Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Bachelorarbeit

Michael Malsch

OMNeT++ basierte Simulation der Kommunikationsinfrastruktur im Flugzeug

Fakultät Technik und Informatik
Studiendepartment Informatik

Faculty of Engineering and Computer Science
Department of Computer Science

Michael Malsch

OMNeT++ basierte Simulation der Kommunikationsinfrastruktur im Flugzeug

Bachelorarbeit eingereicht im Rahmen der Bachelorprüfung

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

Betreuender Prüfer: Prof. Dr. Franz-Josef Korf
Zweitgutachter: Prof. Dr.-Ing. Mark Wiegmann

Eingereicht am: 17. Juli 2018

Michael Malsch

Thema der Arbeit
OMNeT++ basierte Simulation der Kommunikationsinfrastruktur im Flugzeug

Stichworte
Audio-Video Bridging, AFDX, Flugzeugnetzwerk, OMNeT++, Simulation, Rate-Constrained, OMNeT++

Kurzzusammenfassung
Diese Bachelorarbeit simuliert die Kommunikationsinfrastruktur im Flugzeug. Für die Simulation der Kommunikationsinfrastruktur wird ein Flugzeugnetzwerk und eine Kommunikations-Matrix entwickelt. Mit Hilfe der Simulationsumgebung OMNeT++ und dem CoRE4INET Framework, welches Time-Triggered Ethernet implementiert, wird das Flugzeugnetzwerk mit der Kommunikations-Matrix analysiert, evaluiert und bewertet.
Michael Malsch

Title of the paper
OMNeT ++ based simulation of the communication infrastructure in the aircraft

Keywords
Audio-Video Bridging, AFDX, Aircraft Network, OMNeT++, Simulation, Rate-Constrained, OMNeT++

Abstract
This bachelor thesis simulates the communication infrastructure in the airplane. For the simulation of the communication infrastructure an aircraft network and a communication matrix will be developed. With the help of the simulation environment OMNeT ++ and the CoRE4INET Framework, which implements Time-Triggered Ethernet, will be using the aircraft network the communication matrix is analyzed and evaluated.

Inhaltsverzeichnis

Tabellenverzeichnis

Abbildungsverzeichnis

Listings

1 Einleitung

1.1 Motivation

Bis in die späten achtziger Jahre steuerte man Flugzeuge noch rein mechanisch. Über Seilzüge und Stangen wurden die Lenkbewegungen an die Leitwerke weitergegeben. Erst mit dem Airbus A320 wurde diese Technik durch elektronische Systeme, dem sogenannten Fly-by-Wire, 1988 ersetzt.
Heute existieren in den moderneren Flugzeugen viele elektronische Komponenten, die miteinander kommunizieren. Hierzu wurden in der Vergangenheit erste Datenbusse (ARINC 429) eingeführt, die aber aufwändige und schwere Kabelbäume mit sich brachten. Da die Menge und Komplexität der zu verbindenden Komponenten in den Flugzeugen aber immer größer wird, ist die Flugzeugindustrie dazu übergegangen ihre Bussysteme für die Vernetzung weiterzuentwickeln und zu optimieren. Flugzeuge der neusten Generation wie der Airbus A380 und A350 setzen hierbei auf den ARINC 664 Standard. ARINC 664, auch als AFDX bekannt, hat sich bis jetzt aber nur in wenigen Flugzeugen der Großserien wie Airbus A350 und Boeing 787 in der Praxis durchgesetzt. In den kleineren Flugzeugserien wird weiterhin der alte Vorgänger Standard 629 verbaut. Dieser bietet mit nur 2 Mbit/s Datenrate und einem einfachen Basisprotokoll nicht die Vorzüge des AFDX-Standards, der auf einem echtzeitfähigen Ethernet mit 100 Mbit/s Datenraten basiert. Hier wäre ein modernes Bussystem vorteilhaft, welches kostengünstig in der Umsetzung ist und auch die Vorteile des AFDX-Standards mit sich bringt.

1.2 Zielsetzung und Herangehensweise

Diese Arbeit setzt sich mit der Realisierung von Echtzeit Ethernet Kommunikation im Flugzeug auseinander. Als Grundlage dienten hierbei die Ergebnisse der Arbeitsgruppe CoRE Research Group der HAW Hamburg im Automobilbereich, da sich diese technisch bereits bewährt haben. Ziel dieser Bachelorarbeit ist die Praxistauglichkeit des Time-Triggered Ethernet Protokolls im Flugzeugbereich und die technische Bewährung analog des Automobilbereichs. Hierzu wird ein realitätsnahes Datennetzwerk eines Flugzeugs entworfen. Mit Hilfe einer neu entwickelten

Kommunikationsmatrix und der Simulationsumgebung OMNeT++ werden dann verschiedene Nachrichtenprotokolle des Time-Triggered Ethernet Protokoll auf ihre Verwertbarkeit überprüft.

2 Grundlagen

In diesem Kapitel werden Grundlagen vermittelt, die zum Verständnis der Arbeit notwendig sind. Es bietet eine Einführung in die verwendeten Netzwerk-Protokolle. Zunächst wird in Abschnitt 2.1 die Etherneterweiterung TTEthernet erläutert. Dazu gibt es eine Einführung in die verschiedenen Nachrichtenklassen Time-Triggered (TT), Rate-Constrained (RC), Best-Effort-Traffic (BE) und Audio Video Bridging (AVB). Es folgen Grundlagen zum AFDX Protokoll und Definitonen verschiedener Metriken. Abschließend werden die Grundlagen zur OMNeT++ Simulationsumgebung dargestellt.

2.1 TTEthernet

Das Time-Triggered Ethernet (TTEthernet) Protokoll ist ein zeitgesteuertes Kommunikationsprotokoll, das speziell für den Einsatz in sicherheitskritischen Anwendungen, wie z. B. Flugzeug- oder Automobilbereich. Das von der Firma TTTech Computertechnik AG entwickelte Protokoll ist nach SAE AS6802 (vgl. Society of Automotive Engineers - AS-2D Time Triggered Systems and Architecture Committee (2016)) standardisiert. TTE ist eine Erweiterung des Standard Ethernet und unterstützt zeitgesteuerten Datenverkehr mit verschiedenen Nachrichtenklassen über den selben Kommunikationsweg. Es ist vollständig kompatibel mit IEEE 802.3 Ethernet und aufgeteilt in drei Nachrichtenklassen, die sich hauptsächlich in der Priorisierung der Nachrichtenübertragung unterscheiden.

2.1.1 Nachrichtenklassen

TTEthernet definiert drei verschiedenen Nachrichtenklassen:

- Time-Triggered (TT)
- Rate-Constrained (RC)
- Best Effort (BE)

Die Klassen unterscheiden sich in ihrer Priorisierung und Echtzeitanforderungen. In Abbildung 2.1 ist der Aufbau des TTE-Servicelayer dargestellt. Dieser baut auf der Ethernet 802.3 Schicht auf und ist damit kompatibel zu Protokollen in höheren OSI-Schichten, wie zum Beispiel UDP oder TCP.

Time-Triggered (TT)

TT ist die Nachrichtenklasse mit der höchsten Priorität und kann für harte Echtzeitanforderungen verwendet werden. Die TT-Klasse basiert auf dem TDMA-Verfahren. Per Offline-Scheduling werden die Zeitrahmen definiert, zu welchem Zeitpunkt ein Knoten im Netzwerk TT-Nachrichten versendet bzw. empfängt. Dazu sind synchronisierte interne Uhren auf allen teilnehmenden Knoten nötig.

Rate-Constrained (RC)

Die Rate-Constrained (RC) Klasse hat die zweithöchste Priorität. Sie wird für Anwendungen mit weniger strengen Echtzeitanforderungen verwendet. In der RC-Klasse ist zum Gegensatz

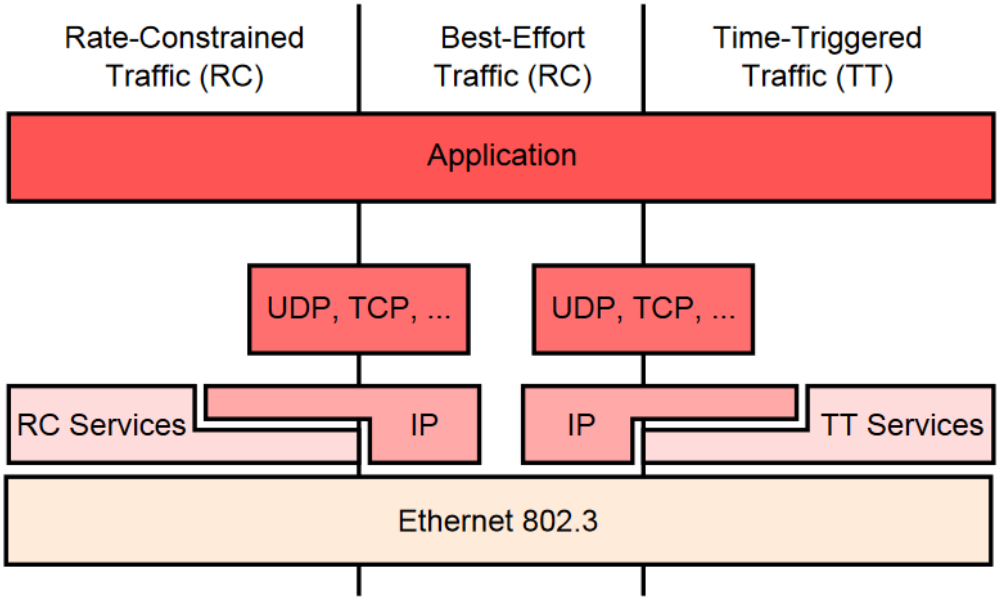

Abbildung 2.1: Aufbau und Kompatibilität des TTE Service Layer Steiner (2009)

zur TT-Klasse die Nachrichtenübertragung zeitunabhängig. Per Offline-Scheduling wird hier das Routing und die nutzbare Bandbreite definiert. Das garantiert, dass die Bandbreite für jedes Paket vordefiniert ist und Verzögerungen und zeitliche Abweichungen definierte Obergrenzen haben. Damit verhält sich die RC-Nachrichtenkommunikation fast gleich zum Standard ARINC 664 besser als AFDX 2.3 bekannt.

Best-Effort (BE)

Die Best-Effort Nachrichtenklassen besitzt die niedrigste Priorität. Hier erfolgt die Nachrichtenübertragung nach dem Standard IEEE 802.3 Ethernetverkehr. Für BE-Nachrichten gibt es keine Garantien, ob ein Paket gesendet wurde. Die von TT- und RC-Nachrichten nicht genutzte Bandbreite steht für die BE-Kommunikation zur Verfügung.

2.2 Audio Video Bridging (AVB)

Die Audio Video Bridging Task Group entwickelte für synchronisiertes und priorisiertes Streaming von Audio- und Videodaten über Netzwerke den Standard IEEE 802.1 besser bekannt als Audio Video Bridging (AVB). AVB ist eine Zusammenfassung verschiedener Standards, die in dem IEEE 802.1BA Standard definiert sind (vgl. Institute of Electrical and Electronics Engineers (2011)). In Abbildung 2.2 ist der Protokollstack mit den verschiedenen Standards abgebildet.

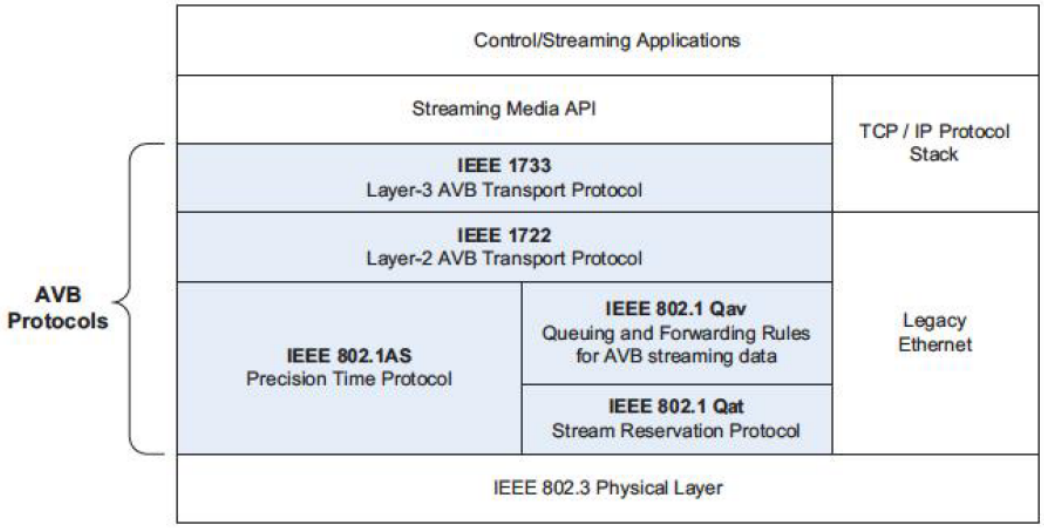

Abbildung 2.2: AVB-Protokollstack mit den verschiedenen Standards Lim u. a. (2012)

AVB setzt sich aus folgenden Standards zusammen:

- IEEE 802.1AS: Timing and Synchronization for Time-Sensitive Applications (gPTP)
- IEEE 802.1Qat: Stream Reservation Protocol (SRP)
- IEEE 802.1Qav: Forwarding and Queuing for Time-Sensitive Streams
- IEEE 802.1BA: Audio Video Bridging Systems

2.3 AFDX

Die Abkürzung steht für **A**vionics **F**ull **D**uple**X** Switched Ethernet und ist im Standard ARINC-664 Part 7 (ARINC (2005)) spezifiziert. AFDX wurde als Flugzeugdatennetzwerk der nächsten Generation konzipiert. Das Protokoll basiert auf dem Ethernet Standard IEEE 802.3 und stellt garantierte Bandbreiten und Dienstgüste (QoS) bereit. Es arbeitet mit logisch, unidirektionale Verbindung Virtual Links (VL) und bestimmbaren Verzögerungszeiten, die über die Bandwidth Allocation Gap (BAG) definiert werden. Die AFDX Netztopologie (Abbildung 2.3) ist sternförmig, d.h. jedes Endsystem (ES) ist über einen Switch mit den anderen Endsystemen verbunden. Die Hauptelemente eines AFDX Netzwerk sind:

- Endsysteme (ES)
- Switches
- Links (VL)

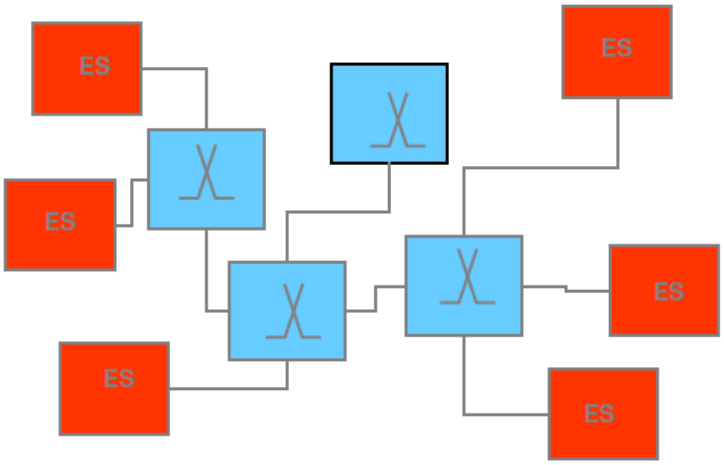

Abbildung 2.3: AFDX Netzwerktopologie (AIM GmbH)

2.3.1 Virtual Link (VL)

Ein Virtual Link (VL) ist eine logische, unidirektionale Verbindung von einem Quell-Endsystem (ES) zu einem oder mehreren Ziel-Endsystemen (Uni- oder Multicast Kommunikation). In der Abbildung 2.4 ist Aufbau dargestellt. Das Routing der Nachrichtenpakete ist statisch und wird über die Virtual Link ID festgelegt. Hier wird der Pfad bestimmt, über welchen Switch und

Port die jeweilige Nachricht geroutet wird. Jeden VL ist eine feste (garantierte) Bandbreite zugewiesen. Die Summe der Bandbreiten der einzelnen VLs ist durch die maximal verfügbare Bandbreite begrenzt. Typische Werte in modernen Flugzeugen wie dem Airbus A350 sind 10MBit bzw. 100Mbit.

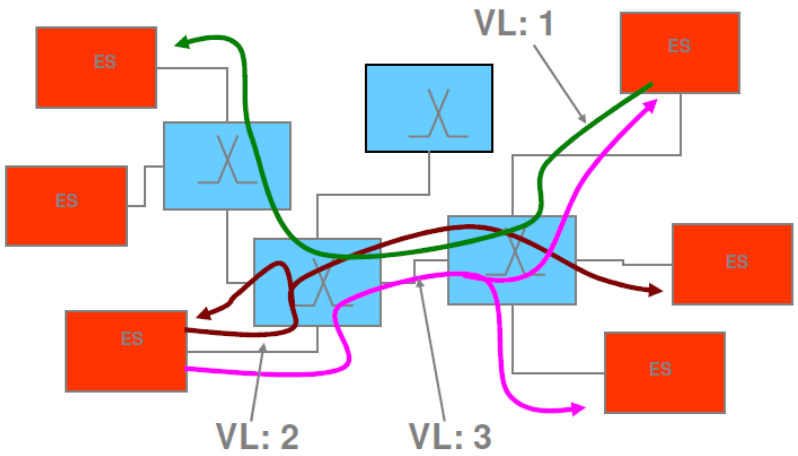

Abbildung 2.4: Darstellung der Virtual Links (AIM GmbH)

2.3.2 Bandwidth Allocation Gap (BAG)

Die Bandwidth Allocation Gap (BAG) eines Virutal Link legt das minimale Zeitintervall zwischen zwei AFDX Nachrichten eines VLs und deren Länge fest. Die BAG verhindert das Versenden von Nachrichten vor Ablauf der BAG-Zeit.

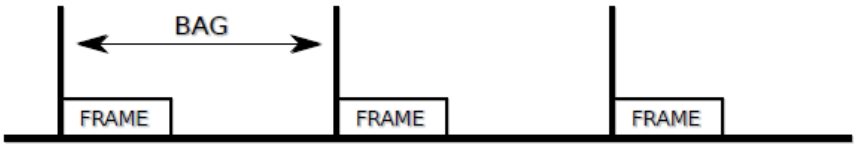

Abbildung 2.5: Darstellung der BAG mit AFDX Frame Aeronautical Radio Incorporated (2009)

Das BAG-Intervall (8 Bit) wird in Millisekunden angegeben und kann die Werte 1, 2, 4, 8, 16, 32, 64 und 128 annehmen. Mit der BAG und FrameSize lässt sich die Bandbreite des VLs bestimmten.

$$maximaleVLBandbreite = \frac{maxFrameSize}{BAG}$$

Bei einer maximalen Framelänge von 1518 Bytes und einer BAG mit 1ms ergibt sich die maximale Bandbreite des VLs von:

$$\frac{1518Bytes}{1ms} = 1518\frac{kBytes}{s}$$

2.3.3 Redundanzmanagement

Ein AFDX-Netzwerk ist so konstruiert, dass zwei unabhängige Verbindungen mit redundanten Switches zwischen jedem Endsystem (ES) existieren. In der Literatur liest man daher oft von Netzwerk A und B (siehe Abbildung 2.6). Im Standardmodus senden die Endsysteme ihre Pakete immer an beide Netzwerke. Daher erhalten die Ziel-Endsysteme immer zwei Kopien der Pakete. Das Redundanzmanagement hat die Aufgabe zu erkennen, ob ein Paket eingetroffen ist und somit das zweite Paket verworfen werden kann. Dazu orientiert sich das Redundanzmanagement an der Sequenznummer eines Paketes. Bei jeder Übertragung werden zwei Pakete mit identischer Sequenznummer versendet. Sobald ein gültiges Paket empfangen wird, wird jedes andere Paket mit der identischen Sequenznummer verworfen.

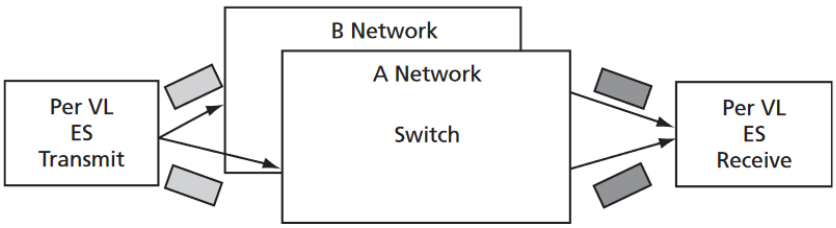

Abbildung 2.6: Schematischer Aufbau eines AFDX-Systems mit Redundanzmanagement Actel (2005)

2.4 Metriken

Metrik definiert im Netzwerkbereich die Kennzahlen zur Analyse der Güte der Verbindungen zwischen Sendern und Empfängern. Sie finden ihren Einsatzbereich in der Analyse der zur Verfügung stehende Bandbreite, Netzauslastung oder Verbindungsqualität. Im folgenden Abschnitt werden verschiedene Metriken vorgestellt.

Kapazität

Payload, der maximal auf einem Link oder Pfad pro Sekunde übertragen werden kann (vgl. Kempf (2008)).
Einheit: Mb/s
Größenordnungen: 54.7 Mb/s - 97.5 Mb/s bei einem Ethernetlink mit 100 Mb/s

Linkauslastung

Die Linkauslastung wird für jeden Link im Netzwerk einzeln gemessen. Sie zeigt die Auslastung eines Links im Verhältnis zur maximalen Bandbreite an, die innerhalb eines Intervalls übertragen werden kann. Das Intervall ist dabei nicht festgelegt, wird jedoch im Allgemeinen pro Sekunde angegeben (vgl. Kempf (2008)).
Einheit: Mb/s oder Prozent
Größenordnungen: Von 0% -100% , 0Mb/s ? Maximale Bandbreite

Verfügbare Bandbreite

Diese Metrik gibt die noch verfügbare Bandbreite auf einem einzelnen Link an. Die verfügbare Bandbreite ist das Gegenstück zur Linkauslastung. Dabei kann die Darstellung dieser Metrik prozentual oder in Mb/s angegeben werden (vgl. Kempf (2008))..
Einheit: Mb/s oder Prozent
Größenordnungen: 100 % - 0 % der maxi. Bandbreite, maximale Bandbreite des Links - 0

Latenz

Die Latenz beschreibt die Verzögerung einer Botschaft vom Absenden bis zum Empfangen. Die Latenz hat großen Einfuss auf die Quality of Service (QoS) in einem Netzwerk. Entscheidend für diese Metrik sind die minimale, die maximale und die durchschnittliche Latenz. Die minimale Latenz gibt die kleinste Verzögerung einer Nachricht zwischen zwei Punkten im Netzwerk an. Die maximale Latenz ist das Gegenstück und gibt die längste Verzögerung einer Nachricht

an. Die durchschnittliche Latenz gibt die durchschnittliche Verzögerung aller Pakete in einem Zeitintervall an. In dieser Metrik ist das Zeitintervall die gesamte Simulationszeit (vgl. Kempf (2008)).
Einheit: Sekunden (s)
Größenordnungen: (Mikrosekunden μs)

Jitter

Die Variabilität der Latenz von Paketen
Einheit: Sekunden (s)
Größenordnungen: (Mikrosekunden μs)

Paketverlustrate (PLR)

Die Metrik Paketverlustrate im Buffer gibt das Verhältnis der Pakete an, die durch einen Bufferüberlauf verloren gehen. Dabei werden alte Pakete mit neu ankommenden Paketen überschrieben. Diese Pakete werden als Dropped bezeichnet. Der Verlust von Paketen führt zum erneuten Senden und damit wiederum zu einem langsamen Netzwerk (vgl. Kempf (2008)).
Einheit: Prozent (%)
Größenordnungen: 0% - 100%

2.5 Simulationsumgebung OMNeT++

Objective Modular Network Testbed (OMNeT++) ist ein objektorientiertes modulares Netzwerk-Simulationsframework für diskrete Ereignisse. Es hat eine generische Architektur, so dass es für verschiedene Problembereiche verwendet werden kann wie z. B.:

- Modellierung von drahtgebundenen und drahtlosen Kommunikationsnetzen
- Protokollmodellierung
- Modellierung von Multiprozessoren und anderen verteilten Hardwaresystemen
- Jede Anwendung, die sich mit dem Ansatz der diskreten Ereignisse und Nachrichtenaustausch modellieren lässt (vgl. OMNeT++ Manual)

Die auf Eclipse basierende Entwicklungsumgebung kann unter Windows, Mac OS X und Linux betrieben werden. OMNeT++ ist eine Open Source-Umgebung, die für akademische sowie private Zwecke unter der GNU General Public License (GPL) kostenlos verwendbar. Für die Erstellung von Modellstrukturen verwendet OMNeT++ eine eigene Sprache namens NED (Network Description). Komponenten für Funktionen werden in der Prgrammiersprache C++ entwickelt.

2.5.1 Modellierungsaufbau

Ein OMNeT++ - Modell besteht aus Modulen, die mit Hilfe von Nachrichten untereinander kommunizieren. Man unterscheidet zwischen zwei Arten von Module:

- *Simple-Modules* Sie implementieren das Verhalten des Modells. Sie sind in C++ geschrieben und unter Verwendung der Simulationsklassenbibliothek.
- *Compound-Modules* Die Simple-Modules können weiter gruppiert werden. Die Anzahl der Hierarchieebenen ist unbegrenzt.

Die Verbindungen zwischen den Modulen heißen Connections und werden mit sogenannten Gates verbunden. Das gesamte Modell wird in OMNeT++ als Netzwerk bezeichnet und ist selbst ein zusammengesetztes Modul. In Abbildung 2.7 ist der Aufbau dargestellt.

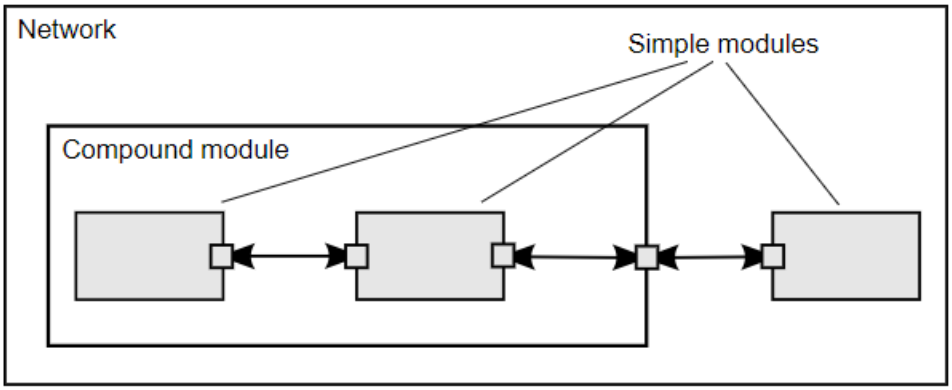

Abbildung 2.7: Aufbau OMNeT Modell mit verschiedenen Modulen OMNeT++ Manual

2.5.2 NED

In folgenden Beispiel der Beschreibungssprache (Network Description) wird der Aufbau von NED verdeutlicht. Es wird dabei ein Netzwerk mit mehreren Knoten (Nodes) aufgebaut. Node1 ist mit einer 100 Mb/s Verbindung über den Switch1 mit Node2 und Node3 verbunden. In Abbildung 2.8 ist das Beispiel dann als Netzwerkbild dargestellt.

```
network small_network
{
        submodules:
                node1: Node1 {}
                node2: Node2 {}
                node3: Node3 {}

        switch1: Switch1 {
                parameters:
                gates: ethg[3];
        }
        connections:
                node1.ethg <--> Eth100M <--> switch1.ethg[0];
                node2.ethg <--> Eth100M <--> switch1.ethg[1];
                node3.ethg <--> Eth100M <--> switch1.ethg[2];
}
```

Listing 2.1: Beispiel der NED-Sprache

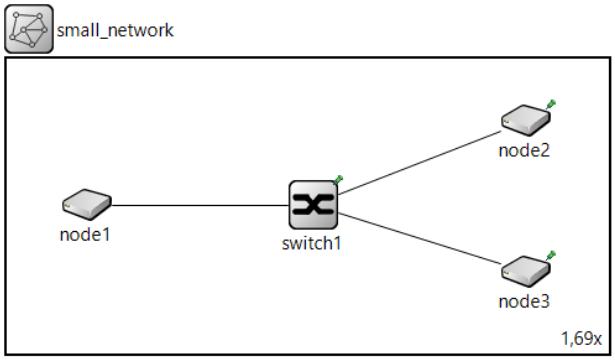

Abbildung 2.8: NED Quellcode grafisch dargestellt

3 Analyse

In diesem Kapitel wird die Modellierung der Kommunikationsinfrastruktur erläutert. Zu Beginn werden die Anforderungen an die Erstellung einer Kommunikationsmatrix erörtert. Anschließend wird auf das Design moderner Flugzeugnetzwerke eingegangen. Im zweiten Abschnitt folgt dann eine Anforderungsermittlung zur verwendeten Simulationssoftware OMNet++.

3.1 Modellierung

Die Bachelorarbeit entsteht im Umfeld der CoRE-Arbeitsgruppe (CoRE RG) der Hochschule für Angewandete Wissenschaften Hamburg. Diese Projektgruppe befasst sich mit modernen Real-time Ethernet-Fahrzeugnetzen im Automobilbereich, daher kann nicht auf vorhandene Entwicklungen aus dem Flugzeugbau zurückgegriffen werden.
Hauptgrundlage für diese Arbeit ist die Entwicklung einer Kommunikationsmatrix und ein Flugzeugnetzwerk. Entsprechende moderne Kommunikationsmatrizen aus Flugzeugen sind öffentlich nicht zugänglich, daher ist eine komplette Neuentwicklung in Eigenregie der Kommunikationsmatrix nötig.
Für die Modellierung und Entwicklung werden etablierte Techniken aus dem Automobilbereich herangezogen bzw. als Vorlage genutzt.

3.1.1 Aufbau einer Kommunikationsmatrix (K-Matrix)

Signale stellen Kommunikationsbeziehungen zwischen einzelnen Endsystemen im Automobil bzw. Flugzeug da. Diese sind sogenannte Sender-Empfänger-Beziehungen, bei denen jede Kommunikationsnachricht die benötigten Informationen von einem Sender zu einem oder mehreren Empfängern für das Netzwerk bereithält.
Alle Kommunikationsbeziehungen in einem Netzwerk können als Sender-Empfänger-Beziehungen tabellarisch in einer sogenannten Kommunikationsmatrix, kurz K-Matrix zusammengefasst werden. Die K-Matrix enthält dann alle kommunikationsrelevanten Informationen des Netzwerks bereit (vgl. Schäuffele und Zurawka (2013)).
Ein Beispiel einer K-Matrix wird in Abbildung 3.1 dargestellt. Ein unterschiedlicher Aufbau

Netzknoten	Nachricht	Signal	ABS-Steuergerät	Motorsteuergerät	Getriebesteuergerät	⋰
ABS-Steuergerät	ABS-1	Raddrehzahl vorne links	S		E	
		Raddrehzahl vorne rechts	S		E	
						...
	ABS-2	Raddrehzahl hinten links	S		E	
		Raddrehzahl hinten rechts	S		E	
Motorsteuergerät	MS-1	Fahrpedalwert		S	E	
		Motordrehzahl	E	S	E	
						...
...	...	...	...	...	...	...

Abbildung 3.1: Kommunikationsmatrix mit einer Sortierung nach Netzknoten Schäuffele und Zurawka (2013)[S. 90].

einer K-Matrix ist möglich. In der linken Spalten sind alle Netzknoten, also alle Steuergeräte, aufgelistet. In folgenden Spalten werden die Nachrichten und die Nutzdaten in Form vom Signalen aufgeführt, die vom jeweiligen Netzknoten gesendet werden. In den nächsten Spalten folgen dann wieder erneut Netzknoten. Hier wird mit E die Empfänger gekennzeichnet, welcher Netzknoten welche Nachricht empfängt und auswertet. Das S beschreibt den Sender der Nachricht.

3.1.2 Eigenschaften der Kommunikationsmatrix

Eine Kombination dieser Kommunikationsmatrix mit den Eigenschaften vgl. (Tindell und Burns (1994)) der einzelnen Nachrichten und den Anforderungen, die sich für jede Nachricht ergeben, ist dabei für eine vollständigere Kommunikationsmatrix anzustreben. Gerade bei zeitkritischen Systemen ist die Angabe von Eigenschaften in den Nachrichten nötig. Ein Aufbau einer solchen Matrix sieht man in Tabelle 3.1.
In der linken Spalte bekommt jedes Signal aufsteigend eine Nummer zugewiesen. Die nächste Spalte enthält die Beschreibung des Signals. In den folgenden Spalten stehen dann die wichtigsten Eigenschaften des Signals. Erst die Signalgröße in bits, dann der Jitter in Millisekunden,

Signal Number	Signal Description	Size /bits	Jitter /ms	Time /ms	Periodic / Sporadic	Sender	Receiver
1	Traction B. Voltage	8	0.6	100.0	P	Battery	V/C
2	Traction B. Current	8	0.7	100.0	P	Battery	V/C
3	Traction B. Temp	8	1.0	1000.0	P	Battery	V/C
4	Auxiliary B. Voltage	8	0.8	100.0	P	Battery	V/C

Tabelle 3.1: K-Matrix mit Eigenschaften der Nachrichten nach Tindell und Burns (1994) Bedeutung der Abkürzungen: B. = batteries, V/C = vehicle controller

danach die Frequenz des Signals in Millisekunden. Daneben kennzeichnet ein P, ob das Signal periodisch oder mit S für sporadisch, versendet wird. Den Abschluss bilden die zwei letzten Spalten, die den Sender und den Empfänger beschreiben.

3.1.3 Flugzeugnetz Design

Moderne Flugzeuge wie der Airbus A380, der Airbus A400 und die Boeing-777 basieren heutzutage auf dem AFDX-Standard. AFDX steht für Avionics Full DupleX Switched Ethernet.
Das Flugzeugnetzwerk ist ein zweifach redundantes, echtzeitfähiges geswitchtes Ethernet. Es besteht aus einem primären Netzwerk A (rot) und einem sekundären Netzwerk B (blau), die stets die gleichen Daten führen und daher redundant zueinander sind.

Abbildung 3.2 zeigt eine schematische Darstellung eines AFDX-Netzes in einem Flugzeug. Das Netzwerk A wird durch rote Rechtecke (Switch) und rote Verbindungen symbolisiert. Netzwerk B wird durch blaue Rechtecke (Switch) und blauen Verbindungen dargestellt. Netz A und B sind identisch im Aufbau, aber unabhängig von einander. Die weißen Rechtecke stehen für die Rechnereinheiten (IMA) der einzelnen Systeme. Sie haben jemals eine Verbindung in das Netz A (rote Linie) und eine Verbindung in das Netz B (blaue Linie).
Die Avionik basiert überwiegend auf der Architektur der Integrated Modular Avionics (IMA). Zusätzliche Input-Output-Module (IOM) dienen dazu, Systeme und Sensoren in das AFDX-Netzwerk einzubinden, die kein eigenes AFDX-Interface besitzen. Dabei sind die Avionik-Funktionen für Klimaanlage, Zapfluft, Cockpit-Datenkommunikation und Bord-Boden-Datenrouting, elektrische Stromversorgung, Treibstoff-Management, Fahrgestell, Bremsen und Lenkung auf insgesamt acht verschiedenen Typen von IMA-Rechnern integriert. Die IMA-Rechner, auch CPIOM (Core Processing Input/Output Module) genannt, basieren auf identischen PowerPC-Prozessoren, jedoch unterscheiden sie sich in den spezifischen Signalschnittstellen für die jeweils auf den Modulen integrierten Systeme.

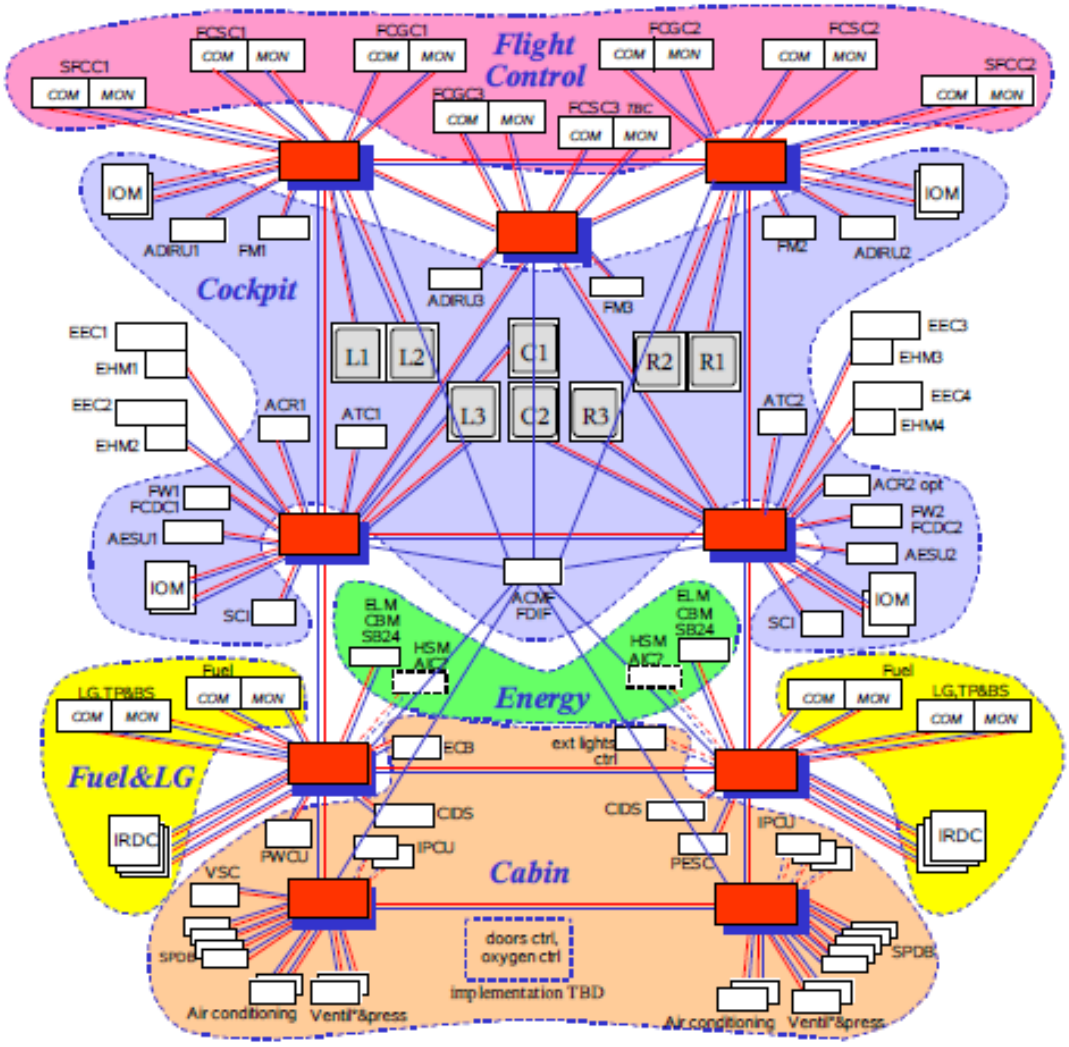

Abbildung 3.2: AFDX-Netzwerk im Flugzeug David und Rockwell (2004)

3.2 Simulation der Kommunikationsinfraktionstruktur

3.2.1 Simulationssoftware OMNeT++

In dieser Arbeit wird das Simulationsframework OMNeT++ verwendet. Durch den modularen Aufbau von OMNeT++ lässt es sich leicht durch Frameworks erweitern. In der Abbildung 3.3 sind die Framework dargestellt, welche die Simulationskomponenten bereitstellen, die für die Simulation benötigt werden.

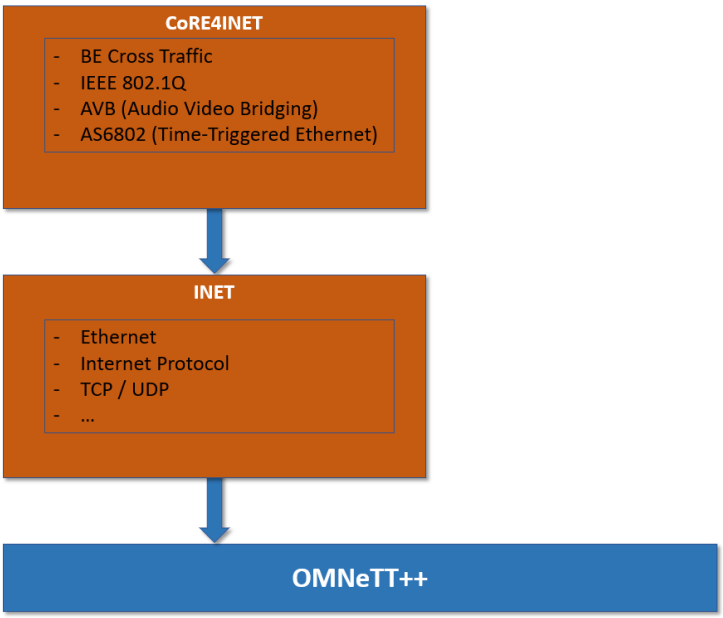

Abbildung 3.3: Modularer Aufbau der Frameworks

3.2.2 INET Framework

Das INET Framework OMNeT++ Community ist eine Open Source Bibliothek verschiedener Netzwerk-Protokolle und Netzwerkkomponenten. Sie erweitert den Funktionsumfang von OMNeT++ mit Protokollen wie z. B. UDP, TCP, SCTP, IPv4, IPv6, Ethernet, IEEE 802.11 etc. Für diese Arbeit wird auf den Ethernet-Stack und typische Netzwerkkomponenten wie Switch, Server und Clients zurückgegriffen.

3.2.3 CoRE4INET Framework

CoRE4INET ist eine Erweiterung des INET-Frameworks für die Simulationumgebung OMNeT++ (vgl. CoRE4INET). Das Framework wurde von der CoRE Research Group entwickelt und beinhaltet folgende Protokolle:

- AS6802
  - Time-Triggered (TT)
  - Rate-Constrained (RC)
  - Best Effort (BE)
- IEEE 802.1 Audio Video Bridging (AVB)

3.2.4 Anforderungen an die Simulation

In der Simulation soll der grundlegende Aufbau eines AFDX-Netzwerks nachgebildet werden. Der Rate-Constrained-Traffic-Modus unterstützt zu großen Teilen die Konzepte des AFDX-Protokolls, dass im Abschnitt 2.3 beschrieben wird. Die Simulation stellt eine sogenannte System-Level-Simulation dar, d.h. der Fokus liegt auf der Bitübertragsungsschicht (Physical Layer).

4 Konzept

In diesem Kapitel wird das Konzept der Kommunikationsmatrix (kurz K-Matrix) präsentiert und die Auswahl der dazugehörigen Systemkomponenten festgelegt. Es werden die Nachrichtenklassen bestimmt und verschiedene Entwürfe für das Flugzeugnetz-Design vorgestellt. Die Konzepte und Entwürfe basieren auf den Ergebnissen der Analyse und den Rahmenbedingungen der verwendeten Simulation.

4.1 Kommunikationsmatrix

Für die Erstellung der Kommunikationsmatrix orientiere ich mich an aus der Analyse vorgestellen Beispiele von Schäuffele und Zurawka (2013) und Tindell und Burns (1994). Die K-Matrix wird dazu in Anlehnung an Abbildung 3.2 in folgende 6 Hauptsysteme unterteilt:

- Cockpit 4.1
- Cabin 4.2
- Engines 4.4
- Energy 4.4
- Fuel & Landing Gear 4.5
- Flight Control 4.6

Den Aufbau der K-Matrix sieht man beispielhaft in Tabelle 4.1. Am Tabellenkopf steht der zugehörige Bereich des Endsystems. In diesem Fall beschreibt die Tabelle die K-Matrix des Cockpits. In der linken Spalte werden die Sender Systeme als Abkürzung genannt. Die Systemkomponenten und ihre Abkürzungen sind den jeweiligen Tabellen 4.1.3 beschrieben. Die nächste Spalte beschreibt die Signale, die der Sender an die jeweiliger Empfänger sendet. In den folgenden Spalten wird die Zeit in Millisekunden genannt, mit welcher Periode der Sender das Signal an die Empfänger gesendet. Dabei unterscheidet die Tabelle, ob das Signal nach dem ARINC-629 Standard 4.1.1 oder nach einem geschätzten Wert versendet wird.

4.1.1 Periodenzeiten der Endsysteme

Moderne Flugzeuge von Airbus und Boeing benutzen den ARINC-629 Standard um verschiedene Systeme miteinander zu verbinden. Airbus nutzt das Bussystem im A340 und A330. Der Begriff ARINC steht für die Abkürzung Firma Aeronautical Radio Incorporated. Der Standard ARINC-629, ist eine Spezifikation, in der festgelegt wird, wie Systeme über das Netzwerk Daten empfangen und senden können.
Da die Firmen Airbus und Boeing keinen Zugriff auf ihre Kommunikationsprotokolle ihrer AFDX-Systeme freigeben, werden die Zeit-Eigenschaften (Sendeperiode), wenn möglich auf Basis der Daten des ARINC-629 entwickelt, damit ebenfalls realitätsnahe Daten mit in die Entwicklung fließen können. Falls für eine Senderkomponente keine Sendezeiten vorliegen, werden die Zeiten durch geschätzte Zeiten ersetzt.

4.1.2 Aufbau K-Matrix Hauptsysteme

Cockpit				
Sender - System	Signal Beschreibung	Periode ARINC 629 / ms	Periode / ms	Receiver
FCDC 1	Status Message	125		ECAM1/ECAM2/FMS1/FMS2
FCDC 2	Status Message	125		ECAM1/ECAM2/FMS1/FMS2
FW 1	Engine Warning1	-	125	ECAM1/ECAM2
...	...	...	...	...

Tabelle 4.1: K-Matrix Flugzeug Cockpit

Cabin				
Sender - System	Signal Beschreibung	Periode ARINC	Periode / ms	Receiver
CIDS1	Info Message1	-	500	FMS1, FMS2, FMS3
CIDS2	Info Message2	-	500	FMS1, FMS2, FMS3
IPCU1	Antifreeze Status1	-	500	FMS1, FMS2, FMS3
...	...	...	...	...

Tabelle 4.2: K-Matrix Flugzeug Cabin

Engines				
Sender - System	Signal Beschreibung	Periode ARINC	Periode / ms	Receiver
FADEC 1	Engine Rotational Speed1	125	-	FCPC1/FCPC2/FCPC3/EEC/EEC2
FADEC 2	Engine Rotational Speed2	125	-	FCPC1/FCPC2/FCPC3/EEC/EEC2
FADEC 3	Engine Rotational Speed3	125	-	FCPC1/FCPC2/FCPC3/EEC/EEC2
...	...	...	...	...

Tabelle 4.3: K-Matrix Flugzeug Engines

Energy				
Sender - System	Signal Beschreibung	Periode ARINC	Periode / ms	Receiver
ELM 1	GeneratorB Output1	-	250	FCPC1/FCPC2/FCPC3/FCSC1/FCSC2
ELM 2	GeneratorB Output2	-	250	FCPC1/FCPC2/FCPC3/FCSC1/FCSC2
HSMU 1	Hydraulic Pressure1	-	500	ECAM1/ECAM2
...	...	...	...	...

Tabelle 4.4: K-Matrix Flugzeug Energy

Fuel & Landing Gear				
Sender - System	Signal Beschreibung	Periode ARINC	Periode / ms	Receiver
FQMS 1	Monitoring Centerof Gravity1	-	250	ECAM1/ECAM2/FMS1/FMS2/FMS3
FQMS 2	Monitoring Centerof Gravity2	-	250	ECAM1/ECAM2/FMS1/FMS2/FMS3
BCS 1	Monitoring Brake TempNose 1	-	250	ECAM1/ECAM2
...	...	...	...	...

Tabelle 4.5: K-Matrix Flugzeug Fuel & Landing Gear

Flight Control				
Sender - System	Signal Beschreibung	Periode ARINC	Periode / ms	Receiver
SFCC 1	Flaps Up1	62.5	-	FMS1/FMS2/FMS3
SFCC 2	Flaps Up2	62.5	-	FMS1/FMS2/FMS
SFCC 1	Slaps Down1	62.5	-	ECAM1/ECAM2
...	...	...	...	...

Tabelle 4.6: K-Matrix Flugzeug Flight Control

4.1.3 Systemkomponenten der K-Matrix

Die Auswahl der Systemkomponenten der K-Matrix wird durch eigene Recherche erstellt, da auf keinen öffentlichen, zugänglichen Datensatz zurückgriffen werden kann. Als Vorlage dient die Abbildung 3.2 des A380 AFDX-Netzwerk von David und Rockwell (2004). Durch Internet und Buch Recherche kommen folgende Komponentenlisten der Systeme zustande. Die Listen sind nach den Hauptbereiche Flight Control 4.7 , Cockpit 4.9 , Energy 4.10 , Fuel & Landing Gear 4.5 und Cabin 4.12 aufgebaut. In ersten Spalte steht die Abkürzung des Systems mit der auch in der K-Matrix 4.6 gearbeitet wird. Den kompletten Namen der Systemkomponente beschreibt die zweite Spalte. Je nach Wichtigkeit ist das System mehrmals redundant. Z. B. in der Tabelle 4.7 Flight Control ist der Flight Control Primary Computer gleich dreimal vorhanden und zusätzlich besitzt jeder Flight Control Primary Computer noch einen Flight Control Secondary Computer.

Flight Control	
Abkürzung	Name
SFCC1	Slats & Flats Control Computer 1
SFCC2	Slats & Flats Control Computer 2
FCPC1	Flight Control Primary Computer 1
FCSC1	Flight Control Secondary Computer 1
FCPC2	Flight Control Primary Computer 2
FCSC2	Flight Control Secondary Computer 2
FCPC3	Flight Control Primary Computer 3
FCSC3	Flight Control Secondary Computer 3

Tabelle 4.7: Bedeutung der Abkürzungen Bereich Flight Control

Engines	
Abkürzung	Name
EEC1	Electronic Engine Computer 1
EEC2	Electronic Engine Computer 2
EEC3	Electronic Engine Computer 3
EEC4	Electronic Engine Computer 4
FADEC1	Full Authority Digital Engine Control 1
FADEC2	Full Authority Digital Engine Control 2
FADEC3	Full Authority Digital Engine Control 3
FADEC4	Full Authority Digital Engine Control 4

Tabelle 4.8: Bedeutung der Abkürzungen Bereich Engines

Cockpit	
Abkürzung	Name
FMS1	Flight Management System 1
FMS2	Flight Management System 2
FMS3	Flight Management System 3
ADIRU1	Air Data Inertial Reference Unit 1
ADIRU2	Air Data Inertial Reference Unit 2
ADIRU3	Air Data Inertial Reference Unit 3
SCI1	Secure Communication Interface 1
SCI2	Secure Communication Interface 2
IOM1	Input Output Module 1
IOM2	Input Output Module 2
ATC1	Air Traffic Control 1
ATC2	Air Traffic Control 2
ACR1	Avionics Communication Router 1
ACR2	Avionics Communication Router 2
FW1	Flight Warning System 1
FW2	Flight Warning System 2
AESU1	Aircraft Environment Surveillance Unit 1
AESU2	Aircraft Environment Surveillance Unit 2
FCDC1	Flight Control Data Concentrator 1
FCDC2	Flight Control Data Concentrator 2
ECAM1	Electronic Centralized Aircraft Monitoring 1
ECAM2	Electronic Centralized Aircraft Monitoring 2
EHA1	Electro Hydrostatic Actuator 1
EHA2	Electro Hydrostatic Actuator 2

Tabelle 4.9: Bedeutung der Abkürzungen Bereich Cockpit

Energy	
Abkürzung	Name
HSMU1	Hydraulic System Monitoring Unit 1
HSMU2	Hydraulic System Monitoring Unit 2
CBMS1	Circuit Breaker Monitoring System 1
CBMS2	Circuit Breaker Monitoring System 2
ELM1	Electrical Load Management 1
ELM2	Electrical Load Management 2

Tabelle 4.10: Bedeutung der Abkürzungen Bereich Energy

Fuel & Landing Gear	
Abkürzung	Name
FQMS1	Fuel Quantity Management System 1
FQMS2	Fuel Quantity Management System 2
BCS1	Braking Control System 1
BCS2	Braking Control System 2
SCS1	Steering Control System 1
SCS2	Steering Control System 2
TPCS1	Tyre Pressure Control System 1
TPCS2	Tyre Pressure Control System 2
LGCIS1	Landing GearControl and Indicating System 1
LGCIS2	Landing GearControl and Indicating System 2

Tabelle 4.11: Bedeutung der Abkürzungen Bereich Fuel & Landing Gear

Cabin	
Abkürzung	Name
SPDB1	Secondary Power Distribution System 1
SPDB2	Secondary Power Distribution System 2
SPDB3	Secondary Power Distribution System 3
SPDB4	Secondary Power Distribution System 4
SPDB5	Secondary Power Distribution System 5
SPDB6	Secondary Power Distribution System 6
SPDB7	Secondary Power Distribution System 7
SPDB8	Secondary Power Distribution System 8
CPCS1	Cabin Pressure Control System 1
CPCS2	Cabin Pressure Control System 2
CPCS3	Cabin Pressure Control System 3
CPCS4	Cabin Pressure Control System 4
AIR-COND1	Air Conditioning 1
AIR-COND2	Air Conditioning 2
IPCU1	Ice Protection Control Unit 1
IPCU2	Ice Protection Control Unit 2
DSCS1	Door (and Escape) Slides Control System 1
DSCS2	Door (and Escape) Slides Control System 2
VCS1	Ventilation Control System 1
VCS2	Ventilation Control System 2
PEPDC1	Primary Electrical Power Distribution Centre 1
PEPDC2	Primary Electrical Power Distribution Centre 2
PWS	Potable Water System
ECB	Electronic Control Box
CIDS1	Cabin Intercommunication Data System 1
CIDS2	Cabin Intercommunication Data System 2

Tabelle 4.12: Bedeutung der Abkürzungen Bereich Cabin

4.2 Auswahl Nachrichtenklassen

In dieser Arbeit wird das Simulationsframework OMNeT++ verwendet, da hierfür im Rahmen der CoRE-Projektgruppe ein CoR4INET-Framework entwickelt wurde, welches Simulationskomponenten für die Simulation folgender Nachrichtenklassen bereitstellt:

- AS6802
  - Time-Triggered (TT)
  - Rate-Constrained (RC)
  - Best Effort (BE)
- IEEE 802.1 Audio Video Bridging (AVB)

Moderne Flugzeugnetze basieren auf dem AFDX-Protokoll. Die Rate-Constrained (RC) Nachrichtenklasse unterstützt zu großen Teilen die Konzepte des AFDX- Protokolls. Die Wahl fällt daher auf die Nachrichtenklasse Rate-Constrained (RC), um einen realitätsnahen Bezug darzustellen.
Zusätzlich wird noch das Streaming basierte Protokoll Audio/Video Bridging (AVB) gewählt, um weitere Simulationsergebnisse zu analysieren und mögliche Vorteile bzw. Nachteile der beiden gewählten Protokolle zu verdeutlichen.

4.3 Sequenznummer für RC- und AVB-Nachrichten

AFDX-Systeme besitzen wie in Abschnitt 2.3.3 beschrieben ein Redundanzmangement inklusive einer Sequenznummer zum Differenzieren der Nachrichtenpakete. Im Frame-Aufbau der RC- und AVB-Nachrichten ist diese Information nicht vorhanden. Durch Änderungung des Frame-Aufbaus von RC und AVB wird die Funktion zum Mitführen einer Sequenznummer ermöglicht.

Abbildung 4.1: Schematischer Aufbau des Frames mit Sequenznummer

In Abbildung 4.1 ist der schematische Aufbau für des AVB- bzw. RC-Frames mit Sequenznummer dargestellt. Es wird einfach ein Feld mit der Sequenznummer zwischen Nachrichtentyp und dem Payload eingefügt. Der ursprüngliche Payload wird dadurch nur unbedeutend verkleinert.

4.4 Entwurf des Flugzeugnetzes

Im letzten Teil der Konzeption wird auf den Entwurf des Flugzeugnetzes eingegangen. Die hier vorgestellten Design-Konzepte beruhen auf der Analyse aus Kapitel 3.1.3. Sie bilden die Grundlage für die verschiedenen Simulationsmodelle für OMNeT++. Durch die Wahl der Nachrichtenklassen in Abschnitt 4.2 ergeben sich Unterschiede im Netzwerk-Design.

4.4.1 Flugzeugnetz-Architekturen

Die Wahl der Nachrichtenklassen RC und AVB hat verschiedene Netzarchitekturen mit unterschiedlichen Typologien zur Folge:

4.4.2 Entwurf des RC-Netzwerks

Die Architektur des RC-Flugzeugnetzes basiert auf einer vermaschten Topologie (siehe Abbildung 4.2). Das Netzwerk besteht aus zwei redundanten Netzen A (blau) und B (rot). Netzwerk A ist gekennzeichnet durch blaue Switches und blaue Verbindungen - Netzwerk B durch rote Switches und rote Verbindungen. Ein Netzwerk besteht jeweils aus 9 Switches. Die weißen Rechtecke stellen die einzelnen Endsysteme dar. Eine Systemkomponente ist mit seinen redundanten Schwestersystemen jeweils an Netzwerk A und B gekoppelt. Die farbliche Gruppierung beschreibt die Zugehörigkeit zum jeweiligen Bereich. Die 6 Bereiche sind Flight Control, Cockpit, Engines, Energy, Fuel & Landing Gear und Cabin. Die Abbildung 4.2 zeigt von oben absteigend die kritischen Systeme für den Betrieb des Flugzeuges.

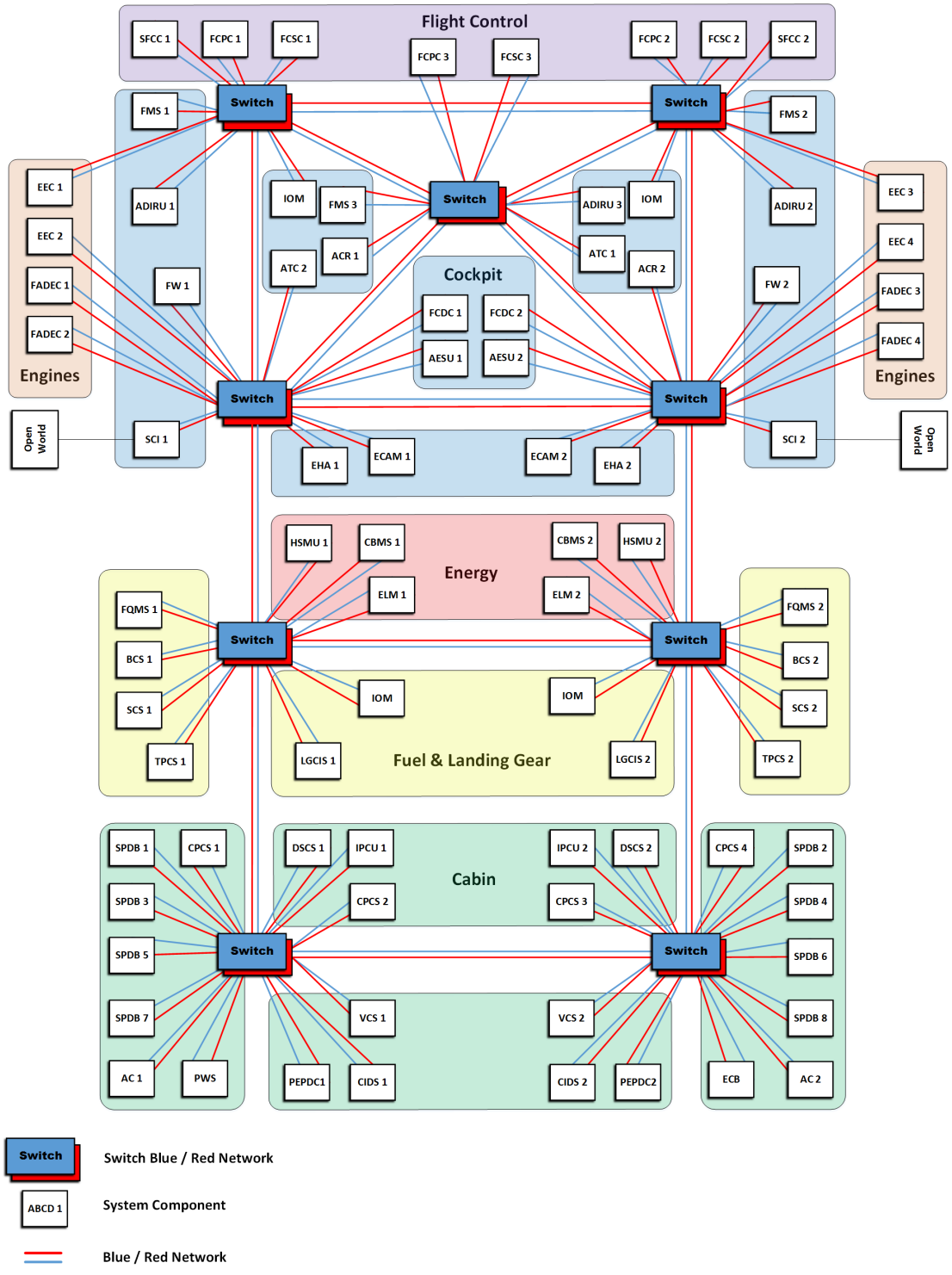

Abbildung 4.2: Konzept RC-Netz-Entwurf

4.4.3 Entwurf des AVB-Netzwerks

Der ursprüngliche Netzentwurf (Abbildung 4.2) erfährt eine Änderung aufgrund des AVB-Protokolls. Das AVB-Protokoll kann auf die vermaschte Netztopologie nicht angewendet werden. Im Gegensatz zur RC-Nachrichtenklasse, die ein statisches Routing besitzt, ist bei der AVB-Nachrichtenklasse ein dynamisches Routing vorhanden. Würde man die ursprüngliche Netztopologie beibehalten, würde es zu so genannten Switching-Loop kommen. Der Effekt entsteht wenn ein Layer-II-Broadcast-Paket an einen Switch verschickt wird. Dies passiert regelmäßig im Zuge der Adressauflösung mittels Address Resolution Protocol. Durch die Ringtopologie kommt es dann zum Broadcast-Sturm. Um diesen Effekt zu vermeiden, wird die Netztopologie verändert. In Abbildung 4.3 ist der Entwurf des AVB-Netzwerks dargestellt. Die Anzahl und Art der Endsysteme sind im AVB-Netzwerk weiterhin mit des RC-Netzwerks identisch.

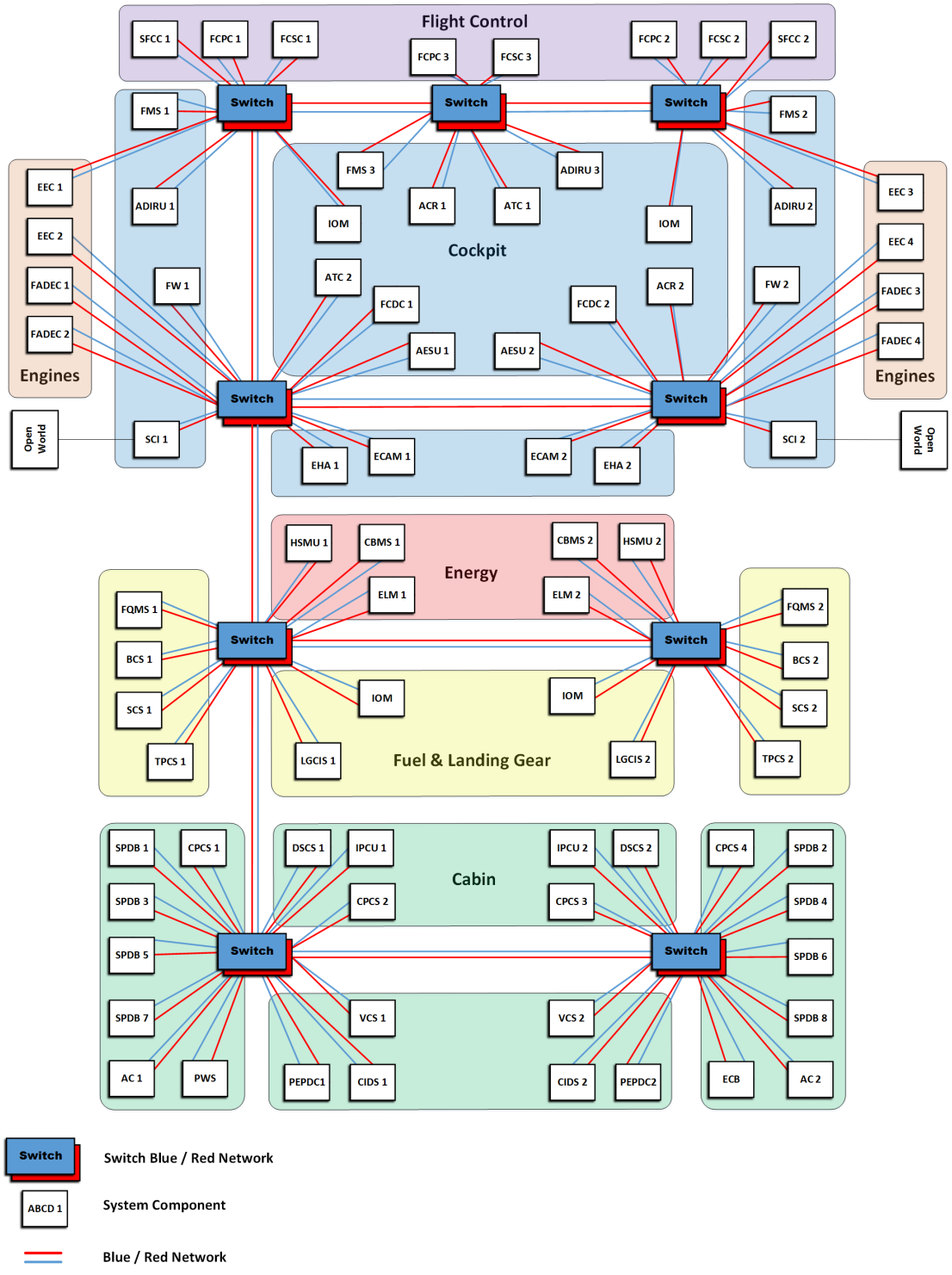

Abbildung 4.3: Konzept AVB-Netz-Entwurf

5 Entwurf des Simulationsmodells

In diesem Kapitel geht es um die Umsetzung der Konzepte. Hier werden die Simulationsmodelle der Nachrichtenklassen AVB und RC entworfen. Im ersten Teil wird der Aufbau des Nodes beschrieben und auf die nötige Erweiterung der Portanzahl eingegangen. Danach folgt die Implementierung der Nachrichten-Differenzierung per Sequenznummer. Im letzten Teil werden die Entwürfe der Simulationsmodelle für die zwei Nachrichtenklassen vorgestellt.

5.1 Aufbau eines Nodes

Um in OMNeT++ ein Netzwerk zu simulieren zu können, werden vorhandene TTE-AVB-Node und TTE-AVB-Switch Module aus dem CoRE4INET Framework benutzt. Die Nodes, die im Netzwerk, die jeweiligen Endsysteme darstellen, müssen für den Netzentwurf modifiziert werden. Daher folgt ein Einblick in den Aufbau und Funktionsweise eines Nodes. Die Abbildung 5.3 zeigt den Aufbau eines Nodes für AVB, RC und Best Effort Nachrichten. Die RC-Komponenten sind gelb, die AVB-Komponenten sind blau und Best Effort-Komponenten sind grün dargestellt. Die Pfeile repräsentieren feste uni- oder bidirektionale Verbindungen über die die Nachrichten verlaufen. Manche Pfeilverbindungen sind aber nur zur Laufzeit sichtbar. vl_1_ctc und vl_1 Module stellen virtuelle Links mit der dazugehörigen virtuellen Link-ID der RC-Nachrichten da. Diese resultieren pro Link in ein „RCIncoming “ (CTC) und ein „RCBuffer“Modul und sind beim Port registriert. Pakete mit der zugehörigen virtuellen Link ID werden an das jeweilige ?Incoming? gesendet. Das Modul „app “, kann mehrere Stimuligeneratoren für verschiedene Nachrichtenklassen beinhalten. Die Module für AVB Nachrichten bestehen aus dem AVBIncoming (avbCTC) und für die verschiedenen AVB-Klassen je einen Buffer (avbABuffer und avbBBuffer). Für das Streaming Reservation Protokoll (SRP) werden die Module srpProtocol und srpTable benutzt. Die Übertragungseigenschaften des SRP für die Kommunikationsteilnehmern Talker und Listener werden im encap Modul in eine Ethernet-Nachricht verkapselt. Das Modul phy stellt den physischen Port des Nodes da.

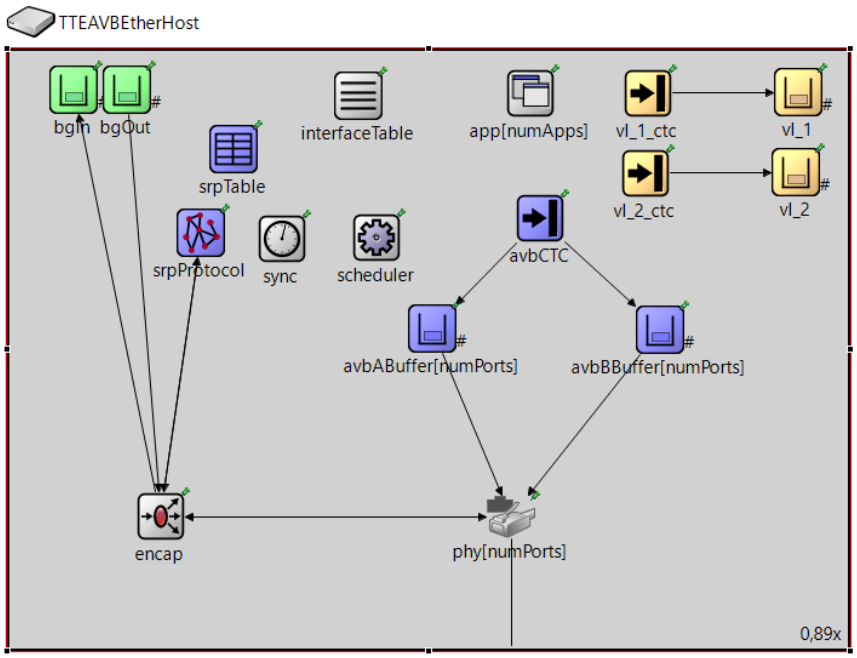

Abbildung 5.1: Aufbau eines AVB-TTE-Nodes mit einem Port

5.2 Porterweiterung des Nodes

Die bisher verwendenden Nodes in OMNeT++ sind für den Automobilbereich ausgelegt und haben daher nur einen Portanschluss je Node. Durch die Redundanz der Systemkomponenten und damit die Koppelung an jeweils zwei Switches, hat eine Erweiterung der Portanzahl des Nodes zur Folge. Der bisher benutzte Node (Abbildung 5.3) wird nur mit einem Port pro Node genutzt. Daher wird der vorhanden Node mit einem Portanschluss weiterentwickelt, damit man zwei Portanschlüsse an einem Node zur Verfügung hat.

Entwicklung SRPEtherEncapAdapter

Durch die Erhöhung der Portanzahl auf zwei Anschlüsse entstehen Probleme für die Funktionalität der AVB Nachrichten im Node. Das „encap“ und „srpProtocol“ Modul sind für mehr als eine Nachricht nicht ausgelegt. Durch die Erhöhung der Portanzahl entsteht ein weiteres „encap“ Modul für das zweite Nachrichtenpaket. Das srpProtocol Modul ist aber nicht für zwei „encap“ Module kompatible. Diese Problematik führt zur Entwicklung des SRPEtherEncapAdapter.

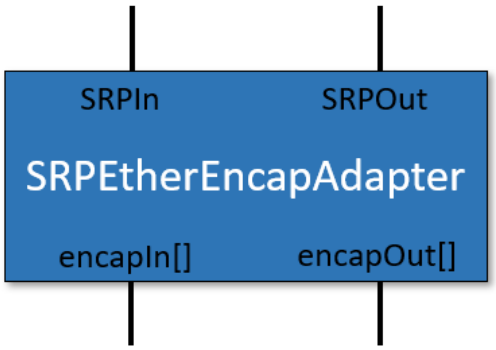

Abbildung 5.2: Schematische Darstellung des SRP-Ethernet-Encap-Adapters

Die Abbildung 5.2 stellt die Funktion des SRPEthernetEncapAdapter schematisch dar. SRPIn und SRPOut stellen die Ein - und Ausgänge des srpProtocol Modul da und encapIn[] und encapOut[] die möglichen multiplen Ein - und Ausgänge der verschiedenen Encap-Module. Durch den Adapter kann jetzt mehr als ein Nachrichtenpaket auf das SRP-Protocol-Modul zugreifen und anschließend in eine Ethernet-Nachricht verkapselt werden.

Node mit 2-Ports

In Abbildung 5.4 ist ein Node mit zwei Ports abgebildet. Der Node besitzt zwei physische Ports und es werden jetzt zwei avbABuffer bzw. avbBuffer generiert. Auch der entwickelte EncapEnthernetAdapater wird als adapter-Modul zwischen dem srpProtocol-Modul und den zwei encap-Modulen dargestellt. Zum Vergleich wird der Node mit einem Port 5.3 daneben abgebildet.

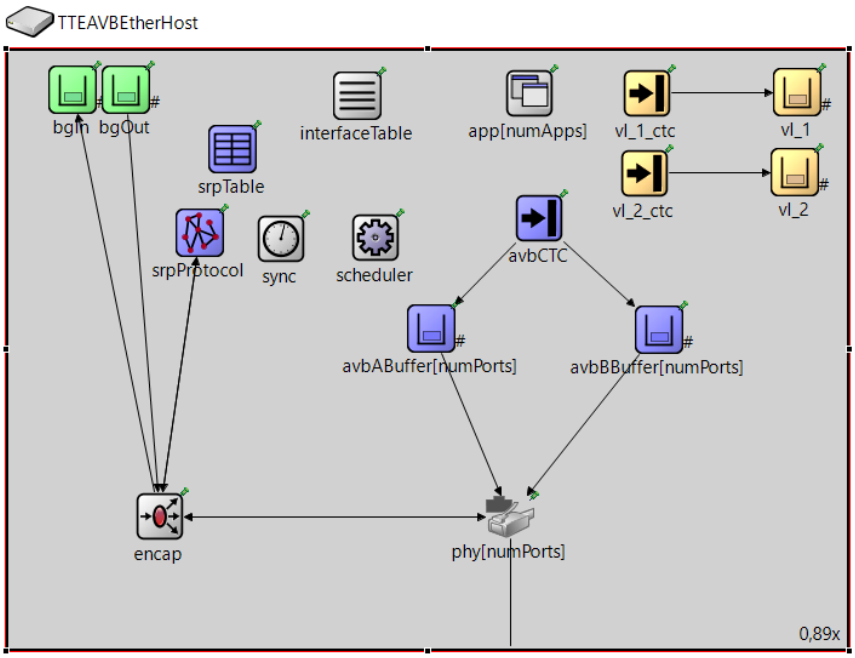

Abbildung 5.3: Aufbau AVB-Node mit einem Port

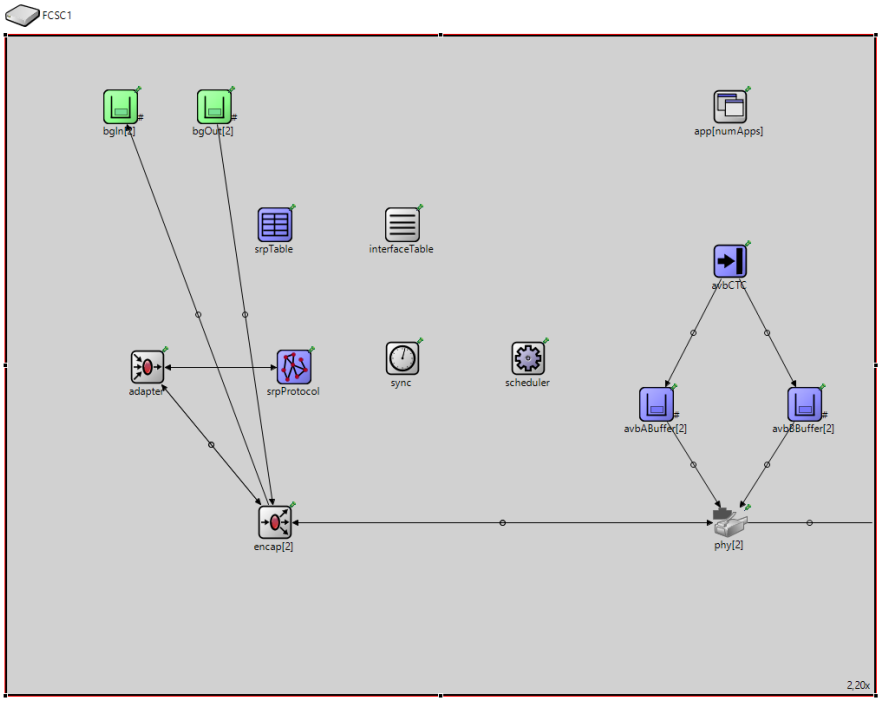

Abbildung 5.4: Aufbau AVB-Node mit zwei Ports und SRP-Adapter für AVB-Nachrichten

5.3 Nachrichten-Differenzierung

Ethernetframe mit Sequenznummer

Um die Nachrichten-Differenzierung zu implementieren wird das CoRE4INET Framework um die Funktion Sequenznummer im Ethernetframe erweitert. Wie schon in den Grundlagen erläutert ist die Simulationsumgebung OMNeT++ auf Modulen aufgebaut. Die Module sind in einer Kombination aus *.ned*, *.cc* und *.h* Dateien zusammengesetzt. Die *.ned*-Daten beschreiben die Architektur eines Moduls oder Netzwerk. Die *.cc* und *.h* Dateien implementieren das Verhalten.

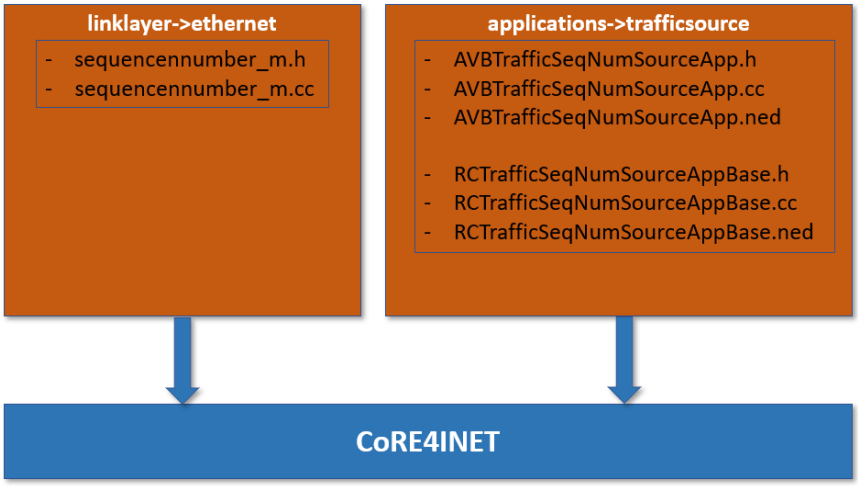

Abbildung 5.5: Schematische Erweiterung des CoRE4INET-Frameworks mit Sequenznummer-Funktion

In Abbildung 5.5 wird schematisch die Erweiterung des CoRE4INET-Framework mit der Sequenznummer-Implementierung dargestellt. Im linken Feld wird die Definition (*squence-number_m.cc* und *sequencenumber_m.h*) des Sequenznummer-Frame abgebildet. Auf rechten Seite sieht man die Implementierung der einzelnen *appilcations* mit Verhalten und Architektur der Nachrichtenklassen AVB und RC, die man in den Nodes als Modul vorfindet.

Funktionsweise

Sendet eine *app* z. B. ein RC-Frame, dann wird die Sequenznummer inkrementiert und mit samt des RC-Frames in ein Ethernet-Frame verkapselt und versendet. Da ein Node immer eine Kopie der Nachricht an beide Ports schickt, haben beide Nachrichten eine identische Sequenznummer.

5.4 Aufbau des Simulationsmodells

Im folgenden wird der Aufbau und die Konfiguration des Simulationsmodells beschrieben.

5.4.1 Aufbau des Netzwerks

Das Simulationsmodell wird in der NED-Sprache beschrieben und besteht aus unterschiedlichen Modulen. In den gezeigten Listings wird aufgrund Größe des Netzwerks nur Ausschnittsweise der Aufbau gezeigt. Listing 5.1 beschreibt die Definition des Netzwerks *plane_network_AVB*. Die verschiedenen Nodes (Endsysteme im Netzwerk) werden als Submodule definiert.

```
network plane_network_AVB
{
submodules:
        SFCC1: SFCC1 {}  // Endsystem Slats & Flats Control Comp. 1
        FCPC1: FCPC1 {}
        SFCC2: SFCC2 {}
        FCPC2: FCPC2 {}
        ...
}
```

Listing 5.1: Submodul-Definition

Unter *types* wird ein Kommunikationsweg (*Channel*) mit 100 MBit\s definiert, der die Eigenschaften eines Ethernet-Link besitzt. Im Abschnitt *connections* werden Nodes und Switches untereinander verdrahtet. In Listing 5.2 sieht man in Zeile 2 wie Port *0* von Switch S1 des blaues Netzwerks über eine 100 MBit Ethernetverbindung mit Port *0* von Switch S2 des blauen Netzwerks verbunden ist.

```
types:
channel ETH_100MBITS extends EtherLink{}

connections:
s1_blue.ethg[0] <--> ETH_100MBITS <--> s2_blue.ethg[0];
s2_blue.ethg[1] <--> ETH_100MBITS <--> s3_blue.ethg[0];
SFCC1.ethg[0] <--> ETH_100MBITS <--> s1_blue.ethg[2];
FCPC1.ethg[0] <--> ETH_100MBITS <--> s1_blue.ethg[3];
...
```

Listing 5.2: Channel- und Connections Definition

5.4.2 Konfiguration des Netzwerks

Nachdem dem Aufbau des Netzwerks folgt die Konfiguration des Netzwerks und der Nodes. In der Datei *omnetpp.ini* können globale spezifische Parameter wie z. B. Dauer der Simulation, Recording von Simulationsdaten und Eventlogs, Einstellungen unterschiedlichen Zyklen konfiguriert werden. Ebenfalls werden hier alle Konfigurationsdateien der Nodes (z. B. FADEC.ini) inkludiert.

5.4.3 Konfiguration der Nodes

Die Konfiguration der Nodes folgt in einer eigenen Datei. Ein Endsystem besteht immer aus zwei Dateien. Einer *.ned* Datei für die Architektur des Systems und einer *.ini* Datei für das Verhalten. Z. B. das *FADEC* Endsystem besteht aus einer *FADEC.ned* und einer *FADEC.ini* Datei. Die *.ini* Datei definiert z. B. welche und wie viele *apps* auf dem Node laufen. dh. welchen Nachrichten der Node sendet bzw. empfängt.

```
[General]
network = plane_network_RC

**.FADEC1.phy[*].mac.address = "7A-C6-0A-82-84-A1"

**.FADEC1.numApps = 10

# Sink app for message "fuelFlow1":
**.FADEC1.app[9].typename = "CTTrafficSinkApp"
...
# Source app for RC message "engineRotationalSpeed1":
**.FADEC1.app[0].typename = "RCTrafficSeqNumSourceApp"
**.FADEC1.app[0].interval = uniform(0.25s, 0.25s)
**.FADEC1.app[0].payload = 500Byte
**.FADEC1.app[0].ct_id = 2090
**.FADEC1.app[0].buffers = "vl_2090"
...
```

Listing 5.3: Aufbau der FADEC1.ini

In Listing 5.3 ist Aufbau einer *Node.ini* am Beispiel der *FADEC1.ini* dargestellt. Hier sieht man dass der Node *FADEC1* zum plane_network_RC gehört und welche Mac-Adresse jeweils sein Port besitzt. Die Anzahl der laufen dem *apps* auf dem Node beträgt 10. Die *Sink app* beinhaltet eingehende und die *Source app* ausgehende Nachrichten. In der *Source app* ist der Nachrichtentyp, Name der Nachricht, der Payload, die VL-ID und die Sendepriode definiert.

5.5 RC- und AVB-Simulationsmodelle in OMNeT++

Nach dem Aufbau und Konfiguration der Netzwerke für AVB- und RC-Nachrichten sind folgende zwei Simulationsmodelle in OMNeT++ entstanden. In Abbildung 5.7 ist das Simulationsmodell für RC-Nachrichten abgebildet. Es folgt im Aufbau dem Netz-Design des RC-Netz-Konzepts 4.2. Netzwerk A wird durch die roten Switches und Verbindungen dargestellt und das redundante Netzwerk B ist jeweils in Blau gekennzeichnet. An den insgesamt 9 Switches je Netz sind die verschiedenen Endsysteme (Nodes) angeschlossen und als graue Module dargestellt. Das Simulationsmodell für AVB-Nachrichten ist in Abbildung 5.6 abgebildet und beruht ebenfalls im Aufbau auf dem Netz-Design des AVB-Netz-Konzepts 4.3.

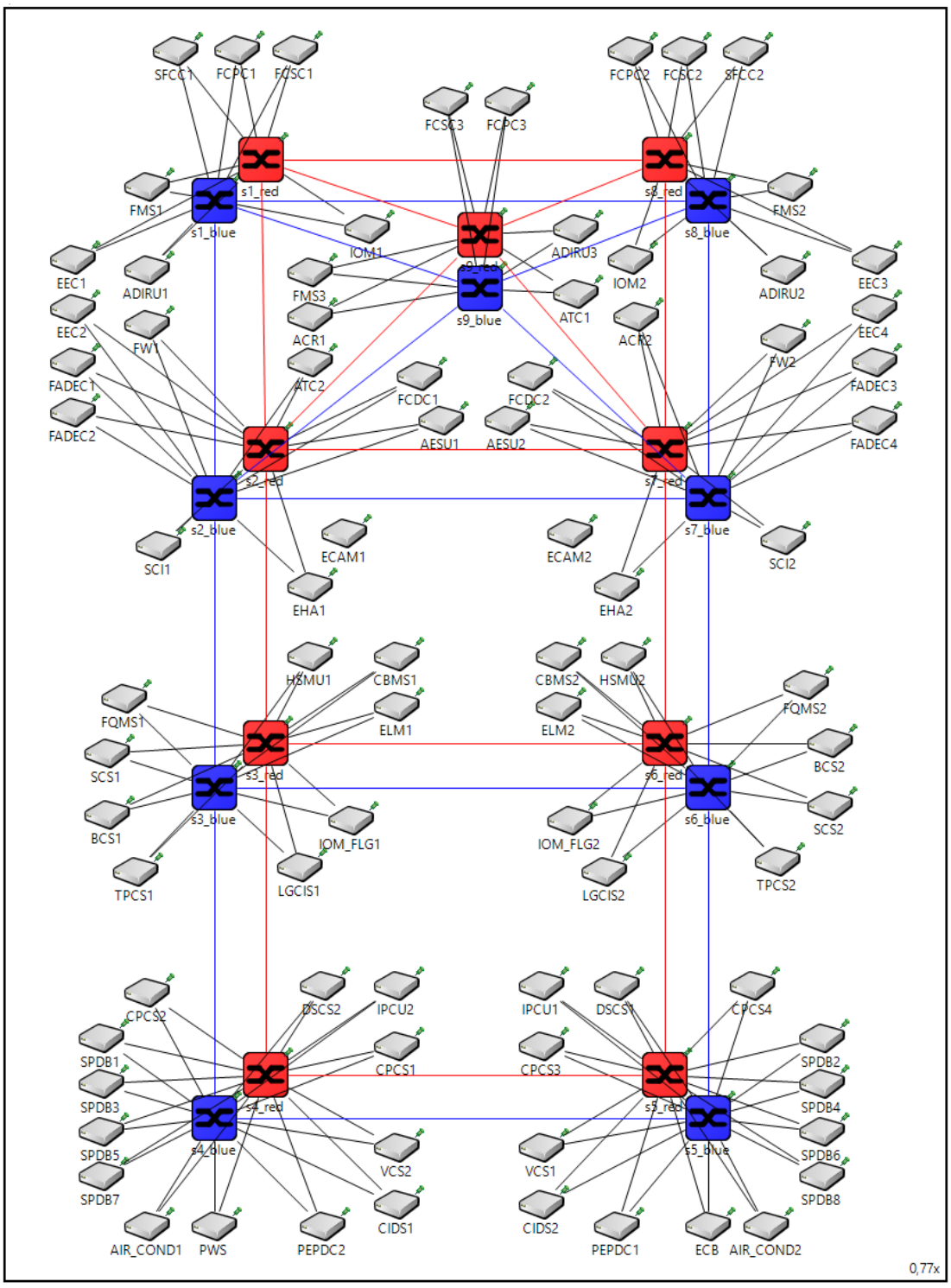

Abbildung 5.6: Entwurf des RC-Flugzeugnetzes in OMNeT++

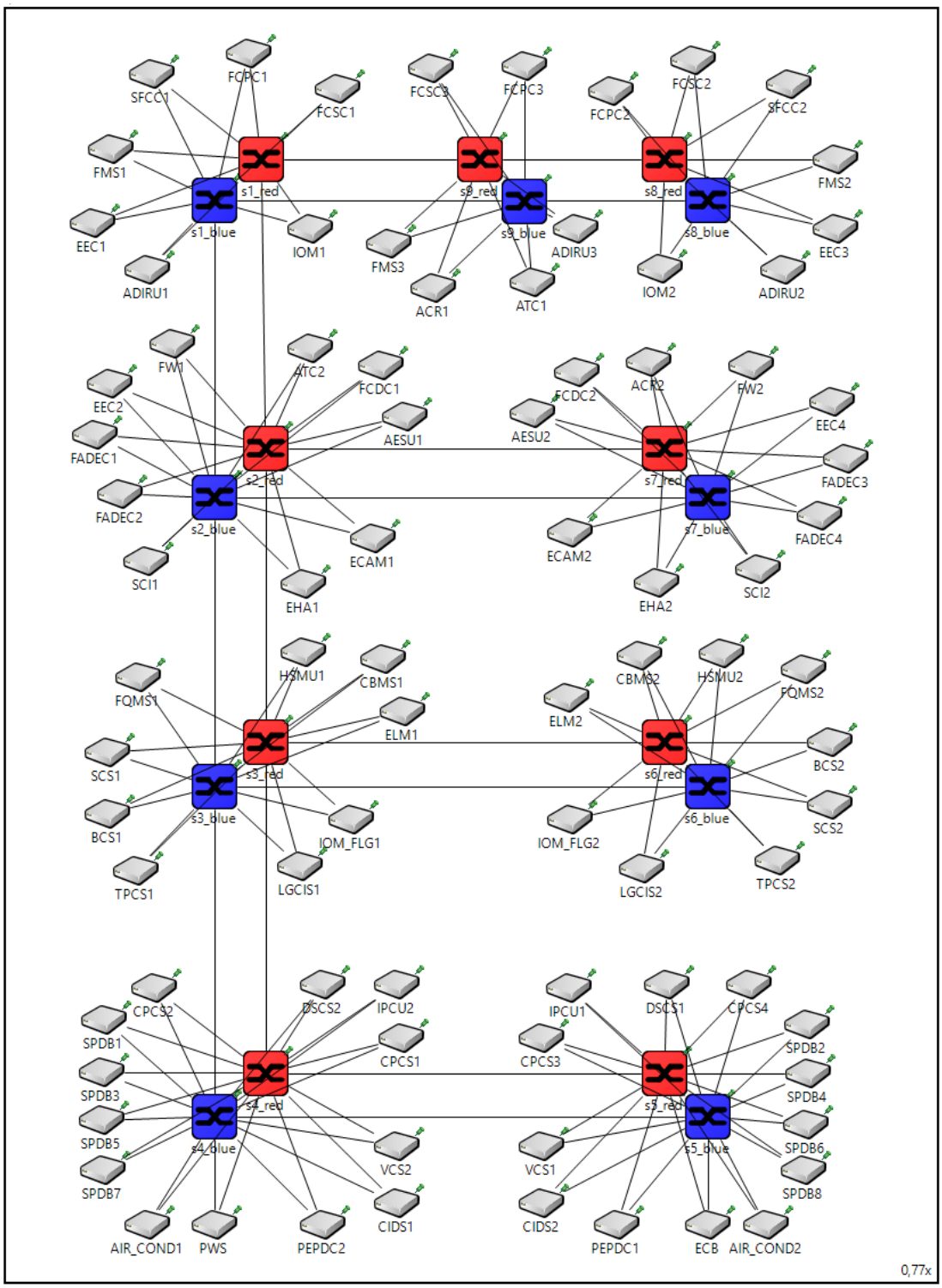

Abbildung 5.7: Entwurf des AVB-Flugzeugnetzes in OMNeT++

6 Tests und Evaluation

Dieses Kapitel befasst sich mit den Tests der neu implementierten Kompetenten und der Evaluation der Kommunikationsmatrix in den erstellten Simulationsmodellen des Flugzeugnetzes aus Kapital 5. Zuerst erfolgt die Validierung der entwickelten Komponenten für die Porterweiterung und der Nachrichten-Differenzierung per Sequenznummer. Im letzten Teil werden die Simulationsmodelle für AVB - und RC-Nachrichten des Flugzeugnetzes behandelt.

6.1 Tests

Komponenten-Validierung

Zum Testen wird ein kleines und übersichtliches Netzwerk, das aus vier Switchen und zwei Endsystemen besteht erstellt. Das Netzwerk wird einmal für RC-Nachrichten und einmal für AVB-Nachrichten konfiguriert.

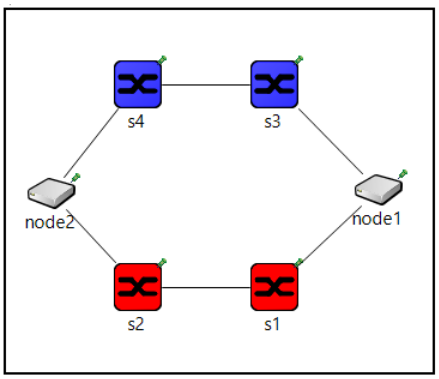

Abbildung 6.1: Test-Netzwerk für AVB- und RC-Nachrichten

Das Test-Netzwerk in Abbildung 6.1 besteht aus zwei Endsystemen Node1 und 2. Node1 ist einmal über eine blaue Verbindung über Switch s3 und s4 mit Node2 verbunden. Die alternative rote Verbindung verläuft über Switch s1 und s2 zu Node2. In der AVB-Konfiguration ist Node1 der Talker und Node2 der Listener. In der RC-Konfiguration sendet Node1 den vl_1 über die blaue Verbindung und vl_2 über die rote Verbindung.

Funktionstests

Das Testnetz muss für die jeweilige Konfiguration die folgende Anforderungen erfüllen:

- *RC-Test-Konfiguration*
  - Node1 sendet erfolgreich gleiches Ethernetframe über und vl_2 zu Node2
  - Node2 empfängt erfolgreich gleiches Ethernetframe über vl_1 und vl_2
  - Die Ethernetframes von vl_1 und vl_2 besitzen die selbe Sequenznummer
  - Beim wiederholten Senden des Ethernetframes über vl_1 und vl_2 erhöht sich die Sequenznummer in beiden Ethernetframes auf die selbe Nummer.
- *AVB-Test-Konfiguration*
  - Talker Node1 erstellt erfolgreich den gleichen Stream zu Listener Node2 über die rote und blaue Verbindung
  - Node2 empfängt erfolgreich das gleiche Ethernetframe über den rote und blauen Kommunikationsweg
  - Ethernetframes, die über die blaue und rote Verbindung gesendet werden, beinhalten die selbe Sequenznummer
  - Beim wiederholten Senden des Ethernetframes über die rote und blaue Verbindung erhöht sich die Sequenznummer in beiden Ethernetframes auf die selbe Nummer.

Das Testnetz erfüllt den Funktionstest mit AVB - und RC-Konfigurationen erfolgreich.

6.2 Evaluation

Nach erfolgreichen Test der Komponenten und deren Funktionsweise, erfolgt die simulationsbasierte Untersuchung der erstellten Kommunikationsmatrix. Für diese Analyse wird die Kommunikationsmatrix in den entwickelten Simulationsmodellen aus Kapitel (5.5) simuliert.

6.2.1 Simulation der Kommunikationsmatrix

Konfiguration

Für die Simulationsmodelle AVB und RC werden verschiedene Basiseinstellungen vorgenommen. Bei Wahl der Einstellungen wird versucht ähnliche Werte wie im Flugzeug zu wählen, damit die Kommunikationsmatrix in einer realitätsnahen Umgebung simuliert wird. Im Airbus A380 wird das AFDX-Netz mit 100Mbit/s betrieben. Dieser Wert ist nicht auf das AVB-Simulationsmodell übertragbar. Bei einer der Übertragung eines Radarbildes der Air Traffic Control (ATC) würde die Bandbreite nicht ausreichen. Bei der Annahme das Bild hätte VGA-Auflösung (640x480 Pixel) mit einer Farbkodierung 3 Byte (RGB-Farbmodell), kommt man zu folgenden Daten:

$$Datenvolumen = 640 * 480 * 3Byte = 921600Byte$$

Bei einem maximalen Payload (1500 Byte) eines Ethernet-Frames, ergibt sich die Anzahl der nötigen Frames pro Bild.

$$Frames = \frac{921600Bytes}{1500\frac{Bytes}{Frame}} = 614$$

Bei 30 Bilder pro Sekunden, ergibt es dann folgende Bandbreite.

$$Bandbreite = 30\frac{Bilder}{s} * 614\frac{Frames}{s} * 1518\frac{Bytes}{Frame} = 223,2\frac{MBit}{s}$$

Die Rechnung zeigt Links mit 100 Mbit/s reichen nicht aus. AVB wird daher mit 1 Gbit/s Links konfiguriert. Die Payload wird in beiden Simulationsmodellen 500Byte betragen. Simuliert werden 15 Echtzeit-Sekunden angestrebt, wobei es hier noch zur Korrektur kommen kann, da die Simulation bei Vortests zeigt, dass sie sehr rechenintensiv ist. Die Zykluszeit der Endsysteme wird aus der Kommunikationsmatrix übernommen. Beide Nachrichtentypen werden ohne nebenläufigem Best-Effort-Verkehr simuliert.

Simulation der RC- und AVB-Simulationsmodelle

Bei der Simulation der Kommunikationsmatrix werden Endsysteme und ihr redundantes Endsystem auf ihre Latenz untersucht. Die Endsysteme ECAM und FCPC werden ausgewählt, da sie Systeme in der Kommunikationsmatrix besonders viel Nachrichtenverkehr verarbeiten.

Anpassung der Simulationszeiten

Die Simulationszeit beider Modelle muss angepasst werden. Die angestrebten 15 Echtzeit-Sekunden sind nicht ideal. Die K-Matrix ist für das AVB-Simulationsmodell zu komplex. Erst nach ca. 24 Stunden Simulation erreicht das AVB-Modell ca. 10 Echtzeit-Sekunden. Ein Abschalten von bestimmten Simulationsroutinen brachte keinen merklichen Vorteil. Daher wird die Simulationszeit auf eine Echtzeit-Sekunden gesenkt. Das RC-Modell ist deutlich performanter. Hier wird die Simulationszeit auf 40 Echtzeit-Sekunden erhöht.

AVB-Netzwerk

Im AVB-Netzwerk wird das Endsysteme ECAM1 (Electronic Centralized Aircraft Monitoring1) und sein Zwillingssystem ECAM2 analysiert. In Abbildung 5.6 erkennt man, dass ECAM1 mit Switch s2 und ECAM2 mit Switch s7 verbunden sind. Danach wird der Flight Control Primary Computer 3 (FCPC3) mit seinem Ersatzsystem Flight Control Secondary Computer 3 (FCSC3) untersucht. Beide sind im Gegensatz zu den ECAM-Systemen, mit dem identischen Switch s9 gekoppelt.

Latenzzeiten im AVB-Netzwerk

In Tabelle 6.1 ist die Ende-zu-Ende Latenz mit Minimum, Durchschnitt und Maximum der Endsysteme ECAM1 und ECAM2 dargestellt. Die Zeiten für den FCPC3 und FCSC3 stehen in Tabelle 6.2.

Endsystem	Minimum	Durchschnitt	Maximum
ECAM1	26,88 μs	101,4324 μs	113,08 μs
ECAM2	47,04 μs	110,6216 μs	135,12 μs

Tabelle 6.1: Ende-zu-Ende Latenz ECAM mit AVB

Endsystem	Minimum	Durchschnitt	Maximum
FCPC3	41,78 μs	54,464 μs	89,2 μs
FCSC3	41,78 μs	57,243 μs	90 μs

Tabelle 6.2: Ende-zu-Ende Latenz FCPC3 und FCSC3 mit AVB

RC-Netzwerk

Im RC-Netzwerk wird das Endsysteme ECAM1 (Electronic Centralized Aircraft Monitoring1) und das Zwillingssystem ECAM2 analysiert. In Abbildung 5.7 erkennt man, dass ECAM1 mit Switch S2 und ECAM2 mit Switch s7 verbunden sind. Danach wird der Flight Control Primary Computer 3 (FCPC3) mit seinem Ersatzsystem Flight Control Secondary Computer 3 (FCSC3) untersucht. Beide sind im Gegensatz zu den ECAM-System, mit dem gleichen Switch s9 gekoppet. M

6.2.2 Latenzzeiten im RC-Netwerk

In Tabelle 6.3 ist die Ende-zu-Ende Latenz mit Minimum, Durchschnitt und Maximum der Endsysteme ECAM1 und ECAM2 dargestellt. Die Zeiten für den FCPC3 und FCSC3 stehen in Tabelle 6.4.

Endsystem	Minimum	Durchschnitt	Maximum
ECAM1	11,416 μs	12,883 μs	47,022 μs
ECAM2	17,124 μs	18,952 μs	84,609 μs

Tabelle 6.3: Ende-zu-Ende Latenz ECAM mit RC

Endsystem	Minimum	Durchschnitt	Maximum
FCPC3	95,84 μs	676,91 μs	1036 μs
FCSC3	95,84 μs	676,91 μs	1036 μs

Tabelle 6.4: Ende-zu-Ende Latenz FCPC3 und FCSC3 mit RC

6.2.3 Auswertung

An den Latenzzeit, von Modulen die mit dem identischen Switch verbunden sind, wie in Tabelle 6.2 und Tabelle 6.4 sind die Latenzzeit fast identisch. Bei den ECAM-Endsystemen, die an verschiedenen Switches gekoppelt sind, kann man schon einen deutlichen Unterschied feststellen. In Tabelle 6.3 ist die maximale Latenz von ECAM2 mit $84,609 \mu s$ fast doppelt

zu hoch wie an ECAM1. Die Auswertung der Latenz von AVB- zu RC-Netzwerk ist wenig aussagekräftig, da beide Netze eine unterschiedliche Netztopologie besitzen, ein Vergleich wäre nicht zielführend.

7 Zusammenfassung und Fazit

Ziel dieser Bachelorarbeit war die Überprüfung der Praxistauglichkeit des TTEhernet-Protokolls im Flugzeugbereich verbunden mit der Fragestellung, ob man mit den Mitteln die TTEthernet bereitstellt, ein funktionierendes Flugzeugnetz betreiben kann.
Hierzu wurde mit Hilfe einer selbst entwickelten Kommunikationsmatrix ein realitätsnahes Datennetzwerk eines Flugzeugs entworfen. Im zweiten Schritt wurden die Nachrichtenklassen RC und AVB in der Simulationsumgebung OMNeT++ simuliert und ausgewertet.
Als Vorbild diente das AFDX-Netzwerks eines Airbus A380.

7.1 Fazit

Zusammenfassend lässt sich anhand mehrerer Faktoren - u.a. Netzwerkdesign, Simulationsergebnisse - festhalten, dass die Nachrichtenklasse AVB sich nicht für das Betreiben eines Flugzeugnetzes eignet, weil in hohem Maße der Sicherheitsaspekt der Redundanz verloren geht. Für das AVB-Protokoll musste das vollvermaschte Netzwerk des A380, dass auf hohe Ausfallsicherheit beruht, aufgegeben werden. Der alternative Netzentwurf ist von der Topologie ein Bussystem, welches wesentlich anfälliger für Kommunikationsstörungen ist. So kommt es zum Beispiel beim Ausfall der Hauptverbindung zum Totalausfall des Netzes. Ein weiterer Nachteil ist, dass es sich im Gegensatz zu den RC- und AFDX-Protokollen nicht um eine statische Konfiguration des Routings handelt, sondern um eine dynamischen Konfiguration. Folglich ist die Nachrichtenklasse AVB nicht ressourcenschonend. Beispielsweise konnte man es sehr gut bei der Konfiguration des AVB-Netzwerks beim Einstellen der Bandbreite erkennen. Hierbei war es zwingend erforderlich die Bandbreite auf ein GBit/s zu erhöhen, um eine Simulation überhaupt zu ermöglichen.
Im Gegensatz zur Nachrichtenklasse AVB eignet sich die RC-Nachrichtenklasse sehr gut im Flugzeugbereich. Per Offline-Scheduling wird hier das Routing und die nutzbare Bandbreite definiert. Das garantiert, dass die Bandbreite für jedes Paket vordefiniert ist und Verzögerungen und zeitliche Abweichungen definierte Obergrenzen haben. Damit verhält sich die RC-Nachrichtenkommunikation bis auf das Redundanzmanagement gleich. Somit kann das TTEhernet-Protokoll auch im Flugzeugbereich Anwendung finden.

Literaturverzeichnis

[Actel 2005] Actel: *Developing AFDX Solutions*. 2005. – URL http://www.actel.com/documents/AFDX_Solutions_AN.pdf. – Zugriffdatum: 2018-01-19

[Aeronautical Radio Incorporated 2009] Aeronautical Radio Incorporated: *Aircraft Data Network, Part 7, Avionics Full-Duplex Switched Ethernet Network / ARINC. 2009 (ARINC Report 664P7-19*. 2009

[AIM GmbH] AIM GmbH : *AFDX Training*. – URL http://www.afdx.com. – Zugriffdatum: 2018-03-01

[ARINC 2005] ARINC: *AIRCRAFT DATA NETWORK PART 7 AVIONICS FULL DUPLEX SWITCHED ETHERNET (AFDX) NETWORK*. 2005. – URL https://standards.globalspec.com/std/1283307/arinc-664-p7. – Zugriffdatum: 2017-08-01

[CoRE Research Group] CoRE Research Group: *CoRE Research Group Website*. – URL http://core.informatik.haw-hamburg.de/en/. – Zugriffdatum: 2017-10-10

[CoRE4INET] CoRE4INET: *CoRE Research Group TTE for INET*. – URL http://tte4inet.realmv6.org/. – Zugriffdatum: 2017-10-10

[David und Rockwell 2004] David, Miller ; Rockwell, Collins: *AFDX Determinism 2004*. 2004. – URL http://www.arinc.com/aeec/general_session/gs_reports/2004/presentations/afdx_determinism_david_miller.pdf. – Zugriffdatum: 2018-03-01

[Institute of Electrical and Electronics Engineers 2011] Institute of Electrical and Electronics Engineers: *IEEE 802.1BA - IEEE Standard for Local and metropolitan area networks - Audio Video Bridging (AVB) Systems / IEEE. September 2011 (IEEE 802.1BA-2011)*. 2011. – Standard

[Kempf 2008] Kempf, Fabian: *RECBAR-Bericht: Metriken in Ethernet-basierten Fahrzeugnetzen*. CoRE Research Group, 2008

[Lim u. a. 2012] Lim, Hyung-Taek ; Herrscher, Daniel ; Waltl, Martin J. ; Chaari, Firas: *Performance analysis of the IEEE 802.1 ethernet audio/video bridging standard. In Proceedings of the 5th International ICST Conference on Simulation Tools and Techniques (SIMUTOOLS '12). ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering).* ICST, Brussels, Belgium, 2012

[OMNeT++ Community] OMNeT++ Community: *INET Framework for OMNeT++ 4.0.*. – URL http://inet.omnetpp.org/. – Zugriffdatum: 2017-08-04

[OMNeT++ Manual] OMNeT++ Manual: *OMNeT++ Manual.* – URL https://omnetpp.org/doc/omnetpp/manual/usman.html. – Zugriffdatum: 2017-03-08

[Schäuffele und Zurawka 2013] Schäuffele ; Zurawka: *Automotive Software Engineering.* Vieweg und Teubner, 2013. – ISBN 978-3-8348-2469-1

[Society of Automotive Engineers - AS-2D Time Triggered Systems and Architecture Committee 2016] Society of Automotive Engineers - AS-2D Time Triggered Systems and Architecture Committee: *Society of Automotive Engineers - AS-2D Time Triggered Systems and Architecture A6802. SAE Aerospace.* 2016. – URL http://standards.sae.org/as6802/. – Zugriffdatum: 2017-11-10

[Steiner 2009] Steiner, Wilfried: *TTEthernet Datafow Concept.* In 8th IEEE International Symposium on Network Computing and Applications), 2009

[Tindell und Burns 1994] Tindell ; Burns: *Guaranteed Message Latencies for Distributed Safety-Critical Hard Real-Time Control Networks.* 1994. – URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.50.5526. – Zugriffdatum: 2018-03-08

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

Hamburg, 17. Juli 2018 Michael Malsch