[Zitt93] Martina Zitterbart:
Hochleistungskommunikation
Skriptum zur gleichnamigen Vorlesung im SS 1993; Institut für Telematik, Universität Karlsruhe; 1993.

[Schu93] Henning Schulzrinne, IETF Arbeitsgruppe AVT:
Issues in Designing a Transport Protocol for Audio and Video Conferences and other Multiparticipant Real-Time Applications
Internet Draft; <draft-ietf-avt-issues-01.txt>; 1993.

[Schu95a] Henning Schulzrinne:
Dynamic Configuration of Conferencing Applications using Pattern-Matching Multicast
5th International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV); Durham, New Hampshire; April 1995.

[Schu95b] Henning Schulzrinne:
Guide to NeVoT 3.30
Internes Dokument; GMD Fokus, Hardenbergplatz 2, 10623 Berlin; 1995.

[Schu95c] Henning Schulzrinne, IETF Arbeitsgruppe AVT:
RTP Profile for Audio and Video Conferences with Minimal Control
Internet Draft; <draft-ietf-avt-profile-04.ps>; 1995.

[Stei93] Ralf Steinmetz:
Multimedia-Technologie
Springer Verlag; Berlin u.a.; 1993.

[Stev92] W. Richard Stevens:
Advanced Programming in the UNIX Environment
Addison-Wesley; Englewood Cliffs, NJ; 1992.

[Tane90] Andrew S. Tanenbaum:
Computer-Netzwerke
Wolfram's Fachverlag; 1990.

[Topo90] C. Topolcic:
Experimental Internet Stream Protocol, Version 2 (ST-II)
RFC 1190; 1990.

[XTP92] Protocol Engines Inc.:
XTP Revision 3.6
Protocol Engines Inc.; Santa Barbara; 1992.

[ZBE&94] L. Zhang, R. Braden, D. Estrin, S. Herzog, S. Jamin:
Resource ReSerVation Protocol (RSVP) - Version 1 Functional Specification
Internet Draft; <draft-ietf-rsvp-spec-05.ps>; 1994.

[MPEG93] ISO IEC JTC 1:
Information Technology - Coding of Moving Pictures and Associated Audio for Digital Storage Media up to about 1.5 Mbit/s
International Standard ISO/IEC IS 11172; 1993.

[Mühl92] Max Mühlhäuser:
Telematik I
Skriptum zur gleichnamigen Vorlesung im WS 1992/93; Institut für Telematik, Universität Karlsruhe; 1992.

[Mühl93] Max Mühlhäuser:
Verteilte multimediale Systeme
Skriptum zur gleichnamigen Vorlesung im SS 1993; Institut für Telematik, Universität Karlsruhe; 1993.

[OSF91] Open Software Foundation:
Guide to OSF/1: a technical synopsis
O'Reilly & Associates; Sebastopol; 1991.

[Oust94] John K. Ousterhout:
Tcl and Tk toolkit
Addison-Wesley; Reading, Massachusetts; 1994.

[OVer94a] Ohne Verfasser:
Multimedia Sound & Motion J300
Drucknummer des Originals: EC-Z2459-19; Digital Equipment GmbH, Hauptverwaltung, Freischützstr. 91, 81927 München; 1991.

[Over94b] Ohne Verfasser:
Multimedia Services for DEC OSF/1 Programming Guide
Order Number: AA-Q0APD-TE; Digital Equipment Corporation, Maynard, Massachusetts, USA; 1994.

[Russ91] Michael Russo:
The new user's guide to the sun workstation
Springer Verlag; New York u.a.; 1991.

[SCFJ94] Henning Schulzrinne, Stephen Casner, Ron Frederick, Van Jacobson:
RTP: A Transport Protocol for Real-Time Applications
Internet draft; <draft-ietf-avt-rtp-06.ps>; 1994.

[JSTS92] K. Jeffay, D.L. Stone, T.Talley, F.D. Smith:
Adaptive, Best-Effort Delivery of Digital Audio and Video Across Packet-Switched Networks
Third International Workshop in Network and Operating System Support for Digital Audio and Video (NOSSDAV); La Jolla, USA; November 1992.

[Kast95] Frank J. Kastenholz:
A Profile for the Transmission of Video Data over RTP
Internet Draft; <draft-kastenholz-loki-00.txt>; 1995.

[KEL90] R. Keller, W. Effelsberg, B. Lamparter:
XMovie: Architecture and Implementation of a Distributed Movie System
Reihe Informatik 12/94; Universität Mannheim, Seminargebäude A5, 68131 Mannheim; 1994.

[Klau94] Peter Klau:
Das Internet: weltweit vernetzt
IWT-Verlag; Vaterstetten bei München; 1994.

[Krüg93] Gerhard Krüger:
Telematik II
Skriptum zur gleichnamigen Vorlesung im SS 1993; Institut für Telematik, Universität Karlsruhe; 1993.

[LiCo83] S. Lin, J. Costello:
Error Control Coding
Prentice Hall; Englewood Cliffs, NJ; 1983.

[LMKQ90] Samuel J. Leffler, Marshall Kirk McKusick, Michael J. Karels, John S. Quarterman:
The Design and Implementation of the 4.3 BSD UNIX Operating System
Addison-Wesley; Reading, Massachusetts; 1990.

[LPG&93] Thomas M. Levergood, Andrew C. Payne, James Gettys, G. Winfield Treese, Lawrence C. Stewart:
AudioFile: A Network-Transparent System for Distributed Audio Applications
Proceedings of the Summer Usenix Conference; 1993.

[Mill92] David L. Mills:
Network Time Protocol (Version 3) Specification, Implementation and Analysis
RFC 1305; 1992.

[Davi88] John Davidson:
An Introduction to TCP/IP
Springer-Verlag; New York u.a.; 1988.

[Dege94] Jutta Degener:
Digital Speech Compression
Dr. Dobb's Journal; December 1994.

[Erik94] H. Eriksson:
Mbone: The Multicast Backbone
Communications of the ACM; Vol. 37, Issue 8; pp. 54-60; 1994.

[Fell60] William Feller:
An Introduction to Probability Theory and Its Applications
John Wiley & Sons, Inc.; New York u.a.; 1960.

[Flan92] James L. Flanagan:
Speech Technology and Computing: A Unique Partnership
IEEE Communications Magazine; Volume 30; pp. 84-89; May 1992.

[HäHu91] Rainer Händel, Manfred N. Huber:
An Introduction to ATM-Based Networks
Addison-Wesley; Wokingham, England u.a.; 1991.

[Hals92] Fred Halsall:
Data Communications, Computer Networks and Open Systems
Addison-Wesley; Wokingham, England u.a.; 1992.

[HeSh92] John Henshall, Sandy Shaw:
OSI praxisnah erklärt: Der Standard für die Computer-Kommunikation
Hanser Verlag; München u.a.; 1992.

[HFG94] Don Hoffman, Gerard Fernando, Vivek Goyal:
RTP Encapsulation of MPEG1/MPEG2
Internet Draft; <draft-hoffman-rtp-mpeg-encap-01.txt>; 1994.

[JPEG92] ISO IEC JTC 1:
Information Technology - Digital Compression and Coding of Continuous-Tone still Images
International Standard ISO/IEC IS 10918; 1992.

C Quellenverzeichnis

Bei ausschließlich elektronisch verfügbaren Publikationen ist zu deren Lokation der Dateiname angegeben.

[ADH&93] Michael Altenhofen, Jürgen Dittrich, Rainer Hammerschmidt, Thomas Käppner, Carsten Kruschel, Ansgar Kückes, Thomas Steinig:
The BERKOM Multimedia Collaboration Service
Proceedings of the ACM Multimedia Conference, Anaheim, California, 1993; Association for Computer Machinery; New York; pp. 457-463, 1993.

[AST94] Michael Altenhofen, Joachim Schaper, Susan Thomas:
The BERKOM Multimedia Teleservices
Second International Workshop IWACA 1994; Heidelberg, Germany; pp. 237-250; Springer-Verlag; Berlin; 1994.

[Bale93] D. Balenson:
Privacy Enhancement for Internet Electronic Mail: Part III: Algorithms, Modes and Identifiers
RFC 1423; 1993.

[Bour82] S. R. Bourne:
The UNIX system
Addison-Wesley; London u.a.; 1982.

[CCIR82] CCIR:
Recommendation 601: Encoding Parameters of Digital Television for Studios
CCIR Genf; vol. 11, Part 1; 1982.

[CCIR91] CCIR:
Recommendation 601-2: Encoding Parameters of Digital Television for Studios
CCIR Genf; Documents CCIR Study Groups Period 1990-1994; 1991.

[ChTo92] Stephen T.-C. Chou, Hideyuki Tokuda:
System Support for Dynamic QOS Control of Continuous Media Communication
Third International Workshop in Network and Operating System Support for Digital Audio and Video (NOSSDAV); La Jolla, USA; November 1992; pp. 94-105; 1992.

[Cust93] Helen Custer:
Inside Windows NT
Microsoft Press; Redmond; 1993.

PMM	Pattern Matching Multicast
PT	Payload Type
RC	Reception Report Count
RFC	Request for Comments
RSVP	Resource Reservation Protocol
RTCP	Real-Time Transport Control Protocol
RTP	Real-Time Transport Protocol
SDES	Source Description
SSCP	Source and Sink Control Protocol
SSRC	Synchronization Source
ST-II	Experimental Internet Stream Protocol Version II
T	Version Type
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
X	Extention Flag
XTP	Express Transport Protocol

B Abkürzungsverzeichnis

ADPCM	Adaptive Pulse Code Modulation
ASC	Application Sharing Component
ATM	Asynchroner Transfer-Modus
AVC	Audiovisual Component
AVCP	Audiovisual Control Protocol
AVM	Audiovisual Manager
AVXP	Audio Video Exchange Protocol
BERKOM	Berliner Kommunikationssysteme
CC	Contributing Sources Count
CCIR	Comité Consultatif International des Radiocommunications
CCITT	Comité Consultatif International Telegraphique et Telephonique
CD	Conference Directory
CDAP	Conference Directory Access Protocol
CIA	Conference Interface Agent
CM	Conference Manager
CMAP	Conference Manager Access Protocol
CNAME	Canonical Name
CSMA/CD	Carrier Sense Multiple Access with Collision Detection
CSRC	Contributing Source
DPCM	Differenz Pulse Code Modulation
FEC	Forward Error Correction
FPN	Forward Error Correction Packet Number
FT	Forward Error Correction Type
IB	Invitation Broker
IBAP	Invitation Broker Access Protocol
IP	Internet Protocol
ICC	Interface Conference Controller
ICMP	Internet Control Message Protocol
IETF	Internet Engineering Task Force
ISO	International Standards Organization
JPEG	Joint Photographic Expert Group
M	Marker Flag
MMC	Multimedia Collaboration Service
MMM	Multimedia Mail Service
MMT	Multimedia Transport Service
MPEG	Motion Picture Expert Group
NeVoT	Network Voice Terminal
OSI-BRM	Open Systems Interconnection - Basisreferenzmodell
P	Padding Flag
PAL	Phase Alterning Line
PCM	Pulse Code Modulation
PDU	Protokolldateneinheit (Protocol Data Unit)

AFRecordSamples (ac, t0, 160, buffer, ANoBlock):

(10)	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**Mittelwert**
Geringe Last	1,46	1,56	1,56	1,85	1,56	1,46	1,85	1,56	1,85	1,56	1,627
Hohe Last	1,46	1,46	1,46	1,56	2,34	1,56	1,56	1,56	1,56	1,56	1,608

Tabellen A-1: Zeitbedarf der Audiofile-Funktionen (in ms)

A.2 Rückgabewerte

AFRecordSamples(ac, t2, 160, buffer, ABlock):

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**> 7**	**Mittelwert**
Geringe Last	0	2	2	0	164	632	187	10	3	5,04
Hohe Last	3	1	0	1	0	154	251	257	300	6,945

AFRecordSamples(ac, t2, 160, buffer, ANoBlock):

	0	**1**	**2**	**3**	**4**	**5**	**Mittelwert**
Geringe Last	15	521	453	11	0	0	1,46
Hohe Last	11	571	359	15	8	36	1,546

AFRecordSamples(ac, t0, 160, buffer, ABlock):

	5	**6**	**7**	**8**	**9**	**10**	**> 10**	**Mittelwert**
Geringe Last	5	240	707	48	0	0	0	6,798
Hohe Last	7	412	490	36	12	1	41	7,441

AFRecordSamples(ac, t0, 160, buffer, ANoBlock):

	0	**Mittelwert**
Geringe Last	1000	0
Hohe Last	1000	0

Tabellen A.2: Rückgabewerte von *AFRecordSamples* (in ms)

A Meßergebnisse Audiofile-Funktionen

A.1 Zeitbedarf

AFOpenAudioConn:

(1)	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**Mittelwert**
Geringe Last	37,1	17,6	15,6	15,6	18,5	17,6	18,5	14,6	15,6	42,9	21,36
Hohe Last	20,5	23,4	18,5	20,5	16,6	15,6	17,6	16,6	17,6	18,5	18,54

AFCloseAudioConn:

(1)	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**Mittelwert**
Geringe Last	2,93	2,93	2,64	2,64	2,73	2,73	2,64	2,73	2,93	2,93	2,783
Hohe Last	2,64	2,64	3,70	2,93	2,83	2,83	2,83	2,83	2,93	2,83	2,899

AFGetTime:

(10)	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**Mittelwert**
Geringe Last	1,27	1,27	1,46	1,37	1,56	1,37	1,37	1,56	1,27	1,27	1,377
Hohe Last	1,66	1,37	1,37	1,66	1,66	1,56	1,27	1,27	1,37	1,56	1,475

AFPlaySamples (ac, t6, 160, buffer):

(10)	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**Mittelwert**
Geringe Last	1,46	1,46	1,56	1,56	1,85	1,56	1,56	1,85	1,56	1,85	1,627
Hohe Last	2,15	1,56	1,56	1,56	1,46	1,56	1,56	1,56	1,56	1,46	1,599

AFRecordSamples (ac, t2, 160, buffer, ABlock):

(1)	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**Mittelwert**
Geringe Last	9,8	10,7	9,8	11,7	9,8	9,8	9,8	9,8	11,7	25,3	11,82
Hohe Last	17,6	13,7	12,7	9,8	9,8	11,7	11,7	11,7	11,7	9,8	12,02

AFRecordSamples (ac, t2, 160, buffer, ANoBlock):

(1)	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**Mittelwert**
Geringe Last	1,27	1,27	1,46	1,37	1,56	1,37	1,37	1,56	1,27	1,27	1,377
Hohe Last	1,66	1,37	1,37	1,66	1,66	1,56	1,27	1,27	1,37	1,56	1,475

AFRecordSamples (ac, t0, 160, buffer, ABlock):

(10)	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**Mittelwert**
Geringe Last	1,46	1,56	1,56	1,46	1,85	1,56	1,56	1,66	1,95	1,85	1,647
Hohe Last	1,95	1,66	1,56	1,56	1,85	1,46	1,56	1,56	1,56	1,46	1,618

jedoch einige Besonderheiten zu beachten sind. Insbesondere die Rückgabewerte einer Funktion zeigten sehr inkonsistentes Verhalten. Daraus resultierte ein erheblicher konzeptioneller Aufwand bei der Anpassung an DEC-Audiofile. Der im Rahmen der Arbeit gemessene Zeitbedarf einiger Funktionen deckte sich nicht mit den Angaben der Entwickler von DEC-Audiofile. Die Portierung hat ebenso gezeigt, daß insbesondere die unterschiedliche Byteanordnung der Systeme von Sun und Digital einen hohen Aufwand bei der Portierung eines Programms erfordern.

Dank

Während meiner Diplomarbeit wurde ich von Markus Hofmann und Claudia Schmidt vom Institut für Telematik und Rudolf Brandner und Markus Jork vom CEC Karlsruhe der Firma Digital betreut. Markus Hofmann und Claudia Schmidt gaben mir bei regelmäßigen Treffen konstruktive Anregungen und Hinweise und betreuten die Arbeit optimal. Alle Betreuer waren jederzeit ansprechbar. Markus Jork und Rudolf Brandner unterstützten mich insbesondere beim implementierungsspezifischen Teil der Arbeit, welcher am CEC Karlsruhe der Firma Digital durchgeführt wurde. Während eines Aufenthalts in Berlin und daran anschließend hatte ich jederzeit die volle Unterstützung von Dr. Henning Schulzrinne von der GMD Fokus in Berlin. Mit ihm konnte ich kreativ Probleme diskutieren, die im Rahmen seines Audio-Konferenzsystems NeVoT, des Realzeit-Transportprotokolls RTP und verwandten Themen auftraten. Prof. Dr. Dr. h. c. Gerhard Krüger vom Institut für Telematik und Joachim Schaper vom CEC Karlsruhe der Firma Digital ermöglichten die Durchführung der Diplomarbeit. Mrs. Spock war für die moralische Unterstützung während der Arbeit verantwortlich. Allen, auch den nicht explizit aufgeführten Personen, gebührt herzlicher Dank.

8 Zusammenfassung

Entsprechend der Aufgabenstellung liegen folgende Ergebnisse vor:

- Die Analyse und ein Vergleich der Protokolle RTP und AVXP, welche für den Transport multimedialer Daten (Audio/Video) geeignet sind.
- Die konzeptionelle Erweiterung des Konferenzsystems MMC um das Protokoll RTP.
- Die Portierung und Integration des Audio-Konferenzsystems NeVoT und des Protokolls RTP von Sun-Arbeitsplatzrechnern in die Systemumgebung von Alpha-Arbeitsplatzrechnern der Firma Digital.

Die Analyse und der Vergleich der Protokolle AVXP und RTP haben ergeben, daß es sich bei RTP gegenüber AVXP um ein wesentlich flexibleres Protokoll handelt. RTP unterstützt Protokollmechanismen, welche mittels AVXP nicht realisierbar sind. Dies ist insbesondere durch die Möglichkeit der Festlegung von anwendungsspezifischen Erweiterungen des Protokolls und das zu RTP gehörende Kontrollprotokoll RTCP begründet.

Die konzeptionelle Erweiterung von MMC um RTP hat gezeigt, daß durch die Verwendung von RTP statt AVXP Mechanismen eingesetzt werden können, die durch AVXP nicht unterstützt werden. Als Beispiele wurden die Interoperabilität von MMC und weiteren Konferenzsystemen, eine adaptive Vorwärtsfehlerbehebung und eine empfängergesteuerte Senderegulierung vorgestellt. Eine Reduzierung der Netzlast ist durch die Verwendung von unterbrochener Übertragung, bei der irrelevante Informationen (Sprechpausen) nicht übertragen werden, möglich. Dies gilt sowohl für RTP als auch für AVXP. Beide Protokolle unterstützen die für eine unterbrochene Übertragung erforderliche Synchronisation. In beiden Fällen ist zusätzlich zur Protokollunterstützung eine Abstimmung in den Endsystemen sowie eine Ruhe-Erkennung (silence-detection) erforderlich. Eine Verringerung von Verzögerungszeiten und -schwankungen ist durch die Verwendung von RTP nicht möglich, da diese Dienstgüteparameter nicht von RTP und AVXP, sondern von den darunterliegenden Protokollen abhängen.

Als weiterführende Arbeiten bieten sich folgende Ansätze an:

- Die Umsetzung der vorgestellten Erweiterung von MMC um RTP.
- Eine Ersetzung des unterhalb von RTP verwendeten Kommunikationssystems. Zum Zeitpunkt der Arbeit wird UDP/IP und CSMA/CD verwendet. Hierfür bietet sich beispielsweise ATM an, bei dem Verzögerungszeiten und -schwankungen erheblich reduziert bzw. ausgeschlossen werden können. Die Kombination von RTP und ATM ergäbe ein gut geeignetes System für die Übertragung von Audio und Video.

Durch die Portierung des Audio-Konferenzsystems NeVoT und des Protokolls RTP sind diese neben den bisher unterstützten Systemen von Sun, Silicon Graphics und Hewlett Packard auch auf Alpha-Arbeitsplatzrechnern der Firma Digital verwendbar. Im Rahmen der Portierung erfolgte eine Anpassung des Audiozugriffs mittels DEC-Audiofile. Dabei hat die Analyse von Audiofile ergeben, daß es sich um eine einfache Programmierschnittstelle handelt, bei der

- Messung der Zeitdauer zwischen *t2* und *t1*. Anhand der Messungen kann ein fester Wert für *ta* ermittelt werden, der beispielsweise dem größten Meßwert von *t2* entspricht. Dieser Wert gilt für alle Programmabläufe.
- Adaptive Bestimmung der Zeitdauer zwischen *t2* und *t1* während des Programmablaufs. Mit dieser Methode kann ständig eine Anpassung von *ta* an *t2* erfolgen. Hierbei ist zu beachten, daß eine Anpassung von *ta* während des Programmablaufs sinnvoll ist, jedoch nur in größeren Abständen erfolgen sollte. Erfolgt hingegen beispielsweise bei jedem Schreibzugriff eine Neuberechnung von *ta*, so entstehen wiederum Lücken oder Überschneidungen bei der Ausgabe. Durch die ständigen Messungen der Zeitdauer zwischen *t1* und *t2* wird mit diesem Verfahren eine Unabhängigkeit vom verwendeten System erreicht.

Aufgrund der höheren Variabilität wurde die zweite Variante gewählt, da sie den entscheidenden Vorteil der Unabhängigkeit vom verwendeten System bietet. Die Messungen der Zeiten zwischen *t1* und *t2* können unterschiedlich ausgewertet werden. Eine Möglichkeit besteht darin, für *ta* den höchsten Meßwert zu verwenden. Dadurch könnten jedoch selten auftretende, extreme Verzögerungen zu einer insgesamt sehr hohen Verzögerung führen. Sinnvoller ist es, nicht alle Schwankungen, sondern nur einen hohen Prozentsatz durch eine Verzögerung zu kompensieren, um mit akzeptabler Verzögerung eine gute Audioqualität zu erreichen.

Der Algorithmus wurde so ausgelegt, daß *ta* nach jedem hundertsten Schreibaufruf neu berechnet wird. Bei einer Paketgröße von 20 Millisekunden erfolgt dies demnach alle zwei Sekunden. Dabei wird *ta* so gewählt, daß mindestens 95 % aller gemessenen Zeitpunkte *t2* zeitlich vor *ta* liegen. Durch dieses Verfahren werden seltene, extrem hohe Verzögerungen von *t2* gegenüber *t1* ignoriert. Da es sich bei diesen Parametern um eine subjektive Beurteilung handelt (Qualität von Audio) erfolgt keine Darstellung der Meßwerte.

Setzen von Parametern

Das Setzen der Parameter für den Audiozugriff erfolgt über die Funktion *AFChangeACAttributes*. Zu den Parametern gehören u.a. Lautstärke, Wahl der Kodierung und Anzahl der Kanäle. Eine Besonderheit besteht bei der Auswahl der Samplefrequenz. Diese muß bereits beim Starten des Servers angegeben werden. Demzufolge kann bei einer Veränderung von Parametern nur eine Überprüfung der Gültigkeit der Samplefrequenz durchgeführt werden, nicht jedoch eine Veränderung der Frequenz. Während des Programmablaufs ist demnach eine Veränderung der Audiokodierung nur möglich, wenn die gleiche Abtastfrequenz verwendet wird.

Schreibzugriff

Unmittelbar nach dem Lesen von Daten wird die Funktion *ahw_write* aufgerufen, welche Daten an das Audiosystem zur Ausgabe übergibt. Für die Umsetzung des schreibenden Zugriffs existieren wiederum mehrere Möglichkeiten. Für die folgenden Ausführungen muß beachtet werden, daß der Funktionsaufruf von *ahw_write* abhängig von *ahw_read* ist und demnach bezüglich der Aufrufzeitpunkte ebenfalls Schwankungen unterworfen ist.

Eine Möglichkeit besteht in der Ausgabe der Daten zum Zeitpunkt des Aufrufs von *ahw_write*. Dazu ist jeweils eine Abfrage der Audiouhr erforderlich, anhand derer die Ausgabe festgelegt werden kann. Aufgrund der Schwankungen der Aufrufzeitpunkte von *ahw_write* ergäben sich durch diese Realisierung Einbußen der Qualität, da sich Pakete überlagern oder Lücken zwischen Paketen entstehen könnten.

Sinnvoller ist eine Orientierung an den für das Lesen von Daten verwendeten Synchronisationspunkten. Werden die empfangenen Audiodaten immer einen festen Zeitpunkt nach den Synchronisationspunkten ausgegeben, so kann eine lückenlose Ausgabe der Audiodaten gewährleistet werden. Für dieses Verfahren ist die Ermittlung des Ausgabezeitpunkts erforderlich. Der Schreibzugriff ist in Abbildung 7-6 dargestellt. In der Abbildung entspricht der Zeitpunkt *t1* dem Synchronisationspunkt, der Zeitpunkt *t2* dem Aufruf der Funktion *ahw_write*. Es ist ein zu lesendes und ein zu schreibendes Paket dargestellt. In *ahw_write* soll die Ausgabe der Daten so koordiniert werden, daß sie zum Zeitpunkt *ta* ausgegeben werden. An den Zeitpunkt *ta* gibt es folgende Anforderungen:

- Der Zeitpunkt *ta* muß nach dem Zeitpunkt *t2* liegen, ansonsten gehen Daten verloren.
- Der Zeitpunkt *ta* muß möglichst nah an *t1* (dem Synchronisationspunkt) liegen, um die Verzögerung so gering wie möglich zu gestalten.

Ziel ist es demnach, den Zeitpunkt *ta* optimal zu bestimmen.

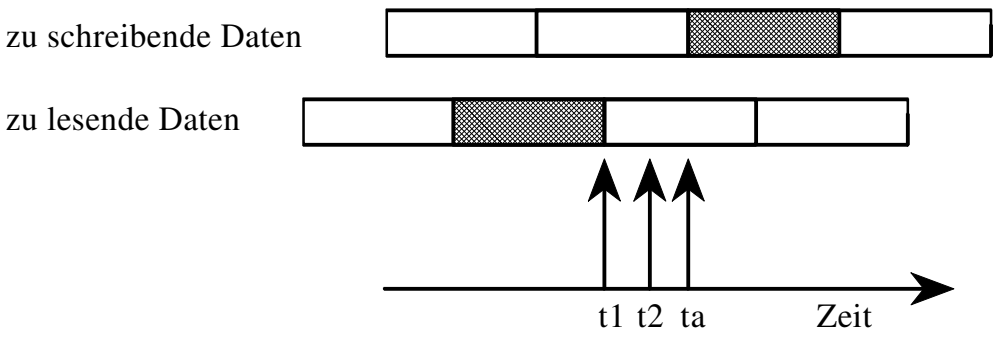


Abbildung 7-6: Schreibender Zugriff

Je weiter der Zeitpunkt *ta* nach hinten verzögert wird, desto unbedeutender werden die Schwankungen von *t2*, da immer mehr Werte von *t2* vor *ta* liegen. Die Ermittlung von *ta* sollte in Abhängigkeit von *t2* erfolgen, um *ta* so nah wie möglich an *t1*, nicht jedoch vor *t2* zu legen. Für die Bestimmung von *ta* gibt es mehrere Möglichkeiten:

Millisekunden hinzuaddiert, kann der errechnete Wert als Anhaltspunkt für den nächsten Zugriff verwendet werden. Dadurch entstehen keine Lücken oder Überschneidungen beim Lesen von Daten. Diese Vorgehensweise wird durch den Pufferspeicher bei Audiofile ermöglicht.

Nach den Ausführungen zum Einsatz der Funktion AFRecordSamples und zu ihren Rückgabewerten ist die Bestimmung für den Zeitpunkt des Aufrufes der Funktion noch offen. Theoretisch ist ein Aufruf der Funktion in Abständen entsprechend dem Paketintervall erforderlich. Bei diesen Intervallen kann es jedoch gelegentlich zu Verzögerungen kommen, da auf den Rechnern i.d.R. mehrere Prozesse den Prozessor beanspruchen. Dabei kann ein Prozeß mit hoher Priorität vorübergehend viel Rechenzeit beanspruchen, was zur Verzögerung von Zeitmessern (und somit dem Aufruf von ahw_read) führen kann. Das Betriebssystem OSF/1 bietet zwei Möglichkeiten von Zeitmessern:

- Ein Timer kann auf einen Zeitwert gesetzt und gestartet werden. Nach Ablauf des Timers erhält ein Prozeß ein Signal, welches den Ablauf des Timers anzeigt [LMKQ90; Kapitel 4.7].
- Ein explizites Warten auf einen Zeitpunkt wird von einem Prozeß mittels des Befehls *select* realisiert [Stev92].

Bei den im ersten Punkt vorgestellten Signalen kann es sehr lange dauern, bis ein Signal bei einer Funktion ankommt. Weiterhin kann ein Signal unter Umständen verloren gehen, was zu inakzeptablen Ergebnissen führen kann. Demzufolge wurde für die Implementierung ein mittels *select* realisierter Zeitmesser verwendet.

Abbildung 7-5 zeigt beispielhaft zwei aufeinander folgende Zugriffe auf Audiodaten. Aufgrund von Verzögerungen wurden die Daten nicht zum Synchronisationspunkt *t1* abgerufen, sondern zu einem Zeitpunkt *t1**, der zeitlich nach *t1* liegt. Der nächste Synchronisationspunkt liegt nun nicht mehr (*t2 - t1*) Millisekunden entfernt, sondern (*t2 - t1**). Der Zeitmesser muß dieser verkürzten Zeitdauer angepaßt werden. Für die Berechnung der Zeitdauer (*t2 - t1**) ist eine Kenntnis der beiden Werte erforderlich. Der Wert *t1** entspricht dem Rückgabewert der Funktion *AFRecordSamples*, der Wert *t2* kann wie folgt berechnet werden. Zu Beginn mehrerer aufeinanderfolgender Lesezugriffe (etwa nach jedem Öffnen des Audiosystems) wird ein Referenzzeitpunkt (der erste Synchronisationspunkt) der Audiouhr genommen. Aufgrund der Kenntnis des Paketintervalls können alle folgenden Synchronisationspunkte berechnet werden, so auch *t2*.

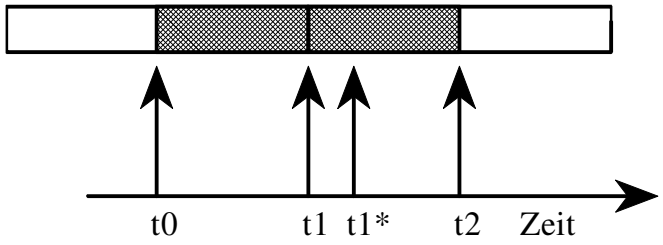


Abbildung 7-5: Synchronisationspunkte aufeinanderfolgender Lesezugriffe

Zusammengefaßt ergibt sich die Problematik, daß zum richtigen Zeitpunkt die Daten effizient gelesen werden müssen. Im folgenden wird zunächst auf den Einsatz der Funktion AFRecordSamples eingegangen, anschließend erfolgen Überlegungen zu den Aufrufzeitpunkten der Funktion. In Kapitel 7.3.2 wurde die Funktion *AFRecordSamples* bereits vorgestellt. Sie kann blockierend (warten auf noch nicht vorliegende Daten) oder nicht-blockierend eingesetzt werden.

Ein erster Ansatz führt zu der Überlegung, AFRecordSamples blockierend einzusetzen und damit einen Abgleich zwischen Audiouhr und Systemuhr durchzuführen. Dabei könnte mit einem Zeitmesser **(Timer)** ein Zeitpunkt ausgewählt werden, der kurz unterhalb des Synchronisationspunktes liegt. Durch ein blockierendes Lesen sollte die Funktion nach Erhalt aller Daten zurückkehren und damit den Synchronisationspunkt genau treffen. In Abbildung 7-4 ist dieser Sachverhalt graphisch dargestellt.

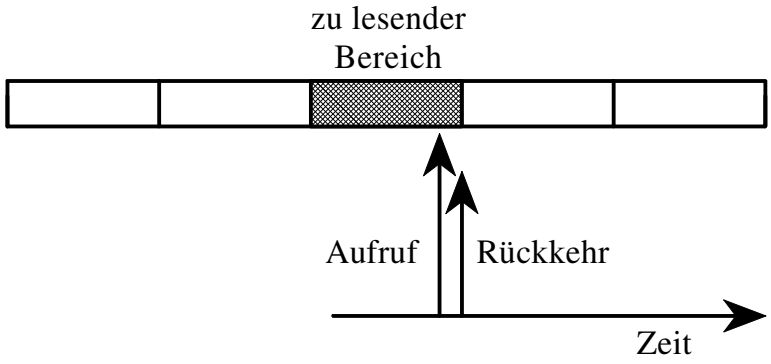


Abbildung 7-4: Blockierendes Lesen mit *AFRecordSamples*

Mehrere Gründe sprechen jedoch gegen die Verwendung der Funktion in dieser Art und Weise. Zunächst entsteht durch den blockierenden Einsatz eine Totzeit, in der das Programm nur auf die Rückkehr der Funktion wartet. Viel gravierender ist jedoch, daß ein Funktionsaufruf für ein blockierendes Lesen nicht unmittelbar nach Vorlage aller Daten zurückkehrt, sondern erst ca. sieben Millisekunden danach. Diese Werte gehen aus Kapitel 7.3.3 hervor. Ein blockierendes Lesen von Audiodaten ist demnach nicht möglich. Diese Überlegungen führen zu dem Schluß, daß nur komplett vorliegende Daten abgerufen werden sollten. Die Zeitpunkte für den Datenabruf sind demnach so zu wählen, daß alle angeforderten Daten vorliegen.

Die in Tabelle 7-3 vorgestellten Meßergebnisse zeigen, daß die Funktion *AFRecordSamples* teilweise keine konsistenten Rückgabewerte (Rückgabewerte der Audiouhr) liefert. Daraus ergibt sich die Notwendigkeit, für die Angabe des Zeitpunkts von zu lesenden Daten berechnete Zeitpunkte anzugeben. Ein Beispiel: zu einem Zeitpunkt *t* werden Daten gelesen, die einer Dauer von 20 Millisekunden entsprechen. Die Meßergebnisse zeigen, daß der Rückgabewert der Funktion nicht immer *t* + 20 Millisekunden entspricht, sondern oftmals darüber liegt. Bei einem nächsten Lesezugriff kann demnach durch das Verwenden des Rückgabewertes eine Lücke entstehen. Werden jedoch zu dem bekannten Zeitpunkt *t* 20

Öffnen und Schließen des Audiozugriffs

Das Öffnen des Audiozugriffs bildet keine zeitkritische Komponente bei der Implementierung, da diese Vorgänge lediglich zu Beginn der Übertragung und am Ende erfolgen. Der Zeitbedarf für das Erstellen und Freigeben von Kontexten kann aufgrund des Zeitbedarfs vernachlässigt werden. Das Schließen des Audiozugriffs am Ende einer Übertragung mittels der Funktion *AFCloseAudioConn* mit einer Dauer von ungefähr 3 Millisekunden ist aufgrund des seltenen Einsatzes ebenfalls unkritisch.

Lesezugriff

Weit aufwendiger als das Öffnen und Schließen des Audiozugriffs ist die konzeptionelle Neugestaltung des Lesens und Schreibens von Daten. Im folgenden wird zunächst auf den lesenden Zugriff eingegangen, daran anschließend erfolgt die Betrachtung des schreibenden Zugriffs. Anhand Abbildung 7-3 wird die Problematik des Lesens von Daten verdeutlicht.

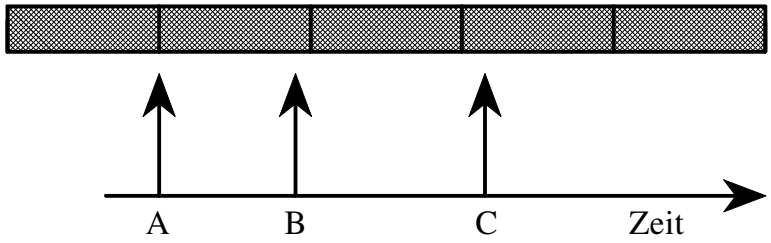


Abbildung 7-3: Synchronisation des Lesezugriffs bei NeVoT

Zunächst sei nochmals erwähnt, daß die Paketintervalle durch die Audiouhr und nicht durch die Systemuhr bestimmt sind. Daraus ergibt sich bereits die erste Konsequenz für die Implementierung. Da bei Audiofile ein explizites Abrufen der Daten erforderlich ist, muß dieser Abruf mittels eines Zeitmechanismus gesteuert werden. Der Zeitpunkt der Audiouhr kann bei Audiofile nur abgerufen werden, zur Bestimmung eines Zeitintervalls für den Abruf muß die Systemuhr verwendet werden. Da für die Ein-/Ausgabe von Audiodaten die Audiouhr maßgeblich ist, muß ein ständiger Abgleich zwischen Audiouhr und Systemuhr erfolgen.

In Abbildung 7-3 sind Paketintervalle und Zeitpunkte, zu denen auf Audiodaten zugegriffen wird, eingezeichnet. Im Optimalfall wird wie zum Zeitpunkt A jeweils genau bei Vorlage aller Daten für ein Paket auf diese zugegriffen. Im folgenden wird hierfür die Bezeichnung **Synchronisationspunkt** verwendet. Wird dieser Zeitpunkt verfehlt, so entstehen Schwankungen zwischen den Versendezeitpunkten der Pakete. Diese Schwankungen müssen entweder beim Sender oder beim Empfänger durch das Einfügen von künstlichen Verzögerungszeiten kompensiert werden. Im Fall B liegen die Daten noch nicht vor, hier bietet sich die Möglichkeit des Wartens oder der Rückkehr zum Leseaufruf zu einem späteren Zeitpunkt an. Fall C zeigt eine Verspätung gegenüber dem optimalen Zeitpunkt des Datenaufrufs. Hierdurch entsteht die erwähnte Schwankung zwischen den Paketankunftszeiten.

*ahw_set(int pr, audio_set_t p, int value)/ahw_get(audio_state_t *as)*
Mit *ahw_set* und *ahw_get* werden Parameter für den Audiozugriff gesetzt und der Status des Systems (Fehler/kein Fehler) abgerufen. Bei *ahw_set* wird angegeben, ob es sich um lesenden oder schreibenden Zugriff handelt (*pr*), der zu setzende Parameter (*p*) und der Wert auf den der Parameter gesetzt werden soll. Zu den Parametern gehören u.a. die Lautstärke und der Ein-/Ausgabekanal (Mikrophon, „line in“ etc.).

*ahw_read(char *buffer, int bytes)/ahw_write(char *buffer, int bytes)*
Die Funktionen *ahw_read* und *ahw_write* führen den konkreten Lese- oder Schreibzugriff durch. In beiden Fällen wird der Speicherbereich angegeben, in den die Daten geschrieben werden bzw. aus dem die Daten gelesen werden.

*ahw_fmt(audio_descr_t *record, audio_descr_t *play)*
Die Funktion *ahw_fmt* bestimmt die Qualität des angeschlossenen Audiosystems. Die Ermittlung der Qualität des Audiosystems (mögliche Kodierungen, Samplefrequenz) muß für jedes System durch individuelle Abfragen erfolgen. Die ermittelte Qualität bestimmt die Parameter für die Ein- und Ausgabe der Audiodaten.

7.3.5 Implementierung

In den letzten Kapitel wurde der Aufbau des Audiozugriffs bei NeVoT erläutert. Im folgenden wird die konkrete Umsetzung des Audiozugriffs erläutert. Dabei steht das Lesen und Schreiben von Daten im Vordergrund.

Die bereits vorgestellte Implementierung der Audiosteuerung bei NeVoT erfolgt nach dem Prinzip des Erhalts einer Meldung, wenn am Audiosystem die erforderliche Datenmenge vorliegt. Die ebenfalls vorgestellten Schnittstellenfunktionen bei DEC-Audiofile ermöglichen diese Art von Lesezugriffen nicht. Vielmehr ist ein explizites Abrufen der Daten erforderlich. Dieser Unterschied erfordert eine komplette Neugestaltung des Audiozugriffs bei der Verwendung von DEC-Audiofile.

Die folgenden Betrachtungen orientieren sich an einer Paketgröße von 20 Millisekunden. Größere Werte erhöhen die Ende-zu-Ende Verzögerung, liefern jedoch bezüglich der zeitlichen Anforderungen des Programms keine Probleme. Wesentlich geringere Werte sind nicht erforderlich, da mit einer Paketgröße von 20 Millisekunden ausreichende Verzögerungszeiten für interaktive Anwendungen erzielt werden können. Bei über Satelliten geschalteten Telefonverbindungen entstehen aufgrund der Ausbreitungsverzögerung von elektromagnetischen Wellen eine Verzögerung von ca. 250 Millisekunden, so daß die hier gewählten 20 Millisekunden zusammen mit der erforderlichen Verarbeitung der Daten eine wesentlich geringere Verzögerung ermöglichen. In anderen Systemen werden beispielsweise Paketgrößen von 33 Millisekunden [JSTS92] verwendet.

Millisekunden über dem theoretischen Wert liegt. Hier liegt eine Parallele zur Zeitmessung der Funktion vor, wo ein Zeitaufwand festgestellt wurde, der ebenfalls etwa um 5 Millisekunden höher liegt als erwartet. Da bei diesem Funktionsaufruf zwischen Aufruf und Rückkehr über 10 Millisekunden liegen, ist diese Art von Einsatz indiskutabel. Es muß beim Lesen von Daten demnach gewährleistet werden, daß die zu lesenden Daten vollständig vorliegen, um ohne blockierendes Lesen auszukommen.

7.3.4 Der Audiozugriff bei NeVoT

Nachdem in den letzten Kapiteln die Möglichkeiten für den Zugriff auf das Audiosystem bei Alpha-Arbeitsplatzrechnern vorgestellt wurden, erfolgt hier die Erläuterung der Programmstruktur bei NeVoT für den Audiozugriff. Die Ausführungen beziehen sich auf die Version 3.30. Dieses Kapitel ergänzt die allgemeinen Erläuterungen zum Programm NeVoT, welche bereits in Kapitel 3 vorgestellt wurden. In Kapitel 7.3.5 wird konkret auf die Implementierung eingegangen.

Wie bereits in den allgemeinen Ausführungen über NeVoT erläutert, ist die Ein- und Ausgabe von Audio gekoppelt, d.h. nach jedem lesenden Zugriff wird auch ein schreibender Zugriff durchgeführt, sofern Daten zum Schreiben vorliegen. Die Intervalle zwischen den Zugriffen entsprechen der in Millisekunden angegebenen Paketgröße bei einem Teilnehmer. Ebenso wie die Paketgröße muß beim Initiieren einer Konferenz auch eine Auswahl der Kodierung erfolgen. Aus der Kodierung und dem Zeitintervall ergibt sich dann die Datenmenge, die bei jedem Zugriff zwischen Audiosystem und Programm transferiert wird. Bei der Entwicklung von NeVoT auf Sun-Arbeitsplatzrechnern wurde ein Mechanismus des Systems verwendet, der das Vorliegen von Audiodaten an ein Programm meldet. Die Struktur des Programms hat folgende Gestalt: Eine Funktion *tx* (= transmitt) wird immer dann aufgerufen wird, wenn die erforderliche Menge an Daten vorliegt. Durch den Aufruf von *tx* erfolgt implizit ein Lesezugriff, eine Übertragung der Daten (falls erforderlich) und ein Schreibzugriff. Empfangene Daten werden davon unabhängig in einem Ringpuffer abgelegt, woraus sie bei einem Schreibzugriff abgerufen werden.

Bei NeVoT existieren für den hardwareabhängigen Teil des Audiozugriffs mehrere Funktionen, die für jedes System neu erstellt bzw. an jedes System angepaßt werden müssen. Die hierfür wesentlichen Funktionen werden im folgenden vorgestellt, sie bilden die Grundlage für die im nächsten Kapitel vorgestellte Implementierung des Audiozugriffs bei Alpha-Rechnern.

ahw_open(int ndelay)/ahw_close(void)
Die Funktionen *ahw_open* und *ahw_close* öffnen und schließen den Zugriff auf das Audiosystem. Der Zugriff wird geöffnet, wenn Daten zur Ausgabe vorliegen oder Daten gelesen werden sollen. Falls keine Daten mehr für die Ein- oder Ausgabe vorliegen, wird der Zugriff geschlossen. Der Parameter *ndelay* (= no delay) gibt an, daß auf angeforderte Daten, sofern sie noch nicht am Audiosystem vorliegen, nicht gewartet werden soll.

mit den Mittelwerten 11,82 und 12,02 Millisekunden überraschen in diesem Zusammenhang. Daraus ergibt sich auch die Empfehlung, diese Funktion nicht wie hier beschrieben einzusetzen.

Im folgenden werden Meßwerte der Funktion *AFRecordSamples* vorgestellt, wie sie bereits im Zusammenhang mit Abbildung 7-2 angesprochen wurden. Die Messungen beziehen sich auf den Rückgabewert der Funktion. Wie bereits im letzten Kapitel erwähnt, sind die Rückgabewerte der Funktion (Zeitpunkte) nicht konsistent. Tabelle 7-3 beinhaltet durchschnittliche Meßergebnisse der Funktionsaufrufe, wobei zur Messung 1000 Iterationen durchgeführt wurden. Die vollständigen Meßergebnisse befinden sich in Anhang A. Jeder Funktionsaufruf *AFRecordSamples* aus Abbildung 7-2 hat einen theoretischen Rückgabewert, auf den sich die Meßergebnisse beziehen. Dieser Wert berechnet sich aus dem Startzeitpunkt des Schreib-/Lesezugriffs und der Menge der zu lesenden Daten. Im Fall 2 (blockierendes Lesen für noch nicht vollständig vorliegende Daten) ist es der Zeitpunkt t6, im Fall 3 (nicht blockierendes Lesen für nicht vollständig vorliegende Daten) ist es t5 und in den Fällen 4 und 5 ist es der Zeitpunkt t4. In den Tabelleneinträgen befinden sich die Differenzen zu den vorgestellten Zeitpunkten. Diese sind zur Übersichtlichkeit in Millisekunden umgerechnet, die Funktion gibt Samplewerte zurück.

Funktionsaufruf	**Last**	**Durchschnittlicher Rückgabewert**
AFRecordSamples (ac, t2, 160, buffer, Ablock)	gering	5,04
	hoch	6,94
AFRecordSamples (ac, t2, 160, buffer, ANoBlock)	gering	1,46
	hoch	1,54
AFRecordSamples (ac, t0, 160, buffer, Ablock)	gering	6,79
	hoch	7,44
AFRecordSamples (ac, t0, 160, buffer, ANoBlock)	gering	0
	hoch	0

Tabelle 7-3: Durchschnittliche Rückgabewerte von *AFRecordSamples* (in ms)

Bei der Analyse der Ergebnisse sind mehrere Aspekte auffällig. Ein Vergleich der Mittelwerte ergibt, daß bei hoher Last die Rückgabewerte um bis zu 37 % voneinander abweichen. Dies ist deshalb verwunderlich, da der Zeitaufwand für die Ausführung der Funktionen nur um ca. 15 Prozent abweicht. Eine plausible Erklärung für das Verhalten gibt es nicht, als Konsequenz muß jedoch bei der Verarbeitung der Rückgabewerte eine ständige Kontrolle stattfinden. Als Kontrolle kann ein Vergleich zwischen errechneten und tatsächlich erhaltenen Rückgabewerten eingesetzt werden.

Weiterhin ist verwunderlich, daß beim Lesen von komplett vorliegenden Daten ein Unterschied im Rückgabewert besteht, je nachdem, ob die Funktion blockierend eingesetzt wird oder nicht. Der Unterschied im Rückgabewert läßt sich auch nicht durch den Zeitbedarf der Funktionen erklären, da beide gemäß Tabelle 7-2 ungefähr 1,6 Millisekunden benötigen. Blockierendes Lesen von noch nicht vollständig vorliegenden Daten liefert einen Wert der Audiouhr, der ca. 5

davon mit zwei Stelle hinter dem Komma. Der Eintrag hinter dem Funktionsaufruf gibt die Anzahl der Wiederholung von Funktionsaufrufen an, die zur Messung verwendet wurden. Tabelle 7-2 zeigt die Mittelwerte der Messungen, die vollständigen Meßergebnisse befinden sich in Anhang A.

Funktionsaufruf / Wiederholungen	**Last**	**Mittelwert**
AFOpenAudioConn / (1)	gering	21,36
	hoch	18,54
AFCloseAudioConn / (1)	gering	2,783
	hoch	2,899
AFGetTime / (10)	gering	1,377
	hoch	1,475
AFPlaySamples (ac, t6, 160, buffer) / (10)	gering	1,627
	hoch	1,599
AFRecordSamples (ac, t2, 160, buffer, Ablock) / (1)	gering	11,82
	hoch	12,02
AFRecordSamples (ac, t2, 160, buffer, ANoBlock) / (1)	gering	1,377
	hoch	1,475
AFRecordSamples (ac, t0, 160, buffer, Ablock) / (10)	gering	1,647
	hoch	1,618
AFRecordSamples (ac, t0, 160, buffer, ANoBlock) / (10)	gering	1,627
	hoch	1,608

Tabelle 7-2: Durchschnittlicher Zeitbedarf der Audiofile-Funktionen (in ms)

Ein erster Vergleich der Meßergebnisse zeigt, daß beim regelmäßig erforderlichen Lesen und Schreiben von Daten keine wesentlichen Unterschiede zwischen geringer und hoher Prozessorlast vorliegen (maximal 7 %). Lediglich beim seltener benötigten Öffnen einer Verbindung existieren Abweichungen von 15 %. Für einen Programmablauf unter hoher Prozessorlast sind demnach keine Besonderen Berücksichtigungen erforderlich. Es ist ebenfalls ersichtlich, daß für das Lesen und Schreiben von Daten ein Mindestaufwand von ungefähr 3,2 Millisekunden kalkuliert werden muß. In [LPG&93] wird als Durchschnittswert für einen Zyklus aus einer Schreib- und einer Leseoperation 0,87 Millisekunden (DEC Alpha 3000/400s; Taktfrequenz 133 MHz) angegeben. Dieser Wert deckt sich nicht mit den vorliegenden Testergebnissen. Der Grund für die hohen Abweichungen zwischen angegebenen und gemessenen Werten ist nicht alleine durch unterschiedliche Systeme bei der Leistungsmessung begründbar und nicht nachvollziehbar. Der Zeitbedarf für die Funktionen *AFOpenAudioConn* und *AFCloseAudioConn* ist insofern unkritisch, da diese Funktionen nur jeweils ein Mal innerhalb eines größeren Zeitraums aufgerufen werden. Besonders auffällig ist das Ergebnis von Beispiel 2 (*AFRecordSamples*). Bei diesem Funktionsaufruf liegt aufgrund der anderen Meßergebnisse die Vermutung nahe, daß der Zeitaufwand ungefähr 6,6 Millisekunden (1,6 Millisekunden Lesezugriff plus 5 Millisekunden Blockierzeit) entspricht. Die Meßergebnisse

7.3.3 Zeitliche Nebenbedingungen

In diesem Kapitel werden zunächst die zeitlichen Nebenbedingungen der Ein-/Ausgabe von Audio dargestellt. Im Anschluß daran werden Messungen der für die Realisierung zur Verfügung stehenden Audiofile-Funktionen und die daraus resultierenden Konsequenzen erläutert.

Bei NeVoT kann die Größe von Audiopaketen vom Anwender gewählt werden. Als Standard sind Pakete mit einer Größe von 20 Millisekunden, unabhängig von der Kodierung, vorgegeben. Daraus resultiert eine obere Zeitschranke für die Verarbeitung von Paketen von ebenfalls 20 Millisekunden. In dieser Zeit muß die entsprechende Datenmenge vom Audiosystem abgerufen und der Inhalt eines ankommenden Pakets an das Audiosystem gegeben werden können. Im folgenden wird zunächst auf das Meßverfahren, dann auf die konkreten Meßwerte von Audiofile-Funktionen eingegangen. Im Anschluß an die Zeitmessung von Funktionsaufrufen erfolgt eine Darstellung gemessener Rückgabewerte der Funktion *AFRecordSamples*.

Als Testumgebung wurden ein Auftraggeber und ein Server gewählt, die beide auf demselben Rechner ausgeführt wurden. Als Testrechner fungierte ein Alpha-Arbeitsplatzrechner 3000/300X mit 96 MB Hauptspeicher, dessen Prozessor mit 175 MHz getaktet wird. Eine erste Analyse ergab - je nach untersuchter Funktion - einen Zeitbedarf von einer bis mehreren Millisekunden. Die vom Betriebssystem zur Verfügung gestellte und verwendete Funktion *gettimeofday* besitzt eine Auflösung von 976 Mikrosekunden. Durch die Messung eines Funktionsaufrufes kann demnach bei Zeiten im Bereich einer Millisekunde eine sehr hohe Meßungenauigkeit auftreten. Aus diesem Grund wurden bei manchen Funktionen eine Messung anhand einer mehrfachen Iteration durchgeführt. Ein entsprechender Vermerk in den Tabelleneinträgen gibt die Anzahl der verwendeten Iterationen an. Die Messungen wurden bei zwei unterschiedlichen Prozessorauslastungen jeweils zehnmal durchgeführt. Die Aufteilung der Prozessorzeit zwischen Anwendungs- und Systemprozessen und freier Zeit ist in Tabelle 7-1 dargestellt.

	Anwendungs-prozesse (in %)	**Systemprozesse (in %)**	**Prozessor frei (in %)**
Geringe Last	5-15	5-15	70-90
Hohe Last	80-95	5-20	0

Tabelle 7-1: Prozessorlast bei Audiofile-Zeitmessung

Die folgenden Tabelle beinhaltet die Meßergebnisse für die in Kapitel 7.3.2 vorgestellten Audiofile-Funktionen. Der Zeitaufwand für die Funktionen *AFCreateAC* und *AFFreeAC* beträgt weniger als 0,05 Millisekunden, aus diesem Grund sind die Ergebnisse nicht aufgeführt. Alle Beispiele aus Abbildung 7-2 sind aufgeführt. In den Feldern sind die Zeitaufwände in Millisekunden für *einen* Funktionsaufruf eingetragen. Werte unterhalb von 5 Millisekunden sind mit einer Genauigkeit von drei Stellen nach dem Komma angegeben, Werte oberhalb

zurückgegeben, im Fall 5 der Zeitpunkt t4. Ob es sich bei diesen Besonderheiten um einen Fehler des Programms oder eine absichtliche Eigenschaft handelt, ist nicht ersichtlich.

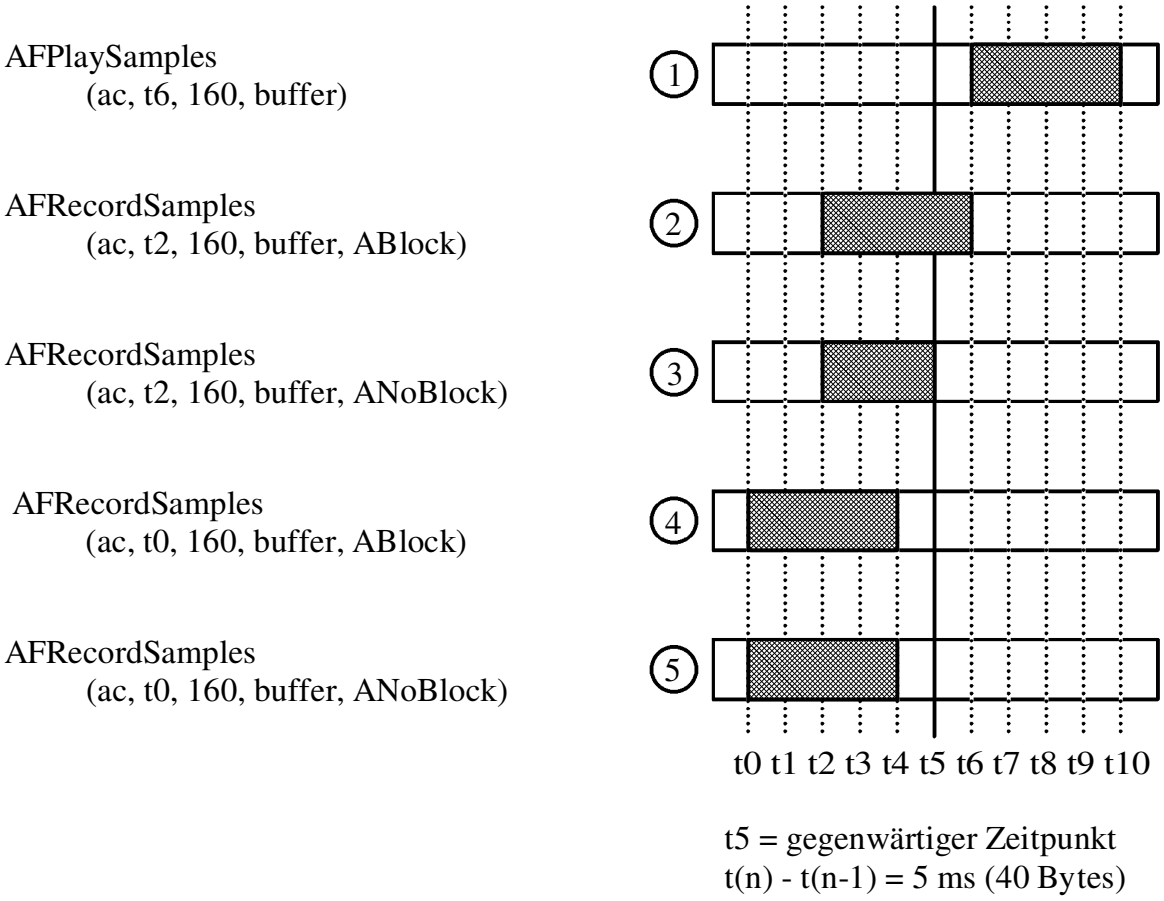


Abbildung 7-2: Beispiel für Schreib-/Leseoperationen

Die vorgestellten Funktionen zeigen, daß mit DEC-Audiofile eine einfache Schnittstelle zur Ein- und Ausgabe von Audio vorliegt. Einige Besonderheiten, insbesondere der Rückgabewert der Funktion *AFRecordSamples,* sind jedoch zu beachten. Des weiteren ist bei Audiofile leider kein Mechanismus vorhanden, welcher das Vorhandensein einer definierbaren Datenmenge am Audiosystem meldet (beispielsweise mittels Unterbrechungsfunktion). Bei der Ein- oder Ausgabe von Audio sind jeweils nur alle Anschlüsse ansprechbar. Die Anschlüsse „line in“ und Mikrophon können ebensowenig gesondert angesprochen werden wie „line out“ und die Lautsprecher. Für die Implementierung kommt erschwerend hinzu, daß Audiofile in der am Rechner elektronisch verfügbaren Dokumentation (man-pages) teilweise fehlerhaft dokumentiert ist (z.B. die Datenstruktur AF_sample_sizes).

(Aufnahme/Ausgabe), Anzahl der Kanäle, Kodierung und Anordnung der Bytes. Das Erstellen eines Kontextes erfolgt anhand der Übergabe der Verbindung (*connection*), des Audiogerätes (*device*) und der Attribute (*attributes*). Ein Kontext (ac) wird durch den Aufruf von *AFFreeAC* mit dem von *AFCreateAC* zurückgegebenen Bezeichner freigegeben.

AFGetTime(ac)
AFGetTime gibt den gegenwärtigen Wert der Audiouhr zurück. Die Audiouhr läuft in Samples. Bei mehreren Messungen wurde eine Auflösung der Uhr festgestellt, die umgerechnet ungefähr einer Millisekunde entspricht. Die Uhrzeit ist maßgeblich für den Einsatz der folgenden beiden Funktionen.

AFPlaySamples(ac, time, buffer, bytes), AFRecordSamples(ac, time, buffer, bytes, block)
Die Funktionen *AFPlaySamples* und *AFRecordSamples* dienen zum Schreiben und Lesen von Daten. Bei beiden Funktionen müssen Kontext, Zeitpunkt der Audiouhr und Speicherbereich (Länge in Bytes) angegeben werden. Beim Lesen muß zudem angegeben werden, ob blockierend (wartend auf alle angeforderten Daten) oder nicht-blockierend gelesen wird. *AFPlaySamples* gibt die Anzahl der erfolgreich abgelegten Bytes, *AFRecordSamples* den gegenwärtigen Zeitpunkt der Audiouhr zurück. Wie bereits oben erwähnt, kann Audiofile Daten für vier Sekunden in der Vergangenheit und in der Zukunft handhaben. Die folgenden Beispiele in Abbildung 7-2 verdeutlichen den Einsatz der beiden Funktionen. Zugriffe auf Daten, die außerhalb des Speicherintervalls liegen, werden aufgrund ihrer praktischen Irrelevanz für interaktive multimediale Anwendungen nicht betrachtet.

Die in Abbildung 7-2 dargestellten Beispiele zeigen einen Schreibzugriff und vier Lesezugriffe. In dem Beispiel werden Pakete mit einer Länge von 20 Millisekunden verwendet, wobei in den 20 Millisekunden 160 Bytes (= 160 Samples) gelesen werden. Das Beispiel entspricht der PCM-Kodierung (Kapitel 2.1.3). Die Abstände zwischen den dargestellten Zeitpunkten entsprechen 5 Millisekunden oder 40 Bytes. Bei den Erläuterungen zu den Beispielen wird u.a. der Rückgabewert der Funktion *AFRecordSamples* angegeben. Dabei handelt es sich um ungefähre Angaben, da keine konsistenten Werte zurückgegeben werden. In Kapitel 7.3.3 werden gemessene Rückgabewerte vorgestellt. Alle abgesetzten Funktionsaufrufe erfolgen zum gegenwärtigen Zeitpunkt (t5).

- Im Beispiel 1 wird ein Zeitpunkt in der Zukunft (5 Millisekunden) gewählt, zu dem die Daten ausgegeben werden sollen. Die Funktion *AFPlaySamples* übergibt die Daten an Audiofile und kehrt unmittelbar zurück.
- Die Beispiele 2 und 3 zeigen einen lesenden Zugriff zu einem gleichen Zeitpunkt. In beiden Fällen sollen 160 Bytes gelesen werden. Im blockierenden Fall wird bis zu Zeitpunkt t6 gewartet, bis alle Daten vorhanden sind. Im nicht blockierenden Fall werden nur 120 Samples gelesen. Rückgabewerte sind ungefähr t6 im Fall 2 und ungefähr t5 im Fall 3.
- Die Beispiele 4 und 5 stellen einen lesenden Zugriff für Daten der Vergangenheit dar. Alle angeforderten Daten liegen beim Server vor. Die Besonderheit liegt im Rückgabewert der Funktion *AFRecordSamples*. Im Fall 4 wird ungefähr der Zeitpunkt t5

sehr große Verzögerungszeiten auftraten. Die Verwendung von TCP entspricht der Nutzung eines weit verbreiteten Protokolls, nicht jedoch einer Optimallösung. Insbesondere wenn Auftraggeber und Auftragnehmer auf einem System laufen, ist die Lösung von TCP als Kommunikationskomponente zwischen den Komponenten nicht effizient.

Der Server hat zwei Pufferspeicher: Jeweils einen für Daten, die von der Audio-Hardware eingelesen werden (über Mikrophon oder einen anderen Eingang), und einen für Daten, die an die Audio-Hardware ausgegeben werden (über Lautsprecher oder Kopfhörer). Beide Pufferspeicher können Audiodaten für 4 Sekunden beinhalten. Beim Lesen kann demnach auf Daten, die maximal vier Sekunden in der Vergangenheit liegen, zugegriffen werden. Zu schreibende Daten können für Zeitpunkte bis zu vier Sekunden in der Zukunft vorgesehen werden. Maßgeblich für Schreib-/Lesezugriffe ist die Uhrzeit des Audiosystems, welche von der Systemuhr unabhängig läuft. Beim Starten des Servers muß die Abtastfrequenz angegeben werden. Ein Server muß bei Bedarf mehrere Aufträge mit unterschiedlichen Parametern von verschiedenen Anwendungen verarbeiten. Der Server läuft als Hintergrundprozeß. Abbildung 7-1 zeigt ein Beispielszenario für den Einsatz von Audiofile mit einem lesenden und einem schreibenden Auftraggeber.

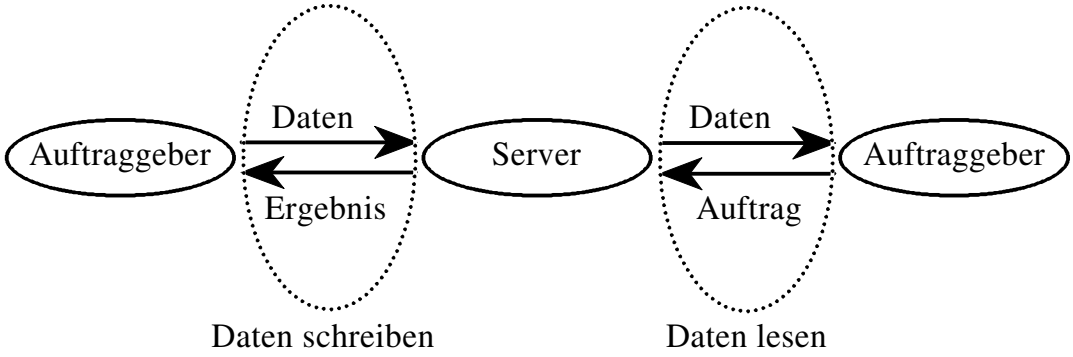


Abbildung 7-1: Audiofile Beispielszenario

In den nächsten Abschnitten werden die für das Lesen und Schreiben von Daten wichtigsten Schnittstellenfunktionen und deren Parameter vorgestellt.

AFOpenAudioConn(audname), AFCloseAudioConn(connection)
Mit diesen beiden Funktionen wird eine Verbindung zwischen einem Auftraggeber und einem Server hergestellt. Innerhalb dieser Verbindung können mittels der in den nächsten Abschnitten vorgestellten Kontexte voneinander unabhängige Schreib-/Lesezugriffe durchgeführt werden. Der Parameter *audname* (Rechnername) kann einen auf einem anderen Rechner laufenden Server bezeichnen. Die Funktion *AFOpenAudioConn* gibt einen Bezeichner für die Verbindung zurück, dieser muß beim Schließen der Verbindung angegeben werden (*connection*).

AFCreateAC(connection, device, attribute_flags, attributes), AFFreeAC(ac)
Diese Funktionen dienen zum Erstellen und Freigeben eines Kontextes. Ein Kontext bedeutet in diesem Zusammenhang eine Kapselung von mehreren Attributen, damit diese nicht bei jedem Zugriff übergeben werden müssen. Zu den Attributen (*attributes*) gehören: Lautstärke

7.3.2 DEC-Audiofile

In diesem Kapitel werden die Grundlagen von DEC-Audiofile [LPG&93] erläutert. Dies erfolgt anhand der Vorstellung der einzelnen Komponenten und der verfügbaren Programmierschnittstelle. Messungen von Audiofile-Funktionen sind Bestandteil von Kapitel 7.3.3, da in diesem Kapitel zeitliche Nebenbedingungen der Portierung dargelegt werden.

DEC-Audiofile wurde von Digital Equipment in Cambridge, USA, entwickelt. Als Unterstützung für flexible Audioanwendungen wurde ein geräteunabhängiges, portierbares und netzwerktransparentes System für computerbasierte Audioanwendungen erstellt. Audiofile wurde in der Version 3R1 (Patchlevel 9i) verwendet, die Quellen sind über das Internet frei verfügbar (z.B. auf dem Rechner crl.dec.com).

Unabhängigkeit von der Audio-Hardware bietet die Möglichkeit, Anwendungen auf Systemen eines Herstellers auszuführen, auch wenn die Systeme mit unterschiedlicher Audio-Hardware bestückt sind. Gemäß dem Stand vom Sommer 1993 läuft Audiofile auf Alpha AXP und RISC Rechnern von Digital, Sparc Arbeitsplatzrechnern von Sun und Indigo Arbeitsplatzrechnern von Silicon Graphics. Die im obigen Abschnitt erwähnte Netzwerktransparenz bezieht sich auf die bei Audiofile erforderliche Kommunikation. Diese wird in den nächsten Abschnitten vorgestellt. Audiofile unterstützt für DEC-Alpha unterschiedliche Audiosysteme, dazu gehört das Standardsystem (baseboard) und die Audio-/Videohardware J300 [OVer94a].

Die Funktionsweise von Audiofile entspricht dem Auftraggeber/Auftragnehmer Prinzip. In der Praxis werden jedoch Auftraggeber und Auftragnehmer i.d.R. auf einem Rechner ausgeführt. Aus der Sicht eines Programmierers besteht Audiofile im wesentlichen aus zwei Komponenten:

- Eine Schnittstelle zum Audiosystem **(Server)** und
- eine Schnittstelle für die Programmierung von Anwendungen **(**Application Programming Interface**; API)**.

Der Zugriff auf das Audiosystem, beispielsweise das Lesen und Schreiben von Audiodaten, wird durch den Server realisiert. Auftraggeber kommunizieren über die von der API zur Verfügung gestellten Funktionen mit dem Server und lassen von diesem Aufträge bearbeiten. Die Auftraggeber und der Server können auf unterschiedlichen Rechnern lokalisiert sein. Ebenso gehören zu Audiofile das anwendungsspezifische Protokoll zur Kommunikation zwischen Server und Auftraggebern und mehrere kleine Anwendungen. Diese verdeutlichen die Arbeitsweise von Audiofile. Die beiden letztgenannten Komponenten werden im folgenden nicht weiter betrachtet.

Der Datentransport zwischen Auftraggebern und dem Server kann gemäß den Autoren „über nahezu jedes Transportprotokoll“ abgewickelt werden. In der Version 3R1 wird TCP [Davi88] verwendet. Die Autoren beschreiben TCP als ausreichend im lokalen Netzwerk-Bereich. Bei transkontinentalen Verbindungen wurden jedoch Probleme aufgrund überlaufender Puffer in Zwischensystemen **(Routern)** festgestellt. Die durch den Pufferüberlauf verloren gegangenen Pakete zogen eine Verkleinerung des Flußkontroll-Fensters (slow-start) nach sich, wodurch

7.3 Ein-/Ausgabe von Audio

Die Realisierung des Audiozugriffs bei NeVoT ist Gegenstand der folgenden Unterkapitel. Nach einer Einleitung wird zunächst der Ausgangspunkt für die Umsetzung des Problems diskutiert, daran anschließend erfolgt die Erläuterung der Lösung.

7.3.1 Einleitung

Im folgenden wird zunächst der Hintergrund der Anpassung von Ein- und Ausgabe von Audio bei NeVoT erläutert und die Motivation für die verwendete Programmierschnittstelle gegeben. Daran schließt sich eine Kurzübersicht über Kapitel 7.3 an.

Der Zugriff auf das Audiosystem erfolgt bei den NeVoT-Versionen für Sun, HP und anderen Arbeitsplatzrechnern ohne zusätzliche Programmierschnittstelle. Diese Vorgehensweise ist beim Einsatz von NeVoT auf Alpha-Arbeitsplatzrechnern weder erforderlich noch wünschenswert, da mit DEC-Audiofile [LPG&93] und MME („Multimedia Extensions“) [OVer94b] zwei Schnittstellen für den Zugriff auf das Audiosystem existieren. Zudem sind Alpha-Rechner teilweise mit unterschiedlicher Audiohardware ausgestattet. Daraus kann bei hardwareabhängiger Implementierung ohne Programmierschnittstelle ein wesentlich höherer Aufwand entstehen. Für die Durchführung der Portierung ist demnach die Auswahl einer Programmierschnittstelle empfehlenswert.

Die Schnittstellen MME und Audiofile unterscheiden sich in ihrer Komplexität und ihrem Einsatzbereich. Ein hier nicht näher vorgestellter Vergleich der zur Verfügung stehenden Funktionen zeigt, daß die Komplexität der Schnittstelle bei Audiofile geringer ist als bei MME. Bei Audiofile kann mittels weniger Funktionen ein schreibender oder lesender Zugriff auf das Audiosystem erfolgen, bei MME ist die Anwendung mehrerer aufwendiger Funktionen erforderlich. Ein Vergleich der Schnittstellen unter Berücksichtigung der Einsetzbarkeit liefert ebenso unterschiedliche Ergebnisse. MME ist eine Ergänzung zum „Microsoft Multimedia Programming Interface“ [OVer94b], welches für Computer mit dem Betriebssystem Windows NT [Cust93] entwickelt wurde. Audiofile läuft auf den Systemen Sparc von Sun, Indigo von Silicon Graphics und RISC- oder Alpha-Arbeitsplatzrechnern von Digital. Die von Audiofile unterstützten Systeme verwenden überwiegend, wie bereits erwähnt, UNIX-ähnliche Betriebssysteme. Demnach kann bei einer weiteren Portierung von NeVoT die Audioschnittstelle Audiofile ggf. wieder verwendet werden. Eine detailliertere Betrachtung der unterschiedlichen Schnittstellen ist nicht Bestandteil dieser Arbeit, vielmehr steht die Problemlösung mittels DEC-Audiofile im Vordergrund.

Abschließend läßt sich zusammenfassen, daß die unabhängigere und einfachere Audioschnittstelle Audiofile für den Einsatz bei NeVoT vorzuziehen ist. Die für Audiofile notwendige Bedingung der ausreichenden Leistungsfähigkeit war aus einer in [LPG&93] dargestellten Leistungsbeschreibung der Schnittstellenfunktionen ersichtlich und wurde durch die Portierung bestätigt.

werden die Daten mit dem höchstwertigen Byte beginnend in den Speicher geschrieben **(big-endian** oder **network-byte-order)** oder mit dem niedrigstwertigen Byte zuerst **(little-endian)**. Diese unterschiedliche Byteanordnung ist historisch bedingt. Systeme eines Herstellers haben aus Kompatibilitätsgründen die Byteanordnung innerhalb der Systemfamilie nicht verändert. Im Laufe der Zeit wurden auf den Systemen immer mehr UNIX-ähnliche Betriebssysteme eingesetzt. Dadurch existieren ähnliche Betriebssysteme mit unterschiedlicher Byteanordnung. Alpha-Arbeitsplatzrechner besitzen eine little-endian Anordnung, Sun-Rechner eine big-endian Anordnung. Dieser Sachverhalt kann u.a. zu Problemen führen, wenn Speicherbereiche mit anderen Werten (z.B. Konstanten) verglichen werden.

Zur Lösung des Problems existieren die Funktionen *htons*, *htonl*, *ntohl* und *ntohs*. Mit diesen Funktionen (network-byte-order to host-byte-order und host-byte-order to network-byte-order) kann die Byteanordnung eines Wertes verändert werden. Beide Funktionen existieren für 16-Bits Typen (s = short) und für 32-Bits Typen (l = long). Die host-byte-order entspricht der Byteanordnung des Systems. Je nach Byteanordnung des Systems wird die Anordnung der Daten entweder verändert oder nicht. Bei der Portierung von NeVoT wurden beide Typen von Funktionen verwendet. Insbesondere zwei spezielle Programmteile erfordern eine Konvertierung zwischen den Anordnungen. Zum einen ist bei der Programmierung von Socket-Schnittstellen eine Angabe der Parameter im big-endian Format vorgeschrieben. Des weiteren schreibt die Spezifikation des von NeVoT verwendeten Protokolls RTP vor, daß alle Werte in den Paketen im Format big-endian gespeichert sind. Neben den vorgestellten Problembereichen wurden auch in anderen Programmteilen Änderungen erforderlich.

7.2.3 Deklarationsdateien und Funktionen

Neben den in den letzten Kapiteln vorgestellten Unterschieden in der Darstellung von Datentypen und Werten existieren zwischen den Betriebssystemen Unterschiede bezüglich der vom System zur Verfügung gestellten Funktionen. Auch die Deklaration von Funktionen, welche in Deklarationsdateien zusammengefaßt sind, unterschiedet sich zwischen den Systemen.

Von NeVoT werden folgende, nicht vom Betriebssytem OSF/1 zur Verfügung gestellte, Funktionen verwendet.

- *infinity*. Diese Funktion gibt den höchsten verwendbaren Wert zurück (die Konstante HUGE_VAL).
- *gmtoff*. Diese Funktion gibt die Zeitzone zurück, in der sich das Systems befindet.

Beide Funktionen müssen explizit programmiert und in einer architekturspezifischen Datei (alpha_sys.c) zur Verfügung gestellt werden.

Des weiteren unterscheiden sich die Deklarationsdateien auf unterschiedlichen Systemen. Die in den Programmdateien verwendeten Befehle *#include* müssen entsprechend angepaßt werden. Ein Beispiel hierfür ist die Funktion *malloc*, welche unter OSF/1 nicht in *malloc.h* sondern in *stdlib.h* deklariert ist.

7.2 Datenstrukturen

Bei der Portierung waren einige Programmteile und Datenstrukturen von NeVoT und RTP anzupassen. Das vorliegende Kapitel gibt Auskunft über die erforderlichen Anpassungen. Dabei handelt es sich um betriebssystemspezifische und hardwarespezifische Aspekte.

7.2.1 Datentypen

Die Programmkomponenten von NeVoT verwenden, wie die meisten anderen C-Programme auch, verschiedene Datentypen. Diese Datentypen haben in unterschiedlichen Rechensystemen teilweise verschiedene Darstellungen bzw. Wertebereiche. Beispielsweise kann es vorkommen, daß ein Integer-Wert auf dem einen System einer vier Bytes großen Zahl entspricht, auf einem anderen System einer zwei Bytes großen Zahl. Daraus resultierende Probleme werden im folgenden erläutert.

Große Teile des Programms sind für die Problematik der unterschiedlichen Datenstrukturen bereits ausgelegt. Dazu existieren eigene Datentypen, wie *u_int8*, *u_int16* und *u_int32*. Diese Datentypen entsprechen vorzeichenlosen Integer-Werten der Größe 8, 16 bzw. 32 Bits. Für die Portierung ist bei diesen Typen lediglich eine entsprechende Definition erforderlich. Bei Alpha entspricht der Typ *u_int32* dem Typ *unsigned int*. Die vorgestellten Datentypen werden insbesondere bei der Beschreibung von Datenpaketen verwendet, wo die Feldgröße durch die jeweilige Spezifikation vorgegeben ist. Da aber nicht ausschließlich die vorgestellten Datentypen verwendet werden, ist darüber hinaus bei Verwendung von Datentypen wie *int*, *float* usw. eine Kompatibilität zu gewährleisten.

Bei NeVoT werden häufig Zeiger auf Datenstrukturen verwendet. Diese weit verbreitete Programmiertechnik ermöglicht eine flexible und effiziente Programmierung. Alpha-Arbeitsplatzrechner besitzen eine 64-Bits Architektur, für die Darstellung von Zeigern werden 64 Bits (= 8 Bytes) verwendet. Andere Systeme, insbesondere Rechner von Sun, auf denen NeVoT entwickelt wurde, verwenden aufgrund ihrer Architektur eine Zeigerdarstellung von 32 Bits. Diese Unterschiede rufen Probleme hervor, insbesondere dann, wenn Zeiger weiterverarbeitet werden (z.B. einer anderen Variablen zugewiesen werden). Für diese Problematik ist eine Lösung erforderlich, die für alle unterstützten Systeme Lauffähigkeit ermöglicht. Zur Lösung kann u.a. Typkonversion **(cast)** eingesetzt werden, mit der ein Datentyp in einen anderen überführt wird. Dabei muß berücksichtigt werden, daß unter Umständen Informationen verloren würden, z.B. wenn ein 64-Bits Wert in einen 32-Bits Wert überführt wird.

7.2.2 Byteanordnung

Bereits im vorangegangenen Kapitel wurde eine unterschiedliche Anzahl von Bytes pro Datentyp vorgestellt. Neben diesen systemabhängigen Besonderheiten unterscheidet sich auch die rechnerinterne Speicherung von Daten. Vor allen Dingen bei Variablen, deren Größe ein Byte überschreitet, gibt es zwei Möglichkeiten, diese Werte im Speicher abzulegen. Entweder

7 NeVoT auf DEC-Alpha

In diesem Kapitel wird die Portierung und Integration des Audio-Konferenzsystems NeVoT und des Protokolls RTP von Sun-Arbeitsplatzrechnern in die Systemumgebung von Alpha-Arbeitsplatzrechnern beschrieben. In Kapitel 7.1 wird dabei auf die allgemeine Problemstellung eingegangen, daran anschließend erfolgen die Erläuterungen zu den erforderlichen Modifikationen des Programms.

7.1 Überblick

Auf Arbeitsplatzrechnern unterschiedlicher Hersteller werden häufig Betriebssysteme eingesetzt, welche eine enge Verbundenheit zu UNIX [Bour82] besitzen. In der Regel sind diese Betriebssysteme den Rechnern der unterschiedlichen Hersteller angepaßt. Dies hat zur Folge, daß die meisten Arbeitsplatzrechner ähnliche aber nicht identische Betriebssysteme besitzen. Diese Gegebenheit macht es wünschenswert, auf einem System entwickelte Programme auch auf anderen Systemen lauffähig zu machen (zu portieren). Dabei können Gemeinsamkeiten genutzt werden, (betriebs-)systemabhängige Besonderheiten müssen jedoch angeglichen werden. Insbesondere bei direkten Zugriffen auf die Hardware sind teilweise grundlegende Programmanpassungen erforderlich.

Im folgenden wird die Portierung des Audio-Konferenzsystems NeVoT und des Protokolls RTP vorgestellt. NeVoT und RTP wurden auf Sun-Arbeitsplatzrechnern [Russ91] entwickelt und seitdem nach mehreren Betriebssystemen portiert. Das Zielbetriebssystem der vorgestellten Portierung ist OSF/1 [OSF91] auf Alpha-Arbeitsplatzrechnern der Firma Digital.

In den Kapiteln 7.2 und 7.3 werden die für die Portierung erforderlichen Veränderungen vorgestellt. Kapitel 7.2 beinhaltet Änderungen an dem Programm und den Datenstrukturen. Dazu gehören beispielsweise die Berücksichtigung von unterschiedlichen Darstellungen von Daten innerhalb des Speichers, Unterschiede von System-Bibliotheken und andere betriebssystemspezifische Besonderheiten. Da diese Änderungen innerhalb des Programms i.d.R. an mehreren Stellen durchgeführt wurden, werden die Modifikationen allgemein vorgestellt.

Bestandteil von Kapitel 7.3 ist die Neugestaltung der Ein-/Ausgabe von Audio in NeVoT. Diese wurde aufgrund komplett verschiedener Schnittstellen zum Audiosystem bei den unterschiedlichen Systemen notwendig. In Kapitel 7.3 erfolgt eine genaue Betrachtung der Voraussetzungen und Möglichkeiten. Anhand dieser Grundlage wird auf die konkrete Realisierung des Audiozugriffs eingegangen.

von Audio und Video. Anhand von drei Anwendungsmöglichkeiten wurde gezeigt, welchen Vorteil die Verwendung von RTP gegenüber AVXP bietet:

- Interoperabilität zwischen MMC und weiteren Konferenzsystemen.
- Adapitve Vorwärtsfehlerbehebung.
- Empfängergesteuerte Senderegulierung.

Über die vorgestellten Möglichkeiten hinaus sind weitere Ansätze möglich, z.B. die Realisierung von Zwischensystemen (z.B. zum Mischen von Audio).

6.3 Zusammenfassung

In diesem Kapitel wurden zunächst die zwei Protokolle AVXP und RTP, die den Transport von Audio und Video unterstützen, verglichen. Die Funktionalität beider Protokolle ist gemäß OSI-BRM in die Transport- und Sitzungsschicht einzuordnen. Bei AVXP handelt es sich um eine spezialisierte Lösung, welche für den Einsatz in dem Konferenzsystem MMC entwickelt wurde. RTP hingegen ist ein allgemeiner Ansatz für die Kommunikation innerhalb und zwischen unterschiedlichen verteilten multimedialen Anwendungen, der von der IETF (Internet Engineering Task Force) spezifiziert wurde. Zum Zeitpunkt der Arbeit stand die Verabschiedung von RTP als RFC (request for comments) unmittelbar bevor.

Der Vergleich der Protokolle AVXP und RTP hat gezeigt, daß es sich bei RTP im Gegensatz zu AVXP um ein wesentlich flexibleres Protokoll handelt. Zu den Vorteilen von RTP gehören ein variabler Paketkopf, eine Versionsidentifikation, eine Unterstützung von Zwischensystemen und eine Unterstützung von Verschlüsselung. Weiterhin bietet das zu RTP gehörende Kontrollprotokoll RTCP die Möglichkeit, Kontrollinformation bezüglich Konferenzteilnehmern und der Qualität der Datenübertragung auszutauschen. Diese Möglichkeit ist bei AVXP nicht vorhanden.

Die Qualität der Datenübertragung (Verzögerungszeiten, -schwankungen und Paketverluste) hängt bei RTP und AVXP von den darunterliegenden Protokollen und Netzen sowie der Netzlast ab. Bei einer Verwendung von UDP/IP und CSMA/CD (Kapitel 2.3) ist aufgrund der unzuverlässigen Übertragung mit Paketverlusten und Verzögerungsschwankungen zu rechnen. An dieser Stelle wäre die Verwendung eines anderen Kommunikationssystems, als Diensterbringer für AVXP oder RTP, wünschenswert. Um Verzögerungszeiten zu reduzieren und Verzögerungsschwankungen zu eliminieren, ist eine andere Netzwerktechnik als CSMA/CD erforderlich. Eine Möglichkeit besteht in der Verwendung des asynchronen Transfer-Modus ATM (Kapitel 2.3). Laut Autoren von RTP kann das Protokoll direkt auf ATM aufgesetzt werden [Schu93; Kapitel 1.1]. Aus der Kombination von ATM und RTP ergäbe sich ein gut geeignetes System zur Übertragung von Audio und Video.

Bei der Analyse des Datentransports in MMC stellt sich die Frage, welche Vorteile der im Rahmen des Projekts BERKOM entwickelte Multimedia Transport Service (MMT), bei dem die Protokolle XTP/ST-II verwendet werden, als Kommunikationssubsystem bietet. Diese Kommunikationskomponenten bieten keine Möglichkeit einer Garantie bezüglich maximalen Verzögerungsschwankungen. Paketverluste lassen sich bei XTP nur durch den Einsatz von Übertragungswiederholung ausschließen. Der von XTP/ST-II unterstützte Mechanismus der Ressourcen-Reservierung kann zu einer Reduzierung der Paketverlustrate führen. Wenn lediglich unzuverlässige Übertragung gefordert wird, bieten XTP/ST-II außer der Ressourcen-Reservierung keine wesentlichen Vorteile gegenüber UDP/IP.

In Kapitel 6.2 wurde die konzeptionelle Erweiterung des Konferenzsystems MMC um RTP vorgestellt. MMC verwendet zum Zeitpunkt der Arbeit das Protokoll AVXP zum Transport

Bei der Verwendung von Gruppenkommunikation kann der Fall eintreten, daß mehrere Empfänger unterschiedliche Qualität verarbeiten können und somit unterschiedliche Rückmeldungen beim Sender eintreffen. Dieser muß bzw. kann aufgrund der empfangenen Informationen die oben aufgeführten Parameter für die Qualität der Übertragung anpassen. Werden unterschiedliche Rückmeldungen erhalten, so muß eine Politik ausgewählt werden, nach der die Datenübertragung angepaßt wird. Hierfür gibt es u.a. folgende Möglichkeiten:

- Orientierung an der niedrigsten gewünschten Qualität.
- Orientierung an der höchsten gewünschten Qualität.
- Anhebung/Senkung der Qualität nicht über einen Schwellwert hinaus.

Entsprechend der Zielsetzung der Übertragung (Konferenz mit hoher Qualität oder Konferenz, an der alle Teilnehmer partizipieren können) muß eine Politik ausgewählt werden.

Im folgenden wird ein anwendungsspezifisches Kontrollpaket (RTCP) vorgestellt, welches zur Informationsübermittlung vom Empfänger zum Sender eingesetzt werden kann (vgl. Kapitel 4.3).

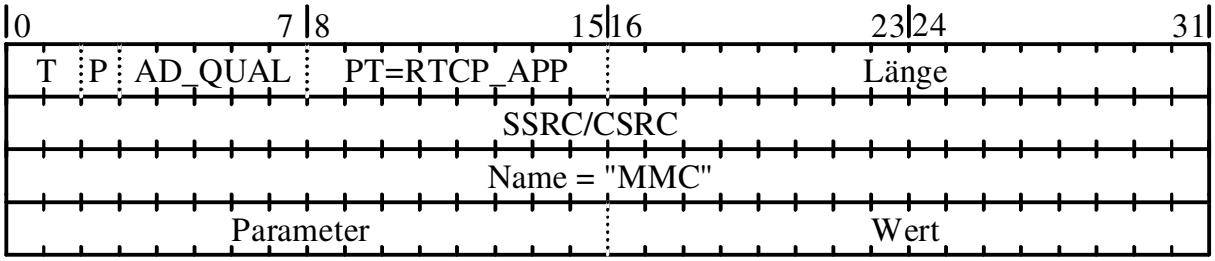


Abbildung 6-3: RTCP-Paket zur empfängergesteuerten Senderegulierung

Das Feld **Name** (4 Bytes) beinhaltet den Namen der Institution oder Einrichtung, welche das Paket spezifiziert hat. In dem Feld sind vier Buchstaben gespeichert. Zur Kennzeichnung des Kontextes MMC wurde der Bezeichner „MMC“ gewählt.

Das mit **AD_QUAL** bezeichnete Feld (5 Bits) entspricht dem Subtyp-Feld des RTCP-Pakets. Für dieses Feld ist der Eintrag einer Konstanten AD_QUAL (adapt quality) vorgesehen.

Die Felder **Parameter** (2 Bytes) und **Wert** (2 Bytes) dienen der Angabe eines Parameters und eines Wertes, auf den der Parameter gesetzt werden soll. Hierfür muß von den MMC implementierenden Unternehmen eine Liste mit Parametern festgelegt werden. Der folgenden konkrete Vorschlag gibt ein Beispiel für den Einsatz eines Kontrollpaketes.

Bei MMC liegt die Festlegung eines Parameters MJPEG_FRATE (Motion-JPEG framerate) nahe, durch den die Anzahl der Bilder pro Sekunde bei der in MMC verwendeten Kodierung Motion-JPEG verändert wird. Motion-JPEG besteht aus JPEG-Einzelbildern, die jeweils komplett übertragen werden (vgl. auch Kapitel 2.1.4). Als Wert ist dann die Anzahl der gewünschten Bilder pro Sekunde eingetragen.

Redundanzpaket gebildet wird. Das Feld **FPN** (FEC packet number; 4 Bits) gibt an, um welches der R Redundanzpakete es sich handelt. Im Anschluß daran folgen die Angaben Zeitstempel, Sequenznummer, Markierungsbit, Nutzdatentyp und Padding für jedes Originalpaket aus dem das Redundanzpaket gebildet wurde. Bei der Bildung von einem Redundanzpaket aus zwei Originalpaketen sind in der Paketkopferweiterung demach die Angaben für zwei Originalpakete enthalten.

6.2.3 Empfängergesteuerte Senderegulierung

Die empfängergesteuerte Senderegulierung bezeichnet die Regelung der Datenrate eines Senders in Abhängigkeit von Informationen von Empfängern. Die folgenden Ausführungen zeigen, in welchen Fällen dieser Mechanismus einsetzbar ist und welcher Nutzen daraus gezogen werden kann. Die Betrachtungen orientieren sich an der Übertragung von Video, da sich daran der geschilderte Sachverhalt am besten erläutern läßt.

Zunächst wird von der Kommunikation zwischen zwei Teilnehmern ausgegangen. Dieser Sachverhalt trifft insbesondere den momentanen Stand der Entwicklung, bei dem AVXP in MMC über UDP realisiert ist. Zwischen je zwei Teilnehmern wird eine Verbindung (Verbindung nicht im Sinne einer verbindungsorientierten Übertragung, sondern als logischer Kanal) zur Kommunikation aufgebaut. Wenn sich zwei Teilnehmer auf die Kommunikation von Video geeinigt haben, so ist eine Kodierung festgelegt, Parameter der Kodierung können jedoch verändert werden (bei MMC). Dazu gehört insbesondere die Bildwiederholfrequenz, aber auch die Bildgröße oder die Farbinformation sind denkbar.

Eine Anpassung der Parameter kann in zwei Fällen sinnvoll sein:

- Der Empfänger stellt aufgrund empfangener Daten fest, daß im Netzwerk Daten verloren gehen (z.B. über den Vergleich von Sequenznummern). Falls diese nicht mit Vorwärtsfehlerbehebung (Kapitel 6.2.2) gesichert werden können, weil die Paketverlustrate zu hoch ist, so ist es sinnvoll, eine Videoübertragung mit geringerer Bildwiederholfrequenz, geringerer Bildgröße oder lediglich schwarzweißer Darstellung zu realisieren.
- Der Empfänger stellt durch geeignete Mechanismen fest, daß die Verarbeitungsgeschwindigkeit entweder für die Bearbeitung der übertragenen Daten nicht ausreicht oder noch ausreichend Verarbeitungsgeschwindigkeit vorhanden ist, um Video in einer höheren Qualität zu empfangen und darzustellen. Dabei ist es sicherlich sinnvoller, sich einer gewünschten Qualität von unten zu nähern, d.h. die Parameter für die Qualität anzuheben. Diese Vorgehensweise ist insbesondere für die Auslastung der beteiligten Systeme sinnvoll.

Unabdingbar für beide Einsatzbereiche ist ein Kontrollprotokoll, welches die Informationen, die zu einer Anpassung der Senderate führen, vom Empfänger zum Sender transportiert. Beim Einsatz von RTP/RTCP in MMC ist dies möglich, nicht jedoch bei der Verwendung von AVXP.

Für einen Empfänger von Daten muß ersichtlich sein, daß Daten durch FEC gesichert sind. Dies ist sowohl für Originaldaten als auch für Redundanz-Daten erforderlich. Bei Redundanz-Daten ist dieser Sachverhalt intuitiv ersichtlich, da die Originaldaten rekonstruiert werden müssen. Bei Originaldaten ist eine Kennzeichnung des Einsatzes von FEC ebenfalls erforderlich, weil dem Empfänger angezeigt werden muß, daß auf gegebenenfalls noch folgende Redundanzpakete gewartet werden kann bzw. muß.

Für die Kennzeichnung des Einsatzes von FEC bestehen zwei Möglichkeiten. Entweder es erfolgt eine Kennzeichnung in der Paketkopferweiterung oder im Feld Nutzdatentyp (PT). Bei der Verwendung des Feldes PT müßte für jeden unterschiedlichen Nutzdatentyp ein neuer Typ (gleiche Nutzdaten, aber mittels FEC gesichert) spezifiziert werden. Aus diesem Grund ist es vorzuziehen, die Kennzeichnung für FEC in der Paketkopferweiterung vorzunehmen. Das Flag für die Paketkopferweiterung **X** muß damit immer gesetzt sein, unabhängig davon, ob Originaldaten oder Redundanzpakete versendet werden. Falls keine FEC unterstützt wird, kann die Paketkopferweiterung ignoriert werden.

Für den Aufbau der O Originalpakete gilt:
Alle Felder des Paketkopfs sind wie bei Übertragung ohne FEC zu verwenden. Dem Paketkopf folgt eine Paketkopferweiterung. Die Erweiterung besteht nur aus vier Bytes. In der Erweiterung ist das Flag **FEC** gesetzt, das Feld **FT** (FEC type; 3 Bits) gibt den Typ der verwendeten FEC an (siehe unten). Das Feld **FPN** (FEC packet number; 4 Bits) hat bei Originalpaketen keine Bedeutung.

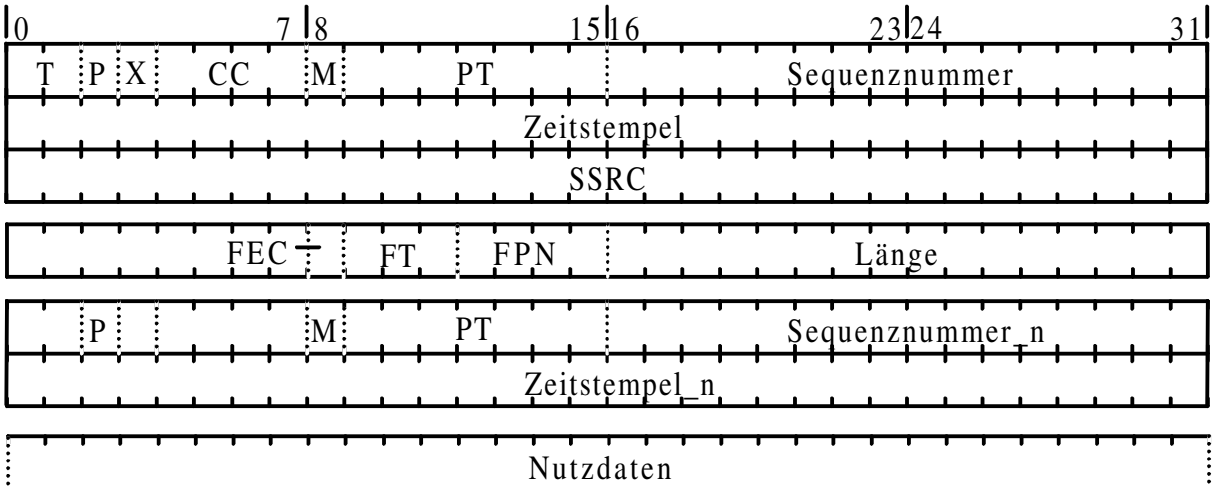


Abbildung 6-2: Paketkopf und -erweiterung für Vorwärtsfehlerbehebung

Für den Aufbau der R Redundanzpakete gilt:
Die Angaben Zeitstempel, Sequenznummer, Markierungsbit, Nutzdatentyp und Padding im Paketkopf sind irrelevant. Die Interpretation der ersten vier Bytes der Paketkopferweiterung ist wie folgt: Das Flag **FEC** zeigt an, daß Vorwärtsfehlerbehebung verwendet wird, der Typ der verwendeten Vorwärtsfehlerbehebung geht wie bei Originalpaketen aus dem Feld FT hervor. So könnte FT = 1 für eine bestimmte Abbilung stehen, bei der aus zwei Originalpaketen ein

Werden Daten mittels Gruppenkommunikation (Multicast) übertragen, so ist nicht auszuschließen, daß einer der Empfänger durch FEC gesicherte Daten nicht interpretieren kann oder möchte. Dieser Fall kann eintreten, wenn einer der Empfänger mitgeteilt hat, daß die Übertragungsqualität entsprechend schlecht ist. Es ist deshalb erforderlich, daß die FEC so durchgeführt wird, daß Redundanzpakete ignoriert werden können, die Original-Nutzdaten aber dennoch extrahiert werden können.

Die Erzeugung von Redundanz kann auf unterschiedliche Weise erfolgen. Im folgenden wird ein Beispiel für die Erzeugung von Redundanz gegeben. Auf der Basis von zwei Nutzdaten-Paketen (Pakete a und b) können neue Pakete erzeugt werden. Falls die Pakete eine unterschiedliche Länge besitzen, so muß das kürzere durch Füllbytes erweitert werden. Im einfachsten Fall wird je ein Bit der Nutzdaten von Paket a und ein Bit der Nutzdaten von Paket b mittels der Operation XOR (exklusives oder) verknüpft. Das daraus entstehende Paket c wird ebenfalls übertragen. Werden zwei der Pakete a, b bzw. c erhalten, können die Pakete a und b rekonstruiert werden. Wird nicht nur jeweils ein Bit verknüpft, so lassen sich aus zwei Paketen noch mehr Redundanzpakete erzeugen. Die Erzeugung solcher Pakete erfolgt mit Hilfe der polynomiellen Algebra [LiCo83], deren Erläuterung den Rahmen dieses Kapitels sprengen würden. Werden mehrere Pakete Redundanz erzeugt, beispielsweise c und d, so ist nach Erhalt von zwei der vier Pakete eine Rekonstruktion von a und b möglich. Das Erzeugen und das Senden von Redundanzpaketen ist erst möglich, wenn die Originaldaten vorliegen. Dadurch kann es bei der Ausgabe zu Verzögerungen kommen.

Aus diesen Ausführungen ergeben sich die folgenden Anforderungen an eine adaptive Vorwärtsfehlerbehebung.

- Unterschiedlicher Grad der Redundanz (d.h. unterschiedliche Sicherung) muß unterstützt werden.
- Die Originaldaten müssen, wenn keine FEC unterstützt wird, extrahierbar sein.
- Die FEC muß für Pakete unterschiedlicher Länge geeignet sein.
- Der Mechanismus der FEC muß erkennbar sein.
- Aus empfangenen redundanten Daten muß eine Rekonstruktion der Originaldaten möglich sein.

Für die Erläuterung des Aufbaus der FEC-Pakete gilt folgende Bezeichnung: Aus **O Originalpaketen** werden **R Redundanzpakete** erzeugt.

Aufbau der FEC-Pakete

Die Paketkopferweiterung bei RTP bietet sich an, die für die FEC erforderlichen Informationen aufzunehmen. In Abbildung 6-2 ist ein Vorschlag für ein Paketformat gegeben, welches FEC unterstützt. Die erste Komponente zeigt den Paketkopf. In der Paketkopferweiterung (Komponenten 2 und 3) sind die bezüglich der FEC erforderlichen Informationen gespeichert. Anhand der Erläuterung der Paketfelder wird gezeigt, daß den oben aufgeführten Anforderungen genügt wird.

Die nachfolgenden Messungen zeigen, in welchen Übertragungsbereichen mit Paketverlusten zu rechnen ist. Die Messungen wurden im Internet mittels des Befehls *ping* durchgeführt. Der Befehl *ping* initiiert die Übertragung von ICMP-Paketen zum angegebenen Empfänger und zurück. Dabei wird die Paketverlustrate ausgegeben. Die Messungen wurden fünfmal, jeweils zu unterschiedlichen Zeitpunkten, wiederholt. Es wurden vier verschiedene Übertragungsstrecken vermessen, von lokalen Verbindungen bis hin zu transkontinentalen Verbindungen. Wesentlich ist dabei nicht der Weg, den die Daten durchlaufen sind, sondern die erhaltenen Ergebnisse. Die Meßergebnisse sind in Tabelle 6-3 dargestellt. Die Ergebnisse zeigen, daß insbesondere bei Weitverkehrsverbindungen hohe Paketverlustraten auftreten. Der Ergebnisbereich bei den Paketverlustraten geht von null bis hin zu ca. 40 %. Somit ist gezeigt, daß in Netzen zum einen Übertragungsstrecken existieren, bei denen in der Regel kein Paketverlust auftritt, zum anderen manche Verbindungen mit hohen Paketverlusten behaftet sind. Bei den Meßergebnissen trat, wenn überhaupt, eine Paketverlustrate von durchschnittlich mindestens vier Prozent auf. Aufgrund der Ergebnisse liegt der Schluß nahe, daß FEC eingesetzt werden sollte, sobald ein geringer Schwellwert (beispielsweise Paketverlustrate > 1%) überschritten wird. FEC sollte jedoch erst bei mehrmaliger Überschreitung des Schwellwerts aktiviert werden, um kurzzeitige Pufferüberläufe zu ignorieren.

Verbindungs-typ	**Messung 1**	**Messung 2**	**Messung 3**	**Messung 4**	**Messung 5**	**Mittelwert**
lokal	0	0	0	0	0	0
campusweit	0	0	0	0	0	0
weitverkehr	7	1	2	3	7	4
trans-kontinental	20	46	42	20	33	32,2

Tabelle 6-3: Paketverlustraten im Internet (in %)

Bereits oben wurde erwähnt, daß Adaptivität auch bezüglich der Priorität der Daten ausgelegt werden kann. Dieser Ansatz wird in [KEL94] verfolgt. Tritt Paketverlust während einer Übertragung auf, so ist es sinnvoll, wichtige Daten besser zu sichern als unwichtigere Daten. Der Sachverhalt unterschiedlicher Priorität bei Nutzdaten läßt sich besonders deutlich an der Übertragung von MPEG-Video (Kapitel 2.1.4) zeigen. Der Empfang von I-Bildern ist wesentlich wichtiger als der Empfang von P- oder B-Bildern. Demzufolge ist es sinnvoller diese Pakete besser zu schützen als die anderen Pakete, d.h. mehr Redundanz während der Übertragung hinzuzufügen.

Kontrollinformationen können ebenfalls mittels FEC gesichert werden. Dies ist insbesondere sinnvoll, wenn es sich um Informationen handelt, die - ebenso wie die Nutzdaten - minimal zeitversetzt beim Empfänger ankommen sollen. Dieser Fall schließt den Transport von Kontrollinformationen über ein Protokoll aus, welches Übertragungswiederholung im Fall von Verlusten einsetzt.

existiert allerdings auch bei Übertragungswiederholung aufgrund von Paketverlust. Es ist demzufolge wichtig, Redundanz nicht im Übermaß hinzuzufügen.

Die folgenden Ausführungen werden die hier vorgestellten Ansätze vertiefen. Ziel ist eine adaptive Vorwärtsfehlerbehebung bezüglich ganzen Paketen, welche sich der Qualität der Übertragung sowie der Priorität der zu übertragenden Daten anpaßt. Nachfolgend steht der Begriff **Adaptivität** für die adaptiven Aspekte der hier vorgestellten adaptiven Vorwärtsfehlerbehebung.

Adaptivität

Adaptivität kann in zwei Dimensionen ausgelegt werden, zum einen bezüglich der Wichtigkeit der zu übertragenden Daten, zum anderen bezüglich der Zeit. Adaptivität bezüglich der Zeit bedeutet hier, daß FEC nur dann eingesetzt wird, wenn Pakete bei der Übertragung tatsächlich verloren gehen. Ein erster Ansatz der FEC besteht in der ständigen Übertragung von Zusatzinformation, unabhängig von der Qualität der Übertragung. Bei einer Übertragung ohne Paketverlusten ist die Notwendigkeit von Vorwärtsfehlerbehebung jedoch nicht notwendig, demnach wird hier der Vorschlag für eine zeitlich adaptive Vorwärtsfehlerbehebung gemacht. Voraussetzung für dieses Verfahren ist die Möglichkeit einer Überwachung der Qualität der Datenübertragung (quality of service monitoring), insbesondere der Überwachung der Paketverluste. Dafür ist ein Kontrollprotokoll erforderlich, welches zwischen zwei Teilnehmern entsprechende Informationen übermittelt. Das zu RTP gehörende Kontrollprotokoll RTCP bietet die erforderliche Funktionalität, AVXP ist zur Verwendung einer zeitlich adaptiven Vorwärtsfehlerbehebung nicht geeignet, da ein Kontrollprotokoll ganz fehlt. Bei RTCP dient der Empfangsbericht (vgl. Kapitel 4.3) zur Übermittlung der erforderlichen Informationen. Das Feld *Zähler der erwarteten Pakete* gibt an, wieviel Pakete rechnerisch zwischen dem Beginn der Übertragung und dem letzten empfangenen Paket liegen. Zusammen mit dem Inhalt des Feldes *Paketzähler des Empfängers* kann die Paketverlustrate bezüglich einer Übertragung ermittelt werden. Es ist keine Definition neuer Kontrollpakete erforderlich. Der Sender kann durch den periodischen Erhalt eines Empfangsberichts vom Empfänger entscheiden, ob FEC zukünftig eingesetzt werden sollte. Dabei sollte in regelmäßigen Abschnitten untersucht werden, ob Paketverluste bei der Übertragung aufgetreten sind. Bei der Entscheidung für den Einsatz einer Vorwärtsfehlerbehebung sollte berücksichtigt werden, daß sie nicht aufgrund eines kurzzeitig aufgetretenen Paketverlusts eingesetzt wird, sondern nur, wenn durch den Erhalt mehrerer aufeinanderfolgender Kontrollpakete ein dauerhafter Paketverlust festgestellt wird. Die Festlegung eines Schwellwerts, ab dem die adaptive FEC eingesetzt werden sollte, ist einer Anwendung überlassen, bei den unten aufgeführten Meßergebnissen wird ein Vorschlag gegeben.

Der folgende Abschnitt beinhaltet Ausführungen über Paketverluste während einer Übertragung. Im Anschluß daran wird die Art und Weise der Vorwärtsfehlerbehebung betrachtet.

übertragenen Audiodaten empfangen kann. Hierfür bietet sich z.B. das in Kapitel 3 vorgestellte NeVoT an. Für die Kommunikation zwischen NeVoT und MMC ist jedoch ein Protokoll erforderlich, welches von beiden Systemen unterstützt wird. Dabei ist es aufgrund der bereits vorgestellten Argumente (u.a. Standardisierung, Flexibilität) sinnvoller, in MMC auch RTP zu unterstützen, da es sich hierbei um ein weiter verbreitetes Protokoll handelt, und nicht um eine spezielle Lösung. Für das hier vorgestellte Szenario ist die Verwendung von AVXP nicht ausreichend, die Verwendung von RTP ist notwendig.

RTP kann in MMC auch zur Kommunikation zwischen MMC und anderen audiovisuellen Konferenzsystemen verwendet werden.

6.2.2 Adaptive Vorwärtsfehlerbehebung

Im folgenden wird dargelegt, wie mittels RTP und RTCP Vorwärtsfehlerbehebung (forward error correction; **FEC**) unterstützt werden kann.

Grundlagen

Vorwärtsfehlerbehebung ist ein Mechanismus, mit dem Übertragungsfehler und Paketverluste kompensiert werden können. Dafür werden zusätzlich zu den Nutzdaten weitere Daten (Redundanz) übertragen, anhand derer eine Kompensation fehlerhafter Übertragung beim Empfänger durchgeführt werden kann. Bis zu welchem Grad einer fehlerhaften Übertragung Nutzdaten rekonstruiert werden können, hängt von der hinzugefügten Redundanz und den Verlusten bzw. Fehlern ab. Der Haupteinsatzbereich der Vorwärtsfehlerbehebung liegt in Anwendungen, bei denen eine Übertragungswiederholung aufgrund der Zeitanforderungen der Anwendung nicht praktikabel ist. Insbesondere bei der interaktiven Übertragung von Audio und Video ist dies der Fall. FEC ist ein Mittel, um die Qualität bei unzuverlässiger Übertragung zu verbessern.

Prinzipiell lassen sich zwei Arten der FEC unterscheiden. Wenn FEC gegen Bitfehler in Paketen eingesetzt wird, dann müssen die fehlerhaften Bits bestimmt und korrigiert werden. Bei FEC gegen Paketverlust ist die Rekonstruktion des verloren gegangenen Pakets erforderlich, die Bestimmung des fehlenden Pakets erfolgt anhand der Sequenznummern der Pakete. Der Paketverlust durch Pufferüberlauf in Zwischensystemen ist heutzutage gravierender als das Auftreten von Bitfehlern. Dies ist unter anderem dadurch begründet, daß die Qualität der Übertragungsmedien steigt (Verwendung von Glasfaser) und beteiligte Zwischensysteme immer stärker aus- und überlastet sind. Insofern ist eine Sicherung von ganzen Paketen, ohne Sicherung von Bitfehlern, sinnvoll. Im Kontext von RTP und AVXP sind Bitfehler von den Protokollen selbst nicht erkennbar, da beide Protokolle keine Prüfsumme besitzen. Fehlerhafte Pakete müßten von den darunterliegenden Protokollen angezeigt werden. Fehlende Pakete können durch Sequenznummer und Zeitstempel erkannt werden. Beim Einsatz von FEC für Paketsicherung muß jedoch erwähnt werden, daß mit diesem Verfahren die Symptome der fehlerhaften Übertragung teilweise nur verstärkt werden, denn durch FEC werden mehr Daten übertragen als ohne Sicherung. Diese Eigenschaft

6.2.1 Interoperabilität

Die folgenden Ausführungen beinhalten Aspekte der Interoperabilität zwischen MMC und weiteren Konferenzsystemen.

Bei der Analyse der Protokolle RTP und AVXP wurde bereits erwähnt, daß RTP von der IETF (Internet Engineering Task Force) spezifiziert wurde, und AVXP einem spezialisierten Protokoll für den Einsatz in MMC entspricht. Das Protokoll RTP wird von mehreren Anwendungen verwendet, in Kapitel 3 wurde eine der Anwendungen, das Network Voice Terminal NeVoT, vorgestellt. Der Einsatz von AVXP ist zum Zeitpunkt der Arbeit auf MMC beschränkt. Aufgrund der für MMC spezialisierten Lösung ist ein Einsatz von AVXP in anderen Anwendungen unwahrscheinlich.

Obwohl es sich bei MMC um ein abgeschlossenes Konferenzsystem handelt, ist die Möglichkeit der Kommunikation mit weiteren Anwendungen wünschenswert. Das folgende Szenario gibt ein Beispiel für diesen Ansatz. Die hier vorgestellten Aspekte beziehen sich ausschließlich auf den Transport von Audio und Video.

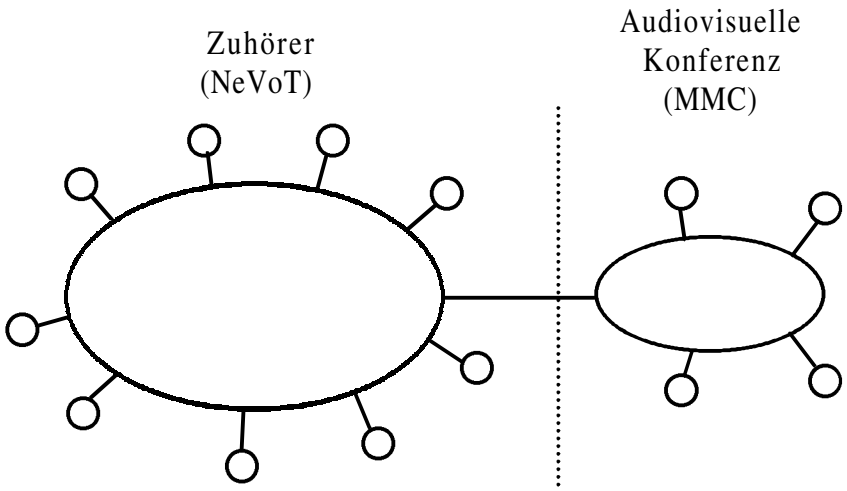


Abbildung 6-1: Interoperabilität von MMC und weiteren Anwendungen

MMC wird größten Teils von industriellen Unternehmen entwickelt. Aus diesem Grund ist es wahrscheinlich, daß die Nutzung des Systems nicht unentgeltlich erfolgen wird. Die Verwendung von MMC wird somit auf Personen mit Nutzungsrechten beschränkt sein. Dies ist bei einem kommerziellen Produkt sinnvoll und erforderlich. Falls aber die während einer Konferenz übermittelten Informationen einer größeren Menge von Personen verfügbar gemacht werden sollen, ist dies hinderlich. Denkbar ist eine audiovisuelle Konferenz unter Verwendung von MMC. Weiterhin wäre die Möglichkeit, nicht zur Benutzung von MMC autorisierten Personen zumindest den Empfang von Audio zu ermöglichen, wünschenswert. Damit könnten Personen, welche an das von MMC verwendete Netzwerk angeschlossen sind, passiv an der Konferenz teilhaben. Dafür ist ein Programm erforderlich, welches die von MMC

eindeutige Einordnung der erhaltenen Daten möglich sein. Dazu dienen die Felder Sequenznummer, Zeitstempel und die Komponentenangabe.

Die Verwendung des Feldes Zeitstempel ist bei AVXP und RTP ähnlich. Bei AVXP ist die Verwendung von Mikrosekunden vorgeschrieben, bei RTP muß die Interpretation des Zeitstempels vor der Verwendung festgelegt werden. Die Verwendung der Sequenznummer unterscheidet sich zwischen AVXP und RTP. Während bei einem aufgeteilten Videobild bei RTP für jedes Teilpaket die Sequenznummer erhöht wird, ist dies bei AVXP nicht der Fall. Für die Unterscheidung der einzelnen Komponenten wird dies durch das Feld Komponentensumme und Komponentennummer erreicht. Diese Vorgehensweise bietet bei AVXP lediglich den Vorteil, daß im ersten zu einer Komponente gehörenden Paket die Gesamtgröße der Komponente übergeben wird (im Feld Offset, vgl. Kapitel 5).

6.1.10 Nutzdatenidentifikation

Bei RTP gibt es ein Feld (PT), welches den Typ der transportierten Nutzdaten identifiziert. Die Werte dieses Feldes müssen durch globale Konventionen festgelegt sein, etwa durch die in Kapitel 4.4 erläuterten Anhangdateien. Bei AVXP ist eine Identifikation des Nutzdatentyps nicht vorgesehen. Nutzdatenidentifikation dient einerseits dem Empfänger zur Unterscheidung verschiedener multimedialer Daten, andererseits ist sie für Zwischensysteme notwendig, wenn Daten gemischt werden. Wie bereits mehrfach erwähnt, ist bei der Übertragung von Audio die Möglichkeit gegeben, Daten mehrerer Sender in Zwischensystemen zu mischen, um bei der weiteren Übertragung zwischen Zwischensystem und den Empfängern weniger Daten zu übertragen. Die Verwendung von 3 Bits für eine Nutzdatenidentifikation in den für Flags reservierten Bytes ist auch mit den oben gemachten Vorschlägen für Flags möglich.

6.1.11 Kontrollinformationen

Der wesentliche Unterschied zwischen AVXP und RTP/RTCP liegt in der Unterstützung von Kontrollinformationen. Die zentrale Systemstruktur von MMC sieht vor, daß alle Kontrollinformationen zwischen den Teilnehmern über zentrale Instanzen, z.B. den AV Manger, laufen. Hier liegt der entscheidende Nachteil von AVXP, es werden keine Kontrollinformationen zwischen den Teilnehmern unterstützt. Bei RTP/RTCP wird Konferenzkontrolle und Überwachung der Datenübertragung unterstützt. Durch das Fehlen solcher Funktionalität bei AVXP sind manche Protokollmechanismen nicht einsetzbar, mit denen gegebenenfalls eine Steigerung der Qualität der Datenübertragung erreichbar wäre. Beispiele für solche Mechanismen werden in Kapitel 6.2 vorgestellt.

6.2 Möglichkeiten

In Kapitel 6.1 wurden die Protokolle AVXP und RTP ausführlich verglichen. Im folgenden werden drei Erweiterungsmöglichkeiten von MMC vorgestellt. Anhand dieser Möglichkeiten wird gezeigt, daß durch die Verwendung von RTP statt AVXP entweder die Qualität der Datenübertragung in MMC oder die Flexibilität von MMC gesteigert werden kann.

Bei AVXP existiert ein Flag (PAUSE), welches einem Empfänger anzeigt, daß für eine gewisse Zeitdauer keine Daten übertragen werden. Somit wird bei AVXP in einem Datenpaket auch Kontroll- bzw. Signalisierungsinformation übertragen. Eine genaue Verwendung dieses Flags ist nicht spezifiziert. Es wird zum Zeitpunkt der Arbeit nicht verwendet. Ursprüngliches Ziel bei der Spezifikation war es, dem Empfänger anzuzeigen, daß für eine nicht spezifizierte Zeit keine Daten empfangen werden müssen. Ungeklärt ist auch, durch welchen Mechanismus ein Empfänger darauf aufmerksam gemacht wird, daß die Datenübertragung fortgesetzt wird.

Bei RTP wird eine unterbrochene Übertragung in anderer Weise unterstützt. Ein Empfänger muß jederzeit für den Empfang von Daten bereit sein. Ein Sender kann, bei entsprechenden Daten, die Übertragung vorübergehend aussetzen. Beim Empfänger ist dann jedoch bei Empfang einer neuen Folge von Paketen eine neue Synchronisation erforderlich. Diese wird durch die Felder Zeitstempel und Sequenznummer sowie das Markierungsbit unterstützt (vgl. Kapitel 2.2 und Kapitel 4).

Der Mechanismus der unterbrochenen Übertragung kann auch von AVXP unterstützt werden. Wie in Kapitel 2.2 gezeigt wurde, ist für die dazugehörige Synchronisation ein Markierungsbit (z.B. BOT = „begin of talkspurt“) oder die Angabe einer Sequenznummer und eines Zeitstempels erforderlich. Zeitstempel und Sequenznummer sind bei AVXP vorhanden, der Platz für ein entsprechendes Flag ist ebenfalls verfügbar.

Ziel der unterbrochenen Übertragung ist es, ohne eine Einbuße der Qualität die Netzwerklast zu reduzieren. Damit ist eine geringere Auslastung der zur Übertragung verwendeten Netzwerke möglich. Insbesondere bei der Audioübertragung zwischen mehreren Teilnehmern ist davon auszugehen, daß nicht alle Teilnehmer gleichzeitig sprechen und somit Pausen bei der Übertragung entstehen.

Bei der unterbrochenen Übertragung muß berücksichtigt werden, daß sich zwar die durchschnittliche Übertragungskapazität über einen längeren Zeitraum reduzieren kann, jedoch nicht die Spitzenwerte der Übertragung. Im Fall einer Übertragung mit CD-Qualität bedeutet dies, daß entweder mit 1411,2 KBits/s gesendet wird oder überhaupt nicht. Bei einer möglichen Ressourcen-Reservierung hätte dies zur Folge, daß 1411,2 KBits/s reserviert werden müßten, um eine Garantie für die Übertragung der Audiodaten zu erhalten. Erfolgt die Abrechnung der Nutzung eines Netzwerks nach maximal genutzter Übertragungskapazität, so spielt es für den Nutzer keine Rolle, ob unterbrochene Übertragung eingesetzt wird. Der Netzbetreiber hingegen könnte daraus Vorteile ziehen und restliche Übertragungskapazität anderen Nutzern anbieten.

6.1.9 Zeitstempel, Sequenznummer und Komponentenangabe

Zeitstempel, Sequenznummer und Komponentenangabe dienen zur eindeutigen Zuordnung eines Datenpakets zu einer Folge von Daten, etwa einem Audio- oder einem Videostrom. Wie bereits mehrfach erwähnt, werden Videobilder oftmals unterteilt und in mehreren Paketen übertragen. Bei der Rekonstruktion einer Videosequenz beim Empfänger muß dann eine

Bei AVXP ist die Verschlüsselung von Daten nicht explizit vorgesehen. Eine Erweiterung dahingehend wäre jedoch, wie im Fall der Versionsnummer, wünschenswert. Für eine solche Unterstützung sind wie im Fall von RTP wenige Schritte erforderlich. Die für Flags reservierten zwei Bytes bieten Platz für die Verwendung eines Flags zur Identifikation der Verwendung von Füllbytes (Padding-Flag). Für die Längenangabe der Füllbytes kann ebenfalls das letzte Füllbyte verwendet werden.

Unabhängig von der Unterstützung der Verschlüsselung durch Füllbytes stellt sich die Frage, ob lediglich die Nutzdaten oder das gesamte Paket verschlüsselt werden soll. Bei einer Verschlüsselung des gesamten Pakets sind Zwischensysteme nicht in der Lage die Daten zu interpretieren, falls sie nicht im Besitz des Schlüssels sind. Ein Mixen von Daten, welches insbesondere bei der Übertragung von Audio sinnvoll ist, ist damit nicht möglich. Jedoch wird durch dieses Verfahren auch vermieden, daß nicht-autorisierte Personen über im Paketkopf befindliche Informationen verfügen oder auf Pakete antworten können.

6.1.7 Offset

Ein 4 Bytes großes Feld bei AVXP dient zur Angabe des Offsets der Daten. Werden zusammengehörende Daten in mehreren Paketen verschickt, so müssen diese beim Empfänger zusammengesetzt werden. Der Offset gibt an, an welcher Stelle die Daten eingefügt werden müssen. Insbesondere bei Videobildern ist dies relevant, da Videobilder i.d.R. fragmentiert übertragen werden. Bei unzuverlässiger Übertragung kann durch Paketverlust oder -verzögerung eine Lücke bei den empfangenen Daten entstehen. In diesem Fall stellt sich die Frage, ob unvollständige Daten bei einem Empfänger überhaupt einen Sinn ergeben. Bei Audiodaten ist eine durch unvollständige Daten entstehende Lücke in der Sprachausgabe sicherlich vertretbar, bei Video hängt die Verwendbarkeit der Daten von der Kodierung ab.

Bei RTP ist ein Offset-Feld nicht im Standard-Paketkopf festgelegt. Die Erweiterbarkeit von RTP bzw. eine weitergehende Spezifikation von Paketen mittels Anhangdateien ermöglichen jedoch ein entsprechendes Feld ohne weiteres. Bei RTP existieren mehrere Anhangdateien, welche die Anordnung von Daten bei der Übertragung von Video festlegen (Vgl. Kapitel 4.4).

6.1.8 Unterbrochene Übertragung

Sowohl bei RTP als auch bei AVXP ist die Unterstützung von Übertragungspausen vorgesehen, es werden jedoch unterschiedliche Ansätze unterstützt. Übertragungspausen entstehen, wenn zwischen Sender und Empfänger während einer Anwendung (beispielsweise einer Konferenz) keine Daten übertragen werden. Dies kann durch automatische Erkennung erfolgen, bei der die zu übertragenden Daten untersucht werden, ob sie relevante Informationen beinhalten. Irrelevante Informationen werden nicht übertragen. Weiterhin kann mittels Kontrollinformation dem Sender angezeigt werden, daß für eine gewisse Zeit keine Daten übertragen werden.

6.1.4 Paketlängenangabe

Der Paketkopf von AVXP beinhaltet ein Feld zur Angabe der Nutzdatenlänge. Bei RTP ist ein solches Feld im Gegensatz zum Kontrollprotokoll RTCP nicht vorhanden. Bei RTCP dient die Längenangabe zur Identifikation beim Transport mehrerer RTCP-Pakete in einem Paket des Kommunikationssubsystems. Bei RTP ist es somit erforderlich, daß bei der Übergabe eines Pakets vom Kommunikationssubsystem an RTP die Paketlänge übergeben wird, ebenso wenn mehrere RTCP-Pakete in einem Paket des Kommunikationssubsystems übertragen werden. Die Übergabe der Paketlänge vom Kommunikationssubsystem ist jedoch ohne weiteres möglich.

AVXP besitzt ein 4 Bytes großes Feld zur Längenangabe des Pakets. Dieses Feld ist erforderlich, da in MMC teilweise nicht alle Nutzdaten eines Pakets interpretiert werden. Bei einer Verwendung von RTP statt AVXP muß entweder über die Verwendung einer Paketkopferweiterung eine Längenangabe des Pakets hinzugefügt werden oder über den Einsatz des Füllbytes-Mechanismus angezeigt werden, daß nicht alle Daten zu interpretieren sind.

6.1.5 Variabler Paketkopf

Bei RTP existiert ein Flag, welches eine Erweiterung des Paketkopfs anzeigt. In diesem Fall ist dem standardmäßigen Paketkopf eine (anwendungsspezifische) Erweiterung angehängt. Bei AVXP existiert keine vergleichbare Möglichkeit. Auch bei der Verwendung des oben erläuterten Feldes für eine Versionsidentifikation wäre noch ausreichend Platz für ein entsprechendes Flag im Paketkopf.

Bei RTP ist ein solcher Mechanismus sicherlich sinnvoll, da das Protokoll für eine Vielzahl unterschiedlicher Anwendungen konzipiert wurde, deren Anforderungen bei der Spezifikation nicht alle erfüllt werden können. Bei AVXP hingegen ist der Einsatzbereich auf MMC eingeschränkt, da es sich um eine spezialisierte Lösung handelt. Insofern ist eine Erweiterbarkeit des Paketkopfs bei AVXP nur bedingt sinnvoll, da den Anforderungen von MMC voll entsprochen werden kann.

6.1.6 Verschlüsselung und Füllbytes

Die Verschlüsselung von Paketen eines Protokolls kann entweder über die Verschlüsselung der Nutzdaten oder des gesamten Pakets erfolgen. Bei RTP ist eine Verschlüsselung des gesamten Nutzdatenpakets bzw. mehrerer Kontrollpakete als eine Einheit vorgesehen. Als unterstützender Mechanismus wird die Verwendung von Füllbytes ermöglicht. Füllbytes sind Daten, die den Nutzdaten angehängt werden, jedoch nicht interpretiert werden. Sie können bei der Verschlüsselung erforderlich werden, wenn Algorithmen lediglich auf ein Vielfaches einer gegebenen Bytegröße ausgelegt sind. Dann müssen die Pakete entsprechend angepaßt werden. Bei RTP zeigt ein Flag im Paketkopf an, ob Füllbytes den Nutzdaten angehängt sind. Das letzte Oktet der Daten beinhaltet die Länge der Füllbytes, welche bei der Interpretation der Daten ignoriert werden müssen. Die Verteilung des Schlüssels ist durch RTP nicht geregelt und muß durch externe Mechanismen vollzogen werden.

Internetadresse und einem Endpunktidentifikator. Dem steht die mangelnde Transparenz und der entscheidende Nachteil, daß bei zufälliger Auswahl der Werte zwei Teilnehmer einer Konferenz den gleichen Wert wählen können, gegenüber. Dieser Fall muß zunächst von einem Teilnehmer erkannt werden (z.B. durch den Empfang von PDUs, welche als Absender die eigene „Adresse“ besitzen), darüber hinaus muß eine Fehlerbehandlung erfolgen. Dabei ist ein Zurückziehen des Senders mit Wechsel der eigenen Adresse eine Möglichkeit. Beide Schritte sind unabhängig von der Wahrscheinlichkeit für das Eintreten einer Kollision erforderlich, wenn fehlerhafte Übertragung vermieden werden soll. Im folgenden wird ein Überblick über die auftretenden Wahrscheinlichkeiten bei einem 4 Bytes großen Identifikator gegeben.

Bezeichnet N die Anzahl der Teilnehmer und L die Bitlänge des Identifikators, so gilt für die Wahrscheinlichkeit p, daß bei N Teilnehmern zwei Teilnehmer zufällig den gleichen Wert verwenden [Fell60; Kapitel II.3] (für große N) ungefähr:

$$p = 1 - e^{\frac{-N(N-1)}{2^{(L+1)}}}$$

In Tabelle 6-1 sind mehrere Wahrscheinlichkeitswerte für den Fall L = 32, in Tabelle 6-2 mehrere Wahrscheinlichkeitswerte für den Fall L = 16, dargestellt.

Anzahl der Teilnehmer	**Kollisionswahrscheinlichkeit (in %)**
50	0,00003
100	0,00012
500	0,0029
1000	0,012

Tabelle 6-1: Kollisionswahrscheinlichkeit für 32-Bits Adreß-Identifikatoren in %

Anzahl der Teilnehmer	**Kollisionswahrscheinlichkeit (in %)**
50	1,85
100	7,27
500	14,9
1000	99,5

Tabelle 6-2: Kollisionswahrscheinlichkeit für 16-Bits Adreß-Identifikatoren in %

Die Ergebnisse zeigen, daß ein Adreß-Identifikator der Länge 4 Bytes ausreichend ist, um Kollisionen nur in sehr seltenen Fällen auftreten zu lassen, bei Adreß-Identifikatoren der Länge 16 Bytes ist mit erhöhtem Aufwand aufgrund von Kollisionen zu rechnen.

Die Möglichkeit der sinnvollen Weiterentwicklung eines Protokolls sollte in jedem Fall unterstützt werden. Dies gilt insbesondere für AVXP, da mehrere Unternehmen an der Entwicklung von MMC arbeiten, welches AVXP verwendet. Folgendes Beispiel dient zur Verdeutlichung: Bei AVXP wird als Quelladresse die vier Bytes große Internetadresse angegeben. Zum Zeitpunkt der Arbeit ist bereits abzusehen, daß aufgrund der Aufteilung der Adressen diese in absehbarer Zukunft erschöpft sein werden und andere (längere) Adressen verwendet werden. IP in der Version 6 verwendet Internetadressen mit einer Länge von 16 Bytes, so daß ggf. eine Anpassung von AVXP und somit eine neue Version erforderlich wird.

Der Aufbau der PDU bei AVXP zeigt, daß ohne weiteres Platz für eine Versionskennung vorhanden ist. Für Zustandsbits **(Flags)** stehen bei AVXP 16 Bits zur Verfügung, von denen zum Zeitpunkt der Arbeit lediglich zwei belegt sind. An dieser Stelle wäre die Verwendung von zwei oder drei Bits als Versionsnummer ohne weiteres denkbar und wünschenswert.

6.1.3 Adreßinformation

Weder AVXP noch RTP verwenden in ihrer PDU eine Zieladresse. Diese Information ist auch nicht erforderlich, da sie von der Anwendung direkt an das Protokoll gegeben werden kann, welches RTP- bzw. AVXP-Pakete überträgt.

Als Absenderinformation werden unterschiedliche Formate verwendet. AVXP verwendet die Internetadresse (4 Bytes) und den Endpunktidentifikator (2 Bytes). RTP verwendet einen zufälligen Adreß-Identifikator der Größe 4 Bytes, der eindeutig einem Endsystem und einem Endpunkt entspricht. Es existiert somit eine bijektive Abbildung zwischen den Adreß-Identifikatoren und allen an einer verteilten Anwendung beteiligten Teilnehmer bei RTP. Die Verwendung von weiteren Absenderinformationen bei RTP ist für die Unterstützung von Zwischensystemen erforderlich (vgl. Kapitel 4.1).

Eine Internetadresse und einen Endpunkt als Adreßinformation zu verwenden, bietet mehr Transparenz als ein zufällig gewählter Wert. Bei End- oder Zwischensystemen kann aus dieser Information der Empfänger direkt (ggf. für Rückinformation) ermittelt werden, wogegen bei einem allgemeinen Identifikator über einen anderen Mechanismus beim Empfänger eines Pakets eine Zuordnung zwischen dem Identifikator und dem eigentlichen Absender erfolgen muß. Bei RTP erfolgt dies über das dazugehörige Kontrollprotokoll RTCP (mittels der Quellenbeschreibung SDES), bei AVXP müßte der Informationsfluß über den zentralen AV Manger laufen, da AVXP ein reines Datenprotokoll ist.

Bei einem Wechsel von 4-Bytes Internetadressen nach 16-Bytes Internetadressen ist bei einem allgemeinen Adreß-Identifikator (RTP) keine Änderung des Paketkopfs erforderlich, jedoch eine Änderung der Kontrollinformation, welche dem Identifikator eine Internetadresse zuordnet.

Die Verwendung eines zufälligen Identifikators hat den Vorteil, daß die Anzahl der erforderlichen Bytes bei entsprechender Wahl (4 Bytes; siehe unten) geringer ist als bei einer

Protokolldateneinheiten

Die Gesamtlänge des Paketkopfs beträgt bei RTP mindestens zwölf Bytes, bei AVXP 26 Bytes. In dieser Weise sind die Paketköpfe jedoch nicht vergleichbar, da sich der Paketkopf bei RTP durch die Hinzunahme mehrerer Quellenangaben im Paketkopf durch Zwischensysteme und eine Paketkopferweiterung vergrößern kann. Die Verwendung der einzelnen Felder ist vielmehr im einzelnen zu prüfen, dies erfolgt in den folgenden Unterkapiteln. Neben der allgemeinen Unterscheidung der Felder besitzt die allgemeine Entwurfsentscheidung feste oder variable Länge des Paketkopfs eine wesentliche Auswirkung. Eine variable Paketkopflänge ist zwar flexibler, jedoch erfordert die Bearbeitung des Paketkopfs einen höheren Aufwand. Eine feste Länge des Paketkopfs eignet sich für die Realisierung in Hardware weit besser als eine variable Länge.

Beide Protokolle haben gemeinsam, daß die Länge der Nutzdaten den Anforderungen der Anwendungen entspricht (**Application Level Framing**). Lediglich bei der Überschreitung der vom Kommunikationssubsystem abhängigen maximalen Paketgröße ist eine Aufteilung der Daten in mehrere Pakete erforderlich. Die Aufteilung der Daten gemäß den Anforderungen der Anwendung unterstützt einen effizienten Programmablauf, da zwischen Protokoll und Anwendung normalerweise keine Zwischenbearbeitung (z.B. verändern der Paketgröße) erforderlich ist. Ein konkretes Beispiel für Application Level Framing bei AVXP und RTP ist der Transport von Audiopaketen einer gewissen Zeitdauer.

Die Byteanordnung entspricht bei RTP und AVXP der „network-byte-order“, d.h. bei einer Datenlänge von mehr als einem Byte wird das höchstwertige Byte als erstes im Speicher abgelegt bzw. übertragen.

Beide Protokolle unterscheiden sich in der unterstützten Protokollfunktionalität. Der wesentliche Vorteil von RTP liegt im dazugehörigen Kontrollprotokoll und in der Unterstützung eines flexiblen Aufbaus der Pakete. Im folgenden werden Unterschiede bezüglich der Protokolldateneinheiten **(PDUs)** und die dadurch entstehenden Konsequenzen erläutert. Es werden ähnliche Felder der Protokolldateneinheiten verglichen.

6.1.2 Versionsidentifikation

RTP verwendet zwei Bits seines Paketkopfs als Identifikator für die verwendete Version des Protokolls. Bei AVXP ist kein entsprechendes Feld vorhanden. Der Vorteil einer vorhandenen Versionsnummer ist plausibel. Wenn Änderungen am Protokoll vorgenommen werden, kann ein End- bzw. Zwischensystem die Versionsnummer erkennen und die Pakete entsprechend interpretieren oder ignorieren, falls die Version nicht unterstützt wird. Ist keine Identifikation der Version möglich, so müssen alle Benutzer des Protokolls zu einem festen Zeitpunkt auf eine neue Version wechseln. Ein nicht erkannter Versionswechsel führt unter Umständen zu fehlerhafter Interpretation von Paketen.

6 Erweiterung von MMC um RTP

In diesem Kapitel wird die konzeptionelle Erweiterung von MMC um RTP vorgestellt. Zunächst erfolgt ein detaillierter Vergleich der Protokolle AVXP und RTP. Daran anschließend werden Verbesserungsvorschläge für eine Steigerung der Flexibilität von MMC und der Qualität der Datenübertragung in MMC gegeben.

6.1 Vergleich von RTP und AVXP

Die Protokolle RTP [SCFJ94] und AVXP [ADH&93] wurden bereits in den Kapiteln 4 und 5 vorgestellt, ebenso wie in den Kapiteln 3 und 5 die Anwendungen NeVoT und MMC, welche diese Protokolle verwenden. In diesem Kapitel folgt ein Vergleich der beiden Protokolle und deren Einsatzmöglichkeiten.

6.1.1 Grundlagen

Die Protokolle RTP und AVXP unterscheiden sich deutlich bezüglich ihres Entwicklungshintergrunds. RTP wurde als Transportprotokoll für unterschiedliche verteilte multimediale Anwendungen entwickelt, AVXP für eine bestimmte Anwendung (MMC). Demzufolge waren und sind bei den Entwurfsentscheidungen von RTP mehrere Aspekte gegeneinander aufzuwiegen, die in der Regel nicht einer Optimallösung für jede einzelne Anwendung entsprechen. Das Protokoll AVXP hat deutlich geringeren Funktionsumfang, was bereits aus einem ersten Vergleich der beiden Spezifikationen hervorgeht. Zu AVXP gibt es kein entsprechendes Kontrollprotokoll, dies schränkt die Einsatzmöglichkeiten von AVXP ein (siehe Kapitel 6.2). Über die zentrale Organisation des Konferenzsystems MMC stehen den Teilnehmern zwar Kontrollinformationen über partizipierende Teilnehmer und über die Konferenz zur Verfügung, aber die Überwachung der Qualität der Datenübertragung (quality of service monitoring) ist nicht möglich. Dieser Nachteil wird sich im folgenden als besonders hinderlich für die Erbringung gewisser Protokollfunktionalität zeigen.

In diesem Kapitel bezeichnet der Begriff **Kommunikationssubsystem** Protokolle der Schichten drei und vier, die ihre Dienste AVXP und RTP zur Verfügung stellen. Bei beiden Protokollen wird zum Zeitpunkt der Arbeit UDP/IP verwendet. Erweiterungen sind vorgesehen. Bei der Entwicklung von AVXP bestand die Planung, die Protokollfamilie des Projekts MMT (Multimedia Transport Service) zu verwenden. Bei MMT werden XTP/ST-II verwendet. Bei RTP stehen neben UDP/IP mehrere Alternativen zur Verfügung (vgl. Kapitel 4). Die Qualität der Übertragung von Daten mittels AVXP und RTP ist u.a. abhängig von der Wahl der darunterliegenden Protokolle.

Die **Internetadresse der Quelle** (4 Bytes) gibt die Adresse des Senders an, der **Endpunkt der Quelle** (2 Bytes) gibt den eindeutigen Identifikator innerhalb des Systems an. Der Endpunkt der Quelle entspricht dem Wert, der mittels SSCP vom AVM an die AVC im Rahmen des Öffnens einer Quelle übergeben wird.

Die **Sequenznummer** (4 Bytes) gibt die Nummer des Pakets innerhalb eines Datenstromes an. Sie dient zusammen mit dem Zeitstempel und den Komponentenangaben zum Erkennen von verlorengegangenen oder verspäteten Paketen, wonach je nach verwendetem Kommunikationssystem (UDP/IP oder ST-II/XTP) entweder eine Fehlerbehandlung eingeleitet wird oder der Fehler ignoriert wird. Bei der Übertragung von Videobildern erhöht sich die Sequenznummer mit jedem neuen Bild, nicht mit jedem Paket. Zur Unterscheidung von einzelnen Komponenten eines Bildes wird das nächste Feld, die Komponenten-Information, verwendet.

Die Komponenten-Information besteht aus der **Komponentennummer** (1 Byte) und der **Komponentensumme** (1 Byte). Mit den Komponenten werden die im letzten Abschnitt erwähnten Teilbilder eines Videobildes unterschieden. Bei N Komponenten ist im ersten Feld ein Element der Menge {1, .. , N} gespeichert, im zweiten Feld der Wert N.

Der **Offset** (4 Bytes) gibt an, an welche Position die übertragenen Daten gehören. Insbesondere bei der Übertragung von Videobildern ist dieser Mechanismus erforderlich, wenn Teilbilder einer Videosequenz aufgeteilt werden. Da beim ersten Paket der Offset immer null ist, wird an dieser Stelle die Gesamtgröße des zu einer Sequenznummer gehörenden Gesamtbildes eingetragen.

Der **Zeitstempel** (4 Bytes) dient als Ergänzung oder Alternative zur Sequenznummer. Der Zeitstempel wird in Millisekunden relativ zu einem zufälligen Anfangspunkt gerechnet und bezeichnet einen Zeitpunkt zu dem die Daten gehören. Im Falle von Synchronisation muß beim Empfänger dieser Zeitstempel für die Ausgabe einbezogen werden.

Bei AVXP sind Markierungsbits **(Flags)** definiert. Zum Zeitpunkt der Arbeit existiert die Spezifikation von zwei Flags:

- **EOS** (End of Stream). Das Flag zeigt an, daß das Ende der Übertragung des Datenstroms erreicht ist.
- **PAUSE**. Das Flag zeigt an, daß für eine gewisse Zeit keine Daten übertragen werden.

Die **Nutzdatenlänge** (4 Bytes) gibt die Anzahl der Nutzdaten-Bytes des Pakets an.

Die Felder Komponentennummer, Komponentensumme, Offset, Zeitstempel und Flags sind optional. Bei AVXP bedeutet dies, daß die Daten zwar verschickt, nicht jedoch interpretiert werden müssen. Alle aufgeführten Felder beinhalten in diesem Fall den Wert null.

Der Vergleich von AVXP und RTP sowie eine Bewertung der Protokolle ist Bestandteil von Kapitel 6.

wird. Hier wird zunächst der Aufbau von AVXP beschrieben, Erläuterungen und Bewertungen hierzu folgen beim Vergleich der beiden Protokolle in Kapitel 6.

AVXP ist ein reines Datenprotokoll, es besitzt gemäß dem Stand vom Mai 1995 keinen Kontrollteil. Dabei muß allerdings berücksichtigt werden, daß über den in Kapitel 5.2 vorgestellten AV Manager ein Kontrollfluß (z.B. aushandeln der Übertragungsqualität) stattfindet. AVXP dient zur Kommunikation zwischen den ebenfalls bereits vorgestellten AV Komponenten, welche die jeweiligen Endpunkte einer Verbindung bei jedem Teilnehmer bilden. Der Umfang von AVXP ist sehr gering, es ist lediglich der Aufbau einer Protokolldateneinheit festgelegt, welche im folgenden Kapitel dargestellt wird.

AVXP verwendet als Subsystem entweder UDP/IP oder XTP/ST-II, wobei UDP/IP zum Zeitpunkt der Arbeit realisiert ist.

5.3.2 Protokolldateneinheiten

Die Protokolldateneinheiten sind für den Transport von Audio und/oder Video gedacht und besitzt eine variable Länge. Abbildung 5-2 zeigt den Aufbau einer Protokolldateneinheit, die einzelnen Komponenten werden im folgenden erläutert.

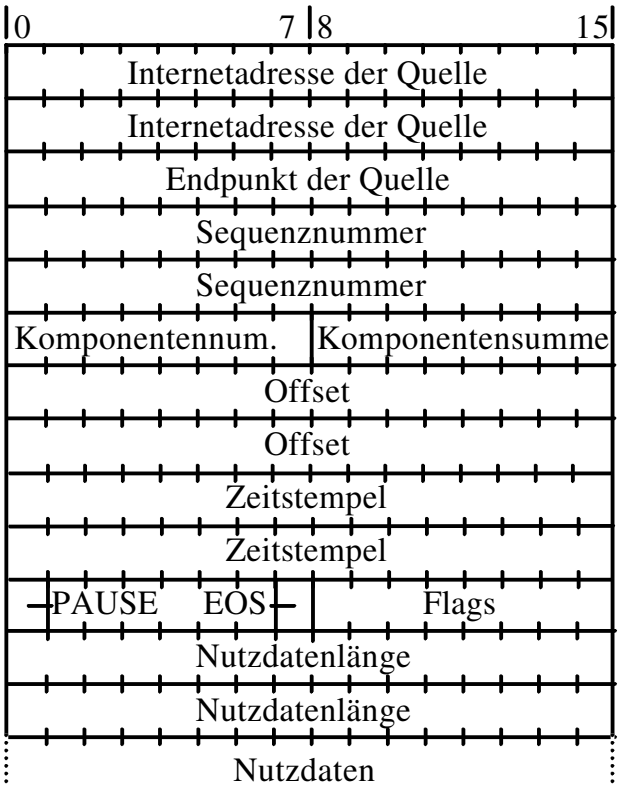


Abbildung 5-2: AVXP Protokolldateneinheit

In beiden Fällen wird die Eingabe der Daten über Token gesteuert, falls nicht ein „Mixing"-Modus (nur in Fall eins) gewählt wird, bei dem alle Teilnehmer zur Eingabe berechtigt sind. Das AVCP *(Audiovisual Control Protocol)* stellt Primitive zur Verwaltung der Verbindungsgruppen zur Verfügung. Dabei werden Anfragen vom CM an den AVM gerichtet, der die entsprechenden Funktionen durchführt. Bei der Etablierung der Verbindungen können unterschiedliche Politiken bezüglich der gewünschten Qualität eingesetzt werden. Entweder die Verbindungen werden nur dann eingerichtet falls eine vorgeschriebene Qualität erreicht wird oder die Verbindungen richten sich nach dem Teilnehmer mit der geringsten Qualität oder für jede einzelne Verbindung wird die optimale Qualität ermittelt.

AVC und SSCP
Die audiovisuelle Komponenten AVC verwaltet die Endpunkte der unterschiedlichen Verbindungen. Jeder Teilnehmer besitzt seine eigene AVC. Mittels *SSCP (Source and Sink Control Protocol)* werden Aufträge vom AVM an die AVC gesendet und Ereignisse vom AVC an den AVM angezeigt. Zu den Aufträgen gehört u.a. das Öffnen oder Schließen einer Datenquelle oder Senke, das Hinzufügen von Empfängern zu einer Datenquelle (*SSCP_AddSinks*) und die Abfrage von Parametern (*SSCP_ListEndpointAVTypes*), anhand derer die beim AVM erläuterte Politik der Qualitätsauswahl durchgeführt werden kann. Zu den Ereignissen von der AVC zum AVM gehört das Schließen des Endpunkts und das Löschen eines Empfängers aus der Empfängerliste.

ASC und ASCP
Die Application Sharing Komponente ASC verwaltet die gemeinsame Nutzung einer Anwendung durch mehrere Teilnehmer. Dabei wird, wie bereits oben erwähnt, die Ausgabe einer Anwendung an alle Teilnehmer geleitet. Die Eingabe zu dieser Anwendung kann von jedem Teilnehmer kommen, falls er im Besitz des Tokens ist. Die Funktionalität ist unabhängig von der Anwendung, d.h. es können alle bei einem Teilnehmer verfügbaren Anwendungen verwendet werden. Zur Kommunikation zwischen ASC und CM existiert das *ASCP (Application Sharing Control Protocol)*. Es beinhaltet Anfragen vom CM an die ASC und Ereignisse von der ASC, die an den CM gemeldet werden. Zu den Anfragen gehört u.a. das Hinzufügen und Löschen von Benutzern einer geteilten Anwendung, das Verwalten der Kontrolle einer geteilten Anwendung und die Verwaltung von Telepointern.

5.3 AVXP - Transport von Audio und Video

Im vorliegenden Kapitel wird das Protokoll AVXP vorgestellt, welches bei MMC als Protokoll für den Transport von Audio und Video zwischen Teilnehmern verwendet wird.

5.3.1 Einleitung

Das Protokoll AVXP (Audio Video Exchange Protocol) ist ein Protokoll, welches speziell für die Datenübertragung in MMC spezifiziert wurde. Hieraus wird bereits deutlich, daß es sich von einem allgemeinen Protokoll zur Übertragung von Audio und Video, wie z.B. das bereits vorgestellte Protokoll RTP, z.T. deutlich unterscheidet, wie dies im Kapitel 6 auch gezeigt

5.2.2 Komponenten

CIA
Der CIA bildet die graphische Benutzerschnittstelle eines Teilnehmers zum Konferenzsystem. Er greift auf das Konferenzverzeichnis zu um Informationen über Teilnehmer zu erhalten. Anfragen bezüglich des Konferenz-Managements werden an den Konferenzmanger CM geleitet. Im Gegenzug wird die vom CM erhaltene Statusinformation über Konferenzen dargestellt.

CD und CDAP
Das Konferenzverzeichnis wurde bereits in Kapitel 5.1.2 erläutert. Auf das CD wird mittels des Protokolls *CDAP (Conference Directory Access Protocol)* zugegriffen. Der Konferenzmanager CM und der CIA greifen mittels dieses Protokolls auf das CD zu. Ein Benutzer muß für den Zugriff entsprechend autorisiert sein. Es gibt drei verschiedene Typen von Benutzern. Ein CD Verwalter darf Benutzer hinzufügen und löschen. Ein Benutzer darf Gruppen einrichten, womit er automatisch Gruppenverwalter wird. Als Gruppenverwalter darf er Gruppeneinträge einrichten bzw. löschen. Der Konferenzmanager CM ist autorisiert, Einträge bezüglich Konferenzen einzutragen bzw. zu löschen.

CM und CMAP
Das Konferenzmanagement wurde ebenfalls in Kapitel 5.1.2 bereits vorgestellt. Die vorgestellte Funktionalität wird vom Konferenzmanager CM erbracht. Er verwaltet dynamische Informationen einer Konferenz. Auf den CM wird mittels des Protokolls *CMAP (Conference Manager Access Protocol)* zugegriffen. Dafür gibt es Primitive zum Öffnen/Schließen einer Konferenz, zum Verwalten einer Chairperson einer Konferenz, zum Hinzufügen und Entfernen von Teilnehmern und für die Tokenverwaltung.

IB und IBAP
Der Einladungsverwalter IB erhält Aufträge vom Konferenzmanager zur Einladung von Teilnehmern mittels des Protokolls *IBAP (Information Broker Access Protocol)*. Der IB gibt Einladungen mittels eines privaten Protokolls an die Teilnehmer weiter. Es existiert ebenfalls ein Primitiv zum Widerrufen einer ausgesprochenen Einladung.

AVM und AVCP
Der AV Manager AVM bietet den Dienst zur Verfügung, mittels audiovisueller Daten zu kommunizieren. Der AVM ist die zentrale Komponente für diese Verwaltung innerhalb einer Konferenz. Zwei unterschiedliche Arten von Verbindungen können etabliert werden.

- Eine Vollvermaschung zwischen allen Teilnehmern zur Übertragung der audiovisuellen Daten der Teilnehmer. Diese werden mittels Mikrophon und Kamera aufgenommen und dann übertragen.
- Eine 1:N Verbindung zwischen der geteilten Anwendung und allen Teilnehmern. In diesem Fall wird die audiovisuelle Ausgabe der Anwendung an alle Teilnehmer geleitet, ein im Besitz des Tokens befindlicher Teilnehmer kann audiovisuelle Daten in die Anwendung eingeben.

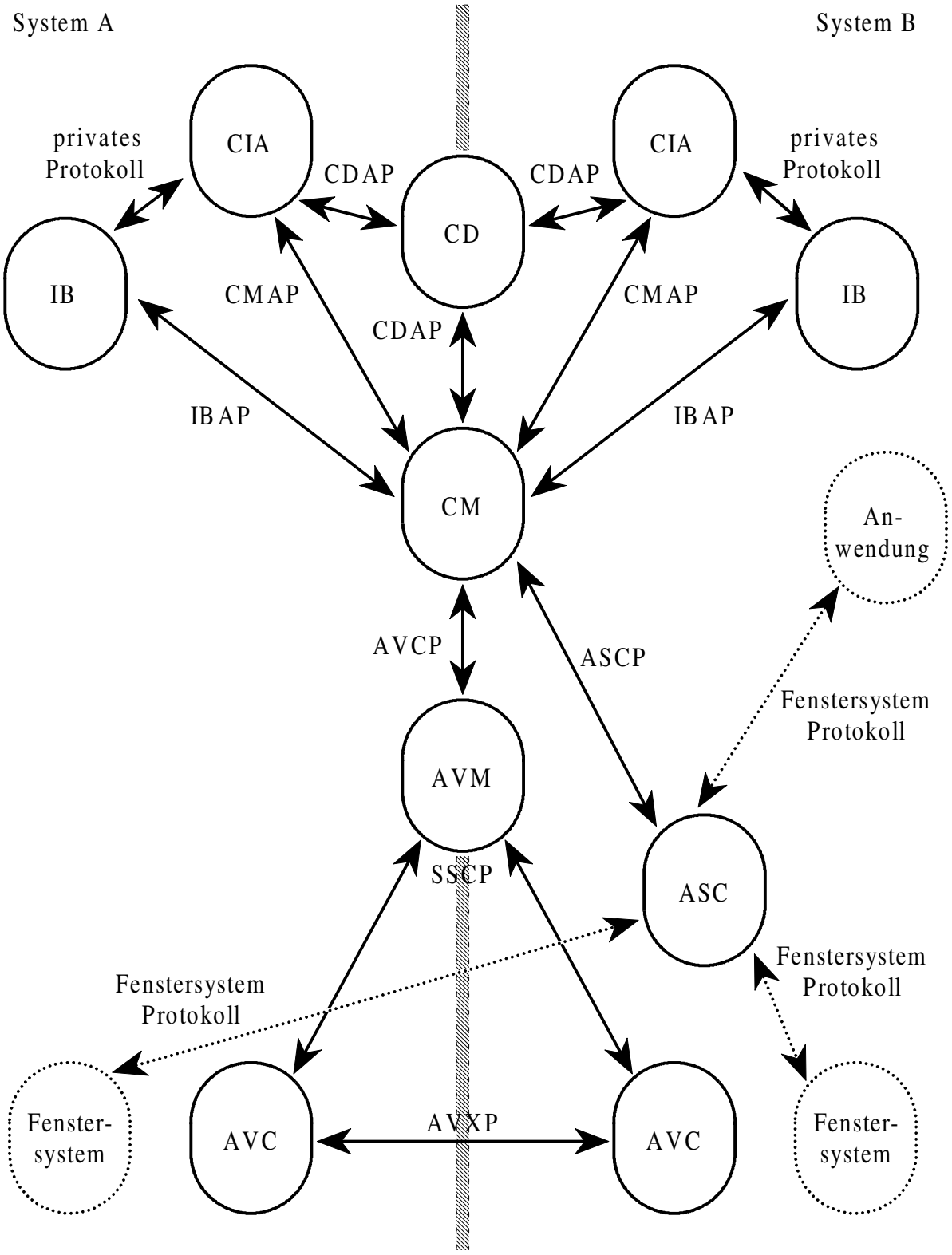


Abbildung 5-1: MMC Systemstruktur

Teilnehmer. Bei der audiovisuellen Kommunikation werden zwischen allen Teilnehmern Verbindungen aufgebaut, über die Audio und Video zwischen Teilnehmern übertragen werden.

5.2 Systemstruktur

Die Erläuterung der Systemstruktur und die Beschreibung der beteiligten Komponenten sind Bestandteil der folgenden Unterkapitel.

5.2.1 Aufbau

Nachdem im letzten Kapitel ein Überblick über MMC gegeben wurde, erfolgt in diesem Kapitel eine Erläuterung der für das Erbringen der Dienste erforderlichen Komponenten sowie der dazugehörigen Protokolle. Zur Verdeutlichung dient Abbildung 5-1, in der die Komponenten und die Protokolle, die zu deren Kommunikation erforderlich sind, dargestellt sind. Komponenten, die einmal pro Konferenz bzw. pro System existieren sind, auf der Mittelachse eingezeichnet; Komponenten, welche spezifisch für einen Teilnehmer sind, befinden rechts und links davon. Die Tabelle 5-1 gibt einen Kurzüberblick über die im folgenden vorgestellten beteiligten Komponenten. Die dazugehörigen Protokolle werden mit den Komponenten zusammen vorgestellt. Eine detaillierte Aufstellung aller Protokollprimitive befindet sich in [ADH&93].

Komponente	Kurzbeschreibung
Konferenzmanager (Conference Manager - **CM**)	Der Konferenzmanager verwaltet Konferenzen, es gibt mindestens einen Konferenzmanager im System.
Konferenzverzeichnis (Conference Directory - **CD**)	Im Konferenzverzeichnis sind statische Daten über Teilnehmer und Gruppen gespeichert.
Konferenzschnittstelle (Conference Interface Agent - **CIA**)	Die Konferenzschnittstelle bildet die Benutzer-schnittstelle für jeden Teilnehmer.
Einladungsverwalter (Invitation Broker - **IB**)	Der Einladungsverwalter empfängt Einladungen für Konferenzen und leitet diese weiter.
Application Sharing Komponente (Application Sharing Component - **ASC**)	Die Application Sharing Komponente ermöglicht die Ausgabe und (nicht gleichzeitige) Eingabe einer Anwendung für mehrere Teilnehmer.
AV Manager (Audiovisual Manager - **AVM**)	Der AV Manager koordiniert den Datenaustausch zwischen Teilnehmern.
AV Komponente (Audiovisual Component - **AVC**)	Die AV Komponente kontrolliert den Daten-austausch eines Teilnehmers.

Tabelle 5-1: MMC Komponenten

- Die Verwendung von internationalen Standards (z.B. ISO oder CCITT) zwecks Kompatibilität und
- die Gliederung des Systems in genau definierte Komponenten und Schnittstellen um einen Teledienst zu realisieren.

5.1.2 Funktionsumfang

Im folgenden wird in einem Überblick der Funktionalität skizziert, die detaillierte Betrachtung der einzelnen Komponenten von MMC ist Bestandteil von Kapitel 5.2.

Das Konferenzsystem bietet folgende Funktionalität:
- Verwaltung eines Konferenzverzeichnisses,
- Konferenzmanagement,
- geteilte Anwendungen **(Application Sharing)** und
- audiovisuelle Kommunikation.

Das Konferenzverzeichnis beinhaltet Informationen über potentielle Benutzer einer Konferenz. Benutzer werden einer Gruppe zugeordnet, welche maßgeblich für den Zugang zu einer Konferenz ist. Darüber hinaus beinhaltet das Verzeichnis Informationen über die Rolle, die ein Mitglied erhalten kann. Es gibt drei verschiedene Rollen: Ein Vorsitzender (Chairperson) hat das Recht, eine Konferenz zu initiieren und zu beenden. Ein Teilnehmer, der im Besitz einer Berechtigungsmarke ist (Token holder), darf privilegierte Funktionen durchführen (vgl. nächster Abschnitt). Sonst ist ein Konferenzmitglied lediglich ein Beobachter (Observer). Informationen bezüglich Konferenzen sind statisch, d.h. sie werden im Laufe einer Konferenz nicht aktualisiert. Demnach ist eine ständig aktualisierte Liste aller Teilnehmer nicht abrufbar. Allgemeine Informationen über Konferenzen sind jedoch verfügbar, anhand dieser können sich Teilnehmer über die Art einer Konferenz informieren.

Das Konferenzmanagement überwacht bestehende Konferenzen und verwaltet die Token. Ein Token erlaubt das Starten und Kontrollieren einer geteilten Anwendung, ein weiteres Token existiert zur Berechtigung der Eingabe von Audio-/Videodaten in eine Konferenz. Für die Verwaltung von Token gibt es zwei Verfahren: Entweder muß es explizit angefordert und zugewiesen werden oder es wird implizit durch den Versuch einer Eingabe angefordert. Die mittels Konferenzmanagement verwalteten Informationen sind dynamisch, d.h. sie werden während einer Konferenz angepaßt.

Beim Application Sharing wird eine Anwendung von einem Benutzer gestartet. Die Ausgabe dieser Anwendung wird an alle partizipierenden Teilnehmer geleitet. Mit diesem Verfahren wird Application Sharing unabhängig von einer bestimmten Anwendung realisiert. Ein im Besitz des Tokens befindlicher Teilnehmer kann Eingaben bezüglich der Anwendung tätigen. Als weiterer Mechanismus kann ein Zeiger **(Telepointer)** durch jeden Benutzer erstellt werden. Ein korrespondierender Zeiger wird daraufhin bei allen anderen Teilnehmern dargestellt. Die Verwendung von Telepointern ermöglicht eine effiziente Zusammenarbeit der

5 Der Multimedia Collaboration Service (MMC)

In diesem Kapitel wird das Projekt MMC vorgestellt. Nach einem Überblick in Kapitel 5.1 folgt die Betrachtung der Komponenten und deren Erklärung in Kapitel 5.2. Der Transport von Audio und Video wird in Kapitel 5.3 gesondert betrachtet.

5.1 Überblick

Die nächsten Unterkapitel geben einen Überblick über den Projekthintergrund und den Funktionsumfang von MMC.

5.1.1 Projekthintergrund

MMC (Multimedia Collaboration Service) [ADH&93] ist ein Teil des Projekts BERKOM (Berliner Kommunikationssysteme) [AST94]. Im Rahmen von BERKOM existieren mehrere Forschungsbereiche. Dazu gehören u.a. MMT (Multimedia Transport Service) [AST94] als Transportsystem für multimediale Daten, MMM (Multimedia Mail Service) [AST94] als multimediales Mail-System und MMC zur multimedialen Zusammenarbeit. Auf MMT und MMM wird im folgenden nicht weiter eingegangen.

Hintergrund des BERKOM Projekts ist die Tatsache, daß immer mehr große Unternehmen und öffentliche Einrichtungen geographisch über weite Strecken verteilt sind. Durch diese dezentrale Aufteilung entsteht ein erhöhter Bedarf an Kommunikation - nicht nur innerhalb, sondern auch zwischen Unternehmen bzw. Einrichtungen. Ein weiterer Aspekt ist der Umzug der Bundesregierung von Bonn nach Berlin, der sich über einen langen Zeitraum erstrecken wird. Einige Einrichtungen der Bundesregierung werden in Bonn bleiben, so daß ein hoher Kommunikationsbedarf u.a. zwischen Bonn und Berlin besteht. Diese Voraussetzungen bilden die Basis für die Gestaltung neuartiger Kommunikationsdienste, die Gegenstand des Projekts BERKOM sind.

Die DeTeBerkom, ein Tochterunternehmen der Deutschen Bundespost Telekom, leitet die Projekte. An der Entwicklung von MMC sind die folgenden Unternehmen beteiligt: Digital Equipment CEC Karlsruhe, GMD Fokus Berlin, Hewlett-Packard Bristol, IBM ENC Heidelberg, Liebig & Ullfors Berlin, Siemens und die TU Berlin (Prozeßrechenzentrum). MMC wird entsprechend den unterschiedlichen Unternehmen auf unterschiedlichen Systemen entwickelt. Implementierungsspezifische Aspekte von MMC beziehen sich in dieser Arbeit auf die Systemlösung von Digital Equipment, welche MMC auf Alpha AXP Rechner unter OSF/1 realisiert. Der betrachtete Stand der Entwicklung bezieht sich auf Mai 1995.

Das Ziel von MMC ist die Entwicklung eines offenen, homogenen Rahmenwerks für multimediale Zusammenarbeit. Für die Entwicklung wurden insbesondere zwei Aspekte berücksichtigt:

bestimmte Anwendung festlegen. Für den Transport von Audio und Video existiert eine Anhangdatei [Schu95c], welche zusammen mit der Spezifikation als grundlegende Anhangdatei festgelegt wurde.

Einer der wesentlichen Aspekte einer Anhangdatei ist die Festlegung der Interpretation der Nutzdaten über das Feld Nutzdatentyp (PT) im Paketkopf von RTP-Paketen. Hierfür enthält [Schu95c] Tabellen, welche eine Zuordnung zwischen Audio-/Videokodierungen und Werten enthält. Die einer Kodierung entsprechenden Werte werden in das Feld Nutzdatentyp eingetragen. Im Rahmen der Zuordnung zwischen Kodierungen und Konstanten erfolgt darüber hinaus eine genauere Spezifikation der Kodierung, bei Audio ist dies beispielsweise die Festlegung der Abtastfrequenz und die Anzahl der Kanäle. Ebenfalls notwendig ist die Spezifikation der Struktur der Daten, beispielsweise die Anordnung der Daten für mehrere Kanäle innerhalb eines Pakets. [Kast95] und [HFG94] enthalten Entwürfe von Spezifikationen für den Transport von Video. Auch die Interpretation der Felder Zeitstempel und Markierungsbit muß in Anhangdateien festgelegt werden.

Ein weiterer Bestandteil von Anhangdateien ist die Festlegung von anwendungsspezifischen Erweiterungen innerhalb von RTP-/RTCP-Paketen. Dies sollte jedoch nur dann erfolgen, wenn die Anhangdatei speziell für eine Anwendung festgelegt ist. Allgemeine Anhangdateien für Klassen von Anwendungen sollten solche Konventionen nicht enthalten.

RTP-/RTCP-Pakete müssen von einem Kommunikationssystem übertragen werden. Aspekte, welche die Schnittstelle zwischen diesem Kommunikationssystem und RTP/RTCP betreffen, können auch Bestandteil von Anhangdateien sein.

4.3.6 Anwendungsspezifische Pakete

Für Anforderungen, die über die bisher genannten Pakete hinausgehen, können anwendungsspezifische Pakete, **(APP-Pakete)** festgelegt werden. Sie dienen dem experimentellen Einsatz. Wenn ein Pakettyp sich als sinnvoll erwiesen hat, dann sollte er standardisiert werden.

Der Inhalt der Felder **P, T, PT** und **Länge** ist wie bei den anderen Kontrollpaketen zu interpretieren.

Das Feld **Subtyp** (5 Bits) dient der Unterscheidung zwischen mehreren anwendungsspezifischen Paketen.

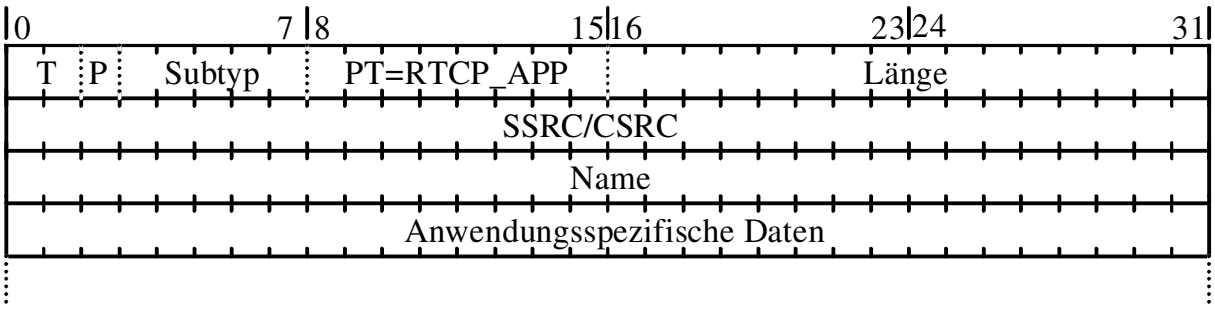


Abbildung 4-11: APP-Paket

Der **Name** (4 Bits) beinhaltet vier ASCII Zeichen, welche eindeutig eine Gruppe von APP-Paketen identifizieren. Dieser Bezeichner legt eine Person/Institution fest, welche die APP-Pakete spezifiziert hat. Innerhalb dieser Gruppe werden die Pakete mittels des Subtyps unterschieden.

Im Feld **Anwendungsspezifische Daten** sind die anwendungsorientierten Daten enthalten. Sie werden nicht von RTP interpretiert. Kapitel 6.2 enthält Beispiele für APP-Pakete.

4.4 Anhangdateien

Die Spezifikation von RTP legt nicht alle Bestandteile des Protokolls eindeutig fest. Dies erfordert für die praktische Anwendung des Protokolls eine Festlegung der offenen Punkte. Dazu gehört die Interpretation der Teile der Protokolldateneinheiten, deren Interpretation in der Spezifikation nicht eindeutig festgelegt ist. Des weiteren muß die Verwendung der anwendungsspezifischen Erweiterungen festgelegt werden. Bei Systemen, die mittels RTP Informationen transportieren, sind für die aufgeführten Aspekte globale Konventionen erforderlich. Zu diesem Zweck werden Anhangdateien (Profiles) verwendet. Sie dienen gewissermaßen als Zusatzspezifikation. Unterschiedliche Anwendungen und Anwendungsbereiche können verschiedene Anhangdateien als Grundlage verwenden. Anhangdateien können einen allgemeinen Inhalt besitzen oder Besonderheiten für eine

Komponente (Konstante)	Wert	Beschreibung
NAME	2	Informelle Bezeichnung des Benutzers, eventuell verbunden mit einer Bezeichnung des Berufs. Vorgesehen ist diese Bezeichnung für die Darstellung in Listen partizipierender Teilnehmer.
EMAIL	3	Electronic-Mail Adresse eines Teilnehmers.
PHONE	4	Telefonnummer eines Teilnehmers.
LOC	5	Ort eines Teilnehmers.
TXT	6	Textangabe bezüglich eines Teilnehmers. Beispiele: Titel einer Rede, Hinweis auf vorübergehende Abwesenheit.
TOOL	7	Verwendetes Anwendungsprogramm.
PRIV	8	Ermöglicht die Spezifikation von weiteren Komponenten. Der Aufbau ist in [SCFJ94; Kapitel 6.5.8] dargestellt.

Tabelle 4-2: SDES Komponenten

4.3.5 Abmeldung

Zur Mitteilung über das Verlassen einer Konferenz können **BYE-Pakete** versendet werden. BYE-Pakete können von Datenquellen, Datensenken oder Mixern versendet werden. Quellen und Senken teilen in den Paketen das Verlassen einer Konferenz mit. Von Mixern versendete Pakete beinhalten den eigenen Identifikator und alle Identifikatoren der Quellen, die von dem Mixer zusammengefaßt wurden. Ein BYE-Paket von einem Mixer zeigt das Einstellen der Funktionalität als Zwischensystem an. An die Liste der Identifikatoren kann noch der Grund des Beendens angehängt sein. Die Länge dieses Feldes muß, wie die Länge des gesamten Pakets, ein ganzzahliges Vielfaches von 32 Bits betragen.

Der Inhalt der Felder **T, P, CC, PT** und **Länge** ist wie bei der Quellenbeschreibung zu interpretieren. Abbildung 4-10 zeigt ein BYE-Paket.

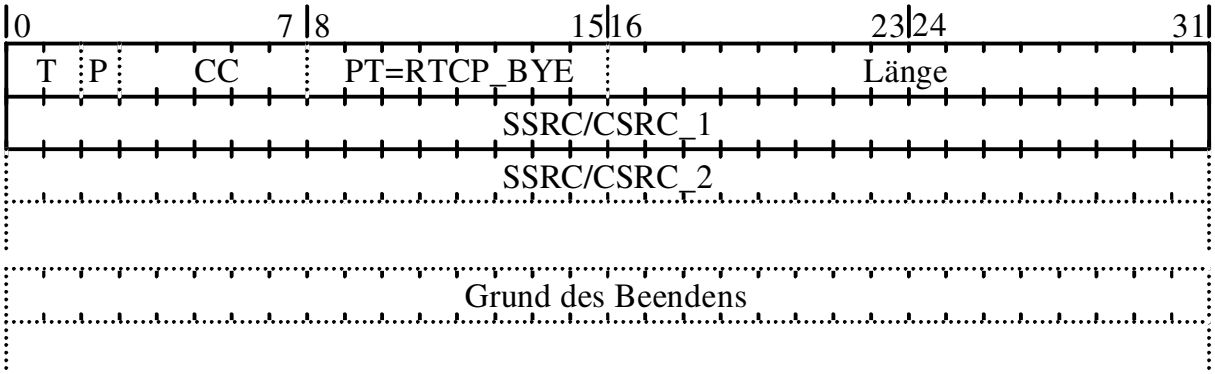


Abbildung 4-10: BYE-Paket

aufgezählt werden. Innerhalb einer Quellenbeschreibung sind die Komponenten direkt hintereinander angeordnet und nicht an 32-Bits Grenzen ausgerichtet. Alle Komponenten enthalten eine Längenangabe, wodurch die Grenzen der Komponenten erkannt werden können. Das Ende einer Liste von Komponenten in einer Quellenbeschreibung wird durch eine oder mehrere Nullen festgelegt. Die Länge einer Quellenbeschreibung beträgt immer ein ganzzahliges Vielfaches von 32 Bits. Die Anzahl der Nullen am Ende einer Komponentenliste richtet sich nach den 32-Bits Grenzen.

Ein Großteil der SDES Komponenten unterscheiden sich nicht in ihrem Aufbau. Daher ist im folgenden nur eine Komponente graphisch dargestellt. Nicht jede Komponente muß in jedem Paket enthalten sein, eine Ausnahme bildet der kanonische Endpunkt-Identifikatior. Er muß in jedem Paket enthalten sein.

Der **Kanonische Endpunkt-Identifikator** (**CNAME**; canonical name) ist ein eindeutiger Identifikator für ein System. Der Identifikator sollte eindeutig innerhalb einer Konferenz sein. Dies ist insbesondere deshalb notwendig, weil der Identifikator SSRC eines Systems sich ändern kann, wenn mehrere Systeme den gleichen Identifikator verwenden. In der Spezifikation wird die Form Benutzer@Rechner oder die Internetadresse des Rechners vorgeschlagen. Werden von einer Quelle durch unterschiedliche Anwendungen mittels RTP Daten übertragen, so sollte zur Vereinfachung ein kanonischer Endpunkt-Identifikator übergreifend für alle Anwendungen festgelegt werden. In Abbildung 4-9 ist der Aufbau des kanonischen Endpunkt-Identifikators dargestellt. Der Inhalt aller Komponenten ist durch ASCII-Zeichen kodiert.

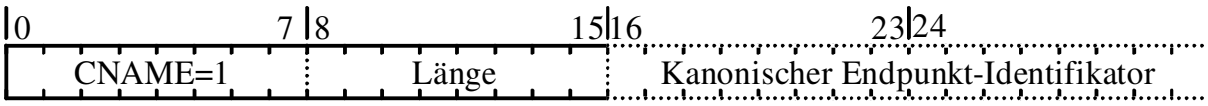


Abbildung 4-9: Kanonischer Endpunkt Identifikator

Weitere Komponenten sind in Tabelle 4-2 dargestellt. Der Wert der jeweiligen Konstante ist im ersten Byte einer Komponente eingetragen. Die Konstanten dienen zur Unterscheidung der Komponenten. Bei einigen Komponenten ist ein Wechsel des Inhalts während einer Konferenz sinnvoll (z.B. TXT).

Empfängerbericht-Paket kann bis zu 31 Empfangsberichte enthalten, weitere Berichte müssen in gesonderten Paketen versendet werden.

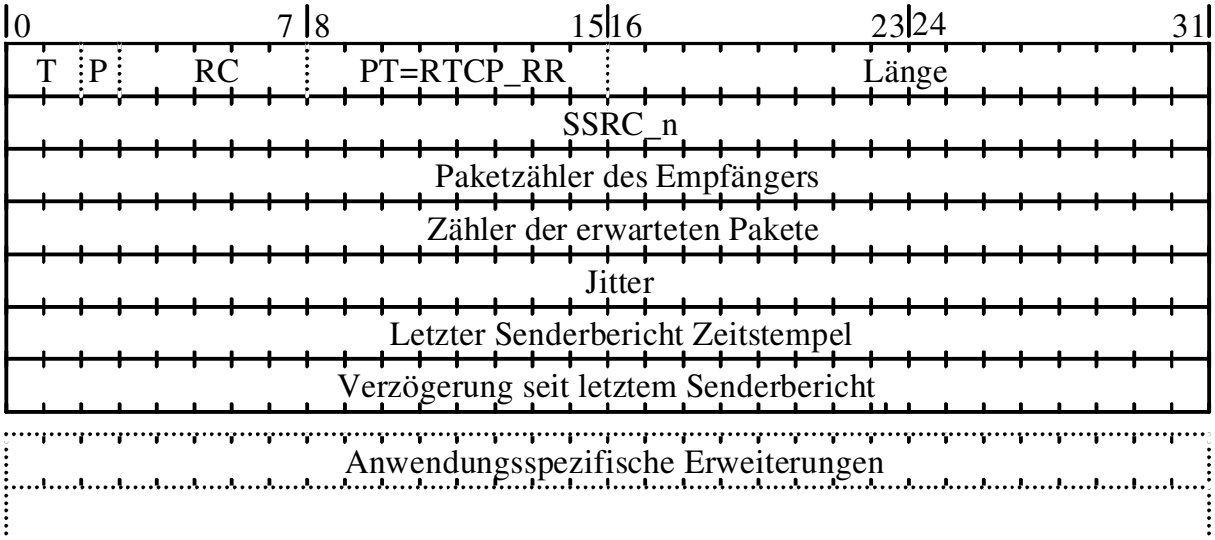


Abbildung 4-7: Empfängerbericht

4.3.4 Quellenbeschreibung

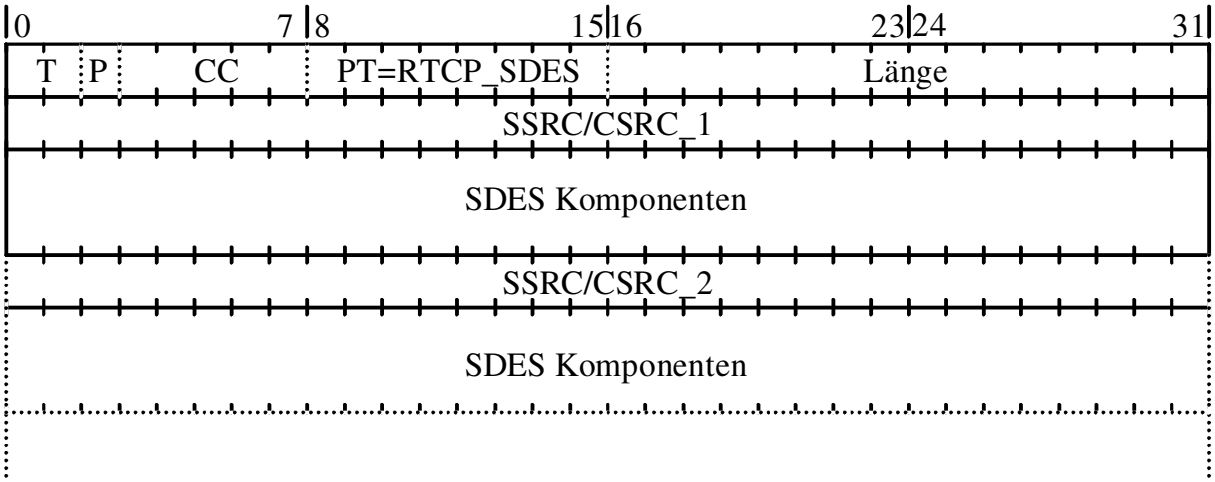


Abbildung 4-8: Quellenbeschreibung

Das Paket **SDES** (source description) ist eine Beschreibung (u.a. Name und Einrichtung des Benutzers) einer oder mehrerer Quellen. Versendet werden die Pakete von Endsystemen, Übersetzern und Mixern. Zwischensysteme senden Quellenbeschreibungen bezüglich sich selbst und der Systeme, von denen sie Daten erhalten und weiterleiten. Endsysteme senden Beschreibungen von sich selbst.

Ein SDES-Paket besteht aus einem Paketkopf und null oder mehreren Quellenbeschreibungen. Null Quellenbeschreibungen sind nach Spezifikation möglich, für die Anwendung jedoch nutzlos. Die Quellenbeschreibung besteht aus mehreren Komponenten, die im folgenden

Die Varianz der Paketankunftszeit ist im Feld **Jitter** (32 Bits) enthalten. Der Wert dient der Überwachung der Qualität der Datenübertragung. In [SCFJ94; Anhang 7] ist ein Algorithmus für die Berechnung des Jitters angegeben. Dabei wird die Varianz der Verspätung der Daten geschätzt. Falls ein System nicht in der Lage ist, den Jitter zu berechnen, dann muß der wert null in das Feld eingetragen werden.

Das Feld **Letzter Senderbericht Zeitstempel** (32 Bits) enthält die mittleren 4 Bytes des NTP-Zeitstempels des letzten empfangenen Senderberichts. Betrachtet wird der Senderbericht der Quelle, bezüglich der der Empfangsbericht erstellt wird.

Im Feld **Verzögerung seit letztem Senderbericht** (32 Bits) ist die Zeit zwischen dem Empfang des letzten Senderberichts und dem Versenden des Empfangsberichts eingetragen. Die Einheit beträgt 1/65536 Sekunden. Dies entspricht der Differenz der Zeitpunkte T2 und T1 in der Abbildung 4-6.

Im folgenden Weg-Zeit-Diagramm ist ein Senderbericht und ein dazugehöriger Empfangsbericht dargestellt. Der Empfangsbericht kann Teil eines Sender- oder Empfängerberichts von *System B* sein. Anhand der letzten beiden vorgestellten Felder können verschiedene Zeiten, beispielsweise T3-T0 oder T3-T0-(T2-T1), bei *System A* berechnet werden.

Die im Feld **Anwendungsspezifische Erweiterungen** eingetragenen Daten werden von RTP nicht interpretiert.

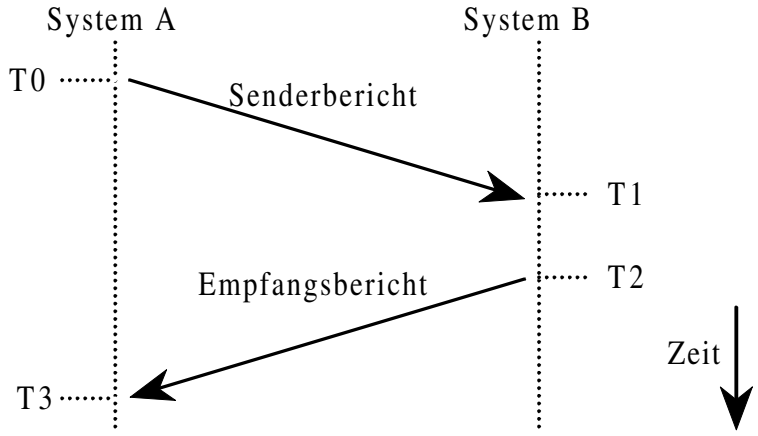


Abbildung 4-6: Weg-Zeit-Diagramm für Berichtpakete

4.3.3 Empfängerbericht

Der Empfängerbericht ist ein RTCP-Paket, welches mehrere Empfangsberichte enthält. Die Komponenten eines Empfangsberichts wurden bereits in Kapitel 4.3.2 erläutert. In diesem Abschnitt ist der Aufbau eines Empfängerberichts dargestellt. Empfängerberichte werden gesendet, wenn eine Anwendung seit einiger Zeit keine RTP-Pakete gesendet hat. Ein

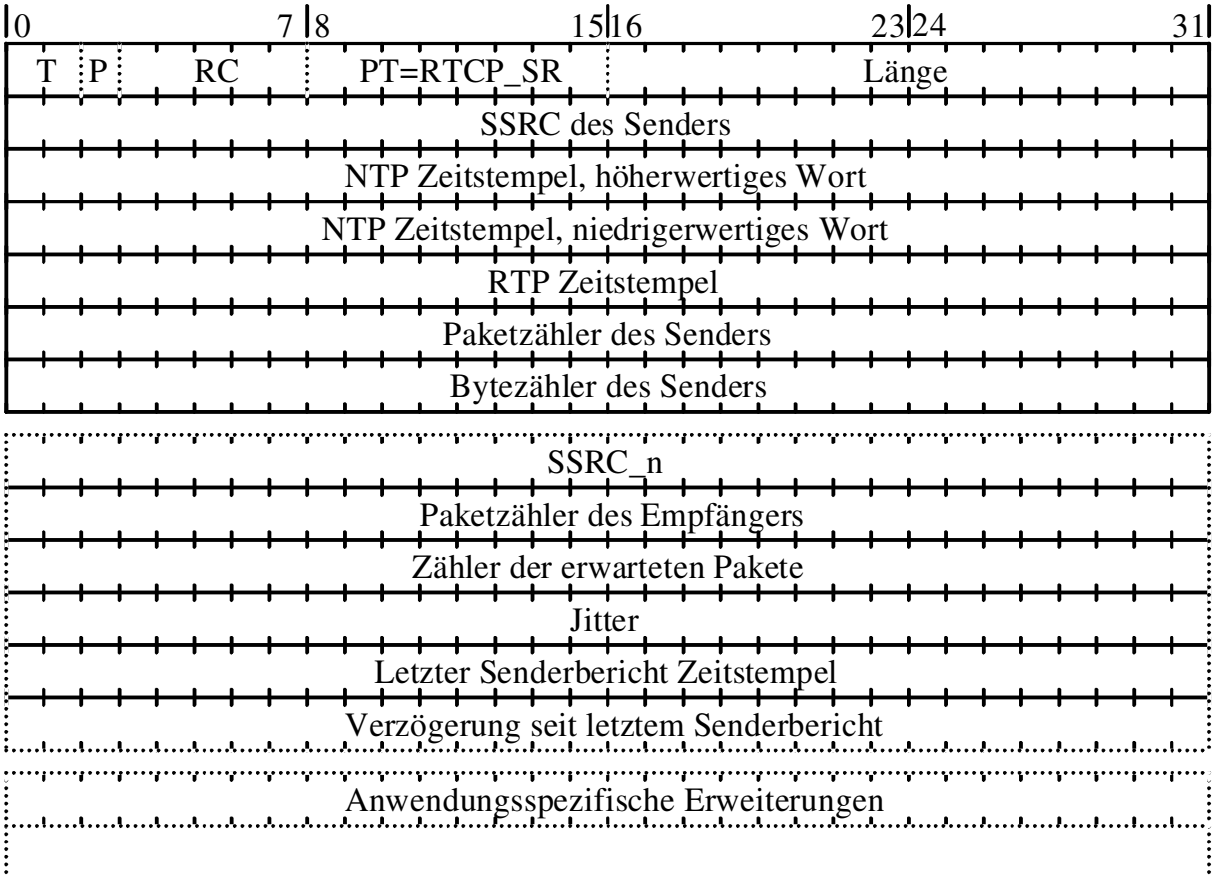


Abbildung 4-5: Senderbericht

Die folgenden Elemente sind Bestandteil eines Empfangsberichts:

Mit dem Feld **SSRC_n** wird der Wert SSRC jenes Systems bezeichnet, auf den sich der Empfangsbericht bezieht.

Der **Paketzähler des Empfängers** (32 Bits) enthält die Anzahl aller empfangenen Pakete seit Beginn der Übertragung von der im Feld SSRC_n angegebenen Quelle. Mittels der Differenz von Paketzählern und den Ankunftszeiten von Paketen kann die Empfangsrate von Paketen errechnet werden.

Der **Zähler der erwarteten Pakete** (32 Bits) beinhaltet die Anzahl der Pakete, die rechnerisch zwischen dem Beginn der Übertragung und der aktuellen Sequenznummer liegen. Zusammen mit den tatsächlich empfangenen Paketen läßt sich die Verlustrate ermitteln. In [SCFJ94; Anhang 8] wird für die Berechnung der rechnerisch empfangenen Pakete folgender Algorithmus angegeben:
Erwartete Pakete = Zyklen * 65536 + aktuelle Sequenznummer - erste Sequenznummer + 1.
Die Anzahl der Zyklen wird vom Beginn der Übertragung gerechnet und jeweils erhöht, wenn ein Paket mit einer niedrigeren Sequenznummer einem Paket mit einer höheren Sequenznummer folgt. Die Anzahl der Zyklen muß mit 65536 multipliziert werden, da dies der Anzahl der Pakete pro Zyklus entspricht.

Empfangsberichte sind Blöcke der Größe 24 Bytes mit Informationen über empfangene Daten. Die Empfangsberichte beziehen sich auf die Quellen, von denen seit dem letzten Senderbericht Daten empfangen wurden. Pro Quelle wird ein Empfangsbericht gesendet. Sind mehr als 31 Empfangsberichte notwendig, so werden diese als Empfängerbericht-Paket gesendet. Dieser Pakettyp ist in Kapitel 4.3.3 erläutert. Am Ende des Senderberichts kann noch eine anwendungsspezifische Erweiterung angehängt sein. Der Aufbau des Pakets ist in Abbildung 4-5 abgebildet, die Felder werden in den nächsten Abschnitten erläutert.

Das Typfeld **T** (2 Bits) dient zur Unterscheidung zwischen RTCP und anderen Protokollen und zur Versionsidentifikation. Wie bei RTP-Paketen wird der Wert zwei verwendet.

Das Füllbit **P** (padding; 1 Bit) zeigt an, daß bei der Interpretation der Nutzdaten am Ende des Pakets eine gewisse Anzahl Bytes ignoriert werden müssen. Die Anzahl der Bytes ist durch das letzte Byte der Nutzdaten festgelegt. Solche Erweiterungen sind zum Teil anwendungsspezifisch notwendig, etwa bei Verschlüsselungsalgorithmen.

Im Feld **RC** (reception report count; 5 Bits) ist die Anzahl der angehängten Empfangsberichte abgelegt. Der Wertebereich liegt zwischen 0 und 31.

Der Pakettyp **PT** (8 Bits) beinhaltet die Art des Pakets, beim Senderbericht ist dies die Konstante RTCP_SR.

Die **Länge** (16 Bits) gibt die Länge des gesamten Pakets in 32-Bits Worten minus eins an.

Die Identifikator des Senders ist im Feld **SSRC des Senders** (32 Bits) abgelegt.

Der **NTP Zeitstempel** (64 Bits) entspricht dem Zeitpunkt der Systemuhr, zu dem der Bericht versendet wurde. Mit NTP Zeitstempeln kann die Laufzeit von Paketen ermittelt werden.

Der Wert des Feldes **RTP Zeitstempel** (32 Bits) entspricht dem Zeitpunkt des NTP Zeitstempels. Der Wert muß entsprechend der globalen Konvention über die Werte von RTP Zeitstempeln interpretiert werden. Dieser Wert kann zusammen mit dem NTP Zeitstempel zur Synchronisation oder zur Bestimmung der nominalen Geschwindigkeit des RTP Zeitstempels verwendet werden.

Der **Paketzähler des Senders** (32 Bits) enthält die Anzahl der von der Quelle gesendeten Pakete seit dem Beginn der Übertragung.

Der **Bytezähler des Senders** Bytes (32 Bits) beinhaltet die Anzahl der von der Quelle gesendeten Bytes seit dem Beginn der Übertragung bis zu diesem Paket. In die Summe werden nur Nutzdaten einbezogen.

nicht über die Netzwerke übertragen werden. Wünschenswert wäre demnach ein Mechanismus, durch den das Senden von Kontrollpaketen unterbunden werden kann. Seitens des Protokolls ist eine Unterstützung einfach zu realisieren. Eine Möglichkeit besteht in der Spezifikation eines neuen Paketes, welches den Wunsch der Unterbindung von Kontrollinformation anzeigt. Hierbei ist jedoch kritisch, welcher der an einer Konferenz partizipierenden Teilnehmer zur Anzeige einer entsprechenden Information autorisiert ist. Diese Problematik müßte seitens der Anwendung gelöst werden. Eine weitere Möglichkeit besteht in der Kennzeichnung aller Datenpakete (eines Datenstroms), für die keine Kontrollinformation gewünscht ist. Dazu müßte in einer Paketkopferweiterung ein entsprechender Vermerk eingesetzt werden. In keinem Fall jedoch kann garantiert werden, daß kein Empfänger von Daten Kontrollinformation bezüglich der Daten zurücksendet. Eine Erweiterung von RTCP um die aufgeführte Funktionalität wäre wünschenswert.

Alle RTCP-Pakete beginnen mit einem einheitlichen Paketkopf, die Länge der Kontrollpakete ist unterschiedlich und wird durch eine Längenangabe im Paket bestimmt. Die Paketlänge beträgt immer ein ganzzahliges Vielfaches von vier Bytes. Durch die Längenangabe ist es möglich, mehrere Kontrollpakete in einem darunterliegenden Protokoll zu versenden. Anhand der Längenangabe können die zusammengefaßten Kontrollpakete beim Empfänger identifiziert werden. Da Kontrollpakete gegenüber Nutzdatenpaketen bezüglich Verzögerung bei der Übertragung eine geringere Priorität besitzen, ist ein Sammeln von Kontrollpaketen durchaus sinnvoll. In den nächsten Kapiteln werden die Paketformate entsprechend der Spezifikation [SCFJ94; Kapitel 6] erläutert.

Bei RTCP gibt es fünf verschiedene Pakettypen. Senderbericht, Empfängerbericht, Quellenbeschreibung und Abmeldung sind in der Spezifikation festgelegt, ein anwendungsspezifisches Paket ermöglicht flexible Pakete, die speziell für eine Anwendung entworfen werden können. Als erstes Kontrollpaket wird immer ein Berichtpaket versendet. Der Pakettyp ergibt sich aus dem Eintrag des Feldes PT, welches in jedem RTCP-Paketkopf enthalten ist. Die folgende Tabelle listet die festgelegten Pakete auf.

Wert	Konstante	Beschreibung
0	RTCP_SR	Senderbericht
1	RTCP_RR	Empfängerbericht
2	RTCP_SDES	Quellenbeschreibung
3	RTCP_BYE	Abmeldung
4	RTCP_APP	Anwendungsspezifisches Paket

Tabelle 4-1: RTCP Pakettypen

4.3.2 Senderbericht

Ein Senderbericht (sender report) wird von Systemen versendet, die innerhalb eines Zeitintervalls (Vgl. [SCFJ94; Anhang 6]) Daten gesendet haben. Er besteht aus einem festen, 28 Bytes langem Teil, dem bis zu 31 Empfangsberichte angehängt sein können.

eine bestimmte Größe der Pakete, die mit der Längenanpassung erreicht werden kann. Verschlüsselt wird das ganze Paket, demnach ist auch die Bearbeitung eines verschlüsselten Pakets in Zwischensystemen nur mit dem entsprechenden Schlüssel möglich. Vorgesehen ist der Verschlüsselungs-Algorithmus DES (Data Encryption Standard) im CBC (Cipher Block Chaining) Modus [Bale93].

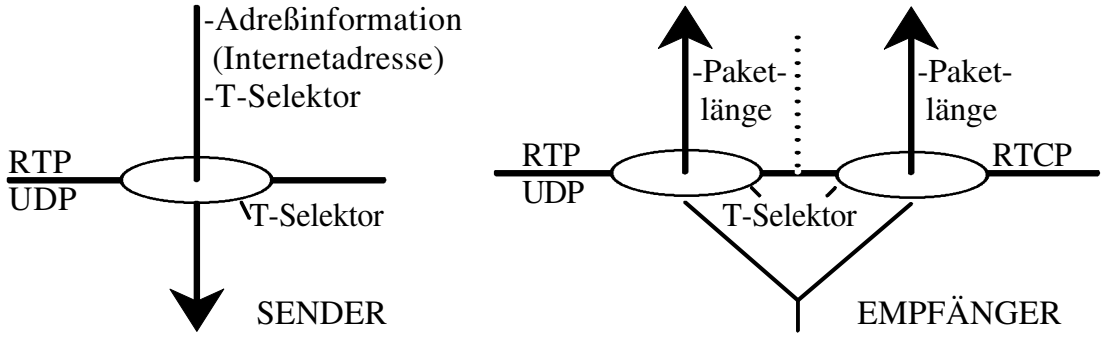


Abbildung 4-4: Schnittstellen-Kontrollinformation und Selektoren

4.3 Das Kontrollprotokoll RTCP

Im folgenden wird das Kontrollprotokoll RTCP erläutert, es erweitert RTP um den Transport von Kontrollinformation.

4.3.1 Einleitung

Das **Realzeit-Transport-Kontrollprotokoll RTCP** (Real-Time Transport Control Protocol) bietet zwei Funktionen [SCFJ94; Kapitel 6]:

- Überwachung der Datenübertragung (z.B. Paketverlust und Laufzeiten) und
- Übermittlung von (Identifikations-) Information innerhalb einer Konferenz.

Die Überwachung der Datenübertragung wird mit Sender- und Empfängerberichten realisiert, die in annähernd periodischen Abständen verschickt werden. Das Zeitintervall zwischen zwei Berichten vergrößert sich mit der Anzahl der Teilnehmer, um die Netzwerke und Endsysteme nicht zu überlasten. Um eine Synchronisation aller Teilnehmer (gleichzeitiges Eintreffen von Berichten) zu vermeiden, werden die Zeitintervalle zudem zufällig variiert. Zur Überwachung von Konferenzen mit wechselnden Teilnehmern existieren Kontrollmechanismen für An- und Abmeldung. Darüber hinaus gibt es die Möglichkeit, anwendungsspezifische Pakete festzulegen.

Die Spezifikation legt fest, daß gewisse Pakettypen in periodischen Abständen versendet werden. Dazu gehören die in den Kapiteln 4.3.2 und 4.3.3 vorgestellten Sender- und Empfängerberichte. Es läßt sich jedoch problemlos ein Szenario entwickeln, bei dem ein Versenden von Kontrollinformationen nicht wünschenswert ist. Beispielsweise bei der Übertragung von Radio über Computernetzwerke ist der Sender der Daten in der Regel nicht daran interessiert, von Empfängern Kontrollinformationen zu erhalten. Diese sollten dann auch

verwendet werden, die dem Aufbau nach unterschieden werden können. Die Spezifikation der Erweiterung ist in der Anhangdatei vorgesehen.

4.2.3 Ergänzende Erläuterungen

Dieses Kapitel beinhaltet weitergehende Erläuterungen zum vorangegangenen Kapitel, die über die ausschließliche Spezifikation des Protokolls hinausgehen.

Der RTP-Paketkopf beinhaltet kein Feld für die Längenangabe des gesamten Pakets. Die Länge des Pakets muß vom Protokoll der darunterliegenden Schicht als Bestandteil der Schnittstellen-Kontrollinformation bei der Paketübergabe übermittelt werden. Mit dem Einsparen der Längenangabe reduziert sich der Aufwand für den Paketkopf.

RTP verwendet als Adreß-Identifikatoren eine 4-Bytes Zahl. Dieser Wert dient als Identifikator für die Endsystemadresse (im Internet 4 Bytes) und den Selektor (mindestens 2 Bytes). Demnach werden bei der Übertragung mindestens 2 Bytes eingespart. Einem Adreß-Identifikator entspricht genau einem Paar bestehend aus Endsystemadresse und Selektor. Die Auswahl des Identifikators erfolgt zufällig. Wie bereits erwähnt, muß beim Erkennen eines anderen Systems mit dem gleichen Identifikator eine Fehlerbehandlung eingeleitet werden. Das Format des RTP Adreß-Identifikators besitzt allerdings den Nachteil, daß Empfänger von Paketen durch einen gesonderten Mechanismus Angaben über den Sender erhalten müssen. Dies erfolgt mittels der in Kapitel 4.3 vorgestellten Kontrollpakete. Weitere Betrachtungen zum Format des Adreß-Identifikators sind Bestandteil von Kapitel 6.1.3.

Zur Identifikation des Protokolls RTP ist innerhalb eines Computersystems ein Selektor **(Port)** notwendig. Über diesen Port werden Pakete zwischen RTP und darunterliegenden Protokollen ausgetauscht. Die Pakete des in Kapitel 4.3 vorgestellten Kontrollprotokolls RTCP können nicht durch ihren Aufbau von RTP-Paketen unterschieden werden. Demnach ist zur Unterscheidung der beiden Pakettypen ein weiterer Selektor für RTCP-Pakete notwendig. Dies ist eine zwar nicht notwendige aber hilfreiche Voraussetzung für die Verwendung zweier unterschiedlicher Protokolle für den Transport der Nutzdaten und der Kontrolldaten. Somit kann für den Transport von Kontrollinformation ein anderes Protokoll verwendet werden. Dafür bietet sich ein verbindungsorientiertes Protokoll an (etwa TCP), welches mittels Übertragungswiederholung eine Übertrag ohne Paketverlust garantiert. TCP ist wiederum für den Transport von Nutzdaten nicht geeignet, da aufgrund von Übertragungswiederholung hohe Verzögerungszeiten entstehen können.
Für die Relation der Ports gilt: Port (RTCP) = Port (RTP) +1.

In Abbildung 4-4 ist die Schnittstellen-Kontrollinformation [Krüg93; Kapitel 1] zwischen RTP und einem darunterliegenden Protokoll (als Beispiel hier UDP) zusammengefaßt. Bei der Quelle wird Adreßinformation übergeben, bei der Senke Längenangabe erhalten.

Die Möglichkeit der Längenanpassung eines RTP-Pakets mittels Füllbit und angehängter Bytes ermöglicht die Verschlüsselung von Paketen: Zum Teil benötigen Verschlüsselungsalgorithmen

Sample erhöht sich der Wert um eins. Pro Paket werden Nutzdaten entsprechend der Länge 20 Millisekunden versendet, das entspricht 160 Samples bzw. Bytes. Somit erhöht sich der Wert des Zeitstempels bei zwei aufeinanderfolgenden Paketen um 160, wenn keine Ruhepause zwischen den Paketen liegt. Wenn Ruhepausen dazwischen liegen, erhöht sich der Wert entsprechend. Mehrere Pakete können auch den gleichen Wert des Zeitstempels beinhalten, wenn sie beispielsweise alle zu einem Bild eines Videos gehören.

Der Identifikator der Synchronisationsquelle wird im Feld **SSRC** (synchronization source; 32 Bit) angegeben. SSRC ist eine global eindeutige, bei der Quelle zufällig bestimmte Zahl, die einen Anwendungsprozeß identifiziert. Erkennt ein System durch den Empfang von Paketen mit dem eigenen Identifikator, daß ein anderes System den gleichen SSRC-Wert verwendet, so muß sich die Quelle zurückziehen und einen neuen SSRC wählen. Zur Unterstützung des Erkennens von anderen Systemen, welche einen identischen SSRC bei der Übertragung verwenden, existiert der bei der Erläuterung des Kontrollprotokolls vorgestellte kanonische Bezeichner. Dieser wird aus Benutzer und Rechner gebildet und ist demnach eindeutig im System. Weiter Betrachtungen über die verwendeten Adreß-Identifikatoren bei RTP sind Bestandteil von Kapitel 6.1.3.

Bis zu 15 **CSRCs** (contributing sources; 32 Bits) können dem festen Paketkopf folgen. Bei Resynchronisation kopieren Mixer die SSRCs der gemischten RTP-Pakete in die CSRC Liste. Die CSRC Liste ermöglicht die Darstellung der ursprünglichen Sender beim Empfänger.

4.2.2 Paketkopferweiterung

Mit der Paketkopferweiterung ist es möglich, individuelle Anpassungen von RTP-Paketen an anwendungsspezifische Anforderungen vorzunehmen. Der Aufbau der Erweiterung ist in Abbildung 4-3 dargestellt. Die Erweiterung besteht aus mindestens 4 Bytes und folgt dem festen Paketkopf und der Liste der CSRCs, wenn das Erweiterungsbit gesetzt ist. Für die Interpretation der Erweiterung und der darin befindlichen Daten ist eine entsprechende Abstimmung zwischen den beteiligten Systemen notwendig. RTP-Systeme, die nicht in der Lage sind, diese Daten zu interpretieren, müssen sie ignorieren.

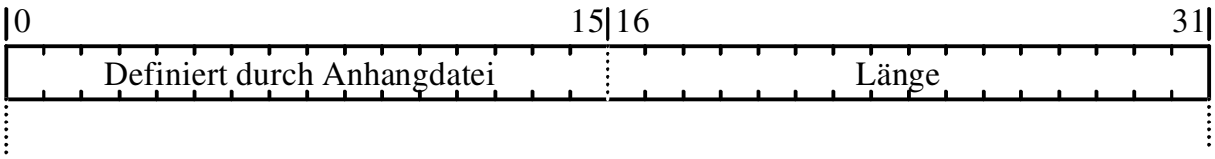


Abbildung 4-3: RTP-Paketkopferweiterung

Die Längenangabe gibt die Länge der Paketkopferweiterung in 4-Bytes-Worten an. Nicht dazugerechnet sind die ersten vier Bytes, die bei der Erweiterung Standard sind. Damit ist null eine gültige Längenangabe. Die Länge muß immer ein Vielfaches von 32 Bits betragen.

Zwecks Unterscheidung der Paketkopferweiterung sind die ersten 16 Bits als Identifikation für den Typ der Erweiterung vorgesehen. Damit können mehrere verschiedene Erweiterungen

Verschlüsselungsalgorithmen eine bestimmte Länge der Nutzdaten oder des gesamten Pakets fordern. Auch Protokolle auf unteren Schichten können eine Längeneinschränkung hervorrufen.

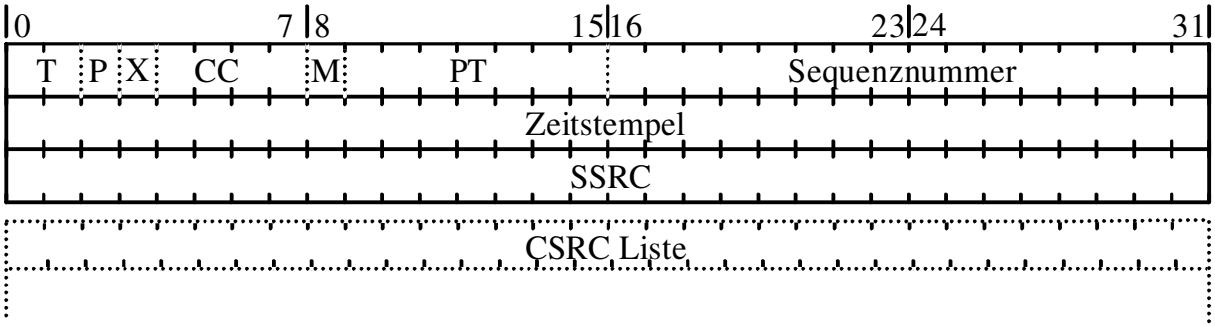


Abbildung 4-2: RTP-Paketkopf

Dem Paketkopf folgt eine Erweiterung (nach der Liste der CSRCs), wenn das Erweiterungsbit **X** (extention; 1 Bit) gesetzt ist. Der Aufbau der Erweiterung ist in Kapitel 4.2.2 dargestellt.

Die Länge der CSRC Liste ist in dem Feld **CC** (CSRC count; 4 Bits) gespeichert. Die Liste folgt dem festen Paketkopf und kommt vor einer etwaigen Paketkopf-Erweiterung.

Die Verwendung des Feldes **M** (marker; 1 Bit) wird durch eine Anhangdatei festgelegt. Es kann verwendet werden für die Anzeige des Beginns einer Synchronisationseinheit. Durch eine Anhangdatei kann die Anzahl der Markierungsfelder vergrößert werden, wenn gleichzeitig das CC Feld verkleinert wird.

Das Feld **PT** (payload type; 7 Bits) zeigt den Typ der Nutzdaten an. Die unterschiedlichen Werte müssen in einer Anhangdatei festgelegt werden. Der Wert von PT dient einer Anwendung oder eines Zwischensystems auf RTP Ebene zur Identifikation des Nutzdatentyps. Beispielsweise ist der Wert 10 vorgesehen für die Übertragung von Audio in CD-Qualität (lineare Quantisierung mit 16 Bits, Abtastfrequenz 44100 Hz, Stereo).

Die **Sequenznummer** (16 Bits) erhöht sich bei jedem abgesendeten Paket um eins. Die Sequenznummer dient einem Empfänger dazu, verlorengegangene Pakete zu erkennen und Reihenfolgevertauschungen festzustellen. Der initiale Wert einer Sequenznummer ist zufällig, da ein fester Wert die Entschlüsselung von verschlüsselten Paketen erleichtern würde.

Der Wert des **Zeitstempels** (32 Bits) gibt den Wert des Zeitstempels des ersten Bytes der Nutzdaten an. Die Interpretation des Zeitstempels (Kapitel 2.2.2) muß global, etwa durch Anhangdateien, festgelegt sein. Der Wert des Zeitstempels erhöht sich entsprechend des Zeitstempeltyps. Der Anfangswert ist wie die Sequenznummer zufällig. Ein Beispiel: Die Sample-Uhr bei der Aufnahme von Audio nimmt 8000 Samples der Größe ein Byte pro Sekunde auf. Das Zeitstempelfeld wird entsprechend der Sample-Uhr interpretiert, d.h. pro

werden als Übersetzer bezeichnet: Befindet sich ein Übersetzer an der Verbindungsstelle zwischen zwei Kommunikationssystemen, beispielsweise ST-II/ATM und UDP/IP/Ethernet, nimmt der Übersetzer die Pakete von der einen und gibt sie an die andere Seite weiter. Auch bei abgeschirmten (isolierten) Netzwerken, die mittels Gruppenkommunikation nicht erreichbar sind, werden Übersetzer eingesetzt. Ein Übersetzer nimmt Pakete entgegen, gibt sie an einen Übersetzer innerhalb des abgeschirmten Netzwerks weiter und von dort können die Pakete ggf. mittels Gruppenkommunikation weitergeleitet werden. Dies entspricht beispielsweise dem Audiostrom, der in Abbildung 4-1 durch *Übersetzer 1* und *Übersetzer 2* von *Quelle 1* zur *Senke* läuft.

Resynchronisation in Zwischensystemen macht in RTP-Paketen eine Unterscheidung zwischen der Quellenangabe notwendig: Es gibt

- die Beitragsquelle (contributing source; **CSRC**) und
- die Synchronisationsquelle (synchronization source; **SSRC**).

Es gibt immer genau *eine* Synchronisationsquelle, während mehrere Beitragsquellen existieren können. Werden Informationen auf einem Weg übertragen, der keine Resynchronisation beinhaltet, so sind CSRC und SSRC identisch. Der Wert ist im SSRC Feld eingetragen. Bei einer Resynchronisation mehrerer Quellen in einem Mixer, setzt der Mixer seine Quellenangabe in das Feld SSRC ein und hängt die synchronisierten Quellen als CSRC Liste an. Im Beispielszenario der Abbildung 4-1 sind unter den Pfeilen SSRC und CSRCs der Pakete eingetragen.

4.2 Protokolldateneinheiten

Die folgenden Unterkapitel beinhalten den Aufbau der Protokolldateneinheiten des Protokolls RTP.

4.2.1 Der Paketkopf

Der RTP-Paketkopf besteht aus mindestens 12 Bytes, der durch eine Liste von CSRCs und eine Paketkopferweiterung (header extention) ergänzt werden kann. Ein Feld der Größe 4 Bits im Paketkopf zeigt an, wieviele CSRCs angehängt sind und ein Feld von der Größe eines Bits zeigt die Erweiterung des Paketkopfs an. Der Paketkopf ist den Nutzdaten vorangestellt. Die folgende Abbildung zeigt den Aufbau des Paketkopfs [SCFJ94], dessen einzelne Felder in den folgenden Abschnitten erläutert werden.

Das Typfeld **T** (2 Bits) gibt den Typ bzw. die Version des RTP-Pakets an. Hier wird die zum Zeitpunkt der Arbeit aktuelle Version zwei vorgestellt. Mit dem Typfeld werden Versionen von RTP unterschieden, es wird auch als Unterscheidung zu anderen Protokollen verwendet (z.B. VAT, Kapitel 3).

Das Füllbit **P** (padding; 1 Bit) zeigt an, ob den Nutzdaten des RTP-Pakets noch Bytes angehängt sind. Ist das Füllbit gesetzt, so beinhaltet das letzte Oktet des Pakets die Anzahl der zu ignorierenden Bytes. Das Anhängen von Daten ist notwendig, wenn

Kontrollprotokolls RTCP Kontrollpakete versendet, deren Verwendung in Kapitel 4.3 dargelegt wird.

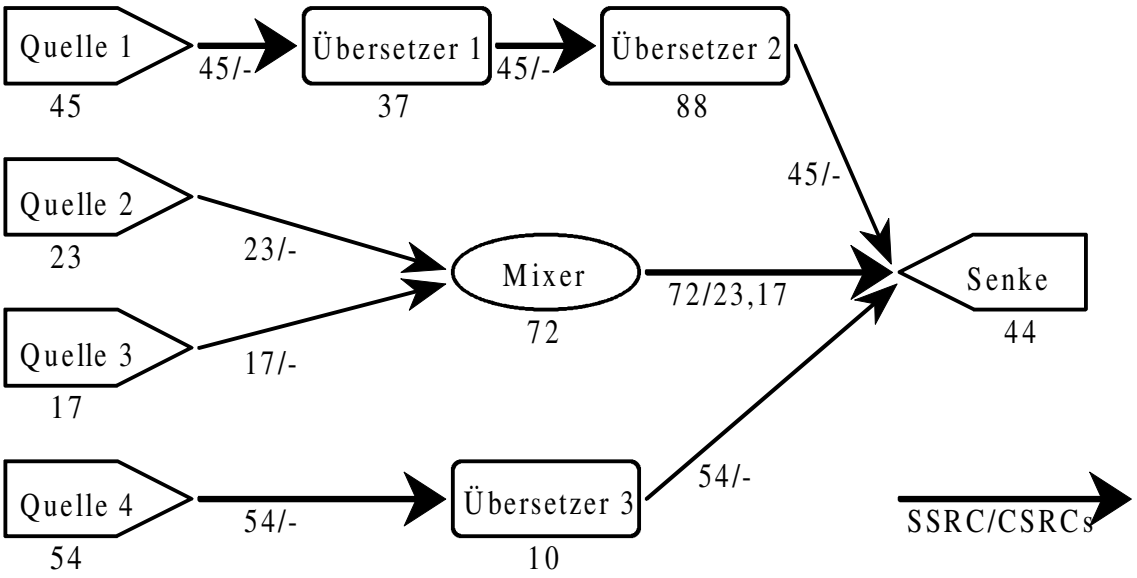


Abbildung 4-1: RTP Beispielszenario

Zwischensysteme auf RTP-Ebene sind definiert als Systeme, die RTP-Pakete bearbeiten. Dabei werden Nutzdaten oder Paketkopf verändert oder Pakete empfangen und weitergeleitet. Zwischensysteme sind nie Ursprung oder Ziel einer Übertragung. Es gibt zwei verschiedene Arten von Zwischensystemen:

- Mixer (mixers) und
- Übersetzer (translators).

Die Zwischensysteme unterscheiden sich in ihrer Funktionsweise:

Werden Audioströme mehrerer Quellen gebündelt oder mehrere Pakete einer Quelle zusammengefaßt, so handelt es sich um eine **Resynchronisation**. Dieser Fall wird als **Mixer** bezeichnet. Der im Beispielszenario abgebildete *Mixer* bündelt die Audioströme der *Quellen 2* und *3*. Der Empfänger *Senke* muß die Synchronisation dann bezüglich des resynchronisierenden Zwischensystems (Mixer) durchführen. Beim Bündeln von Audioströmen kann die Qualität verschlechtert aber nie verbessert werden. Die Qualität wird durch die Wahl der Zielkodierung im Mixer festgelegt. Der Vorteil von Mixern wird an der Grenze zwischen Netzwerken unterschiedlicher Kapazität deutlich: Auf der Seite mit der höheren Kapazität können die Teilnehmer mit einer hohen Audio-Qualität konferieren, ein Mixer bildet das Zwischenglied zum Netzwerk mit geringerer Kapazität. Er setzt die Kodierung um und bündelt noch unterschiedliche Audioströme.

Werden die Nutzdaten eines RTP-Paketes in einem Zwischensystem umkodiert, jedoch nicht mehrere Audioströme gebündelt und nicht mehrere Pakete einer Quelle zusammengefaßt, so liegt keine Resynchronisation vor. Dieser Fall wird als **Übersetzer** bezeichnet. Dies entspricht dem Audiostrom von *Quelle 4* zur *Senke.* Auch die folgenden Arten von Zwischensystemen

- Identifikation des Nutzdatentyps,
- das Kontrollprotokoll RTCP für Kontroll- und Identifikationsfunktion,
- Unterstützung von Zwischensystemen auf RTP-Ebene und
- Erweiterbarkeit und Flexibilität mittels RTP Anhangdateien.

Folgende Aspekte sind nicht Bestandteil von RTP:
- Garantien für pünktliche Auslieferung von Paketen und
- Garantien für andere Dienstgüte-Merkmale (z.B. Bandbreiten-Reservierung).

Entwickelt wurde RTP als Transportprotokoll für multimediale Konferenzsysteme, ist aber nicht auf diesen Bereich beschränkt. Weitere Möglichkeiten der Anwendungen bestehen in verteilten Realzeit-Simulationen, entferntem Speicherzugriff für (multimediale) Daten, Informationsdiensten oder verteilten kooperativen Diensten (Computer Supported Cooperative Work).

Die Spezifikation von RTP läßt einige Fragen offen (z.B. die Interpretation einiger Felder), welche durch Anhangdateien (Profiles) geklärt werden. RTP verwendende Anwendungen müssen sich vorab auf eine oder mehrere Anhangdateien einigen, die in den Anhangdateien festgelegten Konventionen werden damit bindend für die Datenübertragung. Anhangdateien bieten flexible Erweiterungsmöglichkeiten, diese werden in Kapitel 4.4 erläutert.

4.1.2 Beispielszenario und Zwischensysteme

Anhand des folgenden Beispiels einer (Audio)-Konferenz wird die Basisfunktionalität von RTP und die Verwendung von Zwischensystemen verdeutlicht. In Abbildung 4-1 sind mehrere, durch Zwischensysteme verbundene, Teilnehmer einer Konferenz formal dargestellt. Die unterschiedliche Funktionalität der Zwischensysteme wird ebenso wie die Bedeutung der unter den Elementen dargestellten Zahlen in den folgenden Abschnitten erläutert.

Als *Quelle* bezeichnete Teilnehmer sind Ursprung der Übertragung, als *Senke* bezeichnete Teilnehmer sind endgültige Empfänger von Audio. Die Funktionsweise von RTP ist bezüglich Nutzdaten unidirektional, Kontrollinformationen werden bidirektional versendet. Durch Pfeile sind **Audioströme** von Quelle zu Senke gekennzeichnet. Unter den End- oder Zwischensystemen sind die Identifikatoren (bei RTP eine 4-Bytes Zahl) der Systeme angegeben.

Vor Beginn einer Konferenz ist die Bestimmung der Teilnehmer bzw. deren eindeutige Identifikation (Rechneradresse und ggf. Selektor zur Identifikation des Anwendungsprozesses) in einem Netzwerk notwendig. Zur Vereinfachung kann eine Adresse für Gruppenkommunikation für alle partizipierenden Teilnehmer festgelegt werden (entweder temporär oder ständig). Im Internet ist dies mittels des übergelagerten Netzes MBone (Multicast Backbone) [Erik94] möglich. Diese Mechanismen sind jedoch nicht Bestandteil von RTP. Während der Konferenz verschicken die Teilnehmer Audiofragmente, denen ein RTP-Paketkopf vorangestellt ist. Des weiteren werden mittels des zu RTP gehörenden

4 Das Realzeit-Transportprotokoll RTP

In diesem Kapitel wird das Realzeit-Transportprotokoll RTP vorgestellt. Dabei wird zunächst in der Einleitung auf allgemeine Aspekte eingegangen, anschließend folgen die Protokolldateneinheiten, das Kontrollprotokoll und eine Erläuterung von Anhangdateien.

4.1 Überblick

Im folgenden wird nach einem Überblick auf die Grundfunktionalität von RTP eingegangen. Ein Vergleich und eine Bewertung von RTP und dem in Kapitel 5 vorgestellten Protokoll AVXP erfolgt in Kapitel 6.

4.1.1 Funktionsumfang

Das **Realzeit-Transportprotokoll RTP** (Real-Time Transport Protocol) [SCFJ94] ist ein Protokoll, dessen überwiegende Funktionalität gemäß dem OSI-BRM innerhalb der Transportschicht angesiedelt ist. Es stellt anwendungsorientierten Protokollen oder Anwendungen direkt Dienste zur Verfügung. Die zur Verfügung gestellten Dienste bzw. der Aufbau der Protokolldateneinheiten orientiert sich an den Anforderungen von interaktiven verteilten multimedialen Anwendungen. Die Bezeichnung als Realzeit-Protokoll ist insofern irreführend, da bei RTP keine Garantien bezüglich Verzögerungen und Verzögerungsschwankungen erfolgen. RTP nutzt, wie im nächsten Abschnitt erläutert, Dienste der Vermittlungsschicht oder auch der Transportschicht.

Je nach Anforderungen setzt RTP auf ein Protokoll der Vermittlungsschicht oder der Transportschicht auf. Minimal notwendig ist ein ungesicherter Ende-zu-Ende Transport. Nach [Schu93; Kapitel 1.1] kommen hierfür unter anderem UDP, TCP [Davi88], OSI TP1 oder TP4 [Tane90; Kapitel 6] oder auch direkt IP [Davi88], ATM [HäHu91] oder ST-II [Topo90] in Frage. Die Notwendigkeit von UDP beispielsweise hängt davon ab, ob der Dienst „Multiplexen/Demultiplexen“ benötigt wird. Falls dies der Fall ist, so muß UDP verwendet werden, ansonsten kann direkt auf IP aufgesetzt werden. Bei NeVoT (Kapitel 3) wird „Multiplexen/Demultiplexen“ benötigt, deshalb wird auch UDP verwendet.

RTP nutzt folgende Dienste:

- Unzuverlässiger Ende-zu-Ende Transport (verlorengegangene, fehlerhafte oder reihenfolgevertauschte Pakete möglich),
- Multiplexen/Demultiplexen und
- Gruppenkommunikation (Multicast).

RTP bietet Anwendungen die folgende Funktionalität, welche in den nächsten Abschnitten genauer erläutert wird:

- Ende-zu-Ende Transport für kontinuierliche Daten (verbindungsloser Datagrammdienst) durch die Integration von Zeitinformation,

sind zur Identifikation der Pakete deren Zeitstempel (hier in Samples) angegeben. Im Zwischenspeicher sind maximal zwei Pakete, diese werden während einer Ruheperiode gespeichert und jeweils ein älteres durch ein neueres Paket ersetzt. Nach einer zusammenhängenden Audiosequenz werden zwei Pakete übertragen. Paket 160 ist das letzte einer zusammenhängenden Audiosequenz, anschließend werden noch zwei Pakete übertragen (320 und 480). Die Pakete 800 und 960 werden in einem Zwischenspeicher gespeichert und zu Beginn der mit Paket 1120 beginnenden Sequenz übertragen.

Zeitstempel (in Samples)	**Pakettyp (R = Ruhe, S = Sprache)**	**Zwischen-speicher**	**Übertragene Pakete mit Sequenznummer**
160	S		160 (1)
320	R		320 (2)
480	R		480 (3)
640	R	640	
800	R	640, 800	
960	R	800, 960	
1120	S		800 (4), 960 (5), 1120 (6)
1280	S		1280 (7)
1440	S		1440 (8)
1600	S		1600 (9)

Tabelle 3-2: Audioübertragung in NeVoT

Uhren nicht gleich schnell laufen. Dies hat zur Folge, daß Puffer im Audiosystem, die zur Zwischenspeicherung der Audiodaten dienen, unter- oder überlaufen können.

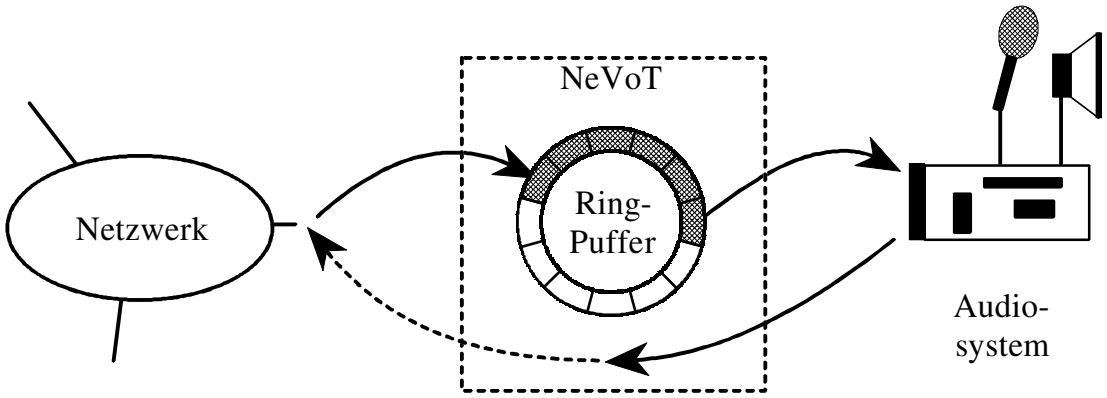


Abbildung 3-4: Zeitliche Kopplung der Ein- und Ausgabe von Audio

Bei NeVoT wird, um ein Auseinanderlaufen der Uhren (clock drift) zu vermeiden, die Audioeingabe mit der Audioausgabe gekoppelt. Dabei wird die Uhr des Audiosystems verwendet, die Systemuhr hat keinen Einfluß auf die Ein-/Ausgabe von Audio.

Jedes Mal, wenn eine gewisse Anzahl von Audiodaten am Audiosystem zur Verfügung steht, werden diese von NeVoT abgerufen, verarbeitet und ggf. versendet. Ankommende Pakete vom Netzwerk werden in einem Ringpuffer gespeichert. Bei jedem Abruf eines Pakets vom Audiosystem wird automatisch ein Paket aus dem Ringpuffer an das Audiosystem gegeben. Mit dieser Vorgehensweise wird die Eingabe und die Ausgabe von Audio zeitlich gekoppelt. Das Verfahren erfordert jedoch, daß die Blocklänge von eingehendem und ausgehendem Audio identisch ist. Abbildung 3-4 zeigt den Fluß der Audiodaten zwischen Audiosystem und Netzwerk.

3.3.2 Unterbrochene Übertragung

Audiopakete werden vor dem Versenden durch NeVoT untersucht, ob sie als Ruhe interpretiert werden können. Die Vorgehensweise der unterbrochenen Übertragung wurde bereits in Kapitel 2.2.1 vorgestellt. Bei NeVoT werden jedoch nicht ausschließlich Pakete einer zusammenhängenden Audiosequenz übertragen. Vor einer zusammenhängenden Audiosequenz wird eine wählbare Anzahl von Paketen übertragen, diese werden während der vorangestellten Ruheperiode in einem Zwischenspeicher abgelegt. Mit Beginn einer zusammenhängenden Audiosequenz werden auch die vorangestellten Audiopakete übertragen. Dadurch vergrößert sich jedoch auch die Ende-zu-Ende Verzögerung zwischen dem Zeitpunkt des Vorliegens der Daten beim Sender und dem Zeitpunkt der Ausgabe der Daten beim Empfänger um die Zeitdauer, die den vorangestellten Paketen entspricht. Nach einer zusammenhängenden Audiosequenz wird ebenfalls eine wählbare Anzahl von Paketen übertragen. Diese Technik (hang-over) verhindert das Abschneiden von Worten deren Endung einen geringen Energiegehalt besitzt und aus diesem Grund als Ruhe interpretiert wird. Tabelle 3-2 zeigt die Übertragung von Audiopaketen im Zusammenspiel mit dem Zwischenspeicher. In den Zellen

PMM
PMM (Pattern Matching Multicast) [Schu95a] ist ein Replikator, der Nachrichten zwischen Prozessen verteilt. PMM dient zur Kommunikation zwischen Komponenten, die Bestandteil eines Konferenzsystems sind, beim Konferenzsystem NeVoT ist dies NEVOT selbst und ICC. PMM ermöglicht die flexible Gestaltung von Konferenzsystemen, bei denen nicht jede Komponente von der Existenz einer anderen Komponente eines Konferenzsystems wissen muß. Für alle an der Verteilung beteiligten Prozesse ist eine Anmeldung bei PMM erforderlich. Nachrichten werden von PMM entgegengenommen und dann an alle angemeldeten Prozesse übertragen. Die PMM-Verbindungen sind mittels TCP realisiert. Diese Vorgehensweise wurde gewählt, da nicht alle Computersysteme lokale Gruppenkommunikation unterstützen [Schu95b; Kapitel 10].

RTP/VAT
Von NEVOT werden zwei Protokolle der Transportschicht verwendet, um Audiopakete zu übertragen. Benutzer des Programms wählen eines der beiden Protokolle aus, welches zum Versenden verwendet wird. Von NEVOT können Pakete beider Protokolle empfangen werden, für mehrere Teilnehmer ist eine Einigung auf ein Protokoll dadurch nicht erforderlich. VAT wird in dieser Arbeit nicht betrachtet. Die Tranportprotokolle bilden die Schnittstelle zwischen NEVOT und dem verwendeten Netzwerk.

TKAUDIO
TKAUDIO ist eine einfache Schnittstelle zum Audiosystem. Mittels TKAUDIO können Lautstärke und Kanäle des Audiosystems eingestellt werden.

3.3 Implementierung

Die Implementierung eines Audio-Konferenzsystems erfordert eine besondere Programmstruktur, um die Verarbeitung und Übertragung von Audio zu ermöglichen. Im folgenden wird auf zwei Besonderheiten der Implementierung eingegangen. Unterschiede zwischen Audiouhr und Systemuhr spielen eine maßgebliche Rolle bei der Portierung von NeVoT, diese wird in Kapitel 7 vorgestellt. Das Prinzip der unterbrochenen Übertragung wurde bereits in Kapitel 2.2.1 erläutert. In Kapitel 3.3.2 wird die konkrete Umsetzung in NeVoT dargestellt.

3.3.1 Audiouhr und Systemuhr

NeVoT versendet Audio in Paketen, deren Länge unabhängig von der Kodierung einer gleichbleibenden Zeitdauer entsprechen. Als Standardwert sind 20 Millisekunden vorgegeben, der Wert kann jedoch verändert werden. Die Hardware zur Steuerung von Audio Ein- und Ausgabe besitzt eine eigene Uhr für die Abtastung der Audiosamples **(Audiouhr)**. Die Geschwindigkeit dieser Uhr kann von jener der Systemuhr abweichen. Werden Audiopakete von einer Anwendung im Takt der Systemuhr an das Audiosystem zur Ausgabe geleitet, dann entsprechen die Zeitintervalle der Pakete nicht den Zeitintervallen der Audiouhr, wenn die

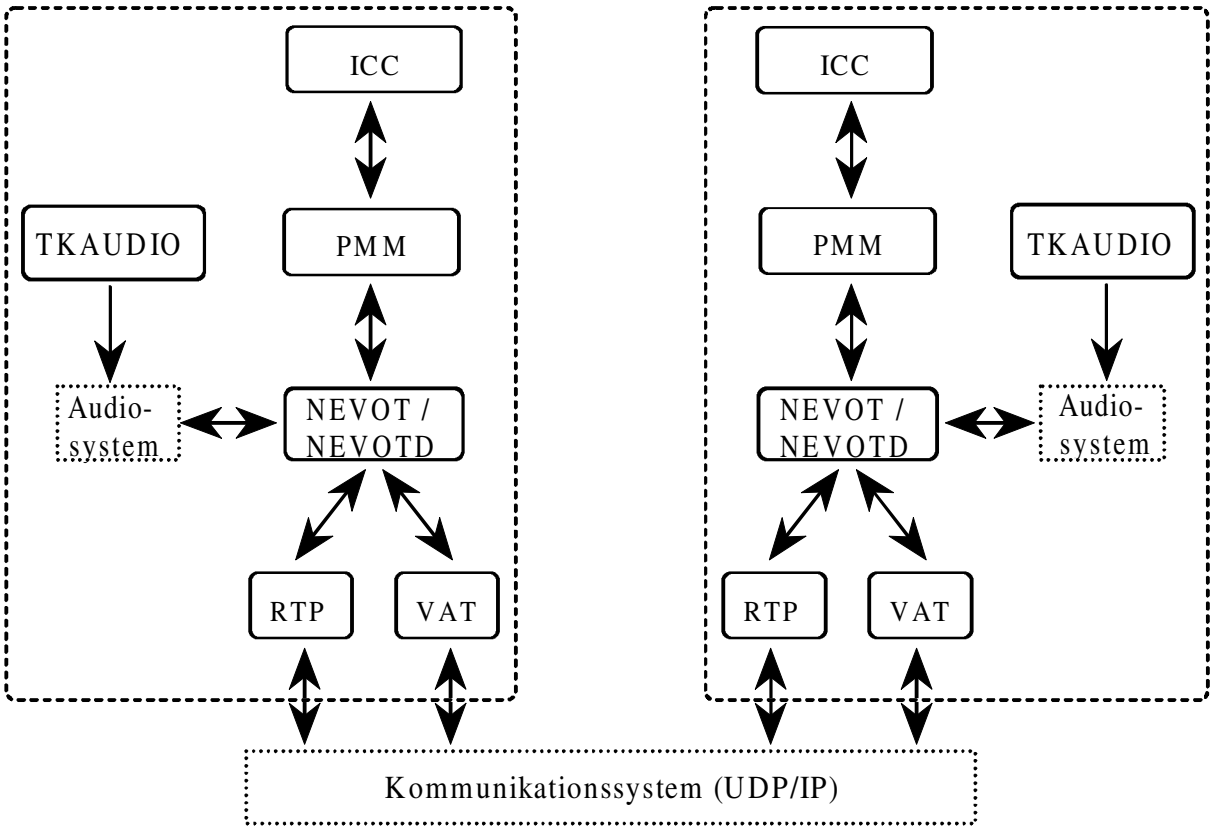


Abbildung 3-3: NeVoT Programmkomponenten

NEVOT/NEVOTD
NEVOT/NEVOTD ist die Hauptkomponente des Programmpakets. Beide Programme besitzen einen identischen Funktionsumfang, jedoch werden die Funktionen von NEVOTD (NEVOT-Deamon) ohne graphische Benutzerschnittstelle angeboten. NEVOT bietet eine Benutzerschnittstelle, die mittels den Kommandosprachen Tk und Tcl [Oust94] realisiert ist. Pro Rechner ist nur eine Instanz, entweder NEVOT oder NEVOTD, notwendig. Im folgenden steht NeVoT für das gesamte Programmpaket, NEVOT stellvertretend für eine der beiden Komponenten NEVOT oder NEVOTD. Der letzte Fall wird lediglich bei der Betrachtung der Komponenten gewählt. NEVOT verwaltet mehrere Konferenzen und steuert den Netzwerkzugriff und die Verarbeitung des Audios für alle Konferenzen. Audio wird von der Audio-Hardware entgegengenommen und an das Netzwerk weitergeleitet. Vom Netzwerk kommendes Audio wird empfangen und ausgegeben.

ICC
ICC (Interface Conference Controller) ist eine allgemeine Kontrollschnittstelle zu Konferenzsystemen. Sie wurde im Rahmen von NeVoT entwickelt, kann aber auch für andere Konferenzsyteme verwendet werden. ICC dient zum Öffnen einer Konferenz und zum Darstellen von Teilnehmern. Jede Konferenz benötigt einen eigenen ICC.

Abbildung 3-2 zeigt die Benutzerschnittstelle von NeVoT. Im oberen Teil ist die Schnittstelle des Programms dargestellt, anhand derer eine Konferenz initiiert, Audio aufgenommen oder abgespielt und die oben erläuterten Parameter eingestellt werden können. Im unteren Teil werden die zu einer Konferenz gehörenden Teilnehmer dargestellt. Dabei entspricht der erste Eintrag immer dem lokalen Benutzer der Konferenz.

Abbildung 3-2: NeVoT Benutzerschnittstelle

3.2 Komponenten

NeVoT ist ein Programmpaket, welches aus mehreren Komponenten besteht. Diese Komponenten, sowie deren Interaktionen untereinander, werden im folgenden vorgestellt.

Abbildung 3-3 zeigt den Aufbau des Programmpakets NeVoT. Dargestellt sind alle beteiligten Komponenten zweier Teilnehmer. Der schematische Aufbau ist in dieser Form auf mehrere Teilnehmer übertragbar.

Konferenz mit diesem Selektor und gibt als Teilnehmer die Rechner der beiden anderen Teilnehmer an. Darüber hinaus führen C und D einen Dialog. Beide starten eine Konferenz mit dem Selektor 3556 und geben den jeweiligen Partner als Teilnehmer an. Das Szenario ist in Abbildung 3-1 graphisch dargestellt.

Das Szenario und die Erläuterung der Vorgehensweise für die Durchführung einer Konferenz zeigen unterschiedliche Anwendungsmöglichkeiten von NeVoT. Sie sind in der folgenden Tabelle zusammengefaßt.

Verbindungstyp	Anwendungsbeispiel
Punkt-zu-Punkt	Dialog (Telefon).
Punkt-zu-Mehrpunkt	Vortrag.
Mehrpunkt-zu-Punkt	Raumüberwachung mehrerer Teilnehmer.
Mehrpunkt-zu-Mehrpunkt	Konferenz mit mehreren Teilnehmern.

Tabelle 3-1: NeVoT Anwendungsmöglichkeiten

Für jede Konferenz können bei einem Teilnehmer mehrere Parameter festgelegt werden. Dazu gehören:

- Die Kodierung der Audiodaten und die Abtastfrequenz. Sie legen die Qualität der Audiosignale für die lokale Ein- und Ausgabe fest.
- Die Anzahl der Kanäle. Die Anzahl der Kanäle legt fest, wieviele der verfügbaren Kanäle des Audiosystems für Ein- und Ausgabe verwendet werden.
- Das Transportprotokoll. NeVoT bietet die Auswahl zwischen den zwei Tranportprotokollen RTP und VAT (siehe unten) zur Übertragung der Daten.
- Die Quellenbeschreibung. Die Quellenbeschreibung beinhaltet Angaben über den Teilnehmer. Dazu gehören Name, Ortsangabe und Telefon. Die Daten der Quellenbeschreibung werden an die anderen Teilnehmer als Kontrollinformation übermittelt.

Während einer Konferenz tauschen Teilnehmer Informationen aus, die in Form von Audio übertragen werden. Über ein an den Computer angeschlossenes Mikrophon werden Audiosignale aufgenommen, digitalisiert, übertragen und bei anderen Teilnehmern ausgegeben. Ebenso können Audiodateien abgespielt werden, diese werden dann wie über Mikrophon aufgenommene Signale übertragen und ausgegeben. Ein Teilnehmer kann die Eingabe von Audio vorübergehend deaktivieren. Ebenso kann die Ausgabe von Audio deaktiviert werden, falls ein Benutzer vorübergehend kein Audio empfangen möchte. Das Programm führt Statistik über den Datentransport von und zu den Teilnehmern. Dazu gehören u.a. die Anzahl der empfangenen und erwarteten Datenpakete und die Anzahl der empfangenen Kontrollpakete. Diese Informationen werden laufend aktualisiert und können vom Benutzer abgerufen werden. Das Beenden einer Konferenz erfolgt durch die Auswahl der Schaltfläche „Leave“. Den anderen Teilnehmern wird daraufhin mittels Kontrollinformation das Ausscheiden des Teilnehmers angezeigt.

3.1.2 Funktionsweise

Die Funktionsweise von NeVoT wird in den nächsten Abschnitten erläutert. Die Reihenfolge der dargestellten Funktionalität entspricht den Schritten zur Durchführung einer Konferenz.

Nach dem Starten des Programms NeVoT bietet sich dem Anwender die Möglichkeit, mehrere Parameter des Programms einzustellen. Zu diesen Parametern gehören Besonderheiten bezüglich Audio, wie die Art und Weise der Lautstärke-Darstellung oder die Aktivierung einer automatischen Lautstärkenkontrolle während einer Konferenz. Ein weiterer Parameter ermöglicht die Aufzeichnung von Informationen über den Ablauf des Programms (Debugging-Informationen), um das Verhalten des Programms nachzuvollziehen. NeVoT kann mehrere Konferenzen innerhalb eines Computersystems steuern, es ist nur eine Instanz des Programms erforderlich. Die hier erläuterten Parameter beziehen sich auf das Programm NeVoT, sie gelten demnach für alle Konferenzen, die gleichzeitig mittels NeVoT auf einem System durchgeführt werden.

Innerhalb des Programms NeVoT kann durch Auswahl der Schaltfläche „Join“ eine Konferenz eröffnet oder einer bestehenden Konferenz beigetreten werden. Hierfür ist die Angabe von zwei Parametern notwendig.

- Angabe der Adressaten. Jeder Teilnehmer legt fest, an welche Adressaten Audio übertragen wird. Es kann eine Liste von Adressen, eine Liste von Rechnernamen oder eine Gruppenkommunikations-Adresse angegeben werden.
- Festlegung des Selektors **(Ports)**. Der Selektor dient zur Unterscheidung mehrerer Konferenzen. Als Standard ist der Wert 3456 vorgesehen.

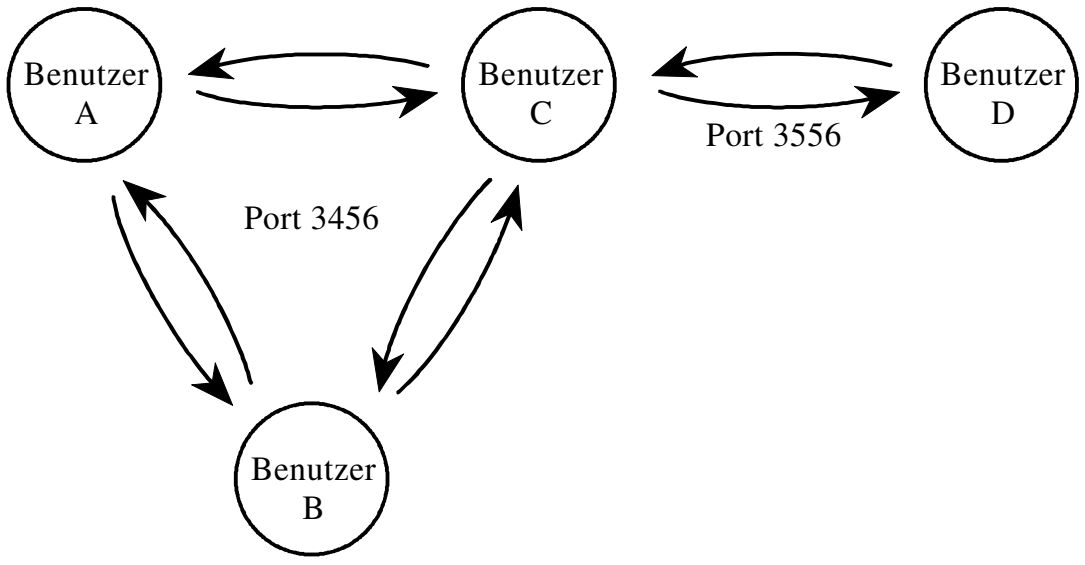


Abbildung 3-1: NeVoT Szenario

Das folgende Beispiel dient zur Erklärung des Adressierungsschemas. Die Computersysteme der Benutzer A, B, C und D sind durch ein Netzwerk verbunden. Benutzer A, B und C haben eine Konferenz mit dem Selektor 3456 festgelegt. Dazu initiiert jeder der drei Benutzer eine

3 Das Network Voice Terminal (NeVoT)

In Kapitel 3 wird das Audio-Konferenzsystem NeVoT [Schu95b] vorgestellt. Dabei wird zunächst das Programm anhand seiner Funktionen erläutert. Anschließend erfolgt eine Betrachtung der Systemkomponenten. Bestandteil des letzten Teils dieses Kapitels sind ausgewählte Aspekte der Implementierung.

3.1 Überblick

In den folgenden zwei Unterkapiteln wird NeVoT anhand eines Überblicks über das Programm und der Erläuterung der Funktionsweise vorgestellt.

3.1.1 Einleitung

Das Audio-Konferenzsystem NeVoT ermöglicht den Austausch von Audio oder Audiodateien zwischen mehreren Teilnehmern. Darüber hinaus erfolgt mittels NeVoT eine Darstellung von Kontrollinformationen bezüglich der Konferenz. Endpunkte der Konferenz sind Computersysteme, die zur Aufnahme und Ausgabe von Audio und zur Präsentation der später vorgestellten Kontrollinformationen dienen.

Für die Verwendung von NeVoT ist ein Arbeitsplatzrechner oder ein Personal Computer notwendig. Des weiteren sind die Peripheriegeräte Mikrophon und Lautsprecher erforderlich. NeVoT gibt es in mehreren Versionen, die unterschiedlichen Computer- und Betriebssystemen angepaßt sind. Unterstütz werden die Systeme SunOS 4.1, Sun Solaris 2.3, SGI Irix 4.0 und Hewlet-Packard HP-UX 9.0.1 [Schu95b; Kapitel 2]. Die Anpassung von NeVoT an Alpha-Arbeitsplatzrechner der Firma Digital mit dem Betriebssystem OSF/1 [OSF91] ist Bestandteil dieser Arbeit.

Eine weitere Grundlage für die Verwendung von NeVoT ist eine Vernetzung der beteiligten Computer. NeVoT wurde unter anderem für den Einsatz im Internet [Klau94] entwickelt. Dies hat den Vorteil, eine gut ausgebaute Infrastruktur nutzen zu können. Demgegenüber steht eine schlechtere Qualität als bei anderen (oftmals kommerziellen) Netzen. Insbesondere Universitäten und andere Forschungseinrichtungen sind über das Internet verbunden. Neben dem Internet können auch andere, beispielsweise lokale Netzwerke, verwendet werden.

NeVoT wurde von Henning Schulzrinne (URL: http://www.fokus.gmd.de/htbin/info/step/hgs) von der Gesellschaft für Mathematik und Datenverarbeitung, Forschungsgruppe offene Kommunikationssystem (GMD Fokus) in Berlin, entwickelt. NeVoT darf frei kopiert werden und ist auf mehreren Systemen im Internet verfügbar (z.B. gaia.cs.umass.edu im Verzeichnis /pub/hgschulz/nevot). Neben ausführbaren Versionen stehen auch die Quellen zur Verfügung.

Für die Durchführung der Ressourcen-Reservierung sind folgende Elemente erforderlich:

- Ein Reservierungsprotokoll, welches Reservierungsinformationen zwischen den Teilnehmern austauscht. Dazu gehört u.a. ein Aushandeln der Ressourcen anhand der Anforderungen und der auf dem Übertragungsweg verfügbaren Ressourcen.
- Ressourcenverwalter, welche die Reservierung in Zwischen- und Endsystemen umsetzen. Eine Aufgabe eines Ressourcenverwalters ist es beispielsweise, eine Paketverarbeitung in einem Zwischensystem innerhalb eines Zeitrahmens durchzuführen.

Durch Ressourcen-Reservierung ist es möglich, die Qualität einer Audio-/Videoübertragung zu garantieren.

kommt es zu Effizienzeinbußen bei der Auslastung des Netzwerks. Diese Übertragungsraten und -techniken mögen für manche, nicht jedoch für alle verteilten multimedialen Anwendungen ausreichend sein.

Bereits heute werden Netzwerke eingesetzt, welche eine Netzwerkkapazität von ca. 622 MBits/s besitzen. Diese Netzwerke arbeiten nach einem asynchronen Transfer-Modus (ATM) [HäHu91]. Bei ATM werden einzelne Zellen über das Netz übertragen. Die Zellen bestehen aus einem Zellenkopf der Größe 5 Bytes und einem Nutzdatenteil der Länge 48 Bytes. Die Zellen werden mit Daten unterschiedlicher Anwendungen gefüllt. Die Belegung der Zellen erfolgt entsprechend der Nachfrage, durch dieses Verfahren ist eine gute Auslastung des Netzwerks möglich. ATM-Netzwerke sind mittels Glasfasertechnik realisiert. Durch die Verwendung von Glasfasertechnik entstehen aufgrund der Lichtgeschwindigkeit in Glasfasern zudem geringe Verzögerungszeiten zwischen dem Versenden und dem Empfangen von Daten.

Leistungsfähige Netzwerke und geringe Verzögerungszeiten bilden eine Grundlage für die Übertragung von Audio und Video. Bei einer Aufteilung von Audio und/oder Video in Pakete zur Übertragung entstehen weitere Anforderungen an Netzwerke. In der Regel werden von der Anwendung Pakete gleicher Größe gebildet. Die Pakete entsprechen einem Zeitpunkt innerhalb des Audio-/Videostroms. Durch die Verwendung von Paketen gleicher Größe entsprechen die Zeiten zwischen den Paketen immer einer konstanten Zeitdauer. Bei der Übertragung dieser Pakete ist es für die Realisierung der Ausgabe wünschenswert, wenn die Zeitpunkte zwischen den Paketen beim Empfang identisch sind mit den Zeitpunkten zwischen den Paketen beim Versenden. Dieser Sachverhalt wird als **isochrone Datenübertragung** bezeichnet. Für die Realisierung einer isochronen Datenübertragung gibt es bei den oben vorgestellten zellenbasierten Netzwerken folgende Möglichkeit: Zu regelmäßigen Zeitpunkten wird eine gewisse Anzahl von Zellen reserviert, welche nur von einer Anwendung verwendet werden dürfen. Treten Verzögerungsschwankungen **(Jitter)** bei der Übertragung auf, so muß vor der Ausgabe der Daten beim Empfänger eine Zwischenspeicherung erfolgen, um eine kontinuierliche Ausgabe der Daten zu ermöglichen. Diese Zwischenspeicherung ist mit einer Verzögerung verbunden, welche bei interaktiven verteilten multimedialen Anwendungen nicht wünschenswert ist.

2.3.2 Ressourcen-Reservierung

Im letzten Kapitel wurden leistungsfähige Netzwerke zur Übertragung von Audio und Video kurz dargestellt. Aufgrund der Verwendung von Netzwerken durch mehrere Benutzer kann, bei entsprechender Auslastung, eine Qualitätseinbuße (z.B. Paketverlust) bei der Übertragung stattfinden. Es existieren bereits mehrere Ansätze zur Reservierung von Ressourcen [Topo90], [ZBE&94]. Durch die Verwendung von Ressourcen-Reservierung können Garantien bezüglich einer Qualität einer Datenübertragung gegeben werden. Parameter der Anforderungen können beispielsweise erforderliche Übertragungskapazität, maximale Verzögerung und maximale Verzögerungsschwankung sein.

Abbildung zwischen der vom Benutzer geforderten Qualität und den dafür erforderlichen Parametern für die Reservierung von Netzwerkkapazität und Rechenleistung in Zwischen- und Endsystemen erfolgen.

- Bereitstellung von Kontrollmechanismen zur Übermittlung von Information zwischen Endsystemen. Die Übermittlung von Kontrollinformationen ermöglicht eine umfassende Information der Anwender bezüglich der Charakteristika der Übertragung und der Teilnehmer (quality of service monitoring). Durch diese Mechanismen ist es für Anwender möglich, selbständig Konsequenzen aus den dargestellten Informationen zu ziehen. Ein Benutzer kann beispielsweise eine Konferenz abbrechen, falls die Qualität nicht seinen Ansprüchen genügt.
- Effiziente Implementierung. Eine effiziente Implementierung ist eine notwendige Bedingung für die Realisierung von verteilten multimedialen Anwendungen. Die höchste Stufe der Qualität eines Dienstes besteht in der Garantie von Rechen- und Kommunikationsleistung innerhalb eines gewissen Zeitrahmens.

Bei der Analyse der vorgestellten Transportprotokolle wird nur auf den zweiten Punkt (Bereitstellung von Kontrollmechanismen) genauer eingegangen, Ressourcen-Reservierung wird kurz in Kapitel 2.3.2 erläutert. Implementierungsspezifische Aspekte sind nicht Bestandteil der Arbeit.

In [ChTo92] wird die Qualität von (kontinuierlichen) Medien folgendermaßen festgelegt: Die Qualität besitzt zwei Dimensionen, eine räumliche und eine zeitliche. Die räumliche Dimension bezeichnet den Grad der Kompression von kontinuierlichen Medien bzw. die Datengröße. Die zeitliche Dimension bezeichnet die Bildwiederholfrequenz bei Video oder die Abtastfrequenz bei Audio (Periodizität der Daten). Durch diese Begriffsverwendung ist es möglich, Medien eine Qualität zuzuordnen.

2.3 Netzunterstützung

Die letzten beiden Kapitel haben die Grundlagen und einsetzbare Protokollmechanismen der Transport- und Sitzungsschicht für den Transport von Audio und Video erläutert. Weiterhin sind für die Übertragung von Audio und Video leistungsfähige Netzwerke erforderlich. Diese Aspekte sind nicht Bestandteil dieser Arbeit, der Vollständigkeit halber wird jedoch im folgenden ansatzweise erläutert, welche Möglichkeiten hierfür existieren.

2.3.1 Leistungsfähigkeit

In Kapitel 2.1 wurden für die Übertragung von Audio und Video erforderliche Übertragungsraten vorgestellt. Insbesondere die Übertragung von Video erfordert hohe Übertragungsraten, welche nicht unter ca. einem MBit/s liegen [Zitt93; Kapitel 1]. Aus diesen Gründen sind leistungsfähige Netzwerke erforderlich, die einen Transport von Audio und Video ermöglichen. Die in vielen Bereichen eingesetzte Netzwerktechnik CSMA/CD (Carrier Sense Multiple Access with Collision Detection) [Hals92] bietet lediglich Übertragungsraten von ca. 10 MBits/s. Bei CSMA/CD kann es, wie der Name bereits andeutet, zu Kollisionen auf dem Netzwerk kommen. Durch eine dadurch erforderliche Wiederholung der Übertragung

der übermittelten Daten: Bei einer Multimediakonferenz werden Audio/Video beim Empfänger oftmals nur ausgegeben und nicht aufgezeichnet oder weiterverarbeitet. Wäre eine Weiterverarbeitung notwendig, so würde bei mehrmaliger Übertragung mit ständigem Ignorieren der Fehler die Qualität von Audio und Video deutlich sinken. Durch konventionellen Dateitransfer übertragene Daten werden oftmals weiterverarbeitet und weiter übertragen. Hier ist das Ignorieren von Fehlern nicht möglich.

2.2.4 Dienstgüte

In den folgenden Abschnitten wird zunächst Dienstgüte im Kontext von verteilten multimedialen Anwendungen erläutert. Daraufhin erfolgt ein Überblick über Möglichkeiten, mit denen auf die beschriebene Dienstgüte Einfluß genommen werden kann.

Für einen Anwender zeigt sich die Dienstgüte einer Audio oder Video übertragenden Anwendung u.a. in der Verzögerungszeit zwischen Sender und Empfänger und der Qualität von Audio und Video. Während die Verzögerungszeit von Anwendern nicht direkt beeinflußt werden kann, so kann die Qualität von Audio und Video durch die Wahl von unterschiedlichen Kodierungen beeinflußt werden. Für höhere Qualität ist bei der Übertragung eine größere Kapazität der Netzwerke notwendig, da mehr Daten übertragen werden müssen. Bei kommerziellen Netzwerken ist mit einer höheren Qualität auch ein höherer Preis für die Übertragung verbunden, wenn eine Abrechnung der Nutzung nach Netzwerk-Kapazität erfolgt. In diesem Fall ist es wünschenswert, daß der Anwender in geeigneter Weise Information über das Verhältnis zwischen Kosten und Qualität erhält, um Einfluß darauf zu nehmen.

Der transportabhängige Teil eines Kommunikationssystems (Schichten 1-4 des OSI-BRM) hat ebenso maßgebliche Auswirkung auf die Qualität einer verteilten multimedialen Anwendung. Maßstäbe sind Kapazität, Bitfehlerrate, Anzahl der verlorengegangenen Pakete pro Zeiteinheit und Verzögerung von Paketen. Zur Verbesserung der Dienstgüte können Protokolle eingesetzt werden, mit denen eine Reservierung von Ressourcen in Netzwerken erfolgen kann. Mit solchen Protokollen und weiteren Mechanismen (u.a. Ressourcenverwaltern in den Systemen) sind Garantien bezüglich Diensten möglich. Reservierung von Netzwerk-Ressourcen und Garantien von Parametern sind nicht Bestandteil der in dieser Arbeit vorgestellten Protokolle RTP und AVXP. Eine Erweiterung dieser Protokolle um Unterstützung von Ressourcen-Reservierung wäre wünschenswert. Die Protokolle ST-II [Topo90] und RSVP [ZBE&94] beinhalten Konzepte zur Ressourcen-Reservierung.

Protokolle der Transportschicht können gemäß den obigen Ausführungen Dienstgütemerkmale u.a. auf folgende Weise berücksichtigen:

- Nutzung von Ressourcen-Reservierung. Die von Protokollen der Vermittlungsschicht angebotene Funktionalität muß von Transportprotokollen in Anspruch genommen werden. Dies bedeutet, daß Anfragen entweder von Transportprotokollen getätigt oder von Anwendungen angeforderte Anfragen durchgereicht werden. Bei der Anforderung von Ressourcen-Reservierung für eine Übertragung von Audio oder Video muß eine

des Zeitstempels den Zeitstempel 220 hat, so kann die zusammenhängende Audiosequenz beginnend mit *Paket 9* mit diesem Mechanismus nicht erkannt werden.

2.2.3 Fehlerbehandlung

In diesem Kapitel werden mehrere Arten der Fehlerbehandlung betrachtet. Anhand der verschiedenen Verfahren wird dargestellt, welche der Mechanismen bei der Übertragung von multimedialen Daten nur bedingt einsetzbar sind.

Aufgrund von Pufferüberlauf in Zwischensystemen, Störungseinflüssen auf Leitungen und anderen Problemen kann es zu Verlust, Verfälschung, Verzögerung und Duplizieren von Pakten kommen. Bei konventionellem Dateitransfer muß bei Eintreten einer der oben aufgeführten Fälle eine Fehlerbehandlung eingeleitet werden. Diese Fehlerbehandlung kann sehr zeitaufwendig sein (z.B. Wiederholung einer Übertragung) und ist somit für Übertragung von multimedialen Daten mit Realzeit-Anforderungen unpraktikabel. Duplikate von Paketen können beim Empfänger mit geringem Aufwand erkannt werden (Überprüfung der Sequenznummern). Anders ist dies bei Verlust, Verfälschung und Verzögerung.

Als vorausschauende Maßnahme gegen Verfälschung und Verlust von Daten kann *Vorwärtsfehlerbehebung* (forward error correction; **FEC**) eingesetzt werden. Dabei werden zu den Nutzdaten weitere Daten (Redundanz) hinzugefügt, mit deren Hilfe eine Rekonstruktion der Nutzdaten im Falle einer Verfälschung möglich sein kann. Bis zu welchem Grad der Verfälschung Nutzdaten rekonstruiert werden können, hängt von der hinzugefügten Redundanz ab. Vorwärtsfehlerbehebung hat eine weitere Verzögerung zur Folge: Geht ein Paket mit der Nummer n verloren oder verspätet sich, so kann zwar das Paket n+1 Daten beinhalten, die eine Rekonstruktion von Paket n teilweise möglich macht, die Verfügbarkeit des Pakets n beim Empfänger verschiebt sich allerdings noch weiter nach hinten. Soll hingegen Paket n-1 Daten von Paket n enthalten, so muß das Absenden beim Sender bis zur Verfügbarkeit aller Daten verzögert werden. Weiterführende Betrachtung zur Vorwärtsfehlerbehebung sind Bestandteil von Kapitel 6.2.2.

Bei Bitfehlern in Paketen oder bei Paketverlusten kann von der Transportschicht oder der darüberliegenden Anwendung eine *Wiederholung der Übertragung* initiiert werden. Da dieses Verfahren durch das Erkennen des Fehlers und die Anforderung der erneuten Übertragung wiederum mit hoher Verzögerung verbunden ist, ist die Übertragungswiederholung bei verteilten Realzeit-Anwendungen praktisch nicht einsetzbar.

Im Gegensatz zum konventionellen Dateitransfer kann das *Ignorieren eines Fehlers* bei der Übertragung multimedialer Daten in Betracht gezogen werden. Dies liegt daran, daß beim Empfänger durch das Fehlen eines Medienfragments bei Sprache nur eine minimale Sprechpause entsteht, bzw. bei Video ein durch mehrere Bilder dargestellter Bewegungsablauf lediglich sprunghaft erscheint. Einen Hinweis auf die Möglichkeit des Ignorierens von Bitfehlern oder verlorengegangenen Paketen gibt ein Vergleich zwischen konventionellem Dateitransfer und der Übertragung von multimedialen Daten bezüglich der Weiterverarbeitung

Zeit zwischen einer Einheit beim Empfänger und einer Einheit beim Sender. Besteht eine zusammenhängende Audiosequenz aus Fragmenten der Länge 20 Millisekunden, dann stehen die Ausgabeeinheiten der Synchronisationseinheit bezüglich Sender und Empfänger in relativer zeitlicher Abstimmung.

Folgende Problematik entsteht bei der absoluten zeitlichen Abstimmung. Werden Pakete immer zu einem festen Zeitpunkt nach Erstellung ausgegeben, dann dürfen die Uhren bei Sender und Empfänger keine Abweichung haben. Kommt es zu einer Abweichung, kann dies zum Über- oder Unterlauf des Empfangspuffers kommen, wenn auszugebende Pakete beispielsweise in einem Ringpuffer zwischengespeichert werden.

Für die Berechnung des Ausgabezeitpunkts von Medien bei einem Empfänger werden in [Schu93; Kapitel 3.6.1] drei Methoden vorgestellt. Basis für diese Methoden bilden die Variablen Erstellungszeitpunkt, variable Verzögerung (Zwischensysteme und Netzwerkzugriff), feste Verzögerung (Ausbreitungsgeschwindigkeit und Protokollbearbeitung), Ankunftszeitpunkt und Ausgabezeitpunkt. Die Erläuterung dieser Verfahren würde den Rahmen dieses Kapitels sprengen, aus diesem Grund wird darauf verzichtet.

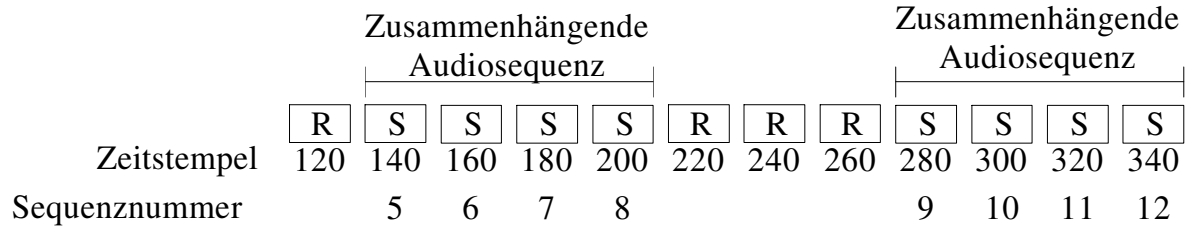
Möglichkeiten des Einsatzes von Synchronisationseinheiten wurden durch die ausgesetzte Übertragung motiviert. Notwendig für die konkrete Anwendung ist dazu das Erkennen von Synchronisationseinheiten beim Empfänger. Zwei Methoden wurden von der IETF Arbeitsgruppe „Audio und Video Transport“ in [Schu93] vorgeschlagen, diese Möglichkeiten können alternativ oder gemeinsam verwendet werden.

- Synchronisationsbit: Dieses Bit im Paketkopf der Audio-/Videodaten zeigt den Beginn einer neuen Synchronisationseinheit an. Probleme treten hier bei nichtgesicherter Übertragung (z.B. UDP [Davi88]) auf. Falls ein Paket verloren geht, kann das darauffolgende Paket nicht als Beginn einer Synchronisationseinheit erkannt werden. Da bei der Übertragung von Audio und Video oft eine unzuverlässige Übertragung gewählt wird, sollte entweder die nächste Möglichkeit gewählt werden oder eventuell das Paket mittels Vorwärtsfehlerbehebung (Kapitel 6.2.2) gesichert werden.
- Zeitstempel und Sequenznummer: Ein Paket wird als Beginn einer Synchronisationseinheit festgelegt, wenn es
  a) den größten Zeitstempel und die größte Sequenznummer bisher beinhaltet und
  b) die Differenz der Zeitstempel zwischen einem empfangenen und dem vorherigen Paket, gerechnet in Sequenznummern, die Differenz der Sequenznummern überschreitet.

  In Abbildung 2-4 wird die Synchronisationseinheit beginnend bei *Paket 9* folgendermaßen erkannt: Die Differenz der Zeitstempel zwischen *Paket 9* und *Paket 8* entspricht 4 Sequenznummern (80 Millisekunden), damit beginnt mit *Paket 9* eine neue zusammenhängende Audiosequenz.

  Komplikationen kann ein zu kleines Feld des Zeitstempels hervorrufen: Falls eine Ruhepause zwischen *Paket 8* und *Paket 9* genau so lange ist, daß *Paket 9* nach Überlauf

so ist bei einer Feldgröße von 10 Bit alle 20,48 Sekunden der gleiche Wert erreicht. Bei zu kurzen Zeitbereichen kann es zu Komplikationen kommen, wie im letzten Abschnitt dieses Unterkapitels erläutert. In Abbildung 2-4 ist eine Folge von Audiofragmenten dargestellt. Unter den Paketen sind Zeitstempel jedes Fragments (in Millisekunden) und die Sequenznummer angegeben. Übertragen werden nur Sprachpakete, Ruhepakete erhalten daher keine Sequenznummer.



R Ruhepaket (ohne Sequenznummer), wird nicht übertragen.

S Sprachpaket (mit Sequenznummer), wird übertragen.

Abbildung 2-4: Fragmentiertes Audio

Zeitstempel können unterschiedlich verwendet bzw. interpretiert werden. Möglichkeiten hierfür sind:

- Entsprechend der Frequenz des Abtasters, der Audio/Video beim Empfänger abtastet (Sample-Uhr).
- Gemäß NTP-Zeitstempel [Mill92] oder einer Teilmenge davon.
- In Mikro- oder Millisekunden.
- Als Paketintervall: Jedes neue Paket erhöht den Zeitstempel. Diese Vorgehensweise entspricht einer einfachen Sequenznummer.

Für die weitere Betrachtung von Synchronisation ist eine Unterteilung der zu synchronisierenden Einheiten hilfreich.

- **Ausgabeeinheit** (playout unit): Gruppe von Paketen, die zu einem Ausgabezeitpunkt gehören. Bei Sprache entspricht dies in der Regel einem Sprachfragment, bei Video kann es eine Menge von Teilbildern sein, die zu einem Gesamtbild gehören.
- **Synchronisationseinheit** (synchronization unit): Sie besteht aus mehreren Ausgabeeinheiten, die alle die gleiche feste Verzögerung zwischen Erstellung und Ausgabe besitzen. Die Verzögerung kann zwischen den Synchronisationseinheiten variieren. Ein Beispiel ist die in Kapitel 2.2.1 vorgestellte zusammenhängende Audiosequenz.

Synchronisations- bzw. Ausgabeeinheiten können bezüglich Sender und Empfänger in unterschiedlicher Relation stehen. Bei der relativen zeitlichen Abstimmung ist der Abstand zwischen den Einheiten bei Sender und Empfänger identisch, gemessen bezüglich der Sample-Uhr. Bei der absoluten zeitlichen Abstimmung besteht eine feste Abstimmung bezüglich der

nicht durch Ruhepakete unterbrochen sind. Ruhepaket beinhalten sprachlich irrelevante Informationen. Dazu zählen Stille oder vernachlässigbare Hintergrundgeräusche. Für ihre Entstehung gibt es zwei Möglichkeiten: Ein Sprecher kann vorübergehend die Eingabe von Audio deaktivieren, wobei dennoch Pakete verschickt werden. Eine andere Möglichkeit besteht in der Untersuchung von Paketen, die eventuell als Ruhepakete interpretiert werden können. Dafür ist ein Maß für den Durchschnitt der Amplitude aller Samples eines Paketes notwendig. Unterschreitet dieser Wert einen vorgegebenen Grenzwert, so wird das Audiofragment als Ruhe interpretiert und nur in Ausnahmefällen übertragen (Kapitel 3.3.2). Bei entsprechender Synchronisation zwischen Sender und Empfänger (Kapitel 2.2.2) ist es ausreichend, nur die Pakete der zusammenhängenden Audiosequenz zu übertragen. Abbildung 2-4 in Kapitel 2.2.2 zeigt ein Beispiel einer zusammenhängenden Audiosequenz.

2.2.2 Synchronisation

Die Übertragung kontinuierlicher Daten erfordert beim Empfänger eine Festlegung der Ausgabezeitpunkte des kompletten Mediums oder der Medienfragmente. Abhängig ist der Zeitpunkt u.a. vom Zeitpunkt des Sendens der Medien beim Sender, wo die Medien erstellt oder gespeichert wurden. Ebenso kann eine zeitliche Abstimmung zwischen mehreren Medien notwendig sein. Diese zeitliche Abstimmung wird als **Synchronisation** bezeichnet. Alleine die Ankunftszeit von fragmentiertem Audio/Video beim Empfänger ist für die Ermittlung des Ausgabezeitpunkts bei vielen verwendeten Kommunikationssystemen unzureichend, da es u.a. aufgrund unterschiedlicher Wegewahl bei der Übertragung zu Schwankungen der Ankunftszeit kommen kann.

Im folgenden wird zunächst auf unterschiedliche Arten der Synchronisation eingegangen. Anschließend werden die dafür notwendigen Mechanismen und Besonderheiten diesbezüglich erläutert.

In [Schu93; Kapitel 3.6] werden drei Arten sich nicht untereinander ausschließender Synchronisation unterschieden:

- **Ausgabe-Synchronisation** (playout synchronization): Ausgabe der Medien zu einem festen Zeitpunkt nach dem Erstellen (zeitversetzt) bei der Quelle.
- **Intra-Medien-Synchronisation** (intra-media synchronization): Ausgabe der Medien zu einem identischen Zeitpunkt bei *allen* Empfängern, z.B. bei verteilten Realzeit-Simulationen.
- **Inter-Medien-Synchronisation** (inter-media synchronization): Synchronisation mehrere Medien bei einem Empfänger, z.B. Video und dazugehöriges Audio.

Formale Mechanismen zur Spezifikation von Synchronisation sind in [Mühl93; Kapitel 6] erläutert.

Die aufgeführten Synchronisationsmethoden benötigen für Medienfragmente eine Zeitinformation **(Zeitstempel)**. Der durch das Feld des Zeitstempels abgedeckte Zeitbereich muß groß genug sein, um Komplikationen nur in sehr seltenen Fällen zuzulassen. Wird der Wert des Zeitstempels bei einem Audiofragment von 20 Millisekunden jeweils um eins erhöht,

- P-Bilder (Predictive Coded Pictures). Bei diesen Bildern wird davon ausgegangen, daß bei aufeinanderfolgenden Bildern oftmals Bildbereiche verschoben sind. Die Kompression basiert auf der Vorhersage von Bewegungen. Bei der Dekompression eines P-Bildes müssen Informationen vorheriger P-Bilder bis zum letzten I-Bild hinzugezogen werden.
- B-Bilder (Bidirectionally Predictive Coded Pictures). Mit diesen Bildern wird der höchste Kompressionsfaktor erreicht. Komprimiert wird anhand einer Vorhersage von vorangegangenen und nachfolgenden Bildern.
- DC-Bilder (DC Coded Pictures). Sie werden für einen schnellen Vorlauf verwendet.

2.2 Protokollunterstützung

Datenübertragung wird zur Kommunikation zwischen Anwendungen bzw. innerhalb verteilter Systeme eingesetzt. Je nach beteiligten Anwendungen unterscheiden sich die Art der Datenübertragung und die benötigten Mechanismen dafür. In diesem Kapitel werden Protokollmechanismen betrachtet, die insbesondere im Kontext von verteilten multimedialen Anwendungen eingesetzt werden oder im Gegensatz zum konventionellen Dateitransfer eine besondere Bedeutung besitzen. Die in diesem Kapitel behandelte Protokollfunktionalität bezieht sich auf die Transportschicht und die Sitzungsschicht des OSI-BRM. Betrachtet werden konzeptionelle Aspekte, die maßgeblich für die Spezifikation von Protokollen sind. Effizienzbetrachtungen der Implementierung von Protokollen sind nicht Bestandteil der Analyse. Die hier vorgestellten Mechanismen bilden die Grundlage für das Verständnis der Protokolle RTP und AVXP, die in späteren Kapiteln vorgestellt werden.

2.2.1 Unterbrochene Übertragung

Bei der Übertragung kontinuierlicher Daten existiert die Möglichkeit, in manchen Fällen die Datenübertragung auszusetzen. Die Grundlagen dieses Verfahrens werden hier dargestellt. Die **unterbrochene Übertragung** kann in den folgenden beiden Fällen eingesetzt werden:

- Der zu übertragende Informationsgehalt ist irrelevant, z.B. Sprechpausen bei Sprachübertragung.
- Der zu übertragende Informationsgehalt ist unverändert gegenüber der zuletzt übertragenen Information, z.B. unveränderte Bilder bei Videoübertragung.
  Auf Sprechpausen in Sprachübertragung wird im folgenden Abschnitt genauer eingegangen, da dieses Konzept bei dem in Kapitel 3 vorgestellten Audio-Konferenzsystem NeVoT verwendet wird.

Mit der unterbrochenen Übertragung können beteiligte Netzwerke vorübergehend geringer belastet werden. Notwendig hierfür ist eine entsprechende Abstimmung bzw. Funktionalität bei Sender und Empfänger, verbunden mit einer Aufwandserhöhung durch die erforderliche Abstimmung in den Endsystemen.

Im folgenden wird fragmentiertes digitalisiertes Audio betrachtet. Hierbei steht der Begriff „zusammenhängende Audiosequenz“ **(talkspurt)** steht für eine Folge von Audiopaketen, die

Nachteil eines erhöhten (Zeit-) Aufwandes gegenüber, der durch den damit verringerten Aufwand bei der Übertragung jedoch i.d.R. kompensiert wird. Es gibt jedoch auch Anwendungen, bei denen eine Kompression seitens der Anwender aufgrund der hohen Sensibilität der Daten nicht erwünscht ist. Als Beispiel hierfür seien verteilte medizinische Anwendungen genannt, etwa die Übertragung von Röntgenbildern, Computertomographien oder Magnet-Resonanz-Tomographien.

Kompression und Dekompression unterscheiden sich in folgenden Merkmalen [Mühl93; Kapitel 2]:

- Der *Kompressionsfaktor* ist ein Maß für das Verhältnis der Größe der unkomprimierten Daten zu den komprimierten Daten.
- Sind Daten nach einer Kompression und Dekompression in ihrem Originalzustand, so handelt es sich um eine *verlustfreie Kompression*, anderenfalls liegt *verlustbehaftete Kompression* vor. Verlustbehaftete Kompression kann beispielsweise bei Bildern eingesetzt werden, da das menschliche Auge gewisse Farbnuancen nicht unterscheiden kann.
- *Symmetrie* oder *Asymmetrie* drückt das Verhältnis zwischen Kodierzeit und Dekodierzeit aus. Bei Symmetrie ist der Aufwand für Kompression und Dekompression ungefähr identisch, bei Asymmetrie ist der Aufwand ungleich verteilt (Mehraufwand bei der Kompression). Asymmetrie ist bei einmaliger Kompression und mehrmaliger Dekompression bei unterschiedlichen Empfängern sinnvoll.
- Die *Komplexität der Algorithmen* hat Auswirkungen auf den Aufwand der Realisierbarkeit. *Parallelisierbarkeit* der Algorithmen ermöglicht eine effiziente Umsetzung auf Parallelrechnern.

Von der **MPEG (Motion Picture Expert Group)** wurden Verfahren zur Kompression von Video standardisiert [MPEG93]. In diesem Abschnitt soll nur die Idee eines Verfahrens erläutert werden [Stei93; Kapitel 5.7]. Das MPEG1-Verfahren besteht aus drei wesentlichen Teilen: Einem Kompressionsstandard für Video (hier betrachtet), einem Kompressionsstandard für Audio und einem Kompressionsstandard für kombiniertes Audio und Video. MPEG verwendet Ergebnisse der **JPEG (Joint Photographic Expert Group)** [JPEG92] zur Kompression von Einzelbildern. Angestrebt wird eine Datenrate von ca. 1,4 MBits/s, vorgeschrieben ist eine maximale Datenrate von 1.856.000 Bits/s. Ausgelegt ist das Verfahren sowohl für symmetrische als auch asymmetrische Kompression. Der Bildaufbau ist bei MPEG1 vorgeschrieben. Das Bild muß aus drei Ebenen bestehen, einer Luminanz-Ebene und zwei Chrominanz-Ebenen (ähnlich dem YUV-Signal). Die Luminanz-Ebene hat dabei eine höhere Auflösung, da sie die wichtigste Information beinhaltet. Die Luminanzkomponente sollte eine Auflösung von 768 * 576 Pixeln nicht überschreiten. Auch sind mehrere Seitenverhältnisse und Bildwechselfrequenzen festgelegt. Die Idee von MPEG besteht in der Kompression von Bewegungen. Es werden vier Arten von Bildern unterschieden:

- I-Bilder (Intra Coded Pictures). Diese Bilder sind einzeln komprimiert und beinhalten keine Bewegungskompression, sie entsprechen einem Standbild. Die Wiedergabe von MPEG-Video kann nur mit I-Bildern begonnen werden, sie dienen als Anker.

geraden Zeilen beschrieben. Die Bildwiederholfrequenz beträgt in Deutschland 25 Bilder pro Sekunde, d.h. die Halbbilder werden mit 50 Hz beschrieben.

Das Farbsignal ist in der Regel in drei Komponenten zerlegt. Bei rein digitaler Verarbeitung wird meist das RGB Signal verwendet, welches aus den Komponenten Rot, Grün und Blau besteht. Aus diesen Farben lassen sich durch additive Farbmischung alle anderen Farben erzeugen. Die Farbwerte sind normiert, so daß gilt: R+G+B = 1.

Eine andere Möglichkeit bietet das YUV Signal, wie es auch im deutschen Fernsehen verwendet wird. Dabei werden nicht drei Farbwerte übertragen, sondern die Helligkeit (Luminanz)

- Y = (0,30 R + 0,59 G + 0,11 B) und die Farbsignale (Chrominanzen)
- U = (B - Y) * 0,493 und
- V = (R - Y) * 0,877 [Stei93; Kapitel 4.3].

Die Verwendung des YUV Signals hat eine historische Bedeutung: Die Übertragung des Y Wertes ist aus Kompatibilitätsgründen für schwarzweiß Fernsehen notwendig.

Von dem CCIR (Comité Consultatif International des Radiocommunications) gibt es eine Empfehlung für die Kodierung des PAL Systems [CCIR82], [CCIR91]. Diese gibt Aufschluß über die Größenordnung des Speicherbedarfs von unkomprimiertem Video. Das Fernsehbild hat 625 Zeilen mit 864 Pixeln pro Zeile und eine Bildwiederholfrequenz von 25 Bildern pro Sekunde. Daraus ergibt sich eine Abtastfrequenz von 13,5 MHz. Kodiert wird mit dem YUV Signal. Die Empfehlung legt fest, daß die Werte von U und V nur mit halber Frequenz abgetastet werden, da Y den Hauptanteil der Information trägt. Pro Abtastwert sind 8 Bits festgelegt, daraus folgt eine Datenrate bzw. Speicherkapazität von:
13.500.000 * 8 + 2 * 8 * (6.750.000) = 216 MBits/s. Die erforderlichen Datenraten sind demnach gegenüber Audio um einige Größenordnungen höher (vgl. Kapitel 2.1.3).

Aus dem oben aufgeführten Beispiel für die Speicheranforderung von unkomprimiertem Video ist die Problematik der Digitalisierung ersichtlich: Eine ausschließliche Digitalisierung von Video ohne Kompression ist aufgrund der anfallenden Datenmenge kaum durchführbar. Eine Lösungsmöglichkeit bietet das schrittweise Digitalisieren und Komprimieren in Realzeit, d.h. jedes Bild wird digitalisiert und komprimiert, bevor das nächste Bild bearbeitet wird. Aus Effizienzgründen werden die Kompressionsalgorithmen oftmals in Hardware realisiert und bilden zudem oft mit der Hardware zur Digitalisierung eine Einheit. Ein Beispiel ist die J300-Hardware von Digital Equipment [Over94a], welche einen JPEG [JPEG92] Kompressions-Chip der Firma C-Cube verwendet.

2.1.5 Kompression

Die Notwendigkeit von Kompression wurde in Kapitel 2.1.4 motiviert. In diesem Kapitel werden einige Merkmale von Kompression erläutert und ein Beispiel für Videokompression ansatzweise dargestellt. Dies dient als Ansatzpunkt und Beispiel für unterschiedliche Ansätze zur Kompression. Dem Vorteil der Datenreduktion steht bei der Kompression immer der

Beim Erkennungs-Synthese-Verfahren wird mittels Spracherkennung die erforderliche Datenrate reduziert. Die Qualität ist jedoch für kommerzielle Produkte noch nicht ausreichend [Stei93; Kapitel 3.5]. Mit diesem Verfahren lassen sich Datenraten von 50 Bits/s erreichen. Ein Problem besteht in der Generierung des Audiosignals bei der Senke. Abbildung 2-3 zeigt den schematischen Ablauf des Erkennungs-Synthese-Verfahrens.

Einen Ausblick auf erreichbare Bitraten gibt James L. Flanagan [Flan92]. Mit einem dynamische Verfahren (dynamic adaption of perceptual weights) und der Modellierung von Artikulation hält er das Erreichen von 2 KBits/s bei Telefonqualität in der Zukunft für möglich. In einer anderen Darstellung beschreibt er die erreichbare Datenrate in Abhängigkeit von Bits pro Abtastwerten bei Abtastung mit der Nyquist-Abtastfrequenz. Hier gibt er ein Bit pro Abtastwert als möglich an.

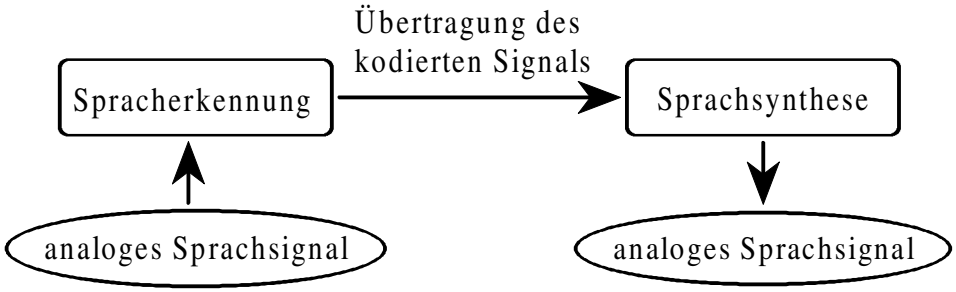


Abbildung 2-3: Erkennungs-Synthese-Verfahren

2.1.4 Video

Neben Audio wird insbesondere Video in verteilten multimedialen Systemen eingesetzt. Nach einer Begriffsdefinition und allgemeinen Aspekten von Bewegtbildern wird kurz auf Eigenschaften von Bildsignalen eingegangen. Diese bilden die Voraussetzung für das Verständnis einer vorgestellten Kodierung von Bewegtbildern. Anhand der Videokodierung wird die erforderliche Datenrate für die Übertragung von Video verdeutlicht.

Begriff **Video** wird im folgenden als Synonym für Bewegtbilder verwendet. Dies bedeutet, daß Video immer unabhängig von eventuell dazugehörigem Audio betrachtet wird. Wird Video und dazugehöriges Audio gemeinsam ausgegeben, so müssen sie zum Zweck der Ausgabe **synchronisiert** werden. Dieser Aspekt wird in Kapitel 2.2.2 erläutert.

Ab einer Bildwiederholfrequenz von ca. 16 Bildern pro Sekunde nimmt das menschliche Auge ein Folge von Bilder nicht mehr als Einzelbilder wahr, sondern als Bewegungsabläufe. Ist die Bildfrequenz unter ca. 50 Bildern pro Sekunde, entsteht ein Flimmereffekt, der durch eine periodische Schwankung der Helligkeitsempfindung hervorgerufen wird. Erst ab ca. 70 Hz verschwindet dieser Effekt. Dabei sind die Werte von der betrachtenden Person abhängig. Beim in Deutschland verwendeten **PAL (Phase Alterning Line)** System besteht ein Fernsehbild aus zwei ineinander verschachtelten Halbbildern, Bildschirme werden nach dem Zeilensprungverfahren beschrieben. Dabei werden abwechselnd alle ungeraden und alle

Kapitel 2.6] muß die Abtastfrequenz mindestens doppelt so groß sein wie die höchste auftretende Frequenz des ursprünglichen Signals, um eine Rekonstruktion des ursprünglichen Signals aus der kodierten Darstellung zu ermöglichen.

Die **Pulse Code Modulation PCM** [Mühl92; Kapitel 2.4] wird schon seit längerer Zeit im Bereich der Telefonie bei digitalen Übertragungsstrecken eingesetzt. Das deutsche Telefonnetz überträgt Signale zwischen 300 und 3400 Hz, als Abtastfrequenz wird (zur Sicherheit) 8 kHz verwendet. Die Abtastwerte werden in einem Kodewort der Länge eines Bytes gespeichert, d.h. für die Übertragung ist eine Kapazität von 8000 Hz * 8 Bits = 64 KBits/s notwendig.

Die Übertragung von Audio mit **Compact Disc Qualität** hat weit höhere Anforderungen an das Kommunikationssystem. Die Abtastfrequenz beträgt 44100 Hz, die Kodewörter haben eine Länge von zwei Bytes. Dazu kommt ein zweiter Kanal für Stereo, der den Aufwand verdoppelt. Daraus ergibt sich eine Übertragungskapazität von: 44100 Hz * 16 Bits * 2 = 1411,2 KBits/s. Der Aufwand bei einer Übertragung ist demnach mehr als 22 Mal höher als bei der Übertragung in Telefonqualität.

Auch ohne eine Interpretation des Signals läßt sich die erforderliche Datenrate bei PCM bei gleicher Qualität reduzieren. Bei der **Differenz Pulse Code Modulation DPCM** wird mittels Differenzen der Abtastwerte kodiert und damit 56 KBits/s erreicht. Als adaptives Verfahren wird die **Adaptive Pulse Code Modulation ADPCM** eingesetzt, hiermit werden 32 KBits/s erreicht. Betrachtungen zur Kompression von Daten sind Bestandteil von Kapitel 2.1.5.

Bei der **Quellenkodierung** werden sprachspezifische Besonderheiten zur Kodierung des Signals verwendet. Dazu gehören die Analyse von bestimmten Frequenzmaxima und die Unterscheidung zwischen stimmlosen und stimmhaften Lauten. Mit einer solchen Technik lassen sich Datenraten von 3 KBits/s erreichen, die Qualität ist jedoch nicht immer voll zufriedenstellend [Stei93; Kapitel 3.5]. Europäische Mobilfunknetze verwenden Techniken mit einer Datenrate von ca. 13 KBit/s [Dege94]. In Abbildung 2-2 ist die Quellenkodierung schematisch dargestellt.

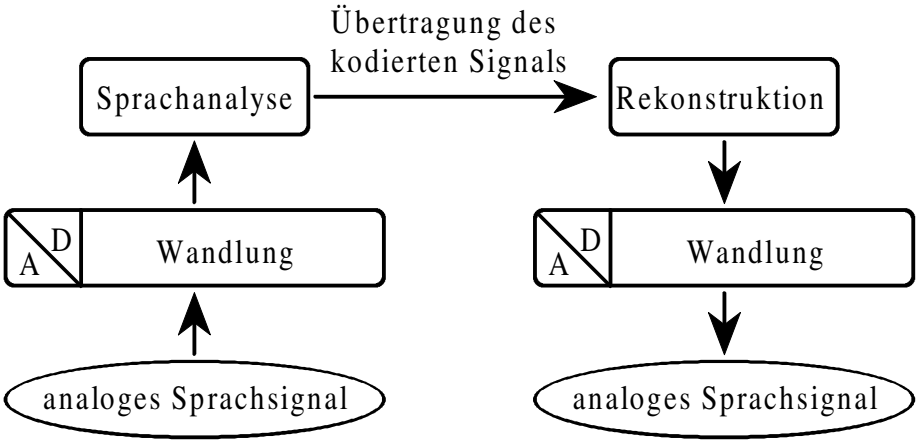


Abbildung 2-2: Quellenkodierung

Medienfragmente) erforderlich, um die geringen Verzögerungszeiten zu erreichen. Bei einer solchen fragmentierten Übertragung werden Audiofragmente in der Größenordnung von mehreren Millisekunden (beispielsweise 20 ms bei NeVoT) verwendet, Videosequenzen werden meistens in Einzelbilder aufgeteilt. Je kleiner die Pakete sind, desto geringere Verzögerungszeiten sind bei paketorientierter Übertragung erreichbar.

Die Betrachtungen dieser Arbeit orientieren sich an der Übertragung mit Realzeit-Anforderungen. Diese stellt die höchsten Anforderungen an das Kommunikationssystem und die beteiligten Systeme.

Die Präsentation der Medien erfolgt mittels Peripheriegeräten, beispielsweise Lautsprecher für Audio oder Bildschirme für Video. Dabei ist eine entsprechende Abstimmung der dargestellten Medien und der Ausgabeperipherie notwendig. Insbesondere müssen die Bildschirme die erforderliche Auflösung besitzen, bzw. angeschlossene Lautsprecher sollten der Qualität des Audios genügen.

2.1.3 Audio

Audio wird schon seit langer Zeit für die Übermittlung von Informationen verwendet (Telefon). Bei einer digitalen Übertragung müssen analoge Audiosignale zunächst in eine digitale Darstellung überführt werden. Dafür gibt es mehrere Ansatzmöglichkeiten, aus denen unterschiedliche Datenraten für die Übertragung resultieren. Im folgenden wird auf Audio allgemein und die unterschiedlichen Möglichkeiten der Übertragung eingegangen.

Audio bezeichnet akustisch wahrnehmbare Signale (Hörschall), die beim Mensch zwischen ca. 20 Hz und ca. 20 kHz liegen. Audio bezeichnet somit Sprache, Geräusche und Musik. Diese Medien sind aufgrund ihrer Informationsdichte für Konferenzsysteme und Informationssysteme gut geeignet, zudem ist Sprache ein natürliches Medium. Im folgenden wird Audio in zwei Darstellungformen betrachtet, zum einen in der Form von Schallwellen und zum anderen in einer digitalen Darstellung.

Zum Zweck der digitalen Übertragung muß Audio digitalisiert werden. Dabei wird das analoge Audiosignal abgetastet, den wertkontinuierlichen Abtastwerten ein diskreter Wert zugeordnet (Quantisierung) und aus den Abtastwerten eine Kodierung erstellt. Die Kodierung kann unabhängig vom ursprünglichen Signal erfolgen, sprachspezifische Besonderheiten berücksichtigen oder mittels Spracherkennung vollzogen werden. Die letzten zwei Möglichkeiten sind nur für Sprachsignale geeignet. Je nach verwendeter Kodierung ist ein unterschiedlicher Speicherbedarf bzw. eine unterschiedliche Datenrate für die Übertragung erforderlich. Die in dieser Arbeit vorgestellten Kodierungen werden in Hinblick auf die bei einer Übertragung erforderlichen Datenrate betrachtet.

Die **Signalformkodierung** betrachtet keine Besonderheiten des ursprünglichen Audiosignals. Parameter für die Signalformkodierung sind die höchste Frequenz des Signals und der Wertebereich für die abgetasteten Audiosignale. Gemäß dem Nyquist-Theorem [Hals92;

Mittels Computern verarbeitete Medien müssen zunächst eine digitale, speicherbare Darstellung erhalten. Dies kann mit der Abbildung realer Objekte auf eine digitale Darstellung im Computer oder die Verarbeitung bereits digital vorliegender Informationen erfolgen. So können Bewegtbilder mit Kameras aufgenommen und digitalisiert, Audiosignale mit Mikrophonen aufgenommen und digitalisiert werden. Weitere Beispiele sind Animationen oder im Computer zusammengesetzte Bildsequenzen.

Die Stadien zwei und vier bezeichnen die Möglichkeit bzw. Notwendigkeit der Verarbeitung und Speicherung von Daten. Verarbeitung der Daten kann als Aufbereitung für die Übertragung notwendig sein, z.B. Umkodieren oder Kompression/Dekompression. Speicherung kann entweder als kurzzeitige Zwischenspeicherung (beispielsweise zur Kompensation von Verzögerungsschwankungen) oder als längere Speicherung, z.B. bei multimedialen Datenbanken, verwendet werden.

Die Übertragung von multimedialen Daten ist auf unterschiedliche Weise möglich. In diesem Abschnitt erfolgt eine Klassifikation gemäß dem Anspruch an das verwendete Kommunikationssystem. Folgende Arten der Übertragung zeitkontinuierlicher Daten sind möglich:

- Übertragung mit Realzeit-Anforderungen, d.h. beim Sender generierte oder vom Sender verschickte Informationen werden minimal zeitversetzt bei einem Empfänger dargestellt. Die Abstände zwischen den Paketen dürfen dabei nicht oder nur minimal variieren (kein bzw. minimaler Jitter). Dies ist die einzige Möglichkeit, um interaktive Konferenzsysteme oder verteilte Realzeit-Simulationen zu realisieren.
- Übertragung ebenfalls noch während des Erstellens oder Abspielens, aber nicht mit Realzeit-Anforderungen. Diese Möglichkeit kann eingesetzt werden, wenn beispielsweise ein Video auf Abruf (on demand) von einem Rechner zu einem anderen übertragen wird. Bei dieser Variante ist für einen Benutzer im Gegensatz zur ersten vorgestellten Möglichkeit eine Verzögerungszeit von mehreren Sekunden zu Beginn des Abrufs vertretbar. Während der Übertragung dürfen die Paketankunftszeiten nur in einem gewissen Rahmen variieren (vgl. erste Möglichkeit), ansonsten können Verzögerungen während der Ausgabe nur mit hohem Aufwand und hoher Verzögerung kompensiert werden. Wichtig für die Übertragung ist hier eventuell die Möglichkeit, dem Benutzer durch schnelles Vor- oder Rückspulen Flexibilität bei der Präsentation zur Verfügung zu stellen.
- Komplette Übertragung der Daten. Eine Auslieferung von Medienfragmenten an eine Anwendung ist nicht vorgesehen, bevor nicht die ganze zu übertragende Einheit (z.B. ein komplettes Video) beim Empfänger vorliegt. Wenn kein Verlust oder keine Verfälschung der Daten auftreten soll, so entspricht dies der traditionellen Datenübertragung in paketorientierten Netzen. Im folgenden wird hierfür der Begriff **konventionelle Dateiübertragung** verwendet. Die Alternative mit Inkaufnahme von Bitfehlern oder Teilverlusten wird im folgenden nicht weiter betrachtet.

Insbesondere für die erste Alternative (Übertragung mit Realzeit-Anforderungen), ist eine Übertragung von sehr kleinen Paketen von Audio bzw. Video **(Medienpakete,**

und Verarbeitung von mindestens einem zeitunabhängigen *und* einem zeitabhängigen Medium. Letzteres wird dabei ausführlicher betrachtet.

2.1.2 Verteilte Systeme

Zunächst wird ein Kurzüberblick über verteilte multimediale Systeme gegeben. Dieser erfolgt anhand der Verarbeitungsschritte, die zwischen Erstellen von Audio/Video und deren Ausgabe liegen. Im Anschluß daran wird auf den Transport von Audio/Video gesondert eingegangen.

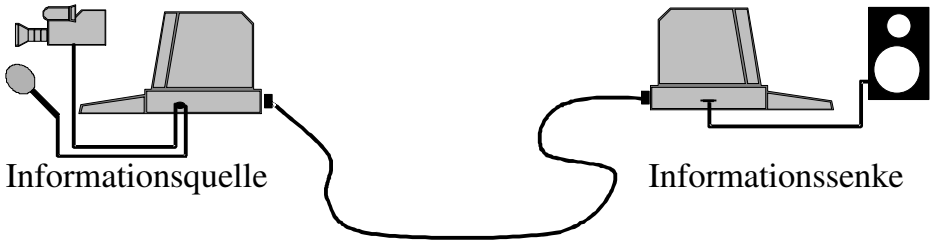


Abbildung 2-1: Verteiltes multimediales System

In Abbildung 2-1 ist ein verteiltes multimediales System, reduziert auf eine Quelle und eine Senke, schematisch dargestellt. Medien durchlaufen dabei vom Erstellen bei einer Quelle bis zur Präsentation bei einer Senke mehrere Stadien. Diese Stadien sind in Tabelle 2-1 dargestellt, sie werden in den nächsten Abschnitten erläutert.

Stadium	Beschreibung	Beispiele
1	Erstellen	Videoaufnahme Audioaufnahme Erstellen von Animationen
2	Verarbeiten/Speichern	Zwischenspeicherung Archivierung Kompression Umkodieren
3	Übertragen mittels Kommunikationssystem	Komplettübertragung Übertragung auf Abruf (on demand) Übertragung in Realzeit
4	Verarbeiten/Speichern	Zwischenspeicherung Archivierung Dekompression Umkodieren
5	Präsentation	Ausgabe von Audio und Video über Lautsprecher bzw. Bildschirm

Tabelle 2-1: Stadien von Medien in verteilten Systemen

2 Verteilte multimediale Systeme

In den folgenden Kapiteln wird auf Aspekte verteilter multimedialer Systeme eingegangen, diese bilden die Grundlage für das Verständnis der darauffolgenden Kapitel. Dabei steht im ersten Teil ein allgemeiner Überblick im Vordergrund. Im zweiten Teil erfolgt eine Betrachtung von Protokollmechanismen der Transport- und Sitzungsschicht, welche beim Datentransport innerhalb verteilter multimedialer Systeme eine besondere Bedeutung besitzen. Hierbei handelt es sich um den Hauptteil dieses Kapitels. Im letzten Teil wird die seitens des Netzwerks erforderliche Unterstützung erläutert, dies dient zur Abrundung des Überblicks.

2.1 Grundlagen

Die folgenden Unterkapitel bilden die Grundlage für das Verständnis verteilter multimedialer Systeme. Neben Begriffsdefinitionen und allgemeinen Ausführungen zu verteilten multimedialen Systemen werden die in diesem Kontext vorwiegend verwendeten Medien Audio und Video bezüglich ihrer Übertragung detailliert betrachtet.

2.1.1 Multimedia

Die hier vorgestellten Begriffsdefinitionen orientieren sich an [Stei93], es wird jedoch nur eine Auswahl der dort definierten Begriffe vorgestellt.

Als **Medium** wird im allgemeinen ein Mittel zur Verbreitung und Darstellung von Informationen bezeichnet. Beispiele hierfür sind Text, Bilder, Grafiken, Bewegtbilder, Sprache und Geräusche. Diese Medien lassen sich unterscheiden nach den Sinnesorganen, über die sie vom Mensch aufgenommen werden. Sie werden als visuelle und auditive Medien bezeichnet, je nachdem, ob sie über das Auge oder das Ohr aufgenommen werden. Andere Medien, die nicht über eines der beiden genannten Sinnesorgane aufgenommen werden, werden im folgenden nicht weiter betrachtet. Der Begriff Medium in dieser Arbeit stellvertretend für eines oder mehrere der hier aufgeführten Medien.

Medien können entsprechend ihrer Darstellung in zeitabhängige und zeitunabhängige Medien unterteilt werden. Zeitunabhängig sind Medien, deren Komponenten untereinander in keiner zeitlichen Relation stehen wie Seiten eines Buches oder Teilbilder eines Gesamtbildes. Die Gültigkeit der Informationen ist demnach auch nicht von der Zeit abhängig. Bei **zeitabhängigen (kontinuierlichen) Medien** ist die Gültigkeit von Informationen abhängig von der Zeit. Bei Video gehört eine Bildsequenz an einen definierten Zeitpunkt, zu einem anderen hat sie keine Gültigkeit. Im Rahmen dieser Arbeit werden als zeitabhängige Medien vorwiegend Audio und Video betrachtet.

Der Begriff Multimedia impliziert durch seine Wortkomposition lediglich die Verwendung mehrerer Medien, nicht jedoch die Verwendung von kontinuierlichen Medien. Entsprechend [Stei93; Kapitel 2.3] steht im folgenden der Begriff **Multimedia** jedoch für die Verwendung

1.3 Überblick

Kapitel 1.3 gibt einen Überblick über die Inhalte der Kapitel und die Vorgehensweise dieser Diplomarbeit.

In Kapitel 2 wird ein allgemeiner Überblick über verteilte multimediale Systeme gegeben. Zunächst werden Grundlagen erläutert, daran anschließend die Charakteristiken von Audio und Video in Hinblick auf ihre Übertragung betrachtet. Im zweiten Teil des Kapitels erfolgt die Betrachtung mehrerer Protokollmechanismen, welche bei der Übertragung von Audio und Video in der Transportschicht und der Sitzungsschicht eingesetzt werden. Der letzte Teil beinhaltet netzwerkspezifische Aspekte zur Übertragung von Audio und Video. Kapitel 2 bildet die Grundlage für die folgenden Kapitel.

In Kapitel 3 wird das Audio-Konferenzsystem NeVoT vorgestellt. Dies dient zum einen zur Einordnung des von NeVoT verwendeten Transportprotokolls RTP und zum anderen zu einem allgemeinen Überblick über das Programm, der für das Verständnis der Portierung von NeVoT erforderlich ist.

In Kapitel 4 wird das Realzeit-Transportprotokoll RTP vorgestellt. Zunächst wird anhand eines Beispiels die grundsätzliche Funktionsweise vorgestellt, daran anschließen erfolgen die Erläuterung der Protokolldateneinheiten und ergänzende Informationen, die über die Spezifikation hinausgehen. Das Kapitel dient als Überblick über RTP sowie als Grundlage für den Vergleich von RTP und AVXP und die konzeptionelle Erweiterung von MMC um RTP.

In Kapitel 5 wird das Konferenzsystem MMC und das dazugehörige Protokoll für den Transport von Audio und Video, AVXP, vorgestellt. Das Kapitel dient als Basis für den Vergleich von RTP und AVXP und die Betrachtungen zur Erweiterung von MMC um RTP.

In Kapitel 6 wird die konzeptionelle Erweiterung von MMC um RTP vorgestellt. Zunächst werden RTP und AVXP verglichen, daran anschließend werden Vorschläge für die Steigerung der Qualität der Datenübertragung bei MMC gemacht, die durch die Verwendung von RTP möglich sind.

In Kapitel 7 wird die Portierung des Audio-Konferenzsystems NeVoT und des Protokolls RTP vorgestellt. Ausgehend von den Voraussetzungen für die Portierung werden allgemeine Modifikationen dargelegt, sowie speziell die Anpassung an das Audiosystem bei NeVoT vorgestellt.

In Kapitel 8 wird eine Zusammenfassung dieser Arbeit und ein Ausblick auf mögliche Ansätze weiterer Arbeiten gegeben.

Engineering Task Force) Arbeitsgruppe „Audio und Video Transport“ (URL: http://www.ietf.cnri.reston.va.us/home.html) erstellt. RTP wurde von dieser Arbeitsgruppe spezifiziert. Für die praktische Umsetzung der vorgestellten Erweiterungsmöglichkeiten, welche durch die Verwendung von RTP in MMC möglich werden, ist zudem eine RTP-Version notwendig, die auf der gleichen Systemumgebung läuft wie MMC.

Das Audio-Konferenzsystem NeVoT wurde auf einem Sun-Arbeitsplatzrechner entwickelt. NeVoT ist ein Programm für interaktive Audiokonferenzen. Die Portierung umfaßt die Modifikation des Programms dahingehend, daß es auch auf Alpha-Arbeitsplatzrechnern der Firma Digital lauffähig ist. Notwendig für die Portierung ist die Anpassung an Besonderheiten des Betriebssystems, welches von Alpha-Arbeitsplatzrechnern verwendet wird. Des weiteren ist aufgrund der Besonderheiten der beiden Systeme eine Neugestaltung der Schnittstelle zum Audiosystem erforderlich. Dabei wurde bei DEC-Alpha die Programmierschnittstelle DEC-Audiofile [LPG&93] verwendet.

1.2 Aufgabenstellung

Nach der Motivation für die Analyse von Protokollen zur Übertragung von Audio und Video wird in diesem Kapitel die Aufgabenstellung konkret vorgestellt.

Bestandteile dieser Arbeit sind:

- Die Analyse der Protokolle RTP (Real-Time Transport Protocol) [SCFJ94] und AVXP (Audio Video Exchange Protocol) [ADH&93], die für die Übertragung von Audio und Video ausgelegt sind.
- Die konzeptionelle Erweiterung des Audio-/Video-Konferenzsystems MMC (Multimedia Collaboration Service) [ADH&93] um das Transportprotokoll RTP.
- Die Portierung und Integration des Audio-Konferenzsystems NeVoT (Network Voice Terminal) [Schu95b] und des Protokolls RTP von Sun-Arbeitsplatzrechnern in die Systemumgebung von Alpha-Arbeitsplatzrechnern der Firma Digital.

Im folgenden wird auf die einzelnen Punkte detaillierter eingegangen.

Die in den Kapitel 4 und 5 vorgestellten Protokolle RTP und AVXP sind Protokolle, deren Funktionalität gemäß OSI-Basisreferenzmodell (OSI-BRM) [HeSh92] im Bereich der Transportschicht und der Sitzungsschicht angesiedelt ist. Sie werden hier vereinfachend als Transportprotokolle bezeichnet. Ihre Funktionalität wird anwendungsorientierten Protokollen oder Anwendungen direkt zur Verfügung gestellt. RTP und AVXP besitzen einen unterschiedlichen Hintergrund bezüglich ihrer Entwicklung. AVXP dient als spezialisierte Lösung, es wurde für eine bestimmte multimediale Anwendung (MMC) entworfen. Im Gegensatz dazu wurde RTP als allgemeine Unterstützung für den Transport von Audio und Video in unterschiedlichen Anwendungen entwickelt. In dieser Arbeit werden beide Protokolle anhand ihrer Funktionen vorgestellt, sowie Gemeinsamkeiten und Unterschiede dargestellt. Die allgemeinen Grundlagen für die von den Protokollen verwendeten Mechanismen werden in Kapitel 2 ausführlich erläutert.

Im Rahmen des Audio- und Video-Konferenzsystems MMC wird zur Übertragung von Audio, Video und anderen Daten das Protokoll AVXP verwendet. Anhand der Analyse und eines Vergleichs der Protokolle AVXP und RTP wird untersucht, ob durch die Erweiterung von MMC um RTP eine Steigerung der Qualität der Datenübertragung innerhalb des Programms möglich ist. Dabei wird insbesondere untersucht, ob durch die Verwendung von RTP

- die Netzlast reduziert werden kann,
- Verzögerungszeiten und -schwankungen reduziert werden können oder
- durch andere Vorteile von RTP über AVXP die Flexibilität von MMC oder die Qualität der Datenübertragung in MMC gesteigert werden kann.

Die Portierung von NeVoT und RTP ermöglichte die Analyse von RTP anhand einer auf Alpha-Arbeitsplatzrechnern lauffähigen Implementierung.

Die im Kontext von NeVoT verwendete Implementierung von RTP dient zudem als Referenz-Implementierung. Sie wurde von Henning Schulzrinne, einem Mitglied der IETF (Internet

werden, wird Audio und Video i.d.R. in Pakete aufgeteilt **(fragmentiert)** und übertragen. Jedes Paket entspricht somit einem Zeitpunkt des zu übertragenden Audio-/Videostroms. Bei gleicher Paketgröße werden bei der Übertragung Pakete in äquidistanten Zeitabschnitten gesendet. Falls die Übertragung gewährleistet, daß die Zeiten zwischen den Paketen beim Empfang der Daten den Zeiten zwischen den Paketen beim Senden entspricht, handelt es sich um einen **isochronen** Datendienst. In diesem Fall kann die Ausgabe der Daten entsprechend dem Ankunftszeitpunkt gesteuert werden.

Treten Verzögerungsschwankungen **(Jitter)** während der Übertragung auf, muß für die Ausgabe beim Empfänger eine zeitliche Abstimmung **(Synchronisation)** zwischen dem Audio-/Videostrom beim Sender und den empfangenen Daten erfolgen. Kapitel 2 wird zeigen, daß zu diesem Zweck i.d.R. eine Zeitinformation für jedes Paket erforderlich ist.

Die erforderliche Zeitinformation ist einer der zentralen Gründe dafür, daß der Transport von Audio und Video nicht alleine durch Protokolle wie UDP, TCP [Davi88] oder XTP [XTP92] realisiert werden kann. Diese Protokolle unterstützen keine Zeitinformation. Es sind demnach neue Protokolle erforderlich, deren Funktionalität gemäß dem OSI-Basisreferenzmodell in die Transport- und die Sitzungsschicht einzuordnen ist. Die Synchronisation ist nur ein Aspekt für die Neugestaltung von Protokollen zur Übertragung von Audio und Video, in den Kapiteln 2, 4 und 5 werden weitere Aspekte vorgestellt. Aus Gründen der Effizienz ist es sinnvoll, für den Transport von Audio und Video Protokolle einzusetzen, die sowohl transportspezifische als auch anwendungsspezifische Funktionen übernehmen.

Der letzte Abschnitt hat gezeigt, daß für die Übertragung von Audio und Video neue Protokolle erforderlich sind. Bei der Suche nach existierenden Ansätzen fällt neben spezialisierten Lösungen ein Protokoll für die Unterstützung verteilter multimedialer Systeme auf: Das Realzeit Transportprotokoll RTP [SCFJ94], welches von der Internet Engineering Task Force (IETF) spezifiziert wurde. Zum Zeitpunkt der Arbeit stand die Verabschiedung von RTP als RFC (request for comments) unmittelbar bevor. Der zentrale Aspekt der Arbeit besteht in der Analyse und dem Vergleich von RTP und dem spezialisierten Protokoll für die Übertragung von Audio und Video AVXP (Audio Video Exchange Protocol) [ADH&93]. Die konkrete Aufgabenstellung wird im nächsten Kapitel vorgestellt.

1 Einleitung

Kapitel 1 beinhaltet Motivation und Aufgabenstellung dieser Arbeit und einen Überblick über die einzelnen Kapitel. In Kapitel 1.1 wird zunächst auf die Motivation für den zentralen Punkt der Arbeit eingegangen, die konkrete Aufgabenstellung und weitere Ausführungen diesbezüglich erfolgen im Anschluß daran. Kapitel 1.3 gibt einen Überblick über den Inhalt der folgenden Kapitel.

1.1 Motivation

Die Steigerung von Rechenleistung unter anderem im Bereich von Arbeitsplatzrechnern und die Integration neuer Peripheriegeräte hat in den letzten Jahren eine neue Dimension von Anwendungsprogrammen ermöglicht. Höhere Speicherkapazität, hochauflösende Bildschirme und neue Peripheriegeräte wie Mikrophone, Lautsprecher und Kameras ermöglichen seit einiger Zeit die Präsentation und Verarbeitung von Sprache und Bewegtbildern. Mit dem Einsatz dieser Medientypen ist es möglich, Informationen im Vergleich zu Text und Einzelbildern in anderer Art und mit höherer Informationsdichte darzustellen.

Ebenso wie die Rechenleistung ist die Vernetzung von Computern aller Art gestiegen. Immer mehr Systeme werden durch lokale, Weitverkehrs- und globale Netzwerke verbunden. Vernetzte Systeme haben den großen Vorteil, daß Daten bzw. Informationen mit verhältnismäßig geringem Aufwand in kurzer Zeit eine große räumliche Strecke überwinden können. Diese Vernetzung macht es wünschenswert, auch Audio und Video global verfügbar zu machen.

Durch die Kombination von Vernetzung und Verwendung sowohl konventioneller Medien (z.B. Text) als auch neuer Medien wie Audio und Bewegtbildern ergibt sich eine neue Klasse von Anwendungen. Es sind dies die **verteilten multimedialen Anwendungen**. Beispiele hierfür sind Konferenzsysteme, Lehrsysteme und Informationsdienste.

Die hier vorgestellten neuen Möglichkeiten verändern das Anforderungsprofil von Kommunikationskomponenten in hohem Maße. Steht bei der Übertragung von konventionellen Dateien (Text, Einzelbildern) die Fehlersicherheit im Vordergrund, so werden bei der Übertragung von Audio und Video andere Aspekte bedeutsam. Dies liegt darin begründet, daß Audio und Video oftmals nicht als Einheit übertragen und anschließend ausgegeben werden, sondern noch während des Sendens bereits mit der Ausgabe beim Empfänger begonnen wird. Das Auftreten von Fehlern kann im Gegensatz zur Dateiübertragung unter Umständen toleriert werden, z.B. wenn es sich um eine kurze Unterbrechung der Audioübertragung handelt. Erforderlich hingegen ist das rechtzeitige Eintreffen der Daten (vor dem Ausgabezeitpunkt) beim Empfänger.

Für die Realisierung interaktiver Anwendungen, bei denen erstellte oder wiedergegebene Daten/Informationen minimal zeitversetzt bei einem oder mehreren Empfängern ausgegeben

7 NEVOT AUF DEC-ALPHA 77

7.1 ÜBERBLICK 77
7.2 DATENSTRUKTUREN 78
7.2.1 DATENTYPEN 78
7.2.2 BYTEANORDNUNG 78
7.2.3 DEKLARATIONSDATEIEN UND FUNKTIONEN 79
7.3 EIN-/AUSGABE VON AUDIO 80
7.3.1 EINLEITUNG 80
7.3.2 DEC-AUDIOFILE 81
7.3.3 ZEITLICHE NEBENBEDINGUNGEN 85
7.3.4 DER AUDIOZUGRIFF BEI NEVOT 88
7.3.5 IMPLEMENTIERUNG 89

8 ZUSAMMENFASSUNG 95

A MEßERGEBNISSE AUDIOFILE-FUNKTIONEN 97

B ABKÜRZUNGSVERZEICHNIS 99

C QUELLENVERZEICHNIS 101

4.2.2 PAKETKOPFERWEITERUNG 38
4.2.3 ERGÄNZENDE ERLÄUTERUNGEN 39
4.3 DAS KONTROLLPROTOKOLL RTCP 40
4.3.1 EINLEITUNG 40
4.3.2 SENDERBERICHT 41
4.3.3 EMPFÄNGERBERICHT 44
4.3.4 QUELLENBESCHREIBUNG 45
4.3.5 ABMELDUNG 47
4.3.6 ANWENDUNGSSPEZIFISCHE PAKETE 48
4.4 ANHANGDATEIEN 48

5 DER MULTIMEDIA COLLABORATION SERVICE (MMC) 51

5.1 ÜBERBLICK 51
5.1.1 PROJEKTHINTERGRUND 51
5.1.2 FUNKTIONSUMFANG 52
5.2 SYSTEMSTRUKTUR 53
5.2.1 AUFBAU 53
5.2.2 KOMPONENTEN 55
5.3 AVXP - TRANSPORT VON AUDIO UND VIDEO 56
5.3.1 EINLEITUNG 56
5.3.2 PROTOKOLLDATENEINHEITEN 57

6 ERWEITERUNG VON MMC UM RTP 59

6.1 VERGLEICH VON RTP UND AVXP 59
6.1.1 GRUNDLAGEN 59
6.1.2 VERSIONSIDENTIFIKATION 60
6.1.3 ADREßINFORMATION 61
6.1.4 PAKETLÄNGENANGABE 63
6.1.5 VARIABLER PAKETKOPF 63
6.1.6 VERSCHLÜSSELUNG UND FÜLLBYTES 63
6.1.7 OFFSET 64
6.1.8 UNTERBROCHENE ÜBERTRAGUNG 64
6.1.9 ZEITSTEMPEL, SEQUENZNUMMER UND KOMPONENTENANGABE 65
6.1.10 NUTZDATENIDENTIFIKATION 66
6.1.11 KONTROLLINFORMATIONEN 66
6.2 MÖGLICHKEITEN 66
6.2.1 INTEROPERABILITÄT 67
6.2.2 ADAPTIVE VORWÄRTSFEHLERBEHEBUNG 68
6.2.3 EMPFÄNGERGESTEUERTE SENDEREGULIERUNG 73
6.3 ZUSAMMENFASSUNG 75

Inhaltsverzeichnis

1 EINLEITUNG 1

1.1 MOTIVATION 1
1.2 AUFGABENSTELLUNG 3
1.3 ÜBERBLICK 5

2 VERTEILTE MULTIMEDIALE SYSTEME 7

2.1 GRUNDLAGEN 7
2.1.1 MULTIMEDIA 7
2.1.2 VERTEILTE SYSTEME 8
2.1.3 AUDIO 10
2.1.4 VIDEO 12
2.1.5 KOMPRESSION 13
2.2 PROTOKOLLUNTERSTÜTZUNG 15
2.2.1 UNTERBROCHENE ÜBERTRAGUNG 15
2.2.2 SYNCHRONISATION 16
2.2.3 FEHLERBEHANDLUNG 19
2.2.4 DIENSTGÜTE 20
2.3 NETZUNTERSTÜTZUNG 21
2.3.1 LEISTUNGSFÄHIGKEIT 21
2.3.2 RESSOURCEN-RESERVIERUNG 22

3 DAS NETWORK VOICE TERMINAL (NEVOT) 25

3.1 ÜBERBLICK 25
3.1.1 EINLEITUNG 25
3.1.2 FUNKTIONSWEISE 26
3.2 KOMPONENTEN 28
3.3 IMPLEMENTIERUNG 30
3.3.1 AUDIOUHR UND SYSTEMUHR 30
3.3.2 UNTERBROCHENE ÜBERTRAGUNG 31

4 DAS REALZEIT-TRANSPORTPROTOKOLL RTP 33

4.1 ÜBERBLICK 33
4.1.1 FUNKTIONSUMFANG 33
4.1.2 BEISPIELSZENARIO UND ZWISCHENSYSTEME 34
4.2 PROTOKOLLDATENEINHEITEN 36
4.2.1 DER PAKETKOPF 36

Ich erkläre hiermit, daß ich die vorliegende Arbeit selbständig verfaßt und nur die angegebenen Quellen und Hilfsmittel verwendet habe.

Karlsruhe, den 31.07.1995

Philipp Harrschar

Kommunikationsunterstützung für multimediale Anwendungen

Diplomarbeit am Institut für Telematik
Fakultät für Informatik
Universität Karlsruhe (TH)

von

Philipp Harrschar

Tag der Ausgabe: 01.02.95
Tag der Abgabe: 31.07.95

Betreuer:
Prof. Dr. Dr. h. c. Gerhard Krüger
Dipl.-Inform. Markus Hofmann
Dipl.-Inform. Rudolf Brandner, CEC Karlsruhe