

## Vorlesung Telematik

Prof. Dr. Dieter Hogrefe  
Dipl.-Inf. Michael Ebner  
Dr. Xiaoming Fu

Wintersemester 02/03

## Literatur

J.F. Kurose, K.W. Ross; Computer Networking: A Top-Down Approach Featuring the Internet Addison-Wesley Publishing Company, Wokingham, England, 2001.

L.L. Peterson, B. S. Davie: Computernetze, dPunkt, 2000

J. Ellsberger, D. Hogrefe, A. Sarma: SDL - Formal Object-Oriented Language for Communicating Systems, Prentice Hall, 1997.

## Inhalt

### Teil 1

- 1 Einführung
- 2 Beschreibung von Kommunikationssystemen mit SDL
- 3 Das ISO-Referenzmodell OSI
- 4 Netztopologien

Telematik (WS 02/03)

Telematik (WS 02/03)

## 1 Einführung

### Aufgaben von Rechnernetzen

#### Datenverbund:

- logische Kopplung von räumlich getrennten Datenbeständen
- nachrichtentechnische Übertragungsmöglichkeit
- Konsistenz- und Aktualitätssicherung mit höheren Systemkonstrukten

#### Funktionsverbund:

- Integration von Geräten und Systemen zur Realisierung spezieller Funktionen
- alle Benutzer können die speziellen Funktionen mitbenutzen

#### Verfügbarkeitsverbund:

- Bereitstellung von Mindestleistung unabhängig vom Ausfall einiger Komponenten
- fehlertolerierende Systeme

#### Leistungsverbund:

- Nutzung von (geographisch) verteilten Komponenten zur Lösung eines Problems
- Ausnutzung paralleler Verarbeitung

#### Lastverbund:

- Einsatz von Ressourcen schwach belasteter Rechner zur Entlastung stark belasteter
- Umverteilung von Aufträgen

Telematik (WS 02/03)

Telematik (WS 02/03)

**Telekommunikation:**

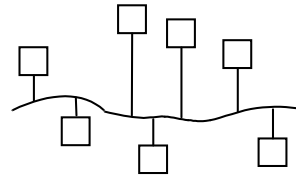
- Austausch von Information aller Art (Telex, Teletex, Telephon, Telefax)
- Dateien
- E-Mail
- Bewegtbild

**Arten von Rechnernetzen****PAN:** (Personal Area Network)

- Ausdehnung 10m
- Typische Übertragungsrate: 1Mbit/sec
- Z.B. Kabelersatz (Bluetooth)

**LAN:** (Local Area Network)

- meist Diffusionsnetz



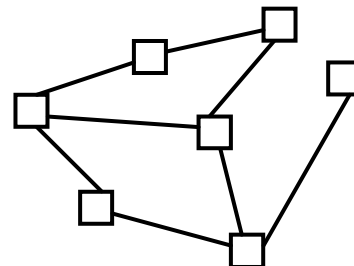
- Ausdehnung: bis 10 km
- typische Übertragungsrate: 100 Mbit/sec
- meist private Betreiber auf privatem Gelände
- bekannte Beispiele:
  - WLAN
  - Ethernet

**MAN:** (Metropolitan Area Network)

- wie LAN
- decken den Kommunikationsbedarf innerhalb von Städten und Ballungszentren
- wie LAN, wird aber nicht vom Anwender selbst betrieben sondern von einem Provider
- meist **Diffusionsnetz**
- Ausdehnung: bis 100 km
- z.B. FDDI, DQDB

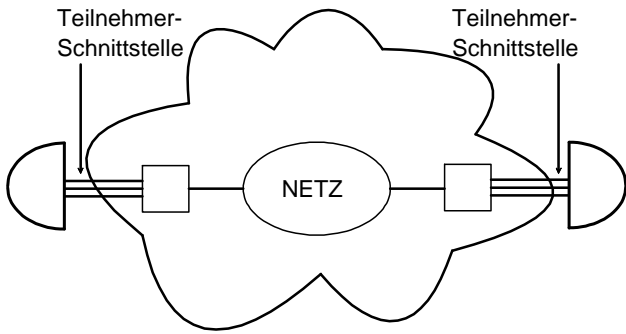
**WAN:** (Wide Area Network)

- klassisches Netz zur Verbindung getrennter Rechenanlagen
- unterliegt keiner räumlichen Begrenzung
- ev. Verwendung von Satelliten zur Überbrückung großer Entfernungen
- arbeiten mit anderen regionalen WANs oder LANs zusammen (oft nur Backbone)
- paketvermittelndes **Teilstreckennetz** oder
- leitungsvermitteltes Netz



- internationale Standards sollen den Übergang zwischen verschiedenen WANs erleichtern, z.B. ATM



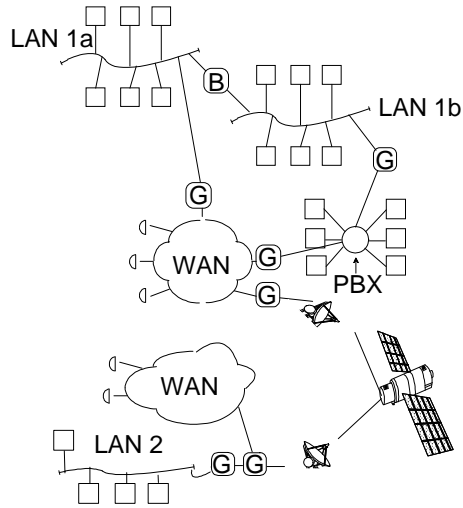


WAN aus Sicht eines Teilnehmers

### Verbindung unterschiedlicher Netzen

Klassifikation von Verbindungen:

- Verbindung von **homogenen** Netzen über **Bridges**
- Verbindung von **heterogenen** Netzen über **Gateways**

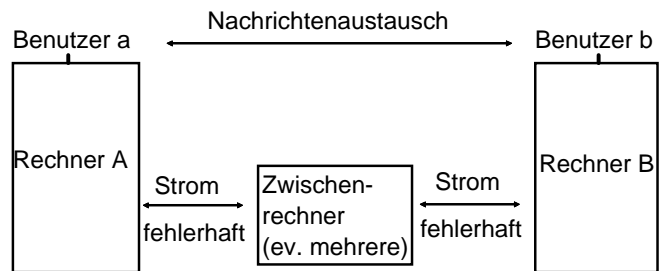


1-19

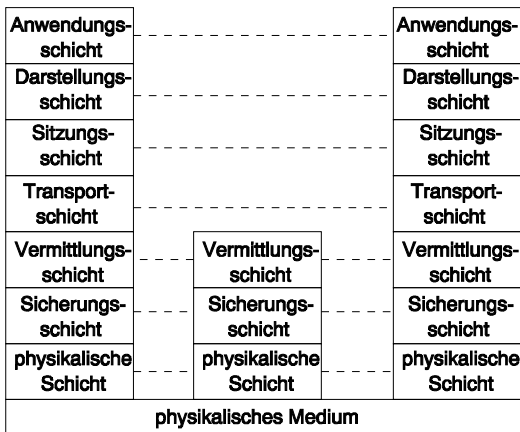
Ideale Benutzersicht:

- Benutzer operiert auf abstrakten Objekten (Dateien, Programmen)
- Benutzer hat keine Einsicht in Implementierungsdetails
- Benutzer weiß nicht, ob ein Objekt lokal ist oder darauf über ein LAN, WAN zugegriffen wird

### Die geschichtete Kommunikationsarchitektur



Zur Bewältigung der Komplexität: Einteilung in Abstraktionsschichten (hier gemäß OSI-Referenzmodell)



## Die Aufgaben der 7 Schichten im OSI-Referenzmodell

### Physikalische Schicht (Physical Layer):

- Übertragung von Bits über ein physikalisches Medium

### Sicherungsschicht (Data Link Layer):

- sichere Übertragung von Datenblöcken (Bitketten fester Länge)

### Vermittlungsschicht (Network Layer):

- Wegewahl durch das Netz
- Flußkontrolle

### Transportschicht (Transport Layer):

- sicherer Transport von Nachrichten durch ein Netz in korrekter Reihenfolge

### Sitzungsschicht (Session Layer):

- Auf- und Abbau von Sitzungen zwischen zwei Benutzern im Netz

### Darstellungsschicht (Presentation Layer):

- Herstellung kompatibler Datenformate zwischen zwei Anwenderprogrammen

### Anwendungsschicht (Applikation Layer):

- enthält die Anwendungsprogramme

## Vorteile offener Rechnernetze

- Herstellerunabhängigkeit
- Zukunftssicherheit

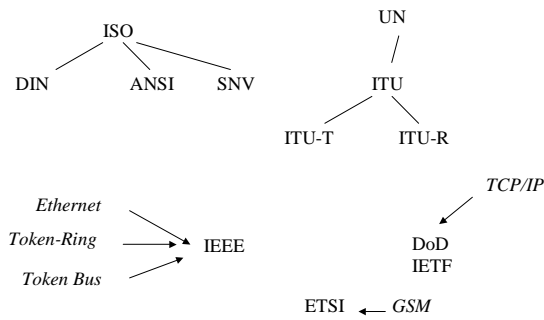
OSI - Basic Reference Model for Open System Interconnection

**offen** = auf öffentlich zugänglichen, international abgestimmten, herstellerunabhängigen Standards basierend

Standards definieren Regeln und Formate für den Informationsaustausch zwischen beteiligten Rechnern (**Kommunikationsprotokolle**).

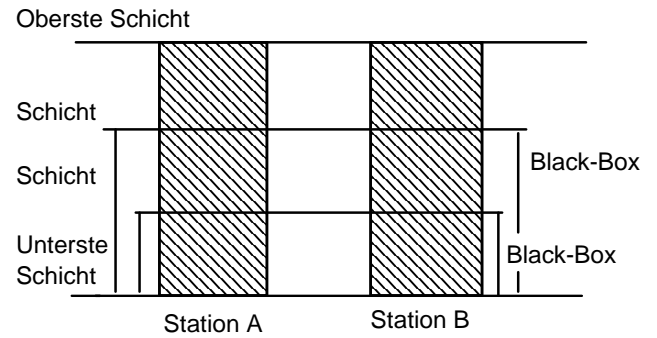
Nur das nach außen sichtbare Verhalten eines Systems wird mit einem OSI-Standard festgelegt.

Wer macht Standards?



Einige Begriffsdefinitionen

Jede einzelne Station (Netzknoten) ist aus einer Folge von Schichten (Layers) zusammengesetzt

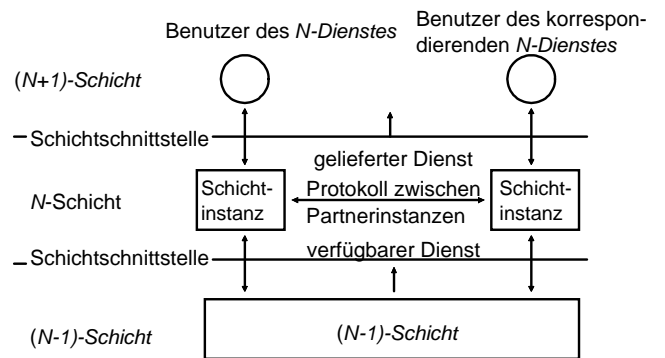
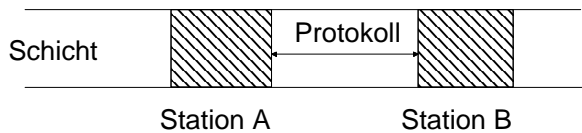


Eine Schicht wird durch mehrere Instanzen (Entities) realisiert.

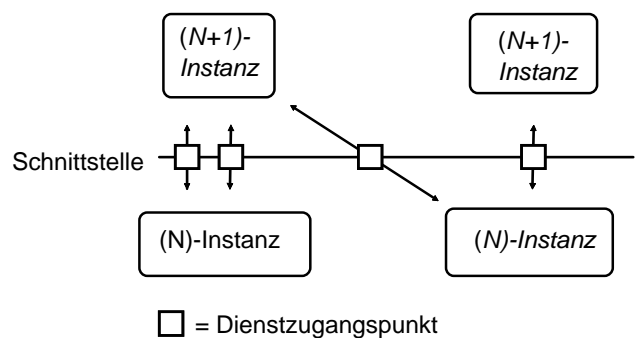
Instanzen der gleichen Schicht in verschiedenen Netzknoten heißen Partnerinstanzen (Peer entities).

Partnerinstanzen arbeiten zusammen, um den Dienst (Service) der Schicht zu erbringen. Dazu benutzen sie den nächst niedrigen Dienst.

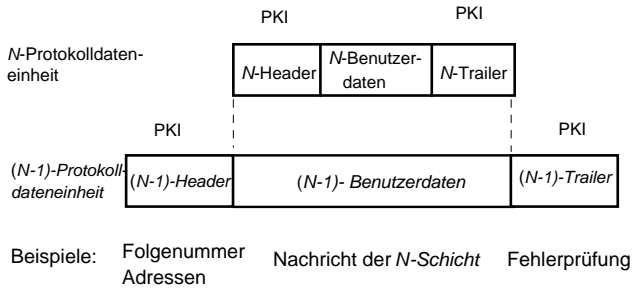
Ein Protokoll (Protocol) ist ein Satz von Regeln für die Kommunikation zwischen Partnerinstanzen.



Auf einen Dienst wird über einen Dienstzugangspunkt (SAP, Service Access Point) zugegriffen.

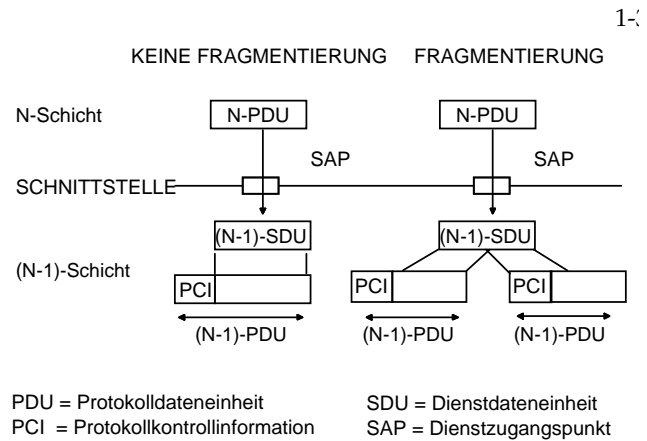


Eine **Protokoll dateneinheit** (PDU, Protocol Data Unit) ist eine Informationseinheit, die zwischen Partnerinstanzen in verschiedenen Stationen als Teil des Protokolls ausgetauscht wird.



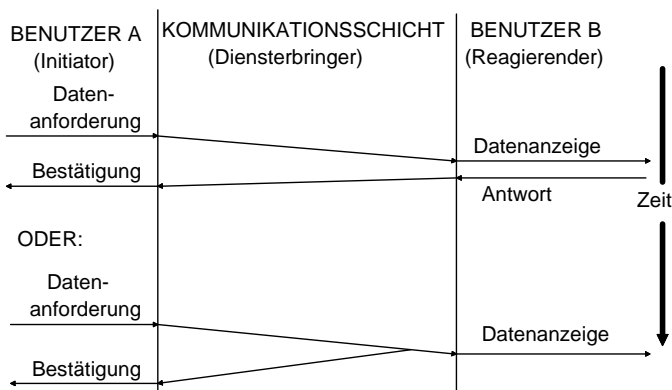
Als **Dienst dateneinheit** (SDU, Service Data Unit) werden Schnittstellendaten bezeichnet, die von einer Schicht an die nächst niedrige oder umgekehrt übertragen werden.

Ein **Dienstelement** (SP, Service Primitive) ist eine abstrakte Repräsentation einer Interaktion **Dienstbenutzer** und **Diensterbringer**.



Es gibt i.w. 4 Sorten von Dienstelementen:

- **Anforderung** (Request)
- **Anzeige** (Indication)
- **Antwort** (Response)
- **Bestätigung** (Confirmation)



### Klassifikation von Diensten und Protokollen

- Verbindungslose Dienste
- Verbindungsorientierte Dienste
- Protokolle

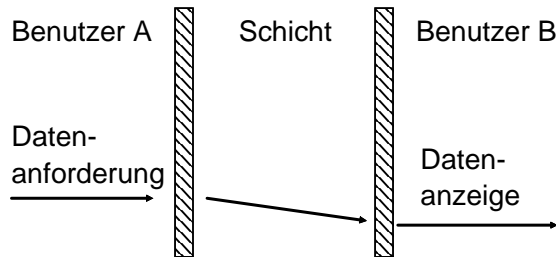
### Verbindungslose Dienste

wesentliche Merkmale:

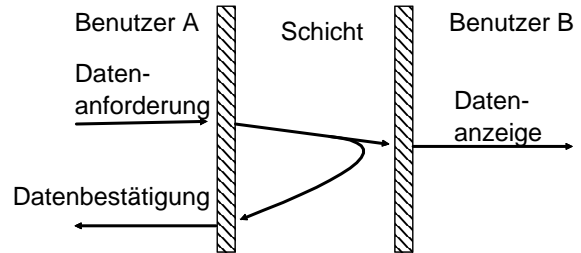
- "Briefverkehr"
- es werden keine Verbindungen zwischen Kommunikationspartnern aufgebaut
- jede Nachricht enthält eine (Quell- und) Zieladresse

**Datagrammdienst:**

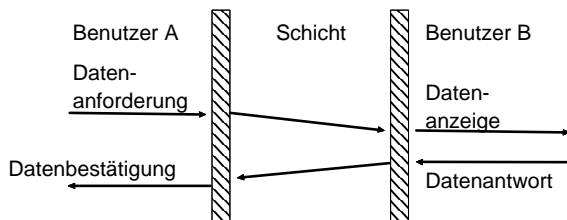
- Sender bekommt keine Bestätigung
- "Senden und Beten"
- Benutzer muß sich absolut auf das Kommunikationssystem verlassen
- Einsatz in LANs und WANs
- Zuverlässigkeit muß in einer höheren Schicht erhöht werden

**Zuverlässiger Datagrammdienst:**

- Dienst mit Übergabebestätigung
- liefert eine Antwort (Response) für jede Benutzernachricht
  - Nachricht erfolgreich gesendet
  - Nachricht nicht erfolgreich gesendet
  - weiß nicht
- Dienst erhöht Zuverlässigkeit der niedrigeren Schichten
- Bestätigung enthält keine Daten

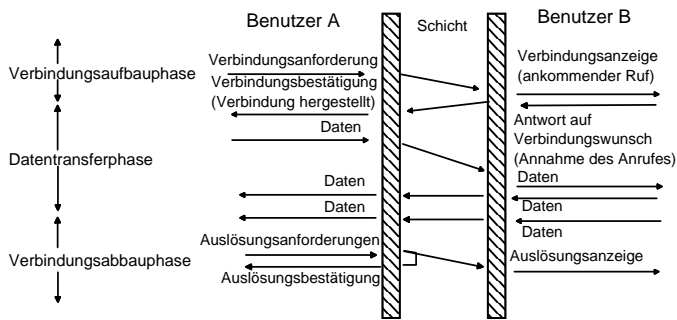
**Anfrage-Antwort:**

- empfangender Benutzer antwortet, nicht nur empfangende Station
- Antwort bedeutet, mindestens eine Anfrage ist empfangen worden
- Fehlantwort bedeutet, keine oder mehrere Anfragen sind empfangen worden

**Verbindungsorientierte Dienste**

wesentliche Merkmale:

- "Telephonverkehr"
- es werden Verbindungen zwischen Kommunikationspartnern aufgebaut
- eine Nutznachricht enthält keine Adresse
- es gibt drei Operationsphasen: Verbindungsaufbauphase, Datenphase, Verbindungsabbauphase
- in Verbindungsaufbauphase können Dienstgüte und Optionen ausgehandelt werden
- nur in der Aufbauphase spielen Adressen eine Rolle, danach nur Verbindungsbezeichner



## Protokolle

Klassifikation ähnlich wie bei Diensten

**zustandsloses Datagrammprotokoll:**

- einfachste Protokollform
- jede Nachricht enthält volle Ziel- und Sendeadresse
- enthält keine Zustandsinformation
- Fehlererkennung aber keine -korrektur

**bestätigtes Datagrammprotokoll:**

- Empfangsbestätigung
- Übertragungswiederholung für verfälschte oder verlorene Nachrichten
- Zustandsinformation i.d.R. nur beim Sender

**verbindungsorientiertes Protokoll:**

- drei Phasen: Aufbau, Transfer, Abbau
- Protokoll enthält verschiedene Zustandsinformation:
  - nichtbestätigte Nachrichten
  - Adresse der entfernten Instanz
  - Nachrichtenfolgennummern
  - Flußkontrollinformation
- Aufbauphase dient zur Synchronisation der Zustandsinformation
- erhöht Zuverlässigkeit des benutzten Dienstes erheblich
- Anwendung für Kommunikation über öffentliche Netze

**Timer-basiertes Protokoll:**

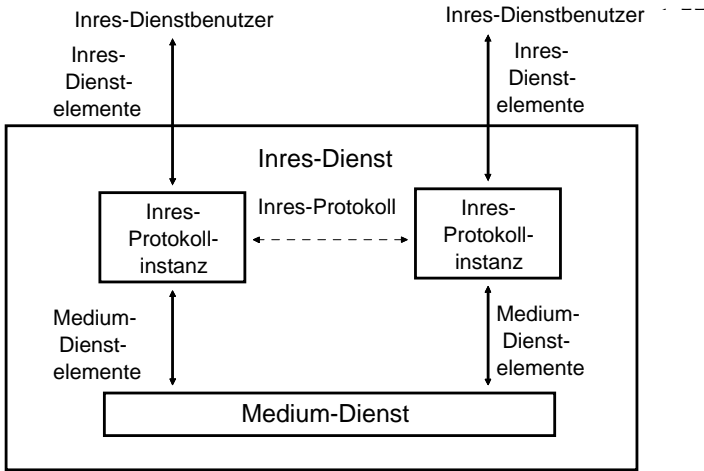
- ähnlich wie verbindungsorientiert, Verlust, Verdopplung, falsche Reihenfolge wird erkannt
- kein expliziter Auf- und Abbau von Verbindungen
- neue Nachricht baut Verbindung auf
- Timer beendet Verbindung

Beispiel: TCP/IP (Protokolle des Internet)

TCP ist ein verbindungsorientiertes Protokoll benutzt aber den verbindungslosen Dienst, den IP zur Verfügung stellt

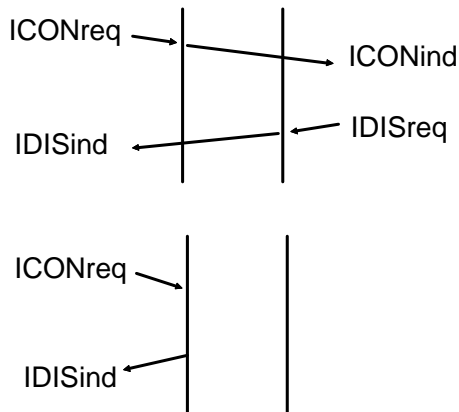
### Beispiel für die Schichtung von Diensten und Protokollen

- Medium-Dienst: Unzuverlässiger Datenaustausch
- Inres-Protokoll: erbringt einen verbindungsorientierten Dienst zum zuverlässigen Datenaustausch
- Inres-Dienst: der vom Inres-Protokoll zusammen mit dem Medium-Dienst erbrachte Dienst

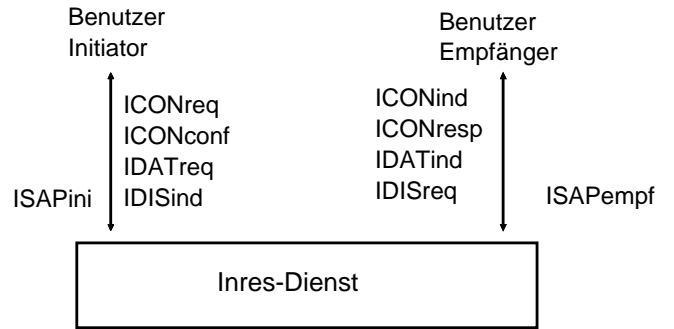


### Der Inres-Dienst

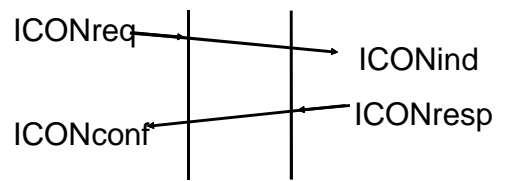
Telematik (WS 02/03)



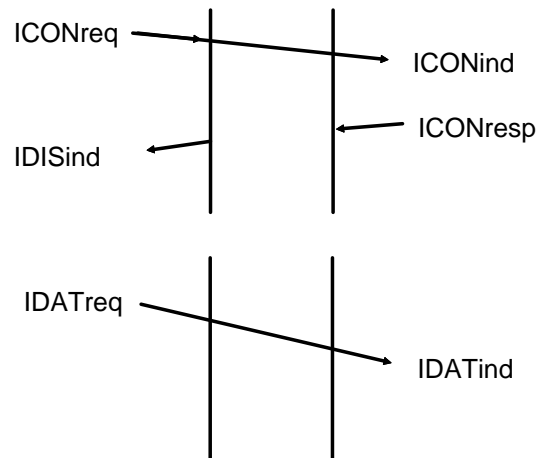
Telematik (WS 02/03)



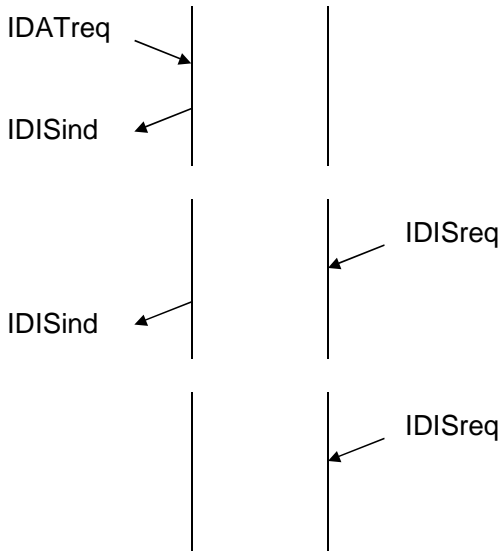
Die SPs IDATreq und IDATind besitzen einen Parameter vom Typ ISDU. Alle übrigen SPs sind parameterlos.



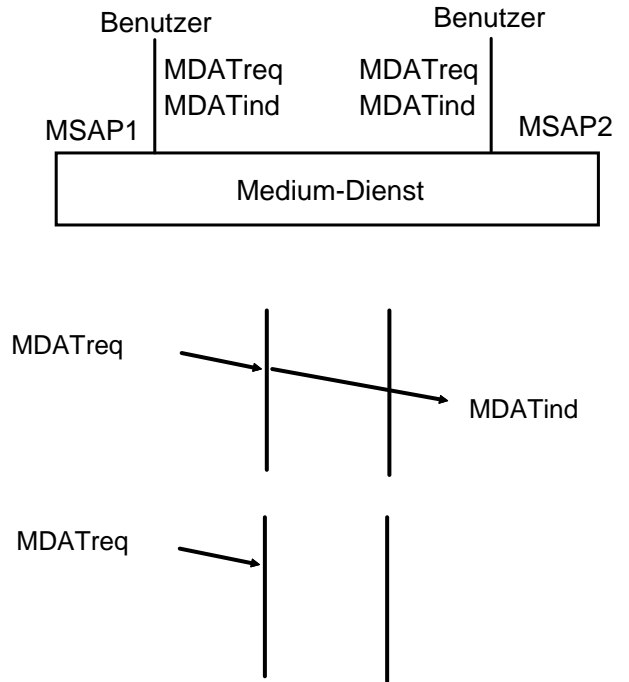
Telematik (WS 02/03)



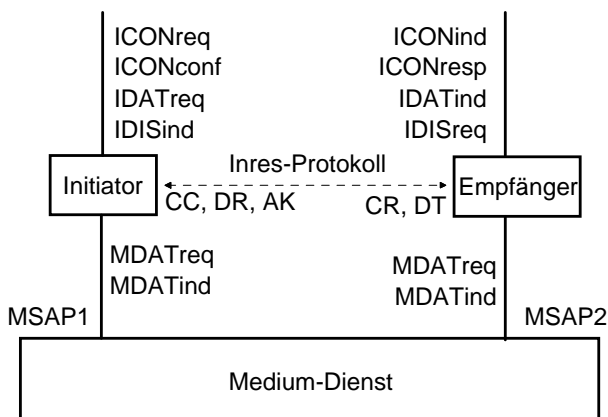
Telematik (WS 02/03)



**Der Medium-Dienst**



**Das Inres-Protokoll**



**Generelle Eigenschaften des Protokolls**

Das Protokoll ist ein verbindungsorientiertes Protokoll, das zwischen zwei Protokollinstanzen *Initiator* und *Empfänger* operiert.

Die Protokollinstanzen kommunizieren durch den Austausch der Protokolldateneinheiten *CR, CC, DT, AK* und *DR*.

Die Kommunikation zwischen den Protokollinstanzen findet in drei Phasen statt: **Verbindungsphase, Datenphase** und **Unterbrechungsphase**.

In jeder der drei Phasen sind nur bestimmte *PDU*s und *SP*s sinnvoll. Unerwartete *PDU*s und *SP*s werden von *Initiator* und *Empfänger* ignoriert.

PDU	Bedeutung	Parameter	entsprechende SPs
CR	Verbindungsaufbau	keine	ICONreq, ICONind
CC	Verbindungsbestätigung	keine	ICONresp, ICONconf
DT	Datentransfer	Folgenummer (1 Bit) Dienstdateneinheit (ISDU)	IDATreq, IDATind
AK	Bestätigung	Folgenummer (1 Bit)	-
DR	Verbindungssabbau	keine	IDISreq, IDISind

## Verbindungsphase

Ein Verbindungsaufbau wird von dem Protokollbenutzer an der Protokollinstanz *Initiator* mit einem *ICONreq* initiiert. Die Instanz *Initiator* sendet daraufhin ein *CR* an die Instanz *Empfänger*.

*Empfänger* antwortet mit *CC* oder *DR*. Im Fall von *CC* sendet *Initiator* ein *ICONconf* an den Benutzer, und die Datenphase wird begonnen. Wenn *Initiator* von *Empfänger* ein *DR* erhält, wird die Unterbrechungsphase begonnen. Wenn *Initiator* keine Antwort auf *CR* innerhalb von 5 Sekunden empfangt, wird *CR* erneut gesendet. Wenn nach 4 Versuchen noch keine Antwort empfangen wurde, beginnt *Initiator* die Unterbrechungsphase.

Wenn *Empfänger* ein *CR* von *Initiator* empfängt, wird dem Benutzer, der mit der Instanz *Empfänger* kommuniziert, ein *ICONind* gesendet. Der Benutzer kann mit *ICONresp* oder mit *IDISreq* antworten. *ICONresp* ist die Verbindungsbestätigung, *Empfänger* sendet ein *CC* an *Initiator* und beginnt die Datenphase. Bei *IDISreq* beginnt *Empfänger* die Unterbrechungsphase.

## Datenphase

Wenn der Benutzer der Instanz *Initiator* ein *IDATreq* übergibt, sendet *Initiator* ein *DT* an *Empfänger* und kann darauf ein weiteres *IDATreq* vom Benutzer empfangen. *IDATreq* enthält als Parameter eine Dienstdateneinheit *ISDU*, mit der der Benutzer Informationen an den Partnerbenutzer übermitteln möchte. Diese Benutzerdaten aus *IDATreq* werden transparent als Parameter in der Protokollateneinheit *DT* übernommen. Nach Aussenden eines *DT* wartet *Initiator* 5 Sekunden auf eine entsprechende korrekte Empfangsbestätigung *AK* vom *Empfänger*. Nach dieser Zeit wird das *DT* erneut gesendet.

Nach 4-maliger erfolgloser Wiederholung beginnt *Initiator* die Unterbrechungsphase.

*DT* und *AK* besitzen als Parameter eine binäre Folgennummer (0 oder 1). Der *Initiator* beginnt nach dem Verbindungsaufbau die Übertragung von *DT* mit der Folgennummer 1. Eine korrekte Bestätigung *AK* enthält jeweils die gleiche Folgennummer. Nach Empfang eines korrekten *AK* kann das nächste *DT* mit der nächsten (d.h. anderen) Folgennummer gesendet werden. Wenn der *Initiator* ein *AK* mit

falscher Folgennummer empfängt, sendet er das letzte *DT* noch einmal. Ebenfalls wiederholt wird *DT*, wenn nach 5 Sekunden kein *AK* empfangen wurde. Ein *DT* kann insgesamt 4 mal wiederholt werden. Danach beginnt *Initiator* die Unterbrechungsphase. Das gleiche gilt bei Empfang eines *DR*.

Der *Empfänger* erwartet nach dem Verbindungsaufbau zunächst ein *DT* mit der Folgennummer 1. Nach dem Empfang eines *DTs* mit erwarteter Folgennummer übergibt er die entsprechenden Benutzerdaten *ISDU* als Parameter eines *IDATind* an seinen Benutzer und sendet eine Bestätigung *AK* mit der gleichen Folgennummer an den *Initiator*. Ein *DT* mit nicht erwarteter Folgennummer wird mit einem *AK*, mit der letzten korrekt empfangenen Folgennummer, beantwortet. Die Benutzerdaten *ISDU* eines inkorrekten *DT* werden ignoriert. Wenn *Empfänger* ein *CR* empfängt, begibt sich *Empfänger* in die Verbindungsphase. Bei *IDISreq* beginnt *Empfänger* die Unterbrechungsphase.

## Unterbrechungsphase

Ein *IDISreq* vom Benutzer führt beim *Empfänger* zum Senden eines *DR* an den *Initiator*. Danach kann *Empfänger* einen neuen Verbindungsaufbauwunsch *CR* vom *Initiator* entgegennehmen.

Beim *Initiator* führt ein *DR* zu einem *IDISind* an den Benutzer. Ein *IDISind* wird auch dann an den Benutzer gesendet, wenn ein *CR* oder *DT* 4 mal erfolglos gesendet wurde. Danach kann eine neue Verbindung aufgebaut werden.

## Gründe für die Schichtung:

- Unabhängigkeit zwischen Schichten
- Flexibilität

Die Implementierung einer Schicht kann beliebig geändert werden, solange der Dienst der bereitgestellt wird, gleich bleibt (Abstraktion, Information Hiding).

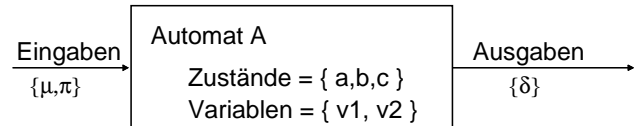
- Physikalische Trennung
- Einfache Implementierung und Wartung
- Förderung der Standardisierung

## 2 Beschreibung von Kommunikationssystemen mit SDL

SDL = Specification and Description Language

Speziell für Erfordernisse in der Telekommunikation entwickelt

Basiert auf der Idee des erweiterten endlichen Automaten



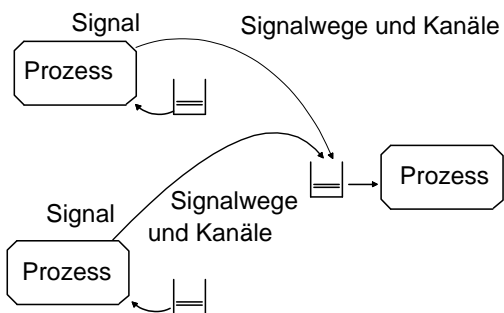
Gesamtmenge der Zustände:

$$Z = \{a,b,c\} \times \text{Typ}(v1) \times \text{Typ}(v2)$$

## Die Sprachelemente von SDL

Basiskonzepte

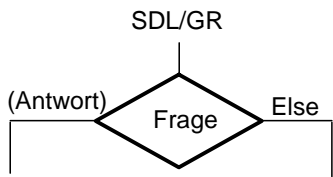
Das Grundelement in SDL ist der Prozeß



Charakterisierung eines Prozesses:

- ein Prozeß befindet sich entweder gerade in einem Zustandsübergang oder wartet auf eine Eingabe;
- es gibt eine Eingabewarteschlange für jeden Prozeß;
- wenn zwei Eingaben einen Prozeß gleichzeitig erreichen, werden sie in zufälliger Reihenfolge in der Eingabewarteschlange gepuffert.

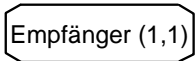
Die zwei syntaktischen Formen SDL/GR und SDL/PR



SDL/PR

DECISION Frage  
(Antwort): .....  
ELSE : .....  
ENDDECISION;

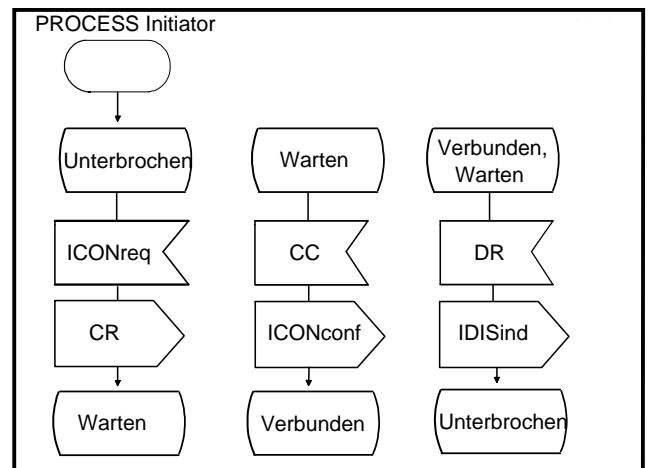
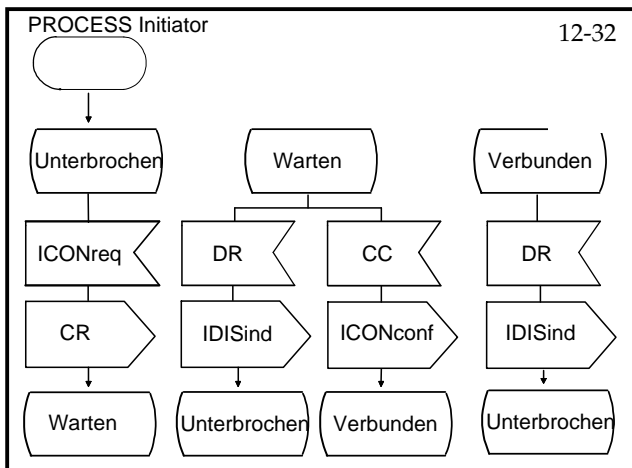
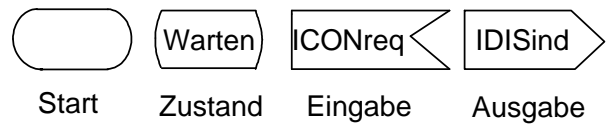
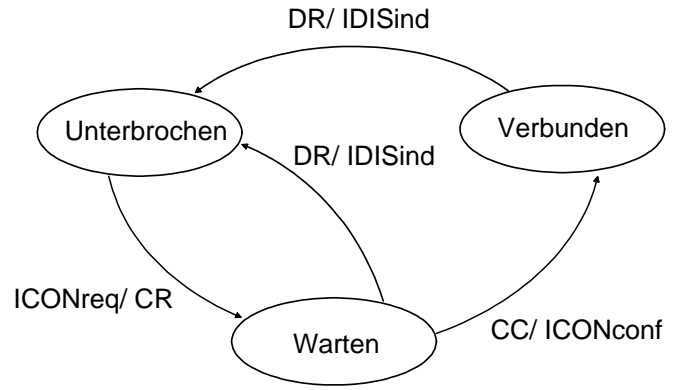
SDL/GR

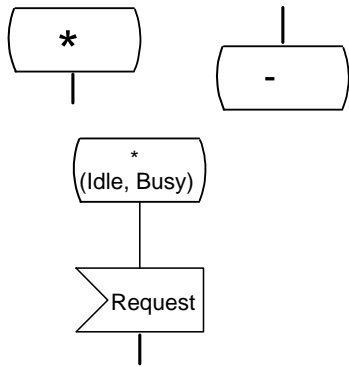


SDL/PR

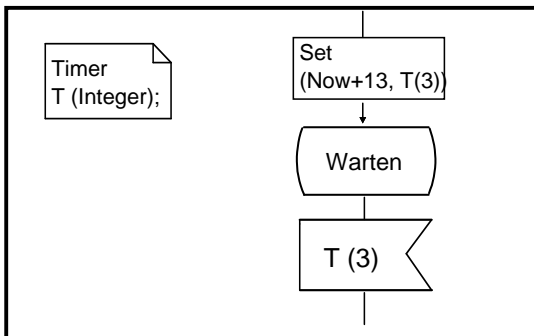
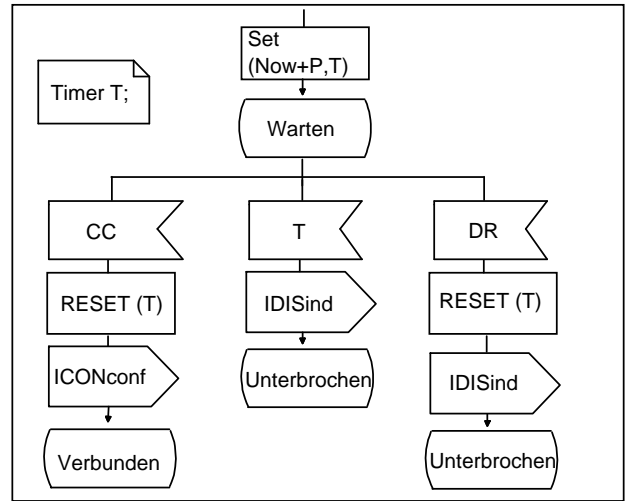
PROCESS Empfänger (1,1);  
REFERENCED;

Zustände und Zustandsübergänge

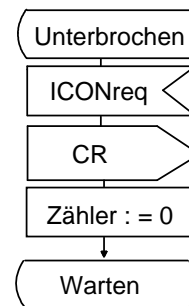
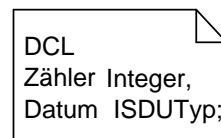




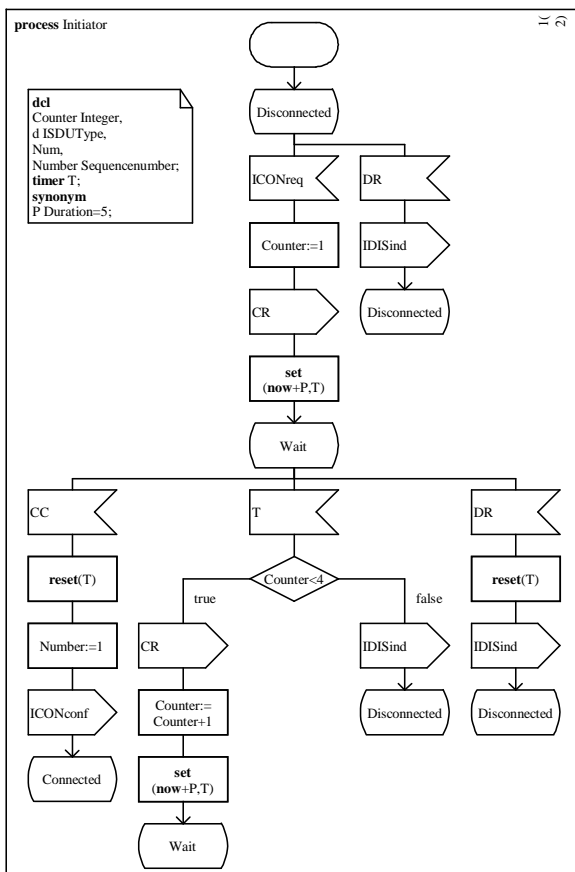
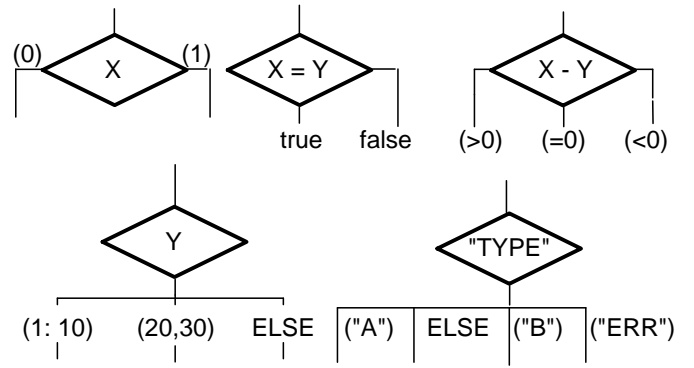
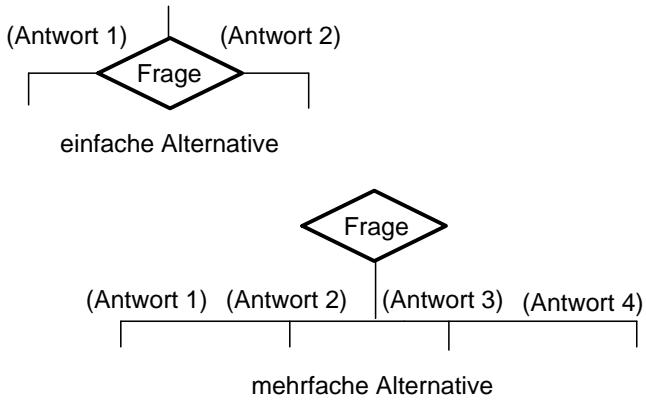
**Zeit in SDL**



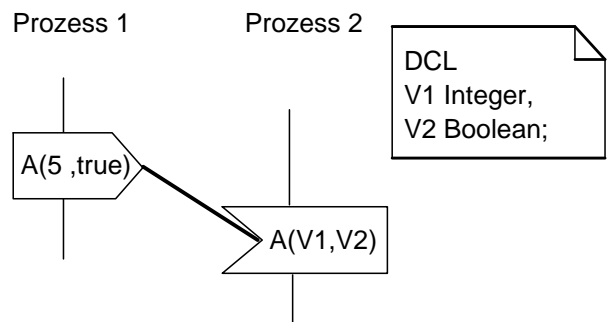
**Deklaration und Gebrauch von Daten**

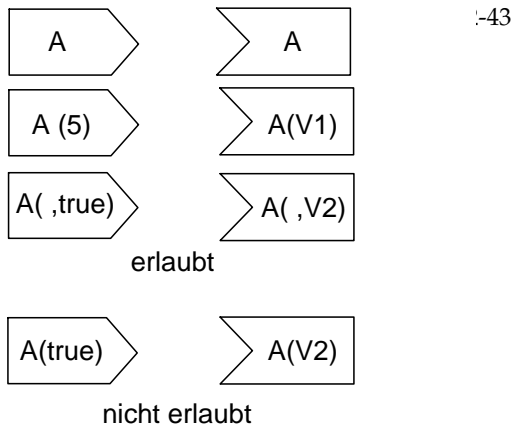


Die Alternative

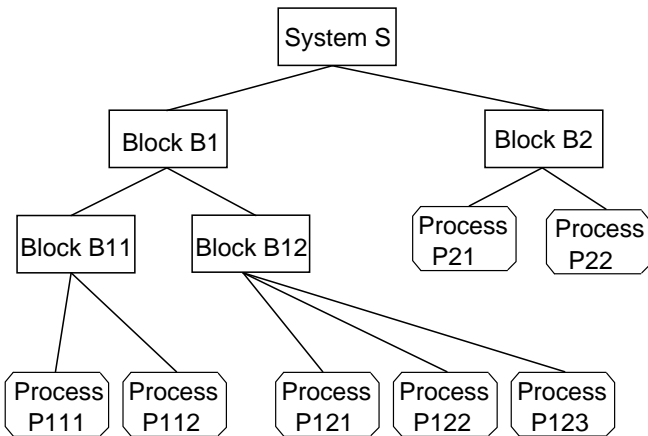
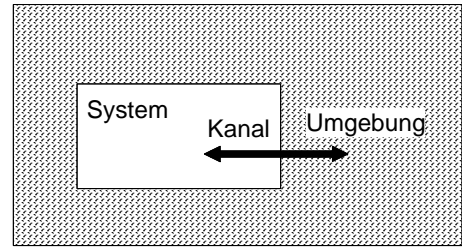


Signale mit Daten

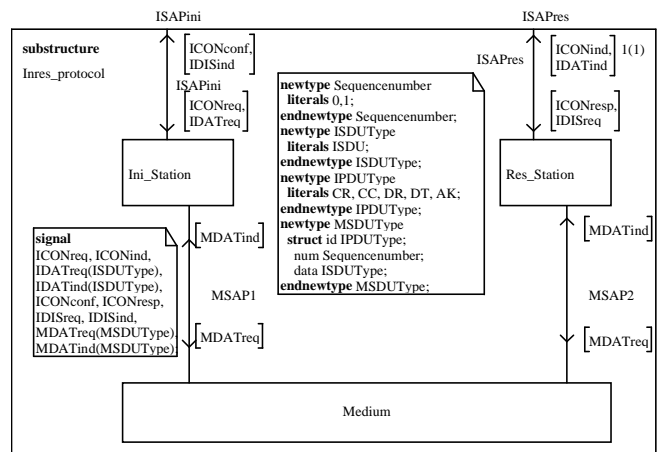




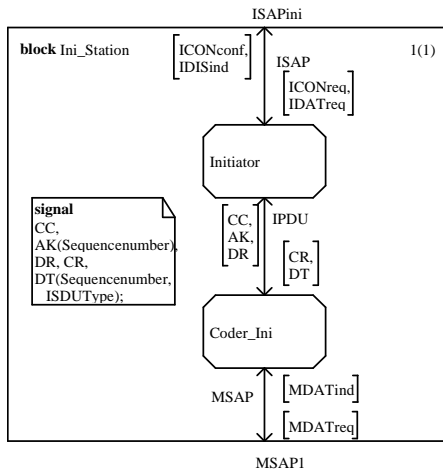
## Strukturierung und Prozeßkommunikation



## Blockinteraktionsdiagramme



## Prozeßinteraktionsdiagramme



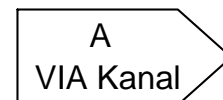
## Prozeßkommunikation und Adressierung

Ein Ziel kann auf unterschiedliche Weise spezifiziert werden:

- durch direkte Angabe einer Zieladresse



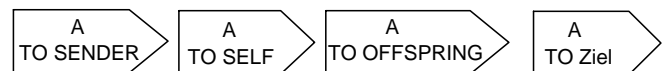
- durch indirekte Adressierung, indem das Ziel eindeutig aus der Systemstruktur hervorgeht,
- durch Angabe eines Kanals oder Signalwegs, über den gesendet werden soll.



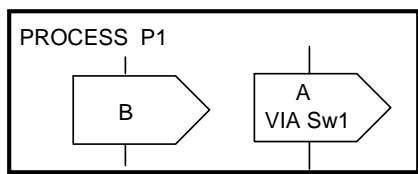
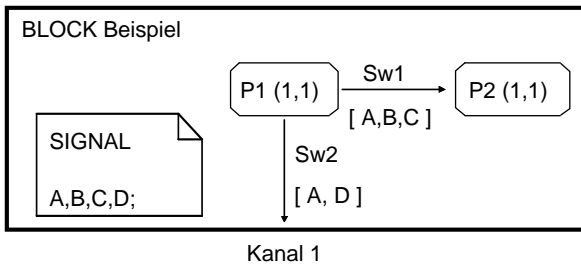
Adresse eines Prozesses:

- **SELF**: dieser Ausdruck liefert die eigene Adresse;
- **SENDER**: dieser Ausdruck liefert die Adresse desjenigen Prozesses, von dem zuletzt eine Eingabe erhalten wurde;
- **OFFSPRING**: dieser Ausdruck liefert die Adresse derjenigen Prozeßinstanz, die zuletzt von der Prozeßinstanz dynamisch erzeugt wurde;
- **PARENT**: dieser Ausdruck liefert die Adresse derjenigen Prozeßinstanz, durch die die Prozeßinstanz dynamisch erzeugt wurde;

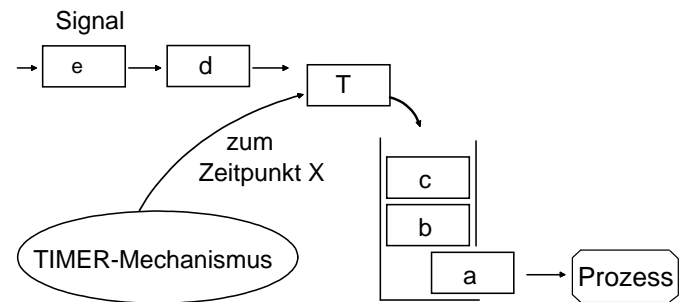
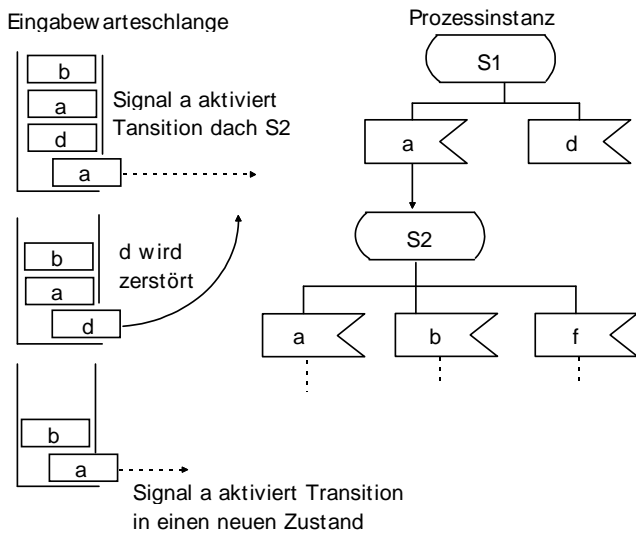
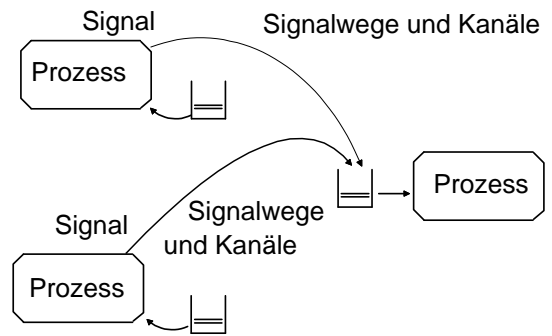
Direkte Adressierung



Indirekte Adressierung und Adressierung durch Angabe eines Wegs

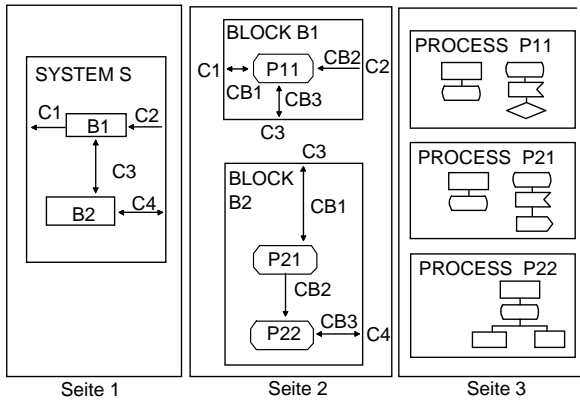


Semantik der Prozeßkommunikation



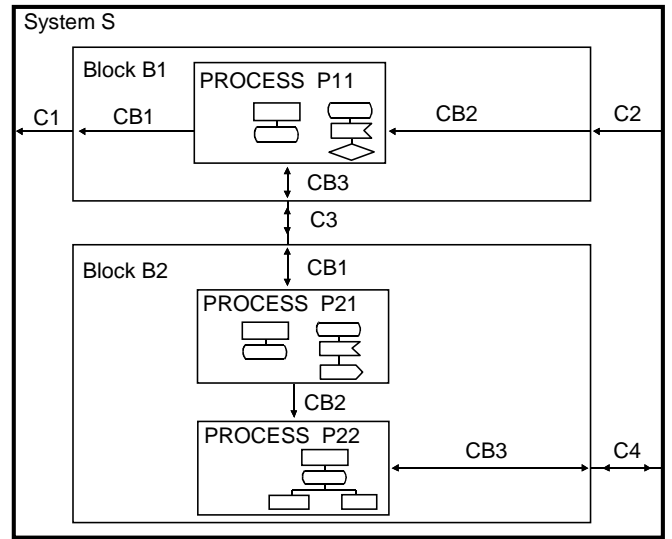
**Dokumentation**

Darstellung durch Referenzierung

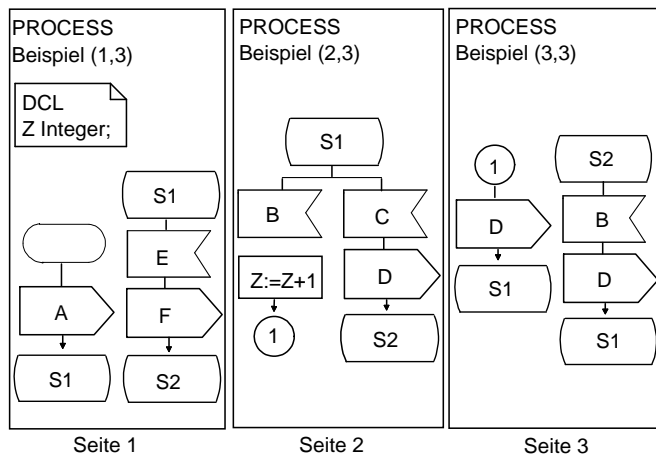


Gemischte Darstellung

Kompromiß zwischen Referenzierung und Verschachtelung



Dokumentation umfangreicher Prozeßdiagramme



**3 Das ISO-Referenzmodell OSI**

OSI = Open Systems Interconnection

Die Entwicklung vieler inkompatibler Standards führte zu der Erkenntnis, daß ein allgemein anerkanntes Basismodell als Rahmen für die Entwicklung von Standards benötigt wird.

Das ISO-Referenzmodell für "Offene Systeme" (OSI) legt 7 Schichten und deren Funktionen fest.

Schichten 1-3 unterscheiden sich erheblich, in höheren Schichten kann Harmonie erreicht werden.

OSI ist selten in reiner Form implementiert.

### 3.1 Die 7 Schichten

#### Die physische Schicht (Schicht 1)

Übertragung von Bits auf einer physikalischen Verbindung

- wieviel Volt werden zur Darstellung einer (dualen) Eins benötigt
- welche Art Verbindungskabel wird benutzt
- wie lang ist dieses Kabel
- erfolgt Bitübertragung gleichzeitig in beide Richtungen
- welche Steckerverbindungen werden benutzt
- welche zusätzlichen Übertragungseinrichtungen sind nötig

Ein typischer Fall ist die Benutzung der Telefonleitung.

Dabei Benutzung eines **Modems** (Modulator-Demodulator, Konvertierung digital-analog)

Ein typischer Standard der physikalischen Schicht ist der RS232-Stecker (9-polig) für die serielle Übertragung von Daten.

Andere Möglichkeiten der Übertragung sind **Radiowellen** und **optische Signale**.

#### Die Sicherungsschicht (Schicht 2)

In der physikalischen Schicht können Fehler auftreten.

Die Aufgabe ist, Fehler zu erkennen und zu beheben.

Zu übertragende Daten werden in Blöcke eingeteilt (Frames).

Ein **Protokoll** übernimmt folgende Aufgaben:

- Festlegung und Erkennung von Anfang und Ende eines Datenblocks
- Erkennung und Behebung von Übertragungsfehlern
- Reihenfolgeüberprüfung beim Senden und Empfangen

Syn.-Zeichen	Start	Kopf	Benutzerdaten	Blockprüfung	Ende
--------------	-------	------	---------------	--------------	------

SYN.-ZEICHEN	- Das Bitmuster, um Bytesynchronisation zu erreichen
START	- Kennung für den Beginn einer Nachricht
KOPF	- Protokollkontrollinformation
BENUTZERDATEN	- Information der nächsthöheren Ebene
BLOCKPRÜFUNG	- Fehlererkennung bzw. Korrekturinformation
ENDE	- Kennung für das Ende einer Nachricht

- HDLC (High level Data Link Control, ISO)
- L2CAP (Bluetooth)
- LAP (Line Access Protocol, ITU-T)

Beispiel-Protokolle der Sicherungsschicht:

### Die Vermittlungsschicht (Schicht 3)

In den Schichten 1 und 2 nur lokale Sicht, Übertragung von Daten zwischen zwei **benachbarten** Rechnern

In der Schicht 3: Sicht des **gesamten** Netzes

insbesondere

- Wegwahl (**Routing**)
- Einteilung der Daten in **Pakete**
- Flußkontrolle
- Reihenfolgekontrolle

Es gibt grundsätzlich zwei verschiedene Dienste, die der Schicht 4 zur Verfügung gestellt werden können:

- **Virtual Circuit Service** (verbindungsorientiert)
- **Datagram Service** (verbindungslos)

bekannte Protokolle der Schicht 3:

- IP (Internet Protocol, IETF)
- X.25/3 (ITU-T)
- Q.931 (auch als ISDN bekannt, ITU-T)

### Die Transportschicht (Schicht 4)

Hat die Aufgabe, die höheren Ebenen von den speziellen Eigenarten des Netzes zu befreien.

Gewährleistet zuverlässigen Ende-zu-Ende-Transportdienst zwischen Benutzerprozessen.

Benutzerprozesse haben den Eindruck, mit benachbarten Benutzerprozessen zu kommunizieren.

- Einteilung der Daten von Schicht 5 in kleinere Einheiten
- Behebung von Reihenfolgefehlern
- Adressen-Handling
- Auf- und Abbau von Transportverbindungen
- Multiplexing oder Splitting auf Verbindungen der Schicht 3
- optimale Ausnutzung (Leistung, Kosten) der Schicht 3 Dienste

bekannte Protokolle:

- TCP (Transmission Control Protocol, IETF)
- UDP (User Data Protocol, IETF)
- IS 8073 (Transport Protocol, ISO)

## Die Sitzungsschicht (Schicht 5)

Eine **Sitzung** ist ein Dialog zwischen zwei Benutzern oder Anwendungsprozessen über ein Rechnernetz

Schicht 5

- baut Sitzungen ordnungsgemäß auf
- unterstützt Sitzungen
- baut Sitzungen ordnungsgemäß ab

Schicht 5 setzt Kontrollpunkte.

Im Fehlerfall der Transportschicht kann eine Sitzung an einem definierten Punkt fortgesetzt werden.

Schicht 5 ist in den meisten Netzen sehr klein und oft in die Anwendungs- oder Transportschicht integriert.

Dienstspezifikation der Schicht 5:

- IS 8326 (Basic Connection Oriented Session Service Specification, ISO)

bekannte Protokolle:

- IS 8327 (Basic Connection Oriented Session Protocol Specification, ISO)

## Die Darstellungsschicht (Schicht 6)

Gleicht Unterschiede in der Informationsdarstellung zwischen Anwendungsprozessen aus.

Erlaubt die Kommunikation von Anwendungsprozessen, die auf verschiedenen Rechnern in verschiedenen Programmiersprachen implementiert sind.

Umfaßt folgende Aspekte:

- verwendeter Zeichensatz
- Codierung zu übertragender Daten (Datenverschlüsselung)
- Darstellung der Daten auf einem Bildschirm oder Drucker

## Die Anwendungsschicht (Schicht 7)

Enthält solche Funktionen, die aus Sicht des Benutzers bzw. Netzanwenders relevant sind.

Schicht 7 stellt als solche keiner höheren Schicht einen Dienst zur Verfügung.

Kann aber selbst in Schichten eingeteilt sein, einige Anwendungsinstanzen können dann anderen einen Dienst zur Verfügung stellen, z.B. für

- Dateitransfer
- Electronic Mail
- verteilte Datenbanken

Protokolle der Schicht 7:

- HTTP
- Telnet
- FTP (File Transfer Protocol)
- SMTP (Simple Mail Transfer Protocol)
- DNS (Domain Name Service)

### 3.2 Punkte, die alle Schichten betreffen

- **Naming**: Identifikation von Objekten/Ressourcen
- **Segmentieren** und **Blocken**
- **Verbindungen** und **Multiplexen**
- **Synchronisation**: Kontrolle in einer Umgebung mit unvorhersehbarer Verzögerung und Fehlverhalten
- **Fehlerüberwachung**: Fehlererkennung und -behebung
- **Flußkontrolle**
- **Prioritäten**

### Naming

der **Name** einer Ressource zeigt an, *was* wir suchen

eine **Adresse** zeigt an, *wo* wir es finden

eine **Route** sagt uns, *wie* man dort hingelangt

**Identifizieren** (Bezeichnen) ist das Herzstück eines Rechensystementwurfs, sowohl verteilt als auch zentralisiert

**Bezeichner** werden in allen Schichten z.B. benötigt für

- Schutz
- Fehlerüberwachung
- Ressourcenverwaltung
- Lokalisieren und gemeinsames Nutzen von Ressourcen

Bezeichner können sehr unterschiedlich sein:

- von Namen hoher Ebenen bis Adressen niedriger Ebenen

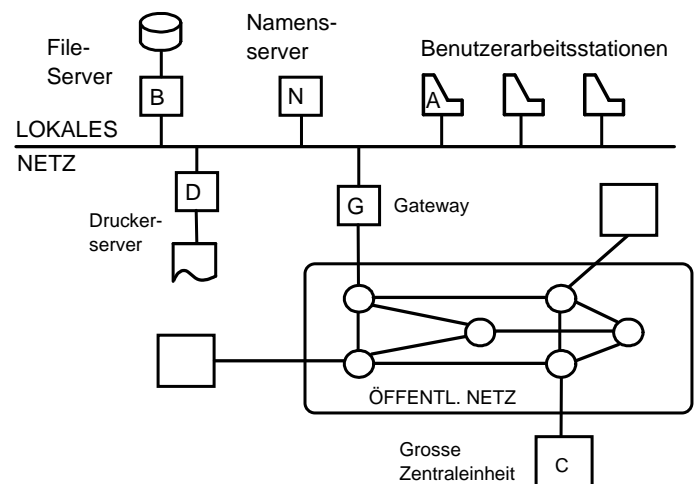
- von benutzerorientierten Namen bis zu maschinenorientierten Adressen

Namen werden auf Adressen abgebildet (**Binding**)

Binding kann:

- **statisch** sein, also zur Übersetzungszeit eine Programms
- **dynamisch** sein, also zur Ausführungszeit

Dynamisches Binding wird oft mit **Nameservern** gemacht.



E-Mail: [hogrefe@itm.mu-luebeck.de](mailto:hogrefe@itm.mu-luebeck.de)

Ideal ist

- logische Sicht: System ist Raum von benannten Objekten

statt

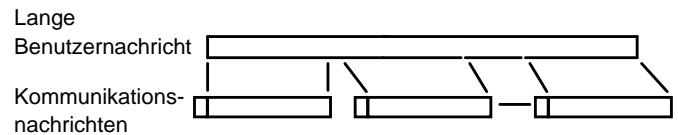
- physikalische Sicht: System ist Raum von Hoststationen, die Objekte enthalten

## Segmentieren und Wiederaussetzen

Manche Benutzerdaten sind zu umfangreich, um als eine einzige Nachricht übertragen zu werden.

Einteilung in kleinere Pakete.

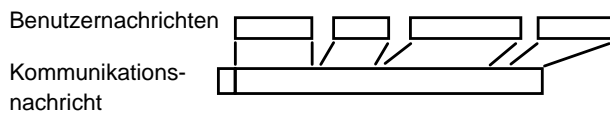
- Interne Puffer in den verschiedenen Schichten haben endliche Größe
- extrem lange Nachrichten monopolisieren gemeinsam genutzte Übertragungswege und andere Ressourcen
- Aufbrechen einer langen Nachricht in kleinere Pakete erlaubt parallele Benutzung von Mehrfachverbindungen (load sharing)
- bei hoher Fehlerrate kann bessere Nutzleistung mit kürzeren Nachrichten erreicht werden.
- Die meisten Netze unterstützen nur Nachrichten bis zu einer bestimmten maximalen Länge



## Blocken

Auch zu kleine Nachrichten sind ungünstig.

Der Verwaltungsaufwand muß in vernünftigem Verhältnis zu den Nutzdaten stehen.



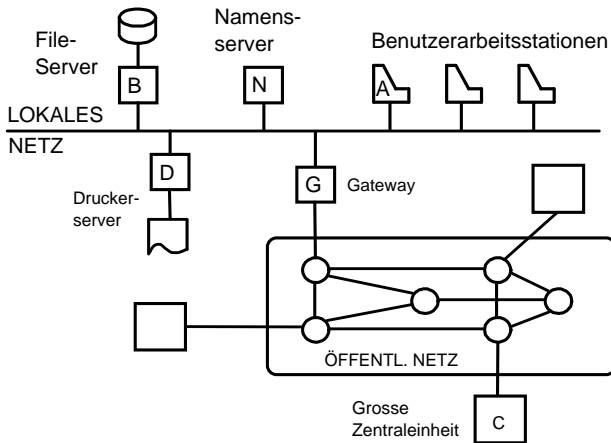
## Verbindungen und Multiplexen

Kommunikationsverbindungen werden benötigt, um **mehrere** Nachrichten von Quelle zu Ziel zu transferieren.

Verbindungen bewahren Zustandsinformation zwecks:

- Wiederaufsetzen nach Fehlern
- Übergabe in der richtigen Reihenfolge
- Segmentieren und Wiederaussetzen

Eine Verbindung ist ein **Nachrichtenstrom** zwischen zwei kommunizierenden Instanzen.

**Beispiel**

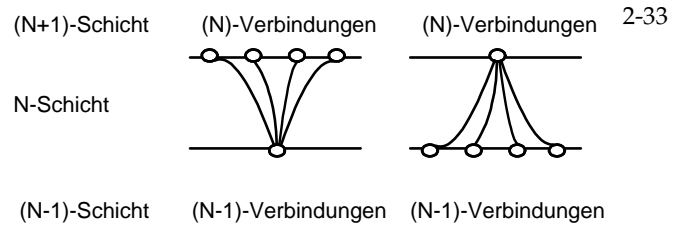
A-B und A-C: zum Kontrollieren eines Filetransfers und zum Initiieren eines Jobs

B-C: zum Übertragen einer Datei

C-D: zum Drucken einer Datei

Große Stationen, z.B. C haben viele entfernte Benutzer aber nur endlich viele physikalische Netzzugänge, daher

**Multiplexing** verschiedener **logischer** Verbindungen auf eine physikalische.



**Aufwärts-Multiplexen**: Verbindungen einer höheren Schicht auf eine Verbindung einer niedrigeren Schicht.

**Abwärts-Multiplexen**: Eine Verbindung einer höheren Schicht auf mehrere Verbindungen einer niedrigeren Schicht.

**Fehlerüberwachung**

Hauptaufgabe der Protokolle: **zuverlässige** Kommunikation

drei Aspekte:

- Erkennung von Fehlern
- Korrektur von Fehlern
- Wiederaufsetzen nach Fehlern (Recovery)

typische Probleme:

- Informationsverfälschung (Bitfehler)
- Verlust von Nachrichten
- Verdopplung von Nachrichten
- Nachrichten in falscher Reihenfolge

Die meisten Protokolle beheben Fehler durch automatische **Wiederholung** der Übertragung.

**Fehler in der Nachricht**: Einfügen redundanter Information

**Reihenfolgefehler**: Vergabe einer Folgenummer

Es gibt auch **nichtbehebbar**e Fehler. Dann müssen ev. mehrere Schichten **reinitialisiert** werden.

**Beispiele: Fehlererkennung und -korrektur in Schicht 2**

Fehlererkennung durch redundante Information, an der der Empfänger eine Verfälschung erkennt.

**Parity Check**: Es werden Bitfolgen betrachtet (zumeist Bytes). Ist die Quersumme gerade, wird an die Bitfolge eine 0 angefügt (gerade Parität) andernfalls eine 1 (ungerade Parität)

**Block Check**: Eine Folge von Bytes wird als Block aufgefaßt. Es werden Zeilen- und Spaltenquersummen gebildet und zum Block hinzugefügt.

Bits 1 2 3 4 5 6 7 8

			x	x				CHK
								CHK
								CHK
			x	x				CHK
								CHK
								CHK
								CHK
								CHK
CHK	CHK	CHK	CHK	CHK	CHK	CHK	CHK	CHK

**Cyclic Redundancy Checksum (CRC):** Am häufigsten verwendete Methode (z.B. HDLC, SDLC).

k-Bit-Nachricht wird als Koeffizientenliste eines Polynoms vom Grad k-1 aufgefaßt

Beispiel: 1011 =  $N(x) = x^3 + x + 1$

Sender und Empfänger kennen ein gemeinsames Generator-Polynom  $G(x)$  vom Grad kleiner k-1  
Beispiel:  $G(x) = x^2 + 1$

Jetzt wird folgender Algorithmus angewendet:

a) habe  $G(x)$  den Grad r. Hänge r 0-Bits an die Nachricht an. Sie enthält dann k+r Bits und entspricht dem Polynom  $x^r N(x)$   
Beispiel: 101100 =  $x^5 + x^3 + x^2$

b) Berechne  $F(x) = x^r N(x) / G(x)$  (mod-2 Division)  
Beispiel:  $F(x) = (x^5 + x^3 + x^2) / (x^2 + 1)$   
 $= x^3 + 1$  Rest 1

c) Subtrahiere den Rest von  $x^r N(x)$  (mod-2 Subtraktion)

Beispiel:  $NC(x) = x^5 + x^3 + x^2 - 1 = x^5 + x^3 + x^2 + 1 = 101101$

Das ist die zu übertragende Nachricht.

Der Empfänger berechnet nun  $NC(x)/G(x)$ .  
Wenn es einen Rest gibt, wurde die Nachricht verfälscht.

Welche Fehler werden erkannt?

Es werden nur solche Fehler nicht erkannt, die Polynomen entsprechen, die durch  $G(x)$  teilbar sind.  
Die Qualität des Verfahrens hängt also von der Wahl von  $G(x)$  ab.

- HDLC:  $G(x) = x^{16} + x^{12} + x^5 + 1$
- ETHERNET:  $G(x) = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + 1$

Wahrscheinlichkeit für Nichterkennung eines Fehlers bei ETHERNET: ca.  $10^{-18}$

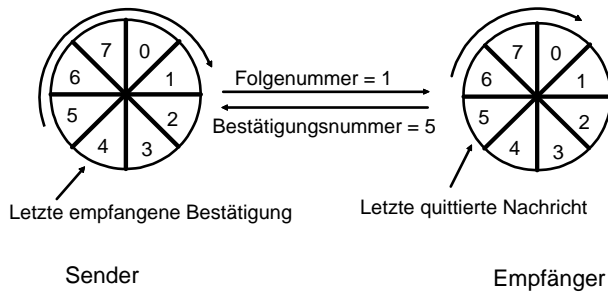
Fehlerkorrektur geschieht in der Regel durch wiederholte Übertragung einer verfälschten Nachricht. Eine Nachricht muß dabei stets quittiert

werden. Trifft für eine Nachricht keine Quittung ein ist ein Fehler aufgetreten.

## Flußkontrolle

Überwachung des Informationsflusses zwischen Sender und Empfänger, um zu verhindern, daß der Sender den Empfänger überflutet.

z.B. Fensterflußkontrolle



Fensterflußkontrolle

## Synchronisation

Für die Kommunikation zwischen zwei Prozessen über ein Netzwerk muß Synchronisation auf verschiedenen Ebenen sichergestellt sein:

### Bit-Synchronisation:

- Empfänger muß Anfang und Dauer eines Signalelements bestimmen.
- Abtasten der Leitung in bestimmten Abständen
- Bitübertragungsschicht

### Byte-Synchronisation:

- Austausch von Zeichen meist in 8-Bit-Einheiten (1 Byte)
- Empfänger muß Anfang und Ende eines Byte bestimmen können

### Block-Synchronisation:

- Daten werden in Blöcken zu mehreren Bytes zusammengefaßt (z.B. eine Nachricht)
- Die Bedeutung eines Bytes hängt von der Position im Block ab

### Zugang zum Übertragungsmedium:

- bei gemeinsamer Nutzung
- ein Benutzer darf nur zu einer bestimmten Zeit Zugriff haben
- der Zugriff muß fair sein

### Protokollsynchronisation:

- kommunizierende Partnerinstanzen haben Zustandsinformation (z.B. Sequenznummer)
- nach Fehlern und Neustart muß Zustandsinformation konsistent gemacht werden

### Prozeßsynchronisation:

- Synchronisation bei Zugriff auf gemeinsam genutzte Ressource (z.B. Daten)

## Prioritäten

Nachrichten eine Priorität zuweisen, um sie im Wettbewerb mit anderen zu bevorzugen

Nachrichten **hoher** Priorität

- haben **niedrige** Verzögerung
- können andere Nachrichten **überholen**

typische Anwendungen

- Multimedia
- Unterbrechung anzeigen
- Protokollsteuerung
- Alarm in Prozeßsteuerungsanwendungen

eine Prioritätszuweisung kann

- statisch sein
- von Nachrichteninhalt abhängig sein

die meisten Protokolle kennen zwei Prioritätsebenen

- normaler Datentransfer
- beschleunigter Datentransfer

Eigenschaften beschleunigten Datentransfers

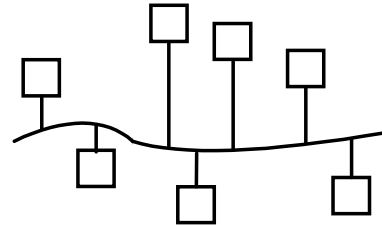
- kurze Nachrichten
- geht an Flußkontrollmechanismen vorbei

## 4 Netztopologien

Netztopologien können grob eingeteilt werden in

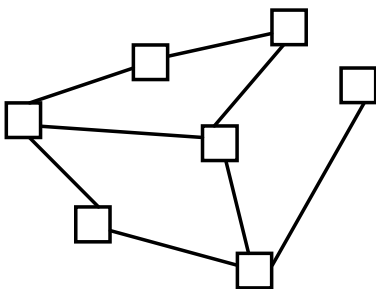
### Diffusionsnetz (Broadcast): (meist LAN)

Alle Stationen sind an ein Übertragungsmedium angeschlossen; Eine Nachricht erreicht somit alle anderen Stationen (broadcast = verbreiten, auch im Sinne von Rundfunk)



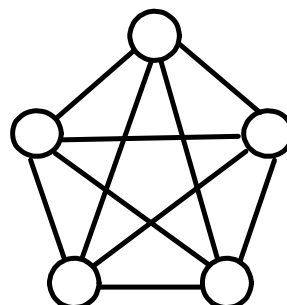
### Teilstreckennetz (Store-and-forward): (meist WAN)

Nachrichten werden in Zwischenknoten auf dem Weg zum Ziel gepuffert und weitertransportiert; unabhängige Punkt-zu-Punkt-Verbindungsleitungen



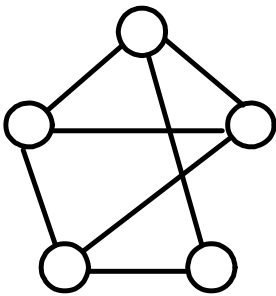
### vollständiger Verbund

- jede Station ist mit jeder anderen Station Punkt-zu-Punkt verbunden
- es können Mehrfachverbindungen zwecks Leistungssteigerung existieren
- kann zuverlässig sein, wegen alternativen Wegen und Routing
- Nachteil: teuer; bei  $n$  Stationen  $((n-1)n/2)$  Verbindungen; jede Station braucht  $(n-1)$  Schnittstellen
- Anwendungen nicht weit verbreitet wegen hoher Kosten; kleine lokale Netze; militärische Anwendungen, wo Redundanz wichtig ist



### vermaschtes Netz (partieller Verbund)

- Punkt-zu-Punkt-Verbindungen zwischen einigen Stationen
- Netzwerk muß Routing-Funktionen besitzen
- Möglichkeit der alternativen Wege, wenn jede Station mit mindestens zwei anderen verbunden ist
- nur zwei zusätzliche Verbindungsleitungen bei zusätzlicher Station
- höhere Verzögerungen
- Platzierung der Verbindungsleitungen gemäß Verkehrsanforderungen
- WAN-Anwendungen, weil hohe Verfügbarkeit und niedrige Kosten

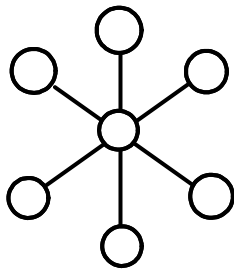


Telematik (WS 02/03)

### Stern

- alle Stationen sind mit einem zentralen Vermittlungsknoten verbunden
- niedrige Kosten
- einfache Routingtabellen
- Verzögerung und Durchsatz wird vom zentralen Knoten bestimmt
- wird verwendet, um Endgeräte (z.B. Terminals) an einen zentralen Rechner anzuschließen
- Nachteil: schlechte Zuverlässigkeit

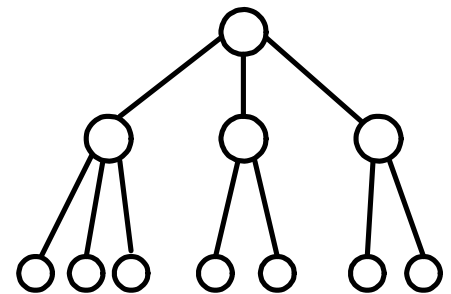
Telematik (WS 02/03)



Telematik (WS 02/03)

### Baum oder hierarchisches Netzwerk:

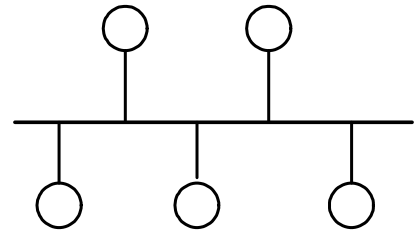
- 3-9
- ähnliche Eigenschaften wie Stern
  - z.B. Telefonnetz



Telematik (WS 02/03)

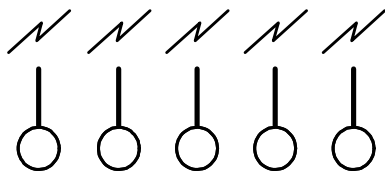
## serieller Bus oder Sammelleitung

- echte Diffusions-Topologie
- gemeinsames Übertragungsmedium
- Kontrollmechanismus für den Zugriff auf das Medium
- günstige Erweiterungskosten, nur ein Interface nötig
- einfache Rekonfigurierung des Netzes
- einfache Kommunikationssoftware, kein Routing nötig, nur Ende-zu-Ende-Flußkontrolle
- gut geeignet für lokale Netze
- Gesamtlänge auf 1-2 km beschränkt (ohne Repeater)
- jegliche Kommunikation bricht zusammen wenn Übertragungsmedium unterbrochen ist.



## drahtloses Netz:

- konzeptionell identisch zu Sammelleitung
- Funk statt Drähte oder Kabel
- es sind keine Kabel zu legen, Stationen können mobil sein
- Empfänger und Sender sind verhältnismäßig teuer



## Ring

- jede Station ist mit ihrer Nachbarstation über eine unidirektionale Verbindungsleitung zusammengeschaltet
- Kommunikation nur in eine Richtung um den Ring
- Ringinterface regeneriert das Signal und besitzt Puffer für wenige Bits (1-16)
- Übertragung kann "Diffusion" sein
- eine zusätzliche Station verursacht nur eine zusätzliche Verbindung
- leichte Rekonfigurierbarkeit
- kein Routing erforderlich
- gut für LAN geeignet
- Gesamtlänge des Rings ist nicht beschränkt
- Nachteil: jegliche Kommunikation bricht zusammen, wenn eine Leitung oder eine Station ausfällt
- Abhilfe: redundante Umgehung von Stationen (doppelt geflochtene Schleife)

