Ein AppleTalk Link Access Protocol basierend auf dem Abstract Personal Communications Manager

Alfred Lupper, Konrad Froitzheim
Universität Ulm
Abt. Verteilte Systeme
7900 Ulm, Germany
lupper@informatik.uni-ulm.de

Für lokale Netzwerke in Ring- oder Busstrukturen sind zahlreiche Protokolle und Systeme implementiert und im Einsatz. Als Übertragungsmedium kommen alternativ ISDN-Kanäle in Frage. Deren leitungsvermittelnde Sternstruktur unterscheidet sich in wichtigen Eigenschaften fundamental von herkömmlichen LAN-Strukturen. Insbesondere die konstante, aber auf 64 kbit/s beschränkte Übertragungsleistung, sowie die leitungsvermittelnde Topologie und die daher fehlende Broadcast-Fähigkeit nehmen Einfluß auf die Implementierung von Kommunikationsprotokollen über ISDN. Dieser Artikel befaßt sich eingehend mit den Randbedingungen, die die Anpassung des AppleTalk-Protokollstapels an die leitungsvermittelnde Sternstruktur des ISDN bestimmen. Hieraus werden Konzepte zur Abbildung von konkurrenten AppleTalk-Transaktionen auf ISDN-Kanäle, zur automatischen Verbindungssteuerung und zur Adressierung in einem leitungsvermittelnden LAN abgeleitet. Schließlich wird ein ISDN Link Access Protocol für AppleTalk vorgestellt, das auf dem an der Universität Ulm entwickelten APCM-Interface für ISDN-Wählverbindungen basiert.

1. Einsatzszenarien

Telekommunikationsanlagen in ISDN-Technik (Integrated Services Digital Network) können neben dem Telefondienst auch zur Datenübertragung verwendet werden. Dabei werden in einem 64 kbit/s-Kanal Daten anstelle von digitalisierter Sprache übertragen. Es liegt nahe, diese Leitungen auch für die Realisierung von Funktionen lokaler Netzwerke (LAN) zu verwenden [FROITZHEIM, 87]. Besonders für die Datenübertragung außerhalb des lokalen Bereiches, d.h. im öffentlichen Netz, empfiehlt sich die Verwendung von ISDN-Kanälen.

ISDN-Verbindungen können zusammen mit LANs in verschiedenen Szenarien eingesetzt werden. Das auf dem Abstract Personal Communications Manager (APCM, [FROITZHEIM, 91]) basierende Link Access Protocol (APCMLAP) bietet einerseits die Möglichkeit bestehende AppleTalk-LANs mit Hilfe von ISDN-Verbindungen über den lokalen Bereich hinaus durch abgesetzte Knoten zu erweitern, andererseits können durch die Installation in einem Router mehrere LANs untereinander verbunden werden. Der Einsatz eines Name Servers ermöglicht außerdem den Aufbau eines vollständig ISDN-basierten AppleTalk-Netzwerkes mit voller LAN-Funktionalität.

1.1 Punkt-zu-Punkt-Verbindungen

Die einfachste Art, zwei Rechner über ISDN miteinander zu verbinden, ist eine Punkt-zu-Punkt-Verbindung, die für die Dauer der Kommunikation bestehen bleibt. Da nur mit einem einzigen Teilnehmer kommuniziert wird, kann die Rufnummer des Zielknotens manuell eingegeben werden. Es ist keine automatische Zuordnung von Knotenadresse und Rufnummer notwendig.

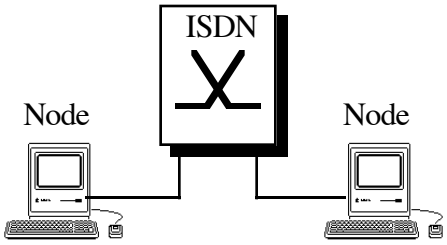

Abbildung 1.1: Punkt-zu-Punkt-Verbindungen über ISDN

Wählverbindungen über einen längeren Zeitraum bestehenzulassen ist jedoch mit Problemen verbunden:

- Gespräche werden im öffentlichen ISDN ohne Berücksichtigung der Datenmenge nach der Verbindungsdauer abgerechnet.
- Die Leitungen der kommunizierenden Knoten werden monopolisiert, d.h. für andere ist 'besetzt'.

Daher ist es unter Umständen auch bei Punkt-zu-Punkt-Verbindungen sinnvoll, eine Verbindungssteuerung einzusetzen, die die Gebührenstruktur im inhouse-Bereich oder im öffentlichen Fernmeldenetz, sowie die Möglichkeit der Monopolisierung berücksichtigt.

1.2 Abgesetzte Knoten in einem bestehenden LAN

Im Zusammenhang mit LANs ist die häufigste Einsatzmöglichkeit für ISDN-Verbindungen die transparente Integration von abgesetzten Knoten. Weder für die abgesetzte Station selbst, noch für die Kommunikationspartner im LAN soll ersichtlich sein, daß die Verbindung über ISDN hergestellt wird.

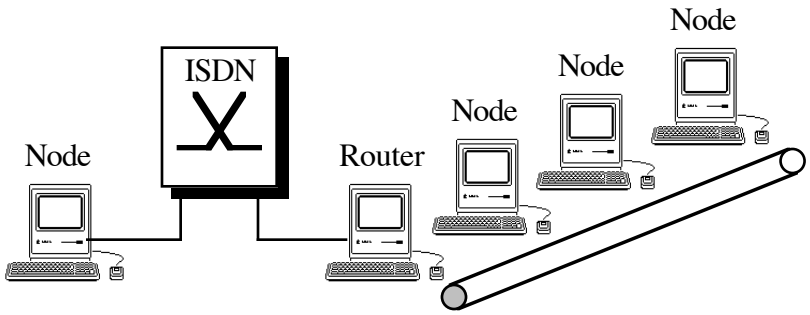

Abbildung 1.2: Integration eines abgesetzten LAN-Knotens über ISDN

Hierzu ist die Installation einer Brücken- oder Router-Funktion in einem Knoten notwendig, der sowohl Zugang zum ISDN, als auch zum LAN besitzt (Abbildung 1.2). Die Adressierung ist ebenso unproblematisch wie im Fall 1.1, da alle Pakete und somit auch Broadcast-Pakete an den Knoten mit der Routing-Funktion gerichtet werden. Der abgesetzte Knoten kann dabei eine eigene AppleTalk-Zone[1] bilden oder aber derselben Zone angehören, wie die übrigen Knoten im LAN auch. Dies ist nur von der Konfiguration des AppleTalk-Routers abhängig.

Die Verbindung zwischen dem Routing-Knoten und dem ISDN-Knoten kann prinzipiell für die Dauer der Kommunikation bestehen bleiben. Aus denselben Gründen wie unter 1.1 ist jedoch die Steuerung der Verbindung sinnvoll.

---

[1] Eine Zone in AppleTalk bezeichnet eine willkürliche Untermenge von Netzwerken im Internet, die eine logische Einheit bildet. Ein gegebenes Netzwerk gehört zu genau einer Zone. Die Netzwerke einer bestimmten Zone müssen in keiner Weise zusammenhängend oder benachbart sein (siehe hierzu auch [APPLE IAT, 86])

1.3 Kopplung von LAN-Segmenten über ISDN

Neben der Integration einzelner Knoten kann die Kopplung von eigenständigen LAN-Segmenten über ISDN erreicht werden. Prinzipiell ist es möglich, die LAN-Segmente zu einer einzigen logischen Zone zu verbinden. Um den Verkehr über die ISDN-Verbindung zu reduzieren, erscheint es jedoch ratsam, für jedes LAN-Segment eine eigene Zone zu bilden.

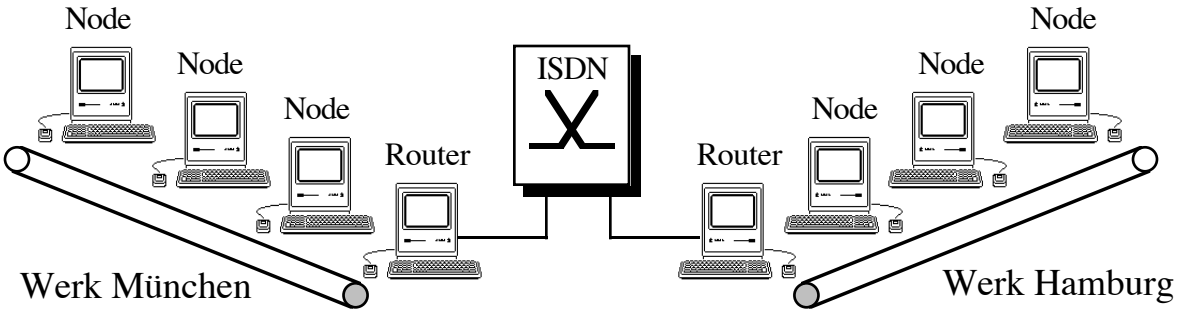

Abbildung 1.3: Kopplung von eigenständigen LANs über ISDN-Verbindungen

Die Adressierung bei der LAN-LAN-Kopplung ist relativ unproblematisch, da ein Router jeweils nur die Rufnummer des gegenüberliegenden Routers kennen muß. Bei mehreren ISDN-Ports an einem Router besitzt jedes LAP des Routers am ISDN eine eigene Rufnummer.

Da LAN-Segmente, die über ISDN gekoppelt sind, sich in der Regel nicht im lokalen Bereich befinden, sondern über das öffentliche Netz kommunizieren, können Verbindungen aus Kostengründen nicht über längere Zeit bestehen. Es ist ebenso nicht möglich, die Dauer einer Wählverbindung zwischen zwei LAN-Segmenten von der Transaktionsdauer zwischen zwei Knoten abhängigzumachen, da die ISDN-Verbindung zwischen den Segmenten keiner Transaktion eindeutig zugeordnet werden kann.

1.4 LAN-Emulation über ISDN

An den beiden vorangegangenen LAN-Szenarien war jeweils ein Standard-LAN beteiligt. Über ISDN kann jedoch auch ein LAN□ ohne das Partizipieren eines realen Buses/Rings aufgebaut werden. Da ISDN nur Punkt-zu-Punkt-Verbindungen unterstützt, muß für jede logische Verbindung auch eine physikalische Verbindung hergestellt und kontrolliert werden.

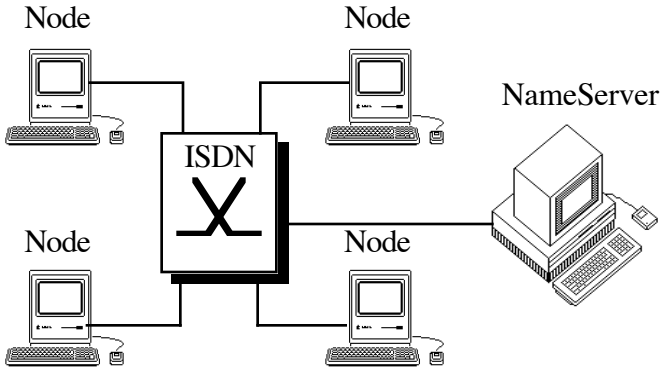

Abbildung 1.4: Emulation einer LAN-Topologie über ISDN mit Name Server

Schwierig bei der LAN-Emulation ist die Adressierung von Knoten, da die LAN-Protokolle normalerweise exzessiven Gebrauch von der Broadcast-Fähigkeit des Mediums machen, um Ressourcen im Netz zu lokalisieren (Name Binding). Diese ist jedoch im ISDN nicht gegeben, so daß eine Methode gefunden werden muß, die gewohnte Adressierung ohne Protokoll-substitution auf höherer Schicht auch bei Punkt-zu-Punkt-Verbindungen zu gewährleisten. Das geschieht typischerweise durch einen Name Server, eine ausgezeichnete Maschine im Netz, die

die Abbildung von Ressource-Namen auf Rufnummern (Mapping) durchführt (siehe Kapitel 6 und [LUPPER, 90]).

Bei der Emulation eines LANs dürfen Verbindungen nur für kurze Zeit bestehen, da ein Knoten in der Lage sein muß, mehrere logische Verbindungen zu unterstützen und die Gefahr der Monopolisierung einer Ressource besteht. Mehrere logische Verbindungen eines Knotens zu unterschiedlichen Zielknoten implizieren einen permanenten Wechsel der physikalischen Verbindung.

1.5 Kopplung ISDN-LAN und Standard-LAN

Schließlich ist eine Konfiguration denkbar, die aus einer Kombination von ISDN-LAN und Standard-LAN besteht. Abbildung 1.5 zeigt eine derartige Anordnung von Knoten. Durch die Konfiguration des Routers ist es möglich, die beiden Segmente zu einer einzigen Zone zu vereinen oder die Segmente als eigenständige Zonen zu betrachten.

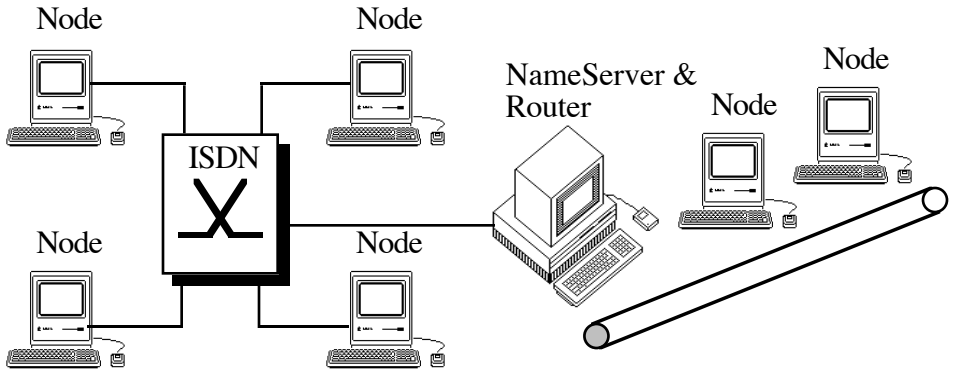

Abbildung 1.5: Emulation einer LAN-Topologie über ISDN mit Name Server

Für die Adressierung im Netzwerk entstehen dann zusätzliche Probleme, da nun die Adressierung im ISDN-Segment auch die Interzonenadressierung unterstützen muß, d.h. eine Station im ISDN-Segment muß auch wissen, welche Zonen vorhanden sind und wo der nächste Router zu finden ist. Etwas einfacher wird die Adressierung, falls Name Server- und Routing-Funktion in einer Station vereint sind, jedoch ist dann die Implementierung eines LAP schwieriger.

Die Problematik der Adressierung läßt sich durch ein Backbone-LAN umgehen. Alle Netzwerkknoten werden dann als einzelne abgesetzte Knoten betrachtet, die voll in das Standard-LAN integriert sind. Diese Konfiguration verursacht jedoch doppelte Verbindungskosten. Außerdem ist es ineffizient, wenn ISDN-Knoten, die sich in München befinden, über ein Backbone-LAN in Hamburg kommunizieren.

Für die Verbindungssteuerung in dieser Konfiguration gilt das bei Punkt 1.4 gesagte.

In Verbindung mit einem Router und durch den Einsatz eines Name Servers lassen sich die unterschiedlichsten Netzwerkkonfigurationen realisieren. Voraussetzung ist, daß das LAP für das ISDN (APCMLAP) mit dem Router kompatibel ist.

2. Restriktionen der leitungsvermittelnden ISDN-Struktur

Zunächst wollen wir uns mit den Restriktionen für die Realisierung von LAN-Funktionen befassen, die uns das Kommunikationsmedium ISDN auferlegt. Die leitungsvermittelnde, sternförmige Struktur des ISDN besitzt dabei gegenüber einer Busstuktur folgende gravierenden Nachteile:

- Die Übertragungsleistung eines einzelnen ISDN-Kanals ist verglichen mit der Übertragungskapazität von typischen LANs mit Busstruktur (ca. 1–100 MBit/sec.) gering.
- Die Verfügbarkeit des zentralen Vermittlers muß stets gegeben sein, da ein Ausfall zur Auflösung des gesamten Systems führt.
- Die **mangelnde Broadcast-Fähigkeit** der Leitungsvermittlung ist ein weiterer schwerwiegender Nachteil, der bei der Konzipierung eines LAN für eine Sternstruktur berücksichtigt werden muß. Dem Broadcasting auf dem Bus steht das Circuit Switching der Vermittlungsanlage gegenüber.
- Während Knoten in LANs mit Busstruktur mehrere virtuelle Verbindungen gleichzeitig unterhalten können, unterstützen leitungsvermittelnde Systeme jeweils nur Punkt-zu-Punkt-Verbindungen zu einem Zeitpunkt.
- Der Aufbau einer Wählverbindung beansprucht sehr viel mehr Zeit als das Herstellen einer logischen Verbindung im LAN mit Busstruktur (Faktor: ca. 100–200).

Dennoch werden ISDN-Verbindungen für LAN-Funktionen eingesetzt. Die Hauptgründe, warum sternförmige Rechnernetze auf ISDN-Basis entstehen, sind folgende:

- Oft kann auf vorhandene leitungsvermittelnde Strukturen, wie digitale Nebenstellenanlagen, zurückgegriffen werden. Somit entstehen nur geringe zusätzliche Kosten für die Installation der Infrastruktur eines Computernetzes.
- Die Zugriffe bei Bussen werden i.a. durch stochastische Methoden und bei Ringen durch deterministische Token-Verfahren geregelt. Bei leitungsvermittelnden Sternstrukturen kann auf eine aufwendige Mediumszugriffskontrolle (MAC) verzichtet werden, da über den Steuerknoten immer nur Punk-zu-Punkt-Verbindungen bestehen. Somit entsteht bei der Datenübertragung kein zusätzlicher Overhead durch die Zugriffskontrolle. Die Übertragungsprotokolle auf Schicht 2 werden dadurch erheblich einfacher.
- Die Gesamtübertragungsleistung eines leitungsvermittelnden Systems kann bei entsprechender Verteilung des Verkehrs ein Vielfaches eines einzelnen ISDN-Kanals betragen (z.B. n * 64 kbit/s).
- Die Verbindung von ein zelnen LANs zum WAN kann nur durch Dienste öffentlicher Telekommunikationsgesellschaften geschehen. Dabei bietet ISDN den kostengünstigsten und leistungsfähigsten Übertragungsdienst.

Um trotz der relativ geringen Übertragungsrate eines einzelnen ISDN-Kanals und der leitungsvermittelnden Struktur des ISDN einen akzeptablen Durchsatz zu erzielen, bedarf es einer behutsamen Implementierung und einer geeigneten Verbindungssteuerung, die die Besonderheiten dieser Kommunikationsstrukturen berücksichtigt und die Vorteile des ISDN nutzt.

3. Restriktionen der Protokolle in AppleTalk

Auf den höheren Schichten der vorliegenden ISDN-LAN-Anbindung und für die LAN-Emulation wurde der AppleTalk-Protokollstapel verwendet. Daher werden in folgenden Abschnitt dessen Protokolle kurz skizziert und anschließend die Möglichkeit zur Substitution der Link-Ebene aufgezeigt .

3.1 Schichten und Protokolle in AppleTalk

AppleTalk wurde nach dem OSI-Schichtenmodell entworfen. Die verschiedenen Funktionen werden nach logischen Gesichtspunkten in die sieben OSI-Schichten eingeordnet. Die unterste, die **physikalische Schicht** bildet die LocalTalk-Hardware, normalerweise ein zweidrahtiges verdrilltes Kabel, das über Induktionskopplungen abgegriffen wird.

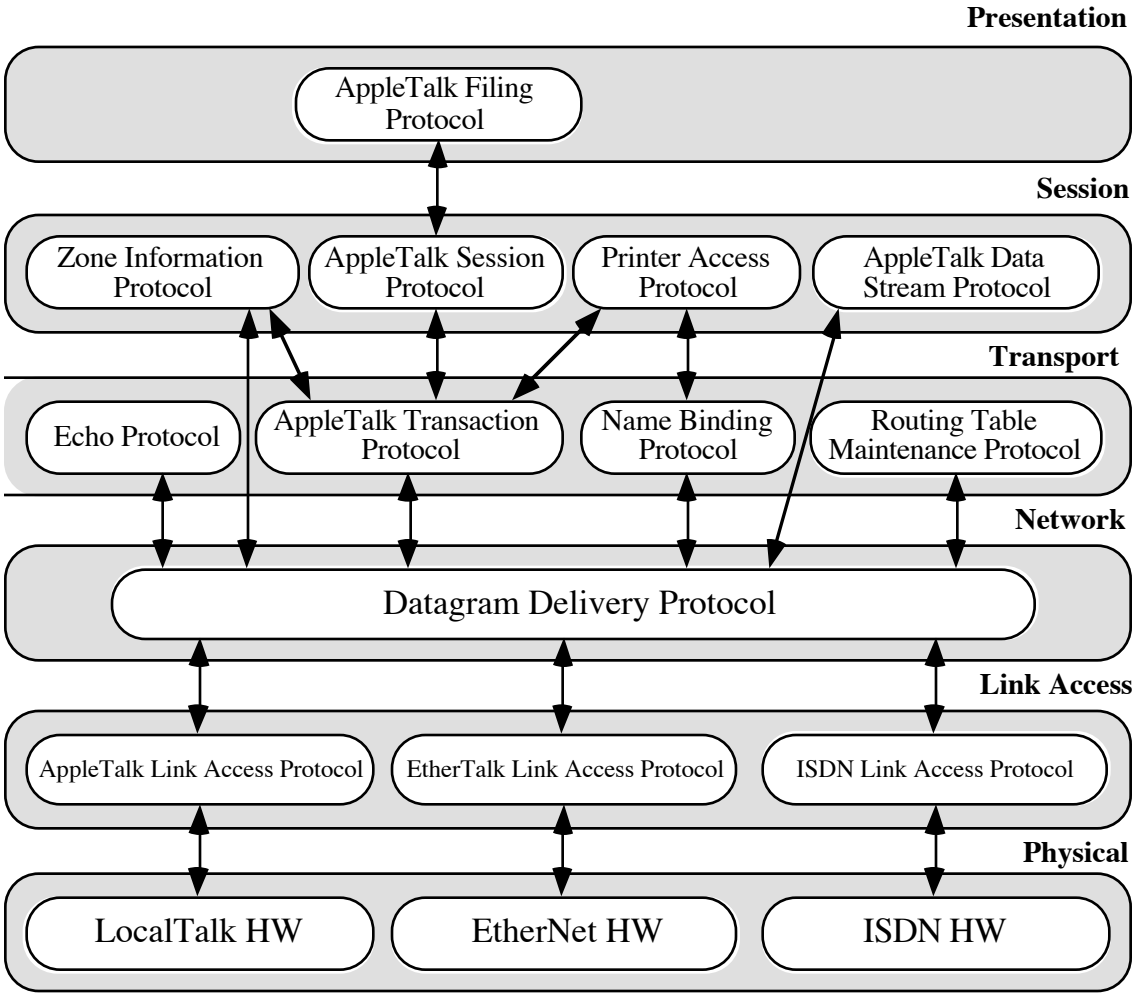

Abbildung 3.1: AppleTalk-Protokolle im ISO-Schichtenmodell

Die darüberliegende **Verbindungsschicht** ist durch das **AppleTalk Link Access Protocol (ALAP)** realisiert, das für die Übertragung von Paketen von Knoten (Node) zu Knoten, die Zugriffskontrolle auf die Hardware und die dynamische Node-Adressierung verantwortlich ist. Für unterschiedliche physikalische Schichten, wie zum Beispiel die Ethernet-Hardware oder die normale Telefonleitung, existieren verschiedene Link Access-Protokolle.

Das **Datagram Delivery Protocol (DDP)** der **Netzwerkschicht** ist für die Übertragung von Datagrammen über Zonengrenzen hinweg an die logischen Adressen in den Geräten, den sog. Sockets, eines Netzwerkes verantwortlich. Es bedient sich dabei der Dienste des ALAP.

Auf der **Transportschicht** des AppleTalk befinden sich mehrere Protokolle mit unterschiedlichen Aufgaben. Die wichtigsten sind das **AppleTalk Transaction Protocol (ATP)** und das **Name Binding Protocol (NBP)**. Das ATP ist für die gesicherte Übertragung von Paketen verantwortlich. Das NBP dient der Zuordnung von Gerätenamen, Gerätetyp und Zonenbezeichnung zu Socket-, Knoten- und Netzwerkadresse. Die Verwaltung von Weglenkungstabellen in AppleTalk-Brücken übernimmt das **Routing Table Maintenance Protocol (RTMP)**. Als Klient des DDP werden RTMP und NBP der Transportschicht zugeordnet, obwohl sie keine Transportfunktionen im Sinne von OSI erfüllen. Ein **Echo Protocol (EP)** dient zum Testen von Verbindungen und zur Bestimmung der Übertragungszeit zwischen zwei Knoten. Alle Protokolle der Transportschicht sind Klienten des DDP.

Über der Transportschicht ist die **Session-Schicht** angeordnet. Auch hier befinden sich mehrere Protokolle mit den verschiedensten Aufgaben (siehe Abbildung 3.1). Ein **Zone Information Protocol (ZIP)** dient dazu, von einer AppleTalk-Brücke die Namen und Adressen weiterer Zonen anzufordern. Das **Print Access Protocol (PAP)** ermöglicht das Drucken über AppleTalk. Ein verbindungsorientiertes Protokoll, das **AppleTalk Data Stream Protocol (ADSP)**, überträgt einen zuverlässigen Vollduplex-Bytestrom zwischen zwei beliebigen Sockets. Bei der Übertragung bleibt die Ordnung des Datenstrom erhalten und es treten keine Wiederholungen auf.

Das **AppleTalk Session Protocol (ASP)** übernimmt den Aufbau, das Abhalten und das Abbrechen einer Session. Über das ASP kann eine Workstation Kommandos an einen Server leiten. Der Server antwortet mit Statusmeldungen oder den angeforderten Daten.

Das **AppleTalk Filing Protocol (AFP)** der darüberliegenden **Präsentationsschicht** ermöglicht einer Workstation den Zugang zu Dateien eines Fileservers. Aufrufe an das Dateisystem des Macintosh OS werden in AFP-Calls verwandelt und an das externe Dateisystem, i.a. einen Fileserver, weitergegeben.

Für die Implementierung eines ISDN-LAN besitzen die Protokolle des AppleTalk Protocol Stacks folgende Vor- und Nachteile:

+ Bemerkenswert an der AppleTalk-Netzwerkarchitektur ist, daß sich auf den Schichten 2 und 3 einfache Protokolle mit wohldefinierten Schnittstellen befinden, die leicht zu substituieren sind.

+ Die Vielzahl der AppleTalk-Dienste bietet eine gute Basis ein ISDN-LAN für verschiedene Anwendungen umfassend zu testen.

– Die Adressierungsprotokolle des AppleTalk nutzen für wesentliche Funktionen die Broadcast-Fähigkeit des Busses und setzen auf der physikalischen Schicht implizit eine Busstruktur voraus.

– Die höheren Protokolle unterstützen gleichzeitig mehrere logische Verbindungen oder Sitzungen mit verschiedenen Netzwerkteilnehmern.

– Die Protokolle der unteren Schichten multiplexen die logischen Verbindungen der höheren Schichten.

– Das AppleTalk Session Protocol tauscht periodisch sog. Tickle-Pakete zwischen Clienten und Servern einer Sitzung aus. Dadurch entsteht Verkehr, ohne daß Nutzdaten ausgetauscht werden.

– Das ASP und das ATP bieten keine Anhaltspunkte für die Implementierung einer transaktions- oder sitzungs-orientierten Verbindungssteuerung.

Die Protokolle des AppleTalk Protocol Stacks wurden für eine paketvermittelnde Busstruktur mit Broadcast-Fähigkeit entworfen und müssen teilweise an die Eigenschaften des ISDN angepaßt werden.

3.2 LAP-Substitution mittels LAP Manager[2]

Da AppleTalk nach dem OSI-Schichtenmodell gegliedert ist, besteht prinzipiell die Möglichkeit verschiedene Protokolle des AppleTalk gegen neue Versionen und Implementierungen auszutauschen. Die diversen Möglichkeiten zur Substitution von Protokollen in AppleTalk wurden in einer vorangegangenen Arbeit [LUPPER, 90] bereits diskutiert und sollen hier nicht erneut behandelt werden. Lediglich die Möglichkeit, das Link Access Protocol im AppleTalk Protocol Stack mit dem LAP Manager zu ersetzen, soll im folgenden kurz aufgezeigt werden.

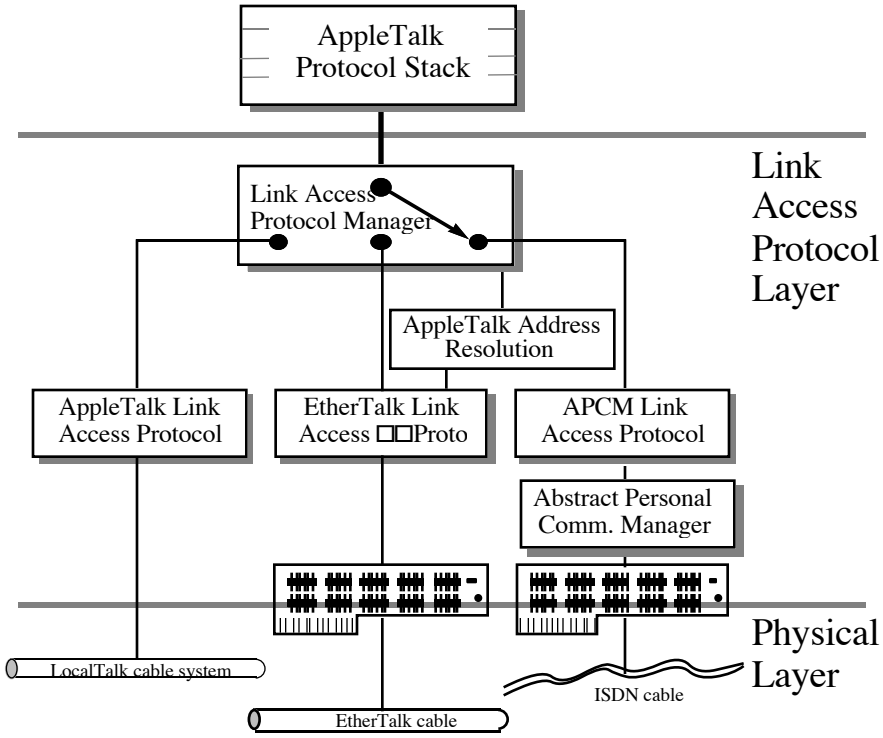

Abbildung 3.2: Wirkungsweise des LAP Managers

Der LAP Manager ist ein Interface zwischen dem LAP und den übergeordneten Schichten (vgl. Abbildung 3.2). Eine Substitution des LAP durch den LAP Manager hat folgende Vor- und Nachteile:

+ Der LAP Manager bietet eine schmale Schnittstelle zum Austausch des LAP.
+ Die Installation eines alternativen Link Access Protocols wird für den Benutzer sehr einfach, er kann problemlos zwischen mehreren AppleTalk-Übertragungsmedien umschalten.
+ Nicht zuletzt wird diese Methode von Apple Computer empfohlen. Deshalb ist zu erwarten, daß die Kompatibilität mit künftigen AppleTalk-Versionen gewährleistet ist.
+ Zur Substitution des LAP ist keine Modifikation des .MPP-Treibercodes notwendig.
– Die vorhandene Dokumentation zum LAP Manager ist in vielen Fällen unzureichend, so daß bei der LAP-Implementierung viele offene Fragen verbleiben, die oft nur mit Detektivarbeit beantwortet werden können.

Zur Einbindung des Link Access Protocols für ISDN in den AppleTalk Protocol Stack wurde der LAP Manager verwendet.

---

2 Eine detaillierte Beschreibung des Einsatzes des LAP Managers wird in [LUPPER, 90] gegeben.

4. Der Abstract Personal Communications Manager

Der Abstract Personal Communications Manager[3] ist ein Application Programmers Interface (API) für Telekommunikation. Er stellt ein einfaches aber mächtiges Bindeglied zwischen Kommunikationsapplikationen und Telefonnetzwerken dar, das die Kontrolle von leitungsvermittelten Verbindungen und die Übertragung von Sprache und Daten gewährleistet.

Der APCM wurde für verschiedene Netzwerke entworfen, besonders geeignet ist er jedoch für ISDN- und PBX-Verbindungen, sowie für das herkömmliche analoge Telefonsystem. Der APCM implementiert verschiedene Protokolle und Interfaces, verbirgt jedoch deren Unterschiede und stellt dem Klienten eine einheitliche Programmierschnittstelle (Datenstrukturen und Routinen) zur Verfügung.

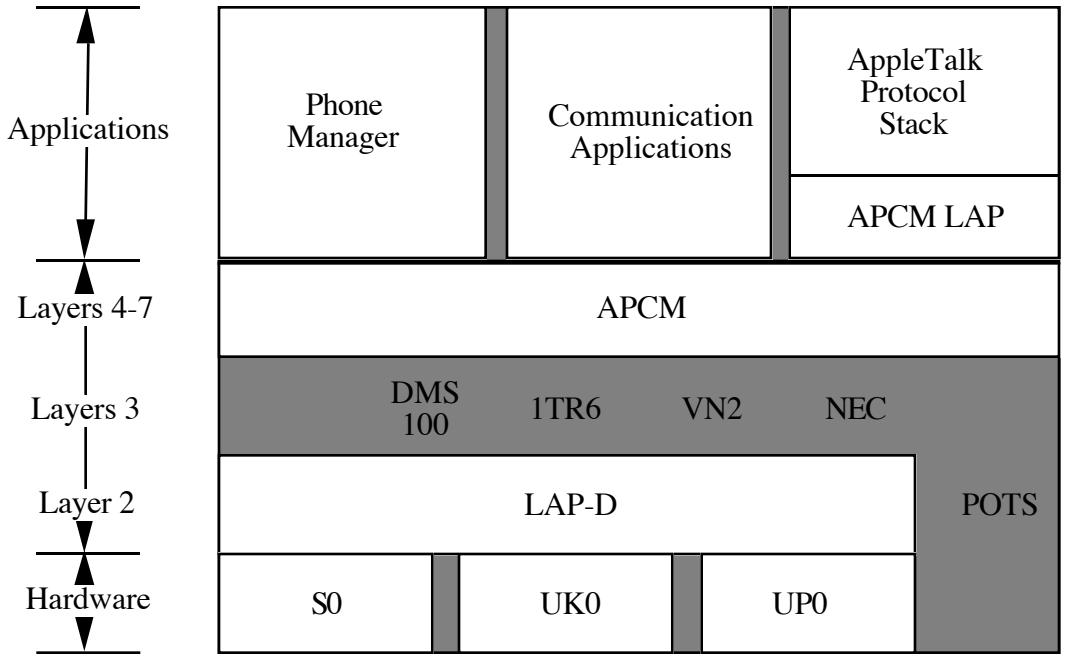

Abbildung 4.1: Die Einbettung des APCM in die Schichten der ISDN-Kommunikation

Die Anforderungen eines ISDN Link Access Protocols an den Verbindungsaufbau und die Datenübertragung werden bereits von einer Untermenge der vom APCM angebotenen Funktionen erfüllt, so daß sich der APCM als Implementierungsplattform anbietet.

4.1 Arbeitsweise des Abstract Personal Communications Manager

Die für das APCMLAP verwendete Implementierung des APCM wurde als Macintosh-Treiber realisiert. Dieser Treiber befindet sich im Systemordner des Macintosh und wird entsprechend der Informationen im ROM der ISDN-Nubuskarte beim Systemstart automatisch in den System Heap geladen und ausgeführt.

Der vorliegende APCM für den Apple Macintosh arbeitet vollständig interruptgetrieben und kann Daten sowohl synchron als auch asynchron übertragen. Der synchrone Aufruf kehrt erst dann zum Klienten zurück, wenn die Daten übertragen wurden. Asynchrone Aufrufe kehren sofort zurück, noch bevor die Daten wirklich gesendet sind. Das Ende der Übertragung wird dem Klienten durch die Ausführung einer Completion-Routine, deren Adresse dem APCM beim Aufruf mit übergeben wurde, angezeigt (Upcall).

---

3 Siehe FROITZHEIM (The Abstract Personal Communications Manager)

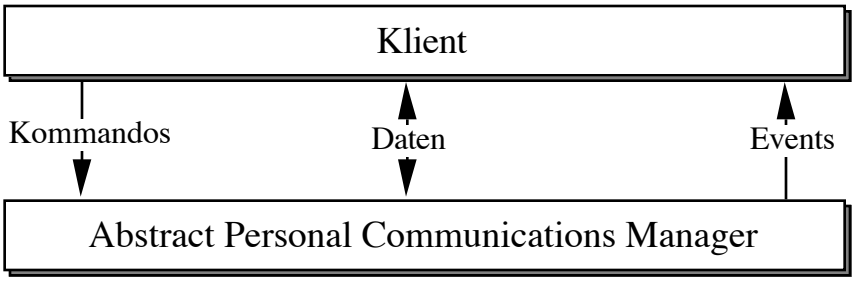

Abbildung 4.2: Wechselwirkungen zwischen APCM und Anwendungsprogramm

Ähnlich wie bei der asynchronen Datenübertragung wird beim Verbindungsaufbau dem Klienten des APCM der Fortschritt über eine Interruptserviceroutine (Notify) gemeldet. Dabei wird als Parameter eine Nachricht übergeben, die den Grund des Aufrufs beschreibt. Diese Nachricht dekodiert die Notify-Routine und aktiviert die entsprechende Behandlungsroutine .

Ein Beispiel für den aktiven Verbindungsaufbau über den APCM ist inder nachfolgenden Tabelle gegeben.

1. Open	*Der Klient registriert sich beim APCM.*
2. Connect	*Verbindungsanforderung mit einer ISDN-Nummer.*
3. Notify	*wird vom APCM gerufen, sobald die Verbindung besteht.*
4. Read/Write	*wird wiederholt zur Datenübertragung gerufen.*
5. Disconnect	*schließt eine Verbindung*
6. Close	*Der Klient meldet sich beim APCM ab.*

Der passive Auf- und Abbau einer Verbindung (ankommender Ruf bzw. Verbindungsauflösung) wird der Applikation wiederum über die Notify-Prozedur mitgeteilt, die daraufhin die notwendigen Aktionen initiiert.

4.2 Datentypen und Funktionen des APCM

Der APCM stellt dem Anwendungsprogrammierer für den Verbindungsaufbau und die Datenübertragung sowohl Datenstrukturen als auch Routinen zur Verfügung. Die Dienstelemente zur Übertragung von Sprache werden hier nicht vorgestellt, da sie für die Implementierung des APCMLAP bedeutungslos sind.

4.2.1 Datentypen des APCM

Um Verbindungen zu identifizieren, verwendet der APCM **Connection References**. Die Art eines ankommenden oder abgehenden Anrufes wird durch **Connection Services** beschrieben. Über **APCMEvents** wird der Klient über Zustandsänderungen des Telefons oder der Verbindung informiert. Fehler werden der Applikation durch die **Causes**-Struktur mitgeteilt. **Callstate** beschreibt den Status der aktuellen Verbindung. Die zu wählende ISDN-Nummer erwartet der APCM in einer eigenen verzeigerten Datenstruktur **ISDNAdr**, die auch mehrere Rufnummern enthalten kann.

4.2.2 Funktionen für das Verbindungsmanagement

Für die Verbindungssteuerung im APCMLAP werden lediglich die Funktionen **Connect** und **Disconnect** benötigt. Die größte Rolle spielt dabei die Funktion **Connect**, die den APCM veranlaßt, eine Verbindung des spezifizierten Typs herzustellen und den Klienten über die Notify-Prozedur zu informieren. Die Art der angeforderten Verbindung wird dem APCM über den Service Parameter mitgeteilt.

4.2.3 Funktionen für die Datenübertragung

Als Funktionen für die Datenübertragung stellt der APCM die Routinen **`APCMRead`** und **`APCMWrite`** zur Verfügung. Sowohl das Lesen als auch das Schreiben kann synchron oder asynchron geschehen. Für den asynchronen Fall muß beim Aufruf jeweils eine Completion-Routine angegeben werden, um das Ende der Ausführung des Befehls anzuzeigen. Außerdem wird als Parameter ein Zeiger auf den Empfangs- bzw. Sende-Puffer übergeben. Der APCM verwaltet asynchrone Read- und Write-Requests in einer Warteschlange.

4.2.4 Lokale Funktionen für die Verwaltung von Verbindungen

Zur Anmeldung eines Klienten beim APCM dient die Funktion **`APCMOpen`**. Als Parameter wird ein Service Array mit den vom Klienten unterstützten Diensten und die Adresse der Notify-Routine des Klienten übergeben. **`APCMClose`** meldet den Klienten beim APCM wieder ab.

Weitere Prozeduren werden vom APCM angeboten, sind jedoch für die Implementierung des APCMLAP nicht von Bedeutung. Unter anderem findet man im APCM-Interface die Prozeduren `NewISDNAdr` und `DisposeISDNAdr`. Diese können von APCMLAP jedoch nicht verwendet werden, da sie unter Umständen Speicher bewegen[4].

4.3 Verwendbarkeit als Basis für das ISDN-LAN

Bei der Verwendung des APCM als Basis für ein Link Access Protocol haben sich folgende Vorteile und Nachteile herausgestellt:

+ Der APCM bietet eine Entkopplung von der ISDN-Hardware.
+ Unabhängigkeit vom verwendeten Signalisierungsprotokoll.
+ Komfortables Application Programmers Interface (API) zum Verbindungsaufbau und zur Datenübertragung.
- Implizite Zeitbeschränkungen durch die Puffergröße der Kommunikationshardware können bewirken, daß im Interrupt Datenpakete während der Übertragung abgebrochen werden und somit verlorengehen.
- Die fehlende Möglichkeit, Pakete synchron ohne Interrupt-Unterstützung zu senden, erzwingt eine Zwischenpufferung von Datenpaketen[5].
- Da auch der Verbindungsaufbau interruptgetrieben ist, kann während eines Interrupts die Verbindung nicht problemlos gewechselt werden.

Dabei ist zu bemerken, daß die Nachteile zum größten Teil durch die (noch nicht vollständige) Macintosh-Implementierung des APCM verursacht werden und keine konzeptuellen Schwächen sind.

Der APCM hat sich für die Implementierung eines AppleTalk Link Access Protocols als nützlich erwiesen. Störend wirkt beim APCM für den Apple Macintosh jedoch die implizite Abhängigkeit von Interrupts. Der Verzicht auf das interruptgetriebene Senden von Datenpaketen im APCM könnte die Implementierung eines LAP für AppleTalk wesentlich vereinfachen.

---

4 Das APCMLAP darf im allgemeinen kein Memory Management direkt oder indirekt verwenden, da andernfalls nicht festgeschraubte Speicherblöcke der Applikationen verschoben werden könnten.

5 Vgl. Abschnitt 7.6.3

5. Die Verbindungssteuerung[6] des APCMLAP

Der folgende Abschnitt beschäftigt sich mit der Wahl einer adäquaten Verbindungssteuerung für die verbindungslosen AppleTalk-Protokolle. Die Verbindungssteuerung des APCMLAP hat zwei Ansprüche zu erfüllen:

- Der **Protokolloverhead**, der durch die Diskrepanz zwischen der Zeit für einen Verbindungsaufbau und der Übertragung eines Paketes entsteht, muß minimiert werden.
- Eine **Monopolisierung** von Netzwerkressourcen ist auszuschließen, so daß mehrere konkurrente Transaktionen von und zu einem Knoten unterstützt werden können.

5.1 Diverse Strategien zur Verbindungssteuerung

Anders als bei LANs oder paketvermittelnden Netzwerken genügt es bei leitungsvermittelnden Systemen zu Beginn einer Übertragung eine Verbindung herzustellen und diese am Ende wieder abzubauen. Durch eine permanente Aufrechterhaltung von Verbindungen entsteht über die Leitung aber eine **Monopolisierung** einer Netzwerk-Ressource (z.B. eines Dateiservers). Andere Teilnehmer sind für die Dauer der Verbindung vom Zugriff auf diese Ressource ausgeschlossen. Eine akzeptable **Strategie zur Verbindungssteuerung** muß daher klären, **wann und für wie lange** eine Verbindung hergestellt wird. Einfluß auf die Wahl dieser Strategie haben folgende Faktoren:

- **Die Kosten eines Verbindungsaufbaus**: Hierzu zählen sowohl die Dauer eines Verbindungsaufbaus als auch die Gebühren, die für einen Verbindungsaufbau anfallen.
- **Die Dauer einer Datenübermittlung**: Sind nur wenige, vereinzelte Pakete zu übertragen oder lange Dateiblöcke?
- **Das Format der Daten**: Insbesondere, ob in den Daten die Ziel- und die Quelladresse enthalten ist oder nicht.
- **Die Frequenz des Zugriffes auf eine Netzwerk-Ressource**: Die Häufigkeit des Zugriffs auf ein Gerät beeinflußt die Anzahl der notwendigen Verbindungen.
- **Die Anzahl der um eine Ressource konkurrierenden Stationen**: Je weniger Stationen auf eine Ressource zugreifen, desto länger können Verbindungen bestehen bleiben.

Unter Berücksichtigung dieser Faktoren und des sequentiellen Ablaufs einer Sitzung haben sich folgende Strategien zur Verbindungssteuerung bewährt (vgl. Abbildung 5.1):

- **Die halbautomatische Verbindungssteuerung** erlaubt dem Benutzer über ein Hilfsprogramm einen Teilnehmer auszuwählen, eine Verbindung zu installieren und wieder abzubrechen.
- **Die sessionorientierte Verbindungsteuerung**[7] hält eine Verbindung für die Dauer einer Session aufrecht. Da eine Sitzung unter Umständen sehr lange dauern kann, entsteht so eine starke Monopolisierung von Netzwerk-Ressourcen. Zum Beispiel zu einem Fileserver im LAN können mehrere Workstations eine Session unterhalten, auch wenn sie zur Zeit nicht auf Dateien zugreifen.
- **Die transaktionsorientierte Verbindungsteuerung** baut jedesmal, wenn ein Request an eine Station gerichtet wird, eine Verbindung auf und wieder ab, wenn die erwartete Antwort erhalten wurde. Diese Strategie wird als **Fast Disconnect** bezeichnet. Oft ist es jedoch schwierig einen Ansatzpunkt für eine Transaktion zu finden. Ferner können an einem Server mehrere Transaktionen gleichzeitig ablaufen.

---

6 Eine ausführliche Diskussion der verschiedenen Strategien zur Verbindungssteuerung auf der Link-Ebene findet man in [LUPPER, 90].

7 FROITZHEIM (LAN-Funktionen), S. 46–47.

- **Die objekt-orientierte Verbindungssteuerung** richtet für jedes zu übertragende Datenobjekt eine neue Verbindung ein. Als Objekte können Dateien, Resourcen, Sektoren oder im Extremfall einzelne Pakete betrachtet werden. Diese Strategie gewährleistet eine optimale Ausnutzung der Schnittstellen durch **mehrere** Geräte. Da jedoch unter Umständen die□ Dauer eines Verbindungsaufbaus[8] im Verhältnis zur Übertragungsdauer eines Datenobjektes sehr lange sein kann, mindert diese Strategie den Datendurchsatz einer Schnittstelle.

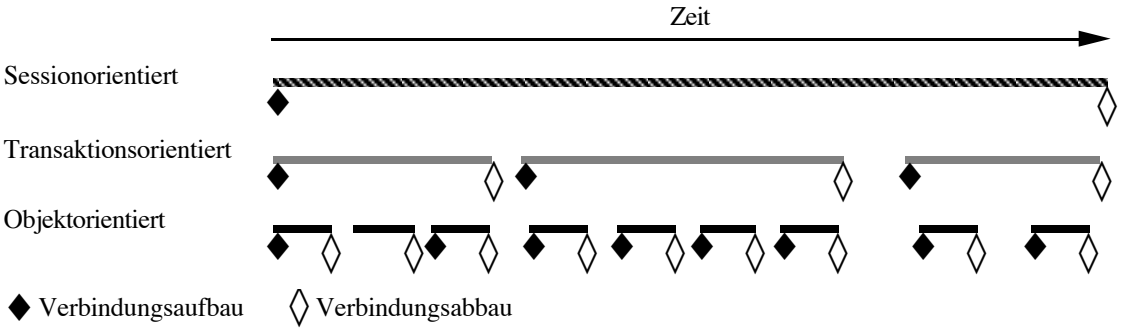

Abbildung 5.1: Verbindungssteuerungsstrategien im Vergleich

Die adäquate Verbindungsstrategie für ein Rechnernetz muß, entsprechend den Rahmenbedingungen, im Einzelfall gewählt werden. Da Parameter, wie Zugriffsfrequenz und Dauer der Datenübermittlung, erfahrungsgemäß stark variieren, ist keine dieser Strategien unter allen Einsatzbedingungen optimal. Die Kombination und **Abwandlung** einzelner Strategien kann zur Verbesserung der Schnittstellenausnutzung durch mehrere Geräte und zur Steigerung des Durchsatzes führen.

5.2 Die retardierende objekt-orientierte Verbindungssteuerung

APCMLAP verwendet eine retardierende objekt-orientierte Verbindungssteuerung, die Verbindungen nach der Übertragung eines Paketes noch für kurze Zeit bestehen läßt. Abgebaut wird erst dann, wenn während dieser Zeit kein weiteres Paket an dieselbe Zieladresse übertragen oder über diese Leitung empfangen wurde. Zusätzlich kann eine Verbindung jederzeit durch die Gegenstation abgebrochen werden. Da es sehr wahrscheinlich ist, daß mehrere Pakete unmittelbar hintereinander an dieselbe Zieladresse zu übertragen sind, bewirkt diese Strategie eine erhebliche Reduzierung der notwendigen Verbindungen (vgl. hierzu Abbildung 5.2).

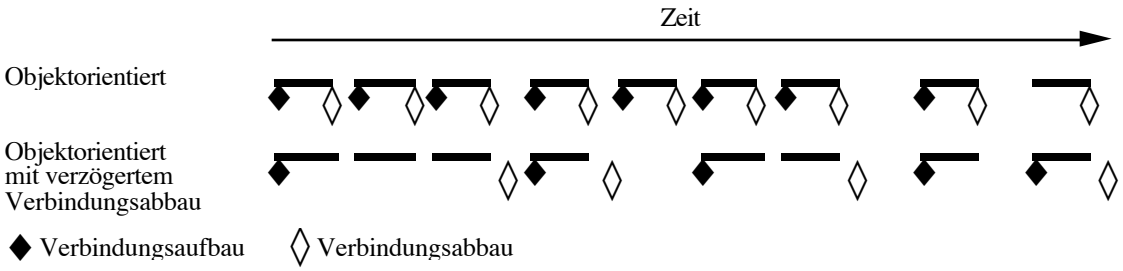

Abbildung 5.2: Passiver versus aktiver Verbindungsabbau

Werden während einer Wählverbindung mehrere Transaktionen mit derselben Gegenstation durchgeführt, so ist das Prinzip des retardierenden Verbindungsabbaus bezüglich der Anzahl der notwendigen Verbindungen besser als eine transaktionsorientierte Verbindungssteuerung. Bei mehreren logischen Verbindungen (Sitzungen) zu verschiedenen Gegenstationen wird jedoch ein ständiger Wechsel der Wählverbindungen notwendig. Im Detail arbeitet die Verbindungssteuerung wie im Flußdiagramm in Abbildung 5.3 dargestellt.

---

8 Ein Verbindungsaufbau kann im öffentlichen ISDN mehr als eine Sekunde dauern.

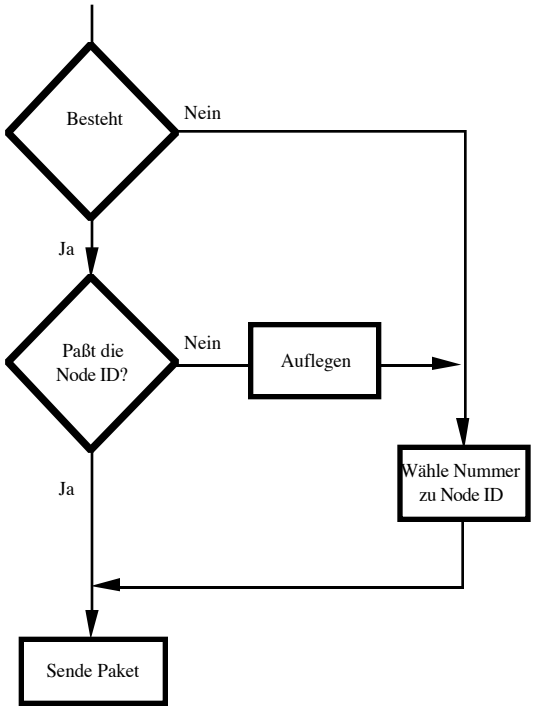

Abbildung 5.3: Schema einer Sendeanforderung

Nach dem aktiven oder passiven[9] Verbindungsaufbau und der Übertragung eines Paketes bleibt die Verbindung zunächst bestehen. **Aktiv abgebrochen** wird sie nur dann, wenn die Zieladresse des zur Übertragung anstehenden Paketes nicht mit der Knotenadresse der Gegenstation übereinstimmt. **Passiv abgebrochen** wird eine Verbindung zeitlich verzögert durch einen Timeout oder durch die Auslösung der Verbindung durch die Gegenstation. Dieser Timeout wird jedesmal, wenn ein Paket empfangen oder übertragen wurde, zurückgesetzt, so daß die Verbindung erst dann abbricht, wenn für eine bestimmte Zeit kein Verkehr auf der Leitung zu verzeichnen war. Dieses Verhalten ist vergleichbar mit dem Nachlaufen eines Diskettenlaufwerks.

Unter ungünstigen Bedingungen kann es auch hier zu einer Monopolisierung kommen. Die Länge des Timeouts kann von Station zu Station unterschiedlich sein. So wird ein Server schon nach sehr kurzer Zeit auflegen, um eine Monopolisierung des Zugriffs durch eine Station zu verhindern. Eine Verkürzung des Timeouts in Abhängigkeit von der Gesamtdauer einer Verbindung kann zur Verringerung der Monopolisierungsgefahr beitragen.

Um die Monopolisierungsgefahr, die bei der eben vorgestellten retardierenden objekt-orientierten Verbindungssteuerung unter ungünstigen Umständen immer noch vorhanden ist, weiter zu reduzieren, ist in ISDN-Systemen ein zusätzliches Leistungsmerkmal „Anklopfen“ nützlich[10]. Eine Verbindung zu einer Ressource im Netzwerk könnte dann solange bestehen bleiben, bis eine weitere Station auf diese zuzugreifen wünscht. Eine weitere Verfeinerung der Heuristiken zur Verbindungssteuerung unter Berücksichtigung der Gesamtdauer einer Verbindung und der Zeitspanne zwischen Verbindungsabbau und erneutem Verbindungsaufbau, ist durchaus denkbar muß jedoch in der Praxis noch erprobt werden.

Den Auf- und Abbau einer Verbindung regelt die retardierende objekt-orientierte Verbindungssteuerung.

---

9 d.h. die Station wird angerufen.

10 Dieses Leistungsmerkmal wird im öffentlichen ISDN bereits angeboten, jedoch zur Zeit von APCMLAP noch nicht berücksichtigt.

6. Naming und Adressierung in APCMLAP

Die Adressierung in AppleTalk basiert auf der **Broadcast-Fähigkeit des LocalTalk-Buses**. In einem leitungsvermittelnden System wie ISDN kann Broadcasting jedoch nur durch sukzessives Anwählen einzelner Stationen realisiert werden[11]. Die **mangelnde Broadcast-Fähigkeit** des ISDN impliziert daher die Anpassung der Algorithmen zur Adressierung im Netzwerk:

- Die dynamische Node-Adressierung ist derart zu modifizieren, daß neue Knoten weiterhin eine **eindeutige** Node ID erhalten.
- Zwischen den Node IDs und den Rufnummern der Knoten muß eine eindeutige Abbildung (**Mapping**) gewährleistet sein.
- Die Adressierungsprotokolle in AppleTalk (NBP, RTMP, ZIP) verwenden Broadcast-Pakete zur Verwaltung verteilter Adreßtabellen. Um die Funktionsfähigkeit dieser Protokolle zu gewährleisten, sind ihre Broadcast-Pakete gesondert zu behandeln.
- Allgemeine Broadcast-Pakete von Klienten sind im Netz zu verteilen.

APCMLAP löst das Adressierungsproblem, indem alle Adressierungs- und Naming-Informationen, die normalerweise über das Netz verteilt sind, in einem Name Server gesammelt werden. Dadurch wird die Information in einem leitungsvermittelnden System für eine einzelne Station leichter erreichbar. Alternative Ansätze zur Realisierung der Adressierung in ISDN-AppleTalk werden in [LUPPER, 90] ausführlich diskutiert.

6.1 Der Name Server zum APCMLAP

Da es in leitungsvermittelnden Netzwerken nur schwer möglich ist, Meldungen an alle Teilnehmer abzusetzen, wie es z.B. beim Name Binding oder der dynamischen Node-Adressierung geschieht, wird ein zentraler **Name Server** eingeführt. Dieser verwaltet ein eindeutiges Verzeichnis aller Node IDs mit zugehöriger Rufnummer und aller sichtbaren, benannten Netzwerk-Ressourcen.

Die grundlegende **Idee** besteht darin, alle Broadcast-Pakete (Zieladresse $FF) an den Name Server zu senden und dort zu behandeln. Dieser reagiert nur auf LAP-Pakete vom Typ $81 (ENQ) und $85 (FND) oder auf Name Binding-Pakete vom LAP-Typ $1 (kurze DDP-Pakete) und DDP-Typ $2 (NBP-Pakete). Broadcast-Pakete anderer Verwaltungsprotokolle, wie RTMP- und ZIP-Pakete, werden zur Zeit noch ignoriert. In einer späteren Version des Name Servers ist eine Behandlung der Interzonen-Verwaltungspakete im Name Server ähnlich den NBP-Paketen vorgesehen.

Aufgaben des Name Servers:

- **Registrieren der Node IDs mit zugehöriger Rufnummer:** Beim Öffnen des .MPP-Treibers wird die Node ID im Name Server auf Eindeutigkeit überprüft und zusammen mit der Rufnummer aus der Anschlußkennung registriert.
- **Auskunftgeben über Rufnummern zu Node IDs:** Über eine spezielle Anfrage kann beim Name Server die Rufnummer zu einer Node ID erfragt werden. Wird die Node ID nicht gefunden oder antwortet der Name Server nicht, so überträgt der Sender sein Paket an die Rufnummer des Name Servers.
- **Suchen und Registrieren von Names Table-Einträgen:** Namen werden vom NBP durch `LkUp`-Pakete beim Name Server gesucht und, falls der Eintrag nicht vorhanden ist (Eindeutigkeit), dort registriert. Existiert der gesuchte Eintrag bereits, so wird ein `LkUpReply`-Paket erwidert.

---

11 Die Möglichkeit zur Einrichtung von Mehrpunktverbindungen auf manchen TK-Anlagen soll hier nicht in Betracht gezogen werden.

- **Verwaltung und Löschen der Einträge im Name Server:** Das Names Directory und das Verzeichnis der Node IDs im Name Server muß gelegentlich auf seine Aktualität überprüft werden. Das ist notwendig, da ein Knoten das Löschen eines Names Table-Eintrags dem Netz nicht explizit durch ein Kontrollpaket anzeigt. Außerdem können Knoten ausfallen und sich nicht ordnungsgemäß abmelden.

  Die Überprüfung der Tabellen des Name Servers geschieht durch regelmäßiges Anwählen inaktiver Knoten durch den Name Server. Dabei wird anhand von Enquiry-Paketen die Gültigkeit der Node ID im Verzeichnis geprüft. Befinden sich im Names Directory des Name Servers Einträge eines Nodes, so sendet der Server zusätzlich `LkUp`-Pakete, um zu testen, ob die Einträge noch im lokalen Names Table des Knotens vorhanden sind. Werden Node ID oder Names Table-Eintrag nicht gefunden, so erfolgt die Löschung im Name Server.

Die dynamische Node-Adressierung mithilfe eines Name Servers erfordert die Modifikation der Adressierungsmechanismen des LAP. Zur Behandlung von Anfragen nach Rufnummern zu Node IDs beim Name Server werden neue Kontrollpakete eingeführt. Für das Name Binding über APCMLAP sind keine Änderung der bestehenden Protokolle notwendig. Auch das Format der NBP-Pakete kann unverändert bleiben, da es bereits alle Informationen enthält, die für das Name Binding über den Name Server notwendig sind.

Die Existenz eines Name Servers ist auch von der Verbindungssteuerung des APCMLAP zu berücksichtigen. Dabei muß es jedoch auch möglich sein, eine Station direkt, ohne den Umweg über den Name Server, anzuwählen.

6.2 Dynamische Node-Adressierung über den Name Server

Die dynamische Node-Adressierung in AppleTalk basiert auf der Broadcast-Fähigkeit der physikalischen Schicht (AppleTalk-Bus). Jeder Knoten im AppleTalk-Netzwerk erhält eine eindeutige Adresse (Node ID), die beim Öffnen des .MPP-Treibers vereinbart wird, indem spezielle Enquiry-Pakete (ENQ) an die selbstgewählte Node ID gesendet werden und von allen Stationen am Bus empfangen werden können. Fühlt sich ein Knoten angesprochen (d.h. er hat bereits diese Node ID) und antwortet auf dieses Paket mit einem ACK-Paket, so ist die Knotenadresse bereits vergeben und der neue Knoten startet einen erneuten Versuch mit einer anderen Node ID.

Da in einem leitungsvermittelnden, sternförmigen Netz keine ständige Verbindung zu allen Stationen besteht, kann der beschriebene Algorithmus nicht in dieser Form übernommen werden. Vielmehr muß eine flexible Methode gefunden werden, die eindeutige Zuordnung der Node IDs in AppleTalk zu den physikalischen Adressen in einem leitungsvermittelnden System, den Rufnummern, zu realisieren:

Beim **Öffnen des .MPP** wird eine Station unter einer Node ID registriert, indem ein **Enquiry Control Frame** (ENQ) mit der Zieladresse (destID) $255 (Broadcast-Paket) an den Name Server übertragen wird. Der Name Server hält zu jeder Node ID die Nummer der anrufenden Station fest, wodurch nach und nach ein Verzeichnis entsteht, das jeder Node ID im Netz eine Rufnummer zuordnet. Durch die Verwaltung dieser Liste kann sichergestellt werden, daß eine Node ID im Netzwerk eindeutig ist.

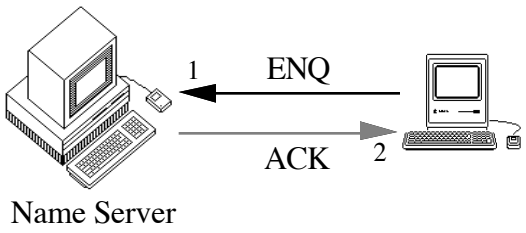

Abbildung 6.1: Registrierung einer Node ID beim Name Server

Der Name Server bestätigt die Eintragung durch einen **Acknowledge Control Frame** (ACK) (Abbildung 6.1). Wurde der Knoten mit Node ID und Rufnummer beim Name Server

registriert, so enthält das ACK-Paket die Node ID des Name Servers. Ist die Node ID im Name Server schon unter einer anderen Rufnummer vorhanden, so enthält das ACK-Paket die Node ID der anrufenden Station. Danach darf die Node ID nicht mehr geändert werden, ohne die Änderung dem Name Server mitzuteilen.

Steht ein LAP-Paket zur Übertragung an und ist die Rufnummer zur Zieladresse des Paketes lokal nicht bekannt, so sucht der sendende Knoten durch ein **Find Node ID-Paket (FND)**[12] an den Name Server nach der Rufnummer, die der Zieladresse (destID) durch die Tabelle im Name Server zugeordnet ist. Findet der Name Server die gesuchte Node ID in seiner Tabelle, so erwidert er ein **Return Number Found-Paket (RND)**[13] mit der zugehörigen Rufnummer (Abbildung 6.2). Wird die Nummer nicht gefunden, erhält der Initiator kein Antwortpaket und das LAP-Paket wird direkt an den Name Server übertragen. Wurde die gesuchte Nummer vom Name Server geliefert, so wird die Verbindung zum Name Server unterbrochen, die ermittelte Nummer gewählt und das LAP-Paket an seine Bestimmungsadresse übertragen.

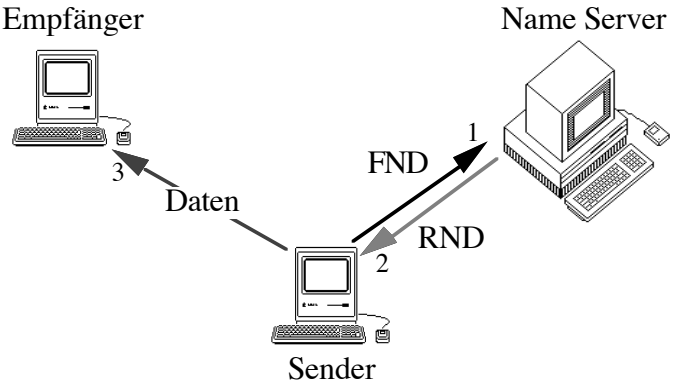

Abbildung 6.2: Suche nach einer Rufnummer beim Name Server

Zusätzlich wird bei jedem Verbindungsaufbau die Eindeutigkeit der Node ID durch den ENQ-ACK-Handshake noch einmal überprüft. Das dient u.a. dazu, die Node ID des anrufenden Gerätes festzuhalten. Dadurch kann der angerufene Knoten erkennen, daß er über eine bestehende Verbindung antworten kann. Außerdem wird die dynamische Node-Adressierung zum Testen der Verfügbarkeit der Datenverbindung verwendet, denn selbst wenn die Vermittlung den Verbindungsaufbau bestätigt hat, kann es noch einige Millisekunden dauern, bis die Datenverbindung wirklich durchgeschaltet ist.

Wird der .MPP-Treiber geschlossen, so muß die Node ID eines Gerätes wieder aus der Liste im Name Server entfernt werden. Das geschieht, indem der Name Server in regelmäßigen Abständen überprüft, ob alle Knoten in seiner Liste, die sich eine bestimmte Zeit nicht mehr gemeldet haben, noch aktiv sind. Ist ein Knoten nicht mehr erreichbar, so wird sein Node ID-Eintrag aus der Liste entfernt.

Das APCMLAP speichert die zuletzt gewählte Rufnummer. Dies kann eine erhebliche Reduzierung der notwendigen Verbindungen zum Name Server bewirken, da sich der Verkehr in einem lokalen Netzwerk typischerweise auf ein oder zwei zentrale Stationen, wie Fileserver oder Drucker, konzentriert. Außerdem ist der zusätzliche Verwaltungsaufwand gering.

6.3 Name Binding-Funktionen über den Name Server

Da Namen für den Anwender besser handhabbar sind als Adressen, beinhaltet AppleTalk ein Name Binding-Protokoll (NBP), um Namen an Socket-Adressen zu binden, auf dem Netzwerk zu suchen und schließlich wieder zu löschen. Das NBP verwendet allerdings Broadcast-Pakete für die Registrierung und Suche von Einträgen im Netzwerk. Somit muß eine adäquate Methode gefunden werden, das Name Binding ohne Broadcasting in einem leitungsvermitteln-

12 Dieser Pakettyp ist kein AppleTalk-Standard LAP-Typ.
13 Dieser Pakettyp ist kein AppleTalk-Standard LAP-Typ.

den System zu realisieren. Eine Strategie für das Name Binding im ISDN muß darauf ausgelegt sein, möglichst keine Veränderungen am bestehenden NBP vorzunehmen.

Der Name Server hält ein zentrales **Names Directory**, das die lokalen Einträge eines jeden Node (Names Table) enthält. Beim Name LookUp über APCMLAP werden die LkUp-Pakete als Broadcast-Pakete direkt zum Name Server geleitet. Daher ist es bei einem Name LookUp nicht mehr notwendig, sämtliche Teilnehmer zu konsultieren. Ist der Eintrag eines LkUp-Paketes noch nicht vorhanden, so übernimmt der Name Server den Inhalt dieses LkUp-Paketes in sein Names Directory. Bei der Registrierung und bei der Suche nach Namen verwendet das NBP jeweils LkUp-Pakete. Anhand des Pakettyps allein kann also nicht entschieden werden, ob es sich um eine Registrierung oder um eine Suche handelt. Beim Name LookUp enthalten LkUp-Pakete jedoch meist sogenannte Wildcards[14] in Namen und können so von Paketen zur Registrierung unterschieden werden. Wildcards in LkUp-Paketen werden daher durch den Name Server aufgelöst und nicht in das Names Directory aufgenommen.

Der Name Server prüft regelmäßig, ob ein Eintrag in der Names Table eines Node noch vorhanden ist. Wurde irrtümlich der Inhalt eines LkUp-Paketes beim Name LookUp in das Names Directory aufgenommen, so wird der Eintrag in der Names Table des entsprechenden Node nicht vorgefunden und daher aus dem Names Directory gelöscht. Andernfalls bleibt er auch nach der Überprüfung in der Tabelle erhalten.

Durch diese Validierung wird auch verhindert, daß Einträge von Knoten, die abgeschaltet wurden, ohne sich beim Name Server ordnungsgemäß abzumelden, im Names Directory und der Liste der Node IDs verbleiben. Dies würde bei Klienten des Name Servers zu unnötigen Wartezeiten führen, da sie nach erfolgreicher Suche erst bei der direkten Anwahl des inaktiven Knotens erkennen könnten, daß dieser nicht mehr bereit ist. Nicht zuletzt belasten „Leichen“ im Names Directory die Suche und Registrierung.

Die Behandlung der einzelnen Funktionsaufrufe des NBP durch das APCMLAP und den Name Server unterscheidet sich im wesentlichen nicht von der in [LUPPER, 90] beschriebenen.

6.4 Behandlung von Interzonen-Protokollen

Grundsätzlich steht es jedem Anwenderprogramm als Klient des LAP oder DDP frei, Broadcast-Pakete zu senden. Derartige Broadcast-Pakete sind äußerst schwierig zu handhaben. Sie müßten durch sukzessives Anwählen an alle Stationen verteilt werden, was aber einen erheblichen Aufwand an Zeit und Verbindungen erfordert. Dadurch können die vom Sender gewünschten Antwortzeiten meist nicht eingehalten werden. Alternative Ansätze hierzu sind durchaus denkbar, müssen aber nicht weiter diskutiert werden, da Broadcast-Pakete, die nicht vom LAP, NBP, ZIP oder RTMP ausgehen, für gewöhnlich in AppleTalk nicht auftreten.

Wie die Names Table des NBP lassen sich auch Weglenkungstabellen und Zone Information Tables im zentralen Name Server verwalten. Das RTMP verteilt sowohl bei der Registrierung als auch bei der Suche nach Brücken Broadcast-Pakete (destID $255) an alle Stationen im Netz. Das ZIP sucht mit Query-Paketen nach der Netzwerknummer zu einem Zone nnamen. Wie die LkUp-Pakete des NBP werden diese Broadcast-Pakete an den Name Server geleitet und dort bearbeitet. Neue Routing Tuples in RTMP-Request-Paketen werden in eine Tabelle aufgenommen, gefundene Tuples werden als Response-Pakete erwidert. Ähnlich wird eine Zone Information Table (ZIT) im Name Server erstellt.

Die Funktionalität der Adressierung des AppleTalk Protocol Stack wird im APCMLAP durch einen Name Server gewährleistet. In einer späteren Version ist zudem die Behandlung der Interzonen-Verwaltungspakete im Name Server vorgesehen.

---

[14] Zeichen, ähnlich dem *.* in MSDOS.

7. Implementierung des APCMLAP

Der folgende Abschnitt beschreibt die Implementierung eines ISDN Link Access Protocols für AppleTalk basierend auf dem APCM. Im Detail werden nur diejenigen Routinen behandelt, die sich wesentlich von den in [LUPPER, 90] beschriebenen unterscheiden.

7.1 Struktur des ADEV APCMLAP

Alle Komponenten des APCMLAP sind in einer System-Datei vom Typ ADEV zusammengefaßt. Dieses ADEV enthält die Programmteile des APCMLAP als einzelne Resourcen. Im wesentlichen lassen sich die Resourcen des APCMLAP in drei Gruppen unterteilen:

- Die ‘adev’-Resource, die die Wechselwirkungen mit dem CDEV Netzwerk bei der Auswahl des verwendeten Netzwerkes übernimmt.
- Die ‘atlk’-Resource, die den eigentlichen Code des APCMLAP enthält.
- Einige weitere Resourcen, die für den Dialog mit dem Benutzer und dem Betriebssystem notwendig sind.

7.1.1 Die ‘adev’-Resource im ADEV APCMLAP

Die ‘adev’-Resource enthält den Code der Prozeduren, die bei der Auswahl des LAP (also des verwendeten Netzwerkes) im Kontrollfeld gerufen werden. Der Code der ‘adev’-Resource wurde aufgrund seiner Systemnähe vollständig in Assembler implementiert und besteht im wesentlichen aus einer Sprungtabelle, den Prozeduren **`GetADEV`** und **`SelectADEV`** sowie der Zeichenkette ‘APCMLAP’, die unter dem APCMLAP-Icon im Kontrollfeld erscheint.

7.1.2 Die ‘atlk’-Resource im ADEV APCMLAP

Der eigentliche Code des APCMLAP ist in der ‘atlk’-Resource des ADEV enthalten. Er besteht aus einer Sprungtabelle, den Installationsprozeduren für den `LAPWrite`-Code, den in Maschinensprache geschriebenen Interfaces (Glue) zum Aufruf der Sende- und Empfangsprozeduren in Pascal und dem eigentlichen APCMLAP-Code in Pascal.

Die Assembler-Prozeduren **`doAInstall`** und **`doAShutDown`** dienen dabei der Installation. Der Glue für die Pascal-Routinen besteht aus den Prozeduren **`LAPWrite`**, **`ReadDispatch`**, **`ReadPacket`** und **`ReadRest`**. Am Ende des Assembler-Codes der ‘atlk’-Resource wird ein Datenblock angelegt, der die globalen Variablen des Pascal-Moduls aufnimmt. Der Pascal-Code des APCMLAP, der eine Vielzahl von Prozeduren enthält, wird zu diesen Assembler-Routinen hinzugebunden.

Da der Code der ‘atlk’-Resource über mehrere Applikationen hinweg aktiv ist, muß er im System Heap untergebracht und „festgeschraubt“ werden. Er darf bei Speicherverwaltungsoperationen nicht verschoben werden.

7.1.3 Weitere Resourcen des ADEV APCMLAP

Neben den Code-Resourcen befinden sich noch weitere Resourcen in der Resource Fork des ADEV. Die BNDL-, FREF- und ICN#-Resourcen dienen dem Anzeigen des APCMLAP-Icons auf dem Desktop. Dieses ist identisch mit dem Icon im Kontrollfeld.

In einer ‘STR ’-Resource ist die Rufnummer des Name Servers abgelegt. Sie wird bei der Auswahl des APCMLAP vom Benutzer über eine Dialogbox, die sich in der ‘DLOG’- und ‘DITL’-Resource befindet, gesetzt.

7.2 Funktionale Aufteilung des APCMLAP in Prozeduren

Die Prozeduren des APCMLAP lassen sich nach ihren Aufgaben grob in fünf Bereiche gliedern (Abbildung 7.1): Die Auswahlprozeduren für das Kontrollfeld, die Glueroutinen zu AppleTalk, die Initialisierungs- und Installationsprozeduren des APCMLAP, die Verbindungssteuerung und die Paketübertragung.

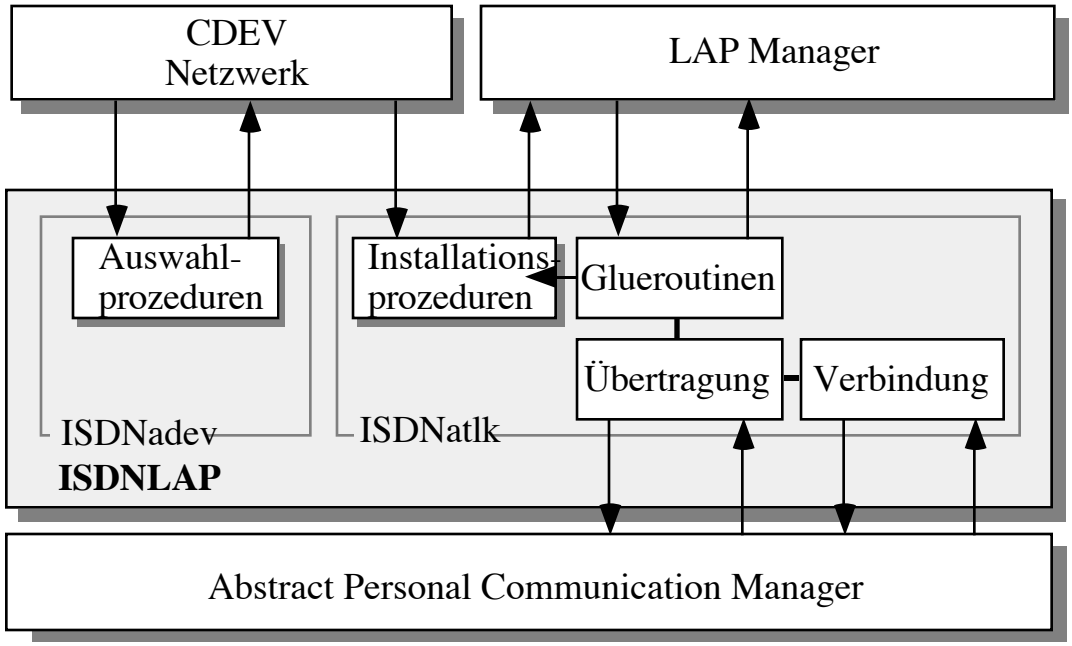

Abbildung 7.1: Das Zusammenwirken der Prozeduren des APCMLAP

- Die Auswahlprozeduren im ‘adev’-Code sind vollständig in Assembler geschrieben und übernehmen beim Anzeigen und bei der Auswahl des LAP den Dialog mit dem CDEV Netzwerk und dem Benutzer.
- Die Installations- und Initialisierungsprozeduren in ‘atlk’-Code übergeben bei der Auswahl die Adresse des `LAPWrite`-Codes an den LAP Manager und initialisieren das APCMLAP.
- Die Assembler-Glueroutinen bilden das Interface zwischen den Pascal-Routinen des APCMLAP und den Routinen des LAP Managers und des Betriebssystems, die ihre Parameter in Registern übergeben (register based).
- Die Pascal-Prozeduren der Verbindungssteuerung übernehmen den Verbindungsauf- und -abbau, die dynamische Node-Adressierung sowie die Suche nach Rufnummern beim Name Server.
- Die Pascal-Prozeduren zur Paketübertragung sind für das Senden und Empfangen von Paketen verantwortlich.
- Verschiedene kleinere Hilfsprozeduren in Pascal unterstützen die Übertragungs- und Verbindungssteuerungsprozeduren.

7.3 Die Aufteilung in Units und das Binden der Moduln

Die Prozeduren des APCMLAP sind in drei Units programmiert. Dabei handelt es sich um zwei Assembler- und ein Pascal-Modul. Im einzelnen besteht der Quellcode des APCMLAP aus folgenden Dateien:

APCMLAP.p	Hier befindet sich der eigentliche LAP-Code mit den Prozeduren zur Verbindungssteuerung und zur Paketübertragung.
APCMatlk.a	integriert die Routinen zur Installation und Deinstallation des `LAPWrite`-Codes im `ATalkHk2` und einige Glueroutinen zum Aufruf der Pascal-Prozeduren aus APCMLAP.p.
APCMadev.a	besitzt zwei Routinen zur Auswahl des APCMLAP im Kontrollfeld.
APCMLAP.r	enthält die übrigen Resourcen des ADEV APCMLAP.
LAPMgrEqu.a	stellt Konstanten zur Verfügung, die von der APCMadev.a- und APCMatlk.a-Unit benötigt werden.
ISDNGlue.p	die Pascal-Interfaceroutinen für die Verwendung des APCM-Treibers
ISDNTypes.p	die Datenstrukturen des APCM-Interfaces
APCMLAP.make	übersetzt die Quellen des APCMLAP und baut die Resourcen des ADEV zusammen.

Die Assembler- und Pascal-Moduln verwenden Toolbox- und OS-Routinen, die jeweils importiert und vom Linker hinzugebunden werden müssen. Um zu verhindern, daß Code zur Bereichsüberprüfung eingebunden wird, muß mit der Compiler-Option {$R-} das Range Checking des Compilers deaktiviert werden. Damit auch die importierten Moduln mit dieser Option übersetzt werden, muß das vor USES geschehen:

```
{$R+}
USES Types, Memory, Resources, OSEvents, OSIntf, Dialogs, Retrace,
         Timer, ToolUtils, Packages, AppleTalk, ISDNTypes, ISDNGlue;
```

Die einzelnen Teile des APCMLAP werden durch den Linker schließlich zu den ‘adev’- und ‘atlk’-Resourcen zusammengebunden. Zur Erzeugung der ‘adev’-Resource wird die ‘adev’-Assemblerunit mit der `SetNumber`-Prozedur aus APCMLAP.p kombiniert. Aus den restlichen Pascal-Prozeduren in APCMLAP.p und dem ‘atlk’-Assemblermodul entsteht durch Linken die ‘atlk’-Resource. Schließlich erzeugt der Resource Compiler (Rez) aus den einzelnen Code-Resourcen das ADEV APCMLAP und das MPW Tool Setfile setzt das Bundlebit, den Typ und den Creator der erzeugten AppleTalk-Erweiterung.

```
APCMLAP ƒƒ APCMLAP.r APCMadev APCMatlk
   Rez   APCMLAP.r -o APCMLAP
   Setfile -a B -t 'adev' -c 'ISDN' APCMLAP
   duplicate -y APCMLAP "{SystemFolder}"APCMLAP
```

Zuletzt wird mit dem Kommando „duplicate“ eine Kopie des APCMLAP im System Folder abgelegt. Somit ist es sofort für das CDEV Netzwerk verfügbar.

7.4 Konstanten, Datenstrukturen und globale Variablen

7.4.1 Konstanten

Einige wenige Konstanten des APCMLAP, wie die Länge von Timeouts und Pakettypen, werden **im Quelltext** (hard coded) untergebracht. Das spart vor allem globale Variablen und/oder Zeit für das Lesen von Resourcen aus der Resource Fork. Dadurch können jedoch Konstanten nur durch Änderung im Quelltext modifiziert werden. Konstanten, die unter Umständen gelegentlich angepaßt werden müssen, sind als **Resourcen** angelegt. Zusätzlich werden verschiedene Konstanten aus den Toolbox- und OS-Interfacedateien verwendet.

7.4.2 Datenstrukturen und Typen

Strukturierte Datentypen in Pascal erlauben einen höheren Abstraktionsgrad als die Assembler-Ebene und erleichtern damit den Zugriff auf Daten. Die Objekte der Datenkommunikation lassen sich durch Pascal-Typen strukturieren und gewinnen somit gegenüber einem unstrukturierten Bitstrom an Anschaulichkeit .

Die im APCMLAP verwendeten Datentypen unterscheiden sich nur unwesentlich von den in [LUPPER, 90] beschriebenen und werden daher nicht weiter erläutert.

7.4.3 Globale Variablen

Temporärer Speicher wird in Pascal-Programmen auf dem Stack angelegt und nach der Beendigung der Prozedur wieder freigegeben. Globale Variablen, die für die Dauer der Aktivierung des APCMLAP benötigt werden, bereiten jedoch Schwierigkeiten. Da es sich bei APCMLAP um Treiber-Code ohne die übliche Applikationsumgebung handelt, können globale Variablen vom Pascal-Compiler nicht wie sonst relativ zum A5-Register adressiert werden.

Die globalen Variablen des APCMLAP sind daher als Datenblock im Assembler-Modul APCMatlk.a angelegt. Sie enthalten nur die Werte, die für die Dauer der Aktivierung des APCMLAP für mehrere Prozeduren relevant sind. Die Struktur der globalen Variablen wird durch den Typ **`GlobalVars`** festgelegt. Die folgende Tabelle listet nur die Felder des Records auf, die speziell für die LAP-Implementierung mit dem APCM von Bedeutung sind.

TerminateDelay	*LongInt*	Leerlaufzeit bis zum Abbruch einer Verbindung.
Name ServerInUse	*Boolean*	gibt an, ob ein Name Server verwendet wird.
actCR		Connection Reference der aktuellen Verbindung
target	*ISDNAdr*	ISDN-Adresse der Verbindung.
targetPtr	*ISDNAdrPtr*	Zeiger auf die ISDN-Adresse
targetHdl	*ISDNAdrHdl*	Handle auf die ISDN-Adresse
HeadofBuffer, TailofBuffer	*INTEGER*	Kopf- und Schwanzzeiger im Ringpuffer
RingBuffer	*ARRAY ..*	Der Ringpuffer zum Versenden von Paketen
ReadPos	*INTEGER*	Aktuelle Leseposition beim Einlesen eines Paketes
RXParams	*rxqel*	Receive Queue Element zum Aufsetzen eines APCMRead-Befehls.
ReceiveBuffer	*aDataField*	Puffer, für eingehende Pakete

7.5 Die Verbindungssteuerung über den APCM

Die Verbindungssteuerung des APCMLAP wird durch die Prozeduren `InitAPCMStuff`, `DoDialing`, `Dialup`, `Hangup`, `Notify`, `ReceiveCall`, `RemoteDisconnect`, `Confirm` und `Terminator` übernommen. Im folgenden wird der Aufbau dieser Prozeduren im Detail beschrieben, sofern sie nicht bereits in [LUPPER, 90] abgehandelt wurden.

7.5.1 Die Initialisierung des APCM für den Verbindungsaufbau

Wird der `LAPWrite`-Code des APCMLAP beim Öffnen des .MPP erstmals gerufen, um ein ENQ-Paket zu übertragen, so aktiviert er über die Prozedur **`InitAPCMStuff`** den APCM. Dazu ruft `InitAPCMStuff` den APCM mit `APCMOpen` und registriert dadurch das APCMLAP als Klient des APCM. Als Parameter wird die Adresse der `Notify`-Prozedur übergeben, die fortan Meldungen des APCM entgegennimmt. Gleichzeitig wird dem APCM der Typ der vom APCMLAP unterstützten Verbindung (`Data64`) mitgeteilt.

Vor dem ersten Senden eines Paketes, d.h. vor dem Öffnen des .MPP, darf `InitAPCMStuff` nicht gerufen werden, da andernfalls Anrufe und Pakete angenommen werden könnten, ohne daß die lokalen Va□riablen des .MPP (`ABusVars`) initialisiert sind.

7.5.2 Die Notify-Prozedur

Die bei der Anmeldung des APCMLAP als Klient des APCM übergebene `Notify`-Prozedur wird immer dann gerufen, wenn der APCM dem Klienten (APCMLAP) Informationen über den Zustand einer Verbindung übergeben will. `Notify` wertet die übergebenen Parameter aus und verzweigt anhand des `notetype`-Parameters zu den einzelnen Behandlungsroutinen, wie `ReceiveCall`, `RemoteDisconnect` oder `Confirm`. Diese werden mit den notwendigen Parametern gerufen und kehren nach der Behandlung wieder über `Notify` zum APCM zurück.

7.5.3 Der Verbindungsaufbau über den APCM

Der Verbindungsaufbau des APCMLAP erfolgt sowohl bei abgehenden als auch bei ankommenden Verbindungen ausschließlich über die Funktionen des APCM. Der Zustand der Verbindung kann jederzeit mit einer Statusabfrage an den APCM geprüft werden. Die verschiedenen Zustände einer Verbindung werden in APCMLAP durch den Typ `LineStatus` repräsentiert. Der aktuelle Status einer Verbindung wird in der globalen Variablen `IsUp` festgehalten.

Bei Verbindungsproblemen sind die **Timeouts** der ISDN-Protokolle[15] für eine Übertragung von AppleTalk-Paketen zu lang. Um dem Benutzer des APCMLAP im Fehlerfall lange Wartezeiten zu ersparen, müssen sie entsprechend verkürzt werden.

7.5.3.1 Der aktive Verbindungsaufbau

Das Wählen beim Verbindungsaufbau übernimmt die Prozedur **`DialUp`**. Als einziger Parameter wird der Prozedur die zu wählende Rufnummer übergeben. Über die globale Variable `IsUP` wird der Zustand der Verbindung vom Typ `LineStatus` zurückgeliefert.

```
FUNCTION DialUp(Number: ISDNNumber);
```

`DialUp` bricht zunächst eine eventuell bestehende Verbindung ab und setzt die Zustandsvariable der Verbindung `IsUP` auf `inProcess`, um anzuzeigen, daß sich eine Verbindung im Aufbau befindet. Anschließend wird ein ISDN-Nummernblock, der dem APCM als Handle übergeben wird, mit der zu wählenden ISDN-Nummer gefüllt. Außerdem wird in das Typfeld eines Service Records der Verbindungstyp `Data64` eingetragen und das `Link`-Feld desselben Records auf den Wert `packet` gesetzt. Die Connection Reference der aktuellen Verbindung wird mit -128 initialisiert.

---

[15] Es kann bis zu 1,5 Sekunden dauern, bis eine Verbindung geschaltet ist.

Die eigentliche Anforderung einer ISDN-Verbindung vom APCM geschieht durch den Funktionsaufruf `Connect`. Dabei wer den die ISDN-Nummer, die Connection Reference und das Service Record als Parameter übergeben. Als Funktionsergebnis liefert der Aufruf eine Fehlernummer zurück, die den Wert Null hat, falls der Aufbau noch nicht abgeschlossen ist.

Sobald die Verbindung endgültig aufgebaut ist, ruft der APCM die `Notify`-Prozedur, die dann anhand der übergebenen Meldung zur `Confirm`-Routine verzweigt. Diese initialisiert die Datenübertragung durch Aufruf der Prozedur `InitDataStuff`. Gleichzeitig wird die Zustandsvariable `IsUp` auf `confirmed` gesetzt und der Ringpuffer und der Timeout der Verbindung zurückgesetzt.

Wurde innerhalb einer bestimmten Zeitspanne vom APCM die Verbindung nicht bestätigt, so wird der Wählvorgang abgebrochen und die Zustandsvariable auf `notConnected` gesetzt. Ist die Leitung besetzt, so ruft der APCM über `Notify` die Funktion `RemoteDisconnect`, die der Zustandsvariable `IsUp` den Wert **`isbusy`** zuweist. Bei allen anderen Fehlercodes ist der Fehler nicht zu beheben und eine Wahlwiederholung macht keinen Sinn.

Es kann bis zu 1,5 Sekunden dauern, ehe eine ISDN-Verbindung durchgeschaltet ist und die Übertragung sich stabilisiert hat. Daher liefert `DialUp` als Funktionsergebnis höchstens `confirmed` und nicht `connected`. Im Anschluß daran muß bei der dynamischen Node-Adressierung mittels **`AcquireAddress`** durch die Übermittlung von Enquiry Control Frames geprüft werden, ob die Datenverbindung bereits durchgeschaltet ist. Wird innerhalb einer gewissen Zeit kein Acknowledge Control Frame empfangen, so bedeutet das, daß keine Datenverbindung besteht und die Wählverbindung wird durch Aufruf der Prozedur `HangUp` zurückgesetzt.

7.5.3.2 Der passive Verbindungsaufbau

Bei der Initialisierung des LAP wird durch `InitAPCMStuff` die Interruptserviceroutine `Notify` installiert, die Meldungen des APCM entgegennimmt. Erhält der APCM einen ankommenden Anruf, so aktiviert er die `Notify`-Prozedur mit dem APCMEvent `ESetupReq` als Parameter. `Notify` ruft hierauf die `ReceiveCall`-Prozedur des APCMLAP mit den für den Verbindungsaufbau notwendigen Parametern.

```
FUNCTION ReceiveCall(theCall: ConnRef; VAR calltype: service;
         VAR Facs: FacHdl; orgAdr: ISDNAdrHdl): InqRes;
```

`ReceiveCall` nimmt einen kommenden Ruf an, wenn AppleTalk initialisiert ist und keine andere Verbindung existiert oder sich in der Aufbauphase (`IsUp`) befindet. Der Zustand von AppleTalk wird mit der Funktion `IsMPPOpen` geprüft. Wenn AppleTalk bereit ist und noch keine Verbindung besteht oder geschaltet wird, dann signalisiert APCMLAP dem APCM die Annahme des Anrufs (call accept), indem an `ReceiveCall` als Funktionsergebnis der Wert `accept` zurückgegeben wird. Die Zustandsvariable der Verbindung (`IsUp`) nimmt nun den Wert `inProcess` an. Schließlich wird die Rufnummer der anrufenden Station der Anschlußkennung entnommen und in einem Zwischenspeicher (Cache) festgehalten.

Ist der MPP-Treiber nicht initialisiert oder hat APCMLAP eine aktive Verbindung, so ignoriert `ReceiveCall` den Wunsch zum Verbindungsaufbau und terminiert mit dem Funktionswert `Ignore`. Das Ergebnis von `ReceiveCall` wird bei der Rückkehr zu `Notify` in das Feld `action` des APCMEvents zurückgeschrieben, das anschließend vom APCM ausgewertet wird.

Wurde die Verbindung angenommen und durch den APCM ordnungsgemäß aufgebaut, so wird wie beim aktiven Verbindungsaufbau die `Confirm`-Routine vom APCM gerufen. Diese setzt den Zähler `LastTransmission` zurückgesetzt und initialisiert das APCMLAP und den APCM mit `InitDataStuff` für die Paketübertragung. Die Variable `IsUp` zeigt nun mit dem Wert `confirmed` an, daß eine Verbindung aufgebaut ist.

7.5.4 Der Verbindungsabbau

Nachdem wir gesehen haben, wie Verbindungen hergestellt werden, beschäftigen wir uns nun mit dem Abbau von Verbindungen. Ein Verbindungsabbau ist immer dann notwendig,

- wenn eine Verbindung zu einem Knoten besteht, dessen Node ID nicht mit der Zieladresse des nächsten zu sendenden Paketes übereinstimmt,
- wenn ein Fehler beim Verbindungsaufbau oder der Node-Adressierung auftritt,
- wenn eine bestimmte Zeit kein Paket mehr übertragen wurde,
- wenn die Gegenseite eine Verbindungsauflösung fordert oder
- wenn eine Verbindung unvorhergesehen unterbrochen wurde und der APCM wieder für einen erneuten Verbindungsaufbau initialisiert werden muß.

7.5.4.1 Der aktive Verbindungsabbau

Ein aktiver Verbindungsabbau wird dann durchgeführt, wenn beim Verbindungsaufbau ein Fehler aufgetreten ist oder eine bestehende Verbindung unterbrochen werden muß, da ein Paket an eine andere Station zu übertragen ist.

Zum Abbrechen einer Verbindung im APCMLAP dient die Prozedur **`HangUp`**. Sie ruft für den Abbau der Verbindung die APCM-Funktion `Disconnect`. Als Parameter wird die Connection Reference der Verbindung und ein Feld zur Angabe von Gründen für den Verbindungsaufbau mitgegeben. `HangUp` benutzt als Grund für den Abbau den Wert `normal`.

`HangUp` wird vom APCMLAP auch dann gerufen, wenn über eine bestehende Verbindung längere Zeit kein Paket mehr übertragen wurde. Das Testen und Abbrechen einer Verbindung durch einen Timeout übernimmt die Prozedur **`Terminator`**. Hierbei handelt es sich um eine VBL-Task, die bei der Installation des LAP durch `InitLAP` in die `VBLQueue` des Betriebssystems eingefügt wurde und vom Betriebssystem periodisch gerufen wird. Unmittelbar nach dem Aufruf besorgt sich `Terminator` die globalen Variablen des APCMLAP und erhöht den Zähler `LastTransmission`, der angibt, wieviele Taskaufrufe seit der letzten Paketübertragung stattgefunden haben. Überschreitet dieser Zähler den Wert **`TerminateDelay`**, und zeigt Variable `IsUp` das Bestehen einer Verbindung an, so löst die VBL-Task die Verbindung mit `HangUp` aus. Bevor die Task beendet wird, muß in `VBLEntry` der `vblCount` wieder auf den Startwert `TaskTime` gesetzt werden.

7.5.4.2 Der passive Verbindungsabbau

Ursache für den passiven Verbindungsabbau ist, daß die Gegenstation den Wunsch hat, die Verbindung zu beenden. Der passive Verbindungsabbau wird dem APCMLAP über die `Notify`-Prozedur und die Prozedur `RemoteDisconnect` angezeigt.

```
PROCEDURE RemoteDisconnect(theCall: ConnRef; Cause: SignedByte);
```

Der Grund für den Aufruf von `RemoteDisconnect` kann aber auch eine besetzte Leitung sein. Dies wird zunächst anhand von `Cause` geprüft und gegebenenfalls die Zustandsvariable `IsUp` auf `isbusy` gesetzt. Andernfalls, wenn die übergebene Connection Reference Number mit der eigenen übereinstimmt, erhält `IsUp` den Wert `notConnected`. Die Node ID der aktuellen Verbindung wird auf Null gesetzt.

7.5.5 Die Adressierung im APCMLAP

7.5.5.1 Dynamische Node-Adressierung und Registrierung der Node ID

Beim Öffnen des .MPP-Treibers muß das APCMLAP die Node ID des Knotens auf dem Netzwerk registrieren. Das geschieht durch Übertragung von ENQ-Paketen an den Name Server, der die Rufnummern und die Node IDs ankommender ENQ-Pakete registriert. Durch die Verbindungssteuerung des APCMLAP werden alle Broadcast-Pakete an den Name Server geleitet. Das RTMP sendet beim Öffnen des .MPP zwei Broadcast-Pakete ($FF) an das Netz (Abbildung 7.2). Dabei werden zum Testen der Verbindung auch ENQ-Pakete an den Name Server übertragen und die Eindeutigkeit geprüft. Ein explizites Anwählen des Name Servers durch das APCMLAP zur Registrierung der Node ID ist daher nicht notwendig. Zusätzlich werden bei jedem Verbindungsaufbau zum Name Servers für einen Name LookUp ENQ-Pakete übertragen.

```
FF 07 01 00 06 01 01 05 01 .........
```

Abbildung 7.2: Ein RTMP-Paket an den Name Server beim Öffnen des .MPP-Treibers

Nach dem Aufbau einer Verbindung wird zuerst eine Folge von ENQ-Paketen übertragen, um die Datenverbindung explizit zu testen. Damit kann gleichzeitig festgestellt werden, ob die eigene Node ID eindeutig ist und auf der Gegenseite die gewünschte Station angetroffen wurde. Ein nachträgliches Ändern der Node ID kommt nur dann vor, wenn sich eine Station nicht am Name Server angemeldet hat.

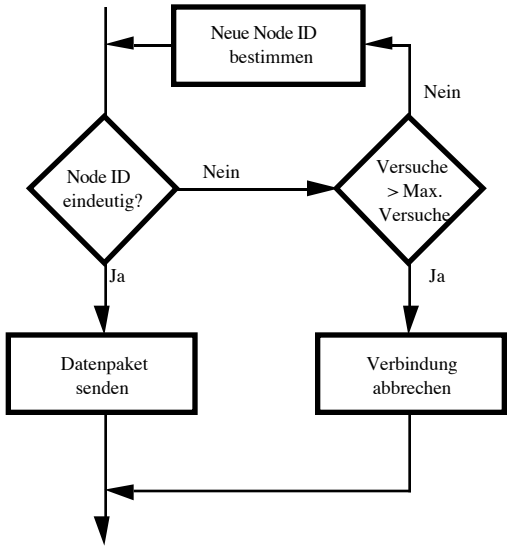

Abbildung 7.3: Dynamische Node-Adressierung im APCMLAP

Zur dynamischen Node-Adressierung und zum Testen der Verbindung wurde in APCMLAP die Pascalprozedur **`AcquireAddress`** implementiert. Beim Aufruf wird der Prozedur ein Parameter `Destination` übergeben, der die Zieladresse der ENQ-Pakete enthält. Bei der Rückkehr teilt die Funktion mit, ob eine Verbindung zur angewählten Station hergestellt werden konnte und ob die lokale Node ID gültig ist.

Die Funktion `AcquireAddress` des APCMLAP unterscheidet sich nicht von der in [LUPPER, 90] beschriebenen Implementierung.

7.5.5.2 Die Verbindungssteuerung des APCMLAP

Steht ein **LAP-Paket** zur Übermittlung an, so ruft der LAP Manager die Assemblerroutine `LAPWrite` und diese anschließend die Funktion `TransmitPacket`.

`TransmitPacket` prüft zunächst, ob die Node ID des Knotens, zu dem eine Verbindung besteht, mit der Zieladresse des Paketes übereinstimmt. Ist das nicht der Fall, so muß eine neue Verbindung eingerichtet werden. Da die Node ID des Name Servers im allgemeinen nicht $255

ist, wird zuvor geprüft, ob es sich um ein Broadcast-Paket handelt und ob eine Verbindung zum Name Server existiert. Trifft das zu, so kann nach dem Testen der Übertragungsleitung durch `AcquireAddress` das Paket übertragen werden. Andernfalls muß zuvor über `DoDialing` die Nummer zur Node ID ermittelt und eine neue Verbindung aufgebaut werden.

Den Aufbau einer Verbindung in `TransmitPacket` übernimmt `DoDialing`. Übergeben wird dieser Prozedur nur die Rufnummer des Name Servers. Damit wird zunächst die Prozedur `FindNumber` aufgerufen, um die Rufnummer der Zieladresse des Paketes zu ermitteln.

```
FUNCTION DoDialing(Number: ISDNNumber);
```

Liefert der Name Server die gesuchte Rufnummer, so wird durch `DialUp` eine Verbindung zur ermittelten Rufnummer hergestellt. Andernfalls bleibt die Verbindung zum Name Server bestehen. Ist die Rufnummer besetzt, so wird nach kurzer Wartezeit die Anwahl solange wiederholt, bis der Verbindungsaufbau Erfolg hatte, ein Fehler aufgetreten ist oder nach einer vorgegebenen Anzahl von Versuchen (`MaxRetrys`) die Leitung immer noch belegt ist. Endet `DoDialing` erfolglos, so ergeht eine Meldung an den Klienten.

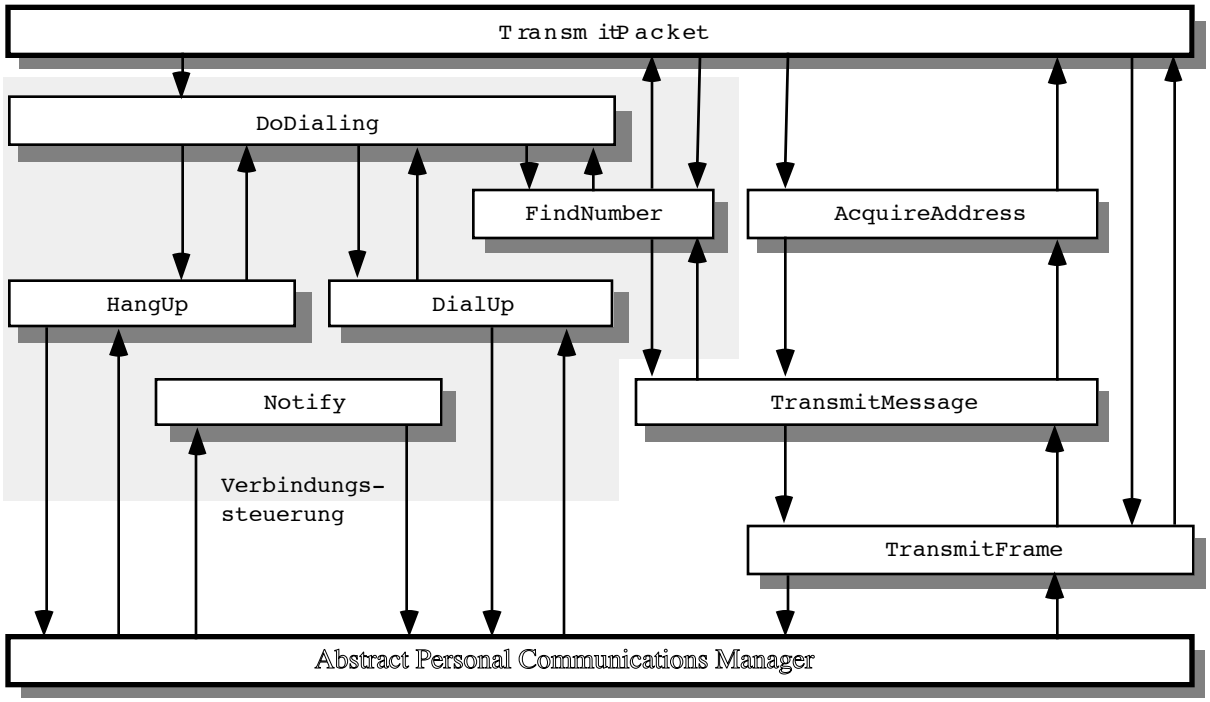

Abbildung 7.4: Die Prozeduren der Verbindungssteuerung

Treten in `DoDialing` oder in `AcquireAddress` Fehler auf, so bricht `TransmitPacket` mit einer Fehlermeldung ab. Kommt es zu keinen Unregelmäßigkeiten, wird das Paket durch `TransmitFrame` schließlich gesendet.

Stehen nach der Übertragung keine weiteren Pakete zur Übermittlung an, so wird die Verbindung durch die VBL-Task `Terminator` abgebrochen. Beim Verbindungsabbau erhält die Variable `LastTransmission` wieder den Wert Null.

7.5.5.3 Die Suche nach der Rufnummer

Beim Öffnen des .MPP wurde die Node ID und die Rufnummer der jeweiligen Station am Name Server registriert. Wird nun eine Rufnummer zu einer Node ID gesucht, so benutzt die Verbindungssteuerung des APCMLAP die Funktion **`FindNumber`**, um vom Name Server die Rufnummer zu einer Node ID zu erfahren. Aufgerufen wird `FindNumber` dabei mit der Nummer des Name Servers und der Node ID, für die eine Nummer zu ermitteln ist. Über den VAR-Parameter `Number` wird die Rufnummer zurückgeliefert. Wurde eine Rufnummer gefunden, so liefert `FindNumber` als Funktionsergebnis TRUE.

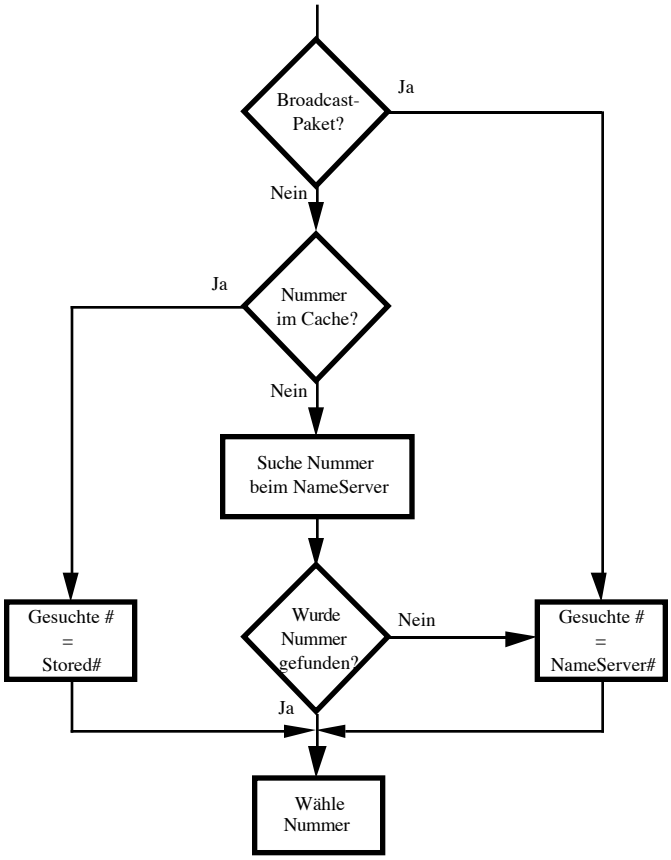

Abbildung 7.5: Der schematische Ablauf der Suche nach einer Rufnummer

Die `FindNumber`-Routine des APCMLAP unterscheidet sich nicht von der in [LUPPER, 90] beschriebenen Implementierung.

7.5.5.4 Zwischenspeichern der Rufnummer

Um die Anzahl der Anfragen nach Rufnummern, die beim Name Server eingehen, und damit die Anzahl der Verbindungen zu reduzieren, wird in der globalen Variablen **`Cache`** die zuletzt gewählte Rufnummer mit zugehöriger Node ID festgehalten. Das Speichern und Auswerten übernimmt bei aktiv aufgebauten Verbindungen die Funktion `FindNumber`.

```
NumberCache = RECORD
                StoredID: BYTE;
                StoredNumber: ISDNNumber;
              END;
```

Ebenso wird durch die Prozeduren `ReceiveCall` und `ReceiveHeader` die Nummer und Node ID derjenigen Station festgehalten, die zuletzt angerufen hat. Das Speichern der letzten Rufnummer führt zu einer erheblichen Reduzierung des Verkehrs mit dem Name Server, da es sehr wahrscheinlich ist, daß die gleiche Station mehrmals hintereinander angewählt werden muß. Wurde eine Node ID unter der im Cache verzeichneten Rufnummer nicht erreicht, so löscht die Prozedur `ClearCache` den Record `Cache`.

Mehrere Nummern im Cache zu halten erscheint nicht sinnvoll, da in den meisten Fällen nur eine Station zur selben Zeit angesprochen wird. Außerdem wird dadurch die Verwaltung der Nummern im Zwischenspeicher komplexer und steht in keinem Verhältnis zum Nutzen.

7.6 Die Paketübertragung im APCMLAP

7.6.1 Die Initialisierung des APCM für die Paketübertragung

Nachdem die Verbindung zwischen zwei Knoten hergestellt ist, ruft `Confirm` die Prozedur **`InitDataStuff.`** Diese aktiviert mit einem `APCMRead`-Aufruf die Datenübertragung. Als Parameter erhält `APCMRead` einen Zeiger auf den Empfangspuffer, die angeforderte Anzahl von Bytes und, da der Aufruf asynchron erfolgt, die Adresse der Interruptserviceroutine **`ReceiveHeader`**, die nach Empfang eines Paketes informiert werden soll. Außer von `Confirm` wird `InitDataStuff` auch von `ReceiverHeader` gerufen, und zwar immer dann, wenn ein empfangenes Paket vollständig behandelt wurde und das Lesen des nächsten Paketes vorbereitet werden muß.

7.6.2 Paketformate und Pakettypen

7.6.2.1 Paketformat

AppleTalk verwendet für die Übertragung von LAP-Paketen ein HDLC/SDLC-artiges Paketformat[16], das auch im APCMLAP beibehalten wird. Abgehende AppleTalk-Pakete (DDP-Frames) werden vom APCM in HDLC-Frames verpackt und über die ISDN-Leitung übertragen.

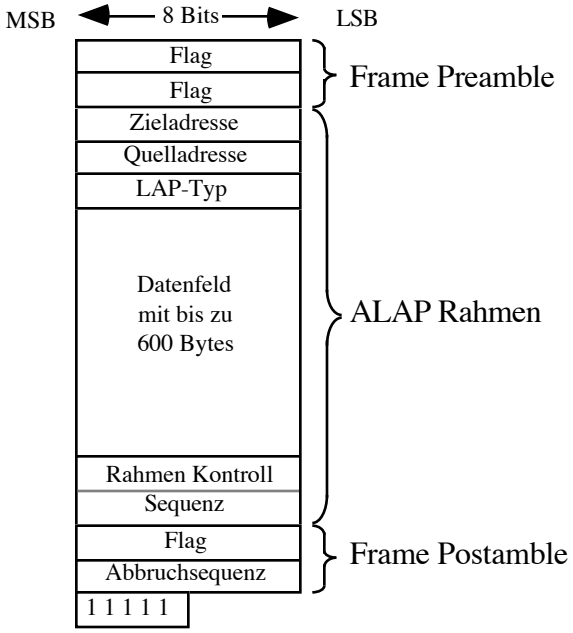

Abbildung 7.6: Das LAP-Paketformat über LocalTalk

7.6.2.2 Pakettypen

APCMLAP verwendet ähnliche Kontrollpakete wie das ALAP. Bekannte Typen von Kontrollpaketen, sofern verwendet, behalten ihre Bedeutung. Um die zusätzlichen Funktionen des APCMLAP zur Suche nach Rufnummern und zum Abbau einer Verbindung erfüllen zu können, werden zwei neue Typen von LAP-Kontrollpaketen eingeführt[17]:

---

16 Vergleiche Paketformat des ALAP in APPLE COMPUTER (Inside AppleTalk).

17 Grundsätzlich ist der Bereich $81–$254 für die Pakettypen der LAP-Kontrollpakete reserviert. In Absprache mit Apple Computer können ungenutzte Typen für eigene Zwecke verwendet werden.

LAP-Typ	ID	Bemerkung
shortDDP	$01	kurzer DDP-Frame
longDDP	$02	langer DDP-Frame
lapENQ	$81	Wird für die dyn. Node-Adressierung, das Registrieren der Rufnummer und zum Testen der Verbindung verwendet.
lapACK	$82	Im Gegensatz zu ALAP erwidert APCMLAP auf ein ENQ-Paket immer ein ACK-Paket.
lapFND	$87	Pakettyp zur Suche nach der Rufnummer beim Name Server.
lapRND	$88	Antwortpaket mit der gesuchten Rufnummer.

7.6.3 Senden eines Paketes mit TransmitFrame

7.6.3.1 Paketübertragung

Wünscht ein Klient des LAP ein Paket zu senden, so ruft er den `LAPWrite`-Code des LAP, auf den die Adresse in `ATalkHk2` verweist. Dabei übergibt er unter anderem einen Zeiger auf die **WDS**[18] (Write Data Structure) des LAP, in der das zu sendende Paket enthalten ist. Die **`LAPWrite`**-Prozedur des APCMLAP ist eine Glueroutine, die die registerbasierenden Parameter für den Aufruf der eigentlichen Pascal-Senderoutinen auf den Stack legt. Schließlich wird von `LAPWrite` die Funktion **`TransmitPacket`** mit einem Zeiger auf die WDS gerufen.

```
FUNCTION TransmitPacket(WDS: WDSEntryPtr): TransmitStatus;
```

`TransmitPacket` prüft vor dem Senden des Paketes die Verbindung zur Gegenstation und stellt sie notfalls her. Ist die Gegenstation erreichbar, so wird die Funktion **`TransmitFrame`** gerufen, die die eigentliche Übertragung übernimmt. Übergeben wird dieser Senderoutine nur ein Zeiger auf die WDS.

```
FUNCTION TransmitFrame(WDS: WDSEntryPtr): TransmitStatus;
```

Hat die Funktion `TransmitFrame` das Paket problemlos übertragen, gibt sie ein positives Funktionsergebnis an `TransmitPacket` zurück (vgl. Abbildung 7.7). Konnte keine Verbindung zur Gegenstation aufgebaut werden oder ist bei der Übertragung des Paketes durch `TransmitFrame` eine Unregelmäßigkeit aufgetreten, so kehrt `TransmitPacket` mit einer Fehlermeldung über `LAPWrite` zum Klienten des LAP zurück.

Da die Macintosh-Implementierung des APCM voll interruptgetrieben ist, können Pakete während eines Interrupts nicht synchron übertragen werden und müssen asynchron gesendet werden. AppleTalk erwartet jedoch, daß bei der Rückkehr von `LAPWrite` die Übertragung von Paketen abgeschlossen ist, und benutzt Puffer, von denen es glaubt, daß ihr Inhalt bereits übertragen wurde, wieder für neue Datenpakete. Dadurch kann es beim asynchronen Senden zum Überschreiben von noch nicht vollständig gesendeten Datenpaketen kommen. Ohne Änderung des APCM läßt sich dieses Problem nur durch einen zusätzlichen Ringpuffer, in den die Pakete kopiert werden, beheben.

Der LAP Manager bietet grundsätzlich die Möglichkeit, Pakete asynchron zu versenden. `LAPWrite` kehrt dann mit einem RTS-Befehl zurück und nicht mit JMP (A0). Von dieser Möglichkeit wird im APCMLAP vorläufig kein Gebrauch gemacht, da noch nicht geklärt ist, welche Variante leistungsfähiger ist.

`TransmitFrame` setzt zunächst den Zähler des Timeouts `LastTransmission` zurück. Anschließend wird geprüft, ob der Ringpuffer voll ist. Ist noch Freiraum, so wird der Kopf des Ringpuffers weitergeschoben, die übergebene WDS durchlaufen und die einzelnen Zeichen der WDS in ein Feld des Ringpuffers kopiert. Enthält das Längenfeld der WDS die Länge 0, so ist die Struktur bis zum Ende durchlaufen und alle Daten sind im Puffer.

---

18 Die Struktur der WDS ist in APPLE COMPUTER (Inside AppleTalk) im Detail beschreiben.

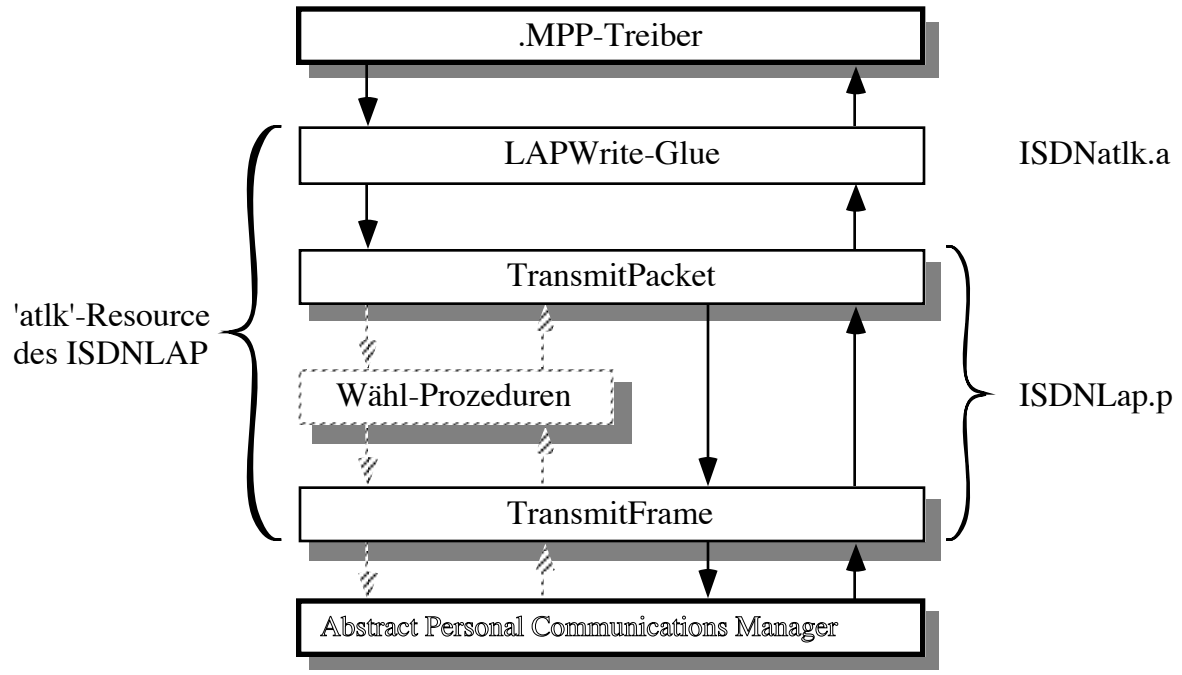

Abbildung 7.7: Die Aufrufkette beim Senden eines Paketes

Für die eigentliche Übertragung eines Paketes mit `APCMWrite` muß zunächst ein Transmit-Parameterblock `TXParams` ausgefüllt werden. In dieser Datenstruktur erwartet der APCM die Adresse des Paketes im Ringpuffer, die Länge des Paketes und die Adresse einer Completion Routine `TransmitComp`, die beim asynchronen Senden gerufen wird, sobald das Paket übertragen wurde. `TransmitComp` hat lediglich die Aufgabe den Schwanz des Ringpuffers nachzuführen.

Hat der APCM den Schreibaufruf in seine Warteschlange aufgenommen und ist dabei kein Fehler aufgetreten, so wird die Funktion `TransmitFrame` mit dem Resultat `TransmitOk` verlassen. Andernfalls wird ein Übertragungsfehler zurückgegeben.

```
FUNCTION TransmitMessage(destAddr: BYTE; Message: BYTE):
                                              TransmitStatus;
```

Eine weitere Prozedur `TransmitMessage` übernimmt das Zusammenbauen einer WDS für ein Kontrollpaket und das Senden über `TransmitFrame`. Als Parameter werden die Zieladresse und der Typ der Meldung übergeben.

7.6.3.2 Fehlerbehandlung beim Senden eines Paketes

Neben Fehlern durch Störungen auf der Übertragungsleitung kann ein **Underrun** des Übertragungsbausteins der ISDN-Karte (HSCX) vorkommen. Dies geschieht immer dann, wenn die Interrupts länger als 4 Millisekunden gesperrt sind. Da das Senden von Paketen im APCM interruptgetrieben ist, bricht in diesem Fall das Paket nach 32 Bytes ab und die Hardware schickt die HDLC-Abortsequenz.

Zur Erkennung und Behebung von Übertragungsfehlern besitzt der APCM den Secure Mode, der verlorene Pakete entsprechend dem HDLC-Protokoll für den Klienten transparent wiederholt. APCMLAP verwendet jedoch lediglich den **Packet Mode**, um eine Duplizierung von Funktionen des APCM in AppleTalk zu vermeiden.

7.6.4 Empfangen eines Paketes mit ReceiveFrame

7.6.4.1 Empfangen eines Paketes

Beim Aufsetzen eines asynchronen `APCMRead`-Aufrufs in `InitDataStuff` wurde die Pascal-Prozedur `ReceiveHeader` des APCMLAP angegeben. Sobald der APCM ein Paket empfangen und im bereitgestellten Empfangspuffer abgelegt hat, wird die `ReceiverHeader`-Prozedur gerufen, um das Paket zu analysieren und an AppleTalk weiterzugeben.

`ReceiveHeader` setzt zuerst den Zähler der Taskaufrufe seit der letzten Paketübertragung (`LastTransmission`) zurück und besorgt sich aus den lokalen .MPP-Variablen einen Zeiger auf die **Read Header Area** (RHA), in die der Kopf des Paketes gelesen werden soll. Anschließend ruft sie die Funktion `ReceiveFrame`, um die ersten 5 Bytes in die RHA zu kopieren. Der Parameter `ReadPacket` gleich TRUE gibt an, daß weitere eventuell im Paket vorhandene Zeichen nicht verworfen werden dürfen.

Kehrt `ReceiveFrame` zurück, so liefert sie über `incomingLength` die Anzahl der tatsächlich gelesenen Bytes. Das Funktionsergebnis zeigt an, ob in `ReceiveFrame` ein Fehler aufgetreten ist. Wurden 5 Zeichen fehlerfrei gelesen, so wird anhand des Pakettyps des Headers in der RHA entschieden, wie mit dem Paket weiter zu verfahren ist.

Handelt es sich um ein fortgesetztes Paket (`continuedFrame`), so kann es sich nur um ein Datenpaket oder ein RND-Paket handeln. Falls die aus dem Packet Header ermittelte Paketlänge sinnvoll ist (3–603), wird anhand des Pakettyps weiter verzweigt. Liegt der Typ im Bereich von **$1–$80**, so wird über die Glueroutine `ReadDispatch` der LAP Manager gerufen. Dieser aktiviert dann anhand des Pakettyps den passenden Protocol Handler, um das restliche Paket einzulesen. Übergeben wird dabei ein Zeiger auf das 6te, nächste freie Byte in der RHA und die Anzahl der Bytes, die noch zu lesen sind. Handelt es sich bei diesem Header um den Anfang eines **RND**-Paketes, so befindet sich im vierten Byte des Headers die Node ID und im fünften die Länge der ermittelten Rufnummer. Über die Prozedur `ReceiveFrame` wird die im Paket enthaltene Rufnummer eingelesen und als aktuelle Rufnummer (`ActualNumber`) übernommen.

Hat `ReceiveFrame` das Paket vollständig gelesen (`closedFrame`), so handelt es sich um ein Kontrollpaket des LAP. Wurde der Kopf eines **ENQ**-Paketes empfangen, dann muß nicht weiter gelesen werden, da diese Pakete nur aus der Zieladresse, der Quelladresse und dem Typ bestehen. `ReceiveHeader` notiert die Quelladresse des Paketes in der globalen Variablen `HisAddress` und im Zwischenspeicher `Cache`. In `IsUp` wird festgehalten, daß die Station angerufen wurde. Schließlich wird über `TransmitMessage` ein ACK-Paket an die Gegenseite übertragen. Gehören die empfangenen Zeichen zu einem **ACK-**Paket, so merkt man sich die Quelladresse des Paketes. Je nachdem, ob diese mit der eigenen Node ID übereinstimmt, d.h. die eigene Node ID eindeutig ist oder nicht, erhält die Variable `AddrStatus` den Wert `InUse` oder `Valid`. Enthält das Feld `LapType` einen unbekannten LAP-Typ oder ist die Länge des empfangenen Paketes nicht korrekt, so wird das Paket verworfen. Bevor `ReceiveHeader` wieder zum APCM zurückkehrt, wird mit `InitDataStuff` ein neuer Read Request aufgesetzt, um das nächste Paket zu lesen.

```
FUNCTION ReceiveFrame(incomingPacket: FramePtr; BytesToRead:
   INTEGER; VAR incomingLength: INTEGER; NoDiscard:BOOLEAN):FrameStatus;
```

Der Funktion `ReceiveFrame` hat lediglich die Aufgabe, ein sich bereits im Empfangspuffer der APCMLAP befindliches Paket stückweise in einen vom Klienten des LAP bereitgestellten Puffer zu kopieren. Dazu wird `ReceiveFrame` ein Zeiger auf den Puffer (`incoming-Packet`), die Anzahl der zu lesenden Bytes (`BytesToRead`), ein VAR-Parameter `incomingLength`, der angibt wieviele Bytes gelesen wurden, und ein Flag `ReadPacket`, welches anzeigt, ob es sich um einen `ReadPacket`- oder um einen `ReadRest`-Aufruf handelt, übergeben.

Zunächst überprüft `ReceiveFrame`, ob die zu lesende Anzahl von Bytes überhaupt noch im Paket vorhanden ist. Trifft das zu, so wird das Funktionsergebnis zunächst auf

`continuedFrame` gesetzt. Andernfalls ist das Funktionsergebnis `closedFrame` und es werden nur noch die im Paket verbliebenen Bytes gelesen.

In `IncomingLength` wird die Anzahl der gelesenen Bytes zurückgegeben. Diese entspricht `BytesToRead`, wenn es sich um einen `ReadPacket`-Aufruf handelt. Andernfalls, wenn ein `ReadRest`-Aufruf stattfindet, wird die Anzahl der verbliebenen Bytes zurückgegeben und das Funktionsergebnis `closedFrame` gesetzt.

Schließlich wird die angeforderte Anzahl von Bytes aus dem Empfangspuffer in den übergebenen Puffer kopiert und der Zeiger, der auf das erste Byte der restlichen Daten im Puffer zeigt, aktualisiert. Danach kehrt `ReceiveFrame` wieder zu `ReceiveHeader` zurück.

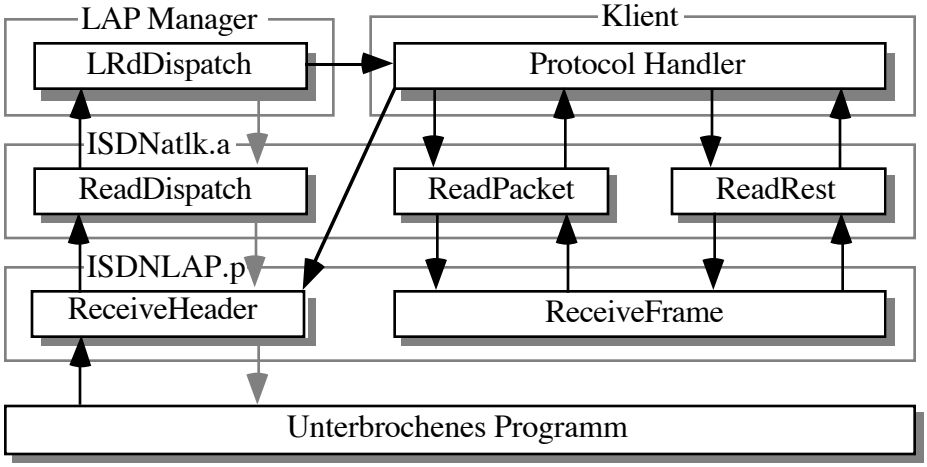

Abbildung 7.8: Call-Mechanismus zwischen APCMLAP und LAP Manager

Die Prozedur `ReceiveFrame` wird aber nicht nur von `ReceiveHeader` zum Lesen des Paketkopfes verwendet, sondern auch von `ReadPacket` und `ReadRest`, um die restlichen Teile eines Frames zu lesen (vgl. dazu Abbildung 7.8 und [Lupper, 90], Abschnitt B.5.3). Dazu müssen lediglich die Parameter passend gesetzt werden.

7.6.4.2 Fehlerbehandlung beim Empfangen eines Paketes

Beim Empfangen eines Paketes können verschiedene Fehler auftreten. Dem Aufrufenden werden sie über das Funktionsergebnis von `ReceiveFrame` mitgeteilt. Die wichtigsten Fehlersituationen und Maßnahmen zur Behebung seien hier kurz erwähnt:

Frame Size Error

Werden weniger als 3 Zeichen empfangen, so kann es sich nicht um ein gültiges LAP-Paket handeln und es wird verworfen. Zusätzlich ist im 4ten und 5ten Byte eines jeden Datenpaketes die Länge der Nutzdaten enthalten. Ist beim Aufruf von `ReadRest` die Anzahl der noch im Paket vorhandenen Zeichen länger als die Anzahl der zu lesenden Zeichen, so□ werden die überschüssigen Bytes verworfen. Anhand der Differenz von angeforderten und gelesenen Zeichen kann der Protocol Handler Fehler erkennen.

Frame Type

Wurde ein Packet mit einem unbekannten Frametyp (Typ > $88) empfangen, so wird es verworfen.

Der APCM hat sich als sehr hilfreich für die Übertragung von Datenpaketen über ISDN erwiesen. Die interruptgetriebene Arbeitsweise des APCM erschwerte jedoch an manchen Stellen die Implementierung des APCMLAP. Außerdem müssen sowohl beim Senden als auch beim Empfangen Daten umkopiert werden, was nicht im Sinne der AppleTalk-Architektur ist.

7.7 Das Interface zum AppleTalk Protocol Stack[19]

7.7.1 Installation des APCMLAP

Die Firma Apple Computer hat für den Austausch des LAP für EtherTalk und andere Implementierungen der AppleTalk-Link-Ebene den LAP Manager geschaffen. Vorgesehen ist zum einen das CDEV Netzwerk zum anderen eine INIT-Resource (ID 18), die beim Systemstart den ausgewählten LAP-Code installiert.

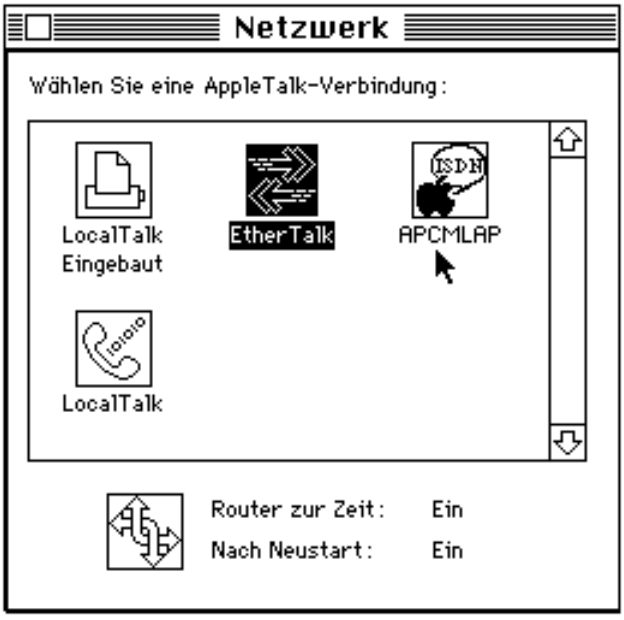

Abbildung 7.9: Das CDEV Netzwerk und zugehörige ADEV im Kontrollfeld

Das CDEV Netzwerk durchsucht bei der Auswahl durch den Benutzer den System Folder nach Files vom Typ ADEV und zeigt alle alternativen LAP-Implementierungen (ADEV) mit ihrem Icon und ihrem Namen im Fenster des Kontrollfelds an (Abbildung 7.9). Dabei fordert das CDEV von jedem ADEV Informationen an. Dies geschieht durch den **`GetADEV`**-Call an den jeweiligen 'adev'-Code des ADEV.

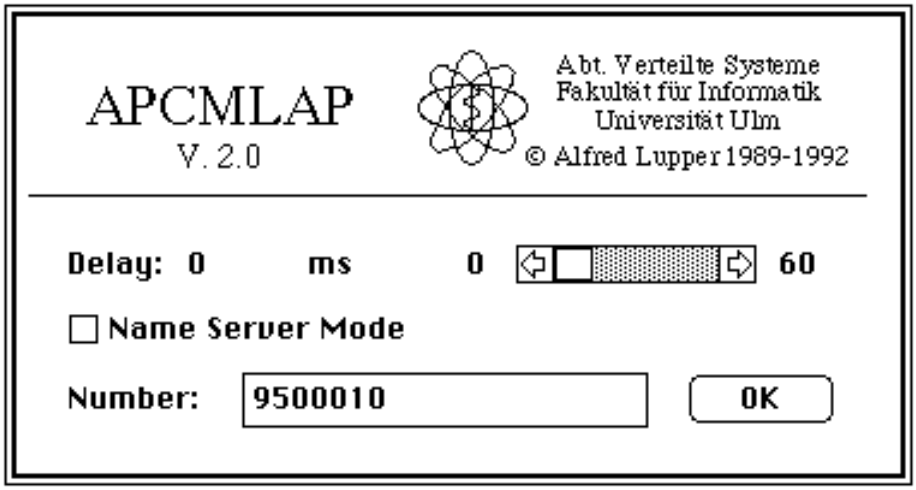

Abbildung 7.10: Eingabe der Nummer des Name Servers bei der Auswahl

[19] Eine detaillierte Beschreibung des Interfaces zum AppleTalk LAP Manager ist in [LUPPER, 90] zu finden.

Wählt der Benutzer durch Anklicken mit der Maus das gewünschte LAP aus, so wird der 'adev'-Code des ADEV durch einen **SelectADEV**-Call über die Auswahl informiert. Die doSelectADEV-Routine in APCMadev.a ruft dabei die Pascal-Prozedur **SetNumber**, welche die Rufnummer des Servers aus einer Resource holt, sie in einer Dialogbox anzeigt und schließlich die geänderte Nummer wieder in die Resource zurückschreibt (Abbildung 7.10).

Bevor allerdings eine neue LAP-Implementierung mit **AInstall** und **LWrtInsert** installiert werden kann, muß das alte LAP aus dem System Heap entfernt werden. Nachdem über **SelectADEV** die 'adev'-Resource ausgewählt wurde, löscht das CDEV den 'atlk'-Code des alten LAP aus dem System Heap. Da der zur Zeit installierte Code von seinem Resource File gelöst (detached) ist, muß sich das CDEV über einen**LWrtGet**-Call an dem LAP Manager die Adresse des aktiven 'atlk'-Codes im System Heap besorgen. Durch einen **AShutDown**-Call an den aktiven 'atlk'-Code wird dieser dann veranlaßt, sich mit einem **LWrtRemove** an den LAP Manager aus dem System Heap und dem ATalkHk2 zu entfernen.

Das Einbinden und der Aufruf der Sende- und Empfangsprozedur durch den LAP Manager entspricht weitgehend der in [LUPPER, 90] beschriebenen Vorgehensweise.

7.7.2 Deinstallation des APCMLAP

Wählt der Benutzer im Kontrollfeld ein anderes LAP als das APCMLAP aus, so muß der Code des APCMLAP deinstalliert und der belegte Speicher ordnungsgemäß zurückgegeben werden. Bevor der LAP Manager den 'atlk'-Code aus den System Heap entfernt, wird deshalb vom CDEV Netzwerk die Routine **doAShutDown** in der 'atlk'-Resource des APCMLAP gerufen, die für die Deinstallation verantwortlich ist.

Nachdem **doAShutDown** einige Register gerettet hat, ruft es die Pascal-Funktion **RestoreOldLAP**, die für die Deinstallation des APCMLAP verantwortlich ist. Wurde diese Prozedur mit FALSE beendet, so wird der LAP Manager mit dem Wert 0 im D0-Register gerufen, worauf dieser eine Fehlermeldung im D0 liefert. Hatte die Deinstallation Erfolg, so wird der LAP Manager mit dem Code für LWrtRemove im D0-Register zum Entfernen des LAP-Codes veranlaßt.

RestoreOldLAP bricht zuerst eine eventuell bestehende Verbindung ab. Anschließend wird von RestoreOldLAP die VBL-Task Terminator aus der VBLQueue entfernt.

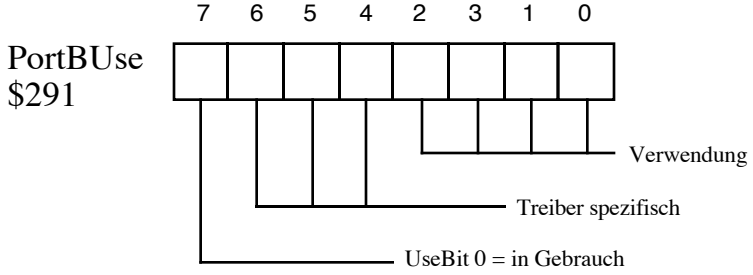

Abbildung 7.11: Das PortBUse-Byte

Schließlich muß in der globalen Betriebssystemvariablen PortUsePtr (Abbildung 7.11) der Wert **$FF** gesetzt werden[20]. Dadurch wird signalisiert, daß der .MPP-Treiber geschlossen wurde und bei einer erneuten Verwendung von AppleTalk zuerst geöffnet werden muß.

Der LAP Manager ermöglicht zusammen mit dem CDEV Netzwerk die Installation eines alternativen AppleTalk Link Access Protocols. Leider ist es zur Zeit noch nicht möglich, mehrere LAPs gleichzeitig zu betreiben.

20 Siehe APPLE COMPUTER (Inside Macintosh), S. 305.

8. Schlußbemerkungen

Die Entwicklung des APCMLAP hat gezeigt, daß die Transaktionsdienste eines für die Bustopologie entworfenen Netzwerkprotokollstapels, der den Spezifikationen des OSI-Schichtenmodells genügt und über exakt definierte Schnittstellen verfügt, durch Substitution der Link-Ebene über die leitungsvermittelnde Struktur von ISDN-Kanälen realisiert werden können.

Mit APCMLAP konnte ferner bekräftigt werden, daß der APCM nicht nur für die Entwicklung von Anwendungen (z.B. PhoneManager) und zur Sprachübertragung geeignet ist, sondern sich ebenso gut als Basis für systemnahe Datenübertragungsprotokolle eignet.

Obwohl die Übertragungsleistung eines ISDN-Kanals mit 64 kbit/s weit niedriger ist als die typischer Übertragungsmedien lokaler Hochleistungsnetze, wird dank einer ökonomischen Verbindungssteuerung und einer effizienten Implementierung ein akzeptabler Durchsatz erzielt. Speziell bei der Übertragung von Dateien mit AppleShare wirkt sich die niedrige Übertragungsleistung des ISDN-Kanals vergleichsweise wenig auf den Gesamtdurchsatz aus.

In ersten Tests mit der Übertragung von Dateien über AppleShare wurde ein absoluter Durchsatz von bis zu **57 kbit/sec.** gemessen. Dies entspricht bei der Übertragung von Dateien durch das Dateisystem einem relativen Durchsatz von **89%** der Bruttoübertragungsrate eines ISDN-Kanals. Durch den AppleTalk Internet Router konnten Dateien immerhin noch mit einem Durchsatz von ca. 35 kbit/sec. übertragen werden. Die Möglichkeit einer zusätzlichen Leistungssteigerung durch transparente Datenkompression auf Basis von Datenpaketen wird zur Zeit geprüft. Hier besteht insbesondere das Problem, daß beim Senden im Interrupt kaum Zeit für die Durchführung der Kompression bleibt.

Die Adaption eines lokalen Netzwerkes an ISDN wurde hier am Beispiel von AppleTalk demonstriert. Sie kann jedoch ohne weiteres auch für andere Protokollstapel, wie Novell Netware, OS/2 LAN-Manager, TCP/IP oder NETBIOS, erfolgen.

9. Literatur

[Apple AACR, 88] Apple Computer: „EtherTalk and Alternate AppleTalk Connections Reference"; Apple Computer Inc., Cupertino, 1988.

[APPLE AFP, 87] Apple Computer: „AppleTalk Filing Protocol"; Apple Computer Inc., Cupertino, 1987.

[APPLE IAT, 86] Apple Computer: „Inside AppleTalk"; Apple Computer Inc., Cupertino, 1986.

[APPLE IM I-V, 86] Apple Computer: „Inside Macintosh I-V"; Apple Computer Inc., Addison-Weseley, Menlo Park, 1986.

[CCITT III.5 85] CCITT: „Red Book III.5: Integrated Services Digital Network (ISDN)"; Series I Recommendations, Genf, 1985.

[CCITT VIII.3, 85] CCITT: „Red Book VIII.3: Data Communications Networks Interfaces"; Recommendations X.20–X.32, Genf, 1985.

[FROITZHEIM, 87] Froitzheim, Konrad: „Implementierung von LAN-Funktionen auf digitalen Nebenstellenanlagen"; Dissertation Universität Augsburg, 1987.

[FROITZHEIM, 91] Froitzheim, Konrad: „Abstract Personal Communications Manager"; Internal Report, Farallon Computing, 1991.

[LUPPER, 90] Lupper, Alfred: „Ein Transaktionsdienst über ISDN-Kanäle mit Protokollsubstitution auf der Link-Ebene"; Diplomarbeit Universität Augsburg, 1990.

[MARONEY, 87] Maroney, T.: „File Servers versus Disk Servers"; MacTutor, Placentia California, Vol. 3, No. 4, S. 73 ff, 1987.

[MOTOROLA, 84] Motorola: „M68000 – Programmer's Reference Manual"; Prentice-Hall, Englewood Cliffs, New Jersey, 1984.

[TANENBAUM, 88] Tanenbaum, Andrew S.: „Computer Networks", Prentice-Hall, Englewood Cliffs, New Jersey, 1989.