

Network Architectures
and Services
NET 2013-12-1

Dissertation

Performance Bounds in Switched Ethernet Onboard Networks

Emanuel Heidinger



Network Architectures and Services
Department of Computer Science
Technische Universität München



TECHNISCHE UNIVERSITÄT MÜNCHEN
Institut für Informatik
Lehrstuhl für Netzarchitekturen und Netzdienste

Performance Bounds in Switched Ethernet Onboard Networks

Dipl.-Inf. Univ. Emanuel M. Heidinger

Vollständiger Abdruck der von der Fakultät für Informatik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktors der Naturwissenschaften (Dr. rer. nat.)

genehmigten Dissertation.

Vorsitzender: Univ.-Prof. Dr. H. M. Gerndt
Prüfer der Dissertation: 1. Univ.-Prof. Dr. G. Carle
2. Univ.-Prof. Dr. J. B. Schmitt
Technische Universität Kaiserslautern

Die Dissertation wurde am 20.06.2013 bei der Technischen Universität München eingereicht und durch die Fakultät für Informatik am 06.11.2013 angenommen.

Cataloging-in-Publication Data

Emanuel Heidinger

Performance Bounds in Switched Ethernet Onboard Networks

Dissertation, December 2013

Network Architectures and Services, Department of Computer Science
Technische Universität München

ISBN 3-937201-39-4

ISSN 1868-2634 (print)

ISSN 1868-2642 (electronic)

DOI 10.2313/NET-2013-12-1

Network Architectures and Services NET-2013-12-1

Series Editor: Georg Carle, Technische Universität München, Germany

© 2013, Technische Universität München, Germany

ABSTRACT

In modern automotive and aeronautic onboard networks, a driving force exists to integrate multiple communication networks onto a single switched Ethernet network. Goals for future onboard communication networks are low weight, low price, high bandwidth, and certifiable safety. Certainly, these objectives influence each other and are difficult to fulfill. Furthermore, an existing certification may not encourage an expensive recertification of novel communication technologies. Aircraft cabin networks are subject to certification, as they not only cover comfort functions such as In-Flight Entertainment, but also safety related functions such as the announcement of evacuation messages or the polling of smoke detectors.

This work addresses the applicability of a switched queuing network for both safety relevant and comfort functions in aircraft cabin networks. Hard performance guarantees on latency and jitter must be fulfilled. The state-of-the-art approach to determine worst case bounds in queuing networks is Network Calculus. The tightness of Network Calculus is subject to ongoing research. Recent work even showed the NP-hardness of determining tight bounds in general networks. In this work, we propose a packetized model based on a mixed-integer program, which is able to express the worst-case in a closed-form representation. The proposed algorithm allows better mapping of packetized switched networks compared to known state-of-the-art Network Calculus approaches.

The presented approach is embedded in the performance evaluation platform DIM-TOOL. This platform provides several performance evaluation techniques, such as network simulation and Network Calculus, in one single toolbox. Having a consistent view on performance bounds allows for a rapid evaluation of the worst case bounds of the cabin network, which in turn is required for the certification process. We show that the hard performance bounds are guaranteed, which opens the investigated switched queuing network for a wide range of applications.

ZUSAMMENFASSUNG

Die Einführung von Switched-Ethernet im Kabinennetz des Flugzeugs verspricht sowohl Gewichts- als auch Kostenersparnis. Da heutige Ethernettechnologien höhere Bandbreiten erreichen, ist es dabei sogar möglich neue Applikationen und Komfortfunktionen bereitzustellen. Die Einführung neuer Technologien in einem sicherheitsrelevanten Kommunikationssystem bedarf allerdings einer Überprüfung im Zertifizierungsprozess. Darüber hinaus ist ein Austausch bestehender Kommunikationssysteme im Hinblick auf die zu erwartenden Kosten zu prüfen. Tatsächlich ist das Kabinenkommunikationssystem bereits sicherheitsrelevant, da beispielsweise der Status von Rauchmeldern erfasst wird und Sicherheitsdurchsagen übertragen werden.

Der Zertifizierungsprozess fordert hierbei harte Garantien hinsichtlich Latenz und Jitter. Der gängige Ansatz zur Bestimmung von Worst-Case Garantien ist Network-Calculus. Die Güte der ermittelten Worst-Case Grenzen ist Gegenstand derzeitiger Forschung und stößt durch die verwendeten mathematischen Operatoren an Ihre Grenzen. Diese Arbeit stellt ein neuartiges, paketbasiertes Modell vor, das den Worst-Case mittels Mixed-Integer-Programming repräsentiert. Diese Repräsentation erlaubt eine verbesserte Darstellung des Worst-Cases, die mit den Network-Calculus Operatoren nicht möglich ist.

Der neue Ansatz wird in das Dimensionierungswerkzeug DIMTOOL eingebettet. Neben diesem neuen Ansatz verfügt das Werkzeug über gängige Techniken der Performanzanalyse wie Network-Calculus und Netzsimulation. Eine konsistente Sicht auf Performance-Garantien ermöglicht eine schnelle und aussagekräftige Bestimmung des Worst-Case, welche im Rahmen des Zertifizierungsprozesses nötig ist. Das entwickelte Werkzeug ist geeignet, um die Garantien in Onboard-Netzen der nächsten Generation zu bestimmen.

ACKNOWLEDGMENTS

This thesis would never have been possible without the continuous help and support of a number of people.

First of all I would like to thank Prof. Dr.-Ing. Georg Carle for the excellent guidance and supervision. I would like to thank Prof. Dr.-Ing. Jens Schmitt for being my second assessor and for his valuable feedback. Additionally, I would like to thank Prof. Dr. Michael Gerndt for chairing the examination committee.

In particular, I would like to thank Prof. Dr. Alexander von Bodisco, and Dr. Nils Kammenhuber, whose thoughtful insights were invaluable for this project, and the wonderful, inspiring people I met on conferences.

It has always been a pleasure to visit the Chair for Network Architectures and Services, TU München. Among the researches of the chair I would like to thank Stephan Günther, Ralph Holz, Dr. Tobias Bandh, Lothar Braun, Nadine Herold, Dr. Holger Kinkelin, Dr. Andreas Müller, Marc-Oliver Pahl, Stephan-A. Posselt, as well as Matthias Wachs.

I would like to thank EADS for giving me the opportunity to do research in the corporation. My thanks go to Josef Schalk and Stefan Schneelee. Among my former colleagues I have to thank Judith Andres, Helga Bayer, Johannes Blanckenstein, Christian Blümm, Stefan Burger, Dr. Thilo Fath, Dr.-Ing. Oliver Hanka, Oliver Keller, Dr.-Ing. Johannes Schels, and Dr. Falk Schubert.

Finally I would like to thank my family, and my wonderful, loving and encouraging girlfriend Simone, for supporting me throughout those years.

NOTATION

- When a notion is introduced, it is written in *italic letters*. The abbreviation (if exists) is given in parentheses and attached to the comprehensive list of abbreviations.
- Established abbreviations or protocol names such as IP or TCP are explained in footnotes.
- Throughout this text we introduce notions and related abbreviation at the first occurrence. After the first occurrence, we use the abbreviated term. An exception to that rule is given when the full notion clarifies the surrounding statement.
- In the field of *Network Calculus* we use the notation given in Equation 0.1:

$$[x]^+ \equiv \max\{x, 0\} \tag{0.1}$$

- We also use the notation given in Equation 0.2:

$$[x]_1 \equiv \min\{x, 1\} \tag{0.2}$$

CONTENTS

Abstract	iii
Zusammenfassung	v
Acknowledgments	vii
Notation	ix
Contents	xiv
1. Introduction	1
1.1 Motivation	1
1.2 Contribution to Research	3
1.3 Thesis Overview	4
2. Background	7
2.1 Aircraft Cabin Management System	7
2.1.1 Safety and Requirements	7
2.1.2 Current Architecture of Cabin Management System	11
2.1.3 Summary	13
2.2 Performance Bounds in Queuing Networks	14
2.2.1 Determining Performance Bounds	14

2.2.2	Quality of Service	16
2.2.3	Network Calculus	30
2.2.4	Summary	37
3.	Next Generation Cabin Management System	39
3.1	General Cabin Layout and Requirements	39
3.2	Single Domain Cabin Layout	40
3.3	Multiple-Domain Cabin Layout	40
3.4	Traffic Characteristics	42
3.5	Configuration Profile	42
3.5.1	Single Domain Layout	43
3.5.2	Multiple-Domain Layout	43
3.6	VLAN Concept	44
3.6.1	Dedicated VLANs	45
3.6.2	Class Based VLANs	45
3.7	Summary	47
4.	An Algorithm for Determining Worst Cases in Packet-Switched Networks	49
4.1	Determining the Exact Worst Case	49
4.2	Related Work	51
4.3	Mixed Integer Programming	53
4.4	Optimization Based Approach	55
4.5	Evaluation of Different Worst Case Calculation Techniques	60
4.6	Summary	67

5. DIMTOOL —Dimensioning the Next Generation Aircraft Cabin	69
5.1 Related Work	70
5.1.1 Network Performance Calculation	71
5.1.2 Configuration and Management	72
5.2 Topology and Flow Description	74
5.3 Performance Calculation Backends	76
5.4 DIMTOOL Architecture	78
5.5 Topology Toolkit	82
5.6 Deployment	84
5.7 Summary	86
6. Evaluation	87
6.1 DIMTOOL Performance Bounds	87
6.1.1 Cabin Intra-Server Communication	87
6.1.2 Single Domain Switched Ethernet Cabin	90
6.1.3 Multiple Domain Switched Ethernet Cabin	92
6.1.4 Summary	100
6.2 Realistic Test System	100
6.2.1 Evaluation Setup	100
6.2.2 Case Study of Layer 2/3 COTS Switches	102
6.2.3 Cabin Demonstrator	111
6.2.4 Applicability of AVB in the Avionic Context	115
6.2.5 Summary	123
7. Conclusion and Outlook	125

Appendices	129
A. DIMTOOL Toolchain	131
B. Network Calculus Bounds	135
C. Topology and Flow Description	141
D. Cabin Demonstrator	143
E. Publications by the Author	145
List of Abbreviations	149
List of Figures	155
List of Tables	159
Bibliography	169

1. INTRODUCTION

1.1 Motivation

Today, different techniques are used for the communication networks within an aircraft cabin, such as LVDS, RS232, 10Base2, ARINC 429 [9], and *Avionics Full Duplex Ethernet* (AFDX) [11]. One of these communication networks is the *cabin intercommunication data system* and is found in modern Airbus airplanes, such as the A380. The cabin intercommunication network is used for the following safety relevant functions: (a) passenger address (i.e., audio announcements from the crew to the passengers), (b) intercabin communication (i.e., telephony between crew members), (c) smoke detection in the lavatories, and (d) passenger call (i.e., the button for calling a stewardess). As a result, the cabin intercommunication data system is still safety relevant, when we consider evacuation messages and smoke detection in the cabin.

Advantages of the current solution are the *existing certification* and its *excellent robustness*. A *drawback* of the current system is its *limited bandwidth*, as it is based on a 10Base2 Ethernet bus. Its limited bandwidth prevents it from being used for resource intensive services such as high quality audio and video transmissions. This means that other cabin services have to use different cables, which run largely in parallel. Since reducing weight is a primary objective in aircraft construction, this is an undesirable situation. In this context, the harmonization of aircraft networks, i.e., the integration of various communication demands and services on board into a single network, is still an ongoing process. Apart from saving weight, further advantages, such as a less complexity and better maintainability, are to be considered.

AFDX is a switched Ethernet solution that provides services to high safety level functions in the avionics domain. A possible solution for the discussed system could be to apply AFDX in the cabin as well. However, for a cabin network, which has less requirements with respect to failure probability and robustness than avionics, AFDX components are too expensive to be a suitable solution. In addition, the performance

bounds that AFDX can guarantee are too high to meet the cabin requirements. In this work, we investigate whether *commercial-of-the-shelf* (COTS) network components like Ethernet switches can be employed in the aircraft cabin. The use of standard Ethernet components takes advantage of recent progress in communication technology, such as BroadR-Reach™ [20], which allows transmitting 100 Mbit/s full-duplex over two wires—a tremendous advantage in terms of saving weight.

A valid question is whether data traffic from *In-Flight-Entertainment* (IFE), e.g., video, Web browsing, or playing games, can use the same network as the cabin traffic. In any case, the cabin network must provide hard performance guarantees for the safety critical applications—for example, not a single packet from a smoke detector is allowed to be lost due to buffer overflow. In addition, the delay of data packets of safety-relevant applications must not exceed given requirements in order to pass the certification process. We therefore have to prove that a network consisting of COTS components is capable of meeting the given safety requirements by using common techniques such as rate shaping and *Virtual LAN* (VLAN) priorities.

Using COTS network components for critical applications requires a few *restrictions* with respect to *traffic profiles*, *bandwidth guarantees*, and *prioritization*. By segregating traffic flows into higher and lower priority traffic profiles, high priority traffic can be coerced into pre-defined token bucket traffic policies, which are in turn enforced by some sort of traffic shapers implemented in the COTS hardware. These traffic descriptions subsequently can be used by analytical methods to approximate and even determine upper bounds within the aircraft cabin domain.

During the certification process of such a novel architecture, hard performance guarantees must be provided. The most important avionics standard that imposes such performances guarantees is DO-214 [98]. Compared to today's TDMA¹ based cabin bus system, it is much more difficult to obtain reliable performance bounds in a switched, packetized queuing network. One established technique to obtain such bounds is *Network Calculus*. However, recent work has shown (such as [16, 103]) that tight bounds cannot be determined by the standard operations of *Network Calculus*. This thesis shows the effect of packetization as another reason for loose bounds in the *Network Calculus*, since *Network Calculus* techniques usually employ an additional delay of a maximum sized frame per traversed node to express the store-and-forward delay in non-preemptive, packetized networks.

¹ Time Division Multiple Access

1.2 Contribution to Research

The contributions of this research are threefold and summarized in this section. We provide (a) a framework for next generation aircraft cabin network based on COTS devices, (b) a novel algorithm based on *Mixed Integer Programming* to determine the exact worst case, and (c) the toolchain DIMTOOL that provides several *performance estimators* under one umbrella. This toolchain is rigorously used for the performance evaluation of different cabin layouts and configurations.

- C1) Recent progress in the field of hardware switching ASICs² and two-wire Ethernet full-duplex solutions has pioneered an employment in aeronautic networks with certain safety demands. We develop a novel next generation aircraft cabin network based on COTS devices. Up to now, it has not been possible to mix several safety domains without raising the lower safety levels to the same level as the highest safety level. We present strategies for mixing the data stemming from applications with different safety levels in the same physical wire by employing reliable *Quality of Service* (QoS) priority mechanisms provided by hardware. We argue that this approach saves weight due to harness savings and to lower harness complexity. The novel aircraft cabin based on switched Ethernet is rigorously evaluated by analytical and simulative methods. We show that a Gigabit backbone is able to give sufficient QoS on latency and jitter requirements for the given traffic loads.
- C2) A novel *algorithm for determining the worst case bounds* in switched Ethernet is presented. The given solution is based on *Mixed Integer Programming* (MIP) and uses an *exhaustive enumeration of packet schedules* following the spirit of model checking. Compared to current solutions, we provide a *closed form solution* similar to the edge-by-edge analysis known from *Network Calculus* (NC) and introduced in Section 2.2. This solution allows a packetized representation which is similar to the packetizer concept introduced in NC. Consequently, we preserve both the superiority of an edge-by-edge analysis and the advantage of packetization.
- C3) The dimensioning tool DIMTOOL is the framework that allows performance calculations and simulations of the aircraft cabin. Existing tools often rely on a *fixed calculation procedure*. The DIMTOOL provides several performance estimators under one umbrella, thus making the performance results of different methods transparent and comparable.

² Application-Specific Integrated Circuit

- C3.1)** At this stage, we provide (a) *Network Calculus*, (b) *Monte Carlo Simulation*, (c) *Worse Case Simulation*, as well as the optimization based approach based on (d) *Mixed Integer Programming*. We employ DIMTOOL to verify the latency constraints in the switched aircraft cabin.
- C3.2)** We show that the NC-FIFO bound does not hold, due to unbounded, arbitrary multiplexing in switches. This unbounded, arbitrary multiplexing must be assumed if COTS switches are employed in the switched aircraft cabin.

1.3 Thesis Overview

The goal of this thesis is to address the employment of switched Ethernet in the aeronautic aircraft cabin. For this, we *analyze the system* in terms of safety and performance requirements while starting from the original cabin core network called *Cabin Intercommunication Data System* (CIDS). Chapter 2 addresses the *background and state-of-the-art*. This chapter discusses the following fields: avionics networks as well as performance bounds in queuing networks. The avionics demands are mapped to the specific use cases of the aircraft cabin. The performance bounds are guaranteed by sophisticated QoS mechanisms; basic state-of-the-art mechanisms are introduced in this chapter. Furthermore we summarize the models and techniques for identifying reliable performance bounds for networks, with a special focus on switched queuing networks. We take a close look at the analytical method NC.

Chapter 3 introduces a *novel network architecture* that covers the requirements and functions as observed in the aircraft cabin. This novel network architecture is not only able to cover the specific needs of the original CIDS, but is also open for the integration of lower safety domains into the same network. This network is therefore able to cover several networks with different safety requirements, such as CIDS, the panel network, and the IFE environment.

Chapter 4 proposes a *novel algorithm* for the identification of worst case bounds in switched networks. Compared to networks where relayed packets conform to a fixed packet size, such as ATM³, the identification of tight performance bounds in switched networks is not as simple, due to the existence of variable packet sizes. By the use

³ Asynchronous Transfer Mode

of MIP, we provide an exhaustive enumeration of schedules, which is similar to techniques known from model checking and which allows the calculation of worst case bounds. To the best of our knowledge, this algorithm is the first of its kind that provides a *closed system description* for switched Ethernet and that allows an identification of performance bounds that considers the existence of variable sized packets.

Chapter 5 introduces a *novel toolchain* that provides several performance estimation techniques, such as Network Simulation, NC, and the MIP approach, under one umbrella. This tool is able to determine comparable performance bounds and help the system engineer through the certification process of the next generation aircraft cabin.

Chapter 6 summarizes the performance results as observed for the different aircraft cabin scenarios introduced in Chapter 3. We address the issue of performance bounds by *simulative* and *analytical* approaches. Measurements that were accomplished in the *realistic test setup* confirm the simulative and analytical considerations.

Chapter 7 draws some conclusions and gives an overview of possible future research.

2. BACKGROUND

This chapter introduces the relevant background. First we introduce the discussed aircraft cabin management system that acts as the target being improved. Later we will discuss performance bounds in queuing networks, how they are defined, how they can be achieved by QoS mechanisms, and how the guarantees are proved by analytical algorithms.

2.1 Aircraft Cabin Management System

The communication system in an aeronautical cabin not only covers comfort functions such as IFE but also safety relevant functions, such as inter-crew communications and evacuation messages. Proving the reliability and deterministic performance bounds is mandatory for the certification of novel airplane types.

There is a need to share components between the different Airbus series and programs. When novel Airbus programs are launched, such as the A30x program or the A350 program, they shall share the same technology in order to reduce efforts in development and to allow cost savings. Already deployed Airbus airplanes can also profit from progress in subsystems by upgrades.

2.1.1 Safety and Requirements

Most cabin core functions are safety-relevant, i.e., a failure of those systems may influence the safety of the crew and passengers. Consider a failure of the audio announcement system. Such a failure may lead to the absence of evacuation messages and safety briefings, so that the safety of the crew and passengers is compromised.

The classification of aircraft functions into safety levels is mandatory for the certification process. *Design Assurance Levels* (DALs) are used to express the safety levels of

DAL	Classification	DAL Definition
Level A	Catastrophic	Catastrophic failure condition for the aircraft
Level B	Hazardous / Severe–Major	Hazardous / severe–major failure condition for the aircraft
Level C	Major	Major failure condition for the aircraft
Level D	Minor	Minor failure condition for the aircraft
Level E	No Effect	No effect on aircraft operational capability or flight crew workload.

Tab. 2.1: Design Assurance Level (Adapted From [99])

systems and functions. These levels range from level A to level E, where A is the highest safety level and E is the lowest safety level. Table 2.1 summarizes the different DALs according to [99] and [107].

The following list summarizes the safety relevant functions in the aircraft cabin.

- **Passenger Address** — A *Passenger Address* (PA) is a cabin function that covers audio announcements from the crew to the passengers. A failure of this system may lead to reduced safety, e.g., due to the absence of evacuation messages.
- **Passenger Call** — A *Passenger Call* provides for calling the stewardess by pressing the button of the *Passenger Service Unit* (PSU). A failure of this system may certainly prevent passengers in need from calling for help from the crew.
- **Cabin Communications** — The telephony system on board is called the *Cabin Interphone*, which is used for coordinative tasks.
- **Pre-recorded audio announcements** — Pre-recorded audio announcements such as boarding music are not necessarily safety relevant per se. It depends on the actual purpose, since pre-recorded audio announcements also cover standardized safety briefings and are thus relevant to safety.
- **Smoke Detection** — Retrieving sensor information from the smoke detectors in the lavatories is certainly safety relevant in order to prevent the outbreak of a fire on board.
- **Lighted Signs** — The most common use cases for lighted signs are the non-smoking sign and the fasten seatbelt sign. Precisely the fasten seatbelt sign is relevant for safety. Passengers are to take a seat during starting, landing, and when turbulences occur.

- Cabin Illumination — The cabin illumination is also part of the CIDS system but usually has lower requirements in terms of safety than the recently mentioned CIDS functions.
- Cabin Video Monitoring System — Video monitoring is a novel use case covered by the cabin system and currently not covered by the CIDS system.

In the following, we summarize the non-safety relevant comfort functions that aim at a convenient stay of the passengers on board. These optional functions are currently covered by additional networks, which results in additional weight. Combining those networks in a single network would be a definite advantage in terms of saving weight.

- IFE — The In-flight entertainment system provides on-board entertainment to the passengers. Common examples for IFE onboard are radio access, watching movies or playing games with other passengers.
- Internet and Email — Some airlines already provide Internet access and Email services to the passengers during flight. The actual Internet connection is usually realized by satellite, the content is distributed by WLAN while the access points are connected by a separate network.
- Mobile Phones — A next challenging use case is the onboard connectivity of mobile phones. There is a definite need for providing those services in the future. Merging this backbone into the cabin system is an important topic, and one currently discussed by manufacturers.

The following end devices participate in the recently mentioned cabin functions. The quantity and actual position in the aircraft is highly dependent on the aircraft type and configuration, and is figured out in Section 3.5.

- *Passenger Service Unit (PSU)* — The passenger service unit is situated overhead. The PSU plays audio announcements, provides a service button for the passenger to call a crew member, reading lights, as well as venting slots.
- *Illumination Ballast Unit (IBU)* — The illumination ballast unit is part of the illumination system in the cabin. Sophisticated versions do provide light scenarios such as a simulation of sunrise.
- Cabin handset — The cabin handset is part of cabin communications. Besides one-to-one calls, the cabin handset may take part in conference calls or provide audio announcements from the crew to the passengers.

- *Flight Attendant Panel (FAP)* — The Flight Attendant Panel provides a frontend for the crew to control the cabin functions, such as control over the illumination system, reading lights, and boarding music.
- *Camera* — Cameras are part of the *Cabin Video Monitoring System (CVMS)* and not yet part of the CIDS. This newer use case is currently covered by a different network.
- *Smoke sensor* — Smoke sensors are polled several times per second in order to detect smoke or fire. They are usually situated in the lavatories or nearby the galley.

Safety-relevant functions can be divided into *safety-relevant signaling* as well as *safety-relevant audio playback*. Signaling messages are expected from sensors such as smoke detectors and from the passenger service units providing the passenger call function. In addition, the quality and availability of audio plays an important role in the CIDS, be it due to the playback of safety briefings and evacuation messages, or due to the cabin phone conversations of crew members. The audio distribution in the aircraft cabin will likely be digital due to the weight advantage over an analog distribution. Due to its noise immunity and the simplification of the physical interfaces, one can improve its performance, flexibility, reliability, and interoperability [2].

The major standards DO-178B, DO-214, DO-254 and ARP 4754A are important for the certification process of the aircraft cabin. From these standards, we later derive requirements concerning worst case signaling delay, low audio delay, and synchronous playback.

- DO-178B: Software Considerations in Airborne Systems and Equipment Certification
- DO-214: Audio Systems Characteristics and Minimum Operational Performance Standards for Aircraft Audio Systems and Equipment Systems and Equipment
- DO-254: Design Assurance Guidance For Airborne Electronic Hardware
- ARP 4754A: Guidelines for Development of Civil Aircraft and Systems

Table 2.2 summarizes the CIDS requirements that have to be proven during the certification process. These requirements have the following consequences: In order to achieve synchronous playback with an accuracy better than 1 ms, we either have to

Function	Max. Latency [ms]	Sync. [ms]
Signaling	100.0	–
Audio Playback	10.0	1
Cabin Interphone	100.0	–

Tab. 2.2: Requirements in CIDS

introduce a common time-base with such an accuracy, or assure that the multi-cast delay difference is lower than 1 ms. If we follow the latter approach, 9 ms are left and can be consumed on the path between the data source and the first multi-cast packet replication.

2.1.2 Current Architecture of Cabin Management System

In this work, we address the cabin networks in Airbus aircraft with a special focus on safety. Basically, the communication networks in an aircraft are classified into the following four domains: (a) *Aircraft Control Domain* (ACD), (b) *Airline Information Services Domain* (AISD), (c) *Passenger Information and Entertainment Service Domain* (PIESD) and (d) *Passenger-Owned Devices Domain* (PODD). These four domains are described in [10] and given in Table 2.3. The CIDS belongs to the domain ACD and describes the major system to control the safety-relevant cabin functions, such as PA, *Prerecorded Audio* (PRAM), and CVMS. The network topology consists of several parallel buses, each one connecting up to 16 intermediate hubs, which in turn act as protocol converters. The communication between the protocol converters and the end devices employs a couple of different protocols, such as LVDS to connect the PSUs, RS485 to connect the smoke sensor, and Ethernet to connect the *Illumination Ballast Units* (IBUs). The backbone of this system uses 10BASE2 on the physical layer, although non-Ethernet compliant frames are used to transport payload. The frames are not compliant to Ethernet frames for the following reasons: A different CRC is used, a shorter IFG¹ is used, and no MAC² address information is available in those frames.

IFE is the entertainment system that allows passengers during flight to watch movies, browse the web, and play games. IFE belongs to the domain PIESD and currently uses separate network infrastructures. Using separate networks has the advantage that the

¹ Inter Frame Gap

² Medium Access Control

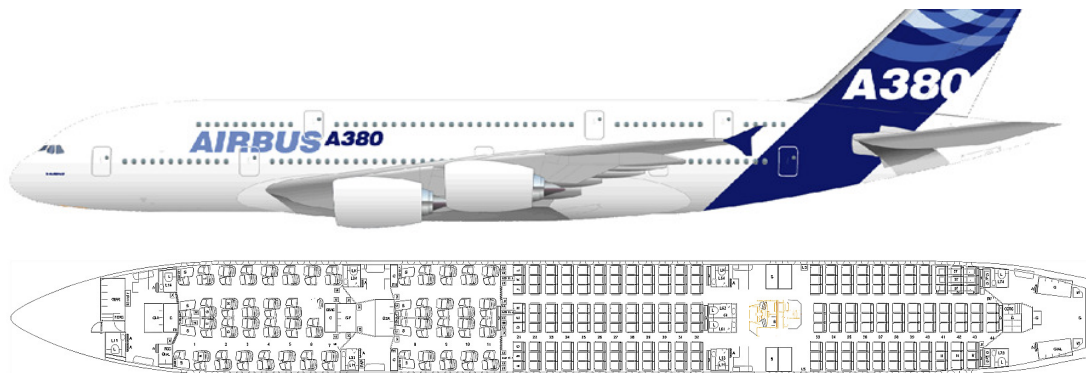


Fig. 2.1: Schematic of Aircraft Cabin, Courtesy of Airbus

different safety levels are not mixed on the same transport medium. Certainly manufacturers have to pay for this by a significant weight increase due to the additional wiring required. A fair question is whether different domains can share the same medium while still adhering to the imposed safety levels.

Figure 2.1 is a schematic for modern aeronautic cabins of the Airbus family. *Three redundant servers* are located at the head of the aircraft cabin, but only one is active at a time. The redundant fall-back alternatives are attached redundantly to the cabin network infrastructure and take over in case of a server error, using hot standby.

The end devices are spread over the whole aircraft cabin. Typically, we find the following types within the CIDS functionality in aircraft cabins: PSU, IBU, smoke detectors, and cabin handsets. The connections to these end devices run largely in parallel: up to 22 lines are used in the largest Airbus family A380. Along the line, up to 16 *Decoder Encoder Units* (DEUs) are attached to the bus, which provide a gateway between CIDS and the attached end devices.

ACD	AISD	PIESD	PODD
Aircraft Control Domain	Airline Information Services Domain	Passenger Information and Entertainment Services Domain	Passenger Owned Devices Domain
Control of the Aircraft	Operations of the Airline	Entertain Passengers	

Tab. 2.3: Aircraft Cabin Domain

Type	Quantity	Description
PSU	1536	Passenger Service Unit
IBU	1512	Illumination Ballast Unit
Handset	10	Cabin Handset
FAP	30	Flight Attended Panel
CVMS	26	Cabin Video Monitoring System

Tab. 2.4: Number of Devices in the A380 Family

Each DEU splits information from arriving frames and distributes them to the connected devices. In order to give information from the end devices back to the server, the DEU collects information from the end devices and assembles them into a CIDS Ethernet frame. So, the DEU acts basically as a protocol converter and gateway, where proprietary Ethernet frames are converted to LVDS, RS485, or CAN, and vice versa. There exist two different line types in the CIDS system: the top-line and the middle-line. The top-line is used for passenger functions at PSU or IBU, the middle-line is used for crew functions, such as crew intercommunication or smoke detection. The number of end devices as well as the number of repeaters in one line depends on the aircraft family, and is given in Table 2.4.

2.1.3 Summary

Functions in the aircraft cabin management system are related to safety. The network must therefore provide guarantees on performance bounds to allow synchronous audio playback, low audio delay, and a maximum worst case signaling delay. Table 2.5 outlines the essence of the preceding section.

Description	Fact
Number of End-Devices in Aircraft Cabin	up to 4000
Maximum Signaling Delay in Aircraft Cabin	100 ms
Maximum Audio Delay in Aircraft Cabin	10 ms
Synchronous Playback in Aircraft Cabin	1 ms
Maximum Ethernet Hop Count	16

Tab. 2.5: Aircraft Cabin Management System — Summary

2.2 Performance Bounds in Queuing Networks

This chapter addresses the methods and techniques that predict the delays and frame losses in switched Ethernet networks. The level of precision is highly dependent on the approach chosen and also on the quality of the traffic model employed. We pay special attention to *Network Calculus*, which allows capturing queuing and transmission delays with the $(\min,+)$ -algebra. The identified bounds are useful when we are interested in bounds that hold under any circumstance, such as those used to evaluate safety critical systems in the aeronautics or automotive industry.

2.2.1 Determining Performance Bounds

In communication networks, there exist *three major approaches* to determining meaningful bounds of real-time applications: (a) analytical methods, (b) network simulation, and (c) measurements. Analytical approaches determine performance bounds by theoretical frameworks like Queuing theory and NC and have the advantage of providing a *strong mathematical boundary*. In the field of performance evaluation, another well accepted method is that of *Discrete Event Simulation* using Monte Carlo methods. Section 5.1 contains a comprehensive overview of network simulation and popular network simulators such as ns-3 [92], OMNeT++ [94] and OPNET [95]. Determining performance bounds by measurements of a real system would certainly provide bounds that are likely to be closer to reality than those from a simulation approach. However, setting up a complete realistic scenario for measurement may be too complex and too expensive, so that a simulative approach would be preferable.

To evaluate safety critical systems, a mixture of these approaches is usually used to prove correct functionality. Analytical methods have the drawback that the performance bounds determined may differ significantly from those of a deployed network. Consider the worst case scenario in a queuing network. The worst case will only be provoked when the packets of interest are delayed by all other crossing packets, which is very unlikely to happen.

Typically, the following delay types are observed in switched Ethernet: (a) Propagation Delay, (b) Transmission Delay, (c) Processing Delay, and (d) Queuing Delay. Current work primarily focuses on determining the worst case values for transmission and queuing delays. Due to the nature of queuing networks, where arbitrary cross traf-

fic may pass the system, this may lead to high variability. In contrast to transmission and queuing delay, values for propagation and processing delay are relatively stable, which allows providing tight bounds. Bounds for the processing delay are highly dependent on the multiplexing architecture assumed, i.e., whether the switch fabric is implemented in hardware or as a software solution.

Proving the correctness of queuing networks, does, however, still contain some unsolved problems—methods known so far either pay by significant overestimation or by computational effort. While overestimated bounds are sometimes tolerable, the computational effort of tight bounds is usually impractical for larger networks [101]. In fact, the problem of finding tight bounds has been proven to be NP-hard [16] for the general case, so that an exponential number of solutions has to be compared in order to find the tightest bound when using the known linear optimization based algorithms of [16, 103].

In the remainder of this section, we introduce another pitfall when determining performance bounds in queuing networks. A simple switching scenario is given in Figure 2.2. The non-FIFO³ may be necessary in this scenario, despite the fact that the queuing policy is FIFO. Bennett et al. [14] found that packet reordering is not a rare event, and naturally occurs due to local parallelism. Rizzo et al. [100] put this fact into the context of NC, and underline the necessity of the non-FIFO bound. As a consequence, we may have to consider the non-FIFO bound instead of the FIFO bound of a queuing system, and this heavily depends on whether the employed switch is able to guarantee real FIFO forwarding or not. This seems to contradict the fact that AFDX networks only consider the FIFO bound. Researchers as well as engineers thereby start from the premise that the switch fabric has a bounded delay on multiplexing. This may be

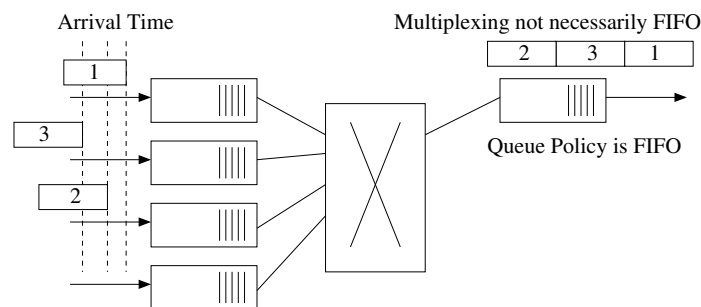


Fig. 2.2: Arbitrary Multiplexing in Switches

³ First In First Out

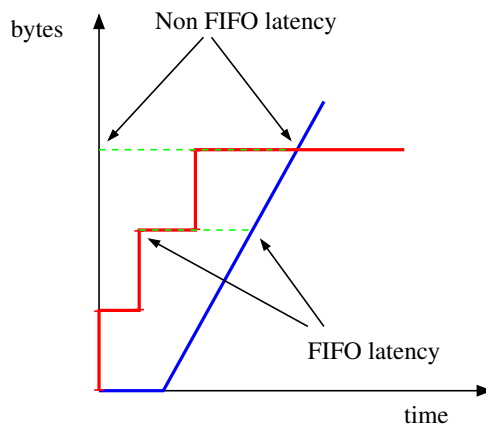


Fig. 2.3: Cumulative Arrival, FIFO Bound Versus Non-FIFO Bound

assumed for AFDX switches. However, we observed non-FIFO behavior during the simulation and measurements of an Ethernet network consisting of standard COTS devices, as is planned for the aircraft cabin.

Figure 2.3 gives a cumulative arrival of the scenario and shows the difference between FIFO and non-FIFO bound. In fact non-FIFO bounds would not be required for performance evaluation if the implemented switch could guarantee FIFO forwarding [100]. In order to do so, the implementation of the crossbar would have to store the concrete arrival time of each packet to guarantee true FIFO behavior. Despite the fact that the concrete implementation details are not published by hardware vendors, it is assumed that most switch implementations do not guarantee FIFO forwarding. Instead maximum bipartite matching algorithms will likely be implemented to achieve high throughput in the crossbar [83].

2.2.2 Quality of Service

Soldatos et al. provide a comprehensive study of the building blocks of QoS [108] and provide the following order and grouping: (a) *Admission control*, (b) *Shaping and policing*, (c) *Signaling and resource management*, (d) *Queuing and scheduling*, (e) *Congestion control and queue management*, (f) *QoS routing*, (g) *QoS policy management*, and (h) *QoS pricing*. We see that the building blocks *Admission control*, *Signaling and resource management*, as well as *Congestion control and queue management* are important in the sense that their dynamic character can achieve good performance in Internet applications and react to sudden changes in the amount of traffic. However, the automotive industry and

the aeronautic industry agree that this dynamic character (be it the online modification of bandwidth reservations, be it TCP with its window-based congestion control algorithm, or be it an active queue management algorithms) is *too difficult to handle* within safety related applications. The building blocks *QoS routing*, *QoS policy management*, and *QoS pricing* are important for providers and the management of ASs⁴.

As a result, we have identified the importance of the following remaining building blocks while addressing the novel aircraft cabin:

- *Shaping and policing* and
- *Queuing and scheduling*

Traffic shaping and policing is well known in the literature [110, 113] and strongly interwoven with the theoretical schemes Token bucket and Leaky bucket algorithm. The networking literature boils these two different algorithms down to the Token Bucket model given in Section 2.2.2, which is able to cover both policing and shaping. We also point out how those shaping and policing algorithms are commonly implemented in hardware.

The building block *Queuing and scheduling* is used in packet-switched networks, as they might contend for access to an outgoing link [108]. Since the sum of the ingress rates that are to be forwarded on the same output link might exceed the outgoing link's capacity, switch buffers, where packets are temporarily queued, are necessary. In order to share bandwidth between competing packets, scheduling algorithms were developed that target scheduling policies, such as achieving a fair share of the bandwidth or minimizing the delay of certain packets.

In order to employ those QoS queuing and scheduling techniques to the traffic flows that flow through the network, an important task is to distinguish different flows by information in the packet headers. Throughout this work, we use field information in the Ethernet header.

Figure 2.4 shows the format of an Ethernet header in accordance with [46, 47, 48]. The fields *Dest* and *Src* store the destination and source MAC addresses. VLANs are an extension to the IEEE 802.1D standard [46] and are defined in IEEE 802.1Q [47]. They are intended to create logical groups in broadcast domains and thus help the administra-

⁴ Autonomous System

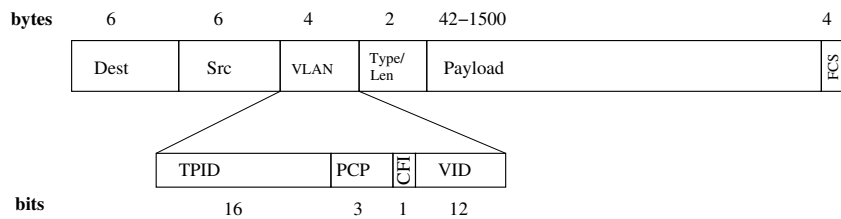


Fig. 2.4: Ethernet Header

tion of VLAN groups. The VLAN tag is optional and consists of a fixed TPID⁵ with the fixed value 0x8100, the PCP⁶, CFI⁷, and VID⁸. The PCP field is a 3-bit wide field and carries priority information. The VLAN identifier is a 12-bit wide field which takes values from 0 to 4095. VID 0, 1, and 4095 are reserved.^{9 10 11} The *Payload* holds the payload with length of 42 to 1500 bytes. The *Type/Len* field defines the protocol and/or length of the payload. The FCS¹² is 32 bits wide and holds a CRC¹³ checksum of the Ethernet frame.

The priority information is mapped to an internal queue in the switch. The internal queues are then served by scheduling algorithms, which can thus provide QoS guarantees even for the lower priority queues. That priority information can also be derived from the IP header, or even from side information available in the switch, such as the ingress or egress port.

Traffic Models

Several models have been proposed to describe network traffic. In the field of NC, the token bucket model serves as the theoretical envelope for upper bound traffic occurrence. In the remainder of this section, we will give an introduction to the single token bucket model and the dual token bucket model. In the field of Network Simulation, we require different models to express arrivals adequately, ideally based on stochastic pro-

⁵ Tag Protocol Identifier

⁶ Priority Code Point

⁷ Canonical Format Indicator, fixed to zero for switched Ethernet, for historical reasons [46]

⁸ VLAN Identifier

⁹ Value 4095 is reserved for implementation use.

¹⁰ The value 0 is the null VLAN id and indicates that only priority information is available.

¹¹ The value 1 is the default port VLAN and used for management operations in managed switches (cf. [90]).

¹² Frame Check Sequence

¹³ Cyclic Redundancy Check

cesses. Common models in network simulation are based on Markov chains and might even create traffic-dependent random variables for the rate and packet size. Some of those models even pay attention to the self-similarity of Internet or Web traffic.

Token Bucket The general token bucket algorithm allows bursts of b token items (usually bytes or frames) that exceed the rate r before limiting or marking operations take place. Interpreting this algorithm as a traffic model allows determining an upper bound to the traffic generation pattern of an arbitrary source. The literature commonly refers to this kind of traffic as being (σ, ρ) -constrained, where σ denotes the burst and ρ the rate.

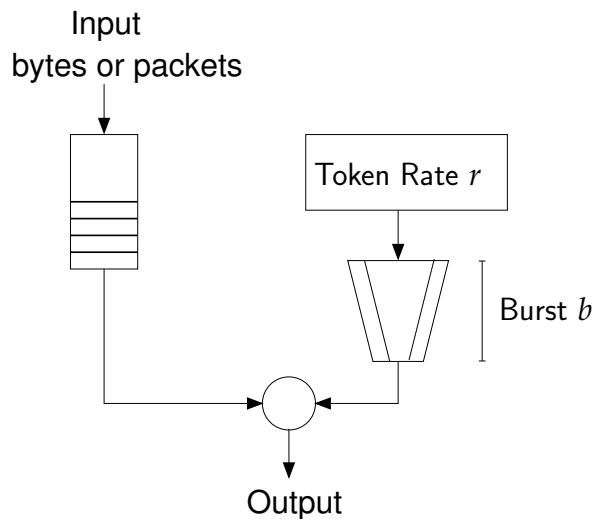


Fig. 2.5: Token Bucket Model

Dual Token Bucket In scenarios where variable bit rates play an important role (such as ATM or IP¹⁴), we require arrivals that conform to *two leaky buckets* (cf. [75]). This second model is also called the dual token bucket model and models the peak rate (r_1) and the average rate (r_2), the peak burst (b_1) and the average burst (b_2).

With this model, computer networks that *usually emit frames at wire speed* can be modeled adequately. As an example, we introduce the *dual token bucket model T-SPEC* known from *IntServ*. The T-SPEC consists of the 4-tuple (p, M, r, b) (cf. [75]), where

¹⁴Internet Protocol

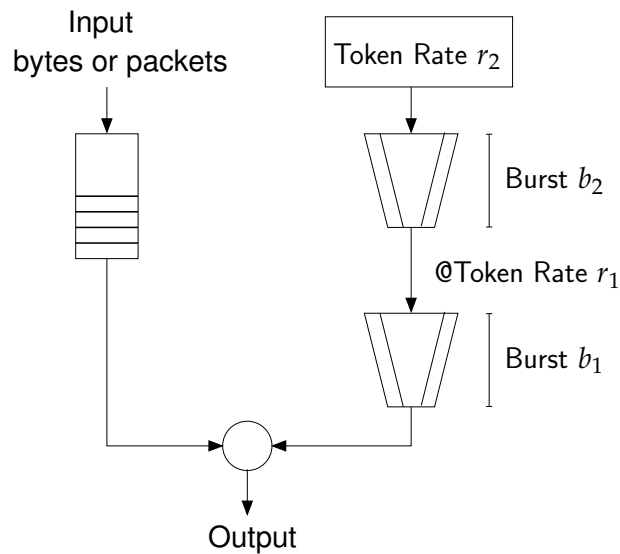


Fig. 2.6: Dual Token Bucket Model

- p is the peak rate,
- M is the maximum packet size,
- r is the rate of the token bucket, and
- b is the burst of the token bucket.

In addition to this dual token bucket model, we can even think of multiple token bucket models having several token buckets in a chain.

On/Off Traffic Model The On/Off traffic pattern is a traditional traffic pattern to model network arrivals by a Markov chain. Figure 2.7 illustrates this traffic model. The data source only generates traffic if the system is in the On state, and creates no traffic in the Off state. In contrast, modern network simulators determine the On and Off time according to some distribution (likely to be exponential or Poisson) and change the state if completely consumed. The traffic composition itself may depend on random variables that determine the traffic rate as well as the packet size.



Fig. 2.7: On/Off Traffic Pattern

Self-Similar Traffic Models In the last century, network arrivals were often modeled as Poisson processes [96]. As a matter of fact, the Poisson distribution was widely used to describe telephone calls, but it turned out that employing an exponential distribution to describe Internet or Ethernet traffic was not appropriate. In order to cope with this novelty, various traffic models have focused on the self-similarity of Internet or Ethernet traffic. The standard literature includes the following. Leland et al. introduce a self-similar model for Ethernet traffic [77]. Paxson et al. introduce a model for TCP traffic [96] and Crovella et al. introduce a model for WWW traffic [25].

Filter Techniques

In this section, we give a short overview of common filtering techniques that are available by standard Layer 2 COTS switching ASICs, as required in the aircraft cabin network.

MAC based filtering The Ethernet header given in Figure 2.4 contains a field source MAC which is used for Layer 2 addressing in the network. The generating source device copies its own MAC address into the source field and the destination MAC address of the addressed Ethernet device. If the destination address is still unknown, the broadcast MAC address is used as the destination address, so that the addressed device can be reached. In order to avoid too many broadcast messages in the network, an address resolution protocol like ARP¹⁵ is used to map the higher layer addresses, such as IP, to MAC addresses. When the packets traverse the network, the Layer 2 Ethernet switch evaluates the MAC address. If the source MAC address is as yet unknown, the switch stores the information of the source MAC address and associated ingress in an internal MAC table. The destination MAC address is used to identify the output port. If the MAC address is known, i.e., found in the internal MAC table, we use that information to determine the output port. Otherwise, if the MAC address is unknown or if the broadcast bit is set in the MAC address, we broadcast that packet to all ports except the ingress port. At this stage, the internal MAC table can also be used for filtering operations, i.e., entries in that table can be used to interdict forwarding of packets with a certain MAC address.

¹⁵ Address Resolution Protocol

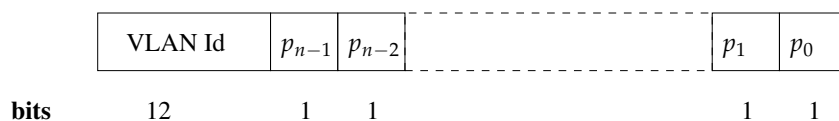


Fig. 2.8: VLAN Table Entry

VLANs Similar to the internal MAC table as introduced in the last paragraph, a VLAN capable switch provides an internal VLAN table. This table specifies VLANs by storing a table entry with the information of the VLAN id and a bit mask of the participating ports. As a matter of fact, common switching ASICs do not always provide the full range of 4096 VLAN entries: the maximum number of VLAN entries may be 16 or 64, and is highly dependent on the actual switching ASIC. Figure 2.8 illustrates the structure of a VLAN table entry.

This VLAN table may also be used to achieve rigorous filtering on a VLAN basis. Tagged Ethernet frames may only egress at those ports whose bit is set in the respective VLAN table entry, otherwise they are discarded. In most switching ASICs, there is also the possibility of specifying the port VLAN ids, i.e., untagged frames will be tagged with the respective port VLAN id and port priority. Some ASICs also provide enforcement to set this information on egress.

This filtering mechanism based on VLANs is extensively used in the novel aircraft cabin to segregate traffic classes and specify conforming forwarding rules.

Filtering based on higher layers There are certainly filtering techniques that evaluate information from layer 3 and up. Known techniques evaluate IP addresses, TOS¹⁶ information, provide stateful filtering or apply deep packet inspection. However, we only provide layer 2 filtering techniques since they may be provided by low cost Ethernet switches. For sophisticated firewall solutions, as might be used in corporate networks, no such low cost solutions are known.

Traffic Regulation

In this part, we address the traffic shaping and traffic policing algorithms that are typically implemented in modern switching hardware, such as the Marvell 88E6097 [80].

¹⁶ Type of Service

We identify the following two major traffic regulation techniques: (a) Traffic shaping, and (b) traffic policing.

A traffic shaper is responsible for shaping incoming traffic into a predefined form where a minimal IFG is guaranteed. The exact value of the minimal IFG is derived from the bandwidth the shaper is intended to guarantee. In contrast to traffic processed by traffic policing, shaped traffic does not exceed the guaranteed maximum bandwidth at any time, i.e., no intermediate bursts are allowed. As a side effect, the overall delay might increase due to delaying packets that do not satisfy IFG. Traffic shapers do not drop packets themselves; however, queues might overflow, so that packet dropping is unavoidable.

Traffic policing is used to check whether incoming traffic conforms to a given traffic description. If not, actions such as packet dropping or flow control might be triggered [48]. Packet dropping due to non-conforming traffic is also called traffic limiting. As a result, traffic limiting allows a specified amount of bursts until packet dropping becomes active.

Figure 2.9 shows the typical mechanism of traffic shaping. A counter (y-axis) is increased by the number of transmitted bytes (or transmitted packets) when packets are forwarded to the output port. The switch is only allowed to send packets to the respective output port if the counter is lower than a certain predefined value (the allowed-to-send barrier). The counter is decreased at a specified slope, and stays at the lower barrier. The illustrated data flow gives the idea of traffic shaping: Packets are forwarded at a constant rate and non-conforming traffic will be queued in the buffer. However, if the queue overflows, packets are dropped.

Figure 2.10 shows a feasible implementation of traffic policing with action drop. A counter that is initially set to zero decides whether an ingressing packet may be forwarded or not. This is done adding the packet size to the actual counter value. If this new value is lower than a specified limit (the burst limit), the packet may egress and the counter takes this new value. Otherwise, a burst exceed action is triggered. In this example, the packet is discarded. Similar to the recently introduced traffic shaper, the counter value is decreased to zero at a specified slope. This slope corresponds to the allowed average bandwidth. As a result, limiting takes place if a certain burst is consumed.

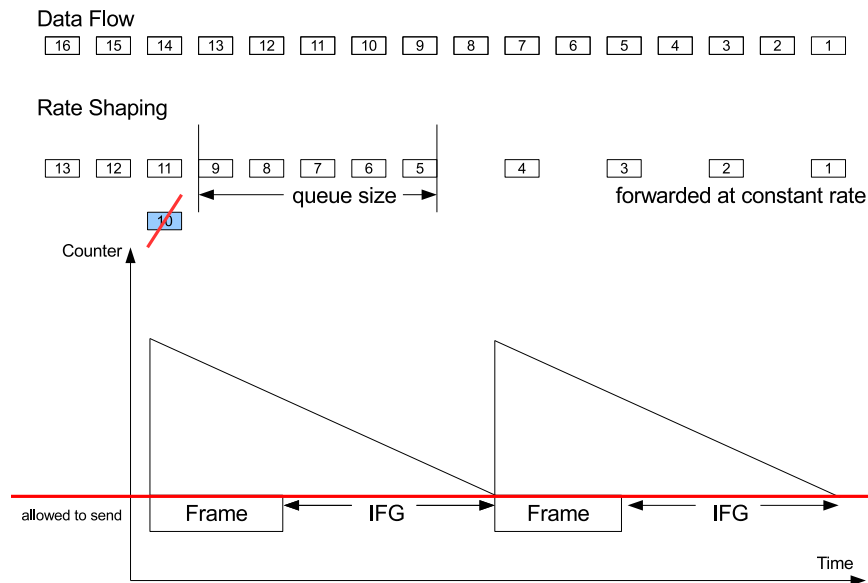


Fig. 2.9: Traffic Shaping

Schedulers in Queueing Networks In the remaining part of this section, we address the QoS building block of queuing and scheduling. Scheduling algorithms allow achieving certain QoS goals, such as sharing bandwidth resources equally or minimizing the end-to-end latency of high priority packets. In the following, we classify the scheduling algorithms into two subcategories: *work-conserving scheduling* and *non-work-conserving scheduling*. A scheduling algorithm is work-conserving if the link is never left idle when there are packets in the queue [108]. Otherwise, they are non-work-conserving. We might figure out at this point that non-work-conserving schedulers are closely related to traffic shapers, i.e., non-work-conserving schedulers might increase IFG with the result of potential increased delays.

If there exists only one output queue per port, the scheduling is obviously FIFO which may be the simplest case of queuing strategy. However, in FIFO queuing discipline, misbehaving senders may exhaust network resources and no differentiation of performance levels is possible [108]. Instead of assuming one single queue, there are usually several queues per port in order to achieve different QoS levels. When several queues are available in the output buffer, the actual queue is chosen by some mapping from priority to internal queue number. The priority is either determined by some field information in the protocol headers, or even determined by some side information such as the ingress port. For processing the different queues, a scheduling algorithm is employed that serves a specific QoS goal.

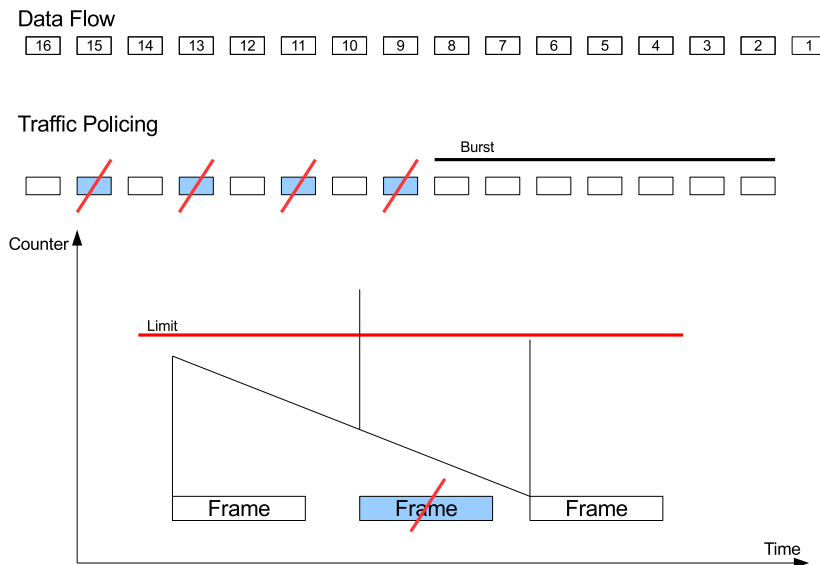


Fig. 2.10: Traffic Limiter

The major work-conserving scheduling algorithms are *Strict Priority Queuing* (SPQ), *Fair Queuing* (FQ) and *Weighted Fair Queuing* (WFQ) [108]. The queuing algorithm SPQ serves each queue after the other according to their assigned internal priority. The queue with a lower priority is only served if no other packets in the higher priority queues are available. As a result, SPQ achieves very low additional latencies for the highest priority class, but the queues of lower priority might get starved due to excessive loads in the queues with higher priority. The risk of starvation is usually tackled by some FQ scheduling approach. For this, FQ serves queues in a round-robin mode, but also takes the number of transmitted bytes into account [108] so that bandwidth is shared fairly. WFQ extends the concept of FQ and allows setting different weights on the bandwidth share (again [108]). As a result, different bandwidth guarantees for each queue can be given.

The *Credit Based Queuing* (CBQ) algorithm is a non-work-conserving scheduling algorithm. This scheduling algorithm not only guarantees a certain bandwidth to a queue, but also that this bandwidth is not exceeded. Hence this scheduling algorithm has to increase the IFG if the queued traffic does not conform to the specified rate. The CBQ has gained a lot of attention in recent years as part of the IEEE standard *Audio-Video Bridging* (AVB). Since this algorithm is fundamental for AVB, which allows synchronous playback of audio and video, we describe it explicitly in the following section.

Audio Video Bridging—AVB

This section discusses the IEEE standard AVB. AVB is a set of IEEE standards developed under the Audio/Video Bridging Task group. As part of 802.1, AVB addresses low latency and highly synchronized playback of audio and video using standard Ethernet. The key factor is providing mechanisms to achieve the objectives of low loss, low jitter, and low latency. A common notable catch phrase is that AVB is able to transmit and play audio with a maximum latency of 2 ms over seven hops. This is achieved by AVB's *Priority Queuing* (PQ) and CBQ algorithms [55]. Present work is covered especially by the following main standards:

- IEEE 1722 — AVB Transport Protocol [53]
- IEEE 802.1AS — Time Synchronization [54]
- IEEE 802.1Qat — Stream Reservation Protocol [52]
- IEEE 802.1Qav — Queuing and Forwarding [50]

The terminology of AVB contains AVB end points, non-AVB end points, AVB bridges, and non-AVB bridges. AVB endpoints are either talkers (AVB endpoints that produce data streams), listeners (AVB endpoints that consume data streams), or both.

IEEE 1722 — AVB Transport Protocol (AVBTP) Figure 2.11 shows the structure of an IEEE 1722 AVB frame. Inside the AVB layer, the standard protocols IEC 61883 [57] are used to carry audio and video payload, i.e., we use a protocol that is called IEC 61883/I-IDC over AVBTP. The IEC 61883 protocols define the format of different payload data, such as the MPEG2-TS data protocol and the Audio and Music data transmission protocol [114]. IEEE 1722 defines the required adaptations that make IEC 61883 ready for transport via AVBTP, i.e., adaption to the IEEE 802 networks as well as adaptations for precise time synchronization.

IEEE 802.1AS— Time Synchronization A mandatory point in the synchronous playback of audio and video is the presence of a common time base. There exist a couple of protocols or mechanisms that are able to distribute a common time base, such as the NTP¹⁷. However, the newer *Precision Time Protocol* (PTP) has gained much acceptance

¹⁷ Network Time Protocol

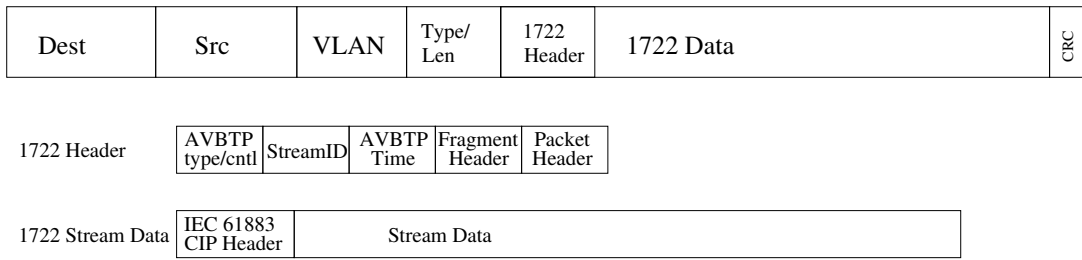


Fig. 2.11: AVB Transport Protocol Frame [93]

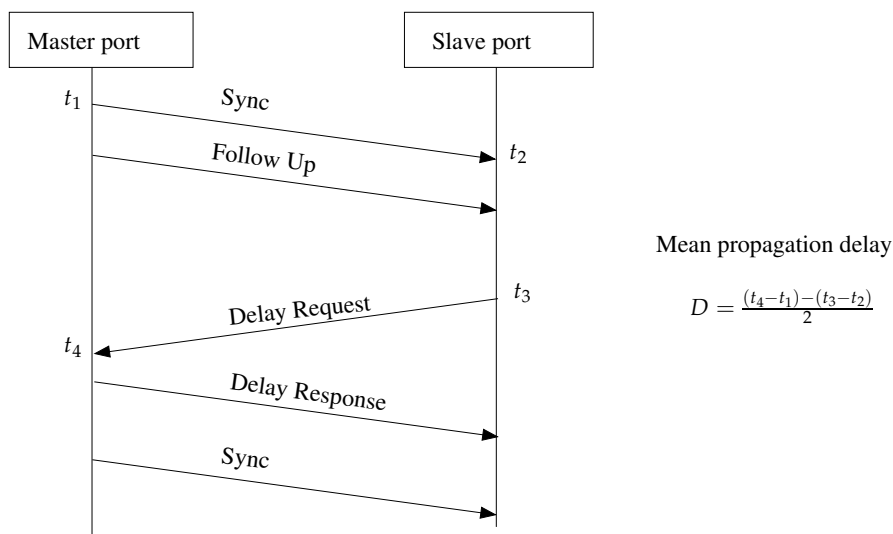


Fig. 2.12: PTP Synchronization (Adapted From [54])

in the last years, as it has some advantages for achieving precision [34]. Figure 2.12 shows the PTP algorithm, which operates as follows [54]: The timestamps t_1, t_2, t_3, t_4 are generated at the respective device in order to measure the delay. The following message types are defined in the standard: Sync message, follow up message, delay request, and delay response. The slave device requires the timestamps t_1, t_2, t_3, t_4 to determine the clock difference from the master clock and to adjust the slave clock. The timestamp t_1 is transmitted with the follow up message to the slave, t_4 is transmitted with the delay response message.

The time synchronization standard IEEE 802.1AS is based on the standard IEEE 1588 [51], i.e., the standard which proposes PTP for network measurements. Clause 7 of 802.1AS introduces the *generalized Precision Time Protocol (gPTP)* has some specifics and differences. The most important are:

- IEEE 802.1AS synchronization packets use the destination (multicast) MAC address (01-80-C2-00-00-0E) [82].
- IEEE 802.1AS provides hardware time stamping, which allows identifying additional delays by intermediate processing [82].
- In gPTP there are only two types of time-aware systems: end stations and bridges. In contrast, IEEE 1588 allows the relay of PTP synchronization messages via components that are not time-aware [54].

The standard conforms to existing standards, such as IEEE 802.3 (Ethernet) and IEEE 802.11 (Wifi) (cf. [54], clause 5). However, since it is mandatory for all participants to be time-aware, the switches and network cards must be capable of performing hardware time stamping. The great benefit of gPTP and, thus, IEEE 802.1AS, is of course the very high precision achieved, since the standard demands hardware time stamping. The drawback of the fact that PTP requires symmetric delays in the delay measurements is also softened by this precision. With respect to AVB bridges, the first respected manufacturers, such as Broadcom [19], Marvell [81], and Netgear [89] provided the first hardware support for IEEE 802.1AS.

IEEE 802.1Qav — Forwarding and Queuing Enhancements To guarantee the requirement of a maximum network delay of 2 ms over seven hops, a special focus must be made on the forwarding and queuing capabilities of the intermediate switches, i.e., the AVB bridges. Provided that we know the mechanisms of rate limiting and rate shaping of modern Ethernet switches, we can quickly introduce the forwarding and queuing mechanisms that constitute the behavior of AVB bridges.

Standard Ethernet COTS switches usually provide two or four FIFO queues, each being assigned an internal priority. The QoS priorities as determined from the Diffserv code-points or the IEEE 802.1q priorities are in turn mapped to these internal QoS classes. Again, AVB bridges extend this concept and add two time-sensitive queues employing the CBQ algorithm. One advantage of this concept as regards forwarding and queuing is that the shaping is done on a per queue basis rather than on a per port basis, which last is very common for standard Ethernet switches. Figure 2.13 shows the model of an AVB output port as it is required by [50].

A remarkable property of CBQ is that whenever a bandwidth reservation is unused or only partly used by the corresponding queue, the remaining bandwidth is devoted

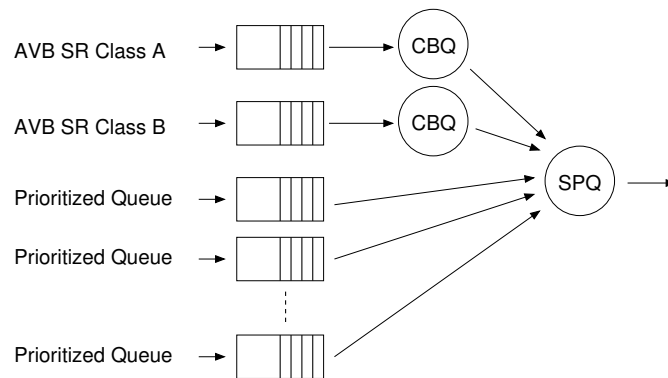


Fig. 2.13: The AVB Output Port

to the next lower traffic queue [55]. Figure 2.14 shows a timing diagram considering the important states in an AVB bridge. Without going into extensive details of this algorithm, we summarize the following abbreviations:

- *noQFs* — number of queued frames,
- *ConFP* — conflicting frame present, higher prioritized frame available or currently sending,
- *idleSlope* — credit rate in bit/s while in idle, i. e., bandwidth guaranteed to queue,
- *sendSlope* — credit rate in bit/s while sending, i. e., remaining bandwidth.

Respected manufacturers such as Broadcom [19] and Marvell [81] provide the first hardware support for AVB queuing.

IEEE 802.1Qat — Stream Reservation Protocol SRP¹⁸ [52] is used to announce and accomplish reservation requests. SRP defines two signaling protocols, the MMRP¹⁹ and the MSRP²⁰ [52]. MMRP is used to register and unregister talkers and listeners, MSRP is used to perform the actual reservation.

The standard defines the structure of the MMRP protocol data units in BNF form, encapsulating the so called FirstValue, which contains MSRP application specific data,

¹⁸ Stream Reservation Protocol

¹⁹ Multiple MAC Registration Protocol

²⁰ Multiple MAC Registration Protocol

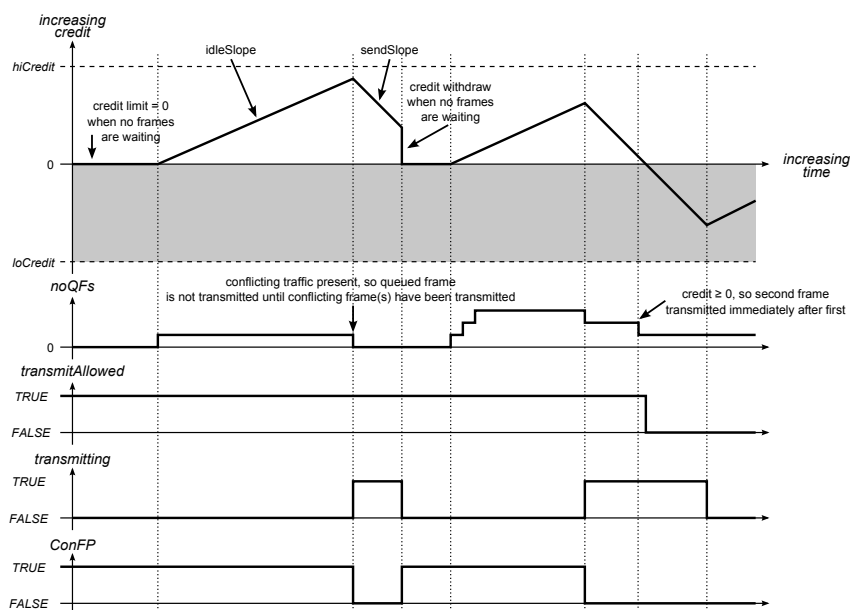


Fig. 2.14: Credit-Based Shaper Queuing and Scheduling Operation [50], as in [55]

such as StreamID, Priority and TSPEC [59]²¹. With this information, the AVB bridges are able to perform the actual reservation by determining concrete values for *idleSlope* and *sendSlope*, as required by 802.1Qav [50].

2.2.3 Network Calculus

Network calculus was first proposed by Cruz in 1991 (cf. [26], [27]). Le Boudec [75] proposed the employment of an algebraic model, which led to the current common understanding of a deterministic network calculus. In recent years, the research community involved in the network calculus has introduced stochastic processes to enable more realistic models of traffic generation and service curves.

Introduction to the Network Calculus

In this section we discuss common techniques for determining worst case delays with NC. It is known (cf. [35, 104]) that restricting the topologies to feed-forward networks

²¹ TSPEC was initially used by IntServ to describe the character of a network flow, paying special attention to intermediate bursts and frame lengths.

is mandatory for NC to obtain stable performance bounds. Simply put, a feed-forward network is a network with a DAG²² or tree topology, or, more formally:

Definition 2.2.1. FEED-FORWARD NETWORKS [104]. *If it is possible to find a numbering of its links such that for any flow through the network the numbering of its traversed links is an increasing sequence, a network is called feed-forward.*

Restricting the topology to feed-forward networks is a good idea in the field of NC, because this property guarantees that NC calculates finite bounds if utilization is less than 1. With respect to computational complexity, an important subcategory of feed-forward networks is the tandem network:

Definition 2.2.2. TANDEM NETWORK [16]. *A tandem network N is a network, such that the induced digraph $G(N)$ is a directed path with no shortcut.*

The theory of NC is based on the work of Cruz [26, 27] and was shifted by Le Boudec towards the $(\min, +)$ -algebra [75]. It is well known that the comfortable use of the $(\min, +)$ -algebra contributed to the success story of NC, making this theoretical framework a valuable tool for determining numerical bounds on network performance.

Deterministic NC expresses network services and traffic flowing over this network in terms of arrival and service curves [75]. We first give a definition of wide-sense increasing functions. These functions are extensively used in NC and have been later used to define arrival and service curves.

Definition 2.2.3. WIDE-SENSE INCREASING (W.S.I.) [45]. *A function f belongs to the set of wide-sense increasing functions F iff (i.e., if and only if):*

$$f(s) < f(t), \forall s < t \quad (2.1)$$

Le Boudec [75] defines two key operations in the min-plus calculus: Min-Plus Convolution (Definition 2.2.4) and Min-Plus Deconvolution (Definition 2.2.5). Min-Plus Convolution is used to define service curves (Definition 2.2.7) and for the concatenation theorem 2.2.1. Min-Plus Deconvolution is primarily used to bound traffic given as an arrival curve that passes a server.

²² Directed Acyclic Graph

Definition 2.2.4. MIN-PLUS CONVOLUTION [75]. Let $f, g \in F$ be two w.s.i. functions or sequences. The min-plus convolution of f and g is given by the function

$$(f \otimes g)(t) = \inf_{0 \leq s \leq t} \{f(t-s) + g(s)\}. \quad (2.2)$$

(If $t < 0$, $(f \otimes g)(t) = 0$).

Definition 2.2.5. MIN-PLUS DECONVOLUTION [75]. Let $f, g \in F$ be two w.s.i. functions or sequences. The min-plus deconvolution of f by g is the function

$$(f \oslash g)(t) = \sup_{u \geq 0} \{f(t+u) - g(u)\}. \quad (2.3)$$

Arrival curves are used to give an upper bound to the arrivals of bit streams. These arrival bounds are used at the traffic generating nodes and between service nodes, which generate traffic themselves due to forwarding.

Definition 2.2.6. ARRIVAL CURVE [75]. Given a w.s.i. function α defined for $t \geq 0$, we say that a flow R is constrained by α iff $\forall s \leq t$:

$$R(t) - R(s) \leq \alpha(t-s) \quad (2.4)$$

We say that R has α as an arrival curve, or also that R is α -smooth.

We now introduce the concepts of service curve and strict service curves, in Definition 2.2.7 and Definition 2.2.8, as well as the concepts of minimum and maximum service curve, in Definition 2.2.9. Service curves are used to express guarantees to flows that traverse network nodes (cf. [75]). Strict service curves are simple service curves that in addition satisfy the requirement that their output is at least equal to the given service curve if backlog is available.

Definition 2.2.7. SERVICE CURVE [75]. Consider a system S and a flow through S with input and output function R and R^* respectively. We assume that S offers the flow a service curve β if and only if β is wide sense increasing, $\beta(0) = 0$ and $R^* \geq R \otimes \beta$.

Definition 2.2.8. STRICT SERVICE CURVE [75]. We say that System S offers a strict service curve β to a flow if, during any backlogged period of duration u , the output of the flow is at least equal to $\beta(u)$.

Definition 2.2.9. MINIMUM AND MAXIMUM SERVICE CURVE [103]. *If the service provided by a system S for a given input function R results in an output function R^* we say that S offers a minimum service curve β respectively a maximum service curve $\bar{\beta}$ if and only if*

$$R^* \geq R \otimes \beta \text{ respectively } R^* \leq R \otimes \bar{\beta} \quad (2.5)$$

The concatenation theorem 2.2.1 uses min-plus convolution to concatenate servers. The end-to-end service curve can then be employed for the arrival curve of interest.

Theorem 2.2.1. CONCATENATION THEOREM [75]. *Assume a flow traverses systems S_1 and S_2 in sequence. Assume that S_i offers a service curve of β_i , $i = 1, 2$ to the flow. Then the concatenation of the two systems offers a service curve of $\beta_1 \otimes \beta_2$ to the flow.*

The concept of *vertical deviation* is defined in Definition 2.2.10, which is mandatory for the later definition of a backlog bound.

Definition 2.2.10. VERTICAL DEVIATION [75]. *Let $f, g \in F$ be two w.s.i. functions or sequences. The vertical deviation $v(f, g)$ is defined as*

$$v(f, g) = \sup_{t \geq 0} \{f(t) - g(t)\}. \quad (2.6)$$

Next, the concept of *horizontal deviation* will be defined. This concept is needed for the later definition of a delay bound.

Definition 2.2.11. HORIZONTAL DEVIATION [75]. *Let $f, g \in F$ be two w.s.i. functions or sequences. The horizontal deviation $h(f, g)$ is defined as*

$$h(f, g) = \sup_{t \geq 0} \{\inf\{d \geq 0 \text{ such that } f(t) \leq g(t + d)\}\} \quad (2.7)$$

The backlog bound gives an upper bound of the maximum buffer usage of a server:

Theorem 2.2.2. BACKLOG BOUND [75]. *Assume a flow, constrained by arrival curve α , traverses a system that offers a service curve β . The backlog $R(t) - R^*(t)$ for all t satisfies:*

$$R(t) - R^*(t) \leq \sup_{s \geq 0} \{\alpha(s) - \beta(s)\} \quad (2.8)$$

The delay bound gives an upper bound of the maximum end-to-end delay of a flow in case of FIFO delivery:

Theorem 2.2.3. DELAY BOUND [75]. *Assume a flow, constrained by arrival curve α , traverses a system that offers a service curve of β . The virtual delay $d(t)$ for all t satisfies $d(t) \leq h(\alpha, \beta)$.*

As mentioned in [103], if FIFO cannot be assumed, the bound on the maximum backlogged period under the assumption of a strict service curve can be used as an alternative delay bound instead of the horizontal deviation. Theorem 2.2.4 describes the output bound.

Theorem 2.2.4. OUTPUT BOUND [103]. *Consider a system S that offers a minimum service curve β and a maximum service curve $\bar{\beta}$. Assume a flow R traversing the system has an arrival curve α . Then we obtain the following output bound α^* for R^* :*

$$\alpha^* \leq (\alpha \otimes \bar{\beta}) \circ \beta \quad (2.9)$$

In the remainder of this subsection, we summarize a common approach that deals with non-FIFO aggregate multiplexing, which is called BLIND MULTIPLEXING[75] and can be determined using the Theorem 2.2.5.

Theorem 2.2.5. BLIND MULTIPLEXING[104]. *Consider a node blindly multiplexing two flows 1 and 2. Assume that the node guarantees a strict minimum service curve β and a maximum service $\bar{\beta}$ to the aggregate of the two flows. Assume that flow 2 has α_2 as an arrival curve. Then*

$$\beta_1 = [\beta - \alpha_2]^+ \quad (2.10)$$

is a minimum service curve for flow 1 if $\beta_1 \in \mathcal{F}$. $\bar{\beta}$ remains the maximum service curve also for flow 1 alone.

In BLIND MULTIPLEXING, no knowledge about the multiplexing and scheduling strategies employed in the intermediate network elements is required.

Two basic concepts are available to determine NC performance bounds [68]:

- Node-by-node analysis
- Edge-to-edge analysis

Schmitt et al. [104] propose the following two systematic algorithms that provide node-by-node and edge-to-edge analysis, *Total Flow Analysis* (TFA) and *Separated Flow Analysis* (SFA). The TFA computes the worst-case delay for each server crossed by the flow of

interest by adding them up [16]. The SFA computes a left-over service curve for every server on the path of the flow of interest, computes the convolution of those service curves, and determines the delay by applying the obtained service curve to the flow of interest [16]. Both TFA and SFA were successfully implemented in the DISCO network calculator [28].

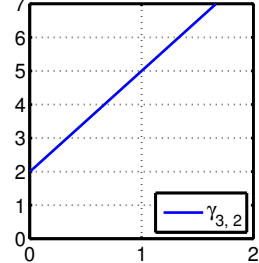
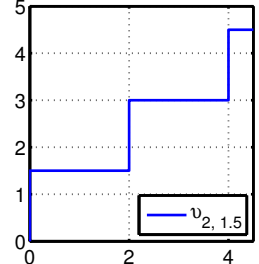
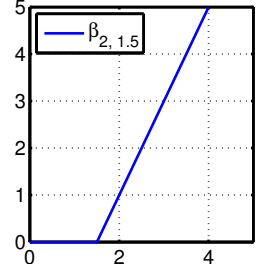
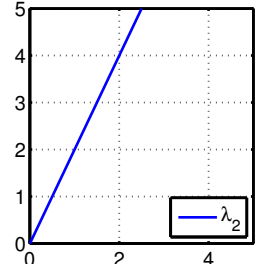
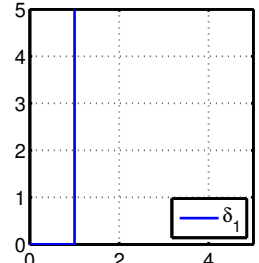
In the field of NC, we know that the following phenomena improve and tighten the derived NC bounds: *Pay Bursts Only Once* (PBOO) and *Pay Multiplexing Only Once* (PMOO). They are addressed by the recently mentioned algorithms. The terminology already indicates the improvements of *Pay Bursts Only Once* and *Pay Multiplexing Only Once*. Imagine a node-by-node analysis, as given by the TFA algorithm. Bursts and multiplexing erroneously are taken into account (“paid”) at every node, rather than just at their first occurrence. This results in looser, i. e., more pessimistic bounds.

These phenomena are addressed by *Separated Flow Analysis* and *Pay Multiplexing Only Once SFA*. The bursts of PBOO are expressed by SFA and the concatenation theorem as introduced by [75]. PMOO is addressed by Schmitt et al. in [103]. They propose *Pay Multiplexing Only Once SFA* (PMOO-SFA), a sophisticated version of SFA that addresses the problem of multiplexing by employing convolution prior to the computation of left-over-service. They also take into account that the order in which the convolution and left-over service curve computation are carried out plays a significant role in the tightness of the bounds.

Table 2.6 shows common arrival and service curves used in NC. The affine arrival curve $\gamma_{r,b}(t)$ is a representator for the generic token bucket model, as introduced in Section 2.2.2. The periodic arrival curve $v_{T,\tau}$ is used for periodic events, such as those generated by regular polling or values sent by sensors. The rate latency service curve $\beta_{R,T}$ bounds the service given by a service element, e. g., an output port. The peak rate service curve $\lambda_R(t)$ gives an upper bound for the service element, e. g., the transmission speed of a given link. The burst delay δ_t is a service curve that adds latency.

Mapping Switched Ethernet to Network Calculus

When addressing end-to-end performance bounds in switched Ethernet, an essential topic is the mapping of the store-and-forward delays of the switches, since fluid flow models do not have a direct counterpart for variable-sized packets. More precisely, we have the following options, known from the standard NC literature (e. g., [75]).

Description	Syntax	Graph
Affine Arrival Curve	$\gamma_{r,b}(t) = \begin{cases} 0 & \text{for } t = 0 \\ r \cdot t + b & \text{for } t > 0 \end{cases}$	
Periodic / Staircase Arrival Curve	$v_{T,\tau}(t) = \lceil (t + \tau) / T \rceil$	
Rate Latency Service Curve	$\beta_{R,T}(t) = R[t - T]^+$	
Peak Rate Service Curve	$\lambda_R(t) = R \cdot t$	
Burst Delay Curve	$\delta_t = \begin{cases} 0 & \text{for } t \leq T \\ \infty & \text{for } t > T \end{cases}$	

Tab. 2.6: Common Curves Used in Network Calculus [75]

- The packet is described as a *discrete burst* (DB) that is already active at time 0. A staircase or (σ, ρ) -constrained arrival curve can be used to model this traffic.
- A *rate latency* (RL) service curve of the form $\beta_{C, l_{max}/C}$ is applied, with C being the capacity of the discussed link and l_{max} the maximum packet size of a frame that is able to delay the flow of interest.
- A *packetizer* (PKT) is introduced that acts as an additional delay element, which releases the whole packet if completely received (cf. [75]). We derive a tightened, stair-case arrival curve that bounds the output.

However, in order to safely apply PKT to an edge-by-edge analysis, having shown more accuracy (cf. [103]), the PKT is usually realized by a RL service curve, a service curve that delays flows by a maximum sized frame and so corresponds to the RL approach. Unfortunately, this additional delay from the maximum sized frame at each server yields more pessimistic bounds than those of the original packetizer developed for the node-by-node analysis. An approach that captures this inaccuracy is given by the *MIP approach* given in Section 4.

2.2.4 Summary

Table 2.7 summarizes the addressed background on performance bounds. We have introduced traffic models that are commonly used in the field of network simulation and NC to characterize traffic arrivals. Two major filtering techniques were identified that are provided by low cost switching ASICs. Additionally, the QoS building blocks *Shaping and policing* and *Queuing and scheduling* present those QoS mechanisms that provide guarantees on static bandwidth and topology dimensioning. As a consequence, those filtering techniques and QoS mechanisms may be candidates for use in the switched Ethernet cabin. At last we have introduced known techniques to determine performance bounds in queuing networks, i.e., Network Simulation, NC, and network measurements.

Topic	Description
Traffic Models	Token Bucket Models On/Off Traffic Models Self-Similar Traffic Models
Addressed Filter Techniques	MAC Addresses VLAN tags
QoS Building Blocks	Shaping and Policing Queuing and Scheduling
Determining Performance Bounds	Discrete Event Simulation Network Calculus Measurements

Tab. 2.7: Performance Bounds — Summary

3. NEXT GENERATION CABIN MANAGEMENT SYSTEM

The network topology we are going to analyze in the remainder of this document is a step towards a COTS enabled CIDS based on switched Ethernet. Throughout this work, we call this novel system *Switched Cabin Management System (SCMS)*. We introduce solutions that provide full IP support in the cabin system, without reducing the predictability and QoS of the DAL-C functions. Section 3.2 addresses the single domain cabin layout that is intended as a replacement to the current CIDS. Section 3.3 introduces the concept of a multi-domain cabin layout. The multi-domain cabin layout is intended to bring several functions from different safety domains to a single network. For the multi-domain cabin, we provide configuration profiles for the A30x, the A350 XWB, and the A380. We finally introduce the VLAN concept, which will be used to segregate the different safety levels and provide QoS guarantees.

3.1 General Cabin Layout and Requirements

Throughout this work we follow an approach with a central server, i.e., safety related traffic only flows from the server to the end devices and vice versa, but never directly between the end devices. This has the advantage of decreasing the multicast delay difference by always forwarding traffic through the server. As a consequence, multicast packets are only multiplied in the downpath.

Recalling the requirements introduced in Table 2.2, Section 2.1, we require the maximum latency of audio playback to be lower than 10 ms. The maximum latency consists of the delay from the end device (likely a handset) to the server, and of the delay from the server to the end device (likely a PSU). The maximum delay from the server to the end device is even tighter (PSU): Only 1 ms of maximum differential delay is allowed in this case. We conclude that the maximum delay for the upstream is 9 ms, and only 1 ms is allowed for the downstream if synchrony cannot be achieved by a common time base.

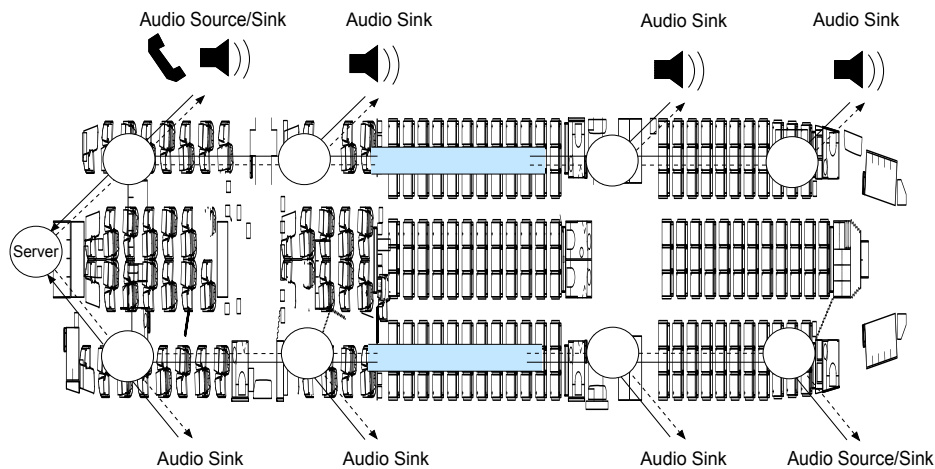


Fig. 3.1: Single Domain Cabin, Courtesy of Airbus

3.2 Single Domain Cabin Layout

A typical topology setup is shown in Figure 3.1, where up to 22 lines are foreseen in the airplane. A maximum topology depth of 16 Ethernet hops is assumed. However, the maximum length is highly dependent on the actual aircraft type and configuration. The later evaluations consider a line consisting of 13 Ethernet switches. This novel cabin layout is based on switched Ethernet and was introduced in [42]. Network simulations showed that the basic functionality can be fulfilled in such a switched environment. This cabin layout covers the domain ACD, and serves the devices PSUs, IBUs, handsets and smoke detectors. This simplified single domain cabin layout is used to show the basic operation of the switched cabin network and does not cover all the devices introduced in Section 2.1.1. The *Flight Attendant Panels* (FAPs) are currently served by the panel network, the CVMS use case is even completely new. This single domain cabin layout will most likely represent a replacement for the original CIDS system, while benefiting from the employment of inexpensive switch ASICs or FPGAs¹ solutions where necessary.

3.3 Multiple-Domain Cabin Layout

In this work, we also present a novel platform that is used to segregate up to four safety domains. The four domains, ACD, AISD, PIESD and PODD, were introduced in

¹ Field Programmable Gate Array

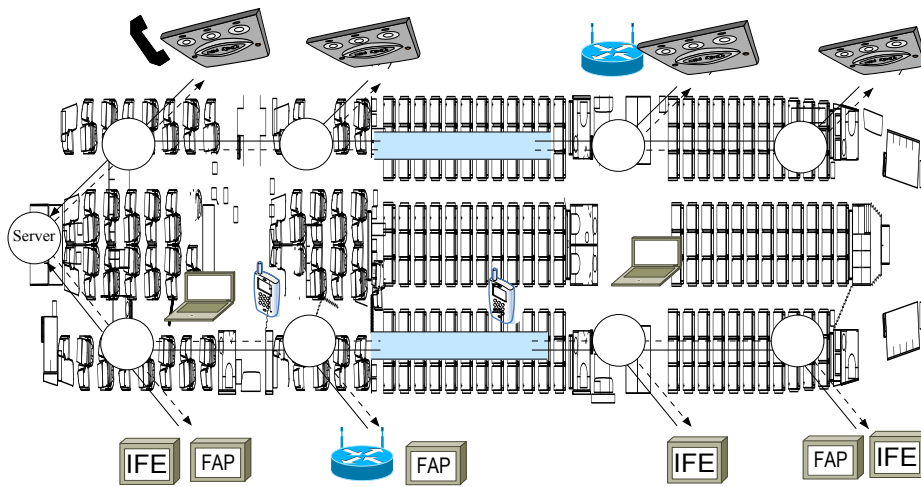


Fig. 3.2: Multi-Domain Cabin, Courtesy of Airbus

Section 2.1.2 and are subject to the segregation. Up to the present, the segregation of domains in the airplane was accomplished by means of a real physical segregation of the different domains, and included the separation of the transmission media and the networks. In terms of saving weight, bringing several domains into one transmission medium is an excellent idea. However, to satisfy the certification process, safety aspects must be addressed, and up to now, it has not seemed that there existed standard components that fulfilled all the desired requirements, be it due to the limited number of VLAN tags that are provided, or due to the fact that ingress traffic regulation is not offered by the switch. Another point is that COTS Ethernet switches have several hundred functions, which will not necessarily be used in an aircraft cabin. However, with respect to the certification process, any function provided by the switch would have to be part of the safety analysis.

The proposed strategy is to take note of the QoS techniques provided by the switch manufacturer and the QoS community, be it sophisticated scheduling algorithms, precise time synchronization with hardware support, or traffic regulation mechanisms. These techniques will be implemented on the NetFPGA platform [88], which arose from the OpenFlow movement and only provides those functions that are strictly necessary for segregation. This platform is based on a Virtex-5 FPGA and is evaluated in Section 6.2.2.

Figure 3.2 shows the topology that covers the four safety domains. The domains share the same network without degrading safety levels.

Device	Packet Size	Bandwidth [kbit/s]
Cabin server	108	27 648
Standard device	108	204
Handset (normal latency)	142	568
Handset (low latency)	64	1632
Camera	$\geq 64, \leq 1518$	≤ 15000
Other	$\geq 64, \leq 1518$	up to wirespeed

Tab. 3.1: Traffic Profiles in the Aircraft Cabin

3.4 Traffic Characteristics

In this section, we summarize the traffic profiles in the cabin system. We introduce a coarse classification of the devices in the cabin system. The standard devices collect data and control cabin functions. Standard devices generate UDP traffic having a packet size of 108 bytes. Throughout this work we confine ourselves to the standard devices PSU, IBU, and smoke sensor. The handsets are used for the PA use case as well as cabin interphone. They generate UDP traffic with a packet size of 142 bytes, respectively, 64 bytes, for the low latency variant. The cameras are used for the CVMS use case and generate arbitrary UDP traffic with no restrictions in terms of packet size. The remaining device type are most likely to generate best effort traffic, i.e., we do not constrain these devices in the way they generate traffic, so that UDP is allowed as well as TCP with no requirement concerning packet size. The traffic profiles of standard devices, handsets and cameras are derived from the original CIDS protocol, given by courtesy of Airbus. Table 3.1 summarizes the assumed traffic profiles.

3.5 Configuration Profile

In the remainder of this chapter, we introduce four different cabin layouts. The first cabin layout addresses a simplified single domain layout as a replacement for the current CIDS bus and solely covers selected cabin functions from the ACD domain. In addition to the single domain layout, we also introduce multi-domain layouts. We target the A30x and A350 programs as well as the A380. The provided configurations are exemplary. The actual end-configuration will likely differ and be highly dependent on the requirements and configuration desires of the airline.

3.5.1 Single Domain Layout

The single domain layout is intended for performance studies and represents a replacement of the current bus solution. This layout consists of up to 13 switches in each line which serves up to 91 PSUs and 13 handsets. Table 3.2 shows the configuration parameters for this layout.

Variable	Value	Domain
DEU per line	13	
PSU per DEU	7	ACD
Handsets per DEU	1	ACD
Handsets per line	13	ACD

Tab. 3.2: Single Domain Configuration

3.5.2 Multiple-Domain Layout

The multi-domain cabin brings together several safety domains into one unified network. This strategy brings weight savings and helps reduce kerosene consumption. We distinguish the following, current, Airbus programs: A30x, A350 XWB, and A380.

The A30x is a current project targeting a consequent further development of the A320 family. The A320 family is the single-aisle jet plane family of Airbus [3]. This airplane family covers short-range and medium-range flights, i.e., up to 5950 km. The maximum number of passengers varies between the A318, A319, A320, and A321. The A318 can transport up to 132 passengers and the A321 has a maximum capacity of 220 passengers [6, 7]. The length of those airplanes ranges from 31.4 m for the A318 to 44.5 m for the A321.

The A350 is a long-range airplane family and is currently under development under the project name A350 XWB. The A350 family provides true long-range capability with seating capacities from 250 to 400-plus passengers [4]. Compared to the existing families A330 and A340, the A350 is a twin-engine airplane. A major advantage is that a high percentage of the fuselage is made of carbon fibre reinforced plastic, which reduces overall weight and results in lower operating costs.

Variable	A30x	A350	A380	Domain
DEU per line	8	15	12	
Cameras per line	4	4	4	ACD
FAP per line	4	4	4	ACD
FAP per line	2	2	2	PODD
FAP per line	2	2	2	AISD
PSU per DEU	8	8	8	ACD
IBU per DEU	0	0	8	ACD
Handsets per DEU	2	1	1	ACD
Handsets per line	4	12	12	ACD
Wireless Sensor AP	1	1	1	AISD
Crew WLAN AP	1	1	1	AISD
Passenger WLAN AP	1	1	1	PIESD

Tab. 3.3: Multiple-Domain Configuration

The A380 is the current flagship of Airbus and also a long-range airplane. The A380 is the world's largest commercial aircraft with capacity ranging from 525 to 853 passengers depending on the actual configuration [5]. It has a range of up to 15 700 km [8]. The A380 has a length of 72.7 m and a wingspan of 79.8 m.

All these different properties have a direct impact on the different cabin configurations. The assumed configuration parameters for the multi-domain cabin layouts are given in Table 3.3, and were specified by Airbus.

3.6 VLAN Concept

In order to guarantee the forwarding through the network on a predefined trajectory, we use a VLAN concept which is introduced in this section. A first version of this VLAN concept was given in [37].

The entries in the switches consist of the information given in Figure 2.8, Section 2.2.2. The 12-bit wide VLAN id specifies the VLAN that is described in the table entry. The bits p_{n-1} to p_0 specify whether a port is in the VLAN or not, i. e., 1 if the port participates in the VLAN, 0 if not. The VLAN rules are deployed in the switches with management protocols like the SNMP² or directly written to the switch's EEPROM³.

² Simple Network Management Protocol

³ Electrically Erasable Programmable Read-Only Memory

The VLAN also holds a *3-bit wide priority field*, which can be applied by certain QoS techniques in the switches. Since we employ the information provided from Layer 2, we have the opportunity to use low cost switching ASICs. In Section 6.2.2 we later address the suitability of low cost switching ASICs in the SCMS.

3.6.1 Dedicated VLANs

The dedicated VLAN concept assigns one VLAN id to each end device in the line. Besides those unique device addresses in the line, we introduce multicast VLANs, which allow addressing several end devices in groups, e.g., all PSUs or all PSUs of a specific class (e.g., business class or economy class). Since some devices occur more often in the aircraft cabin than others, we use Huffman encoding to partition the ranges appropriately [37]. As mentioned in Section 2.2, some of those VLANs are reserved and are thus excluded from the following assignments.

Table 3.4 shows the VLAN ranges for the Airbus A30x, Table 3.5 summarizes the VLAN ranges for the Airbus A350, and Table 3.6 shows the VLAN ranges for the Airbus A380.

3.6.2 Class Based VLANs

The address space of the VLAN id is limited and only allows up to 4096 different values. Three of those are reserved by the standard, so that only 4093 remain for actual assignment. When considering the number of end devices according to Table 2.4, Section 2.1, one can easily see that the number of VLANs will reach its limits in the near future. This problem might be circumvented with the following approach: Class based

VLAN Range	Device Type	Number of Devices / Multicast Groups
2–2047	IBU	1518
2048–3071	PSU	96
3072–3327	Multicast PSU	20
3328–3455	Handset	8
3456–3583	FAP	4
3584–4094	CVMS	26

Tab. 3.4: A30x VLAN Assignment in Each Line

VLAN Range	Device Type	Number of Devices / Multicast Groups
2–2047	IBU	1518
2048–3071	PSU	96
3072–3199	Handset	12
3200–3327	FAP	8
3328–3583	Multicast PSU	20
3584–4094	CVMS	26

Tab. 3.5: A350 VLAN Assignment in Each Line

VLAN Range	Device Type	Number of Devices / Multicast Groups
2–2047	IBU	1518
2048–3071	PSU	96
3072–3328	FAP	30
3328–3583	CVMS	26
3584–3839	Multicast PSU	20
3840–4094	Handset	12

Tab. 3.6: A380 VLAN Assignment in Each Line

VLANs are introduced, which address the device type rather than every single end device. Table 3.7 shows the assignment of VLAN ranges.

With the class based VLAN concept, end devices are not directly addressable on layer 2 without knowing the actual MAC address, so that common address resolution techniques like ARP or NDP⁴ are necessary.

VLAN	Device Type	Number of Multicast Groups
3	IBU	1
4	PSU	1
5	FAP	1
6	CVMS	1
7–26	Multicast PSU	20
27	Handset	1

Tab. 3.7: Class Based VLAN Assignment

⁴ Neighbor Discovery Protocol

3.7 Summary

In this section, we have proposed a novel aircraft cabin architecture, SCMS, based on switched Ethernet. We have introduced the single domain cabin layout as a replacement for the classical CIDS system. The multi-domain cabin layouts are a step towards placing several safety domains on a single network. With the multiple domain approach, we can profit from weight savings and the reduced complexity from reducing the number of required networks. The domains are segregated in the switches by VLAN rules while traffic is served by SPQ.

4. AN ALGORITHM FOR DETERMINING WORST CASES IN PACKET-SWITCHED NETWORKS

Determining the *exact worst case in queuing networks* is known to be difficult and often a hurdle. Bouillard et al. [16] even showed that determining the exact worst case in feed-forward networks is NP-hard. The given approach based on MIP is able to determine the worst case in packetized networks. The presented packetized traffic model describes the worst case schedules in a single MIP instance, and thus benefits from the advantages of both an edge-by-edge analysis and a packetized model, but at the cost of some additional effort. The exhaustive enumeration of schedules is similar to techniques known from model checking and allows the calculation of worst case bounds. The basic idea is to model the case where one packet is delayed by another packet with a Boolean (integer) variable. If this variable is true, i.e., one, the worst case delay is increased by the transmission delay of the previous packet, otherwise it is increased by zero. Compared to the techniques from NC, this model allows better bounds at the cost of requiring additional computational effort, since we approximate an NP-hard problem.

4.1 Determining the Exact Worst Case

In the following section, we explain some of the *reasons why employing the state-of-the-art approach NC* in packetized networks produces *pessimistic results*. Fluid flow models are widely used in the field of NC, which means that one bit after the other is forwarded, instead of packets. This has the drawback that *packetization cannot be modeled exactly*, and simplifications have to be employed. The standard approaches use *discrete bursts (DB)*, *rate latency (RL) service curves*, or *packetizers* to map the packetization to a fluid model. These techniques are given in Section 2.2.3.

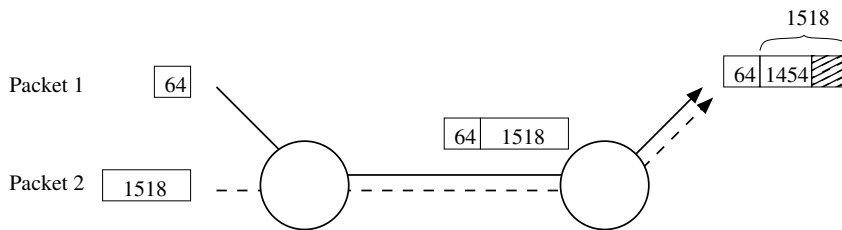


Fig. 4.1: Small packet follows larger packet

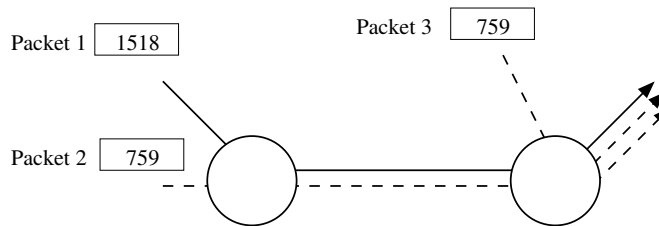


Fig. 4.2: Effectual Burst

The DB- and RL-approaches cannot cover the following situations accurately: If a small packet follows a larger packet over several nodes, the impact of the burst might already be partially spent. Figure 4.1 shows this case: a small 64 byte packet is delayed by a maximum sized packet with length 1518. It turns out that the small packet is delayed at the second server by a smaller portion, i.e., the time it takes to transmit the succeeding 1454 bytes.

Pessimism also occurs when interfering flows join at a second server. Consider the second example shown in Figure 4.2: Packet 1 belongs to the flow of interest. To construct the worst case, Packet 2 delays Packet 1 at the first server, and Packet 3 delays Packet 2, which then can delay Packet 1 at the second server. As a consequence, Packet 1 is delayed by at most one Packet 2 and one Packet 3 (or two times by Packet 2). The standard NC approach would calculate an additional delay of three packet bursts, since bursts are active at any time of the busy period.

The *packetization concept* has been given in [75], and could cover the recently mentioned special cases. However, when inserting additional delay elements in an edge-by-edge analysis, we observe the following pitfall: In order to determine the exact worst case in switched Ethernet, the packetizer has to know the packetization sequence a priori. Especially when the links run at different wire speeds, the analytical methods presented cannot model the exact worst case. This observation is similar to the pitfall of PMOO-SFA, as given in [105] and also outlined in [71]. Due to the commutativity of

(min,+)-convolution, the burstiness of the traffic is always paid for at the rate of the slowest server.

Consider the following example: Three packets (size 125, 250 and 500 bytes) are transmitted over the two sequential servers $S1$ and $S2$, where $S2$ has a faster service rate than server $S1$. Furthermore, we are interested in the worst case latency of the 125 byte packet. The results of the cumulative arrivals are given in Figure 4.3 and Figure 4.4.

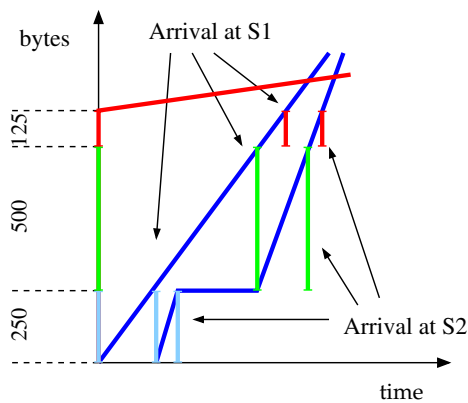


Fig. 4.3: 250 byte packet prior to 500 byte packet

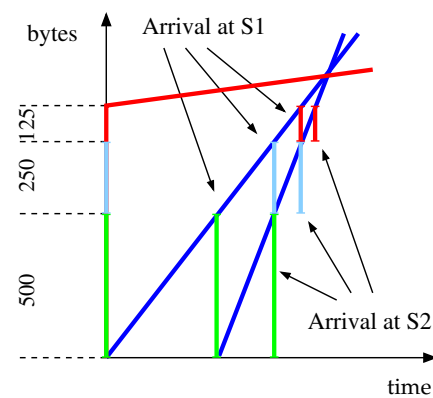


Fig. 4.4: 500 byte packet prior to 250 byte packet

We can see that the packet of interest is 125 bytes and is delayed by one 500 byte packet and one 250 byte packet. Depending on whether the 250 byte packet or the 500 byte packet is transmitted first, we observe different worst cases with the packetization approach. Since any packet sequence is conceivable, the highest worst case latency of all sequences has to be taken into account.

4.2 Related Work

In this section, we summarize recent work that is closely related to the task of determining *tighter bounds* in queuing networks. There are *three major approaches* within the field of determining deterministic performance bounds in packet switched queuing networks: (a) the trajectory approach, (b) model checking with timed automaton, and (c) NC. While the model checking approaches are *exact per definition* but slow, the NC and the trajectory approach have been extended to achieve tighter bounds.

The NC was used in the field of AFDX networks for the certification of packet switched networks. As stated in Section 4.1, NC is usually not able to express the serialization of packet bursts. For this, Frances et al. introduced a technique that improved AFDX bounds significantly ([38]). Bauer et al. [13] call this approach the *grouping technique*.

The trajectory approach was introduced by Martin et al. [79] and further developed by Bauer et al. [13], with a special focus on AFDX networks. The trajectory approach considers a packet from a flow and identifies the busy period of all visited nodes by that packet (cf. [13]). Adding those values gives a first worst-case bound for the flow of interest. The trajectory approach was optimized by the introduction of grouping in [13]. Similar to the grouping technique introduced in the context of AFDX, this extension can cover the following scenario: Flows that share a common link are serialized and cannot arrive at the same time on a switch [13].

The grouping technique or serialization approach is similar to using the NC packer shown in the last section. Common NC approaches use $(\min,+)$ -deconvolution to determine the output curve. However, the delay elements have to be used carefully. When links run at different wire speeds there is the possibility of generating a situation as in Section 4.1. Compared to the NC approaches, the trajectory approach has the advantage that it models packet transmission at wire speed. To do this with NC, one has to use discrete bursts as provided by the staircase arrival curve rather than using an affine arrival curve.

The model checking approaches that determine worst case bounds in packet switched networks are based on timed automata (as used in [21] and [119]). These works use the *model checker UPPAAL* [117]. Charara et al. [21] use a reachability analysis of the states in the system. This approach models the send signals by actions, and only allows changing the state if a condition holds (guard). The signals model the sending of packets, the guard is used to wait until the transmission of the frame has been finished. The methods in [21] verify that a frame is received before a global, arbitrarily chosen transmission delay. The global transmission delay of the frame must then be lower than a given bounded delay (cf. [21]). According to Adnan et al. [1], the latter model checking approach can handle up to ten different flows in the addressed AFDX configuration. They furthermore provide a method to boost the number of flows that can be handled by model checking up to 20 by reducing the search space.

Further literature provides the following insights on tight bounds: Schmitt et al. [103] show that NC is not always able to determine the exact worst case. The NC opera-

tors are not powerful enough to cover all essential cases, so that overestimation has to be taken into account. They furthermore provide an algorithm based on *Linear Programming* (LP) that is able to determine the exact worst case. However this approach requires the enumeration of all potential scheduling possibilities in order to determine the one with the highest impact. In fact, Bouillard et al. [16] showed that, for general networks, this problem is NP-hard.

4.3 Mixed Integer Programming

The novel algorithm to determine the worst-case in a packet switched queuing network is based on *Mixed Integer Programming* (MIP). MIP is a special case of *Linear Programming* (LP), and may require that some or all variables are integral. An LP consists of inequalities and an optimization function that shall be maximized or minimized:

Consider the inequalities 4.1–4.4 and the maximization of the function 4.5:

$$2 \cdot x + y \leq 7 \quad (4.1)$$

$$0.5 \cdot x + y \leq 5 \quad (4.2)$$

$$x \geq 0 \quad (4.3)$$

$$y \geq 0 \quad (4.4)$$

$$5 \cdot x + 3 \cdot y \quad (4.5)$$

Figure 4.5 gives a graphical representation of this LP. The LP solution of this example is $19.6\bar{6}$ where the solution vector is $(x, y) = (1.3\bar{3}, 4.3\bar{3})$. The solution of the MIP (where x and y must be in \mathbb{Z}) is 19, and the solution vector is $(x, y) = (2, 3)$.

When some of the given variables are enforced to be in \mathbb{Z} rather than in \mathbb{R} , the LP turns into a MIP. This difference turns the problem of finding the optimal value into an NP-hard problem. NP-hard problems are difficult to solve and may require an enumeration of an exponential number of solutions in order to determine the best solution.

MIPs are commonly solved by cut-and-branch techniques [87]. The solver thereby starts from the so called LP relaxation, which is the LP instance that is constructed from a MIP instance by omitting the property of integrality. There are efficient algorithms to determine the solution to an LP instance, e.g., the simplex algorithm or the Fourier Motzkin algorithm as shown in [106]. The principle for determining the inte-

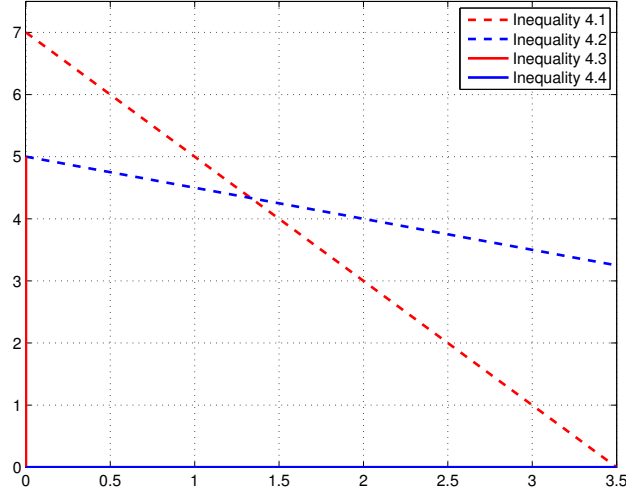


Fig. 4.5: Graphical Representation

ger solution with cut-and-branch is as follows [87]: We successively determine lower and upper bounds of the optimal solution. The lower bound is thereby determined by branching variables in order to scan the search space systematically. The upper bound is determined by starting from the solution of the LP relaxation, and then adding several cutting planes to the inequalities that do not change the solution vector of the MIP.

We now introduce the concept of conditional constraints, which is later used in our novel approach to determine worst cases. A *conditional constraint* is similar to an *if* statement, where the condition in the *then* statement only holds if the *if* condition holds. These conditional constraints are turned into unconditional constraints using an approach which is similar to a technique of [15]. Consider the conditional constraint 4.6:

$$\begin{aligned} &\text{if } a > b \text{ is satisfied} \\ &\text{then } c \leq d \text{ must also be satisfied} \end{aligned} \quad (4.6)$$

As shown in condition 4.7 and condition 4.8, this can be modeled by introducing a binary variable $y \in \{0, 1\}$ and sufficiently large upper bounds M_1 and M_2 (cf. [15]). If $a > b$, then y must be zero due to condition 4.7. But if y is zero, then $c \leq d$ must hold, so that condition 4.8 then holds. Otherwise, if $a \leq b$, then y can also be one, and so, does not enforce $c \leq d$.

$$a \leq b + (1 - y) \cdot M_1 \quad (4.7)$$

$$c \leq d + y \cdot M_2 \quad (4.8)$$

4.4 Optimization Based Approach

In this section, we propose an algorithm that *determines the worst case in a switched network* by translating the general arrival curves of a feed forward topology to MIPs. The global worst case can be determined at once by solving the MIP. A preliminary version of this algorithm was already used in our previous work to determine the worst case execution points in network simulations [41].

This section is structured as follows: First, we introduce the syntax as used in the latter MIP instance. Then we give a simple example showing the basic idea to determine the worst case using MIP. After the basic mechanisms, we give the algorithm to derive the MIP instance from a topology description.

The following variables, indices, constants, and functions are used in the algorithm that derives the MIP instance:

- src, dst : start and end vertex in network $G = (V, E)$
- $i, j \in V$: vertices in network $G = (V, E)$
- $f, g \in \mathcal{F}$: flow f in set of flows \mathcal{F}
- $p, q \in P$: packets p and q of packets P
- $s_{p,i,j}$: start of transmission of packet p from node i to j
- $r_{p,j}$: receive packet p in node j
- $d_{p,i,j}$: queuing delay of packet p between node i and j
- $x_{p,q}$: 1 if packet p queued after packet q , 0 otherwise
- $y_{p,q,i,j}$: burst for which too much was paid if packet p and packet q overlap
- L_p : packet size of packet p
- $w_{i,j}$: wire speed of link between i and j , where $w_{i,j} > 0$
- τ : sum of Ethernet preamble and interframe gap
- κ : processing delay
- $succ(f, src)$: succeeding node of node src in flow f

Besides the network graph as given by $G = (V, E)$, we also require the flow specification \mathcal{F} to construct the MIP. The flows of interest are subject to the optimization problem.

The objective rule 4.9 and the constraint rules 4.10–4.13 are used to build the MIP that allows determining the worst case queuing latency. Algorithm 1 creates the actual MIP from the network graph G and the set of flows \mathcal{F} . The constraints that are built in accordance to constraint rule 4.13 represent the key constraints that capture all possible queuing orders—turning the linear optimization problem into an NP-hard MIP.

Maximize

$$r_{p,dst} - s_{p,src,succ(f,src)} \quad (4.9)$$

Subject to

$$s_{p,i,j} + (L_p + \tau) \cdot w_{i,j} = r_{p,j} \quad (4.10)$$

$$s_{p,i,j} = r_{p,i} + \kappa + d_{p,i,j} \quad (4.11)$$

$$d_{p,i,j} = \sum_{p \neq q} x_{p,q} \cdot (((L_q + \tau) \cdot w_{i,j}) + \kappa) - \sum_{q \in P - \{p\}} y_{p,q,i,j} \quad (4.12)$$

$$x_{p,q} + x_{q,p} \leq 1 \quad (4.13)$$

In fact, the flows in the basic integer program already interfere with other flows, leading to a maximum delay in the objective, i.e., the addressed packet is delayed by all other packets at each hop (expressed in the lines 8 and 9). However, this maximum delay is still virtual, since the determined solutions do not represent a work conserving forwarding, i.e., it is not yet assured that the packets are forwarded at once (links can be idle although backlog is available). A work conserving forwarding is achieved with the conditional constraints in lines 10–14. But we have not yet guaranteed FIFO forwarding, i.e., it is not yet assured that the packets received prior to other packets are forwarded first. This may be guaranteed by the optional conditional constraint given in lines 15–17. The given conditional and either-or constraints are turned into unconditional constraints using the techniques summarized in Section 4.3.

Alg. 1 Mixed Integer Program Creation

```

1: for all  $f \in \mathcal{F}$  do
2:   for all  $p \in f$  do
3:     for all segments  $s$  of paths that packet  $p$  traverses do
4:        $i = s.source$ 
5:        $j = s.destination$ 
6:        $s_{p,i,j} + (L_p + \tau) \cdot w_{i,j} = r_{p,j}$ 
7:        $s_{p,i,j} = r_{p,i} + \kappa + d_{p,i,j}$ 
8:        $d_{p,i,j} = \sum_{q \in P - \{p\}} x_{p,q} \cdot (L_q \cdot w_{i,j} + \kappa) -$ 
9:          $\sum_{q \in P - \{p\}} y_{p,q,i,j}$ 
10:      if  $L_q > L_p$  then
11:        if  $r_{p,i} > s_{q,i,j}$  and  $r_{p,i} < r_{q,j}$  then  $y_{p,q,i,j} \geq (L_q \cdot w_{i,j}) + \tau + r_{p,i} - r_{q,j}$ 
12:      else
13:        if  $r_{p,i} \geq s_{q,i,j} + (L_q \cdot w_{i,j}) + \tau$  then  $y_{p,q,i,j} \geq (L_q \cdot w_{i,j}) + \tau$ 
14:      end if
15:      if fifo then
16:        if  $r_{p,i} > r_{q,i}$  then  $x_{p,q} \leq 0$ 
17:      end if
18:      for all  $q \in P - \{p\}$  do
19:         $x_{p,q} + x_{q,p} \leq 1$ 
20:      end for
21:    end for
22:  end for
23: end for

```

Exemplary: Tandem Scenario We now consider a tandem scenario with four traversing flows in Figure 4.6. This scenario is similar to the one discussed in [103] and was examined in [41] to determine the worst case execution points in network simulations. Table 4.1 shows the token bucket parameters employed.

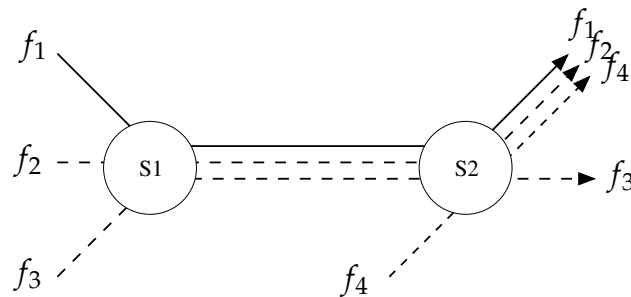


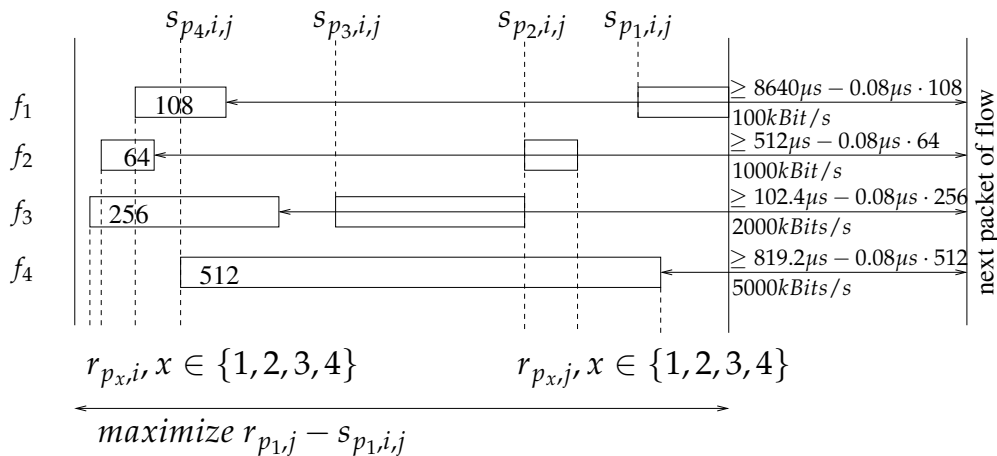
Fig. 4.6: Tandem Scenario with Four Flows

Flow	Burst [bytes]	Rate [kBits/s]
f_1	108	100
f_2	64	1000
f_3	256	2000
f_4	512	5000

Tab. 4.1: Token Bucket Parameters, Four Flows

Figure 4.7 illustrates the queuing situation in the switch. If the investigated flow is f_1 , the packet p_1 of that flow can be delayed by packets p_2 and p_3 at server $S1$ and by packet p_4 at server $S2$. When we are interested in Flow f_2 , the packet p_2 of that flow can be delayed by packets p_1 and p_3 at server $S1$ and by packet p_4 as well as the remainder of packet p_1 (the length of p_1 minus the length of p_2) at server $S2$. This can be expressed by a MIP instance created in accordance with Algorithm 1. Table 4.2 shows the results of the MIP and gives the send and receive time (given in bytes) that enforce the worst case queuing situation.

To solve MIPs one usually applies a cut-and-branch approach as introduced in Section 4.3. Expressed intuitively, we start from the LP relaxation, and successively tighten the upper bound. This upper bound is the safe upper bound for the worst case, i.e., if an exact solution can not be determined in a reasonable amount of time, we can stop the algorithm. When terminating the algorithm before the upper and lower bounds are equal, we have at least a range between the lower and upper bounds in which the worst case is located.

Fig. 4.7: Find Worst Case Between Node i and j for Packet p_1 , 100 Mbit/s

Variable	$p = 1$	$p = 2$	$p = 3$	$p = 4$
$s_{p,src,1}$	148	192	0	
$r_{p,1}$	256	256	256	
$s_{p,1,2}$	576	256	320	172
$r_{p,2}$	684	320	576	684
$s_{p,2,dst}$	1196	320		684
$r_{p,dst}$	1304	384		1196

Worst Case of f_1 , FIFO Assumption

Tab. 4.2: Worst Case Send/Receive Times for Tandem Network (Fig. 4.6)

For a reasonable application of this method in order to find tight bounds, we require that the MIPs be solved in a reasonable amount of time. Perhaps the most important technique to do so is to lower the state explosion to an acceptable level. We now give the major approaches to lowering the search space.

Pre-assignment of Boolean Variables A helpful technique is to assign fixed values to the Boolean variables $x_{p,q}$, so that the packet order is slightly predetermined. More precisely, we cover the following case: If $L_p \geq L_q$ and p does not belong to the flow of interest, we can safely set $x_{p,q} = 1$ (and so $x_{q,p} = 0$).

Restricting the number of packets Furthermore, we should restrict the number of packets of each flow that are analyzed to give a valid bound. To keep the state explosion at an acceptable level, we should not examine an arbitrary number of packets, but only those that are significant for the worst case. We distinguish the following type of bounds:

- FIFO bound — If we are interested in the FIFO bound, it is sufficient to examine one single packet of each flow to determine the exact worst case. In case of FIFO, packets cannot overtake each other, and the earlier received packets are forwarded first.
- Non-FIFO bound — If we are interested in the Non-FIFO bound, it is more difficult to determine the number of packets that are to be examined to calculate the exact bound. One option could be to apply the busy period derived from the NC analysis, and so determine the number of packets. Alternatively we can succes-

sively increase the number of packets of each flow until the results do not change anymore.

Restricting the number of packets and the pre-assignment of Boolean variables is used later, when determining the worst latencies in the aircraft cabin. In examples with over one hundred flows, lowering the states of the MIP is significant in order to obtain at least significant upper bounds.

4.5 Evaluation of Different Worst Case Calculation Techniques

We address the performance of the novel optimization based approach by comparing it to the state-of-the-art approaches introduced in Section 2.2.3. We calculated the NC results using the DISCO network calculator [28] and the *Real-Time Calculus* (RTC) suite [115]. Our algorithm was implemented in C++. The generated MIPs were solved by the standard MIP/LP toolkits ZIMPL [72] and CBC [24].

Tandem Scenario and Feed-Forward Network

We evaluate the different approaches and apply them to the tandem network shown in Figure 4.8 and to the feed-forward network shown in Figure 4.9. For the two introductory examples, we employ the traffic patterns and service curves shown in Table 4.3. Note that we set the processing delay, minimal IFG, and the preamble to zero. For an employment in a realistic scenario, the processing delay should be around $0.05 \mu\text{s}$, and the sum of the minimal IFG and the preamble should be 20 bytes.

Flow	Burst [bytes]	Rate [kbit/s]
f_1	1518	1500
f_2	64	1000
f_3	108	100
Server	Rate Latency [μs]	Rate [kbit/s]
s_1 to s_6	121.44 (RL) / 0.00 (DB)	100000

Tab. 4.3: Token Bucket Parameters for Tandem and Feed-forward Network

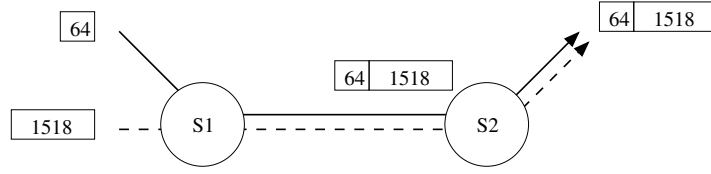


Fig. 4.8: Tandem Scenario with Two Flows

Variable	$p = 1$	$p = 2$	Variable	$p = 1$	$p = 2$
$s_{p,src,1}$	0	1454	$s_{p,src,1}$	0	1454
$r_{p,1}$	1518	1518	$r_{p,1}$	1518	1518
$s_{p,1,2}$	1582	1518	$s_{p,1,2}$	1518	3036
$r_{p,2}$	3100	1582	$r_{p,2}$	3036	3100
$s_{p,2,dst}$	3100	1582	$s_{p,2,dst}$	3036	4554
$r_{p,dst}$	4618	1646	$r_{p,dst}$	4554	4618
Worst Case of f_1			Worst Case of f_2		

Tab. 4.4: Worst Case Send/Receive Times for Tandem Network (Fig. 4.8)

Tandem Scenario In this part, we address the worst case for the tandem scenario given in Figure 4.8. We employ both approaches, i.e., the rate latency method and the discrete burst method. For the NC techniques we give the calculation instructions for TFA, SFA, and PMOO-SFA. The results are summarized in Table 4.5. While TFA applies a node-by-node analysis (so that we have to take the sum of the delays at S1 and S2), both SFA and PMOO-SFA apply an edge-by-edge analysis. The value for PMOO-SFA/DB is not correct for a switched queuing network, since it does not preserve non-preemptive packet bursts. The *MIP approach* delivers worst case send/receive times and we list them in Table 4.4. The values are given in bytes. Multiplying those values by the time it takes to transmit a single byte ($0.08 \mu\text{s}$ for FE or $0.008 \mu\text{s}$ for GbE) gives the actual delay. We conclude that the DB-Approach is superior to the RL-Approach as it considers variable packet sizes. Furthermore our *MIP approach* delivers tighter bounds, but at the cost of additional calculational complexity. In this example, the benefit was about 2% compared to the NC approaches. An illustration is given in Appendix B, Table B.2 and Table B.1.

Approach	Calculation
TFA	$f = f_1 + f_2$ $g = s_1, h = s_2$ $v(f, g), v(f, h)$
RL	$v(f, g) + v(f, h) = 0.2566\text{ms}$
DB	$v(f, g) + v(f, h) = 0.2531\text{ms}$
SFA	$f = f_2$ $g = [s_1 - f_1]^+ \otimes [s_2 - f_1 \odot s_1]^+$
RL	$v(f, g) = 0.4914\text{ms}$
DB	$v(f, g) = 0.2518\text{ms}^1$
PMOO-SFA	$f = f_2, g = [s_1 \otimes s_2 - f_1]^+$
RL	$v(f, g) = 0.3681\text{ms}$
DB	$v(f, g) = 0.1285\text{ms}^2$
MIP	$r_{p_2, dst} - s_{p_2, src, succ(src)}$
DB	0.2480ms

Tab. 4.5: Results for Tandem Scenario

Feed-forward Network In a feed-forward network, flows can take different paths to the same destination or to a later sub-path. This is difficult to handle in the NC, since rejoining flows do not necessarily delay the flow of interest once again, or perhaps only partially. Consider the scenario in Figure 4.9. In fact, f_3 can delay f_2 either at server $S3$ or at server $S8$, but not at both. This is a crucial problem, and as a matter of fact, Bouillard et al. [16] have shown the NP-hardness of determining the exact worst case in feed-forward networks for fluid flow models. This means that when flows have to burst in order to have maximum impact on the delayed flow cannot be easily determined. We observe similar difficulties in the packetized feed-forward network. It cannot be easily determined whether the packets of flow f_2 might only be delayed once by packets from flow f_3 .

The optimization based approach determines the exact worst case of this scenario. The worst case send/receive times are given in Table 4.6. The values are given in bytes.

¹ The DB-PMOO-SFA approach does not obtain correct values in this scenario since packet bursts are not preserved.

² In the DB-SFA approach, we have to add an additional transmission delay, which is lost due to the concatenation of servers.

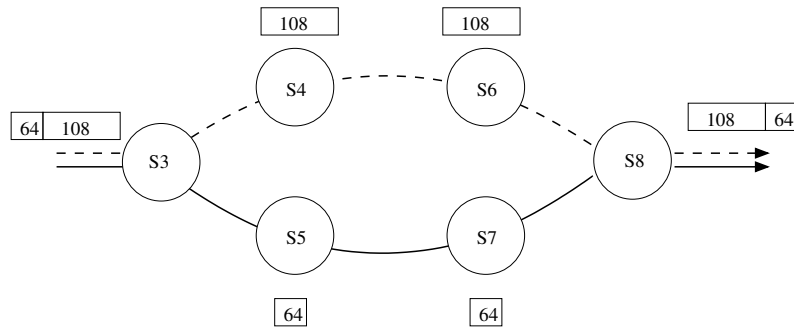


Fig. 4.9: Feed-forward Network with Two Flows

Variable	$p = 2, i = 5, j = 7$	$p = 3, i = 4, j = 6$
$s_{p,src,3}$	504	460
$r_{p,3}$	568	568
$s_{p,3,i}$	676	568
$r_{p,i}$	740	676
$s_{p,i,j}$	740	676
$r_{p,j}$	804	784
$s_{p,j,8}$	804	784
$r_{p,8}$	868	892
$s_{p,8,dst}$	868	932
$r_{p,dst}$	932	1040

Worst Case of f_2

Tab. 4.6: Worst Case Send/Receive Times for Feed-forward Network (Fig. 4.9)

Table 4.7 compares the results for this feed-forward network with the state-of-the-art approaches TFA, SFA, and PMOO-SFA. Both DB- and RL-models are considered. An illustration is given in Appendix B, Table B.4 and Table B.3.

Approach	Calculation
TFA	$a_3 = a_8 = f_2 + f_3, a_5 = a_7 = f_2$ $g_3 = s_3, g_5 = s_5, g_7 = s_7, g_8 = s_8$ $v(a_3, g_3), v(a_5, g_5), v(a_7, g_7), v(a_8, g_8)$
RL	$v = 0.4961\text{ms}$
DB	$v = 0.0378\text{ms}$
SFA	f_2 $g = [s_3 - f_3]^+ \otimes s_5 \otimes s_7 \otimes [s_8 - (f_3 \otimes s_4 \otimes s_6)]^+$
RL	$v(f_2, g) = 0.5124\text{ms}$
DB	$v(f_2, g) = 0.0278\text{ms}^3$
PMOO-SFA	$f_2, g = [s_3 \otimes s_5 \otimes s_7 \otimes s_8 - f_1]^+$
RL	$v(f_2, g) = 0.5019\text{ms}$
DB	$v(f_2, g) = 0.0284\text{ms}^4$
MIP	$r_{p_2, dst} - s_{p_2, src, succ(src)}$
DB	0.02912ms

Tab. 4.7: Results for Feed-forward Network

Switched Aircraft Cabin System

This section compares the results of different NC techniques applied to the Fast Ethernet cabin scenario shown in Figure 3.1, Section 3.2. We give the token bucket parameters of the high priority traffic in Table 4.8. Figure 4.10 shows the results for the upstream, i.e., the delay of packets that flow from an end device to the server. We applied TFA, SFA and MIP and provided discrete models as well as rate latency models. The results are summarized as follows: The discrete burst model using TFA delivers the best realistic results in the discussed scenario and maps with the LP relaxation. However due to state explosion, it was hardly possible to analyze the full network with 104 flows using MIP. Figure 4.11 shows the results for a cabin length of up to eight cascaded switches. We achieved the exact worst case with a scenario of up to 24 flows, i.e., a line length of up to three cascaded switches. For the other scenarios, we stopped the algorithm after 5h and plotted the upper bound. The determined worst case bounds in the upstream path show that the audio delay requirement of 10 ms is met even while using Fast Ethernet.

³ In the DB-SFA approach, we have to add the transmission delay three times, which is lost due to the concatenation of servers.

⁴ The DB-PMOO-SFA approach does not obtain the correct values in this scenario since packet bursts are not preserved.

Flow	Burst [bytes]	Rate [kBits/s]
Server → EndDevices	108	27 648.0
Service Unit → Server	108	204.0
Handset → Server	64	1632.0

Tab. 4.8: Token Bucket Parameters in Aircraft Cabin

Figure 4.12 addresses the downstream in terms of the delay of packets that flow from the server to an end device. Again, the discrete burst model delivers better results than the rate latency model, and the MIP bound outperforms the discrete burst model. With respect to synchronous playback, the maximum worst case bound in the downstream path for Fast Ethernet is higher than the requirement for the maximum differential delay. This topology can thus not guarantee synchronous playback without using a common time-base with regards to avionic requirements.

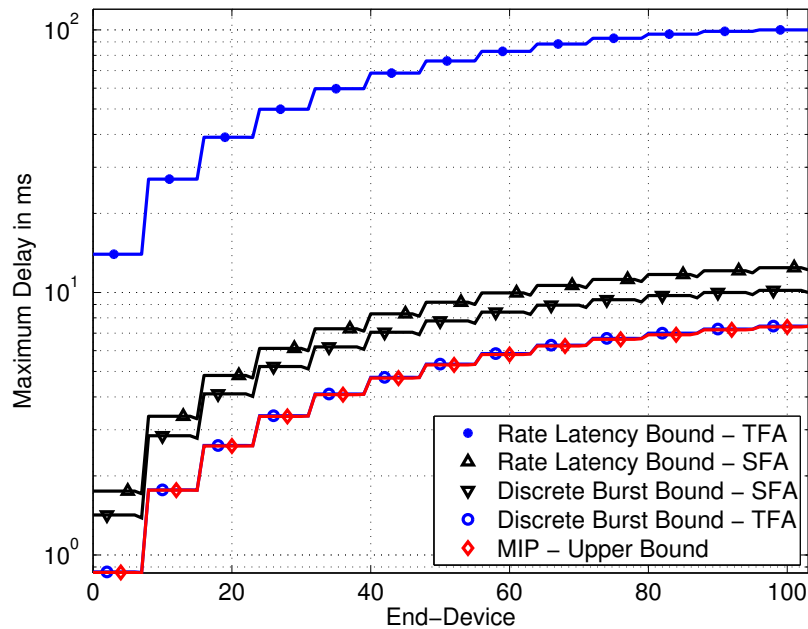


Fig. 4.10: Analysis of Switched Cabin Network—Upstream

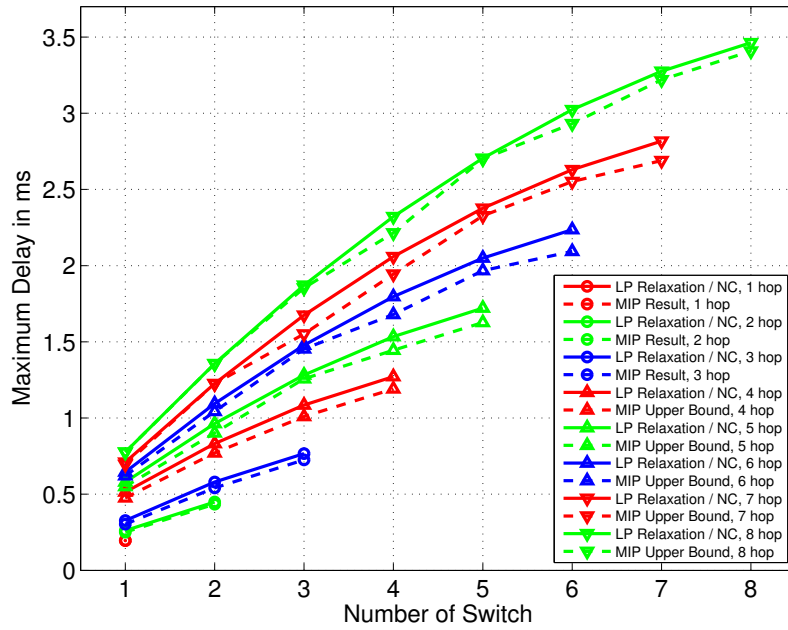


Fig. 4.11: Comparison of LP Relaxation and MIP Bound

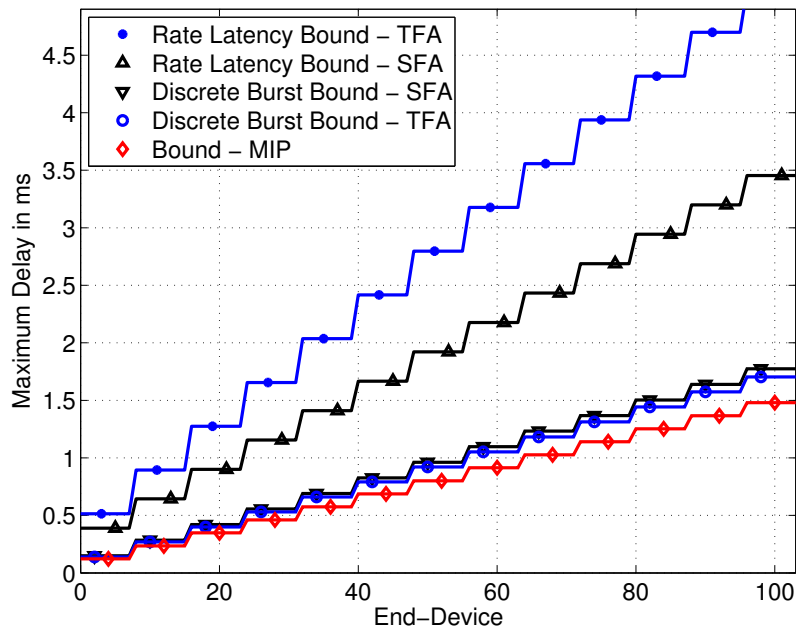


Fig. 4.12: Analysis of Switched Cabin Network—Downstream

4.6 Summary

This chapter introduced a novel algorithm based on *Mixed Integer Programming* that is able to determine the exact worst case in packetized queuing networks. Compared to previous research, we provide a packetized model and express all possible queuing situations by Boolean variables. This approach allows determining the exact queuing situation that leads to the worst case constellation. We also outline the differences from the state-of-the-art approaches. Most techniques are based on the *Network Calculus* and rely on fluid flow models. We found that the fluid flow models do not always provide a good mapping of reality. This is mainly because the exact packetization sequence is unknown and due to the fact that service provided by the transmitting links is not infinitely divisible. Finally, we summarize the approaches that determine the worst case latencies in queuing networks in Table 4.9. The algorithm proposed in this chapter is compared to the state-of-the-art *Network Calculus* approaches. Compared to those methods, our proposed algorithm improves the worst case results in the discussed scenarios by about 1 % to 5 %.

Approach	Fluid Model	Closed Form	Tight Packetized	Tight Fluid	Comp. Effort
TFA	yes	no	no	no	low
Packetized TFA	no	no	no	n/a	low
SFA	yes	yes	no	no	medium
PMOO-SFA	yes	yes	no	no	high
MIP	no	yes	yes	n/a	NP

Tab. 4.9: Comparison of Different Worst Case Approaches

5. DIMTOOL —DIMENSIONING THE NEXT GENERATION AIRCRAFT CABIN

This toolchain fills the gap when different performance calculators are employed to determine performance bounds. Today, worst case computations in certification processes are often hand-crafted or semi-automated. By providing a *consistent view on performance evaluations* we try to ease and speedup certification process. Figure 5.1 summarizes this view on *performance calculation, performance types, aspects, and traffic models*.

The task of performance estimation and performance calculation yielded several approaches in the last decades. There are discrete event simulations based on Monte Carlo studies, methods that aim at the calculation of queuing delays based on queuing theory as well as its alternative NC. The calculated results often fall in the categories of latency, memory utilization, and link utilization. To the best of our knowledge, these tools deliver results that cannot easily be compared since they may assume different models or constraints. In this work, we present the network calculation platform DIMTOOL, which provides consistent view of network performance calculations.

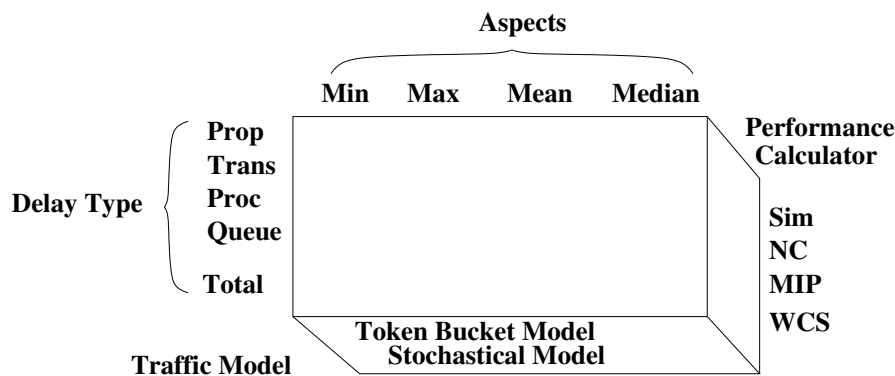


Fig. 5.1: Performance Calculation with DIMTOOL

The structure of this toolchain is *closely related to the certification process* of the novel aircraft cabin. In particular, we separate the following tasks:

- System Integration — The system integrator addresses the requirements of the airlines concerning the cabin layout. The intention is to fulfill those requirements with a few cabin configurations, which will likely be overdesigned to cover all intended use-cases.
- Certification — The certification process requires hard performance bounds for the CIDS functions. They are determined by analytical methods such as NC and required in textual representation.
- Deployment — The bandwidth reservation and forwarding rules are deployed in the network. The deployment takes place in the *Final Assembly Line* (FAL) and may have to be rerun if devices are exchanged.

Ethernet switching technology is very flexible and can thus hardly be covered by the present certification processes. A few standard cabin configurations will be designed and certified to cover most of the use cases desired by the airlines. The deployment will use a protocol to distribute the bandwidth reservation and the forwarding rules to the switches. For the demonstrator, we used SNMP to distribute the configuration into the demonstrator. This protocol will have to be certified as well. Since SNMP has a wide range of commands and data types that are not used in this scenario, a pared-down protocol may be employed that is easier for certification according to DO-178B [97].

5.1 Related Work

The work addressed in this chapter covers related work on network performance calculation and related work on configuration and management protocols. Both are covered by the DIMTOOL suite, either provided by backends or used in the implementation of the configuration tools.

5.1.1 Network Performance Calculation

In this section, we provide an overview of network performance calculation solutions. Throughout this work, network performance calculators are classified by the technology employed. More precisely, we cover *Network Simulation based on the Monte Carlo methods* (Sim), *Network Calculus* (NC) analysis, *Model Checking* (MC) approaches as well as *Worse Case Simulation* (WCS). Table 5.1 gives a comprehensive, not necessarily complete, overview of the available products and their underlying technologies.

A comparison of the results returned from these network performance estimators is difficult because the traffic and network models often differ from each other, and small distinctions may have a significant impact on the results. To resolve this issue, a *generic interface* and *wrappers* are provided that act as *backends* to simplify the comparison of the performance bounds determined. Furthermore, we provide a *Matlab front-end* to the *network calculation tools*.

Computer networks are usually too complex to identify the worst case through network simulation. Popular network simulators are ns-2 [91], ns-3 [92], OMNeT++ [94], OPNET [95], and SSF [109]. The basic idea behind Monte Carlo network simulation is to do individual simulation runs that are based on *different random seeds*. However, it cannot be guaranteed to observe the worst case in one of these runs, so that advanced models and techniques are required. Our previous work [41] presented a method that shifts the bounds achieved in a Monte Carlo simulation towards the worst case observed by analytical models. This approach is referred to as WCS¹.

Recent approaches for performance evaluation may also be based on *real-time scheduling analysis tools*, e.g., ChronVal [56] or SymTA/S [112] to determine performance bounds in communication networks. These tools are usually employed for determining the *worst case schedules* of processes in CPUs, but when those tools are applied to handling *non-preemptive processes*, this technique can be used to model the worst case schedules of frames in a switched network. The model checker UPPAAL [117] was also used to determine performance bounds in the field of Ethernet networks [119]. This is comparable with the recently mentioned *real-time scheduling analyzers*, as they exhaustively enumerate the search space. The approach presented in Section 4 also falls into this category.

¹ Worse Case Simulation

Another well accepted method for determining performance bounds in computer networks is known under the concept of NC [26, 27, 75], being a competitor to classical queuing theory. Mainly academic research of the last decade has brought tools to the community of NC such as DISCO [28], COINC [73], CyNC [102], and RTC [118]. NC was introduced in Section 2.2.3.

Tool	Sim	NC	MC	WCS
DIMTOOL	✓	✓	✓	✓
ns-2 [91]	✓			
ns-3 [92]	✓			
OMNeT++ [94]	✓			
OPNET [95]	✓			
ChronVal [56]			✓	
SymTA/S [112]			✓	
UPPAAL [117]			✓	
DISCO [28]		✓		
COINC [73]		✓		
CyNC [102]		✓		
RTC [118]		✓		

Tab. 5.1: Performance Estimation Tools

In all these performance calculators, the network and the flows traversing the topology have to be specified explicitly by *graphs* and *traffic patterns*. There exist various data formats to describe these environments, which are often tool-dependent. Besides those *tool-dependent data formats*, there exist some *standardized data formats* such as the Common Information Model (CIM) [29] used for network management, or BRITE [18] used for *topology creation*.

5.1.2 Configuration and Management

In this subsection, we give an overview of the common configuration protocols and management architectures. We address SNMP and WBEM² and figure out where those solutions may be applicable in the SCMS. There are certainly other protocols that may be used for configuration tasks, e.g., NETCONF³ [67]. We refer to Fouquet et al. [37] for further information on the integration into SCMS.

² Web Based Enterprise Management

³ Network Configuration Protocol

SNMP The *Simple Network Management Protocol* is a candidate due to its vast distribution in standard products such as routers and switches. The latest version of *Simple Network Management Protocol* is version 3 and is defined in RFC3410 [61].

The information model of SNMP is based on the MIB⁴, a virtual information store which spans a global tree. Collections of related MIB objects are defined in MIB modules, which are in turn defined in the SNMP data definition language. An OID⁵ points to the objects in the MIB tree, so that those objects are addressable by a unique identifier. SNMP defines the following protocol operations: (a) *get*, (b) *get-next*, (c) *get-response*, (d) *set-request*, and (e) *trap*. The *get* and *set-request* operations are used for getting and setting values of MIB objects. With the *get-next* operation we can fully traverse the provided MIB tree, and perform a MIB walk. The traps represent event notifications from the managed device to the manager. The preferred transport protocol for SNMP is UDP.

There exist standardized MIBs such as RFC3418 [62] for SNMP entities, RFC4293 [65] for IP, RFC4022 [63] for TCP, RFC4113 [64] for UDP, and RFC4363 [66] for VLANs. Vendor specific MIBs are provided by manufacturers to provide further functionalities that go beyond the scope of the standardized MIBs. In essence, the switch functionality to configure bandwidth reservation and VLAN entries is usually provided by both vendor specific and standardized MIBs and is highly dependent on the actual implementation. Due to its simplicity and high availability in standard COTS products, SNMP may be a candidate for the configuration in the SCMS.

WBEM The *Web Based Enterprise Management* was developed by the DMTF⁶ and provides a set of management and Internet standard technologies developed to unify the management of distributed computing environments [30]. The DMTF has developed a set of standards such as CIM⁷, CIM-XML, and the CIM Query Language [31]. The WBEM architecture provides a consistent interface to configuration and management protocols, such as SNMP or DMI⁸. These protocols are encapsulated in CIM object providers that forward information to the CIM object manager for integration or interpretation [32]. Figure 5.2 shows the CIM Interop model.

⁴ Management Information Base

⁵ Object Identifier

⁶ Distributed Management Task Force, an alliance of leading companies as Microsoft, Intel, Broadcom

⁷ Common Information Model

⁸ Desktop Management Interface

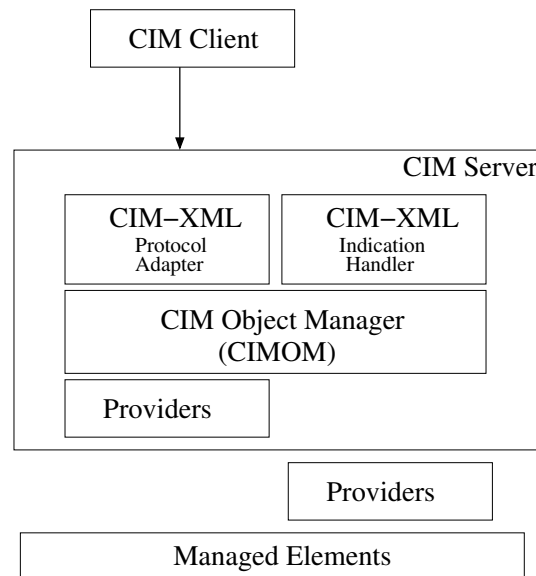


Fig. 5.2: CIM Interop Model

WBEM targets the management of huge, heterogeneous data networks, which might be distributed over several countries. As a consequence of its complexity, WBEM will likely not be accepted as a management platform for the aircraft cabin due to the limited benefits compared to other solutions. In essence, this management architecture is very powerful and feature rich, which may not be necessary for the configuration procedure.

5.2 Topology and Flow Description

There are several data formats available to express network topologies which are highly dependent on the exact field of application and the consequent information content. Some data formats mentioned earlier use XML representation while others use CSV⁹. Our toolbox follows the latter approach and describes topologies via CSV. More precisely we use two sections, one expressing the topology of the network, the other expressing the data flows traversing the network. The EBNF¹⁰ is given in Appendix C.

Listing 5.1 gives an idea of the format we use in this toolchain to express topology and traffic generation. The description covers the tandem scenario given in Figure 5.3; the token bucket rates are given in Table 5.2. The topology section contains the number

⁹ Comma Separated Value

¹⁰ Extended Backus-Naur-Form

```

topo
nodes 8
0,node:10.33.0.102:255.0.0.0,3,switch
3,switch,1,node:10.33.0.103:255.0.0.0
3,switch,2,node:10.33.0.104:255.0.0.0
3,switch,4,switch
4,switch,5,node:10.33.0.105:255.0.0.0
4,switch,6,node:10.33.0.106:255.0.0.0
4,switch,7,node:10.33.0.107:255.0.0.0
#
flows
0, node, { (TokenBucket, 1024, 7.25,
  10.33.0.106, normal(0.202, 0.04)) ) }
1, node, { (TokenBucket, 256, 62.5,
  10.33.0.106, normal(0.202, 0.04)) ) }
2, node, { (TokenBucket, 408, 125,
  10.33.0.107, normal(0.202, 0.04)) ) }
#

```

Listing 5.1: Example Description of Topology and Flows

of nodes and a list of bidirectional edges. The description of each node may contain further information such as the IP address/mask and type of switch. We defined the node types *node* as IP endpoint and *switch* as an Ethernet compliant switch. The flow section contains a list of arrival curves that are to occur in the specified nodes. The

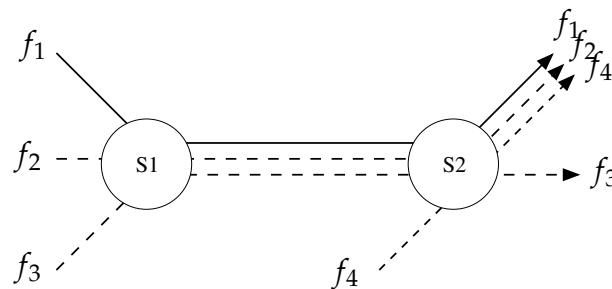


Fig. 5.3: Tandem Scenario with Four Flows

Flow	Burst [bytes]	Rate [kBits/s]
f_1	1024	50
f_2	256	500
f_3	408	1000

Tab. 5.2: Token Bucket Parameters

first two entries specify the described node with numerical id and the type of the node. The remainder contains the arrival curve descriptions, which conforms to a *TokenBucket* arrival curve in this example. Additionally, the destination IP address, as well as the offset of the arrival curve are given. We provide the following arrival curve descriptions: *TokenBucket*, *DualTokenBucket*, *Trace*, and *MMOnOff*.

Some of those arrival curves are better suited for use in some backends than others. In particular, it is not easy to generate reasonable traffic in a Monte Carlo Network Simulation from a pure token bucket curve, due to the fact that the start times do not have a direct counterpart in the token bucket model. On the other hand it is difficult to derive an NC arrival curve from a Markov-modulated on/off process, since the token bucket peak rate may be very high compared to the average rate arising from the Markov-modulated on/off process, unless the parameterized wait times are constant.

5.3 Performance Calculation Backends

The proposed performance evaluator has the ability to address different backends that in turn determine performance bounds. At the time of writing, there exist the following backends: Network Simulation with OPNET, NC analysis using the DISCO network analyzer [104], the rare event simulation as given in [41], and the optimization based approach given in Section 4.

Network Simulation Backend

Network Simulation with Monte Carlo methods is a well accepted technique to estimate performance bounds within a reasonable time, especially when the network topology and the use case hardly allow a real setup. This could be due to the network consisting of a vast number of participating network nodes, or due to the constraints that have to be applied. In general, establishing a realistic testbed of the scenario of interest might be too time consuming and cost intensive. In the Monte Carlo method, the experiment is carried out several times to derive more meaningful results. Since simulation with a computer system implies a deterministic calculation of the result, the processes usually depend on random number generators, which are initialized with different random seeds at each simulation run.

There are popular network simulators that are based on this approach, e.g., SSF, OM-NeT++, and OPNET to name a few. The suggested backend is based on the OPNET network simulator and uses *External Model Access* (EMA) to control OPNET simulations from Matlab. The EMA architecture can be used to create and simulate network scenarios. For this, OPNET provides an interface via shared object/DLL to allow access to the OPNET simulation runtime. Since the computation of several experiments requires a lot of computation time, this backend is run in offline mode, which means that Matlab gets control back while the simulation is still running. We provide methods to check whether the backend has already finished and whether the results are available from the backend.

Network Calculus Analysis Backend

NC is an analytical approach to determine the worst case values for delay and backlog. This approach was introduced in Section 2.2.3. In the NC, we define *traffic envelopes of flows* rather than handling the actual arrival and departure processes known from queuing theory. A detailed introduction to NC is given in Section 2.2.3.

The suggested NC backend uses the DISCO network analyzer. With this toolbox, we obtain a valuable network analyzer class which provides standard network algorithms, such as: (a) Dijkstra's *shortest path algorithm*, (b) an implementation of the *turn prohibition algorithm* [111], and (c) a *topology converter* that converts network graphs to server graphs. The developers of the toolbox have made a lot of progress in recent years in terms of tight NC bounds [103, 105], and so *standard tightening approaches* such as PBOO and PMOO are supported by this toolbox.

Rare Event Simulation Backend

In the pure network simulation backend, the traffic generating nodes have their start time set according to random number generators. This is extremely important because worst case queuing situations of token bucket shaped traffic are only provoked if the offset from each other is at the worst position. For arrival curves in token bucket form, quite a few simulation runs are required to provoke worst case queuing, since offsets are usually not aligned in the most pessimistic way. Depending on the network topology and traversing traffic flows, this simulation backend determines pessimistic offsets and runs a parameterized simulation.

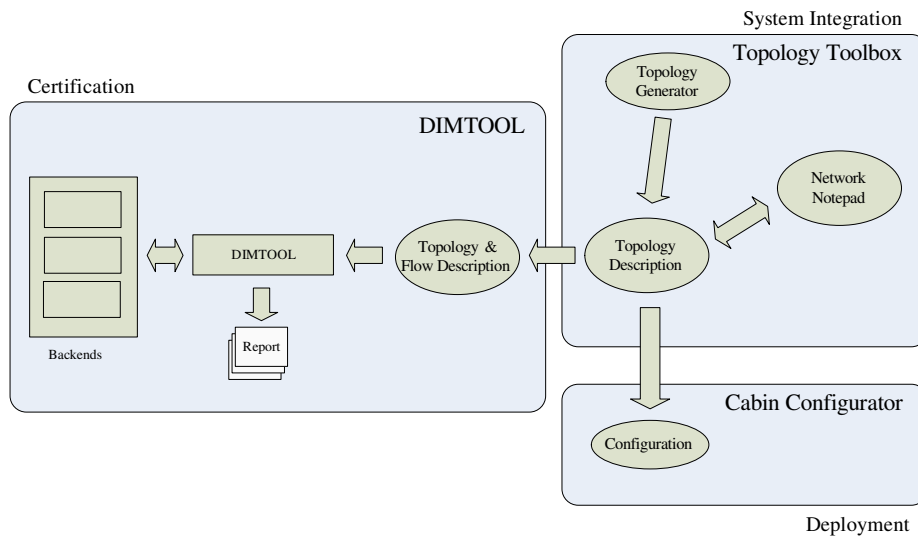


Fig. 5.4: Toolbox DIMTOOL

MIP Analysis Backend

The approach of using a MIP to determine worst case bounds in a switched, non-preemptive queuing network was explained in Section 4. We presented a packetized traffic model that describes the worst case schedules in a single MIP instance using a similar technique as that common in model checking. The worst case is modeled by integer variables that express the fact that one packet is delayed by another packet. This model allows better bounds, since NC approaches commonly have to assume the worst case by a delay of a maximum sized frame.

5.4 DIMTOOL Architecture

Figure 5.4 shows the architecture of the DIMTOOL. On the right we see the tools based on the topology. In order to provide a convenient graphical user interface, we built some import routines for the network editor *Network Notepad* [40] as well as the EADS internal program *Camfigurator*. Within this tool we can easily assemble cabin network configurations that match different airplane types and setups. The *Topology Generator* is mainly used for research and advance development in the field of the aircraft cabin and allows the simulation of various cabin scenarios. This is of special interest when new techniques are elaborated, such as new *scheduling algorithms* in the switches or synchro-

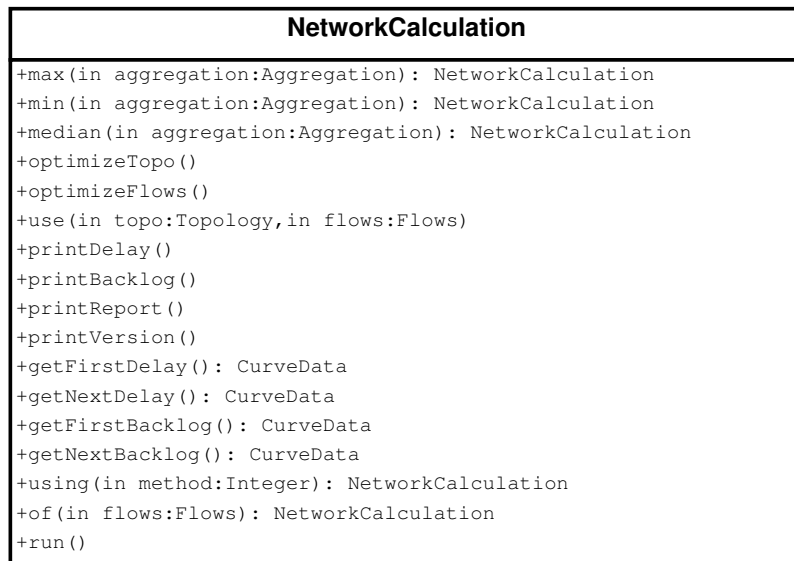


Fig. 5.5: Class NetworkCalculation

nization mechanisms as PTP. The *Configuration* contains the actual bandwidth reservation, VLAN rules, or routing entries. The configuration can then be distributed with standardized management protocols such as SNMP or WBEM. The *Topology Description* is a system independent format based on XML that contains the actual topology as well as the devices hosted on the network. The *Topology Description* is exported to a CSV format that contains both the topology and the flows through the system. Currently, all the export routines and transformations mentioned are provided by a C++ class library based on the *Ultimate++ framework* [116]. The *Topology and Flow Description* is passed to the DIMTOOL, which creates reports from the performance calculator backends. These reports are either in diagram form or represented by text files that are in turn required in the certification processes.

The remainder of this section shows the methods of the DIMTOOL as given in the *NetworkCalculation* class shown in Figure 5.5. These methods address the following tasks: Computational Methods, Execution Methods, Finalization Methods, Result Methods, Global Parametrization Methods, and Optimization Methods.

Computational Methods The expected parameter *aggregation* of the Computational Methods thereby defines how the determined delay bounds will concretely be assembled, e.g., whether to identify the jitter, the delay difference versus variation (in case of multicast flows) or the sum. More precisely we introduce the following methods: (a) *min*, (b) *max*, (c) *avg*, and (d) *median*. The respective function determines the minimum, maximum, average, and median values for delay and backlog.

Execution Methods The execution methods are used to control the performance calculation, i.e., which backend to choose, which topology and cross traffic flows to use, and which flows to investigate. The run method actually triggers the backend and performs the computation or simulation. We introduce the following methods: (a) method *use* specifies the topology and flows to be used, (b) method *of* defines the addressed flows, (c) method *using* chooses the backend to be used, and finally (d) method *run* triggers the actual computation/simulation.

Finalization Methods Some backends provide instantaneous performance calculation as being provided by the NC backend, i.e., returning from the *run* method call implies that NC results are available. The other backends currently use an offline computation, since the computation is very intensive compared to the NC backend, so that the calling Matlab instance would be blocked. For the backends that run in offline mode, we provide finalization methods to check whether the backend has already finished so that the results can be read from the backend. We provide the following methods: (a) method *isFinished* returns true when the simulation/analysis is finished, otherwise *isFinished* returns false, and (b) method *finalize* finalizes the backend. When *finalize* was called on a finished backend, the Result Methods can be executed to read the results from the backend.

Result Methods The result methods are used to process information as detected by the backends. These methods can be called when *isFinished* from the previously stated Finalization Methods returns 1. In order to process the results, we provide print methods and get methods. Print methods create textual representations of the results while the returned objects of the get methods can be used for plotting in Matlab. We also introduce the following methods: (a) method *printDelay* prints the delay report, (b) method *printBacklog* prints the backlog report, (c) method *printReport* prints both the

delay and the backlog report, (d) methods *getFirstDelay/getNextDelay* allows iteration over delay curves, and (e) methods *getFirstBacklog/getNextBacklog* allows iteration over backlog curves.

Global Parametrization Methods The global parametrization methods are used to set the parameterizable topology options globally. This allows the central parametrization of the FIFO assumption, parametrization of the link speed, and setting the processing delay in intermediate switch nodes. We provide the following methods: (a) method *fifo* sets FIFO processing order globally, (b) method *speed* sets the link speed globally, and (c) method *proc* sets the processing delay globally.

Optimization Methods The optimization methods act as an entry point for implementing optimization algorithms for flows and topology. These optimization algorithms are intended to invoke the provided backends in order to optimize aspects of the topology and the flows. In the current version, the Optimization Methods *optimizeTopo* for topology optimization and *optimizeFlows* for flow optimization are provided.

Matlab Functions We embed the given architecture into Matlab and provide seamless integration through Matlab interfaces. Currently the following functions are supported by our framework:

- *dimtool_init* — initializes DIMTOOL temporarily,
- *dimtool_install* — installs DIMTOOL permanently,
- *dimloadtf* — loads the topology and flow descriptions,
- *dimnc* — returns an instance of the class *NetworkCalculation*,
- *dimbar* — plots a bar graph of the delay results,
- *dimbarbacklog* — plots a bar graph of the backlog results,
- *dimplot* — plots a graph of the delay results,
- *dimplotbacklog* — plots a graph of the backlog results,
- *dimprinttofile* — writes the results to a file.

The entry point to all calculation or simulation runs is an instance of *NetworkCalculation*, which is returned by *dimnc*. The interface functions can be called directly on this return value. The function *dimloadtf* loads the topology and flow description in the form described in Section 5.2. This function takes an optional parameter to add the standard values for the interframe gap and preamble seen in switched Ethernet networks. When the *NetworkCalculation* class has been parameterized, the simulation or analysis can be triggered by calling *nc.run()*. If *nc.isFinished()* returns true, the results are available and can be processed further by *dimbar*, *dimplot*, or *dimprinttofile*.

5.5 Topology Toolkit

In the remainder of this chapter we introduce the *Topology Toolkit* as provided by the DIMTOOL suite. The toolkit consists of the *Cabin Topology Generator* and the *Topology Bandwidth Calculator*.

The *Cabin Topology Generator* provides standard cabin layouts for the A30x and A350 projects as well as for the A380. The topology generator takes the input parameters as listed in Table 5.3. The end devices are distributed uniformly in the generated topology so that a regular occurrence of end devices is assured. In order to shift the worst case analysis towards the worst case, we start the distribution of end devices and access points at the outermost DEU.

Parameter	Description
<i>linelength</i>	Maximum depth of cabin tree topology
<i>psu</i>	Number of PSUs per DEU
<i>ibu</i>	Number of IBUs per DEU
<i>cvms</i>	Number of cameras per line
<i>fap1</i>	Number of FAPs per line, domain ACD
<i>fap2</i>	Number of FAPs per line, domain AISD
<i>fap3</i>	Number of FAPs per line, domain PIESD
<i>handsets per DEU</i>	Number of handsets per DEU
<i>handsets per line</i>	Number of handsets per line
<i>wireless sensor access points</i>	Number of wireless sensor access points
<i>crew WLAN access points</i>	Number of crew WLAN access points
<i>passenger WLAN access points</i>	Number of passenger WLAN access points

Tab. 5.3: Input Parameters for the Topology Generator

Alg. 2 Bandwidth Calculation

```

1: function CALCBANDWIDTH(Node  $n$ )
2:   for all  $i \in \text{NEIGHBOURS}(n)$  do
3:     if ALREADYVISITED( $i$ ) then
4:       continue
5:     end if
6:     SETVISITED( $i$ )
7:     for all  $p \in \text{Priorities}$  do
8:        $l \leftarrow \text{GETLOCALBANDWIDTHDOWN}(n, p)$ 
9:       SETLOCALBANDWIDTHDOWN( $i, p, l$ )
10:    end for
11:    CALCBANDWIDTH( $i$ )
12:    for all  $p \in \text{Priorities}$  do
13:       $l \leftarrow \text{GETLOCALBANDWIDTHUP}(i, p)$ 
14:      SETLOCALBANDWIDTHUP( $n, p, l$ )
15:    end for
16:  end for
17: end function

```

The *Topology Bandwidth Calculator* determines the static bandwidth reservations and assigns VLAN tags from the VLAN pool. The VLAN pool provides VLAN tags in accordance with the VLAN concept given in Section 3.6. The static bandwidth reservations and VLAN table entries are determined by a recursive depth-first search. The algorithm is given in Algorithm 2. We perform a depth-first search and visit nodes pre and post-order. The methods *getLocalBandwidthDown* and *setLocalBandwidthDown* are used to propagate the traffic sent from the server to the end devices. These methods are called before the recursion call, i.e., pre-order. The methods *getLocalBandwidthUp* and *setLocalBandwidthUp* are called after the recursion call and are used to determine the estimated traffic bandwidth from the end devices to the server, i.e., upstream. The method *getLocalBandwidthUp* returns the bandwidth that is generated by the end devices and subsequently forwarded towards the server. The traffic forwarded towards the server is multiplexed by the switches so that the aggregate of the end device adds at each switch.

Since the server is the only traffic source in the SCMS concept that sends traffic to end devices, the downstream traffic is determined straightforwardly and set in accordance with Table 3.1, Section 3.5.

Alg. 3 VLAN Assignment

```

1: function ASSIGNVLAN(Node  $n$ )
2:   for all  $i \in \text{NEIGHBOURS}(n)$  do
3:     if ALREADYVISITED( $i$ ) then
4:       continue
5:     end if
6:     SETVISITED( $i$ )
7:     ASSIGNVLAN( $i$ )
8:     if ISENDDEVICE( $n$ ) then
9:        $vlan \leftarrow \text{GETVLANFROMPOOL}(n)$ 
10:      SETVLAN( $vlan$ )
11:    end if
12:  end for
13: end function

```

The Algorithm 3 assigns a VLANs to each end device according to the concept given in Section 3.6. Both algorithms contain recursive function calls and use the methods *alreadyVisited* and *setVisited* to determine the spanning tree of the SCMS topology. The first function call starts at the root node, i.e., at the server.

5.6 Deployment

The *Cabin Configurator* is used to deploy the static bandwidth reservations and forwarding rules into the network as determined by the algorithms given in Section 5.5. Figure 5.6 shows the workflow of the *Cabin Configurator*. At the start of the deployment, we initially distribute arbitrary IP addresses by the DHCP¹¹ mechanism [58]. When the number of expected DHCP acknowledgments is reached after a certain timeout, we move to the next state. In this state, we perform an LLDP¹² topology scan according to IEEE 802.1AB [49]. This topology scan is compared to the normative value of the cabin layout. In the case of consistent matching, we program the bandwidth reservation, the forwarding rules, as well as the final IP addresses of the management interface into the switches. The deployed rules will be reread from the devices, written as a deployment protocol, and compared to the normative values. If any of these checks fails, we move to the end state *Fail*, otherwise we move to the end state *Stop*.

¹¹ Dynamic Host Configuration Protocol

¹² Link Layer Discovery Protocol

Algorithm 4 shows the recursive *Deploy* function. We first calculate the bandwidth per priority that is generated by the end devices and subsequently forwarded towards the server (line 9 to 11). From line 12 to 16 we collect all VLANs that ingress at each port. We thus derive a VLAN port mask that is set in 18 to 20. In line 21 to line 29 we reserve the actual bandwidth that was calculated with Algorithm 2.

A prototype of the *Deploy* algorithm was implemented in C++. This implementation uses the SNMP library SNMP++ [36]. The cabin demonstrator that is addressed in Section 6.2 has been configured with this prototype implementation.

Alg. 4 Deployment

```

1: function DEPLOY(Node  $n$ )
2:   for all  $i \in \text{NEIGHBOURS}(n)$  do
3:     if ALREADYVISITED( $i$ ) then
4:       continue
5:     end if
6:     SETVISITED( $i$ )
7:     DEPLOY( $i$ )
8:      $port \leftarrow \text{GETPORT}(n,i)$ 
9:     for all  $p \in \text{Priorities}$  do
10:       $bw_p \leftarrow bw_p + \text{GETLOCALBANDWIDTHUP}(i, p)$ 
11:    end for
12:     $V \leftarrow \text{GETVLANS}(i)$ 
13:     $V_g \leftarrow V_g \cup V$ 
14:    for all  $v \in V$  do
15:       $P_v \leftarrow P_v \cup \{port\}$ 
16:    end for
17:  end for
18:  for all  $v \in V_g$  do
19:    SETVLANENTRY( $v, P_v$ )
20:  end for
21:  for all  $p \in \text{Priorities}$  do
22:    RESERVEBANDWIDTH( $up, p, bw_p$ )
23:    for all  $port \in \text{Ports}$  do
24:      if  $port \neq up$  then
25:         $bw \leftarrow \text{GETLOCALBANDWIDTHDOWN}(root,p)$ 
26:        RESERVEBANDWIDTH( $port,p, bw$ )
27:      end if
28:    end for
29:  end for
30: end function

```

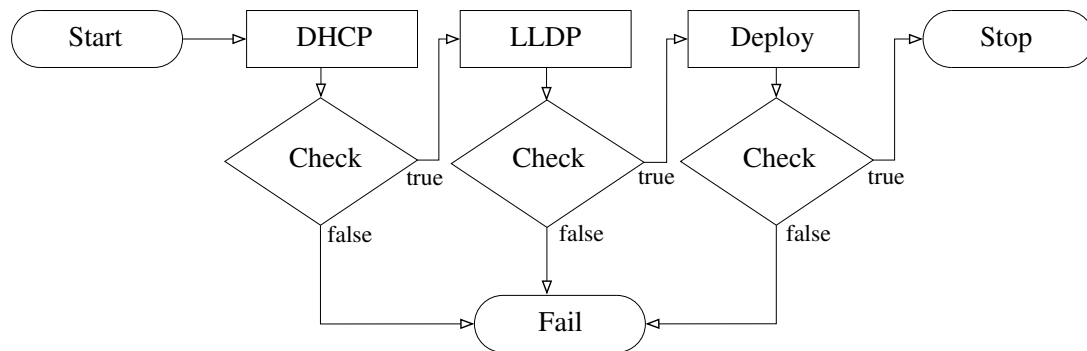


Fig. 5.6: Cabin Configurator Workflow

5.7 Summary

In this section, we introduced the DIMTOOL, a dimensioning tool and performance calculation toolbox for the aircraft cabin. Compared to previous tools, we provided several performance estimation approaches within a single toolbox.

The *Performance Evaluator* provides the approaches NC, Network Simulation, MIP, and Worst Case Simulation, so that both analytical and simulative techniques are given. This provides comparable and transparent performance bounds, which in turn helps find flaws in performance modeling. In addition to the *Performance Evaluator*, the DIMTOOL also has the *Topology Toolkit* (*Cabin Topology Generator* and the *Topology Bandwidth Calculator*) and the *Cabin Configurator*. The *Cabin Topology Generator* provides standard layouts for the aircraft cabin and generates topologies with given input parameters (such as the number of devices or the line length). The *Topology Bandwidth Calculator* is used to determine the static bandwidth reservation and VLAN forwarding rules in the system. Finally, the *Cabin Configurator* deploys the reservations and forwarding rules into the network. For this we provided an implementation based on SNMP that was in turn used in Chapter 6 for setting up the demonstrators.

This novel platform helps get a realistic view of worst case modeling as needed for a switched aircraft cabin. The next chapter takes a closer look at the performance bounds in the aircraft cabin and employs the DIMTOOL for realistic scenarios.

6. EVALUATION

This chapter evaluates the novel aircraft cabin network based on switched Ethernet (SCMS¹). The cabin layouts introduced in Section 3 are evaluated in Section 6.1. For this, we use the DIMTOOL platform given in the last chapter. Section 6.2 addresses a realistic test system. We determine the performance bounds of a simplified mockup and study the applicability of COTS switches in the SCMS.

6.1 DIMTOOL Performance Bounds

The performance bounds are determined with the DIMTOOL platform. Section 6.1.1 evaluates the performance bounds in the *cabin server*. Section 6.1.2 addresses the *single domain* aircraft cabin. The single domain cabin layout is considered as a replacement for the current CIDS², which is based on switched Ethernet. Section 6.1.3 gives the results for the *multi-domain* aircraft cabin. The multi-domain cabin layout has the chief advantage of unifying several safety domains into a single network, which promises clear weight savings.

6.1.1 Cabin Intra-Server Communication

This section addresses the performance bounds observed in the intra-server communication scenario. The communication inside the cabin server is also mandatory when regarding higher safety levels. Besides the scheduling of processes, we are interested in *hard performance bounds* for the inter CPU communication. These hard guarantees are forwarded to real time analysis tools like ChronVal [56] or SymTA/S [112] to determine worst case execution times.

¹ Switched Cabin Management System

² Cabin Intercommunication Data System

Figure 6.1 shows the simplified architecture of the intra-server communication. The cabin server consists of two dual-core CPUs being interconnected by Ethernet. The following functions are covered by the cabin server: (a) PRAM (e.g., boarding music, safety briefing), PA (i.e., audio announcements from crew to passengers), *Cabin Illumination* (CIL) (i.e., cabin illumination with different light scenarios), *Reading Lights* (LI) (i.e., switching of reading lights), IFE (i.e., videos, internet access, games), FAP³ (i.e., control panel for cabin functions), and Cabin Interphone (i.e., crew interphone, conference circuit). Table 6.1 briefly lists the expected traffic flows and the related priority level.

Each of these functions is realized as a partition (similar to process) in the ARINC 653 compliant *operating system*. These partitions run on CPU 0 to CPU 2. The sequencer on CPU 3 controls and manages the communication to the end devices that are connected by Line 1 to Line n. The sequencing unit acts as a gateway to the cabin network, so that the communication between partition and end devices pass a protocol converter.

Figure 6.2 outlines the results of our worst case analysis toolchain. We determined the following bounds for the end-to-end delay: The non-FIFO NC bound, the FIFO NC bound, and the bound determined by Network Simulation. We observe that the NC bounds are relatively tight compared to the worst case observed in the Network Simulation. The reason is that the number of traversed servers has a direct impact on the tightness of the bounds achieved [22, 69]. In this scenario, only one server/node is traversed by the traffic flows.

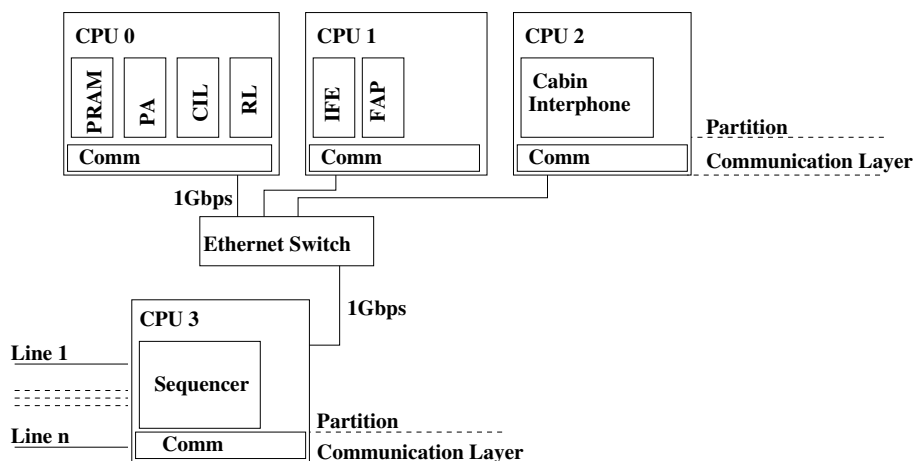


Fig. 6.1: Intra-Server Communication

³ Flight Attendant Panel

Device	Packet Size	Bandwidth [kbit/s]	Priority
PRAM → Seq	86	486.40	5
PA → Seq	64	1.60	7
CIL → Seq	64	1.28	7
LI → Seq	64	0.64	7
IFE → Seq	1522	3125.00	0
FAP → Seq	1522	125.00	7
Seq → PA	64	1.60	7

Tab. 6.1: Parameters in Intra-Server Communication

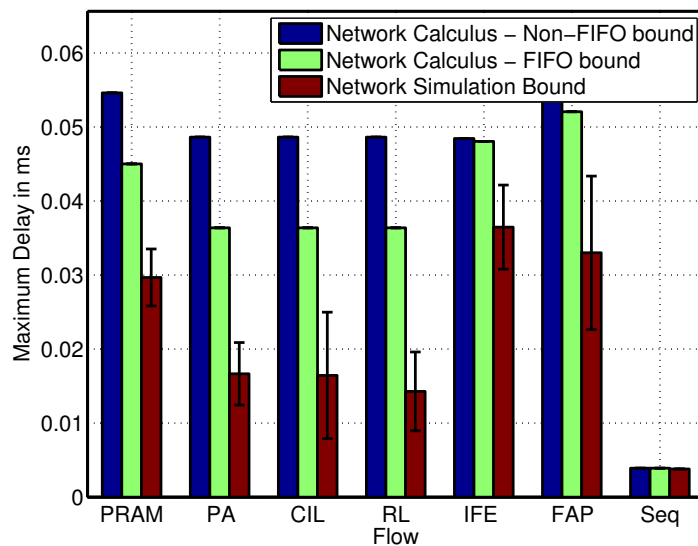


Fig. 6.2: Analysis of CS Using DIMTOOL

The results were obtained by executing the code shown in Listing 6.1 using the DIMTOOL platform: The code invokes an NC analysis on the intra-server communication network. The Matlab calls are forwarded to the wrappers which in turn invoke the chosen backend, in this case the DISCO network analyzer using the SFA algorithm [104]. We set the processing delay used by DISCO and calculate the FIFO bound.

The code given in Listing 6.2 simulates the intra-server communication. The calls are forwarded to the Java wrapper which invokes the network simulator backend. The network simulator backend is based on OPNET, which is called from the OpnetSimulator class. The network traffic is generated from the token bucket traffic model as defined in Table 6.1.

```

tf = dimloadtf('intraserver.txt', 0)
nc = dimnc
agg = Factory.getSumAggregation
nc.use(tf.getTopo, tf.getFlows)
nc.max(agg).of(tf.getFlows).using(2).
    method(2).fifo(1).proc(0.012)
nc.run

```

Listing 6.1: NC Analysis with DIMTOOL, Intra-Server, FIFO

```

nc = dimnc
agg = Factory.getSumAggregation
nc.use('intraserver.txt')
nc.max(agg).of(tf.getFlows).using(1)
nc.run

```

Listing 6.2: Simulation Run with DIMTOOL, Intra-Server

The last result in Figure 6.2, i.e., the non-FIFO bound, is determined by Matlab calls similar to those mentioned above. The parameter of the method *using* decides which backend to choose.

6.1.2 Single Domain Switched Ethernet Cabin

In this use case, we study the worst case delays of a switched version of the CIDS using DIMTOOL. The topology employed is a step towards a COTS-enabled CIDS and was already discussed in [42], where the first simulation results were introduced. The basic *topology* setup follows the *single domain* cabin layout shown in Figure 3.1, Section 3.2. Up to 22 lines are foreseen in a typical airplane, with a maximum depth of 15 Ethernet hops. In this example, 105 high priority flows traverse one line of the simplified cabin network with 13 daisy-chained Ethernet switches. One of these flows acts as a multicast flow from the cabin server to the end devices. The others flow from the end devices back to the server. In fact, we connected seven *Passenger Service Units* and one handset to each switch and employed the traffic patterns shown in Table 6.2.

Figure 6.3 shows the results for the network simulation backend, the WCS backend, and the NC backend. The delays from the WCS are shifted towards the NC bounds by about 10% compared to the standard network simulation. Furthermore, the FIFO bound of the NC analysis does not hold. In this scenario, we do need the non-FIFO bound to be absolutely sure that delays greater than the analytical bound cannot be

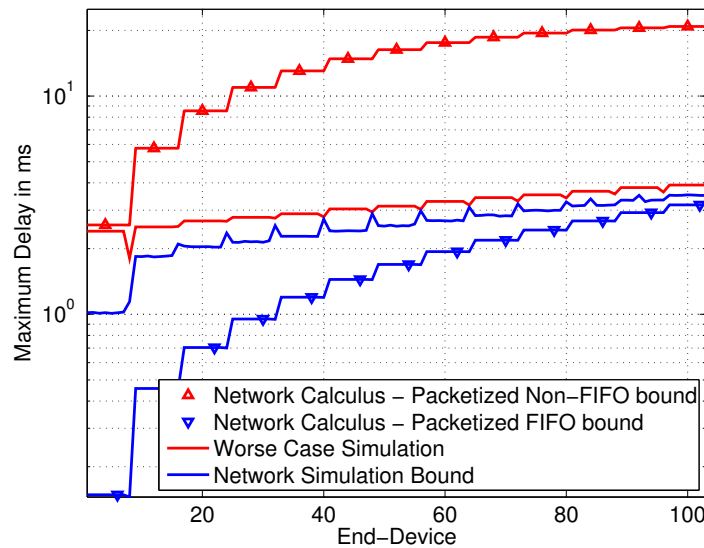


Fig. 6.3: Analysis of SCMS using DIMTOOL

Device	Packet Size	Bandwidth [kbit/s]	Priority
Srv → EndDevices	108	27 648	7
PSU → Srv	108	204	7
Handset → Srv	64	1632	7
Bulk Traffic	$\leq 64, \geq 1518$	8000	0

Tab. 6.2: Parameters in Single Domain Aircraft Cabin

observed. In addition, we observe that the worst case bounds from the NC analysis are getting worse as the number of traversed switches increases. The NC analysis confirms the results from related research, such as [22, 69], which means that we lack tightness when traffic flows traverse several servers.

The results were obtained using the proposed DIMTOOL platform by running Listing 6.3 and Listing 6.4. Both the NC backend and the simulation backend are invoked by the given code. The NC analysis employs the TFA algorithm as given in [104]. The network simulation is performed by the OPNET simulation backend.

The remaining curves of Figure 6.3, i.e., the WCS and FIFO bounds, are determined by similar Matlab calls. The parameter of the method *using* decides which backend should be chosen.

```

tf = dimloadtft('scms.txt', 0)
nc = dimnc
agg = Factory.getSumAggregation
nc.use(tf.getTopo, tf.getFlows)
nc.max(agg).of(tf.getFlows).using(2).
    method(1).fifo(0).proc(0.003)
nc.run

```

Listing 6.3: NC Analysis, Aircraft Cabin, Non-FIFO

```

nc = dimnc
agg = Factory.getSumAggregation
nc.use('scms.txt')
nc.max(agg).of(tf.getFlows).using(1)
nc.run

```

Listing 6.4: Simulation Run with DIMTOOL, Aircraft Cabin

6.1.3 Multiple Domain Switched Ethernet Cabin

This section discusses simulation and analytical results in the multi-domain switched aircraft cabin. The configuration profiles are set according to Section 3.5.2. These cabin profiles merge the different safety domains ACD, AISD, PIESD, and PODD into a single network. Due to the numerous end devices which provide video streaming and high bandwidth services, these cabin profiles require at least Gigabit in the backbone to cover all the static bandwidth reservation inquiries. Table 6.3 summarizes the traffic profiles that occur in the multi-domain scenario. Throughout this section, we apply the OMNET network simulator as well as the NC backend.

The results of the A30x cabin profiles introduced in Section 3.5.2 are shown in the following figures. Figure 6.4 shows the results of the Network Simulation for the devices Passenger Service Unit, Illumination Ballast Unit, Handset, FAP, and Figure 6.5 shows the analytical NC results. Thanks to the Gigabit backbone, the maximum delay of those devices is far below 10 ms. Since the maximum delay of the PSU is even below 1 ms, the multicast delay difference is also lower than this value. As a consequence, the requirements introduced in Section 2.1.1 are met.

Device	Packet Size	Bandwidth [kbit/s]	Priority
Srv → EndDevices	108	27.648	7
Srv → EndDevices	$\leq 64, \geq 1518$	120.000	4
PSU → Srv	108	0.204	6
IBU → Srv	108	0.204	6
Handset → Srv	64	1.632	7
Camera → Srv	$\leq 64, \geq 1518$	15.000	4
FAP → Srv	$\leq 64, \geq 1518$	1.000	4
Wireless Sensor	$\leq 64, \geq 1518$	1.000	2
Crew WLAN	$\leq 64, \geq 1518$	204.000	1
Passenger WLAN	$\leq 64, \geq 1518$	204.000	0

Tab. 6.3: Parameters in Multi-Domain Aircraft Cabin

The results for both the simulation and the NC analysis for the domains with lower safety requirements are shown in Figures 6.6 and 6.7.

Similar to the recently mentioned A30x scenario, Figures 6.8, 6.9, 6.10, and 6.11 show the results for the A350; Figures 6.12, 6.13, 6.14, and 6.15 show the results for the A380.

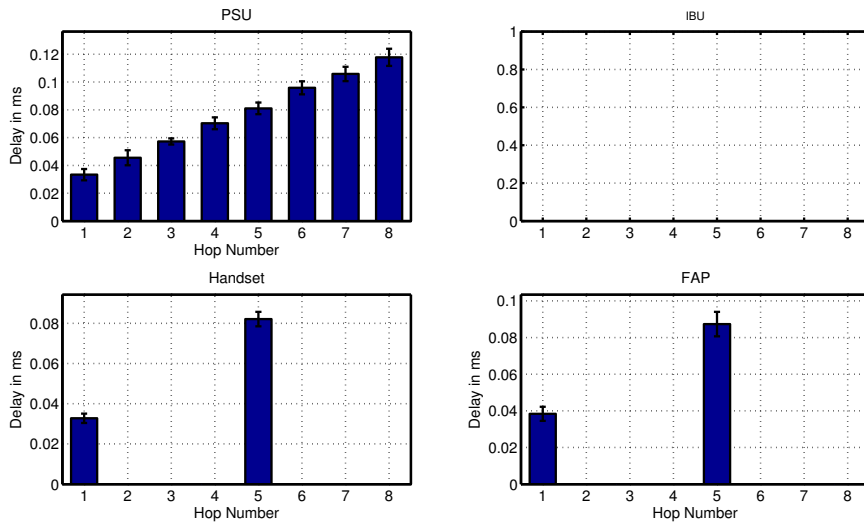


Fig. 6.4: Simulation Results of Multi-Domain Concept using DIMTOOL, A30x Scenario, Passenger Service Unit, Illumination Ballast Unit, Handset, FAP

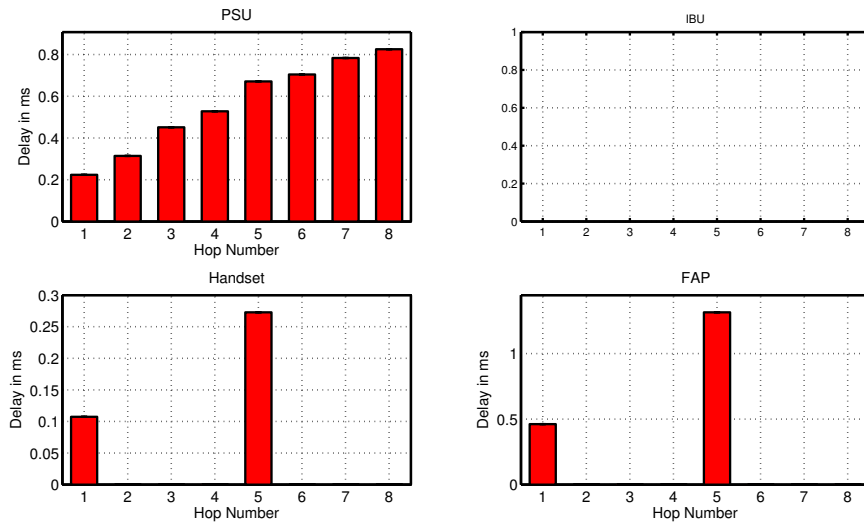


Fig. 6.5: Packetized Non-FIFO NC Results of Multi-Domain Concept using DIMTOOL, A30x Scenario, Passenger Service Unit, Illumination Ballast Unit, Handset, FAP

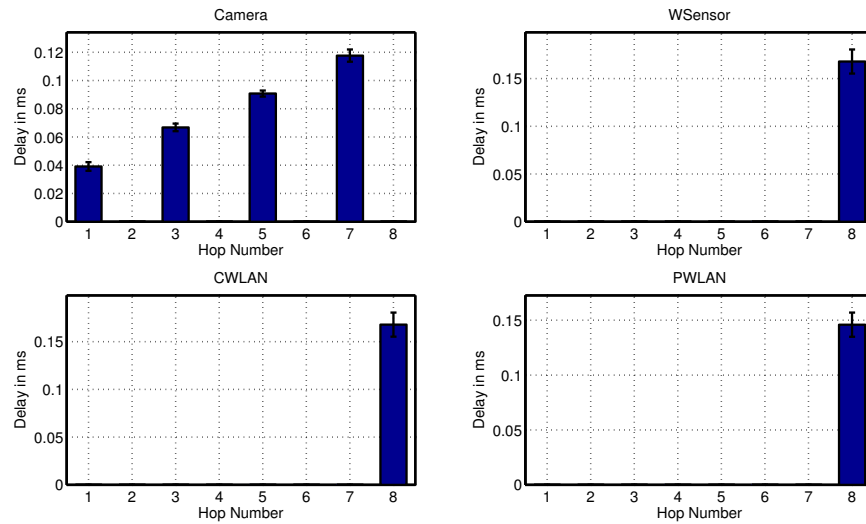


Fig. 6.6: Simulation Results of Multi-Domain Concept using DIMTOOL, A30x Scenario, Camera, Wireless Sensor, Crew WLAN, Passenger WLAN

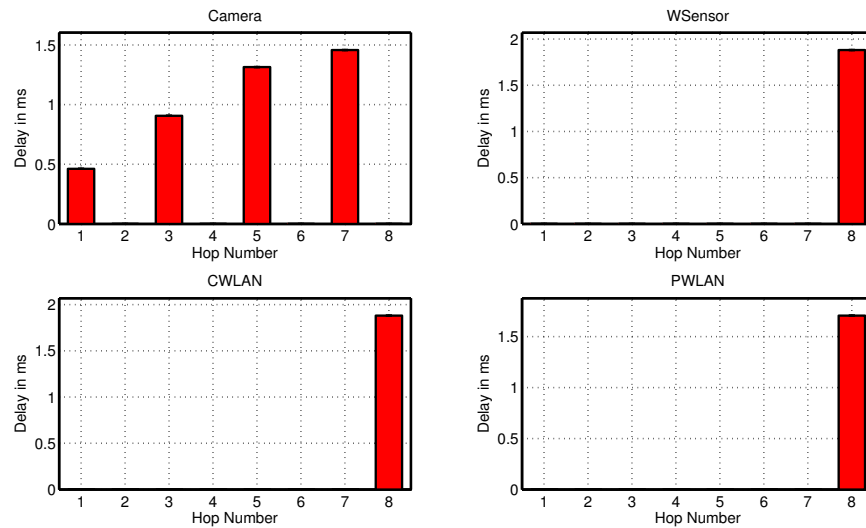


Fig. 6.7: Packetized Non-FIFO NC Results of Multi-Domain Concept using DIMTOOL, A30x Scenario, Camera, Wireless Sensor, Crew WLAN, Passenger WLAN

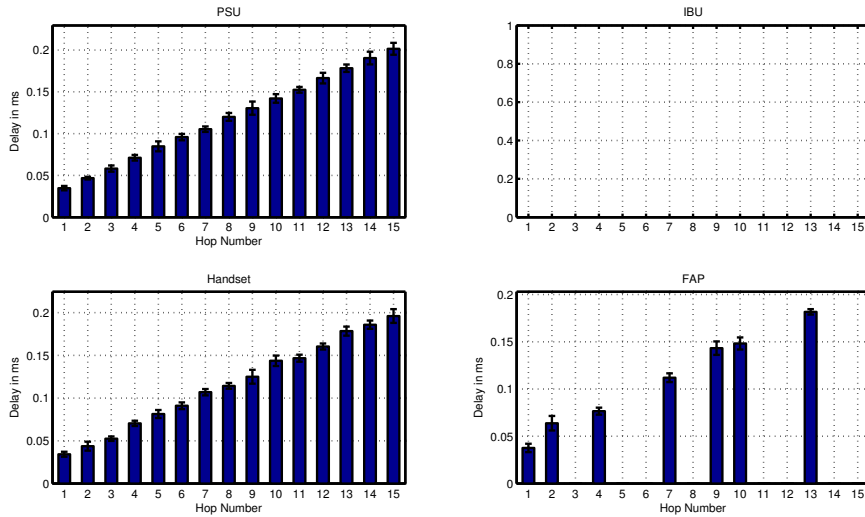


Fig. 6.8: Simulation Results of Multi-Domain Concept Using DIMTOOL, A350 Scenario, Passenger Service Unit, Illumination Ballast Unit, Handset, FAP

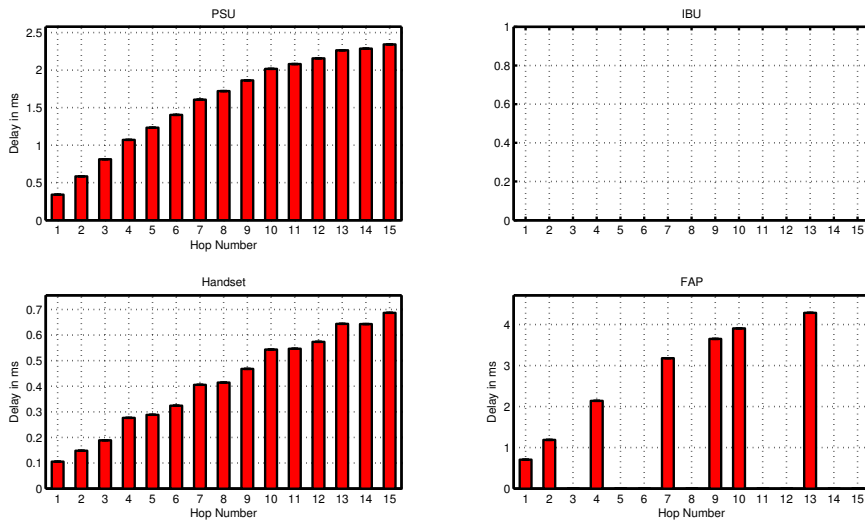


Fig. 6.9: Packetized Non-FIFO NC Results of Multi-Domain Concept Using DIMTOOL, A350 Scenario, Passenger Service Unit, Illumination Ballast Unit, Handset, FAP

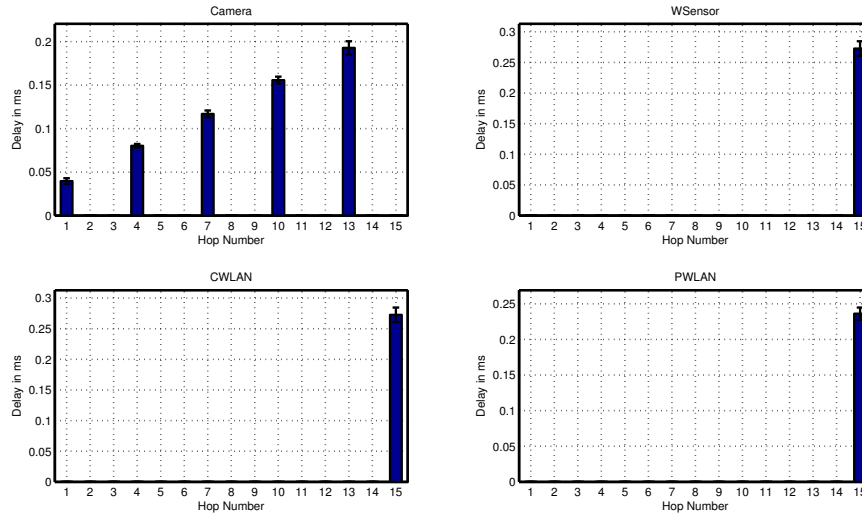


Fig. 6.10: Simulation Results of Multi-Domain Concept Using DIMTOOL, A350 Scenario, Camera, Wireless Sensor, Crew WLAN, Passenger WLAN

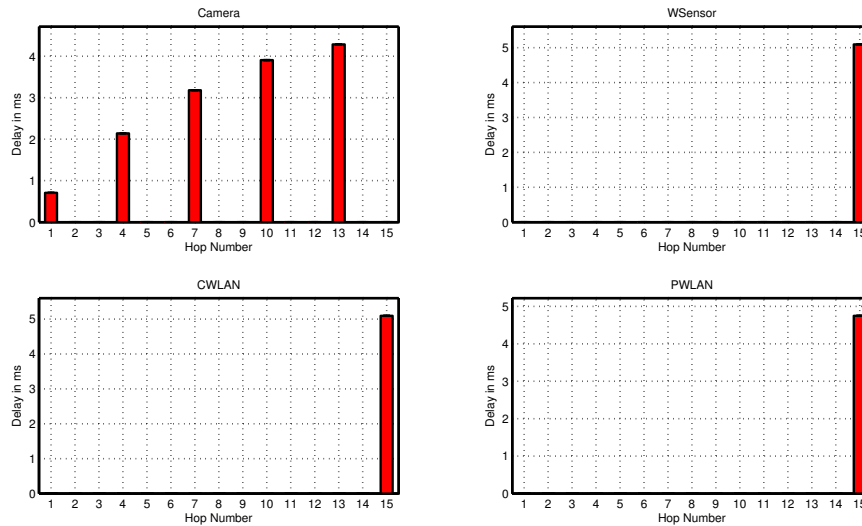


Fig. 6.11: Packetized Non-FIFO NC Results of Multi-Domain Concept Using DIMTOOL, A350 Scenario, Camera, Wireless Sensor, Crew WLAN, Passenger WLAN

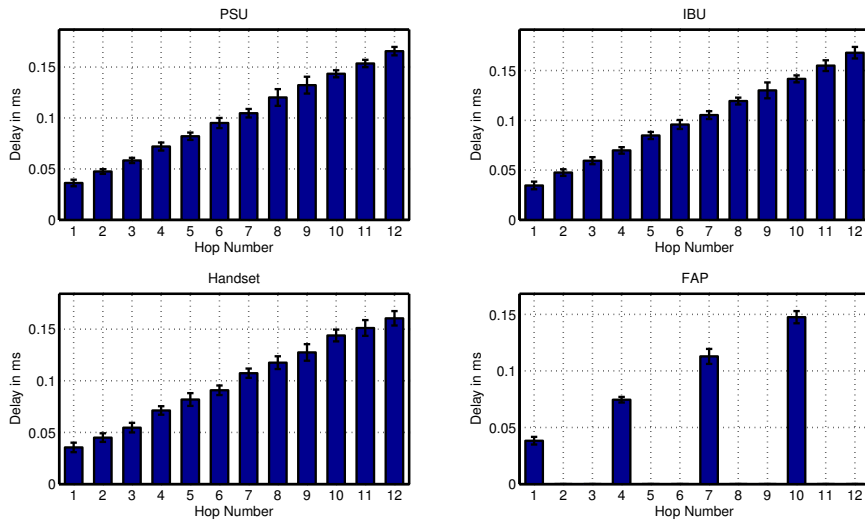


Fig. 6.12: Simulation Results of Multi-Domain Concept Using DIMTOOL, A380 Scenario, Passenger Service Unit, Illumination Ballast Unit, Handset, FAP

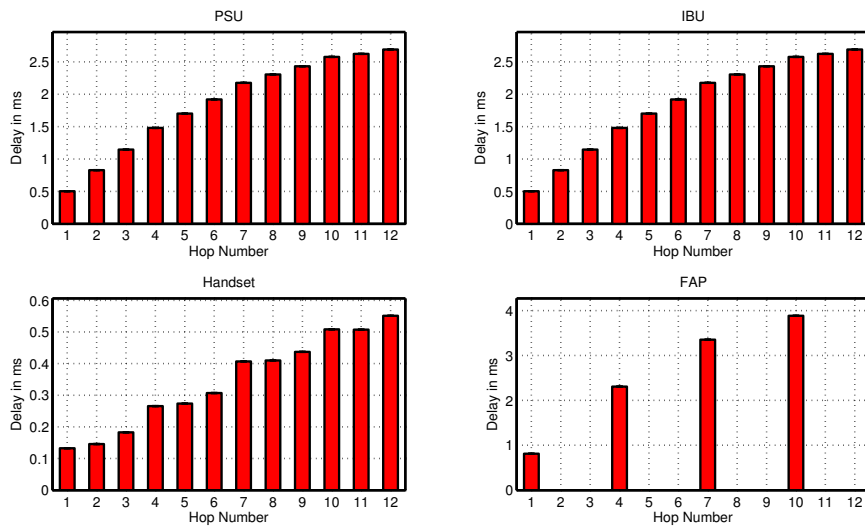


Fig. 6.13: Packetized Non-FIFO NC Results of Multi-Domain Concept Using DIMTOOL, A380 Scenario, Passenger Service Unit, Illumination Ballast Unit, Handset, FAP

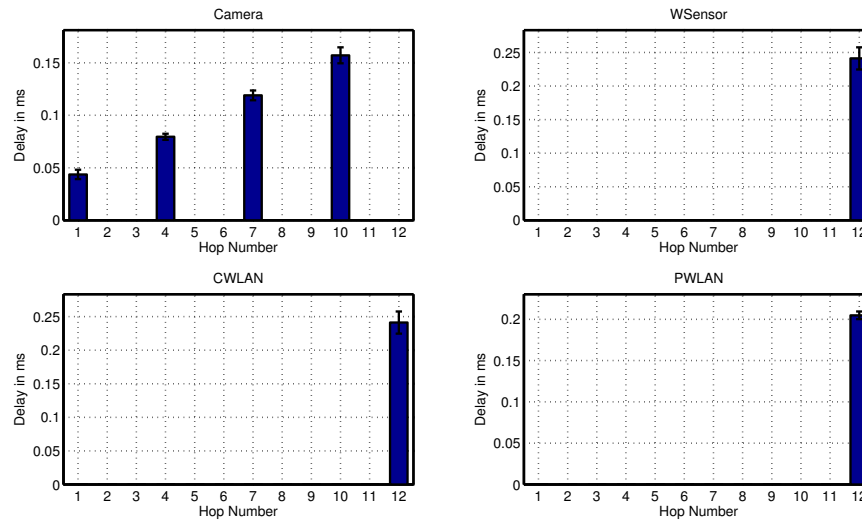


Fig. 6.14: Simulation Results of Multi-Domain Concept Using DIMTOOL, A380 Scenario, Camera, Wireless Sensor, Crew WLAN, Passenger WLAN

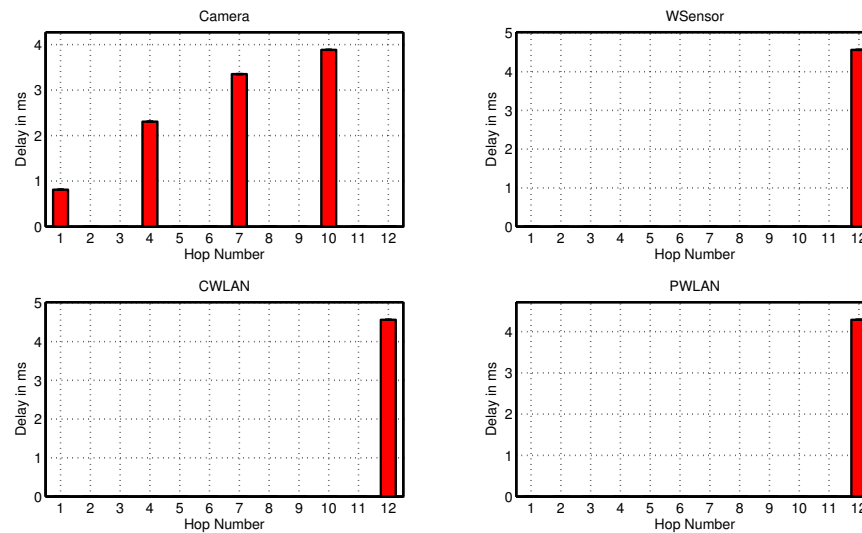


Fig. 6.15: Packetized Non-FIFO NC Results of Multi-Domain Concept Using DIMTOOL, A380 Scenario, Camera, Wireless Sensor, Crew WLAN, Passenger WLAN

6.1.4 Summary

Table 6.4 summarizes the analytical and simulation results concerning the switched Ethernet cabin. We see that both low latency and synchronous playback can only be reached if the backbone runs at Gigabit speed. This is primarily due to the high store-and-forward delay of maximum sized Ethernet frames, which sum up heavily in case of the worst case arrivals. In case of Fast Ethernet, a common time base and rigorous strict prioritization of low latency (handset) packets would be necessary to achieve the requirements. The difficulties of a common time base in DAL-C systems is discussed in Section 6.2.4 when addressing the AVB cabin.

Scenario	Signaling Delay	Audio Delay	Audio Synchrony
Cabin Server	✓	✓	✓
Single Domain CIDS (FE)	✓	-	-
Multiple Domain CIDS (GbE)	✓	✓	✓

Tab. 6.4: Summary of DIMTOOL Performance Bounds

6.2 Realistic Test System

This section shows the performance results as achieved in realistic test setups and cabin mockups. Section 6.2.1 gives an overview of the applied equipment that is used for the performance evaluation. Section 6.2.2 gives a case study on COTS switches that are investigated and evaluated concerning a potential application in the field of cabin communication. Section 6.2.3 shows the performance results that are achieved in a cabin mockup of medium size (6.60 m × 5.85 m, 14 PSUs, 2 handsets). Section 6.2.4 addresses a switched Ethernet cabin that uses AVB to achieve synchronous playback and QoS in terms of latency and differential delay.

6.2.1 Evaluation Setup

The measurements were made using the Anritsu Data Quality Analyzer MD1230B [12] (Figure 6.16), which generated and measured Ethernet data streams. The size and payload of the Ethernet frames can be set on a per-stream basis. To determine packet

Test	Duration[s]
Throughput	60
Latency	120
Frame Loss Rate	60

Tab. 6.5: RFC2544 Default Parameters

reordering, frame loss, and latency, we used the *data pattern TestFrame*, which contains the relevant values, such as the *sequence number* and the *time stamp*, so that we can determine frame loss and latency. Latency is defined as the difference between the time stamp when the first bit of the frame enters the data sink minus the time stamp when the last bit of the frame departs from the source.



Fig. 6.16: Anritsu Data Quality Analyzer MD1230B

The standard RFC2544 [60] specifies a set of tests that determine the performance characteristics of network devices and is provided by the MD1230B. As recommended by the standard, the measurements were made using the standard durations of each trial as given in Table 6.5.

Values for the differential and the absolute audio delay are determined as shown in Figure 6.17. A sine burst generator generates an audio signal at 440 Hz every 200 ms. This audio signal is digitized at the source node and transported via Ethernet to the data sinks. These two data sinks should be positioned so that the delay difference is maximized, i.e., one data sink is very close to the audio source, the other very far away. For these analog measurements we employed the infiniium DS081304B from Agilent that has a sampling rate of up to 40 Gigasamples per second and a bandwidth

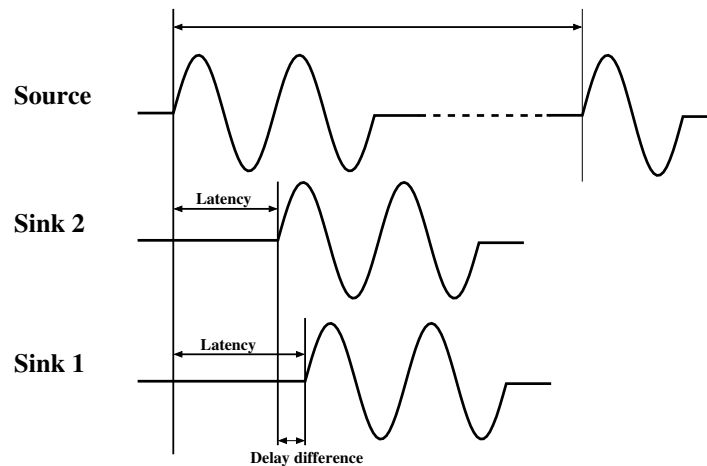


Fig. 6.17: Audio Measurement Setup

of 13 GHz. In addition to this, the infiniium is able to record a histogram of triggered measurements, deriving mean value, median value, and standard deviation.

6.2.2 Case Study of Layer 2/3 COTS Switches

It is a fact that the avionics industry has difficulties when proving and validating new, promising concepts — not only due to the existing high-level standards and certification issues, but also due to the size and complexity of modern aeroplanes. In this field, it is not always possible to provide realistic prototypes — consider the aircraft cabin with typically more than 2000 end devices spread over 22 lines. Avionics bypasses this challenge by an extensive use of simulations, simplified models, and demonstrators. Bearing these constraints in mind, a realistic mapping of simulations and demonstrators is significant when addressing the high accuracy of these simplified models. For this, we address the potential use of COTS components in the aircraft cabin, either managed or unmanaged switches. To clarify the integration of the COTS components, we show the simplified architecture of a managed switch in Figure 6.18. A managed switch usually hosts a switching ASIC, a μ -controller for management tasks, and an EEPROM for storing the configuration data. The switching ASIC provides low level Layer 2 switching of Ethernet packets. The μ -controller addresses the functionality of the switching ASIC usually by the protocols I2C⁴ or SPI⁵.

⁴ Inter-Integrated Circuit

⁵ Serial Peripheral Interface

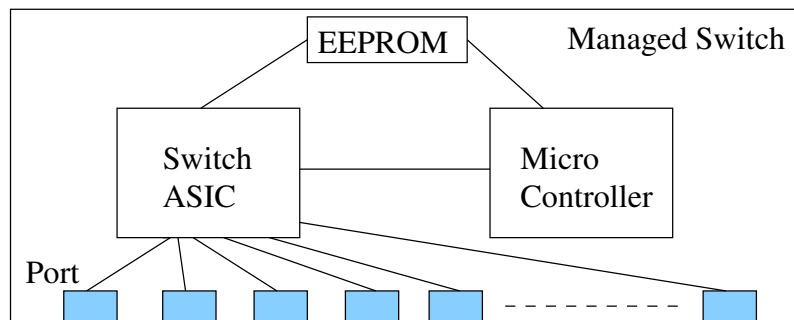


Fig. 6.18: Simplified Managed Switch Architecture

The ASIC devices that are discussed in the remainder of this section limited Layer 3 support, be it the evaluation of the DiffServ Codepoint, the support of IGMP⁶ snooping, or LLDP messages. For more complex functions, such as the mentioned IGMP and LLDP support, an implementation on a μ -controller is necessary. Typical further applications include an SNMP server, a management website, or access via console. The Netgear GS110TP which is addressed in the next section is such a managed solution.

Figure 6.19 shows a switch architecture and gives an insight into state-of-the-art switching ASICs. This architecture covers the major principles found in today's switching ASICs. We analyze [84, 86] and [108]. The classification and filtering block filters or classifies packets according to the header or ingress port id. Traffic policing may be used to limit the incoming traffic or to trigger flow control. In addition, these switches may implement *Virtual Output Queuing* to avoid *Head-Of Line blocking* [108]. For each output port, we set up one virtual output queue at each input port, which is then served by schedulers.

The switch fabric lies at the very core of the switch and tries to forward as many packets as possible. Since $n - 1$ input ports have to be mapped to $n - 1$ output ports, a maximum matching algorithm is likely to be used in those switch implementations (cf. [83]). At the egress level, another classification block determines the queue that is eligible for further handling. After the packet is put in the queue, it waits to be selected by the scheduling algorithms. A scheduling algorithm similar to those introduced in Section 2.2.2 is used to determine the next packet that should be put on the egress port. If shaping is not already provided by the scheduling algorithm, a downstream shaping unit might insert additional time between frames.

⁶ Internet Group Management Protocol

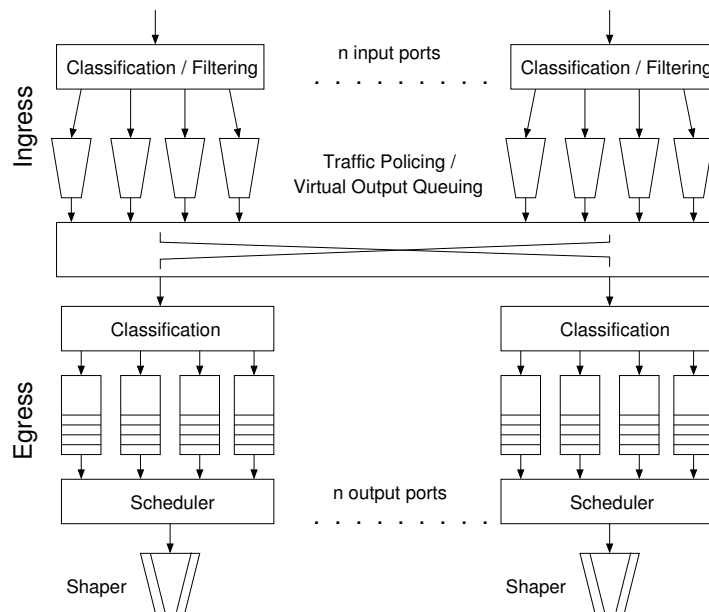


Fig. 6.19: Sample Switch Architecture

Device	Vendor	Type
KS8893MQL [84]	Micrel [85]	ASIC
KS8995MA [86]	Micrel [85]	ASIC
88E6097 [80]	Marvell [81]	ASIC
GS110TP	Netgear [89]	managed
NetFPGA	NetFPGA Project [88]	FPGA

Tab. 6.6: COTS Switches

In the remainder of this section, we address the applicability of the COTS switches listed in Table 6.6. We discuss the features regarding employment in the aircraft cabin, with special focus on the QoS mechanisms introduced in Section 2.2.2.

The Micrel Switch

In this section, we discuss both the 3-port switch KS8893MQL and the 5-port switch KS8995MA. These switching ASICs are located in the low-range price segment. The data sheets of the investigated switches are available without non-disclosure agreements. Apart from the number of ports, there are some differences in terms of switching functionality. Table 6.7 enumerates the major differences between both solutions.

Feature	KS8893MQL	KS8995MA
Number of ports	3	5
Speed	100 Mbit/s	100 Mbit/s
Number of Queues	4	2
Priority Selection	port-based, VLAN, DiffServ	port-based, VLAN DiffServ
Scheduling	SPQ, WFQ	SPQ, WFQ
QinQ	yes	no
Configuration	I2C, SPI, EEPROM	I2C, SPI, EEPROM
Ingress Regulation	Ingress rate limiting	Ingress rate limiting
Egress Regulation	Egress rate shaping	Egress rate limiting

Tab. 6.7: Features of the Micrel Switches KS8893MQL and KS8995MA

Basically the 3-port switch KS8893MQL supports QinQ VLAN tags [47] while the 5-port switch KS8995MA does not. Furthermore, these switches are able to handle more than 16 different VLANs. While the 3-port switch internally distinguishes up to four different priority classes, the 5-port switch distinguishes two priority classes. Having a mixed system leads to the use of two priority classes, which allows handling high and low priority classes differently.

In the following, we use NC to determine the worst case memory consumption in the KS8995MA [86]. We assume the following scenario: Two full speed flows consist of 64byte frames and traversing the switch having the same destination port. These two flows are regulated by a traffic limiter. Figure 6.20 shows the following arrival curves: The composite arrival curve of the incoming flows, the constrained output arrival curve, and the induced memory consumption curve. The arrival curve is thereby built as a pseudo periodic curve (cf. [17]) with period $6.72 \mu s$. The segment is given in Equation 6.1. Multiplication by 2 gives the arrival curve shown in Figure 6.20.

$$\alpha = \max(\delta_{0.64\mu s} \otimes \gamma_{12.5\text{bytes}/\mu s, 0}, 64\text{bytes}) \quad (6.1)$$

We determine the worst case memory consumption, subject to the employed rate limits, by using NC. The results are shown in 6.21. The axes are defined as follows: The x-axis shows the rate limit for Flow 1, the y-axis shows rate limit for Flow 2, and the z-axis shows the maximum backlog.

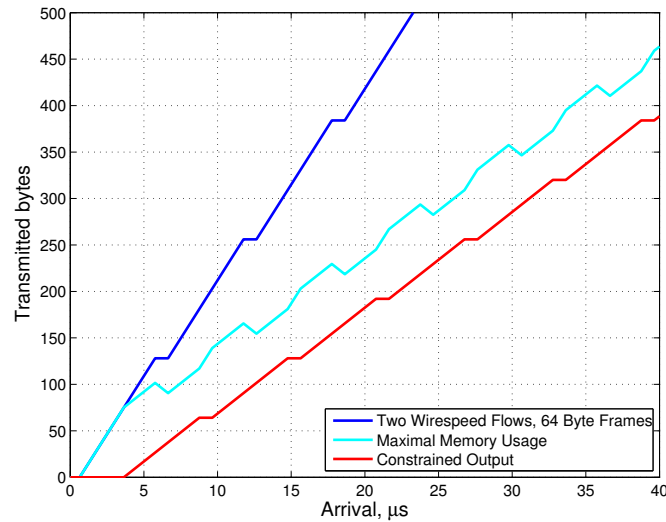


Fig. 6.20: Two Worst Case 64 byte Flows — Memory Consumption

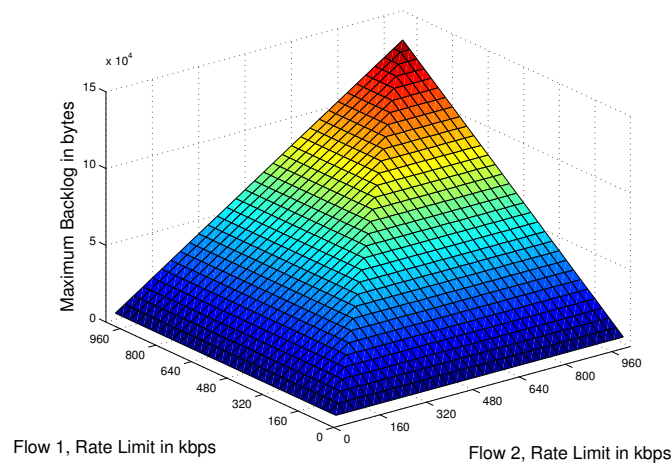


Fig. 6.21: Maximum Memory Usage — Active Rate Limiter

The theoretical considerations are confirmed by a real test setup consisting of one switch, the KS8995MA. Two full-speed flows of 64 byte frames traverse the switch while having the same destination port. Since these flows use *maximum bandwidth* with only one output port, frame drops must occur due to *insufficient buffer space* if the rate limits are not set appropriately, i.e., at a very low level. The results in the test setup are shown in Figure 6.22. The axes are defined as follows: The x-axis is the rate limit for Flow 1, the y-axis shows the rate limit for Flow 2, and the z-axis gives the number of dropped bytes. In contrast to Figure 6.21, Figure 6.22 shows the *dropped bytes* due to *lack of memory*, which is certainly slightly lower than the *hard backlog bound*. The small

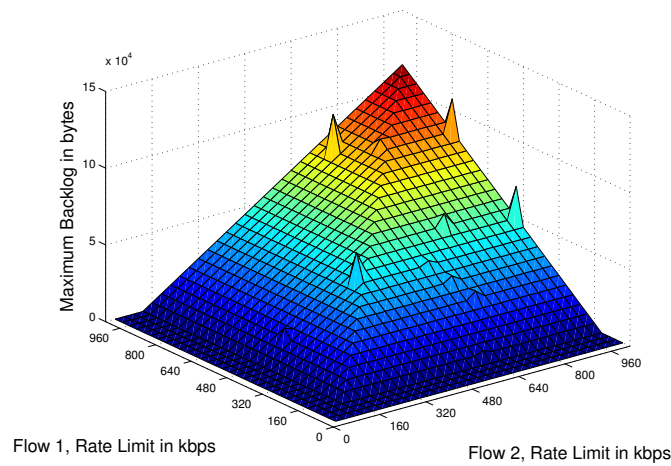


Fig. 6.22: Memory Usage in the Micrel Switch KS8995MA

Packet Size [bytes]	Latency [μ s]	Throughput [%]	Frame Loss Rate [%]
64	13.516	100.00	0.00
128	18.800	100.00	0.00
256	28.950	100.00	0.00
512	49.516	100.00	0.00
1024	90.398	100.00	0.00
1280	110.870	100.00	0.00
1518	130.068	100.00	0.00

Tab. 6.8: RFC2544 Test Results for KS8995MA

band, where no frame dropping occurs in the real test setup, is between 32 kbit/s and 96 kbit/s. Since $\frac{1}{5}$ of the buffer memory (512 kbit) is reserved for each output port, rate limits of up to 100 kbit induce no frame dropping in the output buffer. We can draw an important *conclusion* here: Since traffic limiting cannot prevent the frame dropping induced by *cross traffic*, it is not suitable for employment in the SCMS except, possibly, as an additional policing unit.

Table 6.8 shows the results for RFC2544 of the KS8995MA at a line rate of 100 Mbit/s. We obtain a processing delay ranging from 8.396 μ s for a minimum sized frame to 8.628 μ s for a maximum sized frame.

The switches KS8893MQL [84] and KS8995MA [86] are Layer 2/3 switching ASICs. When addressing their use in the SCMS, the limited number of ports provided (three or five) is a major drawback of these components. The ASICs provide up to 100 Mbit/s

Feature	88E6097
Number of Ports	11
Speed	$8 \times 10/100\text{Mbit/s}$, $3 \times 10/100/1000\text{Mbit/s}$
Number Queues	4
Priority Selection	port-based, VLAN, DiffServ
Scheduling	SPQ, WFQ, programmable weights
QinQ	yes
Configuration	I2C, SPI, EEPROM
Ingress Regulation	Ingress rate limiting
Egress Regulation	Egress rate shaping

Tab. 6.9: Features of Marvell 88E6097

full duplex. According to the performance studies in Section 6.1, a Gigabit backbone is necessary to satisfy the requirements of Section 2.1. This excludes the use of these switches in the backbone. Another drawback is the limited number of distinguishable VLAN addresses. Up to 16 VLAN tags can be handled by the VLAN table. The switches provide rudimentary QoS support as scheduling algorithms and traffic regulation. However, none of these mechanisms is able to prevent a faulty or bubbling device from disturbing other devices of the same safety domain. Compared to the 88E6097, which will be discussed in the next section, the QoS mechanisms can not be adjusted as precisely.

The Marvell Switch

In this section, we address and evaluate the Marvell switch 88E6097. This device provides *eight Fast Ethernet* ports and *three Gigabit Ethernet* ports. Table 6.9 shows the features of this switching ASIC.

The Marvell 88E6097 is a switching ASIC with several hundreds of functions. The switch can distinguish up to 4096 VLAN entries, i.e., it can handle the full range of VLAN tags. The switch provides QoS support in the form of scheduling algorithms and traffic regulation. Compared to the Micrel switches of the last section, the QoS mechanisms are finer-grained. Again, these mechanisms are not appropriate for stopping faulty or bubbling devices. Employment in the SCMS backbone is therefore not recommended.

Packet Size [bytes]	Latency [μ s]	Throughput [%]	Frame Loss Rate [%]
64	15.176	100.00	0.00
128	37.408	100.00	0.00
256	30.692	100.00	0.00
512	51.176	100.00	0.00
1024	92.128	100.00	0.00
1280	112.612	100.00	0.00
1518	131.644	100.00	0.00

Tab. 6.10: RFC2544 Test Results for 88E6097

Table 6.10 shows the test results for RFC2544. We see a processing delay ranging from 10.056 μ s for a 64 byte Ethernet frame to 10.204 μ s for a full-sized 1518 byte Ethernet frame.

Netgear Switch GS110TP

The Netgear GS110TP is a managed switch, and a structure similar to the scheme given in Figure 6.19 is assumed. According to [43], the GS110TP hosts two Broadcom BCM59101 Quad integrated IEEE 802.3af-compliant controllers as well as a Broadcom BCM53312 8-Port GbE + 4-Port GbE highly integrated multilayer switch. The latter information has been confirmed by analyzing the interior of the GS110TP. Table 6.11 shows the features of the GS110TP. The egress regulation is provided on a per queue and a per port basis. The queues are served by the scheduling algorithms SPQ and WFQ to achieve given minimum bandwidth. Additionally, the egress port can be shaped to a specific bandwidth by adjusting the IFG. Table 6.12 shows the test results for RFC2544. We see a processing delay ranging from 3.04 μ s to 3.76 μ s.

Up to 64 VLAN tags can be distinguished by the switch. The GS110TP provides state-of-the-art QoS mechanisms, more precisely: SPQ, WFQ, bandwidth reservation, and egress shaping. The μ -controller provides an SNMP implementation as well as LLDP. Due to the presence of bandwidth reservation, SNMP, and LLDP, we use this managed switch in the remainder of this section in the cabin demonstrator. However, the *lack of ingress regulation mechanisms* does not encourage the employment of this device in the SCMS.

Feature	Netgear GS110TP
Number of Ports	10
Speed	$8 \times 10/100/1000\text{Mbit/s}$, $2 \times \text{SFP } 1000\text{Mbit/s}$
Number Queues	4
Priority Selection	port-based, VLAN, DiffServ
Scheduling	SPQ, WFQ (programmable bandwidth)
QinQ	yes
Configuration	SNMP, LLDP, Web Interface
Ingress Regulation	—
Egress Regulation	Egress rate shaping

Tab. 6.11: Features of Netgear GS110TP

Packet Size [bytes]	Latency [μs]	Throughput [%]	Frame Loss Rate [%]
64	8.160	100.00	0.00
128	14.000	100.00	0.00
256	23.560	100.00	0.00
512	44.960	100.00	0.00
1024	85.880	100.00	0.00
1280	105.440	100.00	0.00
1518	125.120	100.00	0.00

Tab. 6.12: RFC2544 Test Results for GS110TP

NetFPGA

The NetFPGA is a line-rate, flexible, and open platform for research [88]. In this work, the NetFPGA 10G will be used as a platform to study scheduling algorithms. Today, there is an EADS internal switching IP⁷ core, which is able to forward packets at line-rate, do traffic shaping (input and output), and carry out VLAN forwarding rules. The base of the NetFPGA 10G is a Xilinx Virtex-5, and up to four SFP+ (Small Form-factor Pluggable) interfaces are supported in this constellation. Table 6.13 shows the test results for RFC2544. Due to the ingress shaping capability, we observe double latency since frames traverse two cascaded queues. As a consequence, we see a processing delay ranging from 2.506 μs to 4.782 μs .

⁷ Intellectual Property

Packet Size [bytes]	Latency [μ s]	Throughput [%]	Frame Loss Rate [%]
64	3.530	100.00	0.00
128	4.640	100.00	0.00
256	6.894	100.00	0.00
512	11.384	100.00	0.00
1024	20.771	100.00	0.00
1280	24.869	100.00	0.00
1518	29.070	100.00	0.00

Tab. 6.13: RFC2544 Test Results for NetFPGA

Compared to the switching ASICs, the NetFPGA is a flexible solution and can provide the desired QoS mechanisms shaped to the actual needs. We suggest the following QoS and configuration mechanisms:

- Ingress traffic shaping
- Full-range VLAN table
- Simplified SNMP alike configuration protocol
- Topology scan based on LLDP

Yet the need for additional effort in the VHDL development process must be taken into consideration. To overcome this additional effort, a possible solution is to employ IP cores directly from established manufacturers such as Xilinx or Marvell. A cost-benefit analysis would have to be applied to determine the more economical solution. In essence, a hybrid approach will be considered, i.e., we use an FPGA solution in the backbone to get a certain level of protection, and *low cost switching* ASICs inside the safety domains.

6.2.3 Cabin Demonstrator

Figure 6.23 shows the basic network demonstrator that was used to validate the SCMS concept by measurements. The switches H, S1-S12, and T1-T2 are Netgear GS110TPs. Up to eight end devices are served by the switches (typically one handset and seven others, such as PSU, IBU, or Camera). The end-to-end latency is identified by measurements at switch S12, as it is the farthest from the server. The multicast delay difference

is identified by subtracting the latency of the packets which have arrived at S1/T1 from the latency experienced in node S12.

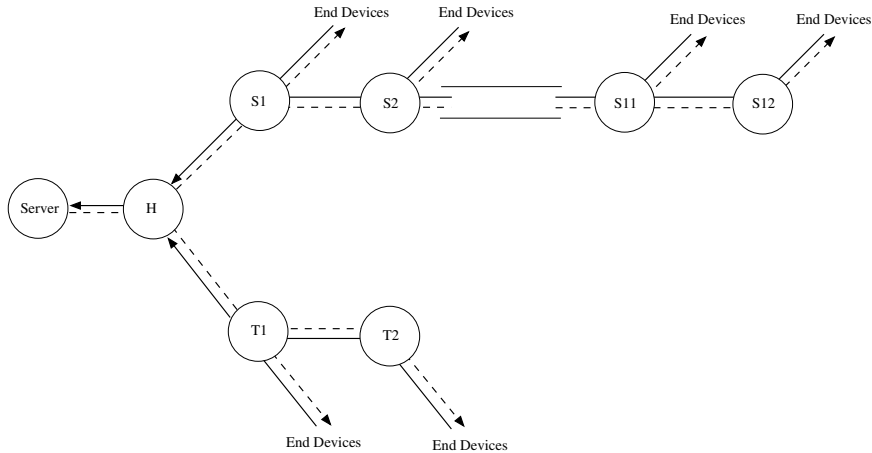


Fig. 6.23: Topology of the SCMS Demonstrator

Table 6.14 shows the traffic pattern employed, as well as the assigned QoS priorities. We employ the following test scenarios in the SCMS demonstrator:

- *No cross traffic*: The subject is the performance of one handset stream. No cross traffic exists.
- *High cross traffic*: We consider the performance of one handset stream. Two full wirespeed (100 MBit/s and 1 GBit/s) streams with maximum packet size flow through the network. One stream covers the upstream path, the other stream covers the downstream path.
- *Heavy cross traffic*: We consider the performance of one handset stream. Two full wirespeed (100 MBit/s and 1 GBit/s) streams at maximum packet size flow through the network in one direction.

Device	Packet Size	Bandwidth [kbit/s]	VLAN Priority	Queue Priority
PSU	108	27	4	2
IBU	108	27	4	2
Handset	142	568	7	3
Camera	$\leq 64, \geq 1518$	5000	0	1

Tab. 6.14: Traffic Pattern and Priorities in SCMS Demonstrator

- *Extreme cross traffic*: The subject is the performance of one handset stream. In addition to *heavy cross traffic*, 11 additional handset streams also having the highest priority flow through the network.
- *Best effort traffic*: The camera ports are subject to the RFC2544 [60] test routine while PRAM and PA at one handset is active.⁸

In the remainder of this performance evaluation concerning the SCMS demonstrator, we determine the performance bounds with respect to latency and multicast delay difference. Each of these measurement sessions lasted 120 s and takes the latency into account. In this way, we can determine not only the end-to-end latency, but also the multicast delay difference, which is important for the PA use case.

- *Latency S12 → S1*: The latency is determined between switch S12 and switch S1.
- *Latency S12 → Srv → S12*: The latency is determined between switch S12 and switch S12. Each packet is sent to the server and reinserted by the server Srv.
- *Multicast Delay Difference S12–T1*: The multicast delay difference is determined by subtracting the receive time stamp at switch S12 from the receive time stamp at switch T1.

Figures 6.24 and 6.25 show the results for latency measurements in the SCMS demonstrator. We address 100 MBit/s and 1 GBit/s. Both the latency and multicast delay difference requirements are fulfilled. However, the multicast delay difference requirement in the case of Fast Ethernet is near the maximum value.

Figure 6.26 shows the histogram for the measurement of S12→S1 in the case of the Fast Ethernet and Figure 6.27 shows the histogram for the measurement of S12→Srv→S12 in the case of the Fast Ethernet. The variability in the *Extreme cross traffic* scenario shows the limits of Fast Ethernet in the SCMS scenario regarding multicast delay difference.

Figure 6.28 shows the histogram for the measurement of S12→S1 in the case of Gigabit Ethernet. Figure 6.29 shows the histogram of the measurement of S12→Srv→S12 in the case of Gigabit Ethernet. The values in the Gigabit backbone confirm the applicability of Gigabit Ethernet in the SCMS — both requirements are fulfilled, providing a broad safety margin.

⁸ The camera ports are located at switches S8 and T2

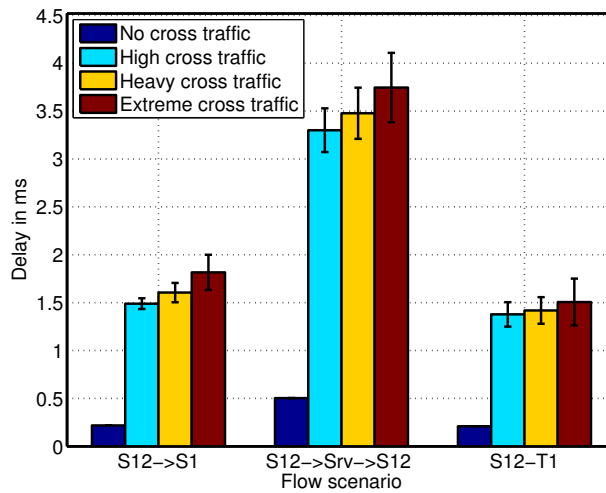


Fig. 6.24: Measurement Results in the SCMS Demonstrator, Fast Ethernet

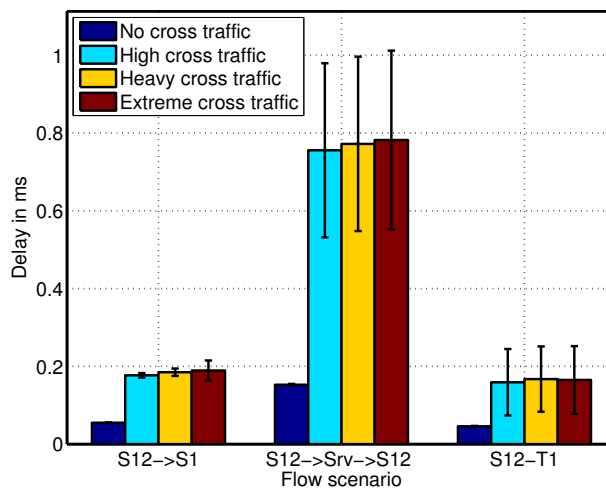


Fig. 6.25: Measurement Results in the SCMS Demonstrator, Gigabit Ethernet

In order to investigate the bandwidth remaining for DAL-E traffic in the cabin network, we summarize the results for the RFC2544 test. This kind of traffic is forwarded with the lowest priority. However, we still require good QoS for DAL-E functions, since this type of traffic will likely be used to implement IFE or high level passenger services. Table 6.15 summarizes the results for the left over service for DAL-E traffic. We can safely devote 90 % of the overall bandwidth to DAL-E traffic.

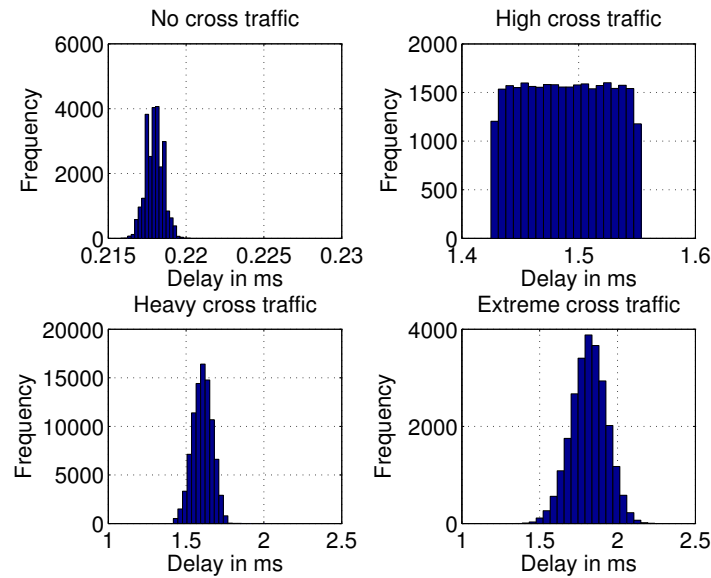


Fig. 6.26: Latency Histogram in SCMS Demonstrator, S12→S1, Fast Ethernet

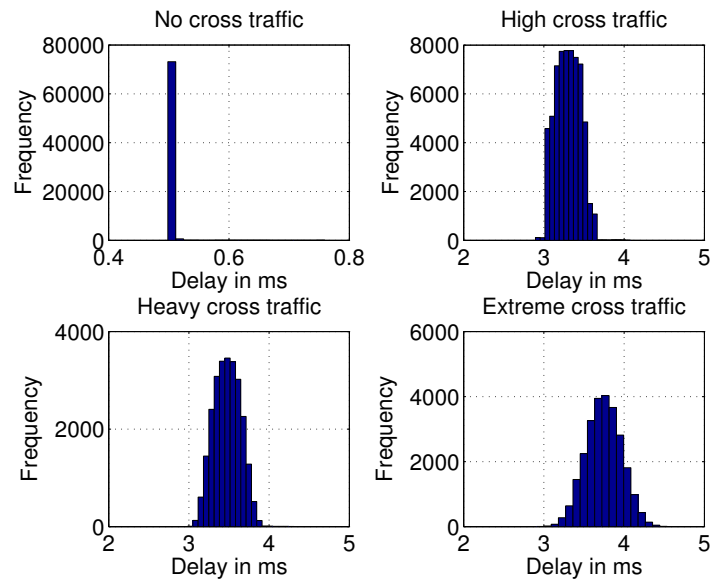


Fig. 6.27: Latency Histogram in SCMS Demonstrator, S12→Srv→S12, Fast Ethernet

6.2.4 Applicability of AVB in the Avionic Context

Today there are various standards and approaches to *guarantee synchronous playback* via different communication systems, such as Apple's AirPlay, Cobranet, and AVB. The aeronautics industry is watching these modern and promising solutions and is trying to

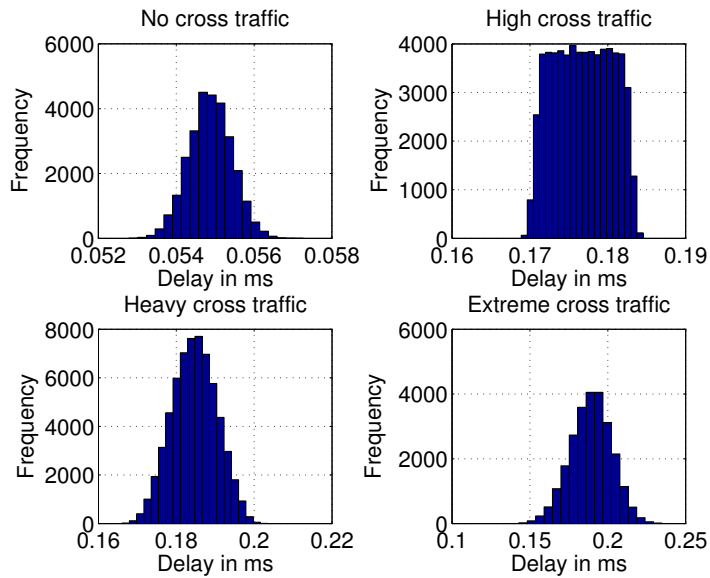


Fig. 6.28: Latency Histogram in SCMS Demonstrator, S12→S1, Gigabit Ethernet

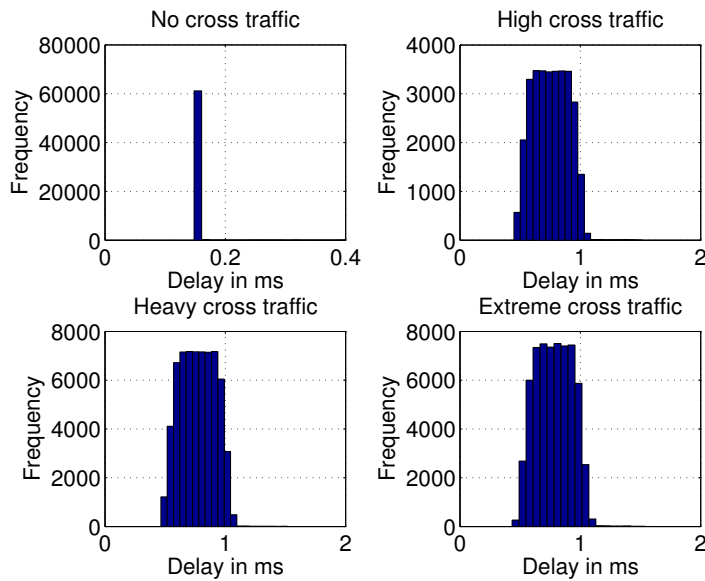


Fig. 6.29: Latency Histogram in SCMS Demonstrator, S12→Srv→S12, Gigabit Ethernet

benefit from the ongoing research and cutting edge technologies, while always taking into consideration how such solutions can fulfill the stringent safety requirements.

To the best of our knowledge, addressing the employment of AVB in an avionics context has not been addressed before. However, there is some previous research in the

Packet Size [bytes]	Latency [μ s]	Throughput [%]	Frame Loss at 100 % Rate [%]
64	422.874	93.00	5.28
128	299.366	95.00	4.22
256	397.924	96.50	3.15
512	535.516	97.00	2.46
1024	818.466	97.50	2.10
1280	1050.642	98.50	2.02
1518	1210.964	98.50	1.95

Tab. 6.15: RFC2544 Test Results for DAL-E traffic, Gigabit Ethernet

field of *industrial Ethernet* and *automotive applications*. Imtiaz et al. [55] discuss the use of AVB in industrial real-time communications and carry out a comprehensive comparison between standard Ethernet and AVB. Lim et al. [78] address the employment of AVB in an Ethernet based in-car network that covers rear seat entertainment and driver assistance. They introduced an AVB model for in-car networks and showed by network simulation that the accuracy remains better than one μ s, as required by IEEE 802.1AS. Kern et al. [70] discuss clock synchrony over two hops and focus on AVB synchrony in a varying temperature range. The results were promising in terms of accuracy, which ranged in the low double digit ns. Garner et al. showed in [39] that the achieved accuracy of gPTP⁹ in a Gigabit network with a maximum of seven hops is better than ± 500 ns. The related research basically addresses the accuracy of clock-synchronization when using AVB-enabled bridges. In contrast, we address AVB over Layer 2 Ethernet switches having no AVB support.

AVB Ethernet Cabin

In this section, we address the applicability of AVB to the aircraft cabin network. Specifically, the goal of this section is to study the performance bounds with respect to audio latency and audio synchrony between different audio sinks with a special focus on the given avionic requirements. In addition, we discuss the proposed technology in terms of its dependability and how to certify it, taking into consideration the demands of avionics.

In the introduction chapter, we already mentioned the safety relevance of cabin com-

⁹ generalized Precision Time Protocol

munications. We again refer to the demonstrator topology shown in Figure 6.23. This mockup shows an Ethernet based solution of the wired aircraft cabin, but as opposed to the approach discussed in the last section, we now use AVB to provide synchronous playback. We will now briefly recall the architecture of the SCMS: The tree topology consists of several physical networks, which run largely in parallel along the aisles. Each network in turn consists of about a dozen daisy-chained Ethernet switches. Typically, we find the following device types attached to those switches that cover the CIDS functionality in an aircraft cabin: (a) PSU, (b) IBU, (c) cabin handset, (d) FAP, (e) CVMS, and (f) smoke detectors. Typically, the number of end devices that are served by the CIDS range from 1000 to 2000 devices in the larger aircraft such as the A380. The concrete number is highly dependent on the configuration as given by the specific airline.

The measurements were carried out as follows: We employed the AVB Audio Endpoint Kit by XMOS, which provides a complete open source AVB stack [120]. At the input of the talker, we put a sine audio pulse, which was repeated every 200 ms. The end-to-end delay was determined by comparing the audio signal at the input of the talker with the audio signal at the output of the listener. The following measurements were accomplished: end-to-end delay between the talker and the listener, and the differential delay between two listeners.

We again refer to Figure 6.17, which gives a comprehensive overview of the measurements carried out. During the experiment, we put additional traffic load on the path from the Talker to Listener 1 and increased the load successively from 0 % to 100 % in steps of 10 %. At each measurement step, *we recorded 1200 samples*. The observed results are evaluated and compared to the DO-214 standard, which recommends minimum standards for aircraft audio systems (microphones, headsets, handsets and loudspeakers) [98] as described in Section 2.1.1.

In this part, we discuss the performance results. Figure 6.30 shows the achieved latency in the cabin network covering both synchronized and unsynchronized AVB. The abscissa shows the percentage of additional low priority traffic that impedes AVB traffic. The ordinate gives the observed latency. Using *clock synchronization*, we achieve promising values for the overall latency from the talker to the listener. We observed delay values of about 2.7 ms, with a 95 %-confidence interval of about $\pm 500 \mu\text{s}$. This latency contains the AVB presentation delay, which is fixed to 2 ms by the AVB standard. When losing synchronization, we observe a latency of about 5 ms, which comes from the *internal playback buffer*, which can hold up to 5 ms of audio data.

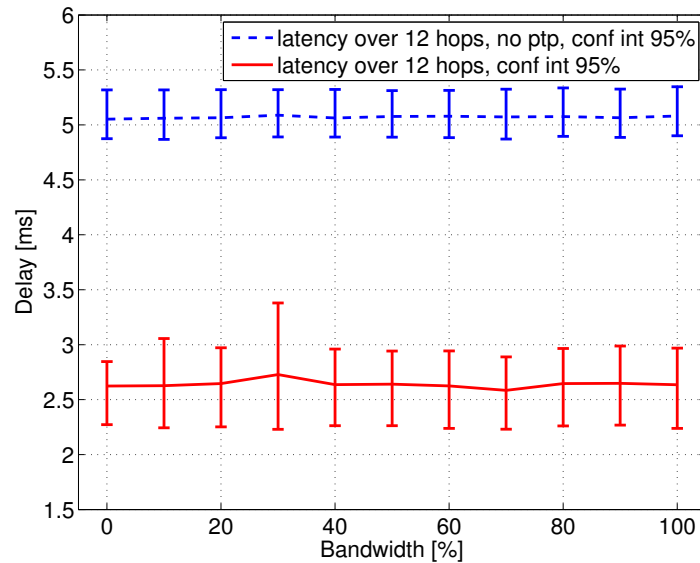


Fig. 6.30: End-to-end Delay Aircraft Cabin

Figure 6.31 shows the achieved differential delay of AVB in the cabin scenario. The observed differential delay in the case of synchronized AVB satisfies the given requirement of the differential playback's being lower than 1 ms. Otherwise, in the case of unsynchronized AVB, we observe differential delays from 3.5 ms to 4.5 ms. Due to the large playback buffer, this significantly exceeds the multicast delay requirement

The performance study shows that losing synchronization in the AVB cabin must be avoided. This implies that there is needed a large effort in software development and the following certification processes.

Safety Aspects of AVB Time Synchronization

As mentioned earlier in this section, AVB basically consists of *four standards*. The *transport protocol* and *pacing* will be relatively straight forward regarding certification. The transport protocol has a well-defined and comprehensible structure so that an implementation with sufficient certification-relevant artifacts should be possible with minor effort. Pacing or traffic shaping on the other side is a well studied field in computer networks, so that scheduling and shaping algorithms achieving certain QoS (such as CBQ) are reliable and well-understood due to their vast distribution. With respect to the stream reservation protocol SRP, industry agrees that dynamic bandwidth reser-

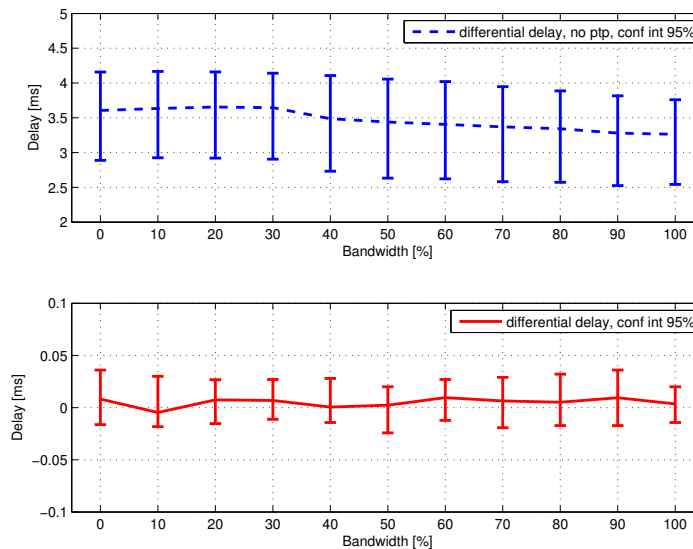


Fig. 6.31: PTP Differential Delay Aircraft Cabin

vation is too complex for safety-relevant networks. A common approach is to disable SRP completely and to specify bandwidth reservations a priori. Such restrictions are common in avionics *in order to limit analysis and test scenarios*. However, with respect to achieving certification of implementations adhering to these standards, the chosen parameters specified for QoS have to be valid and correct. This requires significant analysis efforts, but may be doable.

When we consider the standards mentioned, time synchronization with IEEE 802.1AS remains the last standard we have to address. As does PTP itself, gPTP allows the existence of several master clocks in an AVB network. As defined in IEEE 802.1AS, gPTP client nodes use the BMCA¹⁰ in order to determine the so-called grandmaster of the system. BMCA determines the quality of the master clocks by some parameters, such as the variance of time deviation, or the typical accuracy and class of a master. The following major most intuitive failure cases are summarized:

- Client nodes (masters) send false information in BMCA phase
- Client nodes (masters) send no information in BMCA phase at all
- Client nodes send inconsistent information in BMCA phase
- Two or more clients play grandmaster role (master clock)

¹⁰ Best Master Clock Algorithm

A compromised or faulty grandmaster can propagate a false global time. However this will have no practical relevance if done consistently because each talker and listener will synchronize with the false time. So we will not observe any difficulties with respect to synchronous and slightly delayed playback. However, a client node that sends inconsistent information (or sends information that is inconsistently perceived by receivers) is more critical than no or false but consistently perceived information. Inconsistently perceived information is just as serious an issue as having two or more active grandmasters. In such cases, several talkers and listeners can obtain different global times, which will have a direct impact on the given requirements with respect to synchrony and low latency. In the worst case, agreement is not at all possible, as has been explained in well known literature [74]. Local monitoring techniques may help to limit such scenarios to acceptable levels in practice.

When we address time synchronization in avionics networks, we come across the certification requirements, especially when software algorithms play a major role. Using synchronization with AVB implies the employment of COTS elements regarding time-aware bridges and hardware time stamping. Following [33], we summarize the activities that are generally required to certify COTS elements. For COTS elements, we distinguish between simple, complex, and highly complex integrated circuits or microcontrollers. This classification has a direct impact on the test and verification procedures that have to be covered and whether the intended safety level is achievable with the addressed COTS element at all. The devices used will likely be in the category *complex* for synchronization. Besides collecting and evaluating device data like data sheet and errata sheets, we also have to identify the manufacturing process and the configuration management. In addition to that, the *Product Service Experience* (PSE) has to be verified, i.e., whether the COTS element has sufficient PSE *for the respective* DAL. For the different safety levels the respective instructions have to be employed in order to determine PSE. To draw a short conclusion: A significant amount of effort has to be considered when COTS elements are intended for use in the field of aeronautics networks, where often detailed data from manufacturers is needed. This requirement cannot be easily fulfilled by any COTS vendor.

Similar to the previously mentioned hardware classification and validation, the avionics industry provides standardized processes covering safety aspects of software. Since we will use a software implementation of gPTP, special attention has to be paid to DO-178 [97]. When thinking about the employment of a global synchronization protocol such as PTP or gPTP in safety-critical networks, a significant expense has to be

made for ensuring the correctness of the protocol mechanisms and their implementation. Aerospace standards determine design rules and design processes, so that an implementation of PTP or gPTP can attain certification. This requires rigorous testing, generating certification artifacts (like a sound requirements base and code reviews), and/or rewriting the existing ready-to-use implementations. When regarding the time-awareness of intermediate AVB bridges, we not only have to consider the talker and listener, but also the implementation in intermediate switches. The certification process that would be necessary to go through in order to employ time synchronization in avionics networks will lead to significant costs and additional efforts in order to ensure the correctness and completeness of the implementation and algorithm for its intended use in the avionics architecture.

In conclusion, the efforts required will be highly dependent on the level of assurance needed. E.g., the CIDS is DAL-C in certain configurations [44, 76]. According to [107], DAL-C systems require a maximum failure requirement of 10^{-5} failures/hour. The actual reliability numbers are highly dependent on the failure probabilities of the devices for the failure modes described above. These probabilities are derived from implementation approaches and, hence, cannot be known at this stage. For the purpose of this exercise, we focus on the masters and we assume consistently and inconsistently perceived failures.

Inconsistently perceived *malicious failures* are likely to be device dependent, and could be on the order of 10^{-6} to 10^{-8} failures/hour according to our experience. Consistently perceived *benign failures* are much more likely and could be on the order of 10^{-5} to 10^{-6} failures/hour. Assuming there are three servers in the CIDS for redundancy purposes, any *benign failure mode* can be tolerated for all architectures as the resulting system failure rate derives from two masters failing in order to have one master still being available, which is 10^{-10} failures per hour. For the *malign failure* mode, any failure of the three masters can lead to a system failure, i.e., $3 \cdot 10^{-6}$ to $3 \cdot 10^{-8}$, depending on the master failure rate. As a consequence, AVB may be a *candidate* for DAL-C systems, but further investigations from a reliability perspective are necessary. It should be noted that any real system needs to be evaluated in much more detail, taking into consideration the actual implementation and deployment. However, certification of AVB's synchronization algorithm leads to significant, additional efforts in the certification process.

6.2.5 Summary

In this chapter, we addressed the switched aircraft cabin in realistic test setups. We studied the performance and features of suitable COTS switches of different manufacturers. We covered suitable ASICs and managed switches from Micrel, Marvell, and Netgear. In addition, we briefly introduced our own switch implementation, based on the NetFPGA platform. We pointed out that such a switch implementation based on VHDL might be necessary due to the lack of features in the COTS switches studied. Furthermore, we addressed the applicability of switched Ethernet to aeronautics networks and proved the requirements by a real mockup. We showed that Gigabit Ethernet is necessary to achieve the requirements given in Section 2.1 if no global synchronization is available. We also addressed the applicability of AVB to aeronautics networks and showed that the requirements are basically met when using AVB with time synchronization, even in the case of Fast Ethernet. Compared to the promising results of AVB in terms of audio synchrony and low delays, we pointed out some difficulties when using AVB in the field of aeronautics networks. With respect to certification issues, we see that refraining from time-awareness in Layer 2 switches saves effort in the certification process while still complying with the high-level requirements.

7. CONCLUSION AND OUTLOOK

In this thesis, we have presented a novel design for an aircraft cabin network. The approach is based on switched Ethernet and allows benefiting from recent progress in communications technology. Probably the *most important factor* for the usability of switched Ethernet in automotive and aeronautics networks is the BroadR-Reach™ [20] technology. BroadR-Reach™ allows transmitting 100 Mbit/s full-duplex *over two wires*. Compared to standard Fast Ethernet, which requires four wires to transmit 100 Mbit/s full-duplex, this technology allows of a significant weight reduction. As a result, enabling the aircraft cabin network to use switched Ethernet also for the safety relevant functions has been a key motivation for this thesis. These novel technologies also provide more bandwidth, which can then be used by other applications such as *In-Flight-Entertainment* or Passenger WLAN.

Such a novel approach certainly influences the certification process. The certification authorities, such as EASA¹ and FAA², demand a strict adherence to aviation standards, such as DO-178 for software development [97], DO-254 for hardware development [99], and DO-214 for audio system characteristics [98]. Especially the latter has a direct impact on an aircraft cabin system based on switched Ethernet, since the *maximum worst case latencies* and *Quality of Service* must be guaranteed. *Quality of Service* and worst case scenarios can be determined by common techniques such as network simulation and measurements. However, only analytical approaches are able to determine the hard guarantees that are required in the certification process. A common technique that is used in switched queuing networks is *Network Calculus*, whose tightness is still subject to ongoing research. It has even been shown that determining tight bounds in general networks is NP-hard.

When applying *Network Calculus* to switched Ethernet networks, we found that the widely applied *fluid flow models* have drawbacks when mapping discrete bursts as they occur in those networks. We have proposed a novel technique to determine the worst

¹ European Aviation Safety Agency

² Federal Aviation Administration

case by *Mixed Integer Programming*. We improved the tightness of these bounds in our scenarios by about 5%. Thereby we utilized the variability of micro bursts that cannot be captured by the *Network Calculus* approaches. Compared to other *Model Checking* approaches, we benefitted from the nature of solving *Mixed Integer Programming*: When the computation of the exact worst case bound is too time consuming, we stop the cut-and-branch solver after a certain amount of time and at least determine a safe upper bound that is better than the corresponding *Network Calculus* bound.

The novel approach has been integrated with the proposed performance evaluation tool DIMTOOL, which in turn provides several performance evaluation techniques in a consolidated toolchain. This allows rapid performance evaluations of different aircraft cabin layouts. The analytical results found can easily be verified by network simulations that are also provided by this toolchain. The current version of DIMTOOL provides the following backends: *Network Calculus*, *MIP approach*, *Worse Case Simulation*, and network simulation. The DIMTOOL platform also provides tools for system integration and deployment tasks. These tools are even independent of the actual *Quality of Service* implementations. As a consequence, the methodology developed here can be employed in future cabin designs and does not rely on specific hardware.

This thesis brings the following benefits for the aviation industry: The switched Ethernet cabin is able to guarantee *low audio* and *signaling delay* in the high priority traffic. The remaining capacity can be used by traffic with lesser requirements in terms of safety and reliability, such as *In-Flight-Entertainment*. Compared to established systems, this is an enormous advantage. We have also shown that we can safely employ standard Ethernet COTS hardware and limit the use of FPGAs to the backbone.

Further improvements and evaluations of the presented methods and techniques are thinkable. On the one hand, we would like to investigate stochastic extensions to the *Network Calculus* and thereby especially focus on an employment in the comfort domains. On the other hand, we would like to provide techniques that speedup the *MIP approach*.

Nowadays, the *Network Calculus* community seems to be moving forward towards stochastic methods in the *Network Calculus*. The integration of this *Stochastic Network Calculus* would be a major topic for further extensions to the DIMTOOL. In the *Stochastic Network Calculus*, network traffic that bursts with a *certain severity*, bounded by a *given probability*, has a direct counterpart. In the *Deterministic Network Calculus* it is the peak rate burst that acts as a major reason for unduly pessimistic bounds. To get reasonable

bounds with the *Deterministic Network Calculus*, we thus have to employ rigorous traffic shaping. This is certainly a good idea for the safety relevant network traffic in the aircraft cabin but it may be conservative for DAL-E traffic such as *In-Flight-Entertainment*. This statistical multiplexing gain is an essential property of packet-switched networks and can be taken into account with the *Stochastic Network Calculus* [23].

The *MIP approach* is based on an NP-hard problem. Since Bouillard et al. [16] showed that determining tight bounds in general networks is NP-hard, the employment of *Mixed Integer Programming* does not mean a significant limitation. On the one hand, it is generally believed that solving those problems is inherently difficult. On the other hand, there may be good heuristics that deliver satisfactory approximations. In the *MIP approach*, it is essential to quickly find upper bounds to the maximizations problem. In cut-and-branch techniques, this upper bound is found by adding several cutting planes that do not change the search space of the integer solutions. Sophisticated cut-generators for quickly adding tight cutting planes to the original LP relaxation would be a promising direction for future research. However, we saw that the deterministic *Network Calculus* bounds may already be sufficient in many cases. More precisely, we saw that the requirements of the aircraft cabin can be fulfilled.

At the *system level*, there are two directions that promise significant benefits. On the one hand, engineering efforts are required to make the *novel aircraft cabin system* ready to market. This system has to be developed and certified. Serious statements about cost and weight savings are feasible when this stage has been reached. Rigorous testing will have to take place. Additionally, we studied FIFO and *Strict Priority Queuing* to guarantee the *Quality of Service* performance bounds in the aircraft cabin. For those which are safety related, high priority traffic would be a good idea, since we encounter guaranteed and well-shaped traffic. On the other hand, lower priority traffic such as passenger WLAN and *In-Flight-Entertainment* may encounter starvation issues. The problem of starvation is often faced by sophisticated scheduling algorithms, flow control, and active queue management. Ongoing work will consider those techniques for the aircraft cabin of the future.

Appendices

Appendix A

DIMTOOL TOOLCHAIN

The DIMTOOL is a platform for performance evaluations and performance calculations to be used for the development of a novel aircraft cabin based on switched Ethernet. This toolbox not only provides several performance estimation techniques to determine performance bounds, but also routines and algorithms for cabin dimensioning and deployment.

Figure A.1 shows the architecture of the DIMTOOL toolchain. The *Topology and Flow Description* lies at the very core of the DIMTOOL suite. The CSV based description is forwarded to the Matlab interface, which in turn forwards the description to selected performance evaluation backends. At the time of writing, the following backends exist: network simulation, *Network Calculus* analysis, *Worse Case Simulation*, and the *MIP approach* introduced in Section 4. The topology may be created by a *Topology Generator*, which provides standard cabin layouts for Airbus airlines such as the A380. From the topology we can then derive the actual switch configuration. This switch configuration is deployed with suitable management protocols such as SNMP.

Figure A.2 shows the UML class diagrams of the DIMTOOL package. The class *Network-Calculation* is the basic class where all calculations and evaluations start from. Devices and links are represented by *Nodes* and *Edges* and are stored in the aggregated *Topology*. The theoretical traffic models introduced in Section 2.2.2 have their counterpart in the classes inherited from the *Flow* class. Figure A.3 shows the UML class diagrams inherited from the *Node* class. These classes are in turn serialized and deserialized using an XML representation.

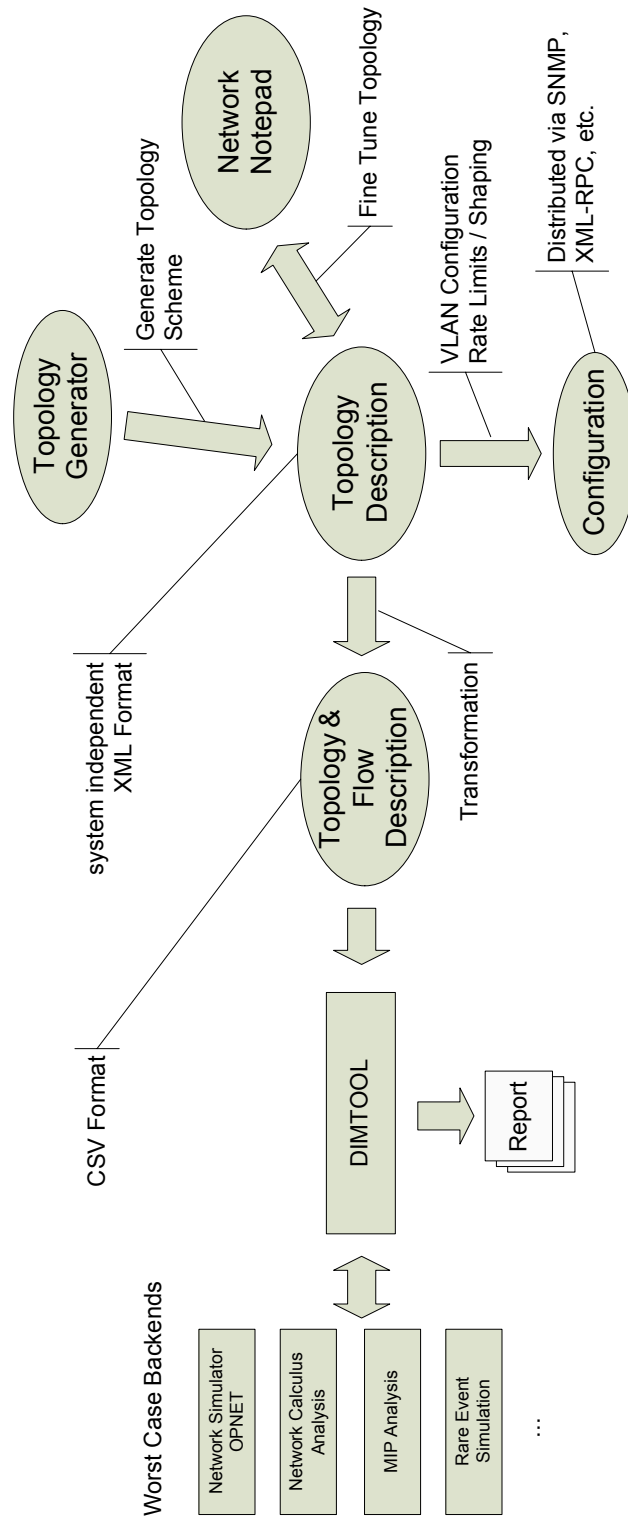


Fig. A.1: Toolbox DIMTOOL

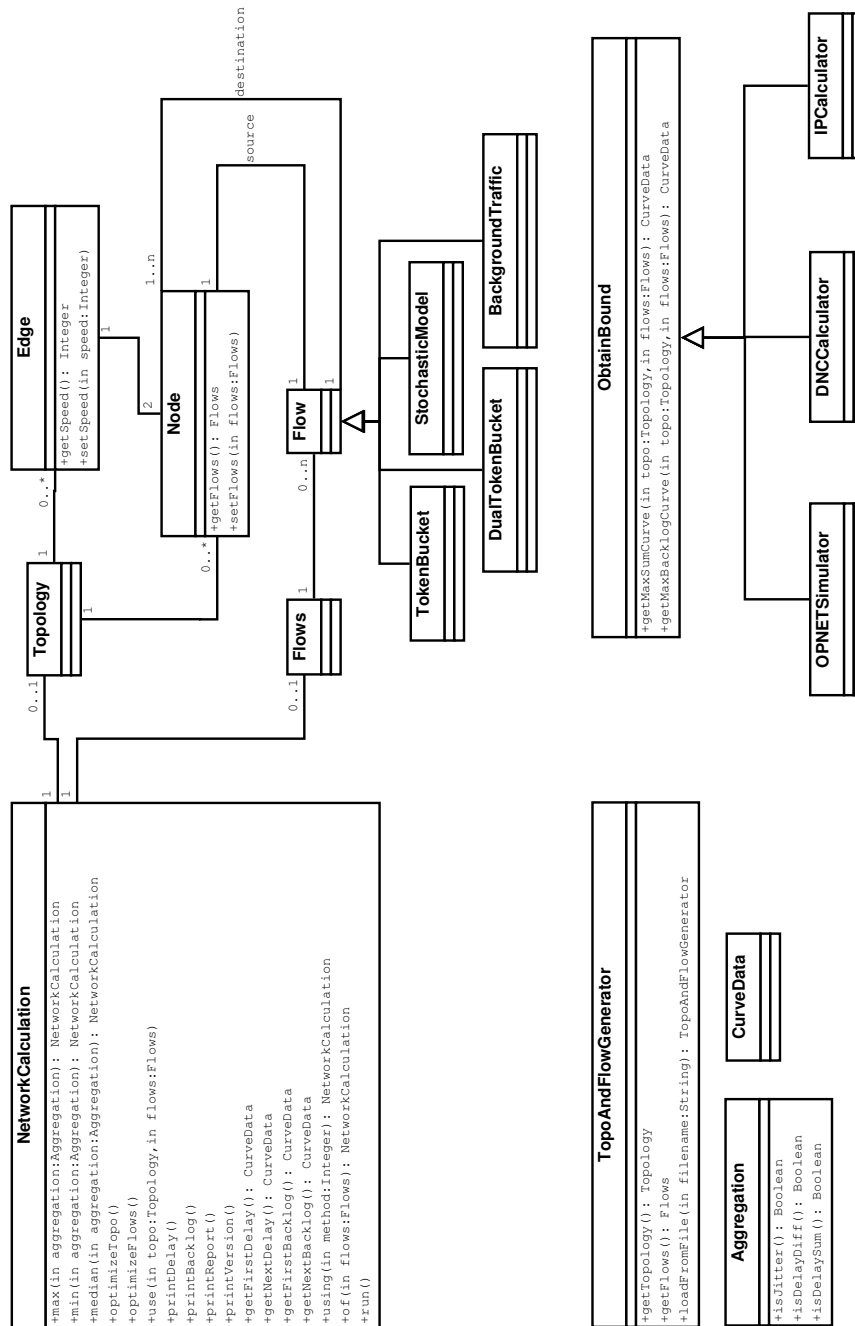


Fig. A.2: DIMTOOL Network Calculation Package

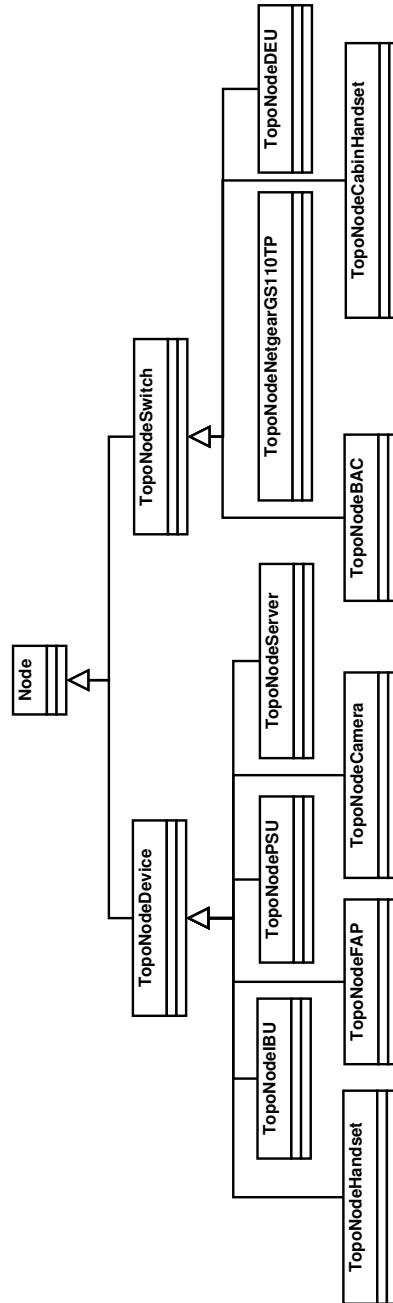


Fig. A.3: DIMTOOL Node Package

Appendix B

NETWORK CALCULUS BOUNDS

Appendix B gives some illustrative examples for the network bound calculation for the sample networks discussed in Section 4.5. They are again given in Figures B.1 and B.2. The alternative DB- and RL-approaches are discussed. Tables B.1–B.4 show the NC calculations using the different approaches. The exact values determined by a model checking approach are 0.2480 ms for the tandem network and 0.02912 ms for the feed-forward network. The transmission speed is 100 Mbit/s, i.e., $0.08 \mu\text{s}$ per transmitted byte. Note that *none of the state-of-the-art NC approaches* can determine that the 64 byte frame can only be delayed once by the 108 byte frame in the feed-forward network.

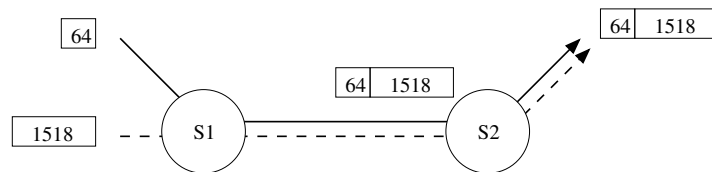


Fig. B.1: Tandem Scenario with Two Flows

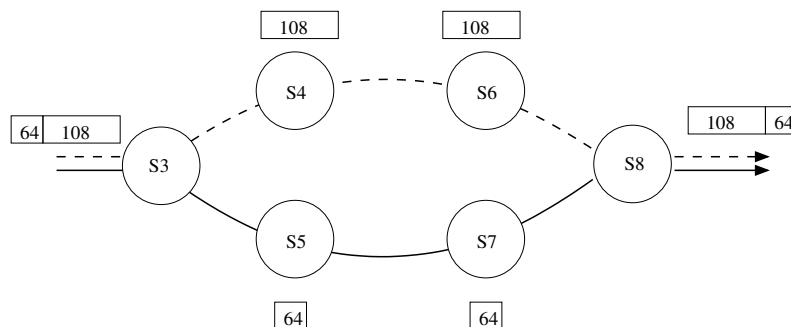
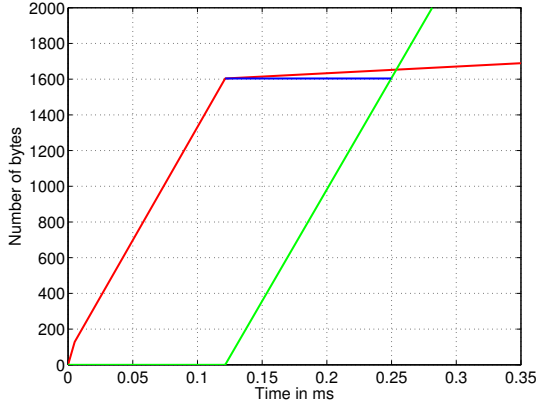
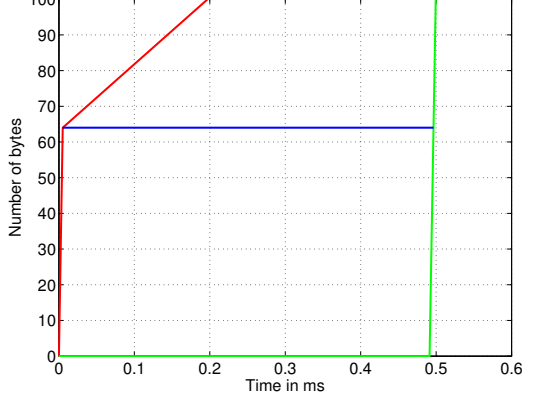
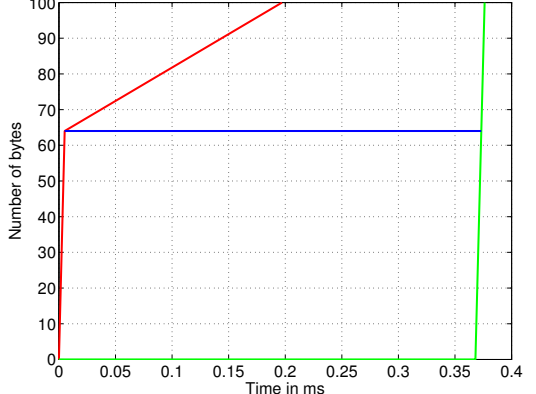


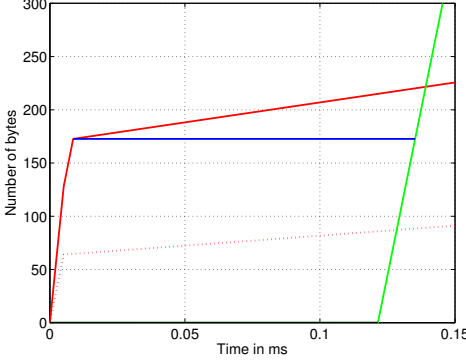
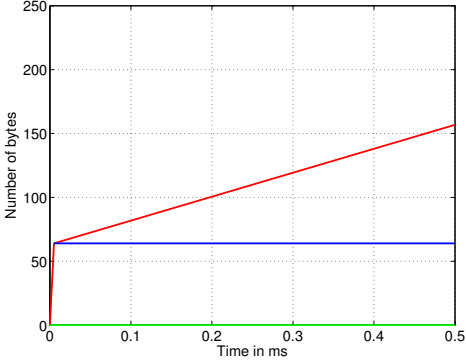
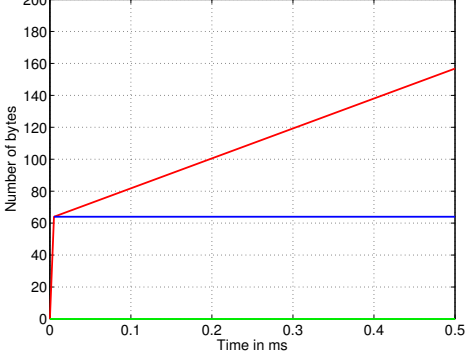
Fig. B.2: Feed-forward Network with Two Flows

TFA	
	$- f = f_1 + f_2 \quad - g = s_1, h = s_2 \quad - v(f, g), v(f, h)$
	$v(f, g) + v(f, h) = 0.2566\text{ms}$
SFA	
	$- f = f_2 \quad - g = [s_1 - f_1]^+ \otimes [s_2 - \ominus [s_1 - f_1]^+]^+$
	$- v(f, g) = 0.4914\text{ms}$
PMOO-SFA	
	$- f = f_2 \quad - g = [s_1 \otimes s_2 - f_1]^+$
	$- v(f, g) = 0.3681\text{ms}$

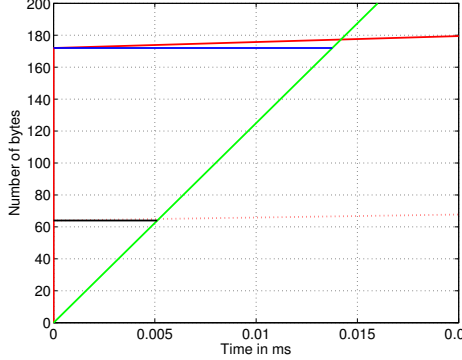
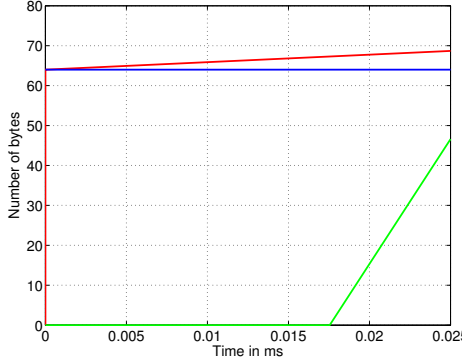
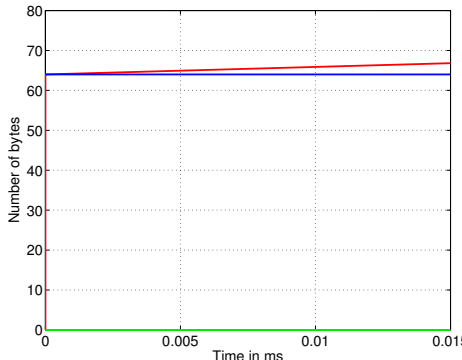
Tab. B.1: Network Calculus Results for Tandem Scenario, RL Approach

TFA	
	$- f = f_1 + f_2 \quad - g = s_1, h = s_2 \quad - v(f, g), v(f, h)$
	$v(f, g) + v(f, h) = 0.2531\text{ms}$
SFA	
	$- f = f_2 \quad - g = [s_1 - f_1]^+ \otimes [s_2 - \ominus [s_1 - f_1]^+]^+$
	$- v(f, g) = 0.2518\text{ms}$
PMOO-SFA	
	$- f = f_2 \quad - g = [s_1 \otimes s_2 - f_1]^+$
	$- v(f, g) = 0.1285\text{ms}$

Tab. B.2: Network Calculus Results for Tandem Scenario, DB Approach

TFA	
	<p> $\color{red}- a_1 = a_3 = f_1 + f_2$ $\color{red}\cdots a_2 = f_1$ $\color{green}- g_1 = s_3, g_2 = s_4, g_3 = s_5, g_4 = s_6$ $\color{blue}- v(a_1, g_1), v(a_3, g_4)$ $\color{black}- v(a_2, g_3)$ </p>
	$v = 0.4961\text{ms}$
SFA	
	<p> $\color{red}- f_2$ $\color{green}- g = [s_3 - f_3]^+ \otimes s_4 \otimes [s_6 - (f_3 \otimes s_3 \otimes s_5)]^+$ </p>
	$\color{blue}- v(f_2, g) = 0.5124\text{ms}$
PMOO-SFA	
	<p> $\color{red}- f_2$ $\color{green}- g = [s_3 \otimes s_4 \otimes s_6 - f_1]^+$ </p>
	$\color{blue}- v(f_2, g) = 0.5019\text{ms}$

Tab. B.3: Network Calculus Results for Feed-forward Network, RL Approach

TFA	
	<p> $\text{--- } a_1 = a_3 = f_1 + f_2$ $\text{... } a_2 = f_1$ $\text{--- } g_1 = s_3, g_2 = s_4, g_3 = s_5, g_4 = s_6$ $\text{--- } v(a_1, g_1), v(a_3, g_4)$ $\text{--- } v(a_2, g_3)$ </p>
	$v = 0.0378\text{ms}$
SFA	
	<p> $\text{--- } f_2$ $\text{--- } g = [s_3 - f_3]^+ \otimes s_4 \otimes [s_6 - (f_3 \otimes s_3 \otimes s_5)]^+$ $\text{--- } v(f_2, g) = 0.0278\text{ms}$ </p>
PMOO-SFA	
	<p> $\text{--- } f_2$ $\text{--- } g = [s_3 \otimes s_4 \otimes s_6 - f_1]^+$ $\text{--- } v(f_2, g) = 0.0284\text{ms}$ </p>

Tab. B.4: Network Calculus Results for Feed-forward Network, DB Approach

Appendix C

TOPOLOGY AND FLOW DESCRIPTION

Listing C.1 shows the syntax of the *Topology and Flow Description* in EBNF. The *Topology and Flow Description* was introduced in Section 5.2.

```
tf      = topo '#' flows '#';
topo    = "topo" "nodes" number node*;
node    = number ',' type ',' '{' '}'';
flows   = "flows" flow*;
flow    = int ',' string ',' '{' <curves> '}'';
curves  = (curve ',' <curves>) | <curve>;
curve   = tb_5 | tb_4 | tb_3 | tb_2 | tb_1 | dtb_5 | dtb_4 |
         dtb_3 | dtb_2 | dtb_1;
starttime = double;
prio    = int;
tb      = "TokenBucket" ',' double ',' double ',' <ip>;
tb_1    = <tb>;
tb_2    = <tb> ',' <starttime>;
tb_3    = <tb> ',' <distribution> ',' '(' <par> ')';
tb_4    = <tb> ',' <distribution> ',' '(' <par> ',' <par> ')';
tb_5    = <tb> ',' <distribution> ',' '(' <par> ',' <par> ')';
         ',' <prio>;
dtb     = "DualTokenBucket" ',' double ',' double ',' double
         ',' double ',' <ip>;
dtb_1   = <dtb>;
dtb_2   = <dtb> ',' <starttime>;
dtb_3   = <dtb> ',' <distribution> ',' '(' <par> ')';
dtb_4   = <dtb> ',' <distribution> ',' '(' <par> ',' <par> ')';
dtb_5   = <dtb> ',' <distribution> ',' '(' <par> ',' <par> ')';
         ',' <prio>;
```

Listing C.1: Topology and Flow Description (1), EBNF syntax

Appendix D

CABIN DEMONSTRATOR

Figures D.1–D.3 show the cabin mockup that was used while approving *Technology Readiness Level* (TRL) 4. The term TRL is used in avionics to describe the readiness of a novel technology or concept. It is the established development process that brings novel technology and concepts into the Airbus programs. TRLs range from 1 to 9 where 1 is the lowest and 9 is the highest technology level. This TRL review shows the safety related functions, such as *Passenger Address* and *Prerecorded Audio*. At TRL 4, a mockup is mandatory to show the basic functioning in a laboratory environment.



Fig. D.1: Cabin Mockup

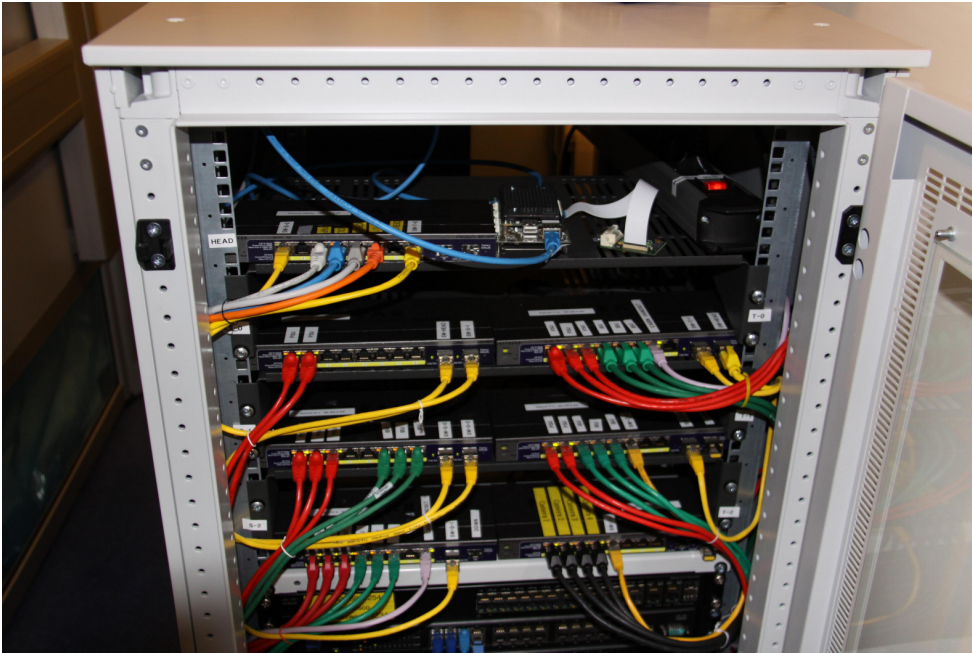


Fig. D.2: Cabin Server Rack



Fig. D.3: Passenger Service Unit

Appendix E

PUBLICATIONS BY THE AUTHOR

Parts of this thesis have been published:

Chapter 4:

- E. Heidinger, N. Kammenhuber, A. Klein, and G. Carle. Network Calculus and Mixed-Integer LP Applied to a Switched Aircraft Cabin Network. In *Proceedings of the 20th International Workshop on Quality of Service, IWQoS 2012, Coimbra*, pages 1–4, 2012.
- E. Heidinger. Rare Events in Network Simulation Using MIP. In *Proceedings of the 23rd International Teletraffic Congress. ITC 2011, San Francisco*, pages 314–315, 2011.

Chapter 5:

- E. Heidinger, S. Burger, S. Schneelee, A. Klein, and G. Carle. DIMTOOL: A platform for determining worst case latencies in switched queuing networks. In *Proceedings of the 6th International Conference on Performance Evaluation Methodologies and Tools, ValueTools 2012, Cargese*, pages 1–9, 2012.
- E. Heidinger, S. Schneelee, A. Klein, and G. Carle. Modeling Aeronautic Networks for Internet Scenarios. In *Proceedings of the 12th Workshop on IP, Euroview 2012, Wuerzburg*, pages 1–2, 2012.

Chapter 6:

- E. Heidinger, F. Geyer, S. Schneelee, and M. Paulitsch. A Performance Study of Audio Video Bridging in Aeronautic Ethernet Networks. In *Proceedings of the 7th IEEE International Symposium on Industrial Embedded Systems, SIES 2012, Karlsruhe*, pages 67–75, 2012.

LIST OF ABBREVIATIONS

ACD *Aircraft Control Domain.* 11, 40, 42–44, 82, 92

AFDX *Avionics Full Duplex Ethernet.* 1, 2, 15, 16, 52

AISD *Airline Information Services Domain.* 11, 40, 44, 82, 92

ARINC *Aeronautical Radio Incorporated.* 1, 88

AVB *Audio-Video-Bridging.* 25, 26, 28–30, 100, 116–119, 121–123

CBQ *Credit Based Queuing.* 25, 26, 28, 119

CIDS *Cabin Intercommunication Data System.* 4, 9–13, 39, 40, 42, 47, 70, 90, 100, 118, 122

CIL *Cabin Illumination.* 88, 89

COTS *commercial-of-the-shelf.* 2–4, 16, 21, 28, 39, 41, 73, 87, 90, 100, 102, 104, 121, 123, 126

CVMS *Cabin Video Monitoring System.* 10, 11, 40, 42, 45, 46, 118

DAL *Design Assurance Level.* 7, 8, 39, 100, 114, 117, 121, 122, 127, 159

DB *discrete burst.* 37, 50, 61–64, 135

DEU *Decoder Encoder Unit.* 12, 13, 43, 44, 82

EMA *External Model Access.* 77

FAL *Final Assembly Line.* 70

FAP *Flight Attendant Panel.* 40, 44–46, 82, 89, 93, 118

FQ *Fair Queuing.* 25

- gPTP* generalized Precision Time Protocol. 27, 28, 120–122
- IBU* Illumination Ballast Unit. 11–13, 40, 42, 44–46, 82, 93, 111, 118
- IFE* In-Flight-Entertainment. 2, 4, 7, 9, 11, 88, 89, 114
- LI* Reading Lights. 88, 89
- LP* Linear Programming. 53, 54, 58, 60, 64, 127
- MC* Model Checking. 71
- MIP* Mixed Integer Programming. 3, 5, 49, 53–56, 58–60, 62, 64, 67, 78, 86
- NC* Network Calculus. 3–5, 14, 15, 18, 30, 31, 34, 35, 37, 49–52, 59–62, 64, 69, 70, 72, 76–78, 80, 86, 88–93, 105, 135
- PA* Passenger Address. 8, 11, 42, 88, 89, 113
- PBOO* Pay Bursts Only Once. 35, 77
- PIESD* Passenger Information and Entertainment Service Domain. 11, 40, 44, 82, 92
- PKT* packetizer. 37
- PMOO* Pay Multiplexing Only Once. 35, 77
- PMOO-SFA* Pay Multiplexing Only Once SFA. 35, 50, 61–64, 67, 136–139
- PODD* Passenger-Owned Devices Domain. 11, 40, 44, 92
- PQ* Priority Queuing. 26
- PRAM* Prerecorded Audio. 11, 88, 89, 113
- PSE* Product Service Experience. 121
- PSU* Passenger Service Unit. 8, 9, 11–13, 39, 40, 42–46, 82, 91–93, 100, 111, 118
- PTP* Precision Time Protocol. 26–28, 79, 120–122
- QoS* Quality of Service. 3, 4, 7, 16–18, 24, 37–39, 41, 45, 100, 104, 108, 109, 111, 112, 114, 119, 120

RL *rate latency*. 37, 50, 61–64, 135

RTC *Real-Time Calculus*. 60

SCMS *Switched Cabin Management System*. 39, 45, 47, 72, 73, 83, 84, 87, 107–109, 111–116, 118, 154, 159

SFA *Separated Flow Analysis*. 34, 35, 61–64, 67, 89, 136–139

SPQ *Strict Priority Queuing*. 25, 47, 105, 108–110

TFA *Total Flow Analysis*. 34, 35, 61–64, 67, 91, 136–139

TRL *Technology Readiness Level*. 143

VLAN *Virtual LAN*. 2, 17, 18, 22, 38, 39, 41, 44–47, 73, 79, 83–86, 105, 108–111

WCS *Worse Case Simulation*. 71

WFQ *Weighted Fair Queuing*. 25, 105, 108–110

LIST OF FIGURES

2.1	Schematic of Aircraft Cabin, Courtesy of Airbus	12
2.2	Arbitrary Multiplexing in Switches	15
2.3	Cumulative Arrival, FIFO Bound Versus Non-FIFO Bound	16
2.4	Ethernet Header	18
2.5	Token Bucket Model	19
2.6	Dual Token Bucket Model	20
2.7	On/Off Traffic Pattern	20
2.8	VLAN Table Entry	22
2.9	Traffic Shaping	24
2.10	Traffic Limiter	25
2.11	AVB Transport Protocol Frame [93]	27
2.12	PTP Synchronization (Adapted From [54])	27
2.13	The AVB Output Port	29
2.14	Credit-Based Shaper Queuing and Scheduling Operation [50], as in [55]	30
3.1	Single Domain Cabin, Courtesy of Airbus	40
3.2	Multi-Domain Cabin, Courtesy of Airbus	41

4.1	Small packet follows larger packet	50
4.2	Effectless Burst	50
4.3	250 byte packet prior to 500 byte packet	51
4.4	500 byte packet prior to 250 byte packet	51
4.5	Graphical Representation	54
4.6	Tandem Scenario with Four Flows	57
4.7	Find Worst Case Between Node i and j for Packet p_1 , 100 Mbit/s	58
4.8	Tandem Scenario with Two Flows	61
4.9	Feed-forward Network with Two Flows	63
4.10	Analysis of Switched Cabin Network—Upstream	65
4.11	Comparison of LP Relaxation and MIP Bound	66
4.12	Analysis of Switched Cabin Network—Downstream	66
5.1	Performance Calculation with DIMTOOL	69
5.2	CIM Interop Model	74
5.3	Tandem Scenario with Four Flows	75
5.4	Toolbox DIMTOOL	78
5.5	Class NetworkCalculation	79
5.6	Cabin Configurator Workflow	86
6.1	Intra-Server Communication	88
6.2	Analysis of CS Using DIMTOOL	89
6.3	Analysis of SCMS using DIMTOOL	91

6.4	Simulation Results of Multi-Domain Concept using DIMTOOL, A30x Scenario, Passenger Service Unit, Illumination Ballast Unit, Handset, FAP	94
6.5	Packetized Non-FIFO NC Results of Multi-Domain Concept using DIMTOOL, A30x Scenario, Passenger Service Unit, Illumination Ballast Unit, Handset, FAP	94
6.6	Simulation Results of Multi-Domain Concept using DIMTOOL, A30x Scenario, Camera, Wireless Sensor, Crew WLAN, Passenger WLAN	95
6.7	Packetized Non-FIFO NC Results of Multi-Domain Concept using DIMTOOL, A30x Scenario, Camera, Wireless Sensor, Crew WLAN, Passenger WLAN	95
6.8	Simulation Results of Multi-Domain Concept Using DIMTOOL, A350 Scenario, Passenger Service Unit, Illumination Ballast Unit, Handset, FAP	96
6.9	Packetized Non-FIFO NC Results of Multi-Domain Concept Using DIMTOOL, A350 Scenario, Passenger Service Unit, Illumination Ballast Unit, Handset, FAP	96
6.10	Simulation Results of Multi-Domain Concept Using DIMTOOL, A350 Scenario, Camera, Wireless Sensor, Crew WLAN, Passenger WLAN	97
6.11	Packetized Non-FIFO NC Results of Multi-Domain Concept Using DIMTOOL, A350 Scenario, Camera, Wireless Sensor, Crew WLAN, Passenger WLAN	97
6.12	Simulation Results of Multi-Domain Concept Using DIMTOOL, A380 Scenario, Passenger Service Unit, Illumination Ballast Unit, Handset, FAP	98
6.13	Packetized Non-FIFO NC Results of Multi-Domain Concept Using DIMTOOL, A380 Scenario, Passenger Service Unit, Illumination Ballast Unit, Handset, FAP	98
6.14	Simulation Results of Multi-Domain Concept Using DIMTOOL, A380 Scenario, Camera, Wireless Sensor, Crew WLAN, Passenger WLAN	99

6.15 Packetized Non-FIFO NC Results of Multi-Domain Concept Using DIM-TOOL, A380 Scenario, Camera, Wireless Sensor, Crew WLAN, Passenger WLAN	99
6.16 Anritsu Data Quality Analyzer MD1230B	101
6.17 Audio Measurement Setup	102
6.18 Simplified Managed Switch Architecture	103
6.19 Sample Switch Architecture	104
6.20 Two Worst Case 64 byte Flows — Memory Consumption	106
6.21 Maximum Memory Usage — Active Rate Limiter	106
6.22 Memory Usage in the Micrel Switch KS8995MA	107
6.23 Topology of the SCMS Demonstrator	112
6.24 Measurement Results in the SCMS Demonstrator, Fast Ethernet	114
6.25 Measurement Results in the SCMS Demonstrator, Gigabit Ethernet	114
6.26 Latency Histogram in SCMS Demonstrator, S12→S1, Fast Ethernet	115
6.27 Latency Histogram in SCMS Demonstrator, S12→Srv→S12, Fast Ethernet	115
6.28 Latency Histogram in SCMS Demonstrator, S12→S1, Gigabit Ethernet	116
6.29 Latency Histogram in SCMS Demonstrator, S12→Srv→S12, Gigabit Ethernet	116
6.30 End-to-end Delay Aircraft Cabin	119
6.31 PTP Differential Delay Aircraft Cabin	120
A.1 Toolbox DIMTOOL	132
A.2 DIMTOOL Network Calculation Package	133
A.3 DIMTOOL Node Package	134

B.1 Tandem Scenario with Two Flows	135
B.2 Feed-forward Network with Two Flows	135
D.1 Cabin Mockup	143
D.2 Cabin Server Rack	144
D.3 Passenger Service Unit	144

LIST OF TABLES

2.1	Design Assurance Level (Adapted From [99])	8
2.2	Requirements in CIDS	11
2.3	Aircraft Cabin Domain	12
2.4	Number of Devices in the A380 Family	13
2.5	Aircraft Cabin Management System — Summary	13
2.6	Common Curves Used in Network Calculus [75]	36
2.7	Performance Bounds — Summary	38
3.1	Traffic Profiles in the Aircraft Cabin	42
3.2	Single Domain Configuration	43
3.3	Multiple-Domain Configuration	44
3.4	A30x VLAN Assignment in Each Line	45
3.5	A350 VLAN Assignment in Each Line	46
3.6	A380 VLAN Assignment in Each Line	46
3.7	Class Based VLAN Assignment	46
4.1	Token Bucket Parameters, Four Flows	58
4.2	Worst Case Send/Receive Times for Tandem Network (Fig. 4.6)	59

4.3	Token Bucket Parameters for Tandem and Feed-forward Network	60
4.4	Worst Case Send/Receive Times for Tandem Network (Fig. 4.8)	61
4.5	Results for Tandem Scenario	62
4.6	Worst Case Send/Receive Times for Feed-forward Network (Fig. 4.9)	63
4.7	Results for Feed-forward Network	64
4.8	Token Bucket Parameters in Aircraft Cabin	65
4.9	Comparison of Different Worst Case Approaches	67
5.1	Performance Estimation Tools	72
5.2	Token Bucket Parameters	75
5.3	Input Parameters for the Topology Generator	82
6.1	Parameters in Intra-Server Communication	89
6.2	Parameters in Single Domain Aircraft Cabin	91
6.3	Parameters in Multi-Domain Aircraft Cabin	93
6.4	Summary of DIMTOOL Performance Bounds	100
6.5	RFC2544 Default Parameters	101
6.6	COTS Switches	104
6.7	Features of the Micrel Switches KS8893MQL and KS8995MA	105
6.8	RFC2544 Test Results for KS8995MA	107
6.9	Features of Marvell 88E6097	108
6.10	RFC2544 Test Results for 88E6097	109
6.11	Features of Netgear GS110TP	110

6.12	RFC2544 Test Results for GS110TP	110
6.13	RFC2544 Test Results for NetFPGA	111
6.14	Traffic Pattern and Priorities in SCMS Demonstrator	112
6.15	RFC2544 Test Results for DAL-E traffic, Gigabit Ethernet	117
B.1	Network Calculus Results for Tandem Scenario, RL Approach	136
B.2	Network Calculus Results for Tandem Scenario, DB Approach	137
B.3	Network Calculus Results for Feed-forward Network, RL Approach	138
B.4	Network Calculus Results for Feed-forward Network, DB Approach	139

BIBLIOGRAPHY

- [1] M. Adnan, J. Scharbarg, J. Ermont, and C. Fraboul. An improved timed automata model for computing exact worst-case delays of AFDX periodic flows. In *Proceedings of the 16th Conference on Emerging Technologies & Factory Automation, ETFA 2011*, pages 1–4. IEEE, 2011.
- [2] Aeronautical Radio, Inc. Draft 2 of ARINC project paper 819: Avionics digital audio distribution. Technical Report Project Paper 819, 2551 Riva Road, Annapolis, MD 21401, USA, Feb 2007.
- [3] Airbus. A320 Family: A318, A319, A320, A321 - A320 photos, pictures, A320 videos, A320 3D view — Airbus — Airbus, a leading aircraft manufacturer. Website, Accessed 2012-09-13. <http://www.airbus.com/aircraftfamilies/passengeraircraft/a320family/>.
- [4] Airbus. A350 XWB: A350-800, A350-900, A350-1000 - A350 photos, pictures, A350 videos, A350 3D view — Airbus — Airbus, a leading aircraft manufacturer. Website, Accessed 2012-09-13. <http://www.airbus.com/aircraftfamilies/passengeraircraft/a350xwbfamily/>.
- [5] Airbus. A380 Family: A380-800 - A380 photos, pictures, A380 videos, A380 3D view — Airbus — Airbus, a leading aircraft manufacturer. Website, Accessed 2012-09-13. <http://www.airbus.com/aircraftfamilies/passengeraircraft/a380family/>.
- [6] Airbus. Dimensions & key data — Airbus, a leading aircraft manufacturer. Website, Accessed 2012-11-07. <http://www.airbus.com/aircraftfamilies/passengeraircraft/a320family/a318/specifications/>.
- [7] Airbus. Dimensions & key data — Airbus, a leading aircraft manufacturer. Website, Accessed 2012-11-07. <http://www.airbus.com/aircraftfamilies/passengeraircraft/a320family/a321/specifications/>.
- [8] Airbus. Dimensions & key data — Airbus, a leading aircraft manufacturer. Website, Accessed 2012-11-07. <http://www.airbus.com/aircraftfamilies/passengeraircraft/a380family/a380-800/specifications/>.
- [9] Airline Electronic Engineering Committee/ARINC Standards. *ARINC Specification 429P1 - 17 Mark 33 Digital Information Transfer System (DITS), Part 1, Functional Description, Electrical Interface, Label Assignments and Word Formats*. 2004.

- [10] Airline Electronic Engineering Committee/ARINC Standards. *ARINC Specification 664P5 - Aircraft Data Network, Part 5, Network Domain Characteristics and Interconnection*. 2005.
- [11] Airline Electronic Engineering Committee/ARINC Standards. *ARINC Specification 664P7 - Aircraft Data Network, Part 7, Avionics Full-Duplex Switched Ethernet (AFDX) Network*. 2005.
- [12] Anritsu. Data Quality Analyzer MD1230B. Website, Accessed 2012-07-02. <http://www.anritsu.com/en-US/Products-Solutions/products/MD1230B.aspx>.
- [13] H. Bauer, J.-L. Scharbag, and C. Fraboul. Improving the Worst-Case Delay Analysis of an AFDX Network Using an Optimized Trajectory Approach. *IEEE Transactions on Industrial Informatics*, 6(4):521–533, 2010.
- [14] J. Bennett, C. Partridge, and N. Shectman. Packet reordering is not pathological network behavior. *IEEE/ACM Transactions on Networking*, 7(6):789–798, 1999.
- [15] J. Bisschop. *AIMMS Optimization Modeling*. Lulu Pr, 2006.
- [16] A. Bouillard, L. Jouhet, and E. Thierry. Tight Performance Bounds in the Worst-Case Analysis of Feed-Forward Networks. In *Proceedings of the 29th Conference on Computer Communications, INFOCOM 2010*, pages 1–9, 2010.
- [17] A. Bouillard and É. Thierry. An algorithmic toolbox for network calculus. *Discrete Event Dynamic Systems*, 18(1):3–49, 2008.
- [18] BRITE. Boston university Representative Internet Topology generator. Website, Accessed 2011-07-03. <http://www.cs.bu.edu/brite/>.
- [19] Broadcom. Broadcom.com. Website, Accessed 2011-01-25. <http://www.broadcom.com>.
- [20] Broadcom. BroadR-Reach PHYs. Website, Accessed 2011-10-03. <http://www.broadcom.com/products/Physical-Layer/BroadR-Reach-PHYs>.
- [21] H. Charara, J.-L. Scharbag, J. Ermont, and C. Fraboul. Methods for bounding end-to-end delays on an AFDX network. In *Proceedings of the 18th Euromicro Conference on Real-Time Systems, ECRTS 2006*, pages 1–10, 2006.
- [22] F. Ciucu, A. Burchard, and J. Liebeherr. A network service curve approach for the stochastic analysis of networks. In *Proceedings of the International Conference on Measurements and Modeling of Computer Systems, SIGMETRICS 2005*, volume 33, pages 279–290. ACM, 2005.
- [23] F. Ciucu and J. Schmitt. Perspectives on Network Calculus: No Free Lunch, but Still Good Value. In *Proceedings of the Conference on Applications, Technologies,*

- Architectures, and Protocols for Computer Communications, SIGCOMM 2012*, pages 311–322. ACM, 2012.
- [24] Computational Infrastructure for Operations Research. COIN-OR: Cbc. Website, Accessed 2012-02-01. <http://www.coin-or.org/projects/Cbc.xml>.
- [25] M. Crovella and A. Bestavros. Self-Similarity in World Wide Web Traffic: Evidence and Possible Causes. *IEEE/ACM Transactions on Networking*, 5(6):835–846, 1997.
- [26] R. Cruz. A Calculus for Network Delay, Part I, Network Elements in Isolation. *IEEE Transactions on Information Theory*, 37(1):114–131, 1991.
- [27] R. Cruz. A Calculus for Network Delay, Part II, Network Analysis. *IEEE Transactions on Information Theory*, 37(1):132–141, 1991.
- [28] Distributed Computer System Lab. DISCO Network Calculator. Website, Accessed 2011-01-24. <http://disco.informatik.uni-kl.de/content/Downloads>.
- [29] DMTF. Common Information Model. Website, Accessed 2011-07-03. <http://www.dmtf.org/standards/cim>.
- [30] DMTF. WBEM — DMTF. Website, Accessed 2012-10-26. <http://www.dmtf.org/standards/wbem>.
- [31] DMTF. Web-Based Enterprise Management (WBEM) FAQs. Website, Accessed 2012-10-26. http://www.dmtf.org/about/faq/wbem_faq.
- [32] DMTF. DMTF Tutorial Introduction. Website, Accessed 2012-10-30. <http://www.wbemsolutions.com/tutorials/DMTF/>.
- [33] EASA. Certification Memorandum - Development Assurance of Airborne Electronic Hardware (Chapter 9). Technical Report EASA CM - SWCEH - 001 Issue 01, Software & Complex Electronic Hardware section, European Aviation Safety Agency, 2011.
- [34] J. Eidson. *Measurement, Control, and Communication Using IEEE 1588*. Springer-Verlag, New York, 2006.
- [35] M. Fidler. Survey of Deterministic and Stochastic Service Curve Models in the Network Calculus. *Communications Surveys Tutorials, IEEE*, 12(1):59–86, 2010.
- [36] F. Fock. SNMP++ v.3x. Website, Accessed 2012-10-25. http://www.agentpp.com/snmp_pp3_x/snmp_pp3_x.html.
- [37] M. Foquet, R. Holz, N. Kammenhuber, and A. Klein. Full IP Cabin Intermediate Report. Technical Report, Internal Communication, 2010.

- [38] F. Frances, C. Fraboul, and J. Grieu. Using network calculus to optimize the AFDX network. In *Proceedings of the International Congress on Embedded Real Time Software, ERTS 2006*, 2006.
- [39] G. Garner, A. Gelter, and M. Teener. New Simulation and Test Results for IEEE 802.1AS Timing Performance. In *Proceedings of the International Symposium on Precision Clock Synchronization for Measurement, Control and Communication, ISPCS 2009*, pages 1–7, 2009.
- [40] J. Green. Network Notepad. Website, Accessed 2012-03-09. <http://www.networknotepad.com/>.
- [41] E. Heidinger. Rare Events in Network Simulation Using MIP. In *Proceedings of the 23rd International Teletraffic Congress, ITC 2011*, pages 314–315, 2011.
- [42] E. Heidinger, C. Heller, A. Klein, and S. Schnee. Quality of Service IP Cabin Infrastructure. In *Proceedings of the 29th Digital Avionics Systems Conference, DASC 2010*, pages 3.D.4–1–3.D.4–10, 2010.
- [43] T. Higgins. NETGEAR GS110TP ProSafe 8-port Gigabit PoE Smart Switch with 2 Gigabit Fiber SFP Reviewed. Website, Accessed 2012-07-11. <http://www.smallnetbuilder.com/lanwan/lanwan-reviews/31217-netgear-gs110tp-prosafe-8-port-gigabit-poe-smart-switch-with-2-gigabit-fiber-sfp-reviewed/>.
- [44] H. Hintze, A. Tolksdorf, and R. God. Cabin Core System - A Next Generation Platform for Combined Electrical Power and Data Services. In *Proceedings of the 3rd International Workshop on Aircraft Technology*, 2011.
- [45] A. Holt. *Network Performance Analysis: Using the J Programming Language*. Springer-Verlag, Berlin, 2007.
- [46] IEEE Computer Society. *IEEE Std. 802.1D, Standard for Local and Metropolitan Area Networks, Media Access Control (MAC) Bridges*. 2004.
- [47] IEEE Computer Society. *IEEE Std. 802.1Q, Standard for Local and Metropolitan Area Networks, Virtual Bridged Local Area Networks*. 2005.
- [48] IEEE Computer Society. *IEEE Std. 802.3, Standard for Local and Metropolitan Area Networks, Carrier Sense Multiple Access With Collision Detection (CSMA/CD) Access Method and Physical Layer Specifications*. 2005.
- [49] IEEE Computer Society. *IEEE Std. 802.1AB-2009, Station and Media Access Control Connectivity Discovery (LLDP)*. 2009.
- [50] IEEE Computer Society. *IEEE Std. 802.1Qav, IEEE Standard for Local and Metropolitan Area Networks, Virtual Bridged Local Area Networks, Amendment 12: Forwarding and Queuing Enhancements for Time-Sensitive Streams*. 2009.

- [51] IEEE Computer Society. *IEEE Std. 1588-2008, IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems*. 2010.
- [52] IEEE Computer Society. *IEEE Std. 802.1Qat, IEEE Standard for Local and Metropolitan Area Networks, Virtual Bridged Local Area Networks, Amendment 14: Stream Reservation Protocol (SRP)*. 2010.
- [53] IEEE Computer Society. *IEEE Std. 1722, IEEE Standard for Layer 2 Transport Protocol for Time Sensitive Applications in Bridged Local Area Networks*. 2011.
- [54] IEEE Computer Society. *IEEE Std. 802.1AS, Timing and Synchronization for Time-Sensitive Applications in Bridged Local Area Networks*. 2011.
- [55] J. Imtiaz, J. Jasperneite, and L. Han. A performance study of Ethernet Audio Video Bridging (AVB) for Industrial real-time communication. In *Proceedings of the 14th Conference on Emerging Technologies & Factory Automation, ETFA 2009*, pages 1–8. IEEE, 2009.
- [56] INCHRON. chronVAL. Website, Accessed 2012-04-01. <http://www.inchron.de/chronval.html>.
- [57] International Electrotechnical Commission. *IEC 61883-1 ed3.0, Consumer audio/video equipment - Digital interface - Part 1: General*. 2008.
- [58] Internet Engineering Task Force. *RFC2131: Dynamic Host Configuration Protocol*. Number 2131 in Request for Comments. 1997.
- [59] Internet Engineering Task Force. *RFC2215: General Characterization Parameters for Integrated Service Network Elements*. 1997.
- [60] Internet Engineering Task Force. *RFC2544: Benchmarking Methodology for Network Interconnect Devices*. Number 2544 in Request for Comments. 1999.
- [61] Internet Engineering Task Force. *RFC3410: Introduction and Applicability Statements for Internet-Standard Management Framework*. Number 3410 in Request for Comments. 2002.
- [62] Internet Engineering Task Force. *RFC3418: Management Information Base (MIB) for the Simple Network Management Protocol (SNMP)*. Number 3418 in Request for Comments. 2002.
- [63] Internet Engineering Task Force. *RFC4022: Management Information Base for the Transmission Control Protocol (TCP)*. Number 4022 in Request for Comments. 2005.
- [64] Internet Engineering Task Force. *RFC4113: Management Information Base for the User Datagram Protocol (UDP)*. Number 4113 in Request for Comments. 2005.
- [65] Internet Engineering Task Force. *RFC4293: Management Information Base for the Internet Protocol (IP)*. Number 4293 in Request for Comments. 2006.

- [66] Internet Engineering Task Force. *RFC4363: Definitions of Managed Objects for Bridges with Traffic Classes, Multicast Filtering, and Virtual LAN Extensions*. Number 4363 in Request for Comments. 2006.
- [67] Internet Engineering Task Force. *RFC6241: Network Configuration Protocol (NETCONF)*. Number 6241 in Request for Comments. 2011.
- [68] Y. Jiang. A Basic Stochastic Network Calculus. In *Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications, SIGCOMM 2006*, pages 123–134, 2006.
- [69] Y. Jiang and Y. Liu. *Stochastic Network Calculus*. Springer-Verlag, New York, 2008.
- [70] A. Kern, H. Zinner, T. Streichert, J. Nöbauer, and J. Teich. Accuracy of Ethernet AVB time synchronization under varying temperature conditions for automotive networks. In *Proceedings of the 48th Design Automation Conference*, pages 597–602, 2011.
- [71] A. Kiefer, N. Gollan, and J. Schmitt. Searching for Tight Performance Bounds in Feed-Forward Networks. In *Measurement, Modelling, and Evaluation of Computing Systems and Dependability and Fault Tolerance*. Springer Verlag, Berlin, 2010.
- [72] T. Koch. *Rapid Mathematical Prototyping*. PhD thesis, Technische Universität Berlin, 2004.
- [73] S. Lagrange. COINC Toolbox. Website, Accessed 2011-05-23. <http://www.istia.univ-angers.fr/~lagrange/software.php>.
- [74] L. Lamport and P. Melliar-Smith. Byzantine clock synchronization. In *Proceedings of the Third Annual ACM Symposium on Principles of Distributed Computing*, pages 68–74, 1984.
- [75] J. Le Boudec. *Network Calculus: A Theory of Deterministic Queuing Systems for the Internet*. Springer-Verlag, Berlin, 2004.
- [76] F. Leipold. *Wireless UWB Aircraft Cabin Communication System*. PhD thesis, Technische Universität München, 2011.
- [77] W. E. Leland, M. S. Taqqu, W. Willinger, and D. V. Wilson. On the Self-Similar Nature of Ethernet Traffic. In *Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications, SIGCOMM 1993*, pages 183–193, 1993.
- [78] H. Lim, D. Herrscher, L. Volker, and M. Waltl. IEEE 802.1 AS time synchronization in a switched Ethernet based in-car network. In *Proceedings of the Conference on Vehicular Networking Conference, VNC 2011*, pages 147–154. IEEE, 2011.

- [79] S. Martin and P. Minet. Schedulability analysis of flows scheduled with fifo: application to the expedited forwarding class. In *Proceedings of the 20th International Parallel and Distributed Processing Symposium, IPDPS 2006*, pages 1–8, 2006.
- [80] Marvell. LinkStreet 88E6096/88E60097/88E6097F Datasheet. Website, 2007.
- [81] Marvell. Marvell Technology Group Ltd. Website, Accessed 2011-01-25. <http://www.marvell.com>.
- [82] K. Matheus, M. Kicherer, and T. Königseder. Audio/Video Transmission in Cars using Ethernet. 2010.
- [83] N. McKeown, V. Anantharam, and J. Walrand. Achieving 100% throughput in an input-queued switch. In *Proceedings of the 15th Conference on Computer Communications, INFOCOM 1996*, pages 296–302, 1996.
- [84] Micrel. KS8893MQL Datasheet, Integrated 3-Port 10/100 Managed Switch with PHYs. Website, Accessed 2011-01-24. http://www.micrel.com/_PDF/Ethernet/ks8893mql.pdf.
- [85] Micrel. Micrel - Innovation Through Technology. Website, Accessed 2012-07-05. <http://www.micrel.com>.
- [86] Micrel. KS8995MA Datasheet, Integrated 5-Port 10/100 Managed Switch. Website, Accessed 2012-09-04. http://www.micrel.com/_PDF/Ethernet/ks8995ma.pdf.
- [87] J. Mitchell. Branch-and-cut algorithms for combinatorial optimization problems. *Handbook of Applied Optimization*, pages 65–77, 2002.
- [88] NetFPGA. NetFPGA. Website, Accessed 2012-07-05. <http://netfpga.org/>.
- [89] Netgear. Computer Networking Products & Equipment From NETGEAR. Website, Accessed 2012-04-23. <http://www.netgear.com>.
- [90] Netgear. GS110TP. Website, Accessed 2012-04-23. <http://www.netgear.de/products/business/switches/smart-switches/GS110TP.aspx>.
- [91] ns-2. The Network Simulator - ns-2. Website, Accessed 2012-04-24. <http://www.isi.edu/nsnam/ns/>.
- [92] ns-3. Website, Accessed 2012-04-24. <http://www.nsnam.org/>.
- [93] D. Ohlsen. IEEE 1722 AVB L2 Transport Protocol. Website, Accessed 2011-01-24. <http://www.ieee802.org/1/files/public/docs2007/avb-dolsen-1722-status-1107.pdf>.
- [94] OMNeT++. OMNeT++ Network Simulation Framework. Website, Accessed 2011-07-03. <http://www.omnetpp.org/>.

- [95] OPNET. Application and Network Performance with OPNET. Website, Accessed 2011-07-04. <http://www.opnet.com/>.
- [96] V. Paxson and S. Floyd. Wide-Area Traffic: The Failure of Poisson Modeling. *IEEE/ACM Transactions on Networking*, 3(3):226–244, 1995.
- [97] Radio Technical Commission for Aeronautics. *DO-178B/ED-12B: Software Considerations in Airborne Systems and Equipment Certification*. 1992.
- [98] Radio Technical Commission for Aeronautics. *DO-214/SC-164: Audio Systems Characteristics and Minimum Operational Performance Standards for Aircraft Audio Systems and Equipment Systems and Equipment*. 1993.
- [99] Radio Technical Commission for Aeronautics. *DO-254/ED-80: Design Assurance Guidance For Airborne Electronic Hardware*. 2000.
- [100] G. Rizzo and J. Le Boudec. “Pay bursts only once” does not hold for non-FIFO guaranteed rate nodes. *Performance Evaluation*, 62(1-4):366–381, 2005.
- [101] J.-L. Scharbarg and C. Fraboul. Methods and Tools for the Temporal Analysis of Avionic Networks. *New Trends in Technologies: Control, Management, Computational Intelligence and Network Systems*, pages 414–438, 2010.
- [102] H. Schioler. CyNC. Website, Accessed 2011-05-23. <http://www.control.auc.dk/~henrik/CyNC/>.
- [103] J. Schmitt, F. Zdarsky, and M. Fidler. Delay Bounds under Arbitrary Multiplexing: When Network Calculus Leaves You in the Lurch... In *Proceedings of the 27th Conference on Computer Communications, INFOCOM 2008*, pages 1669–1677, 2008.
- [104] J. B. Schmitt and F. A. Zdarsky. The DISCO Network Calculator: A Toolbox for Worst Case Analysis. In *First International Conference on Performance Evaluation Methodologies and Tools, ValueTools 2006*, pages 1–10, 2006.
- [105] J. B. Schmitt, F. A. Zdarsky, and I. Martinovic. Improving Performance Bounds in Feed-Forward Networks by Paying Multiplexing Only Once. In *Proceedings of the 14th Conference on Measuring, Modelling and Evaluation of Computer and Communication Systems (MMB)*, 2008.
- [106] A. Schrijver. *Theory of linear and integer programming*. Wiley, 1998.
- [107] Society of Automotive Engineers. *ARP 4754A: Guidelines for Development of Civil Aircraft and Systems*. 2010.
- [108] I. Soldatos, E. Vayias, and G. Kormentzas. On the building blocks of quality of service in heterogeneous IP networks. *IEEE Communications Surveys & Tutorials*, 7(1), 2005.

-
- [109] SSFNet. Scalable Simulation Framework. Website, Accessed 2011-07-03. <http://www.ssfnet.org/>.
- [110] W. Stallings. *High Speed Networks and Internets*. Pearson, 2002.
- [111] D. Starobinski, M. Karpovsky, and L. A. Zakrevski. Application of network calculus to general topologies using turn-prohibition. *IEEE/ACM Transactions on Networking*, 11(3):411–421, 2003.
- [112] Symtavigation. SymTA/S. Website, Accessed 2012-04-01. <http://www.symtavigation.com/symtas.html>.
- [113] A. Tanenbaum. *Computer networks*. Pearson, 2002.
- [114] M. Teener. Higher Level Streaming Standards: Part 1 - IEC 61883. Website, Accessed 2011-01-24. http://www.ieee802.org/3/re_study/public/200505/teener_1_0505.pdf.
- [115] TIK. Modular Performance Analysis with Real-Time Calculus. Website, Accessed 2012-02-01. <http://www.mpa.ethz.ch/>.
- [116] Ultimate++. Ultimate++ framework. Website, Accessed 2012-03-11. <http://www.ultimatepp.org/>.
- [117] UP4ALL. Design Verification for Embedded Systems. Website, Accessed 2012-04-12. <http://www.uppaal.com/>.
- [118] E. Wandeler and L. Thiele. Real-Time Calculus (RTC) Toolbox. Website, 2006. <http://www.mpa.ethz.ch/Rtctoolbox>.
- [119] D. Witsch, B. Vogel-Heuser, J. Faure, G. Marsal, et al. Performance analysis of industrial Ethernet networks by means of timed model-checking. In *Proceedings of the 12th Symposium on Information Control Problems in Manufacturing, INCOM 2006*, pages 1–6, 2006.
- [120] XMOS. Low-cost AVB Audio Endpoint Kit. Website, Accessed 2012-04-23. <http://www.xmos.com/products/development-kits/avb12>.

ISBN 3-937201-39-4
ISSN 1868-2634 (print)
ISSN 1868-2642 (electronic)
DOI: 10.2313/NET-2013-12-1