Adaptation Algorithms for Quality of Service Improvement of Distributed Interactive Multimedia Applications in IP-based Networks

Summary

While the usage of digital communication networks increases, applications require more and more performance of networks and terminals. In particular *interactive distributed multimedia applications* exchange large amounts of data under tight real-time constraints.

It is expected that applications like video conferencing or Internet telephony are used even more in the future. This is accompanied by increased expectations of the users towards these applications: as soon as a service is commonly accepted, a sufficiently high Quality of Service (QoS) is a necessity. At the same time, users wish to get a cost-efficient service which uses the available infrastructure, i. e. the currently existing IP-based Internet. Therefore, it is necessary to run distributed applications with sufficiently high QoS over networks without service guarantees. Since the behaviour of the network cannot be influenced directly (e. g., by reservation of resources or by using differentiating services) in this scenario, the applications themselves have to adapt to the network behaviour.

The goal of every adaptation is to increase the user-experienced QoS, or to keep it at a predefined level. In order to do this, the requirements toward the application have to be described, depending on the concrete scenario in which the application is used. Since a study with a large group of users was beyond the scope of this work, several scenarios are defined and requirements for the QoS are derived. The focus is laid on *interactive* applications, which restricts the time between the generation and the playout of the data. It therefore limits the means of the application to counter the deficiencies of the network.

This report starts with a description of the basic concepts of distributed multimedia applications, analyses the possibilities of an adaptation and demonstrates experimental studies to evaluate the performance of mechanisms and algorithms in several scenarios. It shows when and how an adaptation to the network behaviour can be performed successfully.

While adaptation showed significant improvements for the transmission of audio streams, video streams coded with a robust coding algorithm suffered less degradation due to the network deficiencies, and therefore showed smaller improvement. The buffering capacity of the network, i. e. the number of packets that can be in transit to the receiver at the same time, was identified as a significant problem for an effective adaptation.

Adaptionsalgorithmen zur Erhöhung der Dienstgüte verteilter interaktiver Multimedia-Anwendungen in IP-basierten Netzen

Zusammenfassung

Während sich die Nutzung digitaler Übertragungsnetze erhöht, verlangen Anwendungen immer mehr Leistung von Netzen und Endgeräten. Insbesondere tauschen *verteilte, interaktive Multimedia-Anwendungen* große Datenmengen unter strengen Echtzeitanforderungen aus.

Es wird erwartet, dass Anwendungen wie Video-Konferenzen oder Internet-Telefonie künftig vermehrt eingesetzt werden. Dies wird von gesteigerten Erwartungen der Nutzer an diese Anwendungen begleitet: sobald ein Dienst allgemein akzeptiert wird, muss eine ausreichend hohe Dienstgüte (*Quality of Service*, QoS) bereitgestellt werden. Gleichzeitig wünschen sich die Anwender kostengünstige Dienste, die die verfügbare Infrastruktur nutzen, d. h. das existierende, IP-basierte Internet. Somit ist es notwendig, verteilte Anwendungen mit ausreichend hoher QoS über Netze zu betreiben, die keine Dienstgüten-Garantien geben. Da das Verhalten solcher Netze nicht direkt beeinflusst werden kann (z. B. durch Reservierung von Ressourcen oder durch differenzierende Dienste), müssen sich die Anwendungen selbst an das Verhalten des Netzes anpassen.

Das Ziel jeder Anpassung ist die Erhöhung der vom Nutzer erfahrenen QoS, bzw. das Halten der erfahrenen QoS auf einem vorgegebenen Niveau. Um dies zu erreichen, müssen die Anforderungen an die Anwendung in Abhängigkeit des konkreten Einsatzszenarios beschrieben werden. Da eine Studie mit einer großen Zahl an Nutzern den Umfang dieser Arbeit überstiegen hätte, wurden mehrere Szenarien definiert und QoS-Anforderungen abgeleitet. Die Betonung wurde dabei auf *interaktive* Anwendungen gelegt, was die Zeit zwischen Erzeugung und Ausspielen der Daten begrenzt und somit die möglichen Reaktionen der Anwendung auf die Defizite des Netzes eingeschränkt.

Die Arbeit beginnt mit einer Beschreibung der Grundkonzepte verteilter Multimedia-Anwendungen, analysiert die Möglichkeiten der Anpassung und stellt experimentelle Untersuchungen vor, mit denen die Leistung von Mechanismen und Algorithmen in mehreren Szenarien bewertet wird. Sie zeigt, wann und wie eine Anpassung an das Netzverhalten erfolgreich durchgeführt werden kann.

Es zeigte sich, dass bei der Übertragung von Audio-Strömen durch Adaption deutliche Verbesserungen erreicht werden können, während Video-Ströme im untersuchten, robusten Video-Kodierungsverfahren weniger unter den Defiziten des Netzes leiden, und daher geringere Verbesserungen erfahren. Als großes Problem einer effektiven Anpassung wurde die Pufferkapazität des Netzes identifiziert, also wie viele Pakete sich gleichzeitig auf dem Weg zum Empfänger befinden können.

Inhaltsverzeichnis

Abkürzungen

ADPCM	Adaptive Differential Pulse Code Modulation
ALF	Application Level Framing *(Rahmenbildung in der Anwendungsschicht)*
ALQ	Application Level Quality *(Dienstgüte der Anwendung)*
ATM	Asynchronous Transfer Mode
CSMA/CD	Carrier Sense Multiple Access with Collision Detection
Codec	Coder/Decoder *(Kodierer/Dekodierer)*
CLQ	Communication Level Quality *(Dienstgüte der Übertragung)*
CD	Compact Disk
CSCW	Computer-Supported Collaborative Work *(Rechnerunterstütztes, kooperatives Arbeiten)*
ADSL	Asymmetric Digital Subscriber Line
DVD	Digital Versatile Disk
FIFO	First In, First Out *(Warteschlange)*
FEC	Forward Error Correction *(Vorwärts-Fehlerkorrektur)*
GSM	Global System for Mobile Communications
ISDN	Integrated Services Digital Network
ISO/OSI	International Organisation for Standardization/Open Systems Interconnection
ITU-T	International Telecommunication Union – Telecommunication Standardization Sector
IETF	Internet Engineering Task Force
IP	Internet Protocol
JPEG	Joint Photographic Experts Group
LAN	Local Area Network *(Lokales Netz)*
MAN	Metropolitan Area Network *(Regionales Netz)*
MPEG	Moving Picture Experts Group
NLQ	Network Level Quality *(Dienstgüte des Übertragungsnetzes)*
PC	Personal Computer
PDU	Protocol Data Unit

PCM	Pulse Code Modulation
QoS	Quality of Service *(Dienstgüte)*
RTCP	RTP Control Protocol
RTP	Real-time Transport Protocol
RFC	Request for Comment
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
ULQ	User Level Quality *(Dienstgüte in der Nutzer-Schicht)*
WAN	Wide Area Network *(Weitverkehrsnetz)*
WWW	World Wide Web

1 Einleitung

Mit der steigenden Verbreitung digitaler Kommunikationsnetze verändert sich die Art der eingesetzten Anwendungen, denn sie stellen erhöhte Anforderungen an die Leistungsfähigkeit der Netze und Endgeräte. Solche Anwendungen sind insbesondere *verteilte Multimedia-Anwendungen*, bei denen große Datenmengen unter Echtzeitbedingungen zwischen den Endgeräten ausgetauscht werden.

Es wird allgemein damit gerechnet, dass der Einsatz von Anwendungen wie Video-Konferenzen oder „Internet-Telefonie“ in Zukunft noch stärker zunehmen wird. Einher gehen dabei steigende Erwartungen der Nutzer an die Anwendung: Sobald ein Dienst als alltäglich erfahren wird, wird eine ausreichende Dienstgüte erwartet. Gleichzeitig wünschen die Nutzer aber einen kostengünstigen Dienst, der über die momentan verfügbare Infrastruktur abgewickelt werden kann. Dies bedeutet, dass heute vorhandene (vor allem IP-basierte) Netze für die Übertragung eingesetzt werden sollen. Somit ist es notwendig, verteilte Anwendungen mit genügend hoher Dienstgüte über Netze ohne Dienstgütengarantien zu betreiben. Da hier nicht die Möglichkeit besteht, das Verhalten des Netzes (durch Reservierung von Ressourcen, differenzierende Dienste) direkt zu beeinflussen, müssen sich die Anwendungen selbst an das Verhalten des Netzes anpassen.

Ziel jeder Anpassung ist es dabei, die vom Nutzer erfahrene Dienstgüte zu erhöhen, bzw. auf einem ausreichend hohen Niveau zu halten. Hierzu ist es erforderlich, die Anforderungen an die Anwendung zu beschreiben, die vom konkreten Einsatz-Szenario abhängen. Hierzu sind jedoch Untersuchungen mit einer großen Gruppe von Menschen erforderlich, die hier nicht durchgeführt werden konnten. Daher wurden bestimmte Szenarien definiert und aus ihnen Anforderungen an die Dienstgüte der Anwendungen abgeleitet. Außerdem wurde hier der Fokus auf *interaktive* Anwendungen gelegt. Dies bedeutet, dass zwischen der Entstehung der Daten beim Sender und ihrem Ausspielen beim Empfänger nicht zu viel Zeit vergehen darf, was die Möglichkeiten zum Ausgleich der Defizite des Netzes stark einschränkt.

Die vorliegende Arbeit beginnt mit einer Beschreibung der Grundlagen verteilter Multimedia-Anwendungen, analysiert dann die Möglichkeiten der Anpassung und stellt experimentelle Untersuchungen vor, die die Leistung von Mechanismen und Algorithmen in verschiedenen Szenarien bewerten. Sie schlägt ein Vorgehen zur Entwicklung von Adaptionsalgorithmen für neue Anwendungen vor und zeigt, wann und wie eine Anpassung an das Netzverhalten erfolgreich durchgeführt werden kann.

Nach der Einleitung werden in Kapitel 2 die Grundlagen der Datenkommunikation dargestellt, die dieser Untersuchung zugrunde lagen. Zuerst wird das ISO/OSI-Referenzmodell kurz erläutert, um Mechanismen, Protokolle usw. einordnen zu können. Es wird auf die bekanntesten Vertreter für Netzarchitekturen und -protokolle (TCP/IP, Ethernet und ATM) eingegangen und dann der Begriff der Dienstgüte in Kommunikationsnetzen sowie in Multimedia-Anwendungen eingeführt.

Der nächste Abschnitt bietet einen Überblick über die Grundlagen verteilter Multimedia-Anwendungen. Die bedeutendsten Standards für die Kommunikationssteuerung (H.323 und SIP), sowie für die Datenübertragung (RTP/RTCP) und die Medien-Kodierung werden dargestellt. Danach werden Anwendungstypen definiert und Auswirkungen verschiedener Szenarien auf die Anforderungen an die Dienstgüte von Anwendung und Netz beschrieben.

Es folgt ein allgemeiner Abschnitt über Anpassung in Multimedia-Anwendungen, insbesondere eine Festlegung der Aufgaben von Algorithmen und Mechanismen und ein Vergleich mit der Regelungstechnik. Das Kapitel schließt mit einem Überblick über weitere Multimedia-Anwendungen und -Frameworks, wobei eine Abgrenzung hinsichtlich der Ziele und Methoden zur vorliegenden Arbeit vorgenommen wird.

Kapitel 3 führt in die konkreten Möglichkeiten der Anpassung in Multimedia-Anwendungen ein. Es wird beschrieben, wie ein Adaptionsalgorithmus vorgeht, um die Dienstgüte zu verbessern und wie die Leistung verschiedener Algorithmen verglichen werden kann.

Es folgt ein Abschnitt über die Mechanismen, mit denen Anwendungen die vom Nutzer erfahrene Dienstgüte verbessern können. Zunächst wird als Vorbereitung eine Klassifikation nach vier Kriterien präsentiert.

Der letzte Teil des Kapitels stellt dann Anpassungsmechanismen und ihre Auswirkungen auf die Multimedia-Anwendung und das Netz vor. Sie werden anhand der Klassifikation eingeteilt und ihre Eigenschaften in einer Tabelle am Ende des Kapitels zusammengefasst.

Kapitel 4 erläutert die durchgeführte experimentelle Untersuchung. Das verwendete System, die sog. eXPERIMENTATION PLATFORM, wird hinsichtlich seiner Grundstruktur und Konzepte erklärt. Es folgen Abschnitte, in denen die Implementierungen des Audio- und des Video-Mediums beschrieben werden, um die Funktion der Mechanismen und Algorithmen verstehen zu können.

Der nächste Teil des Kapitels enthält, wie die untersuchten Szenarien nachgebildet wurden. Hierzu wurden zwei WAN-Emulatoren eingesetzt, deren Funktionsweise genauer beschrieben wird. Es werden die Szenarien definiert, innerhalb derer die Algorithmen verglichen wurden. Zum Ende des Kapitels wird eine medienübergreifende Adaption eingeführt. Das Konzept der *Meta-Policy* und die eingesetzten Mechanismen werden vorgestellt.

Die Bewertung der verschiedenen Algorithmen erfolgt in Kapitel 5. Hier werden zunächst die Kriterien festgelegt, nach denen die Leistung der Algorithmen verglichen wird, und dann mittels der Ergebnisse der Experimente die Algorithmen bewertet. Die sich ergebenden Resultate werden mit den erwarteten Werten verglichen und Gründe für auftretende Unterschiede angegeben. Es wird skizziert, was für eine weitere Verbesserung der Anpassung, aber auch der Anwendungen insgesamt, unternommen werden müsste.

Die Ausarbeitung schließt in Kapitel 6 mit einem Fazit über die Vor- und Nachteile adaptiver, verteilter Multimedia-Anwendungen und einer Zusammenfassung sowie einem Ausblick auf mögliche Erweiterungen und Verbesserungen.

Im Anhang findet sich zunächst ein Glossar, es folgt dann eine genaue Beschreibung der durchgeführten Experimente. Anhang B liefert eine Übersicht über die Klassen der eXPERIMENTATION PLATFORM und über einige interessante Mechanismen wie Zeitstempel und die Erfassung von Messwerten. Außerdem werden die zur Berechnung der Dienstgüte verwandten Tabellen dokumentiert. In Anhang C folgen Überlegungen zu Simulationsstudien. Die verwendete Software, sowie die erfassten Messdaten können auf einem zusätzlichen Datenträger (CD-ROM) zur Verfügung gestellt werden.

Anmerkung

Wenn in dieser Ausarbeitung von (Be-)Nutzern die Rede ist, so ist immer auch die weibliche Form der Bezeichnung gemeint. Nur aus Gründen der Lesbarkeit wurde auf Formulierungen wie „Nutzerinnen und Nutzer“ verzichtet.

2 Grundlagen und bekannte Ansätze

2.1 Kommunikationsarchitekturen und Dienstgüte

Zunächst sollen hier ein Überblick über die Grundeigenschaften digitaler Kommunikationsnetze gegeben und sodann einige Technologien vorgestellt werden. Es folgt ein Überblick über die für das Verständnis notwendigen Begriffe der Dienstgüte in Kommunikationsnetzen. Für einen ausführlichen Überblick sei das Buch „Computer Networks“ von Andrew S. Tanenbaum [Tan96] empfohlen.

2.1.1 ISO/OSI-Basis-Referenzmodell

In dieser Architektur (Tab. 2.1 [IT94]) zur systemunabhängigen Kommunikation (engl. *Open Systems Interconnection, OSI*), die 1983 von der ISO (*International Standards Organization*) verabschiedet wurde, werden die Aufgaben, die bei einer Kommunikation durchgeführt werden müssen, auf sieben Schichten aufgeteilt. Jede dieser Schichten darf dabei nur Dienste in Anspruch nehmen, die von der darunter liegenden Schicht (engl. *lower layer*) angeboten werden. Konzeptionell kommunizieren Instanzen der gleichen Schicht über ein sog. *„Peer-to-Peer“* Protokoll. Hierdurch wird eine eindeutige Gliederung der Aufgaben erreicht und es ist problemlos möglich, die Implementierung einer oder mehrerer Schichten auszutauschen.

7	Verarbeitungsschicht	*Application Layer*
6	Darstellungsschicht	*Presentation Layer*
5	Kommunikationssteuerungsschicht	*Session Layer*
4	Transportschicht	*Transport Layer*
3	Vermittlungsschicht	*Network Layer*
2	Sicherungsschicht	*Data Link Layer*
1	Bitübertragungsschicht	*Physical Layer*

Tabelle 2.1: Schichten des ISO/OSI Basis-Referenzmodells

Die Aufgaben der Schichten in Bezug auf eine Multimedia-Übertragung sind wie folgt:

1 – Bitübertragungsschicht: Diese Schicht sorgt für die physikalische Übertragung einzelner Bits. Je nach Technologie werden Bits auf verschiedene Weise dargestellt, z. B. als Lichtimpulse oder modulierte Trägersignale.

2 – Sicherungsschicht: Hier werden Bits zu Gruppen zusammengefasst und als Daten-Rahmen (engl. *data frame*) übertragen. Diese Schicht erkennt Fehler in der physikalischen Übertragung und korrigiert sie, z. B. mittels zusätzlich übertragener redundanter Information oder Wiederholung.

3 – Vermittlungsschicht: Die Funktionen in dieser Schicht sorgen dafür, dass Daten von einem Endgerät ein anderes erreichen können, auch wenn sie nicht direkt miteinander verbunden sind. Hierzu können die Daten einem Pfad folgen, der fest im Netz eingestellt ist, der zu Beginn einer Übertragung festgelegt wird oder der für jede einzelne *Protocol Data Unit (PDU)* neu ermittelt wird.

4 – Transportschicht: Diese Schicht sorgt dafür, dass verschiedene Ende-zu-Ende–*Verbindungen* zwischen zwei Endgeräten unterschieden werden können. Zudem können Protokolle dieser Schicht sicherstellen, dass die Daten korrekt übertragen werden.

5 – Kommunikationssteuerungsschicht: Hier werden Dienste bereitgestellt, um Sitzungen aufzubauen, zu betreiben und wieder abzubauen. Eine Sitzung kann dabei mehrere Verbindungen benutzen und diese z. B. synchronisieren.

6 – Darstellungsschicht: Die Daten einer Kommunikationsbeziehung müssen in einem für alle Teilnehmer verarbeitbaren Weise vorliegen. Hierzu werden in dieser Schicht Daten entsprechend (um-)kodiert.

7 – Verarbeitungsschicht: Die eigentliche *semantische* Verarbeitung der Daten erfolgt in dieser Schicht.

Dieses Referenzmodell hat zwar als Implementierung keine große Bedeutung erhalten, wird jedoch häufig zur Einordnung von Modellen und Konzepten sowie zur Standardisierung benutzt.

Ein wichtiges Prinzip bei der blockorientierten Datenübertragung ist die Verwendung von Nachrichtenköpfen (engl. *Header*). Hierbei kann jede Schicht vor die Daten, die sie von der über ihr liegenden Schicht erhält, Steuer-Information hinzufügen (Abb. 2.1). Mit Hilfe dieser Information kann dann die Schicht im Empfänger entscheiden, wie mit den Daten zu verfahren ist oder an wen sie weitergeleitet werden müssen. Vor der Übergabe an eine höhere Schicht entfernt die Schicht ihren Header, so dass der nächste Header am Anfang der übergebenen Daten steht. Das heißt, dass nicht nur die eigentlichen Daten, sondern auch einige Header vom Sender zum Empfänger übertragen werden müssen.

2.1.2 Netzprotokoll-Architekturen und Übertragungstechnologien

Aus der großen Anzahl der zur Zeit in den unteren vier Schichten des OSI Referenzmodells eingesetzten Protokolle und Technologien seien hier nur TCP/IP (*Transmission Control Protocol/Internet Protocol* [Com00]), *Ethernet* ([MB76], auch IEEE 802.3) und ATM (*Asynchronous Transfer Mode* [IT95]) herausgegriffen und genauer vorgestellt, da sie momentan die meiste Verbreitung finden. In [Tan96] finden sich Beschreibungen weiterer Technologien und Protokolle.

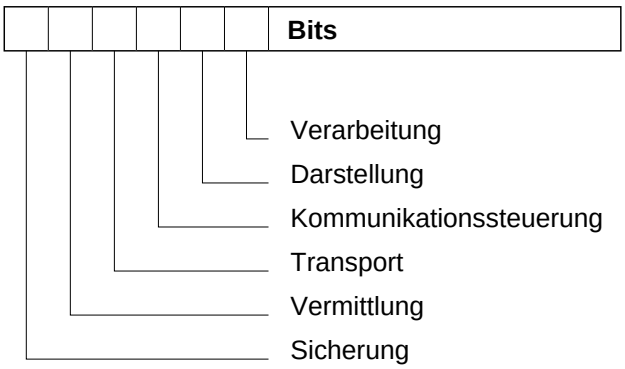


Abbildung 2.1: Verwendung von Headern bei der Übermittlung

2.1.2.1 TCP/IP

TCP/IP ist die zur Zeit am weitesten verbreitete Protokoll-Architektur. Ursprünglich wurde sie im Auftrag des amerikanischen Verteidigungsministeriums für das „ARPA-Net“ entwickelt, nun jedoch ist die Kontrolle an die *Internet Engineering Task Force (IETF)* übergegangen. Durch den Erfolg des *World Wide Web (WWW)* werden heute meist Protokolle dieser Familie zur Verbindung von Datennetzen eingesetzt, u. a. im so genannten „Internet“.

Das Basis-Protokoll ist hierbei das Internet Protocol (IP, STD 5 bzw. RFC[1] 791 [PSC81]), das auf OSI-Schicht 3 (Vermittlung) angesiedelt ist[2]. Als Vermittlungsprotokoll sorgt IP dafür, dass die so genannten IP-Pakete vom Sender zum Empfänger gelangen, wobei für jedes Paket ein Weg erneut festgelegt wird (*Verbindungslose Kommunikation*). Die momentan am häufigsten eingesetzte Version ist IP-Version 4 (IPv4). Da IP-Version 6 (IPv6, RFC 2460 [DH98]) zwar bereits spezifiziert wurde, aber noch nicht weit verbreitet ist, wird hier von IPv4 ausgegangen.

Ein IP-Paket kann bis zu $2^{16} - 1 = 65535$ Byte umfassen. Die Adressierung erfolgt über 4 Byte lange Adressen, wobei diese in einer hierarchischen Struktur angeordnet sind. Hierdurch ist es möglich, Netze logisch zu gruppieren.

Die Beschaffenheit tieferer Schichten wird nicht spezifiziert, wodurch IP sowohl über *Local Area Network*-Technologien (LAN) wie *Ethernet* übertragen werden kann, als auch über *Wide Area Network*-Technologien (WAN) wie ATM oder SDH. Da IP aber keinerlei Anforderungen an die Dienstgüte des darunter liegenden Netzes stellt, ist es andererseits nicht möglich, mit IP eine Dienstgüte für eine Übertragung festzulegen. Um diesen *Best-Effort*-Charakter eines IP-Netzes zu verbessern, wurden jedoch verschiedene Erweiterungsvorschläge (s. 2.1.3.4, S. 11) unterbreitet.

Aufsetzend auf IP werden Protokolle der OSI-Schicht 4 (Transport) spezifiziert, darunter das *Transmission Control Protocol* (TCP, STD 7/RFC 793 [Pos81]). Dieses Protokoll sorgt dafür, dass Verbindungen aufgebaut werden können – wobei dies jedoch nur in den Endgeräten vermerkt ist, d. h. das Transportnetz kennt die Verbindungen nicht. TCP stellt eine korrekte, vollständige Übertragung eines Datenstroms in der richtigen Reihenfolge sicher, indem verloren gegangene IP-Pakete wiederholt versandt werden. Zusammen mit verschiedenen Mechanismen zur Überlastregelung (z. B. AIMD, (engl. *Additive-Increase, Multiplicative-Decrease*)) führt dies zu einem Über-

[1] Die Internet-Standards entstehen aus sog. „*Request for Comments*“ (erhältlich unter http://www.ietf.org/rfc).

[2] Das TCP/IP-Referenzmodell ([Tan96], S. 36) ähnelt dem OSI-Referenzmodell stark, verzichtet jedoch auf die Darstellungs- und die Kommunikationssteuerungsschicht.

tragungsverhalten, das für Echtzeitanwendungen nicht geeignet ist, da sich die Übertragungsverzögerung hierdurch stark erhöhen kann. Eine der wichtigsten Anwendungen von TCP ist der Einsatz als Transportprotokoll für das *Hypertext Transfer Protocol* (HTTP, RFC 2616 [FGMea99]).

Ein weiteres Protokoll der Schicht 4, das auf IP aufsetzt, ist das *User Datagram Protocol* (UDP, STD 6/RFC 768 [Pos80]). Ziel dieses Protokolls ist die möglichst effiziente Übertragung von Daten zwischen Prozessen auf verschiedenen Endgeräten, wobei keine Schutzvorkehrungen getroffen werden. Hierdurch können Pakete sofort weiter gereicht werden, auch wenn frühere Pakete verloren gingen, was die Übertragungsverzögerung so niedrig wie möglich hält. Dies bedeutet aber auch, dass verlorene Pakete in höheren Schichten erkannt werden müssen; ebenso gibt es keine Garantie, dass die Daten in der Reihenfolge des Absendens beim Empfänger ankommen.

UDP wird als Transportprotokoll für weitere Protokolle benutzt, z. B. für das *Real-time Transport Protocol* (RTP, RFC 1889 [SCFJ96], s. 2.2.3.3, S. 16), *Network File System Protocol* (NFS, RFC 1813, [CPS95]) oder das *Network Time Protocol* (NTP, RFC 1305, [Mil92]).

2.1.2.2 Ethernet

Ethernet[3] [MB76] ist eine LAN-Technologie auf Schicht 2 (*Data Link*), d. h. sie dient üblicherweise zur kostengünstigen Übertragung von Daten über relativ kurze Strecken (wenige hundert Meter). Grundprinzip ist die Verwendung eines gemeinsamen Mediums, an das alle Stationen angeschlossen sind. Jede Station lauscht auf dem Medium, ob gerade eine Übertragung stattfindet und darf mit der Übertragung beginnen, falls nicht bereits eine andere Station sendet. Dies erlaubt eine sehr schnelle Übertragung der Daten, da keinerlei Ressourcen angefordert oder fest zugeteilt werden müssen, jedoch mit dem Nachteil möglicher Kollisionen. Bedingt durch die Signallaufzeit kann es vorkommen, dass zwei Stationen gleichzeitig annehmen, das Medium sei verfügbar. Sie fangen zu senden an und die Überlagerung der Signale verstümmelt die Daten. Die Stationen merken dies und stoppen dann die Übertragung – dies führt zu einem starken Abfall der übertragenen Datenmenge, wenn die Zahl der sendewilligen Stationen an einem Medium groß wird. Dieses Protokoll wird als *Carrier Sense Multiple Access with Collision Detection* oder CSMA/CD bezeichnet.

Die übertragenen Daten werden in Rahmen von bis zu 1500 Byte aufgeteilt (d. h. IP-Pakete benötigen u. U. mehrere Rahmen). Die Adressierung erfolgt über eine fest an die so genannte Ethernet-Karte gebundene, weltweit eindeutige Adresse und es gibt keinerlei Dienstgüten-Garantien.

2.1.2.3 ATM

ATM ist eine Technologie, die heute vorwiegend als WAN/MAN-Technologie eingesetzt wird. Ursprünglich als Ersatz für spezialisierte Netze konzipiert, wird ATM [IT95] momentan vor allem im Backbone-Bereich zur Übertragung anderer Protokolle (meist IP) eingesetzt.

ATM ist verbindungsorientiert, d. h. vor Beginn einer Übertragung muss dem Netz mitgeteilt werden, wohin und mit welchen Eigenschaften die Daten übertragen werden sollen. Dies kann in ATM durch Festlegen eines permanenten Pfads (*Permanent Virtual Circuit, PVC*) durch den Netzbetreiber geschehen. Eine Anwendung hat aber auch die Möglichkeit, via Signalisierung einen durchgeschalteten Pfad (*Switched Virtual Circuit, SVC*) anzufordern und einen „Verkehrsvertrag“ über

[3] Benannt in Anlehnung an den „Äther“, der laut einer Theorie aus dem 19. Jahrhundert den Weltraum füllen und die elektromagnetischen Wellen tragen sollte.

dessen Dienstgüte auszuhandeln. Dies erlaubt dem Netz, für jede Verbindung einen Pfad bereit zu stellen und für diesen Dienstgüten-Garantien zu geben.

Für die Übertragung verwendet ATM Pakete fester Größe (so genannte „Zellen“ mit 48 Byte Nutzdaten und 5 Byte Header), die sehr effizient verarbeitet werden können. Für verschiedene Dienste gibt es in ATM so genannte „Adaptions-Schichten“ (*ATM Adaptation Layer*, AAL [IT96b]), z. B. AAL5 zum Transport von Datenpaketen variabler Länge.

Anmerkung: Die ATM-Protokollschichten lassen sich nicht so leicht wie die Schichten der TCP/IP-Architektur den Schichten des ISO/OSI-Modells zuordnen, sondern verteilen sich auf die OSI-Schichten 1 bis 4 ([Tan96], S. 64).

2.1.3 Dienstgüte in Kommunikationsnetzen

2.1.3.1 Allgemein

Um ein Kommunikationssystem bewerten zu können, muss seine *Dienstgüte* (engl. *Quality of Service*, *QoS*) betrachtet werden. Dieses allgemeine Konzept wurde in [ITG97] wie folgt definiert:

> **Definition 2.1 (Dienstgüte):** *Gesamtheit der Qualitätsmerkmale (. . .) aus der Sicht der Benutzer eines betrachteten Dienstes.*

Somit müssen zunächst die jeweiligen „Qualitätsmerkmale“ definiert und bestimmt werden. Aus diesen Werten wird dann über eine weitere Funktion (z. B. eine gewichtete Summe) die Gesamtdienstgüte berechnet.

Die Qualitätsmerkmale eines Kommunikationsnetzes lassen durch die Betrachtung der Übertragung der Daten von einem Endgerät zu einem anderen bestimmen, was sich mit den Parametern nach Tab. 2.2 ausdrücken lässt.

Eigenschaft	Bezeichner	Einheit	Beispiele
Bitfehlerwahrscheinlichkeit	p_{Berr}	1	Glasfaser: 10^{-8}, Funk: 10^{-3}
Paketverlustwahrscheinlichkeit	p_{Ploss}	1	
Übertragungsverzögerung	d_{trans}	s	Internet: 0.05–0.5 s (typisch)
Verzögerungsschwankung	Δd_{trans}	s	
Bitrate	B	$\frac{\text{bit}}{\text{s}}$	ISDN: $64\frac{\text{kbit}}{\text{s}}$, ATM: $155\frac{\text{Mbit}}{\text{s}}$

Tabelle 2.2: Parameter der Netzdienstgüte

Zunächst kann die *Korrektheit* der Daten erforderlich sein. Man spricht hier von einer erlaubten Bitfehlerwahrscheinlichkeit p_{Berr} (engl. *bit error probability*), also mit welcher Wahrscheinlichkeit der Empfänger statt einer logischen `1` eine logische `0` erhält und umgekehrt. Es kann sinnvoll sein, auf eine aufwändige Reduktion der Fehlerrate zu verzichten, wenn eine höhere Schicht die Übertragungsfehler ausgleichen kann. Diese Art von Fehlern tritt praktisch nur in Schicht 1 (Bitübertragung) auf und stellt nur bei drahtlosen Übertragungen ein großes Problem dar. So arbeiten z. B. optische Übertragungen heute mit Bitfehlerwahrscheinlichkeiten im Bereich von ca. 10^{-8}.

Eine weitere Anforderung ist die *Reihenfolge-Sicherung*. Müssen alle Daten in derselben Reihenfolge ankommen, in der sie abgeschickt wurden, so kann dies bedeuten, dass im Empfänger Daten

in der Schicht 4 (Transport) zwischengespeichert werden müssen, um Umordnungen zu korrigieren. Dieses Problem tritt z. B. in den verbindungslosen IP-Netzen auf, wo für jedes Paket erneut ein Weg gesucht wird. Dabei kann es vorkommen, dass ein später abgesandtes Paket einen kürzeren Weg nimmt und daher eher den Empfänger erreicht als ein früher abgesandtes.

Die *Paketverlustwahrscheinlichkeit* p_{Ploss} (engl. *packet loss probability*) ist eine sehr wichtige Größe bei der Bestimmung der Dienstgüte eines Übertragungsnetzes. Sie bezeichnet die Häufigkeit, mit der Pakete oder Zellen verloren gehen. Ursache für den Verlust eines IP-Pakets kann z. B. ein Übertragungsfehler auf einer tieferen Schicht sein, wodurch das Paket nicht mehr rekonstruiert werden kann oder das Paket aufgrund einer falschen Prüfsumme verworfen wird. Es ist auch möglich, dass die Daten durch einen Fehler in der Steuerinformation den Empfänger nicht erreichen – und stattdessen in einer anderen Verbindung als Falschinformation auftauchen.

In Bezug auf die Übertragung von Echtzeit-Daten ist die *Übertragungsverzögerung* d_{trans} (engl. *transmission delay*) besonders wichtig. Sie gibt an, wie lange Daten brauchen, um vom Sender zum Empfänger zu gelangen (gemessen auf Schicht 3). Verursacht wird die Verzögerung v. a. durch zwei Effekte. Zum einen bestimmt die physikalische Übertragungsrate die Dauer, in der ein Paket zwischen zwei Netzknoten ausgetauscht wird. So erreichen optische Übertragungen Raten im Bereich vieler Mbit/s, der Mobilfunk jedoch nur wenige kbit/s. Der zweite wichtige Faktor ist die Verzögerung, die durch Zwischenspeicherung in den Netzknoten und die Signallaufzeit (*propagation delay time*) verursacht wird.

Durch sich ändernde Netzbedingungen kann die Übertragungsverzögerung variieren, z. B. weil Pakete auf verschieden volle Warteschlangen treffen. Dies wird als *Verzögerungsschwankung* Δd_{trans} (engl. *delay jitter*) bezeichnet, und hat vor allem Auswirkungen auf die Größe von Puffern im Empfänger (s. 3.5.3.6, S. 61). Zudem kann es zur *Büschelbildung* (engl. *bursts*) kommen, d. h. in sehr kurzer Zeit erreichen sehr viele Pakete den Empfänger und die momentane Rate nimmt stark zu.

Eine Übertragung zeichnet sich außerdem durch die verfügbare *Bitrate B* (engl. *bit rate*) aus. Je nach Technologie und Verkehrsvertrag kann das Datennetz eine bestimmte Übertragungsrate garantieren (z. B. ATM) oder alle Daten so gut wie gerade möglich („Best-Effort“) übertragen (z. B. Ethernet, IP). Im ersten Fall kann die Anwendung davon ausgehen, dass die Dienstgüte über die Zeit konstant bleiben wird. Im letzteren Fall jedoch kann sich die vom Netz erbrachte Dienstgüte ändern: Je nachdem, welchen Verkehr das Netz sonst noch übertragen muss, kann es zu Schwankungen kommen. So kann es vorkommen, dass durch eine zusätzliche Übertragung auf einem Teil des Wegs plötzlich nicht mehr genug Bandbreite zur Verfügung steht. Dies wird von der Anwendung erst durch Beobachtung der ankommenden Pakete bemerkt.

2.1.3.2 Prinzipien

Es gibt drei Grundhaltungen, die ein Datennetz gegenüber Dienstgüte einnehmen kann:

„Best-Effort“: Das Netz verspricht keinerlei Dienstgüte, sondern „tut sein momentan Möglichstes“. Diese Haltung ist typisch für IP-Netze, da hier durch das Protokoll keinerlei Dienstgüte-Vorgaben gemacht werden können.

Garantien: Die Anwendung handelt mit dem Netz einen Verkehrsvertrag aus, an den sich beide Seiten halten müssen (Beispiel: ATM). In diesem Vertrag garantiert das Netz z. B. eine Mindest-Rate und eine maximale Verzögerungsschwankung und die Anwendung verpflichtet sich, nicht mehr als eine bestimmte Datenmenge auf einmal ins Netz zu schicken. Sendet

der Sender mehr als er darf, so wird der Überschuss vom Netz als *Best-Effort*-Verkehr behandelt oder sogar verworfen. (Auch bezeichnet als „harte/deterministische Garantie“ [Lu00].)

Priorisierung: Das Netz bietet keine festen Werte für Parameter wie Verlustrate und Verzögerung, sondern garantiert nur eine bevorzugte Behandlung des Verkehrs („weiche/statistische Garantie“ [Lu00]). So werden z. B. bei *DiffServ* (*Differentiated Services*, RFC 2475 [BBCea98]) IP-Pakete in Klassen verschiedener Prioritäten eingeteilt. Müssen nun Pakete im Netz verworfen werden, so werden zunächst nur Pakete mit niedriger Priorität verworfen und jene mit hoher weiter gesandt („relative Dienstgüte“). Dies garantiert aber keine bestimmten Eigenschaften für die Übertragung, da die erbrachte Dienstgüte immer noch von der Art des übrigen Verkehrs und seiner Aufteilung auf die Klassen abhängt.

Stellt eine Anwendung fest, dass die vom Netz erbrachte Übertragungs-Dienstgüte nicht ausreicht, so sind ihre Möglichkeiten abhängig von der Art des verwendeten Netzes. Hält sich das Netz an einen Verkehrsvertrag, so kann die Anwendung versuchen, einen anderen Vertrag auszuhandeln, also z. B. mehr Bandbreite zu reservieren. Gibt das Netz jedoch nur „weiche“ oder gar keine Garantien, und kann es auch nicht anders von der Anwendung beeinflusst werden (z. B. dass es andere Verkehre benachteiligt), so bleibt der Anwendung nur, sich selbst an den Zustand der Übertragung anzupassen (s. 2.3, S. 24) oder auf eine Übertragung ganz zu verzichten.

2.1.3.3 Kosten

Die Kosten einer Übertragung stellen sich für den Netzbetreiber und den Nutzer des Übertragungsdienstes sehr unterschiedlich dar. Der Betreiber achtet vor allem auf die Rentabilität seiner Investitionen in das Netz und wird daher anstreben, die von den Kunden einnehmbaren Entgelte zu maximieren. In seine Berechnungen der Kosten zur Bereitstellung von Dienstgüte gehen die Aufwendungen für Geräte und Leitungen, aber auch für Wartungs- und Management-Aufgaben ein.

Das Ziel des Nutzers ist dagegen, entweder eine bestimmte Dienstgüte zu möglichst geringen Kosten zu erhalten, oder bei festen Kosten die bestmögliche Dienstgüte zu erreichen.

2.1.3.4 Dienstgüte in IP-Netzen

Beim Entwurf der TCP/IP-Architektur (s. 2.1.2.1, S. 7) wurde keine Möglichkeit vorgesehen, mit dem Netz eine Dienstgüte für die Übertragung auszuhandeln. In den letzten Jahren wurden daher verschiedene Ansätze vorgeschlagen, die es einer Anwendung ermöglichen sollen, die Behandlung ihres Verkehrs durch das Netz zu beeinflussen.

Um eine *garantierte Dienstgüte* vereinbaren zu können, wurde *IntServ* (*Integrated Services*, RFC 1633 [BCS94]) entwickelt. *Relative Dienstgüte* kann mittels *DiffServ* (*Differentiated Services*, RFC 2475) erreicht werden. Für eine Übersicht über weitere Ansätze sei auf [XN99] verwiesen.

2.1.4 Dienstgüte in Multimedia-Anwendungen

Nach der Definition auf S. 9 besteht die Dienstgüte einer Anwendung aus mehreren Qualitätsmerkmalen, aus denen eine Gesamtdienstgüte abgeleitet wird. Für eine Multimedia-Anwendung können Parameter verschiedener Kategorien zur Bestimmung herangezogen werden [VKBvG95]:

Leistung: Von der Situation und Konfiguration abhängige Größen, die auch durch externe Einflüsse (Übertragungsnetz) verändert werden können. Beispiele: Ende-zu-Ende–Verzögerung, Bandbreite.

Datenformate: Parameter, die von der Anwendung gewählt werden können. Sie haben direkte Auswirkungen auf die Dienstgüte, z. B. Bildauflösung und -rate, Kodierungsverfahren.

Synchronisierung: Versatz (engl. *skew*) zwischen Medienströmen, d. h. der zeitliche Abstand der Ausspielzeitpunkte von zur selben Zeit erfassten Daten, entweder am selben oder an verschiedenen Endgeräten.

Kosten: Entgelte für die Bereitstellung von Ressourcen, die z. B. für einen Netzzugang oder eine bestimmte garantierte Netz-Dienstgüte an den Netzbetreiber entrichtet werden müssen. Der Betrag kann auch von der Art und der Menge des übertragenen Verkehrs abhängen.

subjektive Dienstgüte: Der Eindruck, den ein menschlicher Benutzer als erfahrene Gesamtdienstgüte der Anwendung wahrnimmt.

Letztlich ist für den Einsatz der Anwendung entscheidend, dass die subjektive Dienstgüte, d. h. der Gebrauchswert für den Nutzer, höher ist als die Summe aller auftretenden Kosten. Da jedoch die subjektive Dienstgüte nicht direkt am Nutzer gemessen wird[4], muss man sie aus Parametern abschätzen, die im Endgerät erhoben werden.

Die subjektive Dienstgüte hängt aber nicht nur von der erbrachten Leistung der Anwendung ab, sondern auch von den *Erwartungen*, die der Nutzer an die Anwendung stellt [IKW00]. Dies bedeutet, dass die Anforderungen nicht immer exakt identisch sind, sondern von der jeweiligen Situation, einem sog. *Szenario*, abhängen. Es kann dabei von einem Benutzer nicht erwartet werden, alle Parameter, die eine Anwendung steuern, festzulegen. Stattdessen müssen diese aus wenigen Informationen ermittelt und vor dem Start der Übertragung eingestellt werden.

Der erste Faktor, der die Bestimmung der Parameter hauptsächlich beeinflusst, ist die *Art der Anwendung*. Wie in 2.2.4.4 beschrieben, gibt es verschiedene Typen und für die meisten von ihnen ist je nach konkretem Einsatzfall eine andere Qualität erforderlich, wie unter 2.2.6 beschrieben. Hieraus kann die Anwendung Anforderungen an die Dienstgüte der einzelnen Medien errechnen.

Im nächsten Schritt muss nun die Anwendung versuchen, die ihr zur Verfügung stehenden Ressourcen so einzusetzen, dass sie die Anforderungen erfüllen kann. Sie bildet also die allgemeinen Parameter der erfahrenen Dienstgüte der einzelnen Medienströme auf konkrete Parameter ab. An dieser Stelle kann die Anwendung auch Ressourcen reservieren, sofern das verwendete System oder Netz dies unterstützt. Die Anwendung verhandelt dann mit Verwaltern (engl. *Resource Manager*), die die verfügbaren Ressourcen auf mehrere Anwendungen aufteilen.

Ist es dagegen der Anwendung nicht möglich, Ressourcen zu reservieren (weil dies vom verwendeten System nicht unterstützt wird oder der Benutzer die Kosten hierfür nicht aufwenden will), so kann die Anwendung nur versuchen, aufgrund von Annahmen über die verfügbaren Ressourcen die konkreten Parameter zu bestimmen. Sie beginnt dann mit der Übertragung, muss dann aber stets ihre Annahmen mit der tatsächlichen Verfügbarkeit vergleichen und gegebenenfalls korrigieren (*Adaption*).

[4] Es gibt Ansätze hierzu durch Messung von physiologischen Parametern, z. B. [WS99b].

2.2 Verteilte Multimedia-Anwendungen

Durch die steigende Leistungsfähigkeit der verfügbaren Computersysteme wurde die Realisierung von Anwendungen möglich, die immer mehr Ressourcen und auch spezielle Hardware benötigen. Erst hierdurch war es möglich, z. B. Videosignale zu digitalisieren und zu bearbeiten, erst moderne Rechner erlaubten die ausreichend schnelle Komprimierung der Daten für eine effiziente Übertragung.

Der Preisverfall der Rechenleistung und der Komponenten wie *Frame-Grabber* erlaubt nun immer mehr Anwendern, die für Multimedia-Anwendungen benötigten Computersysteme zu nutzen, und das sogar für private Zwecke. Hierbei werden zunächst sicher klassische Anwendungen (s. u.) zum Einsatz kommen; jedoch ist es wahrscheinlich, dass im Laufe der Zeit neue Anwendungen entwickelt werden, die bisher nur schemenhaft absehbar sind. Denkbar sind z. B. Anwendungen in der „Virtuellen Realität“ oder solche mit taktiler Rückkopplung.

> **Definition 2.2 (Multimedia):** *Verknüpfung mehrerer Medien, z. B. Bild und Ton.*

Der Begriff „Multimedia“ wurde schon für sehr viele verschiedene Anwendungstypen verwandt; sogar für Textverarbeitungen, die Bilder in Dokumente einbinden konnten. In diesem Projekt werden aber nur *kontinuierliche Medien* (engl. *continuous media*) betrachtet, bei denen über eine lange Zeitdauer hinweg Daten übertragen werden und es eine implizite Ordnung gibt: Ein Datum (also ein Bild oder ein Audio-Abtastwert) steht zu allen anderen Daten des Mediums in einer zeitlichen Beziehung hinsichtlich seines Ausspielzeitpunkts. Dies hat zur Folge, dass nach dem Ausspielen eines Datums alle Daten des Mediums, die früher hätten ausgespielt werden müssen, nicht mehr ausgespielt werden dürfen und somit wertlos geworden sind.

Des Weiteren wird in diesem Projekt nicht immer von *mehreren* Medien ausgegangen. Zur Vereinfachung wird bei bestimmten Anwendungen zunächst nur ein Medium, z. B. ein Audio-Strom, betrachtet, jedoch mit der Absicht, die hier gewonnenen Erkenntnisse auf eine Anwendung mit mehreren Medien zu übertragen.

2.2.1 Datenströme

Verteilte Anwendungen zeichnen sich durch den Austausch von Daten aus. Hierbei werden die Multimedia-Daten in zusammen gehörende *Ströme* unterteilt, die vom Sender zum Empfänger übertragen werden. Innerhalb einer Anwendung können mehrere Ströme auch gleichen Typs vorkommen, also z. B. zwei Audio-Ströme von verschiedenen Sprechern. Ein Strom kann umkodiert, mit einem weiteren gemischt oder an mehrere Empfänger gleichzeitig gesandt werden.

Werden für ein Medium periodisch Daten ausgesandt, z. B. 25 mal in der Sekunde ein Videobild oder alle 80 ms ein Rahmen mit Audio-Daten, so spricht man von *kontinuierlichen* Medien (engl. *continuous media*). Für andere Medien (z. B. *Whiteboard*-Daten, s. u.) wird nur bei Bedarf (hier also wenn der Nutzer eine Eingabe macht) Datenverkehr erzeugt, der somit eine unregelmäßigere Struktur aufweist.

Zunächst seien daher die verschiedenen Arten der übertragenen Daten dargestellt.

Audio-Ströme: Ein Audio-Signal wird kontinuierlich mit einer Abtastrate (engl. *sampling frequency*) abgetastet, wobei jeder Abtastwert (engl. *sample*) digital kodiert wird. Die Abtastwerte werden meist nicht einzeln übertragen, sondern vorher zu Rahmen (engl. *fra-*

mes) zusammengefasst, die – typischerweise verlustbehaftet – komprimiert werden können (s. 2.2.3.4, S. 16).

Video-Ströme: Hierbei wird Bildinformation übertragen. Durch die enorme Datenmenge, die unkomprimierte Bilder darstellen, werden diese normalerweise verlustbehaftet komprimiert (s. 2.2.3.5, S. 18). Zum einen kann es sich dabei um Bilder (engl. *frames*) handeln, die ohne weitere Information angezeigt werden können (engl. *intra-frame coding*). Es gibt aber auch die Möglichkeit, nur die Änderungen relativ zu anderen Bildern zu übertragen, so dass die Anzeige erst zusammen mit diesen möglich ist (engl. *inter-frame coding*).

***Whiteboard*-Daten:** Übertragung von Daten, die zu einer *Whiteboard*-Anwendung (s. 2.2.4.4, S. 20) gehören. Dies umfasst eingegebene Tastendrücke bzw. Text, Bewegungen mit der Maus, grafische Operationen wie das Zeichnen von Linien oder Flächen etc.

Signalisierung: Zwischen den Endgeräten wird neben den eigentlichen Multimedia-Daten auch Steuer-Information ausgetauscht, die zum einen zur Initialisierung der Anwendung, d. h. dem Aushandeln der Parameter der Übertragung, zum anderen auch für die Beschreibung der Daten und zum Austausch von Zustandsinformation benötigt wird.

verschiedene Daten-Ströme: Hierunter fallen alle übrigen, sehr anwendungsabhängigen Typen, z. B. die Texte einer *Chat*-Anwendung, die Steuerinformationen für einen Roboter oder die Beschreibung eines Orts im „virtuellen Raum“.

2.2.2 Ressourcen im Terminal

Für die Erbringung eines Dienstes benötigt eine Anwendung Zugriff auf Ressourcen des Terminals. Dabei handelt es sich einerseits um dedizierte Ressourcen, die das Terminal bereitstellen muss, z. B. ein Mikrofon, Lautsprecher, einen Frame-Grabber, einen Bildschirm, nicht zuletzt ein Netz-Interface usw. Für alle diese Ressourcen kann man Zugriffsprotokolle beschreiben, insbesondere bei konkurrierenden Anwendungen drohen andernfalls Zugriffskonflikte oder sogar eine Verklemmung (engl. *deadlock*).

Neben den dedizierten besitzt das Terminal auch gemeinsam benutzte Ressourcen. Dies bedeutet, dass die Anwendung sich diese Ressourcen prinzipbedingt mit anderen Anwendungen (zumindest dem Betriebssystem) teilen muss; genauer: einzelne *threads* (Kontrollflüsse) der Anwendung müssen die Verwendung dieser Ressourcen untereinander abstimmen.

In diese Kategorie fallen insbesondere Speicher und Rechenleistung. Diese Ressourcen sind vor allem in spezialisierten Endgeräten (z. B. „Set-Top-Boxen“, mobile Multimedia-Telefone) knapp, während moderne *Personal Computer* deutlich besser ausgestattet sind. Die Ressourcenvergabe erfolgt durch ein *scheduling*, d. h. die Vergabe der Ressourcen an einzelne Threads. Um die Einhaltung bestimmter *deadlines* („Termine“) sicher zu stellen, können die Threads entweder priorisiert werden, oder Hilfsfunktionen sorgen dafür, dass das Scheduling entsprechend den Terminen erfolgt (z. B. [NCN98], [DFGG96]).

Abgrenzung

Die Untersuchung der Ressourcenvergabe ist nicht der Hauptpunkt der vorliegenden Arbeit (es sei hier auf [NS95], s. 2.4.3.1, S. 28 verwiesen). Hier wird davon ausgegangen, dass die Anwendung dedizierte Ressourcen stets verwenden kann, und dass genügend Speicher zur Verfügung steht. Bei

der Aufteilung der Rechenleistung wird davon ausgegangen, dass das Scheduling schnell genug arbeitet (im Bereich weniger Millisekunden) und daher hierdurch kaum zusätzliche Wartezeiten entstehen. Somit reicht es aus, die absolute Menge der benötigten Rechenleistung zu betrachten und anzupassen.

2.2.3 Relevante Standards

2.2.3.1 ITU-T H.323

Um die Interoperabilität von Multimedia-Anwendungen zu verbessern, wurden Standards für die Datenformate und den Ablauf der Kommunikation verabschiedet. Besonders hervorzuheben ist hierbei der Standard ITU-T H.323[5], der als Rahmen für diverse Standards (s. u.) der ITU-T und der IETF dient. In kurzer Zeit wurde dieser Standard die Grundlage für viele kommerzielle Anwendungen.

Konkret beschreibt der Standard H.225.0 den Aufbau einer Verbindung und der Standard H.245 die Steuerung der Kommunikation und den Austausch von Steuerparametern, z. B. um eine Video-Übertragung zu starten. Die eigentliche Datenübertragung ist in die Formate der zu übertragenden Daten (bzw. die entsprechenden Kodierverfahren, Tab. 2.3, s. 2.2.3.4, S. 16, s. 2.2.3.5, S. 18) und das Übertragungsprotokoll RTP/RTCP (RFC 1889 [SCFJ96], s. 2.2.3.3, S. 16) aufgeteilt. Als Transportprotokolle können TCP und UDP (s. 2.1.2.1, S. 7) verwandt werden, dies ist jedoch nicht festgelegt. H.323 bietet zudem auch die Möglichkeit, andere Kodierverfahren zu verwenden.

Medien-Typ	Standard	Anmerkung
Audio	G.711, G.723.1,	
	G.728, G.729, ...	
Video	H.261	$n \times 64 \frac{\text{kbit}}{\text{s}}$
	H.263	flexiblere Erweiterung von H.261
Daten	T.120	*CSCW*

Tabelle 2.3: Kodierungsverfahren in H.323

Außerdem spezifiziert H.323 die an einer Übertragung beteiligten „Geräte“, also das Endgerät, *Gateways* (Übergänge zwischen Netzen), *Multipoint Control Units* (zur Unterstützung mehrerer Teilnehmer) und *Gatekeeper* (Zugangskontrolle).

2.2.3.2 SIP

Das *Session Initiation Protocol* (*SIP*, RFC 2543 [HSSR99]) dient ähnlich wie H.323 zum Aufbau und zur Steuerung von Multimedia-Verbindungen zwischen Endgeräten. Es verwendet ebenfalls RTP zum Transport der Daten und weitere Protokolle wie SDP (*Session Description Protocol*, RFC 2327 [HJ98]), die von der IETF spezifiziert wurden. [SR99] bietet einen genaueren Überblick.

[5] [TO99] bietet einen detaillierten Überblick.

2.2.3.3 RTP/RTCP

Ziel des *Real-time Transport Protocols* (*RTP*, RFC 1889 [SCFJ96]) ist es, die Übertragung von Echtzeit-Daten über Datennetze zu unterstützen, indem dem Datenstrom Zusatzinformation beigefügt wird, deren Auswertung mit Hilfe des *RTP Control Protocol* (*RTCP*) zur Steuerung der Übertragung benutzt werden kann. RTP nutzt dabei so genannte *Profiles* zur Beschreibung einer Übertragung, z. B. für Audio- und Video-Konferenzen (RFC 1890, [Sch96]). Außerdem können Konvertierer und Mischer von RTP-Daten benutzt werden. RTP legt jedoch weder das Übertragungsprotokoll fest, noch unterstützt es die Reservierung von Ressourcen.

Es wurde das Prinzip der Rahmenbildung auf Anwendungsebene (engl. *Application Level Framing*, ALF [CT90]) genutzt. Dabei teilt bereits die Anwendung kontinuierliche Datenströme in Rahmen auf und fügt ihnen eigene Nachrichtenköpfe hinzu, hier die RTP-Header. Die Übertragung muss dann sicherstellen, dass die Rahmen wieder in derselben Form den Empfänger erreichen, so dass der Nachrichtenkopf ausgewertet werden kann.

Bei der Übertragung fügt RTP jedem Paket folgende Informationen hinzu: Identifikation der Quelle (und ggf. von Bearbeitern der Daten), Nutzlast-Typ, Sequenznummer, Zeitstempel und ein Markierungsbit, dessen Bedeutung abhängig vom Profil ist. Zusätzlich kann ein Profil weitere Header definieren.

Mit Hilfe des RTCP werden Zusatz-Informationen ausgetauscht: Die Quellen beschreiben sich (Name, E-Mail-Adresse etc.), kündigen ihre Teilnahme an (bzw. das Verlassen) einer Konferenz an oder übertragen anwendungsspezifische Daten.

Zusätzlich kann mit RTCP aber auch der Zustand der Übertragung ermittelt werden. Sender senden regelmäßig eine Beschreibung der Übertragung aus ihrer Sicht an alle Teilnehmer (*sender report*), in der sie mitteilen, in welcher Beziehung die Zeitstempel mit der wirklichen Zeit stehen (im NTP-Format) und wie viele Byte und Pakete sie übertragen haben.

Die Empfänger senden Meldungen (*Receiver Reports*) an die Sender, die die Zahl der verlorenen Pakete, die Sequenznummer des letzten erhaltenen Pakets, die Netz-Verzögerung und -schwankung seit der letzten Meldung enthalten. Mit Hilfe dieser Information kann ein Sender auf eine Verschlechterung der Übertragung reagieren, sobald er diese erkennt.

2.2.3.4 Audio-Kodierung

Ein kontinuierliches Audio-Signal wird für die Übertragung zunächst digitalisiert. Mit Hilfe eines Analog-Digital-Wandlers wird die von einem Mikrofon als elektrische Spannung erfasste Schwingung mit einer bestimmten Abtastfrequenz (*sampling frequency*) abgetastet und der ermittelte Wert (Abtastwert, *sample*) gespeichert. Die Anzahl der bit pro Abtastwert bestimmt (zusammen mit der Abtastfrequenz) die Qualität, mit der das Audiosignal digital gespeichert wird. Für gute Telefonqualität reicht eine Abtastfrequenz von 8000 Hz mit einer 14 bit Genauigkeit aus, während für die Speicherung von Musik heute 44.1 kHz bei 2×16 bit (stereo, CD–Qualität) üblich sind. Es werden aber auch Formate bis zu 192 kHz bei 24 bit großen Abtastwerten (je Kanal) verwandt (DVD–Audio). Die Abtastfrequenz bestimmt dabei die maximale Tonhöhe, die digitalisiert werden kann[6], die Anzahl der bit pro Wert beeinflusst u. a. das so genannte „Quantisierungsrauschen“.

Bevor die Daten übertragen werden, werden sie zunächst aufbereitet. Für eine digitale, verbindungsorientierte Sprachübertragung (z. B. ISDN) werden die Audiodaten kodiert, so dass der Da-

[6] Laut dem Abtasttheorem von *Nyquist* darf das Signal keine Frequenzen größer als die Hälfte der Abtastfrequenz enthalten – ggf. muss ein Tiefpassfilter eingesetzt werden.

tenumfang abnimmt. Hierzu werden die 14 bit großen Abtastwerte in einem 8 bit Code dargestellt, der sich die Eigenschaften des menschlichen Ohrs zu Nutze macht. Dessen Empfindlichkeit nimmt bei lauten Tönen ab, d. h. für die Speicherung lauter Töne können weniger Bits verwendet werden als für die Speicherung leiser, bzw. bei gleicher Bitzahl können einem Originalwert größere Amplitudenintervalle zugeordnet werden. Hierzu wurde das „μ-law"–Verfahren entwickelt (ähnlich: „A-law"), das eine annähernd logarithmische Kennlinie verwendet, um die Zuordnung zwischen Abtastwert und Code-Wert herzustellen.

Wird eine verbindungslose Übertragung verwendet, so müssen die Daten in Pakete aufgeteilt werden. Dies bedeutet, dass Audio-Daten in *Rahmen* (engl. *frames*) zusammen gefasst werden, die z. B. 20, 40 oder 80 ms entsprechen. Um auch hier den Datenumfang zu reduzieren, kann man weitere Eigenschaften des menschlichen Hörapparats nutzen, z. B. dass laute Töne leisere mit ähnlich hoher Frequenz unhörbar machen. Ein Rahmen mit Audio-Daten wird dann komprimiert, indem ein Algorithmus die Daten spektral analysiert und bestimmte Frequenzen herausgefiltert werden.

Eine weitere Möglichkeit ist die Berücksichtigung der Art der Daten. So kommen in Sprachdaten nur sehr selten große Sprünge zwischen Abtastwerten vor, d. h. meist ist es effizienter, nur die Unterschiede zwischen den Werten zu übertragen (engl. *differential coding*). Zudem gibt es Verfahren, die den Verlauf des Audio-Signals „vorhersagen" können, so dass nur die Unterschiede zwischen den vorhergesagten und den tatsächlichen Werten übertragen werden müssen (engl. *predictive coding*).

Die Wahl des Kodierverfahrens hat Auswirkungen auf die Art des erzeugten Verkehrs und die vom Nutzer erfahrene Dienstgüte des übertragenen Audio-Stroms. Die maximale Qualität des Signals hängt von der Art der Abtastung und der Kodierung ab; hinzu kommen Faktoren wie Kodierungs-Verzögerungen und die Toleranz der Kodierung gegenüber Verlusten. Tab. 2.4 bietet einen Überblick über gebräuchliche Standards zur Audio-Kodierung (s. a. [Cox97], [IT96a]).

Codec	Bitrate/$\frac{\text{kbit}}{\text{s}}$	Prinzip	Qualität
G.711	64	8 kHz, μ-law	sehr gut
G.722	48, 56, 64	16 kHz, logarithmisch, vorhergesagt, differenziell	gut
G.723.1	5.3, 6.3	lineare Vorhersage/Quantisierung	ausreichend
G.728	12.8, 16	lineare Vorhersage mit niedriger Verzögerung	ausreichend/gut
G.729	8	lineare Vorhersage	gut
GSM	13	Langzeit-Vorhersage	ausreichend

Tabelle 2.4: Standards zur Audio-Kodierung

2.2.3.5 Video-Kodierung

Um einen Bildstrom über ein digitales Netz zu übertragen, müssen zunächst die einzelnen Bilder erfasst werden[7]. Ein „Frame-Grabber“ wandelt hierzu das analoge Video-Signal einer Kamera bildweise in eine digitale Darstellung um. Hierbei kann die Anzahl der erfassten Bilder pro Sekunde, die Bildgröße und die Art der Darstellung eines Bildpunktes (engl. *pixel*) beeinflusst werden, ausgehend vom Format des Video-Signals (z. B. PAL: 50 Halbbilder pro Sekunde, 625 Zeilen à 864 Punkte; NTSC: 59.94 Halbbilder pro Sekunde, 525 Zeilen à 864 Punkte; HDTV: bis zu 30 Vollbilder pro Sekunde und 1080 Zeilen à 1920 Punkten).

Nachdem diese in digitaler Form vorliegen, können sie mit verschiedenen Verfahren komprimiert werden. Hierzu werden einerseits Algorithmen eingesetzt, die für die Speicherung von Einzelbildern entwickelt wurden (Motion-JPEG [JPE]). Andererseits macht man sich zu Nutze, dass sich zwischen aufeinander folgenden Bildern meist nur wenig verändert. Man berechnet dann die Unterschiede und überträgt nur diese. Bekannte Verfahren hierfür sind MPEG und die H.26x-Standards der ITU-T (s. u.).

Bei der Auswahl des Kodierverfahrens (Tab. 2.5) spielen neben der Art des zu übertragenden Video-Stroms die Faktoren Berechnungsaufwand und Verzögerungszeit eine Rolle. So liefert das MPEG-2–Verfahren zwar eine sehr hohe Video-Qualität bei relativ niedriger Datenrate, jedoch werden für die Berechnung der Unterschiede mehrere Bilder benötigt, so dass die algorithmische Verzögerung weit über 100 ms betragen muss, was dieses Verfahren für interaktive Anwendungen weniger geeignet macht.

Verfahren	Bitrate (typisch)	Bildqualität
unkomprimiert	> 2.5 Mbit/s	sehr gut
Motion-JPEG	100 kbit/s bis 2 Mbit/s	gering bis gut
MPEG	500 kbit/s bis 6 Mbit/s	gut bis sehr gut
H.261	$n \times 64$ kbit/s	mittel bis gut
H.263	28.8 bis 300 kbit/s	gering bis gut

Tabelle 2.5: Standards der Video-Kodierung

2.2.3.5.1 ITU-T H.261, H.263 Video-Kodierung

Der 1990 verabschiedete Standard H.261 definiert die Kodierung eines niederbitratigen Video-Stroms. Er legt die verwendbaren Bildformate fest (z. B. „*Common Intermediate Format*“, CIF, 288 Zeilen à 352 Pixel; „*Quarter-CIF*“, QCIF, 176×144) und die Art der Kodierung der Blöcke, aus denen ein Rahmen aufgebaut ist. Die Kodierung der Bildinformation erfolgt durch eine diskrete Kosinus-Transformation (engl. *Discrete Cosine Transformation*, *DCT*) mit verlustbehafteter Quantisierung und anschließender Komprimierung ([Ste99], S. 145 ff.).

Zudem besteht die Möglichkeit, nur die Änderungen zwischen aufeinander folgenden Rahmen zu übertragen (*inter-frame coding*). Hierzu wird ein Mechanismus definiert, mit dem man die neue Position eines Blocks im nächsten Bild angeben kann („*motion compensation*“). Eine genauere Beschreibung dieses Vorgangs findet sich in [Rij96].

[7] Es wird also *nicht* analog zu Audio-Übertragungen das Videosignal der Kamera direkt digitalisiert, wie das z. B. bei Speicherung auf der sog. „Laser-Disc“ der Fall war.

Später wurden die erweiterten und verbesserten Kodierungen H.263 und H.263L/+ definiert, die die benötigte Rate um bis zu 50 % reduzieren können. Anwendungen können für diese Kodierungen optionale Erweiterungen aushandeln; zudem stehen zusätzliche Bildformate (SQCIF, 4CIF, 16CIF) zur Verfügung.

2.2.4 Anwendungstypen

Multimedia-Anwendungen lassen sich grundsätzlich in folgende beiden Kategorien einteilen:

- lokal: Alle verarbeiteten Daten befinden sich bereits am Ort der Verwendung oder werden dort erzeugt bzw. erfasst. Beispiele hierfür sind Multimedia-Lexika, Spiele und Unterhaltungs-Programme wie Filme auf DVD (*Digital Versatile Disk*).
- verteilt: Die Daten werden zwischen zwei oder mehreren Rechnern ausgetauscht, bevor sie dem Nutzer präsentiert werden.

sowie

- vorproduziert, präsentierend: Die Inhalte liegen bereits beim Start der Anwendung vor und müssen nur den Wünschen des Nutzers entsprechend angezeigt werden.
- dynamisch erzeugt, dialogorientiert (interaktiv): Die Inhalte werden erst kurz vor der Anzeige, in Abhängigkeit von Nutzereingaben, erzeugt bzw. erfasst.

2.2.4.1 Lokale, vorproduzierte Anwendungen

Anwendungen diesen Typs laufen ausschließlich im Endgerät ab und es wurde bereits vor dem Start festgelegt, aus welcher Inhaltsmenge der Nutzer auswählen kann.

Multimedia-Präsentation: Hierbei handelt es sich um eine Aufbereitung eines Themengebietes, wobei verschiedene Medien verknüpft werden. Beispiele sind Lexika, die neben Texten auch Film- und Tondokumente anbieten, oder auch Präsentationen, die ein Produkt vorstellen und seine Verwendung erklären.
Die Daten liegen bereits auf einem Massenspeicher (z. B. CD-ROM, DVD) beim Endgerät vor.

Spiele: Moderne Computerspiele zeichnen sich meist durch die Verwendung von Film- und Tonausschnitten und computergenerierter Grafik aus, die von einem Massenspeicher (z. B. CD-ROM, DVD) gelesen werden.

2.2.4.2 Lokale Anwendungen mit dynamischem Inhalt

Auch diese Anwendungen laufen ausschließlich im Endgerät ab, jedoch wird der Inhalt erst während der Interaktion mit dem Nutzer erzeugt.

Konferenzraum-Unterstützung: Diese Anwendung soll die Arbeit einer Gruppe in einem Konferenzraum technisch unterstützen. So kann z. B. ein Teilnehmer für die Gruppe die Diskussion begleitend aufzeichnen, so dass alle auf den Mitschrieb zugreifen und z. B. sofort Änderungen und Anmerkungen anbringen können. Es können auch vorbereitete Dokumente abgerufen und dargestellt werden.

***Virtual Reality*-Spiele:** Hier wird vom lokalen Rechner eine „virtuelle Welt“ erzeugt, in der sich der Spieler (bzw. sein Repräsentant oder „Avatar“) bewegt.

2.2.4.3 Verteilte, vorproduzierte Anwendungen

Kennzeichen dieser Anwendungen ist, dass vorbereitete Inhalte über ein Kommunikationsnetz zum Endgerät transportiert werden.

***Video-on-Demand* (VoD), Abrufdienste:** Nutzer können Videofilme (oder auch Musikstücke) von einem Anbieter zu beliebigen Zeitpunkten anfordern und zu sich übertragen lassen. Es besteht die Möglichkeit, das Ausspielen zu beeinflussen, z. B. den Film zu stoppen oder vorzuspulen. Üblicherweise ist dieser Dienst kostenpflichtig, daher wird TV- bzw. „HiFi“-Qualität von ihm erwartet.

***Near-Video-on-Demand*, Verteildienste:** Diese Anwendungen tragen Ähnlichkeiten mit der Ausstrahlung klassischer Fernseh- bzw. Radioprogramme. Die Nutzer können nicht direkt bestimmte Inhalte anfordern, jedoch aus vielen Angeboten auswählen und zu periodisch wiederkehrenden Zeitpunkten empfangen.

Multimedia-Datenbanken: Ähnlich einer Multimedia-Präsentation können hier vorbereitete Inhalte abgerufen werden. Die Inhalte liegen jedoch nicht im Endgerät bereit, sondern werden erst nach Veranlassung durch den Nutzer von einem Datenbank-Server übertragen. Meist wird hierfür eine grafische Benutzungsoberfläche angeboten.

2.2.4.4 Verteilte Anwendungen mit dynamischem Inhalt

Hier werden Inhalte übertragen, die erst durch die Interaktion mit dem Nutzer (bzw. den Nutzern) entstehen.

(Video-) Telephonie: Anwendung, bei der zwei Menschen miteinander interaktiv kommunizieren. Es wird ein Audio-Strom übertragen; bei der Video-Telephonie zusätzlich ein Bild-Strom. Diese Anwendung ist die Übertragung und Weiterentwicklung des klassischen, einfachen Telefondienstes auf moderne Datennetze.

Video-Konferenz: Eine Besprechung kann mit einer Anwendung diesen Typs räumlich verteilt durchgeführt werden. Es werden Audio- und Video-Ströme ausgetauscht. Zudem wird häufig eine CSCW-Komponente (s. 2.2.5, S. 21) wie *Whiteboard* oder *Application Sharing* eingesetzt.

***Whiteboard*:** Dies ist eine elektronische Version der klassischen Wandtafel. Nutzer an verschiedenen Orten können Änderungen und Erweiterungen an einem gemeinsamen Aufschrieb vornehmen, wobei sie Texte eingeben und Grafik anbringen können.

***Application Sharing*:** Dieses Werkzeug erlaubt es, eine Anwendung, die auf einem Rechner läuft, von einem anderen Rechner aus zu bedienen. Hierzu werden alle Eingaben, die am entfernten Rechner gemacht werden, zu dem Rechner übertragen, auf dem die Anwendung läuft. Ebenso wird alles, was die Anwendung anzeigt, zu dem entfernten Rechner übertragen.

***Tele-Teaching*:** In seiner extremsten Form hat diese Anwendung zum Ziel, mehrere Hörsäle zu einem einzigen zusammenzuschließen. Alle Hörer sollen dabei der Veranstaltung so gut folgen können, als wären sie mit dem Vortragenden im selben Raum. In jedem der Säle soll dabei jeder Hörer alle Möglichkeiten zur Interaktion mit dem Vortragenden und den anderen Hörern haben, z. B. um Fragen zu stellen. Es gibt vereinfachte Varianten, bei denen z. B. kein ständiger Kontakt zwischen den Hörern in verschiedenen Sälen besteht.

Netz-Spiele: Bei diesen Spielen sind die Teilnehmer räumlich getrennt und es werden Daten über ihre Aktionen ausgetauscht. Je nach Typ kann dies einfache oder komplexe Meldungen umfassen und von weiteren Medien, wie einem Audio-Strom, begleitet sein. Beispiele sind Spiele in der „Virtuellen Realität“, in der sich für jeden Spieler ein Repräsentant, ein so genannter „Avatar“, bewegt und mit anderen Avataren interagieren kann.

Telemanipulation (auch *Tele-Robotik*, *Tele-Medizin*): Dies bezeichnet die Fernsteuerung eines Vorgangs durch einen menschlichen Bediener. Dieser muss über entsprechende Rückmeldung den Zustand des Vorgangs erfassen und darauf reagieren können. Meist bedeutet dies die Fernsteuerung eines Roboters, z. B. zur Durchführung von Arbeiten an gefährlichen oder schwer zugänglichen Orten.

2.2.4.5 Abgrenzung

Das vorliegende Projekt beschränkt sich auf die Betrachtung von *verteilten, interaktiven Multimedia-Anwendungen mit dynamisch generiertem Inhalt*.

Dies hat die Auswirkung, dass die Daten nicht im Endgerät vorliegen oder erzeugt werden. Es kann somit davon ausgegangen werden, dass ein Übertragungsnetz benutzt wird, dessen Charakteristik die von der Anwendung erbrachte Dienstgüte beeinflusst.

2.2.5 *CSCW*

Unter *CSCW* (*Computer Supported Collaborative Work*, Tab. 2.6) versteht man die Unterstützung einer Gruppe von Menschen bei der Bewältigung einer komplexen Aufgabe. Die beteiligten Personen können dabei sowohl räumlich als auch zeitlich getrennt voneinander arbeiten.

	gleiche Zeit	verschiedene Zeit
gleicher Ort	Konferenzraum	*Workflow*-Steuerung
	Whiteboard [a]	*Message-Board*
verschiedener Ort	Video-Konferenz	*E-Mail*
	Telephonie	*News*
	Whiteboard [a]	
	Tele-Teaching	

[a] Diese Anwendung kann lokal und verteilt eingesetzt werden.

Tabelle 2.6: CSCW-Anwendungen

Die Anwendungen, bei denen zur gleichen Zeit gearbeitet wird, wurden bereits in 2.2.4.2 und 2.2.4.4 genauer vorgestellt. Die restlichen Anwendungstypen sind:

Steuerung eines Arbeitsablaufs (engl. *Workflow*): Hier wird die Bearbeitung eines Auftrags mit Computerunterstützung organisiert. Zum Beispiel wird eine Akte von Bearbeiter zu Bearbeiter in einer bestimmten Reihenfolge weitergeleitet oder ein Werkstück nacheinander zu verschiedenen Maschinen gebracht.

***Message-Board*:** Elektronische Form einer Pinnwand, an der Nachrichten und Mitteilungen abgelegt und eingesehen werden können. Dabei kann der Zugriff auf einzelne Nachrichten auf bestimmte Personen(gruppen) eingeschränkt werden.

***Electronic Mail* (E-Mail):** Übertragung von Dokumenten durch ein Datennetz. Der Sender gibt die Adresse des oder der Empfänger an und nur dieser erhält die Nachricht. Die Übertragung kann dabei wenige Sekunden, aber auch mehrere Stunden dauern.

***News*:** Bei diesem System werden Dokumente nicht an eine bestimmte Person gerichtet, sondern unter einem Oberbegriff auf einem so genannten *News-Server* abgelegt. Es existieren zudem Verbünde von Servern, z. B. das „Usenet“, die die Dokumente weltweit verbreiten und Interessierten zum Zugriff anbieten. Typischerweise dauert die Verbreitung der Nachrichten im Usenet einige Stunden.

2.2.6 Beispiel-Szenarien

Nach dem Überblick über die Medien und grundlegenden Anwendungstypen werden hier einige Szenarien genauer vorgestellt.

2.2.6.1 Video-Konferenz

Ziel einer Video-Konferenz ist es, den Beteiligten Reisen zu ersparen. Die Teilnehmer müssen nicht zu einem Konferenzort fahren, sondern können von ihrer Arbeitsstelle oder sogar ihrer Privatwohnung aus die Besprechung durchführen.

Bestandteile einer Video-Konferenz sind zunächst die übertragenen Video-Ströme. Parallel dazu wird jeweils ein Audio-Strom übertragen, wobei Lippensynchronität zwischen Audio und Video erreicht werden muss, d. h. das Ausspielen der Ströme muss zeitlich aufeinander abgestimmt werden. Menschen empfinden dabei einen Versatz (engl. *skew*) zwischen Audio und Video von unter 80 ms als nicht störend [Ste96]. Ein weiterer Datenstrom kann von einer Whiteboard-Anwendung generiert werden. Nehmen nun mehr als zwei Teilnehmer an einer Video-Konferenz teil, so wird die Übertragung der Datenströme schwieriger. Der offensichtliche Ansatz, jeden Datenstrom an alle Teilnehmer zu senden, führt schnell zu einer sehr großen zu transportierenden Datenmenge, da bei n Teilnehmern $n(n-1)$ Ströme übertragen würden.

Um Ressourcen im Netz einzusparen, gibt es einerseits intelligente Multicast-Verfahren (Ein Beispiel für die Untersuchung von IP-Multicast findet sich in [LAP99], s. 3.5.2.6, S. 54.), andererseits können spezielle Verarbeitungsknoten in die Übertragung eingeführt werden. So genannte *Multicast Control Units* (MCU) können mehrere Datenströme zu einem gemeinsamen mischen und an die Teilnehmer senden, so dass jeweils nur ein Strom von jedem Teilnehmer zur MCU und ein Strom von der MCU zum Teilnehmer übertragen werden muss, insgesamt also nur $2n$ Ströme. Das Mischen kann dabei einfach in der Auswahl eines der angelieferten Ströme bestehen oder auch die Dekodierung der Video-Ströme, die Anordnung der verkleinerten Bilder in einem neuen Bild und die Kodierung desselben umfassen.

Der Anspruch an die Qualität der übertragenen Medien kann sehr unterschiedlich sein. In einem Arbeitsszenario ist es das Ziel, gemeinsam ein Dokument zu erstellen, sei es ein Text, eine Zeichnung oder das Objektdiagramm eines Programms. Hierzu kann meist auf ein gutes Videobild verzichtet werden, jedoch sind die Audio-Übertragung und die Whiteboard-Funktion besonders wichtig.

Anders bei einer Besprechung, in der es um die Präsentation der Teilnehmer selbst geht. Hierzu muss versucht werden, sowohl die Audio- als auch die Videoübertragung auf einem sehr hohen Niveau zu halten. Besonders in Fällen, in denen sich die Teilnehmer noch nie persönlich begegnet sind, ist es wichtig, alle Nuancen auch der Körpersprache zu erkennen. Typischerweise wird hier weniger Wert auf die Zusammenarbeit an einem Dokument gelegt, so dass die Whiteboard-Funktion hier nicht so wichtig ist.

Für die verwendeten Übertragungsnetze bedeutet dies im ersten Fall, dass auf die Reservierung von Ressourcen verzichtet wird, um Kosten einzusparen. Sollte sich dann bei der Übertragung eine Reduktion der Qualität einstellen, so wird oft die Videoqualität reduziert oder ganz auf eine Video-Übertragung verzichtet. Wird jedoch Wert auf eine hohe Dienstgüte gelegt, so sind die Beteiligten meist auch bereit, für die Bereitstellung von Ressourcen zu bezahlen.

2.2.6.2 Tele-Manipulation

In diesem Szenario steuert ein Bediener z. B. einen Roboter fern. Dies kann ein Roboter in einem Kernkraftwerk sein, ein Greifarm in einer die Erde umkreisenden Raumstation oder auch ein medizinisches Gerät zum Durchführen von Operationen.

Um den Roboter zu überwachen, müssen Informationen über seine Position und seine Umgebung übertragen werden. Dies kann durch eine stereoskopische Videoübertragung vom Ort der Aufstellung des Roboters geschehen. Der Bediener hat dann einen räumlichen Eindruck und kann den Roboter entsprechend steuern[8]. Des Weiteren kann der Bediener eine taktile Rückkopplung erfahren, indem die vom Roboter erfahrenen Kräfte an den Bedieninstrumenten wiedergegeben werden. Dies erleichtert z. B. einem Chirurgen die Durchführung von Operationen.

Die benötigte Bandbreite unterscheidet sich dabei je nach Richtung sehr stark. *Zum Roboter* muss nur Steuerinformation übertragen werden, dies jedoch fehlerfrei und rechtzeitig, während die Zustandsdaten und Videobilder *vom Roboter* deutlich größeren Umfang haben.

Bedingt durch die hohe Wichtigkeit der ausgeführten Tätigkeiten wird hier natürlich auf größtmögliche Zuverlässigkeit Wert gelegt und die Reduktion der Kosten der Übertragung nicht als vordringlich angesehen. Daher kann davon ausgegangen werden, dass für diese Dienste eine Mindest-Dienstgüte zur Verfügung gestellt wird, also eine Reservierung von Ressourcen stattfindet.

2.2.6.3 Spiele

Die ersten Spiele, die über ein verteiltes Computersystem gespielt wurden, gab es bereits ca. 1970 (Beispiel: „Netrek“, http://www.netrek.org). Sie nutzten die damals verfügbare Technik, also Text-Terminals, und erlaubten bereits vielen Mitspielern, in einer simulierten Welt zu interagieren. Im Laufe der Zeit war es möglich, durch grafische Benutzungsoberflächen immer komplexere Spiele zu entwickeln (Beispiel: „FreeCiv“, http://www.freeciv.org). Den meisten Spielen ist gemeinsam,

[8] Diese Steuerung wird komplizierter, wenn die Übertragungsverzögerung sehr groß ist, wie es bei Raumstationen meist der Fall ist. Hierfür werden dann aufwändigere Lösungen benötigt, die die Position von Objekten vorhersagen können [Hir95].

dass die Spieler *Client*-Programme starten, und Daten über einen *Server* austauschen. Die übertragene Datenmenge ist dabei relativ gering, da sie nur aus Zustands- und Steuerinformation besteht. Da diese Spiele meist rundenbasiert sind, stellen sie keine große Anforderung an die Übertragungsverzögerung.

Mit der Entwicklung der „Heim-Computer“ und dann der Verbreitung der „Personal Computer“ (PC) im häuslichen Umfeld, wurde vermehrt Wert auf eine ansprechende grafische Darstellung gelegt. Unter anderem hierdurch entstand ein sehr großer Markt, der die Entwicklung immer leistungsfähigerer Grafikkarten und einen rapiden Preisverfall der Rechenleistung zur Folge hatte.

Die hohe grafische Qualität heutiger, nicht-verteilter Computerspiele führte zu Bestrebungen, verteilte Spiele mit vergleichbarer Qualität zu entwickeln. Da gleichzeitig Hardware zum Aufbau von LANs immer kostengünstiger wurde, hatten Spiele wie „Quake“ (http://www.idsoftware.com) großen Erfolg[9].

Die nächste Stufe der Entwicklung sind Spiele in der „Virtuellen Realität“. Hierbei soll ein möglichst realistischer Eindruck eines Raums und von Personen erzeugt werden, um eine „natürliche“ Interaktion zu erlauben. Dies hat dann zur Folge, dass Sprachdaten (also Audio-Ströme) übertragen werden müssen, aber auch komplexere Daten wie die Lage von Objekten im Raum. Zusätzlich entstehen dann strengere Anforderungen an Übertragungsverzögerungen als dies bei den einfachen, rundenbasierten Spielen der Fall ist.

Um diese Art Spiele zum Erfolg werden zu lassen, ist eine effiziente und kostengünstige Übertragung von großer Wichtigkeit, da Privatpersonen mit sehr hoher Wahrscheinlichkeit nicht bereit sein werden, für die Reservierung von Ressourcen für verteilte Spiele zu bezahlen. Dies lässt der kommerzielle Misserfolg von „Video-on-Demand“-Diensten in der Vergangenheit vermuten. Aufgrund seiner Verbreitung bietet sich somit das Internet als Transportnetz an, wobei jedoch durch das Fehlen einer Möglichkeit zur Reservierung von Ressourcen die Spiele selbst für einen Ausgleich der Unzulänglichkeiten des Netzes sorgen müssen.

2.3 Prinzipieller Ablauf der Anpassung in Multimedia-Anwendungen

In diesem Abschnitt wird der grundlegende Ablauf der Anpassung in Multimedia-Anwendungen vorgestellt. Eine genauere Betrachtung der Algorithmen und die Klassifizierung und Vorstellung von Mechanismen erfolgt in Kapitel 3.

2.3.1 Aufbau der Multimedia-Übertragung

Beim Start einer verteilten Multimedia-Anwendung wird zunächst die Startkonfiguration (Abb. 2.2) ermittelt. Hierzu wird aus dem Szenario, d. h. der Art der Anwendung, die der Nutzer wünscht, und den vorhandenen Ressourcen eine Festlegung verschiedener Parameter der Anwendung vorgenommen (s. 2.1.4, S. 11). Danach kann die Übertragung gestartet werden.

2.3.2 Adaptionsalgorithmen

Während der Übertragung überwacht die Anwendung die erreichte Dienstgüte und prüft regelmäßig, ob sie der erwarteten Dienstgüte entspricht. Ist dies nicht der Fall, so wird ein Adaptionsal-

[9] [Bor00] untersucht den von diesem Spiel generierten Netzverkehr und zieht das Fazit, dass hier die Endgeräte großen Einfluss auf den Dienstgüte haben, da die Reaktionsgeschwindigkeit von der Anzeigegeschwindigkeit abhängt. In dieser Anwendung ist somit das Ziel, allen Spielern gleiche Siegeschancen einzuräumen (Fairness), nicht erfüllt.

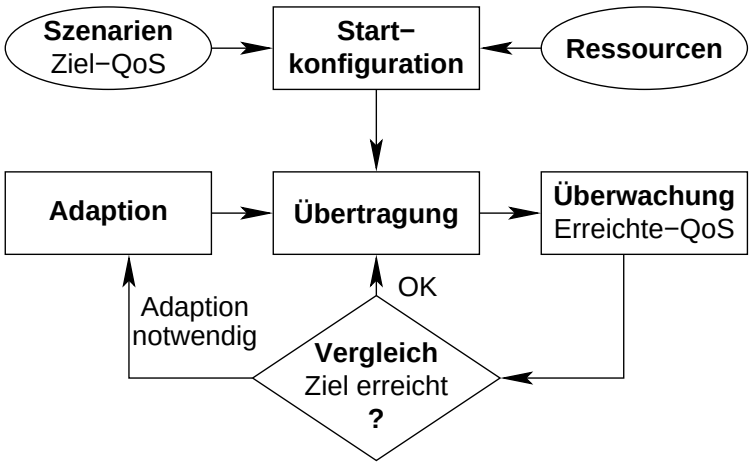


Abbildung 2.2: Adaption im Gesamtsystem

gorithmus ausgeführt, der durch Änderung des Verhaltens der Anwendung die Nutzer-Dienstgüte erhöhen soll. Der Ablauf der Anpassung erfolgt dann in den folgenden Schritten; eine genauere Betrachtung findet sich unter 3.2.

1. Erkennen des Problems (*Symptom*)

   Im ersten Schritt muss der Algorithmus feststellen, welcher Effekt für die reduzierte Dienstgüte verantwortlich ist. Hierzu muss der Algorithmus die von der Anwendung ermittelten Zustandsparameter auswerten, z. B. die Zahl der aufgetretenen Ausspielpausen in einer Audio-Übertragung. Eine hohe Zahl an Pausen kann zu „Knacken“ führen und die vom Nutzer erfahrene Dienstgüte stark beeinträchtigen.

2. Zustandsermittlung (*Ursache*)

   Um effiziente Gegenmaßnahmen treffen zu können, muss der Algorithmus ermitteln, in welchem Zustand sich gerade das Netz und die Anwendung befinden. Aus den Zustandsparametern bildet der Algorithmus nun eine Einschätzung des Zustand von Netz und Anwendung, z. B. über den Puffer-Füllstand der Sound-Karte und die Schwankung der Netz-Verzögerung (*Jitter*). Ist ersterer sehr niedrig, so kann ein hoher Jitter zu einem „Puffer-Unterlauf“ führen, weil die Sound-Karte für einige Zeit keine Daten hat, die sie ausspielen kann.

3. Auswahl des einzusetzenden *Mechanismus*

   Je nach Problem und Zustand der Anwendung und des Netzes muss der Algorithmus einen Mechanismus auswählen, dessen Einsatz die Dienstgüte verbessern soll. Im Beispiel könnte das die Beeinflussung eines dem Puffer der Sound-Karte vorgeschalteten Puffers sein, der Audio-Rahmen künftig länger speichern soll, bevor sie an die Sound-Karte übergeben werden.

4. Durchführen der Adaption

   Der Adaptionsalgorithmus muss die Durchführung der Anpassung in der Anwendung anstoßen. Dies kann z. B. über den Aufruf entsprechender Funktionen erfolgen, oder auch durch das Ändern von Steuerparametern.

2.3.3 Bezug zur Regelungstechnik

Regelungstechnik (s. [Böt98]) ist eine Grundlage der Automatisierungstechnik. Technische Vorgänge werden von ihr beschrieben, analysiert und auf ein Modell abgebildet. In einem „geschlossenen Wirkungsablauf“ (DIN 19226) läuft ein Prozess der Form *messen – rechnen – steuern – regeln* ab. Wird der Prozess durch eine Rückführung (engl. *feedback*) geschlossen, so spricht man von einem *Regelkreis* (Abb. 2.3).

In einem Regelkreis wird versucht, die *Regelgröße* (Istwert) einer *Führungsgröße* (Sollwert) möglichst gut anzunähern. Hierzu wird die Differenz zwischen Ist- und Sollwert ermittelt (bzw. abgeschätzt) und die *Stellgröße* eines *Reglers* um einen daraus berechneten Wert verändert.

Von außen wirken *Störgrößen* auf die *Regelstrecke* ein, die die Abbildung der Stellgröße auf die Regelgröße beeinflussen. Ebenso wird die Messung der Regelgröße stets durch Messrauschen gestört.

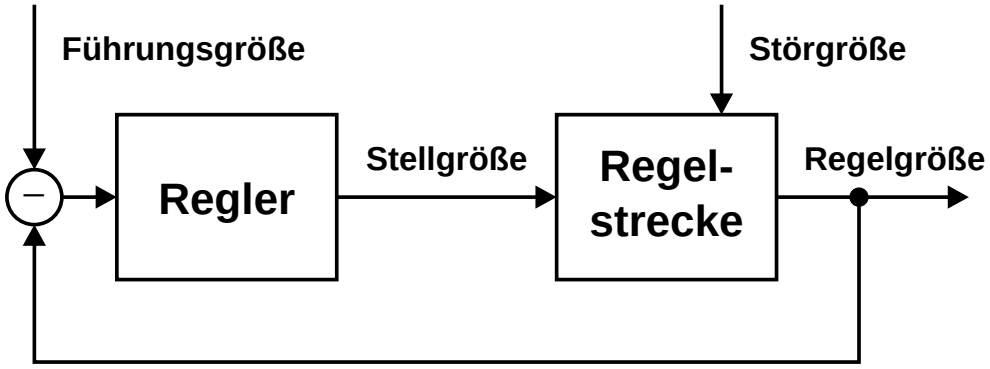


Abbildung 2.3: Allgemeiner Regelkreis

Offensichtlich handelt es sich bei der Anpassung einer Multimedia-Übertragung um einen regelungstechnischen Vorgang. Für das vorliegende Projekt ist besonders der Vergleich der möglichen Reaktionen interessant. In der Regelungstechnik wird aus der Differenz zwischen Soll- und Istwert eine Änderung der Stellgröße berechnet. Dieser Vorgang ist natürlich auch bei einer Multimedia-Übertragung notwendig, aber während bei den meisten Regelungen Regel- und Führungsgrößen mit *kontinuierlichen* Wertebereichen vorliegen (Beispiel: Spannung, Kraft, Geschwindigkeit), sind die Wertebereiche bei einer Multimedia-Übertragung durch Randbedingungen meist diskret. So steht z. B. nur eine kleine, begrenzte Anzahl an Kodierungen zur Verfügung, so dass es nicht sinnvoll erscheint, eine „Linearisierung im Arbeitspunkt“ vorzunehmen. Daher scheint sehr fraglich, ob Standard-Verfahren der Regelungstechnik wie „PID“ (*proportional-integral-derivative*) auf dieses Problem angewandt werden können.

Ein passenderer Ansatz scheint jedoch eine Abwandlung der „Unscharfen Regelung“ (*Fuzzy Control*, [Böt98] S. 189–194) zu sein. Diese Technologie erlaubt es, eine umgangssprachlich formulierte Regelbasis zur Auswahl der Reaktion zu verwenden. Hier kann also „Expertenwissen“ genutzt werden, um die Entscheidung für eine Aktion zu treffen.

Dazu werden zunächst die Werte der Zustandsmessung kategorisiert, d. h. es wird eine unscharfe Zustandsbeschreibung gebildet („Fuzzifizierung“). Diese sagt aus, wie stark verschiedene Aussagen über einen Parameter zutreffen, z. B. ob sein Wert „sehr klein“, „klein“, „mittel“, „groß“ oder

„sehr groß“ ist. Mit Hilfe einer Regelbasis wird aus dem unscharfen Zustand dann die auszuführende Aktion abgeleitet. Eine Regel könnte zum Beispiel lauten:

> „Wenn Parameter p_1 sehr klein ist und Parameter p_2 sehr groß ist, dann wähle Parameter y_1 klein.“

Es muss danach ein konkreter Wert für ein „kleines y_1“ gefunden werden („Defuzzifizierung“), der dann als neue Stellgröße dient.

2.4 Multimedia-Anwendungen und -Frameworks

In den vergangenen Jahren wurden viele Multimedia-Anwendungen[10] (einsetzbare Programmsysteme) und *-Frameworks* bzw. *-Middlewares* entwickelt. Hier sollen deren Ansätze und einige Systeme beispielhaft vorgestellt und hinsichtlich ihrer Relevanz für die vorliegende Untersuchung beurteilt werden.

2.4.1 Klassifizierung

Bei der Entwicklung dieser Systeme wurden meist eines oder mehrere der folgenden Ziele verfolgt:

- Einfacher Aufbau neuer Anwendungen:
  Das Framework stellt z. B. Komponenten bereit, die über eine einfach zu bedienende Benutzungsoberfläche auch von Laien zu komplexen Multimedia-Anwendungen zusammengefügt werden können. Beispiel: *CINEMA* (s. 2.4.2.1, S. 28, [BDHea93]). Oder das Framework unterstützt zusätzlich besondere Merkmale wie Verschlüsselung und Authentifizierung. Beispiel: *Da CaPo++* (s. 2.4.2.2, S. 28, [SCWea99]).

- Unterstützung bei der Formulierung von Dienstgüten-Anforderungen und von Dienstgüten-Garantien:
  Das Framework stellt einem Anwendungsentwickler Dienste zur Verfügung, um z. B. Dienstgüten-Abbildungen und -Reservierungen durchzuführen. Beispiel: *QoS-Broker* (s. 2.4.3.1, S. 28, [NS95]).

- Nachweis der Tauglichkeit eines Programmierkonzepts:
  Um zu zeigen, dass ein neues Konzept bei der Entwicklung eines komplexen Systems Vorteile bringt, wurden oft Multimedia-Anwendungen mit Hilfe dieses Konzepts entwickelt und dann mit Anwendungen verglichen, die auf anderen Konzepten basieren. Beispiele s. 2.4.4, S. 28.

- Unterstützung bei der Ermittlung des Übertragungszustands:
  Diese Frameworks übernehmen für die eigentliche Anwendung die Aufgabe der Überwachung der Dienstgüte der Übertragung und stoßen dann die eigentlichen Anpassungsvorgänge an. Beispiel: *Congestion Manager* (s. 2.4.5.1, S. 29, [BRS99]).

- Untersuchung der Adaption in Multimedia-Anwendungen:
  Diese neueren Ansätze bieten Unterstützung bei der Entwicklung adaptiver Anwendungen. Neben der in diesem Projekt entwickelten EXPERIMENTATION PLATFORM (s. 4.2, S. 64) werden ab S. 30 einige davon vorgestellt.

[10] Einen Überblick über „klassische“ QoS-Architekturen (Stand 1996) bietet [CAH96].

Im Folgenden werden die Anwendungen und Frameworks näher betrachtet.

2.4.2 Einfache Anwendungserstellung

2.4.2.1 CINEMA

Das Ziel dieses Systems war die Bereitstellung einer Sammlung von einfach verwendbaren und universell einsetzbaren Komponenten, die zu komplexen Multimedia-Anwendungen zusammengebaut werden können [BDHea93]. Durch ein „Port“- und „Link“-Konzept können diese Komponenten miteinander verbunden werden. Weitere Ziele waren u. a. die Untersuchung der Synchronisation (s. [RH95]) und Zusammenfassung von Datenströmen.

Abgrenzung:
Durch die Vorgabe einer Struktur der Anwendung vor dem Start ist es nicht möglich, im laufenden Betrieb Komponenten zu ersetzen, also z. B. eine Aufgabe statt von drei Komponenten nur noch von zwei durchführen zu lassen.

2.4.2.2 Da CaPo++

Stiller *et al.* stellen in [SCWea99] die Middleware *Da CaPo++* vor, deren Ziel die Unterstützung der Entwicklung von Multimedia-Anwendungen ist. Aus einer Spezifikation der zu erreichenden Dienstgüte und funktionaler Parameter wie Sicherheitsanforderungen ermittelt das System eine Konfiguration von Modulen und stellt Laufzeitunterstützung bereit. Zusätzlich existiert eine objektorientierte Programmierschnittstelle.

Abgrenzung:
Ziel dieser Middleware war nicht die Unterstützung adaptiver Anwendungen, sondern die einfache Entwicklung und der effiziente Ablauf. Anpassungsmechanismen werden nicht betrachtet.

2.4.3 Dienstgüten-Verarbeitung

2.4.3.1 QoS-Broker

Diese in [NS95] vorgestellte Systemarchitektur hatte auf die Betrachtung von Dienstgüte in verteilten Multimedia-Anwendungen großen Einfluss, weil sie erstmals Abbildungen, Zulassungskontrolle und Aushandlung klar definierte. Es wird ein „Broker“–Konzept vorgeschlagen, also ein Makler, der die Dienstgütenaushandlung zwischen den verschiedenen Ressourcen-Verwaltern für die Anwendung übernimmt.

Abgrenzung:
Das Hauptziel dieser Systemarchitektur war die Aufteilung der verschiedenen Aufgaben bei der Berechnung der benötigten Dienstgüte, wobei der Nutzer selbst die Dienstgüte auf Anwendungsebene festlegen muss. Auf eine Änderung in der Verfügbarkeit der Ressourcen wird mit einer Neuaushandlung reagiert.

2.4.4 Programmierkonzepte

2.4.4.1 Performance-orientierter Entwurf

In [BHHea99] wurde eine Methode zum „performance-orientierten Systementwurf“ vorgestellt und bewertet. Große Anstrengungen wurden unternommen, um die Komponenten des Systems zu

verbessern und so ihre Leistung zu erhöhen, auch indem „schichten- und komponentenübergreifende Optimierungsmöglichkeiten genutzt“ wurden. Ziel war es schließlich, die hier gewonnenen Erkenntnisse zu verallgemeinern, um sie auf andere verteilte Anwendungen anwenden zu können.

Abgrenzung:
Das hier verfolgte Ziel der Bewertung einer Entwurfsmethode ist ein völlig anderes, als die Untersuchung der Adaption in diesem Projekt.

2.4.4.2 Ansatz mit mobilen Agenten

[GOFC98] stellt einen Ansatz vor, der so genannte „mobile Agenten“ einsetzt. Diese können sich gegenseitig bei der Erfüllung ihrer Aufgaben unterstützen und lernen durch die Interaktion mit anderen Agenten, ihren Auftraggebern und den Systemen, in die sie migrieren.

Abgrenzung:
Grundanforderung an die Entwicklung des Systems war hier der Einsatz von mobilen Agenten, um die angedeuteten Vorteile zu erreichen. Es wird jedoch eingeräumt, dass Agenten-basierte Systeme niemals die Leistung traditionell programmierter verteilter Systeme erreichen kann („. . . agent-based systems will never reach the performance of traditional, RPC-based, distributed systems.“, [GOFC98] S. 1277)

2.4.4.3 Ansatz mit Reflexion

Ein interessantes Konzept in der Software-Entwicklung ist die Verwendung von *Reflexion*. Hierbei kann das Verhalten von Objekten nachträglich verändert werden, indem vor und/oder nach Methodenaufrufen andere Funktionen ausgeführt werden. Tassel verwendet in [Tas97] *„Reflective Java“*, um ein Dienstgüten-Management-Modell zu entwickeln, mit dem eine Java-Anwendung mit minimalem Aufwand um eine Dienstgüten-Steuerung erweitert werden kann.

Abgrenzung:
Ein Hauptziel dieses Projekts war, die Programmiersprache „Reflective Java“ zu bewerten. Anpassung erfolgt durch erneute Aushandlung und Ressourcen-Reservierung. Konkrete Maßnahmen werden nicht beschrieben oder untersucht.

2.4.5 Netz-Überwachung

2.4.5.1 Congestion Manager

H. Balakrishnan *et al.* stellen in [BRS99] ein Framework vor, das Anwendungen eine Schnittstelle bietet, über die sie den Zustand des Netzes abfragen können. Zusätzlich übernimmt das Framework die zeitliche Planung der Übertragung, d. h. es legt fest, wann Daten ausgesandt werden können.

Abgrenzung:
Ziel dieses Ansatzes ist es vor allem, die Verträglichkeit von Echtzeit-Multimedia-Übertragungen mit anderen Verkehrstypen (insbesondere TCP) zu ermöglichen. Es wird kaum auf die Art der Anpassung in der Anwendung selbst eingegangen.

2.4.6 Adaptive Anwendungen

2.4.6.1 Odyssey

In [Nob00] stellt B. Noble *Odyssey* vor, eine „Plattform für mobilen, adaptiven Datenzugriff“. Der Ansatz dieses Systems ist die Anpassung der Qualität der Daten an die verfügbaren Ressourcen. Der Begriff „*Fidelity*“ (etwa: Genauigkeit im Vergleich zu einer Referenzdarstellung) als Maß der Datenqualität wird eingeführt. Das System unterstützt dabei die Anwendung bei den Adaptionsentscheidungen, nimmt sie ihr aber nicht ab. Dies wird hier als *application-aware adaptation* bezeichnet. Verschiedene Anwendungen, die dieselben Ressourcen benutzen wollen, kooperieren bei der Vergabe.

In einer Implementation ist die Anwendung ein „Klient“ und teilt *Odyssey* mit, welche Ressourcen sie benötigt. Das System überwacht dann die zugeteilten Ressourcen und alarmiert die Anwendung, wenn sich in der Verfügbarkeit etwas ändert. Die Anwendung kann dann ihre Anforderungen anpassen.

Abgrenzung:
Hier werden keine Hinweise gegeben, wie die Anwendung konkret auf eine Änderungen in der Ressourcenverfügbarkeit reagieren soll. Das System soll sie dabei „nur“ unterstützen.

2.4.6.2 CORBA-Middleware

B. Li und K. Nahrstedt stellen in [LN99] eine Middleware vor, die zwischen Anwendung und Betriebssystem sitzt. Ihre Aufgabe ist es, einerseits Fairness zwischen verschiedenen Anwendungen zu gewährleisten, andererseits aber auch die Anwendung bei der Einhaltung ihrer Dienstgütenanforderungen zu unterstützen. Hierzu wird eine Abbildung der Theorie der Regelungstechnik auf das System vorgenommen, konkret der Ansatz der *Fuzzy Control* (s. 2.3.3, S. 26). In der Middleware wird jede Ressource des Systems von einem „Adaptor“ überwacht, der dann die Anpassung der Anwendung auslöst. Jede Anwendung hat hierzu einen „Configurator“, der die entsprechenden Änderungen an den gewählten Parametern vornimmt.

Bemerkenswert ist hierbei besonders, dass der Versuch unternommen wird, eine Standardmethode der Regelungstechnik (s. 2.3.3, S. 26) zur Berechnung des Anpassungsfaktors zu nutzen, nämlich eine *PID*-Regelung (*proportional-integral-derivative*). Hierzu muss eine Linearisierung im Arbeitspunkt angenommen werden.

Abgrenzung:
Das Projekt ist hochinteressant und relevant im Vergleich zur vorliegenden Arbeit. Es stellt ein System bereit, in das die gewonnenen Erkenntnisse eingebaut werden könnten. Jedoch werden keine konkreten Aussagen über die Eignung und Auswirkungen verschiedener Anpassungsmethoden im Gesamtkontext gemacht, so dass das Ziel ein anderes zu sein scheint.

2.4.6.3 Modell mit mehrdimensionaler räumlicher Darstellung

Ziel des Projekts in [BK99] ist es, adaptiven Anwendungen ein Modell bereit zu stellen, an dem diese den Zustand des Netzes erkennen können. Die Anwendung definiert dabei Arbeitspunkte („Flow-States“), die je nach Netzzustand eingenommen werden.

Die Darstellung des Netzzustands erfolgt dabei mit Hilfe mehrdimensionaler räumlicher Darstellung, wobei die Anwendung einen Teilbereich des Raums definiert, in der sich der Arbeitspunkt der Übertragung befinden soll. Die Anpassung selbst wird der Anwendung jedoch nicht abgenommen,

sondern muss von dieser abhängig von der gelieferten Zustandsinformation durchgeführt werden. In diesem Artikel wird als Beispiel ein Adaptionsalgorithmus für eine eine Audio-Anwendung vorgestellt.

Abgrenzung:
Dieses Framework bietet eine interessante Schnittstelle zur Beobachtung des Netzzustands an. Die Anpassung kann relativ einfach vorgenommen werden, jedoch liegt auch hier der Schwerpunkt nicht in der Untersuchung der möglichen Vorgehensweisen.

2.4.6.4 Multicast-System

Ein *Framework* zur Unterstützung der Entwicklung adaptiver, interaktiver Multicast-Multimedia-Anwendungen stellen A. Youssef *et al.* in [YAWM00] vor. Ziel ist hierbei, eine Menge von Strömen gemeinsam zu verwalten, so dass eine Gesamtdienstgüte (*„Quality of Session“, QoSess*) erreicht wird. Hierzu wird die Ende-zu-Ende–Leistung kontinuierlich von *„monitoring agents“* überwacht und der Zustand in einem Rückkopplungsmechanismus von *„inter-stream adaptation agents“* ausgewertet. Die Anpassung erfolgt dann in der Auswahl der Schichten einer hierarchischen Kodierung, die der Empfänger erhalten will.

Es wird angenommen, dass auf der Vermittlungsschicht die Verteilung der Daten optimiert werden kann, d. h. dass Empfänger, die bestimmte Schichten nicht verwenden können oder wollen, diese auch nicht zugesandt bekommen (hier über das *Internet Group Management Protocol*, IGMP [Dee89]).

Abgrenzung:
Dieses Framework kann nur bei Multicast-Übertragungen effizient eingesetzt werden, was jedoch nicht mit den Zielen dieses Projekts übereinstimmt.

2.4.6.5 Nutzer-beeinflussbare Anpassung

In [OMRW98] stellen Ott *et al.* eine Architektur für adaptive Anwendungen vor, bei denen der Benutzer die Gewichtung der einzelnen Medien verändern kann, indem er sie innerhalb einer dreidimensionalen Landschaft anordnet, also z. B. die Videoübertragung in den Hintergrund rückt.

Die Architektur bietet Abstraktionen für die Ressourcenzuordnung, die durch Dienstgütenaushandlung und -verträge erfolgt. Hierbei wird so genannte „soft-QoS“ (weiche Dienstgüte) verwendet, d. h. es wird nicht erwartet, dass die spezifizierte Dienstgüte garantiert wird.

Abgrenzung:
Schwerpunkt dieser Arbeit ist die Bereitstellung eines Frameworks zur Unterstützung der Anwendungsentwicklung. Besonders sticht dabei die Einbeziehung des Nutzers in die Anpassung der Dienstgüte hervor, indem ihm Gelegenheit gegeben wird, die Gewichtung der Medien zu verändern. Ziel war auch hier nicht die Untersuchung der Möglichkeiten zur Anpassung der Anwendung.

2.4.6.6 QoS-Agenten zur Reservations-Anpassung

Hafid und Bochmann stellen in [HBv98] ein System vor, dass die Reservierung von Ressourcen für eine verteilte Multimedia-Anwendung unterstützt und bei verringerter Verfügbarkeit automatisch für eine Neuaushandlung und Anpassung der Anwendung sorgt.

Dabei kann das System die Konfiguration der Komponenten oder die Anforderungen an die von den Komponenten zu erbringende Dienstgüte ändern.

Abgrenzung:
Diese Arbeit ist ebenfalls hoch interessant, stellt sie doch Protokolle vor, die die Anpassung in den Komponenten veranlassen. Hierbei liegt der Fokus jedoch auf der Ressourcen-Reservierung, d. h. die eigentliche Adaption wird den Komponenten überlassen.

2.4.6.7 Proxy-Server und Umkodierung

[YM99] stellt eine Architektur vor, bei der im Netz so genannte Proxies („Stellvertreter“) Multimedia-Daten umkodieren können. Hierzu handelt der Proxy mit dem Empfänger eine Dienstgüte für eine Video-Übertragung von einem Video-Server aus. Die Video-Daten des Servers werden dann im Proxy dergestalt umkodiert, dass die vereinbarte Dienstgüte erreicht wird. Die Dienstgüte der Übertragung zum Empfänger kann dabei an die Empfangsbedingungen angepasst werden; es findet eine kontinuierliche Abbildung der verfügbaren Ressourcen auf Dienstgüten-Parameter statt.

Abgrenzung:
Die hier vorgestellte Architektur zielt auf eine Anwendung mit mobilen Endgeräten, die auf Video-Daten im Zugangsnetz zugreifen. Da hier Änderungen im Transportnetz notwendig sind, ist dies ein anderer Ansatz als in der vorliegenden Arbeit.

2.4.6.8 System mit Verzögerungskompensation und Glättungsmechanismus

Um das Ausspielen von Medienströmen zu synchronisieren, wird in [TYS99] ein System vorgestellt, das über ein Kommunikationsprotokoll den *Server* so steuert, dass er den Zeitpunkt des Abrufs (engl. *data retrieval time*) der zu übertragenen Daten ändert.

Hierzu wird das System in zwei Ebenen aufgeteilt, die *„Control and Monitoring Plane“* (Steuerung und Überwachung) und die *„QoS Plane“* (Dienstgüte). Jede Ebene besteht aus vier Schichten, wobei in der untersten Schicht (Netz) keine Steuerung stattfindet (vgl. 4.2).

Das System verwendet einen Mechanismus zur Intra-Strom–Synchronisierung durch Änderung des Abfragezeitpunkts (s. 3.5.3.5, S. 60), durch den es zur Auslassung von Medien-Daten (s. 3.5.3.6, S. 61) kommen kann.

Abgrenzung:
Die vorgestellte Architektur zeigt große Parallelen zur EXPERIMENTATION PLATFORM auf, jedoch werden nur zwei Anpassungsmechanismen verwandt, von denen einer (Änderung des Abfragezeitpunkts) für interaktive Anwendung aufgrund der Erhöhung der Verzögerung weniger geeignet erscheint. Die Anpassung erfolgt ohne Betrachtung des Typs des Mediums anhand zweier Messwerte, der Ausspielabweichung (engl. *presentation deviation*) und der Verlustrate (engl. *data loss rate*).

2.4.6.9 Auswahl auf Inhaltsebene

Springer und Schill stellen in [SS00] ein Modell zur Unterstützung von Adaption auf der Anwendungsebene vor. Hierbei werden mit Hilfe von „mobilen Agenten“ abzurufende Daten ausgewählt oder ihre Darstellung (Kodierung) geändert. Ist z. B. das verwendete Endgerät nicht in der Lage,

Sprach-Daten abzuspielen, so werden diese nicht übertragen. Alternativ könnten diese Daten aber von einem Agenten-Programm in Text umgewandelt und dann übertragen und angezeigt werden.

Abgrenzung:
Das hier vorgestellte Modell zielt vor allem auf mobile Endgeräte mit stark eingeschränkter Leistungsfähigkeit. Es soll hier ein Framework geschaffen werden, mit dem die Anpassung auf Anwendungsebene einfach durchführbar ist. Hierbei sind jedoch nicht interaktive Anwendungen das Hauptziel, sondern Abfrage- bzw. Abholdienste wie E-Mail und Datenbanken.

2.4.7 Fazit

Die obigen Abschnitte (und auch [CAH96]) zeigen, dass es bereits eine große Anzahl an Multimedia-Anwendungen und -Frameworks gibt. Es wird jedoch auch deutlich, dass nur wenige davon mit dem Ziel der Untersuchung der allgemeinen Adaption in interaktiven Szenarien entworfen wurden und somit die EXPERIMENTATION PLATFORM als Werkzeug zur Bewertung von Adaptionsalgorithmen gerechtfertigt ist.

3 Anpassung in Multimedia-Anwendungen

3.1 Einführung

Eine adaptive Multimedia-Anwendung ist in der Lage, ihr Verhalten während der Übertragung von Medienströmen zu verändern, um sich an geänderte Bedingungen anzupassen. So kann es vorkommen, dass sich die Verfügbarkeit von Ressourcen in den Endgeräten oder dem Netz während der Übertragung verändert. Es ist möglich, dass sich Eigenschaften der übertragenen Medien oder des Netzes ändern, z. B. kann der Komprimierungsfaktor eines Algorithmus von den Bilddaten abhängen oder es kann sich die Übertragungsverzögerung ändern.

Daher muss in der Anwendung ein „Adaptionsalgorithmus“ ablaufen, der die erbrachte Dienstgüte ständig überwacht und bei Bedarf durch den Einsatz von „Anpassungsmechanismen“ das Verhalten der Anwendung ändert. Dies geschieht mit dem Ziel, die vom Nutzer erfahrene Gesamtdienstgüte (s. 2.1.4, S. 11) zu erhöhen.

Es wird nun zunächst der prinzipielle Ablauf der Adaptionsalgorithmen in einer Multimedia-Anwendung dargestellt und erläutert, wie die Leistung verschiedener Algorithmen bewertet werden kann. Danach werden einige Mechanismen klassifiziert und genauer untersucht.

Vereinfachung des ISO/OSI-Referenzmodells

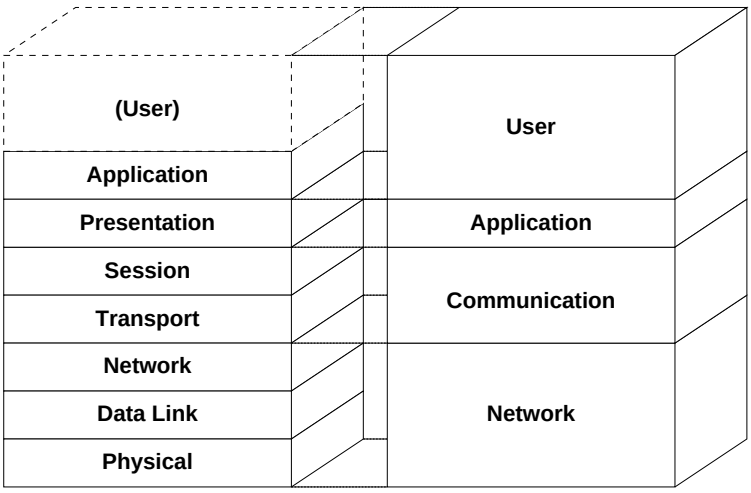


Abbildung 3.1: Zuordnung der OSI-Schichten zum vereinfachten Modell

Für die „vertikale Einordnung“ der Mechanismen wird das OSI-Schichtenmodell vereinfacht (Abb. 3.1). Statt der sieben Schichten werden nur vier betrachtet: Die Schichten 1, 2 und 3 (Bitübertragung – *physical*, Sicherung – *data link* und Vermittlung – *network*) werden zur „Netz-Schicht“ (*network level*) zusammengefasst[1], die Schichten 4 und 5 (Transport und Kommunikationssteuerung – *session*) zur „Kommunikations-Schicht“[2] (*communication level*). Die Darstellungsschicht 6 – *presentation layer* entspricht der „Anwendungs-Schicht“[3] (*application level*) und aus Schicht 7 (Verarbeitung – *application*) und einer zusätzlichen Betrachtung des subjektiven Eindrucks des Benutzers (*user*) von der Gesamtdienstgüte wird die „Nutzer-Schicht“ (*user level*) gebildet.

3.2 Adaptionsalgorithmen

Die folgenden Abschnitte werden zeigen, dass eine adaptive Multimedia-Anwendung viele verschiedene Anpassungsmechanismen zur Auswahl hat. Es stellt sich nun jedoch die Frage, wann welche Mechanismen unter gegebenen Umständen am besten eingesetzt werden. Dies ist die Aufgabe der Adaptionsalgorithmen:

- ❑ Sie beobachten die Dienstgüte der einzelnen Medien-Übertragungen und überwachen die Gesamtdienstgüte der Anwendung,
- ❑ sie entscheiden, wann die Anwendung ihr Verhalten ändern muss,
- ❑ sie wählen die einzusetzenden Mechanismen aus und stellen ihre Parameter ein,
- ❑ und schließlich veranlassen sie die Anwendung, die Änderungen durchzuführen.

Wang und Schulzrinne teilen in [WS99a] die Adaptionsalgorithmen in folgende Kategorien:

- ❑ Sender-gesteuert: Die Entscheidung über die Anpassung wird im Sender vorgenommen. Dies geschieht z. B. über eine Betrachtung der Pufferfüllstände im Sender (s. a. 3.5.2.3, S. 50) oder des von den Empfängern gemeldeten Netzverhaltens.
- ❑ Empfänger-gesteuert: Die Entscheidung über die Anpassung wird vom Empfänger durchgeführt, der das Aussenden der Daten durch den Sender beeinflussen kann. Bietet der Sender die Daten z. B. in mehreren Lagen an (s. 2.4.6.4, S. 31, 3.5.2.6, S. 54), so kann der Empfänger die Auswahl ändern.
- ❑ *Transcoder*–Ansatz: Hierbei ändern spezielle Netzknoten *im Kommunikationsnetz* die Kodierung der Daten (vgl. *Active Networks* [TSSea97]). Dies wird vor allem in Gateways zwischen Netzen verschiedener Technologien eingesetzt, um Netze mit geringerer Bandbreite nicht zu überfordern. (Dieser Ansatz wird hier nicht betrachtet.)

[1] IP-Kommunikation ist somit komplett in dieser Schicht angesiedelt.

[2] Hier liegen dann UDP und RTP.

[3] Da hier die *Anwendung* die Übertragung von Medienströmen ist, wäre hier „Medien-Schicht“ eine etwas genauere, aber einschränkende Bezeichnung.

3.2.1 Ermittlung des Übertragungszustands

Durch eine Änderung der Verfügbarkeit von Ressourcen im Endgerät oder des Verhaltens des Kommunikationsnetzes (d. h. es gibt hier keine garantierte Dienstgüte) ändert sich für den Nutzer auch die von der Anwendung erbrachte Dienstgüte. Um eine Anpassung der Anwendungsparameter veranlassen zu können, muss die Anwendung also den momentanen Zustand kennen und bewerten. Der erste Schritt hierzu ist die Ermittlung des Zustands.

Zustand der Endgeräte

Änderungen in den Endgeräten können von parallel ablaufenden Prozessen verursacht werden. Unter den oben angeführten Voraussetzungen (nur Betrachtung der Rechenleistung, s. 2.2.2, S. 14) muss also nur der Fall betrachtet werden, dass den Prozessen der Anwendung weniger Rechenleistung zur Verfügung steht. Dies wirkt sich auf die Zeitdauer aus, die für die Bearbeitung der Multimedia-Daten benötigt wird und kann im Endgerät durch die Betrachtung der Zeitstempel an den Daten beobachtet werden.

Beispielsweise wird im Sender-Endgerät die Dauer für die Komprimierung eines Video-Bildes gemessen. Nimmt die Dauer zu, so ist dies ein Hinweis auf eine Abnahme an verfügbarer Rechenleistung und kann zu einer Adaptionsentscheidung führen.

Zustand des Kommunikationsnetzes

Im Empfänger kann die Auswirkung des Netzes auf den Datenverkehr durch Messung ermittelt werden. Hierzu werden in den betrachteten Schichten (s. 3.1, S. 35) Messwerte erhoben:

- Netz-Schicht: Viele Geräte in einem Netz können Informationen über den von ihnen transportierten Verkehr sammeln. Dies können z. B. die Länge der Pufferwarteschlangen, die Zahl der verworfenen Pakete, die Zahl der fehlerhaften Pakete, die benutzte und die freie Bitrate sein.
  Anmerkung: In dieser Untersuchung wird jedoch davon ausgegangen, dass diese Art von Information *nicht* zur Verfügung steht, da die Anwendung nicht mit dem Netz kommunizieren kann. Dies ist insbesondere beim momentan am häufigsten eingesetzten Kommunikationsprotokoll *IP* der Fall. Somit muss der Zustand über Mechanismen auf anderen Schichten ermittelt werden.

- Kommunikations-Schicht: Diese Schicht ist die erste Schicht der Anwendung, die die Auswirkungen des Kommunikationsnetzes auf den Verkehr beobachten kann. Ausgehend von der Annahme, dass in der Kommunikations-Schicht des Senders die Mediendaten optimal „verpackt“ werden und *medienunabhängige* Zusatzinformation (Sequenznummer, Zeitstempel) zur Verfügung steht (vgl. RTP, s. 2.2.3.3, S. 16), können hier bereits Aussagen über folgende Effekte gemacht werden:

  - Änderung der Paketreihenfolge: Jedes Paket besitzt eine fortlaufende Sequenznummer, durch die eine Änderung der Reihenfolge bei der Paketankunft erkannt werden kann.
  - Paketverluste[4]: Ebenfalls über die Sequenznummer kann der Verlust eines Pakets erkannt werden.

[4] Übertragungsfehler und Verluste werden aufgrund des Verwerfens verfälschter UDP-Pakete gemeinsam betrachtet (STD 3/RFC 1122/4.1.3.4).

- Übertragungsverzögerung und -schwankung: Die Zeitdauer zwischen Erzeugung von Daten und ihrem Eintreffen beim Empfänger kann über den Zeitstempel ermittelt werden; ebenso deren Schwankung.
- Bitrate („Bandbreite“): Die Menge der übertragenen Daten pro Sekunde kann abgeschätzt werden.

Anmerkung: Obwohl diese Messungen in der Kommunikations-Schicht durchgeführt werden, beschreiben sie das Verhalten der Netz-Schicht und die ermittelten Werte werden daher Parametern der Netz-Schicht (*NLQ*) zugewiesen.

Zustand der Medien-Übertragungen

Sowohl die Verfügbarkeit der Ressourcen in den Endgeräten, als auch der Zustand des Kommunikationsnetzes beeinflussen die Dienstgüte der Medienströme und damit auch die Gesamtdienstgüte der Anwendung.

- ❑ Anwendungs-Schicht: In dieser Schicht können medienspezifische Zustandsparameter ermittelt werden. Für Video-Übertragungen ist dies z. B. die Bildrate, die durch Effekte wie Verzögerungsschwankungen und verlorene Rahmen beeinflusst wird. Hier werden ebenfalls Parameter wie die Bildgröße und -kodierung betrachtet, obwohl sie nicht durch die Übertragung verändert werden. Jedoch kann so in der Anwendungs-Schicht die Medienqualität bewertet werden, indem die Parameter in eine Bewertungsfunktion einfließen, die für den Medienstrom eine Abschätzung der Dienstgüte liefert (s. 4.3.1, S. 71 für Audio).

- ❑ Nutzer-Schicht: Für den Nutzer der Anwendung wird die Dienstgüte durch die Kombination der einzelnen Medien bestimmt. Je nach Szenario wird aus den Qualitätswerten der verschiedenen Medienströme die Gesamtdienstgüte gebildet, indem die gewichtete Summe der einzelnen Werte berechnet wird.

3.2.2 Adaptions-Entscheidung

Der Adaptionsalgorithmus überwacht die erbrachte Dienstgüte der Anwendung und entscheidet, ob eine Anpassung vorzunehmen ist. Parameter, die diese Entscheidung beeinflussen, sind:

- ❑ Prüf-Intervall: Wie lange sammelt der Algorithmus Daten über den Zustand der Anwendung, bevor er über die Anpassung entscheidet?
  Dieser Parameter beeinflusst die *Agilität* der Anwendung, d. h. wie schnell sie auf Änderungen reagiert. Er beeinflusst aber auch die *Stabilität*, d. h. wie häufig die Dienstgüte sich ändert (dies ist meist störend für den Nutzer).

- ❑ Prüf-Kriterien: Je nach Szenario gibt es verschiedene *Regeln*, nach denen über die Notwendigkeit einer Anpassung entschieden wird.
  So kann allgemein ab einer bestimmten Differenz zwischen der Ziel- und der erreichten Dienstgüte eingegriffen werden; es ist aber auch möglich, bestimmte Zustandsparameter eines Medienstroms zu überwachen und bei einer plötzlichen Verschlechterung spezifisch zu reagieren (z. B. bei Abnahme der Bildrate Steuerparameter der Videoübertragung zu ändern).

3.2.3 Auswahl der Anpassung

Stellt der Algorithmus fest, dass eine Anpassung notwendig ist, muss im nächsten Schritt entschieden werden, *welche* Anpassung vorzunehmen ist. Ausgehend vom Zustand der Übertragung muss also entschieden werden, welcher der unter 3.3, S. 40 dargestellten Mechanismen auszuwählen ist. Die Entscheidung erfolgt hierbei in einer „Wenn–dann“ – Form, d. h. es wird mit *Regeln* gearbeitet.

Regeln können dabei in verschiedener Form formuliert sein. Ansätze wie [LN99] verwenden eine allgemeine Regeldarstellung und trennen dabei den Programm-Code von einer Regeldatenbank. So wird dort das „Fuzzy Control Model“ (s. 2.3.3, S. 26) verwendet, bei dem die Regeln in Form einer Matrix die Zustands- mit den Steuerparametern verknüpfen. Möglich sind jedoch auch andere Formen der Implementierung, z. B. angelehnt an Methoden der Künstlichen Wissensverarbeitung (Expertensysteme, Neuronale Netze, ...) und Optimierungsverfahren (kalkülbasierend, evolutionäre Verfahren, ...). Ziel dieser Arbeit ist es insbesondere, solche Regeln zu untersuchen und zu bewerten.

3.2.4 Durchführung der Adaption

Der Adaptionsalgorithmus muss die Anpassung der Anwendung steuern. Dieser Vorgang hängt von der Art der Implementierung ab. So kann der Algorithmus in einem objektorientierten System Methoden der an der Übertragung beteiligten Module aufrufen, um Parameter zu übergeben oder Änderungen zu veranlassen. Es ist zudem möglich, die Verknüpfung der Module miteinander zu verändern, d. h. Module auszutauschen oder neue einzufügen.

Eine weitere Methode ist die Kommunikation über Steuerparameter. Dabei greifen die Module der Anwendung auf Mengen von Parametern zu („Tupelraum“, jeder Parameter besteht aus Bezeichner und Wert), die ihr Verhalten beeinflussen, z. B. eine Zielvorgabe für die Verzögerung in der Form (`"Delay"`, `0.04`). Durch diese Entkopplung muss der Adaptionsalgorithmus nicht über genauere Information über die Implementierung der Anwendung verfügen, sondern kann anonym mit dem Modulen des Systems kommunizieren. Gleichzeitig kann über einen analogen Mechanismus auch der Zustand der Übertragung dem Adaptionsalgorithmus mitgeteilt werden.

3.2.5 Kriterien der Leistungsbewertung

Um die Leistung verschiedener Adaptionsalgorithmen vergleichen zu können, müssen zunächst Kriterien zu ihrer Bewertung festgelegt werden. Diese sind:

- Durchschnittliche Dienstgüte: Wie verläuft die Dienstgüte der Anwendung für ein bestimmtes Übertragungsszenario?
- Agilität: Wie schnell reagiert die Anwendung auf Änderungen in den verfügbaren Ressourcen?
- Stabilität: Hält die Anwendung die Dienstgüte über einen langen Zeitraum konstant oder ändert sie die Dienstgüte sehr häufig?

Offensichtlich sind Stabilität und Agilität gegensätzliche Ziele. Daher hängt die Bewertung letztlich vom betrachteten Szenario und der Toleranz des Nutzers gegenüber Veränderungen ab.

3.2.6 Durchführung der Leistungsbewertung

Um die verschiedenen Implementationen von Adaptionsalgorithmen bewerten zu können, wird ihre Leistung hinsichtlich vorgegebener Übertragungsszenarien beobachtet. Hierzu wird in einem experimentellen Aufbau die Anwendung gestartet und dann das Verhalten des Transportnetzes mittels eines WAN-Emulators (z. B. „The Cloud“ [Clo], „NIST Net“ [NIS01]) variiert.

Hierbei werden zunächst einfache Zusammenhänge untersucht, d. h. es wird nur ein Parameter der Übertragung verändert:

- Schwankung der Übertragungsverzögerung: Um die Reaktion der Anwendung auf einer Änderung der Verzögerung beurteilen zu können, wechselt die Verzögerung zwischen zwei Werten, einem hohen und einem niedrigen, hin und her. Es wird die Reaktion auf verschiedene Wechselfrequenzen untersucht.
- Änderung der Übertragungsbitrate: Die emulierte Rate wird für einige Sekunden abgesenkt, um den Effekt einer geänderten Ressourcen-Aufteilung im Netz nachzubilden. Hierdurch können Puffer volllaufen, was zu Verlusten führen kann. Die anschließende Erhöhung der Bitrate führt zu Paketbüscheln (engl. *bursts*), die das Ausspielverhalten beeinflussen.
- Schwankung der Verlustwahrscheinlichkeit: Es werden Phasen mit erhöhter Verlustwahrscheinlichkeit emuliert, wobei deren Häufigkeit und Länge in die Bewertung einfließen.

Die Experimente erlauben eine Bewertung der Mechanismen hinsichtlich einzelner Effekte und dienen somit der Abschätzung des Zusammenspiels im Rahmen eines Adaptionsalgorithmus.

Der nächste Schritt ist sodann die Bildung spezialisierter Algorithmen für ein zu untersuchendes Szenario (d. h. eine Kombination von Effekten, die während der Übertragung auftreten), die den Zuständen der Übertragung eine Reaktion (einen Mechanismus) zuordnen. Die Leistung der Algorithmen wird sodann im Experiment bewertet und verglichen, mit dem Ziel eine Begründung für unterschiedliches Verhalten zu finden und eine Empfehlung für den Einsatz in bestimmten Szenarien zu geben.

3.3 Schichtenbezogene Funktionsweise der Mechanismen

In diesem Abschnitt wird die Funktionsweise der Mechanismen beschrieben, sortiert nach den Schichten des vereinfachten Modells (s. 3.1, S. 35) in denen die Anpassung stattfindet. Außerdem werden Parameter definiert, die in der jeweiligen Schicht für die Beschreibung des Übertragungs- und Anwendungszustands verwendet werden können. Diese Parameter werden dann in 3.5 verwendet, um konkrete Mechanismen zu beschreiben.

Grundsätzlich gilt für die Parameter: Bei Verwendung des Index „T“ handelt es sich um einen *Zielwert* (engl. *Target-*), der erreicht werden soll. Parameter, die einen *erreichten Wert* (engl. *Achieved-*) repräsentieren, besitzen den Index „A“. Die von einem Mechanismus veränderten Werte werden mit einem „$*$“ markiert.

3.3.1 Nutzer-Schicht

Eine Anpassung in der Nutzer-Schicht hat eine Änderung der Gewichtung der einzelnen Medien zur Folge. Diese Änderung kann auf folgende Art ausgelöst werden:

- Durch den Nutzer: Der Nutzer *ändert das Szenario*, indem er die Gewichtung eines oder mehrerer Medienströme verändert. Die drastischte Möglichkeit ist hierbei das Ab- oder Hinzuschalten eines Medienstroms, z. B. das Beenden der Video-Übertragung in einer Konferenzanwendung. Dies hat eine Änderung der Aufteilung der verfügbaren Ressourcen auf die Medienströme zur Folge, d. h. die Dienstgüten-Anforderung der Anwendungs-Schicht muss neu berechnet werden.
  Für die Durchführung einer solchen Änderung kann die Anwendung dem Nutzer verschiedene Benutzungsschnittstellen anbieten, z. B. kommandozeilen-orientierte (EXPERIMENTATION PLATFORM), grafische (s. 2.4.6.5, S. 31) oder auch einen „Qualitäts-Knopf“ [BRS00]. Für die genaue Untersuchung dieser Art der Anpassung muss das Verhalten bestimmter Nutzergruppen und ihre Toleranz gegenüber Dienstgütenveränderungen betrachtet werden. Dies übersteigt den Umfang dieser Arbeit und wird daher nicht betrachtet.

- Durch eine *Meta-Policy* (s. 3.5.1.1, S. 46): Hierbei ändert die Anwendung die Gewichtung der einzelnen Medien. Die Zieldienstgüte der Medienströme wird hierbei ebenfalls verändert, jedoch nur solange es notwendig erscheint, um die Gesamtdienstgüte zu verbessern. Zum Beispiel kann dies bedeuten, dass zeitweise nur noch Einzelbilder übertragen werden, um Ressourcen für die Audio-Übertragung zu verwenden.
  Eine weitere Möglichkeit ist die automatische Auswahl der Inhalte (s. 3.5.1.2, S. 47). Werden die Daten in mehreren Formaten angeboten, so kann die Anwendung entscheiden, welches davon die beste Dienstgüte liefern wird. Beispielsweise kann statt einer Sprachübertragung die Übertragung eines Textes erfolgen oder statt eines Videofilms können Einzelbilder angezeigt werden.

Parameter

Der Nutzer der Anwendung soll zum einen in der Lage sein, seinen Wunsch an die Dienstgüte Q der Anwendung insgesamt, aber auch an die Dienstgüte $Q(m)$ einzelner Medien m, angeben zu können. Zum anderen muss er die von der Anwendung berechnete momentane Dienstgüte verstehen können. Daher wird auf der Nutzer-Schicht die Qualität durch die Verwendung eines Qualitätsfaktors zwischen 0 (Medium nicht vorhanden) und 10 (Medium in maximaler Qualität) spezifiziert. Die Gewichtung der Medien zueinander folgt aus dem Verhältnis ihrer Gewichtsfaktoren w_m und hängt vom betrachteten Szenario ab (Tab. 3.1).

Symbol	Bereich	Beschreibung
M		Menge der Medien der Anwendung
$Q(m)$	$[0,10]$	Dienstgüte des Mediums $m \in M$
w_m	$[0,1]$	Gewichtsanteil des Mediums $m \in M$ an der Gesamtdienstgüte
Q	$[0,10]$	Gesamtdienstgüte der Anwendung

Tabelle 3.1: Parameter der Nutzer-Schicht

Die erreichte Gesamtdienstgüte Q_A wird aus den Qualitätsfaktoren der beteiligten Medien über eine Gewichtsfunktion gebildet:

$$Q_A = \sum_{m \in M} w_m \cdot Q_A(m) \quad \text{mit} \quad \sum_{m \in M} w_m = 1 \tag{3.1}$$

3.3.2 Anwendungs-Schicht

In dieser Schicht wird die digitale Darstellung der Medienströme verändert:

- Durch Änderung der Behandlung des Datenstroms: Hierbei wird die Art der Daten selbst nicht verändert (Beispiele s. 3.5.2.1, 3.5.2.2 S. 48 ff.).
- Durch Änderung der Kodierungsparameter: Dies hat eine Änderung der übertragenen Daten zur Folge, wobei eine Änderung der Dekodierung nicht immer erforderlich ist. Beispiele: Änderung der Komprimierungsfaktoren, Abtastauflösung und -rate (s. 3.5.2.3, 3.5.2.4, S. 50 ff.).
- Redundanz in der Anwendungs-Schicht: Hierbei werden die Daten mehrfach übertragen, meist in verschiedenen Kodierungen (s. 3.5.2.5, S. 52) und zeitlich versetzt (im Unterschied zu hierarchischen Kodierungen, s. 3.5.2.6, S. 54).
- Durch Änderung des Kodierverfahrens: Dies ändert die Darstellung der Daten grundlegend; es muss ein Austausch von Programmcode erfolgen. Zum Beispiel MPEG statt Motion-JPEG, GSM statt G.711.

Parameter

In der Anwendungs-Schicht befinden sich die Kodierungsverfahren, die die Medien erst übertragbar machen. Dies bedeutet, dass viele Parameter vom konkret verwendeten Verfahren abhängen. Dennoch lassen sich gemeinsame Parameter nach Tab. 3.2 bestimmen.

Symbol	Einheit	Beschreibung
K	–	verwendetes Kodierverfahren/-format
$l_F,\ l_F(t)$	bit	Größe eines Rahmens (konstant oder zeitabhängig)
$C_{cod,K}(l_F)$	%	benötigter Anteil an der Rechenleistung für Kodierung
d_F	s	zeitlicher Abstand zwischen aufeinander folgenden Rahmen
$l_{cod,K}(l_F)$	bit	Größe eines Rahmens nach der Kodierung mit K
p_{drop}	1	Wahrscheinlichkeit eines Verwurfs aufgrund des Überschreitens der *erlaubten Verzögerung*
t_{gen}	s	Zeitpunkt der Erzeugung eines Anwendungs-Datenwertes
t_{out}	s	Ausspielzeitpunkt eines Anwendungs-Datenwertes
$\hat{d} = t_{out} - t_{gen}$	s	Ende-zu-Ende–Verzögerung eines Datenwertes
$\bar{d}$	s	mittlere Ende-zu-Ende–Verzögerung des Mediums

Tabelle 3.2: Parameter der Anwendungs-Schicht

3.3.3 Kommunikations-Schicht

Mechanismen dieser Schicht ändern die Abbildung der Rahmen eines Medienstroms auf einen Paket-Strom (bzw. umgekehrt) oder die Art, wie die Pakete an das Netz übergeben werden:

- ❑ Verkehrsformung (engl. *shaping*): Änderung der Übergabe der Pakete an das Netz, bzw. der Behandlung empfangener Pakete. Zum Beispiel Anpassung der Paketgröße, des Paket-Scheduling (s. 3.5.3.1, S. 55) oder der Paket-Pufferung.

- ❑ Verlustlose Komprimierung: Die Medien-Daten werden für die Übertragung komprimiert, so dass die implizite Redundanz[5] vermindert wird (s. 3.5.3.2, S. 56).

- ❑ Möglichkeit der wiederholten Übertragung: Übertragungsverluste werden durch eine Neuanforderung der verlorenen Daten ausgeglichen (s. 3.5.3.3, S. 57), wobei jedoch auf das rechtzeitige Eintreffen der Daten beim Empfänger geachtet werden muss – dies schließt Mechanismen wie TCP (s. 2.1.2.1, S. 7) aus.

- ❑ Redundanz in der Kommunikations-Schicht: Einfügen von Redundanz, unabhängig von der Art des Medienstroms (Vorwärts-Fehlerkorrektur, *Forward Error Correction*, s. 3.5.3.4, S. 59).

Parameter

Auf dieser Schicht ist der Inhalt der transportierten Daten bereits nicht mehr relevant, sondern der Verkehr wird alleine durch sein Muster von den Parametern nach Tab. 3.3 beschrieben.

Symbol	Einheit	Beschreibung
l_P, $L(n)$, $L(t)$	*bit*	Paketlänge (konstant o. abh. von Paketnummer bzw. Zeit)
$d(l)$	s	Verzögerung für Daten der Länge l
B_{max}	$\frac{bit}{s}$	Maximal verfügbare Bitrate
p_{Floss}	1	Rahmenverlustwahrscheinlichkeit

Tabelle 3.3: Parameter der Kommunikations-Schicht

3.3.4 Netz-Schicht

Die Anwendung kann auf Fähigkeiten der Netz-Schicht in verschiedener Weise zugreifen:

- ❑ Nutzung von Netzdiensten durch Aushandlung: Die Anwendung beeinflusst das Verhalten des Netzes, indem sie einen Verkehrsvertrag aushandelt oder ändert (z. B. *IntServ*, RFC 1633). Sie kann auch Netzknoten umkonfigurieren, z. B. um eine bevorzugte Behandlung des Verkehrs zu erreichen. Sie kann Dienste im Netz nutzen, z. B. *Transcoder*, die die Kodierung eines Medienstroms ändern.

- ❑ Implizite Nutzung von Netzdiensten: Manche Kommunikationsnetze können Verkehrsströme differenziert behandeln. So gibt es für IP-Netze die Technik des IP-Multicast, mit deren Hilfe die Übertragung an mehrere Empfänger effizienter durchgeführt wird. Daneben gibt es Technologien wie *DiffServ* (RFC 2475), die eine Priorisierung von Datenströmen erlauben.

[5] Normalerweise sollte das Kodierverfahren Redundanz aus dem Medienstrom entfernen, jedoch ist dies meist nicht mit der hohen Effizienz spezialisierter Kompressionsalgorithmen der Fall.

Parameter

Die Parameter in Tab. 3.4 beschreiben, wie sich das Netz der Anwendung darstellt, wobei in diesem Projekt die Anwendung das Netzverhalten nicht direkt beeinflussen kann, d. h. sie kann keine Garantien aushandeln.

Symbol	Einheit	Beschreibung
B	$\frac{\text{bit}}{\text{s}}$	Bitrate (Mittel-, Maximalwert, ...)
$D(l)$	s	Verzögerung[6] eines Pakets der Länge l
ΔD	s	Schwankung der Verzögerung
p_{Ploss}	1	Paketverlustwahrscheinlichkeit

Tabelle 3.4: Parameter der Netz-Schicht

3.3.5 Schichtenübergreifende Mechanismen

Ein Beispiel für einen Mechanismus, für dessen sinnvollen Einsatz Eingriffe auf mehreren Schichten erforderlich sind, ist die hierarchische Kodierung für Multicast-Anwendungen (s. 3.5.2.6, S. 54). Hierbei wird in der Anwendungs-Schicht eine Kodierung eingesetzt, die die Medien-Daten in Lagen unterschiedlicher Gewichtung produziert. Der Mechanismus steuert dabei, abhängig vom Zustand des Netzes, die vom Empfänger angeforderten Kodierungslagen. Dies führt jedoch erst in Kombination mit der IP-Multicast–Technik auf der Netz-Schicht zu einer höheren Übertragungseffizienz, da nun nicht mehr alle Daten an alle Empfänger übertragen werden.

Parameter

Tab. 3.5 zeigt den Parameter „Anteil an der Rechenleistung“, der keiner der Schichten zugeordnet werden kann, da diese Ressource in mehreren Schichten verwandt wird.

Symbol	Einheit	Beschreibung
C	%	Anteil an der Rechenleistung

Tabelle 3.5: Restliche Parameter

Die Rechenleistung für eine bestimmte Aufgabe wird durch ihren Anteil an der gesamten Rechenleistung des Endgeräts von 100 % angegeben. Stehen mehr Ressourcen zur Verfügung, z. B. spezialisierte Hilfsprozessoren, so reduziert sich dieser benötigte Anteil C für die Aufgabe, z. B. das Kodieren eines Bildes, und die Dauer nimmt ab.

Beispiel

Die Kodierung eines Bildes benötige in einem Endgerät $C = 10$ %, somit können maximal $100\ \%/C = 10$ Bilder pro Sekunde kodiert werden. Bei einer verfügbaren Rechenleistung von $C_{avail} = 50$ % können nur 5 Bilder pro Sekunde kodiert werden.

[6] Im Gegensatz zu $d(l)$ in der Kommunikations-Schicht ist hier nur die vom Netz benötigte Dauer zur Übertragung von Paketen enthalten, nicht jedoch der Zeitaufwand für Pufferung oder Komprimierung im Endgerät.

3.4 Klassifizierung der Mechanismen

Um Mechanismen einzuordnen, werden folgende Kriterien betrachtet:

1. Ist der Mechanismus abhängig vom Typ oder Format des Mediums?
2. Auf welcher Schicht im (vereinfachten) OSI-Modell ist der Mechanismus angesiedelt?
3. Wie beeinflusst der Mechanismus die Dienstgüte des zu übertragenen Medienstroms?
4. Wie beeinflusst der Mechanismus die Art des übertragenen Verkehrs?

3.4.1 Kriterium: Medienabhängigkeit

Nutzt ein Mechanismus spezifische Eigenschaften eines Mediums oder einer Kodierung aus, so bezeichnet man ihn als *medienabhängig*. Dies hat zur Folge, dass er nicht für alle Medien oder alle Kodierungen einsetzbar ist.
Andere Mechanismen können für alle Medien und Kodierungen eingesetzt werden, z. B. weil sie auf der Kommunikations-Schicht arbeiten und somit unabhängig vom transportierten Medium sind.

3.4.2 Kriterium: Schicht

In welcher Schicht des vereinfachten Modells (s. 3.1, S. 35) arbeitet der Mechanismus, oder greift er über mehrere Schichten hinweg? Eine Einordnung in eine bestimmte Schicht kann eine Einschränkung der Einsetzbarkeit des Mechanismus zur Folge haben, z. B. weil keine Möglichkeit besteht, das Verhalten der Netz-Schicht zu ändern, oder weil Vorgaben der Nutzer-Schicht unbedingt erfüllt werden müssen.

3.4.3 Kriterium: Dienstgüte

Dienstgüte-ändernd: Mechanismen dieses Typs verändern die Qualität der Daten, die übertragen werden. Die neue Dienstgüte begrenzt die maximal mögliche Qualität des Mediums.

Dienstgüte-erhaltend: Diese Mechanismen ändern die Dienstgüte des übertragenen Mediums nicht, sondern nur die Art, wie es übertragen wird.

3.4.4 Kriterium: übertragener Verkehr

Der Einsatz eines Mechanismus kann das Verkehrsprofil des übertragenen Verkehrs verändern. Die Übertragung wirkt sich dann anders aus, so dass der empfangene Verkehr sich ebenfalls ändert.

Änderung des Bitratenbedarfs: Der Einsatz des Mechanismus führt zu einer Änderung der Datenmenge, die pro Sekunde vom Netz übertragen werden muss.

Änderung des Verzögerung: Eine oder mehrere Teil-Verzögerungen werden verändert, so dass sich ebenfalls die resultierende Gesamt-Verzögerung ändert.

Änderung der Verlusttoleranz: Der resultierende Datenstrom ändert seine Empfindlichkeit gegenüber Datenverlusten.

3.5 Mechanismen

An dieser Stelle werden einige typische Anpassungsmechanismen vorgestellt und exemplarisch klassifiziert. Zunächst wird ihre Funktion erläutert, sodann ihre Auswirkungen auf die Übertragung. Für die Beschreibung wird die Nomenklatur nach 3.3 verwendet. Tabelle 3.4, S. 62 enthält eine Übersicht der Eigenschaften der Mechanismen.

3.5.1 Mechanismen der Nutzer-Schicht

Mechanismen der Nutzer-Schicht greifen in die relative Gewichtung der Medienströme zueinander ein und müssen die Erwartungen des Nutzers an die Leistung der Anwendung in die Anpassung mit einbeziehen.

3.5.1.1 Meta-Policy

Ablauf

Die Bedeutung verschiedener Medienströme variiert je nach Anwendungsszenario. Entsteht nun eine Situation, in der die volle erwartete Dienstgüte nicht mehr erreicht wird, so kann versucht werden, Ressourcen von weniger wichtigen Strömen auf wichtigere zu verlagern.

Hierbei muss jedoch das Szenario berücksichtigt werden, d. h. es muss sichergestellt werden, dass die Änderungen die Gesamtdienstgüte der Anwendung verbessern. Eine „Meta-Policy“ verfügt dabei über die Zustandsinformation aller Medienströme und kann anhand des Szenarios entscheiden, wie die Gewichtung der Medien zu ändern ist. Hiervon ausgehend werden die Dienstgüten-Ziele aller Medien neu berechnet.

Anmerkung

Erreichen die Medienströme eine Dienstgüte, die höher ist als die Zielvorgabe, so kann dies eine Verwendung von zu vielen Ressourcen bedeuten. Soll dies vermieden werden, so können die *Kosten* wie eine Medien-Dienstgüte betrachtet werden, d. h. hohe Kosten reduzieren ebenfalls die Gesamtdienstgüte und nur bei $Q_A < Q_T$ muss durch die Meta-Policy eingegriffen werden.

Parameter

Gewichte der Medien	$W = \{w_m \mid m \in M\}$
erreichte Dienstgüte der Medien	$Q_A(M) = \{Q_A(m) \mid m \in M\}$
Zieldienstgüten der Medien	$Q_T(M) = \{Q_T(m) \mid m \in M\}$

Bedingung für Einsatz

Dieser Mechanismus wird eingesetzt, falls individuelle Anpassungen der Medienströme nicht mehr ausreichen, um die Gesamtdienstgüte zu verbessern. Mögliche Reaktionen sind:

- Das Gewicht des Mediums, dessen Gewicht am geringsten ist, wird reduziert; den wichtigeren Medien stehen dann mehr Ressourcen zur Verfügung.
- Die Zielvorgaben aller Medien werden etwas reduziert, wodurch die Reduktion der Dienstgüte gleichmäßiger stattfindet.

- ❑ Die Zielvorgaben aller Medien werden reduziert, bis auf dasjenige Medium, dessen erreichte Dienstgüte am weitesten unterhalb seiner Zieldienstgüte liegt; es werden diesem Medium also Ressourcen zugeteilt.
- ❑ Das Medium, dessen erreichte Dienstgüte am weitesten unterhalb der Zieldienstgüte liegt, wird aufgegeben und seine Ressourcen an die übrigen verteilt.
- ❑ Weitere Reaktionen abhängig vom Szenario und den Medientypen.

Auswirkung

Die Anwendung verwendet neue Zieldienstgüten-Parameter für die Medienströme.

$$Q_T^*(M) \quad = \quad f(W, Q_T(M), Q_A(M))$$

3.5.1.2 Auswahl auf Inhaltsebene

Quelle: [SS00]

Ablauf

In manchen Szenarien besteht die Möglichkeit, zwischen verschiedenen Formen von Mediendarstellungen zu wählen. So können Informationen in einer Datenbank sowohl als Text wie auch in gesprochener Form vorliegen. Statt einer Videoübertragung reicht mitunter auch eine regelmäßige Übertragung von Einzelbildern. Oder ein Medienstrom kann im Sender inhaltlich umkodiert werden, z. B. indem ein Programm Sprache in Text umwandelt und nur diesen übermittelt.

Parameter

Gewichtung (bzw. Auswahl) der Medien $\quad W = \{w_m \mid m \in M\}$

Bedingung für Einsatz

Um diese Art der Anpassung einsetzen zu können, bedarf es spezieller Szenarien. Der Mechanismus muss genau einschätzen können, welche Medien-Ersetzungen der Nutzer zu akzeptieren bereit ist.

Auswirkung

Dieser Mechanismus ändert die Menge der verwandten Medien, d. h. ein übertragenes Medium m_1 wird durch ein bisher nicht übertragenes Medium m_2 ersetzt, indem die Zieldienstgüte des ersten Mediums auf Null und die des zweiten auf einen Wert größer Null gesetzt wird.

$$\begin{aligned} \exists\, m_1, m_2 \in M \quad : \quad & Q_T(m_1) > 0 \;\wedge\; Q_T^*(m_1) = 0 \;\wedge \\ & Q_T(m_2) = 0 \;\wedge\; Q_T^*(m_2) > 0 \end{aligned}$$

3.5.2 Mechanismen der Anwendungs-Schicht

Hier werden auf bestimmte Medientypen oder gar -kodierungen spezialisierte Mechanismen vorgestellt und diskutiert.

3.5.2.1 Synchronisierung von Datenströmen

Quelle: [RH95]

Ablauf

Bei der Übertragung mehrerer kontinuierlicher Datenströme muss für eine Synchronisierung der Ausspielung gesorgt werden, d. h. der Versatz (engl. *skew*) zwischen den Strömen darf nicht zu groß werden. In diesem Artikel wird nun ein Protokoll vorgestellt, mit dessen Hilfe Datenströme synchronisiert werden können, selbst wenn sie an verschiedenen Terminals ausgespielt werden.

Hierbei wird für einen Datenstrom ein Mechanismus eingesetzt, der den Füllstand des Empfangspuffers überwacht. Über- bzw. unterschreitet dieser eine obere Grenze (d_{UTB}, *upper target boundary*) bzw. untere Grenze (d_{LTB}, *lower target boundary*)[7], so setzt die Anpassung ein: Für eine bestimmte Zeitdauer L wird die Ausspielrate erhöht (reduziert). Dies hat zur Folge, dass sich der Puffer leert (füllt), da nun die Füllrate unter (über) der Ausspielrate liegt. Der Mechanismus hält also den Füllstand in einem bestimmten Bereich, so dass Ausspielpausen durch Pufferunterlauf vermieden werden. Ebenso wird die Verzögerung der Daten begrenzt, d. h. es wird verhindert, dass die Pufferdurchlaufzeit die Ende-zu-Ende–Verzögerung zu stark erhöht.

Die Art der Daten begrenzt die mögliche Änderung der Ausspielrate, weil sonst die Qualität des Signals beeinträchtigt wird. So bedeutet beispielsweise die Erhöhung der Ausspielfrequenz eines Audio-Signals auf das $\sqrt[12]{2}$-fache (also um ca. 6 %) eine Erhöhung um einen Halbton.

Parameter

obere Puffergrenze (Verzögerung)	d_{UTB}
untere Puffergrenze (Verzögerung)	d_{LTB}
Dauer der Anpassungsphase	L

Bedingung für Einsatz

Der Mechanismus eignet sich sehr gut für kontinuierliche Ströme, sofern die Ausspielrate fein genug eingestellt werden kann, was für Audio- und Video-Übertragungen meist der Fall ist. Durch die Verwendung von Puffergrenzen kann dieser Mechanismus außerdem generell zur Einstellung einer Pufferverzögerung verwandt werden.

Zu beachten ist jedoch die Auswirkung der Ratenänderung auf die Medienqualität und die Häufigkeit, mit der die Rate angepasst wird. Auch schränkt das Verhältnis von Rahmengröße zur maximal erlaubten Verzögerung die Verwendung ein: Soll beispielsweise die Pufferverzögerung in einem 100 ms-Bereich bleiben und werden 80 ms-Rahmen übertragen, so kommt es zu extremen Schwankungen des Pufferfüllstands.

[7] Diese Grenzen werden in Abhängigkeit von maximaler und minimaler Verzögerung gewählt.

Auswirkung

Die Pufferverzögerung d_{buff} bleibt in einem Bereich[8] $[d_{Bmin}, d_{Bmax}]$. Die Dienstgüte des Audiosignals reduziert sich, abhängig von der Häufigkeit der Ratenanpassungen p_{adapt}, deren Dauer L und deren Ausmaß ΔR. Der Versatz zwischen mehreren Strömen wird reduziert.

$$\begin{aligned} d_{Bmin} &< d_{LTB} \\ d_{Bmax} &> d_{UTB} \\ d_{buff} &\in [d_{Bmin}, d_{Bmax}] \\ Q^*(m) &= Q(m) \cdot f(p_{adapt}, L, \Delta R) \end{aligned}$$

3.5.2.2 Verzögerungsanpassung für Übertragungen von Audio-Paketen

Quelle: [RKTS94]

Ablauf

In diesem Artikel stellen Ramjee, Kurose *et al.* einen Mechanismus mit vier Varianten zur Anpassung der Ausspielzeit von Audiopaketen vor. Dabei wird von einer *Talkspurt/Silence*–Übertragung ausgegangen: der Sender sendet in Sprachpausen („silence“) nichts, sondern nur wenn Aktivität (ein „talkspurt“) festgestellt wird.

Der erzeugte Verkehr besteht dann aus Paketbüscheln für die Talkspurt-Phasen, wobei der Empfänger die Pakete verzögern muss, um den *Jitter* auszugleichen. Nach der *absolute timing method* von Montgomery [Mon83] bestimmt dieser Mechanismus nun den Ausspielzeitpunkt $t_{out,i}$ des ersten Pakets i eines Talkspurts. Der Sendezeitpunkt $t_{send,i}$ und der Empfangszeitpunkt $t_{rec,i}$ werden benutzt, um Schätzungen der Übertragungsverzögerung D_i und deren Schwankung ΔD_i zu berechnen.

$$\begin{aligned} t_{out,i} &:= t_{send,i} + D_i + 4\,\Delta D_i \\ \text{mit} \quad \Delta D_i &= \alpha\,\Delta D_{i-1} + (1-\alpha)\left|D_i - (t_{rec,i} - t_{send,i})\right| \end{aligned}$$

Ohne eine Anpassung des Ausspielzeitpunkts durch Pufferung der Daten entstehen bei großer Schwankung der Verzögerung Ausspielunterbrechungen, d. h. das auszuspielende Paket wurde noch nicht empfangen. Andererseits ist eine zu hohe Verzögerung nicht wünschenswert, da sie die Interaktivität der Übertragung beeinträchtigt.

Parameter

Verschiedene Varianten des Mechanismus unterscheiden sich nur durch die Formeln zur Berechnung der Abschätzung der Verzögerung D_i. Ausgangspunkt ist hierbei die Dämpfung der Schwankung der Verzögerung durch einen Gewichtsfaktor $\alpha \in (0,1]$:

[8] Nicht im Bereich $[d_{LTB}, d_{UTB}]$, da diese Grenzen überschritten werden.

$$D_i \quad = \quad \alpha\, D_{i-1} + (1-\alpha)(t_{rec,i} - t_{send,i})$$

Die erste Variante verwendet verschiedene Werte für α, je nachdem ob sich die Verzögerung erhöht ($t_{rec,i} - t_{send,i} > D_{i-1}$) oder erniedrigt. Die nächste Variante verwendet als D_i die kleinste Verzögerung des vorhergehenden Talkspurts, und die letzte Variante versucht, den so genannten „Spike"-Effekt (die plötzliche Ankunft vieler Pakete in kurzer Zeit) auszugleichen.

Bedingung für Einsatz

Dieser Mechanismus kann bei Audio-Übertragungen eingesetzt werden, sofern der Sender im Talkspurt/Silence-Modus arbeitet. Sendet er jedoch kontinuierlich Audio-Daten aus, so hängt der Ausspielzeitpunkt immer vom Ausspielen der vorhergehenden Daten ab, d. h. die Festlegung des Ausspielzeitpunkts ist hier nicht direkt möglich.

Auswirkung

Die Ende-zu-Ende–Verzögerung der Audio-Übertragung passt sich an die tatsächliche Netzübertragung an. Dies kann aber zu Synchronisationsproblemen (hoher Versatz) mit anderen Medienströmen führen, weil keine Zielverzögerung eingestellt werden kann.

Die Verwurfswahrscheinlichkeit p_{drop} sinkt, da die Verzögerungsschwankungen zur Berechnung der Ausspielzeitpunkte einbezogen wird.

3.5.2.3 Geglättete, adaptive Video-Übertragung

Quelle: z. B. [DRR98]

Ablauf

Duffield *et al.* stellen hier einen Algorithmus zur Anpassung einer Video-Übertragung an ein Netz mit bekannter Bitrate vor. Der Sender beobachtet dabei den Füllstand des Sendepuffers und regelt die Datenrate über die Einstellung der Kodierungs-Parameter. Hierdurch wird erreicht, dass der Pufferfüllstand einen bestimmten Wert nicht überschreitet, d. h. die Verzögerung durch Passieren dieses Puffers bleibt unter einem vorgegebenen Wert, wird „geglättet" (engl. *smoothing*).

Konkret wird z. B. beim Erreichen eines Füllstands von 100 ms die Datenrate durch stärkere Komprimierung der MPEG-Rahmen reduziert, so dass der Pufferfüllstand und damit die Pufferverzögerung sinkt.

Parameter

maximale Sendepufferverzögerung	$d_{SB,max}$
optimale Datenrate	B_{opt}
gewählte Datenrate zum Zeitpunkt t	$B(t)$
maximale Glättung	$\alpha \leq \frac{B(t)}{B_{opt}}$

Bedingung für Einsatz

Die Anwendung dieses Mechanismus eignet sich bei Verwendung einer Videoübertragung mit entsprechenden Parametern. Er geht davon aus, dass im Sender eine Verkehrsratenformung (engl. *traffic smoothing*) erfolgt, für die ihm das Netz die momentan verfügbare Übertragungsbitrate mitteilt, was nicht für alle Netztechnologien möglich ist.

Auswirkung

Dieser Mechanismus erhöht bereits im Sender die Übertragungsverzögerung. Durch die Änderung der Kodierungsparameter schwankt die Bildqualität schon nach der Erzeugung. Erbringt jedoch das Netz die eingestellte Rate, so wird dieser Mechanismus die Übertragungsverzögerung und -verluste insgesamt reduzieren.

3.5.2.4 Regelung der Videokodierungsparameter

Quelle: u. a. [BT94], [WTSB01]

Ablauf

Bolot und Turletti stellen in [BT94] einen Mechanismus vor, bei dem die momentane Paketverlustwahrscheinlichkeit vom Empfänger an den Sender gemeldet wird. Dieser passt dann die Bitrate des erzeugten Video-Stroms an, indem die Parameter des Kodierungsalgorithmus variiert werden.

Hier werden drei Parameter eines H.261-Kodierers (s. 2.2.3.5.1, S. 18) angepasst: Die Bildrate (wie viele Bilder pro Sekunde kodiert werden), die Quantisierungskoeffizienten (diese bestimmen die Genauigkeit der Kodierung bzw. den Kompressionsverlust) und Bewegungs-Erkennungs-Schwelle (engl. *movement detection threshold*).

In [BT94] wird ein Zielwert B_T für die Bitrate B ermittelt: ist die Paketverlustwahrscheinlichkeit p_{Ploss} größer als ein maximal tolerierbarer Wert $p_{tolerate}$, so beträgt die neue Rate einen Bruchteil $B_T = \beta_{reduce} \cdot B$, mit $\beta_{reduce} < 1$, der momentanen Rate oder wird auf einen Mindestwert B_{min} gesetzt. Andernfalls wird der Zielwert mit dem Faktor $\alpha_{gain} > 1$ multipliziert, darf aber einen Maximalwert B_{max} nicht überschreiten. (Die genaue Abbildung der Ziel-Bitrate auf Parameter des Kodierverfahrens wird nicht beschrieben.)

Parameter

Tolerierbare Verlustwahrscheinlichkeit	$p_{tolerate}$
Erhöhungsfaktor	$\alpha_{gain} > 1$
Reduzierungsfaktor	$\beta_{reduce} < 1$
Mindestrate	B_{min}
Maximalrate	B_{max}

Bedingung für Einsatz

Der hier vorgestellte Mechanismus ist speziell auf H.261-Videoübertragungen abgestimmt, kann aber auch auf H.263 übertragen werden. Ein ähnlicher Mechanismus für MPEG wird z. B. in [WTSB01] vorgestellt.

Auswirkung

Die Qualität des Video-Stroms passt sich den Übertragungsverlusten an. Dies hängt jedoch in großem Maße vom verwendeten Kodierverfahren und der Abbildung der Zielrate auf die Kodierungsparameter ab. Die Bitrate regelt sich auf einen Wert ein, bei dem die Verlustwahrscheinlichkeit den Toleranzwert nicht überschreitet, bzw. auf den Mindestwert.

$$\begin{aligned} B_{min} \leq \quad & B \quad \leq B_{max} \\ p_{Ploss} \quad \leq \quad & p_{tolerate} \end{aligned}$$

3.5.2.5 Redundante Audio-Übertragung

Quelle: [KHHC97], [PKHea97]

Ablauf

Um den Verlust von Paketen in Audio-Übertragungen auszugleichen, wird hier jedem Paket die Audioinformation von n_L Vorgänger-Rahmen hinzugefügt. Der Versatz (engl. *Offset*) v_j gibt dabei an, in welchem Paket $i+v_j$ die Ersatzdaten in der Kodierung k_j für den Rahmen i gepackt werden. Wird in einem Paket mehr als ein Redundanz-Rahmen mitgeschickt, so werden die Audio-Rahmen umso stärker verlustbehaftet komprimiert, je älter sie sind. Abb. 3.2 zeigt ein Beispiel mit $n_L = 2$ und $v_j = j$, d. h. $v_1 = 1$ und $v_2 = 2$.

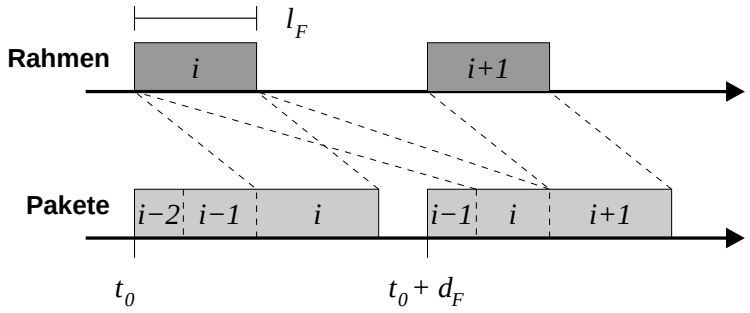


Abbildung 3.2: Zuordnung der Audio-Rahmen zu Paketen

Geht nun Paket i verloren, so kann der Empfänger bei Erhalt des Pakets $i+v_j$ aus diesem den Audio-Rahmen i rekonstruieren, jedoch in der Kodierung k_j. Die Qualität der Kodierungen k_j ist dabei niedriger als die der „Hauptkodierung“ k_0, wobei $q_j \leq 1$ diese reduzierte Güte ausdrückt.

Durch Erhöhung der Redundanz-Stufen oder des Versatzes kann auch der Verlust von mehreren Paketen abgesichert werden. Dieser Mechanismus hat Ähnlichkeit mit selektiver wiederholter Übertragung und Vorwärts-Fehlerkorrektur.

Mögliche Kodierungen (s. 2.2.3.4, S. 16, [CK96]) sind z. B. G.711 (μ-law, 64 kbit/s), G.721 (ADPCM, 16 kbit/s), GSM (13.2 kbit/s), G.723.1 (6.3 oder 5.3 kbit/s), die in dieser Reihenfolge mit abnehmender Audio-Qualität arbeiten.

Parameter

Redundanz-Stufen	n_L
Kodierungsfolge	$k_1, \ldots, k_{n_L}$
Versatz	$0 < v_1 < \ldots < v_{n_L}$
Audio-Rahmen-Länge	l_F
Audio-Rahmen-Dauer	d_F
Audio-Qualität relativ zur Hauptkodierung	$1 = q_0 \geq q_1 \geq \ldots \geq q_{n_L}$

Bedingung für Einsatz

Dieser Mechanismus eignet sich besonders für Audio-Übertragungen, da hier eine große Anzahl an verschiedenen Kodierverfahren zur Verfügung steht und der Umfang der Audio-Daten (im Vergleich zu Video-Daten) gering ist. Jedoch führt die Verwendung dieses Mechanismus zu zusätzlichem Rechenleistungsbedarf C_{cod}, da pro Sekunde $1\text{s}/d_F$ Rahmen der Länge l_F mit n_L verschiedenen Verfahren kodiert werden müssen. Die benötigte Bitrate nimmt um $B_{redundancy}$ zu. Schließlich muss wie bei selektiver wiederholter Übertragung das Ausspielen der Audio-Daten stark verzögert werden, um es den Ersatz-Daten zu erlauben, den Empfänger zu erreichen.

$$
\begin{aligned}
C_{avail} &\geq C_{cod} = \frac{1\text{s}}{d_F} \sum_{i=1}^{n_L} C_{cod,i}(l_F) \\
B_{avail} &\geq B_{redundancy} = \frac{1}{d_F} \sum_{i=1}^{n_L} l_{cod,i}(l_F) \\
d^*_{max} &\geq d^*
\end{aligned}
$$

Auswirkung

Der Rechenleistungsbedarf C^*, die benötigte Bitrate B^* und die Ende-zu-Ende–Verzögerung d^* erhöhen sich.

$$
\begin{aligned}
C^* &= C + C_{cod} \\
B^* &= B + B_{redundancy} \\
d^* &= d + n_L \cdot d_F
\end{aligned}
$$

Die Rahmenverlustwahrscheinlichkeit nimmt ab, denn nur wenn alle Pakete mit Redundanz-Rahmen ebenfalls verloren gehen, ist der Rahmen nicht rekonstruierbar.

$$
p^*_{Floss} = p^{(n_L+1)}_{Ploss}
$$

Die Qualität der Audio-Übertragung nimmt mit dem Faktor q^*, abhängig von der Paketverlustwahrscheinlichkeit p_{Ploss}, ab, da bei Verlust eines Rahmens der erste Redundanz-Rahmen ausgespielt wird. Erreicht auch dieser nicht den Empfänger, so wird der zweite Redundanz-Rahmen ausgespielt usw.

$$\begin{aligned} q^* &= q_0 \cdot (1 - p_{Ploss}) + q_1 \cdot p_{Ploss}(1 - p_{Ploss}) + q_2 \cdot p_{Ploss}^{2}(1 - p_{Ploss}) + \ldots \\ &= (1 - p_{Ploss}) \cdot \sum_{i=0}^{n_L} q_i \cdot p_{Ploss}^{i} \end{aligned}$$

3.5.2.6 Mehrlagige Video-Übertragung

Quelle: [LAP99] (s. a. 2.4.6.4)

Ablauf

Um Multicast-Übertragungen im Internet durchzuführen, wird hier aufbauend auf der IP-Multicast Architektur (vgl. IGMP, [Dee89]) eine Anwendung beschrieben, die eine mehrlagige Video-Kodierung (engl. *multi-layered video coding*) verwendet. Jede Lage der Video-Daten wird dabei an eine eigene Multicast-Adresse gesandt, so dass die Empfänger auswählen können, welche der Lagen sie erhalten wollen. Das IP-Multicast-Protokoll sorgt dafür, dass in einem Teilnetz nicht verwandte Daten dorthin nicht versandt werden (engl. *pruning*).

Erfährt der Empfänger Paketverluste über einer bestimmten Schwelle p_{high}, so gibt er eine Lage auf, d. h. er stellt den Empfang der entsprechenden Multicast-Adresse ein. Ist dagegen der Paketverlust für eine Beobachtungsdauer d_{watch} kleiner als eine untere Grenze p_{low}, so führt der Empfänger ein so genanntes „Join-Experiment“ durch, d. h. er fordert eine zusätzliche Lage an und prüft, ob die Übertragung erfolgreich ist. Ist dies nicht der Fall, so stoppt er den Empfang wieder.

Parameter

Die Aufteilung der Daten auf die Lagen hängt stark von der verwendeten Kodierung ab.

Beobachtungsdauer zum Zeitpunkt t	$d_{watch}(t)$
Paketverlustwahrscheinlichkeit für zusätzliche Lage	p_{low}
Paketverlustwahrscheinlichkeit für Aufgabe einer Lage	p_{high}

Bedingung für Einsatz

Der Einsatz dieses Mechanismus beschränkt sich auf hierarchische Kodierungen und ist nur bei der Verwendung von IP-Multicast sinnvoll.

Auswirkung

Die Netzauslastung wird optimiert, da nur die Daten übertragen werden, die wirklich benötigt werden und den Empfänger erreichen können. Dagegen steht jedoch ein hoher Verwaltungsaufwand im Netz. Zudem sind die Kodierung und Dekodierung aufwändig.

3.5.3 Mechanismen der Kommunikations-Schicht

Diesen Mechanismen ist gemeinsam, dass sie prinzipiell für jedes zu übertragende Medium eingesetzt werden können. Im konkreten Fall muss natürlich die Auswirkung auf die Dienstgüte berücksichtigt werden.

Einige Mechanismen der Kommunikations-Schicht müssen auf Informationen wie die Zugehörigkeit eines Pakets zu einem Medienstrom oder den Zeitstempel eines Datenobjekts zugreifen, was über Protokolle wie UDP (jeder Strom wird über eine „Port-Nummer“ identifiziert) und RTP (führt einen Zeitstempel mit) medienunabhängig erreicht wird.

3.5.3.1 Paket-Scheduling

Literatur: [Zha95]

Ablauf

Typischerweise erzeugt die Kommunikations-Schicht aus einem Medien-Rahmen mehrere Pakete. Dabei bestimmt der Parameter Paketlänge l_P, in wie viele Pakete ein Rahmen geteilt wird. Bei der Übergabe an das Netz kann der Verkehr „geformt“ werden (engl. *traffic shaping*), indem z. B. bestimmte Paketabstände erzwungen werden. Dies verhindert die Übergabe von Büscheln (engl. *bursts*) an das Netz, so dass die Spitzen-Bitrate einen bestimmten, maximal erlaubten Wert nicht überschreitet. Dies könnte durch Pufferüberlauf oder Verwerfen in den Netzknoten zu Paketverlusten führen.

Da die Netz-Schicht als Multiplexer agiert, der mehrere Medienströme zusammenfasst, stehen verschiedene Bedienstrategien (engl. *scheduling algorithms*) zur Verfügung. So ist es z. B. sinnvoll, verzögerungsempfindliche Daten weniger empfindlichen vorzuziehen und eher zu übertragen.

Parameter

Paketgröße	l_P
maximal erlaubte Bitrate	B_{max}
Schedulingverfahren	*Round Robin*, Prioritäten, etc.

Bedingung für Einsatz

Jede Übertragung verwendet ein Scheduling, wobei durch die Änderung der Parameter verschiedene Effekte erreicht werden können. So wird z. B. durch die Änderung der maximalen Bitrate die Dauer der Übertragung eines Rahmens beeinflusst und somit dessen minimale Verzögerung.

Auswirkung

Durch die Änderung der Scheduling-Parameter wird der in das Netz gesandte Verkehr verändert. Der Mindestabstand der Pakete d_P, der minimale Sendeabstand d_{send} zwischen Rahmen und die minimale Dauer der Übermittlung zum Sender d_{min} (mit Paket-Verzögerung $D(l_P)$ im Netz) verändern sich[9] (Abb. 3.3). Die Zusammensetzung des Verkehrsstroms aus Paketen verschiedener Medienströme ändert sich je nach Multiplex-Verfahren.

[9] Die Größe der Paketköpfe wird hier vernachlässigt.

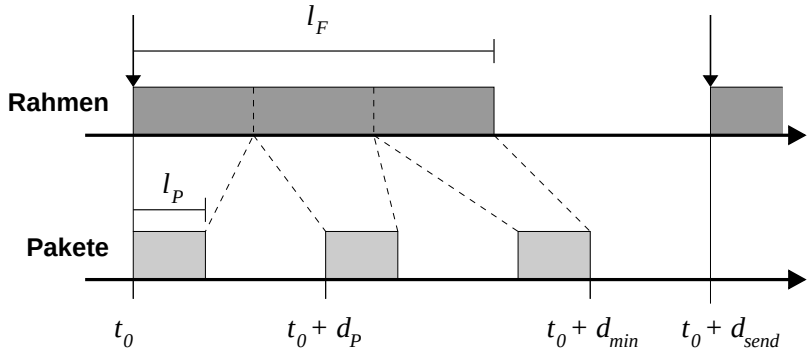


Abbildung 3.3: Paket-Abstand durch Scheduling

$$
\begin{aligned}
d_P &= \frac{l_P}{B_{max}} \\
d_{send} &= \frac{l_F}{B_{max}} \\
d_{min} &= \frac{l_F - l_P}{B_{max}} + D(l_P)
\end{aligned}
$$

3.5.3.2 Verlustlose Komprimierung

Ablauf

Je nach verwendeter Kodierung enthalten Medien-Daten nach der Digitalisierung noch einen Anteil an Redundanz. Dieser Anteil kann entfernt werden, indem Standard-Kompressionsalgorithmen (z. B. nach Lempel-Ziv [ZL77], nach Huffman [LH87]) auf die Daten angewandt werden. Dies geschieht in der Kommunikations-Schicht, somit für die Anwendungs-Schicht nicht sichtbar.

Während es für diese Kompressionsalgorithmen unerheblich ist, von welcher Art die Medien-Daten sind, ist die Kompression von Daten, deren Redundanz bereits entfernt wurde, nicht zu empfehlen. Im besten Fall wird für eine kleine Reduktion der Datenmenge viel Rechenleistung verbraucht; im schlechtesten Fall wird die „komprimierte“ Datenmenge nach Anwendung der Algorithmen durch eingefügte Verwaltungsdaten größer.

Parameter

Netzübertragungsverzögerung	$D(l)$
Bitrate des Medienstroms	B_{SR}
Maximal erlaubte Verzögerung	d_{max}
Rahmengröße	l_F
Maximale Kompressionsbitrate des Endgeräts	B_{compr}
Maximale Dekompressionsbitrate des Endgeräts	$B_{decompr}$
Kompressionsfaktor abh. von Kodierung	c_K

Bedingung für Einsatz

Der Einsatz der Komprimierung ist nur sinnvoll, wenn der Kompressionsfaktor c_K für Daten der verwendeten Kodierung K kleiner 1 ist. Die Leistungsfähigkeit der Endgeräte und der verwandte Kompressionsalgorithmus bestimmen die maximale Kompressions- und Dekompressionsbitrate B_{compr} und $B_{decompr}$, d. h. wie viele Daten pro Sekunde (de-) komprimiert werden können, falls die gesamte Rechenleistung verfügbar ist. Die tolerierbare Verzögerung d_{max} darf nicht überschritten werden, d. h. die Kompression und Dekompression muss entsprechend schnell durchgeführt werden können, indem die benötigten Anteile an der Rechenleistung C_{compr} und $C_{decompr}$ zur Verfügung stehen.

$$\begin{aligned} 1 &> c_K \\ C_{avail,sender} &\geq C_{compr} \\ C_{avail,receiver} &\geq C_{decompr} \\ d_{max} &\geq d^* \end{aligned}$$

Auswirkung

Die neue benötigte Übertragungsbitrate B^*_{SR} sinkt und damit auch die Netz-Verzögerung, somit wird es zusammen mit der zusätzlichen Kompressions- und Dekompressionsverzögerung zu einer Änderung der neuen Übertragungsverzögerung d^* kommen. Es wird zusätzliche Rechenleistung, C_{compr} im Sender und $C_{decompr}$ im Empfänger, für die Bearbeitung des Medienstroms benötigt.

$$\begin{aligned} B^*_{SR} &= c_K \cdot B_{SR} \\ C_{compr} &= \frac{B_{SR}}{B_{compr}} 100\ \% \\ C_{decompr} &= \frac{B^*_{SR}}{B_{decompr}} 100\ \% \\ d^* &= \frac{l_F}{B_{compr}} + D(c_k \cdot l_F) + \frac{c_K \cdot l_F}{B_{decompr}} \end{aligned}$$

Außerdem kann es zu einer Reduktion der neuen Verlustwahrscheinlichkeit auf Rahmenebene p^*_{Floss} kommen, da pro Rahmen weniger Pakete übertragen werden müssen.

$$p^*_{Floss} = 1 - (1 - p_{Ploss})^{\lceil c_K \cdot \frac{l_F}{l_P} \rceil} \quad \leq \quad p_{Floss} = 1 - (1 - p_{Ploss})^{\lceil \frac{l_F}{l_P} \rceil}$$

3.5.3.3 Selektive wiederholte Übertragung

Quelle: [PP96]

Ablauf

Der Empfänger erkennt, dass ein Paket bei der Übertragung verloren ging. Sofern noch die Chance besteht, das Paket *rechtzeitig* zu erhalten, fordert er beim Sender eine erneute Übertragung an[10].

Parameter

Übertragungsverzögerung Sender → Empfänger	D_{SR}
Übertragungsverzögerung Empfänger → Sender	D_{RS}
Rundlaufverzögerung (engl. *Round-Trip Time*)	$D_{RTT} = D_{SR} + D_{RS}$
Zeit bis zur Erkennung eines Verlustes	d_{detect}
Maximal erlaubte Verzögerung	d_{max}

Bedingung für Einsatz

Dieser Mechanismus kann nur eingesetzt werden, wenn das erneut gelieferte Paket mit einer geringeren als der maximal erlaubten Verzögerung den Empfänger erreichen kann. Andernfalls würde das Paket in einer höheren Schicht wegen Verspätung verworfen und es entstünde keine Verbesserung.

Daher muss gelten:

$$\begin{aligned} D_{RTT} &\leq d_{max} - d_{detect} - D_{SR} \\ \text{bzw.} \qquad 2D_{SR} + D_{RS} + d_{detect} &\leq d_{max} \end{aligned}$$

Die Zeit bis zur Entdeckung eines verlorenen Pakets ist abhängig von der eingesetzten Methode. Einerseits kann an einer übersprungenen Sequenznummer erkannt werden, dass ein Paket verloren ging – dann ist d_{detect} der Sendeabstand der Pakete. Wenn aber die Schwankung der Übertragungsverzögerung im Vergleich zum Sendeabstand klein ist, dann kann bei Verstreichen der errechneten erwarteten Ankunftszeit von einem Verlust ausgegangen werden, d. h. d_{detect} ist deutlich kleiner (Größenordnung der Verzögerungsschwankung).

Auswirkung

Die beobachtete Paketverlustwahrscheinlichkeit $p_{Ploss,observed}$ reduziert sich, da ein Paket nur noch dann verloren ist, wenn entweder das Anforderungspaket oder das wiederholte Paket ebenfalls verloren geht. Die Verzögerung wird auf den Mindestwert d_{max} gesetzt. Die benötigte Bitrate erhöht sich in beiden Richtungen. (Annahme: Alle verlorenen Paket werden genau einmal erneut übertragen.)

$$\begin{aligned} p_{Ploss,observed} &= p_{Ploss}\left(p_{Ploss} + (1 - p_{Ploss})p_{Ploss}\right) \\ &= p_{Ploss}^{2}(2 - p_{Ploss}) \\ d^{*} &= d_{max} \\ B_{SR}^{*} &= (1 + p_{Ploss})B_{SR} \end{aligned}$$

[10] Dies ist ein Unterschied zum klassischen *Automatic Repeat reQuest (ARQ)*-Verfahren, bei dem verlorene Pakete immer neu angefordert werden.

3.5.3.4 Vorwärts-Fehlerkorrektur

Quelle: [LC83], [Pur95]

Ablauf

Der Sender fügt den Daten vorsorglich Redundanz hinzu (*Forward Error Correction*, FEC), die eine gewisse Menge an Verlusten[11] ausgleichbar macht, da die verlorenen Daten aus den erhaltenen wieder hergestellt werden können. Hierbei werden Verfahren der Kodierungstheorie angewandt.

Da in paketvermittelnden Netzen Bitfehler meist das Verwerfen eines ganzen Pakets zur Folge haben, wird hier FEC auf Rahmenebene eingesetzt. Es werden für m Datenpakete n Pakete mit Redundanz übertragen, so dass der Verlust von e Paketen ausgeglichen werden kann. Einerseits kann die Redundanz auf alle Pakete verteilt sein, andererseits können $k = n - m$ zusätzliche Redundanz-Pakete gesandt werden.

Eine Implementierung mit einem zusätzlichen Redundanz-Paket kann z. B. alle Pakete bitweise antivalent verknüpfen („XOR“). Die Wiederherstellung eines verlorenen Pakets kann dann ebenfalls durch XOR–Verknüpfung erfolgen, d. h. $e = 1$ und $k = 1$. Typischerweise wird dieses Verfahren für Pakete mit gleicher Größe angewandt. Eine weitere Möglichkeit ist die Übertragung von Prüfbits („Paritätsbit“).

Alternativ kann z. B. ein Bose-Chaudhuri-Hocquenghem (BCH-) Code eingesetzt werden. Jedoch wäre bei dieser Kodierung bereits zum Schutz von einem aus vier Paketen die Übertragung von sieben (!) Paketen nötig ($m = 4, e = 1, n = 7$) – in diesem Fall wäre die XOR–Verknüpfung deutlich effizienter. Zudem benötigen Kodierung und Dekodierung mehr Rechenleistung und Speicherzugriffe.

Parameter

Zahl zu schützender Pakete	m
Zahl der zusätzlichen Pakete	$k > 0$
Zahl zu übertragender Pakete	$n = m + k$
maximale Anzahl rekonstruierbarer Pakete	$e \leq k = n - m$
Effizienz	$\frac{m}{n} < 1$

Bedingung für Einsatz

Dieser Mechanismus kann eingesetzt werden, wenn die verfügbare Bitrate B_{avail} für die zusätzliche Redundanz ausreichend groß ist[12]. Außerdem müssen Rechenleistung und Speicher für die Kodierung und Korrektur vorhanden sein.

$$\begin{aligned} B_{avail} &\geq B_{SR} \cdot \frac{n}{m} \\ C_{avail} &\geq C + C_{coding} \end{aligned}$$

[11] Der mögliche Schutz eines Pakets gegen Bitfehler in den Nutzdaten wird hier nicht betrachtet.

[12] Ggf. muss also zunächst für geringeren Bedarf gesorgt werden!

Auswirkung

Die beobachtete Rahmenverlustwahrscheinlichkeit („Blockfehlerwahrscheinlichkeit“) $p_{Floss} = p_{>e}$ reduziert sich, da nur der Verlust von mehr als e Paketen zu einem Rahmenverlust führt. Die benötigte Bitrate erhöht sich auf das $\frac{n}{m}$-fache. Die neue mittlere Übertragungsdauer d^* innerhalb der Kommunikations-Schicht erhöht sich, da bei Verlust von bis zu e der zu schützenden Pakete auf die k Redundanz-Pakete gewartet werden muss. Wird nur die Verzögerung der Rahmen gemessen, die an die Anwendung übergeben werden können, müssen die Wahrscheinlichkeiten p_0 der fehlerfreien Ankunft der ersten m Pakete und $p_{1\cdots e}$ für 1 bis e Paketverluste durch die Wahrscheinlichkeit $1 - p_{>e}$ dieser betrachteten Fälle normiert werden[13].

$$
\begin{aligned}
p_{Floss} \;=\; p_{>e} \;&=\; \sum_{i=e+1}^{n} \binom{n}{i} \, p_{Ploss}^{\;i} \, (1-p_{Ploss})^{n-i} \\
B^* \;&=\; B \cdot \frac{n}{m} \\
d^* \;&=\; \frac{p_0 \cdot m + p_{1\cdots e} \cdot n}{1 - p_{>e}} \cdot D(l_P)
\end{aligned}
$$

$$
\text{mit} \quad p_0 = (1-p_{Ploss})^m \qquad p_{1\cdots e} = \sum_{i=1}^{e} \binom{n}{i} p_{Ploss}^{\;i} \cdot (1-p_{Ploss})^{n-i}
$$

3.5.3.5 Mediensynchronisierung durch Änderung des Abfragezeitpunkts

Quelle: [TYS99]

Ablauf

Bei der Übertragung kontinuierlicher Medienströme ist es nicht sinnvoll, Daten zu übertragen, die vom Empfänger sicher verworfen werden, da sie nicht rechtzeitig ausgespielt werden können. Treetasanatavorn *et al*. stellen ein Protokoll vor, mit dem der Sender die momentane Übertragungsverzögerung und -schwankung erfährt und hiermit berechnen kann, welche Daten als nächstes zu übertragen sind, so dass sie noch ausgespielt werden können. Dies kann zum Auslassen von Daten führen, die dann nicht versandt werden.

Bei diesem Mechanismus hat die Berechnung der Verzögerung und deren Schwankung großen Einfluss auf die Leistung. Bei einer Überschätzung müssen Daten im Empfänger zusätzlich verzögert werden, bei einer Unterschätzung leidet die Synchronisierung.

Parameter

Berechnung der Verzögerung zum Zeitpunkt t	$d(t)$
Berechnung der Verzögerungsschwankung zum Zeitpunkt t	$\Delta d(t)$

[13] Die Fälle, in denen Rahmen bereits nach Erhalt von $n - e$ Paketen vorzeitig dekodiert werden könnten, sind nicht in der Berechnung enthalten.

Bedingung für Einsatz

Da der Mechanismus den Abfragezeitpunkt im Sender beeinflusst, eignet er sich besonders für vorgefertigte Daten, z. B. Video-on-Demand. Aber auch wenn im Empfänger nicht viel Speicher für die Pufferung der Daten bereit steht, kann dieser Mechanismus Vorteile bringen. Zudem verhindert er, dass unnötigerweise Daten übertragen werden, die der Empfänger verwerfen müsste; er entlastet also das Netz.

Auswirkung

Der Mechanismus erhöht den Signalisierungsverkehr und den Aufwand im Sender. Die Ende-zu-Ende–Verzögerung passt sich der Übertragungsverzögerung an, jedoch kann sich auch der Ausspiel-Jitter erhöhen. (Daher gibt es Erweiterungen zur Glättung des Ausspielens.)

3.5.3.6 Mediensynchronisierung durch Auslassung

Quelle: z. B. [TYS99]

Ablauf

Übertragungseffekte wie Büschelbildung und Verzögerungsschwankung können dazu führen, dass sich der Ausspielpuffer im Empfänger mit Mediendaten füllt und dadurch die Ausspielverzögerung höher ist als gewünscht. Um nun wieder einen Zielwert zu erreichen, kann der Empfänger Daten absichtlich verwerfen (engl. *to discard*), d. h. nicht ausspielen. Je nach Medium beeinflusst dies die Dienstgüte unterschiedlich. Bei Audio-Übertragungen von Sprache können z. B. Verluste von über 10 % der Daten tolerierbar sein [WS96]. Bei einer Video-Übertragung reduziert sich die Bildrate, es kommt zu erhöhtem Jitter.

Der Empfänger zieht dabei für ein Datum die Erzeugungszeit t_{gen} von der erreichbaren Ausspielzeit t_{out} ab und vergleicht diesen Wert $\hat{d}$ mit der maximal erlaubten Verzögerung $\hat{d}_{max}$. Danach ermittelt er die Menge der zu verwerfenden Daten und deren Position im Medienstrom.

Parameter

Maximal erlaubte Verzögerung	$\hat{d}_{max}$
Maximal tolerierbarer Verlust	$p_{discard}$
Maximal tolerierbarer Jitter	Δd_{max}

Bedingung für Einsatz

Dieser Mechanismus kann eingesetzt werden, wenn eine Begrenzung der maximalen Ende-zu-Ende–Verzögerung vorliegt (d. h. bei interaktiven Übertragungen) und das Verwerfen von Daten tolerierbar ist. Insbesondere wenn Alternativen wie die Änderung der Ausspielrate (s. 3.5.2.1, S. 48) nicht zur Verfügung stehen oder nicht ausreichen, um die Anpassung durchzuführen.

Auswirkung

Der Jitter der Daten erhöht sich, da ein Sprung in der Ende-zu-Ende–Verzögerung erfolgt. Die Mediendienstgüte reduziert sich, da z. B. weniger Bilder pro Sekunde angezeigt werden.

3.5.4 Tabellarische Klassifikation der Mechanismen

In Tab. 3.4 werden die Eigenschaften der vorgestellten Anpassungsmechanismen gegenübergestellt. Mit dem Symbol ✓ wird gekennzeichnet, dass der Mechanismus medienabhängig ist. Die Symbole ↑, ↓, und ↕ zeigen an, ob der Wert eines Parameters (Qualität Q der erzeugten Daten, Bandbreite B, Verzögerung d, beobachtete Verluste) erhöht oder erniedrigt wird, bzw. ob beides möglich ist.

	Nutzer-Schicht	Med.	Q	B	d	Verlust
3.5.1.1	Meta-Policy	–	↕	↕	↕	↕
3.5.1.2	Auswahl auf Inhaltsebene	✓	↕	↕	↕	↕
	Anwendungs-Schicht	**Med.**	Q	B	d	Verlust
3.5.2.1	Synchronisierung von Datenströmen	–	–	–	↕	–
3.5.2.2	Verzögerungsanpassung für Übertragungen von Audio-Paketen	✓	–	–	↕	–
3.5.2.3	Geglättete, adaptive Video-Übertragung	✓	↕	↕	↕	↓
3.5.2.4	Regelung der Videokodierungsparameter	✓	↕	↕	–	↓
3.5.2.5	Redundante Audio-Übertragung	✓	–	↑	↑	↓
3.5.2.6	Mehrlagige Video-Übertragung	✓	–	↕	–	↓
	Kommunikations-Schicht	**Med.**	Q	B	d	Verlust
3.5.3.1	Paket-Scheduling	–	–	–	↕	–
3.5.3.2	Verlustlose Komprimierung	–	–	↓	↕	↓
3.5.3.3	Selektive wiederholte Übertragung	–	–	↑	↑	↓
3.5.3.4	Vorwärts-Fehlerkorrektur	–	–	↑	↑	↓
3.5.3.5	Änderung des Abfragezeitpunkts	–	–	↓	↕	–
3.5.3.6	Auslassung	–	–	–	↓	↑

Abbildung 3.4: Klassifikation der Mechanismen

4 Experimentelle Untersuchung

4.1 Einführung

Im Grundlagen-Kapitel wurde motiviert, dass verteilte Multimedia-Anwendungen in Zukunft verstärkt eingesetzt werden. Da sich in der Vergangenheit Netztechnologien, die Ende-zu-Ende–Dienstgüte-Garantien bereitstellen können (z. B. ATM), nicht im erwarteten Umfang durchsetzen konnten, wird hier davon ausgegangen, dass auch in Zukunft Netze mit schwankendem Übertragungsverhalten (vor allem mit IP-Technologie) benutzt werden.

Hier hat also die Anwendung keine Möglichkeit, die Behandlung ihres Verkehrs durch das Netz zu beeinflussen, vielmehr bleibt ihr nur die Möglichkeit, sich selbst an den Zustand der Übertragung anzupassen. Daher ist es das Ziel dieser Arbeit, die *Anpassungsmechanismen* und *-algorithmen* zu untersuchen, die die Anwendung in die Lage versetzen, die von ihr erbrachte Ende-zu-Ende–Dienstgüte zu verbessern.

Diese Untersuchung bewertet einerseits die Mechanismen hinsichtlich ihrer Eignung zur Kompensation verschiedener unerwünschter Verhaltensweisen des Netzes. Sie zeigt andererseits die Entwicklung von Adaptionsalgorithmen für gegebene Szenarien. Sie bewertet die Algorithmen durch Vergleich der Leistung in einem experimentellen Rahmen und durch Analyse der Ergebnisse (s. Kap. 5, S. 103). Sie stellt neue Algorithmen und Mechanismen vor.

Um die Dienstgüte einer Multimedia-Anwendung zu erhöhen, stehen prinzipiell weitere Ansätze zur Verfügung, die jedoch an dieser Stelle *nicht* untersucht werden sollen. Zum einen ist es denkbar, beim Auftreten einer Reduktion der erfahrenen Dienstgüte einen vollständigen Neuaufbau der Kommunikation durchzuführen und die Ressourcen entsprechend des momentanen Zustands zu verteilen. Hierbei muss jedoch beachtet werden, wie sehr der Nutzer durch diese großen Umstellungen gestört wird und dass dies daher nicht sehr häufig durchgeführt werden sollte.

Ebenso kann versucht werden, die Dienstgüte innerhalb der Netze zu erhöhen. Zum Beispiel können Daten unterschiedlich behandelt werden, je nach Art des Datenstroms, oder das Prinzip der Überdimensionierung kann verwendet werden, um die Wahrscheinlichkeit von Verschlechterungen der Übertragung zu reduzieren.

Nicht betrachtet wird daneben die Optimierung der Bestandteile einer Multimedia-Anwendung. So könnten Verbesserungen der Kodierverfahren hinsichtlich Datenrate, Medien-Qualität und Rechenaufwand untersucht bzw. neue Kodierverfahren implementiert werden. Die verwendeten Rechner und Betriebssysteme könnten schneller und effizienter ausgelegt werden.

Ebensowenig ist es beabsichtigt, eine adaptive Multimedia-Anwendung zu implementieren, die als „Produkt“ von Endanwendern genutzt werden kann.

In diesem Kapitel wird zunächst die „eXPERIMENTATION PLATFORM“ vorgestellt, die für die Durchführung der Untersuchungen entwickelt wurde. Insbesondere werden ihre Grundstruktur und die Konzepte für Dienstgüte und Datenfluss genauer erläutert. Es folgen dann Beschreibungen der

Implementierungen des Audio- und Video-Mediums, ihrer Steuer- und Zustandsparameter und der Anpassungsmechanismen und Adaptionsalgorithmen. Danach werden die untersuchten Szenarien vorgestellt und wie das Verhalten eines Übertragungsnetzes emuliert wurde. Das Kapitel wird mit einer Beschreibung einer medienübergreifenden Adaption abgeschlossen.

4.2 Experimentierplattform

Neben der analytischen Untersuchung der Anpassungsmechanismen und -algorithmen wird ihre Leistung *experimentell* bewertet. Hierzu ist es nötig, eine Multimedia-Übertragung in allen ihren Teilen beeinflussen zu können, von der Auswahl der Medien über die Festlegung einzelner Parameter der Anwendung, bis hin zum Netzverhalten.

Dies führte, ausgehend von einem eingeschränkten Prototypen [Zoj98], zu Entwurf und Implementierung der EXPERIMENTATION PLATFORM. Dieses modulare System soll folgende Ziele erfüllen:

- Es wird ein repräsentativer Teil einer Multimedia-Anwendung nachgebildet, d. h. die hier gewonnenen Erkenntnisse können auf andere Anwendungen übertragen werden.
- Die Implementierung schränkt die Art der Anpassung nicht ein. Es ist möglich, völlig verschiedene Ansätze zur Regelung zu implementieren.
- Die Änderung des Verhaltens der Anwendung auf der Ebene des Kommunikationsnetzes und auf der Ebene der Dienstgüten-Regelung ist leicht und schnell durchzuführen.
- Es ist möglich, an Schnittstellen zwischen Modulen sog. „Messpunkte“ zur Erfassung der Verkehrscharakteristik (Rahmen- und Paketgrößen, zeitlicher Abstand zwischen Daten, Verzögerung, usw.) anzubringen.

4.2.1 Grundstruktur

Grundprinzip des Entwurfs der EXPERIMENTATION PLATFORM ist eine Trennung der Anwendung in Schichten (engl. *layers*) und Ebenen (engl. *planes*). Die Aufteilung in Schichten orientiert sich dabei an der in 3.1, S. 35 vorgestellten Vereinfachung des ISO/OSI-Basis-Referenzmodells (s. 2.1.1, S. 5). Die Aufteilung in Ebenen (Abb. 4.1) erfolgt im Hinblick auf die verschiedenen Aufgaben:

- Datenfluss (engl. *data flow*): In dieser Ebene werden die Multimedia-Daten verarbeitet und übertragen.
- Steuerung (engl. *control*): Diese Ebene sorgt für die Steuerung der Datenfluss-Ebene und den Austausch von Steuer- und Dienstgüten-Information.
- Dienstgüten-Steuerung (engl. *QoS-control*): Hier werden die Dienstgüte der Anwendung überwacht und die Adaptionsentscheidungen getroffen. Es erfolgt eine übergeordnete Steuerung durch eine *Meta-Policy* (s. 3.5.1.1, S. 46).

Für die Nutzer-Schicht existieren in der Datenfluss-Ebene keine Module, jedoch wird mittels der Steuerungs-Ebene ein Zielwert in der Dienstgüten-Steuerung festgelegt. Für die eigentliche Übertragung werden die Medien-Daten zunächst in der Anwendungs-Schicht des Senders erfasst, d. h.

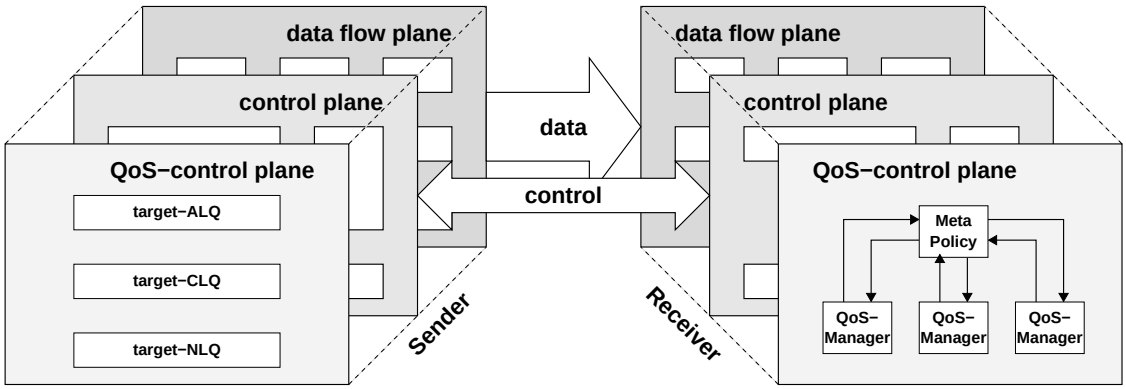


Abbildung 4.1: Struktur der EXPERIMENTATION PLATFORM

die analogen Eingangssignale (Video oder Audio) werden digitalisiert und stehen danach zur Bearbeitung zur Verfügung. Kodierungs- und/oder Komprimierungsmodule erzeugen eine Folge von Rahmen, die von der Kommunikations-Schicht für die Übertragung in Pakete aufgeteilt werden. In der Netz-Schicht erfolgt schließlich die Übertragung abhängig von der verwendeten Technologie. Tabelle 4.1 fasst die Aufgaben der einzelnen Schichten der Datenfluss-Ebene zusammen.

Schicht	Aufgabe
User	(Keine Module in der Datenfluss-Ebene)
Application	Erfassung der Medien-Daten und Kodierung in Rahmen
Communication	Aufteilung der Rahmen auf Pakete
Network	Paket-Übertragung

Tabelle 4.1: Aufgaben der Schichten der Datenfluss-Ebene im Sender

Die Multimedia-Daten werden durch das Netz paketweise vom Sender zum Empfänger transportiert (bei bidirektionaler Kommunikation gilt Analoges in Rückrichtung). Im Empfänger werden die Pakete von einem Modul der Netz-Schicht entgegen genommen und an die Kommunikations-Schicht weiter geleitet. Diese Schicht stellt die Rahmen wieder her, so dass Module der Anwendungs-Schicht die Medien-Daten darstellen (d. h. Videobilder anzeigen oder Audio-Daten ausspielen) können. Außerdem wird in der Anwendungs-Schicht aus den gemessenen Medien-Parametern eine Schätzung der vom Nutzer erfahrenen Dienstgüte des Medienstroms berechnet und in der Nutzer-Schicht für die Ermittlung der Gesamt-Dienstgüte der Anwendung verwandt. Tabelle 4.2 führt die Aufgaben der einzelnen Schichten der Datenfluss-Ebene noch einmal auf.

Die Durchführung der Adaption erfolgt in der Datenfluss-Ebene, über die Steuerungs-Ebene veranlasst von einem *QoS-Manager*–Modul. Während der Übertragung überwachen die Module der Datenfluss-Ebene den Medienstrom und berechnen Zustandsparameter. Deren Werte teilen sie dem zugehörigen QoS-Manager–Modul mit, das dann die momentan erreichte Dienstgüte (engl. *achieved QoS*) errechnet (Abb. 4.2). Falls notwendig bestimmt der Manager neue Werte für die Steuer-Parameter der Datenfluss-Module und überträgt sie als neue Ziel-Dienstgüte (engl. *target-QoS*). Die Module stellen sich dann auf diese neuen Vorgaben ein.

Schicht	Aufgabe
User	(Keine Module in der Datenfluss-Ebene)
Application	Dekodierung der Rahmen und Darstellung der Medien-Daten
Communication	Rekonstruktion der Rahmen aus Paketen
Network	Paketempfang

Tabelle 4.2: Aufgaben der Schichten der Datenfluss-Ebene im Empfänger

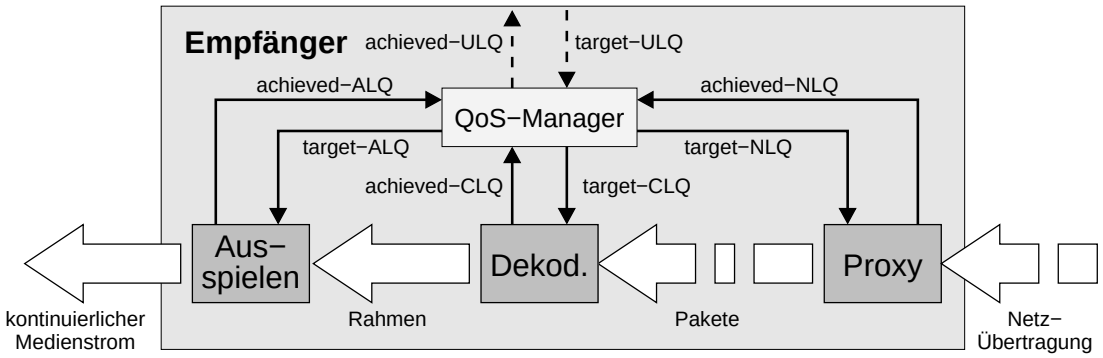


Abbildung 4.2: Medienstrom mit QoS-Manager– und Daten–Modulen

Ist es nicht ausreichend, auf der Ebene eines Medienstroms eine Anpassung durchzuführen, so kann die *Meta-Policy* (s. 3.5.1.1, S. 46) eingreifen und die relative Gewichtung der Medienströme ändern. Die QoS-Manager–Module der Ströme ändern daraufhin die Ziel-Parameter, um die Gesamtdienstgüte zu verbessern.

4.2.2 Dienstgütenkonzept

In der EXPERIMENTATION PLATFORM gibt es folgende wesentlichen Dienstgütenkonzepte. Zum einen wird zwischen *Ziel-* und *erreichter* Dienstgüte unterschieden, zum anderen werden die Dienstgüten-Parameter nach Schichten gruppiert (Tab. 4.3). Die Ziel-Parameter dienen zur Steuerung der Module, indem diese versuchen, die Vorgaben zu erfüllen. Die Festlegung bestimmter Parameterbezeichner definiert eine Schnittstelle, die die Implementierung der Module unabhängig von der Steuerung macht. Die andere Parametergruppe beschreibt den momentanen Zustand aus Sicht eines Moduls, wobei der QoS-Manager durch Betrachtung der Zustände der einzelnen Module den Überblick über die Dienstgüte des Medienstroms erhält und Anpassungsmaßnahmen einleiten kann.

Bevor die Multimedia-Kommunikation beginnt, muss der Ausgangszustand festgelegt werden, d. h. die Steuerparameter erhalten Startwerte zugewiesen. Dies erfolgt durch eine stufenweise Abbildung: Ausgehend von der erwarteten Dienstgüte in der Nutzer-Schicht (target-ULQ) und der Art der Anwendung (Szenario) wird die Ziel-Dienstgüte der Medienströme (target-ALQ) festgelegt. Hieraus folgt eine Beschreibung des Verkehrs, den die Medienströme erzeugen werden (target-CLQ) und als letzter Schritt im Sender die Parameter des Verkehrs, der vom Netz übertragen werden soll (target-NLQ). Die Abbildung wird dabei jeweils im Dienstgüten-Steuerungs–Teil der

	Schicht	Funktion
ULQ	*User Level*	Beschreibung der Erwartung und der Zufriedenheit des Nutzers mit der Anwendung
ALQ	*Application Level*	Dienstgüte und Beschreibung der Medienströme
CLQ	*Communication Level*	Beschreibung der Datenströme
NLQ	*Network Level*	Zustand der Übertragung

Tabelle 4.3: Dienstgütenparameter der Schichten

an der Übertragung beteiligten Module durchgeführt, da sie von der jeweiligen Implementierung der Funktion abhängt.

Es werden Steuer-Parameter mit vereinbarten Bezeichnern in den Modulen und QoS-Managern verwandt. So gibt es beispielsweise den Parameter target(CLQ/BufferDelay), dessen Wert d_{buff} vom Modul XP_CAudioReceiveBuffer zur Berechnung des Übergabezeitpunkts $t_{forward} = t_{gen} + d_{buff}$ eines Audio-Rahmens mit der Erzeugungszeit t_{gen} an das Audio-Interface verwandt wird. Hierdurch lässt sich ein Übertragungs-Jitter ausgleichen. Die Steuerung kann um zusätzliche Parameter erweitert werden, wodurch beliebige Regelungsmechanismen realisierbar sind.

ULQ	Bedeutung
10	beste Qualität
9	ausgezeichnete Qualität
8	sehr gute Qualität
7	gute Qualität
6	befriedigende Qualität
5	ausreichende Qualität
4	nutzbare Qualität
3	noch nutzbare Qualität
2	eingeschränkte Nutzbarkeit
1	sehr eingeschränkte Nutzbarkeit
0	Medium nicht nutzbar

Tabelle 4.4: Bedeutung der ULQ-Werte

Audio-Parameter	Zielwert
SamplingFequency	16925 Hz
AudioDataFormat	ADPCM
FrameLength	40 ms[1]
Delay	40 ms[1]

Video-Parameter	Zielwert
FramesPerSecond	14 fps
CodingFormat	RT-JPEG[2]
Height	288 pixel
Width	384 pixel
Delay	40 ms

Tabelle 4.5: Zuordnung ULQ → ALQ am Beispiel ULQ = 7

[1] Beide Zielwerte könnten nur bei Fehlen jeglicher Verzögerung erreicht werden.

[2] „Real-time JPEG“ [Sch98]

Beispiel

Es werde eine hochqualitative, interaktive Audio- und Videoübertragung betrachtet. Der Nutzer erwarte eine „gute Qualität“, was nach Tab. 4.4 einer Ziel-Dienstgüte von ULQ=7 entspricht. Die *Meta-Policy* leite hieraus eine Gewichtung der einzelnen Medienströme ab und weise den beiden beteiligten Medienströmen die Ziel-Dienstgütenwerte targetULQ(Audio):=7 und targetULQ(Video):=7 zu.

Für jeden Medienstrom existiert ein QoS-Manager, dessen Aufgabe es ist, die ULQ-Vorgabe zu erreichen. Er setzt die ULQ-Vorgabe in die Medien-Parameter um, die dieser Dienstgüte entsprechen, hier z. B. die Parameter nach Tab. 4.5.

Nach Beginn der eigentlichen Übertragung beobachten die Module den Datenstrom, den sie durch die Endgeräte transportieren. So speichert das Bild-Kodierungsmodul die von den kodierten Bildern benötigte Bitrate im Parameter Video.achieved(ALQ/Bandwidth). Regelmäßig wird im QoS-Manager des Mediums m die momentan erbrachte Dienstgüte $Q(m)$ ermittelt (s. 3.3.1, S. 41), indem für jeden betrachteten ALQ-Parameter p dessen Dienstgüte $Q_m(p) \in [0,10]$ und mittels Gl. 4.1 $Q(m)$ bestimmt wird[3].

$$Q(m) = \sum_{w_p} w_p \cdot Q_m(p) \quad \text{mit} \quad \sum_{w_p} w_p = 1 \tag{4.1}$$

Nachdem für jedes beteiligte Medium $m \in M$ sein Dienstgüten-Wert $Q(m)$ bestimmt wurde, berechnet die *Meta-Policy* hieraus die momentan erreichte Gesamtdienstgüte Q der Anwendung. Weicht diese von der erwarteten Dienstgüte ab, so hat die *Meta-Policy* die Möglichkeit, die Gewichtung der Medien zu ändern (d. h. die ULQ-Vorgaben an die QoS-Manager) oder gar konkrete Parameter der Übertragung zu beeinflussen, z. B. die Bitratenaufteilung zwischen Medienströmen.

Die neuen Vorgaben werden wieder von den beteiligten Modulen schrittweise verarbeitet und die eigentliche Übertragung angepasst. Im obigen Beispiel stelle die Anwendung fest, dass die Übertragungsbitrate für die angestrebte Dienstgüte nicht ausreicht und dies zu einer schlechten Audio-Übertragung führt, d. h. die Zahl der „Aussetzer“ durch den Verlust von Paketen mit Audio-Daten überschreitet einen bestimmten Wert. Die *Meta-Policy* entscheidet nun, dass die Audio-Übertragung bevorzugt behandelt wird und reduziert die Zielvorgabe für die Video-Übertragung. Hierdurch sinkt der Bitratenbedarf für den Video-Strom und der Audio-Strom erfährt weniger Verluste. Für genauere Beschreibungen der Adaption sei auf 4.3.4, S. 76 für das Audio-Medium, 4.4.4, S. 86 für das Video-Medium und 4.6, S. 98 für die medienübergreifende Anpassung verwiesen.

4.2.3 Datenflusskonzept

Für jeden Medienstrom wird im Sender ein „Kanal“ erzeugt, der Daten vom Nutzer durch das Terminal dem Netz zuführt. Der Kanal instanziiert die Module, die für die Bearbeitung des Datenstroms zusammengeschaltet werden und die ALQ-Parameter auf implementierungsabhängige Parameter abbilden. Dem Kanal im Sender entsprechend wird auch im Empfänger ein Kanal zur Dekodierung und Anzeige des Mediums aufgebaut. Die erwarteten CLQ- und NLQ-Parameter werden schrittweise durch die konkreten Module des Kanals bestimmt, da diese Werte von Implementierung und Konfiguration abhängen. Zu Beginn der Übertragung liegt dann eine Abschätzung der erwarteten Werte vor.

[3] Die Berechnung der Medien-Dienstgüte kann als beliebige Funktion implementiert werden.

Die Module im Datenfluss können Quellen (engl. *sources*), Senken (engl. *sinks*) oder beides sein. Es können nur Quellen und Senken zusammengeschaltet werden, die denselben Datentyp übergeben bzw. annehmen, also z. B. Audio- oder Video-Rahmen (s. B.2, S. 147). Konvertierungen zwischen verschiedenen Typen erfolgen nur in Modulen, z. B. zerteilt das Segmentierungs-Modul XP_CSegmentation große Rahmen, um sie als Pakete mit einer vom Netz abhängigen Maximalgröße verschicken zu können.

In jedem Sender-Endgerät existiert ein Stellvertreter (engl. *Proxy*) des Empfängers von der Klasse XP_CMediumReceiverProxy, der die eigentliche Übertragung über das Netz kapselt. Je nach verfügbarer Netz-Technologie wird ein spezialisiertes Modul instanziiert, für IP XP_CMedRecProxyIP, für ATM XP_CMedRecProxyATM. Analog wird im Empfänger ein Stellvertreter XP_CMediumTransmitterProxy des Senders erzeugt, bzw. spezialisierte Ableitungen wie XP_CMedTransProxyIP.

Die Erzeugung und Verarbeitung der Daten in den Kanälen wird von einem oder mehreren „ *Threads*“ (Kontrollflüsse, s. B.3, S. 148) durchgeführt, die über gemeinsamen Speicher Daten austauschen. Die Zugriffe werden dabei durch Semaphoren (CMutex, mut*ual*-ex*clusion device*) synchronisiert.

Die Steuer- und Zustandsparameter werden über die *QoS-control*–Ebene ausgetauscht. Bei der Erzeugung des Kanals werden die beteiligten Module registriert, so dass später ihre Funktionen zur Abfrage ihres momentanen Zustands (MakeAbstraction im Empfänger, MediumTransmissionState im Sender) aufgerufen werden können. Diese Funktionen übergeben Parameter-Mengen (XP_CModeSet und XP_CQuality) an den QoS-Manager des Mediums, der ggf. die Adaptionsentscheidung trifft und den Ziel-Parametern neue Werte zuweist. Über die Funktionen MakeSelection (im Empfänger) und ChangeModuleParameters (im Sender) werden diese den Modulen mitgeteilt und sie versuchen, die neuen Zielvorgaben zu erreichen.

4.2.4 Einschränkungen

Bei der Implementierung der EXPERIMENTATION PLATFORM wurden einige, für allgemeine Multimedia-Anwendungen wünschenswerte oder sinnvolle Eigenschaften nicht implementiert, da sie für die Untersuchung der Anpassung in der beabsichtigten Weise nicht erforderlich sind.

Ressourcen-Management: Es findet keine explizite Verwaltung der Ressourcen wie Rechenleistung, Speicher oder Peripherie statt. Durch das präemptive Multitasking des Linux-Systems und kurze Zeitscheiben (2.5 ms), sowie den Verzicht auf parallel laufende Anwendungen scheint dies nicht nötig zu sein.
Ebenso wird auf eine dynamische Aushandlung der Endgeräte-Fähigkeiten beim Aufbau der Verbindung verzichtet, da dies über eine Einschränkung der verfügbaren Kodierungen nachbildbar ist.
Ebenso wurden bestimmte Annahmen über das Kommunikationsnetz getroffen, es wird z. B. von paketvermittelnden Netzen ausgegangen und angenommen, dass Bitfehler in Paketen zu ihrem Verwurf in der Netz-Schicht führen (vgl. UDP, RFC 1122/4.1.3.4).

Uhrensynchronisation: Die Berechnung der Ende-zu-Ende–Verzögerung erfordert eine Synchronisierung der Uhren in den Endgeräten. Hierzu wurde NTP (RFC 1305, [Mil92]) eingesetzt, das eine Genauigkeit im Millisekunden-Bereich erlaubt. Dies ist für eine Multimedia-Anwendung ausreichend.

Verzicht auf Echounterdrückung: Für bidirektionale Audio-Übertragungen wird normalerweise Echounterdrückung (Hard- oder Software) eingesetzt, um zu verhindern, dass der Empfänger das Signal des Senders aufnimmt und zum Sender zurück schickt. In diesem experimentellen System ist diese Funktion nicht erforderlich.

Verzicht auf CSCW: Eine weitere wichtige verteilte, interaktive Anwendung ist CSCW (s. 2.2.5, S. 21). Da die dort verwendeten Medientypen sehr viel größere Verzögerungen tolerieren können ([Ste96]: 500 bis 1250 ms), ist Adaption nicht im selben Maße erforderlich wie bei Audio- und Video-Übertragungen. Zudem wären für die Entwicklung einer Bewertungsfunktion für die Dienstgüte auf Nutzer-Schicht zusätzliche Untersuchungen mit einer großen Anzahl an Testpersonen notwendig.

Verzicht auf Multicast: Obwohl die eXPERIMENTATION PLATFORM mit dem Ziel entworfen wurde, Multimedia-Kommunikation zwischen mehr als zwei Endgeräten zu unterstützen, wird dies hier nicht betrachtet. Es zeigte sich in der Praxis, dass *interaktive* Multicast-Szenarien (z. B. Videokonferenzen mit vielen Teilnehmern) relativ selten auftreten (hingegen sind nicht-interaktive Multicast-Anwendungen immer häufiger anzutreffen), bzw. die Art dieser Anwendungen stark von der hier untersuchten abweicht (z. B. Spiele wie „Quake“, s. 2.2.6.3, S. 24). Zudem wird erwartet, dass sich die hier gefundenen Ergebnisse im Prinzip auf Multicast-Anwendungen übertragen lassen.

Bewertung der Medien-Dienstgüte: Die Bewertung der Dienstgüte erfolgt über Tabellen, die Zustandsparametern einen Qualitätswert zuordnen, und über Funktionen, in denen die Werte für mehrere Parameter in eine gewichtete Summe eingehen (s. 4.3, S. 70). Die Festlegung dieser Werte erfolgte anhand von Literatur und subjektive Vergleiche, jedoch ohne groß angelegte Feldversuche mit vielen Teilnehmern und wissenschaftlicher Auswertung. Dies hätte den Rahmen dieser Untersuchung gesprengt.

Optimierung: Während der Entwicklungszeit der eXPERIMENTATION PLATFORM wurden in der Literatur verschiedene Ansätze zur Optimierung einer Multimedia-Kommunikation vorgestellt, die nicht alle berücksichtigt wurden. Zum Beispiel gibt es einen Vorschlag zur Kompression der Nachrichtenköpfe in RTP-Strömen und optimierte Kodierer für Audio- und Video-Übertragungen, z. B. MPEG.

Keine explizite Betrachtung des Signalisierverkehrs: Die Steuerung des Senders und der Austausch von Zustandsinformation erfolgt über eine TCP-Verbindung und einzelnen UDP-Paketen zwischen Sender und Empfänger. Da dieser Verkehr sehr klein ist (unter 1 kbit/s), wird er nicht berücksichtigt.

4.3 Beschreibung des Audio-Mediums

Dieser Abschnitt stellt die Implementierung der Audio-Übertragung vor und beschreibt den erzeugten Verkehr auf der Netz-Ebene. Es wird erläutert, wie die Dienstgüte anhand der Zustandsparameter bewertet wird und wie die Steuerparameter die Übertragung beeinflussen. Schließlich werden die Anpassungsmechanismen und Adaptionsalgorithmen des Audio-Mediums beschrieben.

4.3.1 Implementierung

Bei der Audio-Übertragung wird das analoge Tonsignal von der Sound-Karte digitalisiert und über ein Geräte-Interface (XP_CAudioInterfaceModule) der Anwendung zur Verfügung gestellt. Die Digitalisierung kann über die Einstellung der Abtastfrequenz, die Kodierung der Abtastwerte und der Zahl der Kanäle (Mono oder Stereo) beeinflusst werden. Da die Übergabe der Audio-Daten blockweise erfolgt, kann über den Parameter ALQ/FrameLength (d_F) die Dauer eines Audio-Rahmens und somit die Mindest-Verzögerung eingestellt werden — der erste Abtastwert kann erst versandt werden, wenn der Rahmen gefüllt ist. Die Länge des Rahmens l_F in bit hängt von der verwandten Kodierung ab (Abb. 4.3).

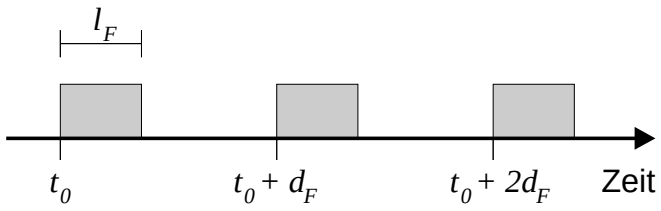


Abbildung 4.3: Generierung von Audio-Rahmen im Sender

Die Audio-Rahmen werden im Audio-Interface XP_CAudioInterfaceModule erzeugt und an ein Modul zur „Weiterleitung“ XP_CAudioSendForward gereicht (Abb. 4.4), das sie mit einer Sequenznummer versieht und als „Vektor-Listen“-Puffer XP_CVectorList interpretiert, so dass sie vom Empfänger-Stellvertreter (*Proxy*) XP_CMedRecProxyIP akzeptiert werden. Danach erfolgt die Übertragung zum Empfänger, wo die Pakete vom Sender-Proxy XP_CMedTransProxyIP an das Pufferungs-Modul XP_CAudioReceiveBuffer übergeben werden, das sie wieder zu Audio-Rahmen konvertiert. Im Pufferungs-Modul werden die Rahmen zwischengespeichert, sortiert nach ihren Erzeugungszeitpunkten, bis sie nach Ablauf der eingestellten Verzögerung target(CLQ/BufferDelay) (gemessen ab dem Erzeugungszeitpunkt) an das Interface-Modul im Empfänger übergeben werden. Hierdurch können Übertragungs-Jitter und auch Änderungen in der Paketreihenfolge ausgeglichen werden. Das Interface-Modul konvertiert die Audio-Rahmen in ein Format, das die Sound-Karte ausspielen kann und schreibt die Daten in den Puffer der Sound-Karte. Von dessen momentanen Füllstand hängt dann der tatsächliche Ausspielzeitpunkt ab, der dann in die Berechnung von $\bar{d}$ =achieved(ALQ/Delay) einfließt. Der Unterschied zwischen der Pufferung auf der Sound-Karte und im Pufferungs-Modul ist, dass sich in letzterem die Verzögerungszeit genau einstellen lässt und dass die Reihenfolge der Audio-Daten geändert werden kann.

Jeder Audio-Rahmen führt einen Typ-Identifikator mit (entspricht dem Parameter ALQ/AudioType), durch den der Empfänger erkennen kann, ob der Sender die Kodierung geändert hat. Der Empfänger kann dann auf das neue Format umschalten, was in dieser Implementierung keine Störungen zur Folge hat. Abb. 4.5 zeigt die Kodierung eines Audio-Typs $k \in K_{au}$ (Tab. B.2, S. 152 im Anhang gibt Beispiele für verschiedene Typen). Die unteren fünf Bit kodieren entweder die verwendete Frequenz oder bezeichnen für das Format „CODEC“ einen bestimmten Kodierer. Dabei sind nicht alle Werte für den Typ möglich. So können beispielsweise ADPCM-Typen (*Adaptive Differential Pulse Code Modulation*, 4 bit pro Abtastwert) nicht im Stereo-Modus verwandt werden.

Als Besonderheit wurden die FEC-Typen (*Forward Error Correction*) FEC0 bis FEC7 definiert, die in einem Paket zwei Audio-Rahmen der Typen $\{k_0, k_1\}$ tragen (s. 3.5.2.5, S. 52). Der Ver-

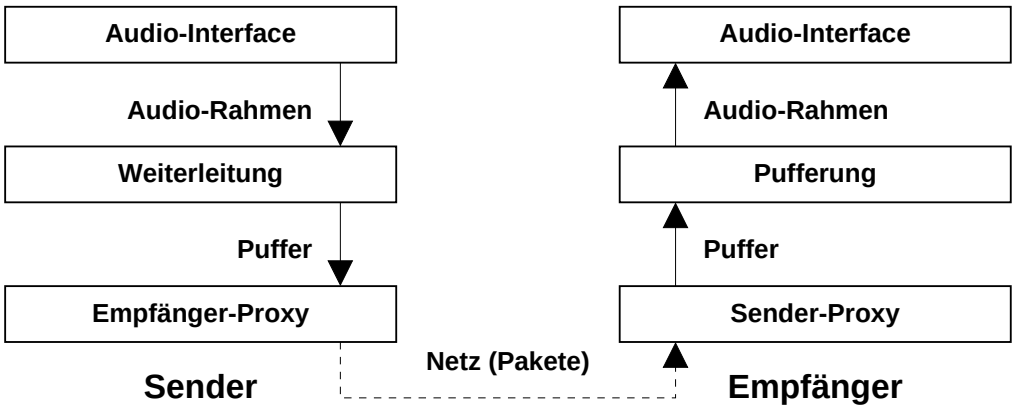


Abbildung 4.4: Aufbau der Audio-Übertragung

satz v_1 der Rahmen vom Typ k_1 gegenüber denen mit dem Typ k_0 wird durch den Parameter ALQ/RedundancyOffset bestimmt.

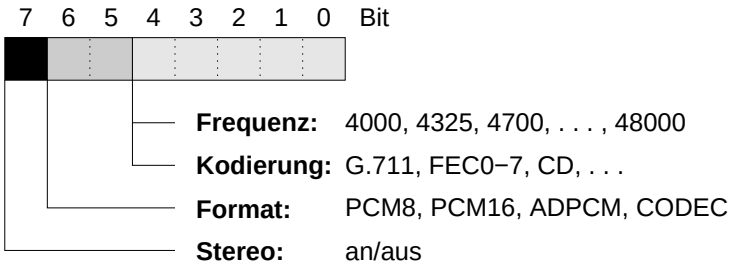


Abbildung 4.5: Kodierung des Audio-Typs

In die Berechnung der vom Nutzer erfahrenen Dienstgüte für die Audio-Übertragung (Gl. 4.2) geht die Qualität $Q(k)$ des Audio-Typs $k \in K_{au}$ nach Gl. 4.3 und die Bewertung $Q(\bar{d})$ der mittleren Ende-zu-Ende–Verzögerung ein. Hier hat die Menge der aufgetretenen Ausspielpausen einen großen Einfluss, dergestalt dass die entsprechende Qualität Q_P (abhängig von der Länge und Anzahl der Pausen) die gesamte Dienstgüte begrenzt. Für die verwandten Werte und Funktionen sei auf Tabellen im Anhang verwiesen (s. B.8.2, S. 151), die ausgehend von der Literatur durch subjektive Tests bestimmt wurden.

$$Q = \min(w_{type} \cdot Q(k) + w_{delay} \cdot Q(\bar{d}), Q_P) \quad (4.2)$$

$$Q(k) = w_{freq} \cdot Q(Freq(k)) + w_{format} \cdot Q(Format(k)) + w_{stereo} \cdot Q(Stereo(k)) \quad (4.3)$$

Für alle implementierten Audio-Typen $k \in K_{au}$ kann die Bitrate $B(k)$ und die ULQ (*User Level Quality*) $Q(k)$ bestimmt werden. Dabei fällt auf, dass bestimmte Typen bei höherer Rate eine niedrigere ULQ besitzen als ein anderer Typ – offensichtlich ist dann der Einsatz dieses Typs nicht sinnvoll. Deshalb wird bei der Initialisierung der Audio-Übertragung die Liste K^*_{au} der effizienten

Audio-Typen berechnet (Gl. 4.4). Die Ausnahme sind dabei die FEC-Typen, die natürlich eine höhere Bitrate benötigen, aber dennoch in bestimmten Situationen eingesetzt werden.

$$K^*_{au} = \left\{ k \in K_{au} \mid \nexists k' \in K_{au} : Q(k') \geq Q(k) \wedge B(k') < B(k) \right\} \tag{4.4}$$

Anmerkung

Nicht alle Audio-Formate werden direkt von der Sound-Karte unterstützt, daher sind im Modul XP_CAudioFrame Funktionen implementiert, die die Audio-Daten konvertieren können, z. B. von PCM16 nach ADPCM und zurück. Außerdem können nicht alle Rahmenlängen verwandt werden, da die Sound-Karte sog. „Fragmente“ aus den Abtastwerten bildet, die aus Effizienzgründen mindestens 16 byte lang sein müssen.

4.3.2 Steuer- und Zustandsparameter

Die Module der Audio-Übertragung werden durch die Parameter nach Tab. 4.6 gesteuert. Die erfassten Zustandsparameter sind in Tab. 4.7 aufgeführt. Die Notation erfolgt in der Form „Schicht/Bezeichner“, z. B. handelt es sich bei ALQ/Delay um den Verzögerungs-Parameter mit dem Bezeichner Delay in der Anwendungs-Schicht des Mediums.

Modul: Parameter	in	Beschreibung
Interface:		
ALQ/AudioType	–	Art der Audio-Daten (Kodierung $\underline{k}$)
ALQ/Delay	s	Ende-zu-Ende–Zielverzögerung d_T
ALQ/DropProbability	1	Wahrscheinlichkeit p_{Fdrop}, einen Rahmen zu verwerfen
ALQ/FrameLength	s	gewünschte Länge $d_{F,T}$ eines Audio-Rahmens
ALQ/RedundancyOffset	1	Versatz v_1 zum Redundanz-Rahmen
ALQ/RepeatProbability	1	Wahrscheinlichkeit $p_{Frepeat}$, einen Rahmen mehrfach auszuspielen
Puffer:		
CLQ/BufferDelay	s	Zielverzögerung nach Pufferung d_{Tbuff}
Proxy:		
NLQ/Bandwidth	bit/s	Bitrate B_T des gesandten Verkehrs

Tabelle 4.6: Steuer-/Zielparameter der Audio-Übertragung

4.3.3 Audio-Anpassungsmechanismen

Die folgenden Mechanismen wurden für die Audio-Übertragung implementiert. Es wird jeweils erst der beobachtete Effekt erläutert und die verantwortliche Ursache erklärt. Danach wird dargestellt, wie der Adaptionsalgorithmus diesen Fall erkennen kann und welche Reaktion erfolgt.

Modul: Parameter	in	Beschreibung
Interface:		
ALQ/AudioDataFormat	–	Format der Abtastwerte
ALQ/Delay	s	erfahrene Ende-zu-Ende–Verzögerung $\bar{d}_A$
ALQ/DSPStereo	1, 0	Stereo ja/nein
ALQ/FrameLength	s	tatsächliche Länge d_F eines Audio-Rahmens
ALQ/LostData	1	Anteil der endgültig verlorenen Rahmen p_{Floss}
ALQ/PauseCount	#	Anzahl der Ausspielpausen
ALQ/PauseLength	μs	Länge der Ausspielpausen
ALQ/SamplingFrequency	Hz	Abtastfrequenz f
Puffer:		
CLQ/BufferDelay	s	Verzögerung d_{buff} der Audio-Daten *nach* Pufferung
Proxy:		
NLQ/Bandwidth	bit/s	Bitrate B_A des empfangenen Verkehrs
NLQ/Delay	s	Verzögerung der Audio-Daten *vor* Pufferung
NLQ/Jitter	s	Verzögerungsschwankung der Pakete
NLQ/PacketLossProbability	s	Wahrscheinlichkeit des Verlust eines Pakets
NLQ/PacketReorderings	1	Anteil der Paketumordnungen pro Sekunde

Tabelle 4.7: Zustandsparameter der Audio-Übertragung

4.3.3.1 Anpassung der Puffer-Verzögerung

Ändert sich die beobachtete Netz-Verzögerung (bestehend aus mittlerer Verzögerung und Verzögerungsschwankung), so kann es vorkommen, dass Rahmen verworfen werden, weil sie die Verzögerungsvorgabe nicht einhalten bzw. Rahmen länger als nötig verzögert werden.

Reihenfolgeänderungen der Pakete durch das Netz können zudem nur ausgeglichen werden, solange die Pakete noch nicht an die Sound-Karte übergeben wurden. Daher muss bei einer hohen Zahl an Umordnungen die Pufferungszeit erhöht werden. Ebenso muss beim Einsatz redundanter Audio-Übertragung die Pufferverzögerung hoch genug sein, um es dem Ersatzrahmen zu erlauben, den Empfänger vor dem Ausspielzeitpunkt eines verlorenen Rahmens zu erreichen.

Ursache: Die Pakete der Audio-Übertragung treffen auf unterschiedlich hoch gefüllte Puffer im Netz, wodurch sich ihre Verzögerung ändert. Das Netz verursacht Änderungen in der Paketreihenfolge. Es wird ein FEC-Audio-Typ eingesetzt.

Erkennung: Die gemessene Netz-Verzögerung achieved(NLQ/Delay) und ihre Schwankung achieved(NLQ/Jitter) ändern sich. Die Zahl der Paketumordnungen achieved(NLQ/PacketReorderings) ist größer als Null. Der Redundanz-Versatz target(ALQ/RedundancyOffset) ist größer Null.

Reaktionen: Die Ziel-Pufferverzögerung wird erhöht, so dass weniger spät ankommende Pakete verworfen werden müssen, bzw. erniedrigt, um die Ende-zu-Ende–Verzögerung zu reduzieren.

Siehe: Der Mechanismus ähnelt 3.5.2.2, S. 49.

4.3.3.2 Anpassung an Erhöhung der verfügbaren Netz-Bitrate

Die Paketverlustwahrscheinlichkeit ist sehr gering (hier unter 1 %) und die empfangene Bitrate entspricht der gesandten. Es kann getestet werden, ob eine höhere Bitrate verfügbar ist und eine frühere Reduzierung der Audio-Qualität zurückgenommen werden kann.

Ursache: Es steht mehr Bitrate zur Verfügung als momentan benötigt wird, bzw. ein Effekt, der die Bitrate im Netz reduzierte (z. B. *Querverkehr*), tritt nicht mehr auf.

Erkennung: Die Paketverlustwahrscheinlichkeit achieved(NLQ/PacketLossProbability) ist kleiner als γ_L und die erreichte Netz-Bitrate achieved (NLQ/Bandwidth) entspricht der gesandten Rate target(NLQ/Bandwidth).

Reaktionen: Es wird auf den nächstbesseren Audio-Typ umgeschaltet, jedoch wird die Ziel-Dienstgüte nicht überschritten.

Siehe: 3.5.2.4, S. 51

4.3.3.3 Anpassung an geringe verfügbare Netz-Bitrate

Die für die Audio-Übertragung verfügbare Bitrate im Netz ist geringer, als die Senderate. Hierdurch kommt es zu Paketverlusten.

Ursache: Im Netz laufen Puffer über und Pakete werden verworfen, da die Zuflussrate die Abflussrate übersteigt. Oder die Übertragung der Pakete dauert zu lange, so dass der Ausspielpuffer leer läuft.

Erkennung: Die beobachtete Paketverlustwahrscheinlichkeit p_{Ploss} ist größer als γ_H. Oder die Pausenlänge achieved(ALQ/PauseLength) ist größer als γ_P und die empfangene Rate B_A ist kleiner als die Senderate B_T. (Da das Netz eine bestimmte Anzahl an Paketen speichern kann, kommt es vor, dass keine Paketverluste auftreten, jedoch dennoch die Senderate zu hoch ist. Dies wird durch den Vergleich mit der empfangenen Rate erkannt.)

Reaktionen: Es wird ein neuer Audio-Typ k^* gewählt, dessen Bitrate $B(k^*)$ die beobachtete Netz-Bitrate B_A nicht überschreitet.
Wurde versucht, einen besseren Audio-Typ zu verwenden, so wird dies zurückgenommen und der vorherige Typ eingesetzt. Der zeitliche Abstand zwischen Verbesserungsversuchen wird verdoppelt (bis zu einem Maximalwert), da die momentan maximal erreichbare Übertragungsrate erreicht wurde und Verbesserungsversuche zu einer unnötigen Beeinträchtigung der erfahrenen Dienstgüte führen.

Siehe: 3.5.2.4, S. 51

4.3.3.4 Auswahl des besten FEC-Audio-Typs

Werden redundante Audio-Daten übertragen, so wird derjenige FEC-Typ verwandt, der die verfügbare Rate am besten nutzt.

Ursache: Die Übertragungsstrecke ist zwar verlustbehaftet, bietet aber dennoch eine bestimmte Bitrate an. Hierfür soll der beste FEC-Typ ausgewählt werden.

Erkennung: Der Anteil achieved(ALQ/LostData) der Rahmen, die nicht durch die redundante Kodierung geschützt werden konnte, ist kleiner als ε_L (größer als ε_H).

Reaktionen: Es wird der nächstbessere FEC-Audio-Typ ausgewählt. (Es wird der nächstschlechtere FEC-Audio-Typ ausgewählt, bzw. es wird eine frühere Verbesserung zurückgenommen und der Abstand zwischen Verbesserungsversuchen verdoppelt.)

Anmerkung: Der wesentliche Unterschied zu den Mechanismen 4.3.3.2 und 4.3.3.3 ist die Verwendung des Parameters ALQ/LostData im Gegensatz zum Parameter NLQ/PacketLossProbability.

4.3.3.5 Steuerung des Pufferfüllstands der Sound-Karte

Der Puffer auf der Sound-Karte muss überwacht werden, denn ein niedriger Füllstand kann zu Ausspielpausen (störendes Knacken) und ein zu hoher Füllstand zu erhöhter Ende-zu-Ende–Verzögerung führen, was die Interaktivität stört.

Ursache: Die Zuflussrate und die Ausspielrate des Sound-Karten-Puffers stimmen nicht überein.

Erkennung: Die erreichte Pufferverzögerung $d_{\underline{buff}}$ = achieved(CLQ/BufferDelay) und die Ende-zu-Ende–Verzögerung $\bar{d}$ = achieved(ALQ/Delay) unterscheiden sich um weniger als $d_{b,min}$ bzw. mehr als $d_{b,max}$.

Reaktionen: Ist der Füllstand zu klein, so wird die Ausspielwiederholwahrscheinlichkeit ALQ/RepeatProbability auf den Wert $p_{Frepeat}$ gesetzt. Ist der Füllstand zu groß, so erhält die Verwurfswahrscheinlichkeit ALQ/DropProbability den Wert p_{Fdrop}. Dabei ist die Dauer $d_{observe}$ eines Beobachtungsintervalls sehr viel größer als die Verzögerungen (einige Sekunden im Vergleich zu wenigen Millisekunden).

$$p_{Frepeat} := \frac{d_{b,min}-(\bar{d}-d_{buff})}{d_{observe}} \quad \text{für} \quad \bar{d}-d_{buff} < d_{b,min}$$

$$p_{Fdrop} := \frac{\bar{d}-d_{buff}-d_{b,max}}{d_{observe}} \quad \text{für} \quad \bar{d}-d_{buff} > d_{b,max}$$

Siehe: 3.5.3.6, S. 61

4.3.4 Audio-Adaptionsalgorithmen

Der Haupt-Algorithmus (Abb. 4.6) startet mit dem vom QoS-Manager vorgegebenen Audio-Typ (abgeleitet aus der Nutzer-Vorgabe) und beobachtet die Übertragung. Er lässt zunächst vom Mechanismus „FEC“ die Rate ermitteln, die das Netz zu übertragen im Stande ist. Da dieser Mechanismus sowohl bei zufälligen Verlusten, als auch bei einer Bitratenbegrenzung prinzipiell funktioniert, kann nach kurzer Zeit entschieden werden, welche der beiden Situationen vorliegt. Treten keine Paketverluste mehr auf (Bedingung [keine Verluste]), so wird der Mechanismus „Bitrate“ aktiviert, der Audio-Typen ohne Redundanz einsetzt, die bei gleicher Bitrate eine deutlich höhere Audio-Qualität als FEC-Typen liefern. Ändert sich das Verhalten des Netzes und treten wieder Paketverluste auf ([Verluste], zufällig oder weil die verfügbare Bitrate reduziert wurde), so wird auf den FEC-Mechanismus zurückgeschaltet. Reduziert sich jedoch die Paketverlustrate, so wird noch nicht umgeschaltet, um dem Bitraten-Mechanismus Gelegenheit zu geben, einen passenden Audio-Typ zu finden.

Gleichzeitig wird parallel der Füllstand des Puffers auf der Sound-Karte geregelt (s. 4.3.3.5, S. 76). Läuft der Puffer leer ([zu wenig]), so wird das Ausspielen einiger Audio-Rahmen wiederholt (Wiederholen). Ist der Puffer zu voll ([zu viel]), so werden Rahmen ungespielt verworfen (Verwerfen).

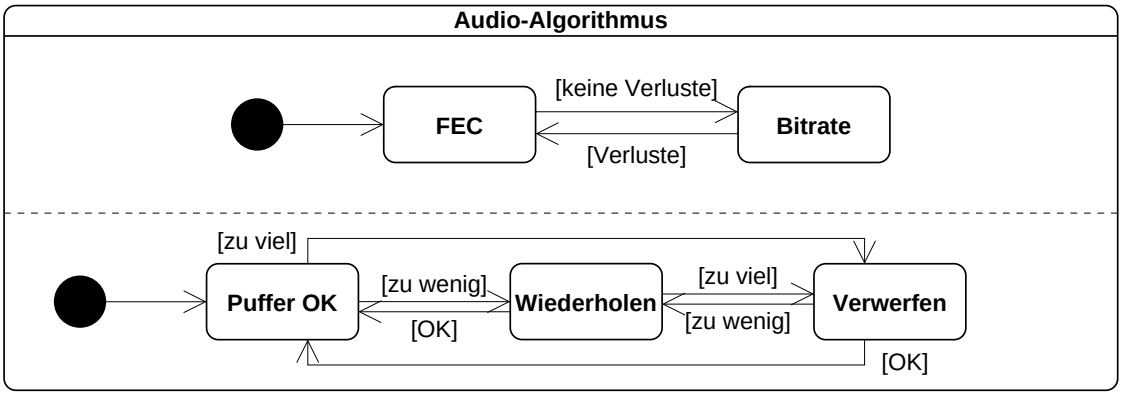


Abbildung 4.6: Zustandsdiagramm Audio-Adaptionsalgorithmus

Die Mechanismen sind über die Zielwerte der Ende-zu-Ende– und Pufferverzögerung gekoppelt. Tabelle 4.8 fasst die Übergangsbedingungen zusammen.

Übergang	Bedingung
[keine Verluste]	$p_{Ploss} < \gamma_L$
[Verluste]	$p_{Ploss} > \gamma_H \ \wedge \ p_{Ploss}(neu) > \alpha \cdot p_{Ploss}(alt)$
[zu viel]	$\bar{d} - d_{buff} > d_{b,max}$
[zu wenig]	$\bar{d} - d_{buff} < d_{b,min}$
[OK]	$d_{b,min} < \bar{d} - d_{buff} < d_{b,max}$

Tabelle 4.8: Übergangsbedingungen des Audio-Adaptionsalgorithmus

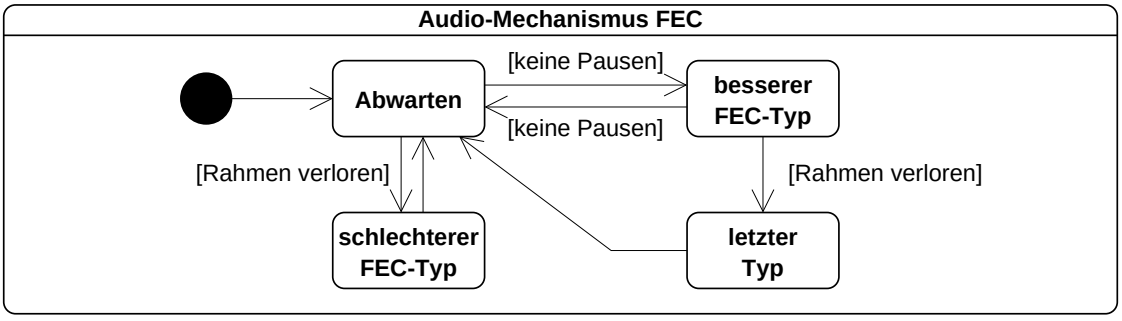


Abbildung 4.7: Zustandsdiagramm Mechanismus FEC

Die Auswahl des passenden FEC-Audio-Typs erfolgt im Mechanismus FEC (Abb. 4.7). Da das System insbesondere nach Änderungen des Audio-Typs einige Zeit braucht, bis wieder verlässliche statistische Daten über den Zustand der Übertragung vorliegen, wartet der Mechanismus zunächst ab (Abwarten). Treten keine Ausspielpausen mehr auf ([keine Pausen]), d. h. die Rahmenverlustwahrscheinlichkeit p_{Floss} ist sehr klein, so konnten die Paketverluste mit Hilfe der redundanten

Audio-Rahmen ausgeglichen werden. Der Algorithmus sucht nun den passenden FEC-Typ, indem er den nächstbesseren FEC-Audio-Typ für die Übertragung benutzt (besserer FEC-Typ) und prüft, ob sich hierdurch die Zahl der Ausspielpausen erhöht. Ist dies nicht der Fall, so wird wieder etwas gewartet, bevor eine weitere Verbesserung versucht wird. Treten jedoch mehr Ausspielpausen auf ([Rahmen verloren]), so macht der Mechanismus die Verbesserung rückgängig (letzter Typ), indem der zuvor benutzte Typ eingesetzt wird. Die anschließende Wartezeit bis zum nächsten Verbesserungsversuch wird erhöht.

Gehen Rahmen verloren, während der Mechanismus sich im Wartezustand befindet ([Rahmen verloren]), so schaltet er auf einen schlechteren FEC-Typ um (schlechterer FEC-Typ). Danach wartet der Mechanismus wieder einige Sekunden ab, bis sich die Übertragung stabilisiert hat.

Treten bei der Übertragung keine zufälligen Verluste auf, so verwendet der Algorithmus den Mechanismus Bitrate (Abb. 4.8), der den nun passenden Audio-Typ auswählt. Sein Ablauf ist analog zur Wahl des FEC-Typs (s. o.), jedoch mit dem Unterschied, dass nicht die fehlenden Audio-Rahmen, sondern die Paketverlustwahrscheinlichkeit p_{Ploss}, die Ziel-Bitrate B_T und die empfangene Bitrate B_A betrachtet werden.

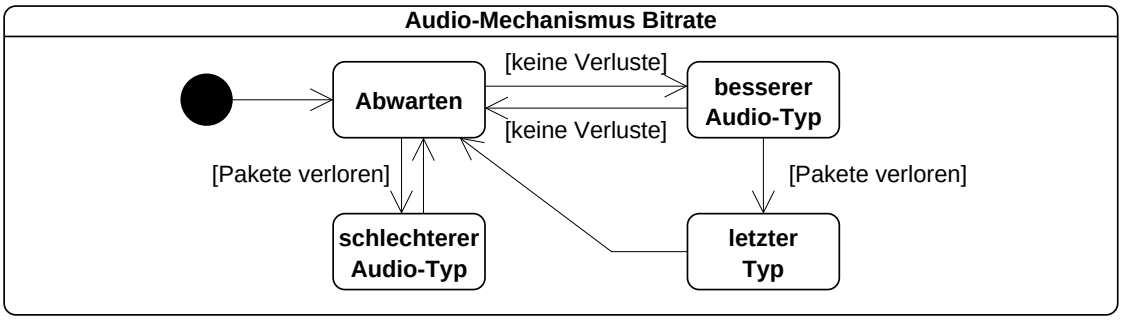


Abbildung 4.8: Zustandsdiagramm Mechanismus Bitrate

Übergang	Bedingung
[keine Pausen]	$p_{FIoss} < \varepsilon_L$
[Rahmen verloren]	$p_{FIoss} > \varepsilon_H$
[keine Verluste]	$p_{Ploss} < \varepsilon_P \wedge B_T \approx B_A$
[Pakete verloren]	$p_{Ploss} > \varepsilon_P \vee B_T > B_A$

Tabelle 4.9: Übergangsbedingungen der Mechanismen

Die Übergänge zwischen den Zuständen sind in Tab. 4.9 aufgeführt und die in dieser Untersuchung verwandten Parameter werden in Tab. 4.10 definiert.

Die Paketverlustwahrscheinlichkeit p_{Ploss} wird nach Gl. 4.5 abgeschätzt, wobei n_{rec} der Zahl der im Beobachtungsintervall angekommenen Pakete und n_{exp} der Zahl der erwarteten Pakete (ermittelt anhand der Sequenznummern) entsprechen.

$$\text{(Paketverlustwahrscheinlichkeit)} \quad p_{Ploss} := 1 - \frac{n_{rec}}{n_{exp}} \tag{4.5}$$

Bedeutung	Symbol	Beispiel
Grenze für hohe Verluste	γ_H	1 %
Grenze für niedrige Verluste	γ_L	0.5 %
Faktor für Verlustverbesserung	α	0.7
Untere Ausspielpuffergrenze	$d_{b,min}$	25 ms
Obere Ausspielpuffergrenze	$d_{b,max}$	80 ms
Länge des Beobachtungsintervalls	$d_{observe}$	5 s
Untere Grenze für Ausspielpausen	ε_L	0.1 %
Obere Grenze für Ausspielpausen	ε_H	0.4 %
Grenze für Verluste	ε_P	1 %

Tabelle 4.10: Parameter der Audio-Adaption

4.4 Beschreibung des Video-Mediums

Hier wird der Aufbau und der Ablauf der Video-Übertragung vorgestellt. Zunächst werden die beteiligten Module und ihre Funktionen erläutert, danach die Steuerparameter, die das Verhalten der Übertragung beeinflussen, und die Zustandsparameter aufgeführt. Es werden dann Anpassungsmechanismen und Adaptionsalgorithmen dargestellt.

Anmerkung

Das zur Zeit bekannteste Übertragungsformat für Video-Ströme, MPEG, eignet sich aufgrund seines Arbeitsprinzips (die Speicherung von Gruppen aufeinander folgender Bilder zur Berechnung der Unterschiede zwischen ihnen) nicht für interaktive Anwendungen und wurde daher hier nicht untersucht.

4.4.1 Implementierung

Video-Bilder werden im Sender (Abb. 4.9) von einem Frame-Grabber–Interface digitalisiert und vom Video-Interface XP_CVideoInterfaceModule als Video-Rahmen XP_CVideoFrame an ein Kodierungs-Modul XP_CVideoToRTJpeg übergeben, das die Rahmen nach dem „RT-JPEG“–Verfahren [Sch98] komprimiert. Es übergibt die entstehenden Puffer mit Zeitstempel (XP_CStampedBuffer) an ein Segmentierungs-Modul XP_CSegmentation, das sie in Pakete segmentiert und diese an den Empfänger-Proxy XP_CMedRecProxyIP weiterreicht (s. 4.2.3, S. 68). Die Paketübertragung erfolgt dann über das Netz zum Sender-Proxy XP_CMedTransProxyIP im Empfänger.

Im Empfänger werden die Segmente vom Sender-Proxy XP_CMedTransProxyIP empfangen und im Modul XP_CReassembly zusammengesetzt. Das komprimierte Bild wird in ein unkomprimiertes dekodiert (XP_CVideoFromRTJpeg) und dieses dann vom Bildschirm-Interface XP_CDisplayFramePerFrame angezeigt.

Die Bilder werden in dieser Implementierung nur im Modul XP_CReassembly gepuffert, da dieses Modul Segmente zwischenspeichert, um sie zusammenzusetzen. In einer abgewandelten Konfiguration können abgeleitete Klassen XP_CSegmentationFEC und XP_CReassemblyFEC verwendet werden, die für jeden Rahmen ein zusätzliches Redundanz-Paket nach dem XOR-Verfahren (s. 3.5.3.4, S. 59) erzeugen, so dass der Verlust eines Pakets ausgeglichen werden kann.

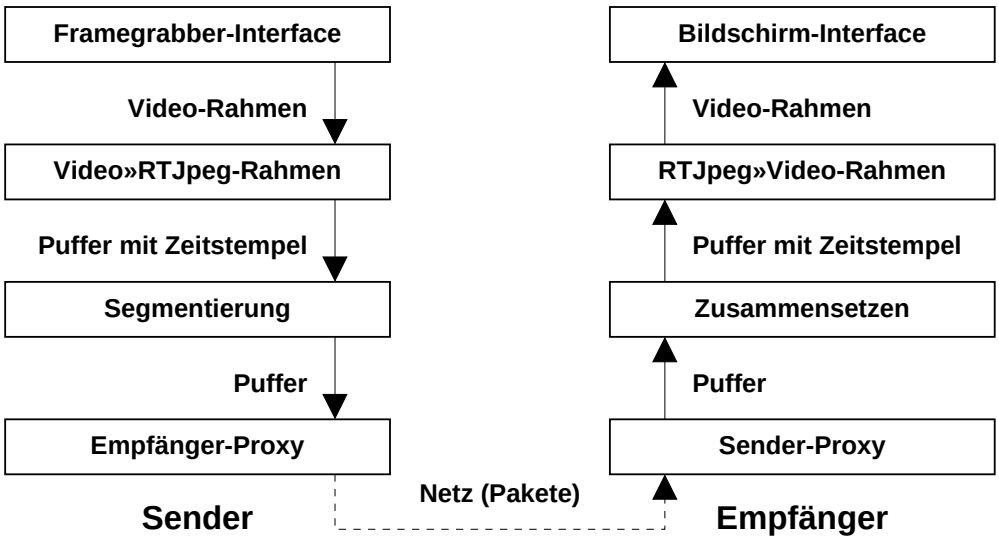


Abbildung 4.9: Aufbau der Video-Übertragung

Außer diesem Aufbau sind noch zwei Varianten möglich, die vom Dienstgüten-Management ausgewählt und während der Übertragung aktiviert werden können. Zum einen ist es möglich, zwischen die Kodierung und die Segmentierung (bzw. das Zusammensetzen und die Dekodierung) ein zusätzliches Modul einzubinden, das die Puffer verlustlos komprimiert (bzw. dekomprimiert, s. 3.5.3.2, S. 56). Dies führt zu einer Reduktion der Bitrate in der Netz-Schicht, jedoch auch zu zusätzlichem Rechenaufwand und einer weiteren Verzögerung.

Blockweise Übertragung

Zum anderen ist es möglich, statt einer *bildweisen* Übertragung eine *blockweise* Übertragung zu verwenden (Abb. 4.10). Hier wird das digitalisierte Bild nicht komplett kodiert, sondern es werden Bildausschnitte einstellbarer Größe, sog. „Blöcke“, kodiert (Abb. 4.11) und in Gruppen an den Empfänger geschickt. Dies hat den Vorteil, dass bei Verlust eines Pakets nur die in diesem Paket befindlichen Blöcke verloren gehen und die Blöcke der anderen Pakete angezeigt werden können. Dagegen geht bei Verlust eines Pakets bei bildweiser Übertragung der komplette Rahmen verloren, bzw. kann nur durch Senden von Redundanz (FEC) gerettet werden.

Im Empfänger werden die (ggf. dekomprimierten) Pakete in einem Pufferungs-Modul XP_CReceiveBuffer zwischengespeichert und um target(CLQ/BufferDelay) verzögert, bevor sie dekodiert und angezeigt werden. Nachteile der blockweisen Übertragung sind die höhere benötigte Bitrate und das Auftreten sog. „Artefakte“, d. h. rechteckiger Bildfehler, wenn Blöcke verloren gehen. Dies ist dann besonders störend, wenn sich der Bildinhalt sehr schnell ändert. Um diesen Effekt in die Dienstgütenabschätzung einfließen zu lassen, meldet das Modul im Parameter ALQ/PictureDistortion die Schwere der Bildstörungen, hier den Anteil der verlorenen Blöcke.

Abbildung 4.12 zeigt den zeitlichen Ablauf der Kodierung eines Video-Rahmens im Sender. Bereits im Frame-Grabber kann ein Jitter ALQ/FrameGenerationJitter entstehen, d. h. eine Schwankung der Abstände zwischen Rahmen, da beim PAL-Videosignal 25 Rahmen pro Sekunde (*frames per second*, fps) digitalisiert, jedoch meist mit einer niedrigeren Rate übertragen werden. Um z. B.

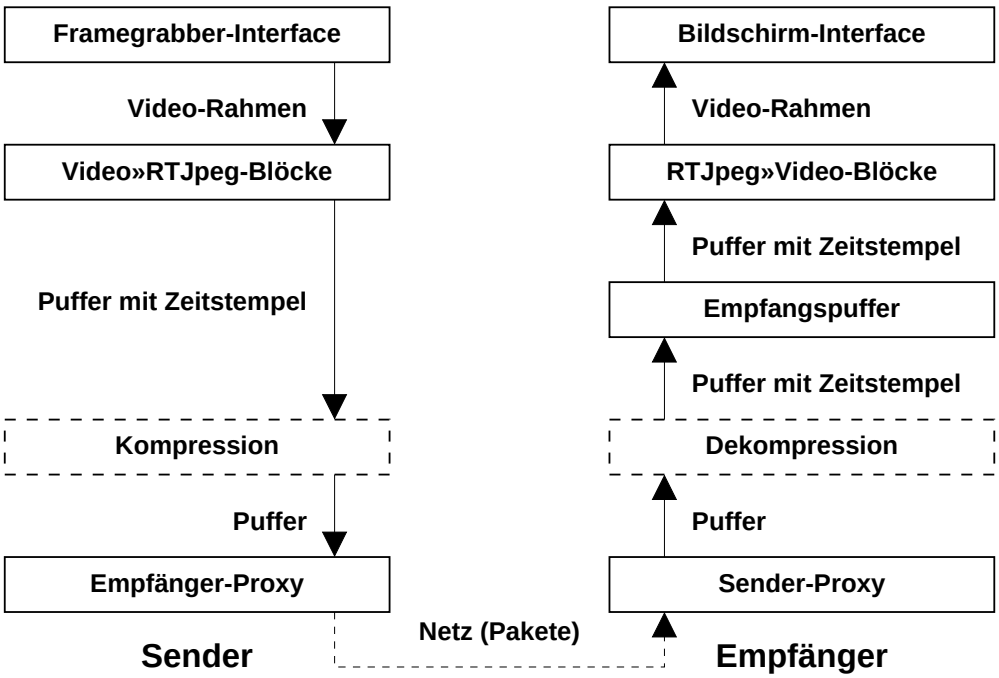


Abbildung 4.10: Aufbau der blockweisen Video-Übertragung mit optionaler Kompression

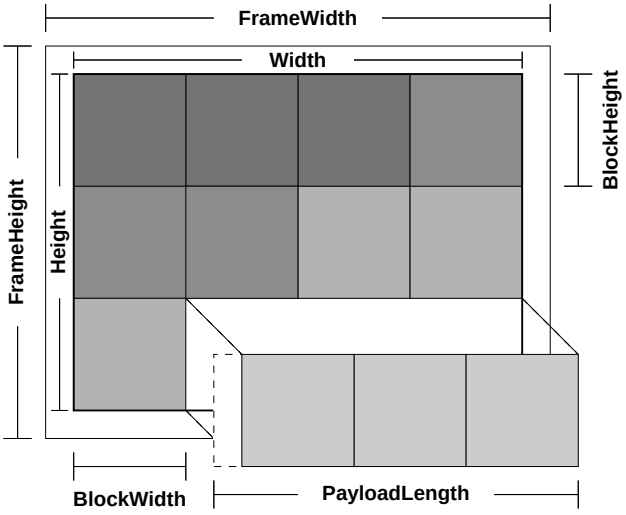


Abbildung 4.11: Aufteilung eines Video-Rahmens in Blöcke

eine Rate von 15 fps zu erreichen, werden zwei von fünf Rahmen nicht benutzt, so dass der Abstand der Erzeugung der verwendeten Rahmen entweder 40 oder 80 ms beträgt.

Es addieren sich weitere Verzögerungen, zunächst ALQ/FrameCodingDuration für die Kodierung und CLQ/CompressionDuration für die (optionale) verlustlose Kompression. Diese Parameter erlauben dem Dienstgüten-Management die Entscheidung, ob genug Rechenleistung im Sender zur Verfügung steht, ob z. B. nur ein kleineres Bild ausreichend schnell kodiert werden kann oder ob

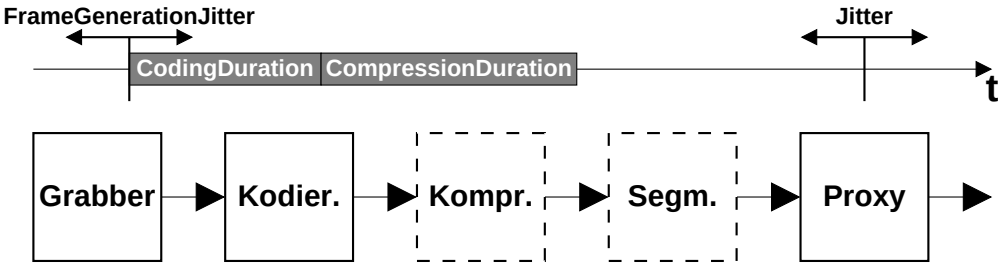


Abbildung 4.12: Generierung des Video-Stroms

auf Kompression verzichtet werden sollte. Das Modul zur Segmentierung der kodierten (und ggf. komprimierten) Daten wird nur bei der bildweisen Übertragung benutzt.

Im Empfänger wird der Video-Datenstrom nach Abb. 4.13 empfangen und angezeigt. Bei der bildweisen Übertragung setzt das Modul XP_CReassembly die Pakete zusammen und verzögert sie, bis seit der Erzeugung des Bildes target(CLQ/BufferDelay) Sekunden verstrichen sind. Optional wird der Puffer dekomprimiert, und danach in ein anzeigbares Format dekodiert. Dieses Bild (bzw. ein Bildausschnitt) wird dann vom Schirm-Modul angezeigt.

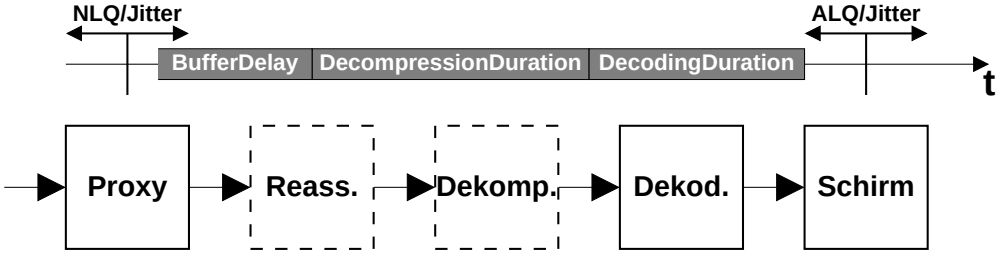


Abbildung 4.13: Empfang und Anzeige des Video-Stroms

Jeder Video-Rahmen führt einen *Header* mit, der seine Größe in Pixeln (*„picture element“*, Bildpunkt) angibt (ALQ/Width und ALQ/Height, bzw. ALQ/BlockWidth und ALQ/BlockHeight, falls es sich um einen Block handelt), außerdem wird der Typ der Bilddaten gespeichert (Tab. 4.11).

Typ	Name	Beschreibung	bit/pixel
R565	RGB 565	5 bit Rot-Wert 6 bit Grün-Wert 5 bit Blau-Wert	16
420P	YUV 420 planar	Y: 1 byte/pixel Helligkeit U: 1 byte Farbwert je 4 pixel V: 1 byte Farbwert je 4 pixel	12
RTjp	Real-time JPEG	Kodierte Video-Rahmen	≈ 1

Tabelle 4.11: Typen der Bildinformation

`R565` ist dabei ein Anzeigeformat des Bildschirms, wobei jeweils 2 byte die Werte für den Rot-, Grün- und Blau-Anteil eines Pixels enthalten. Dagegen werden im `420P`–Format die Helligkeitswerte von den Farbinformationen getrennt gespeichert, daher die Bezeichnung „planar“. Zudem wird zwar die Helligkeit jedes Bildpunkts abgespeichert, jedoch werden die Farbwerte von vier Punkten zusammengefasst, da das Auge empfindlicher für Helligkeits- als für Farbunterschiede ist. Diese Trennung ist eine Voraussetzung für die Kodierung nach dem JPEG–Verfahren, das die Helligkeit der Pixel weniger verlustbehaftet kodiert als die Farbwerte. Da der Frame-Grabber `420P` liefern kann, entfällt ein Konvertierungsschritt im Sender. Schließlich werden kodierte Rahmen mit dem Typ `RTjp` gekennzeichnet.

Bitrate des kodierten Videosignals

Die Größe in bit eines mit dem `RT-JPEG` kodierten Video-Bildes hängt stark von seinem Inhalt ab. Sie kann z. B. für ein Bild der Größe 384×288 pixel bei einem Quantisierungsfaktor $P_Q = 100$ zwischen 6000 und 40000 byte liegen. Dies bedeutet, dass die Senderate des Video-Mediums auch ohne Änderung der Parameter des Senders über die Zeit variiert, in Abhängigkeit des Eingangssignals. In den meisten Systemen wird nun die Bitrate des Senders automatisch auf einen bestimmten Wert eingeregelt, indem Parameter wie Bildqualität und Bild- bzw. Blockrate variiert werden. Dieses Vorgehen würde jedoch in dieser Untersuchung zu dem Problem führen, dass die Adaptionsalgorithmen nicht mehr alle Parameter des Video-Mediums bestimmen können, so dass ein Vergleich ihrer Leistung kaum noch möglich wäre. Stattdessen wird daher stets dasselbe Eingangssignal für die Experimente verwandt (d. h. derselbe Ausschnitt wird übertragen) und bei der Bewertung der Verlauf der Sende-Bitrate in Betracht gezogen.

Berechnung der Video-Dienstgüte

In die Berechnung der Dienstgüte des Video-Mediums nach Gl. 4.6 gehen die Parameter Bildrate f, mittlere Ende-zu-Ende–Verzögerung $\bar{d}$, Quantisierungsfaktor P_Q und Bildbreite P_W und -höhe P_H ein. Schließlich wird die Dienstgüte durch mögliche Bildstörungen P_{Dist} begrenzt (Gl. 4.7), wie sie bei der blockweisen Übertragung durch verlorene Blöcke („Artefakte“) auftreten können. Für die verwendeten Abbildungen des Zustands auf ULQ-Werte und die Gewichtung der Parameter sei auf Tabellen im Anhang verwiesen (s. B.8.3, S. 154).

$$\begin{aligned} Q' &= w_{fps} \cdot Q(f) + w_{delay} \cdot Q(\bar{d}) + \\ &\quad w_{PQ} \cdot Q(P_Q) + w_{PW} \cdot Q(P_W) + w_{PH} \cdot Q(P_H) \end{aligned} \tag{4.6}$$

$$Q = \min(Q', Q(P_{Dist})) \tag{4.7}$$

4.4.2 Steuer- und Zustandsparameter

Die Steuerparameter der Video-Module sind in Tab. 4.12 zusammengefasst, die Zustandsparameter in Tab. 4.13.

4.4.3 Video-Anpassungsmechanismen

Für die Video-Übertragung wurden die folgenden Mechanismen implementiert. Zunächst wird der beobachtete Effekt erläutert und die verantwortliche Ursache erklärt. Es wird dargestellt, wie der

Modul: Parameter	in	Beschreibung
Interface:		
ALQ/Delay	s	Ende-zu-Ende–Verzögerung $\bar{d_T}$
ALQ/FrameHeight	pixel	Höhe des digitalisierten Bildes
ALQ/FrameWidth	pixel	Breite des digitalisierten Bildes
ALQ/FramesPerSecond	fps	Bilder pro Sekunde
Kodierung:		
ALQ/BlockHeight	pixel	Höhe eines Blocks
ALQ/BlockWidth	pixel	Breite eines Blocks
ALQ/Height	pixel	Höhe P_H des übertragenen Bildes
ALQ/PictureQuality	–	RT-Jpeg–Quantisierungsfaktor P_Q
ALQ/Width	pixel	Breite P_W des übertragenen Bildes
Segmentierung:		
CLQ/BufferDelay	s	Zielverzögerung nach Pufferung
CLQ/SegmentSize	byte	maximale Segmentgröße
CLQ/SendRate	bit/s	Ziel-Senderate

Tabelle 4.12: Steuer-/Zielparameter der Video-Übertragung

Modul: Parameter	in	Beschreibung
Interface:		
ALQ/DecodingDuration	s	Dauer der Dekodierung eines Bildes
ALQ/Delay	s	Ende-zu-Ende–Verzögerung $\bar{d_A}$
ALQ/FrameCodingDuration	s	Dauer der Kodierung eines Bildes
ALQ/FrameGenerationJitter	s	Schwankung der Erzeugungsabstände der Bilder
ALQ/FramesPerSecond	fps	Bilder pro Sekunde, f
ALQ/Jitter	s	Schwankung des Anzeigeabstände der Bilder
ALQ/PictureDistortion	1	Störung der Bildqualität P_{Dist}
ALQ/PictureQuality	–	RT-Jpeg–Quantisierungsfaktor P_Q
Kodierung:		
CLQ/CompressionDuration	s	Dauer der verlustlosen Kompression
CLQ/CompressionFactor	1	erreichter Kompressionsfaktor
CLQ/DecompressionDuration	s	Dauer der Dekompression
Proxy:		
NLQ/Bandwidth	bit/s	Bitrate B_A des empfangenen Verkehrs
NLQ/Delay	s	Verzögerung der Video-Daten *vor* Pufferung
NLQ/Jitter	s	Verzögerungsschwankung der Pakete
NLQ/PacketLossProbability	1	Paketverlustwahrscheinlichkeit p_{Ploss}
NLQ/SenderBandwidth	bit/s	Bitrate des ausgesandten Verkehrs

Tabelle 4.13: Zustandsparameter der Video-Übertragung

Adaptionsalgorithmus diese Situation erkennen kann und wie er darauf reagiert. Auf eine Anpassung der Bildgröße wurde verzichtet, weil die Auswirkungen auf die subjektive Dienstgüte zu

groß erschienen. Da die benötigte Dauer zur Bearbeitung der Bilder (Kodierung und Kompression im Sender, bzw. Dekompression, Dekodierung und Anzeige im Empfänger) im Verhältnis zur Netz-Verzögerungszeit klein war, wurde auch auf eine Anpassung hinsichtlich dieser Parameter verzichtet.

4.4.3.1 Geringe verfügbare Netz-Bitrate

Die für die Video-Übertragung verfügbare Bitrate im Netz ist geringer, als die Senderate. Hierdurch kommt es zu Paketverlusten.

Ursache:	Im Netz laufen Puffer über und Pakete werden verworfen, da die Zuflussrate die Abflussrate übersteigt.
Erkennung:	Die beobachtete Paketverlustwahrscheinlichkeit p_{Ploss} ist größer als $\gamma_H + p_{random}$, wobei p_{random} die Abschätzung der zufälligen Paketverlustwahrscheinlichkeit im Netz ist.
Reaktionen:	Die Ziel-Senderate CLQ/SendRate wird reduziert.

4.4.3.2 Erhöhung der verfügbaren Netz-Bitrate

Es steht eine größere Bitrate für das Video-Medium zur Verfügung, als genutzt wird.

Ursache:	Eine Beeinträchtigung der verfügbaren Bitrate (Querverkehr, Reservierungen) ist nicht mehr gegeben.
Erkennung:	Die beobachtete Paketverlustwahrscheinlichkeit p_{Ploss} ist kleiner als γ_L.
Reaktionen:	Die Ziel-Senderate CLQ/SendRate wird erhöht.

4.4.3.3 Niedrige Bildrate

Die beim Ausspielen beobachtete Bildrate ist geringer als der Zielwert.

Ursache:	Das Videosignal braucht mit der Ziel-Bildrate eine höhere Bitrate, als durch die Ziel-Senderate gesendet werden darf. Es werden also weniger Bilder pro Sekunde gesandt, als gewünscht.
Erkennung:	Die erreichte Bildrate achieved(ALQ/FramesPerSecond) ist um φ_L kleiner als die Ziel-Bildrate target(ALQ/FramesPerSecond).
Reaktion:	Der Quantisierungsfaktor P_Q^* (ALQ/PictureQuality) wird mit β_{Qdec} reduziert, wobei ein Minimalwert P_{Qmin} nicht unterschritten wird: $P_Q^* := P_Q - \beta_{Qdec} \cdot (P_Q - P_{Qmin})$
Siehe:	3.5.2.4, S. 51

4.4.3.4 Hohe Bildrate

Die beim Ausspielen beobachtete Bildrate erreicht den Zielwert. Eine mögliche Reduktion der Bildqualität kann zurückgenommen werden.

Ursache:	Das Videosignal benötigt eine niedrigere Bitrate und der Sender erreicht in etwa die Ziel-Bildrate.
Erkennung:	Die erreichte Bildrate achieved(ALQ/FramesPerSecond) ist größer als das φ_H-fache der Ziel-Bildrate target(ALQ/FramesPerSecond).

Reaktion: Der Quantisierungsfaktor P_Q^* (ALQ/PictureQuality) wird mit β_{Qinc} erhöht, wobei der maximale Wert P_{Qmax}, der durch die Zieldienstgüte des Mediums vorgegeben wird, nicht überschritten wird:
$P_Q^* := \min\{P_Q + \beta_{Qinc} \cdot (P_Q - P_{Qmin}),\ P_{Qmax}\}$

Siehe: 3.5.2.4, S. 51

4.4.3.5 Verzögerungsanpassung

Der Ausspielzeitpunkt der Bilder wird an die Übertragungsverzögerung und ihre Schwankung angepasst, um zu verhindern, dass Blöcke oder ganze Bilder aufgrund von Überholungen verworfen werden müssen. Ebenso ist dies zur Synchronisierung mit einem weiteren Medium (z. B. einem Audio-Strom) notwendig.

Ursache: Ändert das Netz die Reihenfolge der Pakete, so können Blöcke späterer Bilder vor Blöcken früherer Bilder ankommen (es ist allerdings unwahrscheinlich, dass ganze Rahmen die Reihenfolge wechseln). Wurde ein Block des späteren Bildes angezeigt, so dürfen die Blöcke des früheren Bildes nicht mehr angezeigt werden, da dies zu Artefakten führen würde.

Erkennung: (Wird ständig durchgeführt.)

Reaktionen: Die Ziel-Verzögerung d_{Tbuff} nach Pufferung target(CLQ/BufferDelay) wird auf einen Wert gesetzt, der von der Ziel-Ende-zu-Ende–Verzögerung $\bar{d_T}$ und den Werten für die Verzögerung im Netz d_{Net}, deren Jitter Δd_{Net} und der Bearbeitungsdauer d_R im Empfänger abhängig ist. Mit der Konstante ε kann der Einfluss des Jitters auf die Pufferzeit eingestellt werden.
$d_{Tbuff} := \max\{\bar{d_T} - d_R,\ d_{Net} + \varepsilon \cdot \Delta d_{Net}\}$

4.4.4 Video-Adaptionsalgorithmus

Der Video-Adaptionsalgorithmus (Abb. 4.14) startet mit einer blockweisen Übertragung (Blocks) und nimmt an, dass die Übertragungsstrecke nicht verlustbehaftet ist ($p_{random} = 0$). Liegt die Paketverlustwahrscheinlichkeit über einer oberen Schranke γ_H und gab es keine Verbesserung ([hohe Verluste]), so schaltet der Algorithmus in einen Zwischenzustand (Observe) und wartet ab. Sinkt die Verlustwahrscheinlichkeit wieder unter die untere Schranke γ_L ([kleine Verluste]), so wird die Übertragung weiter blockweise durchgeführt. Bleibt es trotz des Einsatzes der Anpassungsmechanismen bei Paketverlusten ([Verluste]), schaltet der Algorithmus in den FEC-Modus um (FEC) ($p_{random} > 0$), in dem die Bilder mit einem zusätzlichen Redundanzpaket gegen Paketverluste geschützt werden. Nur wenn die Paketverlustwahrscheinlichkeit wieder unter die untere Schranke sinkt ([kleine Verluste]) geht der Algorithmus wieder in den Rahmen-Übertragungsmodus über. (Die Änderung des Übertragungsmodus erzeugt eine deutliche Störung durch ein kurzzeitiges Absinken der Bildrate, daher darf sie nicht zu häufig durchgeführt werden.)

Die Übergänge zwischen den Zuständen sind in Tab. 4.14 aufgeführt. Die verwandten Parameter werden in Tab. 4.15 definiert. Die Paketverlustwahrscheinlichkeit p_{Ploss} wird nach Gl. 4.5 (S. 78) abgeschätzt.

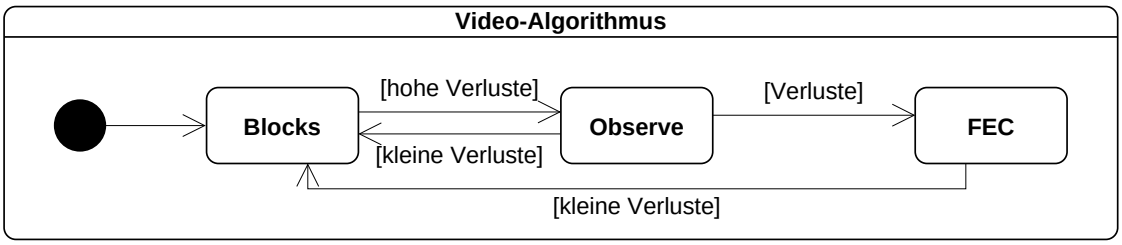


Abbildung 4.14: Zustandsdiagramm der Anpassung des Übertragungsmodus

Übergang	Bedingung
[kleine Verluste]	$p_{Ploss} < \gamma_L$
[Verluste]	$p_{Ploss} > \gamma_L$
[hohe Verluste]	$p_{Ploss} > \gamma_H \wedge p_{Ploss}(neu) > \alpha \cdot p_{Ploss}(alt)$

Tabelle 4.14: Übergangsbedingungen Video-Adaptionsalgorithmus

4.5 Untersuchte Szenarien

Die Leistung eines Adaptionsalgorithmus kann nur in Bezug auf ein Szenario bewertet werden. Neben der konkreten Situation, in der die Anwendung eingesetzt wird, beschreibt es das Netzverhalten, das die Schwankung der erfahrenen Dienstgüte verursacht. Mit dem Ziel, Adaptionsalgorithmen zu entwickeln, die sich auch in der Realität bewähren, führt dies zu den folgenden Anforderungen an die Szenarien:

- Relevanz: Der untersuchte Fall ist ein wirklich vorkommendes Problem, bei dem der Einsatz von Adaptionsalgorithmen möglich ist.
- Beschreibbarkeit, Wiederholbarkeit: Das Verhalten kann in einer Form beschrieben werden, die die reproduzierbare Durchführung von Experimenten erlaubt.

Bedeutung	Symbol	Beispiel
Schätzwert für zufällige Paketverluste	p_{random}	(wird berechnet)
Grenze für hohe Verluste	γ_H	5 %
Grenze für niedrige Verluste	γ_L	2 %
Faktor zur Reduktion des Quantisierungsfaktors	β_{Qdec}	0.1
Faktor zur Erhöhung des Quantisierungsfaktors	β_{Qinc}	0.1
Faktor für Verlustverbesserung	α	0.9
Gewichtung des Netz-Jitters	ε	0.5
Faktor untere Bildratenschwelle	φ_L	0.6
Faktor obere Bildratenschwelle	φ_H	0.8

Tabelle 4.15: Parameter des Video-Adaptionsalgorithmus

- ❏ Komplexität: Das Szenario ist nicht zu komplex, um das beobachtete Verhalten der Algorithmen verstehen zu können.

Eine Szenarienbeschreibung umfasst dabei folgende Punkte:

- ❏ Art der Multimedia-Anwendung: z. B. Video-Konferenz, Telefonie-Anwendung, Spiel, etc.
- ❏ Anforderung an die Dienstgüte durch den Nutzer: Erwartet dieser einen kostengünstigen Dienst, einen mit mittlerer Dienstgüte oder entspricht nur eine ausgezeichnete Qualität seinen Ansprüchen?
- ❏ Einsetzbare Ressourcen: Welche Ausstattung steht für die Anwendung zur Verfügung? Dies umfasst die Klasse der Endgeräte, von spezialisierten Geräten (z. B. Telefon, „Set-Top-Box“) über Geräte mit geringer Leistung (z. B. PDA, *Personal Digital Assistant*) bis hin zu leistungsstarken Geräten (z. B. moderne PC). Zudem wird die unveränderliche Charakteristik des Netzes beschrieben, z. B. die Art des Netzzugangs (Modem, Funkverbindung, LAN, usw.) und des WANs.
- ❏ Netzverhalten: Es wird festgelegt, wie sich das Verhalten des Netzes verändert, z. B. eine Änderung der Verzögerung oder der Paketverlustwahrscheinlichkeit.

4.5.1 Netz-Emulation

Für die experimentelle Bewertung der Leistung eines Adaptionsalgorithmus muss dieser auf Veränderungen in der Netzdienstgüte reagieren. Hierzu ist es also notwendig, über ein Netz mit einstellbarem Verhalten zu verfügen, denn es wäre zwar möglich, die Anpassung bei einer Kommunikation über ein reales WAN zu untersuchen (Abb. 4.15), jedoch ist hier die Wiederholbarkeitsbedingung nicht gegeben. Zudem ist das Wissen um den Zustand eines solchen Netzes stets begrenzt. Es existiert noch die Möglichkeit, die Übertragung gezielt mit Störverkehr zu beeinträchtigen. Hier ist jedoch die Beschreibung der Effekte auf den beobachteten Verkehr problematisch. Theoretisch könnte die Untersuchung auch in einer Simulation erfolgen, jedoch müssten hierzu die kompletten Endgeräte und die Anwendung nachgebildet werden (s. C, S. 157).

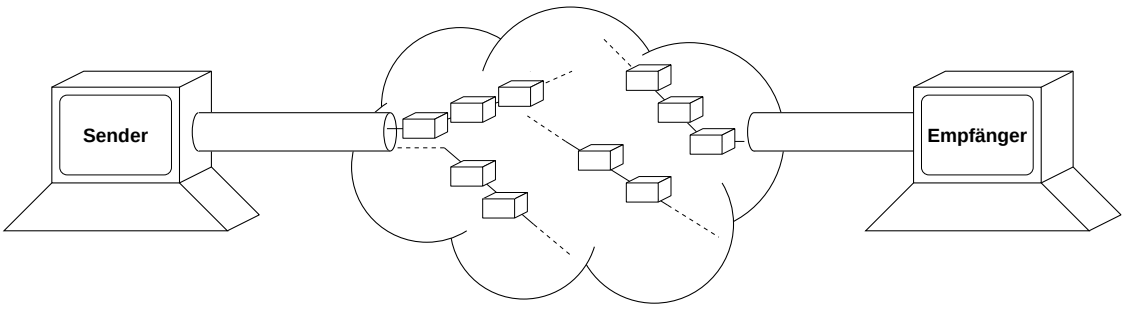


Abbildung 4.15: Kommunikation über ein WAN

Eine realisierbare Möglichkeit ist die *Emulation* eines WAN mit Hilfe eines PC und entsprechender Software. Der prinzipielle Aufbau besteht aus Sende- und Empfänger-Endgerät und dem Emulator-PC. Auf diesem läuft ein geeignetes Programm; in diesem Projekt wurden „The Cloud“ und „NIST

Net“ eingesetzt (s. u.), um auch den Einfluss der verschiedenen Emulationsstrategien auf die Ergebnisse zu untersuchen. Kommunizieren Endgeräte über ein WAN, so besteht die Übertragungsstrecke aus den Zugangsleitungen (z. B. ISDN, ADSL) und den Routern im Netz (Abb. 4.16). Die Emulation muss daher sowohl die Zugänge zum, als auch die Übertragung im WAN nachbilden.

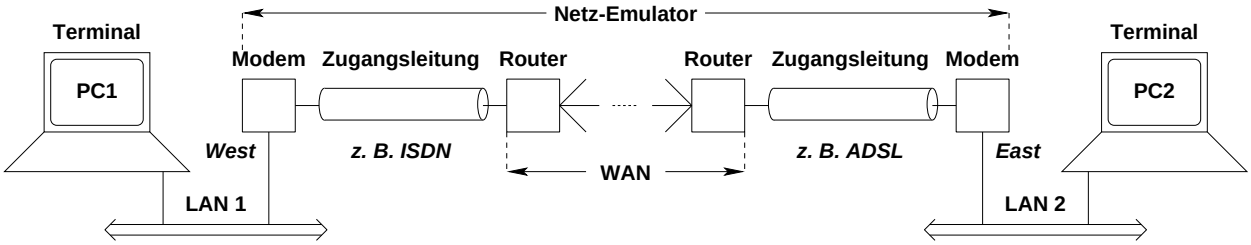


Abbildung 4.16: Struktur der Ende-zu-Ende–Übertragungsstrecke

Die Endgeräte werden so konfiguriert, dass alle Pakete, die für das andere Endgerät bestimmt sind, zunächst über das LAN (hier Ethernet) an ein IP-*Gateway* gesandt werden (Details s. A.4, S. 146). Als dieses Gateway fungiert der WAN-Emulator-PC und nimmt das IP-Paket entgegen. Normalerweise würde das Paket so schnell wie möglich über ein Interface weiter geschickt, über das es den Zielrechner erreichen kann. Das Programm zur WAN-Emulation speichert nun aber das Paket und entscheidet, wie mit ihm zu verfahren ist.

Das Paket kann von der WAN-Emulation verzögert, verworfen und/oder verändert werden. Da Änderungen am Inhalt eines Pakets selten auftreten und meist zu Paketverlusten führen (vgl. UDP, RFC 1122/4.1.3.4), muss dieser Fall nicht gesondert betrachtet werden.

Ein externer Beobachter kann das Verhalten der WAN-Emulation durch die Zeitpunkte beschreiben, zu denen ein Paket vom Emulator empfangen ($t_{receive}$) und wieder ausgesandt wird (t_{send}, s. Gl. 4.8). D nimmt dabei den Wert unendlich an, falls das Paket verloren geht.

$$t_{send} = t_{receive} + D \tag{4.8}$$

4.5.1.1 The Cloud

Bei *The Cloud* (ein kommerzielles Produkt der Firma *Shunra* [Clo], verwandt wurde Version 2.0) handelt es sich um eine Anwendung, die unter Windows NT läuft. Das WAN wird mit einem Netzmodell nach Abb. 4.17 emuliert. Pakete werden auf dem „West“-Gateway empfangen und in den FIFO[4]-Eingangspuffer gespeichert. Dieser hat die Kapazität l_{Qin}, die entweder in byte oder Anzahl an Paketen gemessen wird. Aus diesem Puffer werden die Pakete mit der Rate B_{in} an die WAN-Emulation gereicht (abhängig vom momentanen Pufferfüllstand $l_{Buffer,in}$ um d_{in} verzögert), die für jedes der Pakete eine Verzögerungsdauer d_{WAN} berechnet. Nach Verstreichen dieser Dauer wird das Paket mit der Rate B_{out} entnommen und in den FIFO-Ausgangspuffer des „East“-Gateway geschrieben (sofern es nicht mit der Wahrscheinlichkeit p_{Ploss} verworfen wurde), der die Kapazität l_{Qout} besitzt. Die Verzögerung zwischen der WAN-Emulation und dem Aussenden des Pakets auf das LAN beträgt d_{out} und hängt vom momentanen Füllstand $l_{Buffer,out}$ ab.

[4] *„first in, first out“*

$$D = d_{in} + d_{WAN} + d_{out} \tag{4.9}$$

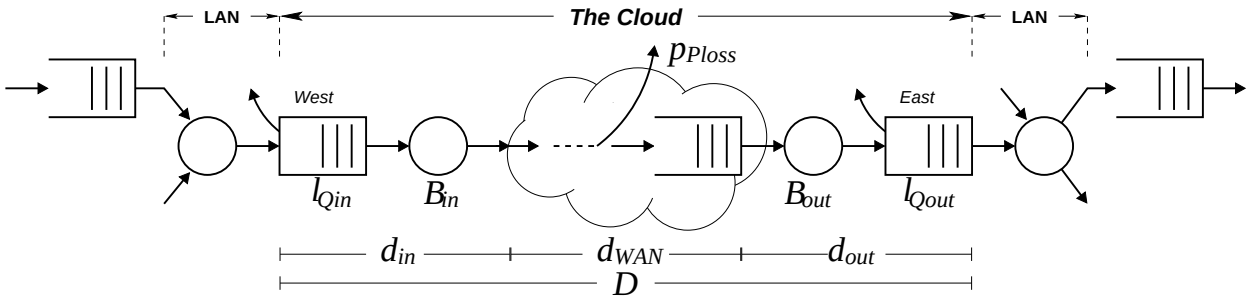


Abbildung 4.17: Netzmodell in *The Cloud*

Um die Emulation zu steuern, können die Parameter über eine Benutzungsoberfläche verändert werden. Für die Bestimmung von d_{WAN}, also wie lange ein Paket im emulierten WAN gespeichert wird, wird zunächst $d'(t)$ nach Gl. 4.10 berechnet. Das Paket wird um mindestens diesen Wert verzögert, jedoch wird sichergestellt, dass sich Pakete nicht überholen.

$$d'(t) = \begin{cases} d_{fixed} & \text{konstante Verzögerung} \\ d_{min} + U(d_{max} - d_{min}) & \text{Gleichverteilung} \\ N(\mu, \sigma) & \text{Normalverteilung} \\ d_{min} + \frac{t \bmod d_{rep}}{d_{rep}}(d_{max} - d_{min}) & \text{lineare Verzögerungsänderung} \end{cases} \tag{4.10}$$

U	gleichverteilte Zufallsvariable
μ, σ	Mittelwert und Standardabweichung
d_{rep}	Dauer des Wiederhol-Intervalls

Paketverluste können sowohl durch eine Paketverlustwahrscheinlichkeit $p_{Ploss} > 0$ (periodische Verwürfe und Burst-Verluste sind ebenfalls einstellbar), als auch durch Puffer-Überläufe hervorgerufen werden. Außerdem kann die Pufferverwaltungsstrategie *Random Early Detection* (RED [FJ93]) gewählt werden, bei der ankommende Pakete mit einer Wahrscheinlichkeit verworfen werden, die vom Füllstand und den Grenzen RED_{low} und RED_{high} abhängt.

Es ist jedoch zu beachten, dass aus den eingestellten Parametern, insbesondere für die Normalverteilung, nicht direkt auf das beobachtete Verhalten geschlossen werden kann. Abhängig von der Art des Verkehrs, der durch den Emulator geleitet wird, treffen Pakete auf unterschiedliche Füllstände der Puffer und werden so im Mittel stärker verzögert, als durch die Raten B_{in} und B_{out} und den Mittelwert μ vorgegeben ist. Dies wird durch die Vermeidung von Paketüberholungen verursacht. Das heißt, ein Paket, das lange verzögert wird, blockiert alle nachfolgenden, auch wenn für diese zufällig eine kürzere Verzögerung bestimmt wurde. Da dieses Verhalten allerdings dem realen Verhalten von Kommunikationsnetzen sehr nahe kommt, ist dies ein sinnvolles Modell.

Abb. 4.18 zeigt ein Beispiel für $\mu = 0.1s$ und $\sigma = 0.03s$ bei 1024 Paketen pro Sekunde. Bei einer niedrigen Paketrate würde die Hälfte der Pakete weniger als 100 ms verzögert, dies ist hier jedoch für kein Paket zu beobachten.

Die Änderung des Netzverhaltens über die Zeit ist mit diesem Programm eingeschränkt möglich. Man kann über eine Datei eine Liste mit Werten für d_{fixed} angeben, die in festen Intervallen von

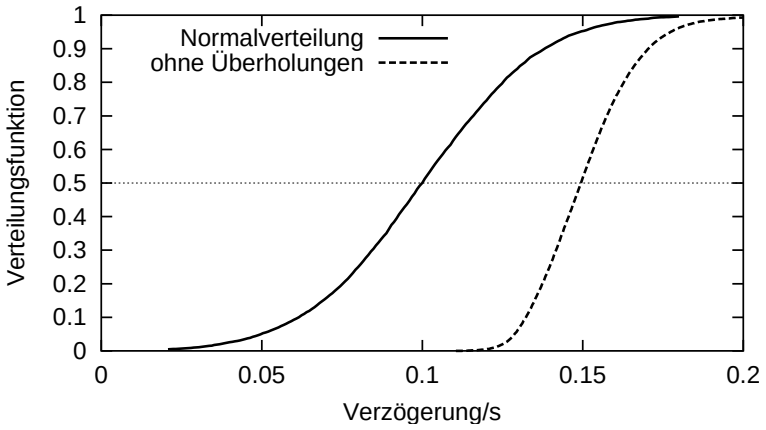


Abbildung 4.18: *The Cloud*, Unterschied Normalverteilung – Beobachtung

jeweils 100 ms bis 2 s Länge[5] emuliert werden. Zudem ist es möglich, Paketverluste periodisch und burstartig zu erzeugen. Es ist jedoch nicht möglich, beliebige Einstellungen abwechseln zu lassen, da das Umschalten zwischen verschiedenen Konfigurationen eine Deaktivierung des Emulators für etwa 300 ms zur Folge hat.

4.5.1.2 NIST Net

NIST Net (ein Copyright-freies *Open Source*–Programm des *National Institute of Standards and Technology* [NIS01], Version 2.0.10, März 2001) ist ein WAN-Emulator, der in den Linux-Kernel eines PC eingreift (es wird ein sog. *Kernel-Modul* installiert) und das *Routing* der IP-Pakete verändert. Neben dem Modul werden noch einige Hilfsprogramme mitgeliefert, mit denen das Verhalten der Emulation bestimmt werden kann.

Der Emulator berechnet für jedes Paket einer eingestellten Route eine Verzögerung, nach der es wieder ausgesandt wird (Gl. 4.11). Es werden zwei Werte berechnet, zum einen die Verzögerung d_{BW} aus der Bitrate B, der Paketlänge l_P und dem Pufferfüllstand l_{Buffer} (Gl. 4.12), als auch eine zufällige Verzögerung d_{rand}. Das Paket wird mit dem größeren der beiden Werte verzögert (Abb. 4.19 links). Dies hat zur Folge, dass Änderungen in der Paketreihenfolge sehr häufig auftreten, sofern die Zufallsverzögerung entsprechend groß ist.

Die Verteilung des Zufallswerts d_{rand} wird über eine Tabelle im Emulator und die eingestellten Parameter bestimmt und entspricht in etwa einer Normalverteilung.

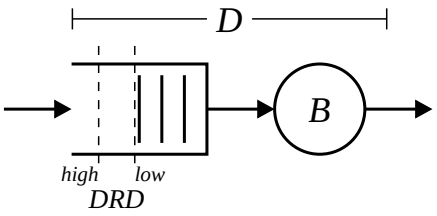
Paketverluste treten entweder nach Festlegung einer Paketverlustwahrscheinlichkeit $p_{Ploss} > 0$ auf, oder können durch einen hohen Pufferfüllstand verursacht werden. Es wird das DRD-Verfahren (*Derivative Random Drop* [Gay96], eine Variante von RED, *Random Early Detection* [FJ93]) eingesetzt: sind mehr als DRD_{high} Pakete im Puffer, so werden neu ankommende mit einer Wahrscheinlichkeit von 95 % verworfen. Ist der Puffer zwischen DRD_{low} und DRD_{high} gefüllt, so wird linear eine Verwurfswahrscheinlichkeit zwischen 5 % und 95 % angenommen (Abb. 4.19 rechts).

Der Vorteil dieses Emulators ist, dass sich die eingestellten Parameter auch im beobachteten Verhalten der Paketverzögerungen wiederfinden lassen. Jedoch sind das Netzmodell und die hierdurch

[5] Bei Angabe von 100 ms beträgt die beobachtete Intervalllänge jedoch etwa 110 ms.

$$D = \max\{d_{BW}, d_{rand}\} \quad (4.11)$$

$$d_{BW} = \frac{l_p + l_{buffer}}{B} \quad (4.12)$$



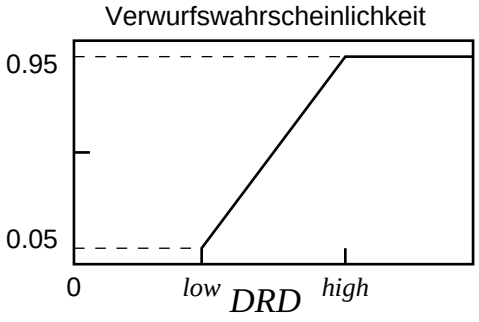


Abbildung 4.19: Netzmodell in *NIST Net*

hervorgerufenen häufigen Paketüberholungen sehr untypisch für heutige Kommunikationsnetze, was bei der Bewertung der Messergebnisse beachtet werden muss[6].

Ein weiterer Vorteil ist, dass der Emulator im Quelltext vorliegt und so nach Belieben angepasst werden kann. Zudem erlaubt das dokumentierte Steuer-Interface die Entwicklung komplexerer Programme zur Einstellung des Emulator-Verhaltens.

4.5.2 Netz-Szenarien

Die Reaktion der Algorithmen auf Änderungen im Verhalten des Netzes ist leichter nachzuvollziehen, wenn man verschiedene Verhaltensweisen getrennt untersucht. Daher wurden zunächst einfachere (pathologische) Fälle betrachtet, um die Algorithmen für diese Fälle zu verbessern. In einer zweiten Phase kann dann die Reaktion dieser Algorithmen in komplexeren Szenarien untersucht werden.

Kriterien für die Wahl der Parameter der Szenarien sind die folgenden:

- Die Bitrate B sollte sich im Bereich gebräuchlicher Zugangstechnologien (z. B. ISDN: 64 oder 128 kbit/s, ADSL: z. B. 768 kbit/s) und der für den Medienstrom üblichen Rate bewegen. Dieser Bereich liegt für Audio-Übertragungen zwischen 17 und 100 kbit/s, für Video-Übertragungen etwa zwischen 500 und 1500 kbit/s.

- Der Mittelwert der Verzögerung μ sollte eine typische Größenordnung besitzen, den Algorithmen jedoch noch Gelegenheit geben, verbessernd einzugreifen. Dies wäre bei Verzögerungen über ca. 200 ms nicht mehr gegeben, daher wurden Mittelwerte zwischen 40 und 100 ms gewählt.

- Die Standardabweichung σ wurde auf ca. 25 % des Mittelwerts der Verzögerung gesetzt. Leider sind kaum Messungen dieses Parameters zu finden, daher wurde ein „interessanter“ Wert gewählt.

- Die Paketverlustwahrscheinlichkeit p_{Ploss} ist nicht der Hauptgegenstand dieser Untersuchung, da hier von leitungsgebundenen Übertragungen ausgegangen wird. Dieser Parameter wird daher meist auf 0 gesetzt und nur in einigen Messungen auf Werte bis zu 20 %.

[6] Die ohnehin notwendige Pufferung der Pakete vor der Verwendung erlaubt es, diesen Effekt weitgehend auszugleichen.

- Die Parameter für die Warteschlangenlänge (RED_{low}, RED_{high} für *The Cloud*, DRD_{low}, DRD_{high} für *NIST Net*) wurden so gewählt, dass die durch die Wartezeit in den Puffern verursachte Verzögerung klein relativ zum Mittelwert der eingestellten Verzögerung ist (s. o.). Leider konnten auch für diese Parameter keine Werte in der Literatur gefunden werden.

- Die Verweildauern in den Netzzuständen d_{high} und d_{low} sollen einen merkbaren Effekt auf die Dienstgüte der Übertragung bewirken und die Agilität der Anpassung bewertbar machen.

4.5.2.1 Konstantes Netzverhalten

Im ersten Schritt wird die Effektivität der Algorithmen geprüft, d. h. wie sie in einem Szenario mit konstantem Netzverhalten die erfahrene Dienstgüte im Vergleich zu einer Übertragung ohne Anpassung verbessern. Insbesondere kann hier der Einfluss verschiedener Parameter auf die Agilität und Stabilität der Algorithmen beurteilt werden, d. h. wie lange der Algorithmus braucht, um sich auf das Netzverhalten, beschrieben mit den Parametern nach Tab. 4.16, einzustellen, und wie gleichmäßig der Algorithmus die Dienstgüte halten kann.

Parameter	Bedeutung	Beispiel
B	Bitrate	100 kbit/s
μ	mittlere Verzögerung	80 ms
σ	Standardabweichung	20 ms
DRD_{low}	untere DRD-Schwelle	5 Pakete
DRD_{high}	obere DRD-Schwelle	10 Pakete
p_{Ploss}	Paketverlustwahrscheinlichkeit	0.02

Tabelle 4.16: Parameter des Szenarios „konstantes Netzverhalten“

***Emulation mit* NIST Net**

Für die Untersuchung der Audio- und Video-Übertragung werden verschiedene Parameter verwandt (Tab. 4.17), da sich die erzeugten Verkehrscharakteristiken dieser Medienströme sehr stark unterscheiden. Die angegebenen Parameter können in *NIST Net* direkt eingestellt werden.

Szenario	B	μ	σ	DRD_{low}	DRD_{high}	p_{Ploss}
1.1	64 kbit/s	40 ms	10 ms	5 Pakete	10 Pakete	0.0
1.2	128 kbit/s	80 ms	20 ms	5 Pakete	10 Pakete	0.0
1.3	64 kbit/s	80 ms	20 ms	5 Pakete	10 Pakete	0.05
1.4	64 kbit/s	80 ms	20 ms	5 Pakete	10 Pakete	0.10
1.5	768 kbit/s	40 ms	10 ms	10 Pakete	20 Pakete	0.0
1.6	768 kbit/s	80 ms	20 ms	10 Pakete	20 Pakete	0.0
1.7	768 kbit/s	40 ms	10 ms	10 Pakete	20 Pakete	0.05
1.8	768 kbit/s	40 ms	10 ms	10 Pakete	20 Pakete	0.10

Tabelle 4.17: Untersuchte Parameter der Szenarien „konstantes Netzverhalten“

Emulation mit The Cloud

In *The Cloud* lassen sich Parameterwerte nur eingeschränkt auswählen. So kann z. B. die Bitrate nicht auf beliebige Werte eingestellt werden, und statt dem DRD-Verfahren wird *Random Early Detection* mit den Werten RED_{low} =8 KB und RED_{high} =16 KB (entspricht im Paket-Modus bei 1500 byte/Paket 5 und 10 Paketen) gewählt. Die Bitrate des „East-Gateway“ wird jedoch auf unbegrenzt *unrestricted* gestellt, so dass effektiv nur ein Puffer aktiv ist – dies erleichtert den Vergleich der Ergebnisse mit der *NIST Net*–Emulation; es sei auf die unterschiedlichen Netzmodelle (s. o.) verwiesen.

4.5.2.2 Verzögerungsänderung

Das Verhalten des Netzes wechselt von einem Zustand mit niedriger Verzögerung und kleinem Jitter („niedrig“) in einen Zustand mit höherer Verzögerung und großem Jitter („hoch“, Abb. 4.20). Dieses Verhalten eines Netzes kann durch plötzlich auftretenden *Querverkehr* erzeugt werden, durch den Pakete in den Warteschlangen der Netzknoten auf eine unterschiedlich hohe Anzahl an Paketen anderer Übertragungen treffen und so unterschiedlich lange verzögert werden.

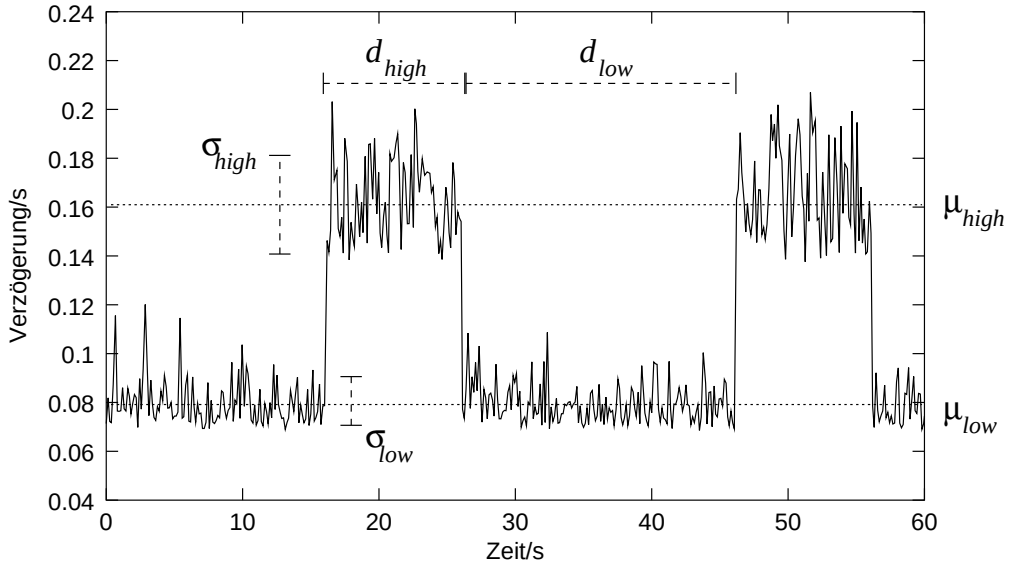


Abbildung 4.20: Beispiel für Szenario „Verzögerungsänderung“

Die Parameter nach Tab. 4.18 bestimmen das Netzverhalten und in Tab. 4.19 werden die Parameter der untersuchten Szenarien aufgeführt. Diese wurden so gewählt, dass zum einen eine zeitweise Beeinträchtigung der Dienstgüte zu erwarten ist (die Werte für die „hoch“–Phase liegen im unteren Bereich von Tab. B.1, S. 152), andererseits die Reaktionsschnelligkeit untersucht werden kann (Dauer der Phasen).

Als Kapazität des Netzes seien hier 2 Mbit/s angenommen, die im Verlauf der Übertragung konstant bleibt. Ebenso sind die Werte für $DRD_{low} = 20$ Pakete und $DRD_{high} = 40$ Pakete konstant. (* In Szenario 2.6 (*NIST Net*) werden im Zustand „hoch“ die Werte $DRD_{low} = 10$ und $DRD_{high} = 20$ eingesetzt.)

Zustand	Parameter	Bedeutung	Beispiel
„niedrig“	μ_{low}	mittlere Verzögerung	80 ms
	σ_{low}	Standardabweichung	10 ms
	d_{low}	Dauer	20 s
„hoch“	μ_{high}	mittlere Verzögerung	160 ms
	σ_{high}	Standardabweichung	20 ms
	d_{high}	Dauer	10 s

Tabelle 4.18: Parameter des Szenarios „Verzögerungsänderung“

Szenario	μ_{low}	σ_{low}	d_{low}	μ_{high}	σ_{high}	d_{high}
2.1	40 ms	10 ms	55 s	200 ms	25 ms	5 s
2.2	40 ms	10 ms	40 s	200 ms	25 ms	20 s
2.3	40 ms	10 ms	40 s	200 ms	25 ms	40 s
2.4	100 ms	20 ms	55 s	400 ms	80 ms	5 s
2.5	100 ms	20 ms	40 s	400 ms	80 ms	20 s
2.6*	100 ms	20 ms	40 s	400 ms	80 ms	20 s

Tabelle 4.19: Untersuchte Parameter im Szenario „Verzögerungsänderung“

Emulation mit NIST Net

Die Parameter nach Tab. 4.19 können mit *NIST Net* über ein *Skript* emuliert werden. Das Netzverhalten entspricht genau den eingestellten Werten, was bedeutet, dass Änderungen der Paketreihenfolge sehr häufig auftreten.

Emulation mit The Cloud

The Cloud erlaubt keine direkte Emulation dieses Verhaltens. Es ist möglich, über eine Datei Verzögerungswerte anzugeben, die von Normalverteilungen mit den gegebenen Parametern erzeugt wurden. Jedoch ist hierbei zu beachten, dass zum einen jeder Wert für mindestens 110 ms gilt, und dass aufgrund des Netzmodells keine Änderungen der Paketreihenfolge auftreten können (s. 4.5.1.1, S. 89). Da die RED–Parameter während des Betriebs des Emulators konstant bleiben müssen, kann Szenario 2.6 nicht emuliert werden.

4.5.2.3 Änderung der Bitrate

Die verfügbare Bitrate im Netz ändert sich von einem Zustand mit relativ hoher Rate zu einem mit geringerer Rate, und umgekehrt. Dieses Verhalten kann z. B. auftreten, wenn von einem Netzmanagement ein Anteil der Bitrate zwischen zwei Knoten fest einem bestimmten Verkehr zugewiesen wird, der dann dem betrachteten *Best-Effort*–Verkehr nicht mehr zur Verfügung steht. Analog kann nach dem Abbau einer Reservierung für den Querverkehr mehr Bitrate für den betrachteten genutzt werden.

Tabelle 4.20 zeigt die Parameter der beiden Zustände, zwischen denen der Netzzustand fluktuiert. Wichtige Parameter sind dabei die Schwellwerte für das DRD-Verfahren (S. 91) in *NIST Net*. Sie bestimmen, wie viele Pakete im emulierten Netz gespeichert werden, wenn die Senderate die

eingestellte Übertragungsrate überschreitet. Sind diese Werte sehr klein, so werden im Netz sehr früh Pakete verworfen. Sind sie jedoch groß, so verzögert sich die Übertragung der Pakete, da sie lange Zeit gespeichert werden.

Zustand	Parameter	Bedeutung	Beispiel
„niedrig“	B_{low}	niedrige verfügbare Bitrate	40 kbit/s
	d_{low}	Dauer	50 s
„hoch“	B_{high}	hohe verfügbare Bitrate	80 kbit/s
	d_{high}	Dauer	50 s
gemeinsam	DRD_{low}	untere DRD-Schwelle	10 Pakete
	DRD_{high}	obere DRD-Schwelle	20 Pakete

Tabelle 4.20: Parameter des Szenarios „Bitratenänderung“

Die Szenarien 3.1 bis 3.3 nach Tab. 4.21 werden für die Untersuchung der Audio-Anpassung verwandt, da diese weniger als 100 kbit/s Bitrate belegt. Dagegen werden die übrigen Szenarien bei Video-Übertragungen eingesetzt, da dieses Medium etwa 1 Mbit/s benötigt.

Szenario	B_{low}	d_{low}	B_{high}	d_{high}	DRD_{low}	DRD_{high}
3.1	64 kbit/s	50 s	96 kbit/s	50 s	5 Pakete	10 Pakete
3.2	64 kbit/s	50 s	96 kbit/s	50 s	20 Pakete	40 Pakete
3.3	32 kbit/s	50 s	96 kbit/s	50 s	5 Pakete	10 Pakete
3.4	768 kbit/s	50 s	1200 kbit/s	50 s	60 Pakete	120 Pakete
3.5	768 kbit/s	50 s	1200 kbit/s	50 s	30 Pakete	60 Pakete
3.6	768 kbit/s	50 s	1200 kbit/s	50 s	5 Pakete	10 Pakete

Tabelle 4.21: Parameter der Szenarien „Änderung der Bitrate“

Emulation mit NIST Net

Die Parameter nach Tab. 4.20 können bei *NIST Net* mittels eines Skripts eingestellt werden, wobei das Programm Angaben in byte/s erwartet. Die Verwendung des DRD-Verfahrens entspricht jedoch nicht exakt dem Verhalten eines Netzes mit festen Puffergrößen, da auch bei hohem Füllstand nicht alle neu ankommenden Pakete verworfen werden.

Emulation mit The Cloud

Eine Emulation der Bitratenänderung ist mit *The Cloud* nicht möglich, da die Zu- und Abflussrate nur auf konstante Werte eingestellt werden können.

4.5.2.4 Verlust-Phasen

Bei diesem Verhalten wechseln Phasen mit keinen oder geringen Paketverlusten mit Phasen hoher Paketverluste ab. Dieses Verhalten kann zum einen durch Querverkehr hervorgerufen werden,

durch den die Pakete des betrachteten Verkehrs auf volle Netz-Puffer treffen und mit höherer Wahrscheinlichkeit verworfen werden. Andererseits kann es durch eine Störung in der Bitübertragungsschicht verursacht werden, z. B. bei drahtlosen Übertragungen[7].

Zustand	Parameter	Bedeutung	Beispiel
„niedrig“	p_{low}	niedrige Verlustwahrscheinlichkeit	0
	d_{low}	Dauer	50 s
„hoch“	p_{high}	hohe Verlustwahrscheinlichkeit	0.1
	d_{high}	Dauer	10 s

Tabelle 4.22: Parameter des Szenarios „Verlust-Phasen“

Wie Tab. 4.22 zu entnehmen ist, wird das Szenario über die Dauer und Häufigkeit der Phasen mit einer erhöhten Verlustwahrscheinlichkeit definiert. Tabelle 4.23 zeigt die untersuchten Parameterkombinationen.

Szenario	p_{low}	d_{low}	p_{high}	d_{high}
4.1	0	50 s	0.02	50 s
4.2	0	50 s	0.05	50 s
4.3	0	50 s	0.10	50 s
4.4	0.05	50 s	0.10	50 s
4.5	0.02	50 s	0.20	50 s

Tabelle 4.23: Parameter der Szenarien „Verlust-Phasen“

Emulation mit NIST Net

Mittels eines Skripts können die Parameter direkt an *NIST Net* übergeben werden, das das gewünschte Verhalten emuliert.

Emulation mit The Cloud

Das gewünschte Verlustverhalten kann in *The Cloud* nur annähernd nachgebildet werden, da hierfür nur die Option *„Congestion“* (Überlastung) zur Verfügung steht. Sie erlaubt, Phasen mit erhöhter Verlustwahrscheinlichkeit zu emulieren, jedoch wird der Abstand zwischen diesen Phasen zufällig gewählt.

4.5.3 Anwendungs-Szenarien

Neben dem Verhalten des Netzes muss auch festgelegt werden, welche Dienstgüte der Nutzer von der Anwendung erwartet und welche Ressourcen für die Multimedia-Kommunikation zur Verfügung stehen.

[7] Bitfehler führen hier zu Paketverlusten.

4.5.3.1 Audio-Kommunikation

Art: Um die Adaptionsalgorithmen für die Audio-Übertragung bewerten zu können, wird zunächst nur ein einzelner Audio-Strom betrachtet. Dieser soll eine gute Dienstgüte erreichen und hohe Interaktivität (d. h. eine kurze Verzögerung) gewährleisten.

Anforderung: Die Kosten der Übertragung sind kein ausschlaggebender Faktor, jedoch soll die Ende-zu-Ende–Verzögerung deutlich unter 200 ms liegen, sowie eine gute Tonqualität (ULQ 7, etwa ADPCM bei 17 kHz) erreicht werden.

Ressourcen: Es werden leistungsstarke Endgeräte verwandt (PC, 350 MHz „Pentium“-Prozessoren, 128 MB RAM), die über eine Zugangsleitung mit bis zu 100 kbit/s an das WAN angeschlossen sind. Die genauen Werte für Zugangskapazität und Verhalten des WANs werden in den Szenarien beschrieben.

4.5.3.2 Video-Kommunikation

Art: Es wird ein einzelner Video-Strom betrachtet, um die Adaptionsalgorithmen bewerten zu können. Die Ziel-Dienstgüte soll „gut“ sein und hohe Interaktivität gewährleisten.

Anforderung: Die Kosten der Übertragung sind kein ausschlaggebender Faktor, jedoch soll die Ende-zu-Ende–Verzögerung deutlich unter 200 ms liegen, sowie eine gute Video-Qualität (ULQ 7, 384x288, 14 fps, $P_Q = 100$) erreicht werden.

Ressourcen: Die Endgeräte sind leistungsstarke PC (s. o.) und die Zugangsleitung zum WAN kann etwa 1 Mbit/s übertragen (abhängig vom Szenario).

4.6 Medienübergreifende Adaptionsalgorithmen

Bei einer *Video-Konferenz* werden mindestens ein Audio- und ein Video-Strom zwischen den Teilnehmern übertragen. Teilen sich die ausgesandten Pakete die Netzressourcen auf ihrem Weg zum Empfänger, so beeinflussen sich die Ströme gegenseitig.

Zudem besteht eine inhaltliche Beziehung zwischen den Strömen, z. B. zwischen dem Bild eines Sprechers und dem Gesagten, so dass eine zeitliche Synchronisierung erfolgen muss. Schließlich ist für die Gesamtdienstgüte der Konferenz eine Gewichtung der Medien erforderlich.

Implementiert wird die Adaption in einer *Meta-Policy* (s. 3.5.1.1, S. 46), die die Ziel-ULQ (*User Level Quality*) eines Medienstroms verändern kann, also die Gewichtung zwischen den Strömen verschiebt. Sie kann außerdem einzelne Ziel-Parameter verändern, insbesondere die Ziel-Werte für die Verzögerung, um die Synchronisierung des Ausspielens der Ströme zu erreichen.

Hier wird ein medienübergreifender Adaptionsalgorithmus skizziert, der zwar implementiert, jedoch aufgrund des Aufwands nicht ausführlich untersucht werden konnte. Am Ende des Kapitels wird ein Eindruck der Leistung dieses Algorithmus gegeben.

4.6.1 Anpassungsmechanismen

In Kapitel 3 wurden zwei Mechanismen vorgestellt, die hier implementiert werden konnten. Die „Auswahl auf Inhaltsebene“ (s. 3.5.1.2, S. 47) steht nicht zur Verfügung, da kein Ersatzmedium für einen Audio- oder Video-Strom vorhanden ist.

4.6.1.1 Änderung der Gewichtung der Medienströme

Die erreichte Gesamtdienstgüte achieved(ULQ) entspricht nicht dem Zielwert target(ULQ).

Ursache:	Für die Medienströme stehen nicht genug Ressourcen zur Verfügung, um beide mit der gewünschten Dienstgüte zu übertragen. Oder die Verteilung der Ressourcen auf die Ströme führt zur Benachteiligung eines Stroms.
Erkennung:	Die erreichten Dienstgüten (Audio.achieved(ULQ) und Video.achieved(ULQ)) beider Ströme liegen unterhalb der Vorgaben. Oder ein Strom erreicht seine Ziel-Dienstgüte und der andere nicht.
Reaktionen:	Die Zielvorgaben der Ströme wird verändert, dergestalt dass sich die Ressourcenaufteilung ändert.
Siehe:	3.5.1.1, S. 46

4.6.1.2 Synchronisation des Ausspielens der Medienströme

Die Ende-zu-Ende–Verzögerung der Medienströme muss aufeinander abgestimmt werden, so dass „Lippensynchronität“ erreicht wird.

Ursache:	Die Verarbeitung und Übertragung der Medienströme dauert unterschiedlich lange.
Erkennung:	Die Audio-Verzögerung Audio.achieved(ALQ/Delay) $\bar{d}_{Aaudio}$ unterscheidet sich von der Video-Verzögerung Video.achieved(ALQ/Delay) $\bar{d}_{Avideo}$ um mehr als die erlaubten Werte $d_{voraus,audio}$ bzw. $d_{voraus,video}$.
Reaktionen:	Die Zielverzögerung des Stroms mit der kleineren Verzögerung wird dem Wert des anderen Stroms angenähert, bzw. der Zielwert des Stroms mit der größeren Verzögerung reduziert.
Siehe:	3.5.2.1, S. 48

4.6.2 Adaptionsalgorithmen

Im ersten (Teil-) Algorithmus (Abb. 4.21) wird die Dienstgüte des Audio- und Video-Stroms überwacht. Ist die Gesamtdienstgüte niedriger als der Zielwert und die erreichte Dienstgüte des Video-Stroms größer oder gleich der des Audio-Stroms ([Audio <= Video]), so wird die Ziel-Dienstgüte des Video-Stroms reduziert (reduziere Video). Erreichen die Medien die reduzierten Zielvorgaben ([Ziel-ULQ erreicht]), so wird versucht, die Ziel-Dienstgüte wieder langsam auf den globalen Zielwert zu erhöhen, wobei bei einer Verschlechterung wieder reduziert wird. Es fällt auf, dass keine Reaktion auf eine schlechte Video-Dienstgüte erfolgt. Dies liegt in der Annahme begründet, dass die Audio-Übertragung wichtiger für die Gesamt-Dienstgüte ist, als die Video-Übertragung. (Der Video-QoS-Manager wird ständig versuchen, die Ziel-Dienstgüte zu erreichen.)

Um das Ausspielen des Audio- und Video-Stroms zu synchronisieren, werden im zweiten (Teil-) Algorithmus die Ende-zu-Ende–Verzögerungen überwacht (Abb. 4.22). Geht das Ausspielen des Audio-Stroms dem des Video-Stroms voraus ([Audio vor Video]), so wird der Audio-Strom stärker verzögert (verzögere Audio), um die Differenz zu minimieren. Geht der Video-Strom dem Audio-Strom voraus ([Video vor Audio]), so wird dieser stärker verzögert (verzögere Video). Wird eine ausreichende Synchronität zwischen den Strömen erreicht ([synchron]), so kann in den Normalzustand (Ausspielen synchron) übergegangen werden. Die Bedingungen und Parameter werden in Tab. 4.24 und Tab. 4.25 aufgeführt.

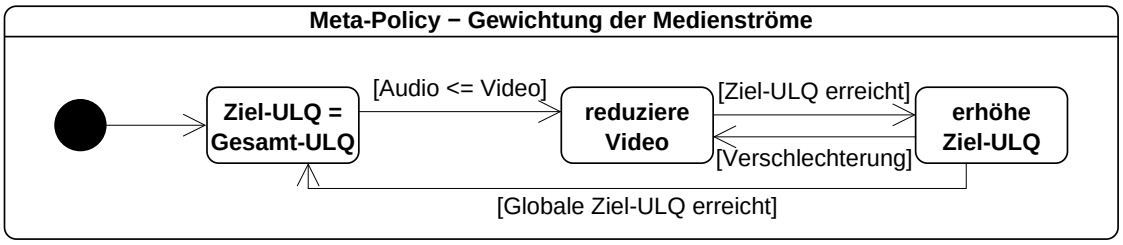


Abbildung 4.21: Zustandsdiagramm zur Änderung der Gewichte der Medien

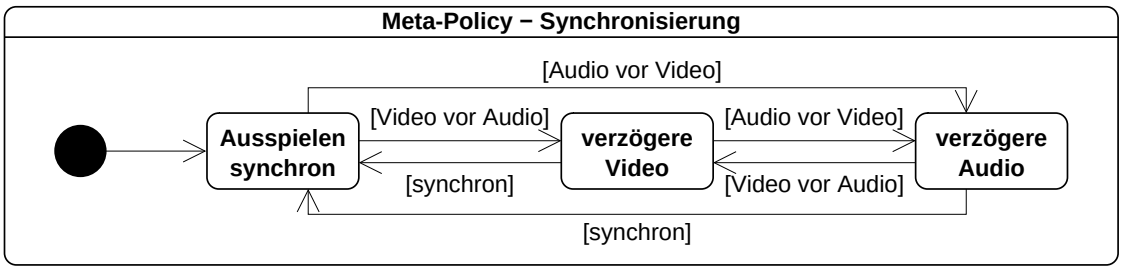


Abbildung 4.22: Zustandsdiagramm zur Synchronisierung der Medien

Übergang	Bedingung
[Audio <= Video]	achieved(ULQ) $<$ target(ULQ)
	$\wedge$ Audio.achieved(ULQ) $\leq$ Video.achieved(ULQ)
[Ziel-ULQ erreicht]	Audio.achieved(ULQ) = Audio.target(ULQ)
[Verschlechterung]	Audio.achieved(ULQ) $<$ Audio.target(ULQ)
[Globale Ziel-ULQ erreicht]	achieved(ULQ) = target(ULQ)
[Audio vor Video]	$\bar{d}_{Avideo} - \bar{d}_{Aaudio} > d_{voraus,audio}$
[Video vor Audio]	$\bar{d}_{Aaudio} - \bar{d}_{Avideo} > d_{voraus,video}$
[synchron]	$-d_{voraus,video} \leq \bar{d}_{Aaudio} - \bar{d}_{Avideo} \leq d_{voraus,video}$

Tabelle 4.24: Übergangsbedingungen Meta-Policy

Bedeutung	Symbol	Beispiel
Maximal erlaubter Vorlauf des Audio-Stroms	$d_{voraus,audio}$	80 ms
Maximal erlaubter Vorlauf des Video-Stroms	$d_{voraus,video}$	120 ms

Tabelle 4.25: Parameter der Meta-Policy

4.6.3 Eindruck von der Leistung des Algorithmus

Die Durchführung einiger Experimente zeigt, dass die gegenseitige Beeinflussung der Medienströme sehr groß ist. Insbesondere führt der Anstieg der Bitrate des Video-Stroms durch eine Änderung des Videosignals zu Verlusten im Audio-Strom, die dessen Dienstgüte reduziert. Die Adaption und die Implementierung des Video-Mediums müsste dergestalt verändert werden, dass die Bitrate des Video-Stroms einen zugewiesenen Wert nicht überschreiten kann.

5 Bewertung

Dieses Kapitel erläutert zunächst die Kriterien, nach denen die Leistung der Adaptionsalgorithmen für die betrachteten Szenarien bewertet wurde. Die einzelnen Messergebnisse der experimentellen Parameterstudien wurden aus Gründen der Übersichtlichkeit in Anhang A, S. 115 ausgelagert. In diesem Kapitel erfolgt die Zusammenfassung der erzielten Erkenntnisse und eine Bewertung der Algorithmen in den verschiedenen Szenarien. Abschließend wird ein Fazit und eine Empfehlung ausgesprochen.

5.1 Kriterien zur Bewertung der Algorithmen

Um die Leistung der verschiedenen Adaptionsalgorithmen für ein Szenario bewerten zu können, müssen zunächst während der Experimente die Dienstgüten-Parameter erfasst werden. Hierzu wird an „Messpunkten“ der zeitliche Verlauf bestimmter Messgrößen aufgezeichnet. So wird z. B. in der Netz-Schicht die gemessene Bitrate, in der Anwendungs-Schicht die erfahrene Ende-zu-Ende–Verzögerung und in der Nutzer-Schicht die *User Level Quality* protokolliert.

Aus den Protokollen werden Kennzahlen gebildet, die das Verhalten des Algorithmus im konkreten Experiment beschreiben. Anhand dieser Werte lässt sich dann die Leistung der Algorithmen vergleichen. An einem Messpunkt X werden Messwerte x_i erfasst (s. B.5, S. 149), wobei die Folge dieser Werte für die Verwendung in einem Adaptionsalgorithmus zum Zeitpunkt t in einen einzelnen Wert $X(t)$ transformiert werden muss. Mittels eines Messpunkts vom Typ XP_CMeasureEvents lassen sich die n Messwerte $x_1,\ldots,x_n$ des Beobachtungszeitraums $[t-d_S,t]$ mitteln (Gl. 5.1, links), summieren (Gl. 5.1, Mitte) oder auch in Einheiten pro Sekunde (Gl. 5.1, rechts) angeben (Abb. 5.1). XP_CMeasureData–Messpunkte „glätten“ die j seit dem Start der Messung erfassten Werte nach Gl. 5.2 (vgl. RTP, RFC 1889/A.8).

$$X(t)=\frac{1}{n}\sum_{i=1}^{n}x_i \quad \text{oder} \quad X(t)=\sum_{i=1}^{n}x_i \quad \text{oder} \quad X(t)=\frac{1}{d_S}\sum_{i=1}^{n}x_i \tag{5.1}$$

$$X(t):=\bar{x}_j \quad \text{mit} \quad \bar{x}_1=x_1 \quad \text{und} \quad \bar{x}_{j+1}=\bar{x}_j+\frac{x_{j+1}-\bar{x}_j}{16} \quad j\geq 1 \tag{5.2}$$

Je nach Modul und Art des Messpunkts können sich die zeitlichen Abstände der Messwerte unterscheiden. Wird z. B. ein Messpunkt beim Eintreffen eines Pakets aktiviert, so unterliegen die Zeitpunkte der gleichen Verzögerungsschwankung wie die Pakete. Es wird in Berechnungen für $X(t)$ der zuletzt erfasste Wert verwandt (s. u.).

Für Parameter, die das System aus eigener Veranlassung regelmäßig erhebt (z. B. die erreichte Nutzer-Dienstgüte *ULQ*), wird ein Abtast-Abstand $d_S(X)$ („*sample distance*“) verwandt.

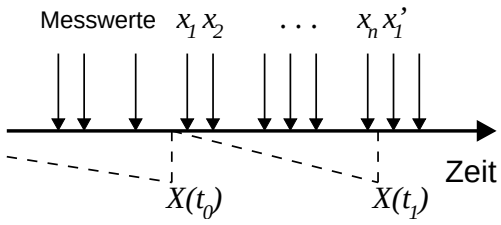


Abbildung 5.1: Transformation einer Messwertfolge

Für die Leistungsbewertung eines Algorithmus werden Kennzahlen nach Tab. 5.1 definiert. Diese werden bei der Durchführung der Experimente eingestellt, erhoben oder nach Tab. 5.2 berechnet.

Kennzahl	Bedeutung
$X(t)$	Wert am Messpunkt X zum Zeitpunkt t
n	Anzahl der Messwerte
d_S	Abstand zwischen QoS-Erfassungen
d_E	Dauer des Experiments E (mit Startpunkt $t = 0$)
$\overline{ULQ}$	Durchschnittliche beobachtete Nutzer-Dienstgüte
$\widetilde{ULQ}$	Medianwert der beobachteten Nutzer-Dienstgüte
ΔULQ	Abstand zwischen maximaler und minimaler Dienstgüte

Tabelle 5.1: Kennzahlen für Leistungsbewertung

Kennzahl		Definition
$X(t)$	$=$	$X\left(\lfloor \frac{t}{d_S} \rfloor d_S\right)$
n	$=$	$\lfloor \frac{d_E}{d_S} \rfloor$
$\overline{ULQ}$	$=$	$\frac{1}{n}\sum_{i=1}^{n} ULQ(i \cdot d_S)$
ΔULQ	$=$	$\max\{ULQ(t)\} - \min\{ULQ(t)\} \quad \text{mit} \quad 0 < t \leq d_E$

Tabelle 5.2: Kennzahlendefinition für Leistungsbewertung

5.2 Audio-Übertragung

Zunächst wird die Übertragung eines Audio-Stroms betrachtet. Der Sender erzeugt eine Folge von Audio-Rahmen und schickt sie in Paketen an den Empfänger, der sie ausspielt (s. 4.3, S. 70). Besonders störend sind beim Ausspielen auftretende Pausen und hörbare Signalsprünge („Knackser“), was auch in die Dienstgütenberechnung (Gl. 4.2, S. 72) eingeht.

Aus der Untersuchung der Auswirkungen des Adaptionsalgorithmus auf die Dienstgüte in den verschiedenen Szenarien (s. A.2.1, S. 115) können folgende Aussagen über die Effizienz der Anpassung getroffen werden.

Der Algorithmus kann sich schnell an die verfügbare Netz-Bitrate anpassen, indem ein anderer Audio-Typ eingestellt wird (Szenarien 1.1 bis 1.4). Dies reduziert die Zahl der vom Netz verworfenen Pakete und somit auch die Ausspielpausen im Vergleich zur Übertragung ohne Regelung deutlich (S. 116).

Problematisch ist für die Audio-Übertragung weniger die absolute Netz-Verzögerung als die Verzögerungsschwankung (S. 117), falls sie wie im Falle von *NIST Net* zu Änderungen in der Paketreihenfolge führt. Der Algorithmus kann durch die Einstellung einer entsprechenden Pufferverzögerung diese Umordnungen ausgleichen, Ausspielpausen vermeiden und so die erreichte Dienstgüte erhöhen.

Da der Audio-Strom sehr empfindlich auf Paketverluste reagiert (bereits 1 % führen zu störenden Knacksern), setzt der Algorithmus Audio-Typen mit Vorwärts-Fehlerkorrektur (FEC) ein, sobald er annimmt, dass diese Verluste unabhängig von der gesandten Bitrate auftreten. Da die FEC-Typen bei gleicher Bitrate eine deutlich geringere Qualität aufweisen, wird erst nach dem Fehlschlagen des Versuchs, durch Reduktion der Bitrate die Verluste zu minimieren, auf einen FEC-Typ umgeschaltet (S. 128).

In den Szenarien mit variabler Verzögerung (2.1 bis 2.6) zeigt sich, dass auch ohne Regelung die Dienstgüte nicht stark abnimmt, solange bestimmte Werte nicht überschritten werden (s. A.2.2, S. 121). Da hier die Audio-Rahmen im Abstand von etwa 40 ms erzeugt werden, kommt es bei einem Wert von 10 ms für die Standardabweichung der Verzögerung selten zu Überholungen und daher müssen auch wenige Rahmen verworfen werden, weil ein jüngerer bereits ausgespielt wurde. Beträgt die Standardabweichung jedoch 25 ms, so sind Paketumordnungen sehr viel häufiger, wodurch Rahmen nicht ausgespielt werden können und Ausspielpausen entstehen. Die Regelung sorgt hier für eine deutliche Verbesserung der Dienstgüte, jedoch um den Preis erhöhter Ende-zu-Ende–Verzögerung.

Bei diesen Szenarien wird der Einfluss des WAN-Emulators auf die Messung deutlich. Während das einfache Netzmodell von *NIST Net* Paketüberholungen zulässt, ist dies im Modell von *The Cloud* nicht vorgesehen. Daher wird die Übertragung durch die Änderung der Netz-Verzögerung nicht beeinflusst, denn der Puffer auf der Sound-Karte füllt sich und gleicht so die Schwankungen aus. Nur die sehr hohe Ende-zu-Ende–Verzögerung von über 0.5 s beeinträchtigt die Dienstgüte. Die Regelung kann hier die Dienstgüte deutlich verbessern, indem sie den Füllstand des Puffers reduziert.

Ändert sich die im Netz verfügbare Bitrate (Szenarien 3.1 bis 3.3), so hat das ohne Adaption zur Folge, dass bei hoher Rate die gewünschte Dienstgüte erreicht wird, jedoch bei niedriger Rate sehr viele Pakete verloren gehen. Der Wert der Dienstgüte springt hier zwischen zwei Extremen hin und her. Da ein Adaptionsalgorithmus die Information, welche Bitrate das Netz gerade übertragen kann, nicht direkt erfahren kann, bleibt nur die Möglichkeit, durch testweise Erhöhung der Senderate eine Verbesserung der Qualität zu erreichen. Da dies aber bei Misserfolg zu deutlichen Störungen führt, wurde hier eine konservative Strategie verfolgt (s. A.2.3, S. 125), d. h. der Algorithmus erhöht den zeitlichen Abstand zwischen Verbesserungsversuchen, sobald ein solcher fehlschlägt. Hierdurch ist es nicht möglich, die erhöhte Bitrate voll auszunutzen. Ebenso wird beim Auftreten von Verlusten nicht sofort die Audio-Qualität reduziert, da es sich um eine kurzfristige Beeinträchtigung handeln könnte. Dies hat insgesamt zur Folge, dass die Verbesserung durch Adaption in diesem Szenario nur gering gegenüber einer Übertragung ohne Regelung ist.

Schließlich verwirft in den Szenarien 4.1 bis 4.5 die WAN-Emulation zufällig Pakete. Ohne Adaption kommt es häufig zu Ausspielpausen, die die Dienstgüte reduzieren. Die Adaption sorgt dage-

gen dafür, dass Audio-Typen mit Vorwärts-Fehlerkorrektur eingesetzt werden, wodurch deutlich weniger Ausspielpausen auftreten. Hier gilt jedoch wieder das zuvor gesagte: Da der Algorithmus nicht über den Netzzustand informiert ist, schaltet er nicht sofort bei Auftreten von Verlusten auf einen anderen Audio-Typ um. Beim Vergleich der Szenarien fällt außerdem auf, dass der Wert der Dienstgüte für 4.1 bis 4.3 deutlich kleiner ist als bei 4.4 und 4.5 (Tab. A.6, S. 130), obwohl bei letzteren stets Paketverluste auftreten. Der Grund liegt in der Tatsache, dass bei ständig auftretendem Verlust der Algorithmus immer einen FEC-Audio-Typ auswählt und nicht auf einen Audio-Typ ohne Redundanz umschaltet. Da im Netz keine Begrenzung der Bitrate vorliegt[1], kann der beste FEC-Audio-Typ eingesetzt werden, wodurch im Vergleich der Szenarien ein deutlich höherer Ressourcenbedarf entsteht (ca. 160 kbit/s statt 70 kbit/s), dies spiegelt der ULQ–Wert jedoch nicht wieder.

Insgesamt lässt sich feststellen, dass die Adaption des Audio-Stroms an das Netzverhalten meist zu einer deutlichen Verbesserung der erfahrenen Dienstgüte führt, wobei jedoch je nach Art des Netzverhaltens die Verbesserung unterschiedlich deutlich ausfällt. Besonders effektiv ist die Anpassung beim Auftreten von Paketüberholungen und wenn der Zustand des Netzes sich nicht zu häufig ändert. Schwierigkeiten hat der Algorithmus vor allem bei der Erkennung der Erhöhung der verfügbaren Bitrate. Offensichtlich könnte hier zusätzliche Information über den Netzzustand die Leistung der Anpassung weiter verbessern.

5.3 Video-Übertragung

Die digitalisierten Bilder werden in der Standardeinstellung blockweise übertragen (s. 4.4.1, S. 80). Der Verlust eines Pakets führt zum Verlust eines Bildausschnitts (einige Blöcke des Bilds), d. h. es treten sog. Artefakte auf, die die erfahrene Dienstgüte stören (s. a. Gl. 4.7, S. 83). Da der Kompressionsfaktor des verwendeten Kodierverfahrens vom Bildinhalt abhängt, schwankt bereits im Sender die Bitrate des Video-Stroms. Der Adaptionsalgorithmus kann sie durch Festlegung einer maximalen Senderate dergestalt steuern, dass die Bildrate (*frames per second*, fps) reduziert wird, um den gewünschten Wert nicht zu überschreiten.

Daher ist es zunächst notwendig, mit Hilfe der Szenarien konstanten Netzverhaltens 1.5 bis 1.8 (s. A.3.1, S. 131) die Effektivität dieser Regelung zu prüfen. Hierbei zeigt sich, dass der Algorithmus einige Zeit braucht, bis er nach einer Änderung des Bildinhalts die Parameter (Senderate und Bildqualität) so angepasst hat, dass nur die verfügbare Bitrate genutzt wird. Die erreichte Dienstgüte erhöht sich im Vergleich zur Übertragung ohne Regelung deutlich, jedoch kann die Anpassung nicht so effektiv sein wie bei der Audio-Übertragung, da die Senderate weder konstant noch vorhersagbar ist.

Der Vergleich der Übertragung ohne Regelung mit der Übertragung mit aktivem Adaptionsalgorithmus zeigt, dass in erstem Fall die Bildrate stets den Zielwert erreicht, jedoch durch die Bitratenbegrenzung im Netz hohe Paketverluste und somit auch sehr viele Bildstörungen auftreten. Durch die Begrenzung der Senderate reduziert sich bereits die Bildrate bei der Erzeugung im Sender und es treten weniger Paketverluste auf, was die Verbesserung des Dienstgütenwerts verursacht.

Auffallend an der Übertragung in den Szenarien 2.1 bis 2.6 (s. A.3.2, S. 136) mit variabler Verzögerung ist, wie wenig sich die Dienstgüte auch ohne Regelung reduziert. Dies liegt an der hohen Fehlertoleranz der Implementierung der Video-Übertragung: Paketumordnungen können nur dann zum Verwurf von Bildblöcken führen, wenn Pakete verschiedener Bildrahmen vermischt

[1] Die verfügbare Rate ist sehr viel größer als die Rate des Audio-Stroms.

werden. Da aber bereits der Abstand der gesandten Bilder ca. 70 ms beträgt (bei einer Bildrate von 14 fps und sehr hoher Senderate), ist die Wahrscheinlichkeit solcher Vermischungen selbst bei einer Standardabweichung der Verzögerung von 25 ms klein, so dass die meisten Bildblöcke angezeigt werden können. Daher kann der Adaptionsalgorithmus die erfahrene Dienstgüte nicht sehr stark verbessern, zumal die plötzliche Erhöhung der Netz-Verzögerung zu Paketverlusten führt, weil die Pufferkapazität des emulierten Netzes überschritten wird. Durch die Regelung der Pufferverzögerung wird zudem die Ende-zu-Ende–Verzögerung größer als ohne Regelung, was den Wert der Dienstgüte weiter reduziert.

Die unterschiedlichen Netzmodelle von *NIST Net* und *The Cloud* wirken sich hier ebenfalls auf die Menge der verlorenen Pakete und verworfenen Blöcke aus, dergestalt dass bei *The Cloud* die Beeinträchtigung der erfahrenen Dienstgüte deutlich geringer ist als bei *NIST Net* (s. A.3.2.2, S. 137). Durch das Einhalten der Paketreihenfolge können sich die Pakete verschiedener Bilder nicht mischen, so dass alle ankommenden Blöcke angezeigt werden. Daher kann hier der Adaptionsalgorithmus auch kaum zu einer Verbesserung der Dienstgüte beitragen, stattdessen können die anfänglichen Versuche, die Übertragung zu verbessern, sogar zu einem im Durchschnitt leicht schlechteren Dienstgütenwert führen.

Erhöht sich die im Netz verfügbare Bitrate (Szenarien 3.4 bis 3.6, S. 140), so führt dies ohne Regelung zu einer Reduktion der Paketverluste und somit auch der Bildstörungen, d. h. insgesamt ergibt sich ein leicht erhöhter Dienstgütenwert im Vergleich zum Szenario 1.5. Der Algorithmus versucht andererseits, die erhöhte Bitrate zu nutzen und erhöht die Senderate, so dass bei der folgenden Reduktion der Bitrate die Paketverluste wieder zunehmen. Daher ist der Wert der Dienstgüte auch praktisch gleich dem im Szenario 1.5, die Verbesserungen und Verschlechterungen heben sich in etwa auf. Interessant ist hier zudem der Einfluss der Größe des Netzpuffers auf die Dienstgüte. Kann das Netz sehr viele Pakete zwischenspeichern, so werden zwar weniger Pakete verworfen, jedoch führt dies auch zu einer großen Ende-zu-Ende–Verzögerung, die die Dienstgüte beeinträchtigt. Bei kleinen Puffergrößen gehen im Netz zwar mehr Pakete verloren, jedoch kann der Adaptionsalgorithmus schneller auf Änderungen reagieren und die Ende-zu-Ende–Verzögerung ist kleiner, so dass dies insgesamt zu einer besseren Dienstgüte führt.

In der letzten Szenarien-Gruppe 4.1 bis 4.5 (s. A.3.4, S. 143) wird die Paketverlustwahrscheinlichkeit im emulierten Netz variiert. Niedrige Werte haben jedoch auch ohne Regelung nur eine geringe Auswirkung auf den Wert der Dienstgüte, da verlorene Pakete nur unmittelbar zu Bildstörungen führen. Der Adaptionsalgorithmus kann daher die Dienstgüte nur wenig verbessern, bzw. gerade erreichen, wenn er versucht, den der Situation entsprechenden Übertragungsmodus (blockweise oder FEC) einzustellen. Bei großen Verlusten wird die Verbesserung deutlicher (Szenario 4.4, S. 143), denn der Algorithmus setzt hier nach kurzer Zeit nur noch Rahmen mit Vorwärts-Fehlerkorrektur ein, wodurch zwar die Bildrate bei hohen Verlusten etwas absinkt, jedoch keine Bildstörungen mehr auftreten. Dies wird, ähnlich wie bei der Audio-Übertragung, durch eine deutlich erhöhte Bitrate erkauft, was in die Berechnung des Dienstgütenwertes aber nicht einfließt.

Im Gegensatz zur Audio-Übertragung kann bei der Video-Übertragung der Adaptionsalgorithmus die erfahrene Dienstgüte nicht immer deutlich verbessern. Dies liegt an der sehr viel fehlertoleranteren Implementierung, bei der selbst hohe Paketverlustwahrscheinlichkeiten und große Verzögerungsschwankungen vergleichsweise geringe Beeinträchtigungen der Dienstgüte zur Folge haben. Zudem erfordert die Schwankung der Bitrate des generierten Video-Stroms bereits eine Regelung der Kodierungsparameter, die durch die Überlagerung mit den vom Netz erzeugten Effekten erschwert wird. Dennoch kann der Adaptionsalgorithmus eine effizientere Nutzung der vorhandenen Bitrate und eine stetigere Dienstgüte erreichen.

5.4 Fazit

Es ist offensichtlich, dass zusätzliche Information über den Zustand des Netzes, bzw. den Aufbau der Übertragungsstrecke, die Leistung der Algorithmen verbessert, da dann schneller geeignete Steuerparameter gefunden werden könnten. Dennoch hat die Untersuchung des Einsatzes der Adaptionsalgorithmen gezeigt, dass eine deutliche Verbesserung der Dienstgüte in den meisten Fällen auch ohne solche Kenntnisse möglich ist. Hierbei kommt es jedoch auf die Eigenschaften des übertragenen Medienstroms an. Reagiert dieser (wie die Audio-Übertragung) sehr empfindlich auf Paketverluste, Verzögerungsschwankungen und Paketüberholungen, so kann der Algorithmus größere Verbesserungen erreichen, als bei Strömen, die bereits sehr fehlertolerant sind (z. B. die vorgestellte Video-Übertragung).

Es wurde gezeigt, wie sich die verschiedenen Effekte des Netzes auf die Dienstgüte auswirken, insbesondere bei Verwendung der WAN-Emulationen *NIST Net* und *The Cloud*, deren Eigenschaften große Auswirkungen auf die Ergebnisse haben. Ebenso wurde deutlich, wie sehr die Berechnung der Dienstgüte auf die Beurteilung der Leistung der Algorithmen einwirkt, da z. B. wünschenswerte Eigenschaften wie ein möglichst kleiner Bitratenbedarf nicht einfließen.

Für die Verwendung dieser Ergebnisse in konkreten Anwendungen wäre es jedoch sehr hilfreich, das typische Verhalten eines realen IP-WANs zu kennen. Hier wären insbesondere die Speicherkapazität des Netzes, die Größenordnung der Verzögerungsschwankung und die Häufigkeit von Paketüberholungen (in Abhängigkeit vom Abstand der Paketsendezeitpunkte) interessant.

6 Zusammenfassung und Ausblick

In der vorliegenden Arbeit wurde untersucht, wie die Dienstgüte einer interaktiven Multimedia-Kommunikationsanwendung durch Anpassung an das Übertragungsverhalten eines Weitverkehr-Datennetzes verbessert werden kann. Hierzu wurden in Kapitel 2 die Grundlagen der Datenkommunikation über paketvermittelnde Netze und von Multimedia-Anwendungen beschrieben. Der Begriff der Dienstgüte wurde eingeführt und der prinzipielle Ablauf einer Multimedia-Anpassung vorgestellt. Das Kapitel schloss mit einer Abgrenzung zu bekannten Anwendungen und Frameworks. Kapitel 3 stellte die Möglichkeiten der Anpassung im Detail vor, indem zunächst eine Klassifizierung erfolgte und dann eine Untersuchung von 14 Mechanismen. In Kapitel 4 wurde die experimentelle Untersuchung erläutert, die mit der „EXPERIMENTATION PLATFORM“ durchgeführt wurde, und wie bestimmte Netzverhaltensweisen mit Hilfe zweier WAN-Emulatoren nachgebildet wurden. Kapitel 5 fasste die durch die umfangreichen Experimente (Anhang A) gewonnenen Erkenntnisse zusammen.

Beim Einsatz interaktiver Multimedia-Anwendungen über Netze, die keine Dienstgütengarantien geben können, kann die vom Nutzer erfahrene Dienstgüte stark schwanken. In dieser Arbeit wurde untersucht, wie sich das Verhalten des Netzes auf die beobachtete Dienstgüte auswirkt, insbesondere wie Änderungen der Bitrate, Verzögerung und Paketverlustwahrscheinlichkeit die Medienströme beeinflussen. Es wurde gezeigt, dass eine Anwendung das Netzverhalten durch Beobachtung des empfangenen Verkehrs gut abschätzen und durch Adaption die erfahrene Dienstgüte deutlich verbessern kann. Mit der EXPERIMENTATION PLATFORM steht nun ein Grundgerüst für die Untersuchung weiterer Typen von Multimedia-Anwendungen zur Verfügung. Die hier entwickelten Adaptionsalgorithmen und Anpassungsmechanismen können aber auch für konkrete Anwendungen übernommen werden.

Durch Anpassung waren bei der Audio-Übertragung sehr große Verbesserungen der Dienstgüte erreichbar, da sie sehr empfindlich auf das Verhalten des Netzes reagiert. Dagegen fiel die Verbesserung der viel robusteren Video-Übertragung deutlich kleiner aus. Zudem erschwerten Eigenschaften der Implementation (insbesondere die variable Sende-Bitrate) die Regelung des Video-Stroms.

Ebenso zeigte sich, dass die Art der Emulation des Netz-Verhaltens sehr große Auswirkungen auf die beobachtete Dienstgüte hat. Die Netzmodelle der beiden eingesetzten Programme *NIST Net* und *The Cloud* unterscheiden sich stark, was sich vor allem auf die beobachtete Verzögerung und die Zahl der Paketumordnungen auswirkt. Insbesondere für die Audio-Übertragung führt dies zu Unterschieden in den Ergebnissen.

Es wurde zudem deutlich, dass die Pufferkapazität des Netzes, also wie viele Pakete vom Netz zwischengespeichert werden, große Auswirkungen auf die Anpassungsfähigkeit der Anwendung hat, wobei es jedoch kaum Angaben über die Größe dieses Parameters in realen Netzen gibt. Ebenso fehlen Aussagen über weitere wichtige Parameter wie den Verzögerungsjitter und die Wahrscheinlichkeit von Paketüberholungen – die häufige Annahme, dass diese vernachlässigbar selten sind,

wird z. B. in [BPS99] bezweifelt. Sofern die Werte dieser Parameter für relevante Weitverkehrsnetze vorliegen, sollten sie mit den hier untersuchten Wertebereichen in Beziehung gesetzt werden. Eine Durchführung der Experimente mit anderen Parametern ist mit kleinem Aufwand möglich und könnte Aussagen für einen konkreten Anwendungsfall liefern.

Während die Implementierung des Audio-Mediums große Ähnlichkeiten mit denjenigen anderer Anwendungen aufweist, gilt dies nicht im selben Maße für das Video-Medium. Daher wäre es sinnvoll, Standard-Video-Kodierer in die EXPERIMENTATION PLATFORM zu integrieren und Adaptionsalgorithmen für sie zu entwickeln. Dies wird durch die modulare Struktur des Systems sehr erleichtert.

Die Beobachtung der Anwendung und des Netzes (bzw. der Netzemulation) während der Übertragung erlaubt es nun, realistische Simulationsmodelle zu entwickeln, mit deren Hilfe die experimentell gewonnenen Erkenntnisse verifiziert werden könnte.[1]

[1] Überlegungen in dieser Richtung sind in Anhang C, S. 157 zu finden.

Glossar

Abtastfrequenz (engl. *sampling frequency*): Rate, mit der ein analoges Signal abgetastet wird.

(Adaptions-) Algorithmus: Rechenvorschrift, die die Zielparameter so ändert, dass die erfahrene Dienstgüte sich der erwarteten Dienstgüte nähert. Eingabewerte sind dabei die Ziel- und Zustandsparameter; ausgegeben werden neue Zielparameter.

***Application Sharing*:** Technik, durch die eine lokale Anwendung von mehreren verteilten Terminals aus bedient werden kann. Die Bedienungskontrolle kann von dem Rechner, auf dem die Anwendung läuft, an einen anderen übergeben werden, wobei die Ein- und Ausgabe übertragen wird.

Ausgabeparameter: Neue Zielparameter, die ein Adaptionsalgorithmus aus den alten Ziel- und Zustandsparametern ermittelt.

***Backbone*-Netz („Rückgrat"):** Datennetz, das über Zugangsnetze erreichbar ist und über das große Datenmengen transportiert werden können.

***Best-Effort-Service*:** Dienst, der keine Garantie über seine Leistung abgibt. Je nach momentanem Zustand kann die Leistung schwanken.

***Client*:** Terminal, das mit einem *Server* kommuniziert, diesem Anforderungen (z. B. eine Datenbankabfrage) übermittelt und von ihm Daten erhält (die angeforderten Daten).

Dienstgüte (engl. *Quality of Service, QoS*): (s. a. Definition nach DIN)
Definition der ITG [ITG97]: „Gesamtheit der Qualitätsmerkmale (...) aus der Sicht der Benutzer eines betrachteten Dienstes."

Endgerät (engl. *terminal*): Gerät, das von einem menschlichen Benutzer bedient wird. Typischerweise umfasst es Ein- und Ausgabeeinrichtungen wie Bildschirm, Tastatur, Maus, Lautsprecher, Mikrofon und Kamera. Es kann sich dabei aber auch um ein spezialisiertes Gerät wie ein Telefon oder eine „Set-Top-Box" handeln.

Erlaubte Verzögerung (engl. *target delay*): Maximale Zeitdauer, die zwischen Erzeugung und Ausspielen eines Datenobjekts liegen darf, um einen interaktiven Dienst zu gewährleisten. Daten, die nur später ausgespielt werden können, müssen verworfen werden.

***Frame-Grabber*:** Gerät zur Digitalisierung von Video-Bildern. Ein angelegtes Videosignal wird abgetastet und jedes Bild digital im Speicher des Rechners abgelegt. Die meisten Geräte sind sehr flexibel hinsichtlich der einstellbaren Bildgröße und Kodierung.

***Framework*:** Bibliothek, die Dienstfunktionen zur Unterstützung der Anwendungserstellung anbietet.

Garantie (engl. *guarantee*): Zusicherung, eine vereinbarte Leistung zu erbringen. Ein Übertragungsnetz kann eine bestimmte Dienstgüte für den transportierten Verkehr garantieren.

Gesamtdienstgüte: Dienstgüte der Kombination der übertragenen Medienströme. Abhängig vom Szenario werden die einzelnen Ströme unterschiedlich gewichtet und ein Wert gebildet, der die Leistung der gesamten Anwendung beschreibt.

Kodierung: Wandlung eines Datenobjekts in eine andere (meist numerische) Darstellung, die bestimmte Vorteile hat. So kann eine Kodierung als Anpassung an eine bestimmte Hardware notwendig sein. Ein Form der Kodierung ist die *Komprimierung*.

Komponente: Abgeschlossener Teil eines größeren Ganzen, der durch ein schnittstellengleiches Objekt selben Typs ausgetauscht werden kann. Beispiele sind Module in einer Anwendung, aber auch einzelne Anwendungen in einem komplexen Szenario.

Komprimierung (verlustlos/-behaftet): Reduktion des Umfangs von Daten. Bei der verlustlosen Komprimierung kann die ursprüngliche Information exakt wieder hergestellt werden (Beispiel: Huffman–Algorithmus [LH87]). Bei der verlustbehafteten Komprimierung wird abhängig von der Art der Daten ein Teil der Information herausgefiltert, der als nicht unbedingt erforderlich angesehen wird (Beispiel: JPEG-Kodierung von Bildern [JPE]).

Mechanismus: Grund-Verhalten einer Komponente eines Mediums. Adaptionsalgorithmen können Mechanismen einsetzen, austauschen oder ihre Parameter verändern, um das Verhalten eines Mediums zu beeinflussen, um z. B. Ressourcen einzusparen. Beispiele sind die Vorwärts-Fehlerkorrektur, aber auch das Ändern der Größe eines Videobilds oder die Veränderung eines Pufferparameters.

Mediendienstgüte: Dienstgüte genau eines Medienstroms. Dieser Wert ist unabhängig von der Gewichtung des Mediums in einem Szenario.

Medienstrom: Kontinuierlich übertragene Folge von Daten, die zu einem Medium gehören.

Medium: [lat. „Mitte“], Mittel(glied), Mittler, Kommunikationsmittel (*Duden*, Band 1) – also eine Darstellung von Information zum Zwecke ihrer Übermittlung. Beispiele: Ton (*Audio*), Bewegtbild (*Video*).

***Middleware* (Zwischenschicht):** Sammlung unterstützender Funktionen. Sie bietet für die Anwendungsentwicklung eine einheitliche Systemschnittstelle an.

Multimedia-Anwendung: Programm, das zur Erbringung einer Aufgabe mehr als ein *Medium* verwendet. Kann lokal (alle Daten befinden sich in einem Gerät) oder verteilt (die Daten müssen zu einem Endgerät übertragen werden) ablaufen. Beispiele: Video-Konferenz, Multimedia-Datenbank, Spiele, usw.

Nachrichtenkopf (engl. *header*): Start eines *Pakets*, der Protokollinformation enthält. Mehrere Köpfe können aneinander gereiht werden.

***Protocol Data Unit* (*PDU*, Protokolldateneinheit):** „Element eines Protokolls, das zwischen gleichen Instanzen in unterschiedlichen Kommunikationssystemen ausgetauscht wird.“ [ITG96]

Querverkehr: Datenverkehr, der nicht zum eigentlich betrachteten Verkehr gehört und der über die selben Netzknoten vermittelt wird, dergestalt dass sich beide Verkehre Ressourcen (Speicher, Verarbeitungskapazität oder Übertragungsbandbreite) teilen.

***Server*:** (auch Datenbank-Server, Video-Server) Zentraler Rechner, der vorbereitete (Multimedia-) Daten bereithält und auf Anforderung verschickt.

Signalisierung (engl. *signalling*): Übermittlung und Verarbeitung von Steuerinformation.

Sitzung (engl. *session*): Menge zusammen gehörender *Verbindungen*, die zeitlich nacheinander oder räumlich nebeneinander bestehen können.

***Skew* (Verzögerungsunterschied):** Abstand der Ausspielzeitpunkte gleichzeitig erfasster Daten, z. B. zwischen Audio-Daten und Video-Bildern.

Szenario: Beschreibung des Einsatzes einer Anwendung. Bestimmt die Anzahl und Art der zu übertragenden Medienströme und die Bedeutung („Gewichtung“) eines Stroms in Relation zu den übrigen.

***Talkspurt*:** Bei der Übertragung gesprochener Sprache kann man zwei Phasen feststellen: eine Sprach-Phase, in der das Sprachsignal eine hohe Energie besitzt, und eine Ruhe-Phase, in der das Signal so leise ist, dass es keine Information trägt und daher nicht übertragen werden muss. Die Datenfolge, die in der Sprach-Phase generiert wird, wird als *Talkspurt* bezeichnet. Sie wird von Ruhe-Phasen (engl. *Silence*) begrenzt.

***Thread* (Kontrollfluss):** Eine sequenzielle Ausführung von Anweisungen. In so genannten *Multitasking*-Betriebssystemen kann die Ausführung eines Threads unterbrochen werden, um andere auszuführen. Das *Scheduling* entscheidet, wann welcher Thread fortgesetzt wird.

Umlaufverzögerung (engl. *round-trip time*): Summe aus Übertragungszeit zwischen Sender und Empfänger und der Zeit für die Übertragung in Rückrichtung.

Verbindung: „Kommunikationsbeziehung zwischen zwei oder mehreren Punkten im Nachrichtennetz“ [ITG96]

Verkehrsformung (engl. *traffic shaping*): Methode, um die Form eines Datenstroms zu verändern. Beeinflusst Parameter wie maximalen Bandbreitenbedarf oder die Länge von Datenbüscheln (*bursts*).

Verzögerung (engl. *delay*): Zeit, die ein Paket zwischen Aussenden und Empfangen benötigt.

Verzögerungsschwankung (engl. *delay jitter*): Statistische Abweichung vom Mittelwert der Übertragungsverzögerung für eine Menge von Paketen.

***Whiteboard*:** Verteilte Anwendung zur Nachbildung einer Wandtafel. Erlaubt es, von mehreren Terminals aus Texte oder Grafiken einzugeben oder vorbereitete Dokumente allen Teilnehmern zu zeigen.

Zielparameter (engl. *target parameters*): Menge von Werten, die Module eines Terminals steuern. Abhängig vom Szenario und der erwarteten Dienstgüte werden die Parameter zunächst initialisiert, später durch den Adaptionsalgorithmus so angepasst, dass die erfahrene Dienstgüte sich verbessert.

Zustandsparameter (engl. *achieved parameters*): Menge von Werten, die die momentane Leistung der Übertragung beschreiben. Sie enthält einerseits die Werte konfigurierbarer Parameter wie Bildgröße oder Kodierung, andererseits gemessene Parameter wie Verlust oder Verzögerung.

A Experimente

A.1 Einführung

In diesem Kapitel werden die durchgeführten Experimente im Detail vorgestellt und auftretende Effekte erklärt. Für eine zusammenfassende Bewertung sei auf Kapitel 5, S. 103 verwiesen. Die Bewertung der Dienstgüte erfolgt mit den Parametertabellen nach B.8, S. 151.

Die in den Schaubildern dargestellten Werte wurden an Messpunkten (s. 5.1, S. 103) durch die Beobachtung des passierenden Datenstroms ermittelt. Die Intervalllänge betrug jeweils 5 Sekunden, in der weit über 100 Pakete vom Sender zum Empfänger gesandt wurden. Somit stehen für die Berechnung gemittelter Werte z. B. für die Verzögerung und den Jitter genügend viele Messwerte zur Verfügung. Jede Messung wurde mehrfach durchgeführt und es wurde festgestellt, dass die Effekte wiederholbar sind. Die angegebenen Werte stammen jeweils von einer repräsentativen Messung.

A.2 Audio-Übertragung

Um die Effektivität der Adaptionsalgorithmen zu überprüfen, wird zuerst ihre Funktion anhand eines unveränderlichen Netzverhaltens, im Vergleich zu einer Übertragung ohne Anpassung, demonstriert. Zudem wird untersucht, wie sich die verschiedenen Netzmodelle der beiden eingesetzten WAN-Emulatoren auf die Dienstgüte auswirken. Anschließend werden Szenarien untersucht, in denen sich das Verhalten des Netzes im Laufe der Zeit ändert.

A.2.1 Konstantes Netzverhalten

Im Gegensatz zur Video-Übertragung hängt die vom Sender erzeugte Bitrate nicht vom „Inhalt“ des Audio-Signals ab, sondern nur von den Kodierungsparametern, also dem Audio-Typ und der Rahmenlänge. Somit muss in diesem Szenario ein effektiver Adaptionsalgorithmus den Netz-Zustand erkennen und die geeignetsten Kodierungsparameter auswählen können.

Zunächst wird eine Übertragung *ohne Anpassung* im Szenario 1.1 (*NIST Net*, s. 4.5.2.1, S. 93) betrachtet. Wie in Abb. A.1 (oben) ersichtlich, erzeugt der Sender eine konstante Bitrate von ca. 70 kbit/s, von denen jedoch nur etwa 59 kbit/s beim Empfänger ankommen[1]. Der Unterschied zur emulierten Rate von 64 kbit/s erklärt sich durch die Paketköpfe (IP und UDP), die im Netz berücksichtigt, jedoch in der Anwendung nicht gemessen werden.

[1] Als erster Wert wird ca. 54 kbit/s gemessen, weil zwischen Aufbau der Verbindung und Start der Übertragung etwa 0.5 s vergehen, in denen keine Pakete gesandt werden.

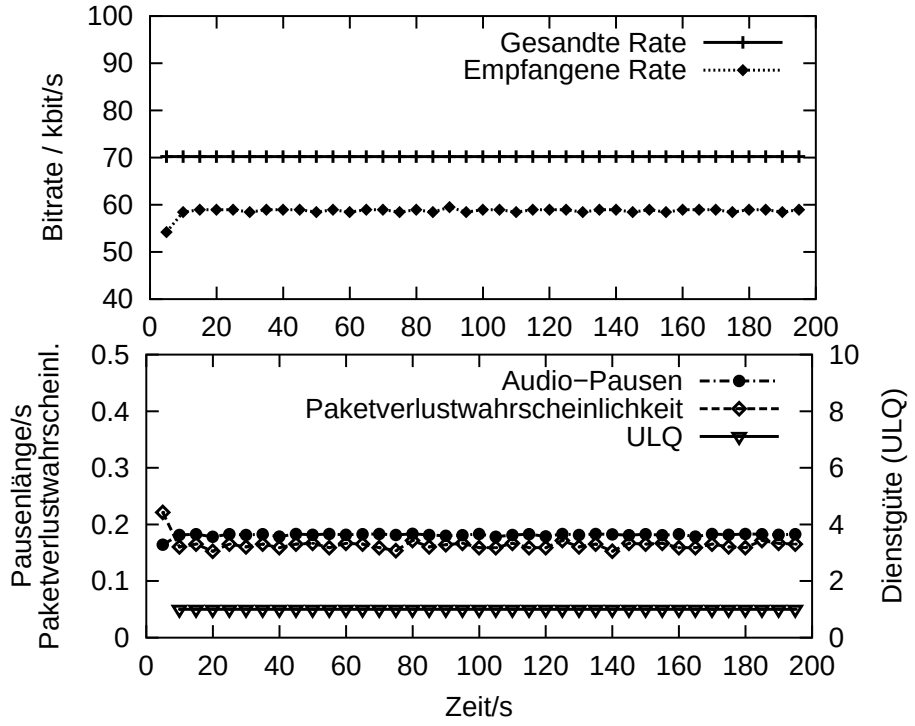


Abbildung A.1: Szenario 1.1 ohne Regelung – Audio-Bitrate (oben), Audio-Pausen, Paketverluste und Dienstgüte (unten)

Der Empfänger beobachtet eine Paketverlustwahrscheinlichkeit von etwa 17 %, die sich in Ausspielpausen (ca. 190 ms/s) niederschlägt und zu einem sehr niedrigen Wert $\overline{ULQ}$ = 1 führt (Abb. A.1, unten).

Der erste Mechanismus „Bitrate“ verwendet nun die empfangene Rate und schaltet auf den Audio-Typ um, der mit dieser Rate auskommen kann. Hierzu überwacht er zusätzlich die Paketverlustwahrscheinlichkeit und die durchschnittliche Größe der Ausspielpausen. Abbildung A.2 (oben) zeigt, dass der Algorithmus, ausgehend vom Start-Typ 82, schrittweise den Typ 79 erreicht. Im unteren Bild kann man den Grund dafür erkennen: zunächst ist die Paketverlustwahrscheinlichkeit größer als der eingestellte Grenzwert von 1 %, weil die Senderate die im Netz eingestellte Rate übersteigt und es durch Pufferüberläufe zu Paketverlusten kommt.

Normalerweise kann jedoch der Algorithmus nicht annehmen, dass das Netzverhalten konstant bleibt. Daher ist es sinnvoll, von Zeit zu Zeit eine Verbesserung der Übertragung zu versuchen (sofern die vorgegebene Maximaldienstgüte nicht bereits erreicht ist), indem auf einen besseren Audio-Typ (hier 80) umgeschaltet wird. Dies wird bei $t = 40\ s$ zum ersten Mal versucht, jedoch ohne Erfolg: die Ausspielpausen nehmen wieder zu, weil wie zuvor die Pufferfüllstände im Netz wachsen. Daher wird zum Audio-Typ 79 zurückgekehrt. Der Algorithmus merkt sich diesen Fehlschlag und verdoppelt die Wartezeit bis zu den nächsten Verbesserungsversuchen ($t = 70\ s, 110\ s, 170\ s$). Jedoch schlägt dieser Versuch fehl und Typ 79 wird wieder eingesetzt.

Für die erfahrene Dienstgüte (Abb. A.2 unten) bedeutet dieses Verhalten (ab ca. $t = 45\ s$ nach der Einschwingphase) ein Pendeln zwischen einem Wert von 6 und 1. Für den beobachteten Zeitraum ergibt sich eine mittlere Dienstgüte von $\overline{ULQ}$ = 4.4 ($\widetilde{ULQ}$ = 6), also eine deutliche Verbesserung gegenüber der Übertragung ohne Regelung. Und in diesem stabilen Szenario ist über einen län-

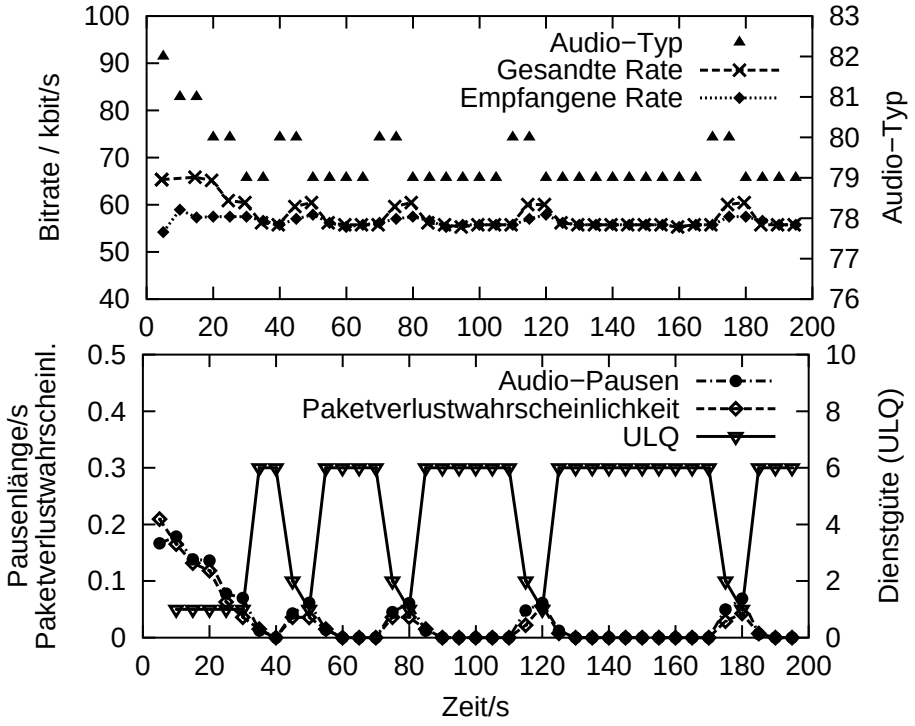


Abbildung A.2: Szenario 1.1 mit Regelung der Bitrate – Audio-Typ und -Bitrate (oben), Audio-Pausen, Paketverluste und Dienstgüte (unten)

geren Zeitraum mit einer Dienstgüte von 6 zu rechnen, nur unterbrochen von den Versuchen zur Verbesserung des Audio-Typs.

Der Vergleich mit der Anpassung beim Einsatz von *The Cloud* ergibt ein praktisch identisches Verhalten. Jedoch unterscheidet sich hier bei sehr ähnlichen Parametern das Verhalten des Netzes: Bei Erhöhung der Senderate steigt die Verzögerungszeit bei *The Cloud* deutlich an, da sich die Pakete im Emulator aufstauen, während sie bei *NIST Net* recht konstant bleibt (Abb. A.3 oben). Der Jitter ist dagegen bei *NIST Net* größer (Abb. A.3 unten). Außerdem führt das einfache WAN-Modell von *NIST Net* dazu, dass, im Unterschied zu *The Cloud*, Paketüberholungen auftreten. Bei einem Jitter-Parameter von $\sigma = 10$ *ms* (Standardabweichung der Verzögerung) kann dies etwa einmal alle 20 s beobachtet werden, bei $\sigma = 20$ *ms* jedoch schon etwa ein- bis dreimal pro Sekunde. Die Häufigkeit hängt mit dem Paketabstand von etwa 40 ms zusammen. Die Umordnungen der Pakete können jedoch korrigiert werden, wenn die Pufferverzögerung entsprechend angepasst wird. Bei *The Cloud* fällt auf, dass die Netz-Verzögerung nach dem Umschalten auf einen besseren Audio-Typ deutlich stärker ansteigt (bis ca. 0.4 s) als bei *NIST Net* und dies zu einer bis zu doppelt so großen Ende-zu-Ende–Verzögerung führt. Dies liegt wieder am Netzmodell, das Paketüberholungen verhindert. Dennoch ist die durchschnittliche Dienstgüte von $\overline{ULQ} = 4.3$ fast gleich der von *NIST Net*, denn durch die hohe Verzögerung kommt es zu weniger Ausspielpausen und die Dienstgüte sinkt nach dem Umschalten nicht so tief ab. Jedoch ist der Medianwert mit $\widetilde{ULQ} = 5$ niedriger.

Im Szenario 1.2 (konstante Bitrate von $B = 128$ kbit/s) ergibt sich bei Verwendung von *The Cloud* keine Notwendigkeit, eine Anpassung durchzuführen, da die Sende-Bitrate für den gewünschten Audio-Typ vom Netz erbracht wird. Allerdings stauen sich ohne Steuerung des Pufferfüllstands der Sound-Karte die Audio-Daten vor dem Ausspielen und die Ende-zu-Ende–Verzögerung liegt bei

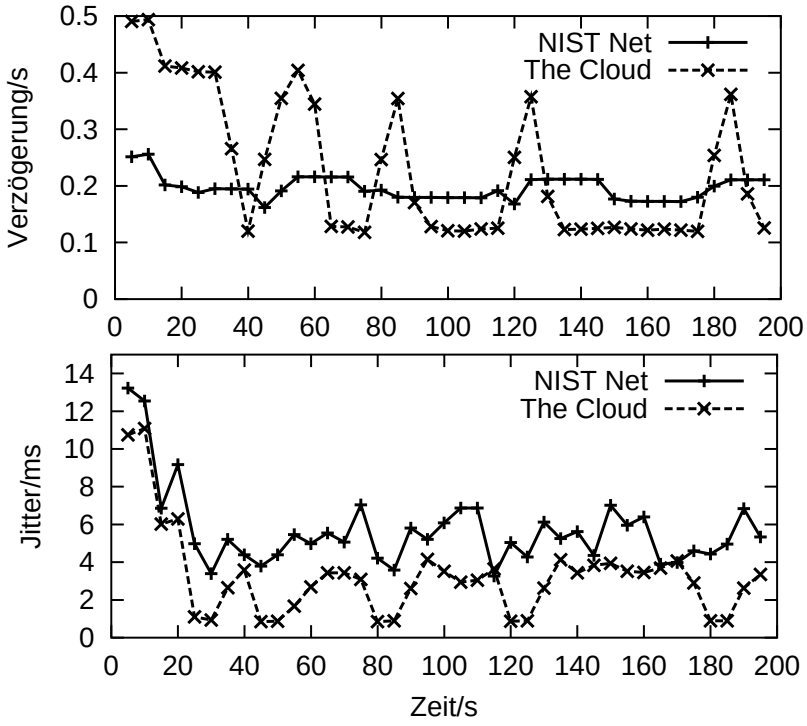


Abbildung A.3: Szenario 1.1 mit Mechanismus „Bitrate“, Vergleich Verzögerung (oben) und Jitter (unten)

etwa 0.6 s, wodurch nur der Wert $\overline{ULQ} = 4$ erreicht wird. Anders liegt der Fall bei der Verwendung von *NIST Net*. Hier treten durch das eingesetzte Netzmodell sehr viele Paketumordnungen auf, die ohne eine Anpassung der Pufferverzögerung nicht ausgeglichen werden können. Denn ist die eingestellte Verzögerung für einen Audio-Rahmen verstrichen, so wird dieser (bei einer Netz-Verzögerung von ca. 120 ms) nach etwa 140 ms ausgespielt. Dies führt dazu, dass alle folgenden Rahmen, die eigentlich vorher hätten ausgespielt werden müssen, nun verworfen werden, was viele Ausspielpausen (ca. 70 ms/s) und einen Wert $\overline{ULQ} = 1.8$ ($\widetilde{ULQ} = 1$) zur Folge hat.

Durch den Einsatz des Mechanismus „Pufferverzögerung“ werden die angekommenen Rahmen im Empfänger länger verzögert (bis ca. 180 ms bei einer Netz-Verzögerung von 120 ms), so dass ihn die meisten der überholten Pakete noch vor der Übergabe an die Sound-Karte erreichen können und dann vorgezogen werden. Hierdurch kommt es zu praktisch keinen Ausspielpausen mehr und die Dienstgüte erreicht insgesamt den Wert $\overline{ULQ} = 5.9$ ($\widetilde{ULQ} = 6$), da die Ende-zu-Ende–Verzögerung mit ca. 220 ms einem *ULQ*–Wert von etwa 4.25 entspricht.

Für den letzten Typ der konstanten Szenarien wird die Paketverlustwahrscheinlichkeit des Netzes auf einen Wert größer Null gesetzt. Ohne Anpassung geht also ein entsprechender Teil der Audio-Daten verloren, was sich direkt in Ausspielpausen auswirkt. In beiden Fällen (1.3 und 1.4) kann somit nur eine Dienstgüte von $\overline{ULQ} = 1$ erreicht werden.

Hier muss der Adaptionsalgorithmus erkennen, dass eine stete Reduktion der Sende-Bitrate keine Verbesserung der Dienstgüte liefert und auf einen Audio-Typ mit Vorwärts-Fehlerkorrektur (S. 71) umschalten. Der Mechanismus zur Regelung der Bitrate stellt keine Reduktion der Paketverlust-wahrscheinlichkeit unter den Schwellwert von 1 % fest und wählt daher stets einen schlechteren Audio-Typ, ist hier also ungeeignet. Der Mechanismus „FEC“ setzt dagegen Audio-Typen mit Vorwärts-Fehlerkorrektur ein (Abb. A.4 oben) und sucht denjenigen aus, für den die wenigsten

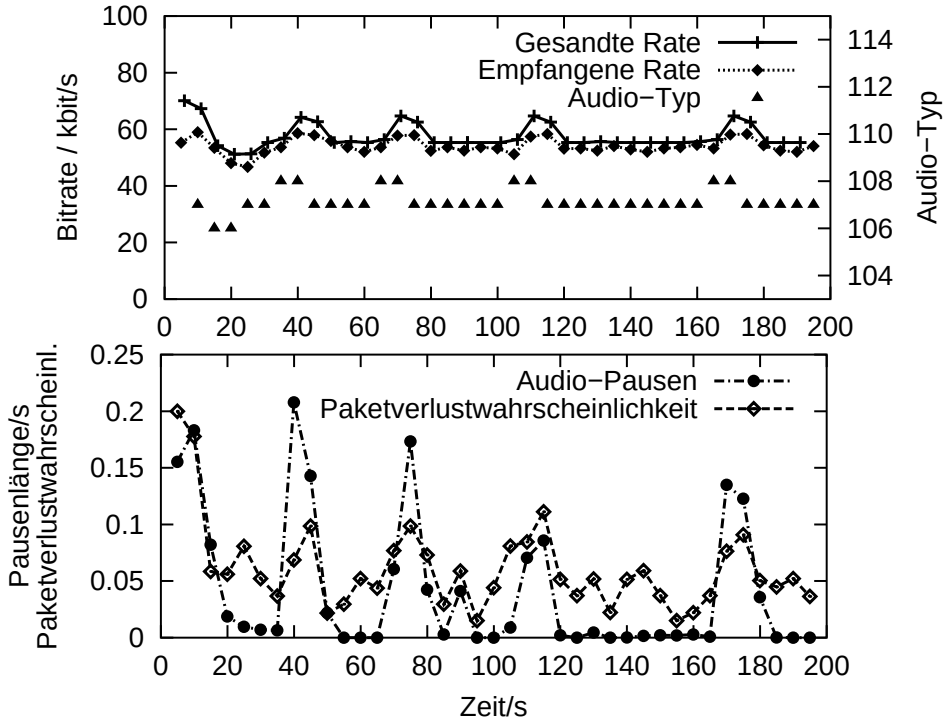


Abbildung A.4: Szenario 1.3 mit Auswahl des FEC-Typs, Bitrate (oben), Audio-Pausen und Paketverluste (unten)

Ausspielpausen auftreten (unten), obwohl die Paketverlustwahrscheinlichkeit mit 5 % im Szenario 1.3 (10 % im Szenario 1.4) sehr hoch bleibt. Man erkennt, dass auch dieser Mechanismus regelmäßig den Versuch unternimmt, die Dienstgüte durch die Wahl eines besseren Audio-Typs zu erhöhen. Die maximal erreichte Dienstgüte beträgt $ULQ = 4$, bei einer durchschnittlichen Dienstgüte $\overline{ULQ} = 3.2$ ($\widetilde{ULQ} = 4$). Dies liegt zum einen an der niedrigeren Qualität des Audio-Typs (z. B. 107: 4.7), aber auch an der erhöhten Verzögerung, denn die Audio-Daten müssen länger gepuffert werden (Ziel-Pufferverzögerung von ca. 240 ms), um den Redundanz-Rahmen das Erreichen des Empfängers zu erlauben. Das Verhalten des Algorithmus im Szenario 1.4 ist identisch, obwohl hier mehr Paketverluste auftreten.

Die Ergebnisse bei Emulation mit *The Cloud* unterscheiden sich nicht wesentlich. Wieder kann das andere Netzverhalten beobachtet werden, das zu einer höheren Verzögerung führt. Die Wahl eines besseren Audio-Typs führt zu besonders hohen Netz-Verzögerungen (ca. 300 ms statt 150 ms).

Da die Anwendung jedoch nicht wissen kann, welche Art von Beeinträchtigungen der Übertragung auftreten werden, muss im übergeordneten Algorithmus die Auswahl des passenden Mechanismus erfolgen, indem das Netzverhalten erkannt wird. Hierzu wurde ein Adaptionsalgorithmus (s. 4.3.4, S. 76) entwickelt, der die vorgestellten Mechanismen je nach Zustand des Netzes einsetzt.

Der Algorithmus startet in der Annahme, dass sich das Netz in einem Zustand mit zufälligem Paketverlust befindet. Die Ergebnisse für die Szenarien 1.3 und 1.4 sind daher identisch mit den oben beschriebenen für den Mechanismus „FEC“, bis auf den Start – es wird erst nach dem zweiten Beobachtungsintervall auf einen neuen Audio-Typ umgeschaltet, da erst dann zuverlässige Informationen gesammelt wurden. Im Szenario 1.1 muss der Algorithmus jedoch erst feststellen, dass keine zufälligen Paketverluste auftreten. Dies erkennt er daran, dass sich nach kurzer Zeit die Paketverluste auf 0 reduzieren, so dass Vorwärts-Fehlerkorrektur nicht mehr notwendig ist (Abb. A.5

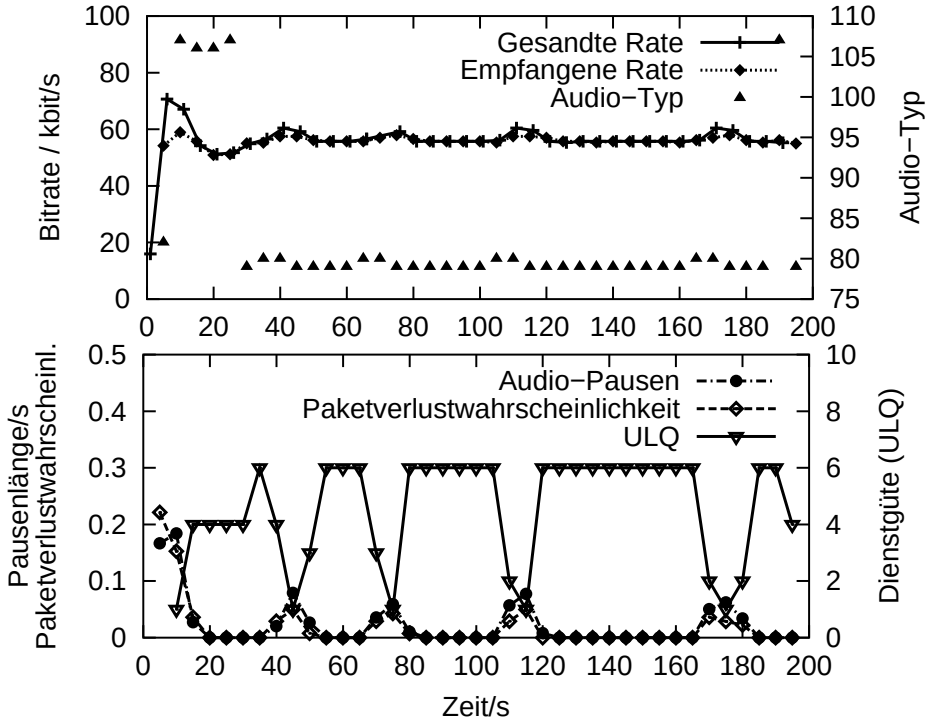


Abbildung A.5: Szenario 1.1 mit automatischer Erkennung des Netzzustands

oben). Ab $t = 30\ s$ verwendet er den Mechanismus „Bitrate“ (die FEC–Audio-Typen besitzen Nummern zwischen 104 und 114), das Verhalten entspricht dem oben beschriebenen und es wird eine durchschnittliche Dienstgüte von $\overline{ULQ} = 3.9$ ($\widetilde{ULQ} = 5$) erreicht. Die Tabellen A.1 und A.2 (S. 129) führen die Dienstgütenwerte für *NIST Net* und *The Cloud* auf.

Es zeigt sich, dass die vorgestellten Algorithmen gut einen konstanten Netz-Zustand erkennen und sich auf diesen einstellen können. Interessant ist nunmehr, wie schnell die Anpassung an eine Änderung des Netzverhaltens erfolgt, dies wird in den folgenden Abschnitten untersucht.

A.2.2 Verzögerungsänderung

In diesen Szenarien ändert sich die Verzögerung, mit der das Netz die Pakete vom Sender zum Empfänger überträgt (s. 4.5.2.2, S. 94). Aufgrund der unterschiedlichen Netzmodelle unterscheidet sich das beobachtete Verhalten für die beiden Emulatoren sehr stark, daher werden die Szenarien einzeln diskutiert.

A.2.2.1 Emulation mit NIST Net

Bei *NIST Net* treten mit den gewählten Szenarien-Parametern sehr viele Paketumordnungen auf, die von der Anwendung durch die Pufferstrategie korrigiert werden können (s. S. 117). Ohne Regelung müssen jedoch die zu spät angekommenen Rahmen verworfen werden, wodurch sich der Puffer der Sound-Karte leert und die Ende-zu-Ende–Verzögerung der Netz-Verzögerung genau folgt (Abb. A.6 oben). Es treten nur kurze Pausen auf, wodurch in den Phasen niedriger Verzögerung ein Wert $ULQ = 7$ (Zielwert) erreicht wird (Abb. A.6 unten). Die Dienstgüte beträgt hier insgesamt $\overline{ULQ} = 5.9$ ($\widetilde{ULQ} = 7$).

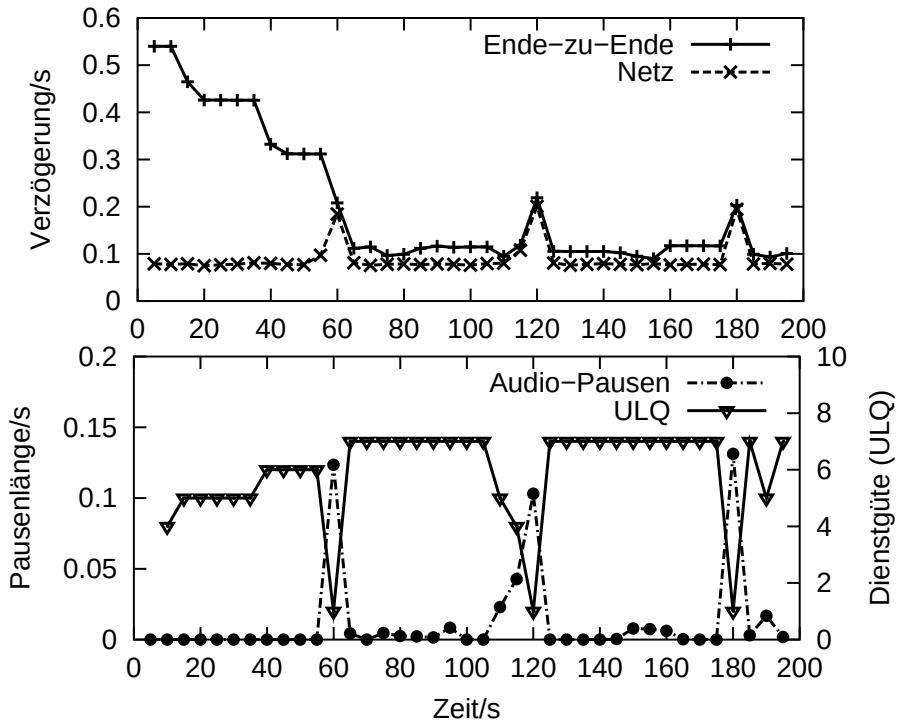


Abbildung A.6: Szenario 2.1 ohne Regelung

Für die geregelte Übertragung sieht das Bild ähnlich aus (Abb. A.7). Es fällt jedoch auf, dass die Ende-zu-Ende–Verzögerung bei etwa 200 ms verharrt, wodurch eine maximale Dienstgüte von $ULQ = 6$ erreicht wird – dies schlägt sich in der durchschnittlichen Dienstgüte von $\overline{ULQ} = 5.4$ ($\widetilde{ULQ} = 6$) nieder, die in diesem Fall sogar niedriger ist als ohne Regelung. Die Änderung der Verzögerung von 5 *s* ist also zu kurz, um durch den Algorithmus eine Verbesserung zu erreichen.

Für das Szenario 2.2 wird ein analoges Verhalten beobachtet. Da sich im geregelten Fall jedoch beim Wechsel von der hohen zur niedrigen Verzögerung einige Pakete im Puffer der Sound-Karte aufstauen, steigt hier kurz die Verzögerung auf ca. 400 ms an. Ohne Regelung wird eine Dienstgüte

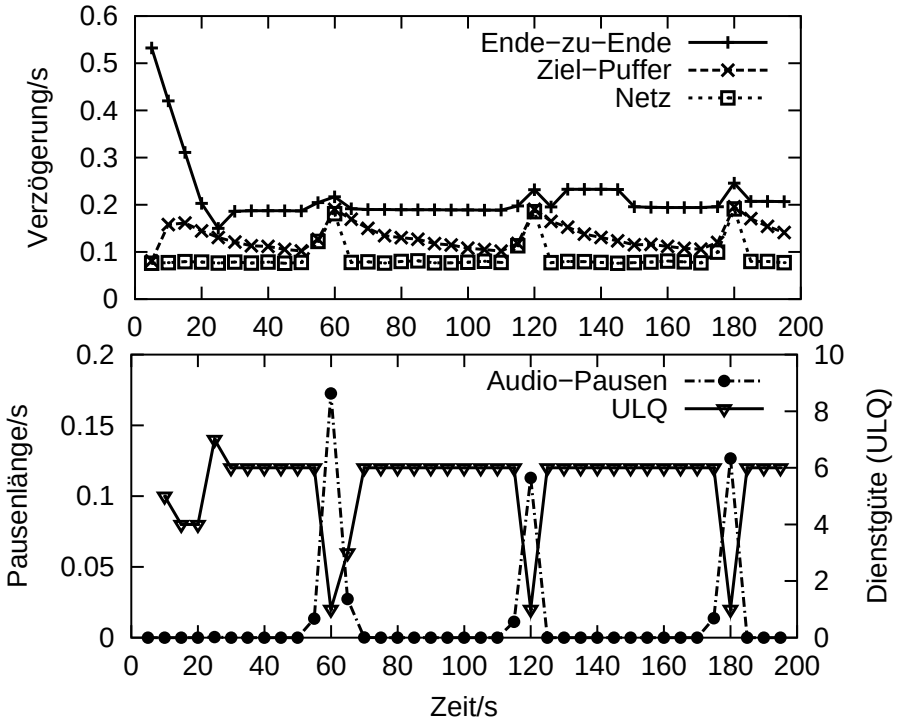


Abbildung A.7: Szenario 2.1 mit Regelung

von $\overline{ULQ}$ = 4.6 ($\widetilde{ULQ}$ = 5), mit Regelung von $\overline{ULQ}$ = 5.0 ($\widetilde{ULQ}$ = 6) erreicht, also eine Verbesserung. Ein sehr ähnliches Bild ergibt sich für das Szenario 2.3. Hier steigt mit Regelung die Verzögerung ebenfalls stark an, jedoch führt sie zu einer deutlichen Verbesserung mit $\overline{ULQ}$ = 4.9 ($\widetilde{ULQ}$ = 5) gegenüber $\overline{ULQ}$ = 4 ($\widetilde{ULQ}$ = 4) ohne Regelung.

Für die letzten Szenarien dieses Typs wurden sehr viel größere Werte gewählt. Insbesondere wurde auch im Zustand niedrigerer Verzögerung eine große Standardabweichung eingesetzt, wodurch hier ohne Regelung sehr viele Pausen auftreten, die zu einer schlechten Dienstgüte von $\overline{ULQ}$ = 2.0 ($\widetilde{ULQ}$ = 2) führen (Abb. A.8).

Die Regelung kann hier die Dienstgüte deutlich verbessern (Abb. A.9). Es treten nur noch dann kurzzeitig Ausspielpausen auf, wenn sich die Verzögerung erhöht, wodurch wieder eine durchschnittliche Dienstgüte von $\overline{ULQ}$ = 5.0 ($\widetilde{ULQ}$ = 6) erreicht wird.

Die letzten beiden Szenarien 2.5 und 2.6 unterscheiden sich nur durch die Größe der Puffer im Netz. Die kleinere Speicherfähigkeit in der „hoch“–Phase in Szenario 2.6 hat zur Folge, dass der Emulator ca. 10 % der Pakete verwirft. Dies führt zu zusätzlichen Ausspielpausen und zusammen mit der sehr hohen Verzögerung zu einer weiteren Verschlechterung. Tabelle A.3, S. 130 fasst die Dienstgütenwerte zusammen.

A.2.2.2 Emulation mit The Cloud

Bei Verwendung von *The Cloud* für die Emulation der Verzögerungsänderung ergibt sich ohne Regelung ein völlig anderes Bild als bei der Verwendung von *NIST Net*. Das Netzmodell führt zu keinen Paketüberholungen, so dass sich der Puffer der Sound-Karte im Empfänger füllen kann

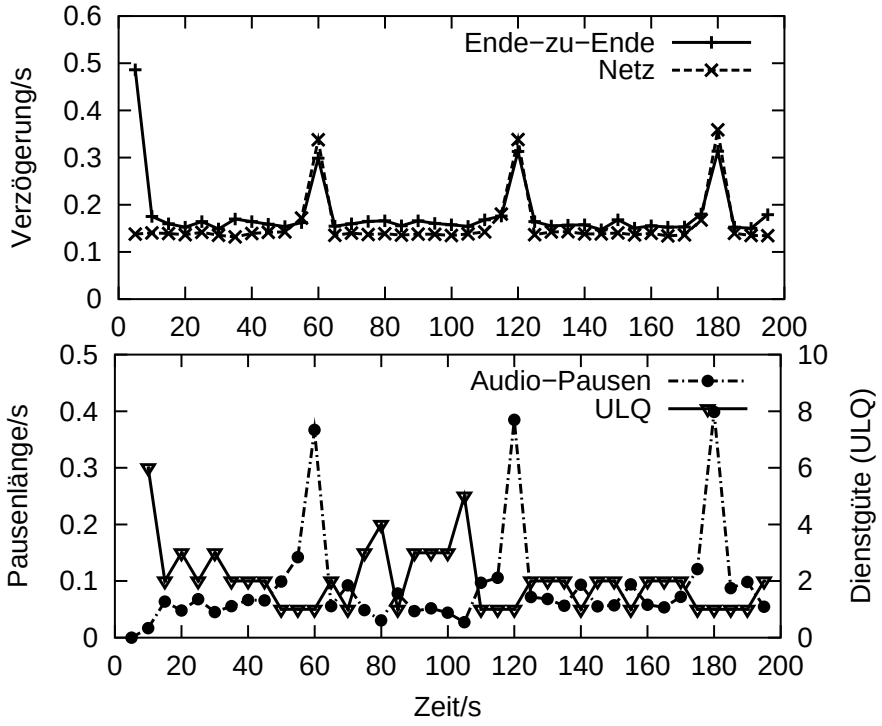


Abbildung A.8: Szenario 2.4 ohne Regelung

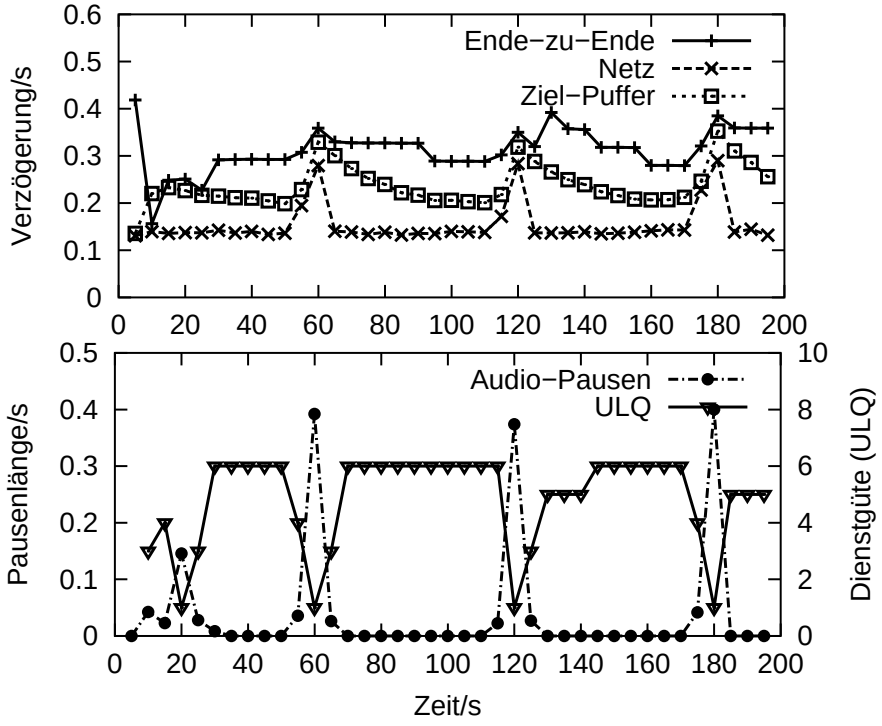


Abbildung A.9: Szenario 2.4 mit Regelung

und die Übertragung durch die variable Netz-Verzögerung kaum beeinflusst wird, sondern konstant etwa 600 ms beträgt (Abb. A.10). Dies entspricht einer ebenfalls konstanten Dienstgüte von 4.

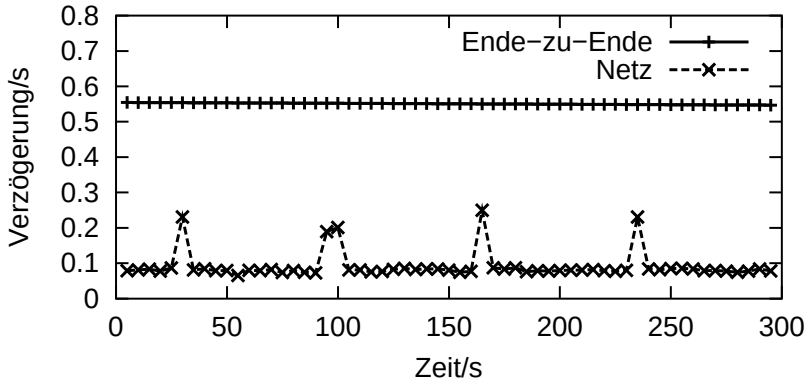


Abbildung A.10: Szenario 2.1 (*The Cloud*) ohne Regelung

Hier bringt der Einsatz des Algorithmus eine deutliche Verbesserung, da die Verzögerung meist niedriger ist als im Fall ohne Regelung. Die Dienstgüte sinkt jedoch nach dem Umschalten vom niedrigen in den hohen Zustand kurz ab, da die Ende-zu-Ende–Verzögerung so stark ansteigt, dass der Puffer der Sound-Karte leer läuft (Abb. A.11, $t = 65\ s,\ 135\ s,\ \ldots$).

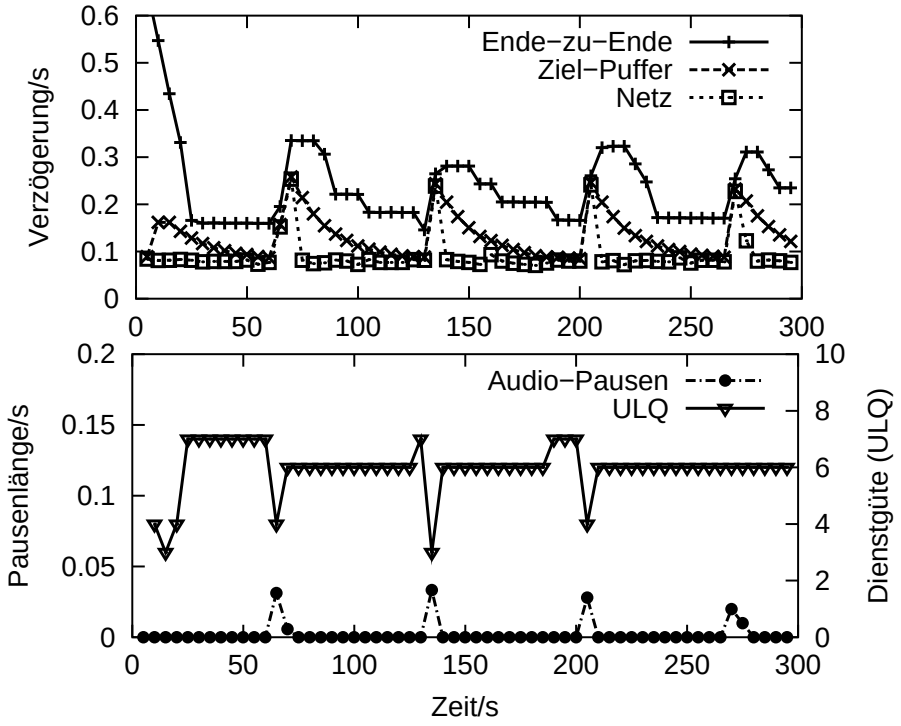


Abbildung A.11: Szenario 2.1 (*The Cloud*) mit Regelung

Tabelle A.4, S. 130 zeigt die durchschnittliche Dienstgüte der Szenarien ohne und mit Adaption.

A.2.3 Änderung der im Netz verfügbaren Bitrate

Für die Szenarien 3.1 bis 3.3 (s. 4.5.2.3, S. 95) wird angenommen, dass sich die für die Übertragung des Audio-Medienstroms im Netz zur Verfügung stehende Bitrate ändert. Im Zustand mit hoher Rate ist mehr Bitrate verfügbar, als für den gewählten Audio-Typ (hier: 96 kbit/s bei ca. 70 kbit/s) benötigt wird. Regelmäßig wird nun in einen Zustand mit niedrigerer Rate (hier: 64 kbit/s bzw. 32 kbit/s) umgeschaltet, und die Anwendung muss sich hierauf einstellen.

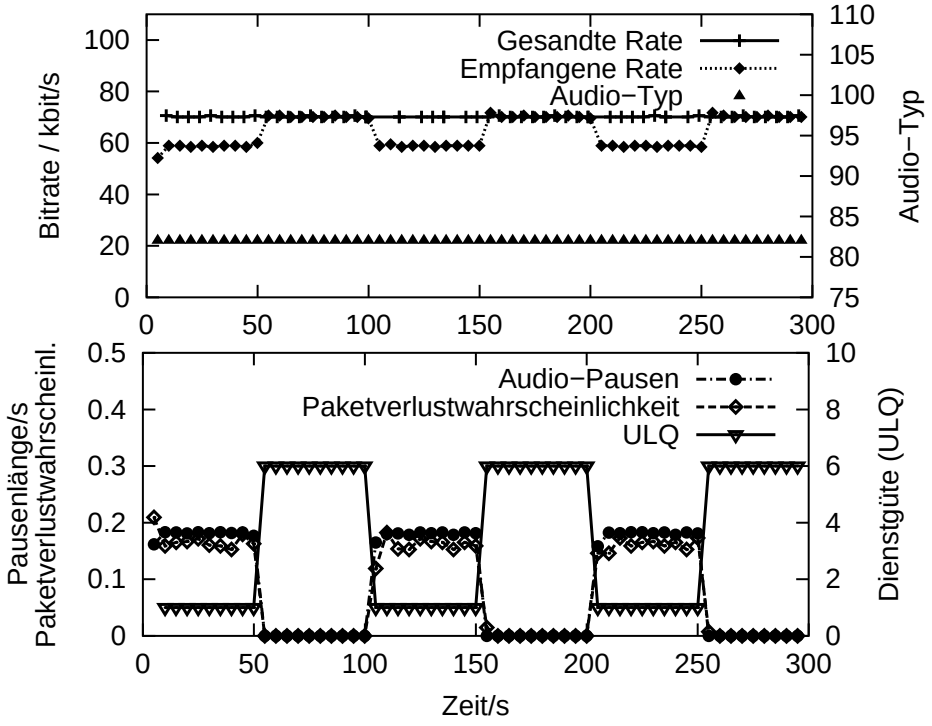


Abbildung A.12: Szenario 3.1 ohne Regelung

Für das Szenario 3.1 (diese Szenarien können nur mit *NIST Net* emuliert werden) ergibt sich ohne Regelung die Dienstgüte nach Abb. A.12. Es wird stets der Audio-Typ 82 verwandt, wodurch in den Phasen mit hoher Bitrate eine gute Dienstgüte erbracht wird (ULQ = 6, da die Verzögerung mit ca. 0.2 s hoch ist), jedoch in den Phasen mit niedriger Bitrate die Dienstgüte stark absinkt (ULQ = 1), da sehr viele Pausen auftreten. Die durchschnittliche Dienstgüte beträgt somit erwartungsgemäß $\overline{ULQ}$ = 3.5 ($\widetilde{ULQ}$ = 3.5).

Wird nun die Übertragung mit dem Adaptionsalgorithmus angepasst, so ergibt sich ein anderes Bild (Abb. A.13). Nach kurzer Zeit stellt der Algorithmus fest, dass keine zufälligen Verluste auftreten und deaktiviert die Vorwärts-Fehlerkorrektur ($t = 30\ s$). Er wählt den Audio-Typ aus, der mit der gemessenen Bitrate auskommen kann (hier Typ 79) und versucht immer wieder Verbesserungen (s. A.2.1, S. 116), bis er zum Zeitpunkt $t = 90\ s$ den Ziel-Typ 82 erreicht hat (abgesehen vom einmaligen Umschalten auf einen FEC-Typ bei $t = 60\ s$). Schaltet das Netz dann zum Zeitpunkt $t = 100\ s$ in den Zustand mit niedrigerer Rate, so nimmt der Algorithmus zunächst an, dass es sich um zufälligen Paketverlust handelt und schaltet wieder auf einen FEC-Audio-Typ ($t = 125\ s$ und $t = 225\ s$) und der Ablauf wiederholt sich. Die erfahrene Dienstgüte erhöht sich hier insgesamt auf $\overline{ULQ}$ = 4.3 ($\widetilde{ULQ}$ = 6). Man kann feststellen, dass sich der Algorithmus nach etwa 30 Sekunden einer neuen Rate anpassen kann.

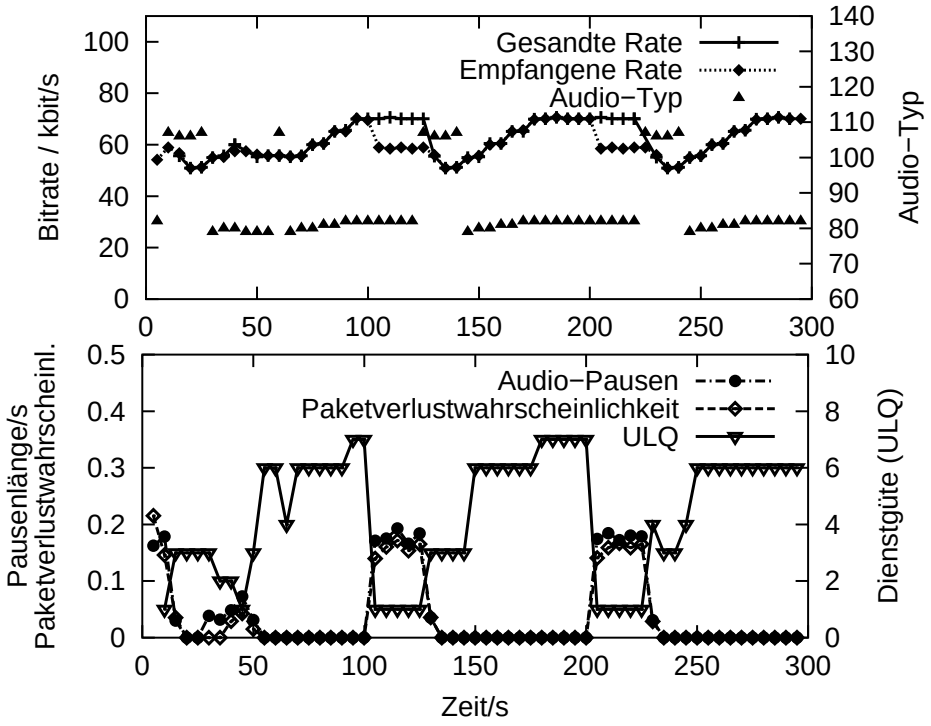


Abbildung A.13: Szenario 3.1 mit Adaption

Anders sieht das Bild für Szenario 3.2 aus, bei dem die Größe der Puffer im Netz vervierfacht wurde. Hier zeigt sich, dass das Netz beim Eintritt in die Phase mit niedrigerer Bitrate so viele Pakete speichert, dass die Netz-Verzögerung auf über 1.5 s stark ansteigt (Abb. A.14).

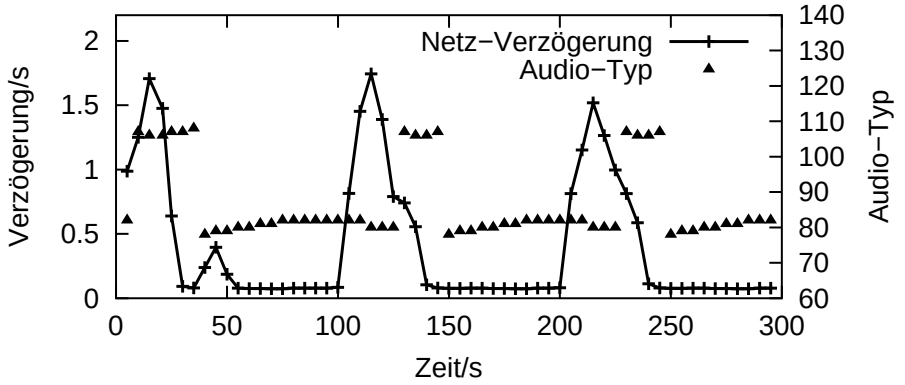


Abbildung A.14: Szenario 3.2, Verzögerungen

Da der Algorithmus aber reagiert und die Sende-Rate reduziert, nimmt die Netz-Verzögerung schnell auf einen niedrigen Wert ab. Die Dienstgüte verbessert sich deutlich mit $\overline{ULQ} = 3.5$ ($\widetilde{ULQ} = 4$) gegenüber der Dienstgüte ohne Regelung von $\overline{ULQ} = 2.5$ ($\widetilde{ULQ} = 2$). Dennoch stellt eine hohe Speicherkapazität des Netzes ein Problem dar, für das bisher keine besonders effektive Lösung gefunden wurde.

Im letzten Szenario 3.3 dieser Gruppe ist die Rate im Zustand „niedrig“ besonders klein (32 kbit/s). Ohne Regelung gehen in diesen Phasen fast 60 % der Pakete verloren, was zu Ausspielpausen von 600 ms/s führt. Die durchschnittliche Dienstgüte liegt dennoch bei $\overline{ULQ} = 3.4$ ($\widetilde{ULQ} = 2$), da der Wert wieder zwischen 1 und 6 pendelt. Der Regelung gelingt es, die Zahl der Ausspielpausen zu reduzieren, indem ein FEC-Typ gewählt wird. Nach etwa 30 Sekunden verbessert sich die Dienstgüte deutlich, erreicht jedoch insgesamt nur den Wert $\overline{ULQ} = 3.0$ ($\widetilde{ULQ} = 3$), da zum einen die Dienstgüte der FEC-Audio-Typen geringer ist, und zum anderen die Ende-zu-Ende–Verzögerung höher ist als ohne Regelung (200 bis 500 ms statt 200 bis 250 ms).

Insgesamt zeigt sich hier, dass der Algorithmus in der Lage ist, die Anwendung innerhalb von etwa 30 Sekunden an die veränderten Bitrate im Netz anzupassen. Die beobachtete Dienstgüte in den einzelnen Szenarien kann Tab. A.5, S. 130 entnommen werden (diese Szenarien können nicht mit *The Cloud* emuliert werden).

A.2.4 Verlust-Phasen

In der letzten Gruppe 4.1 bis 4.5 der Szenarien wird die Zahl der zufälligen Paketverluste variiert (s. 4.5.2.4, S. 96). Ohne Adaption folgen die Ausspielpausen den Paketverlusten, wobei im Szenario 4.1 zu Beginn die Paketverluste dazu führen, dass sich der Ausspielpuffer leert und so die Ende-zu-Ende–Verzögerung ab- und die Dienstgüte zunimmt (vgl. A.2.2.1, S. 121). Später führen die verlorenen Rahmen jedoch zu Ausspielpausen und reduzierter Dienstgüte. Der Vergleich der Dienstgüte mit der Übertragung mit Regelung kann Tab. A.6, S. 130 (für *NIST Net*) entnommen werden.

Hier fällt auf, dass in den Szenarien, in denen stets ein zufälliger Paketverlust auftritt, die Dienstgüte deutlich höher ist als in den anderen Szenarien. Dies liegt daran, dass der Algorithmus dort schnell auf den besten FEC-Audio-Typ umschaltet, der eine Dienstgüte von über 7 hat, jedoch auch eine Bitrate von 160 kbit/s benötigt. In den anderen Szenarien braucht der Algorithmus lange, bis er entschieden hat, dass ein zufälliger Paketverlust vorliegt und er einen FEC-Audio-Typ einsetzt.

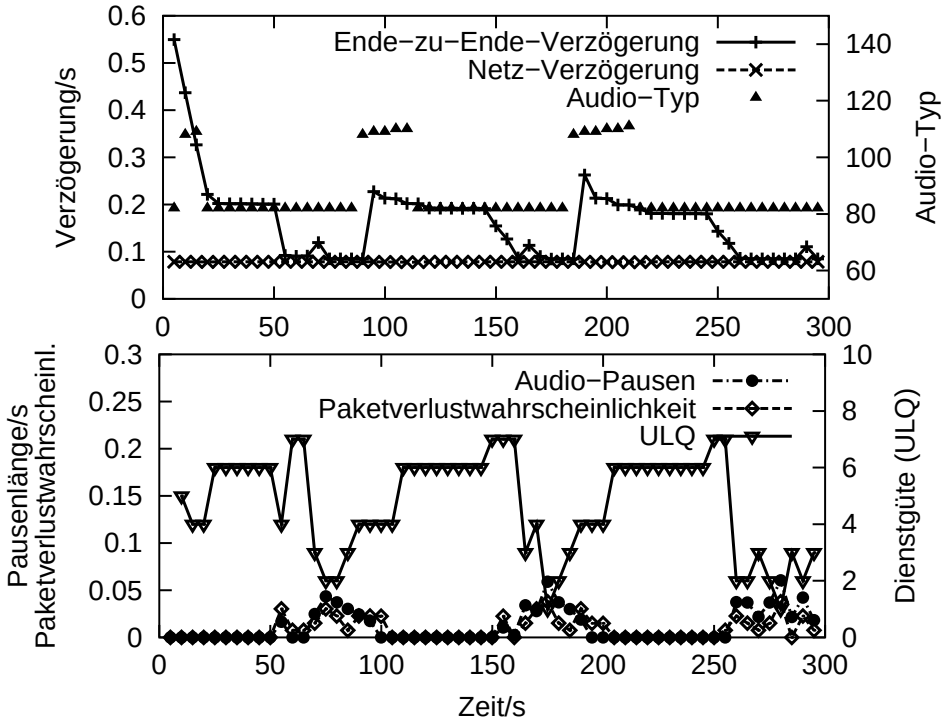


Abbildung A.15: Szenario 4.1 mit Adaption

In Abb. A.15 kann dies für das erste Szenario veranschaulicht werden. Zum Zeitpunkt $t = 55\ s$ bemerkt der Algorithmus, dass die Paketverlustrate gestiegen ist. Doch erst bei $t = 90\ s$ wird auf einen FEC-Audio-Typ umgeschaltet, der schon bei $t = 115\ s$ durch einen normalen Audio-Typ ersetzt wird, da die Paketverluste wieder auf Null gesunken sind. Dieses Verhalten wiederholt sich später bei jeder Änderung der Paketverluste im Netz.

Im Szenario 4.4 (Abb. A.16) stellt der Algorithmus dagegen dauerhaft den FEC-Audio-Typ 114 ein, wodurch sich eine praktisch konstante Dienstgüte von 6 ergibt.

Die Emulation der Szenarien mit *The Cloud* ist nur möglich, indem sog. „Congestion“–Phasen (Überlastung) eingestellt werden. Diese werden jedoch zufällig vom Programm aktiviert, daher ist

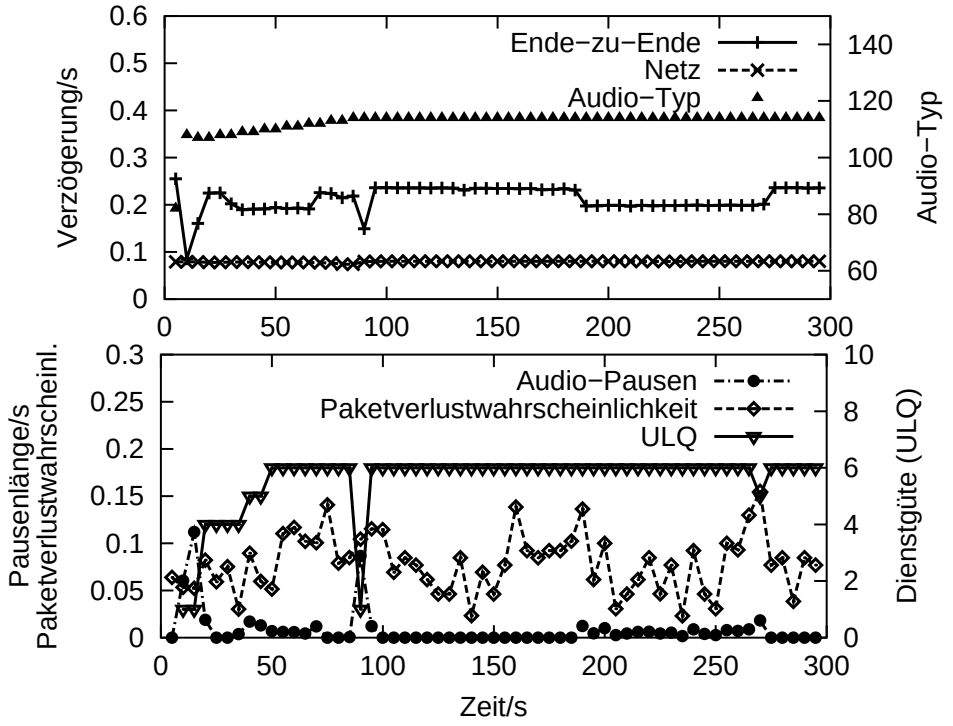


Abbildung A.16: Szenario 4.4 mit Adaption

die Vergleichbarkeit der Ergebnisse nur eingeschränkt gegeben. Tabelle A.7, S. 129 zeigt Messungen über jeweils 200 s, wobei meist jeweils zwei Phasen mit erhöhter Paketverlustwahrscheinlichkeit auftraten.

A.2.5 Tabellarische Übersicht

Die folgenden Tabellen zeigen die erreichte Dienstgüte in den Szenarien mit und ohne Adaption. $\overline{ULQ}$ bezeichnet den durchschnittlichen Dienstgütenwert und $\widetilde{ULQ}$ den Medianwert.

Szenario	1.1	1.2	1.3	1.4
ohne Regelung	1.0 [1]	1.7 [1]	1.0 [1]	1.0 [1]
nur Mechanismus Bitrate	4.3 [6]	5.9 [6]	1.3 [1]	1.0 [1]
nur Mechanismus FEC	2.8 [4]	4.8 [6]	3.2 [4]	3.0 [4]
mit Erkennung des Netzzustands	4.6 [6]	5.4 [6]	2.9 [3]	3.1 [4]

Tabelle A.1: Dienstgüte $\overline{ULQ}$ und [$\widetilde{ULQ}$] für die Szenarien mit konstantem Netzverhalten (*NIST Net*)

Szenario	4.1	4.2	4.3	4.4	4.5
ohne Regelung	4.4 [3]	3.0 [3]	3.3 [3]	1.4 [1]	2.2 [2]
mit Regelung	5.5 [6]	3.9 [4]	5.1 [6]	5.5 [6]	3.6 [4]

Tabelle A.7: Dienstgüte $\overline{ULQ}$ [$\widetilde{ULQ}$] der Szenarien mit variablem Verlust (*The Cloud*, Anmerkungen S. 97 beachten!)

Szenario	1.1	1.2	1.3	1.4
ohne Regelung	1.0 [1]	4.0 [4]	1.0 [1]	1.0 [1]
nur Mechanismus Bitrate	4.3 [5]	5.9 [6]	1.2 [1]	1.0 [1]
nur Mechanismus FEC	2.5 [3]	3.1 [4]	2.8 [3]	3.4 [4]
mit Erkennung des Netzzustands	4.5 [5]	6.6 [6]	2.2 [3]	2.7 [3]

Tabelle A.2: Dienstgüte $\overline{ULQ}$ und $[\widetilde{ULQ}]$ für die Szenarien mit konstantem Netzverhalten (*The Cloud*)

Szenario	2.1	2.2	2.3	2.4	2.5	2.6
ohne Regelung	5.7 [7]	4.8 [6]	4.0 [4]	1.9 [2]	1.8 [1]	1.7 [1]
mit Regelung	5.2 [6]	4.9 [6]	5.1 [5]	4.8 [6]	3.5 [4]	3.0 [2.5]

Tabelle A.3: Dienstgüte $\overline{ULQ}$ $[\widetilde{ULQ}]$ der Szenarien mit variabler Verzögerung (*NIST Net*)

Szenario	2.1	2.2	2.3	2.4	2.5
ohne Regelung	4.0 [4]	4.0 [4]	4.0 [4]	4.0 [4]	4.0 [4]
mit Regelung	6.0 [6]	5.8 [6]	5.8 [6]	5.1 [6]	4.5 [4]

Tabelle A.4: Dienstgüte $\overline{ULQ}$ $[\widetilde{ULQ}]$ der Szenarien mit variabler Verzögerung (*The Cloud*)

Szenario	3.1	3.2	3.3
ohne Regelung	3.5 [3.5]	2.5 [2]	3.4 [2]
mit Regelung	4.5 [6]	3.5 [4]	3.0 [3]

Tabelle A.5: Dienstgüte $\overline{ULQ}$ $[\widetilde{ULQ}]$ der Szenarien mit variabler Bitrate (*NIST Net*)

Szenario	4.1	4.2	4.3	4.4	4.5
ohne Regelung	4.1 [4]	2.9 [3]	2.8 [3]	1.2 [1]	1.9 [1]
mit Regelung	4.7 [6]	4.2 [4.5]	4.0 [4]	5.6 [6]	5.5 [6]

Tabelle A.6: Dienstgüte $\overline{ULQ}$ $[\widetilde{ULQ}]$ der Szenarien mit variablem Verlust (*NIST Net*)

A.3 Video-Übertragung

Zur Untersuchung der Anpassung der Übertragung eines einzelnen Video-Stroms an das Netzverhalten wird dem Sender-PC ein Video-Signal zugespielt, das dieser kodiert und überträgt. Der Standard-Modus ist dabei die Übertragung komprimierter Blöcke (s. 4.4.1, S. 80), d. h. ohne Adaption wird stets dieser Modus eingesetzt. Der Verlust von Blöcken führt dabei zu sog. Artefakten, die sich sehr störend auf die erfahrene Qualität auswirken (Gl. 4.7, S. 83).

Bei allen Messungen wurde dasselbe Video-Signal von einem Videorekorder zugespielt, der Anfang eines Messeberichts. Bei der automatischen Durchführung der Messungen steuerte ein PC den Videorekorder über eine Infrarot-Sendediode fern[2], so dass stets derselbe Ausschnitt übertragen wurde. Jedoch ist es nicht möglich, die Übertragung absolut zeitgleich mit dem Abspielen des Signals zu starten, daher sind die übertragenen Daten (und somit auch bei gleichen Kodierungsparametern die Sende-Bitrate) nicht völlig identisch.

A.3.1 Konstantes Netzverhalten

Bevor die Leistung eines Algorithmus bezüglich eines variablen Netzverhaltens untersucht wird, wird die Fähigkeit zur Anpassung an ein konstantes Verhalten des Netzes betrachtet (s. 4.5.2.1, S. 93). Zuerst werden Übertragungen ohne Anpassung durchgeführt und die Ergebnisse mit der Übertragung mit aktivem Adaptionsalgorithmus verglichen.

Ein weiteres Ziel ist die Abschätzung des Einflusses des Netz-Emulators (*The Cloud* oder *NIST Net*, s. 4.5.1, S. 88) auf die Ergebnisse und die Erklärung möglicher Unterschiede.

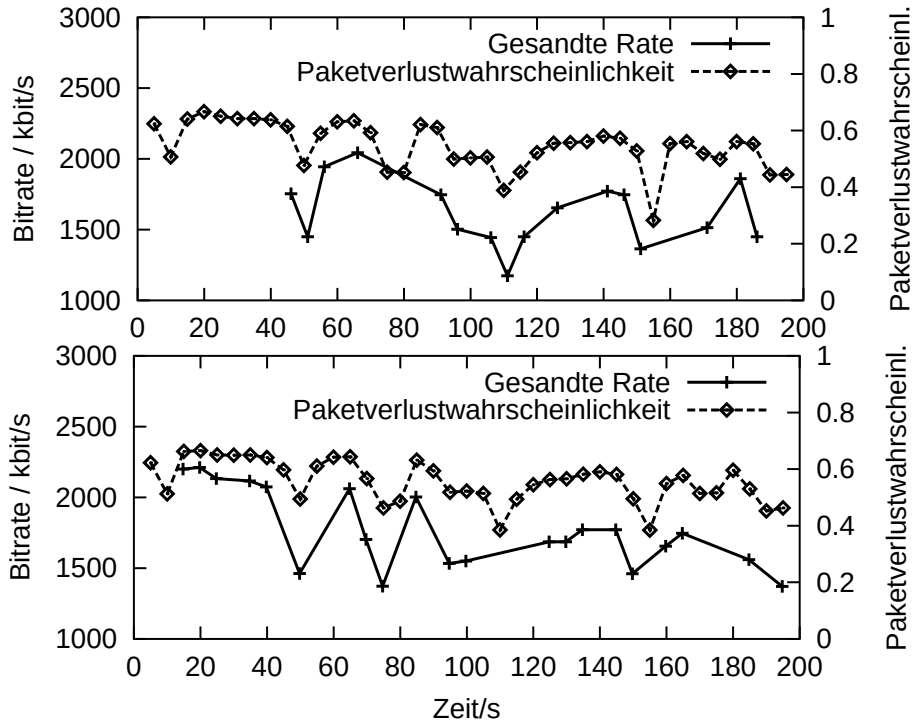


Abbildung A.17: Erzeugte Bitrate und Paketverluste zweier Messungen ohne Regelung

[2] Siehe „Linux Infrared Remote Control“, http://www.lirc.org

Abbildung A.17 zeigt zwei Messungen mit identischen Kodierungsparametern ohne Anpassung (der Quantisierungsfaktor P_Q = 100 bleibt konstant). Die Senderate variiert mit der Änderung des Bildinhaltes und wird in UDP-Statuspaketen an den Empfänger gemeldet. Von diesen Paketen können einige verloren gehen (in der Abbildung erkennbar durch fehlende Markierungen), daher kann sich die Regelung nicht auf diese Meldungen verlassen. Die Paketverluste (hier hervorgerufen durch die Begrenzung der Bitrate im Netz auf 768 kbit/s) werden dagegen lokal im Empfänger mittels der Sequenznummern ermittelt und es zeigt sich, dass sie gut reproduzierbar sind, also die Vergleichbarkeit der Experimente gegeben ist.

Weitere gemessene Werte sind hier eine Rahmenkodierungszeit (ALQ/FrameCodingDuration) von ca. 6–9 ms, einer Komprimierzeit (CLQ/CompressionDuration) von etwa 6 ms, einer Dekomprimierzeit (CLQ/DecompressionDuration) von 1 ms und einer Dekodierzeit (ALQ/DecodingDuration) von 6 ms je Bild bei einer Bildgröße von 384 auf 288 Bildpunkten. Die benötigte Dauer für die Bearbeitung der Bilder sind mit ca. 15 ms im Sender und unter 10 ms im Empfänger klein im Vergleich zu den Verzögerungen, die z. B. im Netz verursacht werden. Daher wurde hier nicht versucht, diese Werte zu optimieren.

Abbildung A.18 zeigt, dass sich die Paketverluste direkt auf die Bildstörungen (Artefakte) auswirken und so die Dienstgüte reduzieren, die nur einen Wert von $\overline{ULQ}$ = 1.8 ($\widetilde{ULQ}$ = 2) erreicht.

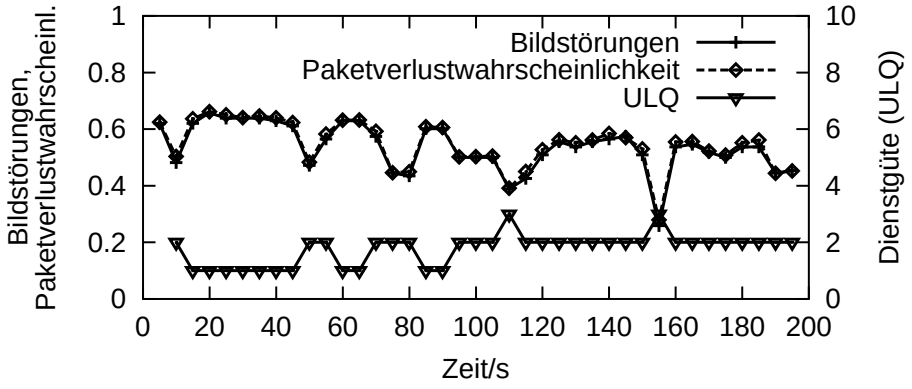


Abbildung A.18: Szenario 1.1 ohne Anpassung – Paketverluste, Bildstörungen (überlagert) und Dienstgüte

Wird nun die Sende-Bitrate durch den Adaptionsalgorithmus geregelt (Abb. A.19 oben), so wird zum einen die Bildqualität über den Quantisierungsfaktor P_Q (ALQ/PictureQuality) dergestalt verändert, dass die Zahl der Bilder pro Sekunde sich dem Ziel-Wert (hier 14 Bilder/s) annähert (Abb. A.19 Mitte). Durch die Änderungen im Bildsignal erhöht sich die Größe der kodierten Rahmen und somit auch bei gleicher Bildrate die Senderate. Ohne Regelung führt diese Erhöhung zu großen Verlusten (z. B. in Abb. A.17, t = 120 s bis t = 150 s), mit Regelung und einer Begrenzung der Senderate zu einer Reduktion der Bildrate. Der Algorithmus reagiert, indem er die Bildqualität reduziert und damit die Länge der kodierten Rahmen verkleinert, was die Bildrate steigen lässt. Erreicht die Bildrate dagegen bei konstanter Bildqualität den Zielwert, so ist das das Zeichen, dass das Videosignal stärker komprimiert werden kann und die Bildqualität wird wieder erhöht (Abb. A.19 Mitte, $t = 180\ s$).

Die Ziel-Senderate (CLQ/CLSendRate) wird vom Sender nicht erreicht; sie dient nur zur Steuerung des Sendens der Pakete und wird über die Zahl der auftretenden Paketverluste geregelt. Abbildung A.19 (unten) zeigt, wie die Rate bei kleiner Verlustwahrscheinlichkeit erhöht, bzw. bei ihrem Ansteigen erniedrigt wird. Außerdem ist sichtbar, dass zum Zeitpunkt $t = 90\ s$ auf die FEC-kodierte

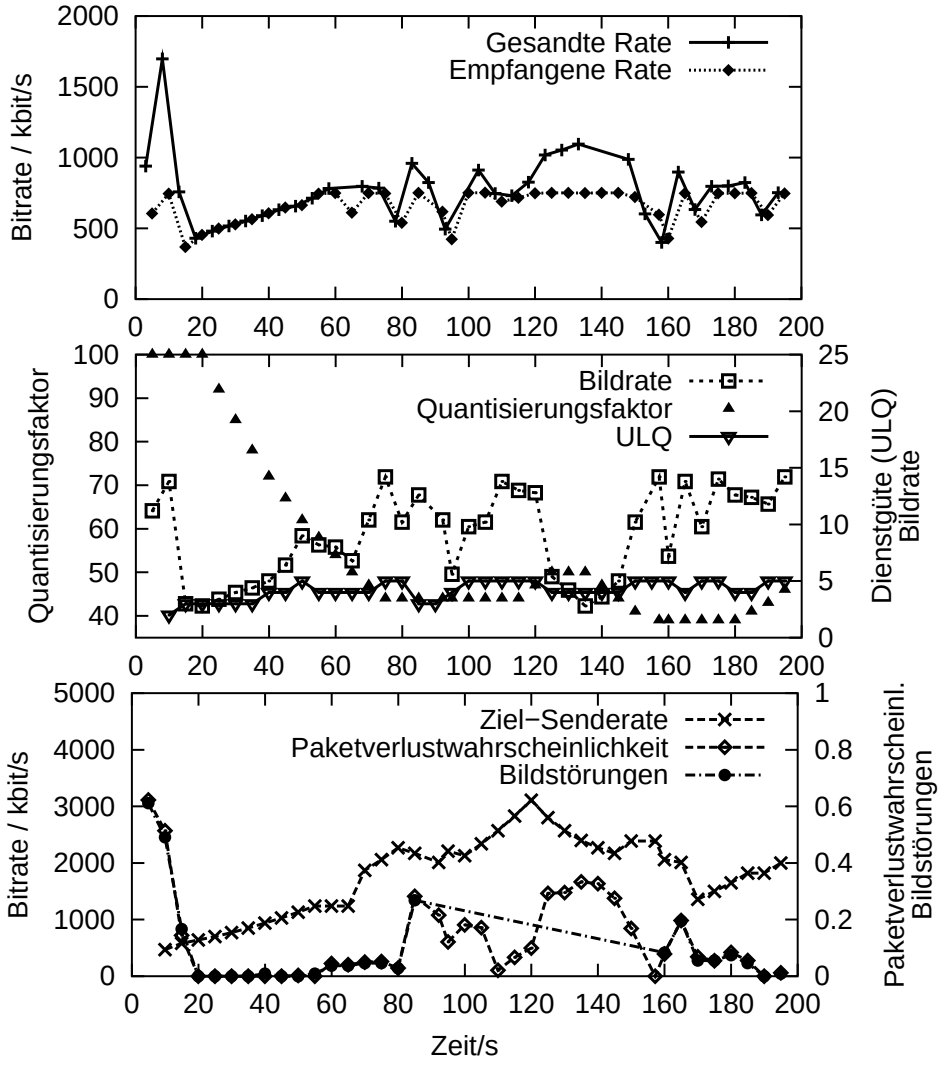


Abbildung A.19: Szenario 1.5 mit Regelung der Bitrate – Video-Bitrate (oben), Bildrate, Quantisierungsfaktor P_Q und Dienstgüte (Mitte), Ziel-Senderate, Paketverlustwahrscheinlichkeit und Bildstörungen (unten)

Übertragung umgeschaltet wurde, da nun keine Bildstörungen mehr auftreten (zu viele Paketverluste führen zu einem unbrauchbaren Bild, was sich auf die Bildrate auswirkt). Zum Zeitpunkt $t = 160\ s$ schaltet der Algorithmus wieder auf die blockweise Übertragung, da die Paketverluste stark abgenommen haben. Insgesamt verbessert sich mit Regelung die Dienstgüte deutlich auf einen Wert von $\overline{ULQ} = 4.2$ ($\widetilde{ULQ} = 4$).

Für das Szenario 1.6 ergibt sich ein sehr ähnliches Bild, woraus folgt, dass Schwankungen der Verzögerung im Netz sehr viel geringere Auswirkungen auf die Dienstgüte haben, als bei einer Audio-Übertragung. Dies bestätigt sich später auch bei der Untersuchung der Szenarien, in denen die Netz-Verzögerung variiert wird (s. A.3.2, S. 136).

Im Szenario 1.7 verwirft das Netz zusätzlich zufällig Pakete mit einer Wahrscheinlichkeit von 5 %. Sobald der Algorithmus erkannt hat, dass sich die Paketverlustwahrscheinlichkeit nicht unter einen sehr niedrigen Wert (2 %) absenken lässt, schaltet er auf die Übertragung von Rahmen mit Vorwärts-Fehlerkorrektur um (in Abb. A.20 bei $t = 105\ s$). Hierdurch treten keine Bildstörun-

gen auf, jedoch geht ein komplettes Bild verloren, wenn mehr als eines der zugehörigen Pakete den Empfänger nicht erreicht. Die Verbesserung der Dienstgüte auf einen Wert von $\overline{ULQ} = 4.1$ ($\widetilde{ULQ} = 4$) ist dennoch deutlich.

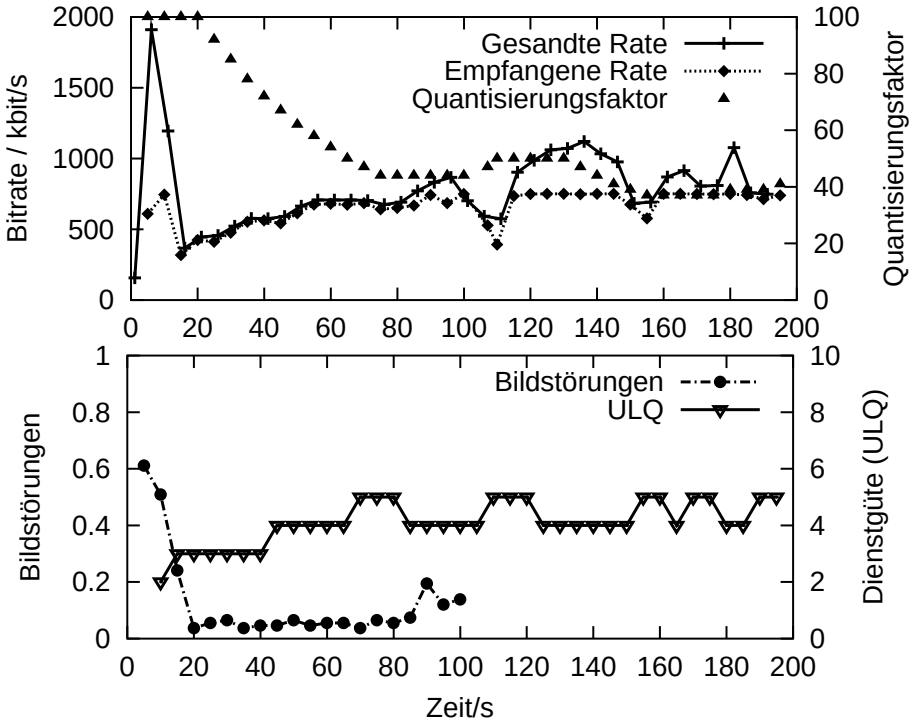


Abbildung A.20: Szenario 1.7 mit Regelung der Bitrate – Video-Bitrate, Quantisierungsfaktor P_Q (oben), Bildstörungen und Dienstgüte (unten)

Schließlich verwirft das Netz im Szenario 1.8 zufällig 10 % der Pakete. Ohne Regelung führt dies wieder auf eine Dienstgüte mit dem Wert $\overline{ULQ} = 1.8$ ($\widetilde{ULQ} = 2$), mit Adaption erreicht das Medium wieder einen Dienstgütenwert von $\overline{ULQ} = 4.1$ ($\widetilde{ULQ} = 4$).

Bei der Emulation mit *The Cloud* ergibt sich ohne Regelung ein fast identisches Bild wie für *NIST Net* (Abb. A.21). Dies unterscheidet sich von der Audio-Übertragung, da dort die Paketumordnungen sehr viel größere Auswirkungen haben. Das unterschiedliche Netzmodell wirkt sich nur leicht auf die Netz-Verzögerung aus, die langfristig nur geringfügig höher ist als bei *NIST Net* (etwa 350 ms statt 300 ms), so dass sich derselbe Durchschnittswert der Dienstgüte ergibt. Für die geregelte Übertragung ergeben sich ebenfalls sehr ähnliche Werte (s. Tab. A.9, S. 144).

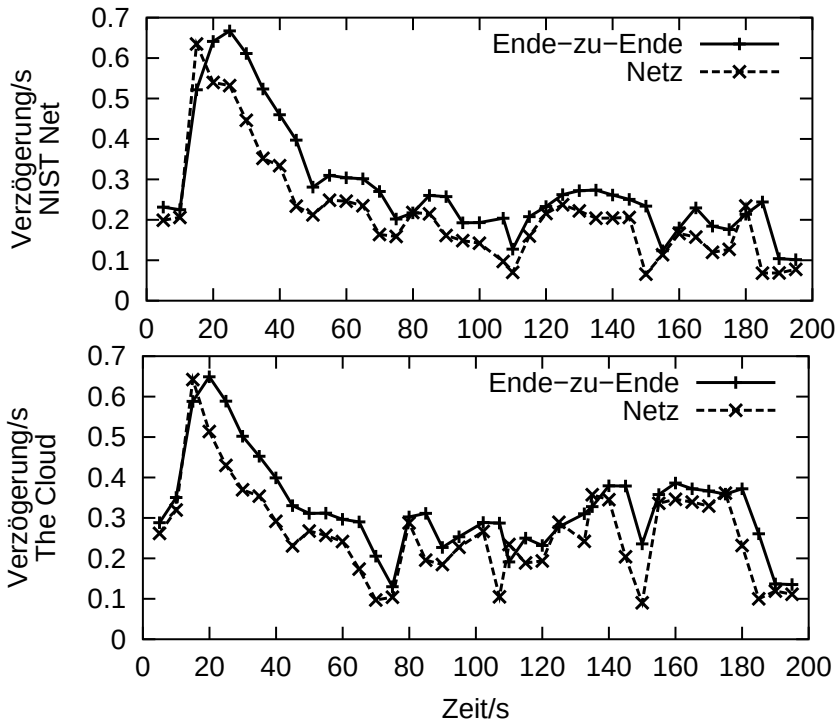


Abbildung A.21: Vergleich der erreichten Verzögerung im Netz und Ende-zu-Ende zwischen *NIST Net* (oben) und *The Cloud* (unten)

A.3.2 Verzögerungsänderung

In den ersten untersuchten Szenarien, in denen sich der Zustand des Netzes ändert, wird die Verzögerung variiert (s. 4.5.2.2, S. 94). Die beiden eingesetzten WAN-Emulationen unterscheiden sich in ihrem Verhalten wesentlich, weshalb sie getrennt betrachtet werden.

A.3.2.1 Emulation mit NIST Net

In Abb. A.22 (oben) kann an der Kurve Netz die Änderung der Netz-Verzögerung abgelesen werden. Die kurzzeitige Erhöhung führt durch das Auffüllen des Puffers im Netz zum Verwurf von Paketen und somit zu Bildstörungen, was die Dienstgüte leicht auf einen Wert von $\overline{ULQ} = 5.7$ ($\widetilde{ULQ} = 6$) verringert (Abb. A.22, unten).

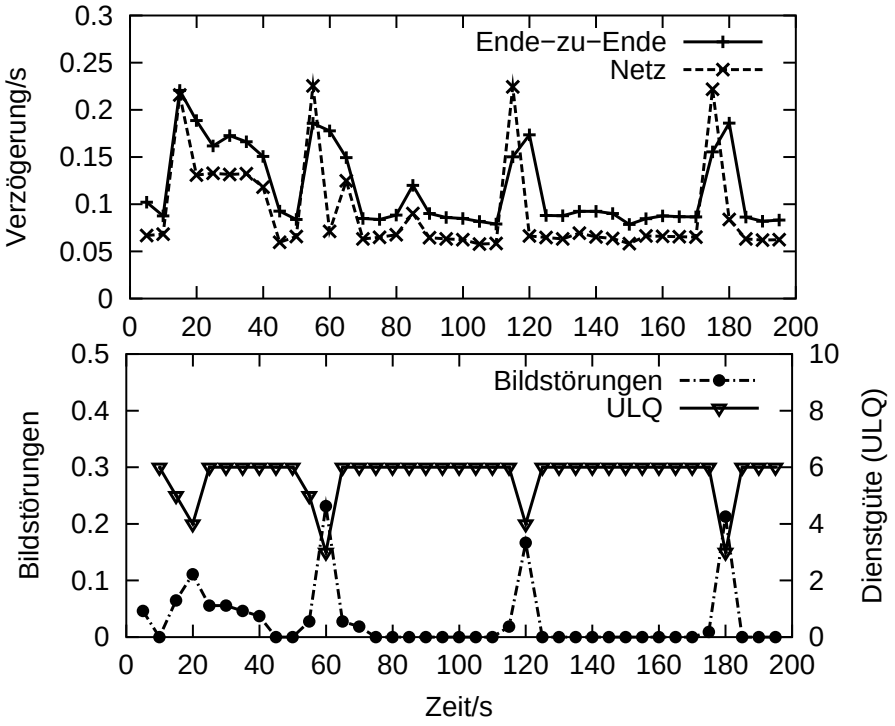


Abbildung A.22: Szenario 2.1 ohne Regelung

In den weiteren Szenarien treten deutlich mehr Paketverluste auf, da das Netz nur 20 bis 40 Pakete puffern kann (*NIST Net* DRD– bzw. *The Cloud* RED–Parameter). Dies bewirkt wieder eine große Zahl von Bildstörungen (z. B. in Szenario 2.5, Abb. A.23) und der Wert der Dienstgüte nimmt bis auf 1 ab. Da aber in den Phasen mit niedriger Verzögerung ein Dienstgütenwert von 6 erreicht wird, ergibt hier sich insgesamt noch ein Wert von $\overline{ULQ} = 3.3$ ($\widetilde{ULQ} = 4$).

Der Adaptionsalgorithmus bringt hier kaum eine Verbesserung der durchschnittlichen Dienstgüte, da er die Pufferverzögerung so einstellt, dass die Ende-zu-Ende–Verzögerung höher ist als ohne Regelung. So beträgt z. B. im Szenario 2.5 der Wert mit Adaption $\overline{ULQ} = 3.9$ ($\widetilde{ULQ} = 4$) (Abb. A.24). Jedoch hat der Einsatz des Algorithmus deutliche Vorteile: seine Verwendung reduziert die Bitrate des Video-Stroms (hier auf etwa 60 %, von ca. 1.6 Mbit/s auf 1.0 Mbit/s, Abb. A.25), der Wert der Dienstgüte ist deutlich stabiler und der Maximalwert der Bildstörun-

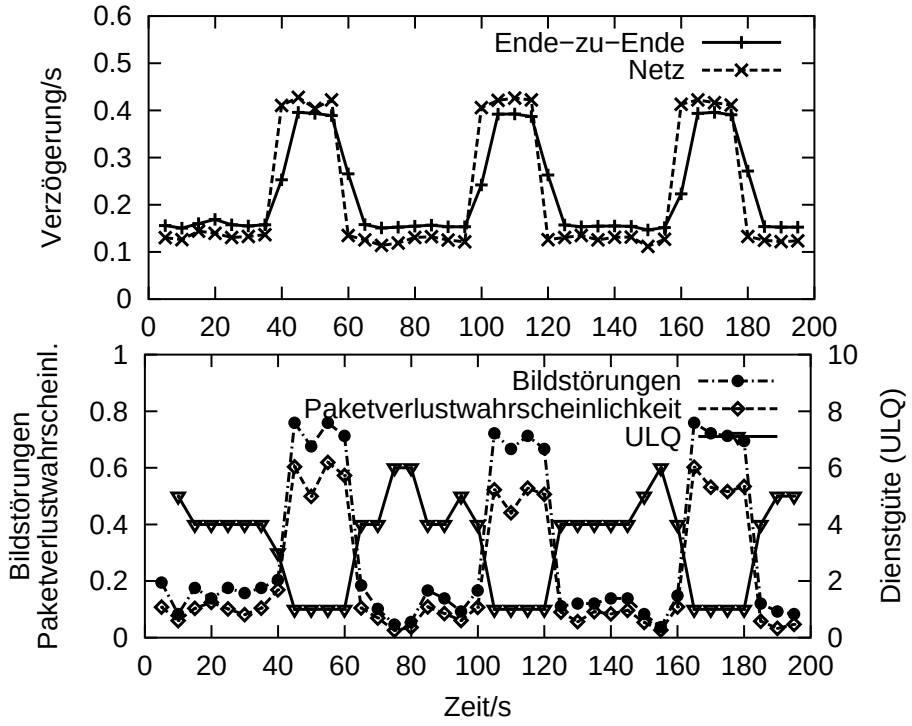


Abbildung A.23: Szenario 2.5 ohne Regelung

gen beträgt etwa 0.4 statt 0.7. Tabelle A.10, S. 145 stellt die Auswirkungen der Szenarien und des Einsatzes des Adaptionsalgorithmus auf die Dienstgüte gegenüber.

A.3.2.2 Emulation mit The Cloud

Bei der Emulation mit *The Cloud* kann im Vergleich zur Emulation mit *NIST Net* festgestellt werden, dass fast keine Paketverluste auftreten. Daher wird die Dienstgüte ohne Regelung auch nur durch die höhere Verzögerung beeinträchtigt und es ergibt sich eine kleine Verschlechterung auf Werte zwischen $\overline{ULQ} = 5.7$ und $\overline{ULQ} = 5.2$ (z. B. Abb. A.26 für Szenario 2.3).

Bei Einsatz des Adaptionsalgorithmus wird die Sende-Bitrate begrenzt (ca. 1.0 Mbit/s statt 1.6 Mbit/s) und dennoch werden praktisch identische Dienstgütenwerte erreicht (Abb. A.27). In Tab. A.11, S. 145 sind die Dienstgütenwerte zusammengetragen.

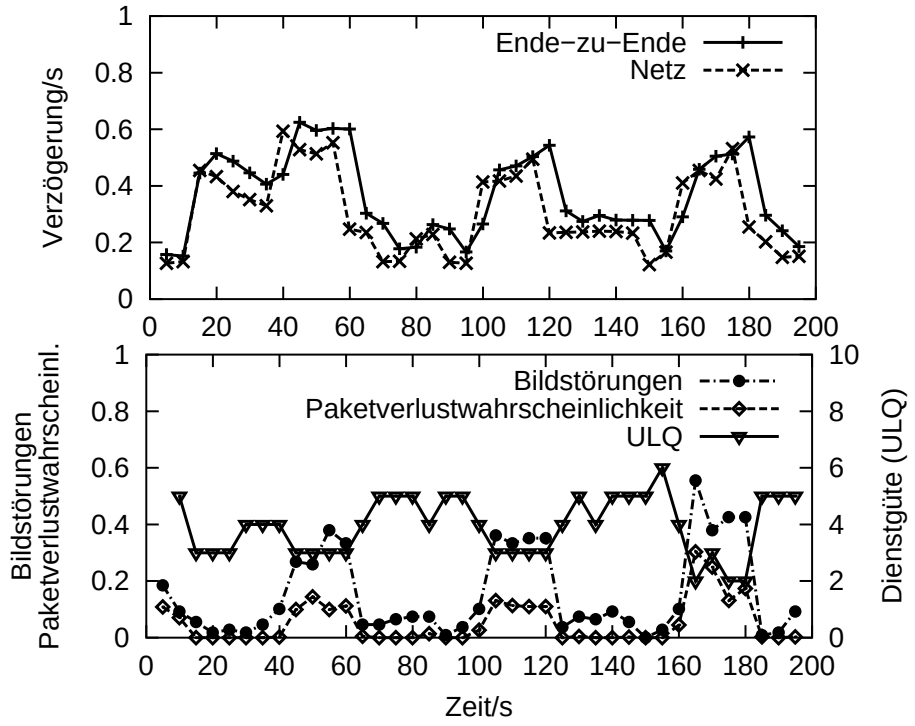


Abbildung A.24: Szenario 2.5 mit Regelung

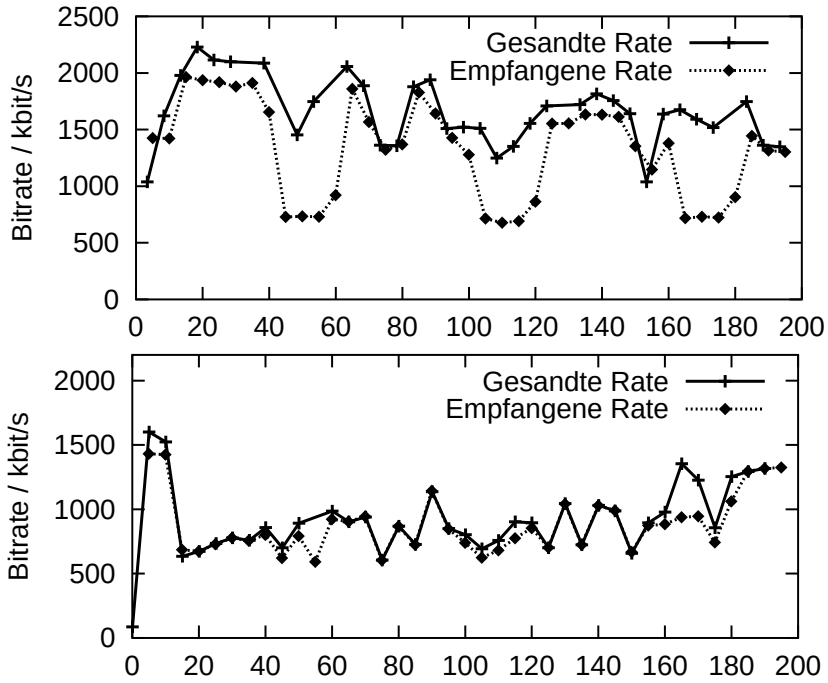


Abbildung A.25: Szenario 2.5, Vergleich der Bitrate ohne (oben) und mit Adaption (unten)

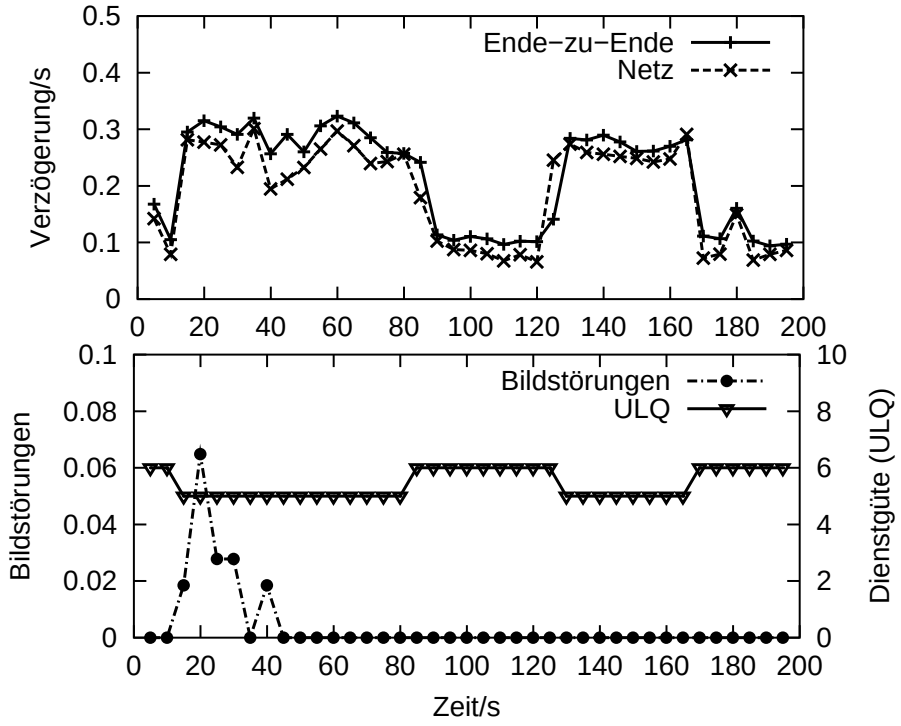


Abbildung A.26: Szenario 2.3 ohne Regelung (*The Cloud*)

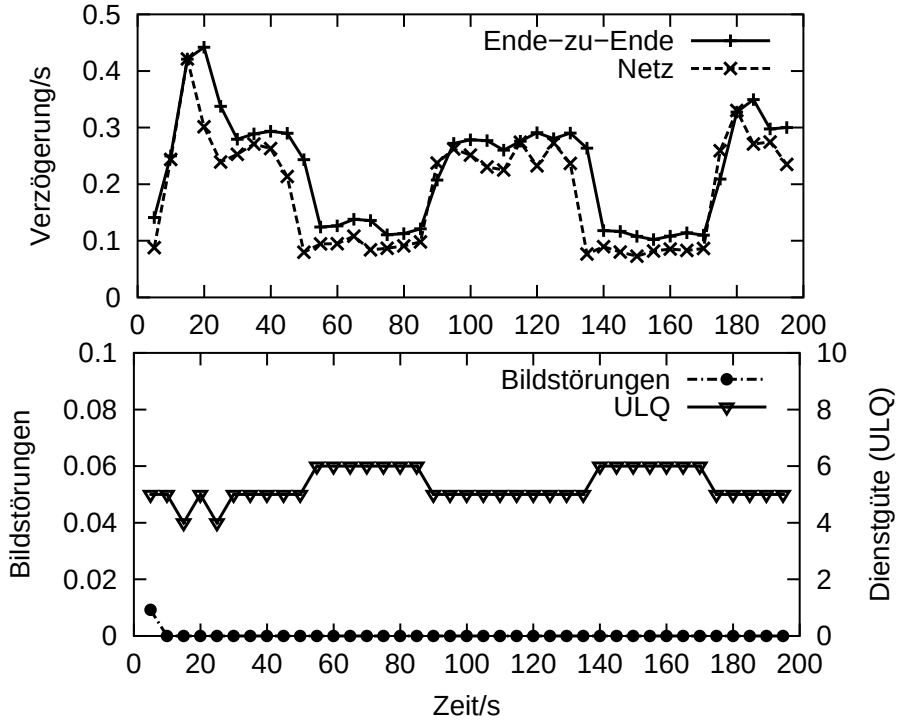


Abbildung A.27: Szenario 2.3 mit Adaption (*The Cloud*)

A.3.3 Änderung der im Netz verfügbaren Bitrate

In den Szenarien dieser Gruppe wurde die von *NIST Net* emulierte Netz-Bitrate periodisch verändert (dies ist mit *The Cloud* nicht möglich, s. 4.5.2.3, S. 95). Ohne Regelung bedeutet die Reduktion der Rate für den Video-Strom eine Erhöhung der Paketverluste und ebenso große Bildstörungen, die die Dienstgüte stark beeinträchtigt. In Phasen hoher Rate werden jedoch gute Dienstgütenwerte erreicht (Abb. A.28).

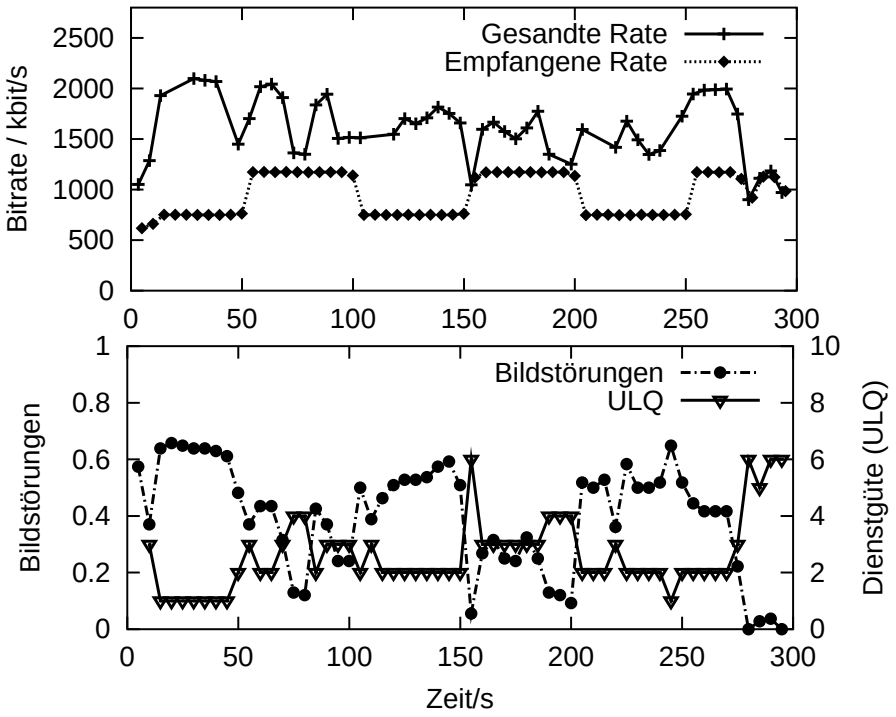


Abbildung A.28: Szenario 3.5 ohne Regelung – Video-Bitrate (oben), Bildstörungen und Dienstgüte (unten)

Die Größe des Puffers im Netz beeinflusst die Übertragung sehr stark. Im Szenario 3.4 kann das Netz zwischen 60 und 120 Pakete zwischenspeichern, was bei einer Rate von ca. 1 Mbit/s und einer Paketgröße von ca. 1500 byte eine Verzögerung von 0.8 bis 1.2 s bedeutet (Abb. A.29 oben) – diese Werte sind für eine interaktive Übertragung kaum zu akzeptieren. Der sehr kleine Puffer im Szenario 3.6 führt zwar zu einer sehr kleinen Verzögerung von etwa 0.1 s, jedoch auch zu sehr hohen Verlusten um 50 %, da die Pakete nicht mehr im Netz gepuffert werden können, und somit auch zu niedrigen Werten für die Dienstgüte (Abb. A.29 unten).

Durch den Einsatz des Adaptionsalgorithmus verbessert sich die Dienstgüte des Video-Stroms deutlich. Im Szenario 3.5 verringert er die Sende-Bitrate dergestalt, dass sich auch die Paketverluste stark reduzieren (Abb. A.30). Die Verzögerung steigt nur noch kurz an, bevor sie wieder unter 0.2 s fällt.

Der Effekt der Größe des Netz-Puffers ist nicht mehr so stark ausgeprägt wie ohne Regelung. Im Szenario 3.4 erhöht sich die Netz-Verzögerung kurzfristig auf etwa 0.8 s, jedoch gibt es fast keine Paketverluste (Abb. A.31 oben). Im Szenario 3.6 ist die Netz-Verzögerung sehr klein (unter 0.2 s), jedoch treten Paketverluste auf (Abb. A.31 unten). Tabelle A.12 zeigt die Änderung der Dienstgütenwerte.

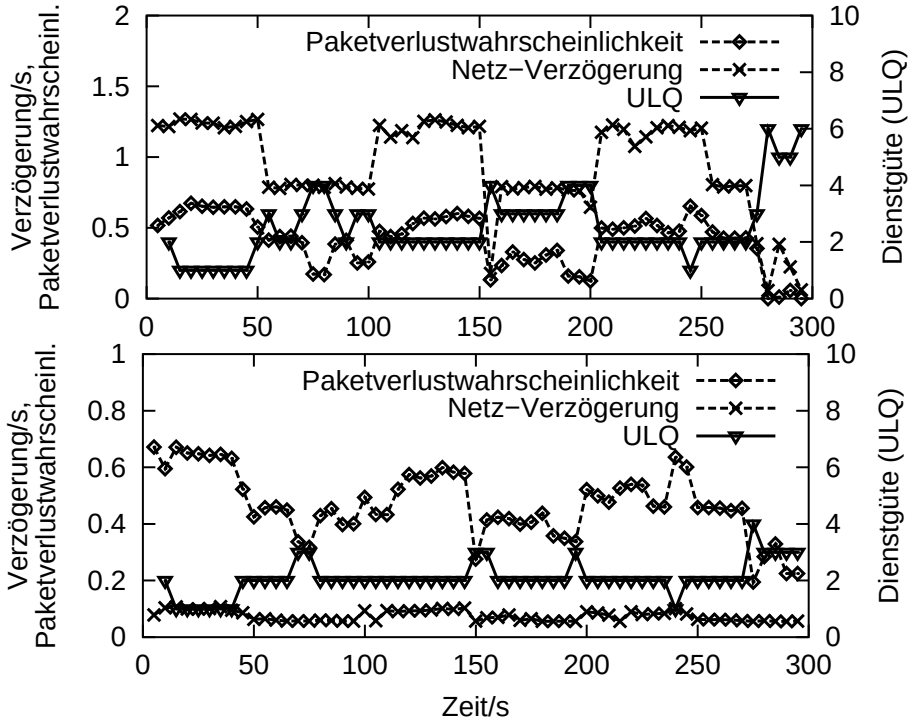


Abbildung A.29: Ohne Regelung – Paketverluste, Verzögerung und Dienstgüte in den Szenarien 3.4 (oben) und 3.6 (unten)

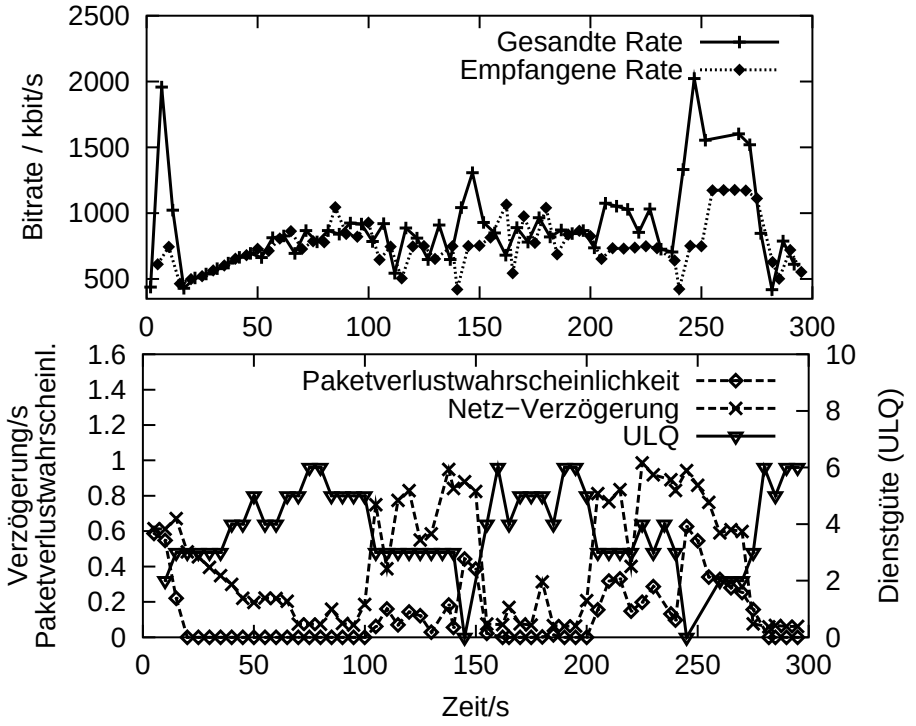


Abbildung A.30: Szenario 3.5 mit Adaption – Video-Bitrate (oben), Paketverluste, Netz-Verzögerung und Dienstgüte (unten)

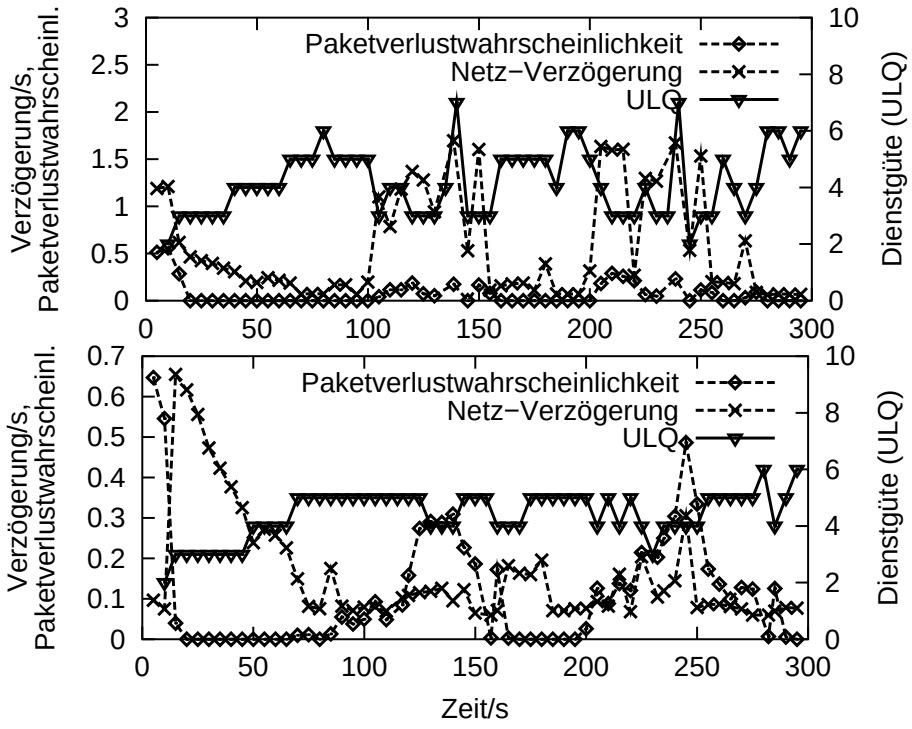


Abbildung A.31: Mit Adaption – Verzögerung, Paketverluste und Dienstgüte in den Szenarien 3.4 (oben) und 3.6 (unten)

A.3.4 Verlust-Phasen

In der letzten Szenarien–Gruppe wird die Wahrscheinlichkeit variiert, mit der das Netz Pakete zufällig verwirft. Dies ist mit *NIST Net* leicht zu implementieren, jedoch mit *The Cloud* nur eingeschränkt (s. 4.5.2.4, S. 96), daher wird dies am Ende des Abschnitts lediglich kurz dargestellt.

Ohne Regelung rufen die Paketverluste wieder direkt Bildstörungen hervor. In den Szenarien 4.1 (2 % Verlust) und 4.2 (5 % Verlust) hat dies nur geringe Auswirkung auf die Dienstgüte, deren Wert $\overline{ULQ}$ = 5.6 ($\widetilde{ULQ}$ = 6) erreicht. Auch noch höhere Verluste von 10 % im Szenario 4.3 oder bis zu 20 % im Szenario 4.5 verschlechtern die Video-Übertragung wenig (bis $\overline{ULQ}$ = 4.5 ($\widetilde{ULQ}$ = 4) in Szenario 4.5). Mit aktiviertem Adaptionsalgorithmus sollte sich daher der Wert der erreichten Dienstgüte nicht ändern, was auch der Fall ist.

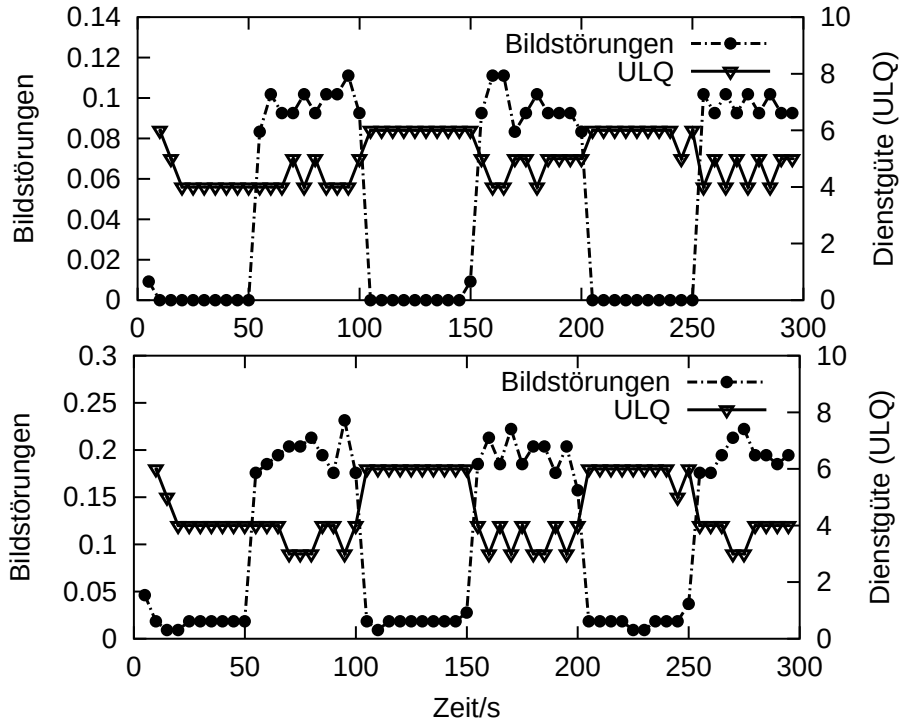


Abbildung A.32: Bildstörungen und Dienstgüte ohne Regelung in den Szenarien 4.3 (oben) und 4.5 (unten)

Abbildung A.33 kann entnommen werden, wie sich die Bildstörungen entwickeln. Im Szenario 4.3 (oben) schaltet das Netz zum Zeitpunkt t = 50 s in den Zustand mit hoher Verlustwahrscheinlichkeit (10 %, mit ebenso großen Bildstörungen). Nach kurzer Zeit erkennt dies der Algorithmus und schaltet zum Zeitpunkt $t = 90\ s$ in den FEC–Modus um, wodurch keine Bildstörungen mehr auftreten (und von der Anwendung auch nicht mehr erhoben werden), daher der Sprung zu t = 110 s, da ab hier das Netz wieder ohne Verluste überträgt. Der Algorithmus schaltet bei $t = 110s$ zurück auf die blockweise Übertragung, bis sich der Ablauf bei t = 150 s wiederholt. Für das Szenario 4.5 (Abb. A.33 unten) verhält sich die Übertragung sehr ähnlich, jedoch gehen bei 20 % Paketverlust auch einige der FEC–geschützen Bilder verloren, so dass die Bildrate nicht ganz so hoch ist wie im vorherigen Szenario. Im Vergleich zur Übertragung ohne Regelung gilt wieder, dass sich die benötigte Bitrate dramatisch reduziert (von ca. 1.5 Mbit/s auf ca. 1.0 Mbit/s), jedoch die Verzögerung höher ist (ca. 0.2 s statt 0.1 s).

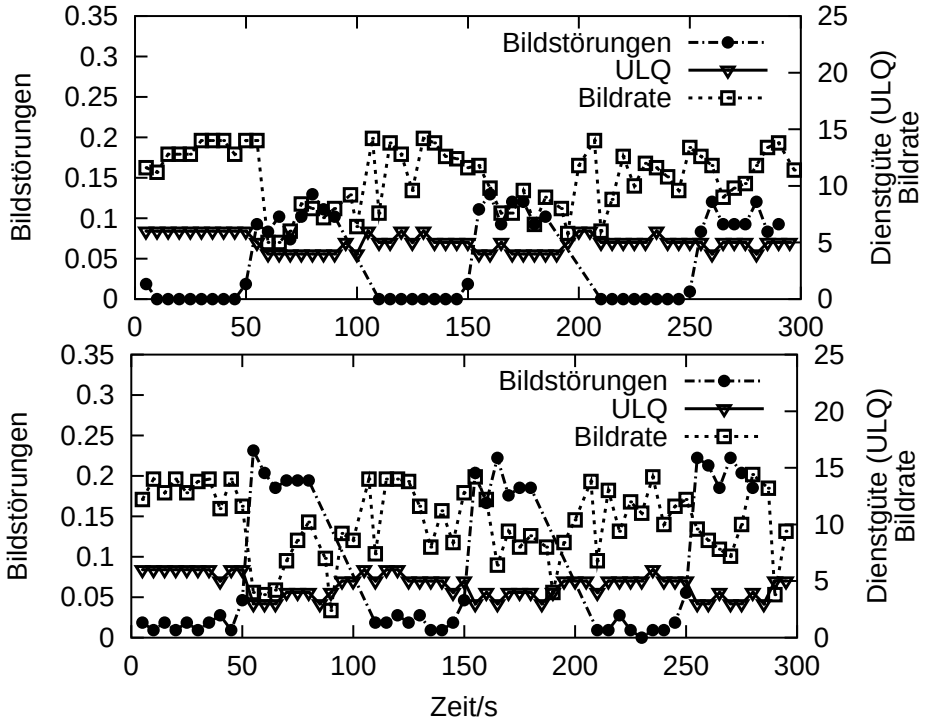


Abbildung A.33: Bildstörungen, Dienstgüte und Bildrate mit Adaption in den Szenarien 4.3 (oben) und 4.5 (unten)

Schließlich zeigt Tab. A.13 die Dienstgütenwerte für die Emulation mit *NIST Net* im Überblick. Tabelle A.14 führt die Werte bei Verwendung von *The Cloud* auf, wobei jedoch die schlechte Wiederholbarkeit des Netzverhaltens mit diesem Emulator berücksichtigt werden muss. Da die Phasen hoher Verluste zufällig aktiviert werden, traten in diesen Messungen ein bis drei solcher Phasen auf, wodurch die Ergebnisse nicht direkt vergleichbar sind.

A.3.5 Tabellarische Übersicht

Diese Tabellen zeigen die Dienstgütenwerte der verschiedenen Experimente mit und ohne Adaption. $\overline{ULQ}$ bezeichnet den durchschnittlichen Dienstgütenwert und $\widetilde{ULQ}$ den Medianwert.

Szenario	1.5	1.6	1.7	1.8
ohne Regelung	1.8 [2]	1.8 [2]	1.8 [2]	1.8 [2]
mit Regelung	4.2 [4]	3.8 [4]	4.1 [4]	4.1 [4]

Tabelle A.8: Dienstgüte $\overline{ULQ}$ und [$\widetilde{ULQ}$] für die Szenarien mit konstantem Netzverhalten (*NIST Net*)

Szenario	1.5	1.6	1.7	1.8
ohne Regelung	1.8 [2]	1.8 [2]	1.8 [2]	1.8 [2]
mit Regelung	3.6 [4]	3.5 [3]	3.9 [4]	4.0 [4]

Tabelle A.9: Dienstgüte $\overline{ULQ}$ und [$\widetilde{ULQ}$] für die Szenarien mit konstantem Netzverhalten (*The Cloud*)

Szenario	2.1	2.2	2.3	2.4	2.5	2.6
ohne Regelung	5.7 [6]	4.9 [6]	4.6 [5]	4.1 [4]	3.3 [4]	3.3 [4]
mit Regelung	4.8 [5]	4.7 [5]	4.9 [5]	4.4 [5]	3.9 [4]	3.4 [4]

Tabelle A.10: Dienstgüte $\overline{ULQ}$ und $[\widetilde{ULQ}]$ für die Szenarien mit Verzögerungsänderung (*NIST Net*)

Szenario	2.1	2.2	2.3	2.4	2.5
ohne Regelung	5.7 [6]	5.5 [6]	5.4 [5]	5.5 [6]	5.2 [5]
mit Regelung	5.8 [6]	5.5 [6]	5.3 [5]	5.4 [6]	4.5 [4]

Tabelle A.11: Dienstgüte $\overline{ULQ}$ und $[\widetilde{ULQ}]$ für die Szenarien mit Verzögerungsänderung (*The Cloud*)

Szenario	3.4	3.5	3.6
ohne Regelung	2.5 [2]	2.6 [2]	2.1 [2]
mit Regelung	4.2 [4]	3.9 [4]	4.5 [5]

Tabelle A.12: Dienstgüte $\overline{ULQ}$ und $[\widetilde{ULQ}]$ für die Szenarien mit variabler Netz-Bitrate

Szenario	4.1	4.2	4.3	4.4	4.5
ohne Regelung	5.6 [6]	5.6 [6]	5.0 [5]	5.0 [5]	4.5 [4]
mit Regelung	6.0 [6]	5.3 [5]	5.0 [5]	5.5 [6]	4.6 [5]

Tabelle A.13: Dienstgüte $\overline{ULQ}$ und $[\widetilde{ULQ}]$ für die Szenarien mit variablen Paketverlust (*NIST Net*)

Szenario	4.1	4.2	4.3	4.4	4.5
ohne Regelung	5.9 [6]	5.9 [6]	5.4 [6]	5.3 [6]	4.5 [4]
mit Regelung	5.9 [6]	5.2 [5]	4.8 [5]	5.1 [5]	4.4 [4]

Tabelle A.14: Dienstgüte $\overline{ULQ}$ und $[\widetilde{ULQ}]$ für die Szenarien mit variablen Paketverlust (*The Cloud*, Anmerkungen S. 97 beachten!)

A.4 Messaufbau

Für die Messungen wurden jeweils drei PC eingesetzt. Auf PC1 und PC2 lief die Anwendung „`terminal`“ unter Linux, als WAN-Emulator diente PC3 mit den Programmen *NIST Net* unter Linux oder *The Cloud* unter Windows NT. Der Emulator-PC besaß zwei Ethernet-Karten, so dass zwei Segmente gebildet werden konnten, an die ausschließlich der Emulator-PC und PC1 bzw. PC2 angeschlossen waren. Um die WAN-Emulationen *The Cloud* und *NIST Net* einsetzen zu können, müssen in den Endgeräten folgende Einstellungen vorgenommen werden:

Den Ethernet-Karten werden IP-Adressen zugewiesen, die in verschiedenen Sub-Netzen liegen:

PC1 `10.0.1.1`
PC2 `10.0.2.2`
PC3 `10.0.1.3` und `10.0.2.3`

Pakete werden über den Emulator-PC in das andere Segment geroutet:

```
PC1$ route add -net 10.0.2.0 netmask 255.255.255.0 gw 10.0.1.3
PC2$ route add -net 10.0.1.0 netmask 255.255.255.0 gw 10.0.2.3
```

Bei Verwendung von *NIST Net* muss darauf geachtet werden, dass „IP-Forwarding“ aktiviert ist, der PC also als Router fungiert. Dies wird unter Linux 2.4.⋆ durch das folgende Kommando erreicht:

```
echo 1 > /proc/sys/net/ipv4/ip_forward
```

Ein Szenario wird unter *NIST Net* über ein Skript gesteuert. Die Parameter für die emulierte Bitrate, Verzögerung und Paketverlustwahrscheinlichkeit werden mittels des Befehls `cnistnet` an den Emulator übergeben. Änderungen im Verhalten werden zeitgesteuert veranlasst, so dass das gewünschte Verhalten erreicht wird. Das Verhalten von *The Cloud* lässt sich dagegen nicht während der Messung verändern, sondern das gewünschte Verhalten muss jeweils von Hand eingestellt werden (s. 4.5.1.1, S. 89).

Gesteuert wurden die Experimente vom Empfänger-PC aus (PC2), auf dem ein sog. „Shell-Skript“ ablief. Für *NIST Net* startet dieses Skript über eine `telnet`–Verbindung ein Szenario-Skript auf dem Emulator-PC, das das gewünschte Netz-Szenario ablaufen lässt. Parallel dazu wird die EXPERIMENTATION PLATFORM gestartet und ggf. der Videorekorder über eine Infrarot-Schnittstelle in den Ausgangszustand gebracht. Während der eigentlichen Messung schreibt das Terminal-Programm Zustandsdaten in Dateien, die später ausgewertet werden können (Kap. 5, S. 103).

B Implementierung der eXPERIMENTATION PLATFORM

B.1 Allgemeines

Die eXPERIMENTATION PLATFORM wurde objekt-orientiert entworfen und in C++ implementiert. Nun werden zunächst die Klassen der Objekte vorgestellt, die von den Modulen übergeben werden und die Multimedia-Daten repräsentieren. Danach werden das Konzept der Datenfluss-Ebene erläutert und die Module der Übertragung beschrieben. Es folgen weitere Konzepte, nämlich die der Zeitstempel, Messwerte und Parameter. Schließlich wird die Funktionsweise der Verwaltungs-Ebene genauer dargestellt.

B.2 Klassen der Übergabe-Objekte

Für die Übertragung von Multimedia-Daten werden in einem Quellen-Modul (das das *Template* XP_TSourceInterface implementiert) Daten-Objekte generiert. Diese werden an ein nachgeschaltetes Senken-Modul weiter gereicht, das die Objekte bearbeitet und an weitere Module übergibt. Dabei kann sich der Übergabetyp ändern, z. B. wird aus einem unkomprimierten Bild ein komprimiertes oder aus einer Menge an Paketsegmenten ein ganzes Paket.

Die in der eXPERIMENTATION PLATFORM verwendeten Typen sind (Abb. B.1):

- XP_CVectorList: Dies ist eine generische Struktur, die mehrere Puffer im Speicher durch eine Liste mit Positionen und Längen bezeichnet. Um einen Nachrichtenkopf (engl. *header*) an den Anfang eines Datenblocks zu stellen, müssen dann die Daten nicht kopiert werden, sondern es reicht aus, einen Zeiger an den Anfang der Liste anzufügen.

- XP_CVectorBuffer: Unterklasse von XP_CVectorList, wobei der Puffer selbst auch von der Klasse verwaltet wird. Dies ist im Empfänger sinnvoll, der Daten zunächst unstrukturiert erhält und dann die Position von Headern eintragen kann.

- XP_CSegmentBuffer: Unterklasse von XP_CVectorBuffer, die zusätzlich eine Segmentnummer speichert.

- XP_CStampedBuffer: Unterklasse von XP_CVectorBuffer, die zusätzlich einen Zeitstempel speichert.

- XP_CAudioFrame: Unterklasse von XP_CSegmentBuffer. Sie dient zur Übergabe von Audio-Daten.

- XP_CVideoFrame: Beschreibt ein Video-Bild im Speicher. Die eigentlichen Bilddaten werden über einen Zeiger angesprochen. Im Objekt werden Daten wie Bildgröße und Kodierungsformat gespeichert.
- XP_DDisplayFrameLineR565: Objekt, mit dem eine Zeile mit Bildpunkten in „RGB 565"-Kodierung (5 bit Rotwert, 6 bit Grünwert, 5 bit Blauwert) übergeben werden kann. Diese Kodierung erlaubt die direkte Darstellung auf dem Bildschirm, wobei die eigentlichen Daten per Zeiger angegeben werden.

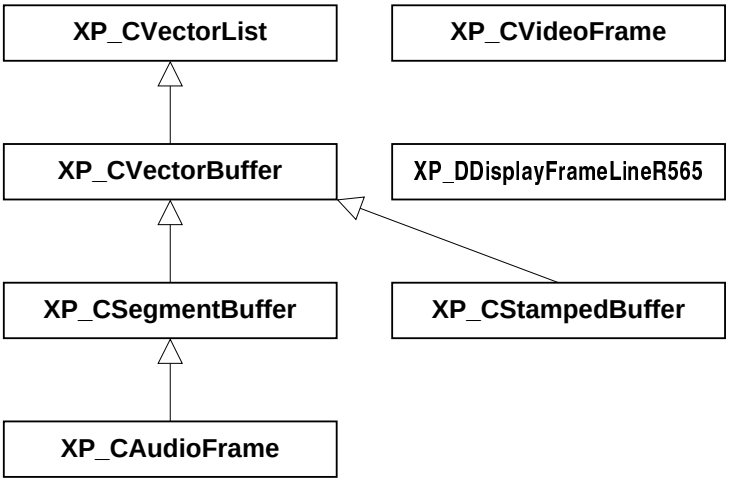


Abbildung B.1: Übergabe-Datentypen

B.3 Module der Datenfluss-Ebene

In diesem Abschnitt werden die in der EXPERIMENTATION PLATFORM verwandten Module genauer vorgestellt, die Multimedia-Daten erzeugen, bearbeiten und/oder anzeigen.

Module der Datenfluss-Ebene werden durch ein Datenübergabe-Interface gekennzeichnet. Hierzu wurden die Templates XP_TSourceInterface XP_TSinkInterface definiert, die jeweils über einen Übergabedatentyp (s. B.2, S. 147) parametrisiert werden.

Die Schnittstellen erlauben verschiedene Übergabearten:

- *„Push"*: Ein Quellen-Modul übergibt Daten an das (bzw. die) angeschlossene(n) Senken-Modul(e), indem es dessen (deren) Take-Methode(n) aufruft. Zum Beispiel übergibt das Video-Interface so Bilder zur Weiterverarbeitung.
- *„Pull"*: Das Senken-Modul fordert bei dem (einem der) angeschlossenen Quellen-Modul(e) ein Datenobjekt an.

Zudem kann das aktive Modul jeweils überprüfen, ob die gewünschte Aktion durchführbar ist oder nicht, so dass Blockierungen vermieden werden. Ist das Quellen-Modul aktiv, so wird es typischerweise die Methode PushAll aufrufen, die ein Datenobjekt an alle angeschlossenen Senken über deren Take-Methoden übergibt.

Eine weitere Besonderheit ist die Verwendung von mehreren Ausführungssträngen (engl. *threads*) bei der Erzeugung und Bearbeitung der Multimedia-Daten. Wird eine Klasse von XP_CSentThreadManager oder XP_CReceivedThreadManager abgeleitet, so wird in ihren Objekten ein Thread gestartet, der den Datentransport durch das Terminal vornimmt. Ein Medienstrom kann dabei auch von mehreren Threads bearbeitet werden, die in einem der Module Daten austauschen. Zum Beispiel erhält der Audio-Empfangspuffer XP_CAudioReceiveBuffer vom Daten-Empfangs-Thread des Sender-Stellvertreters (hier: XP_CMedTransProxyIP) Audio-Rahmen, die das Modul in einen Puffer schreibt. Der Ausspiel-Thread liest dann in regelmäßigen Abständen die Rahmen aus dem Puffer und übergibt sie an die Audio-Schnittstelle.

B.4 Zeitstempel

Die Daten eines kontinuierlichen Medienstroms müssen mit Information über ihre Erzeugungszeit versehen werden. Diese Zeitstempel (engl. *time stamps*) dienen zur zeitlichen Ordnung und zur Berechnung der Verzögerung. In der eXPERIMENTATION PLATFORM wurde für Zeitstempel die Klasse XP_CTimestamp implementiert, deren Objekte über eine Frequenz und einen Startzeitpunkt definiert werden.

Module können die momentane Zeit in Form eines Zeitstempels vom Objekt erhalten. Sie können die Differenz zwischen zwei Stempeln in absoluter Zeit ermitteln lassen oder auch bis zum Verstreichen eines Zeitraums relativ zu einem Zeitstempel warten.

B.5 Erfassung von Ereignissen und Messwerten

Während der Datenübertragung können Ereignisse eintreten, die für die Beurteilung der Dienstgüte benötigt werden. Zum Beispiel können Pakete verloren gehen oder müssen verworfen werden. Für Multimedia-Übertragungen ist es jedoch typisch, dass nur der *momentane* Zustand interessiert. Dies bedeutet dass Ereignisse, die eine gewisse Zeit in der Vergangenheit liegen, die Bewertung nicht beeinflussen sollen.

Um das Zählen solcher Ereignisse zu vereinfachen, wurde eine Klasse XP_CMeasureEvents implementiert. Diese Klasse verwaltet einen Ringpuffer, in den geordnete Ereignisse (Typ CEvent) eingetragen werden. Jedes Ereignis besteht dabei aus einem Zeitstempel und einer Zahl (beliebiger Bedeutung, z. B. als Dauer des Ereignisses interpretierbar). Bei Zugriffen wird überprüft, ob Ereignisse gespeichert sind, die vor dem betrachteten Zeitraum stattfanden. Diese werden dann aus dem Puffer entfernt.

Module können abfragen, wie viele Ereignisse im überwachten Zeitraum eingetreten sind, wobei sie zusätzlich die Summe oder den Durchschnitt der Zahlen erhalten können.

Um die Charakteristik einer Datenübertragung zu erfassen, wurde eine Klasse XP_CMeasureData implementiert, die die Verzögerung, Verluste, Verwürfe und Beschädigungen eines Datenstrom ermitteln kann. Das Modul ruft hierzu beim Eintreffen eines Datenobjekts eine Methode des Messobjekts auf und übergibt den Zeitstempel der Erzeugung. Dies erlaubt zudem die Berechnung der Verzögerungsschwankung.

B.6 Übergabe von Steuer- und Zustandsparametern

Module der Datenfluss-Ebene werden durch *Parameter* gesteuert und speichern den Zustand der Übertragung ebenfalls in Parametern. Hierzu stellt für jedes Medium dessen Dienstgüten-Manager sechs Objekte zur Verfügung: je ein XP_CQuality-Objekt für Ziel- und erreichte Dienstgüte in den Anwendungs-, Kommunikations- und Netz-Schichten (TargetALQ, TargetCLQ, TargetNLQ, AchievedALQ, AchievedCLQ, AchievedNLQ). Die Nutzer-Dienstgüte (TargetULQ und AchievedULQ) wird nur vom Dienstgüten-Manager selbst verwandt.

Beim Aufbau eines Mediums wird vom Nutzer eine Ziel-Dienstgüte vorgegeben, aus der die Meta-Policy die TargetULQ der einzelnen Medien berechnet (abhängig vom Szenario). Der Dienstgüten-Manager des Mediums bildet die ULQ auf eine TargetALQ ab, über die Methode MapULQtoALQ[1].

Bevor die eigentliche Datenübertragung gestartet wird, ruft der Manager in jedem an der Medienübertragung beteiligten Modul der Datenfluss-Ebene die Methode MakeSelection auf. Diese Methoden der Dienstgüten-Steuerungs-Ebene fragen die Zielwerte ab, die das Verhalten des Moduls beeinflussen. Parameter werden dabei über einen Bezeichner in der entsprechenden Struktur angesprochen, z. B. die Zielvorgabe für die Breite (engl. *width*) des Videobildes über[2]:

```
fWidth = ✶QoSManager()->TargetALQ().IntegerParameter( "Width")
```

Zur Ermittlung der erreichten Dienstgüte ruft der Dienstgüten-Manager die Methode MakeAbstraction der Module im Datenfluss auf, die die Zustands-Parameter, die sie während der Übertragung ermittelt haben, in die Strukturen der erreichten Dienstgüte eintragen. Zum Beispiel wird im Stellvertreter-Modul des Senders die momentane Übertragungsbitrate überwacht und in dieser Methode in AchievedNLQ abgelegt:

```
XP_CIntegerParameter ✶achievedBW =
    QoSManager()->AchievedNLQ().IntegerParameter( "Bandwidth");
✶achievedBW = observedBW;
```

B.7 Zugang zur Verwaltungs-Ebene

Während der Laufzeit lässt sich der Zustand viele Module der eXPERIMENTATION PLATFORM abfragen und ihr Verhalten ändern, da sie eine Schnittstelle zur Verwaltungs-Ebene besitzen. Dies erlaubt es zudem, gezielt bestimmte Aktionen auszulösen.

Erreicht wird dies über ein Kommandozeilen-Interface, das die Struktur der Module widerspiegelt (Tab. B.2). Der Nutzer kann mit dem Befehl[3] `cd` das aktuelle Modul wechseln und mit dem Befehl `ls` auflisten lassen, welche Befehle das Modul anbietet und welche Unter-Module sich angemeldet haben (in der Abbildung das Audio-Interface beim Anwendungs-Interface).

Um eine Multimedia-Übertragung zwischen zwei Terminals aufzubauen, muss zunächst eine Steuer-Verbindung zwischen den laufenden Anwendungen hergestellt werden. Das Nutzer-zu-Netz–Modul übernimmt dies (s. Abb. B.3).

Nach erfolgreichem Verbindungsaufbau existiert für das Sender-Endgerät ein Stellvertreter-Modul im Empfänger (analog ein Stellvertreter für den Empfänger im Sender), der über das Kommando-Interface wie ein lokales Modul angesprochen werden kann.

[1] Dies ist eine virtuelle Methode, die in den konkreten Medien-Managern implementiert wird.
[2] vereinfacht
[3] Die Namen der Befehle wurden in Anlehnung an die üblichen Unix *Shell*-Befehle gewählt.

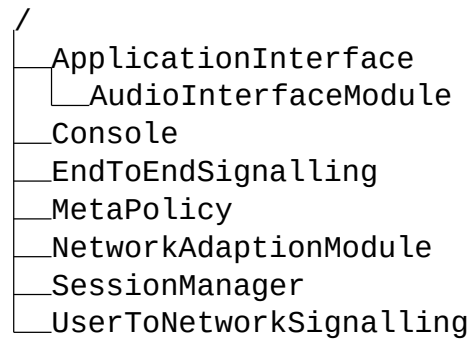


Abbildung B.2: Anwendungsstruktur im Kommando-Interface

```
XP/UserToNetworkSignalling/> ls
Commands:
ReleaseConnection <transmitter Address>
   Release signalling connection to a remote terminal.
SetupConnection <transmitter Address>
   Establish signalling connection to a remote terminal.
XP/UserToNetworkSignalling/> SetupConnection term1
```

Abbildung B.3: Aufbau einer Steuer-Verbindung

Um die Medienübertragung zu starten, wird durch Signalisierung zwischen den Endgeräten (EndToEndSignalling) für den Aufbau eines Mediums gesorgt, hier am Beispiel eines Audio-Stroms mit der Bezeichnung „`med1`“ und der Zieldienstgüte 7. Die Erzeugung des Mediums kann über weitere Parameter beeinflusst werden[4]:

```
XP/EndToEndSignalling/> SetupMedium term1 med1 audio dep 7
```

B.8 Parameter-Tabellen für Medien

B.8.1 Dienstgütenparameter für alle Medien

Die Wertetabelle Tab. B.1 für die Verzögerung wird für alle Medien-Typen verwandt. Die Auswahl der Werte wurde aus [PDLZ99] und [Lu00] abgeleitet und legt großen Wert auf die Interaktivität der Kommunikation.

B.8.2 Dienstgüte der Audio-Übertragung

In Tab. B.2 findet sich für einige Audio-Typen das Audio-Format, die Abtastfrequenz, ob der Typ in Stereo-Übertragungen eingesetzt werden kann (dann würde zu k 128 addiert und die Bitrate sowie ULQ sich ändern), Bitrate und Qualität. Tab. B.4 zeigt die Gewichte, die in Gl. 4.3, S. 72 verwandt werden. Die Abbildung von den Kodierungsparametern auf die Qualität wird in Tab. B.3

[4]Es gibt zusätzlich die Möglichkeit, ein „abhängiges Medium“ zu definieren. Dies dient zur Priorisierung der Medienströme.

Delay	ULQ
480 ms	1
400 ms	2
320 ms	3
240 ms	4
160 ms	5
80 ms	6
70 ms	7
60 ms	8
40 ms	9
20 ms	10

Tabelle B.1: Dienstgüte für Verzögerung

vorgestellt. Abb. B.4 zeigt die 32 möglichen Abtastfrequenzen für die Formate PCM8, PCM16 und ADPCM.

Typ k	Abtastwerte	Frequenz	Stereo	Bitrate[5]	ULQ
0	8 bit	4000 Hz	mögl.	32 kbit/s	4.65
1	8 bit	4325 Hz	mögl.	34.6 kbit/s	4.73
⋮					
31	8 bit	48000 Hz	mögl.	384 kbit/s	9.15
32	16 bit	4000 Hz	mögl.	64 kbit/s	5.35
⋮					
63	16 bit	48000 Hz	mögl.	768 kbit/s	9.85
64	ADPCM	4000 Hz	nein	16.6 kbit/s	3.60
⋮					
95	ADPCM	48000 Hz	nein	192.6 kbit/s	8.10
96	GSM	8000 Hz	mögl.	13.2 kbit/s	1.35
97	G.711	8000 Hz	mögl.	64 kbit/s	5.65
104	`FEC0` {64, 64}	4000 Hz	nein	34 kbit/s	3.60
105	`FEC1` {67, 67}	5100 Hz	nein	42.8 kbit/s	3.87
⋮					
114	`FEC10` {15, 79}	13300 Hz	nein	161 kbit/s	7.68

Tabelle B.2: Beispiele für Audio-Typen

Das Auftreten von Ausspielpausen stört die Audio-Übertragung stark, daher wird der Dienstgütenwert von Q_P (Gl. B.1) begrenzt. In diese Berechnung gehen die gemessenen Werte für die Länge der Pausen (Tab. B.5 links) und die Anzahl der Pausen pro Sekunde (Tab. B.5 rechts) ein. Die Werte wurden in einem Experiment ermittelt.

[5] Auch abhängig von der Rahmen-Größe

AudioDataFormat	ULQ
PCM16	10
MPEG	9
PCM8	8
μ-law	6
ADPCM	5

DSPStereo	ULQ
Stereo	10
mono	9

SamplingFrequency	ULQ
4000 Hz	1
6000 Hz	2
8000 Hz	3
9000 Hz	4
10000 Hz	5
11025 Hz	6
13000 Hz	7
17500 Hz	8
22050 Hz	9
44100 Hz	10

Tabelle B.3: Zuordnung Qualität zu Kodierungsparametern

Gewicht	
w_{type}	4/7
w_{delay}	3/7
w_{freq}	0.5
w_{format}	0.35
w_{stereo}	0.15

Tabelle B.4: Gewichtung der Kodierungsparameter in Gl. 4.3 und 4.2

AudioPauseLength	ULQ
100 ms	1
75 ms	2
50 ms	3
35 ms	4
20 ms	5
10 ms	6
5 ms	7
2 ms	8
1 ms	9
0.5 ms	10

AudioPauseCount	ULQ
200/s	1
150/s	2
100/s	3
50/s	4
35/s	5
20/s	6
10/s	7
5/s	8
2/s	9
1/s	10

Tabelle B.5: Zuordnung Ausspielpausen zu Qualität

$$\begin{aligned} Q_P &= f(\mathit{Pausenlänge}, \mathit{Pausenzahl}) \\ &= \frac{1}{8} \cdot Q(\mathsf{AudioPauseLength}) \cdot Q(\mathsf{AudioPauseCount}) \end{aligned} \tag{B.1}$$

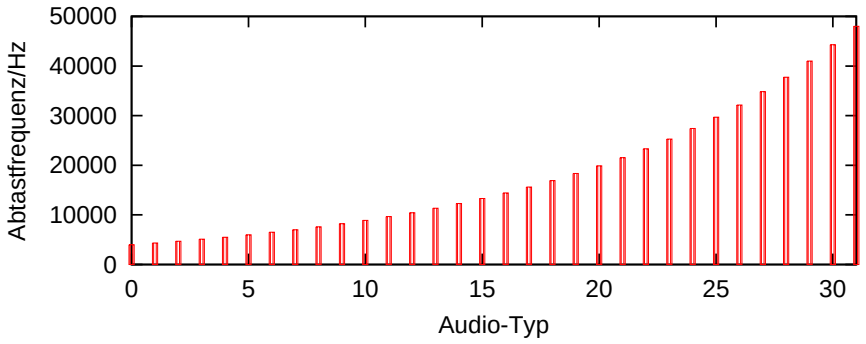


Abbildung B.4: Verwandte Abtastfrequenzen

B.8.3 Dienstgüte der Video-Übertragung

Tabelle B.6 zeigt die Dienstgüte, die den Parametern der übertragenen Video-Rahmen zugewiesen wird[6]. In Tab. B.7 stehen die ULQ-Werte, die der Bildrate zugewiesen werden und Tab. B.8 führt die Gewichtung der Parameter gegeneinander auf.

Width	ULQ
96 pixel	1
112 pixel	2
144 pixel	3
192 pixel	4
272 pixel	5
320 pixel	6
384 pixel	7
576 pixel	8
640 pixel	9
768 pixel	10

Height	ULQ
64 pixel	1
80 pixel	2
112 pixel	3
144 pixel	4
208 pixel	5
240 pixel	6
288 pixel	7
432 pixel	8
480 pixel	9
576 pixel	10

PictureQuality	ULQ
12	1
20	2
24	3
38	4
50	5
75	6
100	7
150	8
199	9
255	10

Tabelle B.6: Zuordnung Bild-Parameter zu Qualität

Gewicht	
w_{fps}	2/9
w_{delay}	1/3
w_{PQ}	2/9
w_{PW}	1/9
w_{PH}	1/9

Tabelle B.8: Gewichtung der Video-Parameter

[6] Das Verhältnis Höhe zu Breite bleibt konstant, jedoch müssen die Werte durch 16 teilbar sein.

FramesPerSecond	ULQ
2 fps	1
3 fps	2
5 fps	3
7 fps	4
9 fps	5
12 fps	6
14 fps	7
17 fps	8
20 fps	9
25 fps	10

PictureDistortion	ULQ
0.80	1
0.60	2
0.40	3
0.20	4
0.10	5
0.07	6
0.05	7
0.03	8
0.01	9
0.00	10

Tabelle B.7: ULQ der Bildrate und Bildstörungen

C Entwurf Simulations-Untersuchungen

Während eine umfangreiche simulative Untersuchung von Adaptionsalgorithmen nicht im Rahmen dieser Arbeit durchführbar war, konnten jedoch Simulationen bereits entworfen und erste Versuche durchgeführt werden (Diplomarbeit Monica Andreu Vazquez, IND Uni Stuttgart, 2001). Ziel der Simulationen ist die Untersuchung von Zusammenhängen, die experimentell nur schwer zu betrachten sind. Dies wird ermöglicht durch die vollständige Kontrolle, die in einer Simulation über das Szenario ausgeübt werden kann.

Die Implementierung verwendet die IND–Simulations-Bibliothek. Mit ihrer Hilfe wurde eine Multimedia-Übertragung modelliert, d. h. die Erzeugung von Audio- und Video-Rahmen, die Bearbeitung dieser Rahmen und die Übertragung über ein simuliertes Netz – inklusive Effekten wie Paketverlust und Verzögerungsschwankung. Es wird ein empfangendes Terminal simuliert, das die Daten puffert und ausspielt, wobei eine Überwachung der erreichten Dienstgüte stattfindet. Es wurde eine Überwachungs- und Regelungseinheit implementiert, die verschiedene Adaptionsalgorithmen einsetzen kann, um die Übertragung an das Verhalten des Netzes anzupassen.

Ein wichtiges Ergebnis der Untersuchungen war, dass die Vereinfachungen und Abstraktionen, die in der Simulation notwendig sind, sehr großen Einfluss auf das Verhalten der Algorithmen haben. Daher sind ausführliche und zeitaufwändige Untersuchungen erforderlich, die aufbauend auf den vorhandenen Überlegungen durchführbar sind.

Literaturverzeichnis

[BBCea98] Blake, S.; Black, D.; Carlson, M.; Davies, E.; Wang, Z.; Weiss, W.: *An architecture for differentiated services*, RFC 2475, IETF, December 1998.

[BCS94] Braden, R.; Clark, D.; Shenker, S.: *Integrated services in the Internet architecture: an overview*, RFC 1633, IETF, July 1994.

[BDHea93] Barth, I.; Dermler, G.; Helbig, T.; Rothermel, K.; Sembach, F.; Wahl, T.: *CINEMA: Eine konfigurierbare, integrierte Multimedia-Architektur*, Beiträge zum GI/ITG Arbeitstreffen Verteilte Multimedia Systeme, Stuttgart, D, Band 5, K.G. Saur, Februar 1993, ISBN 3-598-22407-9, S. 33–47.

[BHHea99] Benz, M.; Hess, R.; Hutschenreuther, T.; Kümmel, S.; Schill, A.: *VideoTransmissionToolkit – ein Resultat performanceorientierten Systementwurfs*, Fachzeitschrift Praxis der Informationsverarbeitung und Kommunikation (PIK), Vol. 22, No. 2, Apr-Jun 1999, pp. 77–86.

[BK99] Bhatti, S. N.; Knight, G.: *Enabling QoS adaptation decisions for Internet applications*, Computer Networks, Vol. 31, No. 7, Elsevier, Amsterdam, NL, April 1999, ISSN 1389-1286, pp. 669–692.

[Bor00] Borella, M. S.: *Source models of network game traffic*, Computer Communications, Vol. 23, No. 4, Butterworth-Heinemann Ltd., February 2000, pp. 403–410.

[Böt98] Böttiger, A.: *Regelungstechnik: eine Einführung für Ingenieure und Naturwissenschaftler*, R. Oldenbourg, München, D, 1998, ISBN 3-486-24530-9.

[BPS99] Bennett, J. C. R.; Partridge, C.; Shectman, N.: *Packet reordering is not pathological network behavior*, IEEE/ACM Transactions on Networking, Vol. 7, No. 6, December 1999, pp. 789–798.

[BRS99] Balakrishnan, H.; Rahul, H. S.; Seshan, S.: *An integrated congestion management architecture for Internet hosts*, Proceedings of ACM SIGCOMM ’99, Cambridge, MA, USA, September 1999.

[BRS00] Bechler, M.; Ritter, H.; Schiller, J. H.: *Quality of Service in mobile and wireless networks: the need for proactive and adaptive applications*, Proceedings of the 33rd Hawaii International Conference on System Sciences (HICSS’00), Maui, USA, January 2000.

[BT94] Bolot, J. C.; Turletti, T.: *A rate control mechanism for packet video in the Internet*, Proceedings of IEEE INFOCOM '94, Toronto, Canada, Volume 1-3, IEEE Computer Society Press, June 1994, ISBN 0-8786-5570-4, pp. 1216–1223.

[CAH96] Campbell, A.; Aurrecoechea, C.; Hauw, L.: *A Review of QoS Architectures*, ftp://ftp.ctr.columbia.edu/CTR-Research/comet/public/papers/96/ CAM96a.ps.gz, March 1996.

[CK96] Cox, R. V.; Kroon, P.: *Low Bit-Rate Speech Coders for Multimedia Communication*, IEEE Communications Magazine, Vol. 34, No. 12, IEEE, New York, USA, December 1996, ISSN 0163-6804, pp. 34–41.

[Clo] *Cloud, The*, http://www.shunra.com/.

[Com00] Comer, D.: *Internetworking with TCP/IP: Volume I; Principles, protocols, and architecture*, Prentice Hall, Upper Saddle River, USA, 2000, ISBN 0-13-018380-6.

[Cox97] Cox, R. V.: *Three New Speech Coders from the ITU Cover a Range of Applications*, IEEE Communications Magazine, Vol. 35, No. 9, IEEE, New York, USA, September 1997, ISSN 0163-6804, pp. 40–47.

[CPS95] Callaghan, B.; Pawlowski, B.; Staubach, P.: *Network File System (NFS) version 3 specification RFC 1813*, IETF, June 1995.

[CT90] Clark, D. D.; Tennenhouse, D. L.: *Architectural Considerations for a New Generation of Protocols*, Proceedings of ACM SIGCOMM '90, Philadelphia, USA, September 1990, pp. 200–208.

[Dee89] Deering, S.: *Host Extensions for IP Multicasting RFC 1112*, IETF, August 1989.

[DFGG96] Demeure, I.; Farhat-Gissler, J.; Gasperoni, F.: *A Scheduling Framework for the Automatic Support of Temporal QoS Constraints*, http://www.infres.enst.fr/∼singhoff/DOC/QOS-ARCHI/POLKA/ demeure_96a.ps.gz, March 1996.

[DH98] Deering, S.; Hinden, R.: *Internet Protocol, version 6 (IPv6) specification RFC 2460*, IETF, December 1998.

[DRR98] Duffield, N. G.; Ramakrishnan, K. K.; Reibman, A. R.: *SAVE: An Algorithm for Smoothed Adaptive Video Over Explicit Rate Networks*, IEEE/ACM Transactions on Networking, Vol. 6, No. 6, December 1998, pp. 717–728.

[FGMea99] Fielding, R.; Gettys, J.; Mogul, J.; Frystyk, H.; Masinter, L.; Leach, P.; Berners-Lee, T.: *Hypertext Transfer Protocol – HTTP/1.1 RFC 2616*, IETF, June 1999.

[FJ93] Floyd, S.; Jacobson, V.: *Random early detection gateways for congestion avoidance*, IEEE/ACM Transactions on Networking, Vol. 1, No. 4, August 1993, pp. 397–413.

[Gay96] Gaynor, M.: *Proactive packet dropping methods for IP gateways*, http://www.eecs.harvard.edu/∼gaynor/final.ps, November 1996.

[GOFC98] Guedes, L. A.; Oliveira, P. C.; Faina, L. F.; Cardozo, E.: *An agent-based approach for supporting quality of service in distributed multimedia systems*, Computer Communications, Vol. 21, No. 14, Butterworth-Heinemann Ltd., September 1998, pp. 1269–1278.

[HBv98] Hafid, A.; Bochmann v., G.: *Quality of Service adaptation in distributed multimedia applications*, ACM/Springer Multimedia Systems, Vol. 6, No. 5, ACM/Springer, USA, 1998, pp. 299–315.

[Hir95] Hirzinger, G.: *The ROboter Technology EXperiment ROTEX*, http://www.robotic.dlr.de/TELEROBOTICS/rotex.html, April 1995.

[HJ98] Handley, M.; Jacobson, V.: *SDP: Session Description Protocol RFC 2327*, IETF, USA, April 1998.

[HSSR99] Handley, M.; Schulzrinne, H.; Schooler, E.; Rosenberg, J.: *SIP: Session Initiation Protocol RFC 2543*, IETF, USA, March 1999.

[IKW00] Ingvaldsen, T.; Klovning, E.; Wilkins, M.: *Determining the causes of end-to-end delay in CSCW applications*, Computer Communications, Vol. 23, No. 3, Butterworth-Heinemann Ltd., February 2000, pp. 219–232.

[IT94] ITU-T: *ITU-T Recommendation X.200 (07/94) – Information technology – Open Systems Interconnection – Basic reference model: The basic model*, ITU-T Rec., International Telecommunication Union, July 1994.

[IT95] ITU-T: *ITU-T Recommendation I.361 (11/95) – B-ISDN ATM Layer Specification*, ITU-T Rec., International Telecommunication Union, Geneva, November 1995.

[IT96a] ITU-T: *ITU-T Recommendation G.113 (02/96) – Transmission Impairments*, ITU-T Rec., International Telecommunication Union, February 1996.

[IT96b] ITU-T: *ITU-T Recommendation I.363.3 (8/96) – B-ISDN ATM Adaptation Layer specification: Type 3/4 AAL*, ITU-T Rec., International Telecommunication Union, Geneva, August 1996.

[ITG96] ITG: *Entwurf für die ITG-Empfehlung 5.2-01 Architekturen und Verfahren der Vermittlungstechnik (Stand: Dezember 1996)*, Begriffswerk des ITG-Fachausschusses 5.2, ITG, D, Dezember 1996.

[ITG97] ITG: *Entwurf für die ITG-Empfehlung 5.2-03 Begriffe der Nachrichtenverkehrstheorie (Stand: Oktober 1997)*, Begriffswerk des ITG-Fachausschusses 5.2, ITG, D, Oktober 1997.

[JPE] JPEG: http://www.jpeg.org/ – *Home site of the JPEG and JBIG committees*.

[KHHC97] Kouvelas, I.; Hodson, O.; Hardman, V.; Crowcroft, J.: *Redundancy control in real-time Internet audio conferencing*, http://www.cs.ucl.ac.uk/staff/O.Hodson/publications/redundancy-avspn97.ps.gz, September 1997.

[LAP99] Li, X.; Ammar, M. H.; Paul, S.: *Video Multicast over the Internet*, IEEE Network, Vol. 13, No. 2, USA, March 1999, pp. 46–60.

[LC83] Lin, S.; Costello, D. J.: *Error control coding: Fundaments and applications*, Prentice Hall, Englewood Cliffs, NJ, USA, 1983, ISBN 0-13-283796-X.

[LH87] Lelewer, D. A.; Hirschberg, D. S.: *Data compression*, ACM Computing Surveys, Vol. 19, No. 3, USA, September 1987, pp. 261–296.

[LN99] Li, B.; Nahrstedt, K.: *A control-based middleware framework for Quality-of-Service applications*, IEEE Journal on Selected Areas in Communications, Vol. 17, No. 9, USA, September 1999, pp. 1632–1650.

[Lu00] Lu, G.: *Issues and technologies for supporting multimedia communications over the Internet*, Computer Communications, Vol. 23, No. 14-15, Butterworth-Heinemann Ltd., August 2000, pp. 1323–1335.

[MB76] Metcalfe, R. M.; Boggs, D. R.: *ETHERNET: distributed packet switching for local area networks*, Computer Networks, Springer-Verlag, NL, 1976, pp. 395–404.

[Mil92] Mills, D. L.: *Network Time Protocol (NTP) version 3 specification, implementation RFC 1305*, IETF, March 1992.

[Mon83] Montgomery, W. A.: *Techniques for Packet Voice Synchronization*, IEEE Journal on Selected Areas in Communications, Vol. 1, No. 5, USA, 1983, pp. 1022–1028.

[NCN98] Nahrstedt, K.; Chu, H. H.; Narayan, S.: *QoS-aware resource management for distributed multimedia applications*, Journal of High Speed Networks, Vol. 7, No. 3,4, 1998, pp. 229–257.

[NIS01] *NIST Net network emulator home page*, http://www.antd.nist.gov/itg/nistnet/, March 2001.

[Nob00] Noble, B.: *System Support for Mobile, Adaptive Applications*, IEEE Personal Communications, Vol. 7, No. 1, Institute of Electrical and Electronics Engineers, New York, USA, February 2000, ISSN 1070-9916, pp. 44–49.

[NS95] Nahrstedt, K.; Smith, J. M.: *The QOS Broker*, IEEE Multimedia, Vol. 2, No. 1, USA, Spring 1995, pp. 53–67.

[OMRW98] Ott, M.; Michelitsch, G.; Reininger, D.; Welling, G.: *An architecture for adaptive QoS and its application to multimedia systems design*, Computer Communications, Vol. 21, No. 4, Butterworth-Heinemann Ltd., April 1998, pp. 334–349.

[PDLZ99] Perkins, M. E.; Dvorak, C. A.; Lerich, B. H.; Zebarth, J. A.: *Speech Transmission Performance Planning in Hybrid IP/SCN Networks*, IEEE Communications Magazine, Vol. 37, No. 6, IEEE, New York, USA, June 1999, ISSN 0163-6804, pp. 126–131.

[PKHea97] Perkins, C.; Kouvelas, I.; Hodson, O.; Hardman, V.; Handley, M.; Bolot, J. C.; Vega-Garcia, A.; Fosse-Parrisis, S.: *RTP payload for redundant audio data RFC 2198*, IETF, September 1997.

[Pos80] Postel, J.: *User Datagram Protocol RFC 768*, IETF, August 1980.

[Pos81] Postel, J.: *Transmission Control Protocol*, RFC 793, IETF, September 1981.

[PP96] Papadopoulos, C.; Parulkar, G. M.: *Retransmission-Based Error Control for Continuous Media Applications*, Proceedings of the 6th International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV '96), Zushi, J, April 1996.

[PSC81] Postel, J. B.; Sunshine, C. A.; Cohen, D.: *The ARPA internet protocol*, Computer Networks, Vol. 5, No. 5, Springer-Verlag, NL, 1981, pp. 261–271.

[Pur95] Purser, M.: *Introduction to Error-Correcting Codes*, The Artech House Telecommunications Library, Artech House, Boston, London, USA, GB, 1995, ISBN 0-89006-784-8.

[RH95] Rothermel, K.; Helbig, T.: *An Adaptive Stream Synchronization Protocol*, Proceedings of the 5th International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV '95), Durham, NH, USA, April 1995.

[Rij96] Rijkse, K.: *H.263: Video Coding for Low-Bit-Rate Communication*, IEEE Communications Magazine, Vol. 34, No. 12, IEEE, New York, USA, December 1996, ISSN 0163-6804, pp. 42–45.

[RKTS94] Ramjee, R.; Kurose, J.; Towsley, D.; Schulzrinne, H.: *Adaptive playout mechanisms for packetized audio applications in wide-area networks*, Proceedings of IEEE INFOCOM '94, Toronto, Canada, Volume 1-3, IEEE Computer Society Press, June 1994, ISBN 0-8786-5570-4, pp. 680–688.

[SCFJ96] Schulzrinne, H.; Casner, S.; Frederick, R.; Jacobson, V.: *RTP: a transport protocol for real-time applications*, RFC 1889, IETF, USA, January 1996.

[Sch96] Schulzrinne, H.: *RTP profile for audio and video conferences with minimal control*, RFC 1890, IETF, USA, January 1996.

[Sch98] Schoeman, J.: *RealTime JPEG library (RTJpeg)*, http://www.ee.up.ac.za/~justin/bttv/, 1998.

[SCWea99] Stiller, B.; Class, C.; Waldvogel, M.; Caronni, G.; Bauer, D.: *A flexible middleware for multimedia communication: design, implementation, and experience*, IEEE Journal on Selected Areas in Communications, Vol. 17, No. 9, USA, September 1999, pp. 1580–1598.

[SR99] Schulzrinne, H.; Rosenberg, J.: *The IETF Internet Telephony architecture and protocols*, IEEE Network, Vol. 13, No. 3, USA, May 1999, pp. 18–23.

[SS00] Springer, T.; Schill, A.: *Automated adaptation for mobile computing based on mobile agents*, Proceedings of the 3rd IFIP/GI International Conference on Trends Towards a Universal Service Market (USM 2000), Munich, D, Springer, September 2000, ISBN 3-540-41024-4, pp. 284–289.

[Ste96] Steinmetz, R.: *Human perception of jitter and media synchronization*, IEEE Journal on Selected Areas in Communications, Vol. 14, No. 1, USA, January 1996, pp. 61–72.

[Ste99] Steinmetz, R.: *Multimedia-Technologie: Grundlagen, Komponenten und Systeme*, Neue Medien, Springer-Verlag, D, 1999, ISBN 3-540-62060-5.

[Tan96] Tanenbaum, A. S.: *Computer networks*, Prentice-Hall, Upper Saddle River, USA, 1996, ISBN 0-13-394248-1.

[Tas97] Tassel, J.: *Quality of Service (QoS) adaptation using Reflective Java*, http://ftp.labs.bt.com/projects/mware/papers/jt_thesis/index.htm, September 1997.

[TO99] Toga, J.; Ott, J.: *ITU-T standardization activities for interactive multimedia communications on packet-based networks: H.323 and related recommendations*, Computer Networks, Vol. 31, No. 3, Elsevier, Amsterdam, NL, February 1999, ISSN 1389-1286, pp. 205–223.

[TSSea97] Tennenhouse, D. L.; Smith, J. M.; Sincoskie, W. D.; Wetherall, D. J.; Minden, G. J.: *A survey of active network research*, IEEE Communications Magazine, Vol. 35, No. 1, IEEE, New York, USA, January 1997, ISSN 0163-6804, pp. 80–85.

[TYS99] Treetasanatavorn, S.; Yoshida, T.; Sakai, Y.: *Media synchronization with adaptive QoS control based on a delay compensation protocol and a smoothing technique*, IEICE Transactions on Communications, Vol. 82, No. 10, J, October 1999, pp. 1595–1608.

[VKBvG95] Vogel, A.; Kerherve, B.; Bochmann v., G.; Gecsei, J.: *Distributed multimedia and QoS: a survey*, IEEE Multimedia, USA, Summer 1995, pp. 10–19.

[WS96] Watson, A.; Sasse, M. A.: *Evaluating audio and video quality in low-cost multimedia conferencing systems*, http://www.cs.ucl.ac.uk/staff/a.sasse/iwc.ps, 1996.

[WS99a] Wang, X.; Schulzrinne, H.: *Comparison of Adaptive Internet Multimedia Applications*, IEICE Transactions on Communications, Vol. 82, No. 6, J, June 1999, pp. 806–818.

[WS99b] Wilson, G.; Sasse, M. A.: *The relationship between media quality and user cost in networked multimedia applications*, Proceedings of Affective Computing Workshop, London, GB, April 1999.

[WTSB01] Wenjing, L.; Tien, C. L.; Sung, L. B.; Bin, W.: *The transmission of MPEG-2 video under usage parameter control*, International Journal of Communication Systems, Vol. 14, No. 2, John Wiley & Sons, GB, March 2001, pp. 125–146.

[XN99] Xiao, X.; Ni, L. M.: *Internet QoS: a big picture*, IEEE Network, Vol. 13, No. 2, USA, March 1999, pp. 8–18.

[YAWM00] Youssef, A.; Abdel-Wahab, H.; Maly, K.: *Enabling Quality of Session control in adaptive multimedia multicast systems*, Interoperable Communication Networks, Vol. 2, No. 2-4, Baltzer Science Publishers, Bussum, NL, January 2000, ISSN 1385 9501, pp. 287–295.

[YM99] Yamazaki, T.; Matsuda, J.: *Adaptive QoS management for multimedia applications in heterogeneous environments: a case study with video QoS mediation*, IEICE Transactions on Communications, Vol. 82-B, No. 11, J, November 1999, pp. 1801–1806.

[Zha95] Zhang, H.: *Service disciplines for guaranteed performance service in packet-switching networks*, Proceedings of the IEEE, Vol. 83, No. 10, USA, October 1995.

[ZL77] Ziv, J.; Lempel, A.: *A universal algorithm for sequential data compression*, IEEE Transactions on Information Theory, Vol. 23, No. 3, USA, 1977, ISSN 0018-9448, pp. 337–343.

[Zoj98] Zojer, S.: *Entwurf und Implementierung einer modularen Experimentierplattform für Audio und Video über ATM*, Uni Stuttgart, E-Technik, IND, September 1998.