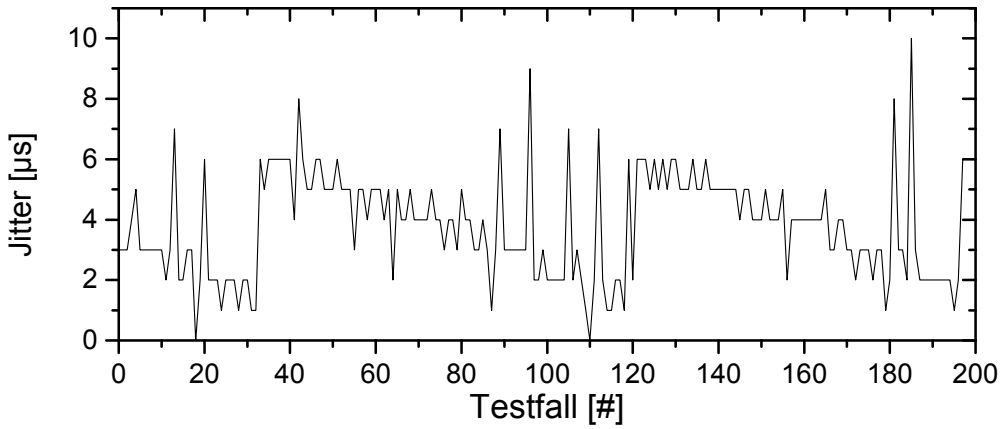

Abbildung 6.9: Laufzeitverhalten: Jitter der Übertragungslatenz der CAN-Nachrichtenübertragung über den RTE Backbone entsprechend des Versuchsaufbaus aus Abbildung 6.8.

Das Abbildung 6.9 zu entnehmende Messergebnis zeigt eine verhältnismäßig konstante Übertragungslatenz, die einen Jitter im Rahmen von bis zu 10 µs aufweist. Es ist einfach zu erklären, warum der Jitter der Übertragungslatenz zwischen 2 Gateways kleiner als der Jitter der Gatewaylogik selbst ist: Dadurch, dass der Job-Task zeitlich vom RTE-Sendetask entkoppelt ist (siehe Abbildung 6.10), und dadurch, dass ein Task-Schedule Zeitreserven für den Jitter des Job-Task enthalten kann, ist die Ende-zu-Ende-Übertragungslatenz zeitlich vom Job-Task entkoppelt.

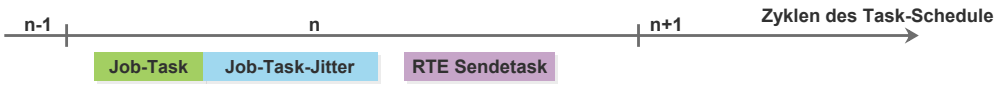

Abbildung 6.10: Veranschaulichung der Entkopplung des Job-Task-Jitters vom gesamten Kommunikationsjitter anhand von Zeitreserven hin zum RTE-Sendetask innerhalb eines fiktiven, stark vereinfachten Task Schedule.

6.3.1.7 Bewertung der gemessenen Laufzeiten

Die Bewertung der gemessenen Laufzeiten kann aus unterschiedlichen Sichten durchgeführt werden. Interessant ist z.B., die Leistungsfähigkeit des Kommunikationsgateways in Bezug zu einem voll ausgelasteten CAN-Bus zu setzen. Darüber hinaus wird aus den Laufzeiten der einzelnen Softwarekomponenten ersichtlich, wo Optimierungsbedarf besteht.
Ein CAN-Bus mit erweiterter 29-Bit Nachrichten-ID, wie er einem VW Golf 7 verwendet wird, verfügt über eine Bruttobandbreite von 500 $^{\text{kBaud}}/_{\text{s}}$. Um die minimale und maximale Nachrichtenfrequenz zu ermitteln, müssen die Längen der einzelnen CAN-Protokollfelder bekannt sein, der Mechanismus „Bit Stuffing“ einbezogen werden und dem Interframe Space von mindestens 3 Bits Rechnung getragen werden.
Wie in Abschnitt 2.1.2 geschildert, verwendet das CAN-Protokoll Bit Stuffing, damit Steuergeräte das Bit Timing weniger präzise bestimmen können müssen. Die Anzahl eingefügter Bit Stuffing Bits kann entweder für jede Nachricht einzeln ermittelt, oder aber pauschal für das Worst Case Szenario ermittelt werden. Das Worst Case Szenario tritt dann ein, wenn sowohl die CAN-Nachrichten-ID als auch eine maximal große CAN-Nutzlast (8 Bytes) ausschließlich aus identischen Bits bestehen. Für Nachrichten mit 29-Bit Nachrichten-ID ergibt sich in diesem Worst Case Szenario eine maximale Anzahl von 5 + „Anzahl der Nutzlast-Bytes“ Bit Stuffing Bits. Wenn zusätzlich noch die Prüfsumme der CAN-Nachricht aus identischen Bits besteht, erhöht sich die maximale Anzahl auf 8 + „Anzahl der Nutzlast-Bytes“ Bit Stuffing Bits.

Anhand dieser Informationen kann die minimale und maximale Länge einer CAN-Nachricht inklusive ihres nachfolgenden Interframe Space berechnet werden (siehe Tabelle 6.1). Auf einem 500 $^{kBaud}/_{s}$ CAN-Bus kann alle 2 µs ein Bit übertragen werden. Daraus folgt, dass bei minimaler Nachrichtengröße ohne Bit Stuffing eine maximale Nachrichtenfrequenz von 7812 $^{Nachricht}/_{s}$ erreicht wird. Die Übertragungsdauer dieser Nachrichten (ohne Interframe Space) beträgt 122 µs.
Analog dazu läßt sich die minimale Nachrichtenfrequenz bei maximal großen Nachrichten und Worst Case Bit Stuffing errechnen: Hier liegt die Nachrichtenfrequenz auf einem 500 $^{kBaud}/_{s}$ CAN-Bus bei 3401 $^{Nachricht}/_{s}$ und die Übertragung einer Nachricht dauert (ohne Interframe Space) insgesamt 294 µs.

Tabelle 6.1: Länge einer CAN-Nachricht und ihre Zusammensetzung (min, max)

Feldname / Ursache	**Länge [Bit]**
Start Of Frame (SOF)	1
Identifier A	11
Substitute Remote Request (SRR)	1
Identifier Extension Bit (IDE)	1
Identifier B	18
Remote Transmission Request	1
Reserved Bits	2
Data Length Code (DLC)	4
Data Field	0 - 64
Checksum (CRC)	15
Checksum (CRC) Delimiter	1
ACK Slot	1
ACK Delimiter	1
End Of Frame (EOF)	7
Bit Stuffing	0 - 16
Interframe Space	3
Summe (min)	67
Summe (max)	147

Die Übertragungsdauer einer CAN-Nachricht von 122 µs + 6 µs Interframe Space bei maximaler Nachrichtenfrequenz offenbart, dass das Kommunikationsgateway nicht in der Lage ist, diese Nachrichtenmenge zu verarbeiten. Die in Abschnitt 6.3.1.5 genannte Laufzeit der Gatewaylogik in Höhe von 235 µs ist fast doppelt so groß wie die Übertragungsdauer auf dem CAN-Bus.
Unter der vereinfachenden Annahme, dass von der Laufzeit der Gatewaylogik bei Verwendung von Nachrichtenbündelung pauschal die in Abschnitt 6.3.1.3 dokumentierte Laufzeit der Ausgabepufferverwaltung in Höhe von mindestens 80 µs abgezogen werden könne, reduziert sich die Laufzeit der Gatewaylogik auf 155 µs - ein Wert, der schon deutlich näher an der minimalen Übertragungsdauer einer CAN-Nachricht liegt.
Es kommen zwei weitere Faktoren begünstigend zur Bewertung der gemessenen Laufzeit hinzu. Zum einen wird in der Automobilindustrie ein CAN-Bus mit einer Bandbreitenreserve von bis zu 50 % betrieben. Leider lässt sich dieser Wert aufgrund fehlender akademischer Quellen nicht weiter präzisieren. Zum anderen ist es mehr als unwahrscheinlich, dass über einen automobilen CAN-Bus nur minimal kleine Nachrichten übertragen werden. Vielmehr haben Auswertungen der mitgeschnittenen Kommunikation des Prototypenfahrzeugs (siehe Abschnitt 6.2) ergeben, dass über 97 % der CAN-Nachrichten eine Nutzlastgröße von 8 Byte aufweisen.
Sobald demnach CAN-Nachrichten mit einer Nutzlast von 8 Byte maximal alle 294 µs - 32 µs (16 Bit Stuffing Bits) = 262 µs bei 100 % Busauslastung am Kommunikationsgateway eingehen, ist das Kommunikationsgateway in der Lage, die Gatewaylogik (235 µs) für eine Nachricht zu durchlaufen, bevor die nachfolgende Nachricht eintrifft. Durch die Verwendung der Nachrichtenbündelung können sogar Rechenzeitreserven gebildet werden.

6.3.2 Untersuchung des Zeitverhaltens des VW-Gateways

Die Tauglichkeit des im Rahmen dieser Arbeit entwickelten Kommunikationsgateways zum längerfristigen Einsatz außerhalb der Laborumgebung wurde durch den Einsatz im Prototypenfahrzeug (siehe Abschnitt 6.2) validiert. Abseits von der allgemeinen Praxistauglichkeit muss ebenso untersucht werden, ob das Kommunikationsgateway in der Lage ist, die momentan im Prototypenfahrzeug durch das zentrale VW-Gateway erledigten Aufgaben zu übernehmen.
Da das VW-Gateway nur auf einer Black Box Ebene analysiert werden kann, können die Laufzeiten der einzelnen Verarbeitungsschritte des Nachrichtenaustauschs zwischen zwei CAN-Bussen nicht detailliert untersucht werden. Es ist nur möglich, die gesamte Verarbeitungszeit des VW-Gateways zu ermitteln. Zu diesen Zweck können zwei unterschiedliche, sich ergänzende, Ansätze gewählt werden: Zum einen kann über die CAN-Nachrichtendatenbank des Logging-PCs (siehe Abschnitt 6.2) ermittelt werden, mit welchem zeitlichen Offset eine auf dem CAN-Bus „x“ sichtbare Nachricht auf dem CAN-Bus „y“ erscheint. Zum anderen kann

diese rechnerisch ermittelte Verarbeitungsdauer des VW-Gateways durch eine Messung des zeitlichen Offset mit einem Oszilloskop validiert werden.
Die folgenden Abschnitte beschäftigen sich mit dem Berechnen, Messen und Auswerten der Verarbeitungszeit des VW-Gateways. Zur Abgrenzung zwischen dem VW-Gateway und dem im Rahmen dieser Arbeit entwickelten Gateway werden diese im Folgenden „VW-Gateway“ bzw. „Core-Gateway“ genannt.

6.3.2.1 Untersuchung anhand der CAN-Nachrichtendatenbank

Die CAN-Nachrichtendatenbank speichert sämtliche auf den an die Core-Gateways angeschlossenen CAN-Busse in Tabellen, welche (mindestens) Spalten für die CAN-Nachrichten-ID, Nutzlastlänge [Byte], Nutzlast und einen Zeitstempel enthalten. Der für jede in einer Tabelle gespeicherte CAN-Nachricht vorgehaltene Zeitstempel besteht aus einem 32-Bit Zählerwert des Core-Gateways in Mikrosekunden-Auflösung. Dieser Zählerwert ist über alle Gateways synchronisiert und repräsentiert eine globale Zeit.
Durch Vergleiche der Zeitstempel unterschiedlicher Datenbankeinträge kann somit der zeitliche Abstand zwischen zwei Nachrichten ermittelt werden. Relevant dafür ist ebenfalls der Zeitpunkt im Nachrichtenverarbeitungsprozess, an welchem der Zeitstempel im Core-Gateway aufgenommen wird: Der Zeitstempel wird jeweils innerhalb der CAN-ISR aufgenommen. Das hat zur Konsequenz, dass eine Nachricht komplett auf den Bus ausgetaktet wurde, bevor sie einen Zeitstempel erhält. Der Zeitstempel spiegelt also den Sendezeitpunkt des CAN-Controllers oder des VW-Gateways plus „Dauer des Austaktens“ wider (siehe auch, Abbildung 6.11).
Aus der Datenbank werden in einem ersten Ansatz Nachrichten isoliert, die auf mindestens zwei Bussen gesichtet wurden. Aufgrund der in VW-CAN-Bussen global eindeutigen Nachrichten-IDs kann somit gefolgert werden, dass diese Nachrichten das VW-Gateway passiert haben müssen.
Um die tatsächliche Verarbeitungszeit des VW-Gateways zu ermitteln, muss die im Folgenden beschriebene Fehlinterpretation ausgeschlossen werden: Aufgrund des zyklischen Sendeverhaltens der CAN-Controller dürfen nicht die Zeitstempel zweier Nachrichten verglichen werden, welche in unterschiedlichen Zyklen auf den jeweiligen Bussen erscheinen. Dadurch könnte die errechnete VW-Gateway-Verarbeitungszeit von der tatsächlichen Verarbeitungszeit um den Faktor „n * Zykluslänge“ abweichen.
Zu diesem Zweck ist eine CAN-Nachricht mit einem offensichtlichen Zyklus zu ermitteln, welche in jedem Zyklus unterschiedliche Nutzlasten aufweist. Dadurch lassen sich in der Datenbank Nachrichten derselben Nachrichten-ID anhand ihrer Nutzlast voneinander unterscheiden.

Unter der in der Praxis sinnvollen Vermutung, dass die Verarbeitungszeit des VW-Gateways kleiner als die Zykluslänge einer solchen Nachricht ist, können somit zwei Nachrichten auf unterschiedlichen CAN-Bussen in eine eindeutige zeitliche Relation gesetzt werden.
Für unterschiedliche Nachrichten konnten anhand dieser Vorgehensweise zeitliche Differenzen zwischen den Zeitstempeln im Intervall von 280 µs (für Nachrichten mit 4 Byte Nutzlast) bis 650 µs (für Nachrichten mit 8 Byte Nutzlast) ermittelt werden. Diese Zeitdifferenzen müssen unter Einbeziehung des Versuchsaufbaus interpretiert und weiterverarbeitet werden (siehe Abbildung 6.11).

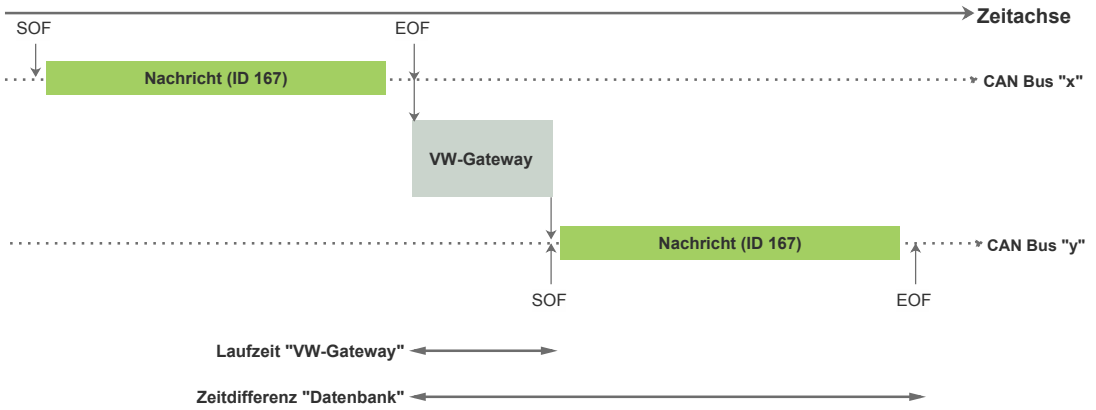

Abbildung 6.11: Versuchsaufbau: Zeitdifferenzen aus der Datenbank entsprechen dem EOF-zu-EOF-Intervall. Die tatsächliche Verarbeitungszeit des VW-Gateways entspricht dem innenliegenden EOF-zu-SOF-Intervall.

Die minimale Differenz der Datenbankzeitstempel von 280 µs für Nachrichten mit 4 Byte Nutzlast muss anhand der in Abschnitt 6.3.1.7 vorgestellten Informationen um die Übertragungsdauer einer solchen Nachricht auf dem CAN-Bus bereinigt werden. Dazu wird die Länge [Bit] solcher Nachrichten ermittelt um daraus anhand des Bit Timing die Übertragungsdauer zu errechnen:

96 Bit (Nachricht) + 12 Bit (Bit Stuffing) = 108 Bit = Nachrichtenlänge

2 $\mu s/Bit$ * 108 Bit = 216 µs = Übertragungsdauer bei 500 $kBaud/s$

280 µs - 216 µs = 64 µs = Verarbeitungsdauer im VW-Gateway

Analog dazu muss die maximale Differenz der Datenbankzeitstempel von 650 µs für Nachrichten mit 8 Byte Nutzlast ebenfalls bereinigt werden. Die Bereinigung bedient sich wieder der in Abschnitt 6.3.1.7 genannten Informationen zur Berechnung der Nachrichtenlänge und Übertragungsdauer:

128 Bit (Nachricht) + 16 Bit (Bit Stuffing) = 144 Bit = Nachrichtenlänge

2 $\mu s/Bit$ * 144 Bit = 288 µs = Übertragungsdauer bei 500 $kBaud/s$

650 µs - 288 µs = 362 µs = Verarbeitungsdauer im VW-Gateway

Da die stark voneinander abweichenden Verarbeitungsdauern intuitiv nicht durch den Unterschied in der Nutzlastlänge von 4 Byte zu erklären sind, muss versucht werden, die Ursache für diese Abweichung nicht durch eine Berechnung, sondern durch Messungen mit einem Oszilloskop zu finden. Darüber hinaus können die im Folgenden Abschnitt vorgestellten Messungen dazu genutzt werden, die Berechnungen insgesamt zu validieren.

6.3.2.2 Validierung des berechneten Zeitverhaltens

Um die im vorherigen Abschnitt berechnete Verarbeitungsdauer des VW-Gateways zu validieren und um die starke Abweichung zwischen minimaler und maximaler Verarbeitungsdauer zu erklären, wird das VW-Gateway anhand des in Abbildung 6.12 gezeigten Versuchsaufbaus mit einem Oszilloskop aus Black Box Sicht untersucht.
Das in Abbildung 6.11 dargestellte Problem, die Übertragungsdauer auf dem CAN-Bus aus der berechneten Zeit herausrechnen zu müssen, ist im Falle der Messung mit einem Oszilloskop nicht gegeben. Hier kann ohne weitere Hilfsmittel das in der Abbildung 6.11 eingezeichnete, innenliegende EOF-zu-SOF-Intervall ausgemessen werden.
Zu diesem Zweck wird das Oszilloskop derart konfiguriert, dass auf eine Nachrichten-ID getriggert wird, von der bekannt ist, das diese CAN-Nachricht das VW-Gateway durchläuft. Das Oszilloskop verfügt über einen internen Speicher, der je nach Abtastrate ein unterschiedlich langes Zeitfenster der CAN-Kommunikation beider ausgemessenen CAN-Busse enthält. Im konkreten Anwendungsfall wurde ein Zeitfenster von 10 ms bei einer Abtastrate von $10 * 10^6$ $Sample/s$ gewählt.
Die Messungen ergaben eine minimale Verarbeitungsdauer durch das VW-Gateway von 60 µs und erklärten ebenfalls die berechnete, maximale Verarbeitungsdauer in Höhe von 362 µs: In Abschnitt 6.3.2.1 wurden die 362 µs fälschlicherweise als Verarbeitungsdauer interpretiert.

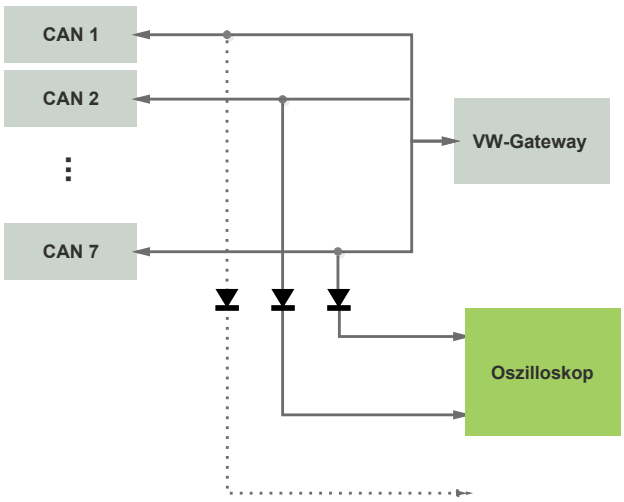

Abbildung 6.12: Versuchsaufbau: Zwei CAN-Busse werden mit einem Oszilloskop abgegriffen, um den zeitlichen Versatz von auf beiden CAN-Bussen erscheinenden Nachrichten zu ermitteln.

Tatsächlich resultiert die große Differenz der Zeitstempel in der Datenbank aber aus dem nicht-preemptiven Verhalten des CAN-Protokolls oder der Priorisierung von CAN-Nachrichten, wie Abbildung 6.13 zu entnehmen ist.

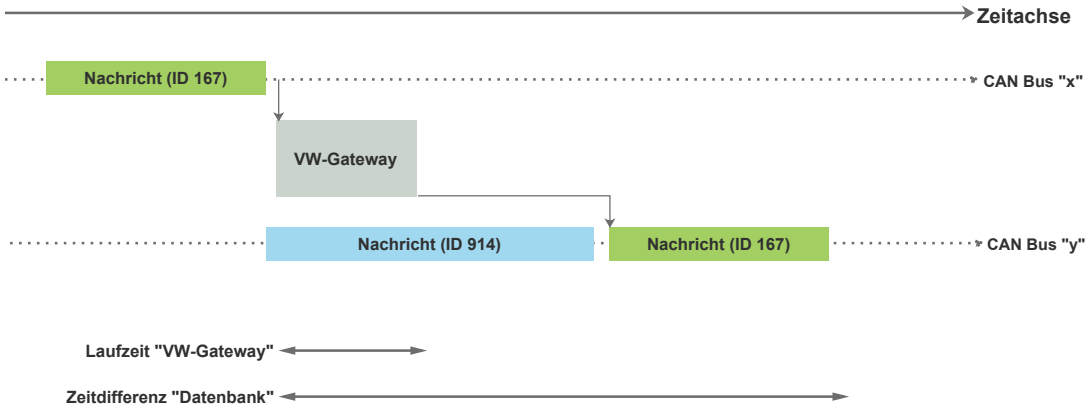

Abbildung 6.13: Veranschaulichung der Ursache für die starke Schwankung der Zeitstempeldifferenzen in der Datenbank: Verzögerungen aufgrund des nicht-preemptiven Verhaltens des CAN-Protokolls (oder aufgrund der Verdrängung durch höher priorisierte Nachrichten) werden in der Datenbank nicht trivial ersichtlich.

6.3.2.3 Bewertung der Zeitanalyse

Es ist nicht ohne Weiteres möglich, der Untersuchung der Verarbeitungszeit des VW-Gateways durch Berechnung und Messung einen Vollständigkeitscharakter zu unterstellen. Die betrachteten Ausschnitte aus der CAN-Kommunikation schränken die Realität immer auf ein zeitlich begrenztes Beobachtungsfenster ein, egal ob die Datenspeicherung in der Datenbank oder im Oszilloskop stattfindet. Es gibt keine Garantie, dass jeder (zeitliche) Sonderfall auch glücklicherweise erfasst wurde.
Unter dieser Einschränkung sind auch die Ergebnisse für die minimale Verarbeitungszeit zu betrachten: Nur mit hoher Wahrscheinlichkeit ist die minimale Verarbeitungszeit des VW-Gateways in Höhe von 60 µs (gemessen) und 64 µs (berechnet) korrekt. Da es nicht möglich ist, die gespeicherten Nachrichten des Oszilloskops in Relation zu konkreten Nachrichten in der Datenbank zu setzen, gibt es ebenfalls unterschiedliche Erklärungsansätze für den 4 µs - Unterschied der gemessenen und berechneten, minimalen Verarbeitungsdauer: Zum einen ist es möglich, dass die Messungenauigkeit des softwarebasierten Zeitstempels die Differenz von 4 µs erklärt. Zum anderen ist aber ebenfalls möglich, dass diese 4 µs schlicht das Resultat der unterschiedlichen, betrachteten Datenmengen ist.
Um die tatsächliche maximale Verarbeitungsdauer des VW-Gateways zu ermitteln, kann nicht auf eine Messung per Oszilloskop zurückgegriffen werden, da die betrachteten Zeitfenster dort verhältnismäßig kurz sind. Ein möglicher Ansatz wäre es, eine Datenbankanwendung zu entwickeln, welche fähig ist, ungewöhnlich hohe Verarbeitungszeiten genau dann auszuschließen, wenn diese durch das nicht-preemptive Verhalten des CAN-Protokolls oder die Nachrichtenverdrängung durch höher priorisierte Nachrichten verursacht wurden.
Aber auch die ermittelte, minimale Verarbeitungsdauer des VW-Gateways erlaubt Schlüsse bezüglich der Ersetzbarkeit des VW-Gateways durch Core-Gateways und den RTE Backbone. Obwohl das Core-Gateway mit einer Laufzeit der Gatewaylogik in Höhe von 235 µs durchaus in der Lage ist, jede Nachricht auf den VW-CAN-Bussen zu verarbeiten, müssen die Zeitgarantien für die Übertragungslatenz von CAN-Nachrichten zwischen unterschiedlichen CAN-Bussen untersucht werden. Die Zeitgarantien müssen gegebenenfalls um die Differenz der Laufzeiten von Core-Gateway und VW-Gateway in Höhe von 175 µs erweitert werden. Prinzipiell erklärt sich die erhöhte Gatewaylogik-Laufzeit des Core-Gateways dadurch, dass das Core-Gateway im Gegensatz zum VW-Gateway in der Lage ist, eine via IDL flexibel konfigurierbare, inhaltsabhängige Nachrichtenbehandlung durchzuführen - es wird die zusätzliche Gatewayflexibilität durch einen erhöhten Bedarf an Rechenleistung erkauft.

7 Übertragung auf weitere automobile Bussysteme

Ein interessanter Aspekt der Erweiterbarkeit des Kommunikationsgateways ist, ob das Kommunikationsgateway bzw. die Architektur und Implementierung des Gateways ebenfalls die Anbindung von weiteren automobilen Bussystemen an einen RTE Backbone zulässt. Eine diesbezügliche Untersuchung muss eine in dieser Arbeit in Bezug auf CAN-Bussysteme vorgestellte Eigenschaft vernachlässigen: den verteilten CAN-Bus. Da die Machbarkeit des Konzepts des verteilten CAN-Bus nur durch eine intensive Untersuchung des CAN-Protokolls analysierbar war, wird für eine Adaption dieses Konzeptes auf andere Bussysteme wie MOST oder FlexRay eine ähnliche Untersuchung der Protokolle notwendig.
Die darüber hinausgehende Kompatibilität automobiler Bussysteme mit dem in dieser Arbeit vorgestellten Lösungsansatz kann aber durch die Untersuchung der Schlüsselmechanismen der fraglichen Bussysteme überprüft werden. Da das Kommunikationsgateway ein Gateway auf Anwendungsebene ist, wird im Folgenden vereinfachend angenommen, dass die Hardwareplattform die zu überprüfenden Bussysteme unterstützt und dass das Betriebssystem API-Funktionen zur Verfügung stellt, welche Eingabedaten und Ausgabedaten analog zu den in Abschnitt 5.3.2.1 präsentierten Voraussetzungen zur Anwendbarkeit der IDL akzeptieren bzw. produzieren.
Die Anwendbarkeit der IDL resultiert in Konsequenzen bezüglich der Fähigkeiten zur Protokollübersetzung: Sobald die IDL verwendet werden kann, können beliebige Nachrichteninhalte generiert werden. Die Datenquelle für diese Nachrichteninhalte kann dann eine Mixtur aus Bestandteilen eingehender Nachrichten, aus vom Betriebssystem bereitgestellten und über Funktionsaufrufe ermittelbaren Daten und aus den Ergebnissen von Manipulationen der Bestandteile eingehender Nachrichten sein.
Die Untersuchung der Schlüsselmechanismen weiterer automobiler Bussysteme beschränkt sich dadurch weitestgehend auf die Untersuchung der Übertragungsmechanismen, speziell der Mechanismen zur Übertragung von zeitgesteuerten und ereignisgesteuerten Nachrichten. Diese Mechanismen werden von allen für moderne Automobilanwendungen sinnvoll verwendbaren Bussystemen zur Verfügung gestellt.

In den folgenden Abschnitten werden die MOST- und FlexRay-Technologien auf eine mögliche Anbindung an den RTE Backbone durch das Kommunikationsgateway untersucht.

7.1 Media Oriented Systems Transport (MOST)

Wie in Abschnitt 2.1.3 vorgestellt, erlaubt MOST die zeitsynchronisierte Kommunikation zwischen bis zu 64 Busteilnehmern in Sterntopologie oder in durch Hub bzw. Switch gebildeten Topologien. Dabei werden in einer MOST-Topologie Frames mit einer konstanten Frequenz (bis 48 kHz/s)übertragen, welche sich je nach Geschwindigkeitsklasse (25 MB/s / 50 MB/s / 150 MB/s) in der Länge unterscheiden. Die folgenden Untersuchungen basieren auf MOST25 mit 25 MB/s.

Der Versand der einzelnen Frames wird vom alleinigen Zeitmaster innerhalb der MOST-Topologie initiiert, woraufhin jeder Frame an alle Busteilnehmer weitergereicht wird. Die Busteilnehmer synchronisieren ihre interne Zeit anhand der Präambel eines Frames (siehe Abbildung 7.1). Der Zugriff auf die unterschiedlichen Datenfelder (synchrone Daten, asynchrone Daten, Control Channel) erfolgt auf zwei unterschiedliche Arten und Weisen: Der Zugriff auf das synchrone Datenfeld ist via TDM organisiert, Zugriff auf asynchrone Datenfelder und den Control Channel erhält ein Busteilnehmer durch ein zu Carrier Sense Multiple Access/Collision Avoidance (CSMA/CA) mit IEEE 802.11e Prioritätsklassen ähnliches Verfahren.

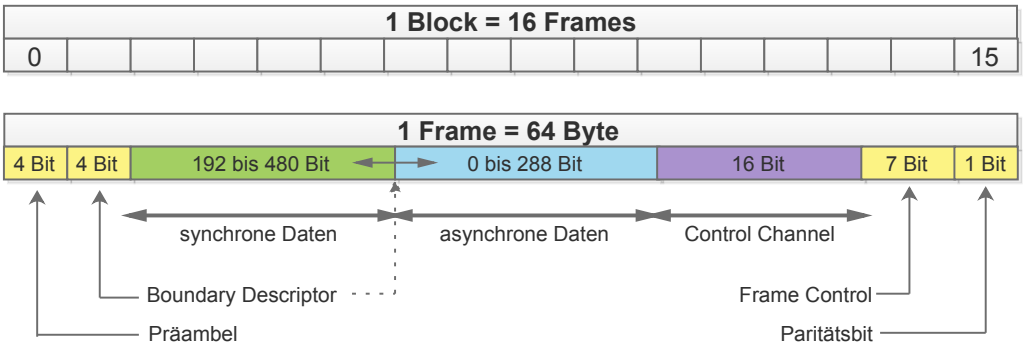

Abbildung 7.1: Struktur eines MOST25-Frames und eines aus 16 Frames zusammengesetzten Blocks. Der Wert innerhalb des Boundary Descriptor Feldes zwischen 6 und 15 definiert die Länge des Feldes für synchrone Daten in Vielfachen von 32 Bit.

Der TDM-basierte Zugriff auf das synchrone Datenfeld erfordert zur Entwurfszeit die Zuweisung von Zeitscheiben an bestimmte Busteilnehmer in einem Schedule. Diese Zugriffsvariante

erlaubt allen Busteilnehmern die Kenntnis von Datenquelle und Datenziel, ohne dass diese Information im synchronen Datenfeld explizit angegeben sein muss. Die Verwaltung der Zugriffsberechtigung findet auf der MAC-Schicht entsprechend des OSI-Schichtenmodells statt. Für das Kommunikationsgateway auf Anwendungsebene bedeutet dies, dass es ohne weitere Hindernisse die ebenfalls zur Protokollübersetzung zwischen CAN und RTE genutzten, anhand der IDL konfigurierten Mechanismen nutzen kann. Allerdings muss die von der IDL verwendete Datenstruktur für Nachrichtenidentifkationskriterien für die Verwendung von TDM so erweitert werden, dass das Kommunikationsgateway den MOST-Schedule verwenden kann, um eine Unterscheidung zwischen über den RTE Backbone zu übertragenden und zu verwerfenden Nachrichten vorzunehmen. Diese Anpassung stellt kein Problem dar, da die Datenstruktur schon jetzt darauf ausgerichtet ist, unterschiedliche Typen von Nachrichtenidentifikationskriterien erkennbar und behandelbar zu machen.
Unter der Voraussetzung, dass das verwendete RTE auf einer physikalischen Schicht basiert, welche in der Lage ist, die konstante Frame-Frequenz des MOST zu bedienen, ist es möglich, das synchrone Datenfeld des MOST auf TT-Nachrichten des RTE abzubilden. Die auf dem entsprechenden VLID konfigurierten Zeitgarantien für TT-Nachrichten müssen an die Zeitgarantien des MOST angepasst werden.

Der Aufbau des asynchronen Datenfeldes eines MOST-Frame ähnelt dem des Control Channels - unabhängig davon, dass der Control Channel in Abschnitten von 2 Byte auf die 16 Frames eines MOST-Blocks verteilt ist, um den Overhead des Control Channels pro Frame zu reduzieren. Das asynchrone Datenfeld und der zusammengesetzte Control Channel beinhalten beide immer zumindest eine Quell- und Zieladresse und die Nutzlast (vergleiche Abbildung 7.2). Die ereignisgesteuerte Übertragung von Daten des asynchronen Datenfeldes oder des

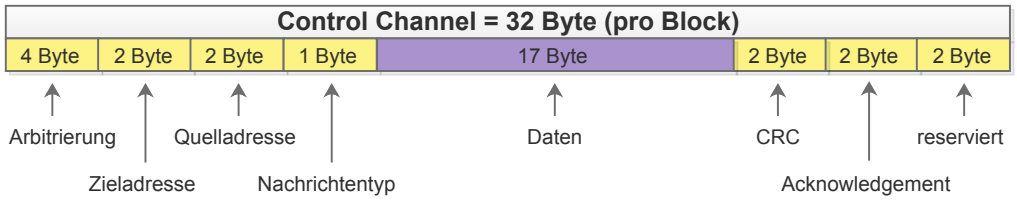

Abbildung 7.2: Struktur einer MOST25-Control-Channel-Nachricht, zusammengesetzt aus je 2 Bytes Control Channel aus den 16 MOST25-Frames eines Blocks.

Control Channel lässt sich leicht auf die RC-Nachrichtenklasse des RTE abbilden. Aus der konstanten Frame-Frequenz, der maximalen Länge des asynchronen Datenfeldes und der Länge

eines zusammengefassten Control Channel des MOST kann die maximale Bandbreite dieser Übertragungsvariante berechnet und die BAG der RC-Nachrichten des RTE entsprechend angepasst werden.
Um auch für asynchrone Daten und Daten des Control Channels eine selektive und zielgerichtete Übertragung über den RTE Backbone zu ermöglichen, ist eine weitere Anpassung der Datenstruktur zur Speicherung von Nachrichtenidentifikationskriterien notwendig. Damit eine Nachrichtenfilterung möglich ist, kann in einem ersten Ansatz z.B. ein Tupel aus Quell- und Zieladresse als Nachrichtenidentifkationskriterium genutzt werden.

7.2 FlexRay

FlexRay ist - wie in Abschnitt 2.1.4 vorgestellt - ein Automobilbussystem mit Schwerpunkt auf zeitlichem Determinismus und Fehlertoleranz unter Vernachlässigung hoher Bandbreitenanforderungen. Nach Weyrich (2012) besteht ein FlexRay-Zyklus nicht zwangsweise aus allen in

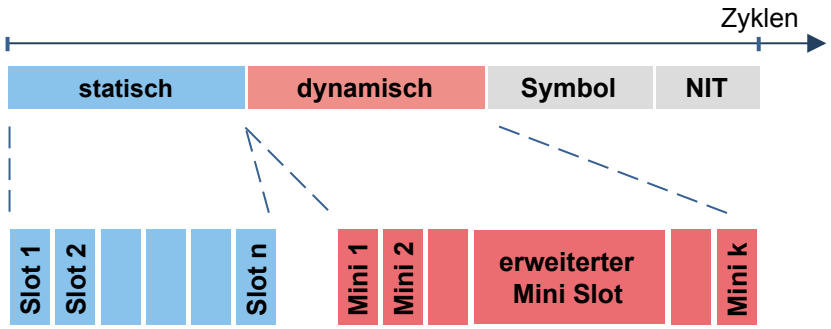

Abbildung 7.3: FlexRay-Zyklus mit statischem und dynamischem Segment, einem erweitertem Minislot, Symbolfenster und NIT. Betriebsmodi ohne dynamisches Segment und/oder Symbolfenster sind ebenfalls zulässig, aber weniger üblich.

Abbildung 7.3 dargestellten Elementen. So sind Zyklen ohne dynamisches Segment und/oder Symbolfenster ebenfalls spezifiziert. In der Automobilindustrie finden aber primär Zyklen mit statischem und dynamischem Segment ohne Symbolfenster Verwendung. Die zeitliche Synchronisation einzelner Busteilnehmer auf die globale Zeit basiert auf der Korrektur der Zeitabweichung und des Uhren-Drifts der internen Oszillatoren der Busteilnehmer und wirkt sich auf die jeweilige Länge der von einem Busteilnehmer eingehaltenen NIT aus.
Der Zugriff eines Busteilnehmers auf das statische Segment ist über das TDMA-Zugriffsverfahren organisiert. Hierzu ist zur Entwurfszeit ein Schedule zu erstellen, der eine Zuordnung zwischen statischem oder dynamischem Slot und einer Frame-ID (siehe Abbildung 7.4) herstellt. Anhand

dieses Verfahrens können die einzelnen Busteilnehmer in Kenntnis der Frame-ID ermitteln, welcher Busteilnehmer Datenquelle bzw. -ziel eines in einem spezifischen, statischen Slot übertragenen Datums ist.

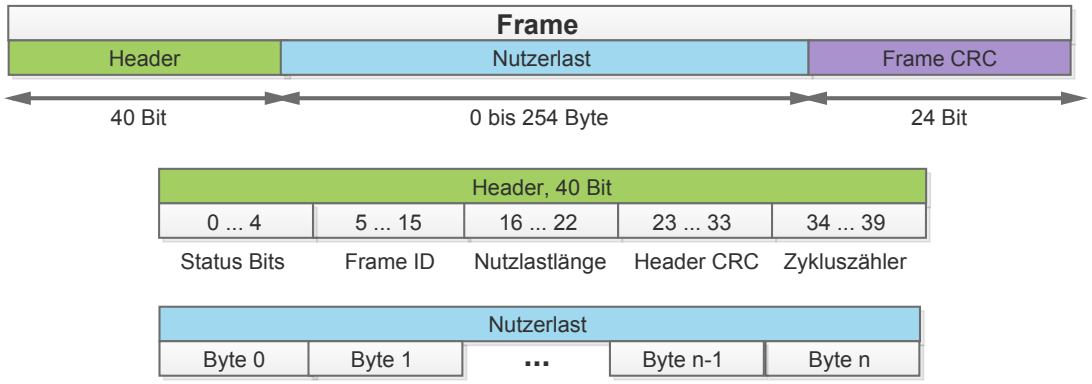

Abbildung 7.4: Veranschaulichung der FlexRay Frame-Struktur aus Header, Nutzlast und Frame-CRC inklusive Aufschlüsselung der Struktur von Header und Nutzlast.

Der Zugriff auf das dynamische Segment erfolgt über das Flexible Time Division Multiple Access (FTDMA)-Verfahren: Sobald ein Busteilnehmer innerhalb des dynamischen Segments ereignisbasierte Nachrichten senden soll, untersucht der Busteilnehmer den zur Nachricht gehörigen Mini-Slot. Sollte in diesem Mini-Slot gerade keine Kommunikation stattfinden, startet der Busteilnehmer den Sendevorgang. Allerdings kann die Nachrichtenlänge im dynamischen Segment die Länge eines Mini-Slots überschreiten. Dies hat zur Konsequenz, dass andere sendewillige Busteilnehmer durch eine bereits begonnene Übertragung im dynamischen Segment verdrängt werden können (vergleiche Abbildung 7.3, erweiterter Mini-Slot).
Analog zum TDM-Verfahren des MOST werden auch das TDMA- und FTDMA-Verfahren unterhalb der Anwendungsebene in der MAC-Schicht entsprechend des OSI-Schichtenmodells abgewickelt. Somit ist keine Anpassung des Kommunikationsgateways auf Anwendungsebene erforderlich. Auf Anwendungsebene kann die IDL in Verbindung mit der FlexRay-Frame-ID als Nachrichtenidentifkationskriterium dazu genutzt werden, um beliebige Protokollübersetzungen vorzunehmen und eine selektive und zielgerichtete Übertragung via RTE auszulösen. In Übereinstimmung mit der Anbindung von MOST an RTE lassen sich zeitgesteuerte Nachrichten (statisches Segment) und ereignisgesteuerte Nachrichten (dynamisches Segment) auf die TT- bzw. RC-Nachrichtenklasse des RTE abbilden.

7.3 Bewertung der Übertragung

Die vorangegangenen Abschnitte haben gezeigt, dass konzeptionell nichts dagegen spricht, MOST oder FlexRay über das Kommunikationsgateway an den RTE Backbone anzubinden. Allerdings gibt es ein potentielles Hindernis aus Sicht der Leistungsfähigkeit der für das Kommunikationsgateway verwendeten Hardwareplattform. Darüber hinaus ist bei der Verwendung eines einzelnen Kommunikationsgateways zur Anbindung mehrerer, unterschiedliche Bussysteme der Task-Schedule des Kommunikationsgateways mit erhöhter Sorgfalt zu erstellen.
Wie im Abschnitt 6.3.1.4 in Abbildung 6.6 gezeigt, besteht ein linearer Zusammenhang zwischen Protokollübersetzungsdauer und Länge [Byte] der zu übersetzenden Nachricht. Die Laufzeit alleine für den Übersetzungsvorgang verdoppelt sich nahezu im Vergleich der Übersetzung von Nachrichten mit 8 Byte Nutzlast (CAN) und Nachrichten mit 254 Byte Nutzlast (FlexRay), was einer Verlängerung der Laufzeit der Protokollübersetzung in Höhe von ca. 50 µs auf ca. 100 µs Gesamtdauer entspricht. Wenn nun die gemessene Gesamtlaufzeit der Gatewaylogik von 235 µs um 50 µs auf 285 µs erhöht würde, könnte das Gateway überschlägig 3500 Nachrichten pro Sekunde verarbeiten. Die der Überlegung zugrundeliegende FlexRay-Nachricht mit 254 Byte Nutzlast hätte eine Gesamtlänge von 262 Byte. Unter der Annahme, dass die Laufzeit der Gatewaylogik ansonsten frei von weiteren Verzögerungen wäre, könnte das Gateway somit nur einen FlexRay-Bus bedienen, der eine maximale Bandbreite von 917 $^{kBaud}/_{s}$ bzw. 850 $^{kB}/_{s}$ aufweist. Dieser Wert ist deutlich von der für FlexRay spezifizierten, maximalen Bandbreite von 10 $^{MB}/_{s}$ entfernt. In einer verwandten Überlegung würde die Frame-Frequenz des MOST von üblicherweise 44,1 $^{kHz}/_{s}$ (bis zu 48 $^{kHz}/_{s}$) einer Nachrichtenverarbeitungsfrequenz des Gateways in Höhe von ca. 4000 Nachrichten pro Sekunde gegenüberstehen. Die sinnvolle Anbindung von MOST und FlexRay bedarf also einer deutlich leistungsfähigeren Hardware, als im Rahmen dieser Arbeit verwendet wurde.
Darüber hinaus muss im Task-Schedule des Gateways für jede zu verarbeitende Nachricht ein Aufruf des Job-Tasks (siehe Abschnitt 5.5.4) eingeplant werden. Ferner muss für jede via RTE zu versendende Nachricht eine Instanz des RTE-Sendetasks eingefügt werden. Dabei definiert der zeitliche Abstand dieser Tasks innerhalb des Task-Schedules eine zusätzliche Latenz in der Übertragung von Nachrichten via RTE Backbone. Es könnte eine Herausforderung sein, einen Task-Schedule zu entwerfen, der - ohne von Überprovisionierung Gebrauch zu machen - dennoch die minimal mögliche Übertragungslatenz für Nachrichten zu garantieren, die aus unterschiedlichen Bussystemen stammen und von einem einzelnen Gateway verarbeitet werden sollen.

8 Zusammenfassung, Fazit und Ausblick

In diesem Kapitel werden die Arbeit und ihre Ergebnisse zusammengefasst. Es wird ein Fazit gezogen, und die Arbeit wird mit einem Ausblick auf zukünftige Arbeiten abgeschlossen.

8.1 Zusammenfassung von Arbeit und Ergebnissen

Die Zielsetzung dieser Arbeit ist es, die inkrementelle Konsolidierung automobiler Bussysteme auf Basis eines RTE Backbone zu erreichen. Zu diesem Zweck wird in dieser Arbeit ein flexibles Kommunikationsgateway implementiert und eine nicht-spezifizierte CAN-Topologie (verteilter CAN-Bus) untersucht. Auf diese Weise können bestehende CAN-Infrastrukturen sukzessive und ohne Veränderung der CAN-Anwendungen mit einem RTE Backbone verbunden werden.

Die Arbeit beginnt mit der Vorstellung etablierter automobiler Bussysteme und deren Eigenschaften und der Präsentation von RTE und des Potentials von RTE als automobiles Kommunikationsnetzwerk. Fortgeführt wird die Arbeit mit der detaillierten Untersuchung der Prozesse und Methoden zum Entwurf und zur Entwicklung automobiler Systeme, in deren Kontext sich diese Arbeit einordnen muss. Automobile Systeme sind - und werden weiterhin im zunehmenden Maße - komplexe verteilte Systeme, die aufgrund ihrer Beschaffenheit und ihres Interaktionsgrades immer mehr Ähnlichkeit mit verteilten Systemen erlangen, wie sie eigentlich aus den Computerwissenschaften bekannt sind. Der Entwicklungsprozess in der Automobilindustrie ist ebenfalls dezentralisiert, indem einzelne Automobilhersteller die Entwicklung der Komponenten eines automobilen Systems nicht selbst durchführen, sondern diese Aufgabe an eine Vielzahl von Dienstleistern und Zulieferbetrieben übertragen. Die Organisation des Entwicklungsprozesses wird anhand des V-Modells durchgeführt, welches hierarchische Prozessebenen und für jede Prozessebene ein Paar aus Anforderung und Validierung bzw. Spezifikation und Verifikation definiert. Da nicht nur komplette Systeme, sondern ebenfalls einzelne Komponenten anhand des V-Modells entwickelt werden, verursacht eine Veränderung der Systemspezifikation ebenso für jede neue oder veränderte Komponente das erneute Durchlaufen des V-Modells. Damit ginge die zeitaufwändige Synchronisation zwischen den einzelnen beteiligten Dienstleistern bzw. Zulieferern einher. Aus diesem Grund

ist es attraktiv, eine Lösung für die Aufgabenstellung dieser Arbeit zu finden, welche nicht nur den Anteil weiterzuverwendender Komponenten maximiert, sondern ebenfalls weitestgehend autark durch einzelne Entwicklergruppen, Dienstleister oder Zulieferer durchgeführt werden kann. Kernpunkt der Lösung sollen flexible Kommunikationsgateways sein, welche alte wie neue Steuergeräte mit dem RTE Backbone verbinden und somit das klassisch verwendete, zentrale Kommunikationsgateway ersetzen müssen.

In Kapitel 4 werden Anforderungen entwickelt, welche insbesondere die unabhängige Entwicklung in einem dezentralisierten Umfeld und die Weiterverwendbarkeit bestehender Komponenten begünstigen sollen. Dabei stellt die Weiterverwendung existierender Steuergeräte und somit ebenfalls die Weiterverwendung der Steuergerätesoftware eine Herausforderung dar, sobald die Bustopologie verändert werden soll, welche die Steuergeräte miteinander verbindet. Eine Veränderung der Bustopologie liegt z.B. dann vor, wenn einzelne Steuergeräte durch native RTE-Steuergeräte ersetzt werden sollen oder aber existierende Steuergeräte zwecks Optimierung des Kabelbaums physikalisch neu platziert und über den RTE Backbone transparent an das vormals verwendete Bussystem angeschlossen werden sollen. Aus dieser Herausforderung resultiert die Anforderung, die CAN-Steuergeräte trotz des Aufbrechens bestehender Partitionierungen und Kollisionsdomänen in einem verteilten, für die Steuergeräte nicht als solchen erkennbaren, CAN-Bus zusammenzufassen. Aus der Notwendigkeit, das üblicherweise verwendete zentrale Kommunikationsgateway in Unkenntnis des exakten Funktionsumfanges des zentralen Kommunikationsgateways zu ersetzen, resultiert die Anforderung, das neu entwickelte Kommunikationsgateway so zu gestalten, dass es flexibel und frei konfigurierbar alle relevanten Protokollübersetzungs- und Datenmanipulationsaufgaben übernehmen kann. Dabei unterliegt das neue Kommunikationsgateway denselben Anforderungen an zeitlichen Determinismus wie das zentrale Kommunikationsgateway. Die Erfüllung dieser Anforderungen muss ebenfalls anhand formaler Verifikationsmethoden überprüfbar sein.

Das Konzept zur Erfüllung der Anforderungen wird in Kapitel 5 vorgestellt. Das für die Weiterverwendung von Steuergerätesoftware trotz Veränderung der Bustopologie relevante Konzept des verteilten CAN-Bus wird aus unterschiedlichen Perspektiven auf seine Machbarkeit untersucht. Es werden die Auswirkung des Teilkonzeptes „verteilter CAN-Bus“ und die damit einhergehende physikalische Trennung auf das Zeitverhalten, die Eigenschaften des CAN-Protokolls sowie auf das Anwendungsverhalten untersucht. Das Untersuchungsergebnis, eine Liste von Restriktionen für die Umsetzbarkeit des Teilkonzeptes, wird in eine Prüfliste für Entwickler überführt, anhand derer die Erfolgsaussichten der Umsetzung vorab bewertbar werden.

Die Emulation des unbekannten Funktionsumfanges des herkömmlichen, zentralen Kommunikationsgateways wird dadurch ermöglicht, dass eine IDL entwickelt wird, welche in der Lage ist, alle sinnvollen Varianten einer Protokollübersetzung oder Datenmanipulation mit Hilfe ihrer Grammatik auszudrücken. Um dem allgegenwärtigen Wunsch nach effizienter Kommunikation zu entsprechen wird das Konzept eines Verfahrens zur Nachrichtenbündelung vorgestellt, welches geeignet ist in Abhängigkeit vom Anwendungsfall die Auslastung des RTE deutlich zu reduzieren. Das neue Kommunikationsgateway selbst wird in einer Softwarearchitektur implementiert, die neben einer hohen Modularität ebenfalls einfache Komponentenschnittstellen aufweist, um Wartbarkeit und Erweiterbarkeit der Kommunikationsgatewayanwendung zu begünstigen.

In Kapitel 6 ist die Auswertung der Implementierung des Kommunikationsgateways dokumentiert. Die Auswertung befasst sich mit drei unterschiedlichen Teilbereichen: der Qualitätssicherung, der Dokumentation des Einsatzes des Gateways in einem Prototypenfahrzeug, sowie der Evaluation der Implementierung in Hinsicht auf das Zeitverhalten und das Potential, ein herkömmliches zentrales Kommunikationsgateway zu ersetzen. Dabei umfasst die Qualitätssicherung die Prozessqualität und Softwarequalität entsprechend des vom V-Modell geleiteten Entwicklungsprozesses und präsentiert das konkrete Vorgehen zur Qualitätssicherung. Die Dokumentation des Einsatzes im Prototypenfahrzeug der Core Arbeitsgruppe ist dazu geeignet, die Praxistauglichkeit und Leistungsfähigkeit abstrakt zu erfassen. Die Leistungsfähigkeit wird durch die Untersuchung des Zeitverhaltens des Kommunikationsgateways und seiner einzelnen Komponenten verfeinert dargestellt und anschließend bewertet. Das Potential, ein herkömmliches zentrales Kommunikationsgateway zu ersetzen, wird durch die rechnerische Analyse und das Messen des Zeitverhaltens des zentralen Kommunikationsgateways des Prototypenfahrzeugt untersucht. Das Ergebnis dieser Untersuchung wird in Relation zum ermittelten Zeitverhalten des neuen Kommunikationsgateways gestellt, um eine abschließende Leistungsbewertung vornehmen zu können.

Um die Flexibilität des Kommunikationsgateways und die Qualität der generischen Konfigurationsmöglichkeit zu testen, wird in Kapitel 7 abschließend der Versuch unternommen, die mit einem Fokus auf die Anbindung von CAN-Bussen an den RTE Backbone betriebene Entwicklung auf weitere Bussysteme zu übertragen. Mit MOST und FlexRay werden hierbei zwei Technologien untersucht, die aufgrund ihrer Leistungsfähigkeit nicht ohne gute Alternativen in absehbarer Zeit abgekündigt werden und deren Migration hin zu RTE aufgrund der Steuergerätekosten wirtschaftlich sinnvoll sein kann - ganz im Gegensatz zu LIN. Der

Übertragungsversuch auf diese Bussysteme wird durch eine Bewertung der Versuchsergebnisse abschlossen, die zeigt, dass das Konzept des Kommunikationsgateways auch auf MOST und FlexRay anwendbar ist.

8.2 Fazit

RTE ist ein guter Kandidat für ein zukünftiges, automobiles Backbone-Netzwerk. RTE verspricht zeitlichen Determinismus im Sinne harter Echtzeitanforderungen, bietet eine hohe Bandbreite und ist aufgrund der Austauschbarkeit der Technologien auf Ebene der physikalischen Schicht in der Lage, auch zukünftige, gewachsene Anforderungen der Automobilindustrie zu erfüllen. Die Fähigkeit, etablierte Bustechnologien an RTE anbinden zu können, ist ein wesentlicher Faktor für die Akzeptanz von RTE im automobilen Einsatz. Die Anbindung etablierter Bustechnologien an RTE mit minimiertem Organisationsaufwand unter Weiterverwendung wesentlicher Teile des Wissens, der Arbeitskräfte, der Software und der Hardware reduziert wiederum die aus einem derartigen Vorhaben resultierende wirtschaftliche Belastung.

Da CAN-Steuergeräte preiswert, zuverlässig und weit verbreitet sind, werden sie nicht in absehbarer Zeit durch native RTE-Steuergeräte ersetzt. Diese Arbeit präsentiert eine Migrationsstrategie zur transparenten Weiterverwendung existierender CAN-Steuergeräte in einer Switch-gestützten Infrastruktur auf Basis von RTE und deren Einschränkungen. Die Migrationsstrategie wird durch aus den Einschränkungen abgeleitete Richtlinien zur Machbarkeitsuntersuchung vervollständigt.

Das im Rahmen dieser Arbeit entwickelte Kommunikationsgateway verwendet eine IDL-gestützte Protokollübersetzung zur Umsetzung der Migrationsstrategie und wird aktiv in dem Prototypenfahrzeug der Core Arbeitsgruppe eingesetzt. Das Kommunikationsgateway ist darüber hinaus optimiert, um seine Verarbeitungszeit und die Bandbreitennutzung des RTE zu reduzieren. Die Untersuchung und Auswertung des Kommunikationsgateways zeigt, dass die Implementierung einer inhaltsabhängigen Nachrichtenbehandlung im Sinne einer Deep Packet Inspection die Hardware aktueller Kommunikationsgateways überlasten kann, aber gleichzeitig einen deutlichen Gewinn an Flexibilität darstellt. Durch die Übertragung des Konzeptes des Kommunikationsgateways auf die Anbindung von MOST und FlexRay werden neben der Flexibilität des Kommunikationsgateways ebenso die Grenzen seiner Leistungsfähigkeit aufgezeigt. Die Auswertung und die allgemein steigenden Anforderungen an Bandbreite und

Echtzeitverhalten in der automobilen Kommunikation heben die ebenfalls stark wachsenden Anforderungen an die automobile Steuergerätehardware hervor.

8.3 Ausblick auf zukünftige Arbeiten

Die auf dieser Arbeit basierenden zukünftigen Arbeiten können primär drei unterschiedliche Themenkreisen bedienen: die Erweiterung der Einsetzbarkeit des Kommunikationsgateways, die Ausdehnung des praktischen Einsatzes im Prototypenfahrzeug der Core Arbeitsgruppe und die Optimierung des Laufzeitverhaltens des Kommunikationsgateways.
Eine Erweiterung der Einsetzbarkeit des Kommunikationsgateways kann z.B. darin bestehen, eine Kompatibilität zu weiteren RTE-Technologien als physikalische Schicht herzustellen. Vielversprechender Kandidat dafür sind OPEN Alliance BroadR-Reach oder Audio/Video Bridging (AVB). Die Attraktivität von AVB ist in der Einsetzbarkeit für Multimedia-Anwendungen begründet, deren Bedeutung durch die verstärkte Integration stationärer und tragbarer Entertainmenthardware in Zukunft weiterhin zunehmen wird. Die Vorzüge von BroadR-Reach liegen nicht nur in der kostengünstigen Verkabelungsmöglichkeit anhand nur eines ungeschirmten Single-Twisted-Pair-Kabels, sondern sind ebenfalls dadurch gegeben, dass BroadR-Reach von der OPEN Alliance Special Interest Group als offener Standard geplant ist.
Um den Einsatz des Kommunikationsgateways im Prototypenfahrzeug der Core Arbeitsgruppe auszudehnen, ist vorstellbar, eine bidirektionale Kommunikation mit den automobilen CAN-Bussen zuzulassen. Aktuell sind die CAN-Busse des Prototypenfahrzeugs durch sogenannte CAN-Dioden gegen den Schreibzugriff durch das Kommunikationsgateway physikalisch geschützt. Um die Ersetzbarkeit des herkömmlichen zentralen Kommunikationsgateways zu untersuchen, wäre es sinnvoll, zumindest für unkritische Teile der automobilen Kommunikation das zentrale Gateway testweise zu umgehen und den Datenverkehr durch das neue Kommunikationsgateway abwickeln zu lassen.
Die Optimierungsmöglichkeiten des Laufzeitverhaltens des Kommunikationsgateways sind vielfältig: Neben algorithmischen Optimierungen könnte ebenso eine Portierung auf eine leistungsfähigere Hardwareplattform vorgenommen werden. Dabei erscheint ein Sonderfall der Portierung besonders interessant: Aktuell werden immer mehr Kombinationen aus herkömmlichen Prozessoren und Programmable Logic Devices (PLD) angekündigt. Durch die Auslagerung besonders laufzeitintensiver Komponenten des Kommunikationsgateways in PLD könnten die Vorzüge zweier Welten kombiniert werden: Die hohe Flexibilität und leichte Anpassbarkeit softwarebasierter Anwendungen würde durch den hohen Parallelisierungsgrad und die Leistungsfähigkeit der PLD ergänzt.

9 Abkürzungsverzeichnis

ABNF Augmented Backus-Naur Form

BNF Backus-Naur Form

BPU Bundle Protocol Unit

ASN.1 Abstract Syntax Notation One

API Application Programming Interface

BAG Bandwidth Allocation Gap

BE Best Effort

CAN Controller Area Network

CM Compression Master

CPF Communication Profile Families

CoS Class of Service

CSMA/CA Carrier Sense Multiple Access/Collision Avoidance

CSMA/NDA Carrier Sense Multiple Access/non destructive arbitration

CT Critical Traffic

CTID Critical Traffic Identifier

DMA Direct Memory Access

ECU Electronic Control Unit

EOF End Of Frame

FIBEX Fieldbus Exchange Format

FTDMA Flexible Time Division Multiple Access

HAW Hochschule für Angewandte Wissenschaften Hamburg

HIL Hardware in the Loop

IDL Interface Description Language

IP Internet Protocol

IP-Core Intellectual Property Core

IOC Information Object Class

IOS Information Object Set

ISO International Organization for Standardization

ISR Interrupt Service Routine

IRQ Interrupt Request

LIDAR Light Detection and Ranging

LIN Local Interconnect Network

LLC Logical Link Control

MAC Media Access Control

MMU Memory Management Unit

MOST Media Oriented Systems Transport

NAT Network Address Translation

NIT Network Idle Time

NRZ Non-Return-to-Zero

OSI Open Systems Interconnection

PAT Port Address Translation

PCF Protocol Control Frames

PDU Payload Data Unit

QoS Quality of Service

RC Rate Constraint

RTE Realtime Ethernet

SC Synchronization Client

SM Synchronization Master

SoC System on Chip

SOF Start Of Frame

TDM Time Division Multiplex

TDMA Time Division Multiple Access

TPU Transport Protocol Unit

TT Time Triggered

V2V Vehicle-to-Vehicle

V2R Vehicle-to-Roadside

VLID Virtual Link Identifier

WCRT Worst Case Response Time

Literaturverzeichnis

[ABI Research 2014] ABI Research: *Ethernet In-vehicle Networking to Feature in 40% of Vehicles Shipping Globally by 2020*. 2014. – URL https://www.abiresearch.com/press/ethernet-in-vehicle-networking-to-feature-in-40-of

[ARINC 664 1 2002] ARINC 664 1: *Aircraft Data Network, Part 1: Systems Concepts and Overview*. 2002

[AUTOSAR Development Partnership 2008] AUTOSAR Development Partnership: *Requirements on Gateway*. 2008. – URL http://www.autosar.org/download/R3.0/AUTOSAR_SRS_Gateway.pdf. – Version 2.0.4, R3.0, rev. 0001

[Benhamou und Estrin 1983] Benhamou, E.A. ; Estrin, J.: Multilevel Internetworking Gateways: Architecture and Applications. In: *Computer* 16 (1983), Sept, Nr. 9, S. 27–34. – ISSN 0018-9162

[Bochmann und Mondain-Monval 1990] Bochmann, G.V. ; Mondain-Monval, P.: Design principles for communication gateways. In: *Selected Areas in Communications, IEEE Journal on* 8 (1990), Jan, Nr. 1, S. 12–21. – ISSN 0733-8716

[CAN in Automation 2014] CAN in Automation: *Growing car sales increase CAN business*. 2014. – URL http://www.can-newsletter.org/engineering/engineering-miscellaneous/nr_growing-car-sales-increase-the-can-business_140107/

[Core Arbeitsgruppe] Core Arbeitsgruppe: *Core Arbeitsgruppe, Hochschule fuer Angewandte Wissenschaften Hamburg*. – URL http://core.informatik.haw-hamburg.de. – Zugriffsdatum: 04.10.2014

[Crocker und Overell 1997] Crocker, David H. ; Overell, Paul: *Augmented BNF for Syntax Specifications: ABNF*. 1997

[Davis u. a. 2007] Davis, Robert I. ; Burns, Alan ; Bril, Reinder J. ; Lukkien, Johan J.: Controller Area Network (CAN) schedulability analysis: Refuted, revisited and revised. In: *Real-Time Systems* 35 (2007), Nr. 3, S. 239–272

[ETAS GmbH 2013] ETAS GmbH: *New High-performance System with EDE Technology from ETAS*. 2013. – URL http://www.etas.com/data/press_room/Press_Release_ETAS_Data_Engine_EDE.pdf

[Felser 2005] Felser, Max: Real-Time Ethernet–Industry Prospective. In: *Proceedings of the IEEE* 93 (2005), Nr. 6, S. 1118–1129

[Hogenmüller und Triess 2013] Hogenmüller, Thomas ; Triess, Burkhard: *Cost Efficient Gateway Architecture for Deterministic Automotive Networks*. 2013. – URL http://standards.ieee.org/events/automotive/12_Hogenmueller_Triess_EDE_Handout.pdf

[IEEE 802.1 2011] IEEE 802.1: IEEE Standard for Local and metropolitan area networks–Audio Video Bridging (AVB) Systems. In: *IEEE Std 802.1BA-2011* (2011), Sept, S. 1–45

[IEEE 802.3 2012] IEEE 802.3: *IEEE Std 802.3 - IEEE Standard for Ethernet*. 2012

[ISO 11898 1 2003] ISO 11898 1: *Road vehicles – Controller area network (CAN) – Part 1: Data link layer and physical signalling*. 2003

[ISO 11898 4 2004] ISO 11898 4: *Road vehicles – Controller area network (CAN) – Part 4: Time-triggered communication*. 2004

[Mercer Management Consulting 2001] Mercer Management Consulting: *Automobiltechnologie 2010: technologische Veränderungen im Automobil und ihre Konsequenzen für Hersteller, Zulieferer und Ausrüster*. Mercer Management Consulting, 2001 (Themen/Mercer Management Consulting). – URL http://books.google.de/books?id=5UEaSQAACAAJ

[Pedreiras u. a. 2005] Pedreiras, P. ; Gai, P. ; Almeida, L. ; Buttazzo, G.C.: FTT-Ethernet: a flexible real-time communication protocol that supports dynamic QoS management on Ethernet-based systems. In: *Industrial Informatics, IEEE Transactions on* 1 (2005), Aug, Nr. 3, S. 162–172

[Philips Semiconductors 1997] Philips Semiconductors: *Determination of Bit Timing Parameters for the CAN Controller SJA 1000*. http://www.mct.net/download/philips/can_timing.pdf. 1997. – Visit: 30.08.2012

[Richards 2001] Richards, Pat: *Understanding Microchip's CAN Module Bit Timing*. http://ww1.microchip.com/downloads/en/AppNotes/00754.pdf. 2001. – Visit: 30.08.2012

[SAE International 2011] SAE International: *Deterministic Ethernet / TTEthernet®*. 2011. – URL http://standards.sae.org/as6802/

[Scharbarg u. a. 2005] Scharbarg, J. ; Boyer, M. ; Fraboul, C.: CAN-Ethernet architectures for real-time applications. In: *Emerging Technologies and Factory Automation, 2005. ETFA 2005. 10th IEEE Conference on* Bd. 2, Sept 2005, S. 8 pp.–252

[Schmidt u. a. 2010] Schmidt, E.G. ; Alkan, M. ; Schmidt, K. ; Yürüklü, E. ; Karakaya, U.: Performance evaluation of FlexRay/CAN networks interconnected by a gateway. In: *Industrial Embedded Systems (SIES), 2010 International Symposium on*, July 2010, S. 209–212

[Seo u. a. 2006] Seo, Suk-Hyun ; Lee, Sang-Won ; Hwang, Sung-Ho ; Jeon, Jae W.: Development of Network Gateway Between CAN and FlexRay Protocols For ECU Embedded Systems. In: *SICE-ICASE, 2006. International Joint Conference*, Oct 2006, S. 2256–2261

[Software Engineering Institute 2007] Software Engineering Institute: *Published Software Architecture Definitions*. 2007. – URL http://www.sei.cmu.edu/architecture/start/glossary/published.cfm. – Stand: 2015

[Tindell u. a. 1994a] Tindell, K. W. ; Burns, A. ; Wellings, A. J.: An extendible approach for analyzing fixed priority hard real-time tasks. In: *Real-Time Systems* 6 (1994), S. 133–151. – URL http://dx.doi.org/10.1007/BF01088593. – 10.1007/BF01088593. – ISSN 0922-6443

[Tindell und Burns 1994] Tindell, Ken ; Burns, Alan: Guaranteeing Message Latencies On Control Network (CAN). In: *In Proceedings of the 1st International CAN Conference*, CiA, 1994, S. 1–2

[Tindell u. a. 1994b] Tindell, K.W. ; Hansson, H. ; Wellings, A.J.: Analysing real-time communications: controller area network (CAN). In: *Real-Time Systems Symposium, 1994., Proceedings.*, Dec 1994, S. 259–263

[Voss 2005] Voss, Wilfried: *A Comprehensible Guide to Controller Area Network*. Northampton, MA, U.S.A. : Copperhill Media Corporation, 2005

[Wallentowitz und Reif 2011] Wallentowitz, Henning ; Reif, Konrad: *Handbuch Kraftfahrzeugelektronik*. 2. Vieweg+Teubner Verlag, Springer Fachmedien Wiesbaden GmbH, 2011. – ISBN 978-3-8348-0700-7

[Weyrich 2012] Weyrich, Prof. Dr.-Ing. M.: *Grundlagen FlexRay*. 2012. – URL http://www.ias.uni-stuttgart.de/lehre/praktika/automatisierung/unterlagen/06-Grundlagen_FlexRay-v10_de.pdf

Hiermit versichere ich, dass ich die vorliegende Arbeit ohne fremde Hilfe selbständig verfasst und nur die angegebenen Hilfsmittel benutzt habe.

Hamburg, 24. März 2015 Jan Depke