[StMü02] *Ralf Steinmetz, Max Mühlhäuser;* **Rechnernetze; Informatik-Handbuch;** Kapitel 6, 3. Auflage, Hanser-Verlag

6 Rechnernetze

R. Steinmetz, M. Mühlhäuser

6.1 Grundbegriffe und Besonderheiten

Alle Grundbegriffe dieses Gebiets müssen immer zusammen mit Kapitel D10 „Verteilte Systeme“ gesehen werden, da beide Kapitel die wesentlichen Bestandteile von Kommunikationssystemen beschreiben. Es werden, ausgehend von der übergeordneten Betrachtung generischer Kommunikationsabstraktionen und -mechanismen, insbesondere die Aspekte der Kommunikation und weniger die Aspekte verteilter Systeme betrachtet.

Als *Rechnernetze* (auch *Kommunikationsnetze* oder kurz *Netze* genannt) bezeichnet man alle für eine Verbindung mehrerer autonomer Rechnersysteme notwendigen Mecha-

nismen und deren Realisierung als Komponenten in Hardware und Software. Rechnersysteme gelten als miteinander verbunden, wenn sie Daten austauschen können. Die Rechnersysteme können heute sowohl PCs (herkömmliche Rechner), als auch beispielsweise multifunktionale ISDN-Telefone oder digitale Fernsehgeräte mit Internetzugang sein, die im allgemeinen nicht an nur einer gemeinsamen Aufgabe arbeiten. Sie sind „offene Systeme", das heißt solche, an die man im Verlauf der Zeit neue Rechner beliebiger Bauart, Datendarstellung und Leistungsfähigkeit anschließen kann. Man nennt die durch ein solches Netz verbundenen Rechnersysteme *Endsysteme*; die innerhalb des Netzes zum Zweck der Datenkommunikation agierenden Systeme bezeichnet man als *Netzknoten* oder kurz *Knoten*. Je nach Funktionalität werden diese Knoten *Dienstübergänge* (*Gateways*), *Vermittlungsstelle*, *Router*, *Brücke*, *Hub*, *Switch* oder *Repeater* genannt. Endsysteme und Netzknoten werden gemeinsam als *Stationen* bezeichnet.

Der Begriff der *Verbindung* hat im kommunikationstechnischen Sinn eine präzise Semantik, die im Folgenden vertieft wird.

6.1.1 Verbindungsorientierte Kommunikation

Als *Verbindung* wird eine für einen wohldefinierten Zeitraum bestehende Beziehung zwischen meist zwei kommunizierenden Instanzen bezeichnet. Diese Beziehung ist dadurch charakterisiert, daß Daten in derselben Reihenfolge empfangen werden, in der sie abgeschickt wurden. Eine Verbindung stellt im allgemeinen auch sicher, daß alle versendeten Daten die empfangende Instanz korrekt erreichen. Verbindungen müssen vor ihrer Benutzung aufgebaut werden. In Analogie zum Telefonieren ergeben sich somit drei Phasen:

1 Verbindungsaufbau (*connect, set-up*): Hier wird die empfangende Instanz angesprochen und der Weg für alle Dateneinheiten festgelegt, siehe auch Bild 5.
2 Datenübertragung (*data transfer*): Nach erfolgreichem Verbindungsaufbau können Daten von der sendenden zur empfangenden Instanz übertragen werden.
3 Verbindungsabbau (*disconnect*): Nach beendeter Datenübertragung wird die Verbindung abgebaut und somit die Beziehung aufgelöst.

Eine verbindungsorientierte Kommunikation ist meist sinnvoll, wenn (1) regelmäßig wiederkehrende Dateneinheiten zu übertragen sind, (2) dies über einen längeren Zeitraum erfolgt und (3) oft auch Dienstgütegarantien (wie die Auslieferung von Datenpaketen in der Sendereihenfolge, Übertragungswiederholung bei Fehlern, Zeitbedingungen, erforderliche Datenrate) einzuhalten sind.

6.1.2 Verbindungslose Kommunikation

Werden Daten versendet, ohne zuvor eine Kommunikationsbeziehung aufzubauen, so heißt die Kommunikation „verbindungslos". Diese Übertragungsart ist eine Analogie zur Briefzustellung: Jeder Brief enthält selber die Zieladresse; er wird isoliert betrachtet und „verarbeitet". Verbindungslose Kommunikation wird meist für einmalige Datenübertragung verwendet und ist somit von kurzer Dauer. In *verbindungsloser* Kommunikation werden Datenpakete unabhängig voneinander verschickt; sie müssen jeweils die vollständigen Empfängerangaben enthalten.

Insgesamt erfordert die Vielzahl der Aufgaben eine möglichst einheitliche Strukturierung des Gebiets durch Festlegung geeigneter Abstraktionsebenen. Dies erfolgt hier in Form eines Schichtenmodells, das weitgehend in der Telekommunikationswelt (Ausgangspunkt Telefonie) und im Internet (Ausgangspunkt Datenwelt) angewendet wird.

6.2 OSI-Referenzmodell und Internetarchitektur

[B1]

Die internationale Standardisierungsorganisation ISO hat mit dem sog. OSI-Referenzmodell (*basic reference model „open systems interconnection"*) drei wesentliche Ziele erreicht: (1) Ein Modell geschichteter abstrakter Maschinen wurde geschaffen, (2) die Terminologie im Zusammenhang mit diesen abstrakten Maschinen wurde einheitlich definiert, und (3) eine konkrete Schichtung mit sieben Schichten wurde festgelegt (siehe auch Bild 3). Mit Ausnahme der Betrachtung gewisser Funktionsmengen als eigenständige Schichten (und im Gegensatz zu etlichen von ISO standardisierten Protokollen) ist das OSI-Referenzmodell heute die Grundlage jeder Netzwerkarchitektur, auch des Internets. Die wesentlichen Konzepte und Begriffe werden im folgenden vereinfacht erläutert.

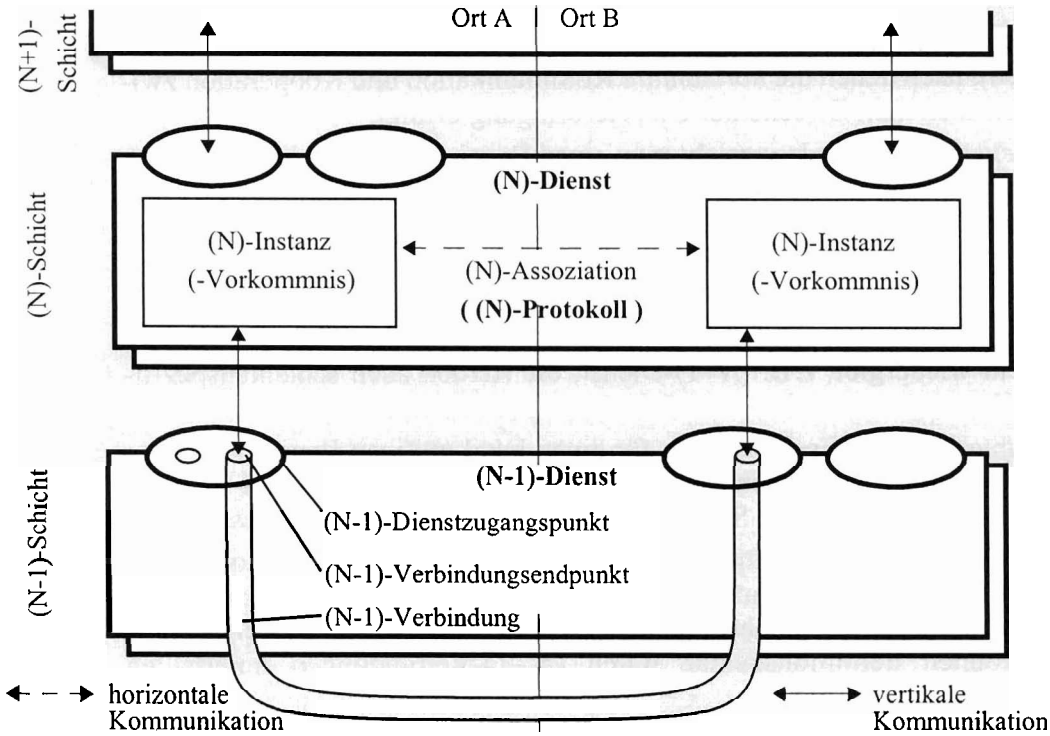

Bild 1 OSI-Referenzmodell, vereinfacht

Bild 1 zeigt die Verbindung einer Schicht mit der über und unter ihr liegenden Schicht. In ähnlicher (aber leider weniger klar strukturierter Weise durch die hier oft unscharfe Trennung von Dienst und Protokoll) wurde dies auch mit dem Internet als sogenannte „Internetarchitektur" verwirklicht.

6.2.1 Architekturelemente

Ein (*Kommunikations-)Dienst* kapselt eine sinnvoll gruppierte Menge von Funktionalitäten und verbirgt sowohl Realisierungsdetails (das „Wie", die sog. „Erbringung") als auch benutzte „tieferliegende" Dienste (sog. „Basisdienste"). Ein Dienst wird *verteilt* und nebenläufig durch kommunizierende Prozesse erbracht.

Indem Dienste einander benutzen und verbergen, entsteht eine Schichtenstruktur. Höhere Schichten bieten höhere Qualität und größere Funktionalität von Diensten.

- Eine *Schicht* enthält einen oder mehrere Dienste; ein Dienst einer Schicht (*N*), bezeichnet als (*N*)-Dienst, benutzt Dienste der nächstniedrigen Schicht als Basisdienste und bietet seine Funktionalität Diensten der nächsthöheren Schicht (Nutzern) an. Dienste derselben Schicht entsprechen einer bestimmten Qualitäts- und Leistungsklasse. Für die niedrigste Schicht ersetzt das physische Übertragungsmedium den Basisdienst, für die höchste Schicht sind Anwendungsprozesse die Nutzer.
- *Instanz* bezeichnet denjenigen Teil einer (verteilten) Diensterbringung, der sich an einem bestimmten Ort befindet – also ein Softwaremodul, das gemeinsam mit anderen Modulen auf anderen Knoten (sog. *Partnerinstanzen*) den Dienst erbringt. In höheren Schichten wird häufig für jeden Nutzer eine eigene Inkarnation einer Instanz aufgerufen, diese wird dann als *Instanz-Vorkommnis* bezeichnet. Niedrigere Schichten sind als Teil des Betriebssystems oder in Firm- oder Hardware *reentrant* realisiert.
- Der Begriff (*Kommunikations-*)*Protokoll* bezeichnet die Festlegungen (Regeln, Abläufe, Prozeduren), nach denen die *horizontale* Kommunikation und Kooperation zwischen Partnerinstanzen zum Zwecke der Diensterbringung erfolgt.
- Unter einem *Dienstzugangspunkt* versteht man einen Bezeichner, über den ein Dienst von seinen Nutzern adressiert werden kann. Meist handelt es sich um eine Adresse und Datenstruktur nach betriebssystemabhängigen Konventionen zur Identifikation von Systemdiensten; pro Dienst und Ort können mehrere Dienstzugangspunkte vorhanden sein; über sie erfolgt die *vertikale* Kommunikation.

Die genannten Begriffe werden häufig mit einem Präfix versehen, das die relative Nummer der Schicht wiedergibt, z.B. (*N*+1)-Dienst; oft werden auch schichtenspezifische Kürzel ohne Klammer verwendet, etwa *T* für Transportschicht.

- *Verbindungsendpunkt* ist der Fachausdruck für einen lokal gültigen Bezeichner einer Verbindung, der zwischen Dienstbenutzer und lokaler Instanz ausgetauscht wird. Er tritt nach Verbindungsaufbau an die Stelle der ausführlichen Empfängeradresse.
- *Assoziation* bezeichnet das „Gedächtnis" über die horizontale Kommunikationsbeziehung zwischen kooperierenden Instanzen oder auf der höchsten Schicht zwischen Anwendungsprozessen; nur für die höchste Schicht ist der Begriff wirklich wichtig. (*N*)-Assoziationen können definitionsgemäß durch (*N*−1)-Verbindungen unterhalten werden.

6.2.2 Dateneinheiten, Dienste und Protokolle

Das OSI-Referenzmodell ordnet auch die Terminologie um die im verteilten System ausgetauschten Dateneinheiten. Sie werden auf dem Weg durch die Schichten mit Zusatzinformationen für die horizontale und vertikale Kommunikation versehen. Die nachfolgenden deutschen Begriffe entsprechen nicht immer den deutschsprachigen Normen.

Übergang zwischen Schichten, vertikale Kommunikation – Dienst. Aufträge eines Benutzers an eine Schicht und Meldungen in Gegenrichtung werden als *Dienstprimitiven* bezeichnet („die Primitive", auch „das Primitiv"), als Datenstrukturen *interface data unit* (*IDU*) genannt. Sie enthalten nach Bild 2 einen *Kommandoteil* (*interface control information, ICI*) und optional *Nutzdaten* (*service data unit, SDU*).

Horizontale Kommunikation zwischen Partnerinstanzen – Protokoll. Nutzdaten werden von der lokalen Instanz über den Basisdienst an die Partnerinstanz gesendet mit dem Auftrag, sie auszuliefern. Die Nutzdaten werden dazu in ein (*Protokoll-*)*Paket* (*pro-*

tocol data unit, PDU) verpackt und mit sog. *Protokollinformation* (*protocol control information, PCI*) versehen (siehe Bild 2) , die beispielsweise den Pakettyp kennzeichnet. Nutzdaten werden meist im Pakettyp *Datenpaket* übertragen, Pakettypen wie *Quittung* enthalten keine Nutzdaten.

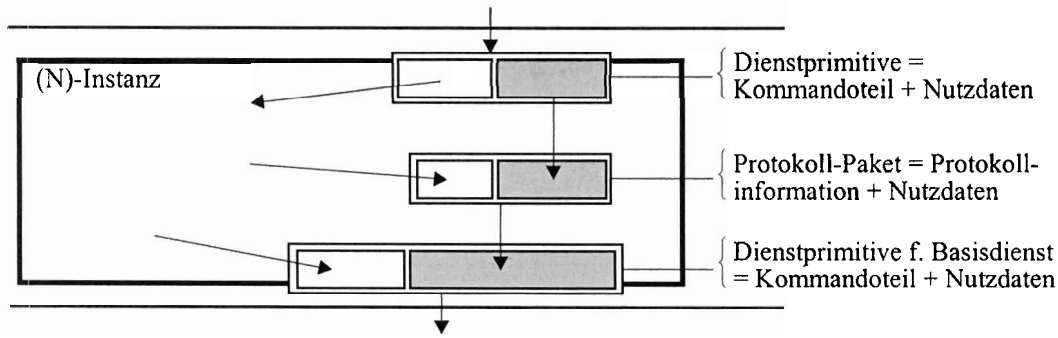

Bild 2 Zusammenhang zwischen Typen von ausgetauschten Dateneinheiten

Bestätigte und unbestätigte Dienste. Dienstprimitiven werden zu Gruppen, sog. *Diensten oder Dienstfunktionen* zusammengefaßt. Man unterscheidet vor allem:

- Bestätigte Dienstfunktionen. Diese bestehen aus vier Dienstprimitiven: *Anfrage, Anzeige, Antwort* und *Bestätigung* (*Req*uest, *Ind*ication, *Resp*onse, *Conf*irmation). Schichten-Kürzel, Dienstfunktions-Kürzel und Art der Dienstprimitive (meist der oben kursiv gedruckte Teil) werden häufig zu einem Bezeichner kombiniert, beispielsweise für eine Dienstfunktion *Connect* (Kürzel *Con*) zum Aufbau logischer Verbindungen in der Transportschicht *T-Con.Req*, *T-Con.Ind*, *T-Con.Rsp* und *T-Con.Cnf*.

  Unbestätigte Dienstfunktionen. Diese bestehen nur aus Anfrage und Anzeige.

6.2.3 Schichtungsvorschlag

Die ISO hat über die Modell- und Begriffsbildung hinaus die Unterteilung von Kommunikationsarchitekturen in sieben Schichten vorgeschlagen (Bild 3). In der Internetarchitektur sind dies vier Schichten.

- OSI-Schicht 1, die *Bitübertragungsschicht Ph*, handhabt den Bitstrom zwischen physisch verbundenen Systemen. Entsprechende Dienste legen die elektrische Darstellung der Bits (Leitungscodierung) und Eigenschaften von Kabeln und Steckern fest.
- OSI-Schicht 2, die *Sicherungsschicht D*, bietet einen gesicherten Paketstrom zwischen physisch verbundenen Systemen. Der Bitstrom wird in numerierte und quittierte Pakete (*Rahmen, frames*) zerlegt. Für lokale Netze, bei denen sich mehrere Knoten ein physisches Medium teilen, hat sich eine Untergliederung in Schicht 2a (Zugangsprotokoll zum Medium: media access control) und 2b (Fehlererkennung und Überlastkontrolle für Empfänger: logical link control) durchgesetzt.
- OSI-Schicht 3, die *Vermittlungs- oder Netzwerkschicht N*, bietet durchgehenden Pakettransport zwischen Endsystemen an. Dort werden Wege zwischen nicht physisch verbundenen Knoten gefunden (Wegewahl) und Pakete weitergeleitet (Vermittlung).
- OSI-Schicht 4, die *Transportschicht T*, leistet die Übertragung von Datenpaketen zwischen Betriebssystemprozessen und verbirgt dabei Topologie und Eigenschaften des Netzes. Diese „Ende-zu-Ende"-Übertragung erfolgt häufig verbindungsorientiert.

OSI-Schicht	Funktionen	Internet-Schicht
1 Bitübertragung *Physical*	Senden von Bits, die als solche korrekt empfangen werden: Mechanik: Steckerart, Kabel/Medium, ... Elektronik: Spannung, Dauer eines Bits, ... Prozedural: unidirektional, bidirektional, ...	Host-an-Netz *Network Interface*
2 Sicherung *Data Link*	Zuverlässige Datenübertragung zwischen benachbarten Endsystemen und Knoten Einführung von Datenrahmen, Fehlererkennung und -beseitigung im Rahmen, Flußregelung, Zugangsregelung, ...	
3 Vermittlung *Network*	Kommunikation Endsystem zu Endsystem Leitwegbestimmung, Überlastkontrolle, Adressierung, ...	Internet *Internet*
4 Transport *Transport*	Kommunikation Anwendung/Prozeß zu Anwendung/Prozeß: Dienstgüte, Adressierung, Verbindungsverwaltung, Flußkontrolle, ...	Transport *Transport*
5 Kommunikationssteuerung *Session*	"Sitzung" über längeren Zeitraum unterstützen: Wiederaufsetzen von Transportverbindungen, ...	Verarbeitung oder Anwendung *Application*
6 Darstellung *Presentation*	Datendarstellung unabhängig von Endsystemen: Einheitliches Datenstrukturzwischenformat, ...	
7 Anwendung *Application*	Anwendungsbezogene Dienste: Elektronische Post, Dateitransfer, Web, ...	

Bild 3 OSI- und Internet-Schichtenmodell

- OSI-Schicht 5, die *Kommunikationssteuerungsschicht S*, erlaubt die Strukturierung einer oder mehrerer aufeinanderfolgender Transport-Verbindungen. Dazu kann das Recht zum Aufruf bestimmter Dienstprimitiven zwischen Nutzern geregelt werden. Die Datenübertragung kann in Abschnitte gegliedert werden, vor allem als Basis für Rücksetz-Operationen.
- In OSI-Schicht 6, der *Darstellungsschicht P*, wird die Datendarstellung für heterogene Endsysteme harmonisiert. Über ein einheitliches Zwischenformat (Transfersyntax) wird die Binärcodierung der Basisdatentypen zwischen Rechnerarchitekturen und Programmiersprachen angepaßt. Definitionen komplexer Datenstrukturen können zwischen kommunizierenden Prozessen ausgetauscht werden (abstrakte Syntax).
- OSI-Schicht 7, die *Anwendungsschicht A*, enthält sog. Dienstelemente für wiederkehrende, allgemeine Aufgaben sowie Dienstelemente für spezielle Anwendungsklassen.

Details der OSI-Schichten 1 – 4 sind Gegenstand dieses Kapitels. **[B2]**

In der Internetarchitektur sind nach Bild 3 die OSI-Schichten 1 und 2 zu einer Schicht „Host-an-Netz" (*network interface*) zusammengefaßt. Außerdem sind die OSI-Schichten 4, 5 und 6 als „Verarbeitungs- oder Anwendungsschicht" (*application*) zusammengefaßt.

6.2.4 Zusammenfassung, Kritik und weitere Normungstätigkeit

Das OSI-Referenzmodell hat wie keine andere Norm zum Verständnis von Kommunikationsdiensten und -protokollen beigetragen, die Sprache vereinheitlicht und Grundlagen für die Weiterentwicklung gelegt. Es ist als Grundwissen über Kommunikation und verteilte Systeme unerläßlich. Inzwischen ist auch klar geworden, daß die strenge Abgrenzung der OSI-Schichten keine schichtenspezifische separate Implementierung (beispielsweise jede Schicht als separater Prozeß) impliziert. Eine solche Folgerung würde zu signifikanten Leistungseinbrüchen führen; beispielsweise durch vielfältiges unnötiges Kopieren der Daten und vielfältige zeitintensive Prozeßkontextwechsel. Kritisiert wurden auch diejenigen (meist anwendungsorientierten) Dienste, die nicht von einem etablierten De-facto-Standard abgeleitet wurden. Viele OSI-Normen wurden von den Gremien IEEE und ITU (früher CCITT) mitgeprägt; siehe auch Kapitel D10, G1 und G2.

6.3 Softwaretechnik für Kommunikationsprotokolle

Bei der *Softwaretechnik für kommunizierende Systeme* ist die Entwicklungsunterstützung für Kommunikationsprotokolle und -dienste (*protocol engineering*) von der Softwaretechnik für verteilte Anwendungsprogramme (*distributed application engineering*) zu unterscheiden. Für erstere sind die nachfolgend beschriebenen Mechanismen weithin bekannt, für letztere gibt es noch kaum weitverbreitete Techniken; Softwaretechnik für verteilte Anwendungsprogramme wird daher nur fallweise im zweiten Teil dieses Kapitels erwähnt.

Spezifikation und Entwurf von Protokollen und Diensten. Dienste sind definiert durch die spezifizierten *Dienstprimitiven*, die zugehörigen *Orte* (Knoten, Dienstzugangs-, Verbindungsendpunkte) sowie *Ablauffestlegungen,* insbesondere die vom Nutzer einzuhaltende Abfolge von Dienstprimitiven und die vom Dienst zugesicherten Reaktionen hierauf. In der abstrakten Beschreibung eines Dienstes entfallen konkrete Ortsbezeichner; meist werden nur zwei kooperierende Nutzer behandelt, die dann häufig *Initiator* und *Beantworter* oder gar völlig abstrakt *A*, *B* usw. genannt werden. Im Unterschied zu den Vor- und Nachbedingungen sequentieller Programme sind Dienstabläufe sehr viel häufiger nichtdeterministisch in dem Sinne, daß eine vom Nutzer übergebene Dienstprimitive unvorhersehbar unterschiedliche Reaktionen des Dienstes hervorrufen kann.

Protokollbeschreibungen verfeinern Dienstbeschreibungen, indem sie zeigen, wie die Abfolgen von Dienstprimitiven (gemäß Ablauffestlegungen) unter Nutzung des *Basisdienstes* durch Austausch von Protokoll-Paketen zwischen Instanzen realisiert werden.

Weg-Zeit-Diagramme. Zur intuitiven Beschreibung von Diensten und Protokollen sind Weg-Zeit-Diagramme weit verbreitet, obwohl sie mangels ausreichender theoretischer Fundierung und mangels Flexibilität oft nur von begrenztem Nutzen sind. Für ein erstes Verständnis von Diensten und Protokollen reichen sie jedoch aus, siehe als Beispiel Bild 4 . Die zeitlichen Abläufe werden von oben nach unten aufgetragen, wobei gleiche Höhe für Gleichzeitigkeit steht. Orte des Geschehens werden horizontal angeordnet: bei Diensten meist zwei, die Schnittstellen Initiator-Dienst und Dienst-Beantworter; bei Protokollen häufig vier, nämlich Initiator-Instanz1, Instanz1-Basisdienst, Basisdienst-Instanz2 und Instanz2-Beantworter.

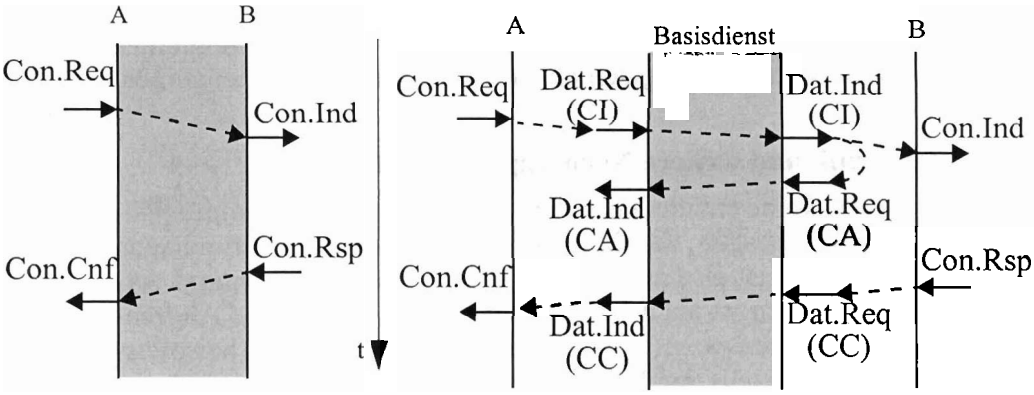

Bild 4 Dienst- (links) und Protokollbeschreibung (rechts) einer bestätigten Dienstfunktion Connect (Con) mittels Weg-Zeit-Diagramm. Gestrichelte Pfeile deuten nur Zusammenhänge an, keine Protokolldetails. CI und CC: benachrichtigen die Partnerinstanz vom erfolgten Con.Req oder Con.Rsp, CA: quittiert CI-Empfang. Dat: Basisdienstfunktion für Datenübertragung.

Zustandsautomaten. Sie gelten wie Weg-Zeit-Diagramme als intuitiv verständlich, erlauben zudem die Modellierung von Alternativen und formale Handhabung. Protokolle und Dienste sind (infolge von Übertragungsfehlern, Nebenläufigkeit usw.) nichtdeterministisch und wechseln möglicherweise (auf gegebener Abstraktionsstufe) spontan Zustände. Das läßt sich mit herkömmlichen endlichen Automaten nicht ausdrücken.

Bild 5 zeigt einen einfachen Dienstautomaten für den Verbindungsaufbau. Dienstprimitiven fungieren als Eingaben (Aufträge) oder Ausgaben (Meldungen) von Zustandsübergängen. Die Orte des Auftretens (Initiator, Beantworter) sind nicht graphisch darstellbar, können aber als Teil der Bezeichner ausgedrückt werden. Im Gegensatz zu Bild 4 können Alternativen modelliert werden: Der Beantworter kann den Verbindungsaufbauwunsch ablehnen. Auch Nichtdeterminismus ist beschreibbar, hier die Option, daß der Dienst mangels Ressourcen den Verbindungswunsch per Verbindungsabbau-Anzeige ablehnt.

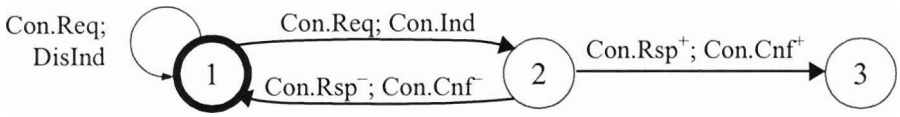

Bild 5 Dienstautomat. Abkürzungen zusätzlich zu Bild 4: Dis.Ind: Verbindungsabbau (Disconnect)-Anzeige des Dienstes; „+" und „–": Annahme oder Ablehnung durch den Beantworter

Dies führt zu verschiedenen Übergängen bei Eingabe von *Con.Req* in Zustand 1. Die Zustandsbezeichner sind für die formale Behandlung irrelevant und könnten sinntragender gewählt werden (1 = Ruhe, 2 = Warten auf Beantworter, 3 = Verbindung aufgebaut). Zur *Protokollbeschreibung* werden Automaten verwendet, die die beteiligten Instanzen und den Basisdienst beschreiben. Häufig werden Phasen des Ablaufs (Verbindungsaufbau, -abbau, Datenaustausch) getrennt modelliert. Initiator und Beantworter werden mit verschiedenen, auf diese Funktion reduzierten Automaten beschrieben.

Zu den wichtigsten verbreiteten formalen Beschreibungsverfahren für Dienste und Protokolle gehören neben streng formalisierten Verfahren auf Automatenbasis noch Petrinetze, temporale Logik sowie die algebraische Spezifikation.

Verifikation: Dienst- und Protokollspezifikationen können einzeln auf (un-)erwünschte Eigenschaften hin geprüft werden (Verklemmungsfreiheit, Abwesenheit unerreichbarer Zustände usw.), was als *Dienst-* oder *Protokollüberprüfung* bezeichnet wird. *Protokollverifikation* weist nach, daß ein spezifiziertes Protokoll einen gegebenen spezifizierten Dienst tatsächlich erbringt. Dies sei für automatenbasierte Spezifikation erläutert.

Typischerweise ist nachzuweisen, daß der *parallele* Ablauf mehrerer Automaten (zwei Instanzen plus Basisdienst) an der Dienstschnittstelle das Verhalten des Dienstautomaten ergibt. Dabei müssen die parallelen Automaten in einen einzigen *Produktautomaten* verwandelt werden; dieser enthält als Zustände Vektoren der Zustände der Teilautomaten (wenn etwa alle drei Teilautomaten ihren Startzustand mit 0 bezeichnen, dann startet der Produktautomat im Zustand 000). Vom Startzustand aus wird sukzessive der Produktautomat aufgebaut, indem pro Zustandsvektor geprüft wird, welche Teilautomaten unmittelbar Zustandswechsel vollziehen können. Das den Zustandswechsel im Teilautomaten auslösende Ereignis markiert auch den Übergang des Produktautomaten. Das Verfahren terminiert mit einer Zustandsanzahl, die maximal der Potenzmenge der Zustandsanzahl der Teilautomaten entspricht (Zustandsexplosion). Dann werden alle Ein-/Ausgaben gestrichen, die nicht an der Dienstschnittstelle sichtbar sind, und es wird versucht, die Verhaltensäquivalenz von Produkt- und Dienstautomat nachzuweisen. Exakte Vorgehensweise, Äquivalenzbegriffe, Beweismethoden und Beherrschung der Zustandsexplosion beschäftigen die Forschung.

Standardisierte Sprachen und Techniken: Mit *Lotos* hat die ISO eine Sprache für algebraische Spezifikation standardisiert, mit *Estelle* für erweiterte Automaten. Ereignisse werden in Estelle mit Bedingungen und Prioritäten verknüpft und Ausgaben mit Zuweisungen. Die Pascal-ähnliche Sprache kennt Module mit Interaktionspunkten, die über Kanäle zusammengeschaltet werden. Zustandsübergänge werden wie folgt programmiert:

```
from <Zustand>
  when <Eingabe> provided <Bedingung> priority <prio>
  to <Zielzustand>
  begin <Ausgaben, Zuweisungen, ...> end;
```

SDL schließlich wurde aus der Kommunikationstechnik übernommen. Es existiert in einer programmiersprachlichen und einer graphischen Variante und unterscheidet Zustände, Sende- und Empfangsoperationen, Flußdiagramm-ähnliche Abfolgen und (möglicherweise vom eingehenden Datum abhängende) Verzweigungen.

Conformance-Test. Für die Testphase eignen sich die auf Testdaten beruhenden Verfahren der sequentiellen Programmierung nicht. Einerseits führt Nichtdeterminismus bei gleichen Eingaben zu unterschiedlichen Ergebnissen, andererseits bestimmen Verhaltensdetails im Protokollablauf, ob Instanzen verschiedener Hersteller gemeinsam einen spezifizierten Dienst erbringen können. Hier hat sich eine wohldefinierte Vorgehensweise, der sog. *Conformance-Test,* etabliert, bei der eine zu testende Instanz oder Schicht im Zusammenwirken mit einem lokalen oder entfernten, zertifizierten Referenzsystem getestet werden (anstatt alle Implementierungen aller Hersteller gegeneinander auszutesten).

6.4 Protokollmechanismen

Die Erfahrung hat gezeigt, daß unterschiedliche Protokolle vergleichbare in sich abgeschlossene Teilfunktionen enthalten. Diese werden als *Protokollmechanismen* bezeichnet. Trotz Auffassungsunterschieden über ihre Anzahl und Aufteilung bilden sie ein kanonisches methodisches Wissen. Im folgenden werden sie in sieben Klassen unterteilt.

Basis-Protokollmechanismen:

- *Datentransfer und Vorrang-Datentransfer*: Da der Datentransfer der eigentliche Zweck von Protokollen ist, wird dieser meist unbestätigte Protokollmechanismus fast immer verwendet. Hinzu kommt bisweilen der sog. *Vorrang-Datentransfer* (*expedited data transfer*). Vorrangdaten werden garantiert vor nachfolgenden normalen Daten ausgeliefert und können vorher abgesandte normale Daten überholen. Sie unterliegen nicht der normalen Flußkontrolle (siehe weiter unten).
- *Verbindungsverwaltung*: Erfolgreicher Verbindungsaufbau sowie Verbindungsablehnung durch den Beantworter oder Dienst wurden in Bild 5 bereits vorgestellt. Der Ablauf wird weiter verkompliziert durch die Gefahr verlorengegangener, duplizierter oder stark verspäteter Protokoll-Pakete. Bei der bestätigten Dienstfunktion Verbindungs*abbau* (*disconnect*) werden in vielen Protokollen ausstehende Datenübertragungen noch ausgeführt. Dies ist bei der unbestätigten Dienstfunktion Verbindungs*abbruch* (*abort*) nicht der Fall.
- *Numerierung*: Einige nachfolgend beschriebene Protokollmechanismen zur Fehlerbehandlung und Systemleistungsanpassung setzen die Numerierung von Datenpaketen voraus. Diese kann nur in verbindungsorientierten Diensten erfolgen. Sie ermöglicht auch die reihenfolgetreue Auslieferung der Pakete an den Empfänger.
- *Quittierung*: Korrekter Empfang von Datenpaketen wird mit einem speziellen Protokoll-Pakettyp (*acknowledgement, ACK*) quittiert oder in einem reservierten Feld eines Datenpaketes, das in Gegenrichtung übertragen wird (*Huckepack-Quittung, piggybacked acknowledgement*). Sammelquittungen für mehrere Datenpakete sind üblich.

Protokollmechanismen zur Fehlerbehandlung

- *Zeitüberwachung*: Beim Absenden eines Datenpaketes kann ein Zeitgeber (*timeout*) gestartet werden. Läuft er ab, ohne daß eine Quittung eingetroffen ist, dann schließt der Sender auf Verlust des Datenpaketes oder der Quittung. Das Datenpaket wird erneut gesendet, im häufigen Wiederholungsfalle gilt der Empfänger als nicht mehr erreichbar. Zu kurze oder zu lange Zeitintervalle verringern deutlich die Systemleistung.
- *Prüfsummen-Erstellung und -Auswertung*. Das wichtigste Prüfsummen-Verfahren, der *cyclic redundancy check* (*CRC*) beruht auf Modulo-2-Arithmetik (bitweises XOR). Im Protokoll ist eine Bitfolge der Länge $n + 1$ (Generator, Prüfpolynom) festgelegt. Der Sender fügt am Ende der zu sichernden Daten n Nullen an. Die entstehende Bitfolge P wird durch den Generator dividiert, der Divisionsrest R wird von P subtrahiert. Da dieser Rest höchstens n Bit lang und XOR-Addition mit XOR-Subtraktion gleich sind, bleiben die Nutzdaten unverändert. Bei fehlerfreier Übertragung kann der Empfänger die Nachricht *ohne Rest* durch den Generator teilen. Er streift dann die letzten n Bits ab und liefert die Nutzdaten aus. Da Protokollinformation und Nutzdaten meist nicht getrennt gesichert werden, müssen fehlerhafte Pakete völlig verworfen werden, um die Fehlinterpretation gestörter Protokollinformation zu vermeiden. Aus lückenhaft numerierten Datenpaketen kann der Empfänger später aber meist auf ausgefallene Pakete schließen und die nachfolgend beschriebenen Mechanismen anwenden.
- *Rücksetzen und Wiederholen*. In vielen verbindungsorientierten Protokollen kann die Empfangsinstanz per „negativer Quittung" die nochmalige Übertragung eines Datenpaketes anfordern. Bei schwerem Fehler und einfachen Protokollen wird die Verbindung auf den Zustand vor der betreffenden Übertragung *zurückgesetzt*. Besser ist es, nur das ausgefallene Paket zu *wiederholen* (selective repeat) und Pakete mit höherer Nummer in der Empfangsinstanz zwischenzuspeichern.

- *Vorwärts-Fehlerkorrektur.* In der Multimedia-Kommunikation und Hochgeschwindigkeitsnetzen können Rücksetzen und Wiederholen intolerable Stockungen erzeugen. Der Sender muß dann soviel Redundanz beifügen, daß der Empfänger selbsttätig ausgefallene Nutzdaten rekonstruieren kann. Da schnelle glasfaserbasierte Netze kaum Bitfehler erzeugen, bisweilen aber Pakete verlieren, muß die Rekonstruktion ganzer Pakete möglich sein.

Protokollmechanismen zur Längenanpassung (paarweise Protokollmechanismen)

- *Segmentierung / Reassemblierung.* Häufig sind Sendeinstanzen in der Lage, lange Datenpakete aufzuspalten; die Empfangsinstanz setzt die Fragmente vor der Auslieferung wieder zusammen. Segmentierung ist nötig bei Nutzdaten, die länger sind als das Maximum des Basisdienstes; sie kann die Leistung steigern, weil Knoten Pakete erst nach vollständigem Empfang auf der nächsten Teilstrecke weiterleiten.
- *Blockung / Entblockung und Verkettung / Trennung.* Für kurze Pakete kann das Verhältnis von Protokollinformation plus Quittungsverkehr zu Nutzbits sehr ungünstig werden, vor allem bei mehreren zu durchlaufenden Schichten. Bei Blockung werden mehrere Nutzdaten-Bitfolgen, womöglich für verschiedene Empfänger, in ein Protokoll-Paket zusammengepackt. Werden sie einzeln verpackt, dann konkateniert und in nur einer Dienstprimitive dem Basisdienst übergeben, spricht man von Verkettung.

Protokollmechanismen zur Systemleistungsanpassung

Flußsteuerung, Flußkontrolle. Dieser Mechanismus schützt die Empfangsinstanz und den empfangenden Nutzer vor Überlast (siehe Abschnitt 6.6.3).

Staukontrolle. In überlasteten Netzen kann der Durchsatz drastisch sinken. Explizite Staukontrolle, etwa durch eine feste Anzahl von umlaufenden Containern (die leer oder voll sein können), ist selten; häufig begrenzt das Netz statt dessen die Anzahl angebotener Verbindungen (in der Telefonie üblich) und den maximalen Flußsteuerungs-Sendekredit pro Verbindung (siehe Abschnitt 6.6.3).

- *Ratenkontrolle.* In der Kommunikation multimedialer Daten über geographisch große Distanzen ist die Datenmenge, die ein Sender zwischenspeichern kann, schnell erschöpft. Das Warten auf Quittungen führt dann zu Stop-and-Go-Betrieb. Statt eine Flußsteuerung müssen sich Sende- und Empfangsinstanz an eine maximale Datenmenge pro Zeiteinheit (Rate) halten.

Protokollmechanismen zur Übertragungsleistungsanpassung

- *Multiplexen / Demultiplexen.* Oft werden mehrere Verbindungen auf eine Verbindung der darunterliegenden Schicht abgebildet. Diese Vorgehensweise ist für Transportdienste typisch, deren Ende-zu-Ende-Verbindungen einzelne Prozesse bedienen, die aber über verbindungsorientierte Vermittlungsdienste meist zwischen zwei Rechnern nur eine Verbindung einrichten.
- *Teilung / Vereinigung.* Zur Durchsatzsteigerung kann auch eine Verbindung auf mehrere Verbindungen verteilt werden (z.B. die Nutzung mehrerer ISDN-Kanäle).

Nutzerzentrierte Mechanismen

- *Verbindungsklassen.* Dienste mit vielen Protokollmechanismen fassen Untermengen ihres Angebots zu Klassen zusammen; die Auswahl erfolgt beim Verbindungsaufbau.
- *Rechtevergabe.* Dieser Mechanismus wird vor allem in Schicht 5 eingesetzt; das Recht zum Absetzen gewisser Dienstprimitiven wird zeitlich an bestimmte Nutzer gebunden.
- *Dienstgüte-Verhandlung.* Dienstgütemerkmale (*Quality-of-Service, QoS*) werden in Kapitel E2 behandelt. Beispiele sind die Restfehlerwahrscheinlichkeit und der Durchsatz. Sie können „ausgehandelt" werden, indem der Initiator bei der Verbindungs-

aufbau-Anforderung (Con.Req) Wunschgüten angibt, die von Basisdienst und Beantworter im Verlauf des Verbindungsaufbaus akzeptiert oder reduziert werden. Das Resultat wird dem Initiator in der Verbindungsaufbau-Bestätigung mitgeteilt.

Ausgehend von den bisher dargestellten allgemeingültigen Prinzipien und Schichten werden im Folgenden die wichtigsten Aspekte in Bezug auf die Bitübertragungs-, Sicherungs-, Vermittlungs- und Transportschicht als Bestandteil des Internets dargestellt.

6.5 Netztopologien

Netze können grob unterteilt werden in *lokale Netze* (*LAN*: *local area network*) und *Weitverkehrsnetze* (*WAN*: *wide area network*). Ein LAN verbindet Stationen an einem Standort miteinander, seine Ausbreitung liegt im Bereich von Metern bis einigen Kilometern. LANs können in verschiedenen Topologien realisiert werden.

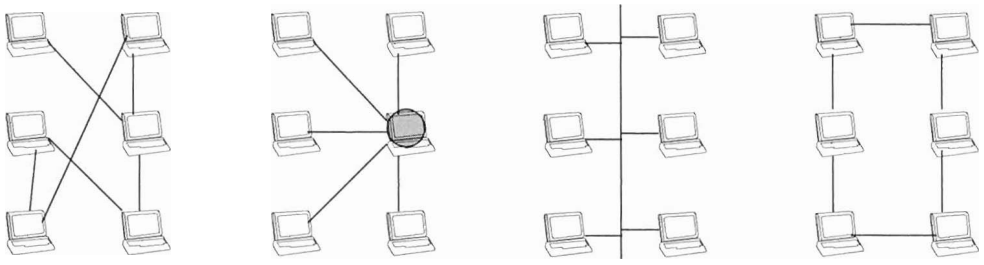

Bild 6 Beispiele für LAN-Topologien: ein irreguläres Netz aus Punkt-zu-Punkt-Strukturen, ein Stern mit einer zentralen Vermittlungsstation und geteilt genutzte Verbindungen in Bus- oder Ringanordnung

In Bild 6 sind vier LAN-Topologien zu sehen; bei der Punkt-zu-Punkt-Struktur sind jeweils genau zwei Stationen (d.h. zwei Knoten oder zwei Endsysteme oder ein Knoten und ein Endsystem) physisch miteinander verbunden. Jede LAN-Topologie hat spezielle Eigenschaften. So sind Bus- und Ringtopologie besonders einfach zu verwalten und zu erweitern; sie benötigen aber eine vergleichsweise komplexe Zugangssteuerung, damit mehrere Stationen die sie verbindenden physischen Verbindungen gemeinsam nutzen können. Eine Sternstruktur zeichnet sich durch die sternförmige Anordnung vieler Punkt-zu-Punkt-Strukturen aus. Das impliziert einen einfachen Zugriff und möglicherweise hohen Durchsatz, es erfordert dazu als entsprechend leistungsfähige zentrale Station und oft spezielle Hardware, sogenannte *Switches*. Irreguläre Strukturen sind schwerer zu verwalten und zu nutzen; sie sind nur dann angebracht, wenn spezielle Anforderungen diese Verbindungen erfordern. Beispiele hierfür sind komplett oder sehr redundant vernetzte Rechner-Cluster.

Bild 7 zeigt ein Weitverkehrsnetz, WAN, das Stationen über ein sehr weites Gebiet verbinden kann. Die einzelnen Stationen sind zumeist in Subnetzen zusammengefaßt, die jeweils aus einem oder mehreren LANs aufgebaut sind.

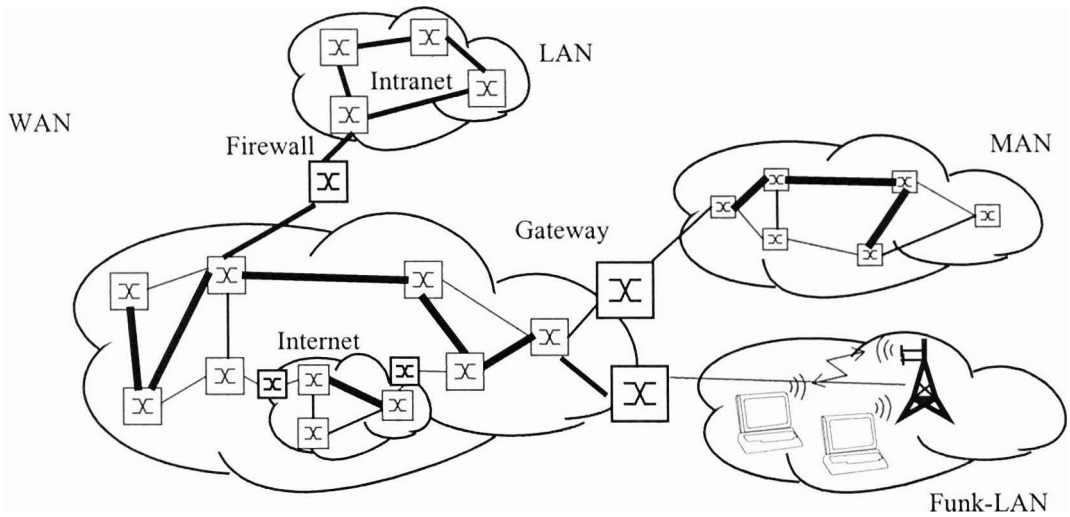

Bild 7 Ein Weitverkehrsnetz, das lokale Subnetze über Vermittlungsrechner (Gateways und Firewalls) verbindet

Oft werden weitere Kategorien von Netzen benannt. Namen wie *MAN* (*metropolitan area network*), *CAN* (*car area network*) oder *BAN* (*body area network*) bezeichnen zumeist LANs mit besonderen Eigenschaften für spezielle Anwendungs- oder Ausbreitungsgebiete. MANs zeichnen sich beispielsweise durch ihre besondere physische Größe im Bereich von bis zu 100 Kilometern aus. Meistens sind sie als Ringe implementiert.

6.6 Zugangssteuerungsverfahren

Den gemeinsamen Zugriff mehrerer Stationen auf ein Netz regelt ein Zugangssteuerungsverfahren; es ist Bestandteil der Sicherungsschicht. Ein solches Verfahren muß vor allem bei gemeinsam benutzten Übertragungsmedien - Busse oder Funk - den alleinigen oder ungestörten Zugriff während der Übertragung sichern und für Fairneß des Zugangs für die verschiedenen Stationen sorgen.

Die Zugangssteuerungsverfahren unterscheiden sich, je nachdem die Netze aus Punkt-zu-Punkt-Strukturen aufgebaut sind oder aus gemeinsam benutzten physischen Verbindungen bestehen. Bei gemeinsam benutzten physischen Verbindungen werden meist zwei verschiedene Methoden eingesetzt: Bei gemeinsamem Bus oder gemeinsamem drahtlosen Medium wird CSMA/CD benutzt (Abschnitt 6.6.2), bei Ringanordnungen werden Marken (*token*) im Kreis weitergereicht, um etwa die Sendeberechtigung zu bestimmen. Zu den letzteren Verfahren gehören zum Beispiel der *Tokenring* und *FDDI*.

6.6.1 Standards

Das wichtigste Standardisierungsgremium für Zugangssteuerungsverfahren ist das IEEE Projekt 802, genannt „LAN/MAN Standards Committee“ (*LMSC*). Diese Gruppe arbeitet in Koordination mit nationalen und internationalen Organisationen, um die Zugangssteuerung und hiermit zusammenhängende Problemstellungen zu standardisieren. Es

werden somit Standards bezüglich der OSI-Schichten 1 und 2 bearbeitet. Einige der Ergebnisse sind inzwischen von der International Standards Organization (*ISO*) als internationale Standards veröffentlicht.

Das LMSC gliedert sich in Untergruppen, die weiter spezialisierte Themen behandeln. Die Namen dieser Gruppen benennen zugleich die entsprechenden Protokolle. Einige wichtige sind:

- IEEE 802.3 – CSMA/CD Working Group (standardisiert das Ethernet)
- IEEE 802.5 – Token Ring Working Group
- IEEE 802.11 – Wireless LAN (*WLan*)

6.6.2 Busbasierte lokale Netze

Die in lokalen Netzen mit Abstand am meisten verwendeten Techniken benutzen Busse. Dazu gehören insbesondere das weithin eingesetzte Ethernet und lokale Funknetze.

CSMA/CD. Ein grundlegendes Verfahren dieser Netze ist CSMA/CD (*carrier sense multiple access collision detection*): Bevor eine Datenübertragung beginnt, wird der Zustand des Netzes von der sendewilligen Station geprüft. Überträgt keine andere Station Daten, darf die sendewillige Station zu diesem Zeitpunkt versuchen zu senden. Eine wichtige Fähigkeit der Stationen in einem solchen Netz ist es, *gleichzeitig horchen* und *senden* zu können. „Horchen" bedeutet Lesen aller an der eigenen Station verfügbaren Daten des Netzes. Versuchen mehrere Station gleichzeitig, eine Datenübertragung zu beginnen, so überlagern sich die Daten auf dem Bus. Sendende Stationen erkennen das an den verfälschten eigenen Daten auf dem Bus. Jede Station versucht anschließend nach einer zufällig gewählten Zeit (während eines sich je nach Anzahl der Kollisionen ändernden Zeitintervalls), die Übertragung von neuem zu beginnen.

Ethernet. Ethernet ist ein CSMA/CD-Verfahren, das heutzutage fast alle anderen drahtgebundenen LANs verdrängt: Es ist sehr günstig und flexibel einsetzbar. Das heute verfügbare Ethernet bietet eine Datenrate von 10 Mbit/s (IEEE 802.3a bis 802.3j und 100 Mbit/s (*Fast Ethernet*, IEEE 802.3u, 802.3y). Es existieren seit einiger Zeit auch Varianten des Ethernets mit 1 Gbit/s (*Gigabit Ethernet*, IEEE 802.3z bis 802.3ab) und 10 Gbit/s. Ab 1 GBit/s werden die Netze für *Backbones* (schnelle Verbindungen zwischen LANs) verwendet.

Beim Aufbau eines Hochgeschwindigkeits-Ethernets (von 1 GBit/s aufwärts), das auf dem ursprünglichen CSMA/CD basiert und über eine Glasfaser läuft, ist die zuverlässige Kollisionserkennung schwierig. Ein weiteres Problem ergibt sich auf Grund der großen Datenraten, da hierdurch die Zeit zur Übertragung von Daten sehr klein wird, was in einer drastisch reduzierten Ausdehnung eines möglichen CSMA/CD-Netzes resultiert. Deshalb werden bei Gigabit-Ethernet andere Netztopologien und Ergänzungen zu CSMA/CD eingesetzt.

Lokale Funknetze. Lokale Funknetze wurden erstmalig 1997 im Standard IEEE 802.11 definiert, zuerst im Bereich von 2.4 GHz (IEEE 802.11b), später mit 5 GHz (802.11.a) für Übertragungsraten von 5.5 Mbit/s bis 54 Mbit/s eingesetzt. Diese Netze benutzen mehrere Trägerfrequenzen gleichzeitig durch das Verfahren *OFDM* (*orthogonal frequency division multiplex*). Der Zugriff auf diese Frequenzen wird dabei nicht nur durch CSMA/CD geregelt, sondern zusätzlich werden Datenpakete an nur einen Empfänger nach dem Empfang bestätigt.

6.6.3 Flußsteuerung

Das Problem, daß ein schneller Sender mit seinem Datenfluß einen langsameren Empfänger überfordert, tritt zwischen benachbarten Stationen auf. Somit muß dieser Datenfluß (eine Folge von Dateneinheiten, Rahmen, Paketen) „abgebremst" oder wieder „beschleunigt" werden; hierzu kommen Mechanismen zur Flußsteuerung zum Einsatz. Flußsteuerung muß sowohl zwischen unmittelbar benachbarten Stationen erfolgen (auf der Sicherungsschicht) und zwischen miteinander kommunizierenden Endsystemen (auf der Transportschicht). Damit wird dann weder die Station als Ganzes (Flußsteuerung auf der Sicherungsschicht) als auch eine Anwendung auf einem Endsystem (Flußsteuerung auf der Transportschicht) überflutet.

Ein wichtiges Problem dabei ist, daß der Sender alle Pakete vorrätig halten muß, solange ihre erfolgreiche Übertragung nicht gesichert ist. Man löst es mit dem Sliding-Window- und dem Kredit-Mechanismus.

Sliding-Window-Mechanismus. Hier besitzen Sende- und Empfangsstation je einen Ring von Puffern, um Datenpakete zwischenzuspeichern, deren Übertragung noch nicht bestätigt wurde. Puffer, die bestätigte Daten enthalten, werden ihrer Reihenfolge im Ring nach wieder benutzt. Dadurch entsteht ein Fenster von gesendeten Datenpaketen, die gleichzeitig unbestätigt sein können.

Die einfachste Art von Sliding-Window ist *Stop-and-Wait*: Sie benutzt nur einen einzigen Puffer. Das bedeutet, daß jeweils ein Datenpaket gesendet und gespeichert wird, bis die sendende Station seinen Empfang bestätigt. Das Paket wird nochmals gesendet, wenn die Bestätigung nicht innerhalb einer festgelegten Zeit eintrifft.

Bei einem Fenster, das größer als ein Datenpaket ist, werden die Pakete nach den Puffern, in denen sie beim Sender gespeichert waren, numeriert. Der Empfänger bestätigt unter Angabe dieser *Sequenznummer*. Auch hier gilt: Wird ein Paket nicht innerhalb einer bestimmten Zeit bestätigt, sendet es der Sender von neuem. Hier ist es aber möglich, eine Sequenz von direkt aufeinander folgenden Paketen durch eine einzige Quittung zu bestätigen. Um stets eine eindeutige Zuordnung der Sequenznummern zu gewährleisten, muß die Anzahl dieser Nummern (also der Puffer beim Sender) mehr als doppelt so groß sein wie das Sendefenster.

Neben dem Nachteil, daß für jede Verbindung hinreichend viele Puffer benötigt werden, hat diese Methode den Vorteil, daß mit steigender Anzahl der Sendepuffer die Methode der Bestätigung direkt aufeinander folgender Pakete durch eine einzige Quittung zunehmend Speicher sparen kann.

Kreditmechanismus. Hier fordert zu Beginn einer Übertragung der Sender beim Empfänger eine jeweils individuell zu bestimmende Pufferkapazität an; der Empfänger gewährt diese oder bestätigt eine dem Empfänger genehme geringere Pufferkapazität (als Kredit bezeichnet). Von nun an kann der Sender entsprechend dem Kredit vermindert um die jeweils ausstehende Anzahl der Quittungen Daten versenden. Der Empfänger bestätigt korrekt empfangene Pakete und gewährt daran angepaßt einen weiteren Kredit (in Abhängigkeit von seinem noch verfügbaren Puffer).

6.7 Wegeleitverfahren

Wegeleitverfahren (*routing*) dienen als Bestandteil der Vermittlungsschicht dazu, daß Pakete in einem Netz einen günstigen Weg zum Empfänger finden oder ein günstiger Weg Ihnen vorgeschrieben wird. Im Internet dienen sie dazu, daß die Pakete des verbindungs-

losen *Internet-Protokolls* (*IP*) ihren Weg zum Endsystem finden. Das Endsystem selber wird beim Internet über die heutige 32 Bit lange Internetadresse adressiert.

Wegeleitverfahren unterscheiden sich nach der Art der angestrebten Kommunikation: *Punkt-zu-Punkt* (*unicast*) als die meistverwendete Beziehung zwischen genau zwei Kommunikationspartnern, *Punkt-zu-Mehrpunkt* (*multicast*) mit der Kommunikation zwischen einem und einer Gruppe vom Partnern und *Rundsenden* (*broadcast*) als Kommunikation zwischen einem und allen anderen. Hier werden nur die grundlegenden Verfahren (primär für eine Punkt-zu-Punkt-Kommunikation) erläutert.

6.7.1 Überflutungsverfahren (Flooding)

[B3]

Das Überflutungsverfahren, *Flooding*, ist ein sehr einfaches Wegeleitverfahren. Es ändert seine Vorgaben nicht mit veränderter Netzsituation und ist damit statisch. Ein statisches Wegeleitverfahren wird, zusammen mit eventuell notwendigen Daten, bei Inbetriebnahme von den Netzknoten (hier von den Vermittlungsrechnern oder Routern) geladen und dann nicht mehr verändert.

Der Mechanismus der Überflutung besteht aus den folgenden beiden Regeln:

1 Jede Vermittlungsstation im Netz sendet jedes Paket, das sie empfängt (und das nicht für sie selbst ist), an jede ihrer unmittelbaren Nachbarstationen weiter. Pakete werden hierdurch dupliziert.

2 Es wird kein Paket an jeweils die Nachbarstation, von der es empfangen wurde, gesendet.

Diese Regeln können durch weitere Vorgaben ergänzt werden, um die hohe Anzahl von Duplikaten eines Pakets im Netz einzuschränken:

- Jedes Paket trägt einen Schrittzähler (*Hop Counter*) der mit jedem Durchlaufen durch einen Knoten (*Hop*) um eins dekrementiert wird. Pakete mit einem Hop Counter von Null werden von dem Knoten verworfen.
  Die Wirkung dieser Regel kann, wenn Wissen über die Entfernung zur Zielstation vorhanden ist, durch jeweils geeignete Initialisierung dieses Zählers verbessert werden.
- Pakete werden nur „in Richtung des Zieles" übertragen. Das setzt Wissen über die geographische Netzstruktur und eine passende Topologie voraus.

Das Überflutungsverfahren ist extrem robust. Es kann garantieren, daß eine Information alle verbundenen Stationen erreicht, und es verwendet in seiner einfachsten Form kein Vorwissen über die Netztopologie. Daher eignet es sich sehr gut zur Verbreitung von Kontrolldaten zwischen den Knoten (etwa für komplexere Routingverfahren) und zur Verwendung während der Initialisierungsphase anderer Wegeleitverfahren.

6.7.2 Distanzvektorverfahren

Im Gegensatz zum Überflutungsverfahren ist das Distanzvektorverfahren dynamisch; es paßt sich an den Netzwerkzustand an. Sein einfaches Prinzip besteht darin, daß benachbarte Vermittlungsstationen Leitweginformationen austauschen. Im Einzelnen wird dies durch die folgenden Regeln erreicht:

1 Jede Station kennt die „Entfernung" zu ihren Nachbarn. Die Entfernungen werden als Zahlenwerte angegeben. Die Metrik kann etwa aus einer Gewichtung der Anzahl der Teilstrecken (zu den unmittelbaren Nachbarn immer eins), aus der Verzögerungszeit

(bestimmt durch die Sendung von Echopaketen zu den Nachbarn), aus der Länge der jeweiligen Sendewarteschlange und anderen Kennzahlen bestehen.

2 Jede Station sendet ihre Liste (der ihr bekannten Entfernungen zu allen ihr bekannten Stationen) an ihren Nachbarn. Zur Funktionsfähigkeit des Verfahrens ist es wichtig, die Häufigkeit dieses Vorgangs geeignet zu wählen und auch die Menge der zu übertragenden Information zu begrenzen. Typisch sind hier eine Wiederholung alle 30 Sekunden und eine Eingrenzung der ausgetauschten Entfernungsinformationen auf Ziele, die mit maximal 15 Hops zu erreichen sind.

3 Jede Station empfängt die Liste ihrer unmittelbaren Nachbarn und berechnet mit dieser Liste und den eigenen Messungen eine neue, eigene Liste.

   1 Dazu werden zuerst alle Entfernungseinträge der empfangenen Listen aller Nachbarstationen um die jeweilige Entfernung der Stationen vergrößert.

   2 Es wird zu jedem Ziel der Eintrag mit der geringsten resultierenden Entfernung festgehalten. Außerdem wird die Station vermerkt, aus deren Liste dieser Eintrag übernommen wurde. Diese Station ist dann jeweils der Ausgangspfad in Richtung des Ziels des jeweiligen Eintrags.

Die Gruppe der Distanzvektorverfahren enthält den verteilten Bellman-Ford-Algorithmus, den Ford-Fulkerson-Algorithmus und das Routing Information Protocol (*RIP*). RIP wurde im Internet und seinem Vorgänger ARPANET anfangs benutzt.

6.7.3 Linkzustandsverfahren

Im Internet wurde das Distanz-Vektor-Verfahren aus dem vorigen Abschnitt vom Linkzustandsverfahren abgelöst. Das Linkzustandsverfahren mißt wie sein Vorgänger in einer Entfernungsmetrik die Distanz zu seinen unmittelbaren Nachbarn und verteilt diese Information:

1 Es wird eine Liste mit den Entfernungen und Adressen zu den unmittelbaren Nachbarstationen erzeugt.

2 Die erzeugte Liste wird zu einem Paket aufbereitet und an alle anderen Vermittlungsstationen (nicht nur die Nachbarstationen) verteilt.

3 Aus der eigenen Liste und den empfangenen Listen möglichst vieler anderer Stationen werden die besten Leitwege und die entsprechenden Ausgangspfade bestimmt.

Eine Variante des Linkzustandsverfahrens ist Intermediate System – Intermediate System (*IS-IS*). Es wurde von der Firma DEC für ihr DECNET entwickelt, wurde als *ISO CLNP* in NSFNET (dem Forschungsnetz der US-amerikanischen National Science Foundation in den späten siebziger Jahren) und in einer weiteren Variante (*NLSP*) von der Firma Novell in ihrem Produkt Netware eingesetzt.

Eine weitere wichtige Variante des Linkzustandsverfahrens ist das im Internet heute eingesetzte Open Shortest Path First (*OSPF*).

6.7.4 Weitere Verfahren

Spanning Tree. Beim statischen Spanning-Tree-Verfahren wird jedem Empfänger der Baum der minimalen Pfadlänge zwischen dem Sender und Empfänger zugeordnet. Die Minimalität bestimmt sich auch hier nach Metriken, die sich neben dem geographischen Abstand und der Anzahl der Hops auch nach den Kommunikationskosten und Verzögerungen richten, welche wiederum von den verfügbaren und benutzten Übertragungsraten abhängen.

Backward Learning - Algorithmus. Ohne zusätzliche Kommunikation kommt der Backward Learning-Algorithmus aus: Hier verwendet jede Station Informationen, die in den empfangenen Datenpaketen implizit über das Netz enthalten sind. Eine Grundvoraussetzung dieses Verfahrens ist die Symmetrie von Verbindungen: Wenn ein Paket von einer bestimmten Quelle innerhalb einer bestimmten Anzahl Hops über die Nachbarstation S ankommt, geht das Verfahren davon aus, daß ein Paket in der Gegenrichtung die obige Quelle über Station S in ebensovielen Hops erreichen kann. Eine weitere Einschränkung des Verfahrens ist seine eingeschränkte Adaptivität: Jede Station merkt sich das Optimum aller ermittelten Pfade ohne Wissen darüber, ob diese Pfade immer noch existieren.

Mehrfach-Leitwegbestimmung (Multipath Routing). Um die maximale Übertragungsrate der verfügbaren Wege gleichmäßiger auszunutzen und so höhere Übertragungsraten und höhere Zuverlässigkeit zu erreichen, erlaubt die Mehrfach-Leitwegbestimmung nicht nur einen, sondern eventuell mehrere Pfade zum Ziel. Die Datenpakete werden dann mit bestimmten Häufigkeiten abwechselnd über die verschiedenen Pfade weitergesandt.

Hierarchische Leitwegbestimmung. Nach dem Ansatz „teile und herrsche" geht die hierarchische Leitwegbestimmung vor. Sie unterteilt das gesamte Netz hierarchisch in Regionen; jede Station muß nur noch die Ziele innerhalb der eigenen Region und eine Station der enthaltenden Region kennen. Dadurch kann die Größe der notwendigen in einem Knoten zu speichernden Daten (die *Routing-Tabellen* und damit auch der Entscheidungsaufwand der Verarbeitung in dem Knoten) unabhängig von der tatsächlichen Netztopologie begrenzt werden.

6.8 Transportprotokolle

Die Transportschicht stellt Prozesse auf kommunizierenden Endsystemen in Beziehung. In vielen Betriebssystemen sind diese Dienste der Transportschicht als sogenannte *Sokkets* implementiert, die von einem Prozeß als Abstraktion einer Verbindung zu einem anderen, möglicherweise entfernten, Prozeß angefordert werden können. Für die Erstellung einer solchen Kommunikationsbeziehung ist im Idealfall nur die Adresse des Zielprozesses, bestehend aus Endsystemadresse (im Internet die Internetadresse) und Prozeßport, notwendig. Zusätzliche Konfigurationsangaben können Aspekte wie Puffergrößen und Wartezeitangaben betreffen.

6.8.1 User Datagram Protocol - UDP

Das *User Datagram Protocol* (*UDP*) stellt einen sehr einfachen Dienst zur Verfügung: Ein Sendeprozeß kann einzelne Pakete, sog. *Datagramme*, absenden; Empfangsprozesse können einzelne Pakete empfangen. UDP stellt keine Verbindung her, und es gibt keinerlei Garantien, Bestätigungen, Flußkontrolle oder Reihenfolgeerhaltung. Falls eine dieser Eigenschaften benötigt wird, so muß sie von der benutzenden Anwendung realisiert werden.

Dementsprechend ist dieses Protokoll sehr einfach und benötigt zu seiner Ausführung vergleichsweise wenig Betriebsmittel (d.h. wenig Prozessorkapazität, Speicherplatz und geringe Datenübertragungskapazität). Sein Einsatz ist insbesondere bei Echtzeitübertragung sinnvoll, wo Daten schnell ungültig werden (siehe Abschnitt 6.8.3) und bei jeglicher Übertragung an viele Empfänger.

6.8.2 Transmission Control Protocol - TCP

Im Gegensatz zu UDP ist das *Transmission Control Protocol* (*TCP*) verbindungsorientiert. Nach dem Aufbau einer Verbindung wird die korrekte und geordnete Übertragung der Pakete garantiert, d.h., bei Paketverlusten oder zu langen Verzögerungen empfangen die beiden beteiligten Prozesse Fehlersignale. Das erlaubt es, TCP-Verbindungen zur Übertragung von Byte-Folgen, *nicht* als isolierte Pakete zu benutzen. Die Verbindungen werden immer bidirektional (Voll-Duplex) eingerichtet und verwendet; damit können also gleichzeitig Daten in beiden Richtungen zwischen den Endsystemen übertragen werden.

Startet ein Prozeß eine TCP-Verbindung, so erfolgt ein aufwendiger dreiphasiger Verbindungsaufbau mit (1) einer Verbindungsanfrage (bei TCP *SYN* genannt) die eine initiale Sequenznummer enthält, und (2) einer Bestätigung (bei TCP *ACK* genannt) vom Empfänger und dann (3) wiederum einer Bestätigung vom Sender. Diese drei Phasen sind zur Vermeidung von Fehlern (Verlust und Verdopplung von *SYN*-Paketen) notwendig. Solange eine Verbindung besteht, sorgt der Sliding-Window-Mechanismus aus Abschnitt 6.6.3 für die vollständige und geordnete Übertragung. Bei Fehlern werden Pakete erneut gesendet. Auch der Abbau einer Verbindung wird explizit signalisiert und bestätigt.

TCP erhöht aus Fairneßgründen die maximale Menge der zu einem Zeitpunkt zu sendenden Daten in Abhängigkeit von der Dauer der Verbindung bis zu einem auftretenden Fehlerfall. Somit ist es nicht möglich, einen kontinuierlichen Datenstrom zu übertragen, wenn die Datenrate über einem sehr niedrigen Schwellwert liegt. Dieser Effekt (*slow start*) führt dazu, daß der als TCP-artig (*TCP-friendly)* bezeichnete Datenstrom nur noch bedingt für die Übertragung von Audio- und Videodaten verwendet werden kann. TCP eignet sich zur fehlerfreien Übertragung von Daten, wie es bei einem Dateitransport oder elektronischer Post erwünscht ist.

6.8.3 Quasi Echtzeittransport - RTP

RTP (*real-time transport protocol*), [RFC1889] wird heute als de-facto-Standard für Video- und Audio-Datenübertragung im Internet angesehen. RTP ist allerdings (genau betrachtet) weder ein Echtzeit- noch ein Transportprotokoll. Es ist eher ein anwendungsnahes Protokoll, das einige für die Audio- und Videokommunikation erforderlichen Eigenschaften sicherstellt.

RTP ermöglicht eine Charakterisierung der Dateninhalte, das Einfügen von Sequenznummern und Zeitstempeln, eine Überwachung des Datenempfangs, und es ermöglicht eine Multicast-Kommunikation (falls darunterliegende Schichten sie auch als Dienst anbieten). RTP verwendet im allgemeinen selber UDP als Transportprotokoll: Es werden somit die RTP-Pakete als Nutzdaten in UDP-Pakete verpackt und übermittelt. Die Verwendung von UDP erlaubt ein Multiplexen, erzwingt keinen TCP-artigen Datenverkehr und veranlaßt selber keine Übertragungswiederholungen (die der Einhaltung der Zeitgarantien nicht förderlich wären). RTP wird stets zusammen mit dem *RTCP* (*RTP control protocol*) verwendet. RTCP ist für die Signalisierung zur Überwachung der Qualität von den übertragenen RTP-Paketen verantwortlich.

6.9 Zusammenfassung und Weiterführendes

Die zunehmende Anzahl der Kommunikationskomponenten, -protokolle und -mechanismen verdeutlicht die Komplexität dieses Bereichs der Informatik (der hier immer zusammen mit den verteilten Systemen zu sehen ist). Kommunizierende Systeme werden mobil, sie werden allgegenwärtig, und in Zukunft wird eine Vernetzung oft spontan erfolgen. Neuere Entwicklungen sind u.a. bei solchen „Ad-hoc-Netzen", einer geeigneten Dienstgüte, der Mobilität als auch im Hinblick auf eine dem Menschen adäquate (nahtlose) Kommunikation zu erwarten.

Abschließend sei an dieser Stelle Herrn Dr. M. Liepert für seine Beiträge und kritischen Anmerkungen gedankt.

Literatur

IEEE Project 802 LMSC.: Overview and guide to the IEEE 802 LMSC. http://grouper.ieee.org/groups/802/overview2000.pdf, IEEE, 2000

Kurose, J.F.; Ross, K.W.: Computer networking: a top-down approach featuring the Internet. Boston San Francisco New York: Addison Wesley 2001

Lockemann, P. C.; Krüger, G.; Krumm, H.: Telekommunikation und Datenhaltung. München Wien: Carl Hanser Verlag 1993

Mühlhäuser, M.; Schill, A.: Software Engineering für verteilte Anwendungen: Mechanismen und Werkzeuge. Berlin: Springer-Verlag 1992

Peterson, L. L.; Davie B. S.; Clark D.: Computer networks: a systems approach. 2nd edition, San Francisco: Morgan Kaufmann 1999. Deutsch: Computernetze; ein modernes Lehrbuch. Heidelberg: dpunkt-Verlag, 2000 (Anmerkung: Sehr gute Übersetzung)

Steinmetz, R.: Multimedia-Technologie: Grundlagen, Komponenten und Systeme. Berlin: Springer 2000

Tanenbaum, A. S.:Computer Networks; Third Edition. Upper Saddle River: Prentice Hall 1996 Deutsch: Computer Netzwerke. 3. Aufl., Upper Saddle River: Prentoce Hall 2000 (Anmerkung: Englische Version trotz guter Übersetzung vorzuziehen)