

Universität Kaiserslautern

- AG Integrierte Kommunikationssysteme -  
und  
AG Systemsoftware

**Adaption einer IP-basierten  
Multimedia-Applikation  
auf ATM**

Jörg Illerich

Betreuer:

Dipl. Inform. Bernd Reuther  
Dipl. Inform. Michael Baentsch  
Prof. Dr. Paul Müller

Mai 97

# Inhaltsverzeichnis

<b>1</b>	<b>Einleitung</b>	<b>4</b>
1.1	Motivation . . . . .	4
1.2	Thema und Zielsetzung . . . . .	5
1.3	Aufbau der Ausarbeitung . . . . .	5
<b>2</b>	<b>Einführung</b>	<b>7</b>
2.1	Multimedia . . . . .	7
2.2	Rechnernetze Grundlagen . . . . .	9
2.2.1	OSI-Referenzmodell . . . . .	9
2.2.2	Verbindungsdienste . . . . .	10
2.2.3	Vermittlungstechniken . . . . .	10
2.2.4	Multiplexverfahren . . . . .	11
2.2.5	Übermittlungsverfahren . . . . .	12
2.3	Internet Protokolle IP, TCP, UDP . . . . .	13
2.4	Notwendigkeit von Protokollen mit Dienstgütegarantien . . . . .	15
2.5	Protokolle mit Dienstgütegarantien . . . . .	15
2.5.1	RSVP . . . . .	15
2.5.2	ST2 . . . . .	19
2.5.3	RTP . . . . .	21
2.5.4	Asynchroner Transfer Modus (ATM) . . . . .	24
2.5.4.1	ATM-Konzepte . . . . .	24
2.5.4.2	ATM-Referenzmodell . . . . .	27
2.5.4.3	Dienstklassen . . . . .	29
2.5.4.4	ATM-Adaptionsschicht . . . . .	30
2.5.4.5	ATM im Einsatz . . . . .	32

<i>INHALTSVERZEICHNIS</i>	3
2.5.4.6 LANE, Classical IP, MPOA . . . . .	33
2.5.5 Vergleich zwischen RSVP, ST2 und ATM . . . . .	36
2.6 Applikation VCR . . . . .	38
2.6.1 Systemarchitektur . . . . .	38
2.6.2 Software Feedback Mechanismus für Client/Server Syn- chronisation . . . . .	39
2.6.3 Software Feedback Mechanismus für die QoS Kontrolle . . .	40
2.6.4 Implementierungsaspekte . . . . .	41
<b>3 Adaption einer IP-basierten Applikation</b>	<b>43</b>
3.1 Gründe für die Adaption . . . . .	43
3.2 Probleme bei der Adaption . . . . .	44
3.3 Durchführung der Adaption . . . . .	45
<b>4 Netzwerkdienste</b>	<b>52</b>
4.1 Leistungsumfang der Dienste . . . . .	53
4.2 Dienstangebot . . . . .	53
4.2.1 Datagramme, Datenströme, Dateiübertragung . . . . .	54
4.2.1.1 Datagramme . . . . .	54
4.2.1.2 Datenströme . . . . .	54
4.2.1.3 Datenübertragung . . . . .	54
4.2.2 Audio- und Videoübertragung . . . . .	55
4.2.3 Email, News, Fax . . . . .	55
4.2.3.1 Email . . . . .	55
4.2.3.2 News . . . . .	56
4.2.3.3 Fax . . . . .	56
4.3 Einflußmöglichkeiten auf die Dienste . . . . .	57
<b>5 Zusammenfassung</b>	<b>59</b>
<b>Literaturverzeichnis</b>	<b>61</b>

# Kapitel 1

## Einleitung

### 1.1 Motivation

Klassische Datennetze (wie z.B. das IP-basierte Internet) wurden entwickelt für die Übertragung von Daten ohne Echtzeitanforderungen, wie Dateien und Emails. Für diese Art von Daten spielt es keine Rolle, daß die zur Verfügung stehende Bandbreite zwischen mehreren Sendern aufgeteilt wird und sich der Durchsatz nach dem best effort Prinzip richtet, d.h. je nachdem wieviele Stationen gerade gleichzeitig senden, ist der Anteil der zur Verfügung stehenden Bandbreite und damit auch die Datenübertragungsrate je Station hoch oder niedrig. Ändert sich die Anzahl der sendewilligen Stationen, so ändert sich damit auch die Datenübertragungsrate jeder Station. Für eine Anwendung, die ihre Daten von einem solchen Netz erhält, kann dies einen ständig wechselnden eingehenden Datenumfang bedeuten.

Mit der Einführung von verteilten Multimedia-Anwendungen wurden neue Ansprüche an die Netzwerke gestellt, da nun auch kontinuierliche Daten mit Echtzeitanforderungen (z.B. Video- und Audiodaten) übertragen werden sollen. Für die Übertragung solcher Daten wird u.a. eine begrenzte Ende-zu-Ende Verzögerung und eine geringe Verzögerungsvarianz benötigt, wenn die Präsentation der Daten auf Empfängerseite gleichmäßig und ohne Pausen geschehen soll. Ein IP-basiertes Datennetz kann diese Anforderungen nicht garantieren, so daß es für die Übertragung solcher Daten ungeeignet ist.

Dies war u.a. der Grund für die Forderung nach Unterstützung von Diensteintegration durch Netzwerke. Statt mehrerer auf einen bestimmten Datentyp spezialisierter Netze (z.B. Telefonnetz und Datennetz) zu haben, sollte die gleichzeitige Übertragung von verschiedenen Datentypen (wie Sprache, Video, Daten) über ein einziges Netzwerk möglich werden.

Dies kann durch den Asynchronen Transfer Modus (ATM) erreicht werden, der sowohl die Vermittlungs- als auch Übertragungstechnik beschreibt. Gegenüber IP-basierten Datennetzen können hier einer Verbindung eine beim Verbindungsaufbau ausgehandelte Dienstgüte während ihrer gesamten Dauer garantiert werden. Durch diese Garantien durch das Netzwerk wird eine sinnvolle Übertra-

gung von Multimedia-Daten erst möglich. Ebenso sind bei ATM Konzepte zur Diensteintegration enthalten.

Vorliegende Anwendungen setzen zumeist auf TCP/IP auf, direkt auf ATM aufsetzende Anwendungen sind dagegen bisher kaum vorhanden. Daher erklärt sich der Wunsch, den bestehenden Anwendungen die Dienstgütegarantien von ATM zugänglich zu machen. Für den Ablauf IP-basierter Anwendungen auf ATM (ohne Änderungen am Quelltext vorzunehmen) gibt es zwar entsprechende Ansätze (z.B. LAN-Emulation, Classical IP über ATM), aber diese ermöglichen den Anwendungen nicht, den Vorteil der Dienstgütegarantien zu nutzen. Dafür müssen an diesen Anwendungen Änderungen vorgenommen werden, damit diese u.a. beim Verbindungsaufbau mittels Signalisierungstechniken die von der Anwendung benötigte Dienstgüte vom Netz anfordern.

Ein Problem ergibt sich dabei bei der Bestimmung der für eine Anwendung notwendigen und vom Netzwerk zu erbringenden Dienstgüte. Der Einsatz von Kodierungs- und Komprimierungsverfahren zur Übertragung von Multimedia-Daten hat starken Einfluß auf die benötigte Dienstgüte. Daher erweist sich die Angabe der benötigten Dienstgüte bei ungenauer Kenntnis über diese Verfahren als schwierig.

## 1.2 Thema und Zielsetzung

Diese Arbeit beschäftigt sich mit der Problematik der Adaption einer IP-basierten Multimedia-Applikation auf ATM. Anhand einer Beispielanwendung soll aufgezeigt werden, wie die für die Anwendung notwendige Dienstgüte bestimmt werden kann. Darüberhinaus werden die bei der Adaption aufgetretenen Probleme und deren Lösungen erläutert.

Ein weiterer Schwerpunkt der Arbeit liegt in der Entwicklung eines Ansatzes für vom Netzwerk bereitgestellte Dienste zur einfachen Realisierung von verteilten Multimedia-Anwendungen.

## 1.3 Aufbau der Ausarbeitung

Die weitere Ausarbeitung ist wie folgt gegliedert:

### Kapitel 2: Einführung

In diesem Kapitel werden die für die weitere Ausarbeitung wichtigen Begriffe erläutert. Hier wird eine Definition von Multimedia(-Applikationen) und deren spezielle Anforderungen an Netzwerkprotokolle angegeben. Nach einer Beschreibung von (TCP/)IP wird auf die Notwendigkeit von Protokollen mit Dienstgütegarantien eingegangen. Es folgen einige Beispielprotokolle, die dies leisten (RSVP,ST2). Anschließend wird ausführlich auf ATM eingegangen und

ein Vergleich mit RSVP und ST2 vorgenommen. Abschließend wird die Beispielanwendung VCR beschrieben.

Kapitel 3: Adaption

In diesem Kapitel wird anhand der Beispielanwendung beschrieben, wie eine Adaption einer IP-basierten Anwendung nach ATM vorzunehmen ist und was es dabei zu beachten gibt. Insbesondere wird auf die Problematik der Bestimmung von QoS-Parameter eingegangen.

Kapitel 4: Dienste

Hier wird erörtert, wie man Anwendungen besser für die Übertragung von Multimedia-Daten unterstützen kann, indem man sie weitestgehend von Netzwerk-, Kodierungs- und Komprimierungsinternas befreit. Dies soll dadurch geschehen, daß bereits vom Netzwerk entsprechende Dienste (wie z.B. Videoübertragung) einer Anwendung bereitgestellt werden.

Welche Funktionalitäten solche Dienste haben könnten und welche Probleme bei der Realisierung und Nutzung solcher Dienste auftreten können, wird in diesem Kapitel erläutert.

Kapitel 5: Zusammenfassung

Zum Schluß folgt eine Zusammenfassung der wichtigsten Ergebnisse dieser Arbeit.

# Kapitel 2

## Einführung

### 2.1 Multimedia

In [Hein96] wird ein *Multimedia-System* wie folgt beschrieben:

*Ein Multimedia-System ist gekennzeichnet durch die rechnergesteuerte und integrierte Erzeugung, Speicherung, Darstellung und Manipulation von Informationen und deren Austausch zwischen Kommunikationspartnern. Multimediainformationen müssen sowohl kontinuierliche (Video, Audio) als auch diskrete (Text, Grafik) Bestandteile aufweisen. Besitzt der Anwender die Möglichkeit, den Empfang und die Darstellung der Informationen zu kontrollieren, so spricht man von interaktiven Multimediasystemen.*

Die Übertragung von Grafiken, Sprache und insbesondere von Bewegtbildern stellt bei einer verteilten Anwendung hohe Ansprüche an das Übermittlungssystem. Man benötigt neben einer hohen Bandbreite, um die z.B. bei einem Video große Menge an Informationen bewältigen zu können, auch Garantien für die Einhaltung von Netzwerkparametern, wie z.B. maximale Sendeverzögerung und Verzögerungsvarianz. Gerade für Videoübertragungen dürfen die Übertragungsraten und Sendeverzögerungen nicht zu sehr schwanken, da es sonst zu Ruckeln der Darstellung kommen kann. Andererseits ist insbesondere für Audiodaten eine möglichst verzögerungsfreie Übertragung notwendig, da es sonst bei einer Sprachübermittlung zwischen zwei Benutzern zu nicht tolerierbaren Sendeverzögerungen kommen kann.

Da mit einer Multimedia-Anwendung die verschiedensten Informationen bearbeitet werden, gilt es, deren spezielle Ansprüche in Einklang zu bringen und durch eine entsprechende Wahl und Konfiguration des Übermittlungssystems Rechnung zu tragen.

Dies ist insbesondere von Bedeutung, da viele Netzwerke folgende Nachteile besitzen [dePr93]:

- dienstabhängigkeit: Das Netzwerk ist darauf spezialisiert, nur Informationen eines Dienstes zu transportieren, für das es entworfen wurde. Daher

haben spezialisierte Netzwerke große Schwierigkeiten, sich an wechselnde oder neue Dienstanforderungen anzupassen.

- ineffizient: Ressourcen des Netzes, die einer Anwendung zur Verfügung gestellt werden, aber nicht benötigt werden, können anderen Anwendungen nicht zugänglich gemacht werden.

Wünschenswert wäre daher ein Netzwerk, das dienstunabhängig ist und alle verfügbaren Ressourcen zwischen verschiedenen Diensten teilen kann. Dies hätte die Vorteile, flexibel und zukunftssicher zu sein. Darüberhinaus wäre ein solches Netzwerk effizient in der Nutzung der verfügbaren Ressourcen und im Betrieb kostengünstiger (da nur ein Netzwerk installiert und gewartet werden muß).

Ebenso wäre eine Unterstützung eines weiten Spektrums von Datenraten sinnvoll, um eine möglichst gute Anpassung an die jeweilige Anforderung vornehmen zu können.

## 2.2 Rechnernetze Grundlagen

Nachfolgend werden einige Begriffe aus dem Bereich Rechnernetze erläutert, die in den nachfolgenden Unterkapiteln verwendet werden.

### 2.2.1 OSI-Referenzmodell

Das *OSI-Referenzmodell* (Open Systems Interconnection) [X200] von ISO (International Standard Organisation) dient als Rahmen für die Entwicklung und Normung von Kommunikationsdiensten und Protokollen. Es handelt sich dabei um eine Schichtenarchitektur, wobei die Kriterien für die Einführung einer Schicht eine klare Zuordnung von Funktionalitäten und Diensten, einem möglichst geringen Informationsfluß zwischen den einzelnen Schichten und die Berücksichtigung bereits international genormter Protokolle waren.

Das Modell besteht aus folgenden sieben Schichten:

1. Bitübertragungsschicht (physical layer)
2. Sicherungsschicht (link layer)
3. Vermittlungsschicht (network layer)
4. Transportschicht (transport layer)
5. Sitzungsschicht (session layer)
6. Darstellungsschicht (presentation layer)
7. Anwendungsschicht (application layer)

Die Bitübertragungsschicht (Schicht 1) ist wie der Name schon sagt für die Übertragung von Bits über ein physikalisches Medium zuständig. Sie befaßt sich zudem mit Fragen der Kodierung, Modulation und der Betriebsart.

Die Sicherungsschicht (Schicht 2) ist für die Übertragung von Rahmen zwischen zwei direkt verbundenen Einheiten und der Strukturierung empfangener Bitfolgen zuständig. Hier findet zudem Fehlerdiagnose, Fehlerbehebung und Flußkontrolle statt.

Die Vermittlungsschicht (Schicht 3) hat die netzwerkweite Vermittlung von Paketen und die Auswahl von Paketleitwegen (Routing) als Aufgabe. Zudem kann hier eine Kostenabrechnung stattfinden.

Die Transportschicht (Schicht 4) realisiert die Übertragung von Nachrichten, die Verbindungssteuerung und Flußkontrolle jeweils zwischen Quell- und Zielrechner. Darüberhinaus werden hier die Nachrichten in einzelne Pakete segmentiert und wieder zusammengesetzt.

Die Sitzungsschicht (Schicht 5) dient der Dialogkontrolle und der Synchronisation. Letzteres kann z.B. durch Einführung von Sicherungspunkten zum Wiederaufsetzen einer unterbrochenen Verbindung geschehen.

Die Darstellungsschicht (Schicht 6) nimmt eine Kodierung/Dekodierung in eine

Transfersyntax vor. Dies ist insbesondere bei der Vernetzung heterogener Rechner notwendig. Darüberhinaus werden hier Daten komprimiert und verschlüsselt. In der Anwendungsschicht (Schicht 7) schließlich werden die Anwendungsprotokolle (z.B. Dateitransfer, electronic mail) realisiert.

### 2.2.2 Verbindungsdienste

Die *verbindungslose Kommunikation* arbeitet nach dem Postprinzip. Jede Nachricht trägt die volle Bestimmungsadresse und wird unabhängig von allen anderen durch das System geschleust. Es erfolgt keine automatische Empfangsbestätigung. Zudem ist es möglich, daß sich zwei Nachrichten eines Absenders an den selben Empfänger überholen können.

Die *verbindungsorientierte Kommunikation* arbeitet ähnlich einem Telefongespräch. Ein Dienstanutzer muß erst eine Verbindung aufbauen, dann kann er die Verbindung zur Übertragung nutzen und schließlich wieder abbauen. Mittels verbindungsorientierter Kommunikation können fehlerfreie und Reihenfolge erhaltende Dienste realisiert werden [Tan92].

### 2.2.3 Vermittlungstechniken

Bei der *Paketvermittlung* werden die zu übertragenden Daten in Pakete variabler (bis zu einem Maximalwert) Länge verpackt. Da diese Pakete unabhängig voneinander durch das Netz geschleust werden, muß jedes Paket Informationen über Absender und Empfänger mit sich führen. Anhand dieser Information kann ein Paket bei jedem Verbindungsknoten des Netzes entsprechend weitergeleitet werden, bis es schließlich sein Ziel erreicht. Es besteht zwischen Absender und Empfänger lediglich eine logische Verbindung, da die einzelnen Pakete auf unterschiedlichen Wegen zum Ziel gelangen können (z.B. könnte die Auslastung einzelner Teilstücke des Netzes Einfluß auf die Weiterleiterichtung haben).

Diese Vermittlungart hat als Vorteil, daß die Gesamtbandbreite des Netzes effizient ausgenutzt werden kann. Durch das Teilen der einzelnen physikalischen Verbindungen benachbarter Verbindungsknoten durch die logischen Verbindungen zwischen den Kommunikationspartnern kann die Leitung, wenn zwei Kommunikationspartner gerade mal keine Daten austauschen, durch andere genutzt werden. Dieses Teilen hat aber auch den Nachteil, daß die einem einzelnen Kommunikationspaar zur Verfügung stehende Bandbreite stark wechselhaft sein kann. Paketvermittlung hat einen weiteren Nachteil. Zwischen je zwei benachbarten Verbindungsknoten werden Fehlererkennungs- und -behandlungsmechanismen durchgeführt. Dies ist aber für die Realisierung von sehr hohen Datenraten (im Bereich von einem Gigabit pro Sekunde) zu zeitaufwendig.

Eine vereinfachte Version dieser Vermittlungstechnik stellt die *schnelle Paketvermittlung* dar. Dabei führt man Fehlererkennung und -behandlung nur noch zwischen Endpunkten des Netzes, aber nicht mehr zwischen den im Netz installierten Vermittlungsknoten durch. So lassen sich die Vermittlungsknoten sehr

vereinfachen, wodurch sich der Durchsatz durch das Netz stark erhöhen läßt. Dies ist insofern gerechtfertigt, da die Fehlerwahrscheinlichkeit von Netzen in der Vergangenheit sehr verbessert werden konnte.

Bei der *Leitungsvermittlung* wird vor dem eigentlichen Datentransfer eine physikalische Verbindung aufgebaut. Dabei können vom Netz entsprechende für die Übertragung notwendige Ressourcen (z.B. eine bestimmte Bandbreite) für diese Verbindung angefordert werden. Diese stehen dann den Kommunikationspartnern exklusiv für ihre Verbindung zur Verfügung. Vorteil dieser Verbindungsart ist die Möglichkeit der exklusiven Nutzung der Bandbreite, sofern diese ständig benötigt wird. Wenn dies nicht der Fall ist (z.B. bei burstartigem Datenverkehr), wirkt sich dies nachteilig auf die Ausnutzung der Bandbreite aus, da ungenutzte Bandbreite nicht an andere Kommunikationspaare abgegeben werden kann. Ein weiterer Nachteil sind die langen Verbindungsaufbauzeiten, die die Anwendung abwarten muß, bevor sie etwas senden darf.

Zur effizienten Behandlung von burstartigem Verkehr wurde die sogenannte *schnelle Leitungsvermittlung* vorgeschlagen. Der Unterschied zur 'normalen' Leitungsvermittlung liegt darin, daß Ressourcen beim Verbindungsaufbau nicht wirklich fest allokiert werden, sondern nur bei aktuellem Bedarf, d.h. beim Transfer von Daten auf dieser Verbindung, belegt und wieder freigegeben werden. Problematisch an diesem Verfahren ist allerdings, daß es zu Situationen kommen kann, in denen eine neue Ressourcenanforderung einer bestehenden Verbindung nicht mehr erfüllt werden kann, weil momentan nicht mehr ausreichend freie Ressourcen vorhanden sind [Zitt95].

#### 2.2.4 Multiplexverfahren

Multiplexverfahren ermöglichen die Übertragung mehrerer voneinander unabhängiger Datenströme über ein und dasselbe physikalische Medium.

Beim *synchronen Zeitmultiplexen* werden die Übertragungsrahmen in konstant große Zeitschlitze unterteilt. Diese können nun durch die verschiedenen Datenströmen genutzt werden. Damit die Übermittlung eines Datenstromes synchron geschieht, erhält ein Datenstrom immer nur dieselbe Position innerhalb der einzelnen Übertragungsrahmen. Man nennt in diesem Zusammenhang die sich durch die Positionen im Übertragungsrahmen ergebenden Bereiche auch Kanäle.

Beim *asynchronen Zeitmultiplexen* werden die zu übertragenden Datenströme jeweils in Datenpakete zerlegt und getrennt übertragen. Es existiert keine feste Zuordnung zwischen Daten und Kanälen, wie es beim synchronen Zeitmultiplexen der Fall ist. Um eine Zuordnung der Datenpakete zu den Sendekanälen zu ermöglichen, ist es notwendig, die Datenpakete jeweils mit einer eindeutigen Kennung pro Kanal zu versehen.

### 2.2.5 Übermittlungsverfahren

Im Zusammenhang der Übertragungs- und Vermittlungstechniken wurde der Begriff des *Transfermodus* eingeführt. Dieser umfaßt Aspekte, die sowohl das Übertragen, daß Multiplexen als auch das Vermitteln in Telekommunikationsnetzen betreffen.

Aus den obigen Konzepten wurden in der Praxis drei verschiedene Transfer-techniken eingeführt, Synchroner Transfer Modus (STM), Paket Transfer Modus (PTM) und Asynchroner Transfer Modus (ATM). Abbildung 2.1 zeigt deren Unterschiede.

	STM	PTM	ATM
Zeitmultiplexverfahren	synchron	asynchron	asynchron
Vermittlungstechnik	Leitungs- vermittlung	Paket- vermittlung	schnelle Paketverm.
Verbindungsdienst	verbindungs- orientiert	verbindungs- orientiert	verbindungs- orientiert
Beispiel	ISDN	Frame-Relay	B-ISDN

Abbildung 2.1: Vergleich der Transfertechniken

Zu obiger Abbildung soll noch folgendes ergänzend angemerkt werden:

Beim synchronen Transfermodus besteht zwischen den Kommunikationspartnern eine physikalische Verbindung, wohingegen beim Paket-Transfermodus und beim asynchronen Transfermodus nur eine virtuelle Verbindung besteht. Beim PTM ist die Länge eines Paket variabel, wohingegen bei ATM jedes Paket die gleiche feste Länge hat, man spricht daher von einer Zelle statt von einem Paket.

Ein genauere Besprechung von ATM wird in Kapitel 2.5.4 vorgenommen. Die beiden anderen Transfertechniken wurden nur der Vollständigkeit wegen vorgestellt, um Vorteile von ATM gegenüber PTM und STM besser darstellen zu können.

## 2.3 Internet Protokolle IP, TCP, UDP

Als OSI-Schicht 3 Protokoll bietet *IP* nur einen ungesicherten, verbindungslosen Datagram-Dienst, ohne Flußkontrolle und Fehlerbehandlung. Diese werden für die Anwendungen (wie etwa FTP oder Telnet) in verbindungsorientierten Transportprotokollen wie dem Transport Control Protocol (TCP) implementiert. Anwendungen, die diese Mechanismen nicht benötigen, nutzen als Transportprotokoll das User Datagram Protocol (UDP).

In der Kontrollinformation eines IP-Paketes werden neben den 4-Byte-Adressen zur Identifikation von Sender- und Empfangsrechner auch die variable Paketlänge (maximal 64 KB) und das übertragene Transportprotokoll (UDP, TCP oder andere) kodiert. Eine Zuordnung der übertragenden Daten zu der entsprechenden Anwendung geschieht durch die in den UDP/TCP-Protokollen zu Adressierungszwecken verwendeten Ports. Im Internet unterscheidet man drei Klassen von IP-Adressen. Die ersten Bits geben an, zu welcher Klasse die Adresse gehört, darauf folgt jeweils eine unterschiedlich lange *netid* und *hostid*.

Class A: Identifier: 0, *netid* (7 Bits), *hostid* (24 Bits)

Class B: Identifier: 10, *netid* (14 Bits), *hostid* (16 Bits)

Class C: Identifier: 110, *netid* (21 Bits), *hostid* (8 Bits)

Die Netzwerkbezeichner (*netid*) werden vom Internet Network Information Center (NIC) an Organisationen mit Internetanschluß vergeben. Die Rechnerbezeichner (*hostid*) werden von den Subnet-Betreibern selbst gewählt.

Durch die Definition von speziellen Adressen und Adreßräumen unterstützt IP neben der Punkt-zu-Punkt-Kommunikation auch Punkt-zu-Mehrpunkt- (Broadcast) und Mehrpunkt-zu-Mehrpunkt-Kommunikation (Multicast).

Weitere Aufgaben von IP sind die Fragmentierung und Reassemblierung (Zusammenbau) von Datagrammen. Darüberhinaus findet hier das Routing der Datagramme statt.

*UDP* stellt einen verbindungslosen Datagram-Dienst zur Verfügung. Multicast und Broadcast sind durch entsprechende Parametrisierung des Protokoll Ports möglich. Als Erweiterung gegenüber IP stellt UDP mehrere Verbindungen pro Rechner (durch die Verwendung mehrerer Ports) bereit.

*TCP* stellt dagegen einen verbindungsorientierten Dienst zur Verfügung. Dieser Dienst ist zuverlässig, d.h. es gibt keine Nachrichtenverluste oder Duplikate und die Pakete werden fehlerfrei ausgeliefert. Aufgrund der Punkt-zu-Punkt-Verbindung (da verbindungsorientiert) steht kein Multicast oder Broadcast zur Verfügung.

Die Programmierung von Anwendungen aufbauend auf UDP oder TCP geschieht meistens durch die Verwendung von sogenannten Sockets. Diese einheitliche Schnittstelle zur Programmierung von Netzwerkanwendungen wurde von den Entwicklern des BSD-Unix entworfen. Dabei handelt es sich um eine Reihe von

Kernel-Routinen, die zum Aufbau einer Verbindung zwischen zwei Rechnern und zur Übertragung von Daten benötigt werden. Es folgen die wichtigsten Socket-Aufrufe [Tan92]:

Befehl	Beschreibung
<code>socket</code>	erstellt einen neuen Socket
<code>bind</code>	assoziiert einen ASCII-namen zu einem zuvor erstellten Socket
<code>listen</code>	erstellt eine Warteschlange, zur Speicherung der eingehenden Verbindungsanforderungen
<code>accept</code>	entfernt eine Verbindungsanforderung aus der Warteschlange oder wartet auf eine
<code>connect</code>	startet eine Verbindung mit einem entfernten Socket
<code>shutdown</code>	beendet eine Verbindung auf einem Socket
<code>send</code>	sendet eine Nachricht durch den vorgegebenen Socket
<code>recv</code>	empfängt eine Nachricht auf einem vorgegebenen Socket
<code>select</code>	prüft eine Gruppe von Sockets, ob einer gelesen oder beschrieben werden kann

Abbildung 2.2: Die wichtigsten Socket-Aufrufe

Einem durch Aufruf von `socket` erzeugten Socket kann Pufferplatz zugeordnet werden, damit er eingehende Verbindungsanforderungen speichern kann. Dies geschieht durch den Aufruf von `listen`. Damit ein anderer Prozeß eine Verbindungsanforderung an einen Socket senden kann, muß der Socket einen Namen erhalten. Diese Zuordnung geschieht mittels `bind`. Das Warten auf eine Verbindungsanforderung geschieht durch den Aufruf von `accept`. Ist bereits eine Anforderung vorhanden, wird sie aus der Warteschlange des Sockets entfernt, andernfalls wird der Prozeß bis zum Eintreffen einer Anforderung blockiert. Zum Initiieren einer Verbindung zu einem entfernten Socket dient der Aufruf `connect`. Mittels `send` und `recv` können Nachrichten gesendet oder empfangen werden. Ein Abbau der Verbindung geschieht durch Aufruf von `shutdown`. Diese Funktionsaufrufe finden sich auch in der VCR-Software (s. Kapitel 2.6).

## 2.4 Notwendigkeit von Protokollen mit Dienstgütegarantien

Verteilte Multimedia-Anwendungen besitzen wie bereits in Kapitel 2.1 erwähnt sowohl diskrete als auch kontinuierlichen Datenströme. Sollen diese kontinuierliche Datenströme, die zudem meist noch Echtzeitanforderungen stellen, über ein Netz übertragen werden, so benötigt man gewisse Dienstgütegarantien vom Netz, damit die Daten auf der Gegenseite in akzeptabler Weise präsentiert werden können. Soll z.B. durch die Anwendung eine Sprachkommunikation ermöglicht werden, so darf die Zeit zwischen Sprechen und Wiedergabe des Gesprochenen nicht zu groß werden. Daher muß die maximale Verzögerung eines Datenpaketes bis zur Auslieferung hier stark begrenzt sein (z.B. 250 ms).

Wird IP (UDP,TCP) als Protokoll zur Datenübertragung verwendet, hat man diesbezüglich aber das Problem, daß IP aufgrund seiner best-effort QoS-Semantik, die oben genannten Dienstgütegarantien nicht gewähren kann. IP wurde für die Übertragung von Daten ohne Echtzeitanforderungen (z.B. Dateien) ausgelegt. Für diese Art von Daten reicht best-effort als QoS-Semantik völlig aus. Für die Übertragung von Multimediadaten bedeutet best-effort aber, daß Datenpakete viel zu spät oder in zu unregelmäßigen Abständen beim Empfänger eintreffen können (was zu Sprechpausen und Ruckeln führen würde), wenn andere Anwendungen zwischendurch erheblichen Datenverkehr auf dem Netz erzeugen. Dieses Verhalten ist aber je nach Ausprägung dieser 'Störungen' nicht mehr akzeptabel. Um diese Beeinflussung durch andere vermeiden zu können, benötigt man ein Protokoll, das einer Datenverbindung eine bestimmte zuvor ausgehandelte Dienstgüte garantiert.

## 2.5 Protokolle mit Dienstgütegarantien

Im folgenden werden Protokolle beschrieben, mit denen es möglich wird, für Verbindungen Dienstgüte zu garantieren.

### 2.5.1 RSVP

Das Resource Reservation Protokoll (RSVP) [RSVP96] erlaubt Anwendungen unidirektionale Datenströme zu einem oder mehreren Empfängern mit einer bestimmten garantierten Dienstgüte aufzubauen. RSVP übernimmt die Aufgabe, Ressourcen Reservierungsanforderungen der Anwendungen an die entsprechenden Knoten im Netzwerk weiterzuleiten.

Beim Entwurf von RSVP wurden folgende Ziele verfolgt [RSVP93]:

1. Unterstützung von heterogenen Empfängern:  
Innerhalb einer Multicastgruppe können Empfänger stark von einander abweichende Eigenschaften besitzen. Beispielsweise könnte durch unterschiedliche Hardware die Verarbeitungsgeschwindigkeit in den Empfängern

stark variieren. Zudem könnten die Pfade, über die die Empfänger erreicht werden, unterschiedliche Kapazitäten haben. Daher soll die Möglichkeit geschaffen werden, Reservierungen zugeschnitten an die jeweiligen Bedürfnisse der Empfänger vornehmen zu können.

2. Fähigkeit zur Anpassung an Änderungen innerhalb von Multicastgruppen: Die Mitgliedschaft in einer Multicastgruppe kann sich dynamisch ändern, es kann ein neues Mitglied zur Gruppe hinzukommen oder ein Mitglied die Gruppe verlassen. Eine Änderung wird um so wahrscheinlicher, je größer die Gruppe ist. Insbesondere bei großen Gruppen sollte daher nicht jedesmal bei Änderungen das Reservierungsprotokoll erneut gestartet werden müssen, sondern eine automatische Anpassung durch das Protokoll erfolgen.
3. Angabe der benötigten Ressourcen durch die Anwendung, um die Nutzung der Netzwerkressourcen effizient vornehmen zu können: Innerhalb einer Multicastgruppe sollte es möglich sein, auf gemeinsamen Pfadabschnitten die Ressourcen zu teilen. Durch eine genaue Angabemöglichkeit der benötigten Ressourcen können die aktuellen Bedürfnisse einer Gruppe ermittelt und so eine Verschwendung von Ressourcen vermieden werden.
4. Empfängern die Möglichkeit geben, zwischen Sendern zu wechseln: Ein Empfänger sollte die Kontrolle darüber haben, welche Pakete über die reservierten Ressourcen transportiert werden, statt der Angabe, was bei ihm lokal von den eingegangenen Informationen verworfen wird. Darüberhinaus sollte der Empfänger die Möglichkeit haben, zwischen Sendern wechseln zu können, ohne das neue Reservierungen vorgenommen werden müssen.
5. Fähigkeit zur Anpassung an Änderungen der Pfade: In einem großen Netzwerk kann es zu Pfadänderungen aufgrund von Änderungen der Auslastung oder Topologie kommen. Eine Anpassung an diese Änderungen ist die Aufgabe eines Routingprotokolls. Obwohl RSVP keine Routingfunktionalität enthält, sollten dennoch automatisch die Reservierungen von Ressourcen entlang der neuen Pfade vorgenommen werden, sofern diese dort verfügbar sind.
6. Protokolloverhead gering halten: Die Anpassung an Änderung von Pfaden sollte nicht zu aufwendigen Erkennungsmechanismen und damit verbundenen großen Nachrichtenaufkommen führen.
7. Durch modulares Design sollen heterogene unterliegende Technologien verwendet werden können.

Diese Entwurfsziele wurden durch nachfolgende Entwurfsprinzipien erreicht:

1. Reservierungsanforderungen werden vom Empfänger vorgenommen:  
Der Empfänger ist für das Anfordern von Ressourcen und der Beibehaltung von Ressourcenreservierungen für die Dauer der Verbindung zuständig. Dies geschieht in Form einer QoS-Anforderung (*Flowspec* genannt). Die Flowspec enthält Parameter wie maximale Datenrate, maximale Paketgröße und Paketrate.
2. Reservierung von Paket-Filterung trennen:  
Die Reservierung beschreibt den Umfang der reservierten Ressourcen und wer die Kontrolle darüber hat. Wohingegen die Paket-Filterung die Pakete nach einer Vorgabe des Empfängers auswählt, die diese Ressourcen nutzen dürfen. Die Vorgabe des Filters kann so dynamisch ohne Änderung der reservierten Ressourcen gewechselt werden.
3. Verschiedene Reservierungsarten bereitstellen:  
RSVP stellt drei Reservierungsarten bereit: no-filter, fixed-filter und dynamic-filter. Die Reservierungsart no-filter dient dazu keine Filterung der Datenpakete vorzunehmen lassen, so daß alle Pakete, die an diese Multicastgruppe gerichtet werden, die reservierten Ressourcen benutzen. Mit den beiden anderen Reservierungsarten kann eine Filterung der Datenpakete nach Absender vorgenommen werden, damit nur diese die reservierten Ressourcen nutzen können. Die Einstellung fixed-filter bedeutet dabei, daß für die gesamte Dauer der Verbindung nur Datenpakete der bei der Reservierungsanforderung angegebenen Sender an den Empfänger weitergereicht werden. Wohingegen bei der Reservierungsart dynamic-filter der Empfänger die Filterung während der Verbindung ändern und somit zwischen Sendern wechseln kann.
4. 'Soft state' Ansatz im Netzwerk:  
Dieser Ansatz bedeutet, daß Reservierungen einer zeitlichen Begrenzung unterliegen (timeout), bis zu der sie erneuert werden müssen, wenn sie weiter aufrecht erhalten werden sollen. Dies bietet den Vorteil, daß bei einer Verbindungsunterbrechung nach einiger Zeit auf jeden Fall die zugehörigen Ressourcen automatisch frei gegeben werden. Dies gilt sowohl für eine ungewollte Unterbrechung als auch für eine gewollte (z.B. wenn der Sender gewechselt wird). Durch diesen Ansatz kann auch der Kommunikationsaufwand bei Änderungen der Multicastgruppe minimiert werden. Diese Vorteile bedeuten aber auch einen gewissen Overhead, da jetzt periodisch refresh-Nachrichten verschickt werden müssen.
5. Protokolloverhead minimieren:  
Der RSVP Overhead ergibt sich durch die versendeten RSVP Nachrichten und deren Größe. Einen entscheidenden Einfluß hat damit die Häufigkeit mit der refresh-Nachrichten für die Reservierungen versendet werden. Daher versucht RSVP verschiedene Reservierungsnachrichten zu einer zusammenzufassen.

## 6. Modularität:

Bei dem Entwurf von RSVP wurde darauf geachtet, an von RSVP benötigte Komponenten möglichst wenig Voraussetzungen zu stellen. Eine Beschreibung der Voraussetzungen erfolgt nach der Darstellung der RSVP umgebenden Architektur.

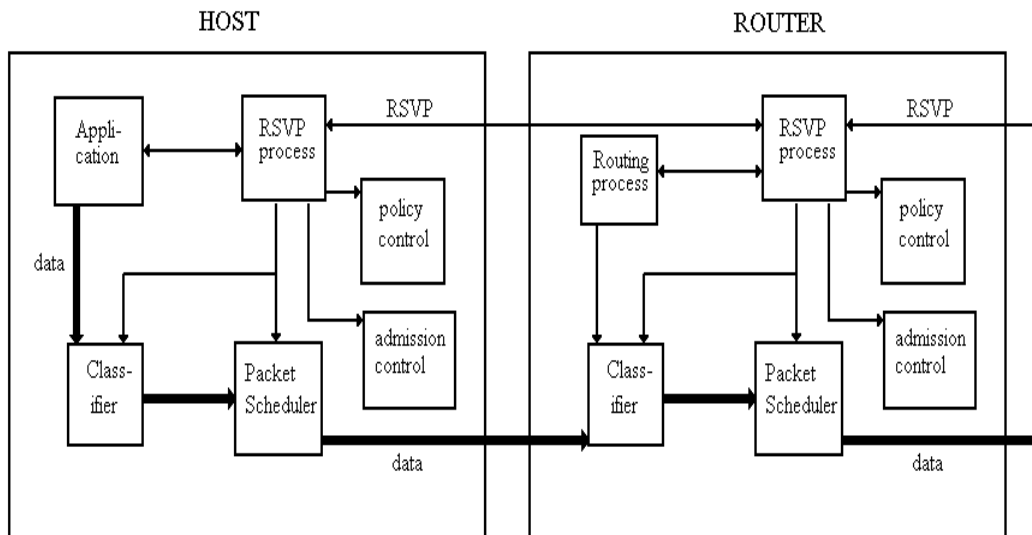


Abbildung 2.3: RSVP in Hosts und Routern

Abbildung 2.3 zeigt die umgebende Architektur des RSVP-Protokolls. Der *RSVP Prozeß* ist für das Weiterleiten der RSVP Nachrichten zuständig. Die *admission control* verwaltet die lokalen Ressourcen, insbesondere auch, welche Ressourcen bereits reserviert sind und welche noch zur Verfügung stehen. Die *policy control* ist für die Überprüfung der Berechtigung einer Reservierung zuständig. Der *Classifier* ermittelt die zu einem Datenpaket zugehörige QoS-Klasse, anhand der im Paketheader enthaltenen Informationen über den Absender. Der *packet scheduler* entscheidet, wann ein Datenpaket an den nächsten Knoten weitergeleitet wird.

RSVP besitzt folgende Voraussetzungen:

Für den Aufbau der Pfade wird ein Routingprotokoll benötigt, daß sowohl unicast als auch multicast Routing zur Verfügung stellt. RSVP ist nur für das Übertragen der Flowspec zuständig. Die Interpretation der Flowspec und die Vergabe durch Ressourcen geschieht durch die *admission control*.

RSVP kennt zwei Arten von Nachrichten, die RESV-Nachricht und die PATH-Nachricht. Mittels RESV wird die QoS-Anforderung vom Empfänger zum Sender geschickt. Damit den zwischen Sender und Empfänger liegenden Routern jeweils die vorherige Station bekannt ist, werden entlang der zuvor erstellten Route vom Sender zum Empfänger PATH-Nachrichten verschickt. Dies geschieht jeweils von Zwischenstation zu Zwischenstation, wobei jede PATH-Nachricht die IP-Adresse der vorhergehenden Zwischenstation enthält. Somit kann dann für die

QoS-Anforderung die Route zwischen Sender und Empfänger rückwärts durchlaufen werden.

Nun zur Funktionsweise:

Die Anwendung des Empfängers stellt eine entsprechende QoS-Anforderung an den lokalen RSVP Prozeß, welcher daraufhin die Anforderung mittels eines Kontrollpaketes zu allen Knoten des Datenpfades bis hin zum Sender befördert. In jedem dieser Knoten wird das Kontrollpaket dann an die admission control und die policy control weitergeleitet. Entscheidet sowohl die admission control, daß der Knoten für die Erfüllung der Anforderung noch genügend freie, noch nicht reservierte Ressourcen besitzt, als auch die policy control, daß eine Berechtigung für die Reservierung vorliegt, so werden entsprechende Parameter an den packet classifier und den packet scheduler übergeben. Bei einer negativen Antwort wird eine Fehlerbenachrichtigung an den Absender der QoS-Anforderung gesendet. In einen Knoten eingehende Datenpakete passieren dann zuerst den packet classifier, der anhand der bei der Reservierung übergebenen Parameter eine Zuordnung zu der entsprechenden QoS-Klasse vornimmt. Daraufhin kann der packet scheduler dann entscheiden, wann das Paket weitergeleitet wird.

Als Besonderheit des Resource Reservation Protokolls soll an dieser Stelle nochmals die Möglichkeit hervorgehoben werden, die Dienstgüte einer Verbindung dynamisch ändern zu können. Bei Anwendungen mit burstartigen Datenverkehr ist somit eine genauere Reservierung von benötigten Ressourcen möglich, weil nicht für die gesamte Dauer der Verbindung Ressourcen belegt werden müssen, die dem Maximaldurchsatz entsprechen. Stattdessen kann die Reservierung von Ressourcen den aktuellen Bedürfnissen entsprechend vorgenommen werden.

Ein Vergleich mit den nachfolgend vorgestellten Protokollen erfolgt in Kapitel 2.5.5.

### 2.5.2 ST2

Beim Internet Stream Protokoll Version 2 (ST2) [RFC1819] handelt es sich um ein experimentelles verbindungsorientiertes Ressourcen-Reservierungsprotokoll, das als Entwurfsziel hatte, Ende-zu-Ende Echtzeit-Garantien bereitzustellen. Es erlaubt Anwendungen unidirektionale Datenströme zu einem oder mehreren Empfängern mit einer bestimmten garantierten Dienstgüte aufzubauen. Diese Garantie wird durch Bandbreiten-Reservierung entlang des Übertragungsweges zusammen mit entsprechenden Netzwerkzugangs- und Paket-Schedulingmechanismen auf allen Knoten sichergestellt. Diese Maßnahmen gewährleisten, daß Pakete mit Echtzeitanforderungen innerhalb ihrer Deadlines ausgeliefert werden.

ST2 besteht aus zwei Protokollen: ST für den Datentransport und SCMP (Stream Control Message Protokoll) für alle Kontrollfunktionen. Typische SCMP-Nachrichten sind CONNECT und ACCEPT um einen Stream aufzubauen, DISCONNECT und REFUSE um einen Stream zu schließen, CHANGE um die Dienstgüte eines Streams zu verändern und JOIN um eine Anforderung zu stellen, einem Stream hinzugefügt zu werden.

ST2 basiert auf einem zwei Phasen-Kommunikationsmodell. Im ersten Schritt werden die Echtzeitkanäle (im Standard mit Streams bezeichnet) für den nachfolgenden Datentransfer aufgebaut. Dies beinhaltet die Wegewahl zum Ziel und das sender-basierte Reservieren der benötigten Ressourcen. Im Gegensatz zu RSVP ist hier also der Sender für das Aushandeln der Dienstgüte zuständig. In der zweiten Phase geschieht dann der Datentransfer über die zuvor eingerichteten Kanäle.

Das Kommunikationsmodell von ST2 besteht aus folgenden Komponenten: einem Setup-Protokoll, einem Datentransferprotokoll, einer Anforderungsspezifikation, einer Routingfunktion und einem lokalen Ressourcenverwalter.

Das Setup-Protokoll dient dem Aufbau, der Aufrechterhaltung und dem Abbau von Echtzeitströmen bzw. -kanälen.

Das Datentransferprotokoll ist für die Übermittlung der Echtzeitdaten zu den Zielen entlang der zuvor durch das Setup-Protokoll erzeugten Kanäle zuständig. Die Datenpakete enthalten einen global eindeutigen Streambezeichner, wodurch eine Zuordnung zu einem Kanal sichergestellt wird.

Mittels der Anforderungsspezifikation (mit FlowSpec bezeichnet) werden die QoS-Anforderungen der Anwendung festgelegt. Diese wird dann bei jedem Knoten vom lokalen Ressourcenverwalter benutzt, um entsprechende Ressourcen zu belegen und damit den Anforderungen zu genügen. Mittels eines Versionsfeldes ist ST2 in der Lage verschiedene Formate von Anforderungsspezifikationen zu unterstützen. Dabei sind zwei Formate zwingend vorgeschrieben: die Null FlowSpec und die ST2+ FlowSpec. Die Null FlowSpec ist zum Testen des Streamaufbaus vorgesehen. Zum Beschreiben der Echtzeitanforderungen einer Anwendung ist die ST2+ FlowSpec vorgesehen. Diese besteht aus der Angabe einer QoS Klasse, einer Vorrangenteilung und den drei QoS Grundparametern Nachrichtengröße, Nachrichtenrate und Ende-zu-Ende Verzögerung.

Es wurden zwei QoS Klassen definiert: 'predictive' und 'guaranteed'. Bei 'predictive' darf die ausgehandelte Dienstgüte für kurze Zeitintervalle während des Datentransfers verletzt werden, wohingegen bei 'guaranteed' die ausgehandelte Dienstgüte strikt eingehalten werden muß. Für die Anwendung bedeutet letzteres, daß sie Reservierungen für das maximale von ihr erzeugte Datenaufkommen, also der peak rate, vornehmen muß. Dies hat den Nachteil, daß Ressourcen reserviert werden, die eventuell kaum genutzt werden.

Die Vorrangenteilung bestimmt die Wichtigkeit, eine Verbindung aufzubauen. Dadurch wird also geregelt, welche Verbindung bei der Vergabe von Ressourcen bevorzugt behandelt wird.

Die Nachrichtengröße bezeichnet das Maximum an Daten, das an einem Stück versendet werden darf. Die Nachrichtenrate gibt an, wieviele Nachrichten pro Sekunde versendet werden dürfen. Die Ende-zu-Ende Verzögerung gibt die maximale Verzögerung für einen Stream vom Aussenden bis zum Empfang einer Nachricht in Millisekunden an. Für diese Grundparameter kann die Anwendung ein Intervall angeben. Als untere Schranke dient dabei ein Wert, den die Anwendung schlechtestenfalls akzeptieren kann und als obere Schranke wird der von der Anwendung gewünschte Wert vorgegeben. In jedem Knoten wird vom

lokalen Ressourcenverwalter überprüft, ob der gewünschte Wert erfüllt werden kann. Ist dies nicht möglich, aber eine Erfüllung der Anforderung oberhalb der vorgegebenen unteren Schranke, wird in der Anforderungsspezifikation das Feld für den verwendeten Wert entsprechend angepaßt. Kann auch die untere Vorgabe nicht eingehalten werden, so kann der Stream nicht entlang dieses Pfades aufgebaut werden.

Weitere mögliche FlowSpecs sind z.B. die HeiTS, BerKom Flowspec und die von [RFC1190] und [RFC1363]. Diese beinhalten Parameter wie durchschnittlichen und maximalen Durchsatz, Ende-zu-Ende-Verzögerung und Verzögerungsvarianz eines Streams.

Die Weiterleitung der Anforderungsspezifikation an jeden Knoten geschieht durch das Setup-Protokoll. Die endgültige Anforderungsspezifikation wird mit einer ACCEPT-Nachricht von jedem Empfänger zurück an den Sender übermittelt. Ein Ändern der Dienstgüte und damit der belegten Ressourcen während der Datenübertragungsphase ist durch den Sender mittels einer CHANGE-Nachricht möglich.

Die Routingfunktion stellt dem Setup-Protokoll einen Pfad zu einem Ziel zur Verfügung. Dieser Pfad bleibt dann für die gesamte Lebensdauer des Streams gleich.

Die Aufgabe des lokalen Ressourcenverwalters ergibt sich unmittelbar aus dem Namen. Ressourcen können dabei z.B. Pufferplatz zum Zwischenspeichern von Daten und Netzwerkadapter sein. Zusätzlich muß hier anhand der Anforderungsspezifikation überprüft werden, ob noch genügend freie Ressourcen für den neuen Kanal zur Verfügung stehen. Falls dies nicht der Fall ist, wird diese Anforderung abgelehnt. Während des Datentransfers wird durch entsprechendes Scheduling des Ressourcenzugriffs die QoS-Anforderungen sichergestellt, indem z.B. Datenpakete von Anwendungen mit kürzeren garantierten Verzögerungszeiten vor Datenpaketen mit weniger strikten Verzögerungszeiten weitergeleitet werden. Gegebenenfalls wird durch ein Prioritätsfeld die Reihenfolge geklärt.

Um die Anzahl an belegten Ressourcen durch Teilung der Ressourcen zwischen mehreren Streams (Bandwidth Sharing) zu minimieren oder belegte Ressourcen im Fehlerfall gleich zu behandeln (Fate Sharing), können Streams zusammengruppiert werden. Dies macht z.B. bei einer Sprachkonferenz Sinn, bei der immer nur ein paar der Teilnehmer gleichzeitig sprechen und somit die Ressourcen zwischen den Teilnehmern geteilt werden können.

### 2.5.3 RTP

Das *Real Time Transport Protokoll* (RTP) [RFC1889, RTP96] stellt Anwendungen mit Echtzeit-Eigenschaften (z.B. Anwendungen, mit denen Audio- oder Videokonferenzen ermöglicht werden) Ende-zu-Ende Transportfunktionen zur Verfügung und liefert durch Monitoring den Kommunikationspartnern Informationen über die Netzwerkleistung, damit diese auf Veränderungen der Übertragungsqualität reagieren können. Eine weitere Unterstützung für Anwendungen

wird in Form von Zeitstempel und Sequenznummern geboten, die zur Synchronisation und Reihenfolgeerhaltung dienen.

Die wichtigsten Ziele bei der Entwicklung von RTP waren die folgenden:

1. Flexibilität gegenüber Anwendungen:  
Um möglichst viele Anwendungen unterstützen zu können, stellt RTP lediglich häufig benötigte Funktionen bereit. Bei Bedarf kann RTP durch anwendungsabhängige Funktionen ergänzt werden.
2. Unabhängigkeit von unterliegenden Protokollen:  
RTP kann auf verschiedene Protokolle aufsetzen, beispielsweise UDP oder auch ST-2.

An die unterliegenden Protokolle stellt RTP kaum Bedingungen. Benötigt werden eine Demultiplex-Funktionalität, um Daten- und Kontrollströme voneinander trennen zu können (bei Verwendung von UDP als unterliegendes Protokoll durch Verwendung unterschiedlicher Portnummern), eine Längenangabe des Paketes, da ein RTP-Paket diese nicht selbst enthält. Soll RTP Multicast-Kommunikation bieten, muß dies von den unterliegenden Protokollen zur Verfügung gestellt werden. Dagegen werden keine Annahmen über die Zuverlässigkeit der unterliegenden Protokolle gemacht.

RTP besteht aus zwei Teilen, dem Real Time Transport Protokoll (RTP) für die eigentliche Datenübertragung und dem RTP Control Protokoll (RTCP) für die Überwachung der zur Verfügung stehenden Dienstgüte und den Austausch von Informationen über die Teilnehmer einer RTP-Sitzung.

*RTP* bietet die folgenden Funktionen:

- Identifikation des Nutzlasttyps
- Sequenznummerierung
- Zeitstempel

Datenpaketen eines Senders wird durch RTP eine Sequenznummer und Zeitstempel hinzugefügt, auf Empfängerseite werden diese Informationen aber nicht von RTP ausgewertet, sondern erst von der darüber liegenden Anwendung. Dies steht im Gegensatz zu sonst üblichen Protokollarchitekturen, wo normalerweise beim Sender hinzugefügte Informationen auf Empfängerseite auf der zur Sender korrespondierenden Ebene ausgewertet werden.

Der Header eines RTP-Paketes enthält ein Feld, das den Typ der Nutzlast (bzw. des verwendeten Kodierungsverfahrens) angibt. Dadurch wird eine dynamische Änderung des Kodierungsverfahrens im Verlauf einer Sitzung möglich (um auf eine Änderung der Übertragungsqualität zu reagieren).

Die Sequenznummer wird bei jedem gesendeten RTP-Paket automatisch um den Wert 1 erhöht. Dies ermöglicht dem Empfänger die ursprüngliche Reihenfolge beim Versenden wieder herzustellen.

Der Zeitstempel beschreibt den Abtastzeitpunkt des ersten Bytes der übertragenden Dateneinheit. Somit wird eine Synchronisation innerhalb eines Datenstromes sowie zwischen unterschiedlichen Datenströmen möglich.

*RTCP* stellt durch den periodischen Austausch von Kontrollpaketen unter den Sitzungsteilnehmern Informationen über die Übertragungsqualität zur Verfügung. Ein Sender kann anhand dieser Informationen seine Übertragungsparameter (z.B. Kodierungsverfahren) an die momentane Netzsituation anpassen. Durch die Kontrollpakete können zudem Netzwerkprobleme erkannt werden.

Sind Sitzungsteilnehmer an unterschiedlich leistungsfähige Netzsegmente angeschlossen, ermöglicht RTP durch sogenannte Mixer und Übersetzer, daß, statt alle Teilnehmer zu einer Kodierung für eine schmalbandige Übertragung zu zwingen, für die Teilnehmer entsprechend der Leistungsfähigkeit der Netzsegmente, an die sie angeschlossen sind, die Kodierung gewählt wird. Der Mixer ordnet für die Teilnehmer eines breitbandigen Netzes die Daten nur neu an, wohingegen bei einem schmalbandigen Anschluß die Daten erst resynchronisiert werden und dann in eine dafür geeignete Kodierung übersetzt werden.

RTP unterscheidet sich also sehr von den oben vorgestellten Protokollen. Es bietet zwar eine Unterstützung für Anwendungen, die zeitkritische Daten verschicken, die Auswertung von durch RTP hinzugefügten Informationen muß aber von der Anwendung durchgeführt werden.

Im Gegensatz zu RSVP und ST2 führt RTP weder Ressourcen Reservierungen durch, noch garantiert es eine Dienstgüte. Zudem stellt es keine Mechanismen bereit, um eine Auslieferung innerhalb zeitlicher Schranken von sich aus zu gewährleisten (obwohl der Name dies vermuten läßt). Es stellt auch keine Fluß- oder Überlaufkontrolle noch Zuverlässigkeit oder Reihenfolgeerhaltung bereit.

### 2.5.4 Asynchroner Transfer Modus (ATM)

Die Forderung nach Dienstintegration, z.B. der gleichzeitigen Übertragung von Audio, Video und Daten wird immer stärker. Durch die Integration unterschiedlicher Anwendungen werden gegenüber der reinen Datenkommunikation bisher nicht bekannte Anforderungen an die Übertragungs- und Vermittlungstechnik gestellt. Traditionelle Techniken wurden entweder für Sprach- oder Datenkommunikation aber nicht für die gleichzeitige Benutzung beider entworfen und sind deshalb nur bedingt für die Unterstützung neuartiger dienstintegrierender Anwendungen geeignet. Daher wurde der *Asynchrone Transfer Modus* (ATM) eingeführt, um der Forderung nach Dienstintegration einfacher entsprechen zu können.

#### 2.5.4.1 ATM-Konzepte

In [Zitt95] werden die ATM-Konzepte wie folgt beschrieben:

*Der asynchrone Transfermodus ATM wurde im Hinblick auf die Realisierung dienstintegrierender Kommunikationsnetze entworfen. ATM gewinnt dabei insbesondere durch die hohen und vielfältigen Dienstanforderungen, die sich im Bereich fortgeschrittener Anwendungen abzeichnen, zunehmend an Bedeutung, da er das zu deren Realisierung notwendige Leistungspotential sowie die Flexibilität in der Diensteunterstützung bereitstellt. Als herausragendes Merkmal der ATM-Technologie wird die Kombination aus Dateneinheiten fester Länge (sog. ATM-Zellen) und virtuellen Verbindungen im Netz angesehen.*

Ein wesentliches Konzept bei ATM ist darüberhinaus, die Vorteile von Paket- und Leitungsvermittlung zu vereinen (s. Kap. 2.2.3).

ATM-Zellen haben eine feste Länge von 53 Bytes, bestehend aus 5 Bytes für den Kopf und 48 Bytes für Nutzdaten (bzw. auch für Kontrollinformationen höherer Schichten). Diese Zellen werden für die Übertragung asynchron auf eine Leitung gemultiplext. Durch die feste Länge können die Zellen durch die netzinternen Vermittlungsknoten wesentlich effizienter behandelt werden als variabel lange Pakete, da die Längenanalyse entfällt. Der Kopf jeder Zelle (s. Abbildung 2.4) enthält unter anderem Information für die Vermittlung der Zellen im Netz.

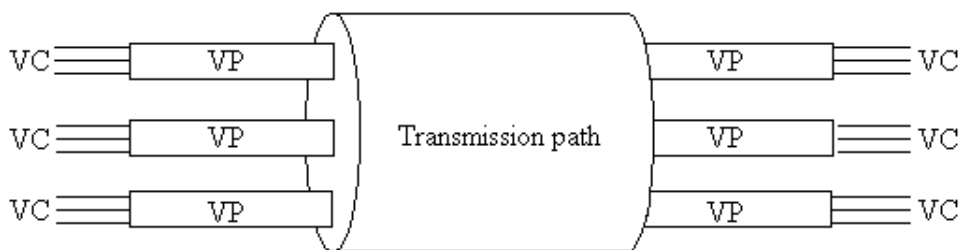
GFC	VPI		Byte 1
VPI	VCI		Byte 2
VCI			Byte 3
VCI	PTI	CLP	Byte 4
HEC			Byte 5

GFC: Generic Flow Control      VPI: Virtual Path Identifier  
VCI: Virtual Channel Identifier    PTI: Payload Type Identifier  
CLP: Cell Loss Priority            HEC: Header Error Control

Abbildung 2.4: ATM-Zellenkopf an der Teilnehmer-Netz-Schnittstelle

Die Länge der ATM-Zelle ist von entscheidender Bedeutung, wenn eine Vielzahl von Diensten (insbesondere Audio und Daten) integriert werden soll. Datenverkehr verlangt, im Gegensatz zum Sprachverkehr, in der Regel größere Längen für die Übertragungseinheiten, um den zusätzlichen Overhead (durch die Übertragung des Zellkopfes) niedrig zu gestalten. Für den Sprachverkehr sind relativ kurze Pakete notwendig, um Verzögerungen beim Senden möglichst klein zu halten.

Um hohe Datenraten von 155 Mbit/s, 622 Mbit/s oder auch 2.4 Gbit/s erreichen zu können, ist es wichtig, die Aufgaben der einzelnen involvierten Netzkomponenten so einfach wie möglich zu gestalten. Für ATM hat dies insbesondere zur Folge, daß Fehlererkennungs- und -behebungsfunktionen weitgehend aus den netzinternen ATM-Vermittlungsknoten in die an den Grenzen eines ATM-Netzes angesiedelten ATM-Endsysteme ausgelagert werden. Die ATM-Vermittlungsknoten (auch ATM-Switches genannt) sind damit im wesentlichen für das reine Weiterleiten von ATM-Zellen verantwortlich.



VC Virtual channel

VP Virtual path

Abbildung 2.5: virtuelle Kanäle, virtuelle Pfade und Übertragungsweg

In Abbildung 2.5 wird der Zusammenhang zwischen virtuellen Kanälen, virtuellen Pfaden und einem Übertragungskanal dargestellt. Ein Übertragungskanal kann mehrere virtuelle Pfade umfassen und jeder virtuelle Pfad kann aus mehreren virtuellen Kanälen bestehen. Dieses Konzept erlaubt das Gruppieren von mehreren virtuellen Kanälen mit den gleichen Endpunkten, d.h. gleicher Quelle und gleichem Ziel. Dies macht dann Sinn, wenn eine Anwendung einen aus mehreren Datenströmen bestehenden Dienst realisieren soll. Ebenso kann ein Nutzer einen virtuellen Pfad bei einem Netzbetreiber mieten und kann die darin enthaltenen virtuellen Kanäle nutzen.

Datenübertragung zwischen zwei Benutzer geschieht bei ATM verbindungsorientiert, es wird also vor dem eigentlichen Datenaustausch eine virtuelle Verbindung aufgebaut. Dies geschieht durch entsprechende Tabelleneinträge in den Vermittlungsknoten, durch die eine Zuordnung zwischen den Eingangs- und Ausgangspunkten festgelegt wird. Durch die in jeder Zelle mitgeführte Marken (VPI, VCI s. Abbildung 2.4) kann eine eindeutige Zuordnung zu einer logischen Verbindung

stattfinden. Vergleicht man die Vermittlung in ATM-Netzen mit der Leitungsvermittlung so besteht ein wesentlicher Unterschied darin, daß sich lediglich die momentan aktiven Verbindungen die Bandbreite auf dem Übertragungsmedium teilen, also keine starre Kanalzuordnung existiert. Die Reihenfolge der Zellen einer Verbindung bleibt erhalten, es werden lediglich Zellen unterschiedlicher Verbindungen miteinander gemultiplext. Letzteres könnte allerdings dazu führen, daß es zu, für manche Anwendungen (z.B. Videoanwendungen) nicht mehr akzeptablen Schwankungen im Datenstrom kommt. Dies kann durch entsprechende Angaben beim Verbindungsaufbau (Signalisierungsphase) verhindert werden, in dem entsprechende Ressourcenreservierungen angefordert werden. Dies geschieht durch die Angabe entsprechender Dienstgüteparameter (quality of service). Diese beeinflussen die Vergabe von Pufferspeicher oder auch die Bearbeitungsreihenfolge der beim Vermittlungsknoten eingehenden Zellen. Dabei sind die maximale Zellenrate und der akzeptierbare Jitter (Variation in der Verzögerung) Pflichtparameter. Das Netzwerk muß dann entscheiden, ob die Anforderungen erfüllt werden können. Falls dies der Fall ist, wird die Verbindung aufgebaut und das Netzwerk muß dann die Erfüllung der Dienstanforderungen für den gesamten Zeitraum der Verbindung garantieren. Es wird sozusagen ein Vertrag zwischen Anwendung und Netz abgeschlossen.

In [Flan94] sind einige Dienstgüteparameter aufgeführt, die in der Datentransferphase von Interesse sind:

- cell loss ratio: fehlerhafte Zellen geteilt durch Gesamtzahl Zellen (gute und fehlerhafte) in einem Zeitintervall
- cell transfer delay: Zeit vom ersten gesendeten Bit im Netzwerk bis das letzte Bit der Zelle beim Ziel eintrifft.
- cell delay variation: Maß für das 'Klumpen' von Zellen, wenn sie das Netzwerk passieren. Durch dieses Klumpen entstehen Lücken, so daß es zu Verzögerungen im Datenstrom kommt.
- sustainable cell rate: (in Zellen pro Sekunde) durchschnittlicher oder konstanter Verkehr, der von einem Sender ausgehen darf.
- peak cell rate: (in Zellen pro Sekunde) Gibt die maximal zulässige Zellrate an. Zellklumpen erhöhen die momentane Rate, nicht aber den Durchschnitt. Daher sollte die peak rate immer etwas größer gewählt sein als die sustainable cell rate.
- Burst tolerance / maximum burst size: Gibt an, wie lange mit peak cell rate gesendet werden darf (ohne dabei die sustainable cell rate zu überschreiten).

In [Zitt95] werden darüberhinaus noch folgende Dienstparameter beschrieben:

- Bitfehlerrate: Rate der Bitfehler im Nutzdatenfeld zu den Bitfehlern in den gesamten, im Verlauf einer Verbindung übertragenen, Nutzdaten.

- **Ende-zu-Ende-Verzögerung:** Zeit, die zwischen dem Senden des ersten Bits und dem Empfangen des letzten Bits einer Zelle verstreicht. Diese Zeit setzt sich aus Verarbeitungs- und Übertragungsverzögerungen zusammen und wird zudem von den Warteschlangen in den Vermittlungsknoten beeinflusst.

Die vom ATM-Forum definierte Teilnehmer-Netz-Schnittstelle (User Network Interface, UNI) verfügt über die Verkehrsparameter maximale Zellenrate (peak cell rate), burst tolerance und einen Parameter zur Definition der mittleren vom Netz aufrechtzuerhaltenden Zellenrate (sustainable cell rate).

#### 2.5.4.2 ATM-Referenzmodell

Das ATM-Referenzmodell [I321] ist in Abbildung 2.6 dargestellt. Gegenüber dem OSI-Referenzmodell hebt es sich durch vertikale Ebenen ab, zusätzlich zu den horizontalen Schichten. Es werden die folgenden Ebenen unterschieden:

- die Benutzerebene,
- die Kontrollebene und
- die Managementebene.

Die *Benutzerebene* ist für den Transport von Benutzerdaten zuständig. Sie enthält Funktionen zur Fluß- und Fehlerkontrolle. Höheren Schichten stellt die Benutzerebene individuell zugeschnittene Dienste zur Verfügung.

Die *Kontrollebene* enthält Signalisierungsfunktionen für den Aufbau, die Überwachung und den Abbau von Verbindungen.

Die *Managementebene* besteht aus zwei unterschiedlichen Kategorien von Managementfunktionen, dem *Ebenenmanagement* und dem *Schichtenmanagement*. Das Ebenenmanagement ist für die Koordination zwischen den verschiedenen Ebenen verantwortlich und unterliegt keiner geschichteten Architektur.

Die Funktionen für das Schichtenmanagement sind jeweils einer spezifischen Schicht zugeordnet. Diese Funktionen dienen der Verwaltung von Ressourcen und Protokollparametern der aktiven Protokollinstanzen.

Es werden die folgenden Schichten unterschieden:

- die physikalische Schicht
- die ATM-Schicht und
- die ATM-Adaptionsschicht.

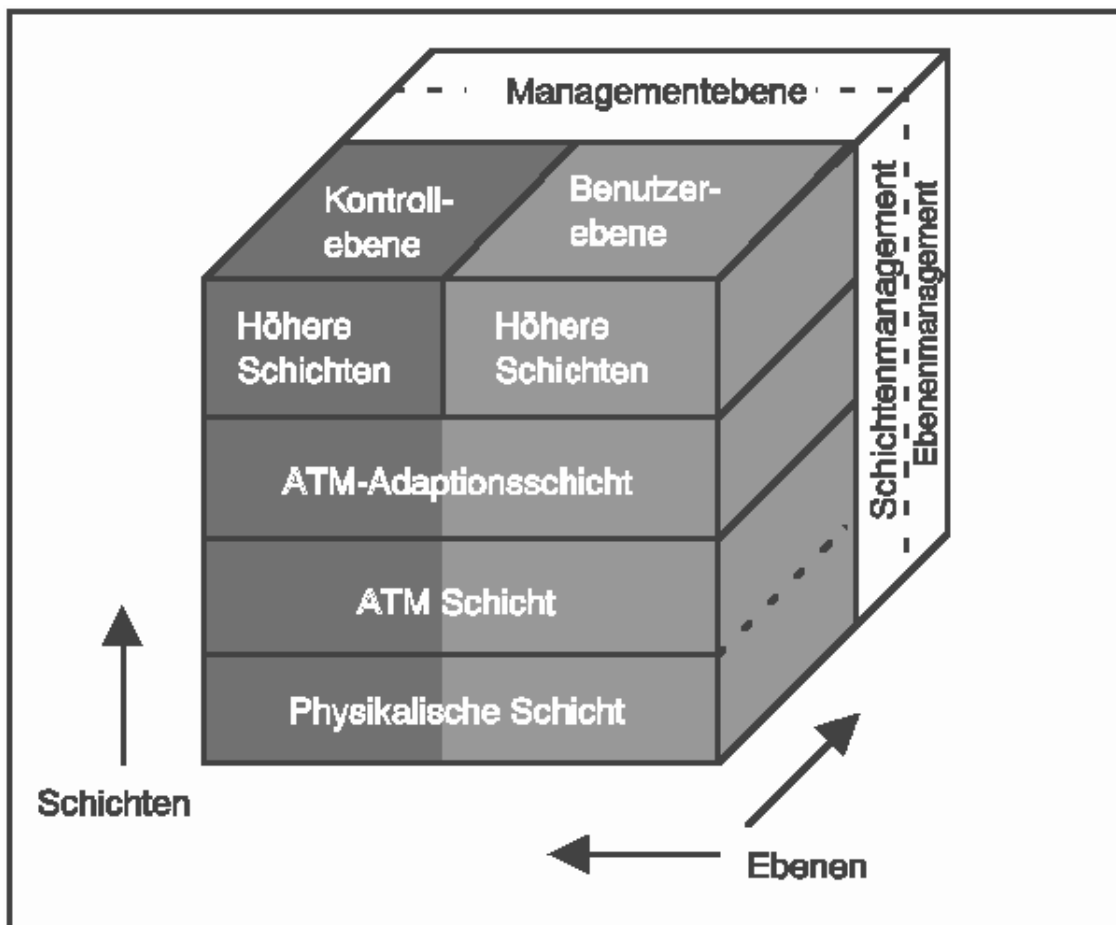


Abbildung 2.6: ATM-Referenzmodell

Die *physikalische Schicht* besteht aus zwei Teilschichten, der Teilschicht des physikalischen Mediums (Physical Medium Sublayer, PM-Teilschicht) und der Übertragungskonvergenz-Teilschicht (Transmission Convergence Sublayer, TC-Teilschicht). Die *PM-Teilschicht* enthält die von dem jeweils eingesetzten physikalischen Medium (z.B. Kupfer- oder Glasfaserkabel) abhängigen Funktionen. Sie ist für die Übertragung von Bits auf dem Medium verantwortlich und enthält die dafür notwendigen Funktionen.

Die *TC-Teilschicht* ist vom Medium unabhängig. Hier werden Zellgrenzen erkannt, Übertragungsrahmen generiert und eine Prüfsumme für die ATM-Zelle berechnet.

Die *ATM-Schicht*, die oberhalb der physikalischen Schicht liegt, umfaßt Funktionen zur Generierung und Extrahierung des Zellenkopfes zu sendender bzw. empfangener Zellen. Sie ist auch für Multiplexen und Demultiplexen von Zellen im Netz verantwortlich. Die ATM-Schicht ist weiterhin für das korrekte Vermitteln von Zellen basierend auf den darin enthaltenen VPI- und VCI-Kennungen zuständig.

Die ATM-Schicht ist, wie auch die physikalische Schicht, Teil der ATM-

Vermittlungsknoten im Netz, wobei die Eigenschaften der ATM-Schicht unabhängig von der Ausprägung der physikalischen Schicht sind.

Die *ATM-Adaptionsschicht* (eine genauere Erläuterung folgt in Kapitel 2.5.4.4) nimmt die Anpassung der von den höheren Schichten geforderten (s. Kapitel 2.5.4.3) und dem von der ATM-Schicht erbrachten Diensten vor. Dazu gehört die Fehlerkontrolle und die Synchronisation. Darüberhinaus ist diese Schicht für das Segmentieren und Reassemblieren von Zellen zu größeren Protokolldateneinheiten zuständig. Die Funktionen der ATM-Adaptionsschicht sind abhängig vom jeweils zu erbringenden Dienst an der Dienstschnittstelle der ATM-Adaptionsschicht zu den Anwendungen.

Die Funktionen der physikalischen Schicht und der ATM-Schicht sind für die Benutzer- und Kontrollebene identisch, dagegen können in der ATM-Adaptionsschicht und den höheren Schichten auf den diesen Ebenen unterschiedliche Funktionen Verwendung finden.

Eine Abbildung der Schichten des ATM-Referenzmodells auf das OSI-Referenzmodell ist nicht eindeutig möglich, daher gibt es in der Literatur durchaus unterschiedliche Sichtweisen. Der Dienst der ATM-Schicht wird ungefähr mit demjenigen der physikalischen Schicht im OSI-Referenzmodell gleichgesetzt, wobei die ATM-Schicht allerdings mehr Fähigkeiten anbietet. Die ATM-Adaptionsschicht wird, nachdem die Verbindung etabliert ist, mit der Sicherungsschicht des OSI-Referenzmodells verglichen.

Manche vergleichen die ATM-Adaptionsschicht auch direkt mit der Transportschicht des OSI-Referenzmodells. Die ATM-Schicht könnte in diesem Sinne wegen der integrierten Vermittlungsfunktionalität mit der Schicht 3 des OSI-Referenzmodells assoziiert werden.

### 2.5.4.3 Dienstklassen

Zur Unterscheidung der verschiedenen Arten für Informationsübermittlung erwiesen sich folgende Kriterien als nützlich [Zitt95]:

- Zeitbeziehung zwischen Sender und Empfänger: benötigt/nicht benötigt
- Bitrate: konstant oder variabel
- Verbindungsart: verbindungslos oder verbindungsorientiert

Dienste bei denen die *Zeitbeziehung* benötigt wird, sind z.B. Audio- und Videoübertragung, da dabei die *Verzögerung* zwischen Sender und Empfänger in bestimmten Grenzen liegen muß, um den Benutzererwartungen zu genügen. Keine *Zeitbeziehung* wird dagegen z.B. bei der Datenübertragung zwischen LAN's benötigt.

Ein Beispiel, wo die *Bitrate* variabel ist, stellt die klassische Datenkommunikation dar, bei der Daten in unregelmäßigen Abständen über das Netz gesendet werden.

Dagegen liefern Sprach- und unkomprimierte Videoübertragung eine konstante Bitrate.

Aus den Kriterien mögliche Kombinationen enthält Abbildung 2.7.

Dienst	Klasse A	Klasse B	Klasse C	Klasse D
Merkmal				
Zeitbeziehung	benötigt		nicht benötigt	
Bitrate	konstant	variabel		
Verbindungsart	verbindungsorientiert			verbindungslos
Beispiel-dienste	Fern-sprechen	deltamod. Video	Daten-übertrag.	Verbindung zwischen LAN's
empfohlener AAL-Typ	Typ 1	Typ2	Typ 3/4,5	Typ 3/4

Abbildung 2.7: Dienstklassen und deren ATM-Adaption Layers

*Klasse A* ist für Anwendungen mit konstanten Bitraten (z.B. Sprachübertragung) ausgelegt. Sie emuliert sozusagen einen leitungsvermittelten Dienst. Als typische Anwendung der Klasse A kann das Telefonieren angesehen werden.

Für Anwendungen mit variablen Bitraten (z.B. Übertragung von komprimiertem Video) ist *Klasse B* gedacht. Durch die bestehende Zeitbeziehung zwischen Sender und Empfänger können zudem traditionelle Realzeitdienste (z.B. entfernte Steuerungen) realisiert werden.

Die Klassen C und D sind für die Realisierung von traditionellen Datendiensten zuständig. *Klasse C* stellt einen verbindungsorientierten Datendienst mit variabler Bitrate ohne Zeitbeziehung zwischen Sender und Empfänger zur Verfügung. Die *Klasse D* unterscheidet sich von Klasse C nur dadurch, daß sie nur einen verbindungslosen Datendienst zur Verfügung stellt.

#### 2.5.4.4 ATM-Adaptionsschicht

Intern kann die ATM-Adaptionsschicht (ATM Adaption Layer, AAL) [Zitt95] in zwei Unterschichten aufgeteilt werden (s.Abb. 2.8):

- die Segmentier- und Reassembler-Schicht (SAR)
- und die Konvergenzschicht.

Die *SAR-Schicht* folgt unmittelbar auf die ATM-Schicht und hat im wesentlichen die Aufgabe zu sendende Pakete in ATM-Zellen zu segmentieren und empfangene Zellen in Pakete zu reassemblieren.

Die *Konvergenzschicht* ist oberhalb der SAR-Schicht angesiedelt. Ihr interner Aufbau hängt von dem zu erbringenden Dienst ab. Im einfachsten Fall kann sie z.B. leer sein. Die Konvergenzschicht wird weiter unterteilt in eine dienstspezifische Teilschicht (service specific sublayer, SS) und eine gemeinsame Teilschicht

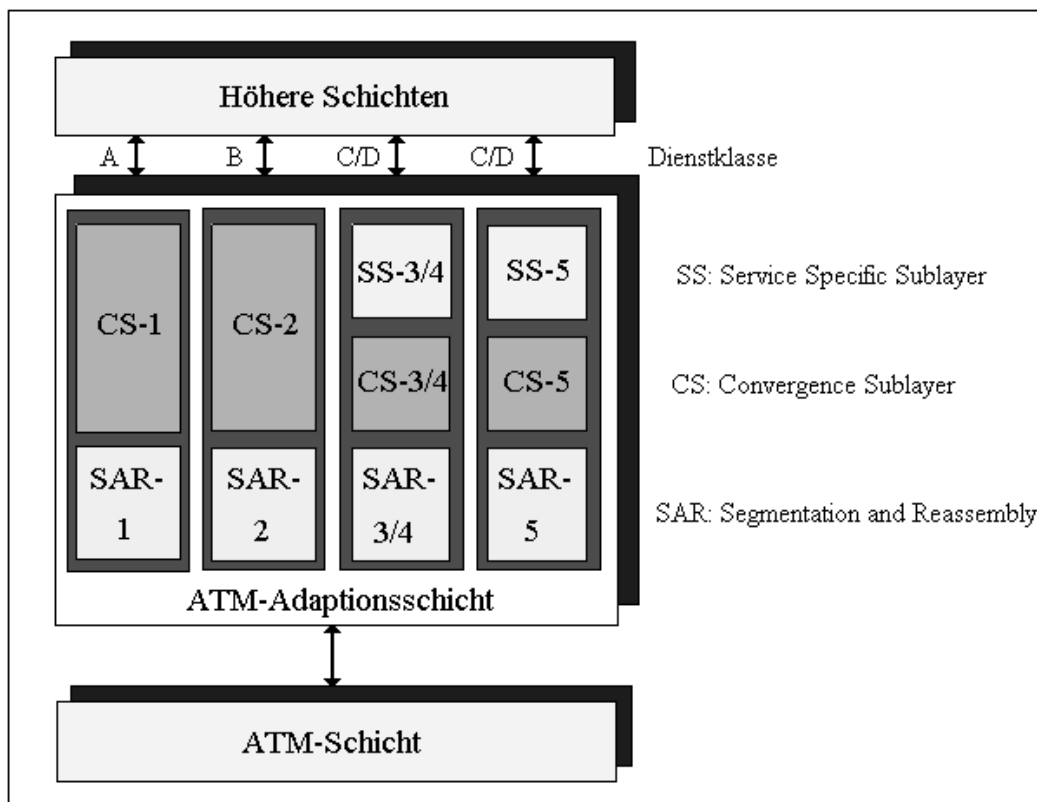


Abbildung 2.8: Aufbau der ATM-Adaptionsschicht

(common sublayer, CS). Die nachgestellten Zahlen beziehen sich auf den zugeordneten AAL-Typ.

Für die vier Dienstklassen (A-D) sind in der ATM-Adaptionsschicht unterschiedliche Ausprägungen der ATM-Adaptionsschicht definiert, die den jeweils geforderten Dienst erbringen.

- **AAL-Typ 1:**  
Dieser Dienstyp wurde für einen Dienst der Klasse A, also eine Übertragung mit fester Zeitbeziehung zwischen Sender und Empfänger und konstanter Übertragungsrate ausgelegt. Neben der Segmentierung und Reassemblierung der Benutzerdaten, steht das Ausgleichen der Verzögerungsvarianz durch den Einsatz von Puffern im Vordergrund. Durch den Einsatz einer Sequenznummer ist es möglich verlorengewandene Zellen oder Änderungen in der Reihenfolge festzustellen. Die Sequenznummer wird dazu durch eine Checksumme geschützt. Für Nutzerinformation verbleiben 47 Byte pro Zelle.
- **AAL-Typ 2:**  
Gegenüber AAL-Typ 1 ist AAL-Typ 2 für eine Datenübertragung mit variabler Bitrate ausgelegt. Die Unterstützung einer Zeitbeziehung zwischen

Sender und Empfänger ist ebenfalls enthalten. Dieser AAL-Typ ist somit für einen Dienst der Klasse B ausgelegt. Zum gegenwärtigen Zeitpunkt ist nicht klar, ob AAL-2 noch konkret spezifiziert werden wird.

- AAL-Typ 3/4:  
Der AAL-Typ 3/4 entstand durch Zusammenlegung der ursprünglich definierten AAL-3 und AAL-4. Dieser Diensttyp ist für die klassische Dateiübertragung (variable Bitrate, keine Zeitbeziehung) vorgesehen, d.h. einen Dienst der Klasse C bzw D. Die SAR-Schicht und sowie ein Teil der Konvergenzschicht sind für beide Dienstklassen identisch, lediglich die dienstspezifischen Konvergenzprotokolle unterscheiden sich. Zum Datenaustausch zwischen Dienstanutzer und AAL-3/4 gibt es einen nachrichtenorientierten und einen stromorientierten Modus. Beide Modi stellen zudem noch wahlweise eine gesicherte oder ungesicherte Übertragung zur Verfügung. Für Nutzerinformationen verbleiben aufgrund der AAL-Informationen lediglich 44 Byte pro Zelle.
- AAL-Typ 5:  
Es werden wie bei AAL-Typ 3/4 auch ein nachrichtenorientierter und stromorientierter Modus mit gesicherter oder ungesicherter Übertragung zur Verfügung gestellt. Dieser AAL-Typ ist insbesondere für die Übertragung von größeren Datenpaketen (bis 64 KB) vorgesehen. Um den Protokoll-overhead möglichst gering zu halten können hier, bis auf in der letzten Zelle jeweils 48 Byte pro Zelle für Nutzerinformationen genutzt werden.

#### 2.5.4.5 ATM im Einsatz

An dieser Stelle soll erwähnt werden, wo ATM heute Einsatz findet und welche Vorteile ATM gegenüber bisherigen Techniken bringt:

ATM wird als Grundlage in verschiedenen neueren Netzen eingesetzt. Für das Breitband-ISDN (B-ISDN) ist beispielsweise ATM als darunterliegende Netzwerk-Technologie geplant. Auch der Datendienst SMDS (Switched Megabit Data Service) verwendet eine ATM-basierte Netzstruktur.

Gegenüber der in vielen Datennetzen verwendeten Netzwerk-Technologie hat ATM den Vorteil, beliebig ausbaubar zu sein und eine gesicherte Bandbreite und andere Dienstgüte-Garantien den Nutzern bereitzustellen. Dagegen sind diese Datennetze meist in ihrer Ausbaustufe begrenzt. Nur durch Repeater und Switches kann z.B. ein Ethernet-Netz bis zu einem gewissen Bereich erweitert werden. Durch die z.B. bei Ethernet möglichen Kollisionen auf dem Medium oder ähnlichen Problemen kann keine Garantie über die dem Nutzer zur Verfügung stehende Bandbreite gemacht werden.

Große Netzbetreiber in den USA (z.B. AT&T) setzen bereits heute ATM-Technologie ein. Die Gründe dafür werden aus der nachfolgenden Auflistung ersichtlich.

In [Flan94] werden folgende Vorteile von ATM aufgeführt:

- für den Netzbetreiber:

- Kombinieren aller Dienste auf einem single switch backbone spart enorm Geld für Aufbau und Support.
  - keine manuell geschalteten Leitungen mehr
  - der Betreiber kann sich auf Upgrades, Problemlösung und Service konzentrieren, statt auf ständig neue Installationen.
  - weniger Wartungsarbeit reduziert die Kosten.
  - durch die Vergabe beliebiger Bandbreiten, kann Benutzeransprüchen besser gerecht werden.
- für den Netznutzer:
    - beliebige Bandbreite möglich (bis zum Maximalen der Verbindung)
    - höhere Verfügbarkeit und Zuverlässigkeit der Leitungen durch automatische Fehlerbeseitigungsmaßnahmen in den Backbone-Netzen.
    - geringe Kosten für Datendienste
    - vereinfachtes Netzwerkmanagement: Statt Komplexität des Verwalten eines privaten Backbones relativ einfache Verwaltung von Zugangsverbindungen.
    - Möglichkeit des Zusammenlegens von vielen Weitbereichsverbindungen in eine einzelne physikalische Leitung durch Multiplexen vieler virtueller Verbindungen.

Eine weitere Einsatzmöglichkeit von ATM, nämlich bestehenden IP-basierten Anwendungen mehr Durchsatz zu ermöglichen, ergibt sich durch die nachfolgenden Konzepte.

#### 2.5.4.6 LANE, Classical IP, MPOA

Bei dem Einsatz von ATM als lokales Netz oder der Verbindung mehrere lokaler Netzwerke durch ein ATM-Netzwerk tritt das Problem auf, daß die in den lokalen Netzwerken verwendeten Protokolle nicht für die Übertragung über ein ATM-Netz ausgelegt sind. So tritt z.B. das Problem auf, daß bei bisher üblichen LAN-Techniken eine direkte Unterstützung für Multicasts zur Verfügung steht, bei dem verbindungsorientierten ATM-Netz aber nicht. Desweiteren muß eine Adreßabbildung zwischen Adressen im LAN und ATM-Adressen durchgeführt werden. Daher wurden verschiedene Ansätze entwickelt, um keine Änderungen an den bestehenden Protokollen und Anwendungen bei Verwendung eines ATM-Netzes vornehmen zu müssen. Diese Ansätze werden nachfolgend skizziert. Ein Nachteil dieser Ansätze ist, daß bei Anwendungen, die unterschiedliche Datenströme mit unterschiedlichen Dienstgüteanforderungen erzeugen, nicht für jeden Datenstrom separate Dienstgüten festgelegt werden können.

**LANE**

LANE [LANE95, Fore96, Cisc95] steht für LAN-Emulation und verfolgt wie der Name schon sagt, das Ziel, auf einem ATM-Netz die Merkmale eines LANs (z.B. die Realisierung von Broadcast Domains) zu emulieren, um für lokale Netzwerke ausgelegte Anwendungen ohne Änderungen über ein ATM-Netz ablaufen lassen zu können. Dazu wird aufbauend auf der AAL-5 Schicht eine LAN-Emulationsschicht (LE-Schicht) eingeführt (s. Abb. 2.9). Diese bietet nach oben entweder eine IEEE 802.3 Ethernet oder eine 802.5 Token Ring Schnittstelle an, auf der Protokolle wie TCP/IP aufsetzen. Nach unten muß sie den bei ATM für eine Übertragung notwendigen Verbindungsauf- bzw. abbau einleiten. Ebenso müssen hier die in einem LAN möglichen Multicasts realisiert werden. Desweiteren muß hier die Adressabbildung von den 48-Bit-MAC-Adressen auf das 64-Bit Adressenformat E.164 umgesetzt werden.

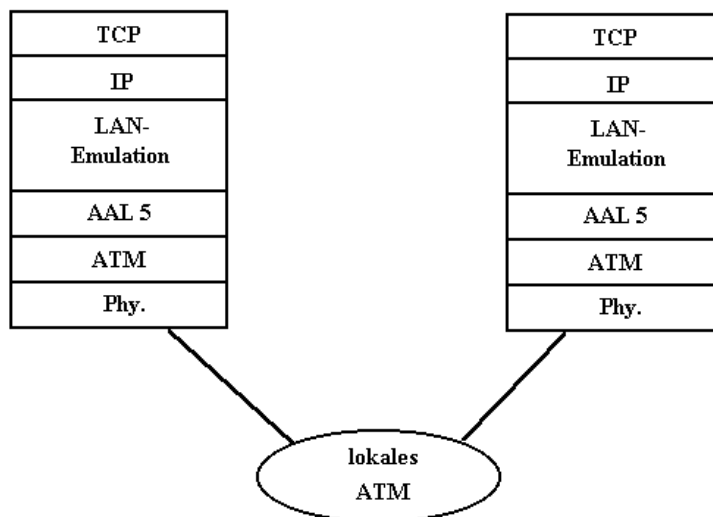


Abbildung 2.9: LAN-Emulation

Dies wird in den folgenden Komponenten realisiert:

LAN Emulation Client (LEC), LAN Emulation Server (LES) und ein Broadcast und Unknown Server (BUS). Der LAN Emulation Client befindet sich in Endsystemen und übernimmt das Weiterleiten von Daten und die Adreßauflösung. Ein LEC bietet auch eine der Standard LAN Schnittstellen für die Kommunikation mit höheren Schichten. Jeder LEC hat eine eindeutige ATM Adresse und wird mit einer oder mehreren MAC Adressen assoziiert. Ein LES übernimmt die Kontrollfunktionen zur Koordination. Bei ihm können LECs ihre ATM und MAC Adressen registrieren. Der LES wird damit die zentrale Anlaufstation für Anfragen der Adreßauflösung. Lediglich wenn die angefragte Adresse nicht bei ihm registriert wurde, muß der LES bei weiteren LECs nachfragen. Der BUS übernimmt die Verteilung der Datenpakete mit Multicastadressen.

Mittels LANE können darüberhinaus noch virtuelle LANs (VLANs) erstellt werden. Durch die Verbindung mehrerer emulierter LANs durch einen ATM

Backbone können so örtlich voneinander getrennte Arbeitsgruppen im selben virtuellen Netz arbeiten.

### **Classical IP over ATM**

Um IP über ein ATM-Netz zu betreiben, wird ein Mechanismus benötigt, der die IP Adressen auf ihre korrespondierenden ATM-Adressen umsetzt. Man stelle sich dazu zwei über ein ATM-Netz verbundene Router vor. Wenn ein Router über ein LAN Interface ein IP-Paket erhält, muß er anhand seiner next-hop Tabelle entscheiden, an welchen seiner Ausgabeports er das Paket weiterreichen muß. Ergibt dieses Nachsehen in der Tabelle, daß das Paket an ein ATM Interface weitergeleitet werden muß, muß der Router eine Adressenauflösungstabelle zu Rate ziehen, um die ATM Adresse des next-hop Routers zu erhalten. In [RFC1577] wird ein Protokoll beschrieben, das für den automatischen Aufbau der Adressenauflösungstabelle und die Adressauflösung selbst sorgt. Dazu wird dort das Konzept eines logischen IP Subnetzes (LIS) eingeführt. Ein LIS besteht aus einer Gruppe von IP Knoten, die mit einem einzelnen ATM-Netzwerk verbunden sind und zum gleichen IP Subnetz gehören. Jedes LIS besitzt einen ATMARP Server der ATMARP Anfragen entgegen nimmt und entsprechend beantwortet. Sein Wissen erhält er dadurch, daß jedes Mitglied beim Eintritt ins LIS eine Verbindung mit dem ATMARP Server aufnimmt und diesem seine IP Adresse mitteilt. Die Einträge beim ATMARP Server erlöschen automatisch, wenn die jeweilige Station nicht mehr auf eine Nachfrage (Inverse ARP) des ATMARP Servers antwortet.

### **MPOA**

MPOA steht für Multiprotocol Encapsulation over ATM und erlaubt den Einsatz verschiedener Schicht 3 Protokolle über ATM. Mit MPOA ist es möglich, virtuelle Netzwerke zu definieren. Datentransfer zwischen direkt an ATM angeschlossenen Hosts wird ohne den Umweg über einen Router durch MPOA ermöglicht, indem einmal für die Zieladresse die zugehörige ATM-Adresse ermittelt wird und so daß bei weiteren Datentransfers direkt die ATM-Adresse als Zieladresse verwendet werden kann. Gegenüber LANE kann so vor allem in großen Netzwerken eine Performancesteigerung erzielt werden. MPOA befindet sich zur Zeit noch in der Entwicklung.

### 2.5.5 Vergleich zwischen RSVP, ST2 und ATM

Nachfolgend werden die Eigenschaften der Protokolle RSVP und ST2 mit denen von ATM verglichen und die Unterschiede aufgezeigt. Da RTP in seinem Leistungsumfang stark von den anderen Protokollen abweicht und insbesondere keine Dienstgüte garantiert, wurde es nicht in den Vergleich aufgenommen. Abbildung 2.10 enthält eine Übersicht [RFC1821].

Kategorie	RSVP	ST2	ATM
Orientierung	empfängerbasiert	senderbasiert	senderbasiert
Zustände	weich (refresh/timeout)	hart (expliziter Abbau)	hart (explizites Löschen)
QoS Aufbauzeitpunkt	getrennt von Verbindungseinrichtung	gleichzeitig mit Streamaufbau	gleichzeitig mit Verbindungseinrichtung
QoS Änderung möglich ?	dynamisches QoS	dynamisches QoS	statisches QoS (fest nach Setup)
Richtung der Ressourcenbelegung	unidirektional	unidirektional	bidirektional (unicast) unidirektional(multicast)
Heterogenität	Empfänger-Heterogenität	Empfänger-Heterogenität	gleicher QoS für alle Empfänger

Abbildung 2.10: Vergleich zwischen RSVP, ST2 und ATM

Bei RSVP sind vorgenommene Reservierungen nicht permanent. Stattdessen unterliegen die Reservierungen in den Hosts und Routern einer zeitlichen Begrenzung (timeout), bis zu der sie erneuert werden müssen, wenn sie weiter aufrecht gehalten werden sollen, ansonsten werden die Ressourcen danach freigegeben. Bei ST2 und ATM bleiben die Ressourcen für die Dauer der Verbindung reserviert und müssen explizit wieder freigegeben werden.

Ein weiterer Unterschied liegt in dem Zeitpunkt, zu dem die Ressourcen reserviert werden. Bei RSVP geschieht dies getrennt von der Verbindungseinrichtung, wohingegen bei ST2 und ATM Ressourcen gleichzeitig mit dem Streamaufbau bzw. der Einrichtung der Verbindung reserviert werden. Bei ATM sind diese Reservierungen und somit der QoS nach dem Verbindungsaufbau fest (statischer QoS), wohingegen sie bei RSVP und ST2 auch während der Datenübertragung geändert werden können (dynamischer QoS).

In Bezug auf eine Kommunikation von einem Sender zu mehreren Empfängern existieren folgende weitere Unterschiede. Bei RSVP und ST2 können für jeden Empfänger unterschiedliche Dienstgüten vereinbart werden (Empfänger-Heterogenität), wohingegen bei ATM für alle Empfänger die gleiche Dienstgüte ausgehandelt wird. Beim empfängerbasierten RSVP ist dies nicht weiter verwunderlich, da jeder Empfänger selbst für die Aushandlung der Dienstgüte mit dem Sender zuständig ist. Beim ST2 Protokoll, obwohl wie ATM auch senderbasiert, ist es den Empfängern möglich, die vom Sender vorgegebene Dienstgüte-Anforderung, falls notwendig, abzuschwächen. Bei ATM ist dies nicht vorgesehen.

RSVP und ST2 bieten durch die dynamisch veränderbare Dienstgüte und einer individuellen Anpassung der Dienstgüte an die Empfänger gewisse Vorteile in Bezug auf genauere Ressourcenbelegung. Dieser Vorteil birgt aber auch neue Probleme in sich. Soll z.B. die Anzahl der Ressourcen zwischenzeitlich reduziert und später wieder erhöht werden, so könnte der Fall eintreten, daß zum Zeitpunkt der Erhöhung der Ressourcen nicht mehr genügend freie Ressourcen zur Verfügung stehen, da die ehemals zu einer Verbindung gehörenden Ressourcen mittlerweile anderen Verbindungen zugeteilt worden sind. Bei ATM ermöglichen Mechanismen, wie z.B. das statistische Multiplexen von Ressourcen, daß Ressourcen zwischen Verbindungen geteilt werden können und dabei die ausgehandelten Dienstgüten weiterhin garantiert werden können. Als nachteilig könnte sich bei RSVP und ST2 erweisen, daß es keinen Kontrollmechanismus gibt, der überprüft, ob der Sender nicht mehr als ausgehandelt wurde, sendet, da in diesem Fall eventuell vom Netzwerk selbst die niedrigere ausgehandelte Dienstgüte nicht mehr garantiert werden kann. Bei ATM wird durch den leaky bucket Algorithmus gewährleistet, daß die ausgehandelte Dienstgüte auch bei einer Überschreitung durch den Sender garantiert werden kann, gegebenenfalls werden zuviel gesendete Zellen verworfen.

Zusammenfassend läßt sich also feststellen, daß alle drei Protokolle Vor- und Nachteile besitzen. Inwieweit sich diese im Einsatz auswirken, hängt allerdings stark von den Anforderungen der Anwendung ab, die diese Protokolle nutzen will. Für die Übertragung eines kontinuierlichen Datenstroms (z.B. eines Videos) reicht statisches QoS vollkommen aus. Zudem wird der Vorteil des dynamischen QoS durch eine wesentlich aufwendigere Protokollbehandlung erkauft, da für eine Änderung der Dienstgüte Änderungen der Ressourcenbelegung auf jeder Zwischenstation zwischen Sender und Empfänger durchgeführt werden müssen. Änderungen der Dienstgüte während der Verbindung erscheinen daher nur von Vorteil, wenn sie längerfristig sind. Bei ATM könnte eine Änderung der Dienstgüte dadurch erreicht werden, daß die bestehende Verbindung abgebaut und durch eine neue Verbindung mit geänderter Dienstgüte ersetzt wird.

## 2.6 Applikation VCR

Bei VCR handelt es sich um einen verteilten, Echtzeit MPEG Video und Audio Player, der vom Department of Computer Science and Engineering des Oregon Graduate Institute of Science and Technology in Portland, Oregon entwickelt wurde [VCR]. Der Player ist dafür ausgelegt, einen Videodatenstrom und Audio-datenstrom von verschiedenen Servern entgegenzunehmen und dann synchronisiert durch einen Client abzuspielen. Unterstützte Datenströme sind MPEG-1 Video und 8-bit 8KHz  $\mu$ -law Audio. Ein Screenshot des Clients ist in Abbildung 2.11 dargestellt. Über die gewöhnlichen Player-Funktionen hinaus kann mit variabler Geschwindigkeit abgespielt und zu einer beliebigen Position gesprungen werden. Das besondere an diesem Player ist, daß er durch einen Software Feedback Mechanismus versucht, Schwankungen im Durchsatz der Datenübertragung auszugleichen und damit eine vom Benutzer vorgegebene Darstellungsqualität (in Form einer gewünschten Bildanzeigerate) sicherzustellen.

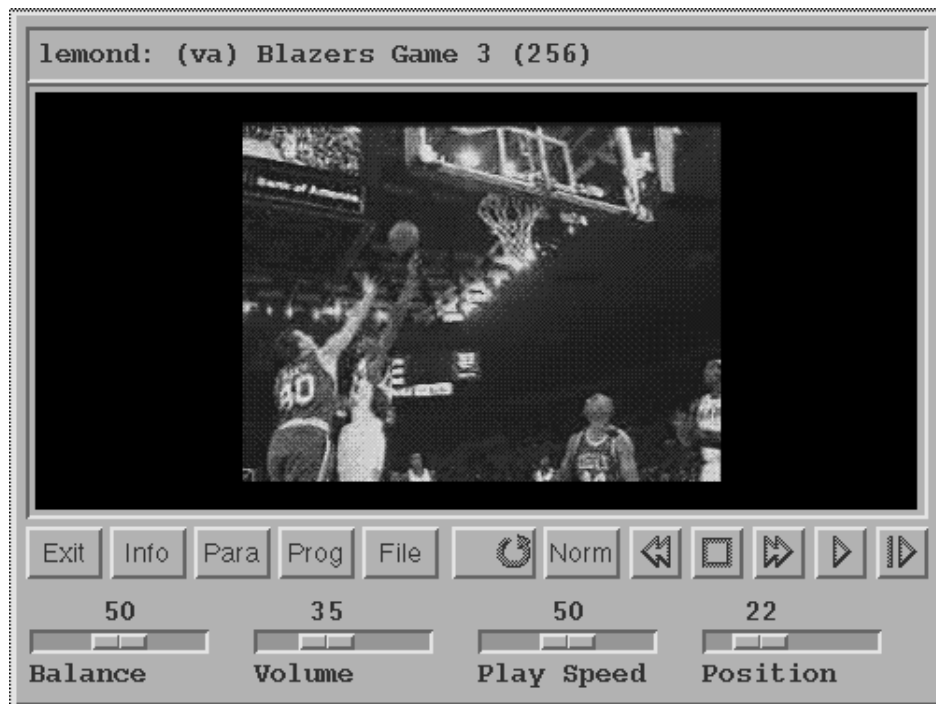


Abbildung 2.11: VCR Client

### 2.6.1 Systemarchitektur

Abbildung 2.12 zeigt die zum Player gehörende Systemarchitektur. Insgesamt gibt es fünf Komponenten: Einen Videoserver (VS), einen Audioserver (AS), einen Client und Video- und Audio-Ausgabegeräte. Der Client besteht aus einem Video Decoder und einem Controller, der die Wiedergabe beider Datenströme steuert und ein Benutzer-Interface bereitstellt. Der Client, der Videoserver und

der Audioserver können auf verschiedenen durch ein Netzwerk verbundenen Rechnern ablaufen. Dagegen müssen sich die Video- und Audio-Ausgabegeräte auf dem selben Rechner befinden wie der Client.

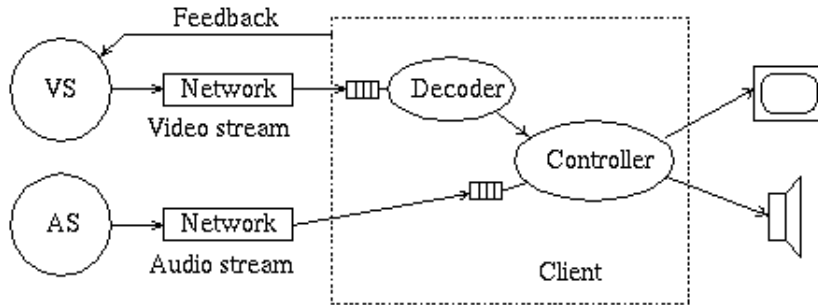


Abbildung 2.12: Systemarchitektur des Players

Während der Wiedergabe senden Video- und Audioserver die Datenströme zum Client. Dieser puffert die Datenströme, um Netzwerk Jitter auszugleichen. Darüberhinaus werden hier die Videobilder decodiert, die Audiodaten aufbereitet und an die Ausgabegeräte übergeben. Der Player spielt ein Programm (bestehend aus einem Video und Audiodatenstrom) ab, indem dessen logische Zeit (definiert durch Sequenznummern für jedes Bild und Block des Audiosamples) auf die Systemzeit des Rechners, wo der Client läuft, abgebildet wird. Dies geschieht wie folgt: Angenommen die Systemzeit, bei dem Bild( $i$ ) dargestellt wird, sei  $T_i$ , und die momentane Abspielgeschwindigkeit sei  $P$  Bilder pro Sekunde, dann muß Bild( $i + 1$ ) zur Zeit  $T_{i+1} = T_i + \frac{1}{P}$  abgespielt werden. Die Synchronisation zwischen den beiden Datenströmen wird beim Client erreicht, indem er Audioblöcke und Videobilder mit den selben Sequenznummern zur gleichen Zeit abspielt.

### 2.6.2 Software Feedback Mechanismus für Client/Server Synchronisation

Mit Hilfe des Software Feedback Mechanismus soll der Player die Fähigkeit erhalten, sich an dynamisch ändernde Umgebungen anpassen zu können, um eine möglichst konstante Darstellungsqualität zu gewährleisten. Der Mechanismus bezieht sich nur auf die Videoübertragung, da davon ausgegangen wird, daß für die Audioübertragung immer ausreichend Bandbreite vorhanden ist.

Der Mechanismus funktioniert wie folgt (s. Abb. 2.13): Beim Client wird die momentane Zeit<sup>1</sup>  $T_c$  gemessen und die Zeit des Servers  $T_s$ , wie sie der Client sieht. Aus der Differenz der beiden Zeiten wird die 'raw server work ahead' Zeit  $T_{rswa} = T_s - T_c$  bestimmt. Die 'server work ahead' Zeit  $T_{swa}$  (also die Zeit, die

<sup>1</sup>alle nachfolgenden Zeiten sind absolute Zeiten

der Server, der Darstellung voraus, Pakete versendet) ergibt sich durch Eliminieren des Hochfrequenzjitteranteils, indem  $T_{rswa}$  durch einen low-pass Filter (F1) geschickt wird. Der Kontrollalgorithmus vergleicht dann  $T_{swa}$  mit der 'target server work ahead' Zeit  $T_{tswa}$  (also der Vorgabe<sup>2</sup> für die server work ahead Zeit) und reagiert entsprechend.

Der momentane Level des Netzwerkjitters  $J_{net}$  ergibt sich durch  $|T_{rswa} - T_{swa}|$ , nachdem letzteres über einen low-pass Filter (F2) geschickt wurde. Anhand von  $J_{net}$  wird dann der neue Wert für die Vorgabe von  $T_{tswa}$  berechnet. Ist die Verzögerungsvarianz durch das Netzwerk groß, muß die Zeit, die der Server im voraus arbeiten soll, entsprechend groß ausfallen, damit die Puffer ausreichend gefüllt bleiben, denn eine große Verzögerungsvarianz impliziert auch größere Verzögerungen. Andererseits muß darauf geachtet werden, daß die Zeit nicht zu groß wird, damit der Puffer nicht überläuft. Somit wird sicher gestellt, daß der Puffer für die eingehenden Videopakete weder leer- noch überläuft.

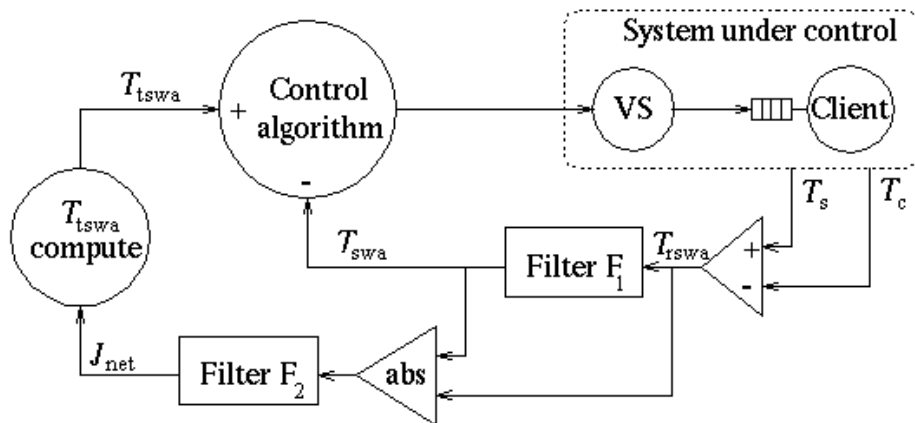


Abbildung 2.13: Aufbau des Synchronisation-Feedback-Mechanismus

### 2.6.3 Software Feedback Mechanismus für die QoS Kontrolle

Da die vom Benutzer vorgegebene Darstellungsqualität<sup>3</sup> höher sein kann als die effektiv zur Verfügung stehende Bandbreite des Videokanals es erlaubt, kann es in diesem Fall zu willkürlichen Bildverlusten kommen. Daher soll anhand der beim Client festgestellten Dienstgüte beim Videoserver (also schon bei der Quelle) Bilder verworfen und somit erst gar nicht übertragen werden.

Der Software Feedback Mechanismus für die QoS Kontrolle ist in Abbildung 2.14 dargestellt. Zu Beginn der Videoübertragung wird die 'target frame rate'  $F_t$ , mit der der Videoserver die Bilder sendet, auf die Benutzervorgabe  $F_u$  gesetzt. Die

<sup>2</sup>im Programm läßt sich dieser Wert als relativer Wert einstellen

<sup>3</sup>zur Zeit nur bestimmt durch die Bilddarstellungsrate

Bilddarstellungsrates wird beim Client gemessen und um Schwankungen auszugleichen über einen low-pass Filter geschickt. Diese gefilterte Bilddarstellungsrates  $F_d$  wird durch den Kontrollalgorithmus mit  $F_u$  und dem momentanen Wert von  $F_t$  verglichen. Anhand dieser Information wird ein neuer Wert für  $F_t$  berechnet und dem Videoserver übermittelt.

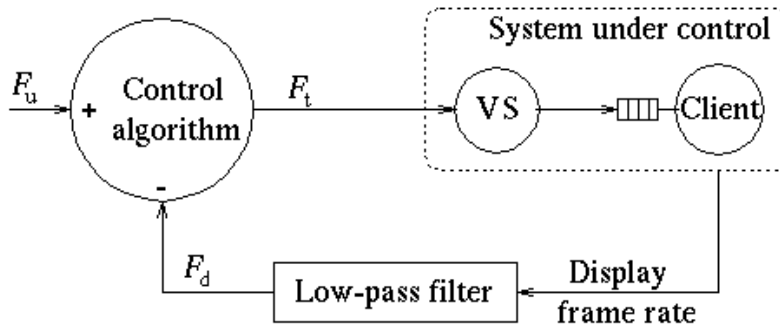


Abbildung 2.14: Aufbau des QoS-Kontroll-Feedback-Mechanismus

### 2.6.4 Implementierungsaspekte

Diese Applikation dient hier als Beispiel für eine Multimedia-Applikation. Interessant wird sie durch die beiden Audio- und Video-Datenströme mit deren unterschiedlichen Dienstgütereigenschaften.

Für die Adaption dieser Anwendung auf ATM sind nachfolgende Informationen von Interesse:

Der Player ist in der Programmiersprache C geschrieben, dessen Quellcode frei verfügbar ist. Abbildung 2.15 zeigt den internen Aufbau des Players in Hinblick auf Kommunikation. Der Client setzt sich aus mehreren Prozessen zusammen, die miteinander über geteilten Speicher, Semaphoren, Pipes und Signalen kommunizieren.

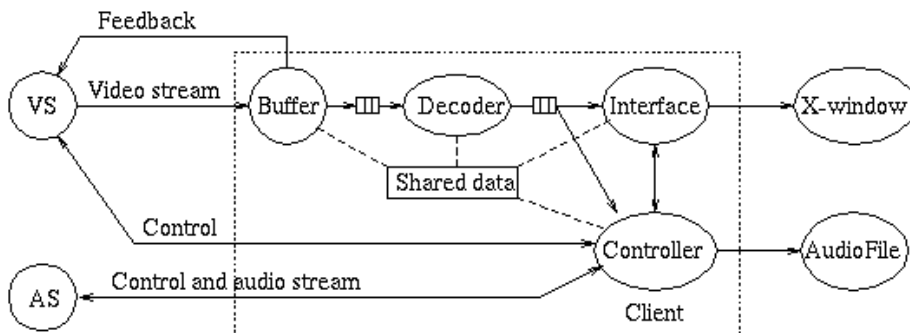


Abbildung 2.15: interner Aufbau des Players

Der Client sendet Kontrollnachrichten an den Audioserver (AS) und empfängt von ihm die Audiodaten. Der Videosever(VS) sendet die Videobilder zum Client, nachdem diese in kleinere Pakete entsprechend den Anforderungen des Übertragungsprotokolls zerlegt wurden. Der Buffer-Prozeß des Client setzt diese dann wieder zusammen, bevor er sie an den Decoder-Prozeß weiterreicht. Die Darstellung und das Benutzerinterface werden durch weitere Prozesse realisiert.

## Kapitel 3

# Adaption einer IP-basierten Applikation

### 3.1 Gründe für die Adaption

Bei ATM können beim Verbindungsaufbau Dienstgüteparameter ausgehandelt werden, die dann für die Dauer der Verbindung garantiert werden. Nun stellt sich die Frage, welche Möglichkeiten es gibt, die vorhandenen IP-basierten Anwendungen auf ATM ablaufen zu lassen.

Die in Kapitel 2.5.4.6 vorgestellten Konzepte ermöglichen zwar vorhandene IP-basierte Applikationen über ATM ablaufen zu lassen, ohne Änderungen an ihnen vornehmen zu müssen, haben aber für die Übertragung von Multimediadaten zwei Nachteile: Zum einen sind sie durch den Protokolloverhead ineffizient und zum anderen, am entscheidendsten, kann keine separate Dienstgüte für unterschiedliche Datenströme genutzt werden. Gerade dies ist aber für die Unterstützung von Multimedia-Anwendungen notwendig, da die dort vorkommenden Datenströme stark abweichende Dienstgüteanforderungen stellen.

Um den Anforderungen bestehender Multimedia-Anwendungen von Netzwerkebene her möglichst gut nachkommen zu können, müssen daher Änderungen an diesen Anwendungen vorgenommen werden:

- Datenströme sollten sauber voneinander getrennt werden, indem pro Datenstrom eine eigene Verbindung benutzt wird.
- für jede Verbindung muß dann den Anforderungen der übertragenen Daten entsprechend eine bestimmte Dienstgüte ausgehandelt werden.
- bei Nutzung von ATM sollte die Kommunikation auf direkte ATM-Kommunikation (Native ATM) umgestellt werden. Dies hat den Vorteil, daß der Protokolloverhead minimiert wird. Dazu ist eine Umstellung auf ATM-Adressen und ATM-Übertragungsfunktionen notwendig.

## 3.2 Probleme bei der Adaption

Unabhängig vom verwendeten Dienstgüte garantierenden Protokoll tritt bei der Adaption ein zentrales Problem auf, nämlich welche Dienstgüte benötigt die zu adaptierende Applikation und wie müssen hierfür die QoS-Parameter gewählt werden. Diese Problematik soll im folgenden allgemein und anhand der Beispielanwendung VCR (s. Kapitel 2.6) erläutert werden.

Zunächst wird auf die benötigte Bandbreite eingegangen. Das zu übertragene Datenvolumen einer Multimedia-Applikation hängt stark von der gewählten Kodierung und dem verwendeten Kompressionsverfahren ab.

Um analoge Audiodaten übertragen zu können, müssen diese zuvor in eine digitale Form überführt werden. Dies geschieht normalerweise durch Abtasten des Signals zu regelmäßigen Zeitpunkten und anschließender Quantisierung. Dies wird auch als Puls Code Modulation bezeichnet. Die Qualität der Kodierung hängt dabei von der Abtastfrequenz und der Anzahl der Quantisierungsstufen ab. Für Sprachübertragung (z.B. beim digitalen Telefon) hat sich eine Abtastrate von 8000 Hz und eine Quantisierung mittels 8 bit als ausreichend herausgestellt. Für die Übertragung ist somit eine Datenrate von 64 kbit/s notwendig. Diese steigt stark an, wenn man in CD-Qualität übertragen will. Dazu sind eine Abtastrate von 44,1 KHz und eine Quantisierung mit 16 bit notwendig. Dies führt bei Stereosignalen zu einer Datenrate von 1,4 Mbit/s. Um die benötigte Datenrate zu reduzieren wurden verschiedene Kompressionsverfahren (DPCM,ADPCM,LP) entwickelt, auf die hier aber nicht näher eingegangen werden soll.

Die Übertragung von Videodatenströmen stellt noch größere Anforderungen. Soll das Video mit Filmqualität (bildschirmfüllend (640x480) und 25 Bilder pro Sekunde, 3 Byte Farbinformation je Pixel) übertragen werden, so würde man eine Datenrate von 22 Mbit/s benötigen. Die benötigte Datenrate läßt sich durch Kompressionsverfahren (z.B. MPEG) drastisch (auf z.B. 1,5 Mbit/s) reduzieren.

Je nach verwendetem Kompressionsverfahren ergibt sich ein Datenstrom mit konstanter oder mit variabler Bitrate. Je nach Anwendung kann auch die Verzögerung bis zur Anzeige beim Empfänger eine Rolle spielen. Wichtiger ist aber, daß es nicht zu Ruckeln der Bilder aufgrund von zu starken Variationen in der Auslieferungsverzögerung (jitter) kommt.

Es zeigt sich, daß die Wahl der Kodierung (bzw. Komprimierungsverfahren) starken Einfluß auf die benötigte Dienstgüte hat. Daher benötigt man genauere Kenntnisse über die in der Anwendung verwendeten Kodierungen, will man die benötigte Dienstgüte möglichst genau angeben. Dies ist besonders wichtig, wenn sich die Kosten für die Übertragung nach der zur Verfügung gestellten Qualität richten. Ansonsten könnte man einfach sehr großzügige Angaben machen, dies erscheint aber in Hinblick auf gute Netzauslastung nicht sinnvoll.

Bei der Applikation VCR sind die gewählten Kodierungs- und Kompressionsverfahren bekannt. VCR unterstützt Audiosamples, die mit 8 Bit und 8 KHz

gesampelt wurden. Daher wird für die Audioübertragung eine Bandbreite von 64 KBit pro Sekunde benötigt. Aufgrund der konstanten Datenrate kommt nur CBR (Constant Bit Rate) als zu verwendende Dienstklasse in Frage.

Die Anforderung der Videoübertragung kann dagegen nicht so leicht bestimmt werden. Dies liegt daran, daß bei dem verwendeten Komprimierungsverfahren MPEG-1 [ISO11172] die Bildgröße variabel bis zu einer Maximalgröße von 720x576 Pixel/Bild bei 30 Bildern/Sekunde ist. Die benötigte Bandbreite für die Videoübertragung ist daher abhängig von der im Video verwendeten Bildgröße. Die Applikation VCR könnte so erweitert werden, daß der Server vor dem Versenden einer MPEG-Datei, deren Bildgröße und Framerate ermittelt und dann die Anforderung entsprechend formuliert. Hier besteht allerdings das Problem, daß keine Erfahrungswerte über den genauen Zusammenhang zwischen Bildgröße und Framerate und der dafür benötigten Bandbreite bestehen. Nur für die Maximalgröße ist dies bisher bekannt. Um die Anforderung daher unabhängig von der im Video verwendeten Bildgröße zu formulieren, muß die Bandbreite für die maximale Bildgröße gewählt werden. Bei diesem Wert kommt es zu einer Datenrate von ca. 1,5 Mbit pro Sekunde. Die Festlegung der übrigen QoS-Parameter geschieht in Anlehnung an [ATM94-0640]. Als zu verwendende Dienstklasse für MPEG-1 wird dort CBR mit einer peak und sustainable bit-rate (s. Kapitel 2.5.4.1) von 1,5 Mbit pro Sekunde empfohlen. Für die maximale Verzögerung wird dort ein Wert von 1 Sekunde angegeben. Das Angeben eines Wertes für Verzögerungsvariation zwischen Zellen entfällt bei einem CBR Dienst, da diese beim entsprechenden ATM Adaption Layer durch den Einsatz von Puffern beseitigt wird.

Für andere im Multimediabereich verwendete Komprimierungsverfahren (z.B. MPEG-2, H.261, H.262) sind in [ATM94-0640] ebenfalls QoS Parameter angegeben.

Es stellt sich die Frage, ob es nicht bessere Möglichkeiten gibt, Multimedia-Anwendungen von Netzwerkeite aus zu unterstützen, indem z.B. durch Anbietung von Diensten durch das Netzwerk. So könnte der Anwendungsbenutzer bzw. Programmierer weitestgehend von der Festlegung von der Vielzahl der QoS Parameter befreit werden. Diese Fragestellung soll in Kapitel 4 untersucht werden.

### 3.3 Durchführung der Adaption

Zu Beginn der Adaption der Anwendung VCR mußte untersucht werden, inwieweit bereits eine klare Trennung der einzelnen Datenströme vorliegt. Bei VCR gibt es folgende Datenströme:

- Datenstrom für die Übertragung der Videodaten
- Datenstrom für die Übertragung der Audiodaten
- Datenstrom für die Steuerung der Videoübertragung
- Datenstrom für die Steuerung der Audioübertragung

- Datenströme für die Kommunikation mit dem Darstellungsinterface
- Dateizugriffe

Davon müssen nur die ersten vier von UDP bzw. TCP auf ATM umgesetzt werden. Die Datenströme für die Kommunikation mit dem Darstellungsinterface benutzen kein UDP oder TCP sondern UNIX-Sockets, da diese Kommunikation lokal auf dem gleichen Rechner bleibt. Diese brauchen daher nicht auf ATM umgesetzt werden.

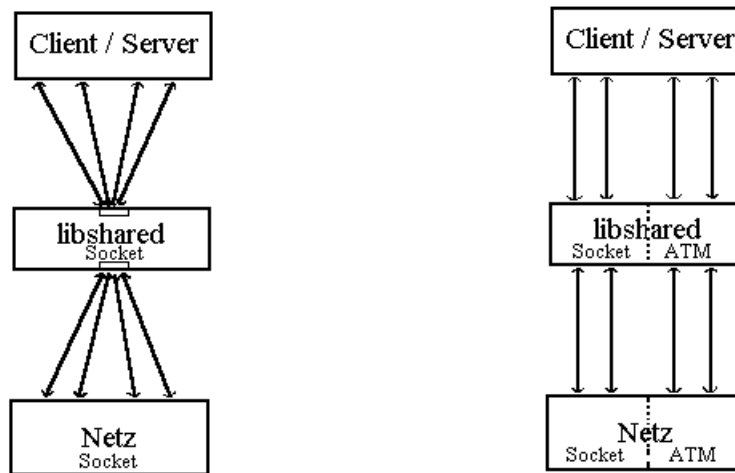


Abbildung 3.1: Datenströme vor (links) und nach (rechts) der Umstellung

VCR besitzt für die Übertragung von Daten eine spezielle Bibliothek (`libshared`). Als Problem stellte sich dabei heraus, daß bei VCR einige Funktionen dieser Bibliothek von mehreren verschiedenen Datenströmen verwendet werden, insbesondere von Datenströmen, die auf ATM umgesetzt, als auch von Datenströmen, die nicht umgesetzt werden sollen. Diese Gleichbehandlung der Datenströme ermöglichte die Nutzung der selben Funktionen der Bibliothek für verschiedene Datenströme. Dies ist aber jetzt wegen der Umstellung auf ATM nicht mehr möglich (s. Abb. 3.1). Für diese Funktionen wurde dann jeweils die vorliegende Variante unverändert übernommen (da diese ja noch für die nicht auf ATM umzusetzenden Datenströme benötigt werden) und zusätzlich eine Variante erstellt, die ATM-Funktionen der API der Firma FORE benutzt.

Für diesen Zweck mußte zunächst für jeden Aufruf einer Funktion aus dieser Bibliothek festgestellt werden, für welche Datenströme sie verwendet wird und gegebenenfalls der Aufruf durch den Aufruf der ATM-Variante dieser Funktion ersetzt werden. Dazu mußte zurückverfolgt werden, an welcher Stelle und mit welchen Parametern die jeweilige Funktion der Bibliothek aufgerufen wurde. Dies erwies sich als recht aufwendig, da teilweise erst bei zwei Aufrufshierarchien höher definiert wurde, für welchen Datenstrom der Parameter für den Filedescriptor stand.

Weitere Probleme beim Anpassen dieser Bibliothek gab es bei der Abbildung der Socketbefehle auf die Funktionen der API der Firma FORE. Die Funktionen der API der Firma FORE haben zumeist den gleichen Namen (mit vorangestelltem `atm_`) wie die Socketvariante. Die Semantik der Funktionen der API der Firma FORE und der Socketbefehle unterscheidet sich aber zum Teil erheblich. Zusätzlich werden zum Teil andere bzw. zusätzliche Parameter verwendet. So kommen z.B. bei `atm_bind()` und `atm_connect()` Parameter für den QoS hinzu. Erschwerend kommt bei der Abbildung hinzu, daß die Dokumentation der API der Firma FORE (man-pages) zum Teil etwas ungenau formuliert ist, so daß die genaue Semantik erst durch Ausprobieren klar wird.

Wegen der zum Teil gravierenden Unterschiede kann nachfolgende Tabelle daher nur als grobe Übersicht dienen:

Socket-Befehl	API-Befehl der Firma FORE
<code>socket()</code>	<code>atm_open()</code>
<code>bind()</code>	<code>atm_bind()</code>
<code>connect()</code>	<code>atm_connect()</code>
<code>listen()</code>	<code>atm_bind()</code> <sup>1</sup>
<code>accept()</code>	<code>atm_listen</code> <sup>2</sup> , <code>atm_accept()</code>
<code>select()</code> <sup>3</sup>	<code>atm_listen()</code> <sup>4</sup>
<code>read()</code>	<code>atm_recv()</code>
<code>write()</code>	<code>atm_send()</code>
<code>close()</code>	<code>atm_close()</code>

Erhebliche Unterschiede gibt bei den Funktionen `listen()` und `atm_listen()`. Während mit `listen()` die maximale Länge der Warteschlange für eingehende Verbindungsaufbauwünsche festgelegt und die Bereitschaft für die Entgegennahme derselben bekundet wird, wird mit `atm_listen()` zeitlich nicht begrenzt auf das Eintreffen eines Verbindungsaufbauwunsches gewartet. Das Festlegen der Warteschlangenlänge muß mit `atm_listen()` (Parameter `qlen`) vorgenommen werden. Warten auf einen Verbindungsaufbauwunsch ähnlich `atm_listen()` geschieht dagegen mit `accept()`.

Bei VCR wird die Funktion `select()` verwendet, um zu testen, ob Nachrichten eingegangen sind, bevor diese weiterverarbeitet werden. Dies ist an zwei Stellen wichtig: Zum einem, wenn auf einen Verbindungsaufbauwunsch gewartet werden soll, zum anderen, wenn auf eingehende Daten gewartet wird. Im ersten Fall wird anschließend ein `accept()` aufgerufen, welches durch `atm_listen()` zu ersetzen ist und im zweiten Fall folgt auf das `select()` ein `read()`, welches durch

<sup>1</sup>Parameter `qlen` bei `atm_bind()` gibt die maximale Länge der Warteschlange an und entspricht damit Parameter `backlog` bei `listen()`

<sup>2</sup>Das Extrahieren des ersten Verbindungsaufbauwunsches aus der Warteschlange analog zu `accept()` geschieht mit `atm_listen()`.

<sup>3</sup>`select()` ist kein Teil der Socket-API, es ermöglicht, an mehreren Sockets gleichzeitig zu horchen, ob etwas angekommen ist.

<sup>4</sup>Im Gegensatz zu `select()` kann `atm_listen()` nur an einer Verbindung horchen. Außerdem kann bei `atm_listen()` nicht wie bei `select()` die Wartedauer begrenzt werden.

durch `atm_recv()` zu ersetzen ist. Der Aufruf von `select()` verhindert in beiden Fällen, daß der darauffolgende Befehl aufgerufen wird, wenn gar keine Daten eingegangen sind, da in diesem Fall der darauffolgende Befehl blockieren würde. Bei `select()` läßt sich die Wartedauer durch einen Parameter zeitlich begrenzen. Nach Ablauf dieser Zeit werden in VCR die Funktionen, in denen `select()` aufgerufen werden, verlassen und erst später erneut aufgerufen. So kann zwischendurch auch an anderen Ports ein Test auf eingehende Daten durchgeführt werden.

Dadurch ist es möglich auf Video- und Audiodaten zu warten und in Abhängigkeit davon, welche jeweils zuerst eintreffen, entsprechend entgegenzunehmen. Die rechtzeitige Entgegennahme der zeitkritischen Audio- und Videodaten ist wichtig.

Ohne diese Funktion hätte man möglicherweise folgendes Problem: Wenn blockierend auf eingehende Daten an einem Port gewartet wird, können in der Zwischenzeit auf anderen Ports eingehende Daten nicht bearbeitet werden. Und wenn diese Daten dann schließlich entgegengenommen werden können, (weil mittlerweile auf dem anderen Port die Blockade durch eingegangene Daten aufgehoben wurde,) könnte deren Bearbeitung bereits zu spät sein, weil deren Zeitbezug nicht mehr stimmen könnte.

Tests ergaben, daß `select()` nicht vollständig mit den Filedescriptoren der API der Firma FORE zusammenarbeitet. Es wird nicht erkannt, daß auf einem ATM-Port ein Verbindungsaufbauwunsch, ausgelöst mit `atm_connect()`, eingegangen ist. Dagegen erkennt `select()` an einem Port eingehende Daten, die mit `atm_send()` an diesen Port gesendet wurden. Somit bereitet der zweite Fall (s.o.) kein Problem, der erste dagegen schon, da es in der API der Firma FORE keine vergleichbare Funktion gibt.

Der Aufruf von `atm_listen()` blockiert so auf jeden Fall solange, bis ein Verbindungsaufbauwunsch eingeht. Die Funktion `ComGetConnPair` der VCR-Software (Verzeichnis `shared/com.c`) in der `atm_listen()` aufgerufen wird, wird aber ständig neu aufgerufen, damit weitere Clients vom Server mit Daten versorgt werden können.

Wenn nun ein vom Server abgespalteter Prozeß einen Client bedient und der ursprüngliche beim erneuten Aufruf von `atm_listen()` blockiert, kommt es zu der Fehlermeldung *'interrupted system call'*. Daher mußte diese Funktionalität beim Server entfernt werden. Er kann damit nur noch einen Client bedienen.

Ein weiteres Problem mußte beim auf `atm_listen()` folgenden Aufruf von `atm_accept()` gelöst werden: Vor dem Akzeptieren und Herstellen einer Verbindung mit dem Kommunikationspartner, der den Verbindungsaufbauwunsch geäußert hat, muß ein neuer Port eingerichtet werden, über den dann die Kommunikation durchgeführt wird. Dies ist bei der Verwendung von BSD Unix-Sockets nicht unbedingt notwendig, wenn man wie bei VCR getan, die Option `SO_REUSEADDR` verwendet.

Das eben Beschriebene sah im Quelltext vorher so aus:

```
// Auszug aus der Funktion ComGetConnPair des VCR-Programms
// (shared/com.c)
...

FD_ZERO(&read_mask);
if (fd_inet >= 0) FD_SET(fd_inet, &read_mask);

tval.tv_sec = 1; /* wait at most 1 second */
tval.tv_usec = 0;

nfds = fd_inet;
nfds ++;
errno = 0;

// auf Verbindungsaufbauwunsch 1 Sekunde lang warten
if ((fd = select(nfds, &read_mask, NULL, NULL, &tval)) == -1)
{
    if (errno == EINTR) return -1;

    fprintf(stderr,
        "Error ComGetConnPair: pid %d failed on select():",
        getpid());
    perror("");
    return -1;
}
// nach Ablauf von 1 Sekunde diese Funktion verlassen, falls
// bis dahin noch kein Verbindungsaufbauwunsch eingegangen ist
if (fd == 0) return -1;

fd = -1;

// Verbindung herstellen
if (fd == -1 && fd_inet >= 0 && FD_ISSET(fd_inet, &read_mask)) {
    addrlen = sizeof(struct sockaddr_in);
    fd = accept(fd_inet, (struct sockaddr *)&peeraddr_in, &addrlen);
    if (fd == -1) {
        fprintf(stderr,
            "Error ComGetConnPair: pid %d failed to accpet on fd_inet:",
            getpid());
        perror("");
    }
}
...
```

Nach der Umstellung folgendermaßen:

```
// Auszug aus der Funktion ComGetConnPair des VCR-Programms:
// (shared/com.c)
...
// Auf Verbindungsaufbauwunsch warten
if (atm_listen(fd_inet,&conn_id,&calling,&qos,&aal)==-1)
{
    fprintf(stderr,
        "Error ComGetConnPair: pid %d failed on atm_listen():",
        getpid());
    perror("");
    return -1;
}

// *** neuen SAP fd_new erzeugen fuer atm_accept ***

fd_new=atm_open(atm_device_name,O_RDWR,&connection_info);

ssap=0; // vergibt automatisch neuen SAP
qlen=2;
if (atm_bind(fd_new,ssap,&ssap,qlen)<0)
{
    fprintf(stderr,"ComGetConnPair: atm_bind fehlgeschlagen\n");
    atm_error("");
    exit(1);
}
// *****

// Verbindung mit conn_id ueber fd_new herstellen
res = atm_accept(fd_inet,fd_new,conn_id,&qos,atm_dataflow);
if (res == -1) {
    fprintf(stderr,
        "Error ComGetConnPair: pid %d failed to accept on fd_inet:",
        getpid());
    perror("");
}
...

```

Noch nicht endgültig geklärt ist, inwieweit Abfragen auf bestimmte Fehlernummern nach Socketbefehlen, die durch API Funktionen der Firma FORE ersetzt wurden, durch neu hinzu gekommenen Fehlernummern der API ersetzt werden müssen (zum Teil liefern die API Funktionen neben zusätzlichen auch dieselben Fehlernummern der Socket-API zurück).

Ein weiteres Problem ergab sich dadurch, daß die maximale Paketlänge (MTU) bei der API der Firma FORE nur 9184 Bytes beträgt, wohingegen bei TCP 64 KByte. Es wurde daher notwendig, die Daten vor dem Übertragen und somit dem Übergeben der Funktion `atm_send()` auf kleinere Längen zu bringen, indem Pakete in mehrere Teilpakete aufgeteilt wurden. Diese Funktionalität war schon in VCR enthalten (Funktion `send_to_network`), es mußte lediglich noch der Wert der maximalen Paketlänge angepaßt werden.

Es zeigte sich, daß sich das Adaptieren der Anwendung VCR auf ATM als recht aufwendig und kompliziert erwies. Zudem mußten aufgrund der eingeschränkten Funktionalität der API der Firma FORE bei der auf ATM basierten Version von VCR Einschränkungen (s.o.) vorgenommen werden.

## Kapitel 4

# Netzwerkdienste

Zu Beginn dieses Kapitels soll anhand einer Beschreibung der heute üblichen Netzwerke erläutert werden, warum die Bereitstellung von Diensten von Netzwerkseite aus wünschenswert ist.

Anwendungen bekommen momentan von ATM Netzwerkseite her nur eine sehr rudimentäre Schnittstelle für den Transport von Rohdaten angeboten. Dabei muß eine Anwendung eine Vielzahl von Parametern beim Verbindungsaufbau übergeben, insbesondere wenn sie eine bestimmte Dienstgüte vom Netz benötigt. Der Zusammenhang zwischen der von der Anwendung benötigten Dienstgüte und den daraus resultierenden Parametern ist nicht unbedingt immer klar ist.

In Bezug auf die Daten selbst ergibt sich folgendes Bild: Die Aufbereitung der Daten, die Kodierung und Kompression unterliegt allein der Verantwortung der Anwendung, von Netzwerkseite gibt es dafür keinerlei Unterstützung. Dies hat zwar den Vorteil, daß von Netzwerkseite her keine Einschränkungen bezüglich der zu übertragenden Daten existieren, bedeutet aber auch, daß erheblicher Aufwand in jede Anwendung allein für die Aufbereitung der Rohdaten gesteckt werden muß, wenn die Übertragung der Daten effizient und kostengünstig geschehen soll.

Insgesamt gesehen erscheint die Realisierung einer netzbasierten Anwendung damit als enorm aufwendig. Es stellt sich daher die Frage, wie die Anwendung von dieser Last und von der Notwendigkeit der Kenntnis von Netzwerkinternas befreit werden kann. Durch die Anbietung verschiedener häufig von Anwendungen benötigter Dienste, könnte sowohl der Zugang zum Netzwerk erleichtert werden, als auch der Anwendung ein Großteil der Arbeit für die Aufbereitung der Daten abgenommen werden.

Dieser Ansatz soll daher im nachfolgend weiter verfolgt werden. Insbesondere wird dabei versucht werden, eine Vielzahl offener Fragestellungen zu beantworten. Einige davon sind z.B. was sollen die Dienste leisten, welche unterschiedlichen Dienste sollen angeboten werden, welche Einflußmöglichkeiten sollen auf den Dienst ermöglicht werden, wie soll die Schnittstelle zum Dienst aussehen und weitere Fragen die Realisierung der Dienste betreffend.

## 4.1 Leistungsumfang der Dienste

Bevor genauer auf einen einzelnen Dienst im speziellen eingegangen wird, soll an dieser Stelle erst einmal auf grundsätzliche Leistungsmerkmale der Dienste und die sich daraus ergebenden Vorteile eingegangen werden.

Die jeweiligen Dienste sollten die Anwendung von Fragen der Kodierung, Kompression, Ressourcenreservierung und Netzwerkinternas entlasten, indem sie diese Aufgaben selbst übernehmen. Für die Anwendung sollte ein einfacher Zugang zum Dienst realisiert werden, indem vorgefertigte Funktionen zur Benutzung eines Dienstes bereitgestellt werden. Durch diese Funktionen und durch die Tatsache, daß die Anwendung jetzt nur noch die zu übertragenden Daten in der vorliegenden statt in komprimierter Form bereitstellen muß, wird eine leichte Erweiterbarkeit der Anwendungen um neue von den Diensten bereitgestellten Funktionen erreicht.

Die Komprimierung der Daten innerhalb des Dienstes bietet weitere Vorteile. Dadurch, daß die Anwendung nur über die vorgefertigten Funktionen auf den Dienst zugreifen kann, kann die interne Realisierung der Dienste, insbesondere die verwendeten Komprimierungsverfahren, verändert werden, wenn die Dienstschnittstelle unverändert bleibt. So kann flexibel auf Verbesserungen in der Komprimierungstechnik eingegangen werden, indem man die bisher verwendeten Verfahren durch die verbesserten ersetzt. Die Anwendung kommt so automatisch in den Genuß einer besseren Leistung und ohne an ihr Veränderungen vorgenommen zu haben.

Der Einsatz von Diensten bietet einen weiteren Vorteil: Durch die vom Dienst vorgegebene Schnittstelle, werden verschiedene Anwendungen potentiell zueinander kompatibel, wenn sie auf dem selben Dienst aufsetzen, denn die Anwendungen übergeben dem Dienst und bekommen vom Dienst nur unkodierte Rohdaten.

## 4.2 Dienstangebot

Um die verschiedensten Anwendungen unterstützen zu können, ist eine Vielzahl an unterschiedlichen Diensten denkbar:

- Datagramme
- Datenströme
- Dateiübertragung
- Videoübertragung (Datenbankabruf, Konferenz)
- Audioübertragung (Datenbankabruf, Konferenz)
- Email, News
- Fax

Die Aufgaben und Funktionen dieser Dienste wird nachfolgend erläutert. Neben unmittelbar einsichtigen Funktionen werden unter anderem auch Funktionen beschrieben, die zur Zeit noch nicht unbedingt durch entsprechende Anwendungen realisiert werden.

### 4.2.1 Datagramme, Datenströme, Dateiübertragung

Zu einem sollten die 'klassischen' Netzwerkdienste, wie die Übertragung von Datagrammen und Datenströmen, angeboten werden, damit auch weiterhin die Möglichkeit besteht, über die vom Netz angebotenen Dienste hinaus in den Anwendungen eigenständige Dienste zu realisieren. Dabei sollte aber eine bessere Unterstützung der Anwendung durch die Dienste angeboten werden, als dies durch bisherige Programmier-Bibliotheken getan wird.

#### 4.2.1.1 Datagramme

- `send / receive`: versenden und empfangen von Datagrammen.

Als Erweiterung wären hier eine Unterstützung einer automatischen Empfangsbestätigung sinnvoll, als auch eine garantierte Auslieferungszeit.

#### 4.2.1.2 Datenströme

- `open`: Datenstrom einrichten
- `write / read`: Daten schreiben / lesen
- `close`: Datenstrom abbauen

Hier sollte der Dienst die Anwendung durch beim `open` angeforderte Echtzeit-Garantien und garantierten Durchsatz unterstützen. Dadurch wäre eine gleichmäßige Auslieferung gewährleistet. Die dafür notwendigen Ressourcen-Reservierungen werden vom Dienst vorgenommen.

#### 4.2.1.3 Datenübertragung

- fehlerfreie Übertragung
- `get / put, directory`-Funktionen

Sofern der Dienst nicht selbst Dateien für den Abruf bereitstellt, ist der Dienst nur für die Kommunikation zwischen Sender und Empfänger zuständig, da dann der Empfänger die restlichen Funktionen bereitstellen muß. Dieser Kommunikationsdienst sollte zuverlässig sein, d.h. durch entsprechende Maßnahmen eine fehlerfreie Übertragung sicherstellen.

Bietet der Dienst selbst Dateien an, müssen vom Dienst Funktionen zum Directory-Handling und zum Übertragung der Dateien angeboten werden. Dabei wäre dann eine Unterstützung zum Fortsetzen von abgebrochenen Übertragungen sinnvoll, um komplette Neuübertragungen zu vermeiden.

### 4.2.2 Audio- und Videoübertragung

Für die Unterstützung von Multimedia-Anwendungen sind auf jeden Fall Dienste zur Video- und Audioübertragung notwendig. Hierbei sind jeweils zwei Arten zu unterscheiden, die jeweils einen unterschiedlichen QoS benötigen. Zum einen sollte der Abruf eines Videos bzw. Audiosamples aus einer Datenbank unterstützt werden. Zum anderen sollte auch eine Konferenz in Form von Video und/oder Audio ermöglicht werden. Eine Konferenz stellt gegenüber einem Abruf aus einer Datenbank strengere Anforderungen an die Ausbreitungsverzögerung. Wenn eine sinnvolle Konversation möglich sein soll, sollte die Verzögerung einen Wert von ca. 250 ms nicht überschreiten. Eine weitere Variante der Dienste Video- und Audioübertragung wäre eine Möglichkeit einer Anrufbeantworterfunktion, also das Hinterlassen von Video- und Audionachrichten, wenn der gewünschte Gesprächspartner gerade nicht erreichbar ist. Dies könnte unter Nutzung des Dienstes Email geschehen.

### 4.2.3 Email, News, Fax

Als weitere Kommunikationsdienste sollten das Versenden und Empfangen von Emails, News und Faxen ermöglicht werden. Hierbei wird davon ausgegangen, daß der jeweilige Dienst die Emails, News, Faxe für den Abruf zwischenspeichert.

#### 4.2.3.1 Email

- Übersicht über die eingegangenen Mails
- einzelne Email zum Lesen übertragen
- automatische Empfangsbestätigung beim Lesen erzeugen
- Email versenden (direkt oder zeitversetzt)
- Weiterleitung von Email
- Email-Filter

Der Dienst Email könnte neben dem normalen Nachrichtentransport die Funktion eines zeitversetzten Versendens ermöglichen. So könnte man z.B. sich selbst per Email an Termine erinnern, Bekannten am richtigen Tag Glückwünsche übermittelt, usw. Um die Zuverlässigkeit von Emails zu erhöhen, wäre eine automatische Benachrichtung an den Sender sinnvoll, wenn der Empfänger die Nachricht

abrufen. Der Sender weiß damit, daß und wann der Empfänger die Nachricht gelesen hat. Die Funktion eines Email-Filters ist gleich mehrfach sinnvoll. Zum einen kann damit unerwünschte Mail direkt vom Dienst gelöscht werden, indem der Dienst für den Anwender eine schwarze Liste von unerwünschten Absendern verwaltet. Zum anderen ist damit eine gezielte (vom Inhalt abhängige) Weiterleitung von Emails z.B. an verschiedene Ansprechpartner möglich.

#### 4.2.3.2 News

- Gruppenliste (alle Gruppen, neue Gruppen)
- Nachrichtenübersicht einer Gruppe
- bestimmte Nachrichten zum Lesen übertragen
- Nachricht versenden
- Nachrichtenauswahl durch Selektion mittels Anfragesprache
- automatisches Dekodieren von Binärdateien in Nachrichten

Der Dienst News könnte z.B. über eine Datenbankabfragesprache verfügen, über die der Nutzer gezielt nur die Informationen abfragen kann, die er auch benötigt. Dies könnte z.B. auch mit dem Dienst Email kombiniert werden, in dem die Ergebnisse vorgegebener Selektionen automatisch regelmäßig per Email dem Benutzer zugesandt werden. Ein Benutzer könnte sich so z.B. eine persönliche 'Zeitung' zusammenstellen. Für das Netz hätte dies den Vorteil, daß weniger Information unnötig transportiert würde und somit die Netzauslastung geringer wäre. Für den Netznutzer ist dies auch in Hinblick auf anfallende Nutzungskosten sinnvoll, wenn er gezielter Informationen abrufen kann.

#### 4.2.3.3 Fax

- Fax versenden
- Fax für Abruf bereitstellen

Bei Faxen könnte unterschieden werden, ob der Empfänger über das Netz erreichbar ist oder nur via Telefonleitung. Bei Erreichbarkeit über das Netz könnte wahlweise das Fax ausgedruckt werden oder alternativ auch als Email zugestellt werden. Umgekehrt könnte bei über Telefon beim Netz eingehenden Faxen mittels Zeichenerkennungsalgorithmen (OCR) der Empfänger ermittelt werden und diesem das Fax dann über das Netz zugestellt werden. Damit wäre jeder Netznutzer in der Lage, Faxe zu versenden und zu empfangen, ohne selbst ein Faxgerät zu benötigen.

Es zeigt sich also, daß eine Konvertierung zwischen einzelnen Diensten durchaus sinnvoll erscheint. Weitere Beispiele könnten die Umwandlung zwischen Email oder News in Sprache und umgekehrt sein.

Bei den Überlegungen, welche Dienste angeboten werden sollen, stellt sich eine weitere Frage. Ist eine Grundmenge an Diensten ausreichend, aus der dann weitere Dienste zusammengesetzt werden können (z.B. aus Videoübertragung & Audioübertragung → Videokonferenz) bzw. welche Vor- bzw. Nachteile hätte das Zusammensetzen eines neuen Dienstes aus mehreren anderen gegenüber der Definition eines neuen Dienstes unabhängig von bestehen Diensten?

Das Zusammensetzen eines neuen Dienstes aus bereits bestehenden hätte natürlich die typischen Modularisierungsvorteile, also daß nur vergleichsweise wenig neu implementiert werden muß und daß man von Änderungen an unterliegenden Diensten direkt profitieren kann, ohne an dem neu erstellten Dienst ebenfalls Änderungen vornehmen zu müssen. Andererseits könnten bei dieser Art der Realisierung eines Dienstes Probleme bei der Synchronisierung von zwei Diensten auftreten (beispielsweise bei Nutzung der Dienste Video und Audio, Stichwort Lippensynchronität). Würden dagegen die Audio und Videodaten von einem Dienst kodiert und komprimiert, z.B. mit MPEG-2, so wäre die Synchronität beim Abspielen auf jeden Fall gewährleistet.

### 4.3 Einflußmöglichkeiten auf die Dienste

Obwohl die Dienste die Anwendungen von Internas der Komprimierung und Netzwerkparametern befreien, ist das Einstellen von Parametern durch die Anwendung nach wie vor nötig. Durch diese Parameter soll der Anwendung bzw. dem Anwender die Möglichkeit gegeben werden, Einfluß auf die benutzten Dienste zu nehmen. Insbesondere sollte der Anwender die Qualität des Dienstes festlegen können, da davon direkt die Kosten der Nutzung des Dienstes bzw. des Netzes abhängen, denn je höher die Qualität, desto höher das zu übertragende Datenvolumen, desto höher die anfallenden Kosten.

Es stellt sich daher die Frage, wie diese Qualitätsschnittstelle zwischen Anwender und Dienst aussehen sollte. Dafür ist es wichtig, zuvor zu überlegen, wie das Aushandeln der Dienstgüte mit dem Dienst und gegebenenfalls zusätzlich mit einem Empfänger ablaufen soll. Dieses Aushandeln ist notwendig, da zum einen von Dienstseite aus zu wenig Ressourcen zur Verfügung stehen könnten und zum anderen Einschränkungen durch die Hardware des Empfängers vorgenommen werden könnten.

In [Dant94] werden diesbezüglich drei unterschiedliche Abläufe unterschieden:

1. Der Dienstanutzer unterbreitet dem Dienst einen Qualitätsvorschlag. Dieser kann sowohl vom Dienst als auch Empfänger abgeschwächt werden. Die sich danach ergebene Qualitätseinstufung wird zurück an den Dienstanutzer gesendet.
2. Der Dienstanutzer gibt ein Intervall bestehend aus gewünschter und schlechtestenfalls akzeptierter Qualität vor. Der Dienst darf nur den Wert für gewünschte Qualität abschwächen, nicht aber die Untergrenze. Der Empfänger entscheidet dann aus diesem Intervall über den wirklichen Wert.

Gegebenfalls muß die Verbindung vom Dienst oder Empfänger abgelehnt werden.

3. Die Vorgabe des Dienstnutzers ist zwingend. Entweder sie wird sowohl von Dienst und Empfänger akzeptiert oder die Verbindung kommt nicht zustande.

Bei der ersten Möglichkeit kommt normalerweise immer eine Verbindung zustande. Allerdings kann durch die Abschwächung des Netzes und des Empfängers die Qualität so stark reduziert worden sein, daß der Dienstnutzer damit nicht zufrieden sein könnte.

Die zweite Möglichkeit gewährleistet beim Zustandekommen einer Verbindung, daß die Qualität des Dienstes zumindest oberhalb der vom Dienstnutzer vorgegebenen Untergrenze liegt. Andererseits besteht hier die Gefahr, daß die vorgegebene Untergrenze vom Dienst oder Empfänger nicht erfüllt werden kann und somit keine Verbindung zustande kommt.

Die dritte Möglichkeit ist die am stärksten einschränkende. Sie hat wie die zweite Möglichkeit den Nachteil, daß Verbindungen nicht zustande kommen können. Dem Dienstnutzer bleibt dann nichts anderes übrig, als zu warten und später erneut seine Qualitätsanforderung zu stellen, in der Hoffnung, daß jetzt mehr Ressourcen zur Verfügung stehen, oder seine Anforderung abzuschwächen.

Die zweite Möglichkeit scheint die meisten Vorteile zu bieten. Zum einen wird den Ansprüchen des Dienstnutzer so weit wie möglich nachgekommen (wenn man davon ausgeht, daß die Anforderungen des Dienstnutzers nicht unnötig durch Dienst oder Empfänger abgeschwächt werden), zum anderen kann die Notwendigkeit eines erneuten Aushandelns häufig vermieden werden, sofern das vorgegebene Intervall nicht zu eng gewählt wird.

In diesem Zusammenhang stellt sich die Frage, wie die Qualitätsvorgaben des Anwenders genau aussehen sollen. Soll der Anwender z.B. bei einer Benutzung des Dienstes Videoübertragung nur zwischen verschiedenen Vorgaben (z.B. geringe, gute, sehr gute Qualität) auswählen können oder soll ihm die Möglichkeit gegeben werden, gezielt die Qualität in Form von Parameter (Bildgröße, Anzahl Farben, Bilder pro Sekunde) festzulegen?

Durch die Angabe von Parametern hätte der Anwender mehr Einfluß auf den Dienst. Allerdings würde damit auch die Nutzung des Dienstes komplizierter werden, da sich nun der Anwender wieder mit der Festlegung von Parametern beschäftigen muß. Andererseits wird ein Anwender die Bedeutung dieser Art von Parametern (gegenüber den Netzwerk-QoS-Parametern) eher abschätzen können. Damit würde allerdings das Aushandeln der Qualität aufwendiger, da nun die Einhaltung einer Vielzahl von Werten überprüft werden muß. Hier könnte man sich auch die Vorgabe in Form einer Gewichtung vorstellen, z.B. lieber größeres Bild und dafür weniger Farben oder weniger Bilder pro Sekunde. Hier stellt sich die Frage, inwieweit vorhandene Videokonferenzsysteme (insbesondere die verwendeten Kameras) überhaupt dem Benutzer eine Wahl der Bildgröße, Farben und Bildraten gestatten. Dies kann an dieser Stelle leider wegen Mangel an entsprechenden Informationen nicht geklärt werden.

## Kapitel 5

# Zusammenfassung

In dieser Arbeit wurden die Anforderungen von verteilten Multimedia-Anwendungen an die verwendeten Übertragungsprotokolle erläutert. Neben einer hohen Übertragungsrate, um dem bei Multimedia-Anwendungen zumeist großen Datenvolumen gerecht zu werden, wird insbesondere eine begrenzte maximale Sendeverzögerung als auch eine begrenzte Verzögerungsvarianz benötigt. Diese Dienstgütekriterien müssen vom Übertragungsprotokoll für die gesamte Dauer einer Verbindung garantiert werden, wenn die Präsentation der Multimediadaten auf Empfängerseite in einem akzeptablen Bereich (z.B. kein Ruckeln bei einer Videoübertragung, keine zu große Sendeverzögerung bei einer Sprachkonferenz) geschehen soll.

Diesen Anforderungen wird das TCP/IP-Protokoll nicht gerecht, da es keine zeitlichen und quantitativen Garantien bietet. Es wurden daher die Protokolle RSVP, ST2 und ATM vorgestellt, die durch ein spezielles Zusammenspiel mit der Hardware in der Lage sind bestimmte Dienstgütekriterien zu garantieren.

Anhand der auf (TCP/)IP aufsetzten Beispielanwendung VCR, die Audio- und Videodatenströme von unterschiedlichen Servern synchronisiert abspielen kann, wurde auf die Probleme bei der Adaption einer IP-basierten Multimedia-Anwendung auf ATM eingegangen. Es wurde die Umstellung auf Kommunikationsfunktionen der API der Firma FORE beschrieben, wobei allerdings aufgrund der eingeschränkten Mächtigkeit der vorliegenden API der Firma FORE Einschränkungen in der Funktionalität der Anwendung VCR vorgenommen werden mußten. Darüberhinaus wurde die Problematik bei der Festlegung der Dienstgüteparameter erläutert.

Letzteres führte zu der Überlegung, ob nicht eine bessere Möglichkeit geschaffen werden könnte, Multimedia-Anwendung bzw. die Erstellung derselben zu vereinfachen. Dazu wurde in einem ersten Ansatz Netzwerkdienste vorgestellt, die allen Anwendungen zur Verfügung stehen sollten. Diese Netzwerkdienste sollen bereits die am häufigsten von einer Multimedia-Anwendung benötigten Funktionen enthalten und so die Anwendung von Fragen der Kodierung, Kompression und Netzwerkinternas zu befreien. Es wurden einige als sinnvoll erachtete Dienste aufgeführt und auf die Einflußmöglichkeiten auf diese Dienste durch den Benutzer

eingegangen. Dabei wurden die Vor- und Nachteile verschiedener Möglichkeiten erläutert.

In weiteren Arbeiten werden das Konzept der Netzwerkdienste weiterverfolgt und genauer ausarbeitet werden. Andere Arbeiten werden sich mit der Erstellung einer Socket-API beschäftigen, die intern direkt auf ATM-Kommunikationsfunktionen aufsetzt und die Funktionalität von TCP/UDP bietet. Dadurch wäre es möglich, den auf IP aufsetzenden Anwendungen ohne größeren Aufwand die Dienstgütegarantien von ATM bereitzustellen.

# Literaturverzeichnis

- [ATM94-0640] M. Schwartz, D. Beaumont, *Quality of Service Requirements for Audio-Visual Multimedia Services.*, ATM Forum, ATM94-0640, Juli 1994
- [Cisc95] Anthony Alles, *ATM Internetworking*, Cisco Brochures, Mai 1995
- [Dant94] A.Danthine, O.Bonaventure, *From best effort to enhanced QoS*, 1994,  
<ftp://www-run.montefiore.ulg.ac.be/pub/papers/94/R94-09.ps.gz>
- [dePr93] Martin de Prycker, *Asynchronous Transfer Mode, Solution for Broadband ISDN*, Second Edition, 1993
- [Flan94] William A. Flanagan, *ATM User's Guide*, First Edition, April 1994
- [Fore96] FORE-Systems *LAN Emulation, Virtual LANs, and ATM Internetworks*, ATM White Paper, Jan. 1996
- [Hein96] Bernd Heinrichs, *Multimedia im Netz*, Springer Verlag, 1996
- [I321] CCITT Recommendation I.321; *B-ISDN Protocol Reference Model and its Application*, Genf, 1991
- [ISO11172] ISO/IEC JTC1: *Information Technology - Coding of Moving Pictures and Associated Audio for Digital Storage Media up to about 1.5 Mbit/s*, Draft International Standard ISO/IEC DIS 11172, 1992
- [LANE95] ATM Forum Specification, *LAN Emulation Over ATM Specification - Version 1*, Februar 1995
- [RFC1190] C. Topolcic, *Experimental Internet Stream Protocol, Version 2 (ST-II)*, RFC 1190, Oktober 1990
- [RFC1363] C.Partridge, *A Proposed Flow Specification*, RFC 1363, September 1992
- [RFC1577] M.Laubach, *Classical IP and ARP over ATM*, RFC1577, Januar 1994

- [RFC1819] L.Delgrossi, L.Berger, *Internet Stream Protocol Version 2 (ST2)*, Protocol Specification - Version ST2+, RFC 1819, August 1995
- [RFC1821] M.Borden, E.Crawley, B.Davie, S.Batsell, *Integration of Real-time Services in an IP-ATM Network Architecture*, RFC 1821, August 1995
- [RFC1889] H.Schulzrinne, S.Casner, R.Frederick, V.Jacobson, *RTP: A Transport Protocol for Real-Time Applications*, RFC 1889, Januar 1996
- [RSVP93] L.Zhang, S.Deering, D.Estrin, S.Shenker, D.Zappala, *RSVP: A New Resource ReSerVation Protocol*, IEEE Network, September 1993  
<ftp://parcftp.xerox.com/pub/net-research/rsvp.ps.Z>
- [RSVP96] R.Braden, L.Zhang, S.Berson, S.Herzog, S.Jamin, *Resource ReSerVation Protocol (RSVP)*, Version 1 Functional Specification, Internet Draft, November 1996
- [RTP96] Armin Schieber, *RTP (Real Time Transport Protocol)*, August 1996  
<http://eratosthenes.informatik.uni-mannheim.de/~whd/seminar-ss96/Armin.Schieber/>
- [Tan92] A.S. Tanenbaum, *Computer-Netzwerke*, Wolfram's Verlag, 2. Auflage, 1992
- [VCR] Shanwei Cen, Calton Pu, Richard Staehli, Crispin Cowan, Jonathan Walpole, *A Distributed Real-Time MPEG Video Audio Player*,  
<http://cse.ogi.edu/DISC/projects/synthetix/Player/>
- [X200] CCITT Recommendation X.200; *Reference Model for Open Systems Interconnection for CCITT Applications*, Blue Book, Fascicle VIII.4, Genf, 1989
- [Zitt95] Martina Zitterbart, *Hochleistungskommunikation, Band 1: Technologie und Netze*, Oldenbourg, 1995

## **Erklärung**

Hiermit erkläre ich, Jörg Illerich, daß ich die vorliegende Diplomarbeit selbständig verfaßt und keine anderen als die angegebenen Hilfsmittel verwendet habe.

Kaiserslautern, den 7. Mai 1997