# Kommunikation und Datenhaltung

## 1. Einführung




**Prof. Dr. Martina Zitterbart**
Dipl.-Inform. Martin Röhricht
[zit | roehricht]@tm.uka.de



# Symbole


[Stal06] Literaturverweis/Webseite …


diese Folie ist zur Information – nicht prüfungsrelevant



Verweis auf andere „Zitterbart-Vorlesung“ (hier z.B. Multimediakommunikation)

# Entwicklung der Telekommunikation: Gestern, Heute, …

- 1878: Erstes Telefonnetz
  - Wenige Teilnehmer, nur Sprache, analog



- 1989: D-Einführung Integrated Services Digital Network (ISDN)
  - Sprache digital übertragen, Zusatzdienste (Makeln, Konferenzschaltung,…)
  - 64 kbit/s Standard-Übertragungsrate (B-Kanal)
- 1999: Deutschlandstart der Digital Subscriber Line (DSL) Technologie



  - 768 kbit/s Übertragungsrate zum Teilnehmer (parallel zum Telefondienst)
- 2008: 20,9 Millionen DSL-Anschlüsse in Deutschland
  - Asymmetric DSL2+ mit 16 Mbit/s zum Teilnehmer
  - Ausbau Very High Data Rate Digital Subscriber Line (VDSL) für über 25 Mbit/s

[Bund09]

# Entwicklung der Telekommunikation: … und Morgen

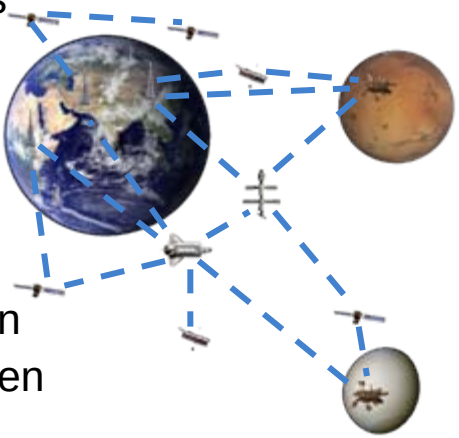
- Zukunft: Konvergenz von Netzen:
  - Anwendungen entscheidend, nicht mehr Netzzugang („Alles aus einer Steckdose“)
  - Aktuelles „Hype“-Beispiel: Triple Play [Spie06, Spie07]
    - Telephonie / Voice over IP
    - Internetzugang
    - hochauflösendes Fernsehen: HDTV-Signal benötigt etwa 5-25 Mbit/s



*Ein Anschluss: Drei Anwendungen*

⇨ Auch in Zukunft mehr Teilnehmer, Dienste und höhere Datenraten

- Im Großen: Interplanetary Internet
  - Kommunikation innerhalb des gesamten Sonnensystems
  - Vielerlei Einschränkungen:
    - Sehr große Ausbreitungsverzögerungen, schwache Signalpegel, Probleme der Energieerzeugung
  - Standardisierung von Architektur und Protokollen
- Wie im Kleinen: Sensornetze
  - Netz von winzigen, drahtlos kommunizierenden Rechnern
  - Arbeiten selbst-organisierend in Ad-hoc Netzen zusammen
  - Überwachen Umwelt mittels Sensoren



# Anwendungen der Kommunikation heute von großer Bedeutung

- Nutzung des Internet Bestandteil des alltäglichen Lebens
  - World Wide Web
  - E-Mail
  - Chat
  - Internet-Telefonie
  - Online-Spiele
- Beispiele beliebter Internet-Anwendungen
  - Skype
    - ~ 15 Mio. Benutzer gleichzeitig online
  - StudiVZ
    - ~ 13 Mio. Benutzer registriert
  - Facebook
    - 175 Mio. aktive Benutzer
    - 3 Milliarden Minuten werden jeden Tag auf Facebook verbracht
    - Mehr als 18 Mio. Benutzer aktualisieren ihren Status mindestens einmal pro Tag
  - World of Warcraft
    - ~ 12 Mio. Abonnenten

# Kommunikation – Mehr als Netzzugang und Anwendungen

- Kommunikation über das Internet benötigt mehr als
  - *„Kabel rein“* und
  - *„Anwendung starten“*
- Zwischen physikalischem Internet-Zugang und Nutzen von Anwendungen
  - Vielzahl von Problemen muss gelöst werden
    - Verteiltheit
    - Nebenläufigkeit
    - Unter Berücksichtigung von
      - Robustheit
      - Mobilität
      - Qualität
      - Sicherheit

→ Gegenstand dieser Vorlesung

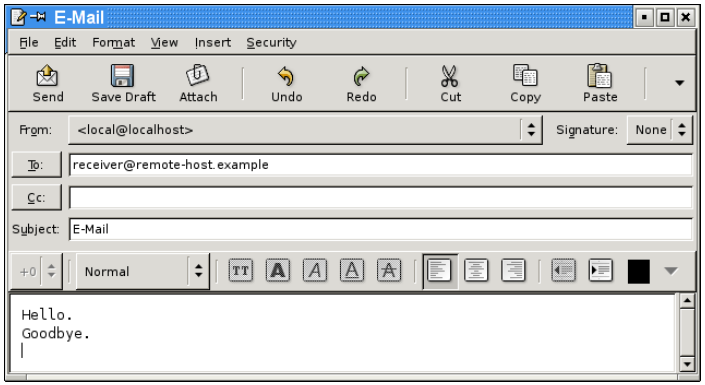
# Kapitelübersicht

1. Einführung
2. Physikalische Grundlagen
3. Protokollmechanismen
4. Geschichtete Architekturen
5. Sicherungsschicht: HDLC
6. Beschreibungsmethoden
7. Sicherungsschicht: Lokale Netze
8. Netzkopplung und Vermittlung
9. Die Transportschicht
10. Anwendungssysteme
11. Middleware

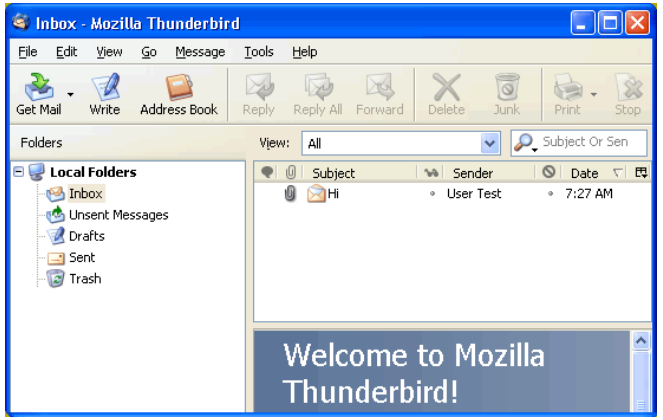
1.1 Beispiel E-Mail
1.2 Beispiel Peer-to-Peer

# 1.1 Beispiel E-Mail

- E-Mail
  - Immense Nutzung [Stand 2007]: [Radi07]
    - 1,2 Milliarden E-Mail-Nutzer mit 2,4 Milliarden E-Mail-Accounts
    - 170 Milliarden E-Mails / Tag (72% Spam)
  - Praktisch als allgemein verfügbar wahrgenommen
  - Funktioniert meist problemlos
  - Anwendung trivial:

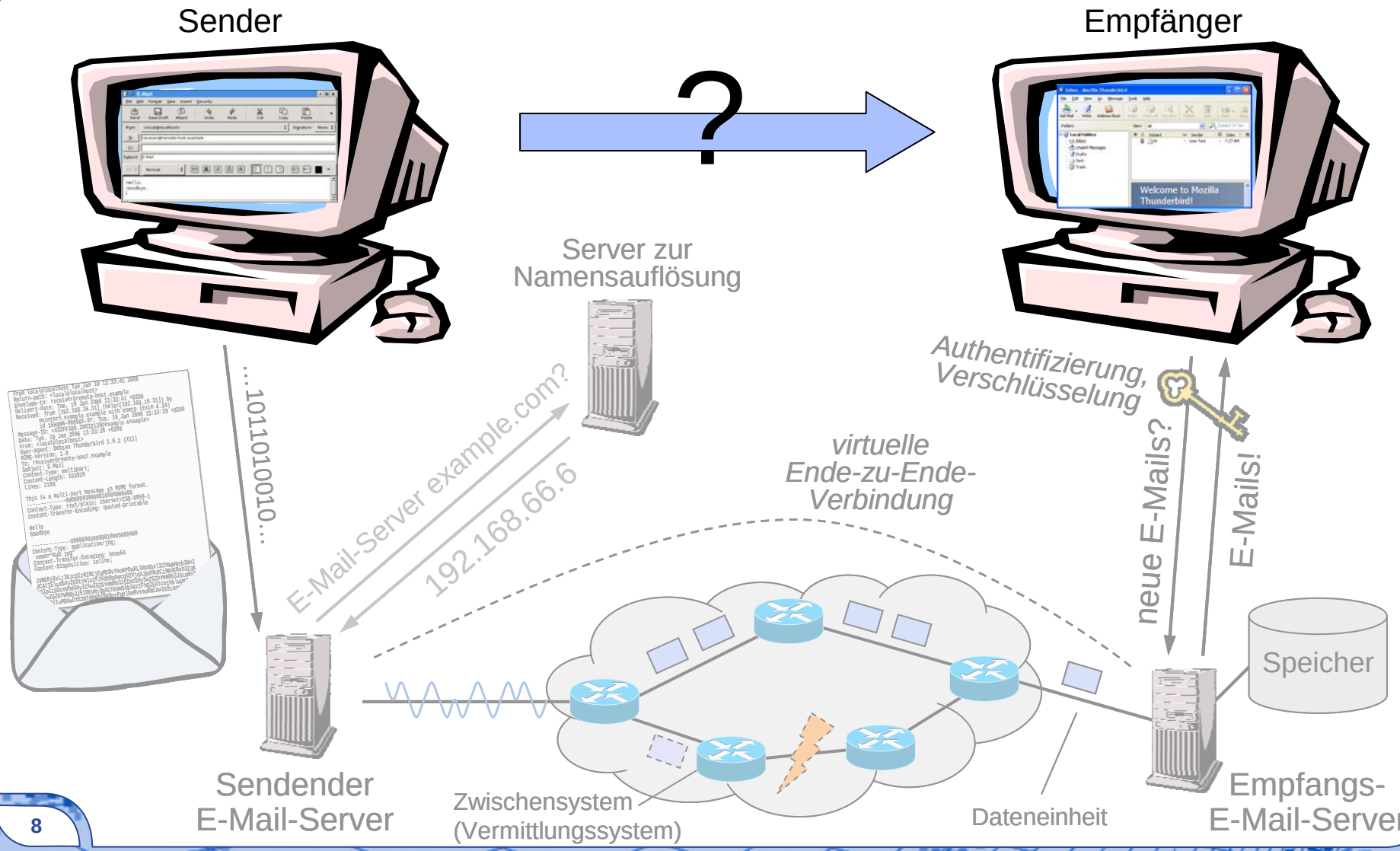



*Verfassen und Absenden einer Nachricht*




*Empfangen einer Nachricht*

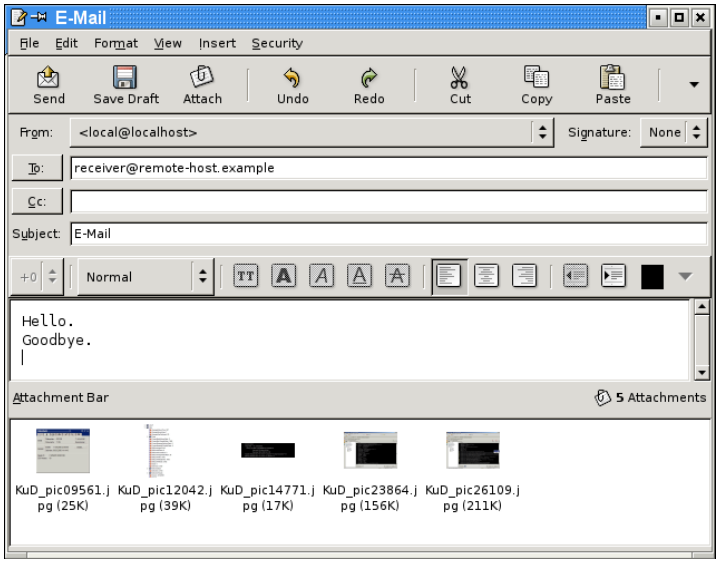
# Beispiel E-Mail – „Big Picture“



# Beispiel E-Mail

- Aber:
  Was passiert alles „hinter den Kulissen“?

- E-Mail-Kopf, Text und Anhänge transportfähig machen




Codierung

```
From local@localhost Tue Jan 19 13:33:41 2006
Return-path: <local@localhost>
Envelope-to: receiver@remote-host.example
Delivery-date: Tue, 19 Jan 2006 13:33:41 +0200
Received: from [192.168.16.31] (helo=[192.168.16.31]) by
        mxintern.example.example with esmtp (Exim 4.34)
        id 1EHgNN-00058X-9T; Tue, 19 Jan 2006 13:33:29 +0200
Message-ID: <432FF388.20632139@example.example>
Date: Tue, 19 Jan 2006 13:33:28 +0200
From: <local@localhost>
User-Agent: Debian Thunderbird 1.0.2 (X11)
MIME-Version: 1.0
To: receiver@remote-host.example
Subject: E-Mail
Content-Type: multipart;
Content-Length: 153929
Lines: 2198

This is a multi-part message in MIME format.
--------------000809030800010905000400
Content-Type: text/plain; charset=ISO-8859-1
Content-Transfer-Encoding: quoted-printable

Hello
Goodbye

--------------000809030800010905000400
Content-Type: application/jpg;
 name="KuD.jpg"
Content-Transfer-Encoding: base64
Content-Disposition: inline;

JVBERi0xLjIKJcDIzNINCjEgMCBvYmoKPDwKL1RpdGxlIChNaWNyb3NvZ
dCAtIEludGVyZG9tYWluIFJvdXRpbmcgU2VjdXJpdHkpCi9BdXRob3IgK
b3IpCi9DcmVhdG9yICh3ZGZGYWN0b3J5IHd3dy5wZGZmYWN0b3J5LmNvb
ICh3ZGZGYWN0b3J5IDEuNjQgXChXaW5kb3dzIFhQIEdlcm1hblwpKQovQ
ICh EOjIwMDUwOTE2MTQ0NzU3KQo+PgplbmRvYmoKNCAwIG9iago8PAovR
```

# Beispiel E-Mail

- E-Mail an eigenen E-Mail-Server (zum Versand an Empfänger) übergeben
  - Nutzung des lokalen Netzes
    - z.B. Ethernet oder WLAN
- Verbindung mit Empfangsserver
  - Eigener E-Mail-Server benötigt zur Zustellung Verbindung zum E-Mail-Server des Empfängers
    - Was ist eine Verbindung?
    - Wie wird sie auf- und wieder abgebaut?
    - Worauf „lauscht“ der Empfangsserver und wie erreicht man ihn?

# Beispiel E-Mail

- Empfängersuche: Zustellung an welchen Server?
  - Auflösen der Empfänger-E-Mail-Adresse in Netzadresse
    - Abfrage an Domain Name System (DNS)
    - Globale, verteilte, hierarchische Datenbank
      - Hochgradig performant [Stand: September 2007] [Karr08]
        - Belastung aller „Hauptserver“ 118 000 Anfragen / Sekunde, somit 8µs pro Anfrage
        - Im Durchschnitt 10 Milliarden Anfragen pro Tag
      - Skalierbar
        - Allein fast 13 Millionen .de-Domains [Deni09]

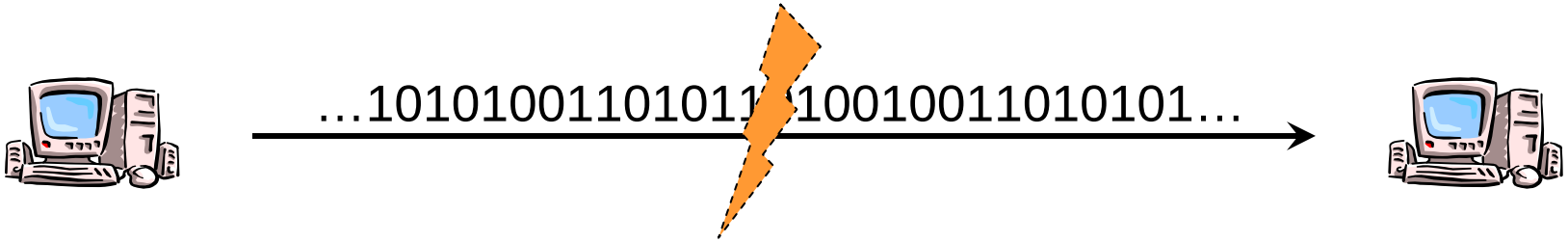
*Beispiel: Suche E-Mail-Server-Adresse der Domain „@tm.uka.de“:*

```
> dig tm.uka.de MX
[…]
;; ANSWER SECTION:
tm.uka.de.              86400   IN      MX      20 iramx2.ira.uni-karlsruhe.de.
tm.uka.de.              86400   IN      MX      10 iramx1.ira.uni-karlsruhe.de.

> dig iramx1.ira.uni-karlsruhe.de
[…]
;; ANSWER SECTION:
iramx1.ira.uni-karlsruhe.de. 29132 IN   A       141.3.10.80
```

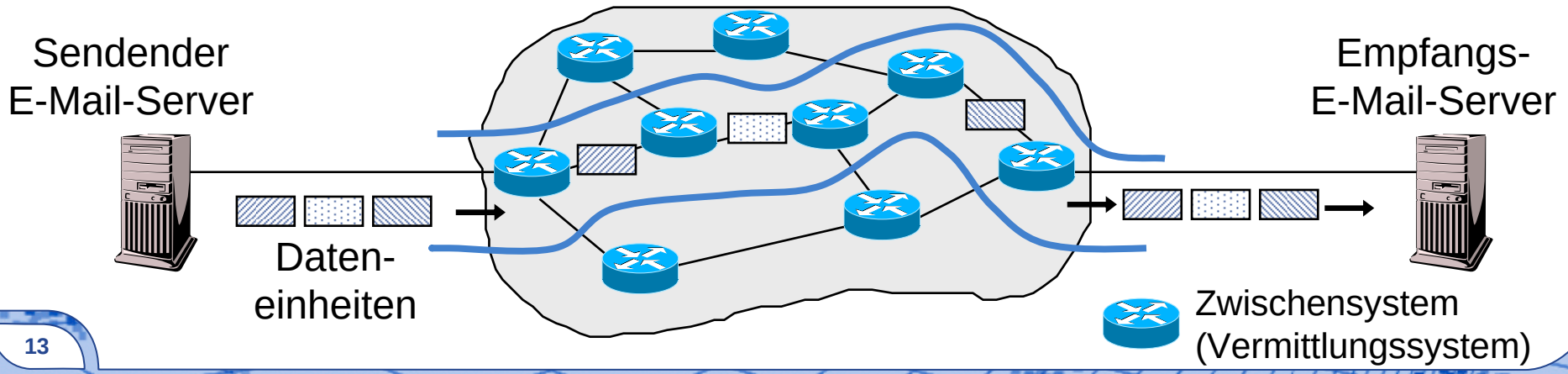
# Beispiel E-Mail

- Zuverlässiger Transport zum Empfänger
  - Übertragung muss mit dem unzuverlässigen Internet zurechtkommen, z.B. durch Mechanismen wie
    - Prüfsummen
    - Zeitgeber (Timer)
    - Nötigenfalls Wiederholung von Daten
  - Protokolle regeln die Kommunikation
    - Satz von Regeln und Formaten, nach denen die Interaktion abgewickelt werden muss
    - Dienen der Abstimmung der beteiligten Prozesse und stellen die erforderliche Disziplin her



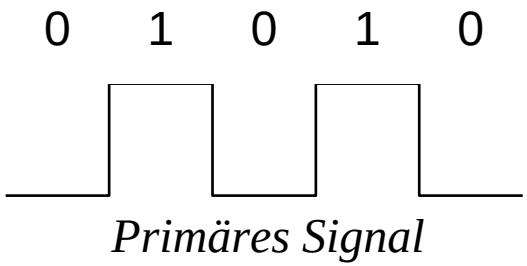
# Beispiel E-Mail

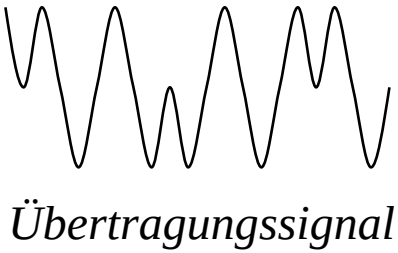
- Transport über das paket-vermittelte Internet
  - Datenstrom in „Dateneinheiten“ (Pakete) zerteilen
    - Jedes Paket benötigt Informationen zum voneinander unabhängigen Transport und dem späteren Wiederzusammensetzen des Ausgangsstroms
  - Wegewahl (Routing)
    - Zwischensysteme müssen Weg(e) zum Ziel (hier: Empfangsserver) finden
      - Wege können sich dynamisch ändern (z.B. nach Leitungsausfall)

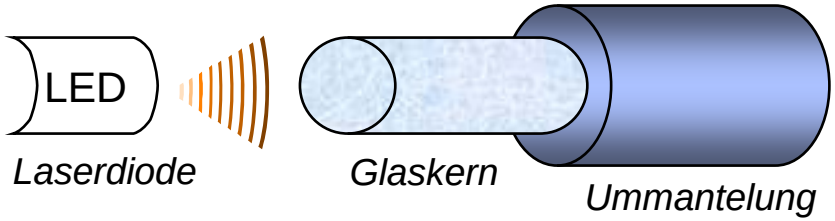


# Beispiel E-Mail

- Transport über physikalische Leitungen
  - Codierung in Bitstrom
    - verschiedene Codierungsverfahren
  - Modulation auf das genutzte Medium
    - z.B. Funkwellen, Glasfasern, Twisted-Pair-Kabel
  - Störeinflüsse in Betracht ziehen
    - z.B. durch Fehlerkorrekturmechanismen robuste Übertragung anstreben



# Beispiel E-Mail

- Empfangen und Bereithalten der Nachricht für den Empfänger
  - verschiedene Formen möglich
    - von einfachen Dateien bis zur Datenbank
  - hohe Anforderungen an Skalierbarkeit
    - viele Abfragen und quasi pausenlos eingehende E-Mails
    - GMX und 1&1-Gruppe: 5 Milliarden E-Mails/Monat, ca. 2000 E-Mails/s ⇨ 0,5 ms „Zeit“ pro E-Mail
    - T-Online täglich 1 Milliarde Spam-E-Mails, 30 Millionen zugestellte E-Mails [Stand: April 2006]
  - Mechanismen, um gleichzeitigen Zugriff zu regeln
    - Locking-Verfahren
    - Backup im Betrieb
- E-Mail-Aktion (Eingang, Abrufen, Löschen, Verschieben) in etwa vergleichbar mit Datenbank-Transaktion



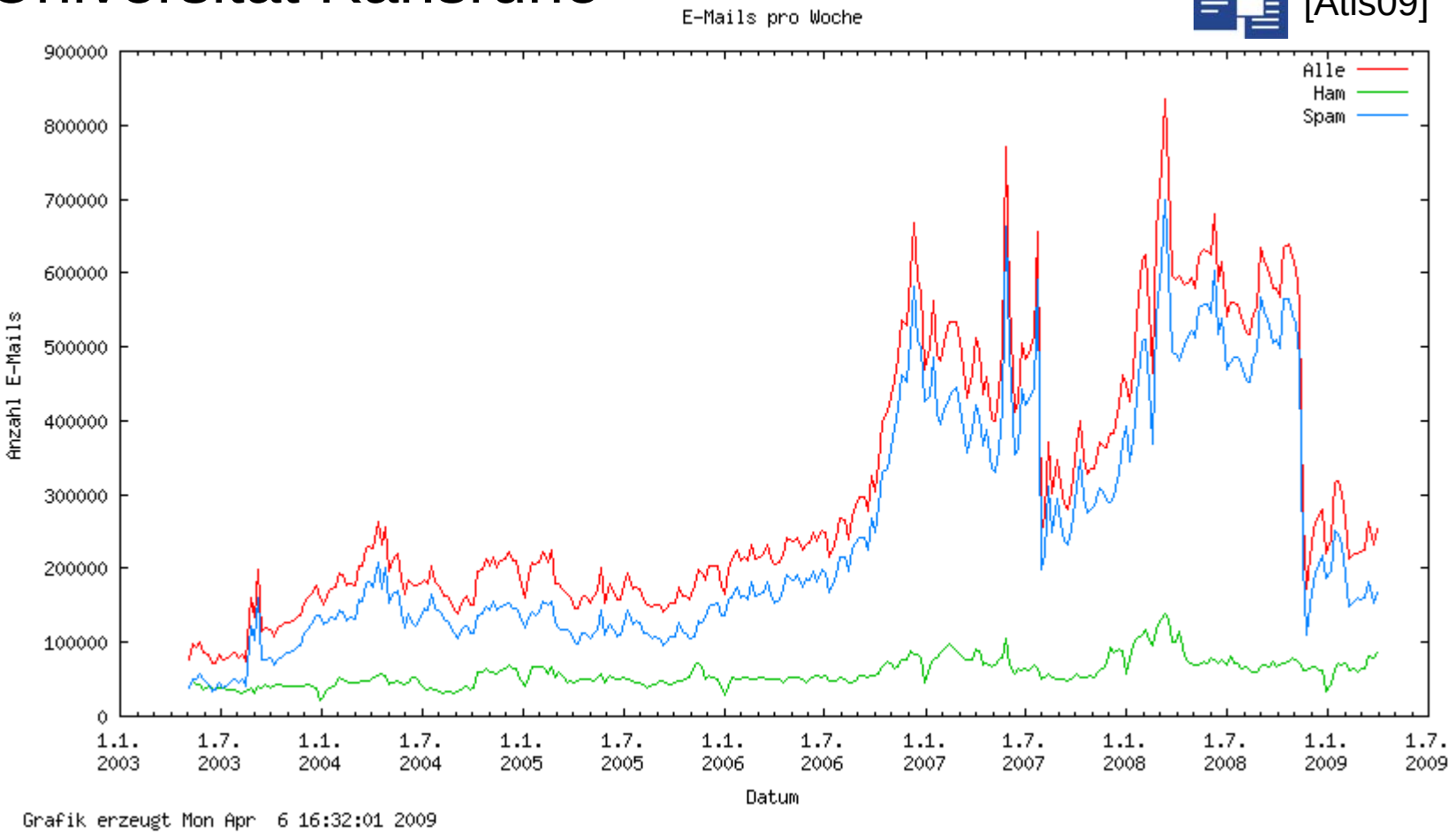
[Eins09]

[Heis06]

# Beispiel E-Mail: Spamaufkommen

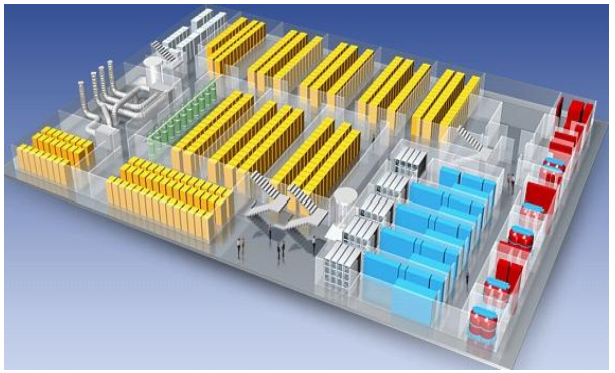
- Spamaufkommen an der Fakultät für Informatik der Universität Karlsruhe

[Atis09]



# Beispiel E-Mail: Exkurs: Das Internet ist nicht virtuell

- Rechenzentrum der 1&1 AG in Karlsruhe
  - 2000m² Grundfläche UG, 1200m² Dachfläche Technik
  - Investitionskosten: 12 Mio. Euro
  - 11 Räume für 660 Racks
    - mehr als 25000 Server
    - 4000 TeraByte Speicherkapazität
  - 5 Elektroversorgungsblöcke (je 2 MW Leistung)
  - über 8 MW Leistungsaufnahme
    - (Stadt Karlsruhe: ca. 361 MW)
  - über 70.000.000 kWh pro Jahr
  - 5 Generatoren mit jeweils 2.4 MW
    - 48.000 Liter Diesel/Tag
  - 8 Kaltwassersätze mit je 700 kW (= 5600 kW)
  - 61 Umluftkühlgeräte mit je 100 kW





[Eins09]

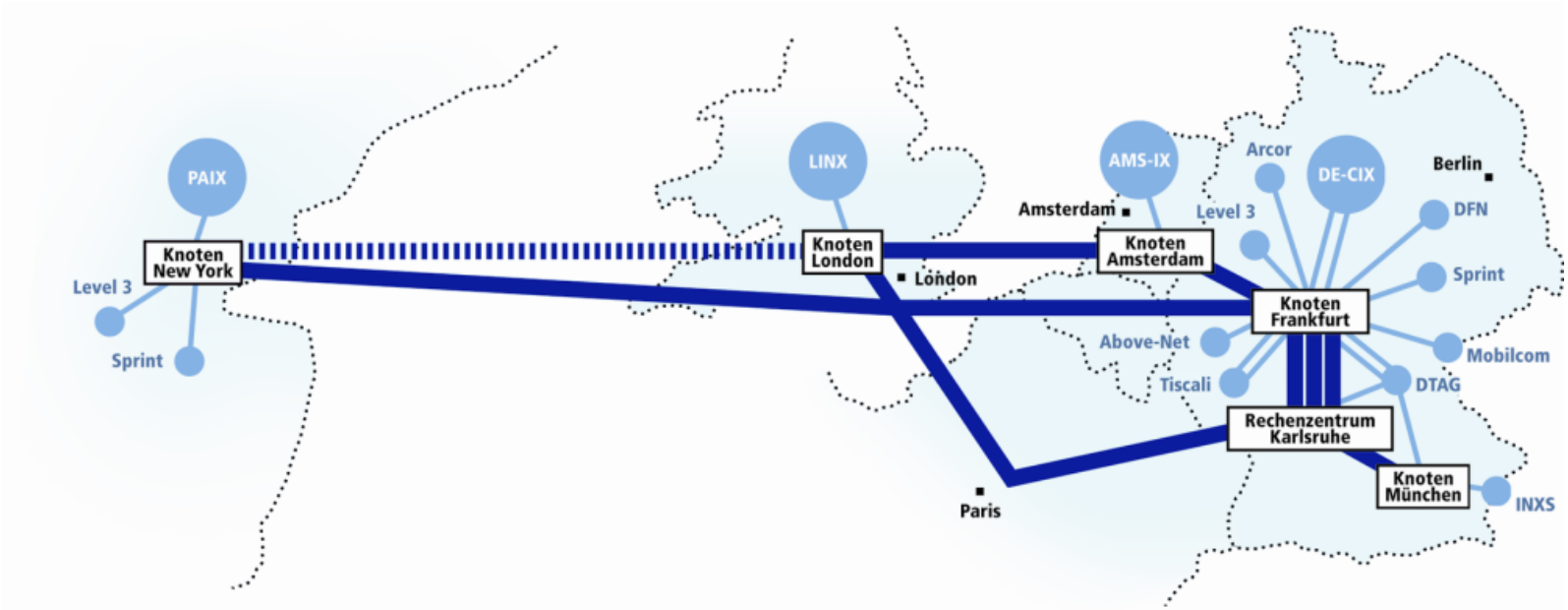


# Beispiel E-Mail: Exkurs: Das Internet ist nicht virtuell

- Stromausfall in Karlsruhe am 30.01.2008 [Heis08]
  - Brand am Umspannwerk am Rheinhafen
  - Kompletter Stromausfall im Westen und Osten der Stadt ab 17:35
    - Etwa 100.000 Bewohner betroffen
  - Erst ab 19:00 wieder umfassende Stromversorgung

- Notfallplan des 1&1-Rechenzentrums
  - Nach Unterschreitung der Netzspannung innerhalb definierten Bereichs Abtrennung von öffentl. Stromnetz
  - Akkugepufferte unterbrechungsfreie Stromversorgung (USV) liefert Strom für etwa 17 Minuten
  - Gleichzeitiges Anspringen von 4 Dieselmotoren
    - Motoröl ständig auf Betriebstemperatur
    - Diesel vorrätig für 24 Stunden Betrieb
  - Nach etwa 20 Sekunden beginnen Aggregate, Akkus nachzuladen

# Beispiel E-Mail:
# Exkurs: Das Internet ist nicht virtuell

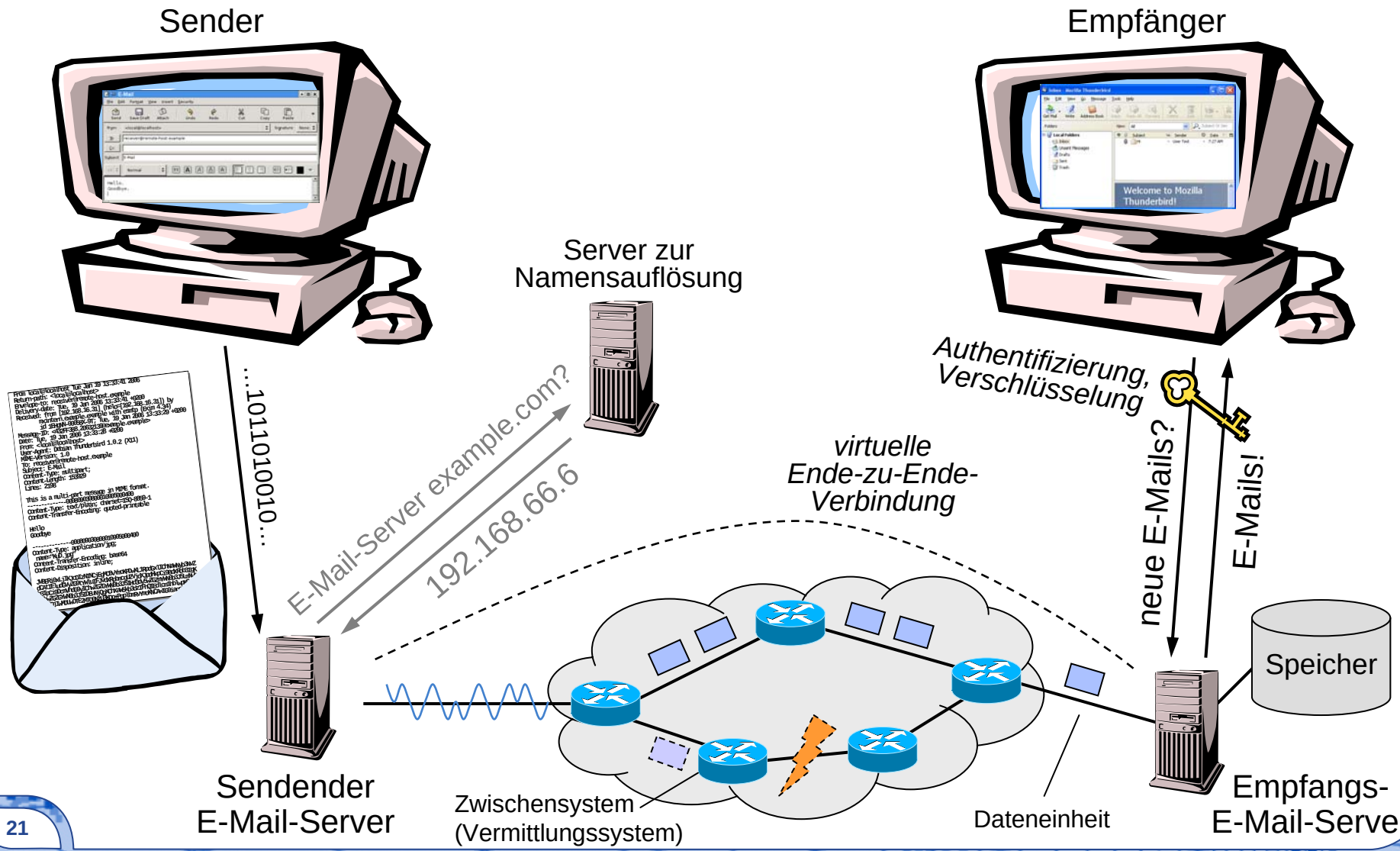



- Netzanbindung der 1&1 AG in Karlsruhe
  - 40 Gbit/s Außenanbindung
    - 2 x 3000 Mbit/s-Anbindung DE-CIX
    - 1 x 2500 Mbit/s und 1 x 1000 Mbit/s zu Mobilcom
    - 3 x 1000 Mbit/s zu LINX, Level3
    - 2 x 1000 Mbit/s zu Tiscali
    - 1 x 1000 Mbit/s zu INXS und Sprint
  - 6500 TeraByte Transfervolumen / Monat

# Beispiel E-Mail

- Abholen der Nachricht durch den Empfänger
  - Protokoll zum Abruf der E-Mails vom Server
  - Authentifizierung und Autorisierung
    - Empfänger muss sich „ausweisen“, darf nur eigene E-Mails abrufen
  - Auspacken der Nachricht aus dem Transportformat
    - Nachrichtenkopf darstellen
    - Text darstellen
    - Anhänge abspeichern
- Allein beim Thema E-Mail noch viele weitere Aspekte und Techniken, z.B. bezüglich
  - Skalierbarkeit
  - Verschlüsselung
  - etc.

# Beispiel E-Mail – „Big Picture“



# 1.2 Beispiel Peer-to-Peer (P2P)

- Grundidee: Keine festen Client-/Server-Rollen im Netz
  - Clients können auch Server sein und Dienste anbieten
  - Prominentes Beispiel in junger Zeit: Filesharing („Tauschbörsen“)
    - Eigentlicher Zweck: Entlastung zentraler Server und deren Leitungen

- Fand in kurzer Zeit große Verbreitung: [Stand 2005]

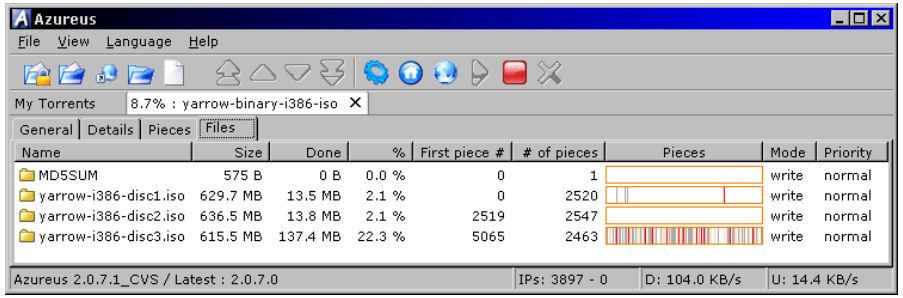


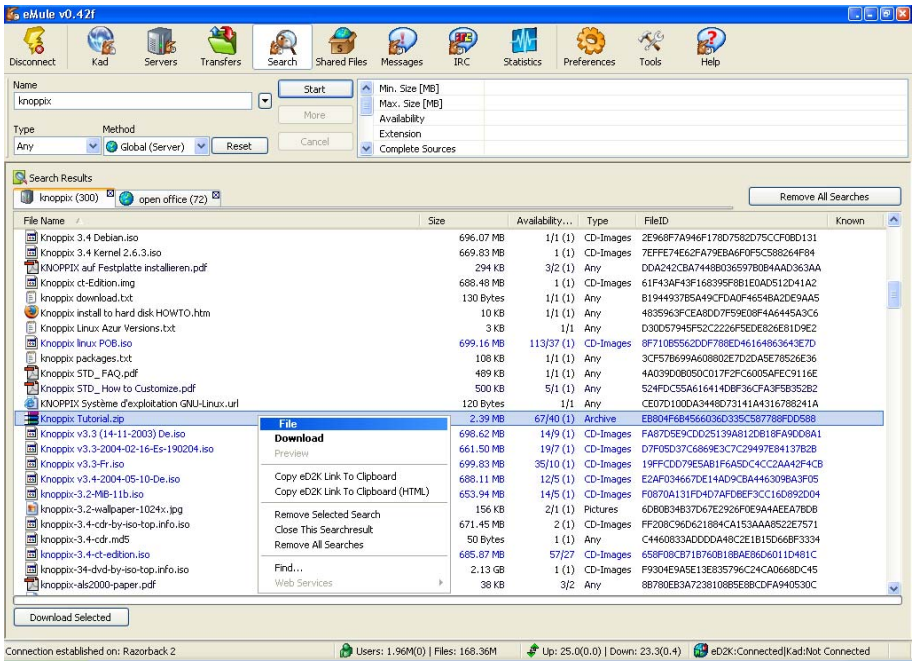
[BigC05, Sand05]

  - Anteil P2P-Verkehr an Upload-Bandbreite europäischer Provider: 85% (Download: 60%)
  - ca. 60 Millionen Nutzer allein in den USA
  - Weltweit zu jedem Zeitpunkt ca. 9 Millionen Teilnehmer angemeldet
  - Bittorrent hierbei nicht erfasst, schätzungsweise bis zu 10 Millionen zusätzliche Nutzer

# Beispiel Peer-to-Peer

- Einfache Nutzung
- Häufig problemloser Betrieb




*Client installieren,*

*ins P2P-Netzwerk einloggen,*

*Inhalte suchen,*

*von verfügbaren Quellen herunterladen*

# Beispiel Peer-to-Peer

- Aber: Welche Techniken sind hierzu unabdingbar notwendig?
  - Im Grunde technische Meisterleistung:

    "*In my humble opinion, the concept of 8.5 million computer users connected peer-to-peer through the Internet at one time is incredible (regardless of how many networks are involved).*"

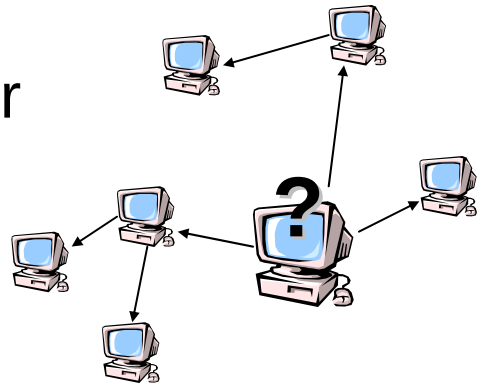
    [John Brewer, member of the Governor's and Legislative Task Force for E-Commerce, Oklahoma City]

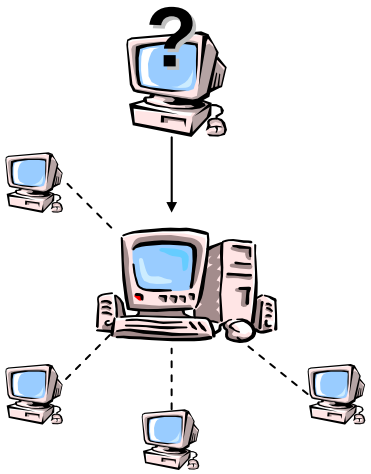
# Beispiel Peer-to-Peer

- Einloggen in ein P2P-Netz
  - Wo einloggen, wenn es keine zentralen Server gibt?
    - Verbundene P2P-Clients können Teilnehmerinformationen austauschen
    - Aber: Wo Anschluss finden, wenn noch nicht verbunden?
    - Sogenanntes Bootstrapping-Problem
  - Die meisten „Lösungen“ heutzutage „schummeln“
    - Client-Software bringt Listen von gut verfügbaren Teilnehmer-Systemen mit
    - Zentrale Ablage von Teilnehmer-Systemen (z.B. im WWW)
    - Widersprüche zum puristischen P2P-Konzept
      - Vorgefertigte Listen können veralten
      - Zentral abrufbare Listen können ausfallen oder abgeschaltet werden

# Beispiel Peer-to-Peer

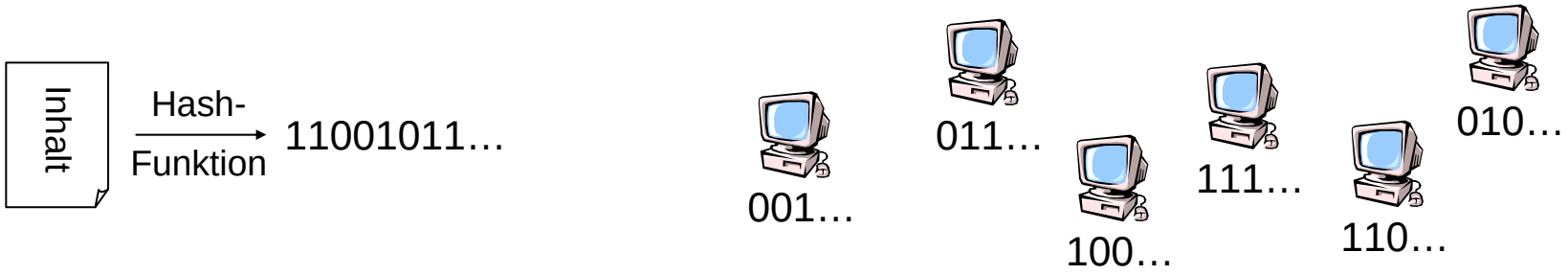
- Inhalte suchen
  - Ziel: Innerhalb der Millionen Teilnehmer diejenigen finden, die gewünschte Inhalte vorhalten
    - Suche muss auch effizient sein (Skalierbarkeit)
      - Gnutella-Netz aufgrund von Suchanfragen-Überlastung zeitweise zerfallen
  - Verschiedene Möglichkeiten:
    - Suchanfragen an alle Teilnehmer im Netz
      - verbietet sich aus Skalierbarkeitsaspekten (s.o.)
    - Listen von Quellen für Inhalte führen
      - z.B. auf besonders leistungsfähigen Client-Systemen
        - verteilte Indizes



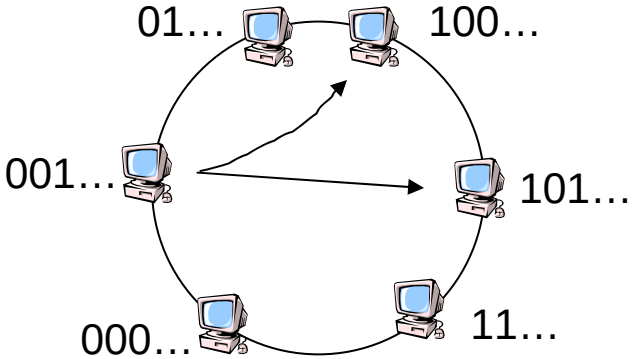



# Beispiel Peer-to-Peer

- Verschiedene Möglichkeiten (2):
  - Clients sind für bestimmte Inhalte dediziert „zuständig“
    - Hash-Werte als eindeutiger Schlüssel für Inhalte
    - Aufteilung des möglichen Suchraums auf die Clients




  - Suchanfragen „clever“ durch das P2P-Netz weiterleiten
    - z.B. an Clients, die voraussichtlich weiter wissen könnten



# Beispiel Peer-to-Peer

- Inhalte abrufen
  - Verbindungsaufbau zu den Teilnehmern, die gesuchte Inhalte vorhalten
    - Probleme durch Adress-Umsetzungen (*Network Address Translation*, NAT) und Firewalls
  - Sicherstellen, dass Download wirklich gesuchter Inhalt ist
    - Prüfsummen, Hash-Verfahren
  - Abruf von mehreren Quellen gleichzeitig
    - Aufteilung gesuchter Inhalte in Stücke
    - Sicherstellen, dass Stücke zusammenpassen
  - „Fairness“ fördern
    - Es kann nicht nur Downloader geben
    - Nutzer sollten auch Inhalte weiterverteilen

# Beispiel Peer-to-Peer

- Inhalte publizieren
  - Unterschiedliche Verfahren zur Einspeisung von Inhalten in ein P2P-Netz
    - Simples Zur-Verfügung-Stellen von Inhalten und entsprechenden Eintrag in die Suchmöglichkeiten
      - z.B. Publikation von Zeigern auf die Inhalte im WWW
    - Aktiver Upload auf die Teilnehmer, deren Client für den Inhalt zuständig ist
- Ausloggen aus einem P2P-Netzwerk
  - Durchschnittliche Verweildauer eines Clients: 1,5 h
  - Je nach P2P-Protokoll evtl. sinnvoll, Daten für die der Client zuständig ist zuvor auf andere Clients zu verteilen

# Beispiel Peer-to-Peer

- Sicherheit / Robustheit von P2P-Netzen
  - Neben Implementierungsfehlern teils auch grundlegende Konzeptprobleme
  - Beispiele:
    - Zentrale Elemente ausschalten (in nicht vollständig P2P-basierten Netzen)
      - Such-/Anmeldeserver
      - Inhaltslisten
    - Unfaire Clients
      - Nur Herunterladen, aber keine Inhalte weiterverteilen
    - Bösartige Clients
      - Gefälschte Inhalte einspeisen
      - Suchanfragen nicht beantworten und nicht weiterleiten
      - Überlastung verschiedener Teile des Systems (Denial of Service, DoS)
        - Überhäufung mit Suchanfragen, Ab-/Anmeldungen, etc.

# Zusammenfassung

- Einige Fragestellungen und Anforderungen „hinter den Kulissen“ von
  - E-Mail
  - Peer-to-Peer

- Die Vorlesung KuD präsentiert
  - Grundlagen, Konzepte und Lösungen für
  - Verfahren, Protokolle und Architekturen

  um solche Systeme tatsächlich zu realisieren
  - Gibt grundlegenden Überblick
    - Vertiefung in weiterführenden Lehrveranstaltungen

# Quellen


- [Atis09] Abteilung Technische Infrastruktur, http://www.atis.uka.de/874.php
- [BigC05] BigChampagne Online Media Measurement, http://www.bigchampagne.com
- [Bund09] Bundesnetzagentur – Jahresbericht 2008, http://www.bundesnetzagentur.de
- [Deni09] DENIC, http://www.denic.de/de/domains/statistiken/index.html
- [Eins09] 1&1 Internet AG, http://www.1und1.de/index.php?&srcArea=ln&page=tech
- [Heis06] heise Newsticker, 29.4.06, http://www.heise.de/newsticker/meldung/72324
- [Heis08] heise Newsticker, 31.01.08 http://www.heise.de/newsticker/meldung/102805
- [Karr08] Daniel Karrenberg, Internet Society Member Briefing, DNS Root Name Servers – Frequently Asked Questions, Februar 2008, http://www.isoc.org/briefings/020/
- [Radi07] radicati Group, http://www.radicati.com
- [Sand05] Netzwerkausrüster Sandvine, http://www.sandvine.com/
- [Spie06] Spiegel 07/2006 „Alles wächst zusammen“ (online verfügbar)
- [Spie07] Spiegel 37/2007 „Die große Wette“ (online verfügbar)

# Kommunikation und Datenhaltung

## 2. Physikalische Grundlagen




**Prof. Dr. Martina Zitterbart**
Dipl.-Inform. Martin Röhricht
[zit | roehricht]@tm.uka.de

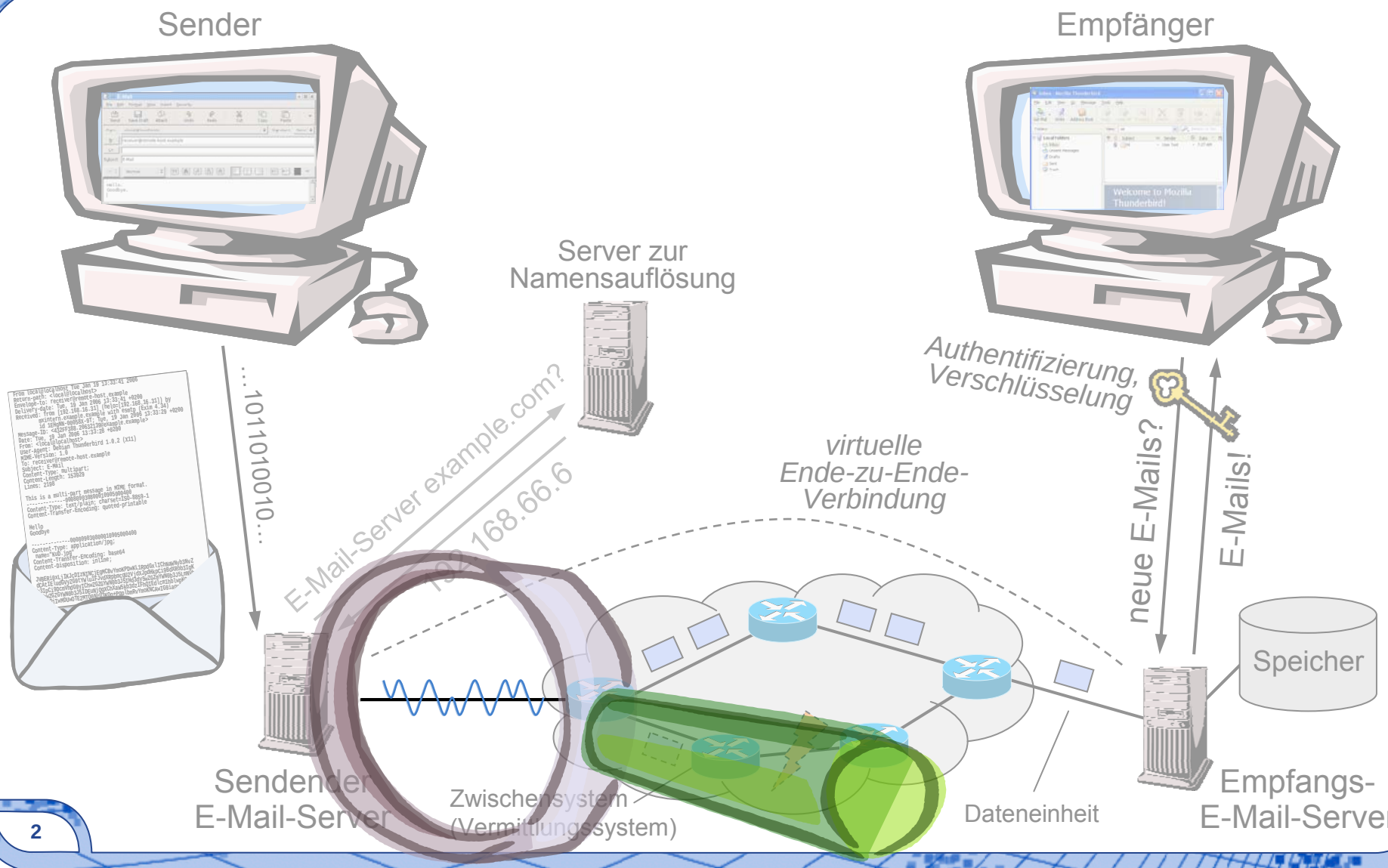


# Kapitelübersicht

1. Einführung
2. Physikalische Grundlagen
3. Protokollmechanismen
4. Geschichtete Architekturen
5. Sicherungsschicht: HDLC
6. Beschreibungsmethoden
7. Sicherungsschicht: Lokale Netze
8. Netzkopplung und Vermittlung
9. Die Transportschicht
10. Anwendungssysteme
11. Middleware

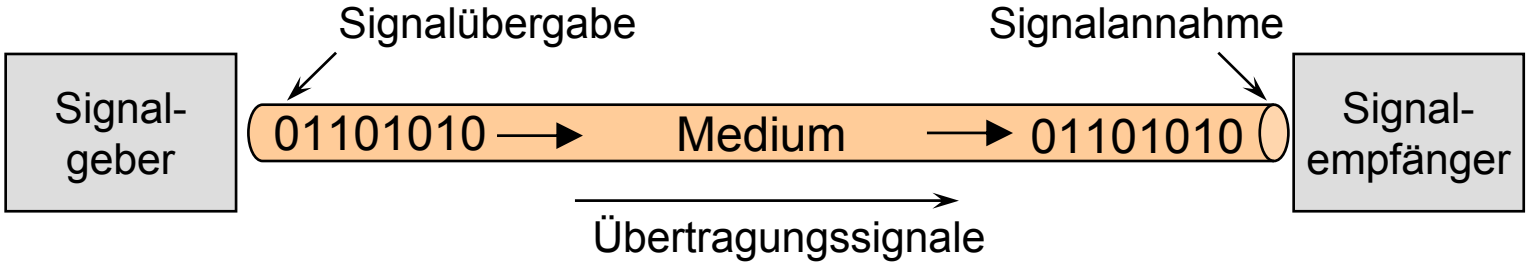
2.1 Grundlagen und Begriffe
2.2 Signalübertragung
2.3 Digitalisierung
    2.3.1 Abtasttheorem
2.4 Übertragungsmedien
2.5 Digitale Signalübertragung
2.6 Kanalkapazität
2.7 Übertragungszeit
2.8 Modulation / Demodulation
2.9 Serielle und parallele Übertragung
2.10 Synchronisation
2.11 Mehrfachnutzung / Multiplexing

# 2. Physikalische Grundlagen



# 2.1 Übertragungssystem, Grundlagen, Begriffe

- Grundlage jeder Kommunikation ist die Signalübertragung
  - Transport von Signalen über ein geeignetes Medium, das diese Signale über eine räumliche Distanz weiterleitet

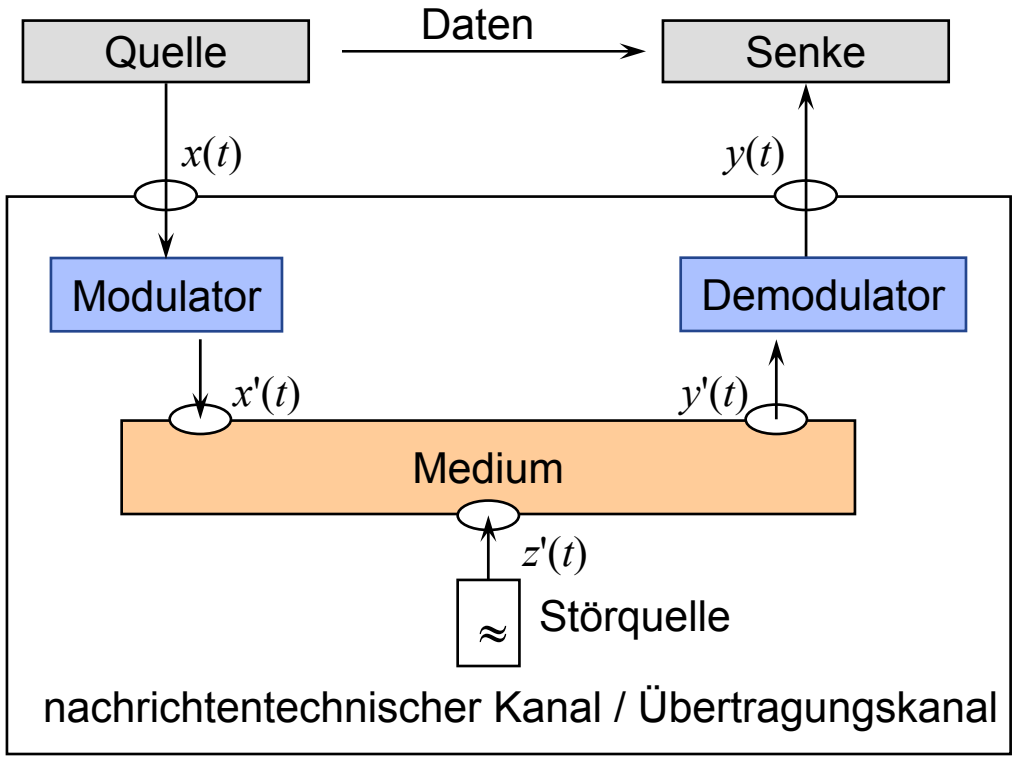



- Verkürzender Sprachgebrauch
  - Übertragungssignal = Signal
  - Signaltransportmedium/Übertragungsmedium = (physikalisches) Medium
  - Signalgeber, Signalquelle = Sender
  - Signalempfänger, Signalsenke = Empfänger
  - physikalisch-technisches Transportsystem für Signale = Übertragungsweg

- Signalübertragung wird in der Nachrichtentechnik meist als Nachrichtenübertragung bezeichnet

# Physikalisches Medium

- Verwendung eines physikalischen Mediums zur Übertragung von Daten.

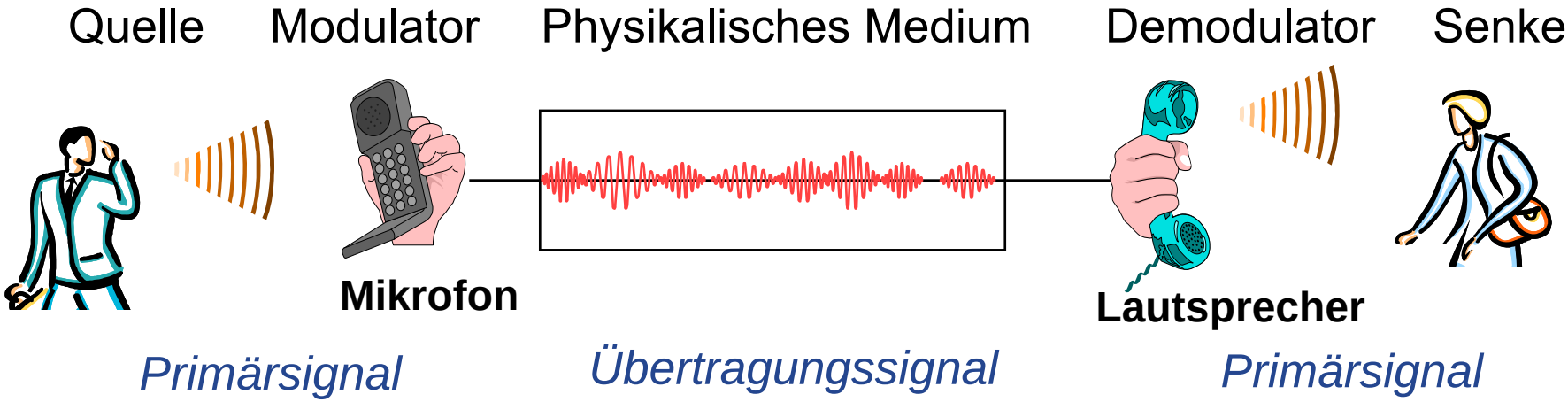



Primärsignale $x(t)$, $y(t)$:
quellen-/senkenbezogene physikalische Größen

Signale $x'(t)$, $y'(t)$, $z'(t)$:
leitungsbezogene physikalische Größen

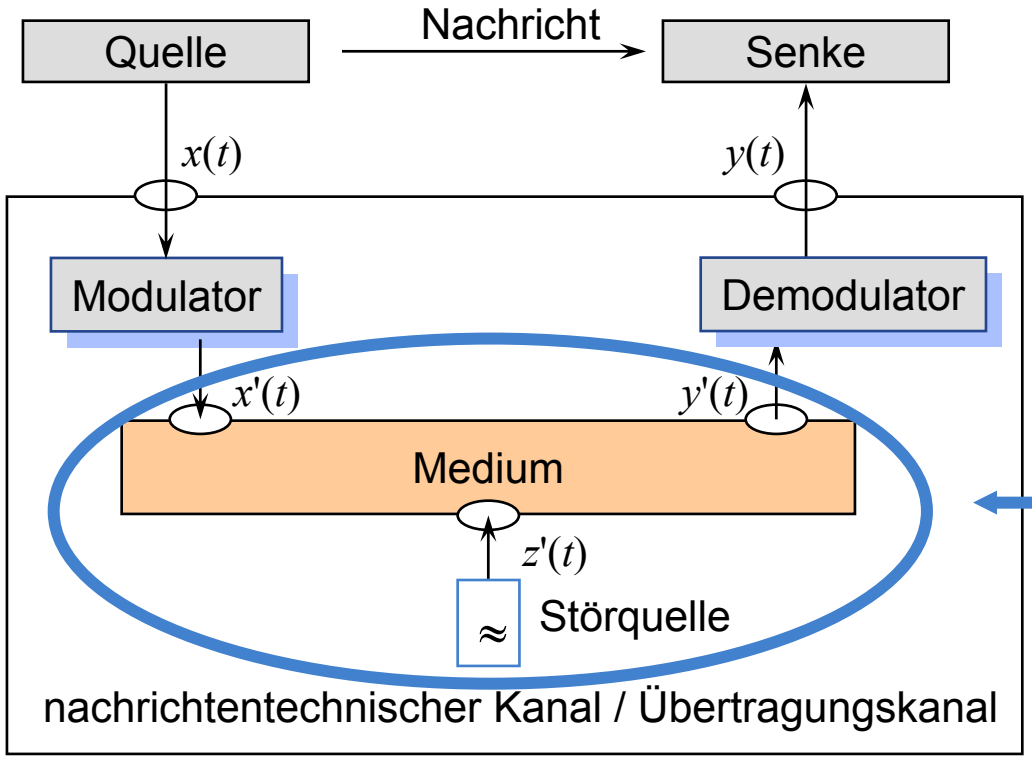
Physikalisches Medium,
z.B. elektrische Leitung:
$y'(t) = F(x'(t); z'(t))$

# Beispiel Fernsprechnetz



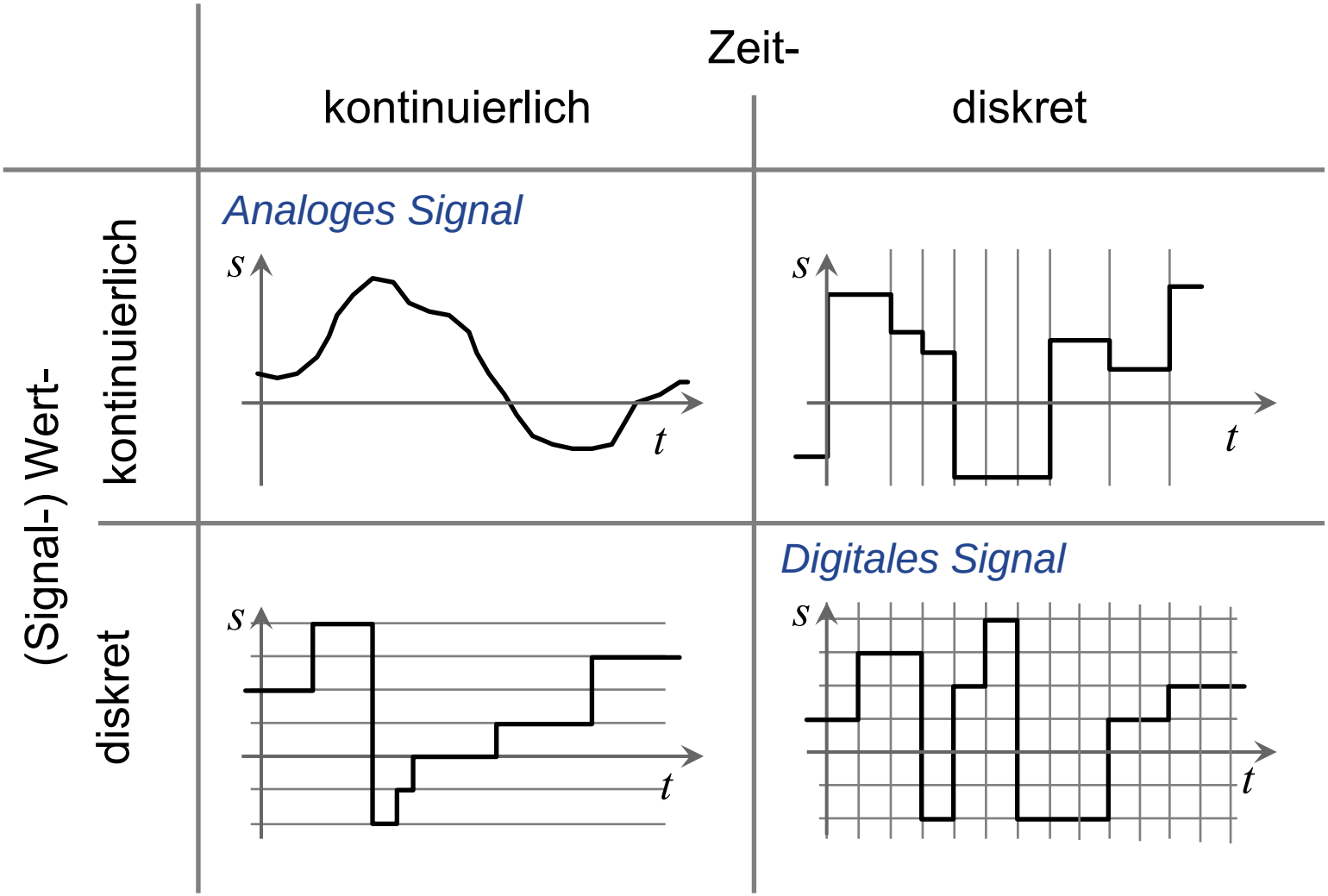

- Primärsignal (hier akustisch) wird durch Modulator in ein elektrisches oder optisches Signal umgewandelt
- Das Übertragungssignal kann analog oder auch digital sein

# 2.2 Signalübertragung




Abschnitt 2.2

# Einteilung der Signale in Signalklassen



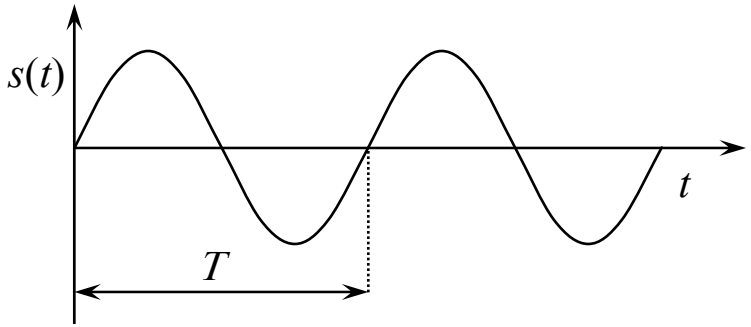
# Periodische Signale

Signalmuster wiederholt sich: $s(t+T)=s(t) \qquad -\infty < t < \infty$

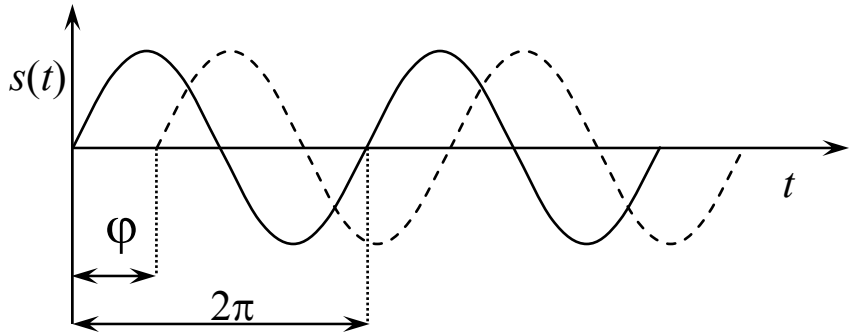
Beispiel: Sinusschwingung

- Kenngrößen: Periode $T$, Frequenz $1/T$, Auslenkung $s(t)$, Phase $\varphi$

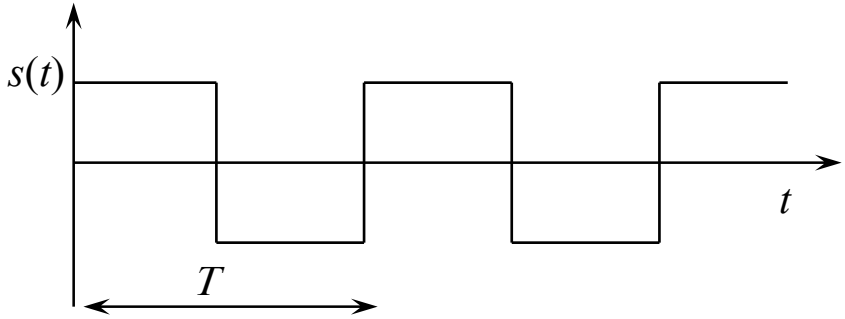
Sinus-Schwingung (zeitkontinuierlich)




Beispiel von Phasendifferenz $\varphi$

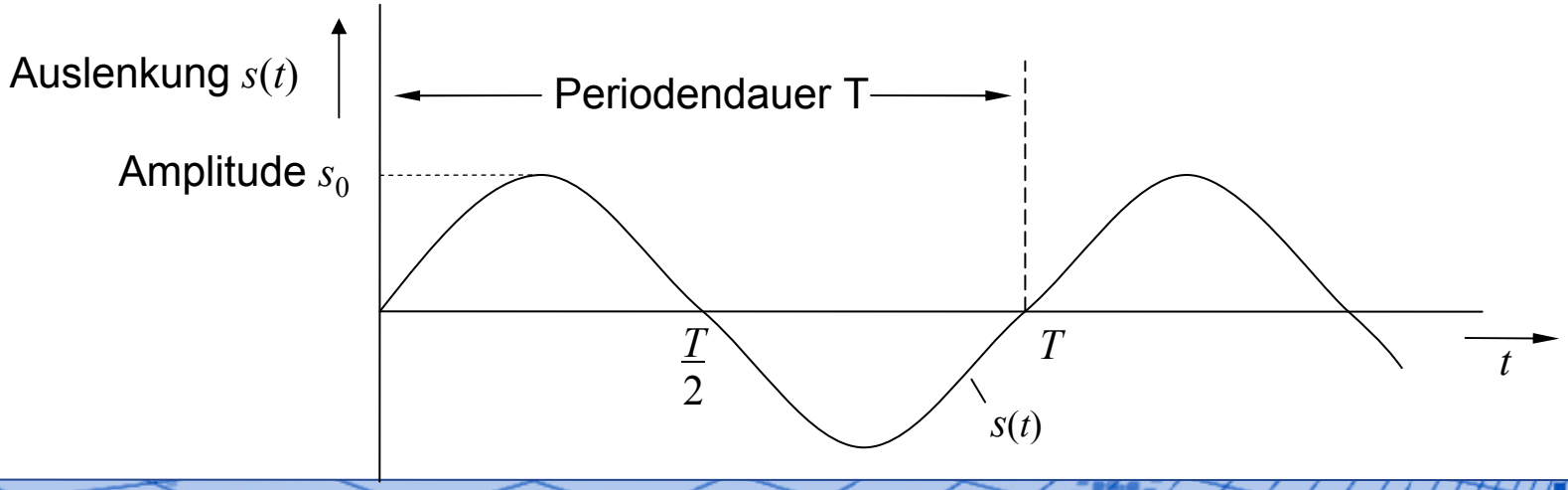



Rechteck-Schwingung (zeitdiskret „idealisiert“)



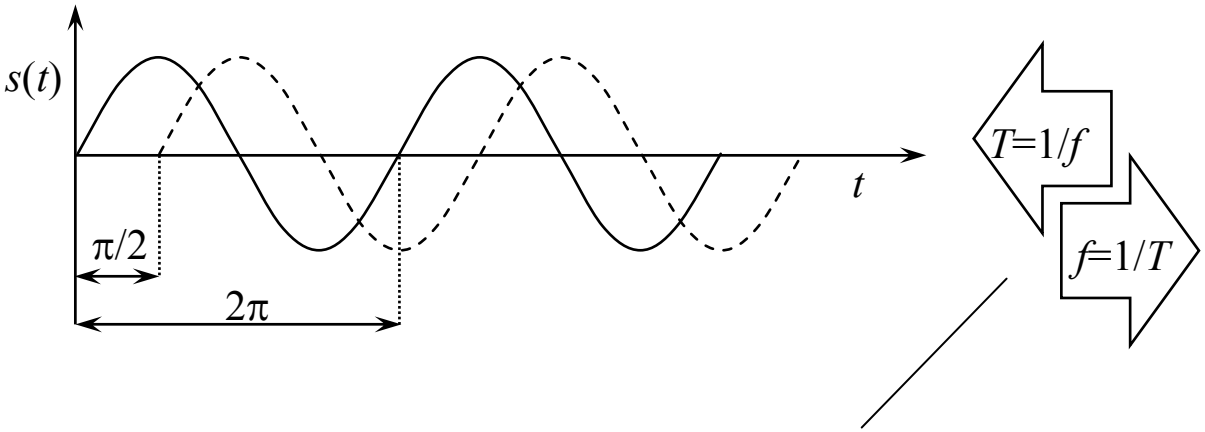
# Sinusschwingung

- Frequenz $f$ [Hertz]
  - Schwingungen pro Sekunde, $T = 1/f$
- Kreisfrequenz (Winkelgeschwindigkeit)
  - $\omega = 2\pi * f$ = Anzahl der Kreisumläufe (Winkel $2\pi$) pro Sekunde
- Innerhalb einer Zeitdauer $t$ wird ein Winkel $\varphi$ durchlaufen
  - Dann gilt $\varphi = \omega \cdot t = 2\pi f \cdot t$
- Allgemein $s(t) = s_0 \cdot \sin(\omega t + \varphi)$

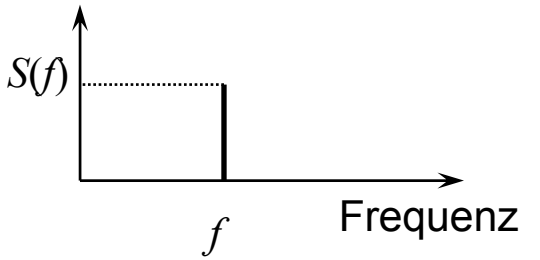


# Zeitdarstellung/Frequenzdarstellung

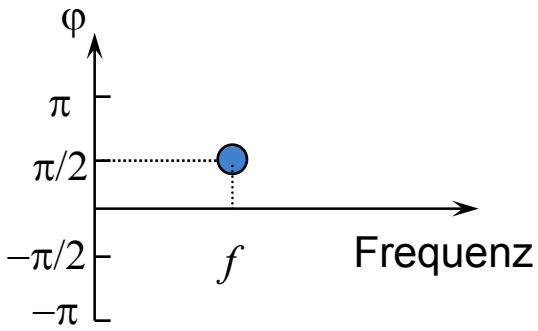
- Zeitfunktion (Zeitdarstellung)
  - Zuordnung von Signalwert und Zeit
- Frequenzfunktion (Frequenzgang, Spektrum)
  - Zuordnung von Werten sinusförmiger Signale und der Frequenz




Übergang zwischen Zeit- und Frequenzfunktion




Amplituden-Frequenzgang




Phasen-Frequenzgang

# 2.3 Digitalisierung analoger Daten

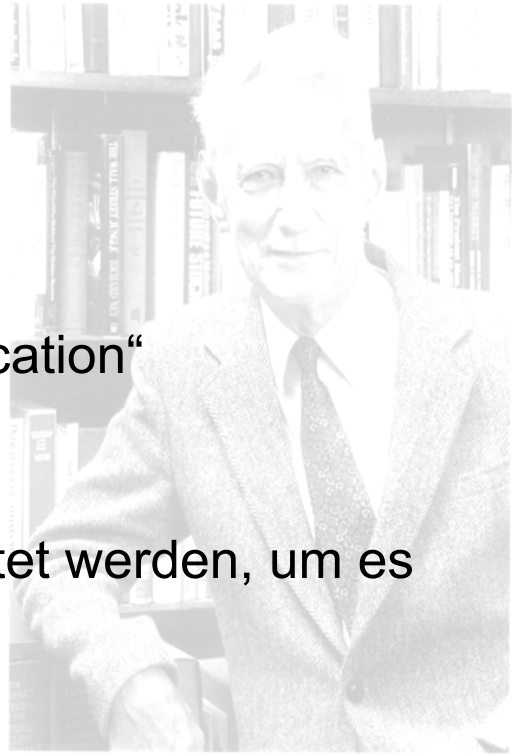
- Pulse Code Modulation (PCM)
  - Wandlung analoger Signale in digitale Signale (z.B. bei digitalen Fernsprechkanälen)
    - Abtastung $\Rightarrow$ zeitdiskretes Signal
    - Quantisierung $\Rightarrow$ zeit- und wertdiskretes Signal

- z.B. Digitalisierung von Sprache
  - Vorgaben
    - Frequenzspektrum der menschlichen Stimme
    - Fähigkeiten eines Fernsprechkanals

- **Abtasttheorem von Shannon und Raabe**
  - Die Abtastfrequenz $f_A$ muss größer als die doppelte obere Grenzfrequenz $f_{Grenz}$ sein, also
    - $f_A > 2 \times f_{Grenz}$

# Exkurs … zur Informationstheorie

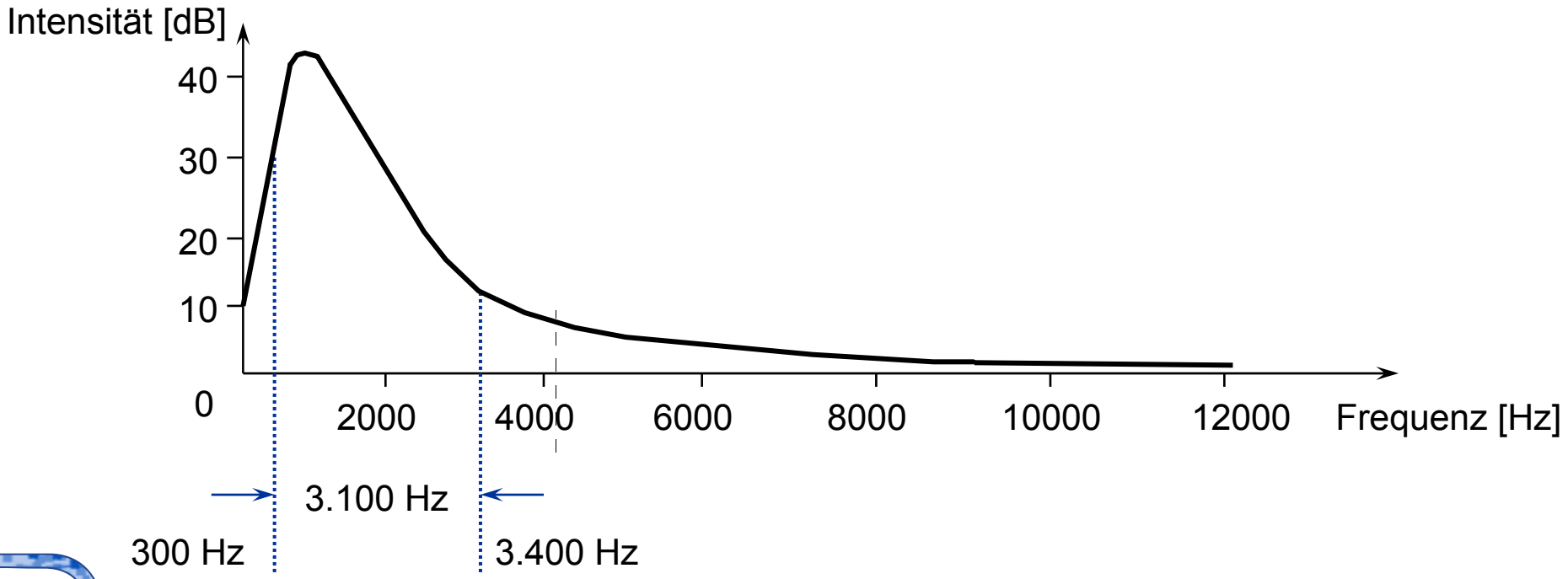
- C.E. Shannon (1916 – 2001)
  Begründer der Informationstheorie
  - sollte besser *Kommunikationstheorie* heißen, da die Semantik der zu übertragenden Daten nicht interpretiert wird
  - 1941-1956: Bell Labs
  - 1949: „The Mathematical Theory of Communication“
  - Ab 1956: MIT
- Abtasttheorem
  - Wie häufig muss ein analoges Signal abgetastet werden, um es ohne Verluste rekonstruieren zu können?
- Kanalkapazität
  - „man kann Rauschen durch erhöhte Bandbreite bekämpfen“
- Informationsgehalt
  - Empfänger weiß über den Zustand des Systems weniger als der Sender



„In jedem Taschentelefon – Handy genannt – geht ein Stück Shannon durch die Straße.“
H. Zemanek; „Claude E. Shannon“, it+ti, 4/2001

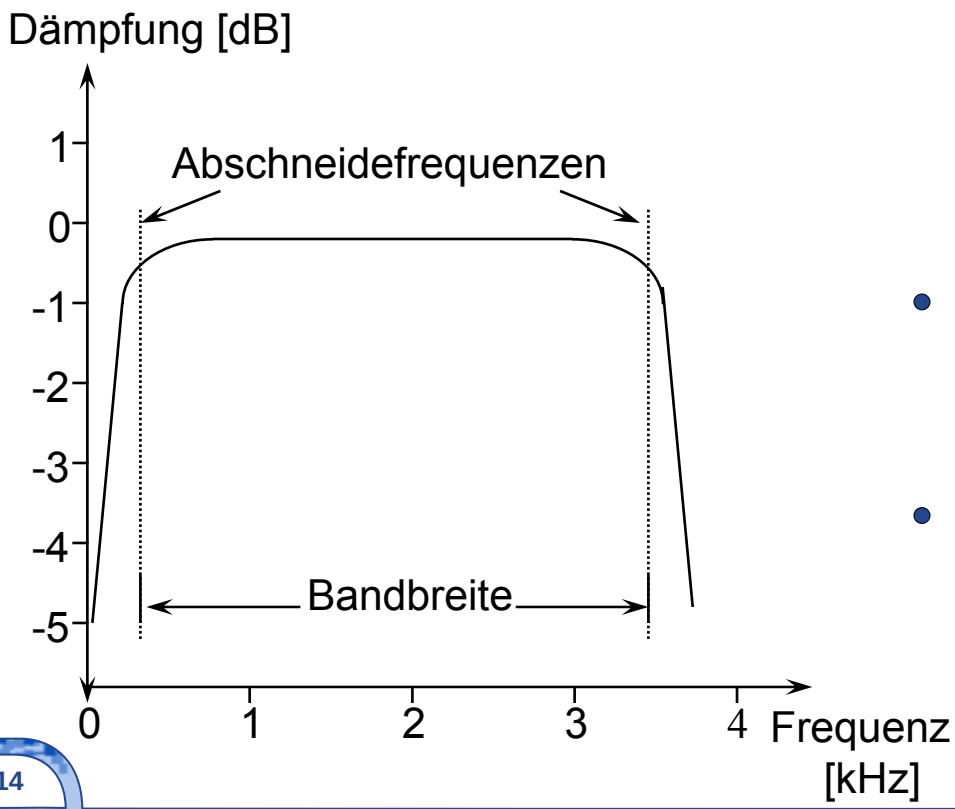
# Frequenzspektrum eines Signals

- Signale können ein „natürlich“ begrenztes – kontinuierliches – Frequenzspektrum umfassen oder durch technische Mittel auf einen Ausschnitt ihres Spektrums begrenzt werden
  - Beispiel: ITU-Standardtelefonkanal
    Kontinuierliches Frequenzspektrum der menschlichen Stimme



# Bandbegrenztes Medium

- Ausgangssituation
  - physikalische Medien übertragen stets nur ein endliches Frequenzband




- Signale müssen an die Übertragungscharakteristik des Mediums angepasst werden
- Bandbreite von Übertragungskanälen
  - Bandbreite in Hz: Frequenzbereich, der über ein Medium (einschließlich der im Übertragungskanal enthaltenen Filter, Verstärker usw.) übertragen werden kann
- Bandbreite ergibt sich aus der **Differenz der höchsten und niedrigsten Frequenzen**
- Festlegung von Abschneidefrequenzen

# Digitalisierung analoger Daten (Fortsetzung)

- Pulse Code Modulation (PCM) für digitale Fernsprechübertragung
  - Grenzfrequenz 3.400 *Hz*
  - Abtastfrequenz 8.000 *Hz* (> 6.800 *Hz*)
    - Abtastperiode:
      $T_A = 1/f_A = 1/8000\ \text{Hz} = 125\ \mu\text{s}$
  - Kodierung der Signalwerte 8 *bit*
    - 256 Quantisierungsintervalle
      Bei binärer Codierung 8 Bit erforderlich
  - Die Datenrate für einen digitalisierten Fernsprechkanal
    - Datenrate = Abtastfrequenz * Codewortlänge

8000/s * 8 bit = **64 *kbit/s***

# 2.3.1 Abtasttheorem

- Problem
  - Wie häufig muss ein analoges Signal periodisch abgetastet werden, um es verlustfrei in ein digitales Signal umzuwandeln?

- Abtasttheorem nach Shannon

  Wird das Signal $x(t)$ periodisch mit einer Frequenz abgetastet, die höher als die doppelte höchste im Signal auftretende Frequenz $f_{\text{Grenz}}$ ist, enthalten die so erfassten Werte alle Informationen des ursprünglichen Signals.

  - Die folgenden Signale werden verwendet
    - $x(t)$ ist bandbegrenztes Signal mit Grenzfrequenz $f_{\text{Grenz}}$
    - $p(t)$ ist Abtastsignal mit einer Abtastfrequenz von $f_{\text{A}}$ und damit Intervallen zwischen den Abtastzeitpunkten $T_{\text{s}} = 1 / f_{\text{A}}$
    - $x_{\text{s}}(t) = x(t) \cdot p(t)$ ist das abgetastete Signal
  - Damit gilt
    - $x(t)$ kann aus $x_{\text{s}}(t)$ exakt wiedergewonnen werden, wenn $f_{\text{A}} > 2 f_{\text{Grenz}}$
    - In diesem Fall entsteht keine Beeinträchtigung aufeinanderfolgender Symbole (Intersymbol Interference, ISI)

# Beweis Abtasttheorem

- Das Signal p(t) ist ein periodisches Signal und kann damit folgendermaßen dargestellt werden
  - $p(t) = \sum_{n=-\infty}^{\infty} P_n e^{j2\pi n f_s t} \longrightarrow x_s(t) = x(t)p(t) = \sum_{n=-\infty}^{\infty} P_n x(t) e^{j2\pi n f_s t}$



[Stal06]

  - Fouriertransformierte von $x_s(t)$

$$X_s(f) = \int_{t=-\infty}^{\infty} x_s(t) e^{-j2\pi f t} dt$$

    - Einsetzen von $x_s(t)$

$$X_s(f) = \int_{-\infty}^{\infty} \sum_{n=-\infty}^{\infty} P_n x(t) e^{j2\pi n f_s t} e^{-j2\pi f t} dt$$

$$X_s(f) = \sum_{n=-\infty}^{\infty} P_n \int_{-\infty}^{\infty} x(t) e^{-j2\pi (f - n f_s) t} dt$$

  - Aus der Definition der Fouriertransformierten
    - Wobei $X(f)$ die Fouriertransformierte von $x(t)$ ist

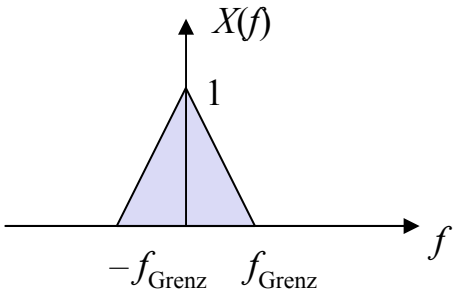
$$X(f - n f_s) = \int_{-\infty}^{\infty} x(t) e^{-j2\pi (f - n f_s) t} dt$$

  - Durch Einsetzen in die obige Gleichung ergibt sich

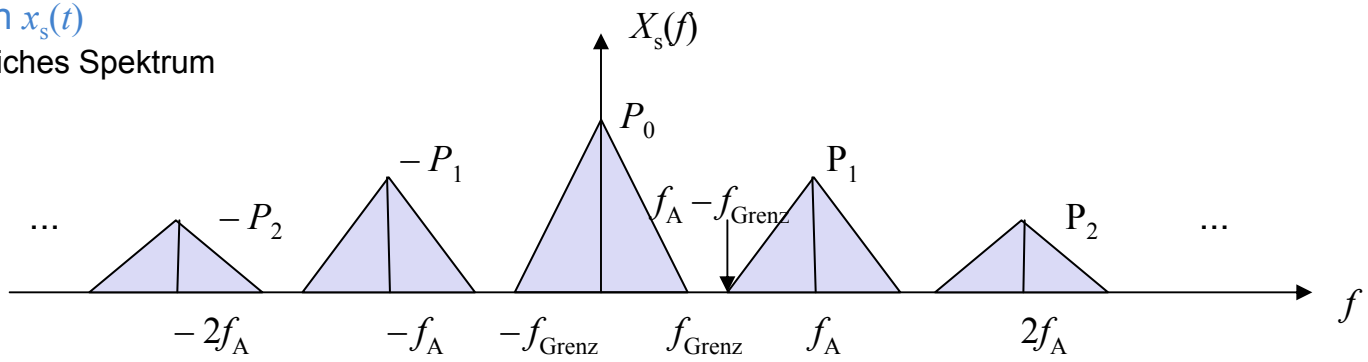
$$X_s(t) = \sum_{n=-\infty}^{\infty} P_n X(f - n f_s)$$

# Beweis Abtasttheorem

- Interpretation der letzten Gleichung
  - Annahme: Bandbreite von $x(t)$ ist im Bereich von 0 bis $f_{\text{Grenz}}$
  - Spektrum von $x(t)$
    - Bandbegrenztes Spektrum




  - Das Spektrum von $x_s(t)$ setzt sich aus zwei Komponenten zusammen: Zum einen dem Spektrum von $x(t)$; zum Anderen dem zu den entsprechenden Harmonischen der Abtastfrequenz übersetzten Spektrum von $x(t)$.
    Dabei wird das jeweilige Spektrum mit dem entsprechenden Koeffizienten der Fourier-Reihe von $p(t)$ multipliziert.
    - Spektrum von $x(t)$ erscheint in $X_s(f)$
  - Spektrum von $x_s(t)$
    - Unendliches Spektrum



# Beweis Abtasttheorem

- Für $f_A > 2 f_{Grenz}$ ergibt sich bei den Harmonischen keine Überlappung

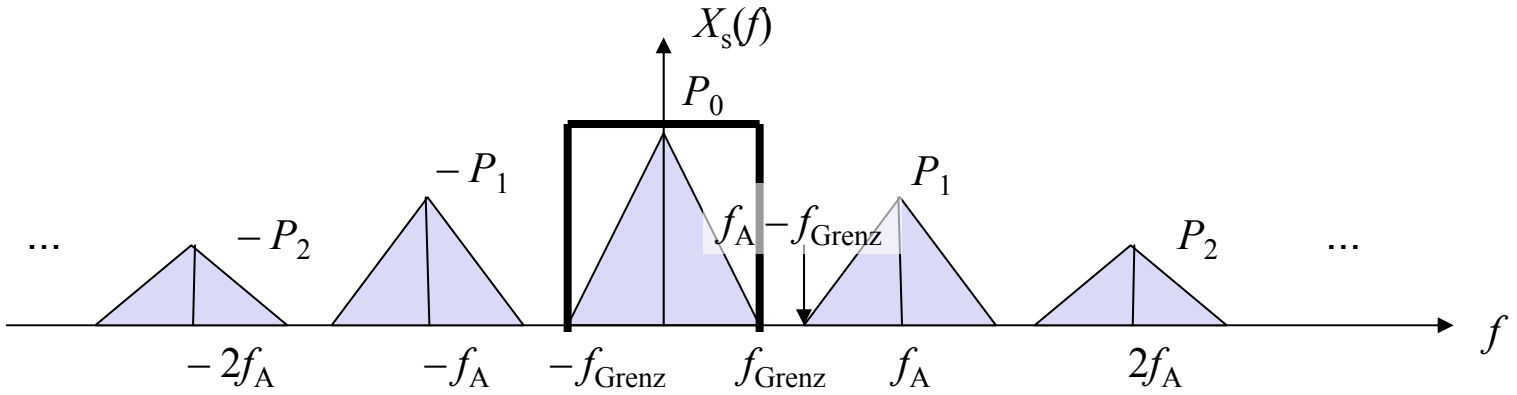
$$\frac{-f_A}{2} < f < \frac{f_A}{2}$$

- $X_s(f) = P_0 X(f)$

- Das Spektrum von $x(t)$ kann z.B. durch die Verwendung eines Tiefpassfilters

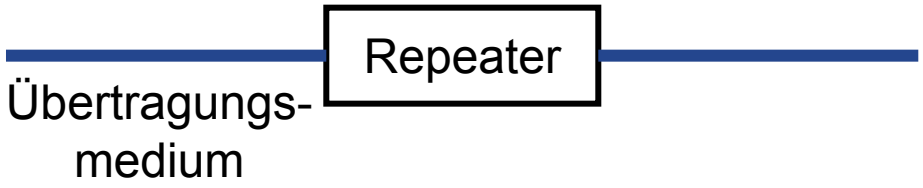
$$rect(f) \text{ mit } 2 f_{Grenz} < f < 2(f_A - f_{Grenz})$$

wieder gewonnen werden



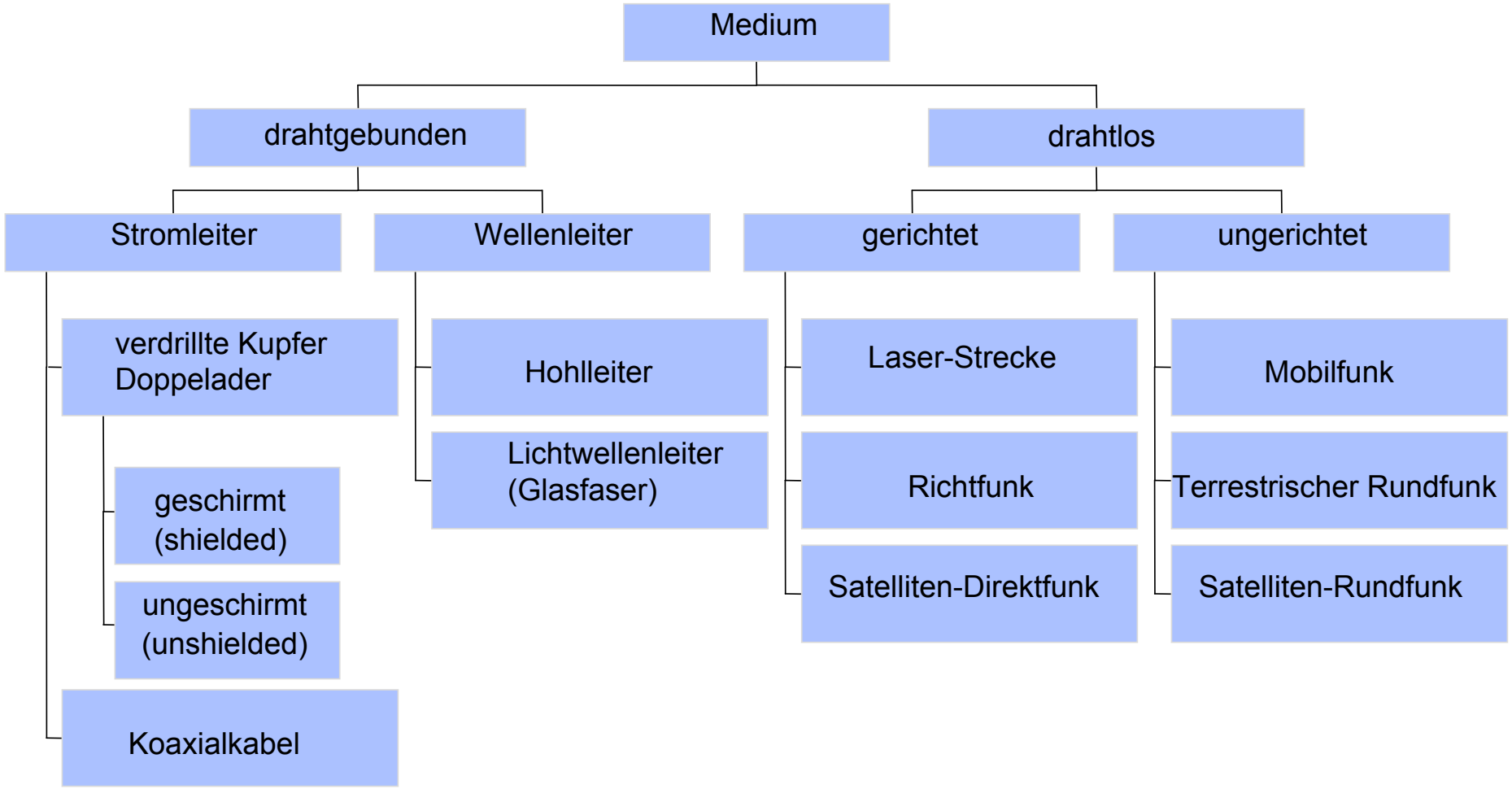
# 2.4 Übertragungsmedien

- Aufgabe
  - Physikalische Verbindung benachbarter Kommunikationssysteme
- Randbedingungen
  - Überbrückbare Distanz
  - „Verlängerung“ durch Einsatz sogenannter Repeater
    - Arbeiten als Signalverstärker

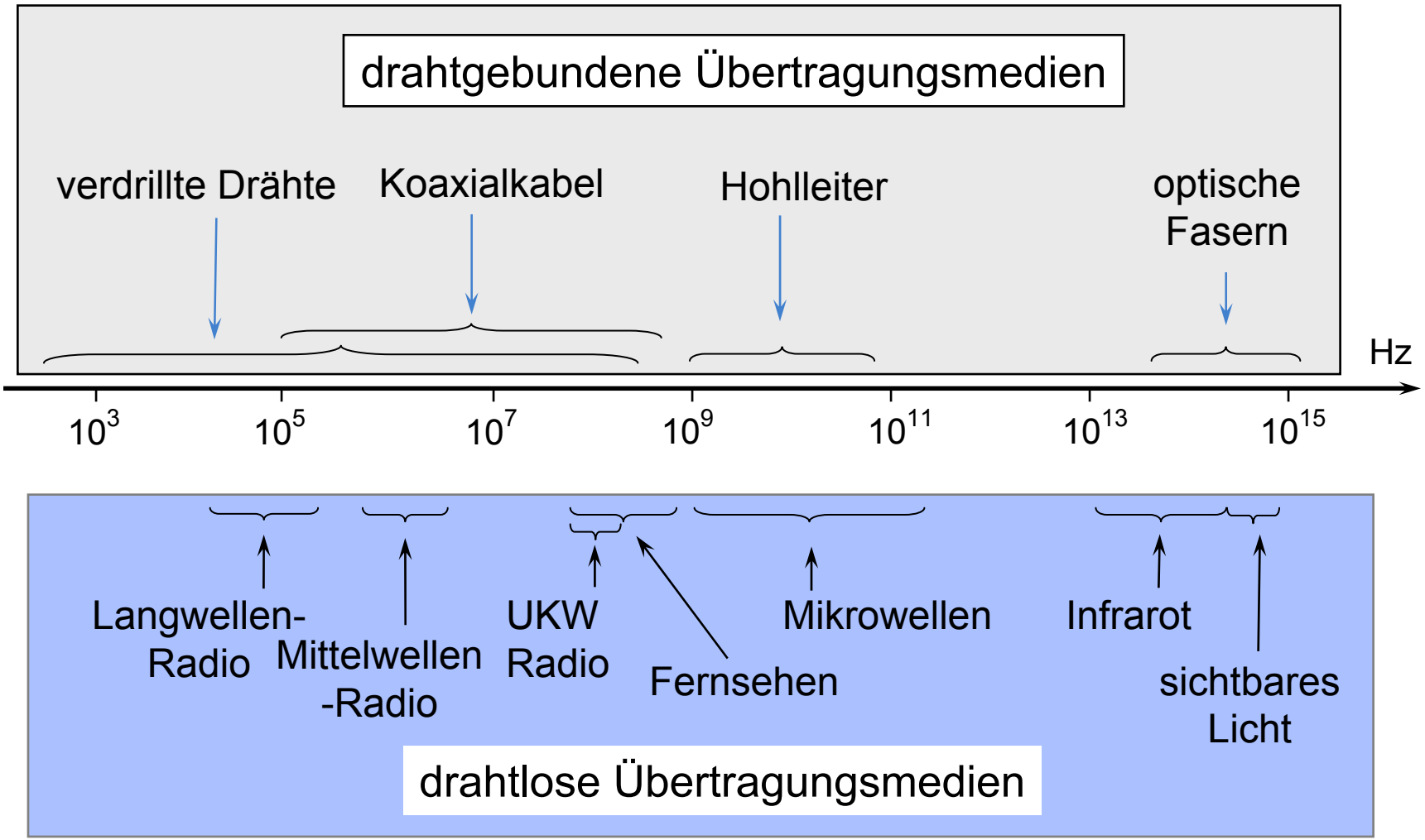



- Netze
  - Setzen verschiedene Übertragungsmedien ein, z.B.
    - Lichtwellenleiter intern im Netz
    - Drahtloser Anschluss ans Netz (z.B. Handy) bzw. im Netz

# Klassifikation



# Elektromagnetisches Spektrum

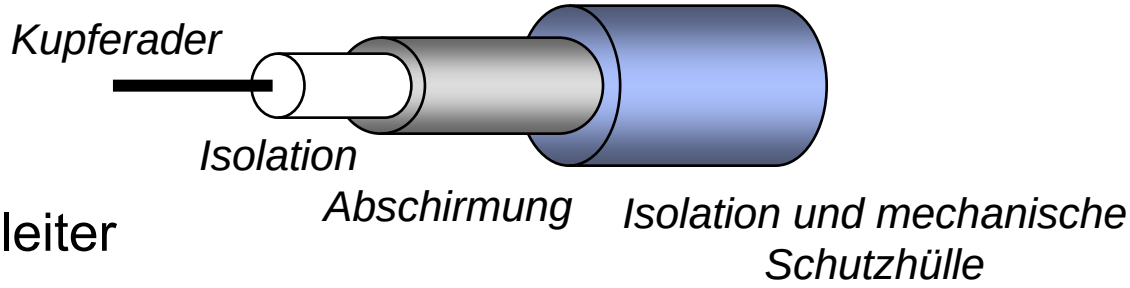


# Drahtgebundene Medien

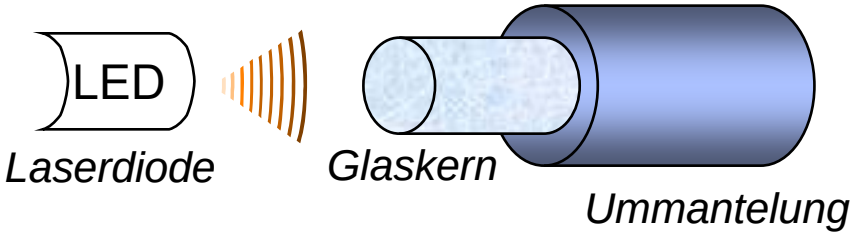
- Verdrilltes Adernpaar




- Koaxialkabel




- Lichtwellenleiter (Glasfaser)



# Verdrilltes Adernpaar

- Übertragungseigenschaften
  - Kategorie 3 UTP (Unshielded Twisted Pair)
    - Kategorie 3 Kabel sind in öffentlichen (und privaten) Gebäuden meist schon vorhanden ➔ Telefonverkabelung.
    - Datenraten bis 16 Mbit/s.
    - Dämpfung [dB pro 100m]: 1 MHz, 2,6 / 4 MHz, 5,6 / 16 MHz, 13,1
  - Kategorie 5 UTP
    - Kategorie 5 wird als neue Standardverkabelung in öffentlichen Gebäuden verwendet
    - Datenraten bis 100 Mbit/s
    - Dämpfung [dB pro 100m]: 1 MHz, 2,0 / 4 MHz, 4,1 / 16 MHz, 8,2 / 25 MHz, 10,4 / 100 MHz, 22,0
  - Unterschiede Kategorie 3 und 5 UTP
    - Die Anzahl der Drehungen sind bei Kategorie 5 deutlich höher (eine Drehung pro 0,6 – 0,85cm), als bei Kategorie 3 (eine Drehung pro 7,5 – 10cm)
    - Kategorie 5 hat eine geringere Dämpfung als Kategorie 3
  - Kategorie 6 UTP
    - Typische Leistung: 1 – 300 MHz
    - Datenraten bis 2,4 Gbit/s möglich
    - Dämpfung [db pro 100m]: 1 MHz, 2,0 / 4 MHz, 3,8 / 16 MHz, 7,6 / 100 MHz, 19,8 / 200 MHz, 29,0 / 300 MHz, 36,4
  - Kategorie 7 (Shielded Screen Twisted Pair)
    - Vier einzeln abgeschirmte Adernpaare führt zu hervorragendem Dämpfung-Übersprech-Verhältnis
    - Datenraten bis 10 Gbit/s möglich
    - Betriebsfrequenzen bis 600 MHz

# Exkurs: Dezibel – Dämpfung, Verstärkung

- Häufig auftauchende Größe zur Bewertung der Intensität ist das Dezibel (dB)
  - Gibt  dimensionsloses Verhältnis an, z.B. zweier Pegel, wie etwa Spannung, Leistung, Strom, Lautstärke, etc.
  - keine physikalische Einheit (eher eine „Hilfsgröße“)
  - Herkunft: Einheit Bel (B), benannt nach Alexander Graham Bell, 1 dB = 1/10 B
  - Findet insbesondere Anwendung beim Ausdruck von Gewinn (Verstärkung) oder Verlust (Dämpfung) eines Signals
    - z.B. durch eine Bearbeitung, eine Übertragung, eine Leitung, Störungen, beim Durchlaufen einer Schaltung, etc.

# Exkurs: Dezibel – Dämpfung, Verstärkung

- Zwei Bezugsarten möglich
  - relative Pegel: Bezug auf Vergleichswert
  - absolute Pegel: Bezug auf Referenzwert, z.B. dBV mit Bezug auf Spannung von 1V
- Vorteile der Größe dB: logarithmische Darstellung ⇨ einfache Berechnungen mit Pegeln (meist Summen statt Multiplikationen)

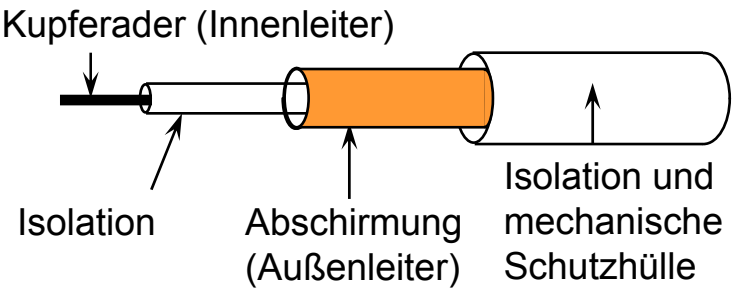
- Bei Pegelberechnungen wird unterschieden
  - quadratische Größen: Energie, Leistung, Intensität
  - lineare Größen: Schalldruck/Lautstärke, Spannung, Stromstärke, dann gilt für den relativen Pegel $L_U$ zweier Spannungen $U_1$ und $U_2$:

$$L_U = 20 * \lg \frac{U_1}{U_2} \text{ dB}$$

- Übertragungstechnik arbeitet meist mit (linearen) Amplituden, es gilt dann beispielsweise
  - +6dB entspricht einer Verdopplung, -6dB einer Halbierung der Ausgangs- im Vergleich zur Eingangsspannung
  - +20dB entspricht einer Verzehnfachung

# Koaxialkabel

- Aufbau
  - Dicke Kupferader, innere Isolierung, Metallgeflecht, äußere Isolierung
  - Außenleiter umschließt Innenleiter zylindrisch
    - Dazwischen befindet sich ein Dielektrikum aus Kunststoffen oder Gasen
    - Signalausbreitung erfolgt im Dielektrikum

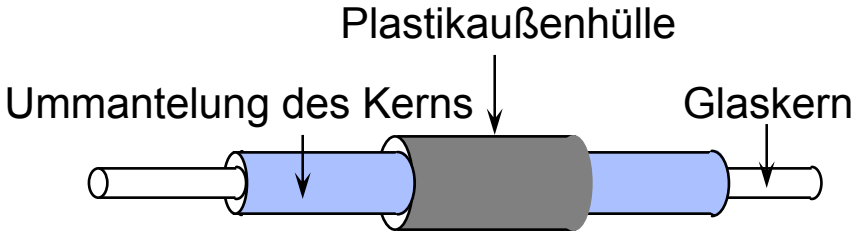



- Eigenschaften
  - Rauschen wird gut von der inneren Leitung fern gehalten
  - Frequenz bis zu 500 MHz, auf kurzen Distanzen auch 750 MHz und mehr (Kabelfernsehnetze)
  - Bandbreite: bis 1 GHz (für kurze Strecken)
  - bis zu 500 Mbit/s; Repeaterabstand ca. 1-10 km
  - Fehlerwahrscheinlichkeit ca. $10^{-7}$
- Beispiele
  - Telefonnetz (netzintern), Kabelfernsehen, früher: lokale Netze

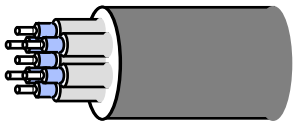
# Lichtwellenleiter

- Aufbau
  - Faser
    - Innen: dünnes Glas oder Plastik (2 – 100 µm)
      - Dünne Faser: etwa halbe Größe eines menschlichen Haars
  - Kabel
    - Besteht in der Regel aus mehreren Fasern (z.B. bis ca. 2000)
- Übertragungsprinzip: Totalreflexion des Lichts

[Haaß97]




Single core



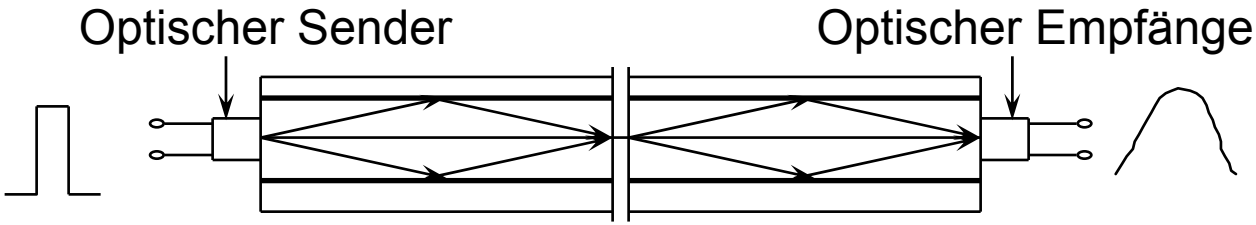
Multicore

- Eigenschaften
  - Wellenlängen (850 nm, 1300 nm, 1550 nm, 1610 nm)
  - Bandbreiten im Bereich mehrerer THz verfügbar
  - Repeaterabstand ca. 10 – 100 km

Video: Herstellung eines Lichtwellenleiters

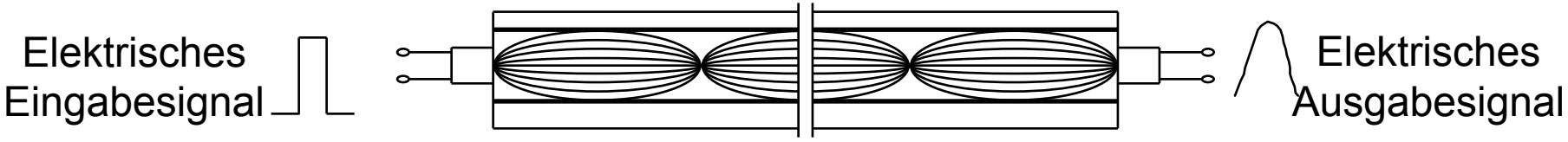
# Typen von Lichtwellenleitern

## Multimode-Faser mit Stufenindex



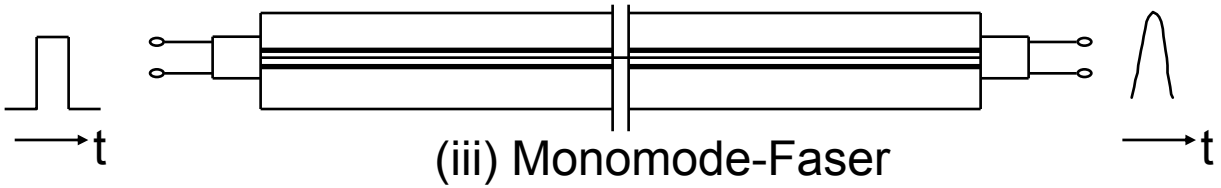

(i) Multimode-Faser mit Stufenindex

## Multimode-Faser mit Gradientenindex




(ii) Multimode-Faser mit Gradientenindex

## Monomode-Faser



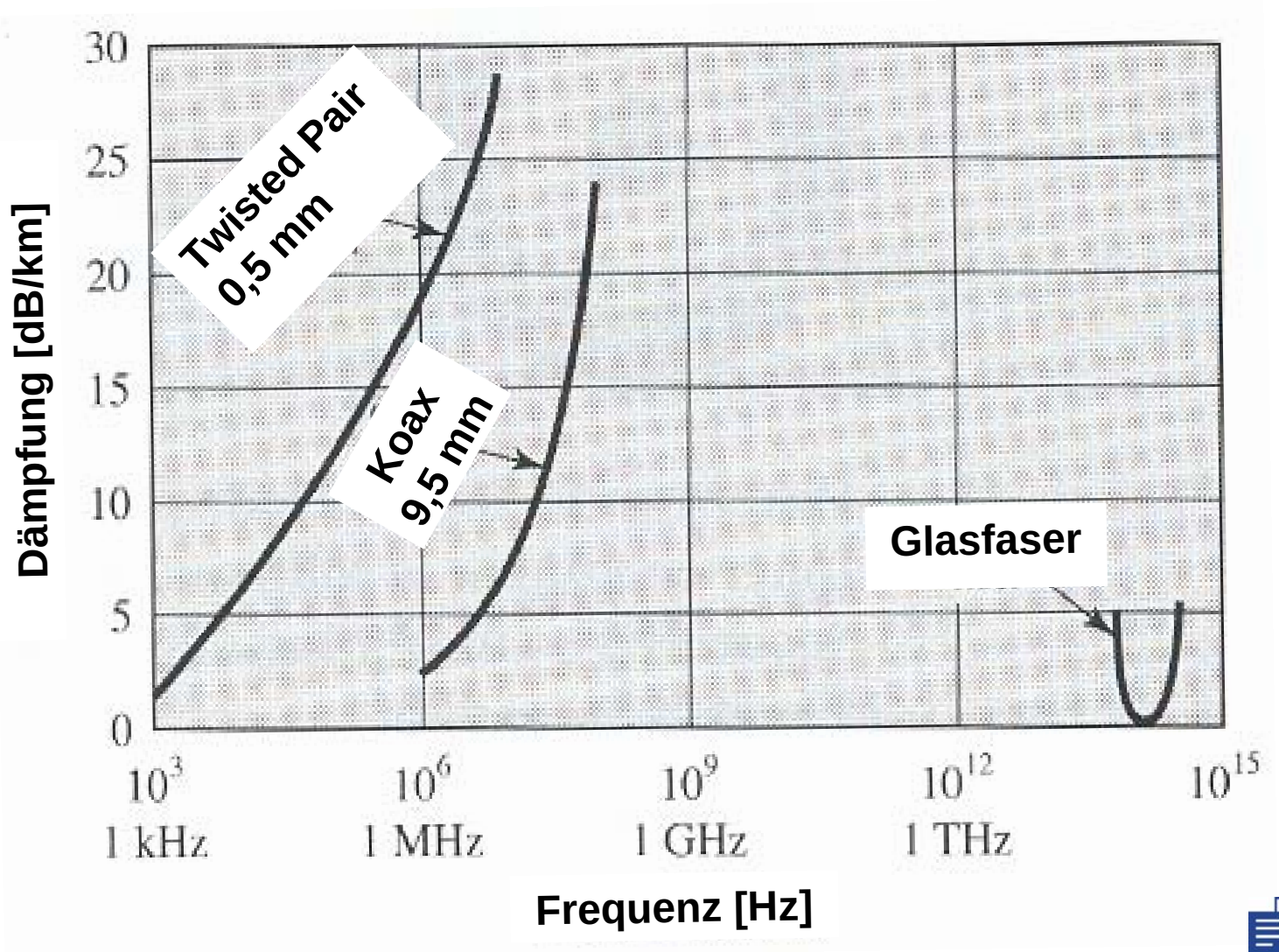

(iii) Monomode-Faser

# Vergleich der Übertragungsmedien

| | Frequenzbereich | Typische Dämpfung | Typische Verzögerung | Repeater Abstand |
|---|---|---|---|---|
| **Twisted Pair** | 0 – 3,5 kHz | 0,2 dB/km (bei 1kHz) | 50 µs/km | 2 km |
| **Twisted Pair (mehrere Paare pro Kabel)** | 0 – 1 MHz | 0,7 dB/km (bei 1kHz) | 5 µs/km | 2 km |
| **Koaxial Kabel** | 0 – 500 MHz | 7 dB/km (bei 10 MHz) | 4 µs/km | 1 – 9 km |
| **Glasfaser** | 186 – 370 THz | 0,2 – 0,5 dB/km | 5 µs/km | 40 km |

[Stal06]

# Vergleich der Übertragungsmedien bzgl. Dämpfung



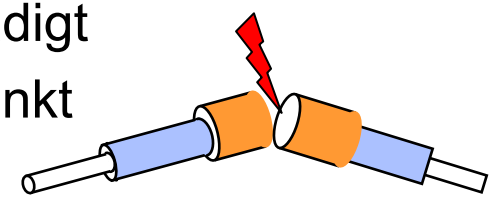

[Stal06]

# Robustheit drahtgebundener Medien

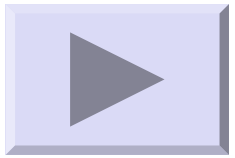
- Bisher näher betrachtet
  - Verdrilltes Adernpaar, Koaxialkabel, Lichtwellenleiter
    - Verbinden benachbarte Kommunikationssysteme
    - Können große Distanzen überbrücken (Repeater)
- Noch nicht betrachtet
  - Zuverlässigkeit und Robustheit solcher Verbindungen
    - Auswirkungen eines Kabelbruchs auf das gesamte Netz
      - Wie kann es zu einem Kabelbruch kommen?
    - Routen zwischen zwei Endsystemen nicht mehr gültig
      - Automatische Adaption des Netzes möglich?
      - Adäquate Routingverfahren nötig
        (siehe Kapitel 8 „Netzkopplung und Vermittlung“)

# Dez. 2006: Kabelbruch in Asien

- Kabelbruch 15 km vor der Küste Taiwans nach einem Seebeben der Stärke 6,7
  - Mindestens 6 Kabel von China Telecom beschädigt
  - Internetverbindung nach Asien stark eingeschränkt
    - Notkapazitäten der Provider zugeschaltet
    - Neue Routen sollen beschädigten Abschnitt meiden
  - Reparatur des Unterseekabels wird 2-3 Wochen dauern
  - Einige Zahlen
    - Ca. 100 Millionen Internetnutzer sind betroffen
    - Antwortzeiten zum Teil verdreifacht, falls Verbindung überhaupt hergestellt werden kann
    - Internet-Telefonate zwischen
      - Taiwan und USA gingen auf ca. 40% zurück
      - China/Japan und USA gingen auf ca. 10% zurück
    - Internetverkehr in Vietnam auf ca. 30% zurückgegangen



Video: Verlegung des ersten Tiefseekabels



# TAT-14

- Trans Atlantic Telephonecable Number 14“
  - Verbindet seit 2001 Europa mit Nordamerika
  - Gesamtkosten: 1,2 Milliarden Euro
  - 15000 Kilometer, ca. 5 cm dick
  - Befindet sich 1 Meter tief im Meeresboden
  - Verstärker alle 50-70 Kilometer erforderlich
  - Doppelte Kabeltrassen




https://www.tat-14.com/tat14/stations.jsp

# Aktuelles: Glasfaser durch Bodensee

- Am 19.November 2007 wurde ein neues Glasfaser-Kabel im Bodensee verlegt
  - Verbesserung des Datentransfers zwischen Nord- und Südufer
    - Zwischen Konstanz und Friedrichshafen
  - Länge: 26 km
  - Liegt in Tiefen bis 200m abseits der Ankerlinien
  - 192 einzelne Glasfasern im ca. 2,5 cm dicken Kabel






- Historisches
  - 1856: Erste Telegrafenverbindung zwischen Deutschland und der Schweiz durch Seekabel im Bodensee
  - 1892: Neues Kabel kann Telegramme und Telefon-gespräche übertragen
  - 1906: Erstes „Pupin-Kabel“ der Welt zur Verbesserung der Sprachqualität
  - 1955-1987: Weitere Kupfer- und erstes Glasfaserkabel für Telefonverkehr

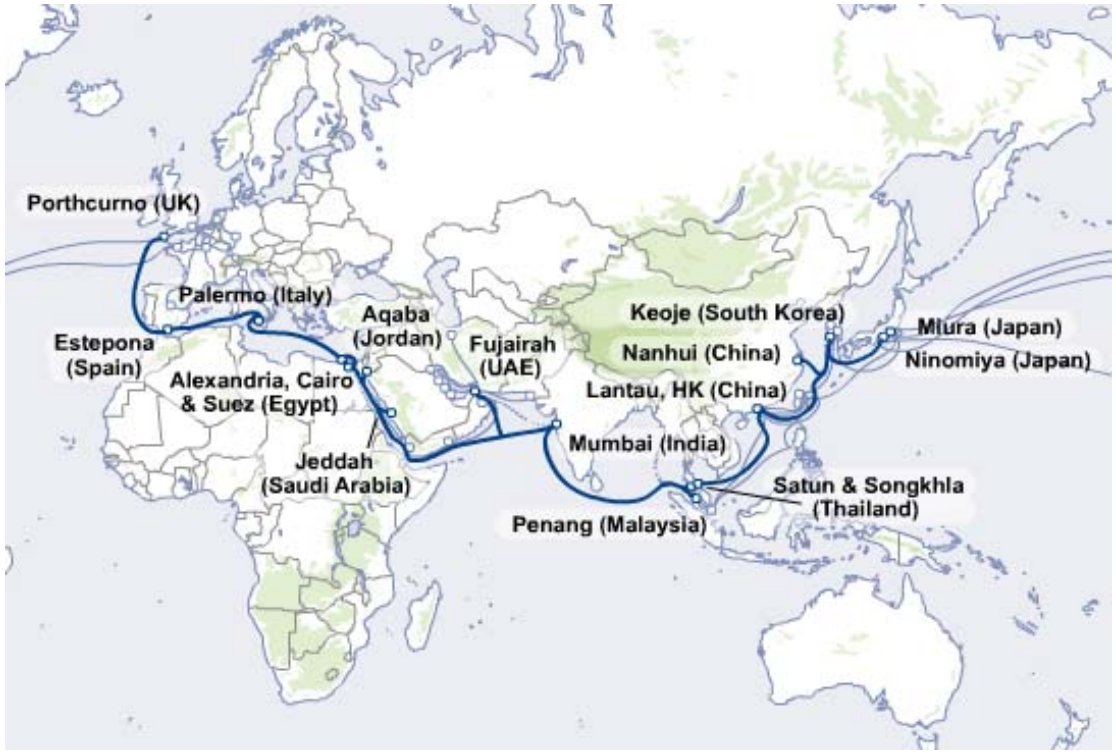


[Hei08a]

# Aktuelles: Seekabel

- 30.01.2008: Zwei Glasfaser-Backbones im Mittelmeer beschädigt
  - Ägypten nur noch mit 30 Prozent Bandbreite angeschlossen
  - Indien nur noch mit 40 Prozent Bandbreite
  - Flag Telecom Group Betreiberin des FEA-Kabels (Flag Europe Asia)
    - Reicht von Großbritannien bis Japan (rund 27.000 km)
    - Beide Kabel Kapazität von fast 620 Gigabit pro Sekunde

[Hei08b]




[Flag08]

# Aktuelles: Seekabel

- 01.02.2008: Riss des FALCON-Kabels um Arabische Halbinsel 56 km vor Dubai
  - Verbindungen nach Ägypten, Vereinigte Arabische Emirate (VAE), Katar, Kuweit, Bahrain, Iran, Irak und Oman gestört
  - Verantwortlich war ankerndes Schiff
    - Bergung eines fünf Tonnen schweren Ankers

 [Hei08c,Hei08e]

- 01.02.2008: Ausfall Glasfaser-Backbone zwischen Katar und VAE
  - Probleme mit Stromversorgung

 [Hei08d]

- 19.12.2008: Ausfall der Seekabel SEA-ME-WE 4, SEA-ME-WE 3 und FLAG-FEA im Mittelmeer
  - Vermutlich durch ankerndes Schiff zwischen Sizilien und Ägypten
  - Umrouten des meisten Verkehrs zw. Europa und Asien über USA

 [Hei09]

- Transsibirische Überlandleitung zwischen Europa und Asien in Planung
  - Soll Datenleitungen per Seekabel oder über Nordamerika verkürzen
    - Laufzeitverkürzung Dateneinheit von >300 ms auf etwa 160 ms

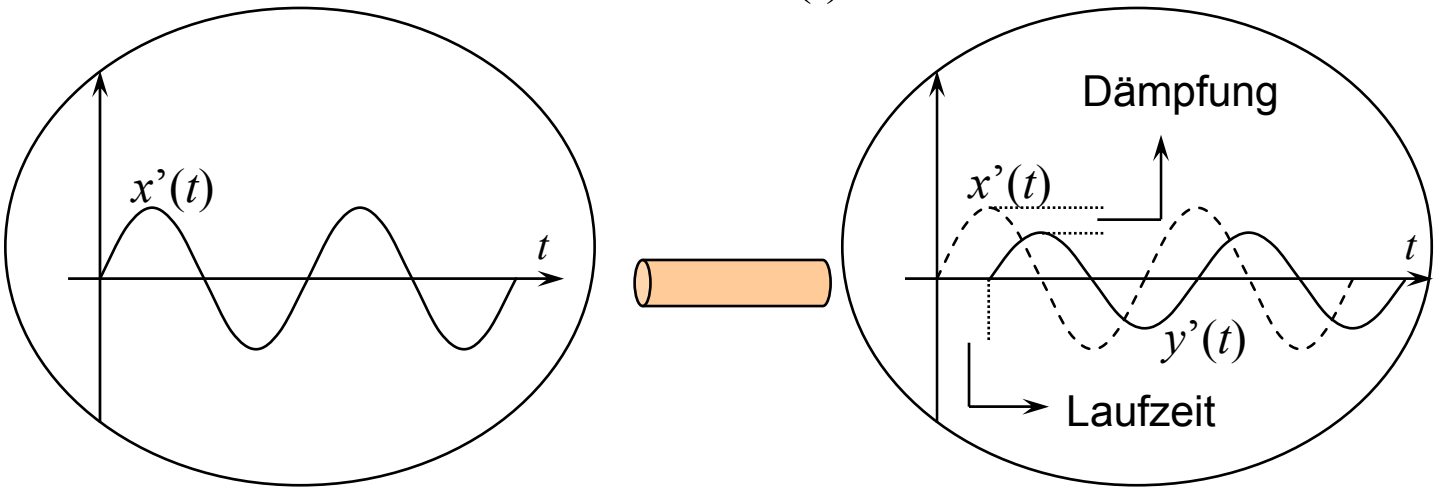
 [Hei08f]

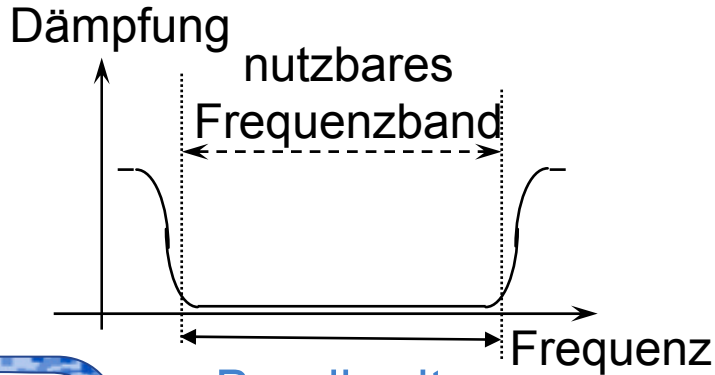
- Aber: Auch Fehler bei der Übertragung möglich

# Signalübertragung über ein Medium

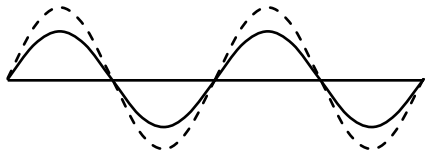
- $y'(t) = F(x'(t), z'(t))$

- $x'(t)$: Eingabesignal
- $y'(t)$: Ausgabesignal
- $z'(t)$: Störeinfluß

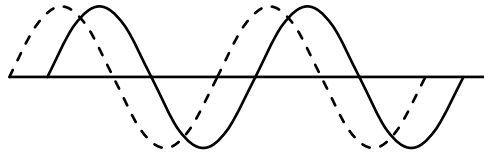



Bandbreite



Dämpfungsverzerrung



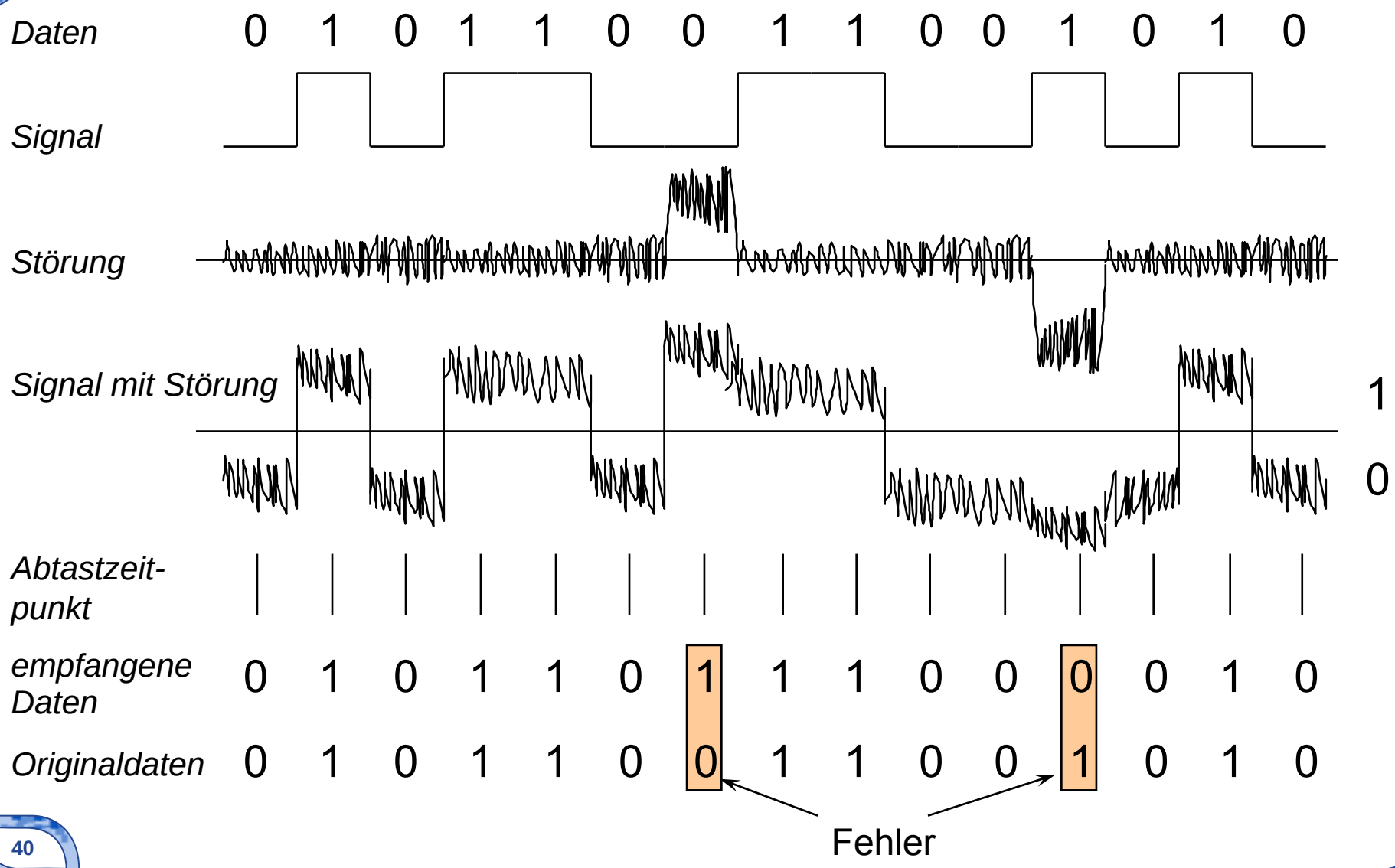
(Lauf-)Zeitverzerrung

# Übertragungsstörung durch Rauschen

- Neben der systematischen Beeinflussung des Signals durch
  - Dämpfung
  - Laufzeitverzerrung
- können Signalstörungen durch
  - transiente, stochastische  Prozesse
  - weißes Rauschen
  - Echobildung (durch zeitverschobenes Eingabesignal)
  - Nebensprechen (gegenseitige Medienbeeinflussung)
  - Brummsignale (niederfrequente Störsignale)
  - Störimpulse (kurzzeitig mit hoher Amplitude)
- auftreten

- Lange anhaltende Störungen: Bündelfehler
  - Echobildung, Nebensprechen, (thermisches) Rauschen, Anschalten von induktiven Lasten (Motor), 50Hz, Netzbrummen stets auf einer Leitung, ...

# Beispiel: Auswirkung von Störungen



# 2.5 Digitale Signalübertragung

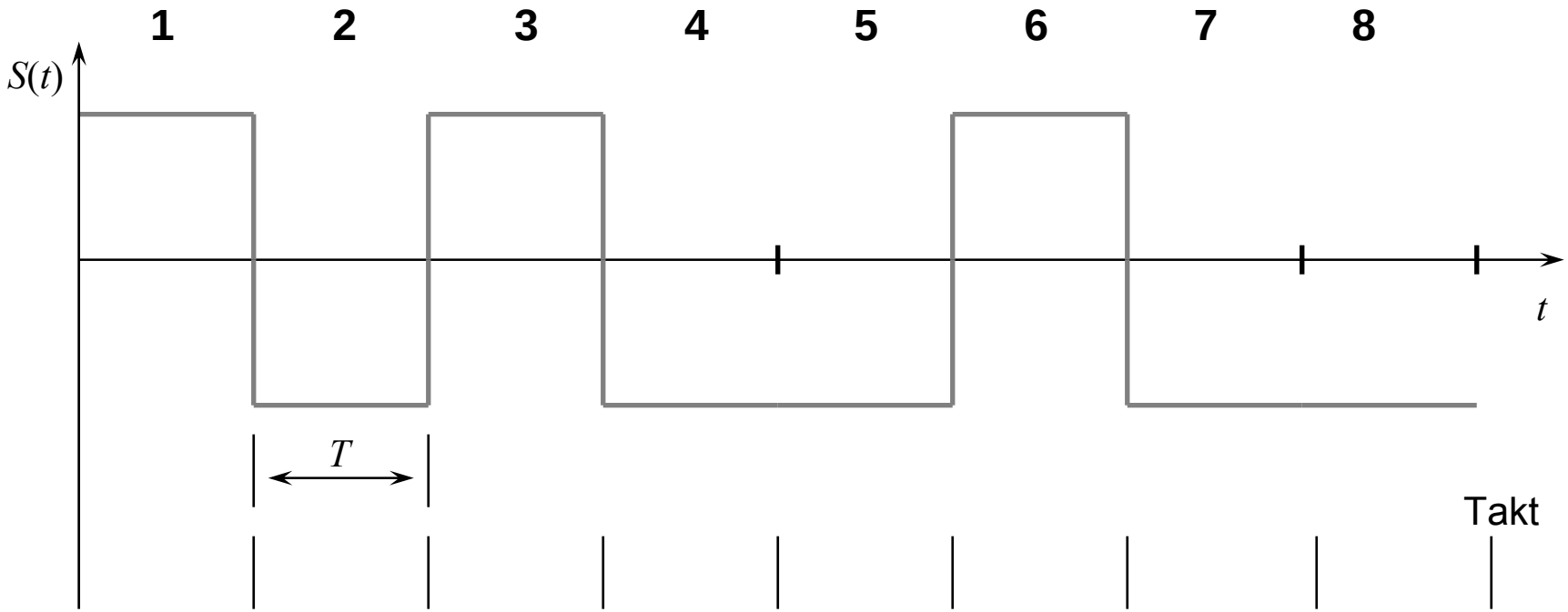
- Schritt
  - Charakteristisch für zeitdiskrete Signale ist die Existenz eines minimalen Zeitintervalls $T_{\mathrm{Min}}$ zwischen aufeinanderfolgenden – möglichen – Änderungen der Signalkoordinate (Schrittdauer, kurz: Schritt als Signal definierter Dauer)
  - Wichtig: Digitales Signal mit fester Schrittdauer $T$ (Schritt-Takt)
- Isochrones (isochronous) Digitalsignal
  - Ein Digitalsignal ist isochron, wenn seine Kennzeitpunkte, d.h. die Zeitpunkte des Übergangs von einem Signalelement zum nächsten, in einem festen Zeitraster liegen
- Anisochrones (anisochronous) Digitalsignal
  - Ein nicht-isochrones Digitalsignal

# Übertragungs- vs. Schrittgeschwindigkeit

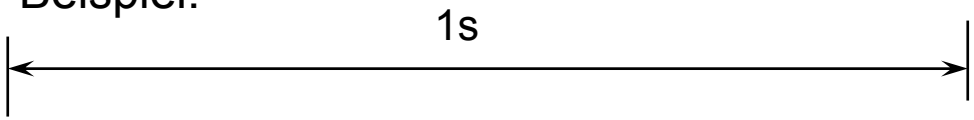
- Schrittgeschwindigkeit (Baudrate)
  - Zahl der Signalparameter-Zustandswechsel
  - Einheit: baud (1/s) (nach Jean Maurice Baudot)
  - bei isochronen Digitalsignalen
    - Kehrwert der Schrittdauer $1/T$
  - auch als *Baudrate* bezeichnet
- Übertragungsgeschwindigkeit (Bitrate, Datenrate)
  - Anzahl der übertragbaren Bitstellen pro Zeiteinheit
  - Einheit: bit/s
- Schrittgeschwindigkeit = Übertragungsgeschwindigkeit
  - Nur für binäre Signale, bei denen jeder Schritt als Signalelement genau ein Bit als Codeelement darstellt.

# Schrittgeschwindigkeit – Beispiel

Schrittfolge:

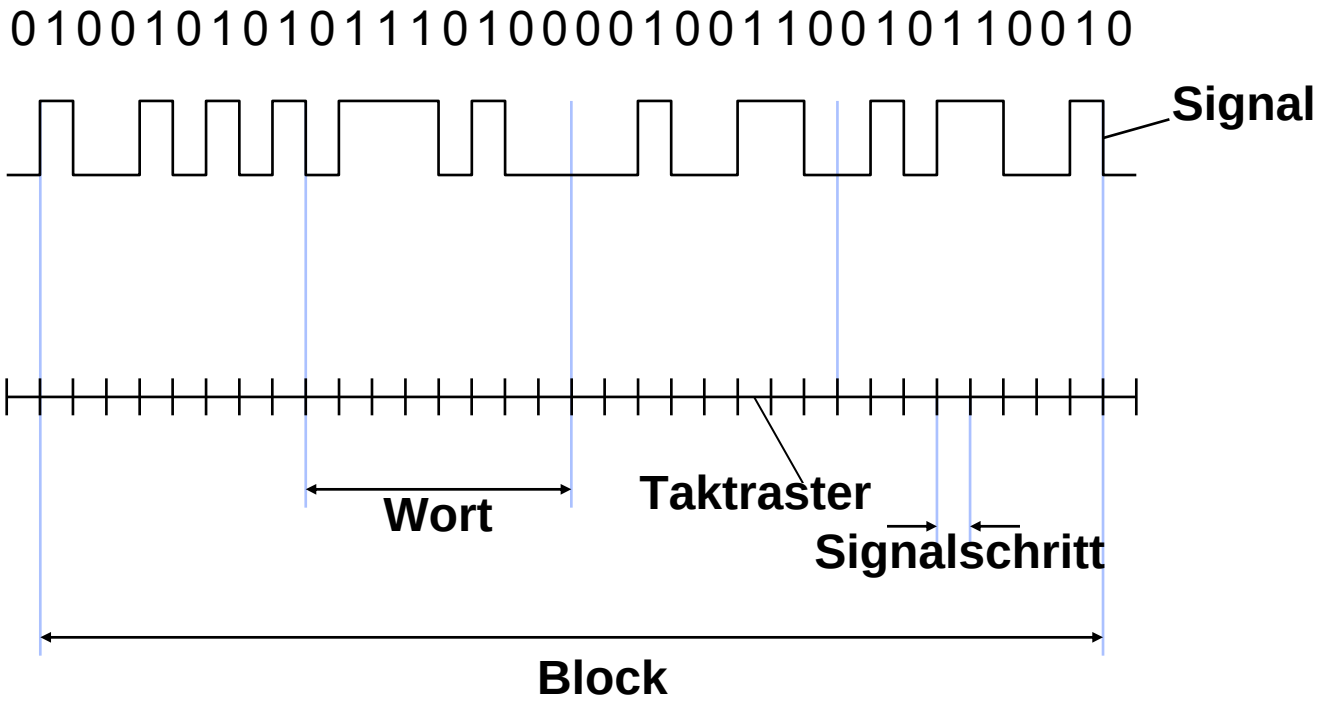



Beispiel:



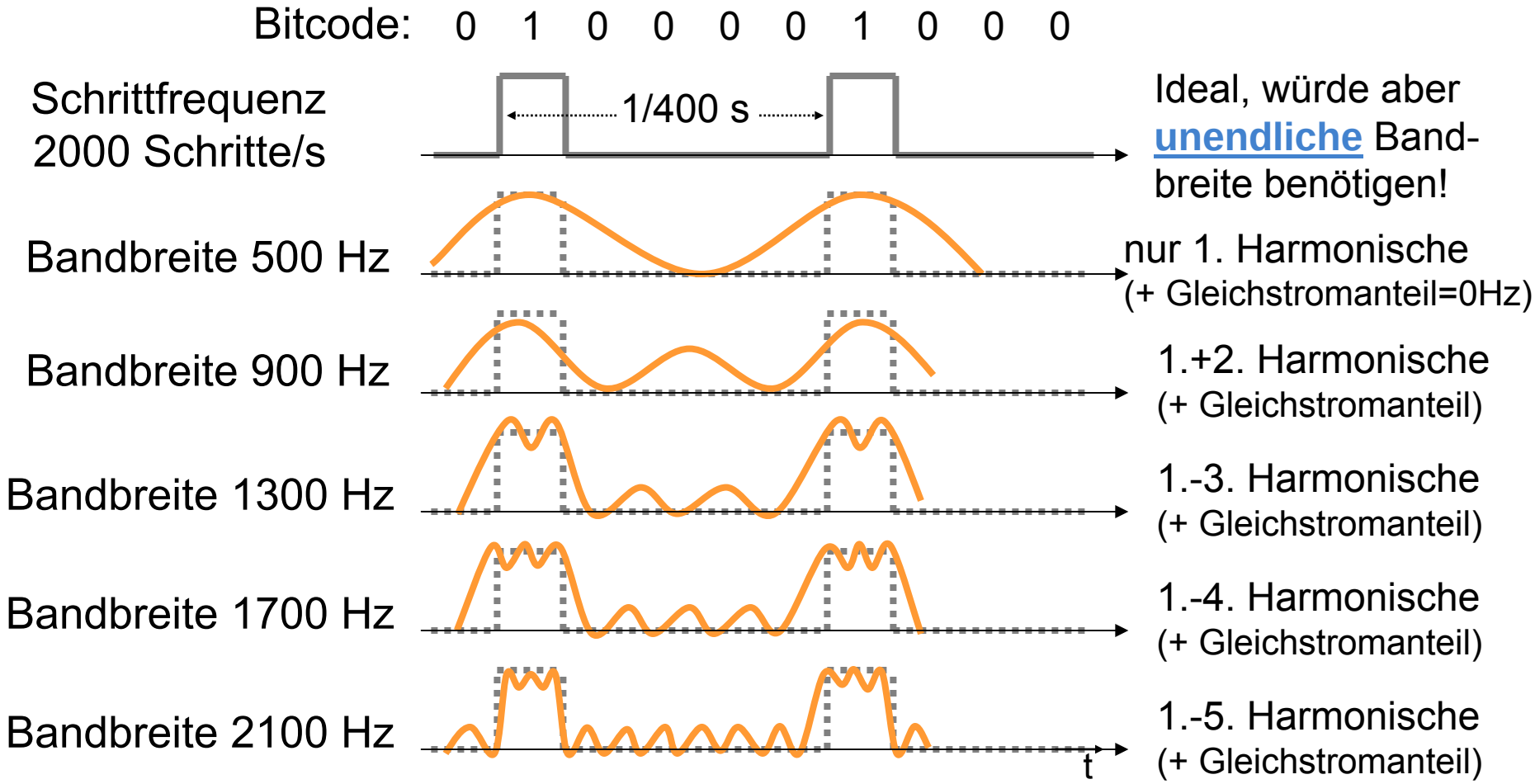

⇨ Schrittgeschwindigkeit 5 baud

# Digitales Signal: Begriffe




- Wort
  - Gruppe kleiner fester Anzahl von Schritten (5 bis 8)
    - bei binärer Übertragung Oktett = 8 Schritte
- Block
  - logisch zusammenhängende größere Einheit

# Bandbreite und digitales Signal




- Min. Bandbreite für Übertragung einer bel. Bitfolge mit bestimmter Schrittfrequenz nötig
- genaue Berechnung der min. Bandbreite nach den Formeln von Shannon/Nyquist

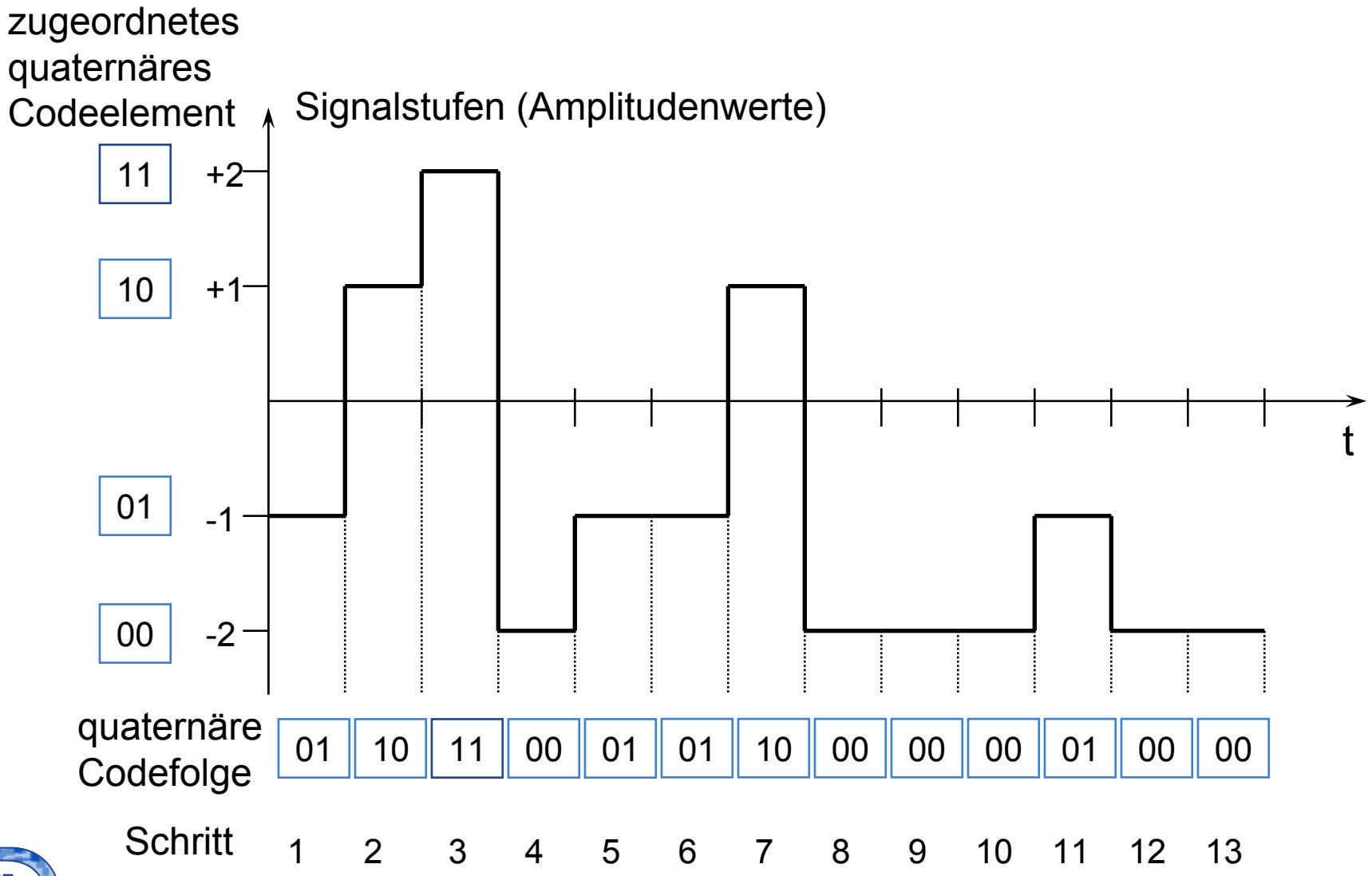
# Zwei- und mehrwertige Digitalsignale

- Zweiwertiges Digitalsignal (Binärsignal)
  - Digitales Signal mit nur zwei Werten des Signalparameters (Digitales Signal, bei dem die Signalelemente binär sind)

- Mehrwertiges (mehrstufiges) Digitalsignal
  - Die (diskrete) Signalkoordinate kann mehr als zwei Werte annehmen
    - Beispiel: DIBIT = zwei Bit pro Koordinatenwert (quaternäres Signalelement)
  - Die Anzahl $n$ der diskreten Werte (Kennwerte, Stufen), die ein Signalelement annehmen kann, wird wie folgt gekennzeichnet:

    $n = 2$ binär (binary)
    $n = 3$ ternär (ternary)
    $n = 4$ quaternär (quarternary)
    ...
    $n = 8$ oktonär (octonary)
    $n = 10$ denär (denary)

# Mehrwertiges Digitalsignal – Beispiel



# 2.6 Kanalkapazität

- Problem
  - Was für eine Datenrate kann auf einer Datenleitung (Kanal) unter den existierenden Bedingungen erzielt werden?
- Datenrate
  - Größe zur Bestimmung der Kapazität eines Übertragungssystems
  - Meist gemessen in übertragenen Bit pro Sekunde (bit/s)
- Fehlerfreier Kanal
  - Datenrate ist nur durch die Bandbreite limitiert
  - Nach Nyquist gilt
    - Für eine Schrittrate von $2B$ ist ein Signal der Frequenz $B$ ausreichend
      - Max. Schrittgeschwindigkeit [baud] = 2 * Bandbreite [Hz]
    - Umkehrung gilt auch
    - Bei binären Signalen: Datenrate = 2B bit/s
    - $C = 2B \operatorname{ld} M$
      - $M$: Anzahl der Signalwerte
      - Erhöhung von $M$ führt zu Erhöhung der erzielbaren Datenrate

# 2.7 Übertragungsdauer

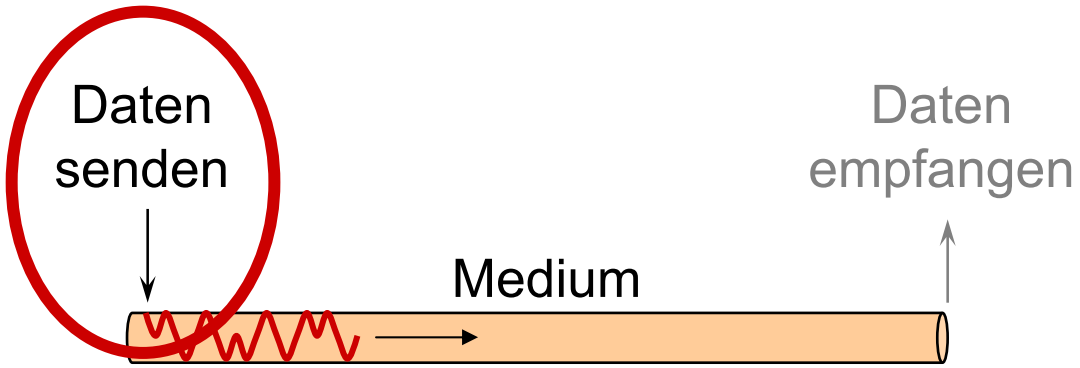
- Allgemeines Verständnis von „Übertragungsdauer“
  - Wie lange dauert es, Daten vom Sender zum Empfänger zu transportieren?
- Achtung: Setzt sich aus verschiedenen Komponenten zusammen
  - Diese hängen jeweils von unterschiedlichen Eigenschaften ab
- Wichtigste Komponenten
  - Sendezeit $T_S$
    Dauer, um Daten „auf das Medium zu legen“
  - Ausbreitungsverzögerung $T_A$
    Dauer, bis Daten über das Medium beim Empfänger eintreffen
    - oft auch als Verzögerung oder Laufzeit bezeichnet

# Sendezeit

- Sendezeit $T_S$: Zeit zwischen Beginn und Abschluss der Sendung
  - Nur auf Senderseite betrachtet
  - Abhängig von
    - Datenmenge $L$
    - Datenrate $d$ des Mediums (Leitung/Verbindung)
      - Anzahl der pro Zeiteinheit übertragenen Daten (z.B. bit/s)

$$T_S = \frac{L}{d}$$

- Achtung: Nach Abschluss der Sendung sind die Daten noch nicht beim Empfänger!
  ⇨ Ausbreitungsverzögerung

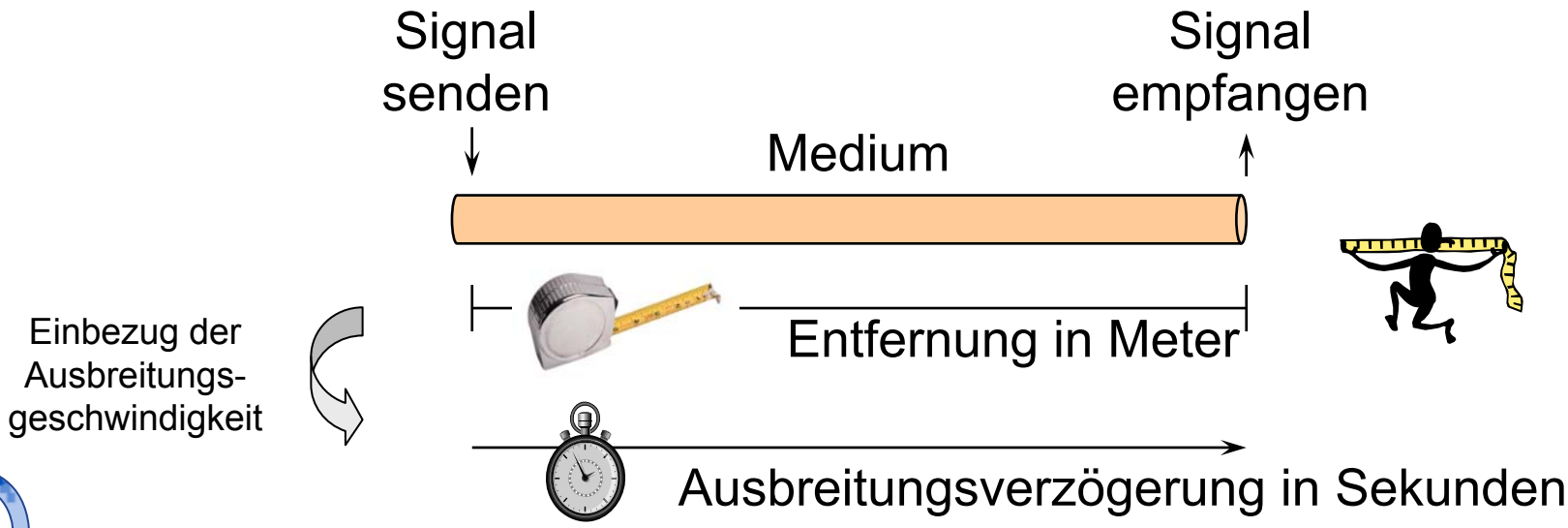


# Ausbreitungsverzögerung

- Ausbreitungsverzögerung $T_A$: Zeitspanne zwischen Absenden eines Signals und dessen Eintreffen am anderen Ende des Mediums

$$T_A = \frac{m}{v}$$

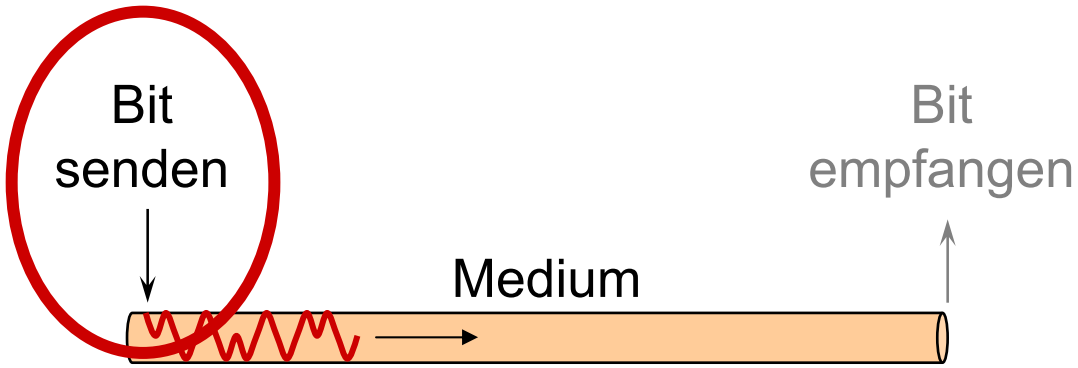
  - Abhängig von
    - Ausbreitungsgeschwindigkeit $v$
      - physikalische Signalgeschwindigkeit
        - in üblichen Medien (Kabel, Glasfaser) etwa 2/3 der Lichtgeschwindigkeit
    - Länge des Mediums $m$



# „Welche Länge hat ein Bit auf der Leitung?“

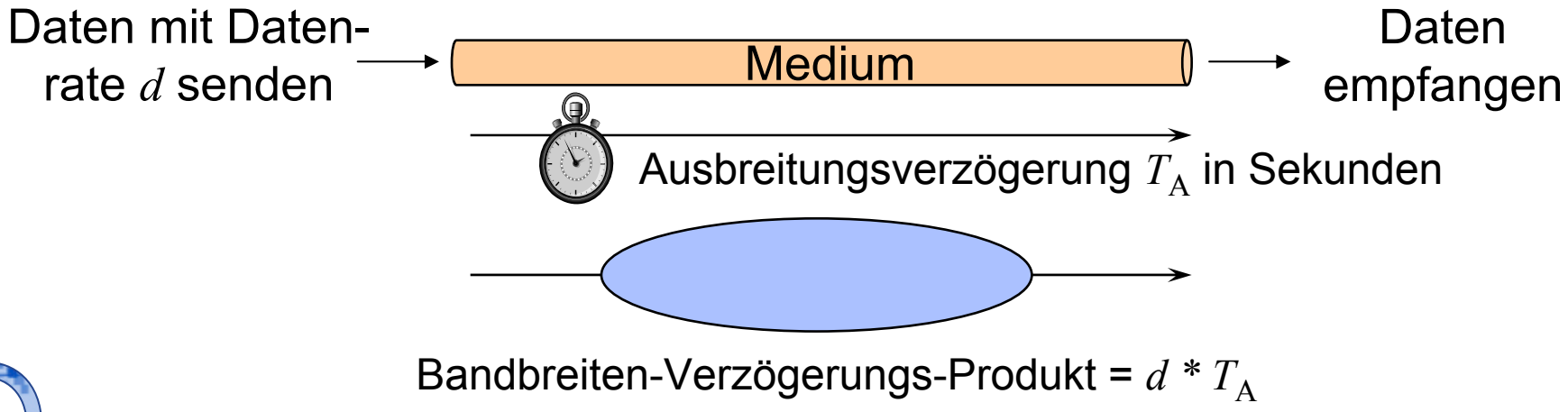
- Interessante Fragestellung nach physikalischer Länge (in Metern) eines Bits auf der Leitung
  - Abhängig von Datenrate $d$ und Ausbreitungsgeschwindigkeit $v$
    - „Wie weit sind die ersten Signalschwingungen des Bits schon gewandert, bevor das Bit vollständig auf das Medium gelegt wurde?“

$$\text{phys. Länge 1Bit} = \frac{1\text{Bit}}{d} * v$$



# Bandbreiten-Verzögerungs-Produkt

- Welche Datenmenge kann sich gleichzeitig „auf dem Medium“ befinden?
  - Übertragungsmedium als Datenspeicher!
  - Abhängig von Datenrate $d$ und Ausbreitungsverzögerung $T_A$
    - (letztere wiederum abhängig von Länge des Mediums m und Ausbreitungsgeschwindigkeit $v$)
    - „Wie lange braucht das erste Bit, bevor es am anderen Ende der Leitung ankommt und wie viel Daten können bis dahin noch zusätzlich auf die Leitung gegeben werden?“
  - Auch als Bandbreiten-Verzögerungs-Produkt bezeichnet
    - Datenrate wird häufig auch (ungenau) als Bandbreite bezeichnet



# Datenraten

- Typischerweise gemessen in

| Einheit | | Bezeichnung |
|---|---|---|
| bit/s | | *„bit pro Sekunde“ / „bit per second“* |
| kbit/s | 1000 bit/s | *„Kilo bit p.s.“* |
| Mbit/s | $10^6$ bit/s | *„Mega bit p.s.“* |
| Gbit/s | $10^9$ bit/s | *„Giga bit p.s.“* |
| Tbit/s | $10^{12}$ bit/s | *„Tera bit p.s.“* |
| Pbit/s | $10^{15}$ bit/s | *„Peta bit p.s.“* |

  - Achtung: Stets Multiplikator (1000 vs. 1024) beachten
  - In der Übertragungstechnik ist 1 Kbit/s meist 1000 bit/s
  - Siehe auch  [Kros91]

# Datenraten in LAN, MAN & WAN

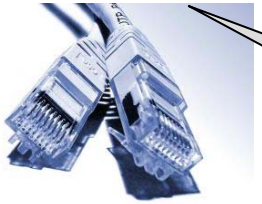
- um 1960: Erste Modems, wenige hundert bit/s
  - Nur von Telcos zugelassene Geräte erlaubt.
- 1968: Akustikkoppler, 300 bit/s
  - Alle erlaubt, da keine direkte Verbindung mit Leitung.
- 1981: Smartmodem (von Hayes Communications), 300 bit/s
- 1996: Modems mit 56 kbit/s (K56flex/X2, ab 1998 V.90)
- 1989: Deutschland-Einführung ISDN, 64 kbit/s B-Kanal
- 1995: LAN: Fast Ethernet, 100 Mbit/s
- 1998: LAN: Gigabit Ethernet, 1 Gbit/s
- 1999: Deutschland-Start von DSL, 768 kbit/s
- 2002: LAN: Erste 10 Gigabit Ethernet Typen, 10 Gbit/s
- 2006: Asymmetric DSL2+, bis zu 16 Mbit/s
- 2006: 6,4 TBit/s über 1000 km Glasfaser
  - DWDM mit 160 verschiedenen Wellenlängen
- ~ 2010 (geschätzt): LAN: 100 Gigabit Ethernet, 100 Gbit/s



Download 100 MByte: 1 Monat

sowieso schwierig: ~ 10 MByte Festplatten





Download 100 MByte: < 4 h





Download 100 MByte: <1 min

Download 100 MByte: 1/8 ms

# Beispiel: Bandbreiten-Verzögerungs-Produkt

- Mobilfunk
  - AMPS $d$ = 13 kbit/s $m$ = 3 km $T_A$ = 10µs → 0,13 bit
  - PCS, GSM $d$ = 300 kbit/s $m$ = 3 km $T_A$ = 10µs → 3 bit
  - UMTS $d$ = 14,4 Mbit/s $m$ = 3 km $T_A$ = 10µs → 144 bit
- Wireless Local Area Networks (WLAN)
  - Infrarot $d$ = 4 Mbit/sm = 10 m $T_A$ = 33ns → 0,002 bit
  - Bluetooth $d$ = 2,1 Mbit/s $m$ = 100 m $T_A$ = 333ns → 0,01 bit
  - 802.11g $d$ = 54 Mbit/s $m$ = 150 m $T_A$ = 500ns → 0,4 bit
  - 802.11n $d$ = 248 Mbit/s $m$ = 250 m $T_A$ = 833ns → 3,2 bit
  - 802.16 (WiMax) $d$ = 134 MBit/s $m$ = 75 km $T_A$ = 250µs → 511 bit
- Satelliten und Interplanetar
  - Geostationär (GEO) $d$ = 50 Mbit/s $m$ = 35 863 km $T_A$ = 270 ms → 200 kbit
  - Mars $d$ = 1 Mbit/s $m$ = 45 Mio. km bis 400 Mio. km
    - $T_A$ = 2,5 min bis 22 min → 3 Mbit bis 22 Mbit
  - Jupiter $d$ = 1 Mbit/s $m$ = 590 Mio. km bis 970 Mio. km
    - $T_A$ = 33 min bis 54 min → 32 Mbit bis 53 Mbit
  - Pluto $d$ = 1 Mbit/s $m$ = 4.275 Mio. km bis 7.525 Mio km
    - $T_A$ = 237 min bis 418 min → 237 Mbit bis 418 Mbit

→ Unterschiedliche Rahmenbedingungen erfordern ganz unterschiedliche Protokolle

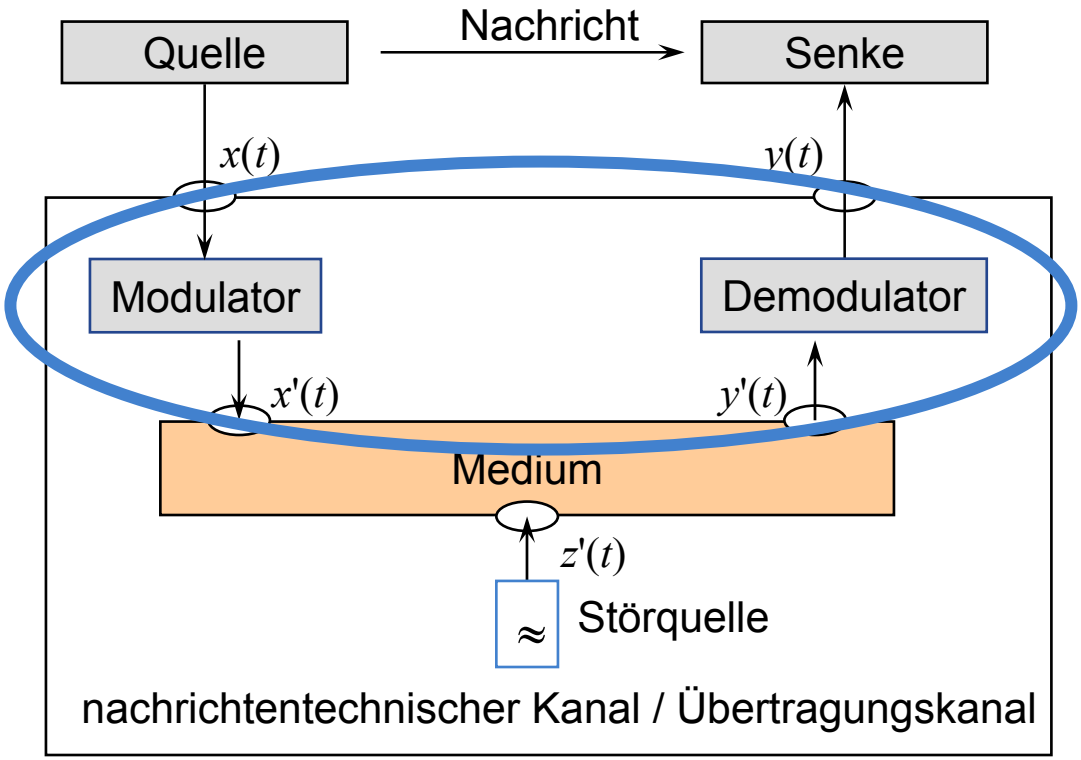
# Übertragungsdauer (2)

- Je nach Betrachtungsweise spielen noch weitere Komponenten bei der Übertragungszeit eine Rolle
  - Zeit für Protokollbearbeitung
    - Daten codieren, mit Prüfsummen versehen, etc.
  - Dauer der Bearbeitung in Zwischensystemen
    - insbesondere bei Paket-basierter Übertragung
      - Daten auspacken, prüfen, nächstes Ziel suchen (Wegewahl) anpassen, neue Prüfsummen berechnen, etc.
  - Wartezeit in Warteschlangen
  - etc.
- Stets beachten, was genau im Kontext gesucht/betrachtet wird

# Kanalkapazität bei Störungen

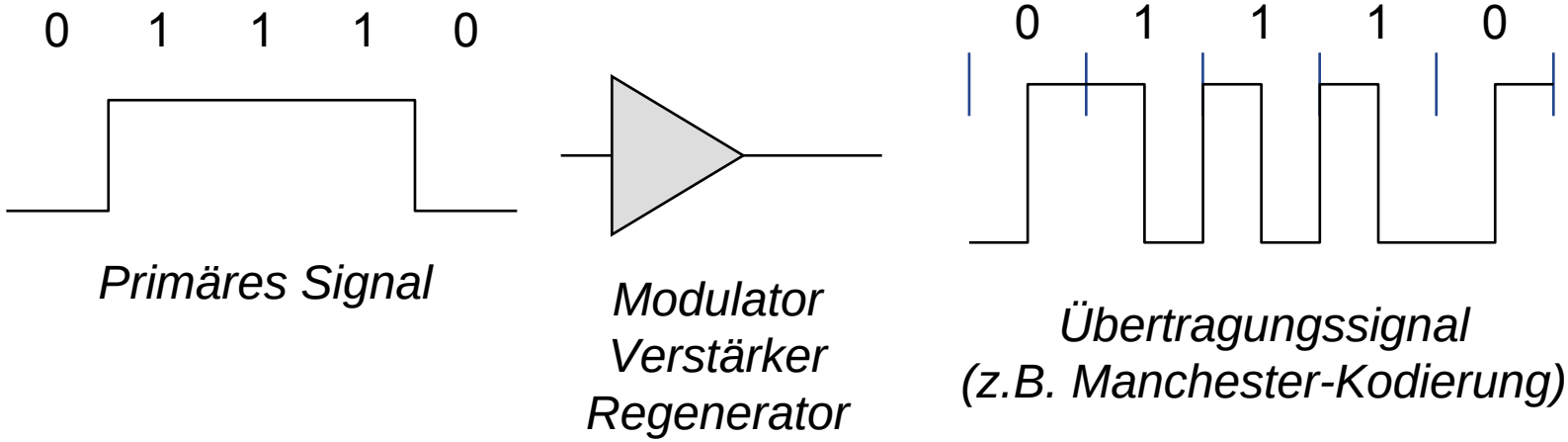
- Beobachtung
  - Bei höherer Datenrate werden Bits „kürzer“ und damit verfälschen Störungen mehr Bits als bei niedriger Datenrate
  - Wichtig ist hier der Signal-Rausch-Abstand (*Signal-Noise-Ratio*, SNR)
    - SNR [dB] = 10 $\log_{10}$ (Signalenergie / Rauschenergie)
    - Energie eines Signals ist proportional zum Quadrat der Amplitude
- Shannon-Hartley-Gesetz
  - Gibt eine obere Grenze für die auf einer Datenleitung erzielbare Datenrate in Abhängigkeit des Signal-Rausch-Abstandes
  - $C = B \operatorname{ld}(1 + S/R)$
    - $C$ ist die Datenrate in bit/s
    - $B$ ist die Bandbreite (gemessen in Hz) des Kanals
    - $S$ ist Energie des Signals
    - $R$ ist Energie der Störquelle

# 2.8 Modulation / Demodulation



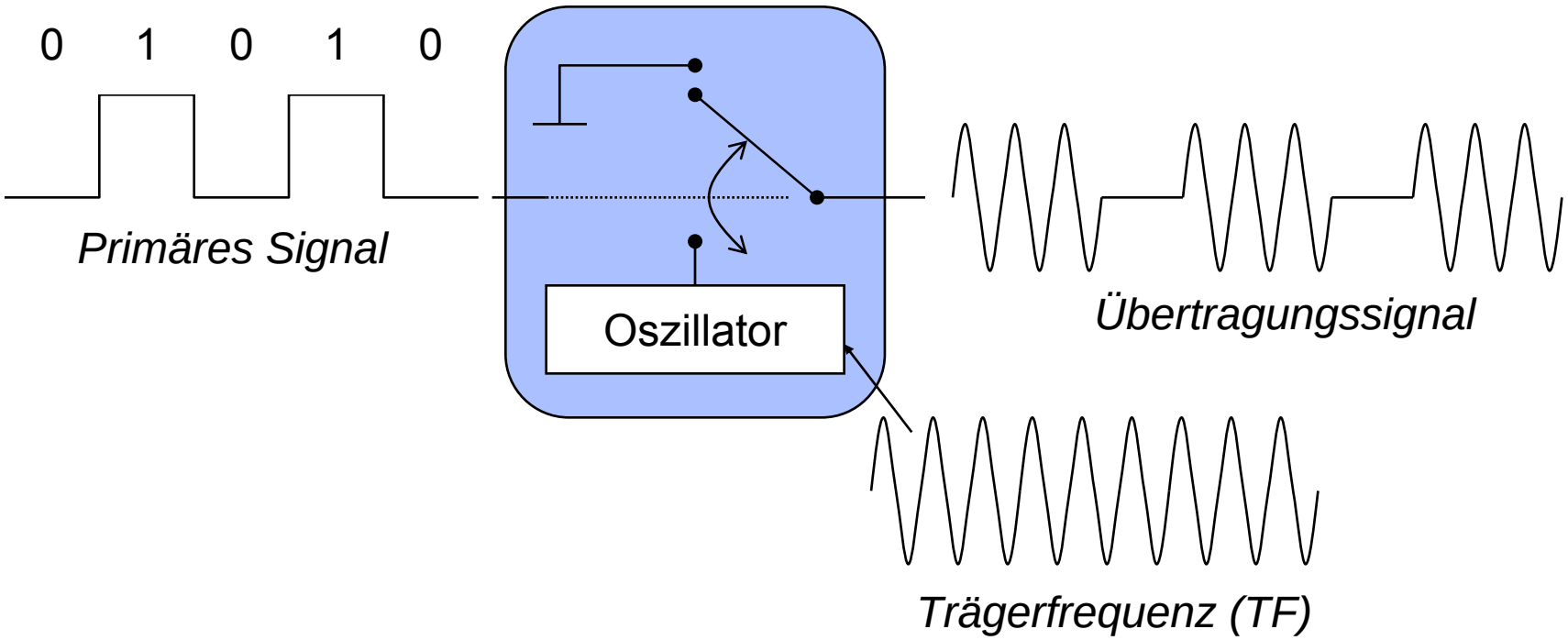

- Zur Übertragung erforderlich
  - Umformung des primären Quellsignals in das Eingabesignal des Mediums
    ⇨ Modulation
  - Rückformung des Ausgabesignals in das primäre Senkensignal
    ⇨ Demodulation
- Bei digitalen Kanälen wird im wesentlichen unterschieden zwischen
  - „direkter“ Weitergabe des Quellsignals: Umformung digital → digital
  - Aufprägung des Quellsignals auf harmonische Trägerschwingung d.h. Quellensignale werden einer Trägerfrequenz aufmoduliert: Umformung digital → analog

# Basisbandübertragung



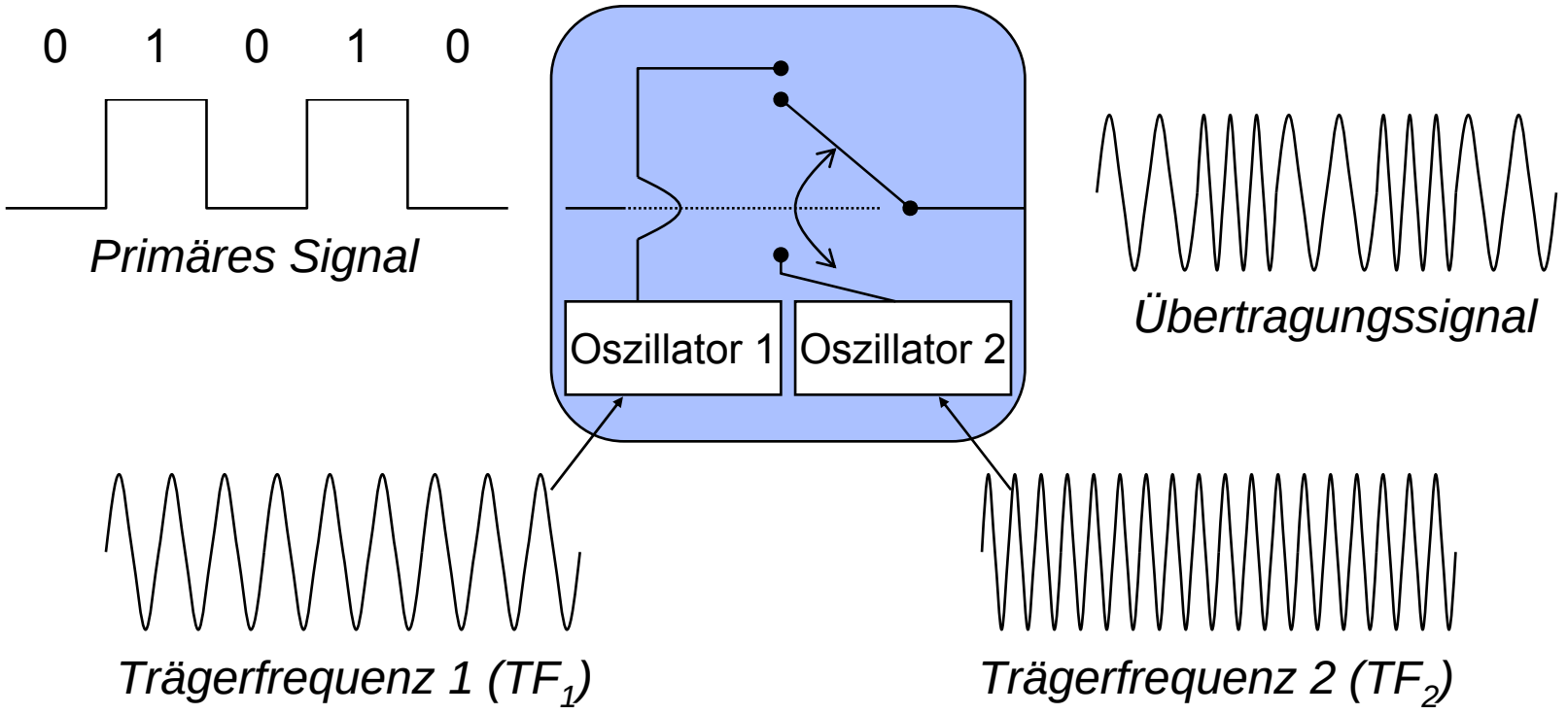

- Einfaches Verfahren
- Primäres Signal muss an Mediencharakteristik angepasst werden
- Umformung digital → digital

# Amplitudenmodulation



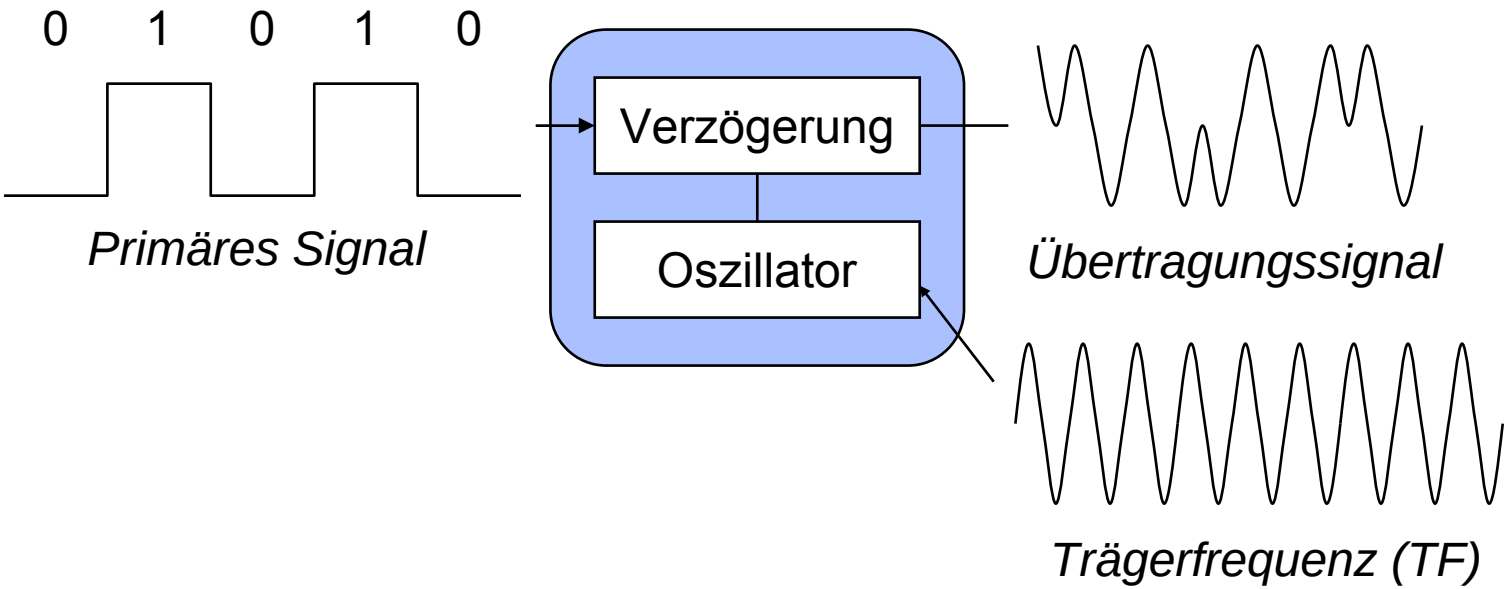

- Primäres Signal wird durch Amplitudenveränderung auf Trägersignal moduliert
- Amplitudenmodulation ist sehr störanfällig

# Frequenzmodulation




- Primäres Signal wird durch gezielte Änderung der Trägerfrequenz moduliert
- Frequenzmodulation ist das unter anderem auch bei UKW-Rundfunk eingesetzte Modulationsverfahren

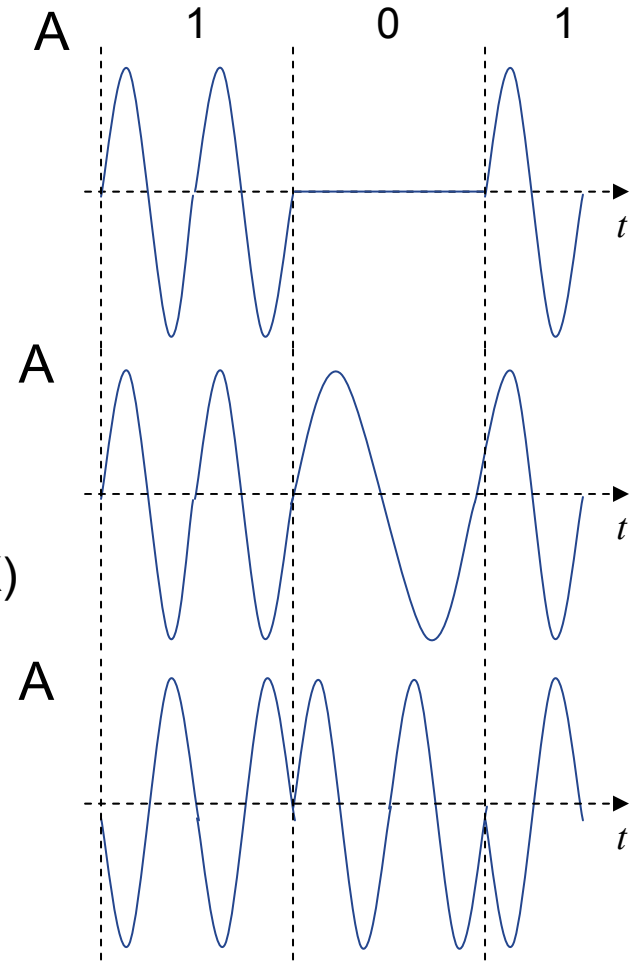
# Phasenmodulation




- Primäres Signal wird mittels gezielter Phasensprünge des Trägersignals moduliert
  - „*Phase modulation with reference phase*“
    0 = Phasendrehung des Referenzsignals um 180°
    1 = keine Phasendrehung (Referenzsignal)
  - „*Differential two-phase modulation*“
    0 = keine Phasendrehung [am Taktbeginn]
    1 = Phasendrehung [am Taktbeginn]
- Phasenmodulation ist das beste, aber auch aufwendigste Verfahren

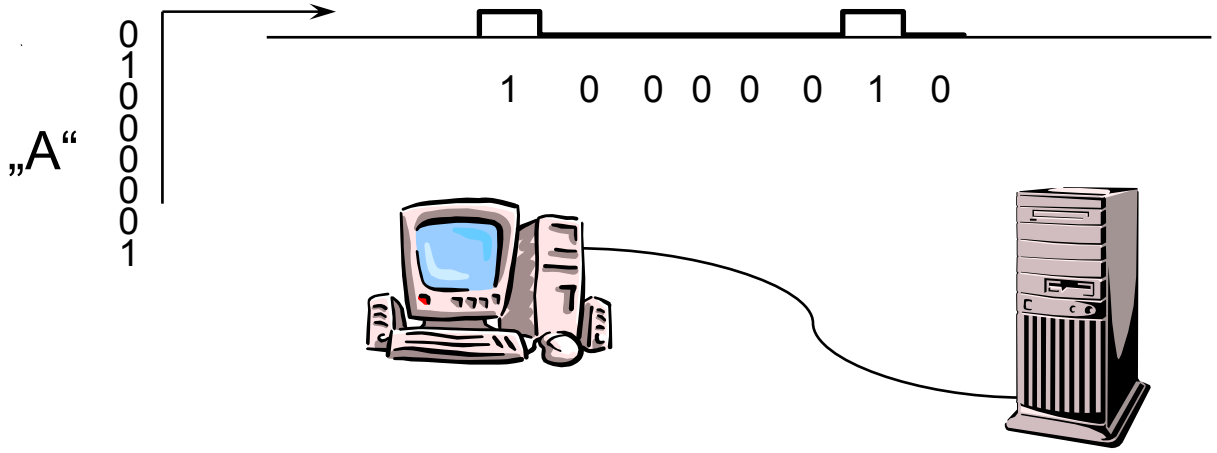
# Modulation: Zusammenfassung

- Amplitudenmodulation
  - technisch einfach
  - benötigt wenig Bandbreite
  - störanfällig
- Frequenzmodulation
  - größere Bandbreite
  - für Telefonübertragung
  - mit oder ohne Phasensprünge
  - Binary Frequency Shift Keying (BFSK)
    - Zwei Frequenzen 0: $f_1$, 1: $f_2$
- Phasenmodulation
  - komplexe Demodulation mit Trägerrückgewinnung
  - relativ störungssicher

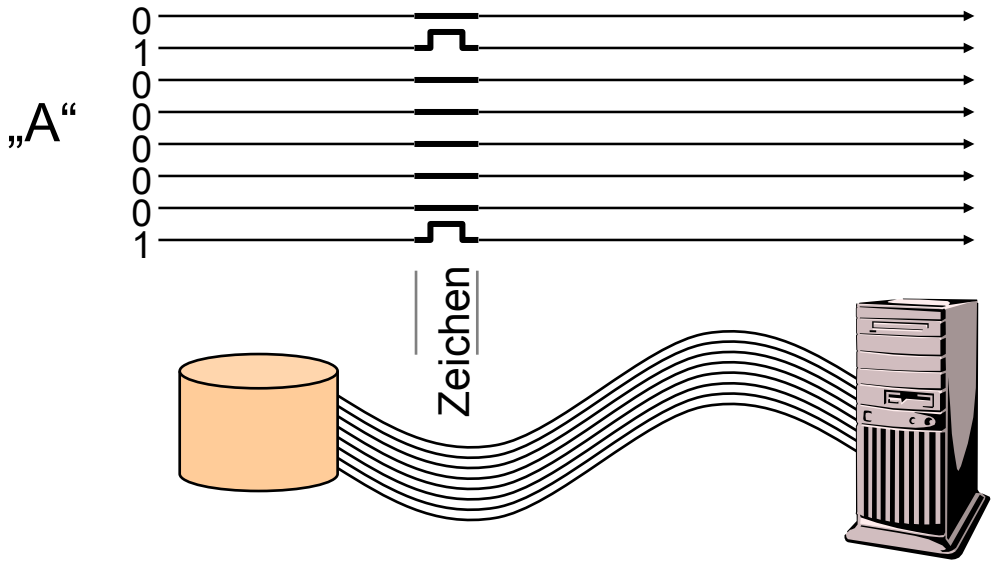


# 2.9 Serielle und parallele Übertragung

- Seriell




- Parallel

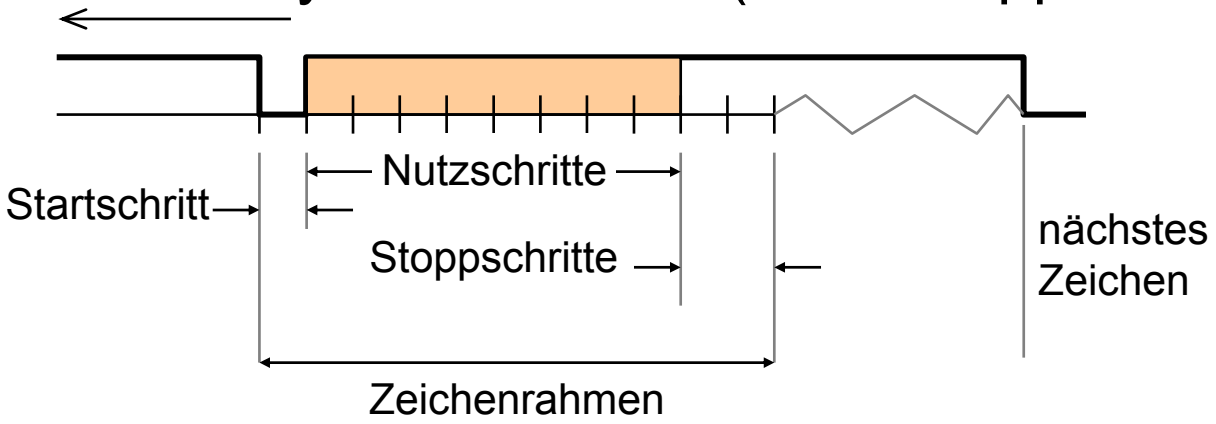


# 2.10 Synchronisation

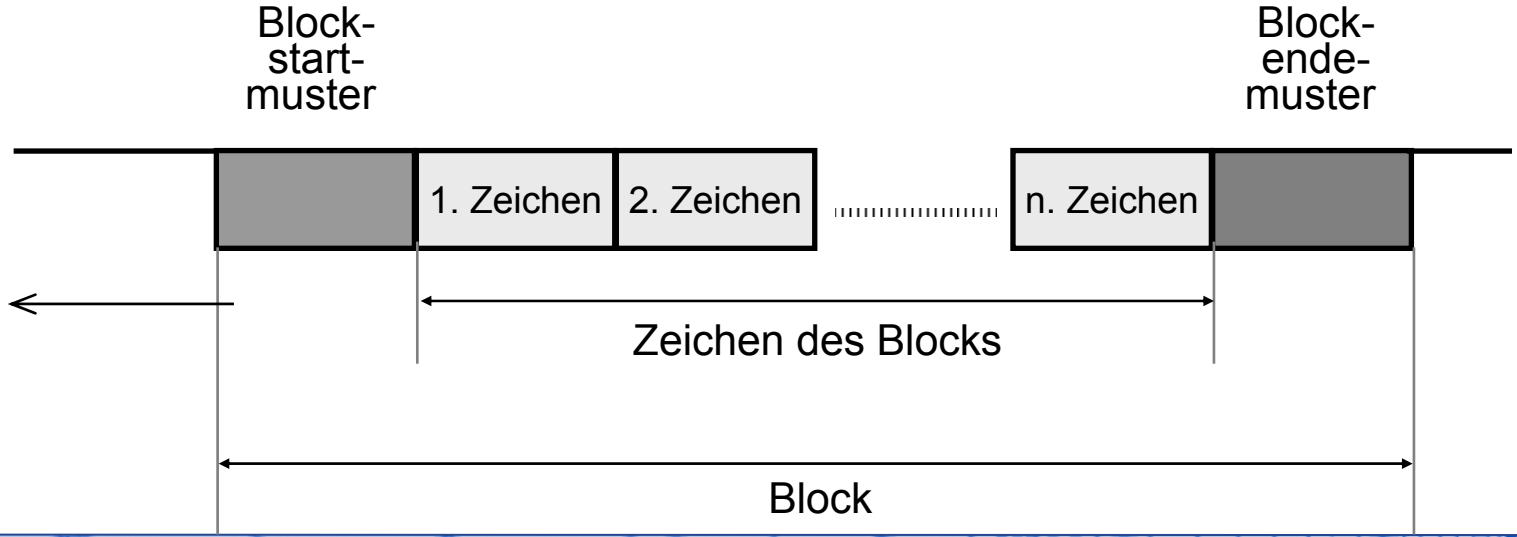
- Grundlegend für eine erfolgreiche Datenübertragung
  - Bestimmung des Abtastzeitpunkt
    → Synchronisation
- Formen der Synchronisation
  - Sende- und Empfangstakt unterliegen gemeinsamen Konventionen und werden diesen folgend von Quelle und Senke unabhängig voneinander unter Nutzung eines Taktgenerators bestimmt
    - äußerst stabile Taktgeneratoren erforderlich
  - Übertragung des Taktrasters auf eigenem parallelen Kanal
    - beschränkt auf Nahbereich
  - Übertragung des Taktrasters mit dem Signal
    - Ableitung des Taktrasters aus dem Signalverlauf
  - Punktuelle Synchronisation eines weitgehend unabhängigen Taktgenerators bei der Senke durch das Signal
    - nur beschränkte Frequenzkonstanz erforderlich

# Zeichenweise Synchronisation und Blocksynchronisation

- Zeichenweise Synchronisation (Start/Stopp-Betrieb)




- Synchronisation (nur) für jeden Block

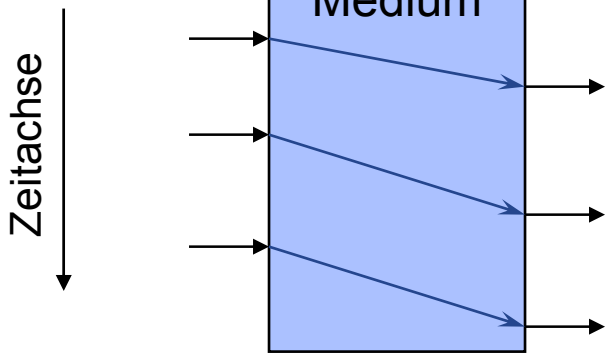


# 2.11 Mehrfachnutzung von Medien




- Dedizierter Kanal
  - Kanal verbindet genau eine Quelle mit genau einer Senke
  - Betriebsarten

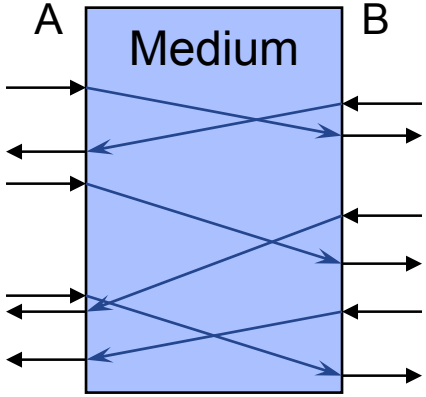
*simplex*




**Komm. nur in eine Richtung**

- Telex, Feuermelder

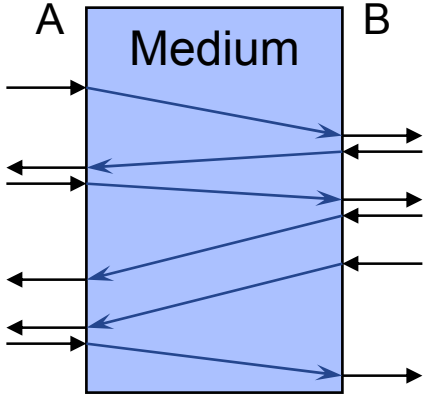
*duplex*




**Komm. gleichzeitig in beide Richtungen**

- Telefon

*halbduplex*




**Komm.richtung wechselt**

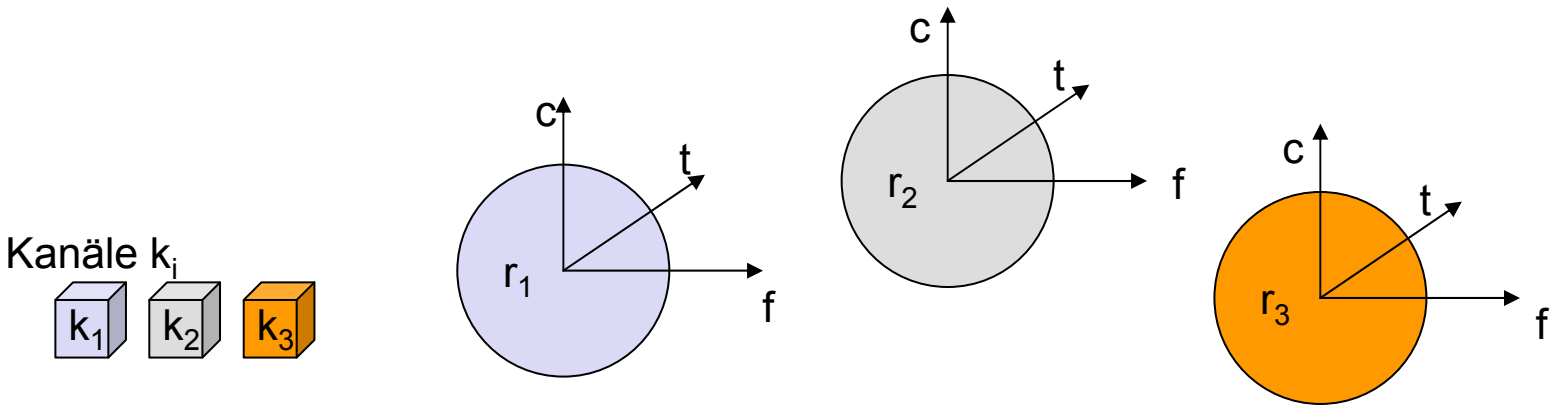
- Aber jeweils nur in eine Richtung genutzt
- Wechselsprechen (häufig beim Sprechfunk, z.B. Walkie-Talkie)

# Multiplextechniken

- Mehrfach genutzter Kanal, Sammelkanal
  - Ziel: Mehrfachnutzung des gemeinsamen Mediums
    - Mehr als zwei Dienstnehmer greifen auf dasselbe Medium zu
- Hierfür sind Multiplextechniken notwendig
- Multiplexen ist in vier Dimensionen möglich
  - Raum (r), Zeit (t), Frequenz (f) und Code (c)
  - Wichtig: Genügend große Schutzabstände nötig!

# Raummultiplex (SDMA)

- Einteilung des Raums in Sektoren, gerichtete Antennen
  - Space Division Multiple Access (SDMA)
- „Kupfermultiplex“
  - Zuordnung dedizierter Leitungen
- Vgl. Zellenstruktur von Mobilfunknetzen

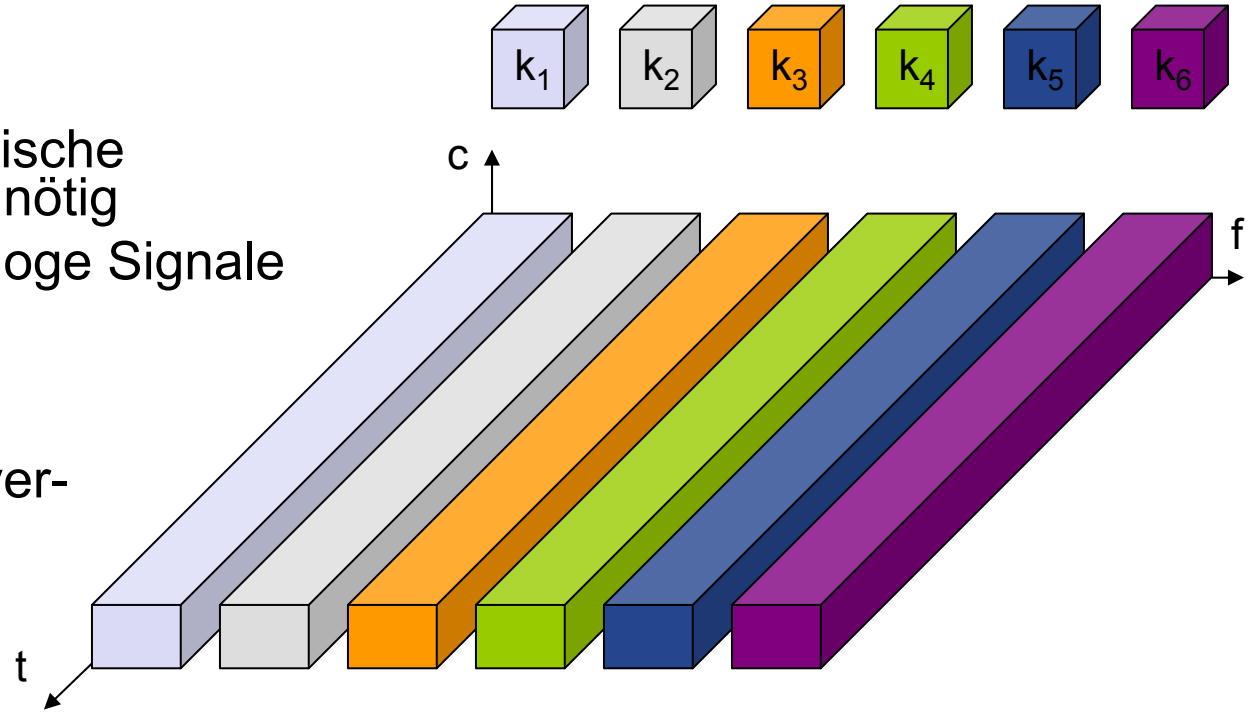


# Frequenzmultiplex (FDMA)

- Gesamte verfügbare Bandbreite wird in einzelne Frequenzabschnitte aufgeteilt
- Übertragungskanal belegt Frequenzabschnitt über gesamten Zeitraum
  - Frequency Division Multiple Access (FDMA)

- Vorteile
  - Keine dynamische Koordination nötig
  - Auch für analoge Signale

- Nachteile
  - Bandbreitenverschwendung bei ungleichmäßiger Belastung
  - Unflexibel

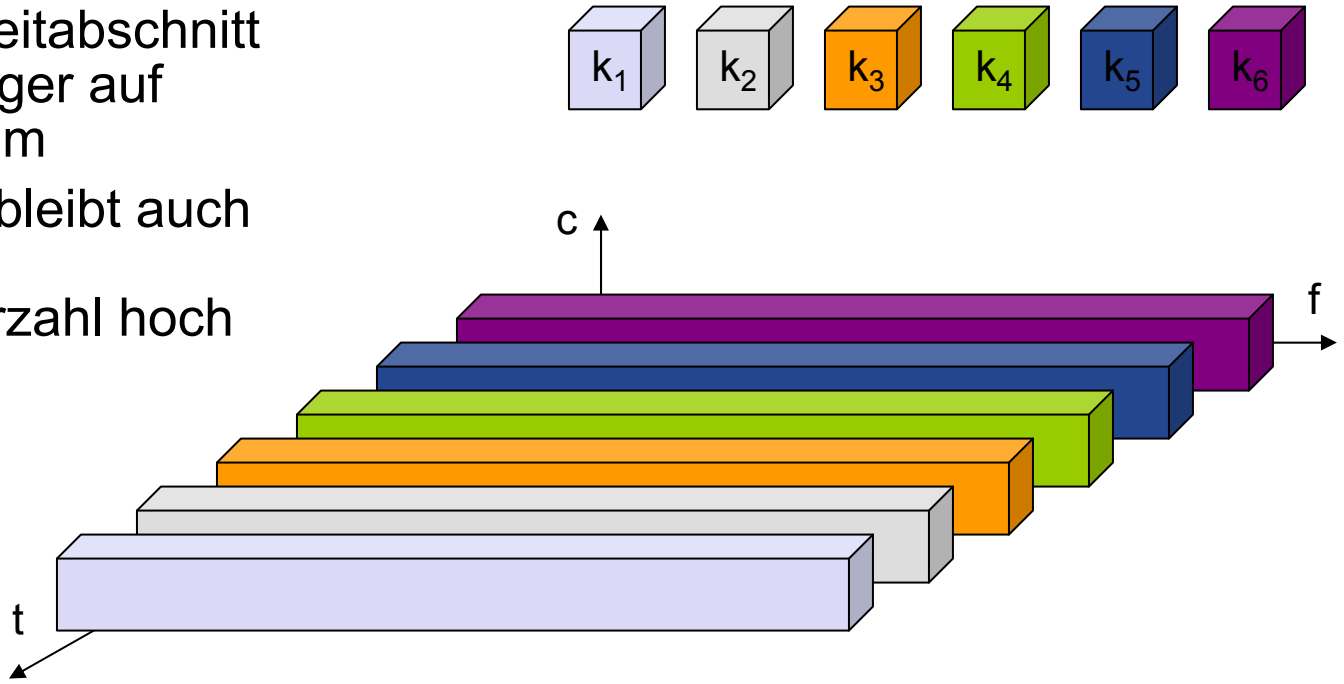


# Zeitmultiplex (TDMA)

- Kanal belegt gesamten Frequenzraum für einen gewissen Zeitabschnitt
  - Time Division Multiple Access (TDMA)

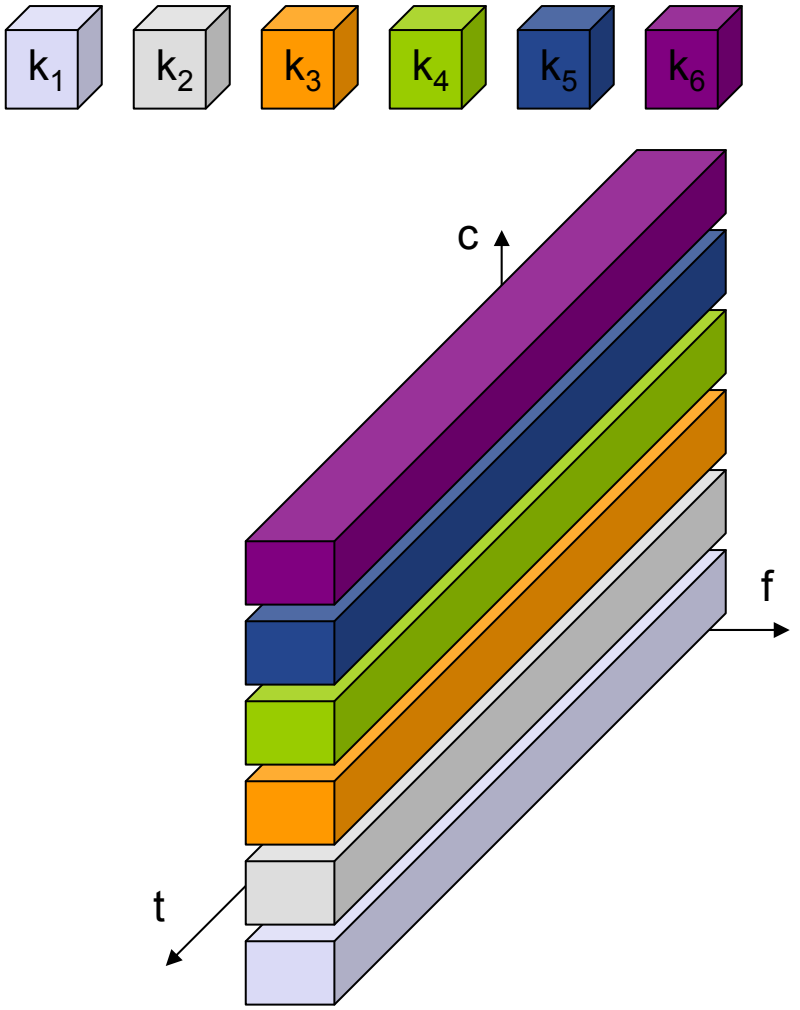
- Vorteile
  - In einem Zeitabschnitt nur ein Träger auf dem Medium
  - Durchsatz bleibt auch bei großer Teilnehmerzahl hoch

- Nachteile
  - Genaue Synchro-nisation nötig



# Codemultiplex (CDMA)

- Vorgehensweise
  - Alle Stationen operieren zur gleichen Zeit auf derselben Frequenz
  - Signal wird vom Sender mit einer für ihn eindeutigen Pseudozufallszahl verknüpft
  - Empfänger kann mittels bekannter Sender-Pseudozufallsfolge und Korrelationsfunktion das Originalsignal restaurieren
- Nachteil
  - Höhere Komplexität wegen Signalregenerierung
  - Alle Signale müssen beim Empfänger gleich stark ankommen
- Vorteile
  - Keine Frequenzplanung erforderlich
  - Sehr großer Coderaum (z.B. $2^{32}$) im Vergleich zum Frequenzraum
  - Vorwärtskorrektur und Verschlüsselung leicht integrierbar
- Realisierung: Spreizspektrumtechnik



# Zusammenfassung

- Systemblick auf Übertragungskanal
- Signalübertragung
- Signalklassen
- Abtasttheorem
  - Repräsentation eines analogen bandbeschränkten Signals als zeitdiskretes Signal
- Kanalkapazität
  - störungsfreier sowie gestörter Kanal
- Modulation
- Mehrfachnutzung

- Kapitel soll Einsichten zu folgenden Fragen geben
  - Was wird tatsächlich übertragen?
  - Welche Datenraten sind möglich?
  - Wie „stabil“ ist die Kommunikation im Übertragungskanal?

# Übungen

2.1 Erläutern Sie die Funktionsweise der Digitalisierung analoger Daten

2.2 Welche Rolle spielt in diesem Kontext das Abtasttheorem?

2.3 Wie groß ist die Standard-Datenrate eines Sprachkanals und aus welchen Zusammenhängen leitet sie sich ab?

2.4 Erläuten Sie den Unterschied zwischen Übertragungs- und Schrittgeschwindigkeit und nennen Sie deren Einheiten

2.5 Aus welchen Einzelkomponenten setzt sich die Übertragungszeit zusammen und wovon hängen diese jeweils ab?

2.6 Wie „lang“ ist eine Dateneinheit von 850 bit bei einer Datenrate von 1 Mbit/s?

2.7 Wieviele bit befinden sich gleichzeitig auf einer 1000 Mbit/s schnellen Glasfaserverbindung mit 1000 Kilometern Länge?

2.8 Nennen Sie unterschiedliche Modulationstechniken und erklären Sie deren Arbeitsweise

2.9 Erläutern Sie verschiedene Multiplex-Techniken und geben Sie Beispiele für deren Anwendung

# Literatur I


[Flag08] Flag Telecom; http://www.flagtelecom.com/

[Haaß97] W. Haaß; Handbuch der Kommunikationsnetze; Springer Verlag, 1997

- Gute knappe Zusammenfassung zu Übertragungsmedien und Multiplexverfahren

[Hals05] F. Halsall; Computer Networking and the Internet; Addison-Wesley, 2005

- Kapitel 1

[Hei08a] heise Newsticker; http://www.heise.de/newsticker/meldung/99240

[Hei08b] heise Newsticker; http://www.heise.de/newsticker/meldung/102751

[Hei08c] heise Newsticker; http://www.heise.de/newsticker/meldung/102870

[Hei08d] heise Newsticker; http://www.heise.de/newsticker/meldung/102931

[Hei08e] heise Newsticker; http://www.heise.de/newsticker/meldung/103135

[Hei08f] heise Newsticker; http://www.heise.de/newsticker/meldung/105984

[Hei09] heise Newsticker, http://www.heise.de/newsticker/meldung/120760

# Literatur II


[KlPi96] Klimant, Piotraschke, Schönfeld; Informations- und Kodierungstheorie; Teubner Verlag, 1996

[Kros91] K. Kroschel; Datenübertragung, Springer-Verlag, 1991

- Fundierte Diskussion von Aspekten der Datenübertragung
- Geht weit über das in der Vorlesung angesprochene hinaus

[KrRe04] G. Krüger, D. Reschke; Lehr- und Übungsbuch Telematik; Hanser Fachbuchverlag, 2004

- Kapitel 3 & Abtasttheorem, Weiterführende Behandlung von Codes

[NIST98] NIST Ref., Prefixes for binary multiples, http://physics.nist.gov/cuu/Units/binary.html

[Spie08] Spiegel Online; http://www.spiegel.de/netzwelt/tech/0,1518,533040,00.html

[Stal06] W. Stallings; Data & Computer Communications; Prentice Hall, 2006

- Kapitel 3 bis 5

# Kommunikation und Datenhaltung

## 3. Protokollmechanismen




**Prof. Dr. Martina Zitterbart**
Dipl.-Inform. Martin Röhricht
[zit | roehricht]@tm.uka.de



# Kapitelübersicht

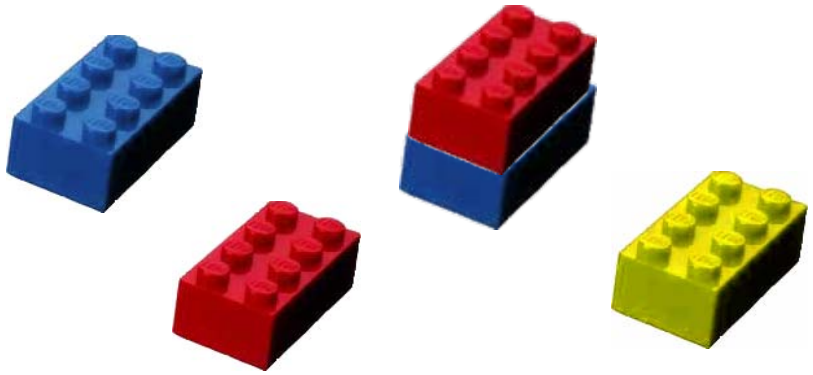
1. Einführung
2. Physikalische Grundlagen
3. Protokollmechanismen
4. Geschichtete Architekturen
5. Sicherungsschicht: HDLC
6. Beschreibungsmethoden
7. Sicherungsschicht: Lokale Netze
8. Netzkopplung und Vermittlung
9. Die Transportschicht
10. Anwendungssysteme
11. Middleware

3.1 Fehlertypen, Fehlerursachen
3.2 Mechanismen zur Fehlererkennung und -behebung
3.3 Fehlerkontrolle bei Bitfehlern
3.4 Fehlerkontrolle bei Paketfehlern
3.5 Flusskontrolle
3.6 Verbindungen

# … Austausch von Daten

- Ziel
  - Geräte bzw. Anwendungen möchten Daten austauschen

- Protokolle erforderlich, die Formate und Regeln des Datenaustauschs festlegen
  - Protokollmechanismen stellen die Bausteine der Protokolle dar



# 3.1 Fehlertypen, Fehlerursachen

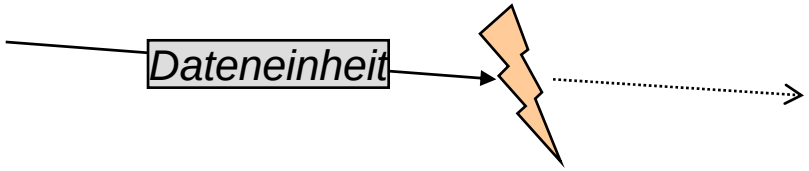
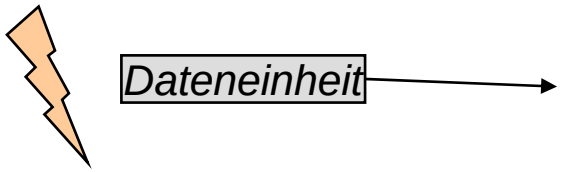
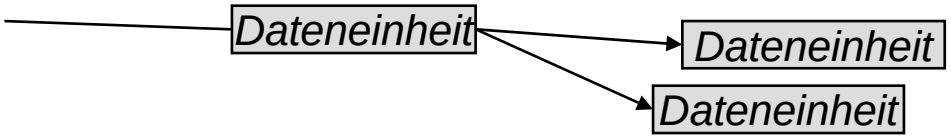
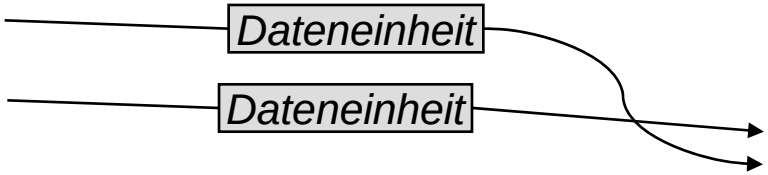
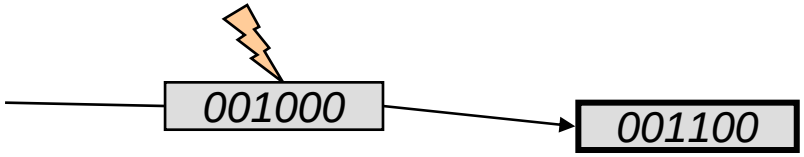
- Verfälschung von Bits bei der Übertragung - Bitfehler
  - Beispiel
    - Null-Bit werde durch 0 Volt repräsentiert; Eins-Bit durch 5 Volt
    - Entscheidungsschwelle sei 2,5 Volt
    - Übertragung ist nicht optimal: Rauschen, Signaldämpfung
    - Ergebnis: Empfänger empfängt Signalwert von 3 Volt, obwohl ursprünglich 0 Volt gesendet wurde
      - Es handelt sich dann um ein sogenanntes „umgekipptes“ Bit, d.h. um einen Bitfehler
  - Fehlerursachen
    - Rauschen
    - Verlust der Bit-Synchronisation

# Fehlertypen, Fehlerursachen

- Verfälschung von Dateneinheiten – oft als Paketfehler bezeichnet
  - Begriff Dateneinheit in der Vorlesung häufig als Oberbegriff verwendet für
    - Rahmen, Nachrichten, Pakete, Segmente …
  - Fehlerarten
    - Verlust/Duplizierung einer Dateneinheit
    - Empfang einer Phantom-Dateneinheit
    - Abweichung der Empfangsreihenfolge von Dateneinheiten
  - Fehlerursachen
    - Überlastung von Zwischensystemen
    - Unterschiedliche Wege durch das Netz
    - Verfrühte Datenwiederholung
    - …

# Fehlertypen

- Paket-fehler

| | |
|---|---|
|  | Verlust |
|  | Phantom-Dateneinheit |
|  | Duplizierung |
|  | Reihenfolge-vertauschung |
|  | Verfälschung |

- Bit-fehler

# Fehlertypen

- Weiterhin wird unterschieden zwischen
  - Einzelbitfehler
    - Z.B. Rauschspitzen, die die Detektionsschwelle bei digitaler Signalerfassung überschreiten
    - Ein einzelnes Bit ist fehlerhaft
  - Bündelfehler
    - Länger anhaltende Störung durch Überspannung, Starkstromschaltprozesse etc.
    - Mehrere direkt aufeinanderfolgende Bits sind fehlerhaft
  - Synchronisationsfehler
    - Empfänger kann den Anfang eines Bits nicht korrekt detektieren
    - Alle Bits werden falsch erkannt

# Fehlerhäufigkeit

- Maß für die Fehlerhäufigkeit

$$\text{Bitfehlerrate} = \frac{\text{Summe gestörte Bits}}{\text{Summe übertragene Bits}}$$

- Typische Wahrscheinlichkeiten für Bitfehler
  - Analoges Fernsprechnetz: $2*10^{-4}$
  - Funkstrecke: $10^{-3} – 10^{-4}$
  - Ethernet (10Base2): $10^{-9} – 10^{-10}$
  - Glasfaser: $10^{-10} – 10^{-12}$



[Holz91]

# Fehlerauswirkungen

- Fehlerauswirkungen sind u.a. abhängig von der Datenrate
  - Rechenbeispiel ... eine Störung von 20 ms führt
    - Bei Telex (50 bit/s, Bitdauer: 20 ms)
      - zu einem Fehler von 1 Bit
      - → Einzelbitfehler
    - Bei ISDN (64 kbit/s, Bitdauer: 15,625 µs)
      - zu einem Fehler von 1280 Bit
      - → Bündelfehler
    - Bei ADSL2+ (16 Mbit/s, Bitdauer: 62,5 ns)
      - zu einem Fehler von ca. 320 kbit
    - … und bei Gigabit-Ethernet (1 Gbit/s)?

# … Erwartungshaltung

- Unzuverlässige Kommunikation
  - Sender und Empfänger sind „glücklich“, wenn möglichst viel Daten korrekt beim Empfänger ankommen
  - Bei Fehlern werden keine weiteren Maßnahmen unternommen
- Zuverlässige Kommunikation
  - Sender und Empfänger erwarten, dass alle gesendeten Daten korrekt beim Empfänger ankommen
    - Alle Daten korrekt
    - In der richtigen Reihenfolge
    - Ohne Duplikate
    - Ohne Phantom-Dateneinheiten
  - Bei Fehlern sind entsprechende Maßnahmen erforderlich

# 3.2 Mechanismen zur Fehlererkennung und -behebung

- Bezüglich Bitfehlern
  - Fehlererkennende Codes
  - Fehlerkorrigierende Codes
- Bezüglich kompletter Dateneinheiten
  - Sequenznummern
  - Zeitüberwachung
  - Quittungen
  - Sendewiederholungen
- Redundanz
  - Fehlererkennung von Datenfehlern beim Empfänger durch Hinzufügung von Redundanz beim Sender

# Code

- Gegeben
  - eine Menge $A$ und ein (endliches) Alphabet $B$
- Code
  - injektive Abbildung $f: A \rightarrow B^*$
- Codewort
  - Für $a \in A$ heißt $f(a)$ ein Codewort von $f$
- Beispiele
  - Morse Code
  - ASCII Code
  - …

| | | |
|---|---|---|
| A .- | N -. | 0 ----- |
| B -... | O --- | 1 .---- |
| C -.-. | P .--. | 2 ..--- |
| D -.. | Q --.- | 3 ...-- |
| E . | R .-. | 4 ....- |
| F ..-. | S ... | 5 ..... |
| G --. | T - | 6 -.... |
| H .... | U ..- | 7 --... |
| I .. | V ...- | 8 ---.. |
| J .--- | W .-- | 9 ----. |
| K -.- | X -..- | Fullstop .-.-.- |
| L .-.. | Y -.-- | Comma --..-- |
| M -- | Z --.. | Query ..--.. |

# 3.3 Fehlerkontrolle bei Bitfehlern

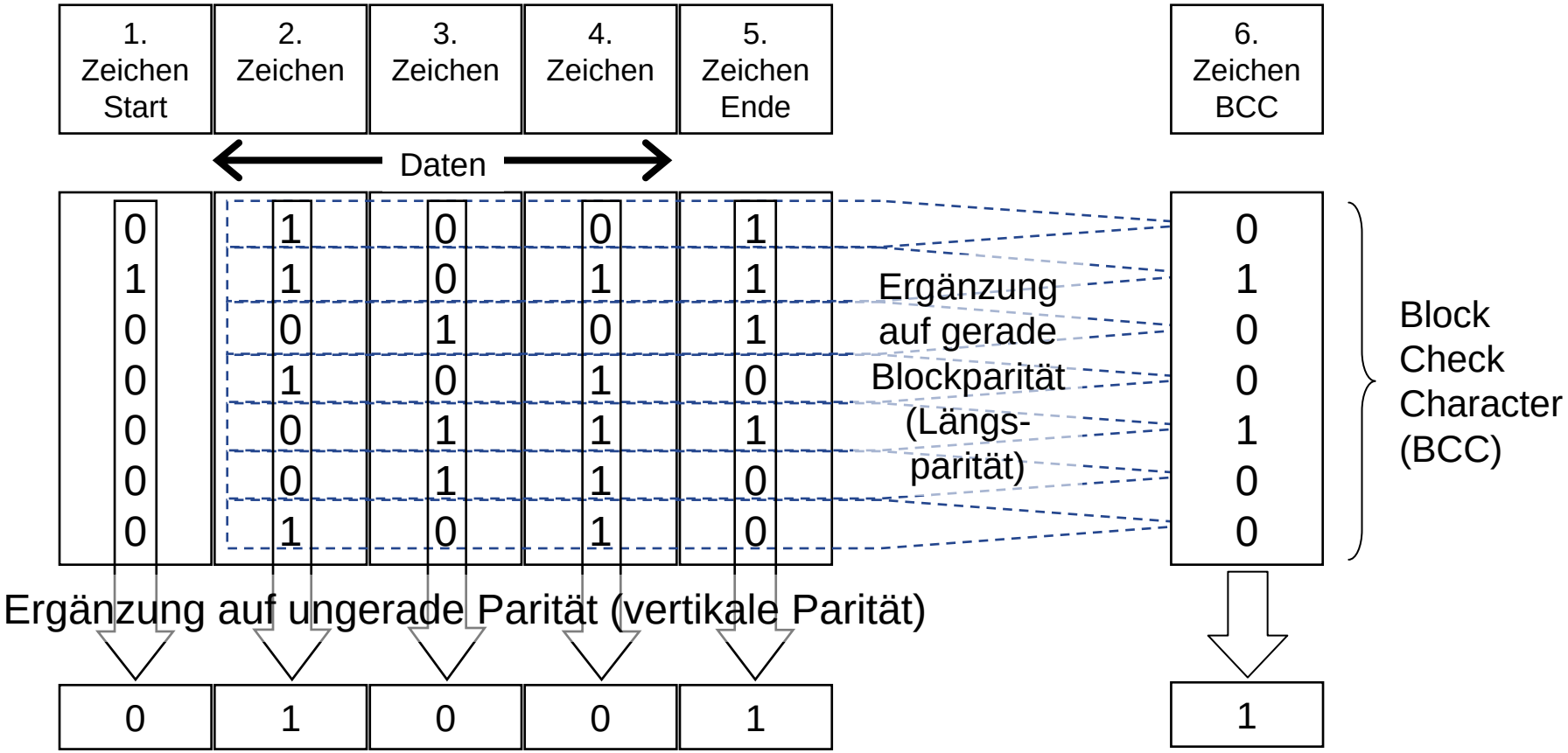
- Problem
  - Wie können Bitfehler beim Empfänger oder in netzinternen Zwischensystemen erkannt werden?
- Grundlegende Ansätze
  - Ausnutzung der „Distanz“ zwischen gültigen Codewörtern, d.h. nicht alle Codewörter, die mit den vorhandenen Bits erzeugt werden können, sind gültig
    - z.B. Hamming-Abstand
  - Hinzufügen von Redundanz bei der Übertragung
    - z.B. Paritätsbits, CRC, …

# 3.3.1 Paritätsbits

- Zu einer „Einheit“ wird jeweils ein redundantes Bit hinzugefügt.
  - Gerade Parität
    - Es wird auf gerade Anzahl von 1-Bits ergänzt
  - Ungerade Parität
    - Es wird auf ungerade Anzahl von 1-Bits ergänzt
- Folgende Varianten werden unterschieden
  - Vertikale Parität
    - An jede einzelne Einheit (Zeichen) – bestehend aus $n$ Bits – wird ein Paritätsbit angefügt (d.h. ein Paritätsbit pro Reihe)
    - Erkennung von Bitfehlern ungerader Anzahl (1-Bitfehler, 3-Bitfehler etc.)
  - Längsparität
    - An eine Folge von Einheiten wird ein dediziertes Prüfzeichen angefügt. Dieses enthält jeweils ein Paritätsbit pro Spalte (d.h. pro $n$-tem Bit aller Zeichen)
    - Auch als Block Check Character (BCC) bezeichnet.

# Paritätsbits: Beispiel

- Paritätssicherung bei Zeichen-basierter Übertragung
  - Modulo-2-Arithmetik



# 3.3.2 Hamming-Abstand

- Hamming-Abstand $d$ einzelner Codewörter
  - Anzahl der Bitpositionen, in denen sich zwei Codewörter $c_1$ und $c_2$ unterscheiden
  - Beispiel
    - `d(10001001,10110001)=3`
      - Anzahl der Eins-Bits von $c_1$ XOR $c_2$

- Hamming-Abstand $D$ des vollständigen Codes $C$

$$D(C) := \min\{d(c_1, c_2) \mid c_1, c_2 \in C, c_1 \neq c_2\}$$

- Der Hamming-Abstand bestimmt die Fähigkeit eines Codes, Fehler zu erkennen und zu beheben
  - Erkenne $k$-Bitfehler: Hamming-Abstand von $k + 1$ notwendig
  - Behebe $k$-Bitfehler: Hamming-Abstand von $2k + 1$ notwendig

# Hamming-Abstand

- Codewort besteht aus $m$ Bits
  - Welche Anzahl Prüfbits $r$ werden benötigt, um 1-Bitfehler zu beheben?




$n = m + r$

  - $2^m$ legale Codewörter können mit $m$ Bits erzeugt werden
  - Allgemein gilt: Pro Codewort $c_i$ der Länge $n=m+r$ existieren $n$ illegale Codewörter gleicher Länge mit Hamming-Abstand 1
    - Jeweils mit einem komplementierten Bit
      - Nur Codewort $c_i$ selbst hat Hamming-Abstand 0
  - Es soll $2^m$ legale Codewörter im Code geben, von denen jedes $n+1$ Codewörter belegt ($n$ illegale und ein legales, $c_i$ selbst)
  - Hierzu müssen $(n + 1)2^m$ Codewörter mit $n=m+r$ Bits darstellbar sein
    - $(n + 1)2^m <= 2^n \rightarrow (m + r + 1) <= 2^r$ untere Grenze für $r$

# Beispiele

- Fehlererkennender Code
  - Code mit einem einzigen Paritätsbit (gerade oder ungerade)
  - Hamming-Abstand = 2
  - Erkennen eines 1-Bitfehlers möglich (oder aller Fehler mit einer ungeraden Anzahl Bits)

- Fehlerbehebender Code
  - Code: **00000 00000, 00000 11111, 11111 00000, 11111 11111**
  - Hamming-Abstand = 5
  - Korrektur von 2-Bitfehlern möglich
  - Beispiel: **00000 00111 → 00000 11111**

# 3.3.3 Cyclic Redundancy Check (CRC)

- Anstelle eines einzigen Paritätsbits wird hier eine Sicherungssequenz angefügt.
  - auch als FCS (Frame Check Sequence) bezeichnet

- Basiert auf Division in Modulo-2-Arithmetik; keine Überträge
  - Entspricht bitweiser **XOR**-Operation
    - **1+1 = 0+0 = 0, 1+0 = 0+1 = 1**
  - Beispiele für bitweise XOR-Operation

```
 10011011      00110011      11110000      01010101
+11001010     +11001101     -10100110     -10101111
---------     ---------     ---------     ---------
 01010001      11111110      01010110      11111010
```

- Bitstrings als Repräsentation von Polynomen, z.B. Dateneinheit **10011010** entspricht Polynom $M(x) = x^7 + x^4 + x^3 + x^1$

- Dateneinheit wird als unstrukturierte Bitfolge aufgefasst, d.h. auch die Anzahl der zu prüfenden Bits ist beliebig
  - Es werden auch keine ganzzahligen Vielfache von 8 Bit gefordert

# Vorgehensweise CRC

- Gleiches Generatorpolynom **G(x)** für Sender und Empfänger
  - Höchstes und niederwertigstes Bit von **G(x)** müssen 1 sein
- Prüfsumme (Checksum) wird berechnet
  - Dateneinheit mit **m** Bits entspricht **M(x)**
  - Prüfsumme entspricht Rest **R** der Division $(x^r M(x)) / G(x)$
    - **r**: Grad des Generatorpolynoms; **m > r**
    - $x^r M(x)$ fügt **r** Nullstellen an das Ende der Dateneinheit
- Prüfsumme wird zur um Nullen erweiterten Dateneinheit hinzugefügt
  - Entspricht der Addition des Restes: $x^r M(x) + R$
- Empfänger überprüft Dateneinheit
  - Division der empfangenen Dateneinheit durch **G(x)**
    - Ist der Rest der Division Null, dann wurde kein Fehler erkannt
    - Ist der Rest ungleich Null, dann ist die empfangene Dateneinheit fehlerhaft

# CRC: Beispiel

- Generatorpolynom **G(x) = 1101**; Nachricht **M(x) = 1001 1010**

```
1001 1010 000     /    1101    =   1111 1001
--------------------------------
1101
-------
 100 1
 110 1
 --------
  10 00
  11 01
  --------
   1 011
   1 101
   --------
     1100
     1101
     -------
        1 000
        1 101
        --------
          101
```

- Welche Bitfolge wird übertragen?

# Erkennen von Bitfehlern mit CRC

- Erkennen aller Einzelbitfehler
  - $x^k$ und $x^0$ dürfen nicht gleich Null sein
- Nahezu alle Doppelbitfehler
  - **G(x)** muss mindestens drei Terme besitzen
  - sämtliche Doppelbitfehler
    - $(x^k + 1)$ nicht durch **G(x)** teilbar ($\forall$k <= Länge der Dateneinheit)
- Jede ungerade Anzahl an Bitfehlern
  - **G(x)** muss den Faktor **x+1** enthalten
- Alle Bursts mit bis zu **m** Bitfehlern
  - für **m** = Grad von **G(x)**

# Bekannte Generatorpolynome

- International genormt sind u.a. folgende Generatorpolynome
  - CRC-12 = $x^{12} + x^{11} + x^3 + x^2 + x + 1$
  - CRC-16 = $x^{16} + x^{15} + x^2 + 1$
  - CRC-CCITT = $x^{16} + x^{12} + x^5 + 1$
  - CRC-32 = $x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$

- CRC-16 und CRC-CCITT entdecken
  - alle Einzel- und Doppelfehler
  - alle Fehler ungerader Anzahl
  - alle Fehlerbursts mit der Länge $\leq 16$
  - 99,997 % aller Fehlerbursts mit der Länge 17 (gilt nicht für CRC-16)
  - 99,998 % aller Fehlerbursts mit der Länge 18 und mehr (gilt nicht für CRC-16)

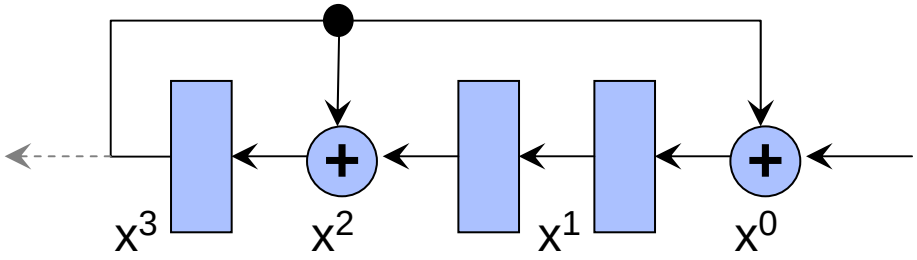
# CRC-Implementierung in Hardware

- Realisierung in Hardware
  - Benutzung von rückgekoppelten Schieberegistern
  - CRC kann während des „Durchschiebens“ durch das Schieberegister berechnet werden
- Prinzip
  - Daten werden bitweise empfangen und durchlaufen das Schieberegister
  - Rückkopplung durch ein XOR-Gatter erfolgt am Eingang und an den Stellen, an denen Bits im Generatorpolynom auf 1 gesetzt sind
    - Ohne das höchste und niederwertigste Bit
  - Nach Durchschieben der Dateneinheit und ihrer angehängten Nullen steht Prüfsumme im Register
    - Zu beachten: Soll Prüfsumme ebenfalls „aus dem Register geschoben werden“, noch etwas mehr Schaltungslogik nötig

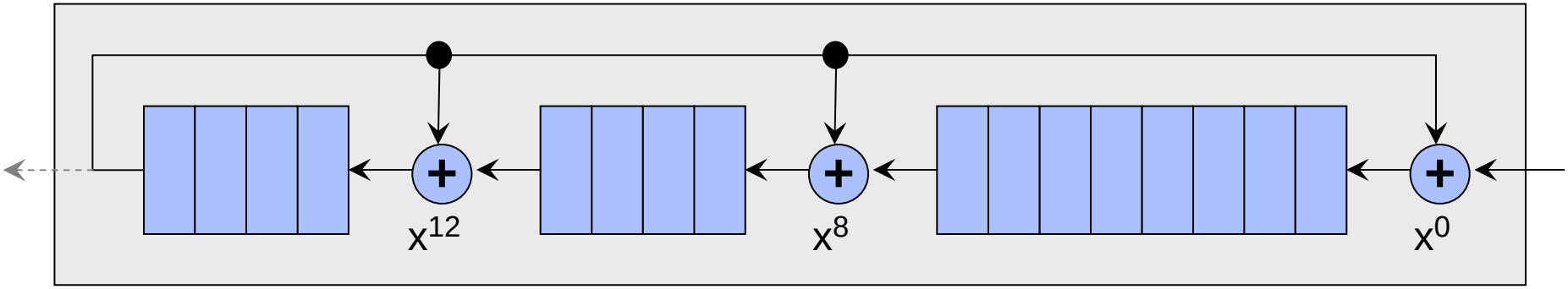
# Beispiele Schieberegister

- G(x) = 1101

  $G(x) = x^3 + x^2 + 1$




- $G(x) = x^{16} + x^{12} + x^8 + 1$




XOR-Gatter

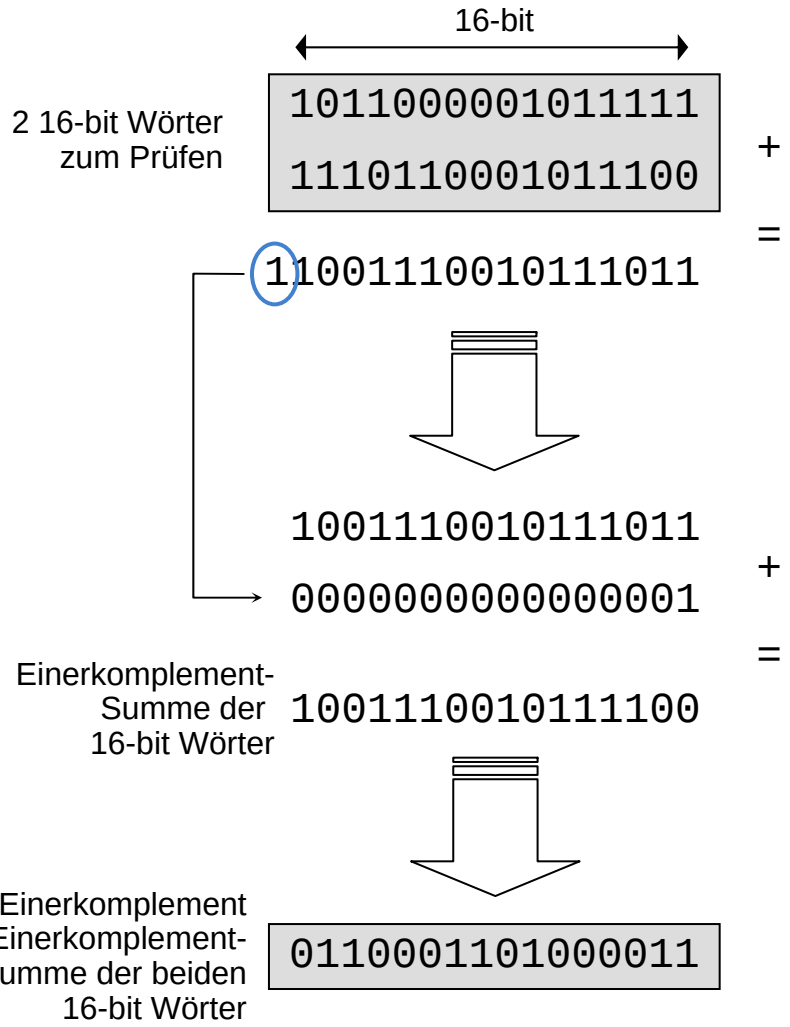
1 Bit-Schieberegister

# 3.3.4 Internet-Prüfsumme

- In den Internet-Protokollen wird eine andere Variante für Prüfsummen eingesetzt
  - Speziell für eine Realisierung in Software ausgelegt
  - In den Eigenschaften zur Fehlererkennung nicht so gut wie CRC
    - Wörter in falscher Reihenfolge können nicht erkannt werden
  - Verwendet bei: Internet Protocol (IP), Universal Datagram Protocol (UDP), Transmission Control Protocol (TCP)
- Prinzip
  - Aufaddieren aller übertragenen Wörter, wobei eine Wortlänge von 16 Bit zugrunde liegt (die Wörter werden als **`Integer`** aufgefasst)
  - Resultat der Addition wird wie beim CRC mit der Dateneinheit übertragen
  - Empfänger führt die gleiche Addition der empfangenen Wörter durch
- Implementierung
  - Addition unter Verwendung des Einer-Komplements

# Prüfsummen: Beispiel IPv4

- Einerkomplement zur Berechnung von BCC
- Internet Protocol Version 4
  - Protokollkopf besitzt Prüfsumme für Werte des Protokollkopfs
    - Muss bei jedem Router neu berechnet werden, da sich Werte wie Time to Live im Protokollkopf ändern
  - Prüfsummenberechnung
    - Addition aller 16-bit Wörter
      - Übertrag auf das Least Significant Bit addieren
    - Einerkomplement der Summe bilden




[Benv05]

# 3.3.5 Vorwärtsfehlerkorrektur

- Vorwärtsfehlerkorrektur (Forward Error Correction – FEC)
  - Fehlerkorrigierende Codes
- Soll dazu dienen, verloren gegangene Dateneinheiten zu rekonstruieren
- Beispiel:
  Zu senden sind die Dateneinheiten

  0101 - D1
  1111 - D2
  0000 - D3
- Dazu wird über XOR eine weitere Dateneinheit berechnet

  1010 - D4
- Diese vier Dateneinheiten werden jetzt an den Empfänger gesendet



[Hals05]

# Vorwärtsfehlerkorrektur – Ablauf

- Der Empfänger muss nur drei der vier Dateneinheiten korrekt empfangen, um die fehlende Dateneinheit rekonstruieren zu können
  - Verknüpft korrekt empfangene Dateneinheiten mit XOR und rekonstruiert so die fehlende
    - D1 geht verloren

      1111
      0000
      1010

      0101 - D1
    - D2 geht verloren

      0101
      0000
      1010

      1111 - D2
    - D3 geht verloren

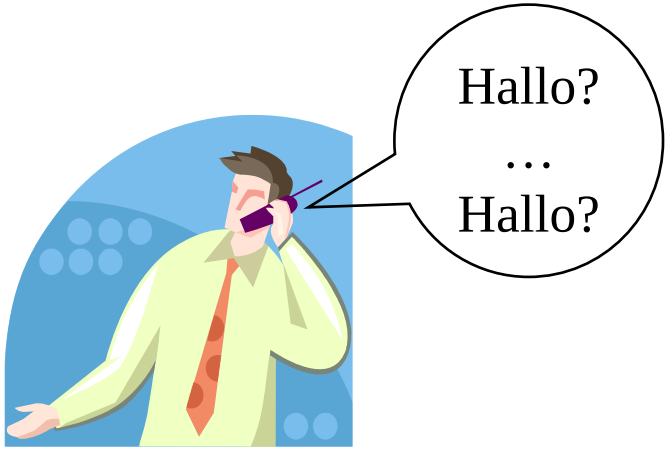
      0101
      1111
      1010

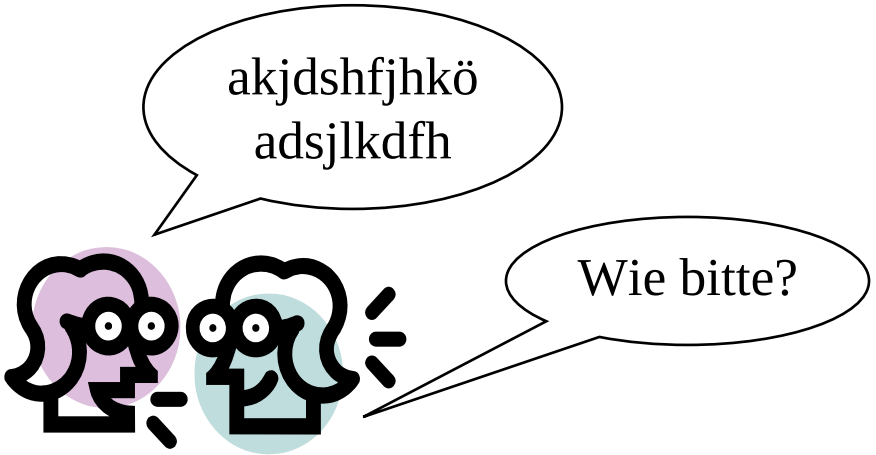
      0000 - D3
  - Der Empfänger muss wissen, welche Dateneinheit verloren gegangen ist ...

# 3.4 Fehlerkontrolle bei Paketfehlern

- Mit Paritätsbits und Prüfsummen können Bitfehler erkannt werden, allerdings nur dann, wenn die Dateneinheit beim Empfänger ankommt oder in den netzinternen Zwischensystemen analysiert wird
- Zur Erkennung von Fehlern bzgl. kompletter Dateneinheiten (Paketfehler) sind zusätzliche Mechanismen erforderlich
  - Sequenznummern (*Sequence Number*)
  - Zeitgeber (*Timer*)
- Auch wenn alle Dateneinheiten empfangen wurden, können die genannten Mechanismen erforderlich sein
  - Nennen Sie Gründe hierfür!
- Zur Behebung der Fehler werden die folgenden Mechanismen verwendet
  - Quittungen (*Acknowledgements*)
  - Sendewiederholungen (*Retransmissions*)

# … Verständigungsprobleme?









1, 2, 3, 5, … etwas fehlt hier …

# 3.4.1 Sequenznummern

- Problem
  - Woher weiß der Empfänger, ob
    - die Dateneinheiten in der richtigen Reihenfolge ankommen?
    - keine Duplikate enthalten sind?
    - keine Dateneinheiten fehlen?
- Mechanismus
  - Die Dateneinheiten (oder die Bytes) werden durchnummeriert
  - Eine entsprechende Kennung wird mit jeder Dateneinheit übertragen
  - Diese Kennung wird als Sequenznummer bezeichnet
- Fehlerszenarien
  - Sequenznummern helfen, wenn Dateneinheiten nicht ausgeliefert werden
    - Beispiele?

# Sequenznummernraum

- Definition
  - Teilmenge der natürlichen Zahlen, die den möglichen Sequenznummernraum umfasst
- Größe
  - Bei einer Länge der Sequenznummer von $n$ Bit umfasst der Sequenznummernraum $2^n$ Sequenznummern
- Erforderliche Länge der Sequenznummer?
  - Ziel
    - Overhead durch Übertragung von Sequenznummern gering halten
  - Parameter
    - $Tl$: maximale „Lebenszeit“ einer Dateneinheit [Sekunden]
    - $Tw$: Maximaler Zeitraum in dem eine Sendewiederholung durchgeführt wird [Sekunden]
    - $Tq$: Maximale Zeit bevor Empfänger nach Erhalt der Daten eine Quittung sendet [Sekunden]
    - $D$: Maximale Datenrate des Senders [Dateneinheiten/Sekunde]
  - Untere Schranke
    - $2^n \geq (2\ Tl + Tw + Tq) * D$

# 3.4.2 Quittungen

- Problem
  - Wie erfährt der Sender, dass eine Dateneinheit überhaupt nicht bzw. nicht korrekt beim Empfänger angekommen ist?
- Mechanismus
  - Der Empfänger teilt dem Sender mit, ob er eine Dateneinheit empfangen hat oder nicht
    - Hierzu werden spezielle Dateneinheiten, sogenannte Quittungen, versendet (ACK: Acknowledgement)
- Varianten
  - Positive Quittung
    - Empfänger teilt dem Sender mit, dass er die entsprechenden Daten erhalten hat
  - Negative Quittung
    - Empfänger meldet dem Sender, dass er die entsprechenden Daten nicht erhalten hat (z.B. bei falscher Reihenfolge)
      - NACK: Negative Acknowledgement

# Quittungen

- Weitere Varianten
  - Selektive Quittungen
    - SACK: Selective Acknowledgement
    - Die Quittung bezieht sich auf eine einzelne Dateneinheit
    - Beispiel
      - Negative selektive Quittung, falls der Verlust einer Dateneinheit vom Empfänger vermutet wird (NACK)
      - Lässt sich mit NACKs eine zuverlässige Kommunikation realisieren?
  - Kumulative Quittungen
    - Die Quittung bezieht sich auf eine Menge von Dateneinheiten, die in der Regel durch eine obere Sequenznummer beschränkt ist
    - Beispiel
      - Positive kumulative Quittung, die besagt, dass alle Dateneinheiten bis zur angegebenen Sequenznummer korrekt empfangen wurden
- Quittungen werden oftmals in Kombination mit Zeitgebern verwendet

# 3.4.3 Zeitgeber

- Problem
  - Woran merkt ein Sender, dass eine Dateneinheit nicht angekommen ist?
- Mechanismus
  - In Abhängigkeit einer zeitlichen Obergrenze wird vermutet, dass eine Dateneinheit nicht mehr beim Empfänger eintrifft
  - Sender kann dann Sendewiederholung starten
- Implementierung
  - Welcher Wert wird für den Zeitgeber gewählt?

# 3.4.4 Automatic Repeat Request: ARQ

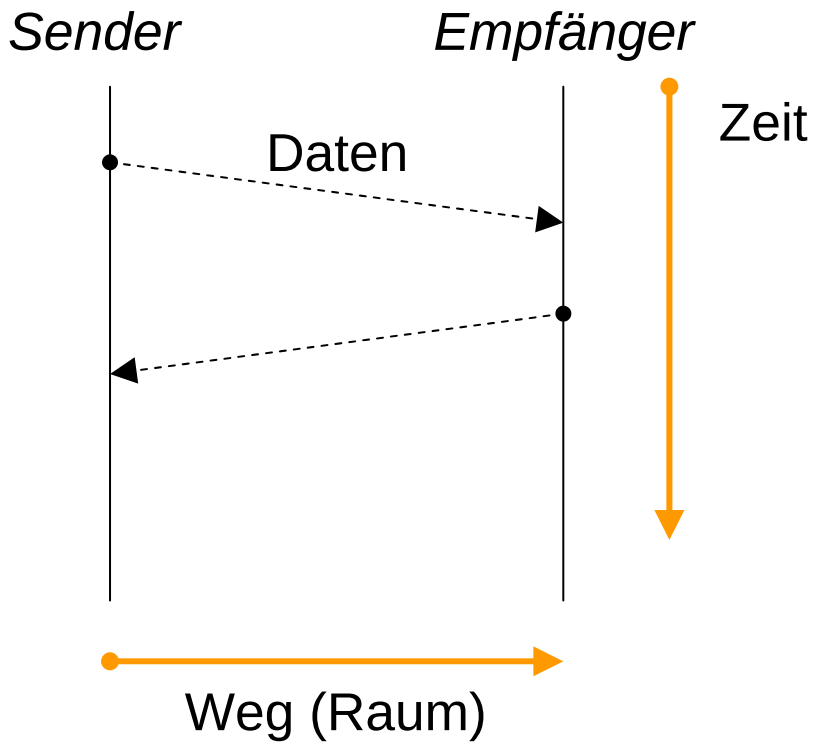
- Grundlegende Variante zur Sendewiederholung
  - Sender erhält positive Quittungen über den Erhalt einer Dateneinheit vom Empfänger
  - Sender kann Sendewiederholungen ausführen, falls Quittung ausbleibt
- Varianten
  - Wann werden Quittungen versendet?
  - Wann werden Sendewiederholungen veranlasst?
- Situation
  - Eine Reihe verschiedener ARQ-Mechanismen ist verfügbar, wobei zwischen grundlegenden Unterschieden und Implementierungsfeinheiten differenziert werden muss
  - Im folgenden werden grundlegende Varianten diskutiert

# Stop-and-Wait

- Grundlegendes, einfaches ARQ-Verfahren
  - Sender wartet auf die Quittung zu einer gesendeten Dateneinheit, bevor er eine neue Dateneinheit senden darf
  - Falls keine Quittung empfangen wird, erfolgt die Wiederholung der Dateneinheit nach dem Ablauf des Zeitgebers
  - Es können Duplikate von Dateneinheiten entstehen
    - Wie?
- Einsatz beim Medienzugriff in drahtlosen lokalen Netzen

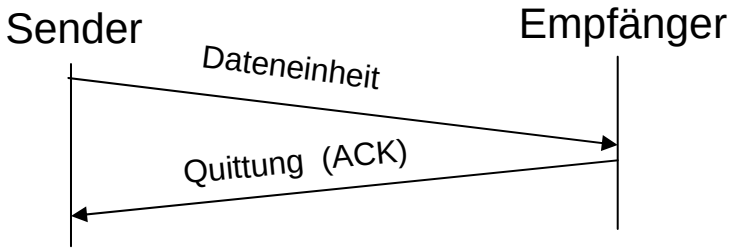
# Einschub: Weg-Zeit-Diagramm

- Beschreibungsmethode zur Darstellung von Abläufen in Kommunikationssystemen
  - Vorwiegend für einfachere Sachverhalte geeignet

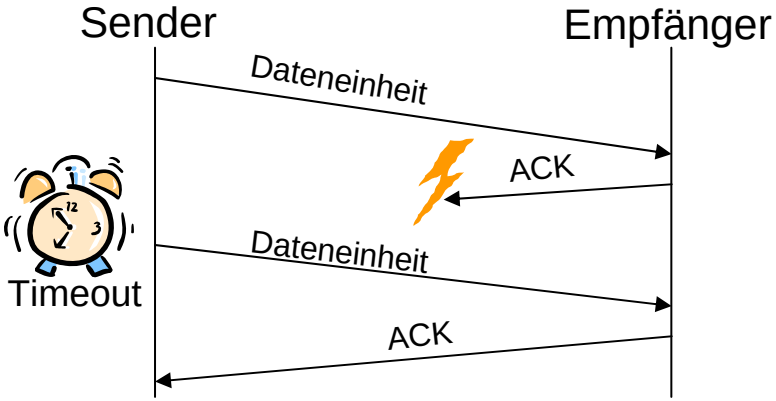


# Stop-and-Wait: Szenarien

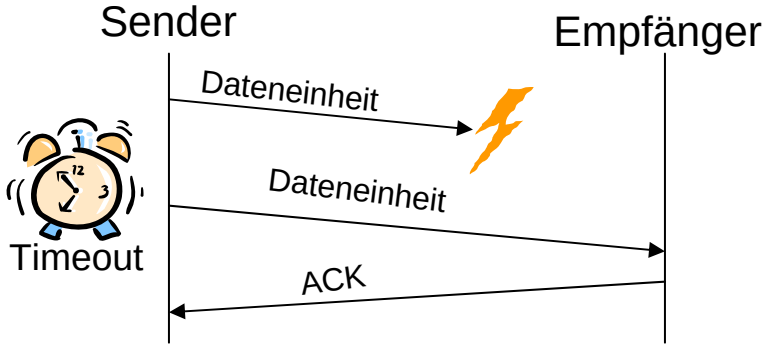
- Regulärer Ablauf



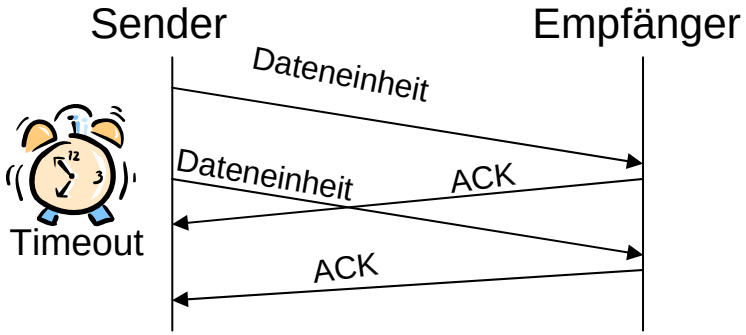

- Verlust einer Quittung
  - Unterschied zum Verlust einer Dateneinheit?




- Verlust einer Dateneinheit



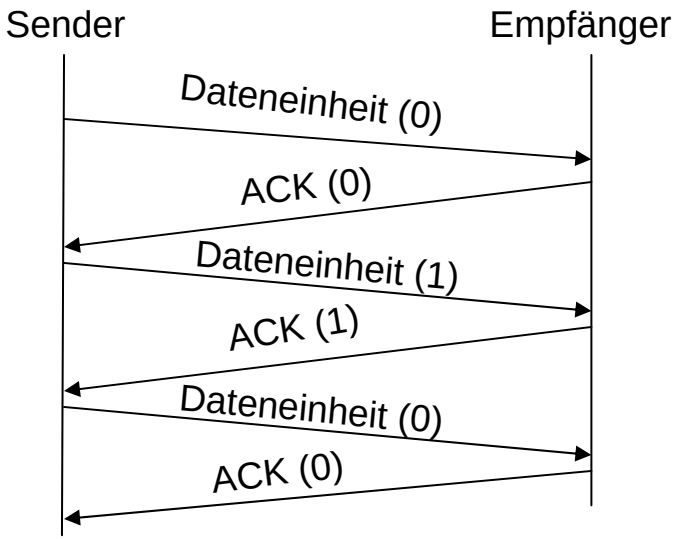

- Zu schneller Ablauf des Zeitgebers
  - Problem?




- Frage: Auf welchen Wert wird der Zeitgeber gesetzt?
- Frage: Unterscheiden sich Verluste bzw. Übertragungsfehler in der Behandlung?

# Stop-and-Wait: Sequenznummern

- Problem
  - In den vorangegangenen Szenarien besteht die Möglichkeit, dass der Empfänger eine Dateneinheit doppelt erhält
    - Er kann dies nicht erkennen
      - Konstruieren Sie einen solchen Ablauf!
- Mechanismus
  - Sequenznummern
    - Die Dateneinheiten werden mit einer Kennung versehen, die es dem Empfänger ermöglicht, diese zu unterscheiden
    - Für Stop-and-Wait ist eine Sequenznummer von einem Bit ausreichend (0 und 1)
- Ablauf

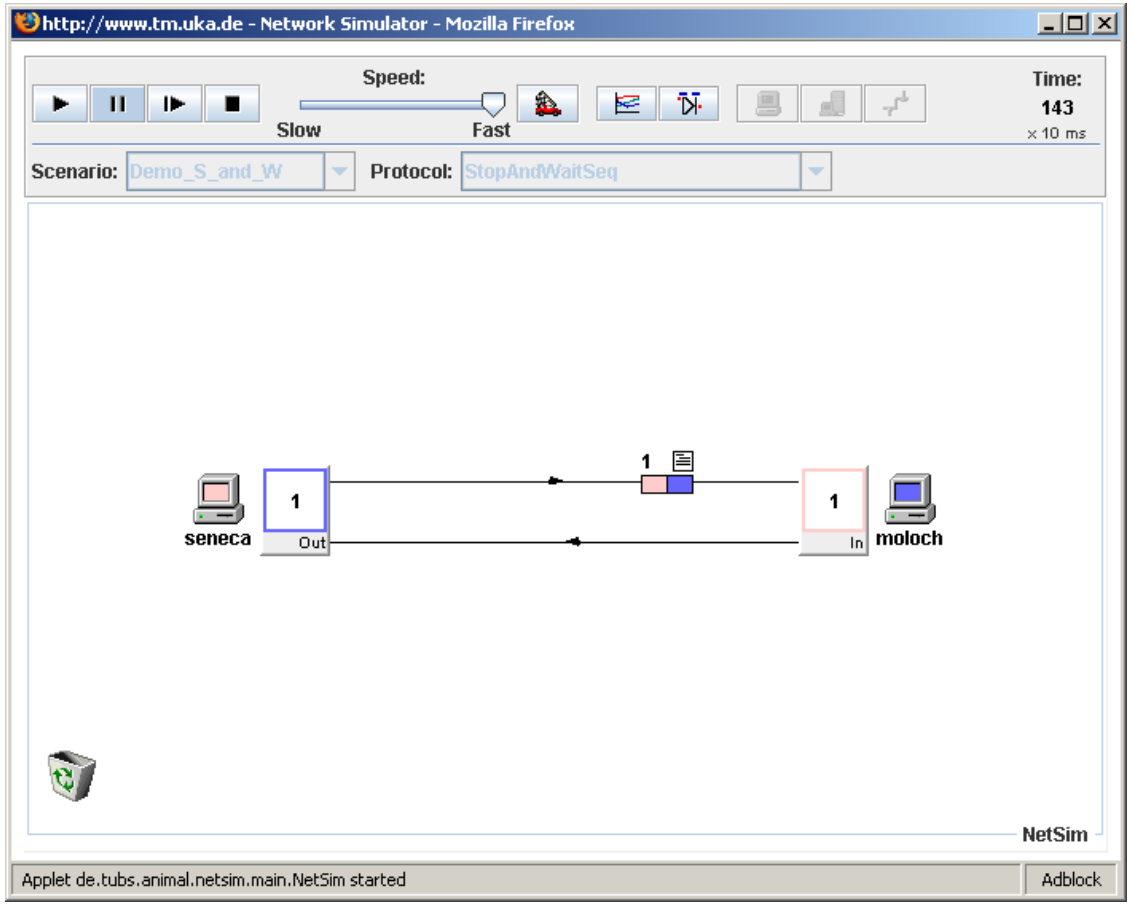


# Java-Applet: Stop-and-Wait

Testen Sie hierzu unsere Java-Applets und unsere Lernumgebung im Internet:

**http://www.tm.uka.de/lehre/aktuell/vl/telematik/sim/simulation.html**

- Einfaches Szenario mit zwei direkt verknüpften Rechnern
- Testen Sie die Leistungsfähigkeit der Fehlerkontrolle
  - Experimentieren Sie mit unterschiedlichen Verlust- bzw. Fehlerraten
  - Experimentieren Sie mit unterschiedlichen Datenraten



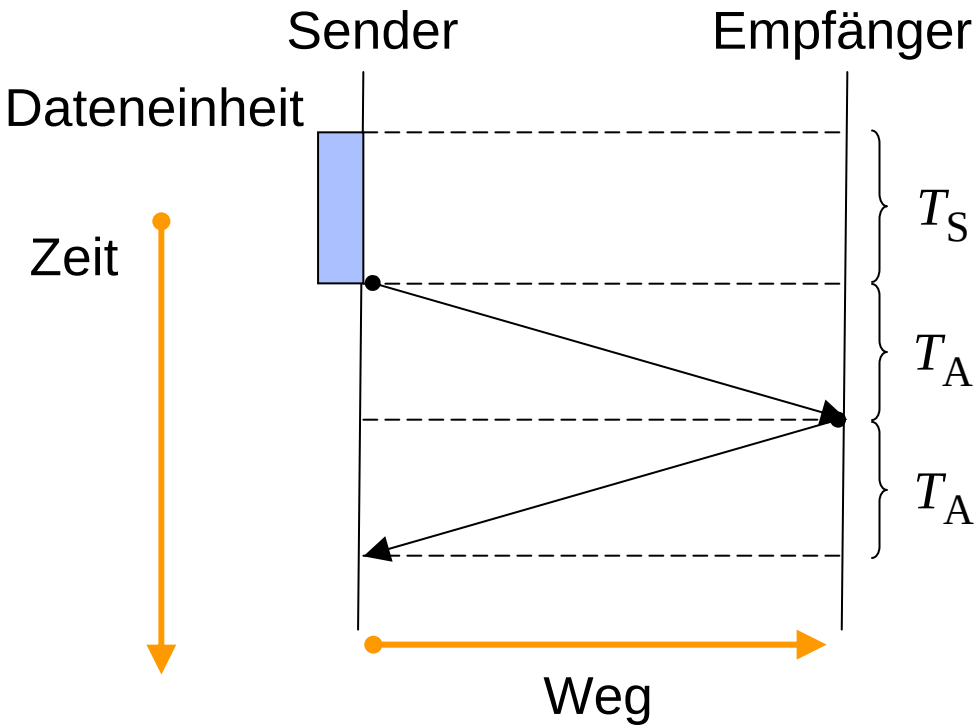
# Leistungsbewertung von Stop-and-Wait

- Sehr einfaches Verfahren
- Wie sieht es mit der Leistungsfähigkeit aus?
- Durchsatz (engl. *Throughput*)
  - Maß für die Menge an Daten, die pro Zeiteinheit über ein Netz übertragen werden kann
- Auslastung (engl. *Utilization* – $U$)
  - Maß für die Nutzung einer Ressource (z.B. Übertragungsmedium)
  - Verhältnis von tatsächlicher Nutzung zu möglicher Nutzung
    - Tatsächliche Nutzung: erfolgreiche Übertragung einer Dateneinheit
    - Mögliche Nutzung: Wieviel Daten hätten in dieser Zeit übertragen werden können?

# Leistungsbewertung von Stop-and-Wait

- Annahme
  - Fehlerfreie Kommunikation
- Welche Parameter haben einen Einfluss?
  - Wie schnell ist eine komplette Dateneinheit bzw. eine Quittung auf das Medium gesendet?
    - Sendezeit – $T_S$
      - $T_S$ = Länge Dateneinheit / Datenrate
  - Wie lange benötigt ein Bit vom Sender zum Empfänger (und umgekehrt)?
    - Ausbreitungsverzögerung – $T_A$
      - $T_A$ = Länge des Mediums / Ausbreitungsgeschwindigkeit
  - Wieviel Zeit wird für die Verarbeitung einer Dateneinheit bzw. einer Quittung benötigt?
    - Verarbeitungszeit – $T_V$
- Vereinfachungen
  - Verarbeitungszeit der Dateneinheit wird vernachlässigt
  - Sende- und Verarbeitungszeit einer Quittung werden vernachlässigt
    - Quittung kurz; Verarbeitung schnell

# Gesamtübertragungsdauer




- Insgesamt benötigte Zeit $T_{Ges}$
  - $T_{Ges} = T_S(\text{Daten}) + T_A + T_V(\text{Daten}) + T_S(\text{Quittung}) + T_A + T_V(\text{Quittung})$
- Mit Vereinfachungen
  - $T_{Ges} = T_S(\text{Daten}) + 2\,T_A = T_S + 2\,T_A$

# Auslastung des Mediums

- Auslastung $U$
  - Tatsächliche Nutzung
    - Sendezeit der Dateneinheit: $T_S$
  - Mögliche Nutzung
    - Zeitintervall vom Beginn des Sendens der Dateneinheit bis zum vollständigen Empfang der Quittung: $T_{Ges}$

$$U = \frac{T_S}{T_{Ges}} = \frac{T_S}{T_S + 2T_A}$$

$$U = \frac{1}{1 + 2T_A / T_S}$$

    - Mit $a = T_A / T_S$ ergibt sich

$$U = \frac{1}{1 + 2a}$$

# Parameter a

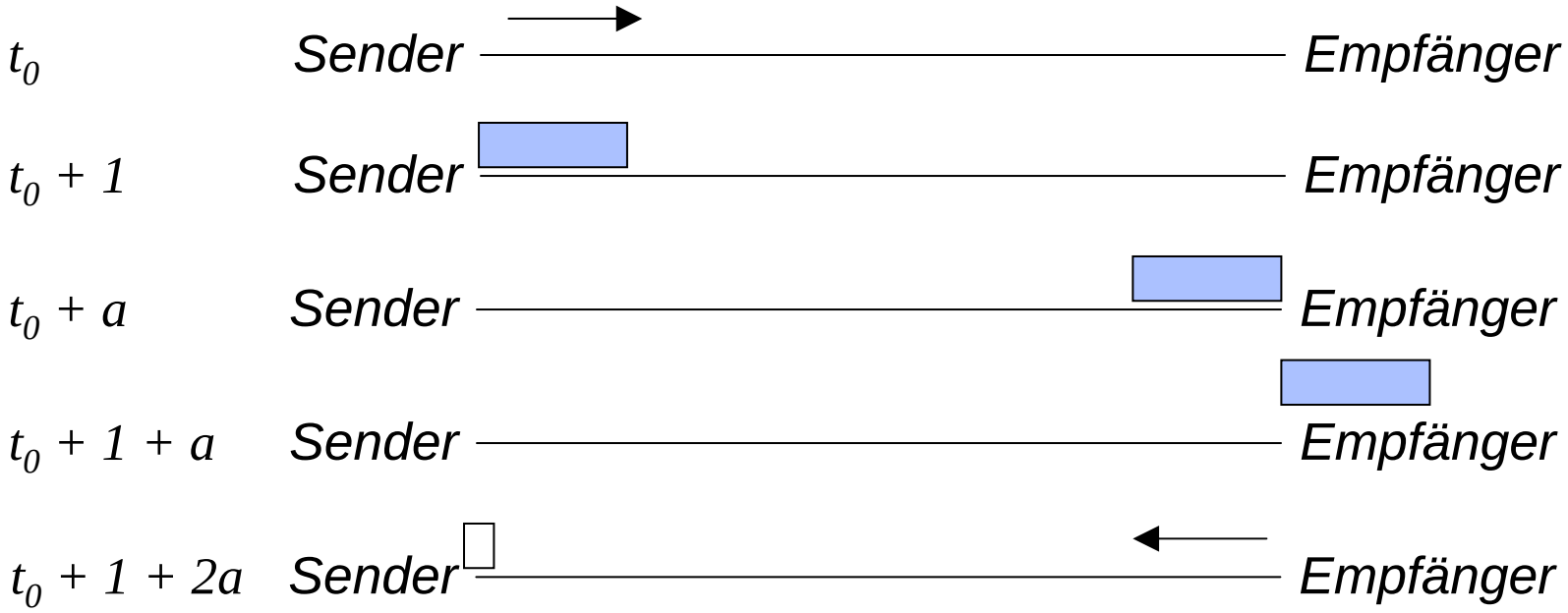
- Häufig bei Leistungsbetrachtungen verwendet
- Definition
  - $a = T_A / T_S$
  - $a$ = Ausbreitungsverzögerung / Sendezeit
- Weitere Parameter
  - Länge des Mediums: $m$
  - Ausbreitungsgeschwindigkeit: $v$
  - Länge der Dateneinheit: $l$
  - Datenrate: $d$
- Damit

$$a = \frac{m/v}{l/d} = \frac{md}{vl}$$

- Interpretation
  - Bei fester Länge der Dateneinheit und fester Distanz zwischen den kommunizierenden Systemen
    - Parameter $a$ ist proportional zum Produkt aus Datenrate und Länge des Mediums
  - Parameter $a$ repräsentiert Länge des Mediums in Bit ($m * d/v$) im Vergleich zur Länge der Dateneinheit ($l$)

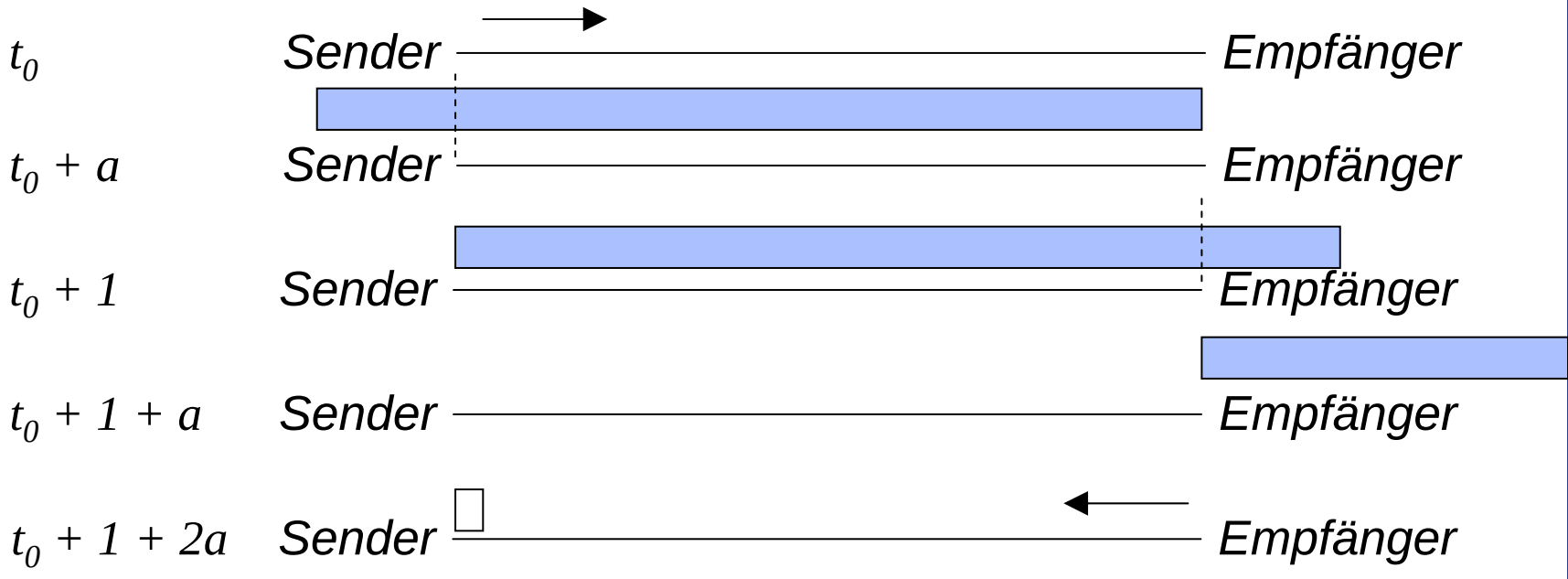
# Beispiel zu Parameter a

- Normalisierung
  - Annahme: Sendezeit einer Dateneinheit betrage 1
- Damit gilt
  - $a = T_A$
  - Für $a > 1$: Ausbreitungsverzögerung > Sendezeit



# Beispiel zu Parameter a

- Für $a < 1$: Ausbreitungsverzögerung < Sendezeit



# Auslastung des Mediums: Beispiel

- Dateneinheiten der Länge 1000 Bit
- Fehlerrate vernachlässigbar
- Datenraten 1 kbit/s und 1 Mbit/s
- Ausbreitungsgeschwindigkeit:
  - (a) und (b) $2 * 10^8\ m/s$
  - (c) $3 * 10^8\ m/s$
    - (a) Verdrilltes Adernpaar, Länge des Mediums 1 km
      - Auslastung?
    - (b) Standleitung, Länge des Mediums 200 km
      - Auslastung?
    - (c) Satelliten-Verbindung, Länge des Mediums 50 000 km
      - Auslastung?

# Bewertung von Stop-and-Wait

- Bisher: fehlerfreie Übertragung angenommen
- Jetzt: Berücksichtigung von Fehlern

- Fehlerfall
  - Sender erhält keine korrekte Quittung

- Annahme: $(n-1)$ konsekutive Sendewiederholungen
  - Benötigte Zeit hierfür
    - $T_{\text{Ges}} = T_{\text{S}} + (n-1)\,(\text{Timeout} + T_{\text{S}}) + 2\,T_{\text{A}}$
  - Annahme
    - Timeout entspricht zweifacher Ausbreitungsverzögerung $T_{\text{A}}$
    - Dann gilt: $T_{\text{Ges}} = n\,(T_{\text{S}} + 2\,T_{\text{A}})$
  - Damit gilt für die Auslastung

$$U = \frac{T_S}{T_{Ges}} = \frac{T_S}{n\left(T_S + 2T_A\right)}$$

  - … in der Regel $n$ nicht fest: Erwartungswert hierfür?

# Bewertung von Stop-and-Wait

- Annahmen
  - $p$ sei Wahrscheinlichkeit, dass eine einzelne Dateneinheit fehlerhaft übertragen wird
  - Quittungen sind nie verfälscht
- Wahrscheinlichkeit für genau $k$ Übertragungsversuche?
  - $k-1$ fehlerhafte Übertragungsversuche gefolgt von einer geglückten Übertragung
    - Wahrscheinlichkeit hierfür: $p^{k-1}(1-p)$
  - $n$ = Erwartungswert [Anzahl Übertragungen]

$$n = \sum_{i=1}^{\infty} (i \times p[\text{genau } i \text{ Übertragungen}]) = \sum_{i=1}^{\infty} i p^{i-1}(1-p) = \frac{1}{1-p}$$

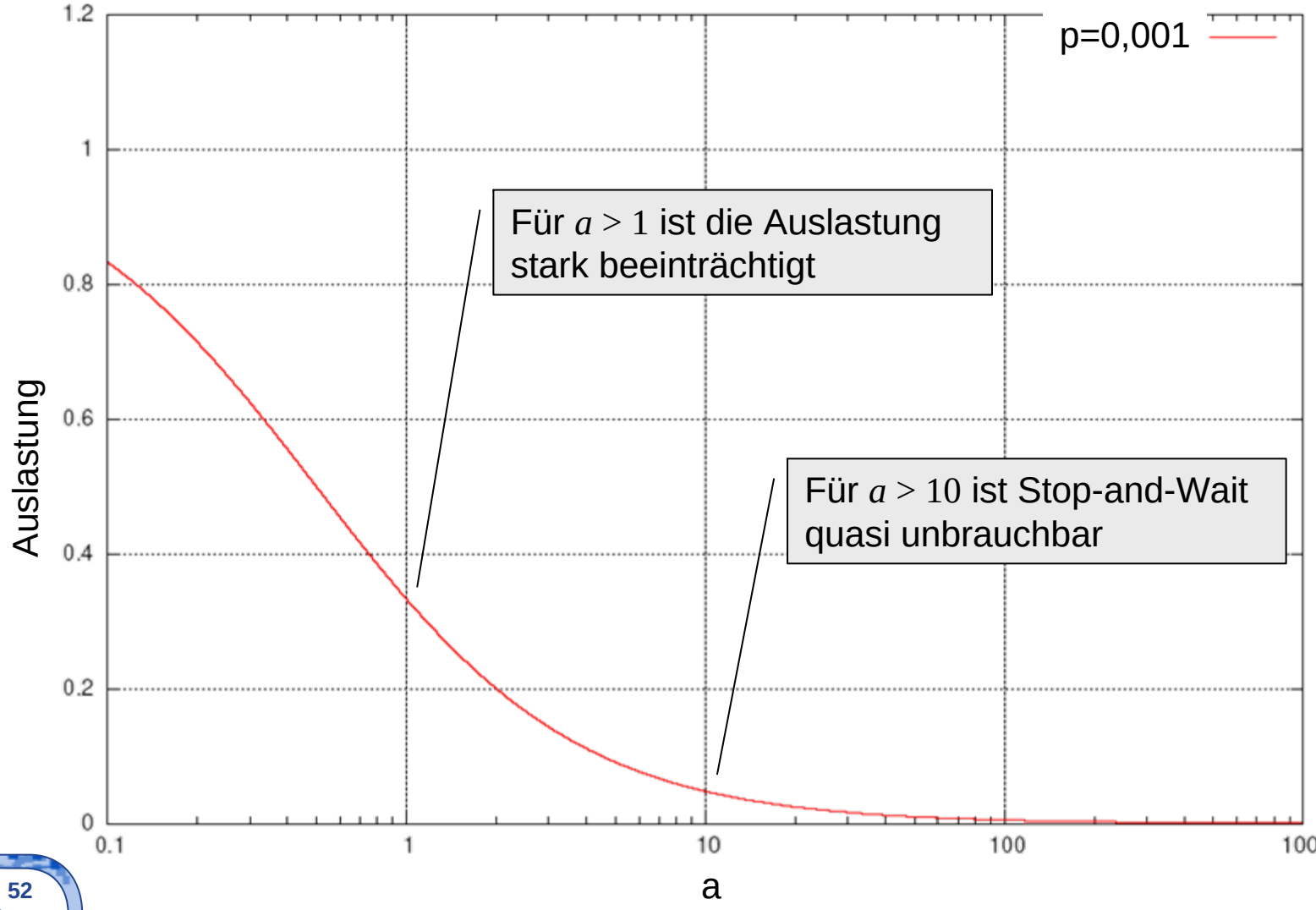
für $-1 < x < 1$ gilt:

$$\sum_{i=1}^{\infty} i x^{i-1} = \frac{1}{(1-x)^2}$$

  - Damit gilt für die Auslastung

$$U = \frac{1}{n(1+2a)} = \frac{1-p}{1+2a}$$

# Bewertung von Stop-and-Wait




[Stal06]

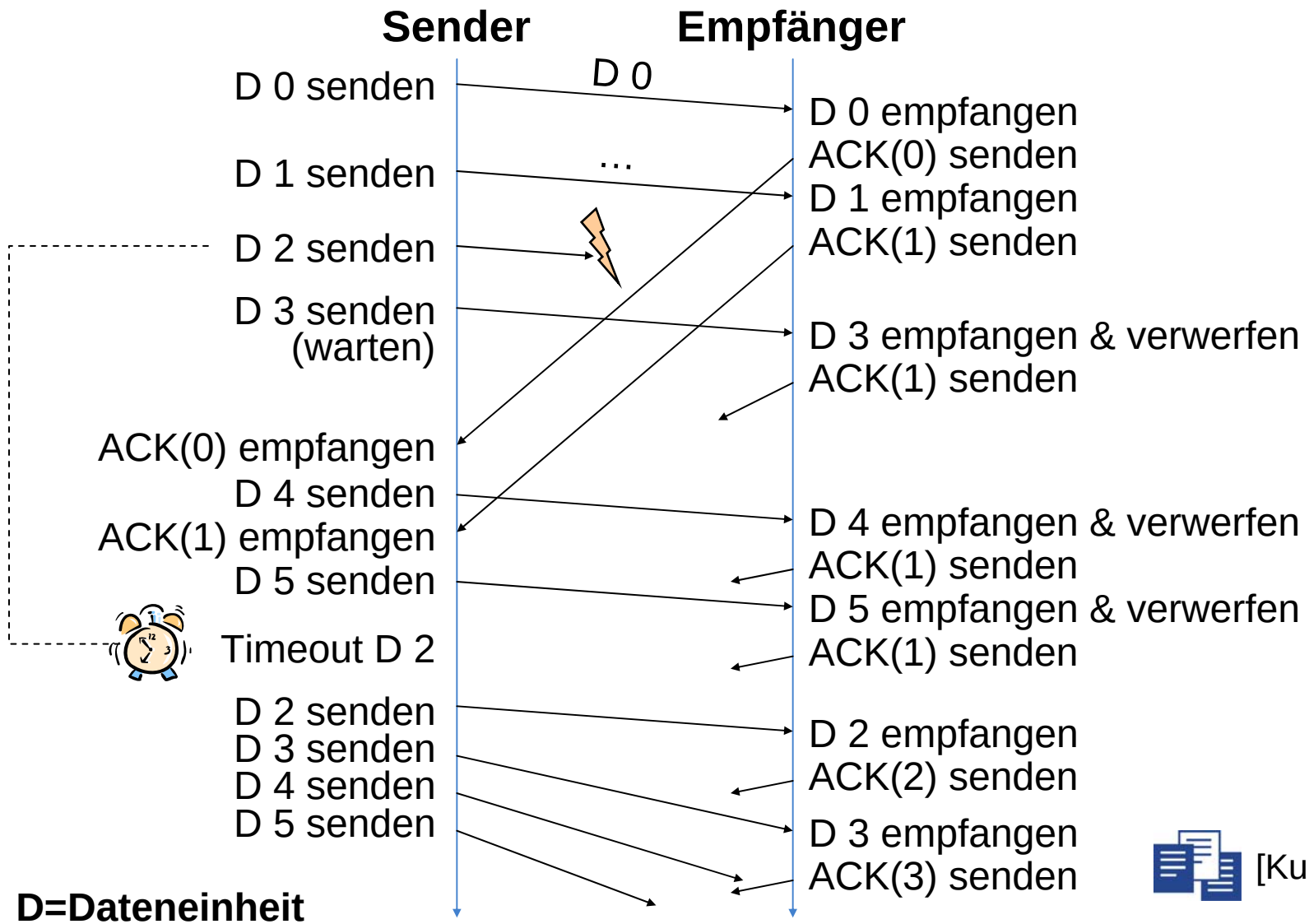
# Go-Back-N ARQ

- Ziel
  - Erhöhung der Leistungsfähigkeit im Vergleich zu Stop-and-Wait
    - Warten auf Quittung vor dem Senden der nächsten Dateneinheit vermeiden
- Verfahren
  - Der Sender kann mehrere Dateneinheiten senden bis er eine Quittung erhalten muss
  - Die maximale Anzahl der nicht quittierten Dateneinheiten ist begrenzt
    - typischerweise durch ein so genanntes Sliding Window
  - Verschiedene Varianten möglich (nächste Folie)

# Go-Back-N ARQ – Varianten

- Variante 1
  - Empfänger quittiert korrekt empfangene Dateneinheiten wie bei Stop-and-Wait
    - Die Quittung erfolgt kumulativ, d.h. für mehrere Dateneinheiten auf einmal
      - Kumulative Sequenznummer gibt an, bis wohin die Daten korrekt empfangen wurden, d.h. es handelt sich um positive Quittungen
      - Es kann auch jede Dateneinheit quittiert werden
- Variante 2
  - Nicht korrekt empfangene Dateneinheiten werden mit einer negativen Quittung (NACK) bestätigt
    - Sender wiederholt daraufhin ab dieser Sequenznummer alle gesendeten Dateneinheiten
      - Go-Back N, wobei N die Sequenznummer in der negativen Quittung ist
- Frage
  - Wo ist eine Pufferung der Dateneinheiten erforderlich? Wie viele müssen gepuffert werden?

# Go-back-N: Beispielablauf




[KuRo07]

**D=Dateneinheit**

# Bewertung von Go-Back-N

- Bemerkung
  - Fenstergröße $W$ begrenzt die Anzahl gesendeter und noch nicht quittierter Dateneinheiten
  - Parameter und Normalisierung wie bei Stop-and-Wait
- Annahme
  - Fehlerfreie Übertragung
- Zwei Fälle müssen unterschieden werden
  - Fenstergröße $W \geq 1+2a$
    - Sender kann ohne Pause senden
    - Übertragungsabschnitt ist 100% ausgelastet
  - Fenstergröße $W < 1+2a$
    - Sender kann nach dem Aufbrauchen des Fensters nicht weiter senden

$$U = \begin{cases} 1 & W \geq 1+2a \\ \dfrac{W}{1+2a} & W < 1+2a \end{cases}$$

# Bewertung von Go-Back-N

- Jetzt: Berücksichtigung von Übertragungsfehlern
- Herleitung ähnlich wie bei Stop-and-Wait, aber
  - Im Fehlerfall werden $K$ Dateneinheiten wiederholt anstelle einer einzigen
  - $n$ = Erwartungswert [Anzahl übertragener Dateneinheiten für eine geglückte Übertragung]
    - $f(i)$: Komplette Anzahl übertragender Dateneinheiten, falls die ursprünglich gesendete Dateneinheit $i$ mal übertragen werden muss

      $$f(i) = 1 + (i-1)K = (1-K) + Ki$$

    - Daraus ergibt sich für den Erwartungswert $n$

      $$n = \sum_{i=1}^{\infty} f(i)\, p^{i-1}(1-p)$$

# Bewertung von Go-Back-N

- Einsetzen von $f(i)$

$$n = \sum_{i=1}^{\infty} ((1-K)+Ki)\, p^{i-1}(1-p)$$

$$n = (1-K)\sum_{i=1}^{\infty} p^{i-1}(1-p) + K\sum_{i=1}^{\infty} ip^{i-1}(1-p)$$

$$n = 1-K+\frac{K}{1-p} = \frac{1-p+Kp}{1-p}$$

für $-1 < x < 1$ gilt:

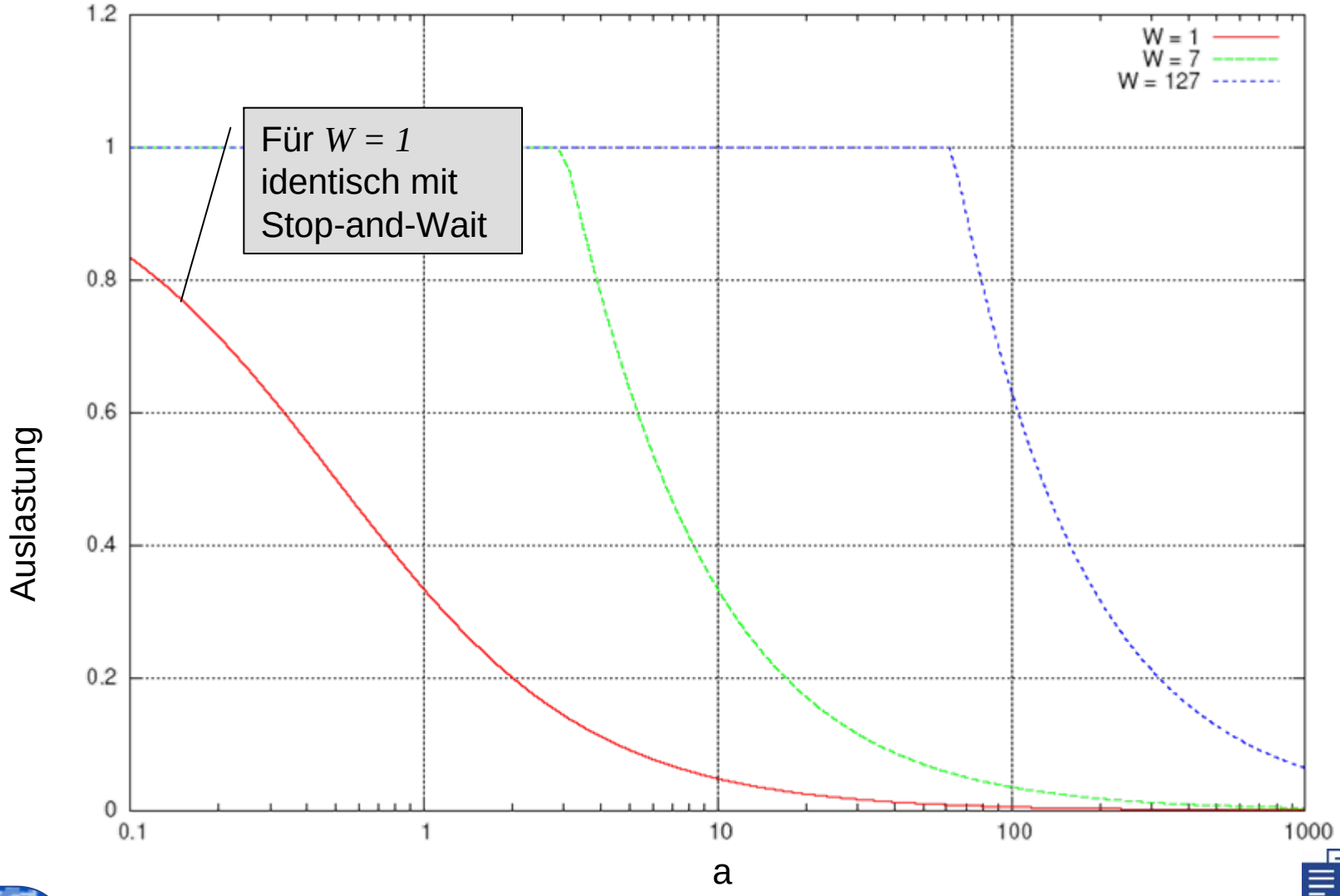
$$\sum_{i=1}^{\infty} x^{i-1} = \frac{1}{1-x}$$

  - Für $W \geq (1+2a)$: $K$ ungefähr $1+2a$
  - Für $W < (1+2a)$: $K = W$

- Damit

$$U = \begin{cases} \dfrac{1-p}{1+2ap} & W \geq 1+2a \\ \dfrac{W(1-p)}{(1+2a)(1-p+Wp)} & W < 1+2a \end{cases}$$

# Bewertung Go-Back-N




[Stal06]

# Selective Reject ARQ

- Ziel
  - Erhöhung der Auslastung im Vergleich zu Stop-and-Wait
  - Reduzierung des Datenaufkommens im Vergleich zu Go-Back-N
- Verfahren
  - Der Sender kann mehrere Dateneinheiten senden, bis er eine Quittung erhalten muss
  - Die maximale Anzahl der nicht quittierten Dateneinheiten ist begrenzt (wie bei Go-Back-N)
    - Der Empfänger sendet eine negative Quittung, wenn er einen Fehler erkennt
      - Diese Quittung bezieht sich auf eine einzelne Dateneinheit
    - Der Sender wiederholt genau die Dateneinheit mit der bei der negativen Quittung angegebenen Sequenznummer
      - Nur die nicht korrekt empfangenen Dateneinheiten werden vom Sender wiederholt
  - Fragen
    - Wo ist eine Pufferung der Dateneinheiten erforderlich? Wie viele müssen gepuffert werden?
    - Vor- und Nachteile von Go-Back-N und Selective Reject im Vergleich?

# Bewertung von Selective Reject

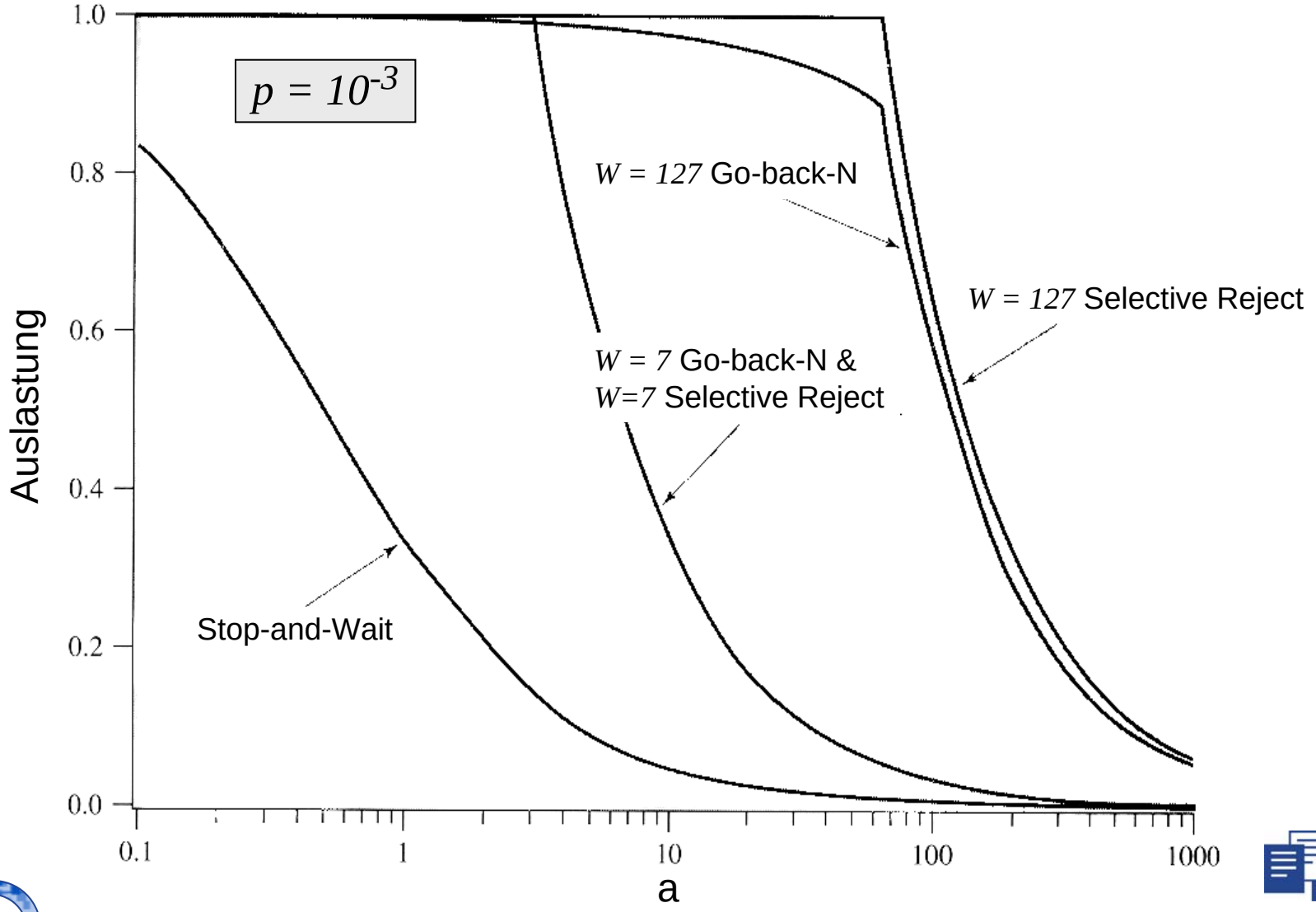
- Herleitung wie bei Stop-and-Wait
  - Es wird immer nur eine Dateneinheit wiederholt

$$n = \frac{1}{1-p}$$

  - Fallunterscheidung wegen Fenster

$$U = \begin{cases} 1-p & W \geq 1+2a \\ \frac{W(1-p)}{1+2a} & W < 1+2a \end{cases}$$

# Vergleichende Bewertung




[Stal06]

# 3.5 Flusskontrolle

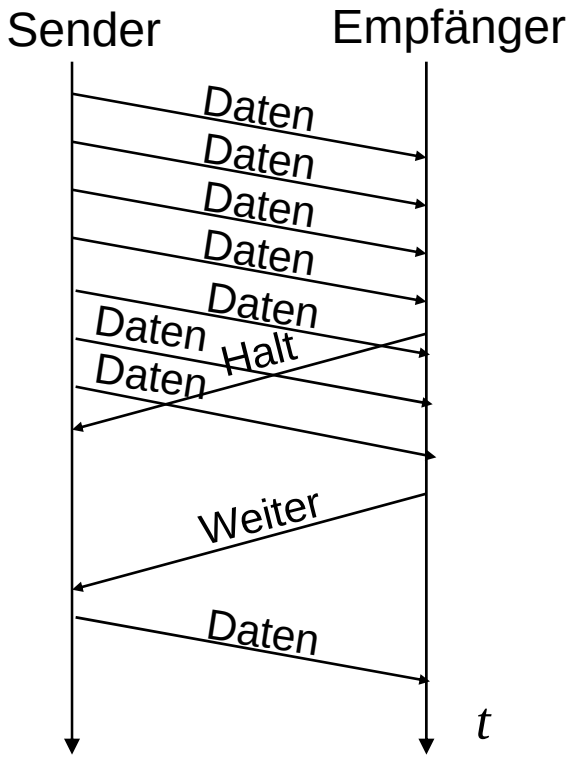
- Problem
  - Wie viele Dateneinheiten dürfen vom Sender hintereinander gesendet werden, ohne dass der Speicher beim Empfänger überläuft (d.h. Verluste von Dateneinheiten entstehen)?
- Anforderungen
  - Einfachheit
  - Möglichst geringe Nutzung von Netzressourcen
  - Fairness
  - Stabilität
- Varianten
  - Closed Loop
    - Rückkopplung, um zu verhindern, dass Empfänger „überschwemmt“ wird ⇨ Quelle adaptiert ihren Datenstrom entsprechend
  - Open Loop
    - Beschreibung des Verkehrs mit anschließender Ressourcenreservierung und Überwachung des eingehenden Verkehrs
    - Hier nicht weiter behandelt

# Closed-Loop Flusskontrolle

- Varianten
  - 1. Generation: Berücksichtigt die Leistungsfähigkeit des Kommunikationspartners
    - Halt-und-Weiter
    - Stop-and-Wait
      - Kombiniert Fehler- und Flusskontrolle
    - Kreditbasierte Flusskontrolle: statisches Fenster
  - 2. Generation: Berücksichtigt auch die Leistungsfähigkeit des Netzes
    - Messungen
      - Explizit vs. Implizit
    - Kontrollvariante
      - Dynamisches Fenster vs. dynamische Rate
    - Kontrollpunkt
      - Ende-zu-Ende vs. Hop-by-hop

# Flusskontrolle mit Halt-/Weiter-Meldungen

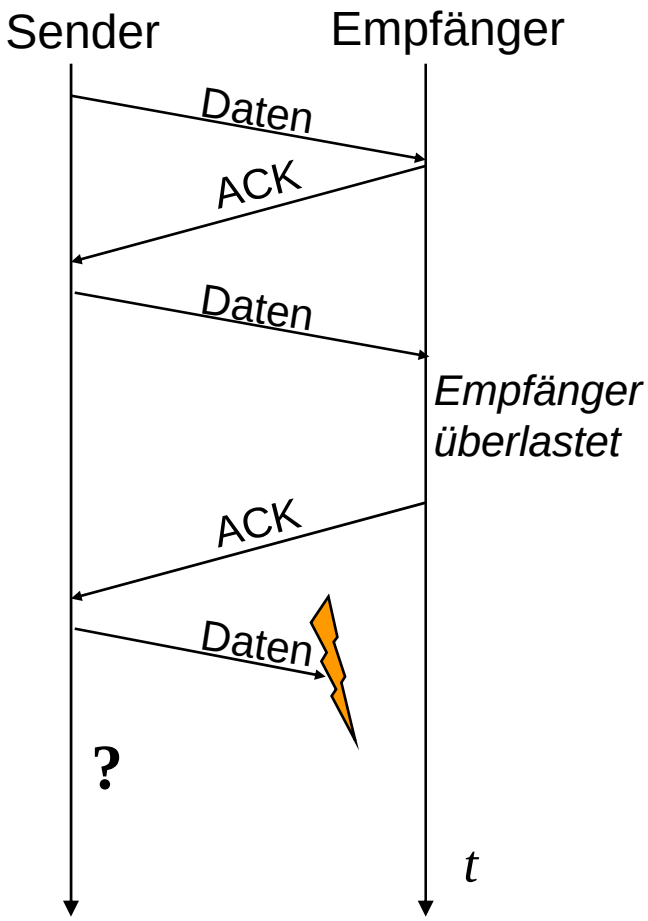
- Sehr einfache Methode
  - Meldungen
    - Halt
    - Weiter
  - Kann der Empfänger nicht mehr Schritt halten, schickt er dem Sender eine Halt-Meldung
  - Ist ein Empfang wieder möglich, sendet der Empfänger die Weiter-Meldung
- Bewertung
  - Nur auf Vollduplex-Leitungen verwendbar
  - Bei hohen Verzögerungen nicht effektiv
  - Probleme bei Verlust der Halt-Meldung
- Beispiel: Protokoll XON/XOFF
  - Für serielle Leitungen
  - Mit ISO 7-Bit-Alphabetzeichen
    - XON ist $DC_1$ (Device Control 1)
    - XOFF ist $DC_3$ (Device Control 3)



# Implizite Flusskontrolle

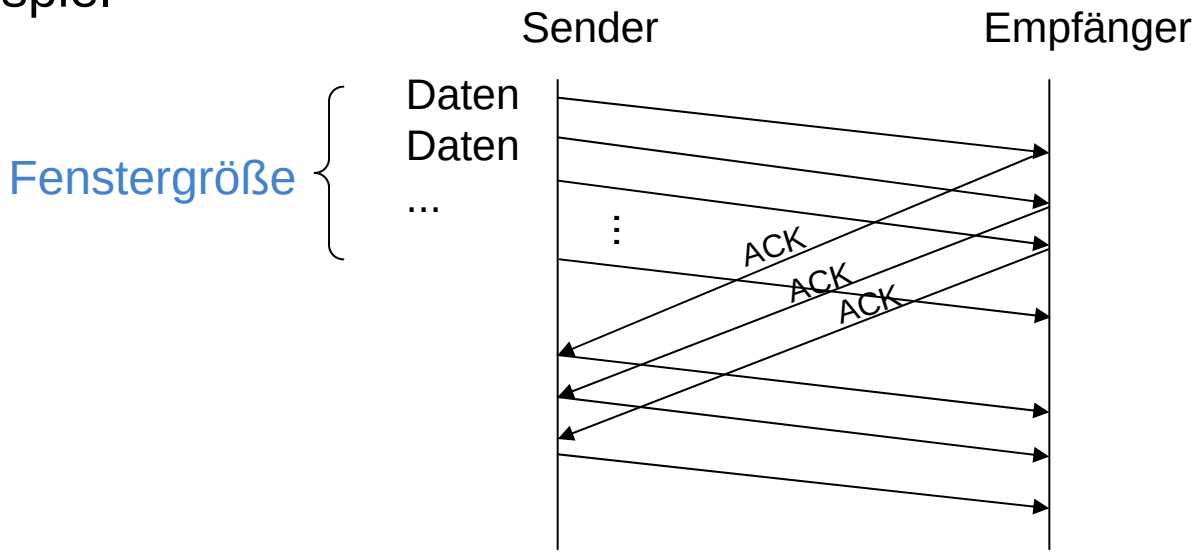
- Funktionsweise
  - Durch Zurückhalten der Quittung (z.B. ACK/NACK) kann der Sender gebremst werden
  - Das bedeutet, dass ein Verfahren zur Fehlererkennung für die Flusskontrolle mitbenutzt wird
  - Beispiel: Stop-and-Wait

- Problem
  - Der Sender kann nicht mehr unterscheiden,
    - ob seine Dateneinheit verloren ging, oder
    - ob der Empfänger die Quittung wegen Überlast zurückgehalten hat



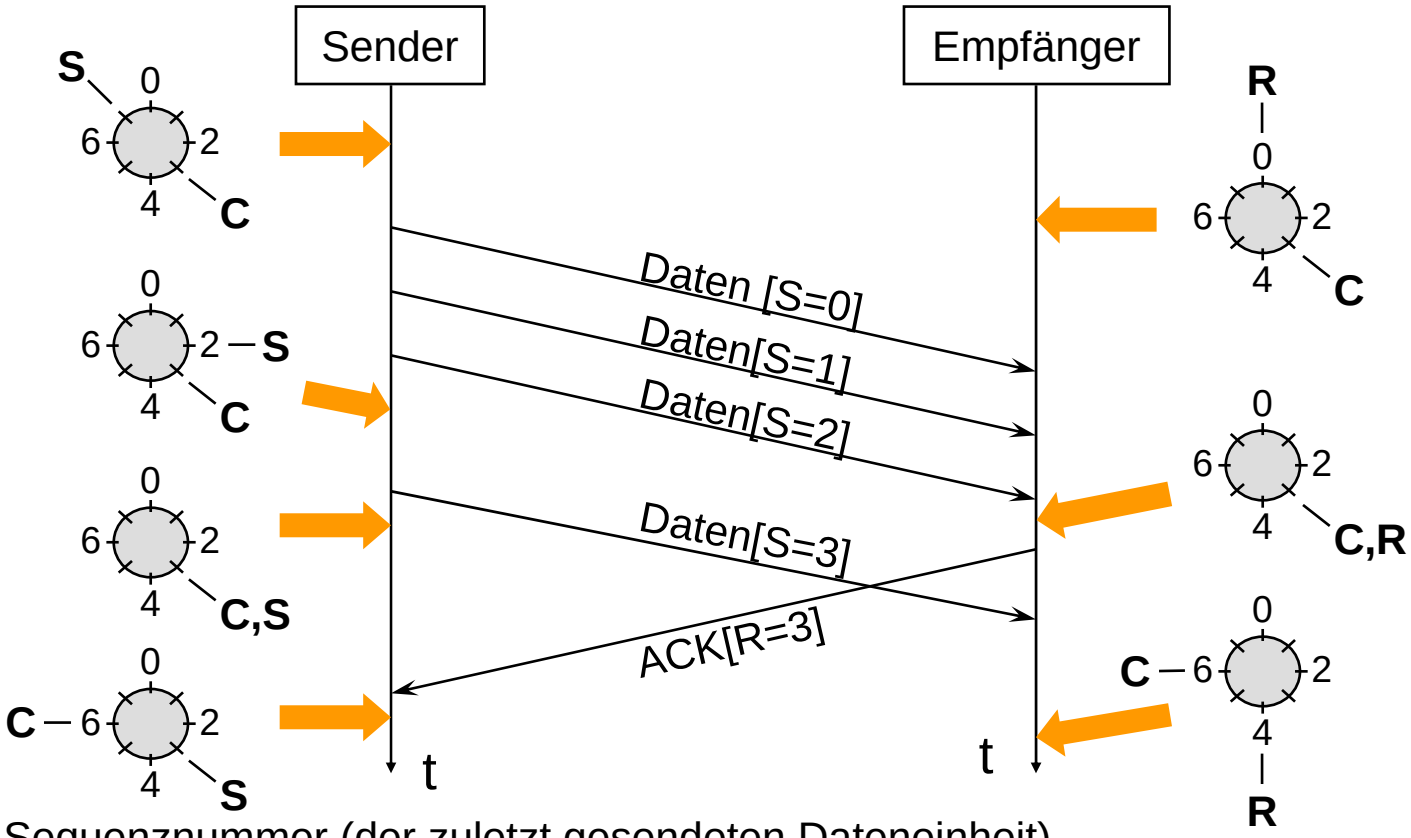
# Kreditbasierte Flusskontrolle

- Prinzip
  - Sender kann bis zu einer maximalen Anzahl Dateneinheiten (bzw. Bytes) senden, ohne eine Quittung zu empfangen
  - Maximale Anzahl der Dateneinheiten repräsentiert die Pufferkapazität des Empfängers und wird als (Sende-)Kredit bezeichnet
  - Oftmals als fortlaufendes Fenster bezeichnet (Englisch: Sliding Window)
    - Fenster wird dann mit jeder empfangenen positiven Quittung weitergeschaltet
    - Empfänger kann meist zusätzlich den Kredit explizit bestimmen (z.B. in TCP)
- Beispiel



# Kreditbasierte Flusskontrolle: Sliding Window

- Beispiel: Fenstermechanismus (Kredit 4) für eine Senderichtung

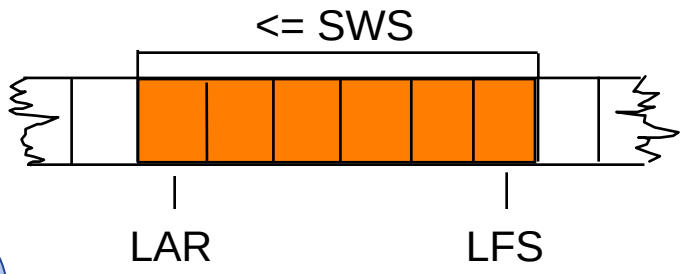



S: Sende-Sequenznummer (der zuletzt gesendeten Dateneinheit)
R: Nächste erwartete Sende-Sequenznummer = Quittierung bis Empfangs-Sequenznummer R-1
C: Oberer Fensterrand (maximal erlaubte Sequenznummer)

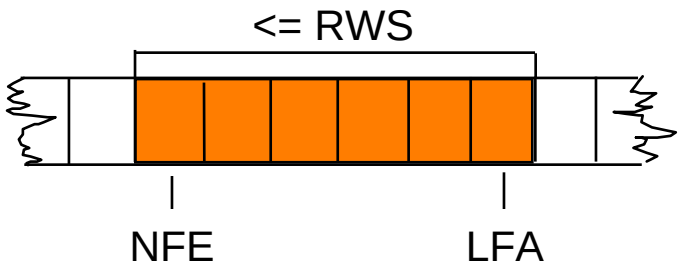
- Nachteil: Kopplung von Fluss- und Fehlerkontrolle

# Flusskontrolle mit Sliding Window

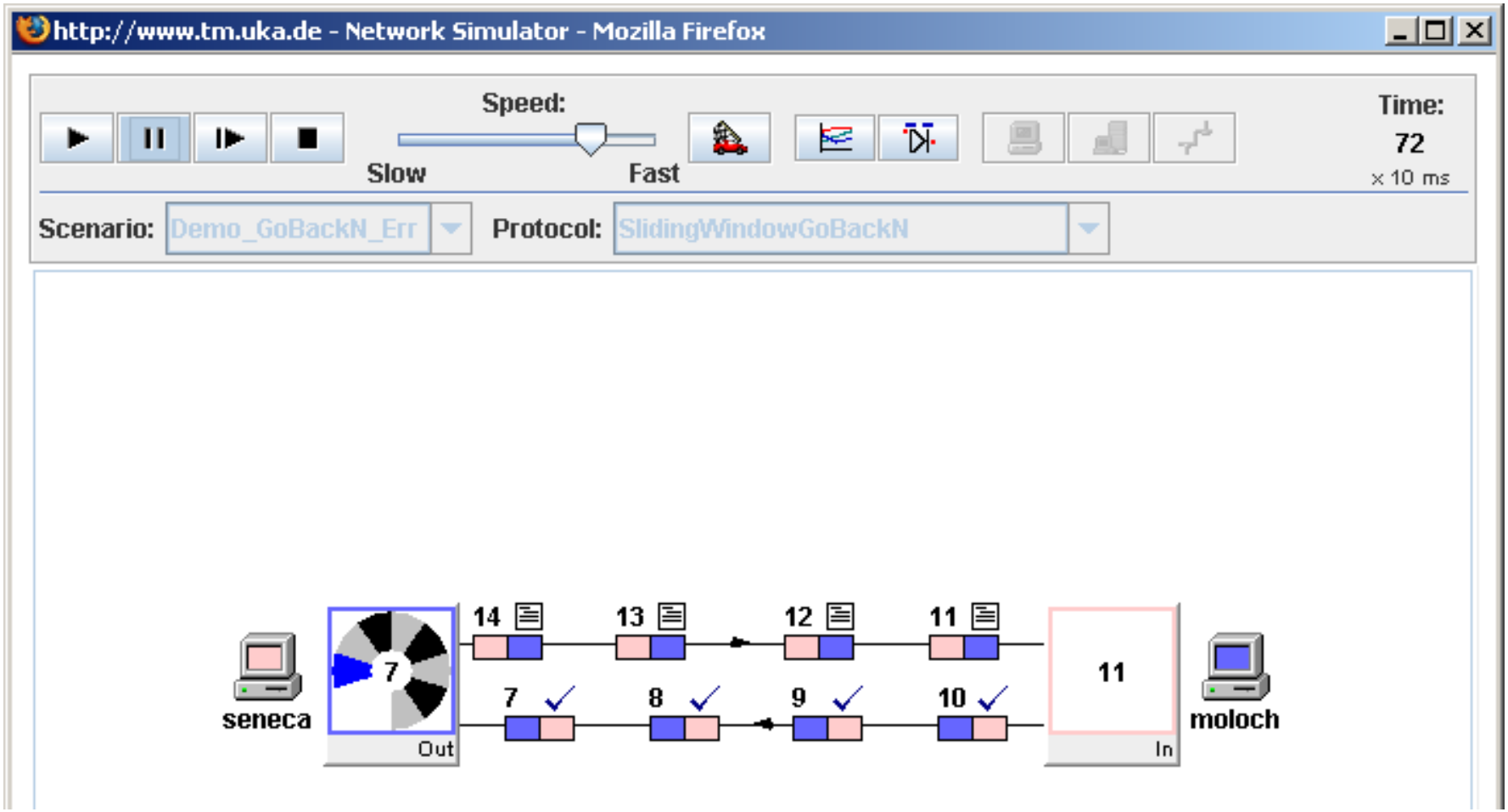
- Sender
  - SWS: Send Window Size (max. Anzahl ausstehender Dateneinheiten bzw. Bytes)
  - LAR: Last ACK Received (Sequenznummer der letzten quittierten Dateneinheit bzw. Bytes)
  - LFS: Last Frame Sent (Sequenznummer der letzten gesendeten Dateneinheit bzw. Bytes)
- Invariante
  - LFS – LAR + 1 <= SWS




- Empfänger
  - RWS: Receiver Window Size (max. Anzahl nicht in Reihenfolge empfangener Dateneinheiten bzw. Bytes)
  - LFA: Last Frame Acceptable (Sequenznummer der letzten empfangbaren Dateneinheit bzw. Bytes)
  - NFE: Next Frame Expected (Sequenznummer der nächsten in Reihenfolge erwarteten Dateneinheit bzw. Bytes)
- Invariante
  - LFA – NFE + 1 <= RWS

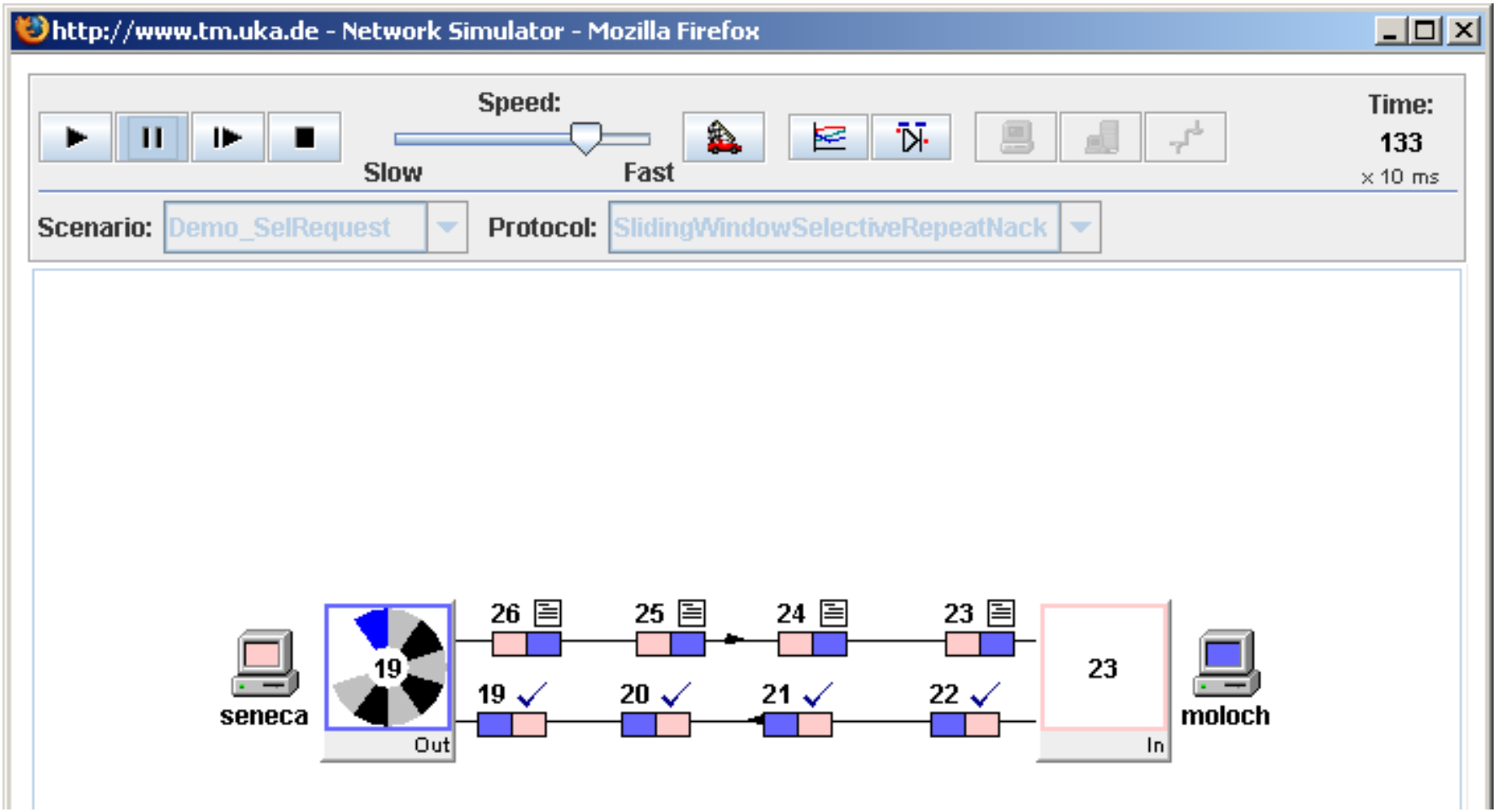


# Java-Applet: Go-back-N und Flusskontrolle




**http://www.tm.uka.de/lehre/aktuell/vl/telematik/sim/simulation.html**

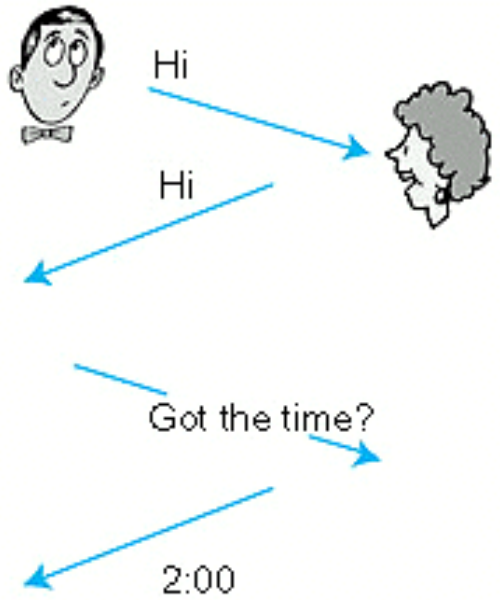
# Java-Applet: Selective Reject und Flusskontrolle




**http://www.tm.uka.de/lehre/aktuell/vl/telematik/sim/simulation.html**

# 3.6 Verbindungen

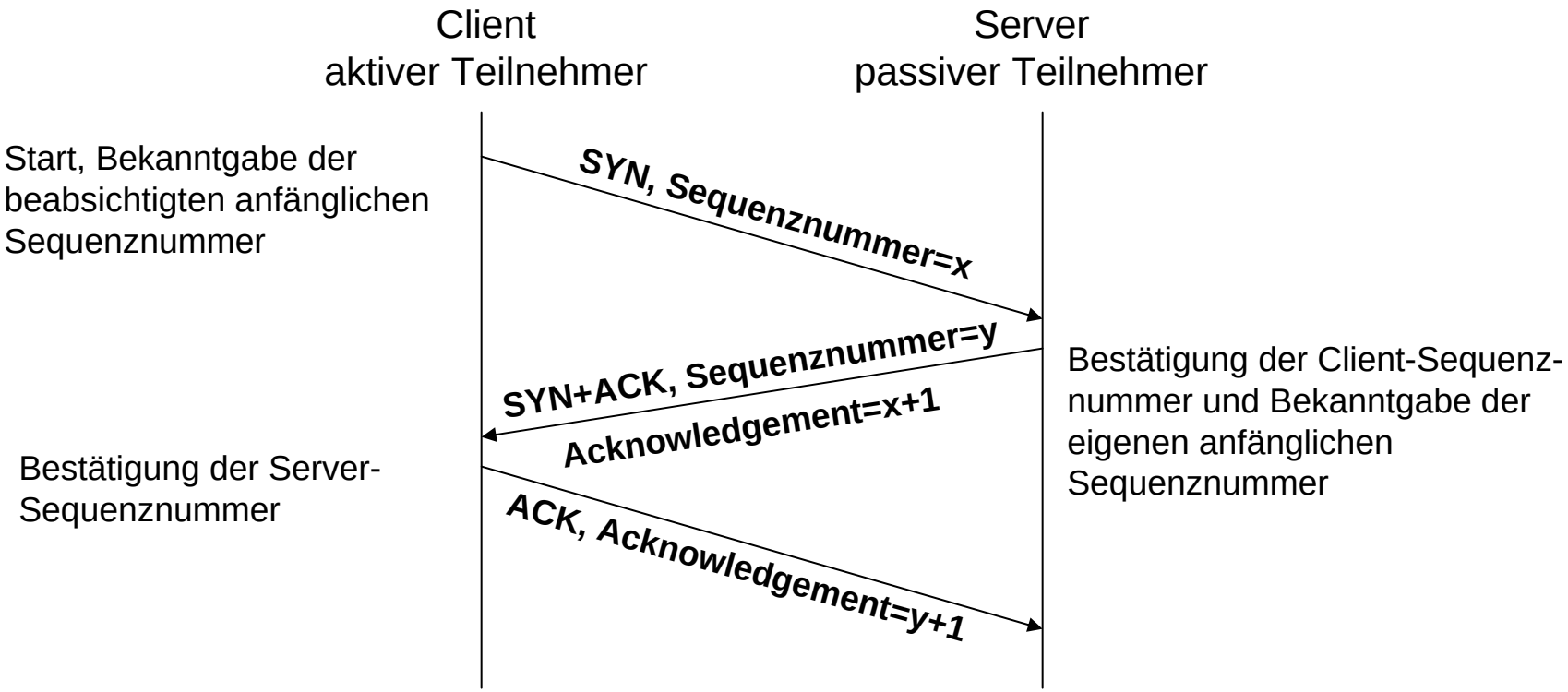
- Verbindungen
  - stellen eine Kommunikationsbeziehung zwischen den Kommunikationspartnern her
  - dienen dem zuverlässigen Austausch von Informationen (Daten)
  - Es wird bei jedem Kommunikationspartner ein sogenannter Verbindungskontext etabliert
    - Sequenznummern, Fenstergröße etc.
- Beispiel



# Verbindungslos vs. Verbindungsorientiert

- Verbindungslose Kommunikation
  - Informationen werden versendet, ohne vorherigen Aufbau einer Verbindung
    - im Beispiel würde also die Begrüßung wegfallen
  - Vorteil: schnelle Datenversendung möglich
  - Nachteil: keine Möglichkeit der Kontrolle, ob der Kommunikationspartner überhaupt zuhört bzw. zuhören kann
- Verbindungsorientierte Kommunikation
  - dem Informationsaustausch geht Aufbau einer Verbindung voraus
  - Vorteil: Aushandlung von Kommunikationsparametern möglich (3-Wege-Handshake)
    - sonst: benötigte Fenstergrößen, verwendeter Code, verwendete Fehlerkontrollmechanismen, Sequenznummern ...
  - Nachteile
    - eigentlicher Datenaustausch verzögert
    - Overhead der Verbindungsetablierung und -verwaltung kann höher sein als der eigentliche Datenaustausch (kurze Daten)

# 3-Wege-Handshake




- Das Feld Acknowledgement identifiziert jeweils die nächste erwartete Sequenznummer

# Zusammenfassung

- Grundlegende Protokollmechanismen
  - Werden uns als Bausteine in vielen Protokollen immer wieder begegnen
- Java Applets: Beispiele für Simulation
  - Animation des Ablaufs
  - Dokumentation des Ablaufs
  - Leistungsbewertung
- Grundlegende Techniken der Leistungsbewertung
  - Viele Vereinfachungen, aber grundlegende Aussagen möglich

# Zusammenfassung

**Fehlerkontrolle**

ARQ-Verfahren:

Stop-and-Wait
Go-Back-N
Selective Reject

Mechanismen:

Sequenznummern
Zeitgeber
Quittungen
Sendewiederholungen

**Flusskontrolle**

Halt/Weiter
Sliding Window

**Verbindungs-management**

3-Wege-Handshake

# Übungen

3.1 Es sind Daten im Umfang von 1 MByte zu übertragen. Die Größe einer Dateneinheit betrage 2000 Byte, die Ausbreitungsverzögerung betrage 20 ms. Vergleichen Sie die erzielbare Auslastung von Stop-and-Wait mit Go-Back-N.

3.2 Wann versendet der Empfänger jeweils Quittungen?

3.3 Welches Ziel hat die Flusskontrolle?

3.4 Konstruieren Sie ein Schieberegister, dass Sie für die Division durch $x^7 + x^5 + x^4 + x + 1$ verwenden können.

3.5 Berechnen Sie den CRC für die Dateneinheit 11101100011010 1 mit dem Generatorpolynom 110011

3.6 Knoten A sendet mit Go-Back-N und Sliding-Window Daten an Knoten B. Die Fenstergröße betrage 4 Bit. Zeichnen sie die Fenster auf beiden Seiten

- Bevor A anfängt Daten zu senden.
- Nachdem A die Dateneinheiten 0, 1 und 2 gesendet und B 0 und 1 quittiert hat.
- Nachdem A die Dateneinheiten 3, 4 und 5 gesendet hat und B 4 quittiert hat.

3.7 Welche Fehlerarten kennen Sie?

3.8 Eine Störung von 10 ms führt bei einer Datenrate von 100 Mbit zu wieviel gestörten Bits?

# Literatur


[Benv05] Ch. Benvenuti; Understanding Linux Network Internals; O'Reilly, 2005

[Hals05] F. Halsall; Computer Networking and the Internet; Addison-Wesley, 2005

- Kapitel 1.4, Anhang C

[Holz91] G. J. Holzmann; Design and Validation of Computer Protocols; Prentice Hall, 1991
Auch online unter: http://spinroot.com/spin/Doc/Book91.html

[KoBu94] W. P. Kowalk, M. Burke; Rechnernetze; Teubner Verlag, 1994

[KuRo07] J. F. Kurose, K. W. Ross: Computer Networking: A Top-Down Approach, Addison-Wesley, 2007

[Stal06] W. Stallings; Data and Computer Communications, Prentice Hall, 2006

- Kapitel 7

# Kommunikation und Datenhaltung

## 4. Geschichtete Architekturen




**Prof. Dr. Martina Zitterbart**
Dipl.-Inform. Martin Röhricht
[zit | roehricht]@tm.uka.de



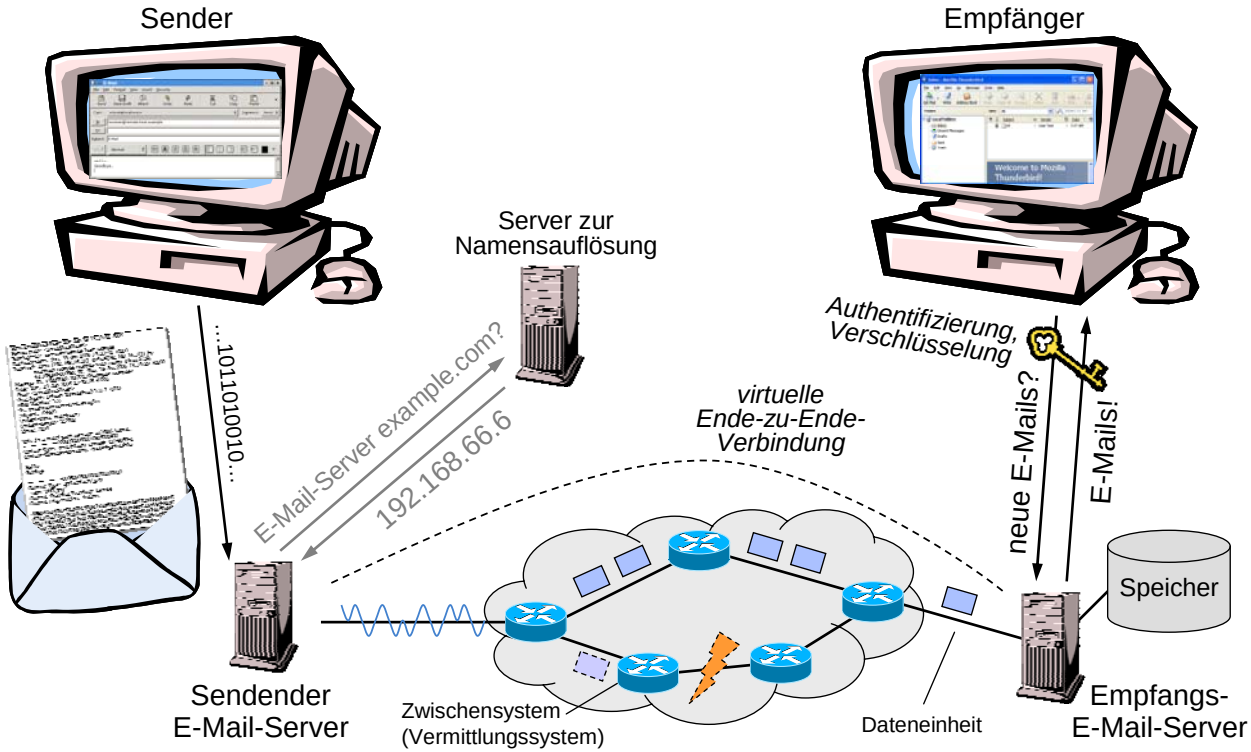
# Kapitelübersicht

1. Einführung
2. Physikalische Grundlagen
3. Protokollmechanismen
4. Geschichtete Architekturen
5. Sicherungsschicht: HDLC
6. Beschreibungsmethoden
7. Sicherungsschicht: Lokale Netze
8. Netzkopplung und Vermittlung
9. Die Transportschicht
10. Anwendungssysteme
11. Middleware

4.1 Motivation
4.2 Grundmodell der Kommunikation
4.3 Dienst
4.4 Geschichtete Architekturen
4.5 ISO/OSI-Basisreferenzmodell
4.6 ISO/OSI versus Internet
4.7 Vorlesungsüberblick

# 4.1 Motivation

- Kommunikationssysteme müssen vielfältige Aufgaben erledigen
  - Austausch von Daten zwischen benachbarten Geräten
  - Zielgerichtetes Weiterleiten der Daten zum Ziel
  - …



# Motivation

- Zielsetzung
  - Standardisierung, um Kommunikation in heterogenen Netzen zu ermöglichen
- Beobachtung Ende der 70er Jahre
  - Etablierung Herstellerspezifischer Netze
    - Z.B. IBM, DEC
- ISO (International Organisation for Standardization)
  - 1977 wurde Komitee „Open Systems Interconnection“ gegründet
    - Definition einer Architektur, die als Rahmenwerk zur Standardisierung von Protokollen dienen kann
    - Als „OSI Reference Model“ bezeichnet
    - Arbeit in nur 18 Monaten beendet
    - Heute noch grundlegendes Modell der Kommunikation

# Historisches zum Referenzmodell

- 1978
  - ISO schlägt Konzipierung eines Rahmenwerks zur Koordinierung der Verbindung heterogener Systeme vor
- 1983
  - Referenzmodell wird internationaler Standard
  - → Referenzmodell wurde vor den Protokollen entwickelt
    - Problem: Modell und Anforderungen klaffen etwas auseinander
- Fundamentale Konzepte
  - Dienste
  - Schnittstellen
  - Protokolle
- Kommentare
  - Recht stark von Telekommunikationssicht geprägt
    - Dort heute teilweise so auch noch im Einsatz (z.B. Signalisierung)
  - „Internet-Welt“ lebt mit einfacherem Model

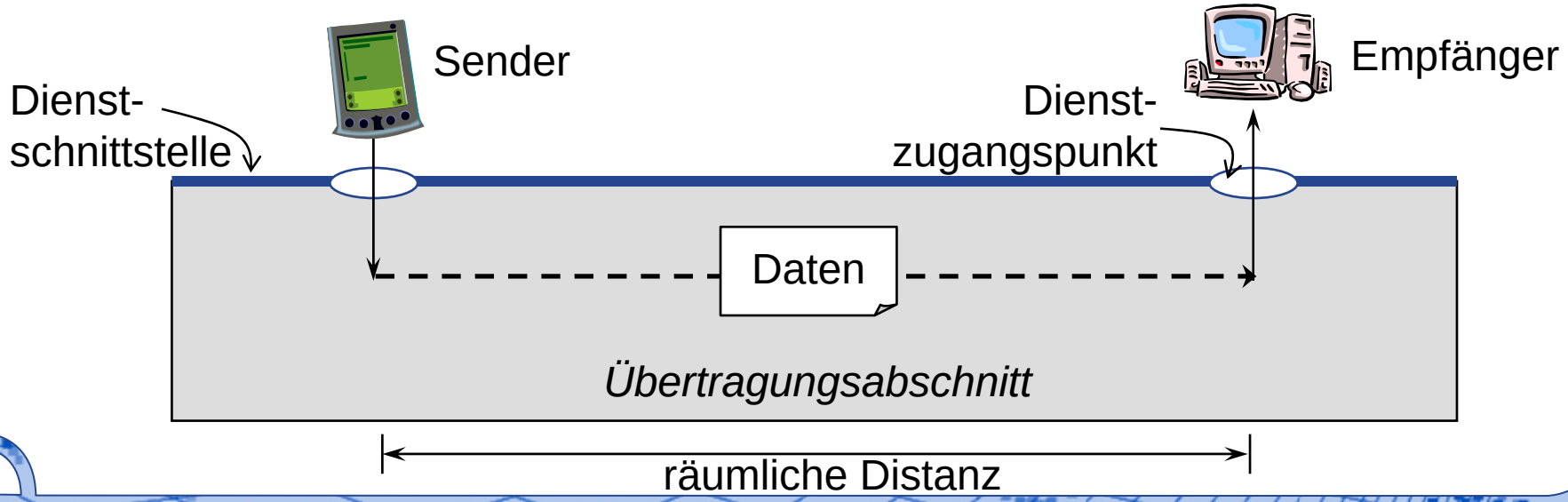


[Zimm80]

# 4.2 Grundmodell der Kommunikation

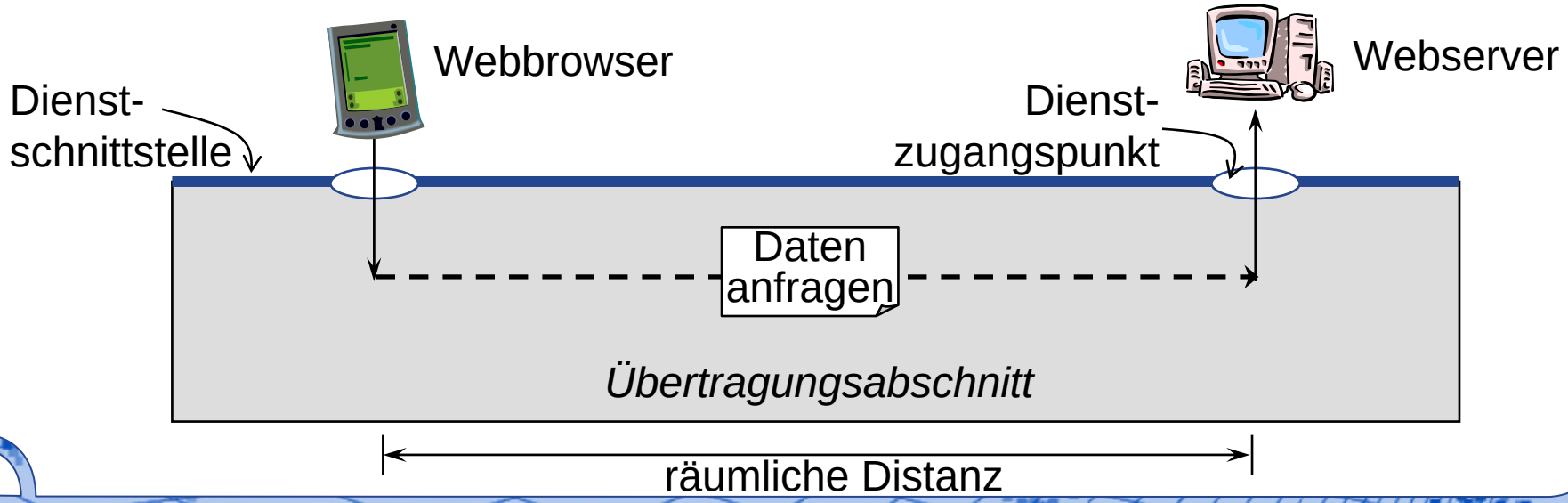
- Grundvorgang
  - Daten überbrücken räumliche Distanz zwischen Sender und Empfänger
- Grundkomponenten
  - Sender und Empfänger
  - Übertragungsabschnitt
    - Bereich zwischen einem Sender und einem Empfänger zur Überbrückung der räumlichen Distanz
  - Dienstschnittstelle (Service Interface)
    - Beschreibt, wie die Übergabe von Informationen zwischen den Schichten erfolgt
  - Dienstzugangspunkte (Service Access Points – SAPs)
    - Bieten Zugang zu Kommunikationsdienst

[Haaß97]



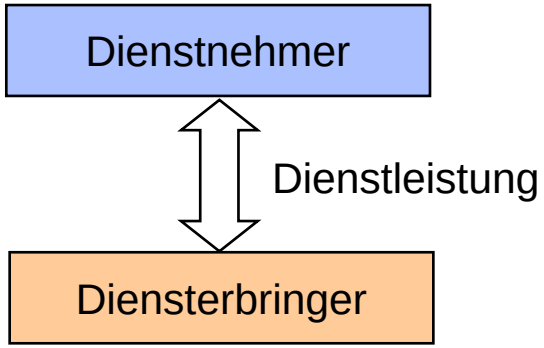
# Beispiel Sockets

- Kommunikationsdienst
  - Internetverkehr
- Diensterbringer
  - Internet-Anbieter (z.B. T-Online)
- Dienstschnittstelle
  - Sockets Programmierschnittstelle
- Dienstzugangspunkt
  - TCP-Socket oder UDP-Socket
- Teilnehmer
  - Client (z.B. Webbrowser) und Server (z.B. Webserver)
- Medium
  - Internet: verschiedene Transportmedien wie WLAN, Glasfaser, Kupferkabel

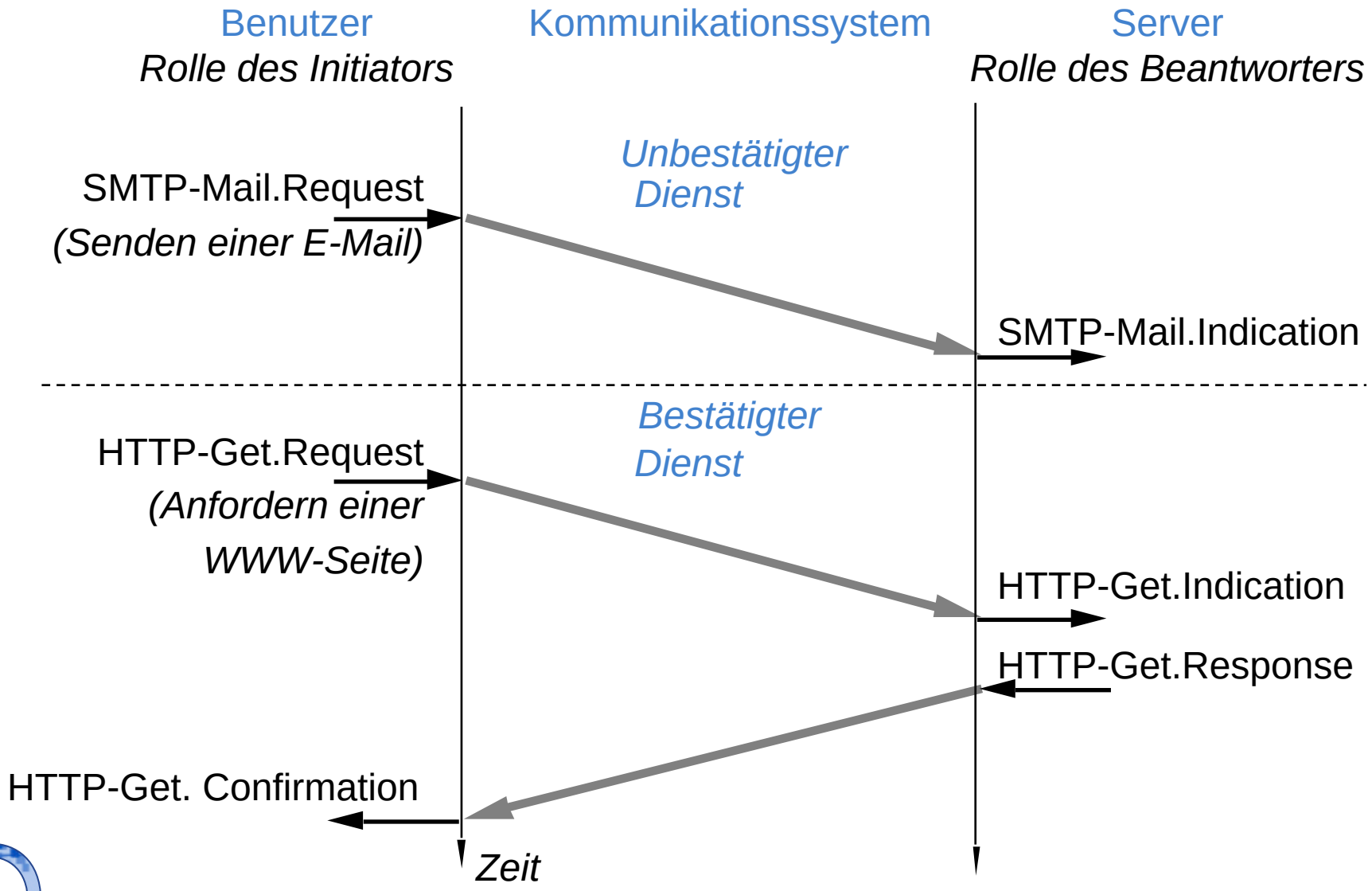


# 4.3 Dienst

- Bündelung zusammengehöriger Funktionen
- Dienstfunktionalität
  - Einzelne Teile eines Dienstes können unabhängig voneinander in Anspruch genommen werden
- Dienstprimitiv
  - Einzelvorgänge einer Dienstfunktion
- Dienstprozedur
  - Zusammensetzung einer Dienstfunktion aus ihren Dienstprimitiven
  - Bemerkung: Begriff im Umfeld der Kommunikationssysteme eher unüblich
- Dienstehierarchie
  - Dienst baut auf anderen Diensten auf
    - Diensterbringer
      - Bieten einen Dienst an
    - Dienstnehmer
      - Nehmen einen Dienst in Anspruch



# Grundformen von Diensten



# Dienst und Verbindung

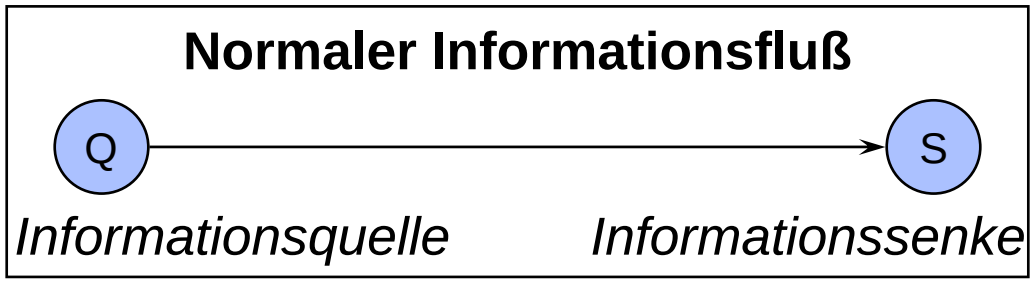
- Verbindungsloser Dienst (Datagramm-Dienst)
  - Dienstleistung ohne Verbindungskontext
    - Dienstleistung wird vollständig durch Anforderung und die mitgegebenen Kontrollparameter definiert
  - „Weg“ zwischen den Kommunikationspartnern muss mit jedem neuen Dienstaufruf auch neu ermittelt werden

- Verbindungsorientierter Dienst
  - Dienstleistung im Kontext, d.h. Diensterbringung abhängig von früher erbrachten Dienstleistungen
    - Kontextinformation enthält z.B. Information über Größe des Flusskontrollfensters, aktuelle Sende- und Empfangssequenznummern
  - Vorteil
    - Weniger Kontrollinformation muss übertragen werden
  - Nachteil
    - Aufwand für Herstellung und Verwaltung des Kontextes, z.B. für Verbindungsaufbau

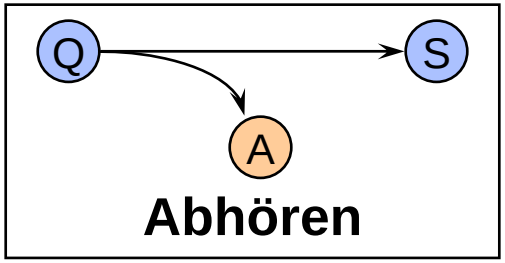
# Dienstqualität

- Nicht nur Funktionalität, sondern auch Qualitätsparameter sind maßgeblich für Kommunikationsdienste
  - Dienstqualität: In welcher Qualität wird die Funktionalität erbracht?
- Qualitätsparameter lassen sich grob fünf Hauptaspekten zuordnen
  - Angemessenheit
    - Eignung des Dienstes für das vorgesehene Einsatzgebiet
  - Technische Leistung
    - z.B. Laufzeit, Antwortzeit, Sende- und Empfangsrate, Durchsatz
  - Kosten
    - Investitions- und Betriebskosten zur Erbringung des Dienstes
  - Zuverlässigkeit
    - Verhinderung von störenden Einflüssen
  - Schutz
    - Verhinderung von bewussten Eingriffen

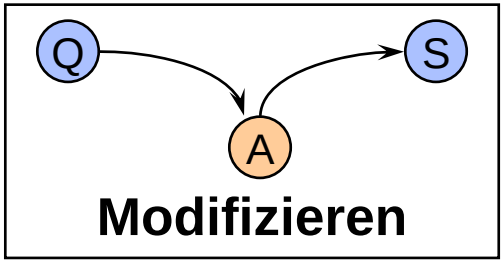
# Qualität: Sicherheitsgefahren und Schutzmaßnahmen

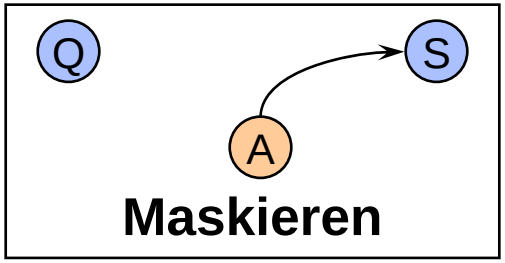


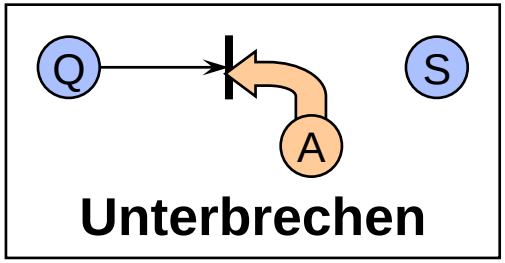

Passiv:




Aktiv:



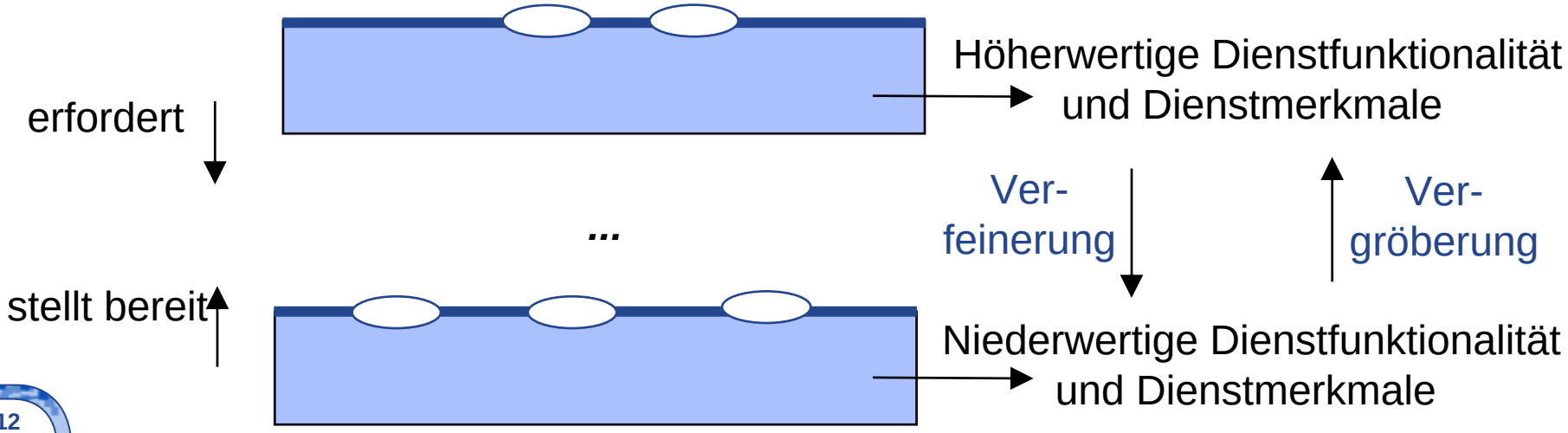



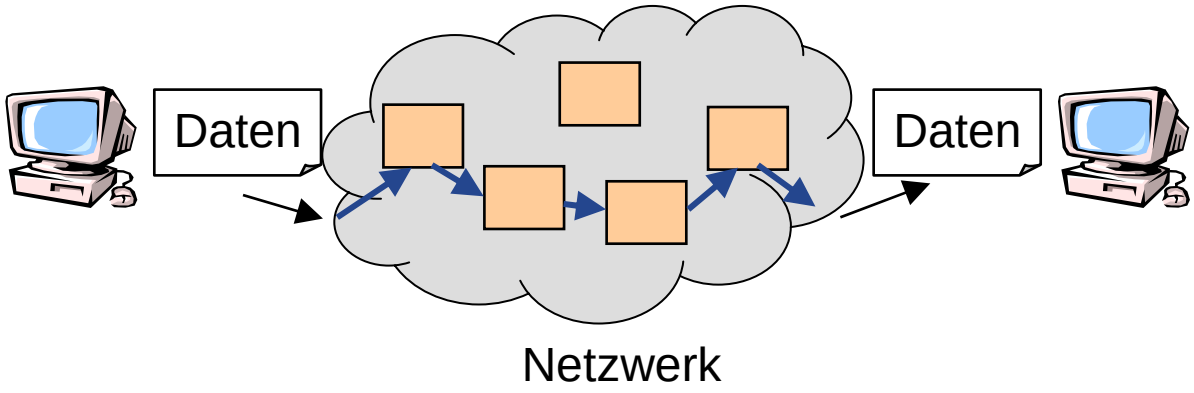
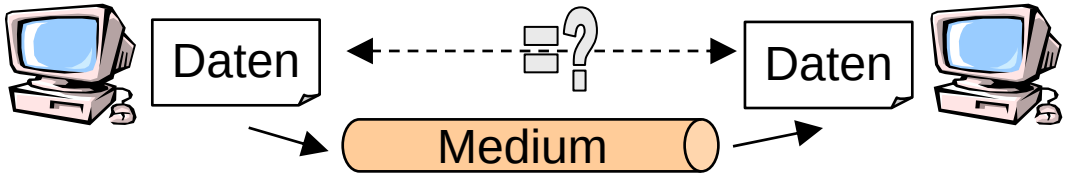

- Schutzmaßnahmen
  - Verschlüsselung (kryptographische Codes)
  - Schaffung vertrauenswürdiger Systeme
    - Authentisierung, Autorisierung

# 4.4 Geschichtete Architekturen

- Verfeinerung des Grundmodells
- Beobachtung
  - Kommunikationssysteme sind komplexe Systeme, zu deren Beherrschung eine über das Grundmodell hinausgehende Gliederung sinnvoll erscheint
- Prinzip
  - Einführung einer Abstraktion auf der Basis von Schichten
    - Bereitstellen von Diensten an der Schnittstelle nach oben
    - Nutzung von Diensten an der Schnittstelle nach unten



# Vereinfachte geschichtete Architektur

| Aufgabe | Aufgabenstellung (Szenario) |
|---|---|
| **Ende-zu-Ende-Übertragung** |   |
| **Gerät-zu-Gerät-Übertragung** |   |
| **Bitübertragung** |   |

# Vereinfachte geschichtete Architektur

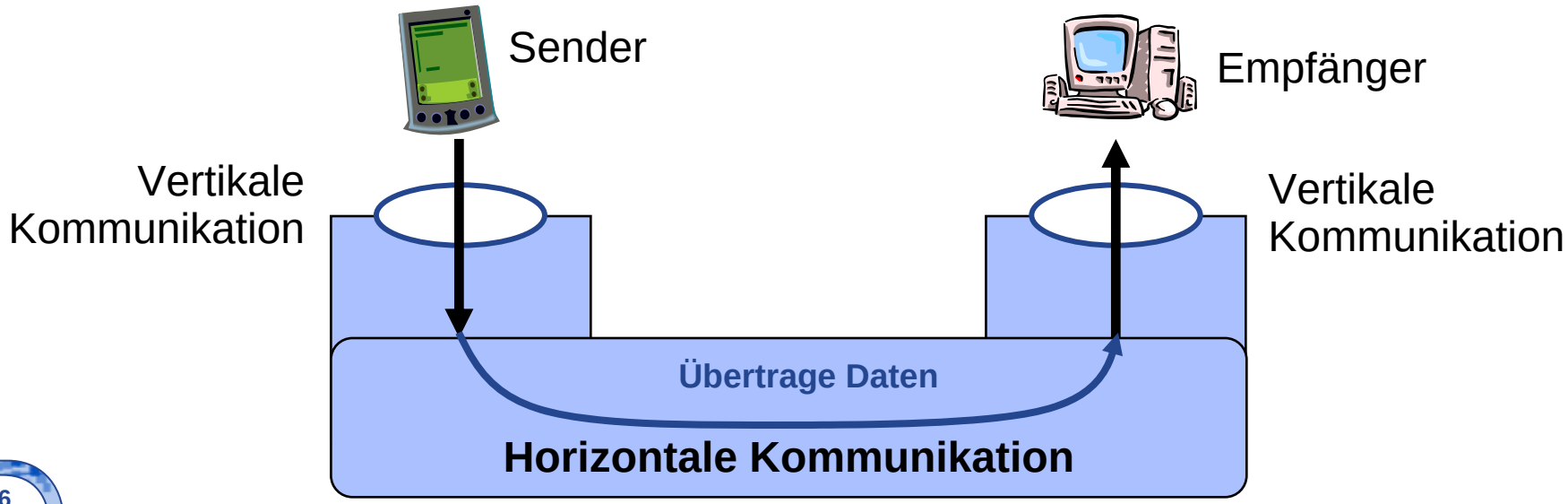
| Aufgabe | Dienstfunktionalität |
|---|---|
| **Ende-zu-Ende-Übertragung** | Verknüpfen der Übertragungsabschnitte zu einer die Endgeräte verbindenden Strecke |
| **Gerät-zu-Gerät-Übertragung** | Übertragung des Bitstroms zwischen physikalisch benachbarten Geräten (Übertragungsabschnitt) |
| **Bitübertragung** | Übertragung des Bitstroms in Form eines Signalverlaufs |

# Protokolle und Protokollinstanzen

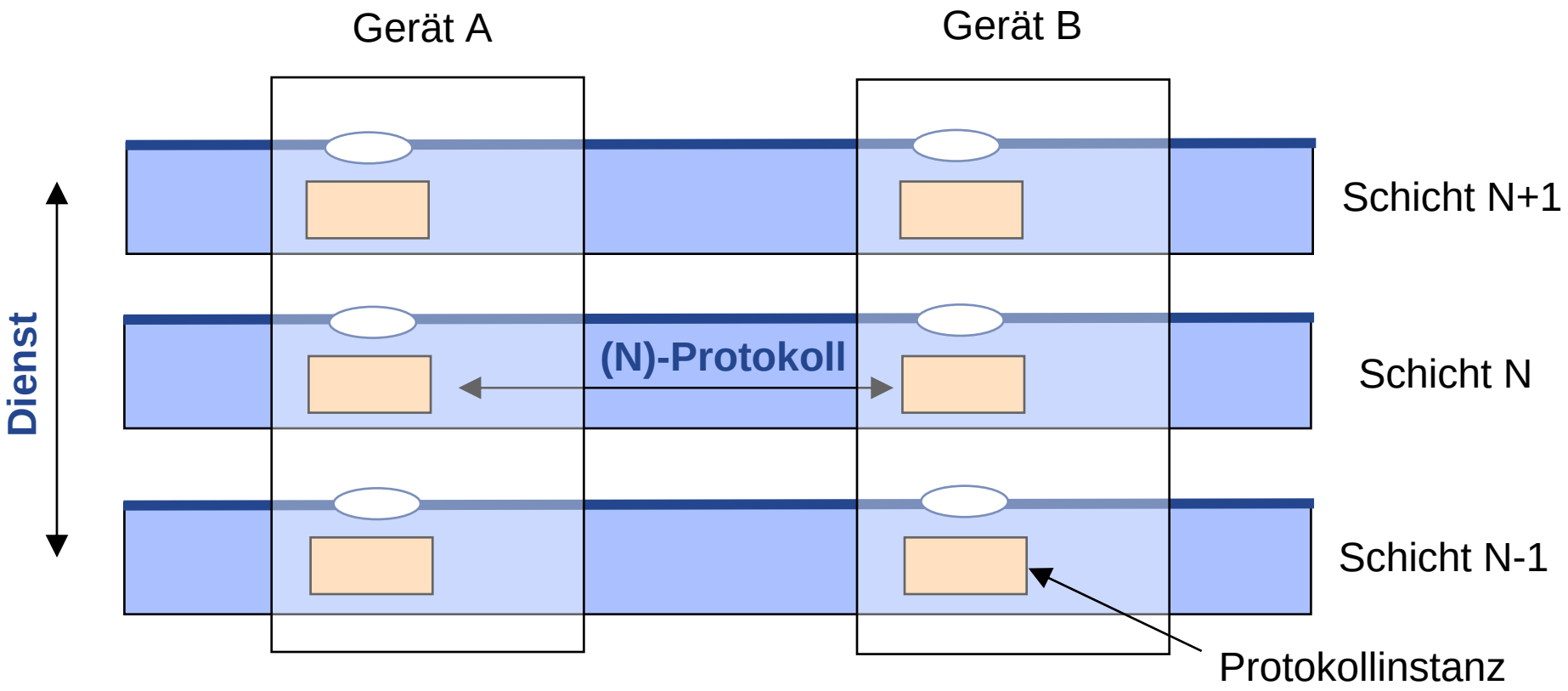
- Dienst
  - Innerhalb einer Schicht erbracht
    - Durch Zusammenwirken der Protokollinstanzen gemäß standardisiertem Protokoll
      - Horizontale Kommunikation
  - Schicht zieht sich über gesamtes Kommunikationssystem hinweg
    - Schicht ist nicht auf ein einzelnes Gerät limitiert
- Protokoll
  - Regeln und Formate für den Austausch von Daten zwischen Protokollinstanzen einer Schicht
    - Protokollinstanzen realisieren Protokolle in den Schichten
      - Ablauf innerhalb einer Schicht wird durch Protokoll geregelt
      - Protokollinstanzen können sich auf dem gleichen oder auf unterschiedlichen Geräten befinden

# Horizontale und vertikale Kommunikation

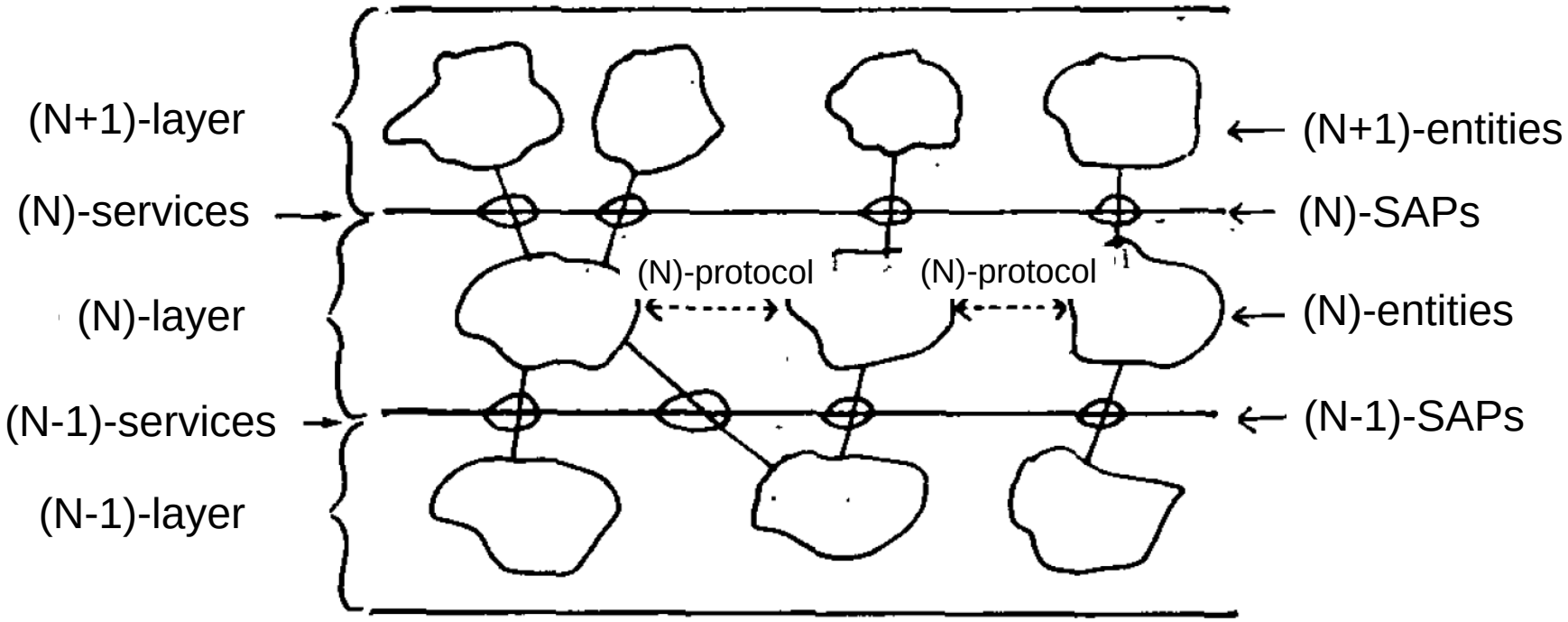
- Horizontale Kommunikation
  - Protokollinstanzen einer Schicht tauschen Daten untereinander aus um den geforderten Dienst zu erbringen
- Vertikale Kommunikation
  - Protokollinstanzen benachbarter Schichten kooperieren miteinander
  - Protokollinstanz der Schicht N greift auf Dienste der Protokollinstanz in Schicht N-1 zu
    - Dieser Austausch erfolgt in der Regel innerhalb eines Geräts (z.B. Notebook)



# Protokolle und Protokollinstanzen



# Instanzen, Dienstzugangspunkte und Protokolle




- Zur Erläuterung aus [Zimm80]
  - „… the (N) entities add value to the (N-1) services they get from the (N-1) layer and offer this value-added service, i.e., the (N) service to the (N+1) entities.“

# 4.5 ISO/OSI-Basisreferenzmodell

- Ziel
  - Offene, herstellerunabhängige Standards für geschichtete Architekturen, Protokolle und Dienste erforderlich
    - „Kommunikationspartner müssen gleiche Sprache sprechen“
  - Bereitstellung eines international standardisierten Modells als Rahmenwerk
- ISO/OSI-Basisreferenzmodell
  - Ist ein logisches Modell
  - Dient der gedanklichen Strukturierung von Kommunikationssystemen
  - Keine Strukturierung für eine Implementierung!
    - Beispielsweise Effizienz
      - Daten werden beim Austausch zwischen Schichten möglichst nicht im Speicher kopiert

# Wieviele Schichten?

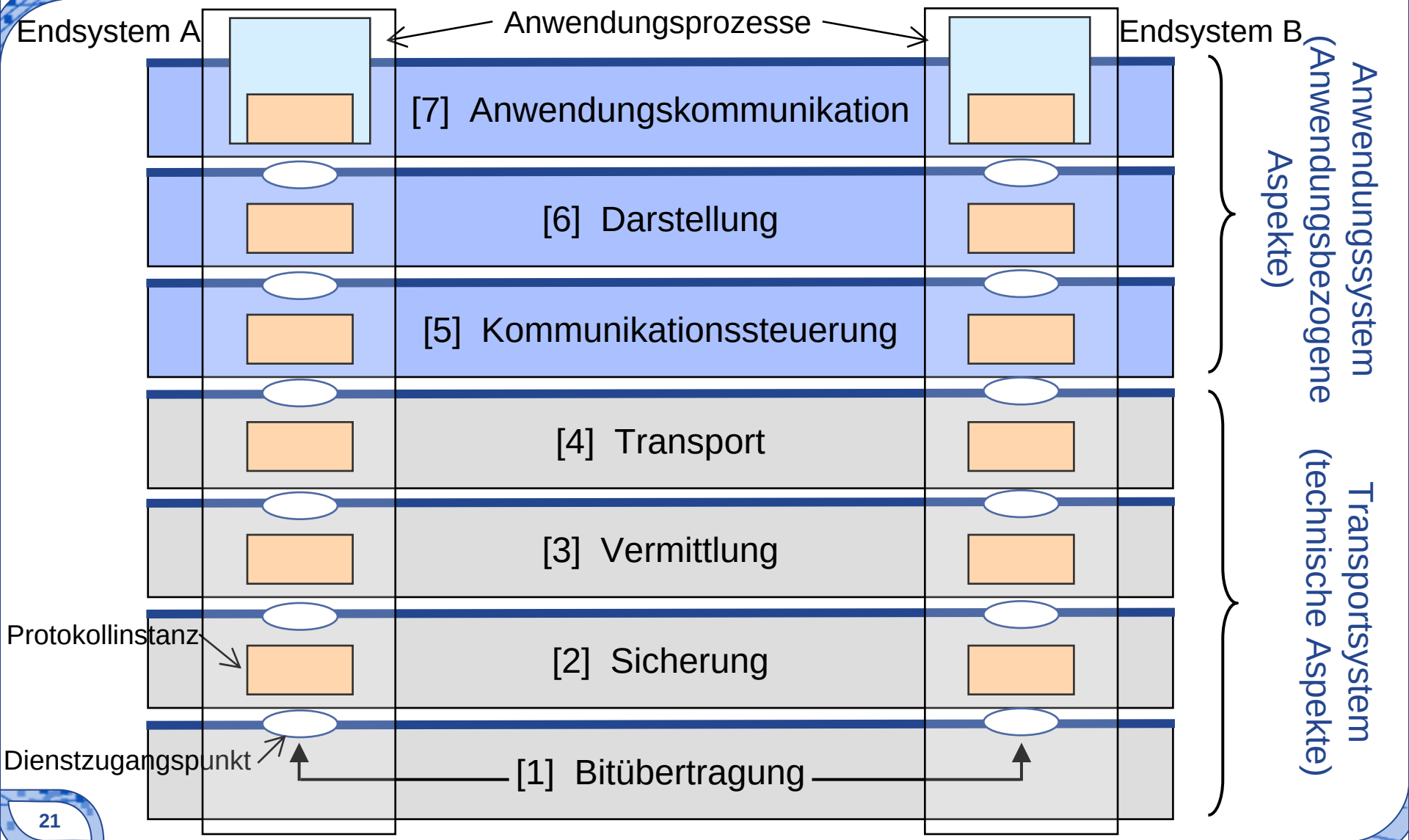
- Grundlegende Prinzipien



[Zimm80]

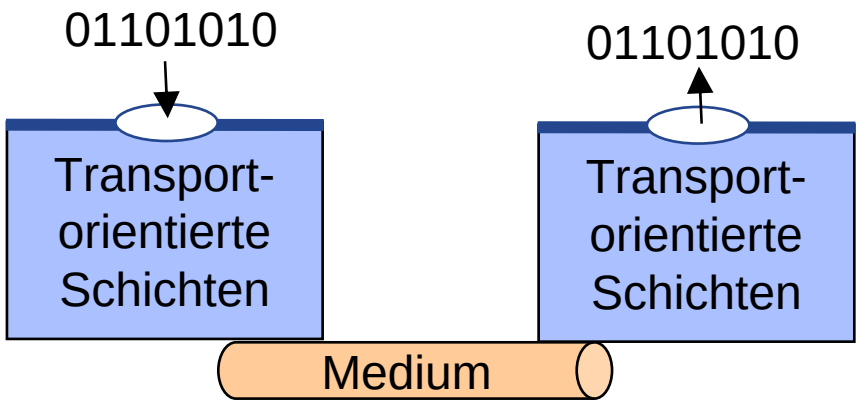
  - Nicht zu viele Schichten
  - Dienstbeschreibung soll klein sein und Anzahl der Interaktionen zwischen Schichten gering
  - Unterschiedliche Schichten für verschiedene Aufgaben
  - Ähnliche Funktionen in der selben Schicht sammeln
  - Schicht muss technologischen Fortschritten entsprechend anpassbar sein, ohne Redesign von benachbarten Schichten zu erfordern
  - Unterschiedliche Abstraktion bei der Behandlung von Daten
  - Funktionen oder Protokolle einer Schicht müssen änderbar sein, ohne Änderungen in anderen Schichten zu erfordern
  - Schicht hat nur Schnittstellen zu ihren benachbarten Schichten
  - Unterschichten innerhalb einer Schicht sind möglich
  - Unterschichten können übergangen werden, Schichten nicht

# ISO/OSI-Basisreferenzmodell



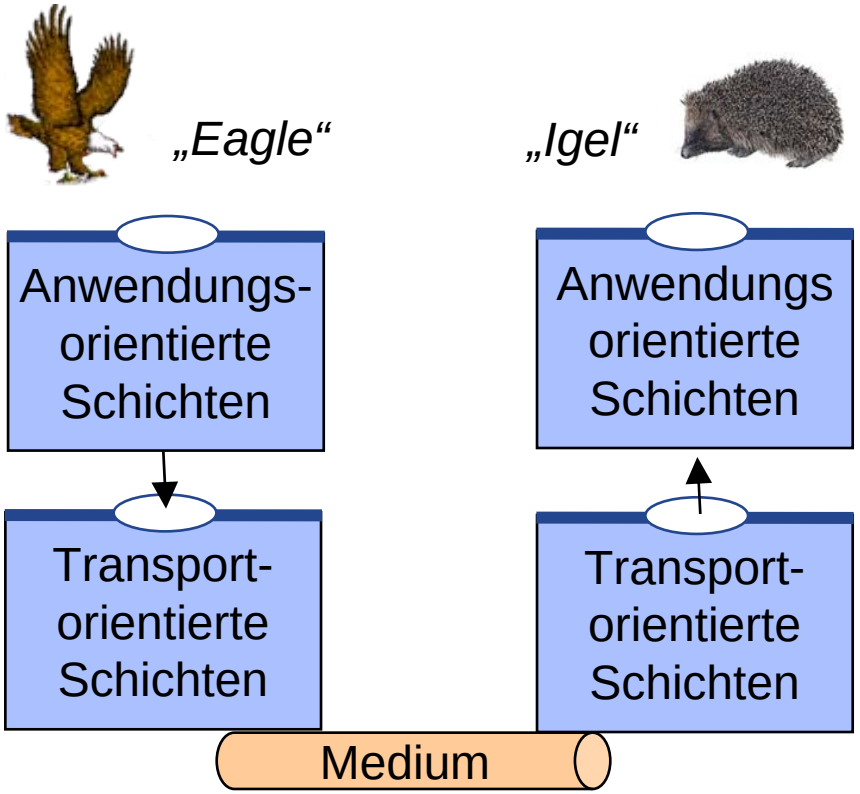
# Aufgabenbereiche

- Transportsystem
  - Transportorientierte Schichten (1-4)
  - Transparente Ende-zu-Ende-Übertragung von Daten
    - Inhalt (Semantik) der Daten transparent
    - Kein Bezug zu Kooperationsbeziehung der Dienstnehmer
    - Elementare Datenübertragung
      - Nur den Bedürfnissen des Datenaustauschs und des bereitzustellenden Dienstes unterstellt



# Aufgabenbereiche

- Anwendungssystem
  - Anwendungsorientierte Schichten (5-7)
  - Anwendungsbezogene Aspekte
    - Semantik der Daten ist wichtig
      - Informationsdarstellung und -austausch
    - Kooperation der Teilnehmer unter formalen Gesichtspunkten berücksichtigt, wie z.B.
      - Steuerung des Ablaufs
      - Kompensation von Fehlverhalten durch verteilte Transaktionen



# Die transportorientierten Schichten

- Schicht 1: Bitübertragungsschicht (Physical Layer)
  - Übertragung von Bits
  - Verwendung von Leitungscodes etc.
  - Keine Pufferung
  - Bietet keine Zuverlässigkeit an
  - Problem: mögliche Störungen der Übertragung
  - Ziel: feste Übertragungsqualität
- Schicht 2: Sicherungsschicht (Data Link Layer)
  - Erweitert nachrichtentechnischen Kanal zum abstrakten „gesicherten Kanal“
    - Erkennung und Behebung von Fehlern der Bitübertragungsschicht möglich
  - Übertragung von Daten zwischen Dienstnehmern direkt verbundener Geräte
  - Gliederung eines Bitstroms in Dateneinheiten
  - Pufferung sowohl beim Sender als auch beim Empfänger

# Die transportorientierten Schichten

- Schicht 3: Vermittlungsschicht (Network Layer)
  - Verknüpft Übertragungsabschnitte zu Ende-zu-Ende-Strecken
  - Wegewahl im Kommunikationssystem
    - Finden geeigneter Wege für die Ende-zu-Ende-Datenübertragung
  - Adressierung der Geräte
  - Multiplexen
- Schicht 4: Transportschicht (Transport Layer)
  - Übertragung von Daten zwischen Anwendungen
  - Abstrahiert von Diensten der Vermittlungsschicht
    - Dem Dienstnehmer bleiben sämtliche Aspekte der Datenübertragung verborgen
  - Fehlererkennung und -behebung
  - Pufferung
  - Adressierung von Transportdienstbenutzern
  - Multiplexen

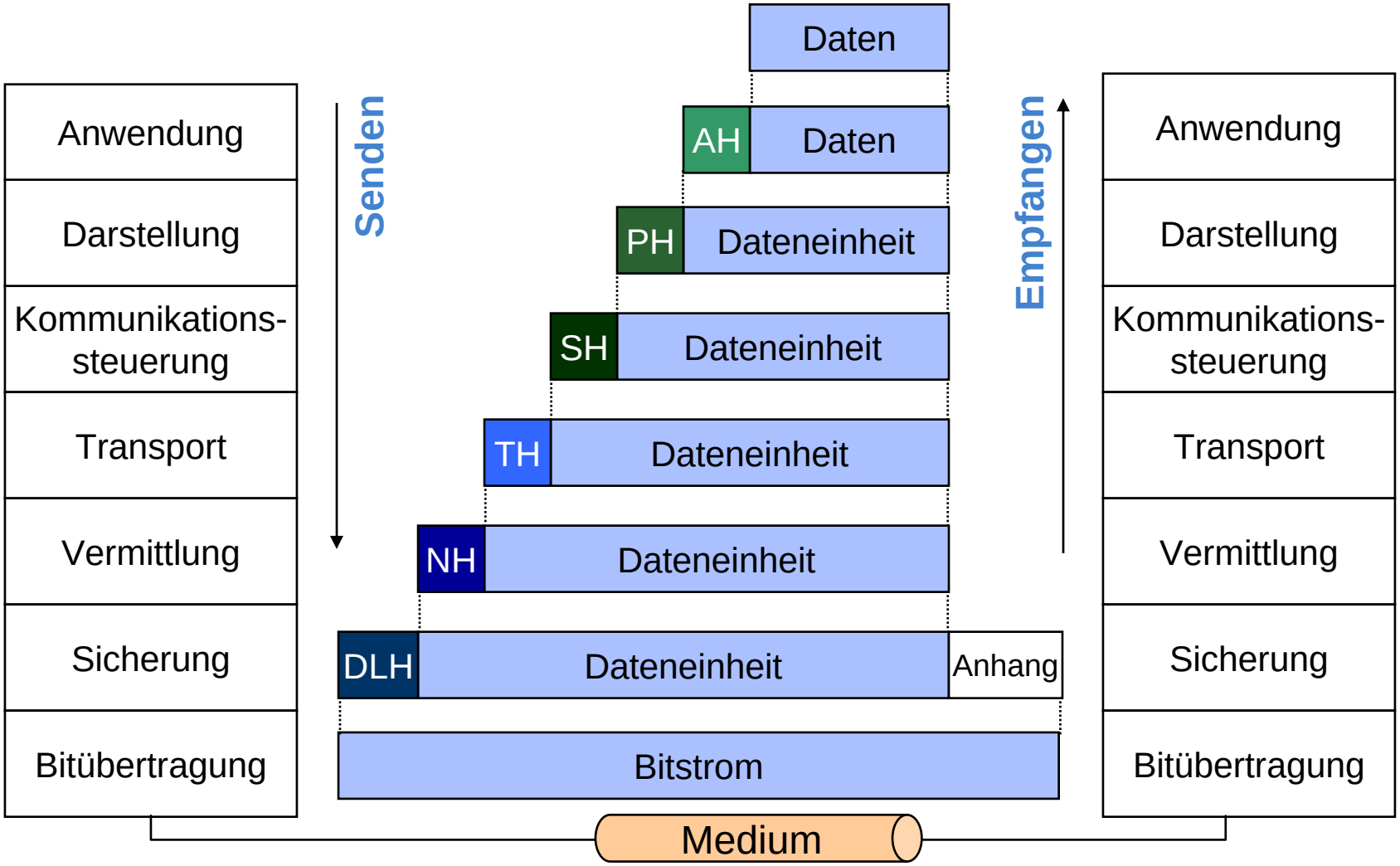
# Die anwendungsorientierten Schichten

- Schicht 5: Kommunikationssteuerungsschicht (Session Layer)
  - Bietet Nichtunterbrechbarkeit von Kommunikationsbeziehungen
  - Gliederung des Datenaustauschs nach Gesichtspunkten der Anwendung
    - Ablaufsteuerung und -koordination
    - Bereitstellen sogenannter Sitzungen
    - Rücksetzvereinbarung
- Schicht 6: Darstellungsschicht (Presentation Layer)
  - Kommunikation zwischen heterogenen Geräten möglich
    - Z.B. unterschiedliche Darstellung der Daten (Integer → 16 oder 32 bit?)
  - Einheitliche Darstellung der Daten (Syntax)
    - Überführung in bzw. aus der lokalen Syntax erforderlich
  - Beibehaltung der Semantik der Information
- Schicht 7: Anwendungsschicht (Application Layer)
  - Austausch von anwendungsabhängigen Daten
    - Aufbau und Zweck durch Anwendung bestimmt
  - Beispiele
    - E-Mail, Dateitransfer, WWW

# Kapselung der Daten

- Ziel
  - Austausch von Information innerhalb einer Anwendung
- Geschichtete Architektur
  - Information wird durch alle Schichten durchgereicht, sowohl beim Empfangen als auch beim Senden
  - Daten werden beim Senden in jeder Schicht gekapselt
    - Kontrollinformation, die für den ordnungsgemäßen Ablauf des Protokolls der Schicht benötigt wird
    - Kopf: Kontrollinformation vor den Daten
      - Z.B. Adressen in Dateneinheiten
    - Anhang: Kontrollinformation hinter den Daten
      - Z.B. Prüfsummen in Dateneinheiten
  - Beim Empfangen muss Kontrollinformation vor der Weitergabe an die darüber liegende Schicht entfernt werden

# Kapselung der Daten




AH: Application Header
PH: Presentation Header

SH: Session Header
TH: Transport Header

NH: Network Header
DLH: Data Link Header

# Dateneinheiten und Kontrollinformation

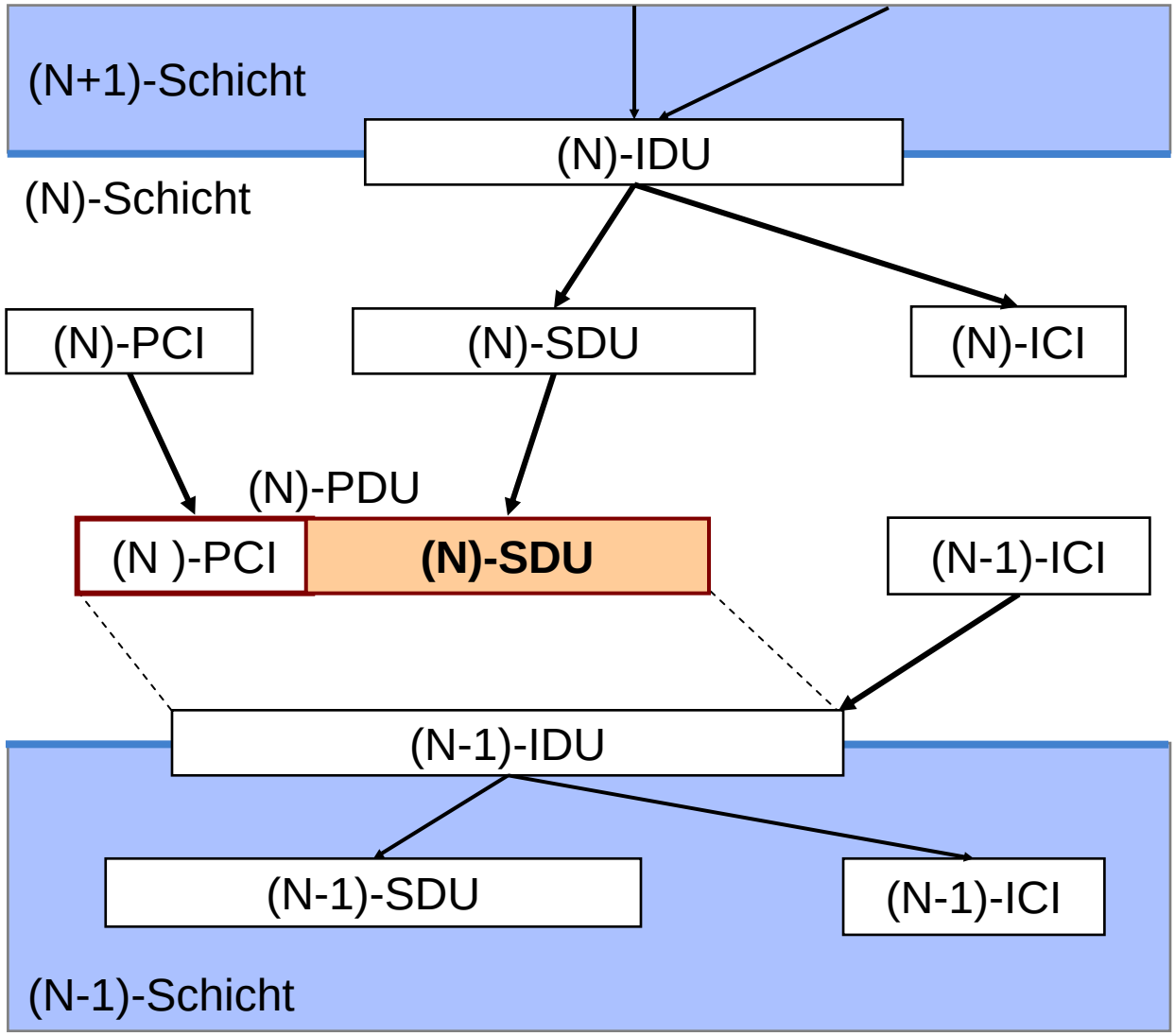
- (N)-Schnittstellendateneinheit (*Interface Data Unit* – IDU)
  - Dateneinheit, die die (N+1)-Schicht an die (N)-Schicht über den Dienstzugangspunkt übergibt
    - Enthält Nutzdaten (N)-SDU und Schnittstellenkontrollinformationen (N)-ICI
- (N)-Schnittstellenkontrollinformation (*Interface Control Information* – ICI)
  - Enthält Information über den auszuführenden Dienst, z.B: Parameter des Dienstes, Länge der SDU, ...
- (N)-Dienstdateneinheit (*Service Data Unit* – SDU)
  - Daten der (N+1)-Schicht (=(N)-PDU), die transparent übertragen werden
    - PDU: Protocol Data Unit
- (N)-Protokollkontrollinformationen (*Protocol Control Information* – PCI)
  - Information die zwischen (N)-Instanzen ausgetauscht wird, um den Protokollablauf zu steuern
    - Horizontale Kommunikation

# Zusammenspiel der Schichten

- Grundlegendes Prinzip
  - (N+1)-Instanz übergibt an der Dienstschnittstelle nach unten eine (N)-IDU, bestehend aus
    - Transparent zu übertragenden Nutzdaten: (N)-SDU
    - Steuerinformation: (N)-ICI
  - Zur Übertragung der (N)-SDU wird gemäß den Formatvorschriften des Protokolls eine (N)-PCI erzeugt
    - Bildet gemeinsam mit (N)-SDU die (N)-PDU
  - (N)-PDU wird transparent zwischen Protokollinstanzen der Schicht N übertragen
    - Hierzu werden Dienste der Schicht (N-1) in Anspruch genommen
      - Entsprechende Kontrollinformation (N-1)-ICI erzeugt
      - (N)-PDU und (N-1)-ICI bilden (N-1)-IDU

# Zusammenspiel der Schichten

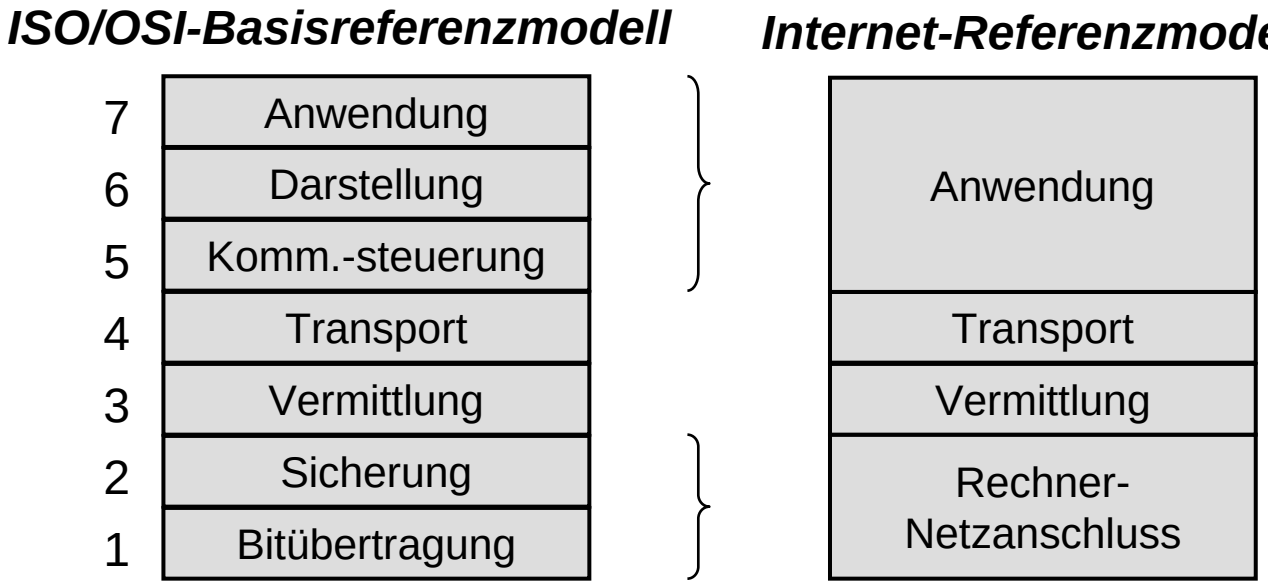
- Schematische Darstellung



# Anmerkungen

- ISO/OSI-Referenzmodell ist der „Klassiker“
  - Gut geeignet zur logischen Strukturierung
  - Oft weit weg von der Strukturierung einer Anwendung
  - Manchmal zu detailliert und komplex … sagen zumindest Kritiker
- Tipp
  - Versuchen Sie in diesem Modell zu „leben“
    - Ordnen Sie immer wieder die Problemstellungen in Kommunikationssystemen sorgfältig in dieses Modell ein
    - Das erleichtert die Beherrschung des umfangreichen Stoffs ungemein
- Modellierung vs. Implementierung
  - Schichten werden nicht separat implementiert
  - Daten werden zwischen Schichten möglichst nicht kopiert

# 4.6 Modell von ISO/OSI versus Internet




- Wesentliche Änderungen im Internet-Referenzmodell
  - Aufgaben der Schichten 5 und 6 vollständig in Anwendung verlagert
  - Schichten 1 und 2 zu einer Schicht zusammengefasst
- Interessant im Internet-Referenzmodell
  - Die konkreten Protokolle, die das Modell ausfüllen

# ISO/OSI versus Internet

- ISO/OSI
  - Modelltechnische Grundlagen für Kommunikationssysteme
    - Richtungsweisender Standard
    - Protokolle der ISO/OSI haben wenig praktische Bedeutung
  - Allerdings
    - Viele individuelle Anpassungen des Referenzmodells
      - z.B. in der Internet-Welt oder bei Systemen der Mobilkommunikation
- Internet-Aktivitäten
  - Allgemein akzeptierte Protokollstandards
    - TCP/IP-Protokollfamilie
    - Referenzmodell ist von untergeordneter Bedeutung

# 4.7 Vorlesungsüberblick: Schichten

- … Vorgehensweise
  - **Bottom-up**

***ISO/OSI-Basisreferenzmodell***




1. Einführung
2. Physikalische Grundlagen
3. Protokollmechanismen
4. Geschichtete Architekturen
5. Sicherungsschicht: HDLC
6. Beschreibungsmethoden
7. Sicherungsschicht: Lokale Netze
8. Netzkopplung und Vermittlung
9. Die Transportschicht
10. Anwendungssysteme
11. Middleware

# Vorlesungsüberblick: Prinzipien

**Top-down:**
**Prinzipien,**
**Mechanismen**
**Beschreibungsmethoden**

**Schichtenübergreifend:**

1. Einführung
2. Physikalische Grundlagen
3. Protokollmechanismen
4. Geschichtete Architekturen
5. Sicherungsschicht: HDLC
6. Beschreibungsmethoden
7. Sicherungsschicht: Lokale Netze
8. Netzkopplung und Vermittlung
9. Die Transportschicht
10. Anwendungssysteme
11. Middleware

# Zusammenfassung

- Grundlegende Modelle und Begriffe für Kommunikationssysteme
  - Schichten
  - Protokolle
  - Dienste
  - Dienstzugangspunkte

- … abstrakt aber wichtig!
  - Ordnen Sie immer wieder die Problemstellungen in das ISO/OSI-Basisreferenzmodell ein!
    - Hilft fürs Verständnis
    - … auch wenn es nicht unbedingt praxisrelevant ist

# Übungen

4.1 Nennen und erklären Sie die im Zusammenhang mit Diensten und deren Zusammensetzung und Inanspruchnahme auftretenden Begriffe.

4.2 Welche Grundformen von Diensten werden bezüglich der Mechanismen *Bestätigungen* und *Verbindungen* unterschieden und worin besteht jeweils der Unterschied?

4.3 Erläutern Sie detailliert den Aufbau einer geschichteten Architektur, sowie deren Prinzipien, Komponenten und deren Zusammenspiel.

4.4 Wozu dient ein „Dienstzugangspunkt“ und wo befinden sich solche innerhalb einer Schichtenarchitektur?

4.5 Was sind „Protokollinstanzen“?

4.6 Was wird unter „vertikaler“ und „horizontaler Kommunikation“ verstanden und zwischen welchen Komponenten finden diese statt?

4.7 Was sind (N)-PDUs, wer tauscht diese miteinander aus und wie setzen sie sich zusammen?

4.8 Was sind die Vorteile einer geschichteten Architektur?

4.9 Welcher Nachteil kann sich aus einer „zu feinen“ Schichtung und den daraus resultierenden (zu) zahlreichen Schichten ergeben?

4.10 Vergleichen Sie das ISO/OSI-Basisreferenzmodell mit dem des Internets und diskutieren Sie kurz die Unterschiede.

# Literatur


- [Haaß97] W. Haaß; Handbuch der Kommunikationsnetze; Springer Verlag, 1997
  - Kapitel 3
- [JaAg90] B. Jain, A. Agrawala; Open Systems Interconnection; Elsevier, 1990
  - Behandelt das ISO/OSI-Basisreferenzmodell im Detail
  - Behandelt ebenfalls die ISO/OSI-Protokolle
- [SpJa93] O. Spaniol, K. Jakobs; Rechnerkommunikation; VDI-Verlag, 1993
  - Detaillierte Behandlung ISO/OSI-Basisreferenzmodell und ISO/OSI-Protokolle
- [Zimm80] H. Zimmermann; OSI Reference Model – The ISO Model of Architecture for Open Systems Interconnection; IEEE Transactions on Communications, Vol. 28, No. 4, April, 1980
  - Klassiker zum Thema ISO/OSI-Referenzmodell

# Kommunikation und Datenhaltung

## 5. Sicherungsschicht: HDLC




**Prof. Dr. Martina Zitterbart**
Dipl.-Inform. Martin Röhricht
[zit | roehricht]@tm.uka.de



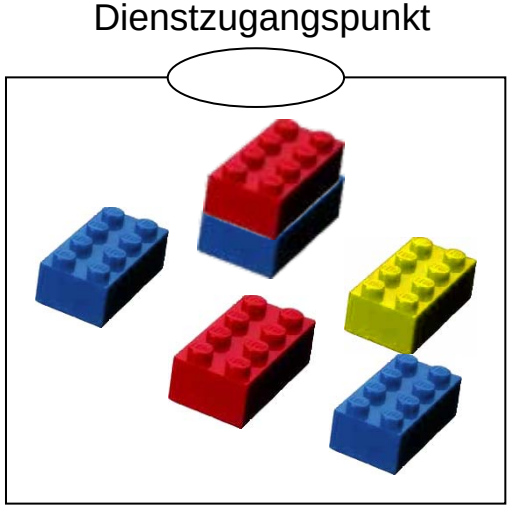
# Kapitelübersicht

1. Einführung
2. Physikalische Grundlagen
3. Protokollmechanismen
4. Geschichtete Architekturen
5. Sicherungsschicht: HDLC
6. Beschreibungsmethoden
7. Sicherungsschicht: Lokale Netze
8. Netzkopplung und Vermittlung
9. Die Transportschicht
10. Anwendungssysteme
11. Middleware

5.1 Basis-Szenario
5.2 Entwicklung von HDLC
5.3 Konfiguration und Betriebsarten
5.4 HDLC-Dateneinheiten
5.5 Protokollablauf

# Motivation

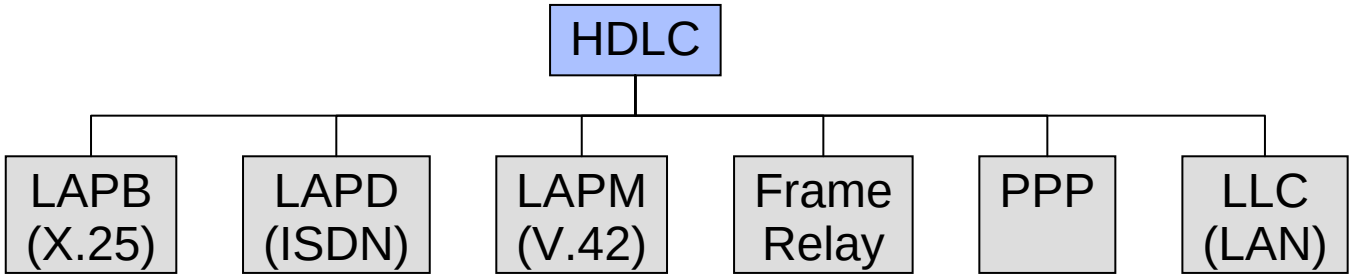
- … wo stehen wir in der Vorlesung?
  - … eine Reihe von Bausteinen für die Kommunikation aus Kapitel 3
  - … physikalische Grundlagen aus Kapitel 2
- … eigentlich interessieren uns komplette Protokolle und die von ihnen angebotenen Dienste!
  - Definition Protokoll?
  - Definition Dienst?




- Ziel dieses Kapitels
  - Präsentation eines kompletten Protokolls
    - Verwendung der Bausteine aus Kapitel 3
  - Auswahl eines Beispielprotokolls
    - … HDLC (High-level Data Link Control) – ein Klassiker

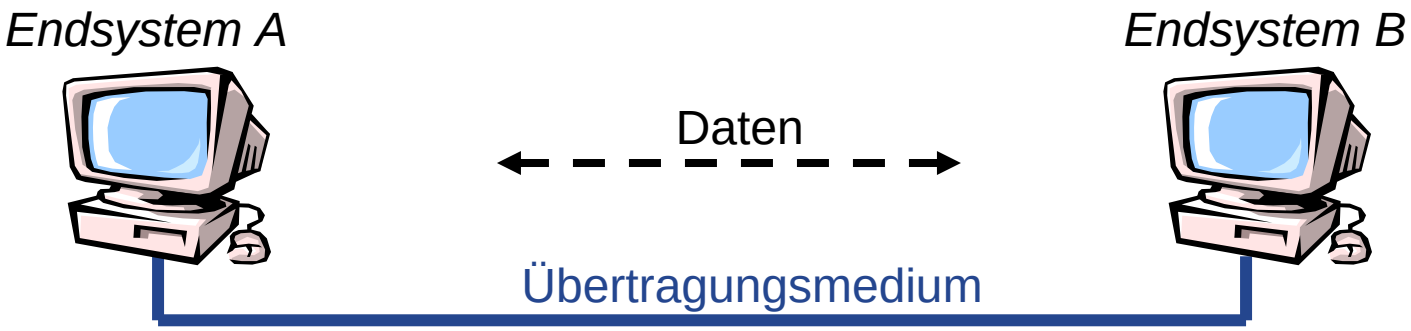
# HDLC – nur ein Klassiker?

- Unbewusst bei den meisten im täglichen Einsatz
  - Mobilfunktelefonie über GSM
  - Einwahl per DSL ins Internet mittels PPP
- HDLC bildet Basis vieler Sicherungsschicht-Protokolle
  - Große Vielzahl an Derivaten
    - X.25 Protokolle (LAPB)
    - V.42 Protokoll für Modems (LAPM)
    - Frame Relay Protokolle (LAPF)
    - ISDN Protokoll (LAPD)
    - GSM Protokoll (LAPM)
    - IEEE 802.2 Logical Link Control Protokoll (LLC)



# 5.1 Basis-Szenario für HDLC-Einsatz

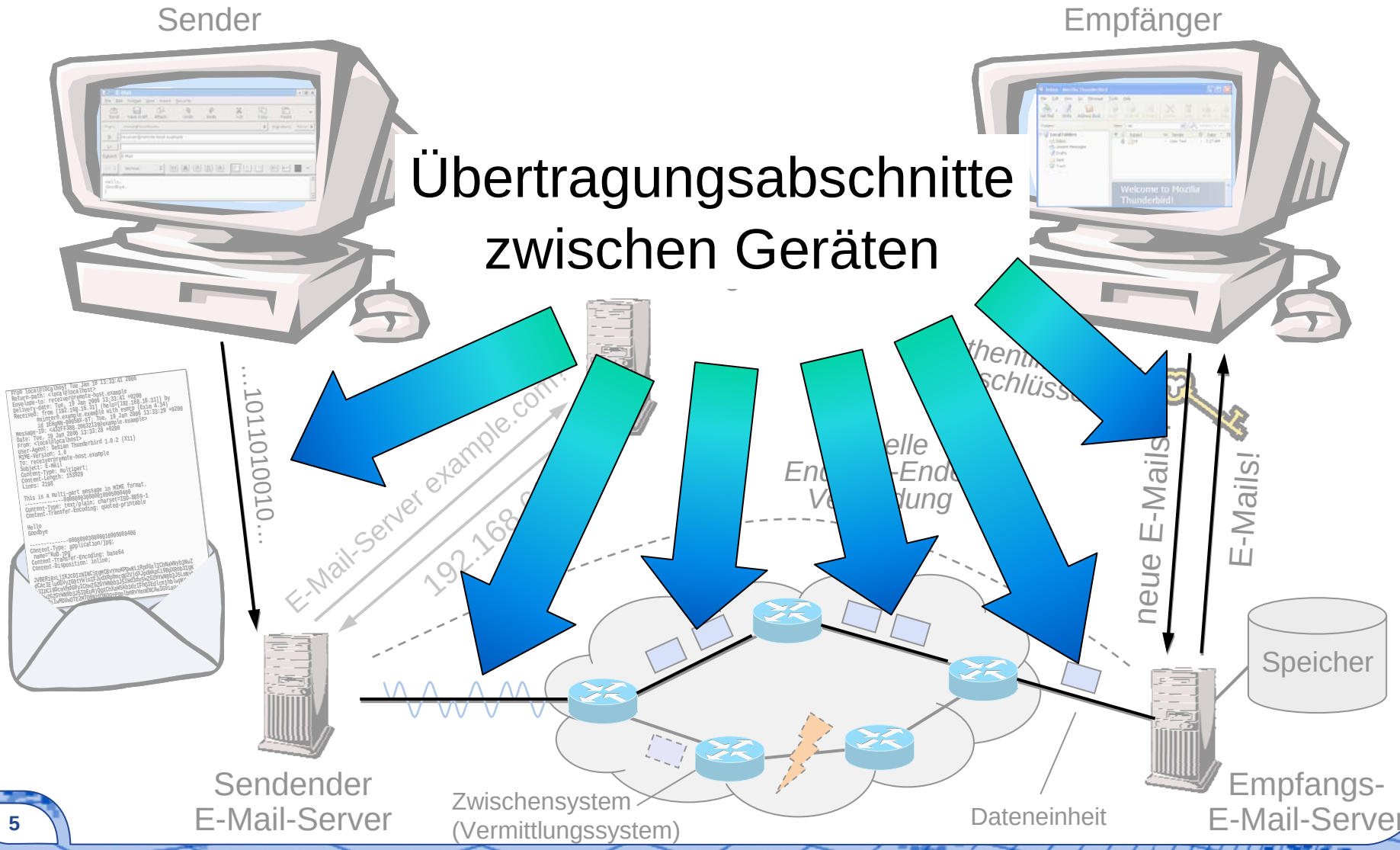
- Szenario




- Ziel
  - Endsysteme A und B sind über ein Übertragungsmedium direkt miteinander verbunden und wollen über dieses zuverlässig Daten austauschen

- … Aufgabe der Sicherungsschicht

# … in unserem Beispiel

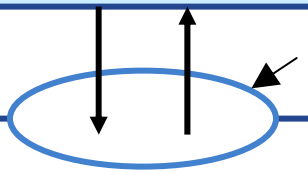


# Aufgaben der Sicherungsschicht

**2 Sicherungsschicht**

- Fehlererkennung und -behebung
- Strukturierung des Datenstroms
- Medienzugangskontrolle bei geteilten Medien

  ⇨ Sicherungsschicht bietet abstraktes Medium „gesicherten Kanal“



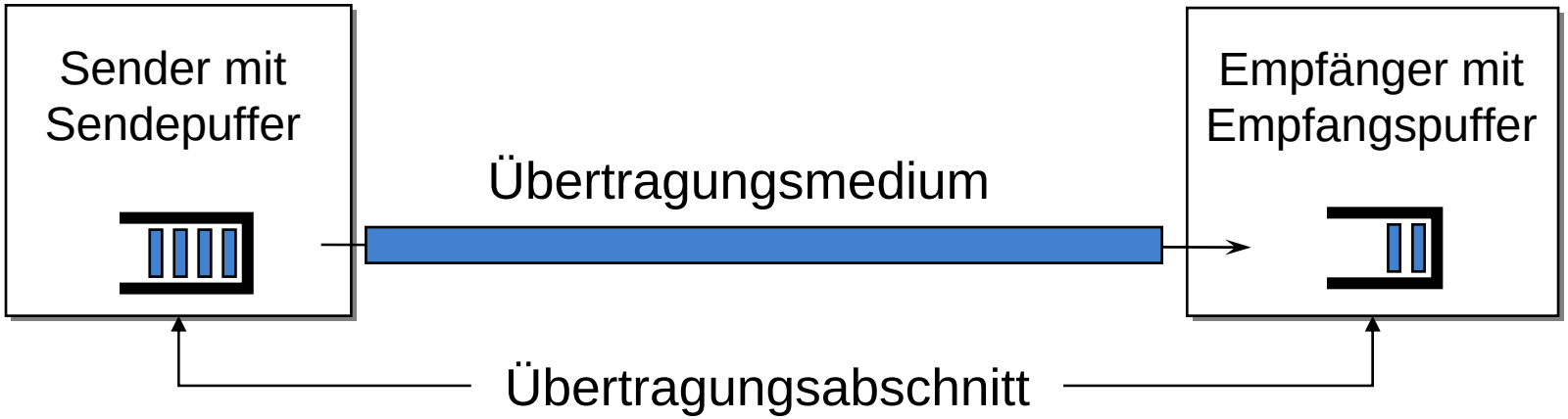
Schicht-1-Dienstzugangspunkt

**1 Einige Probleme der physikalischen Schicht**

- unerwünscht hohe Störungen der Übertragung
- kaum beeinflussbare Übertragungsqualitäten
- keine Pufferung möglich

# Übertragungsabschnitt

- Grundlegendes Modell




- Übertragungsabschnitt
  - Umfasst konzeptionell beide Pufferspeicher und das Übertragungsmedium
  - Übertragungsmedium wird häufig als nicht speichernd bezeichnet
    - Bei schnellen Netzen bzw. bei Netzen mit langen Übertragungswegen stimmt dies nicht ganz
      - Wie viele Daten befinden sich bei einer Datenrate von 1 Gbit/s auf einem 100 km langen Übertragungsmedium?

# 5.2 Zur Entwicklung von HDLC

- Bisher
  - Zeichen-orientierte Protokolle
    - Wenig flexibel, halb-duplex, Abhängigkeit vom Code, nur eine Funktion pro gesendeter Dateneinheit
- Zielsetzung
  - Hohe Flexibilität
  - Adaptierbarkeit für verschiedene Anwendungen und Konfigurationen
  - Übertragung sowohl halb-duplex als auch vollduplex
  - Hohe Zuverlässigkeit
  - Hohe Effizienz
  - Punkt-zu-Punkt- sowie Punkt-zu-Mehrpunkt-Kommunikation
- Entwicklung eines bitorientierten Protokolls
  - Mitte bis Ende der 70er Jahre
  - Reihe unterschiedlicher Protokolle wurde gleichzeitig entwickelt
    - HDLC ist der ISO-Standard


[Gree83]


[ISO02]

# Resultierende Aufgaben

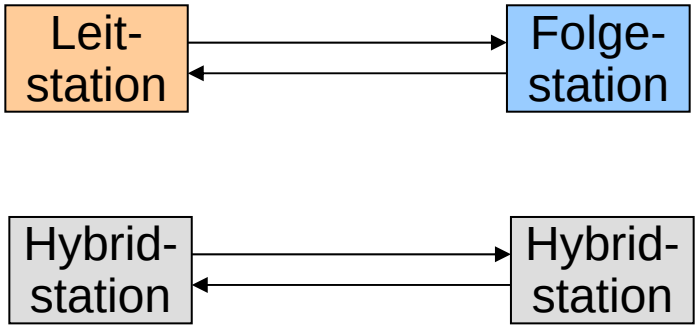
- Bereitstellung eines zuverlässigen Dienstes
  - Fehlererkennung
    - CRC zur Erkennung von Bitfehlern
  - Fehlerbehebung
    - Quittungen und Sendewiederholungen
    - Einsatz eines ARQ-Verfahrens
    - Flusskontrolle
  - Erhaltung der Übertragungsreihenfolge
    - Sequenznummern
  - Auf- und Abbau von Verbindungen

# 5.3 Konfiguration und Betriebsarten

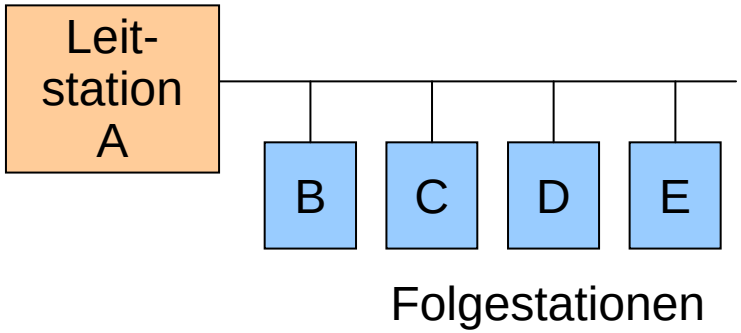
- Stationstypen
  - Leitstation
    - sendet Befehle an Folgestationen
  - Folgestationen
    - senden Meldungen an Leitstation
  - Hybridstationen
    - fungieren sowohl als Leit- als auch als Folgestation
- Verbindungskonfigurationen
  - Asymmetrisch
    - Eine Leitstation mit einer Reihe von Folgestationen
  - Symmetrisch
    - Nur für Punkt-zu-Punkt-Kommunikation mit Hybridstationen

# Kommunikationsformen

- Punkt-zu-Punkt




- Punkt-zu-Mehrpunkt

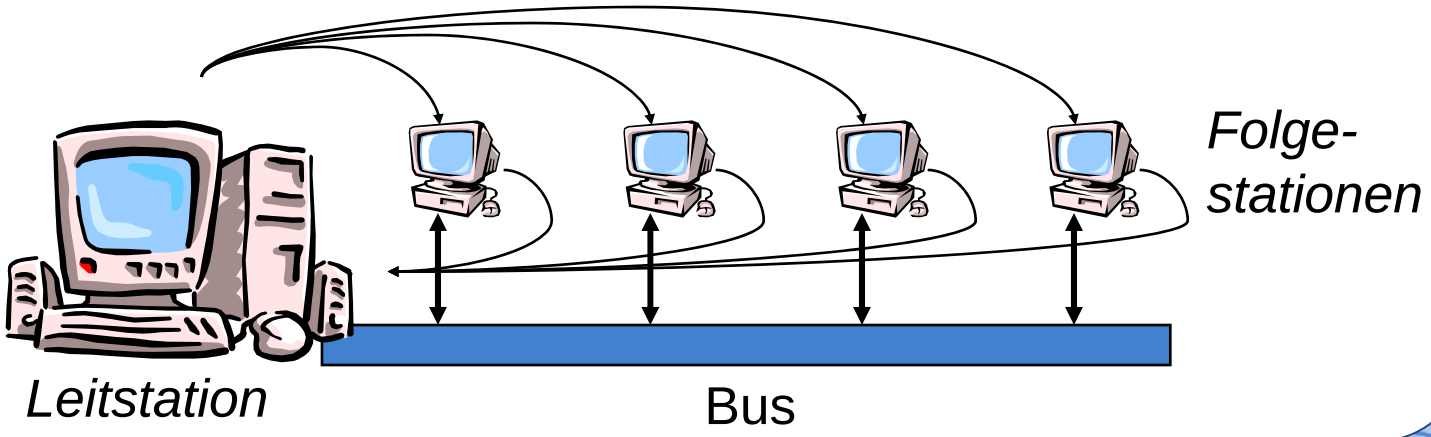


# Betriebsarten

- Drei Betriebsarten werden unterstützt
  - Aufforderungsbetrieb (*Normal Response Mode*, NRM)
    - Asymmetrische Konfiguration
    - Eine Folgestation kann nur nach einem Sendeaufruf („Polling“) durch die Leitstation senden
    - Leitstation ist zentrale Kontrollstelle
  - Spontanbetrieb (*Asynchronous Response Mode*, ARM)
    - Asymmetrische Konfiguration
    - Die Leit- oder Folgestation kann übertragen, falls die Leitung frei ist
    - Leitstation ist zentrale Kontrollstelle
  - Gleichberechtigter Spontanbetrieb (*Asynchronous Balanced Mode*, ABM)
    - Symmetrische Konfiguration
    - Nur für Punkt-zu-Punkt-Kommunikation
    - Stationen sind vollkommen gleichberechtigt

# Aufforderungsbetrieb

- Eigenschaften
  - „Intelligente“ Leitstation hat zentrale Kontrolle
    - „Dumme“ Folgestationen sind vom Design her weniger komplex
  - Jegliche Kommunikation erfolgt über die Leitstation
    - Keine direkte Kommunikation zwischen den Folgestationen
    - Leitstation fragt Folgestationen gemäß Abfragetabelle („Polling Table“) ab
    - Folgestationen antworten nur nach Aufforderung
  - Logische Struktur (Leit- und Folgestationen)
    - Physikalische Struktur dadurch nicht festgelegt (z.B. Stern, Bus)
- Schema



# Spontanbetrieb

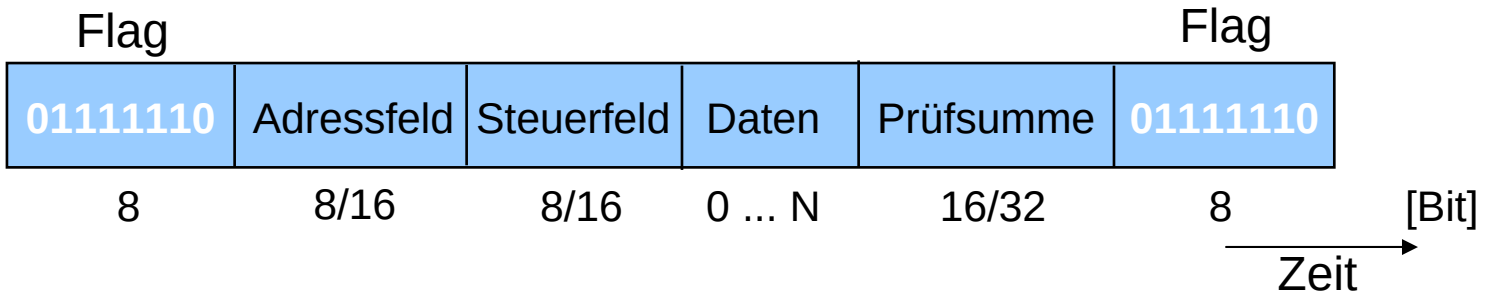
- Eigenschaften
  - Folgestation darf ohne explizite Aufforderung senden
  - Gut geeignet, falls
    - Leit- und Folgestation „frei“ (ohne Pollingstrategie) miteinander kommunizieren möchten
  - Bei Punkt-zu-Mehrpunkt-Kommunikation
    - Nur eine der Folgestationen kann zu einem Zeitpunkt aktiv sein
  - Leitstation hat nach wie vor die Kontrolle über die Datenübertragung
    - … dies ist ein wesentlicher Unterschied zum gleichberechtigten Spontanbetrieb, wo beide Stationen absolut gleichberechtigt sind

# Anmerkungen aus  [Carl80]

- Gleichberechtigter Spontanbetrieb war ursprünglich nicht enthalten
  - Aber
    - „political considerations provided some of the motivation for inclusion of the asynchronous balanced mode (ABM). The NRM and ARM modes had been defined first, and were pretty well fixed in place, and generally accepted internationally, when it was observed that they did not quite satisfy all of the „requirements“. “…
    - „the overall control of the data flow and responsibility for system error recovery resided in only one of the stations – the designated primary station. For many applications this was considered to be unacceptable. For example, when interconnecting governments, corporations, independent systems, etc., the thought of being the „secondary“ to another, dependent upon another for one‘s operation and livelihood, was generally unacceptable.“
  - Nicht nur technische Gründe spielen manchmal eine Rolle bei grundlegenden Entwurfsentscheidungen

# 5.4 HDLC-Dateneinheiten

- Ziel
  - Festlegung des Formats der dem Protokoll zugrunde liegenden Dateneinheiten
    - Erinnerung: Protokolle umfassen Regeln und Formate
- Format von HDLC-Dateneinheiten
  - In der Literatur oft als HDLC-Rahmen (*HDLC Frame*) bezeichnet
  - Bestandteile ergeben sich direkt aus den Aufgaben
    - Wo beginnt/endet eine Dateneinheit?
    - Wie können Fehler erkannt und behoben werden?
    - Für wen bzw. von wem ist/stammt die Dateneinheit
    - Wie können die Regeln für den Ablauf umgesetzt werden?
  - Typische Darstellung von Dateneinheiten

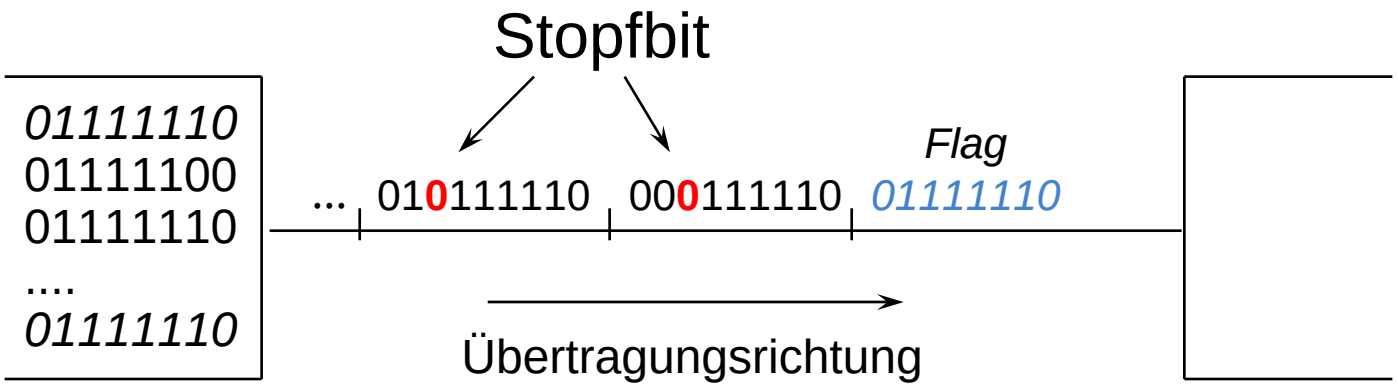



  - Flags kennzeichnen Anfang und Ende
    - Ende-Flag kann gleichzeitig Anfangs-Flag der folgenden Dateneinheit sein

# Codetransparenz durch Bitstopfen

- Ziel
  - Übertragung soll unabhängig von einem Code sein
- Problem
  - Flag (01111110) ist ausgezeichnete Bitfolge
  - Was passiert, wenn Flag in den anderen Feldern der Dateneinheit auftritt?
- Vorgehensweise
  - Bitstopfen
    - Sender fügt zwischen den Flags nach 5 aufeinanderfolgenden Binärzeichen „1“ ein Binärzeichen „0“ ein
    - Empfänger entfernt nach 5 aufeinanderfolgenden Binärzeichen „1“ ein folgendes Binärzeichen „0“
    - Bitstopfen erfolgt beim Aussenden bzw. Empfangen des Bitstroms
      - Prüfsumme wird beim Sender vor (!) dem Bitstopfen und beim Empfänger nach (!) dem Bitstopfen berechnet bzw. überprüft

# Bitstopfen: Beispiel

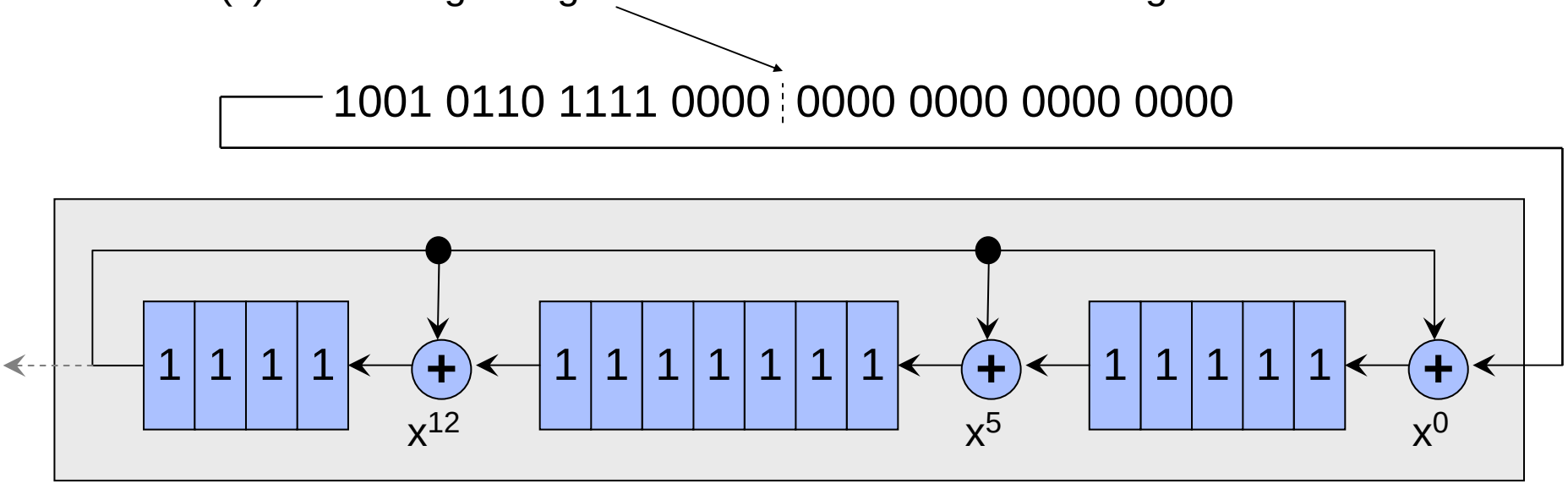


# Prüfsumme

- Verfahren
  - CRC-16 mit Generatorpolynom
    - $G(x) = x^{16} + x^{12} + x^5 + 1$
  - Berechnet über
    - Adressfeld, Steuerfeld und Daten
      - Inhalt zwischen Flags, aber ohne Flags (Bitstopfen erfolgt danach)
- Variante bei der Berechnung
  - Von Standardberechnung abweichend / zusätzlich:
    - Schieberegister zu Anfang mit Einsen gefüllt
    - Der berechnete Rest wird invertiert (Einer-Komplement) bevor er als Prüfsumme mit übertragen wird
    - Beim Empfänger
      - Schieberegister mit Einsen gefüllt
      - Division von Adressfeld, Steuerfeld, Daten, Prüfsumme und 16 angehängten Nullen durch Generatorpolynom
      - Rest, der erfolgreiche Übertragung anzeigt, besteht nicht aus Nullen (!), sondern entspricht Bitmuster 0001 1101 0000 1111
  - Hierdurch „robustere“ Prüfsumme
    - z.B. gegen Anfügen/Löschen von Nullen

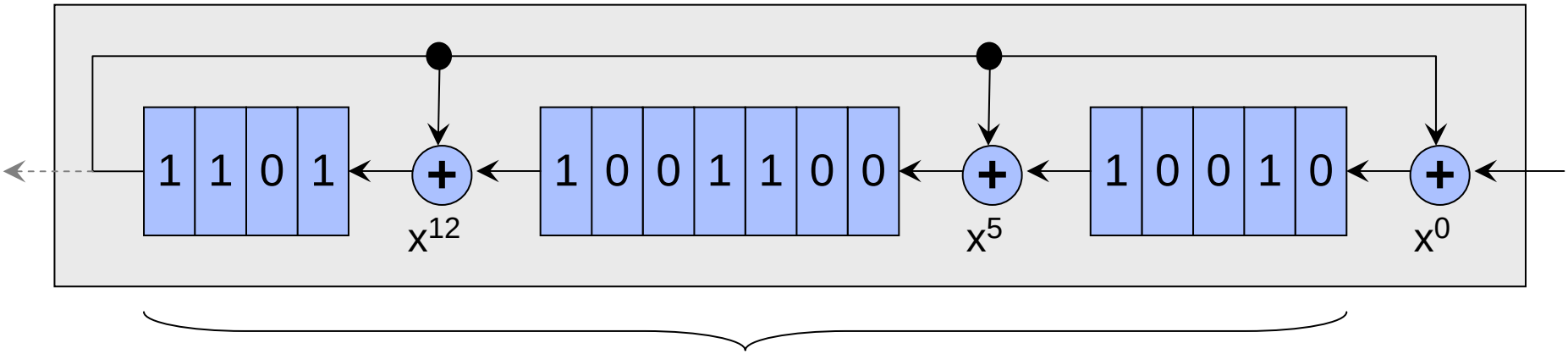
# Beispiel Prüfsumme

- Generatorpolynom: $G(x) = x^{16} + x^{12} + x^5 + 1$
  G(x) = 1 0001 0000 0010 0001
  Grad von G(x) = 16
- Daten: M(x) = 1001 0110 1111 0000
- Prüfsummen-Berechnung beim Sender
  - Schieberegister zu G(x) mit Einsen initialisieren
    - Beachten beim Vergleich mit „manueller Berechnung“!
  - M(x) mit 16 angehängten Nullen durch das Schieberegister schieben



# Beispiel Prüfsumme

- Einer-Komplement des Rests bilden (invertieren) und als Prüfsumme mit übertragen
  - z.B. durch aufaddieren auf M(x) mit angehängten Nullen




Invertiert (Prüfsumme): 0010 0110 0110 1101

\+ 1001 0110 1111 0000 0000 0000 0000 0000

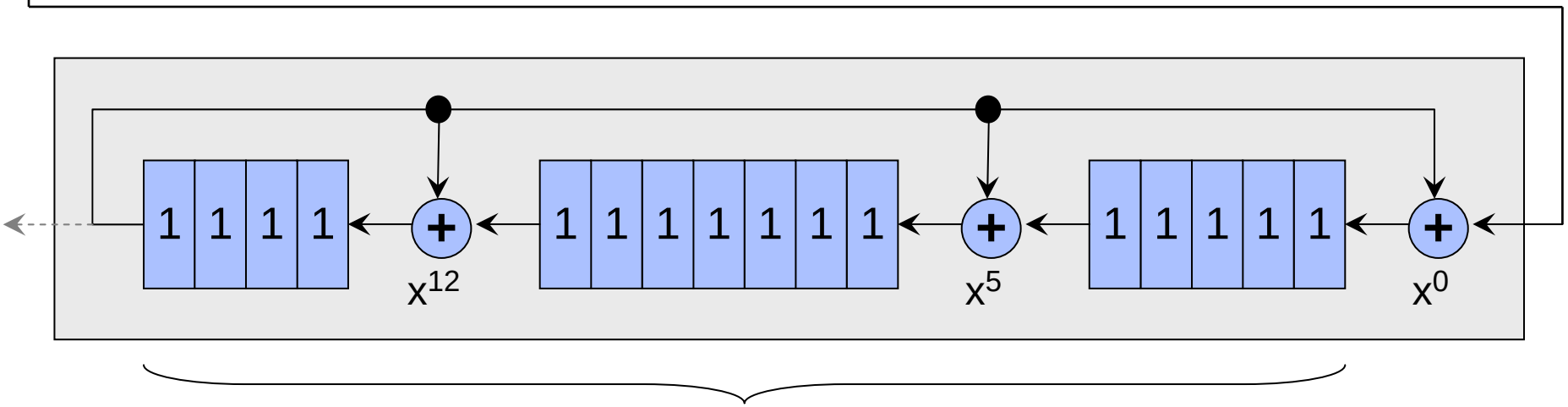
---

**1001 0110 1111 0000 0010 0110 0110 1101** wird übertragen

# Beispiel Prüfsumme

- Prüfsumme überprüfen beim Empfänger
  - Schieberegister mit Einsen initialisieren
  - An empfangene Daten (mit Prüfsumme) 16 Nullen anhängen
  - Durch Schieberegister schieben
  - Rest 0001 1101 0000 1111 zeigt erfolgreiche Übertragung an

1001 0110 1111 0000 0010 0110 0110 1101 | 0000 0000 0000 0000




Rest nach Berechnung: **0001 1101 0000 1111**

⇨ Übertragung (mit hoher Wahrscheinlichkeit) fehlerfrei

# Adressierung

- Basis-Adressfeld
  - 8 Bit zur Kodierung der Adresse
  - Bei Befehlen (Leitstation)
    - Adressierung der Empfängerstation
  - Bei Meldungen (Folgestation)
    - Adressierung der Sendestation
- Erweitertes Adressfeld
  - Größe des Adressfelds kann ein Vielfaches von 8 Bit sein
  - Gekennzeichnet durch eine 0 an der Bitposition 1
    - Größe der Adresse Vielfaches von 7 Bit
  - Wie wird das Ende des erweiterten Adressfeldes kodiert bzw. erkannt?
- Broadcast-Adresse
  - Alle Bits werden auf Eins gesetzt
  - Dateneinheiten werden von allen Stationen empfangen

# Steuerfeld

- Aufgabe
  - Identifiziert den Zweck der Dateneinheit

| **01111110** | 8/16 bit | 8/16 bit | n bit | 16/32 bit | **01111110** |
|---|---|---|---|---|---|
| Flag | Adressfeld | Steuerfeld | Nutzdaten | Prüf-summe | Flag |

| | | | |
|---|---|---|---|
| **0** | N(S) | P | N(R) |

I-Dateneinheit
[I=Information]

| | | | |
|---|---|---|---|
| **1 0** | S S | P/F | N(R) |

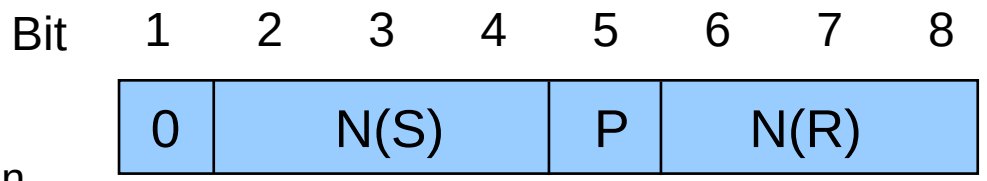
S-Dateneinheit
[S=Supervisory]

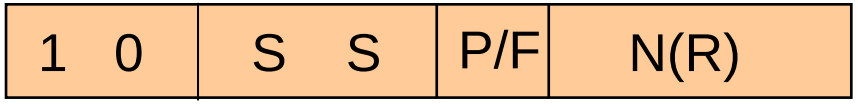
| | | | |
|---|---|---|---|
| **1 1** | M M | P/F | M M M |

U-Dateneinheit
[U=Unnumbered]

# HDLC-Dateneinheiten

- I-Dateneinheiten
  - Datenübertragung
    - Bei gesetztem Poll-Bit muss Empfänger mit Quittung reagieren
    - Die Sequenznummer N(R) quittiert den korrekten Empfang aller Dateneinheiten bis N(R)-1
- S-Dateneinheiten
  - Steuerung des Datenflusses
    - z.B. Sendeaufruf, Bestätigung empfangener Daten, ...
    - Kodierung der Befehle bzw. Meldungen durch *S*-Bits (Supervisory Funktionsbits)
- U-Dateneinheiten
  - Steuerung der Verbindung
    - keine Empfangsfolgenummern
    - Kodierung durch *M*-Bits (Modifier Funktionsbits)



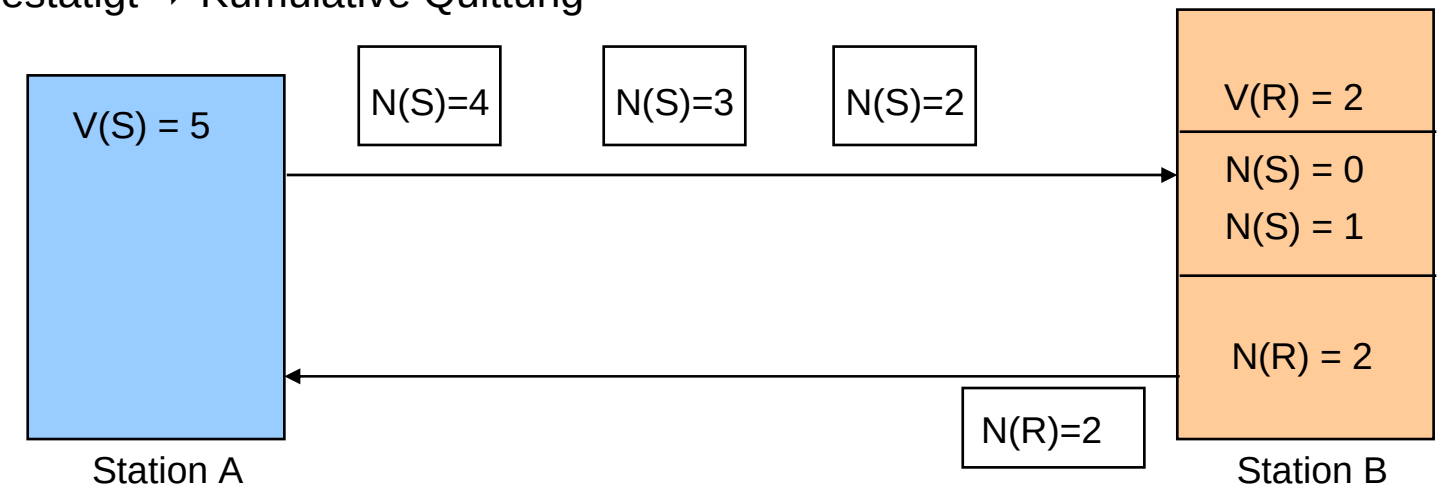




| | |
|---|---|
| N(S) | Sende-Sequenznummer |
| N(R) | Empfangs-Sequenznummer |
| P | Poll-Bit bei Befehlen |
| F | Final-Bit bei Meldungen |

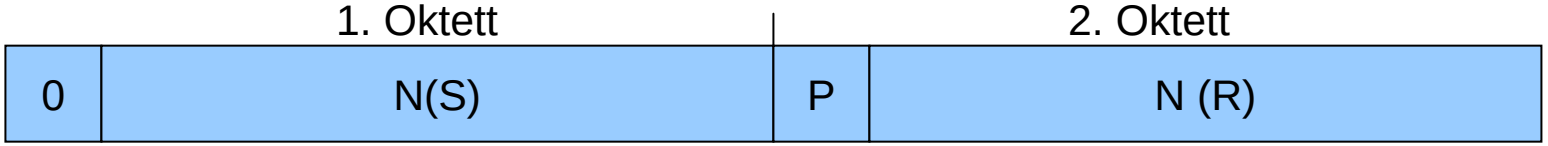
# Sequenznummern

- Sende-Sequenznummern N(S)
  - Sendefolge-Zähler V(S) gibt laufende Nummer der als nächstes zu sendenden I-Dateneinheit einer Station an
  - N(S) der I-Dateneinheit ergibt sich direkt aus V(S)
  - V(S) wird nach dem Senden einer I-Dateneinheit um 1 erhöht
- Empfangs-Sequenznummern N(R)
  - Empfangsfolge-Zähler V(R) erhält laufende Nummer der Sendefolge-Nummer N(S) der als nächstes erwarteten I-Dateneinheit
  - N(R) kann einer I- oder S-Dateneinheit mitgegeben werden
    - damit werden alle I-Dateneinheiten mit N(S) < N(R) als korrekt empfangen bestätigt ⇨ Kumulative Quittung
- Beispiel

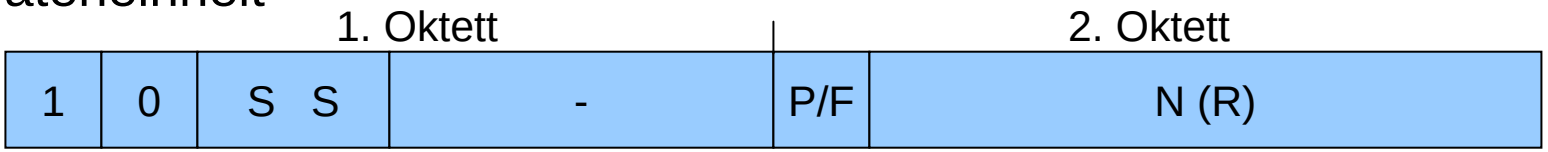


# Erweiterte Sequenznummern

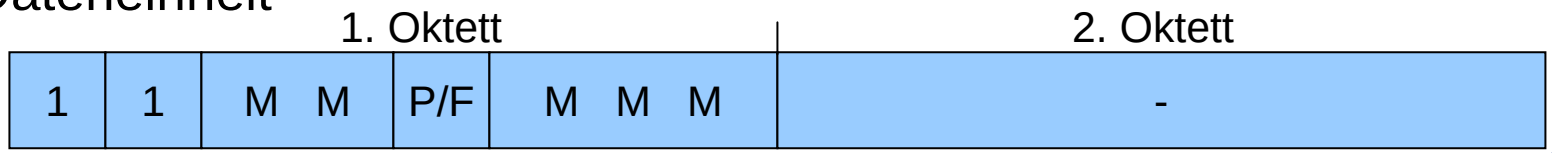
- Sendekredit
  - Durch 3-Bit Sequenznummern limitiert
- Erweiterung
  - Erhöhung des Sendefensters durch Sequenznummer von 7 Bit (*Extended Operation*) oder 15 Bit (*Super Operation*)
- HDLC-Dateneinheiten (Extended Operation)
  - I-Dateneinheit




  - S-Dateneinheit




  - U-Dateneinheit



# Markierungsbit P/F (Poll/Final)

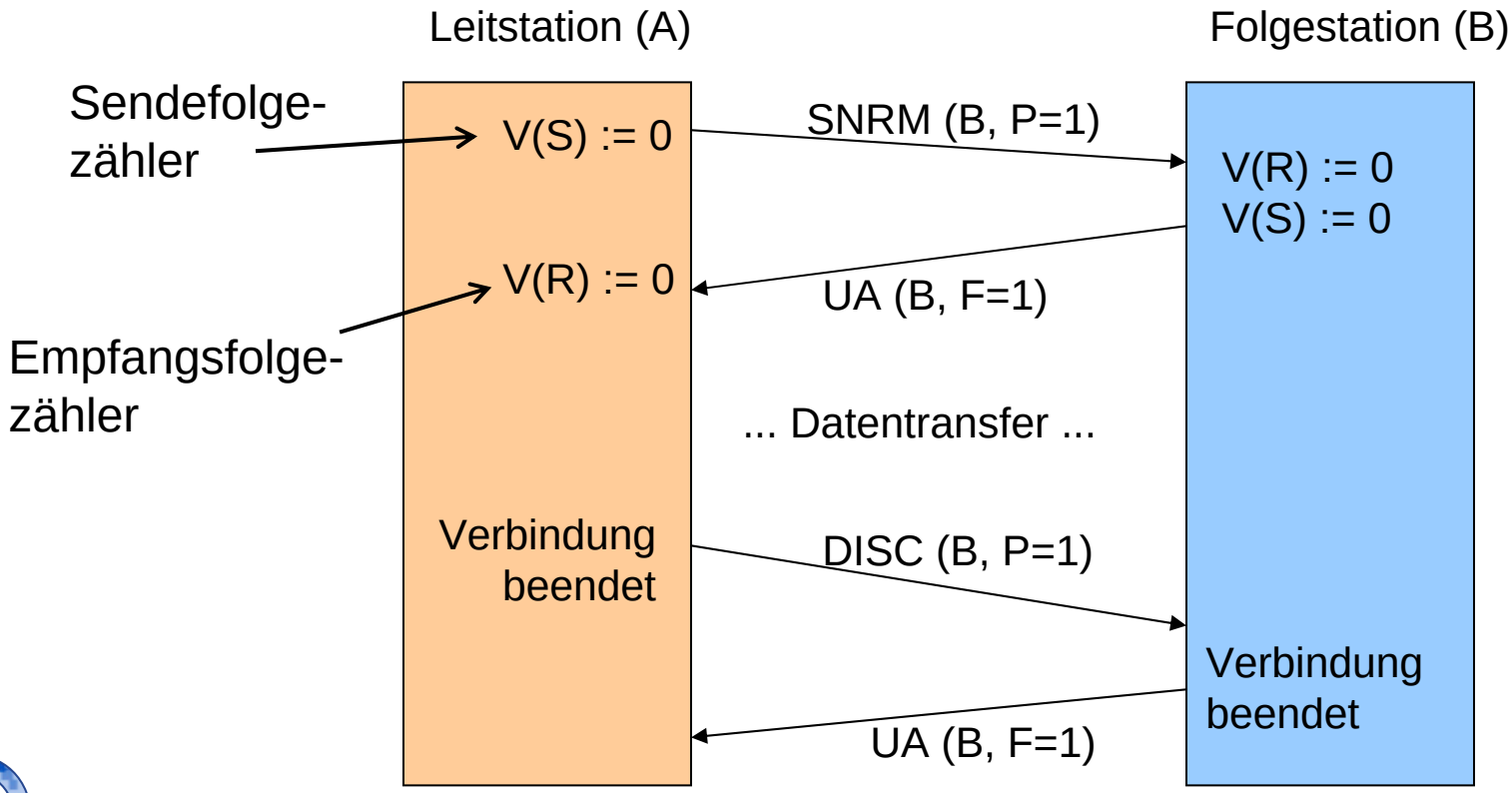
- Unterschiedliche Benutzung in Befehlen und Meldungen sowie in den verschiedenen Betriebsarten
  - PF=1 in Befehlen (Poll)
    - Anforderung einer Meldung bzw. einer Folge von Meldungen
  - PF=1 in Meldungen (Final)
    - Bestätigung des Empfangs eines Befehls mit PF=1, d.h. Meldungen als Antwort auf Befehle mit PF=1
- Normal Response Mode (NRM)
  - Folgestation darf nach Senden einer Meldung mit PF=1 als Antwort auf Befehl mit PF=1 keine weiteren Dateneinheiten ohne Erlaubnis durch die Leitstation senden
- Asynchronous Response Mode (ARM) und Asynchronous Balanced Mode (ABM)
  - Auf einen Befehl mit PF=1 muss vorrangig durch eine oder mehrere Meldungen mit PF=1 geantwortet werden; jedoch sind weitere Meldungen mit PF=0 möglich

# 5.5 Protokollablauf

- Gliederung in
  - Verbindungsmanagement und
  - Datentransfer
  - Damit drei Phasen
    - Verbindungsaufbau, Datentransfer und Verbindungsabbau
- Verbindungsmanagement
  - Vor jedem Datentransfer muss eine logische Verbindung etabliert werden
    - Was kennzeichnet eine logische Verbindung?
    - Was ist der Unterschied zwischen einer logischen Verbindung und einer physikalischen Verbindung?
- ... Eine solche Trennung ist typisch für alle Protokolle, die ihren Dienstnehmern einen zuverlässigen Dienst anbieten
  - ... Was bedeutet nochmal zuverlässig?

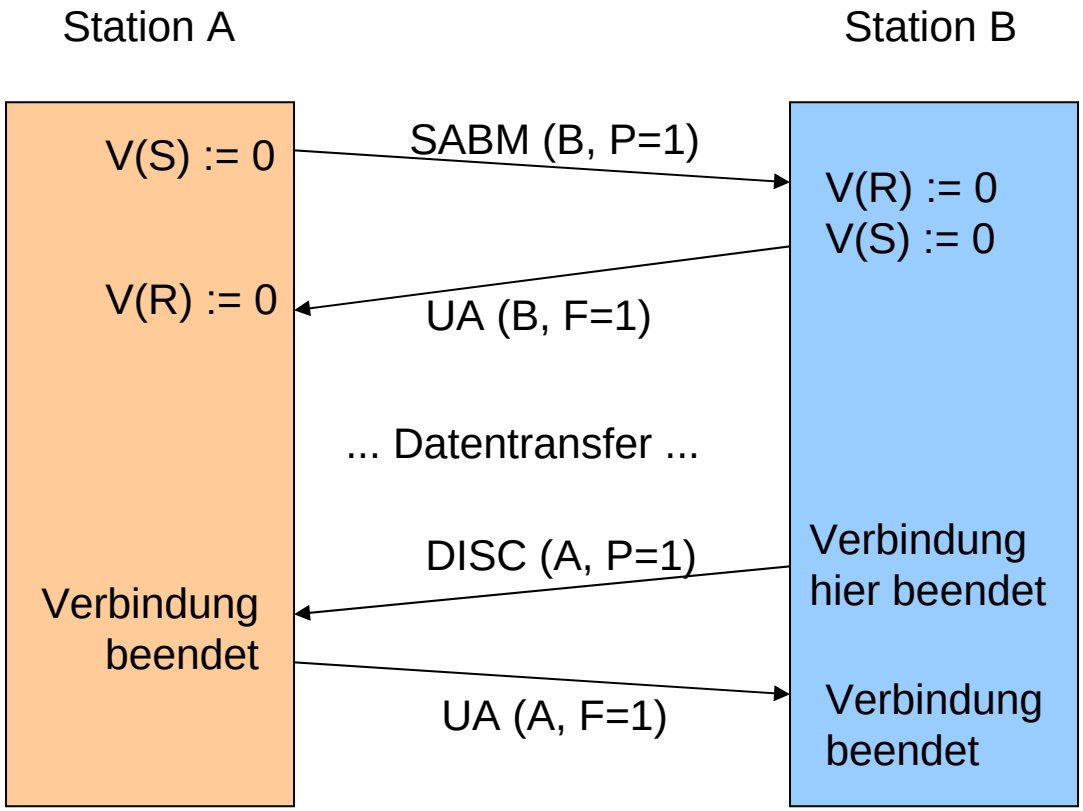
# 5.5.1 Verbindungsauf- und -abbau

- Verwendung von U-Dateneinheiten für die Signalisierung des Verbindungsaufbaus
  - Normal Response Mode (NRM)



# Verbindungsauf- und -abbau

- Asynchronous Balanced Mode (ABM)



# 5.5.2 Überblick U-Dateneinheiten

- U-Dateneinheiten
  - Für Auf- und Abbau von Verbindungen und Behandlung von Sonderfällen
- Befehle
  - SNRM (Set Normal Response Mode), SARM (Set Asynchronous Response Mode), SABM (Set Asynchronous Balanced Mode), ...
    - Starten einer bestimmten Betriebsart
    - Wirkung: Rücksetzen von V(S) und V(R) in den beteiligten Stationen
  - DISC (Disconnect)
    - Hebt alten Betriebszustand auf und geht in Wartezustand über ohne Datenaustausch
  - UA (Unnumbered Acknowledge)
    - Bestätigung des Erhalts der obigen U-Befehle durch die Folgestation
  - UP (Unnumbered Poll)
    - Anforderung von Meldungen der Folgestation(en) durch die Leitstation
  - UI (Unnumbered Information)
    - Es können Daten ohne Erhöhung der Sequenznummer gesendet werden
    - Beispiel: Zustandsinformation, Kenngrößen des Übermittlungsabschnitts
- Zu beachten
  - Unterschiedliche Bedeutung der Steuerfunktionen in Befehlen und Meldungen

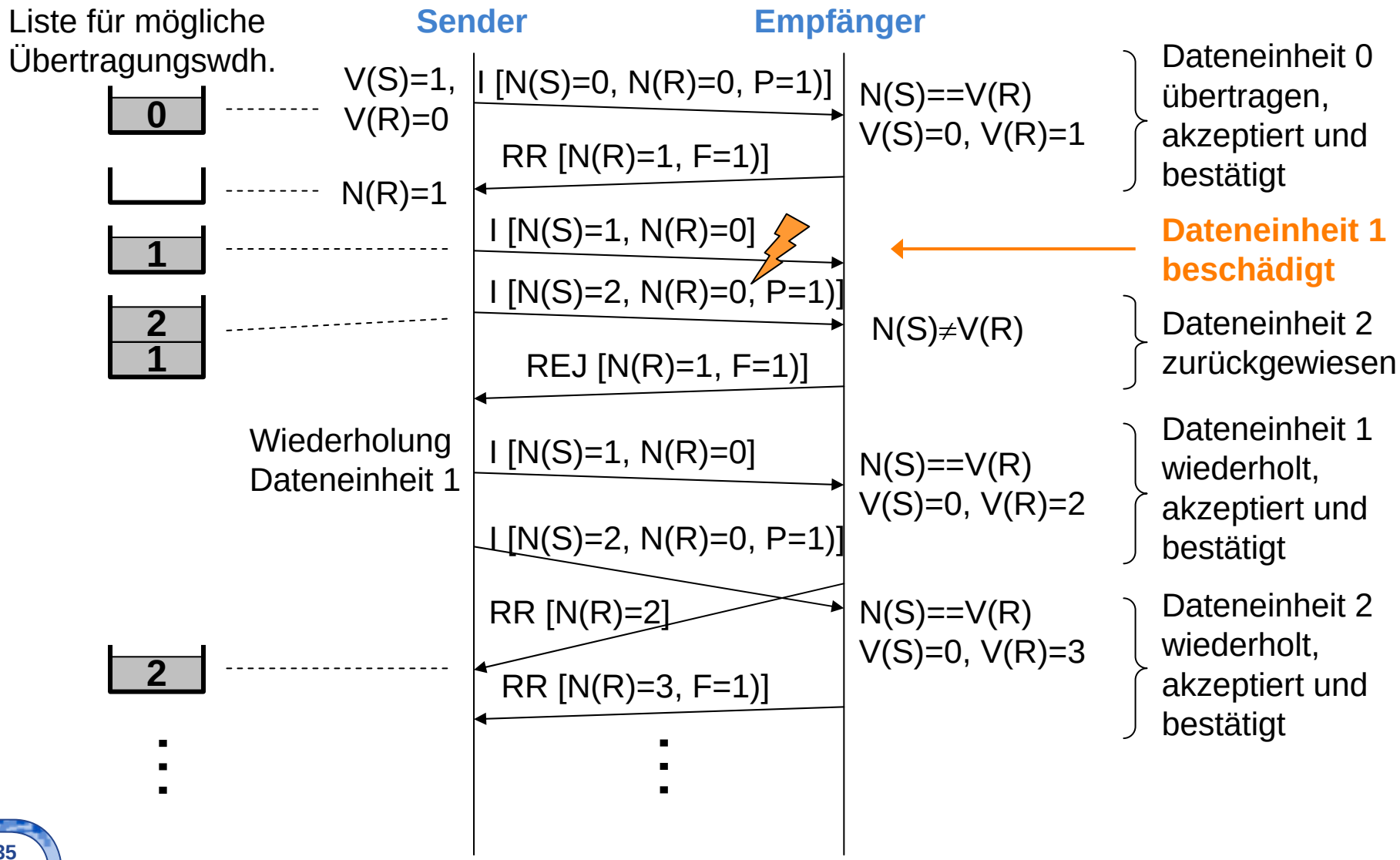
# Datentransfer

- Für die Ablaufsteuerung während des Datentransfers sind die folgenden vier S-Dateneinheiten wichtig
  - RR-Dateneinheit (*Receive Ready*), Kodierung 00 (Bit 3 und 4)
    - Station gibt Empfangsbereitschaft für I-Dateneinheit bekannt (Befehl oder Meldung)
    - Gleichzeitig werden schon empfangene I-Dateneinheiten durch N(R) bestätigt
    - Leitstation erteilt durch RR-Befehl Sendeaufruf an Folgestation
  - RNR-Dateneinheiten (*Receive Not Ready*), Kodierung 01
    - Station gibt bekannt, dass sie zur Zeit keine I-Dateneinheiten empfangen kann
    - Aufhebung dieses Zustands durch RR, REJ, UA oder SABM

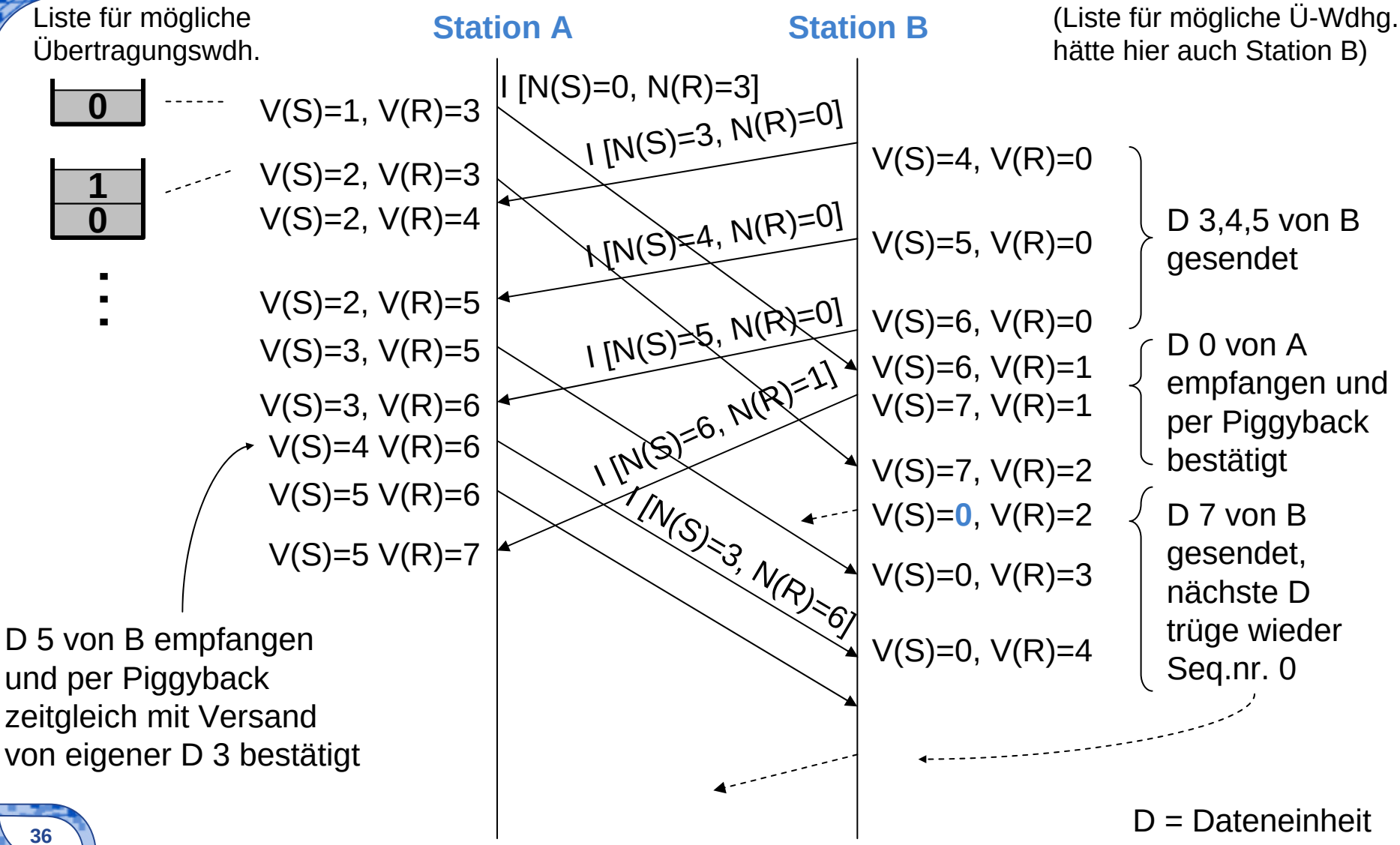
# Datentransfer

- REJ-Dateneinheiten (*Reject*), Kodierung 10
  - Anforderung der Wiederholung von I-Dateneinheiten bei Fehler
  - Bestätigung I-Dateneinheiten mit N(S) <= N(R)-1 und Anforderung der Dateneinheiten mit N(S) >= N(R)
  - Ablaufunterbrechung der normalen Datenübertragung
    - Wird beim Empfang der I-Dateneinheiten mit N(S) = N(R) der REJ-Dateneinheiten wieder aufgehoben
- SREJ-Dateneinheiten (*Selective Reject*), Kodierung 11
  - Gezielte Wiederholung der I-Dateneinheit mit N(S) = N(R) der SREJ-Dateneinheit
    - z.B. bei Bitfehler in I-Dateneinheit

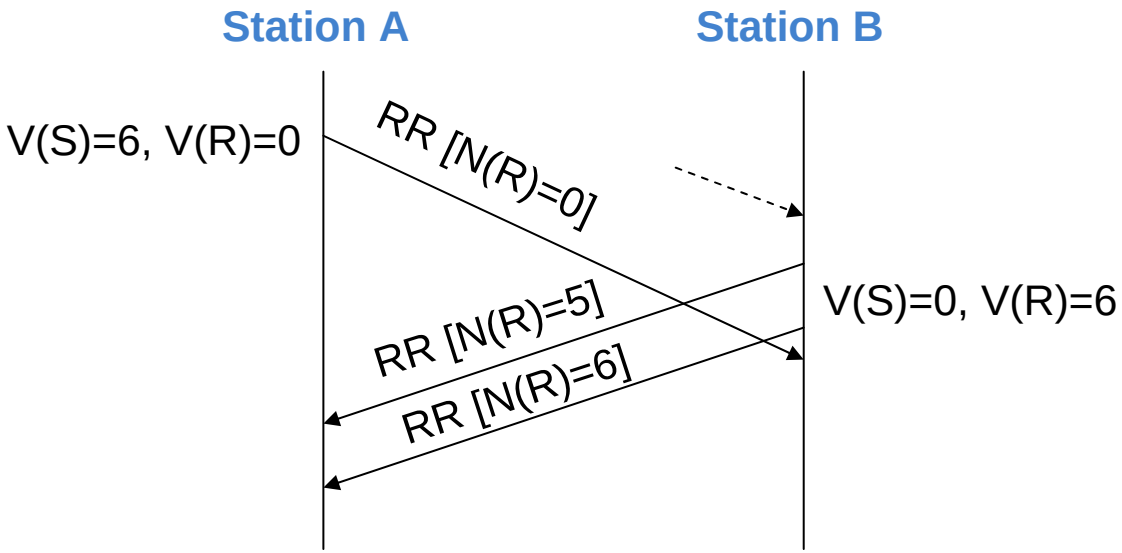
# 5.5.3 Beispiel: NRM ohne Piggyback ACKs



# Beispiel: ABM mit Piggyback ACKs



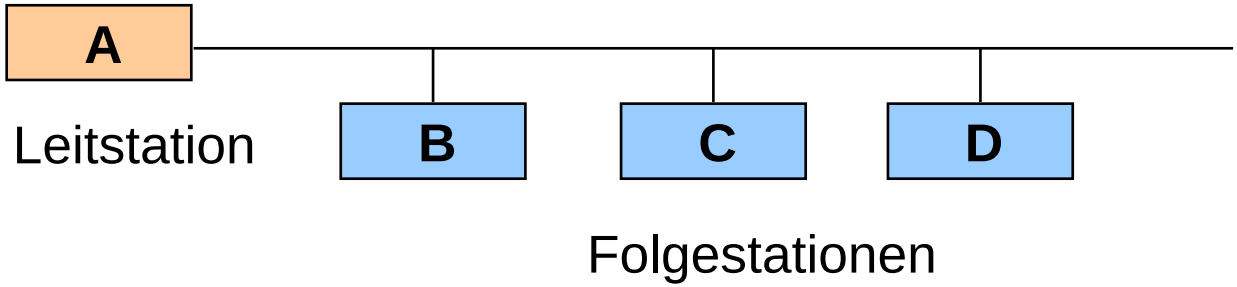
# Beispiel: ABM mit Piggyback ACKs (Fortsetzung)




- Flusskontrolle
  - Übertragung pausiert, sobald V(S) = letztes N(R) empfangen + K
    - mit K = maximale Fenstergröße
      - hier: 7, aufgrund 3 Bit breiter Sequenznummer (modulo 8)
  - Wird jeweils nur in einer Richtung angewendet
    - Unabhängig von anderer Richtung
  - S- und U-Dateneinheiten fließen unabhängig von der Flusskontrolle

# 5.5.4 Beispiel: Mehrere Folgestationen

- Beispielkonfiguration

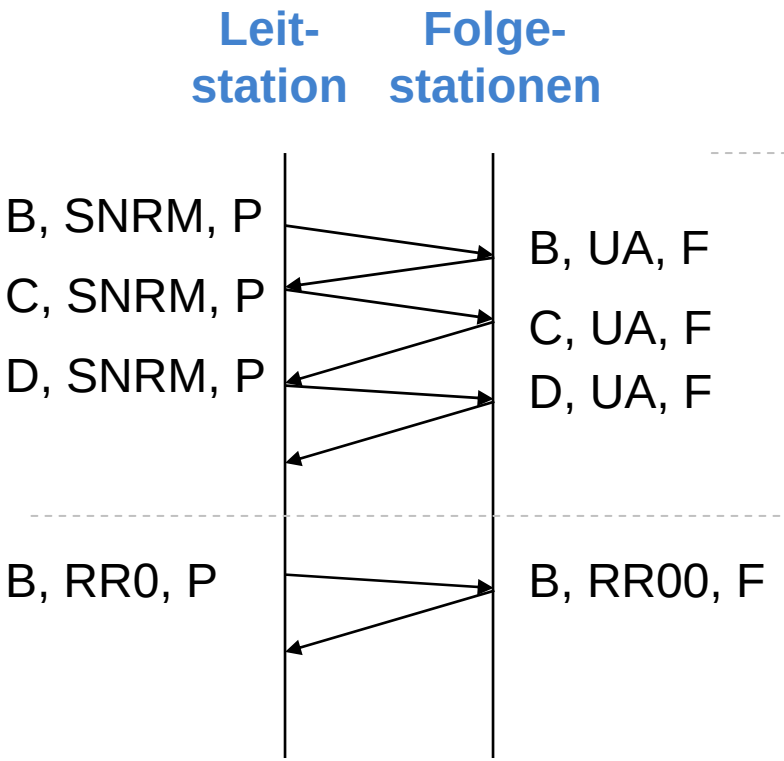



  - Das folgende Beispiel stellt den Ablauf von HDLC bei ungestörtem Betrieb dar
- Folgende Notation wird verwendet: X, Y sr, Z
  - X Adresse des Ziels bzw. Senders
  - Y Kurzbezeichnung Befehl / Meldung
  - s aktueller Wert der Sendefolgenummer
  - r aktueller Wert der Empfangsfolgenummer
  - Z Poll (P)- oder Final (F)-Bit auf 1
    - Keine Erwähnung, wenn PF nicht gesetzt

  } falls vorhanden

  - Beispiel
    - X, Y sr, Z → A, SNRM, P

# Fortsetzung Beispiel

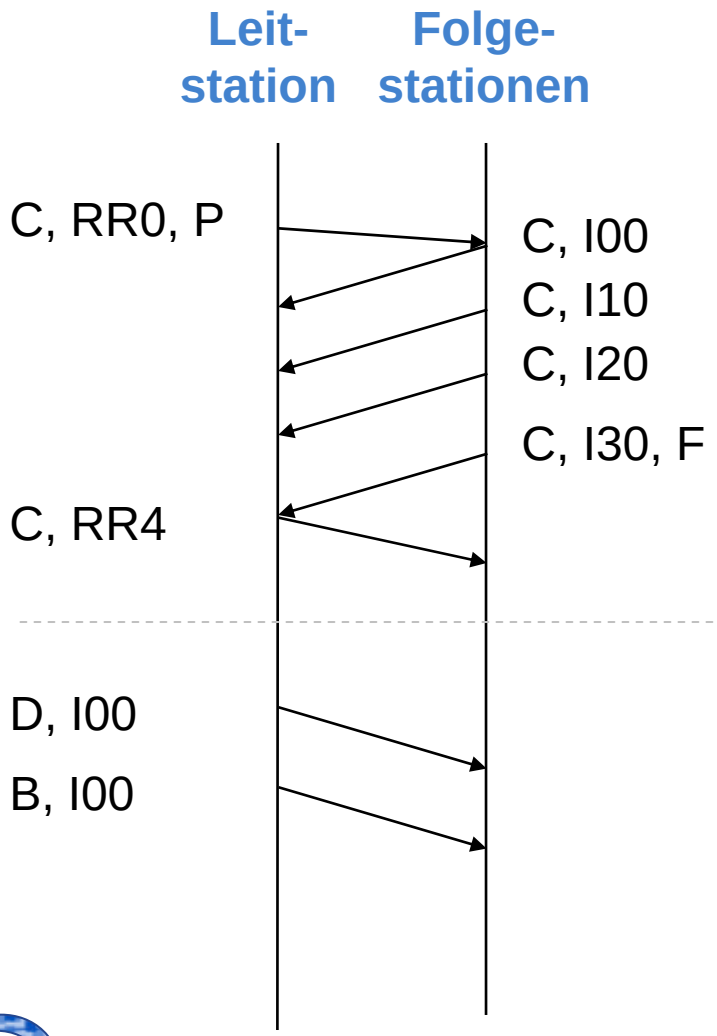



- Zu Anfang: Disconnected Mode

- B,SNRM,P und B,UA,F
  - Aktivierung der Folgestationen B,C,D durch Leitstation A
  - … ebenso für C, D

- B,RR0,P und B,RR0,F
  - Sende-Aufruf der Station B (Polling), kein Sendebedarf von B

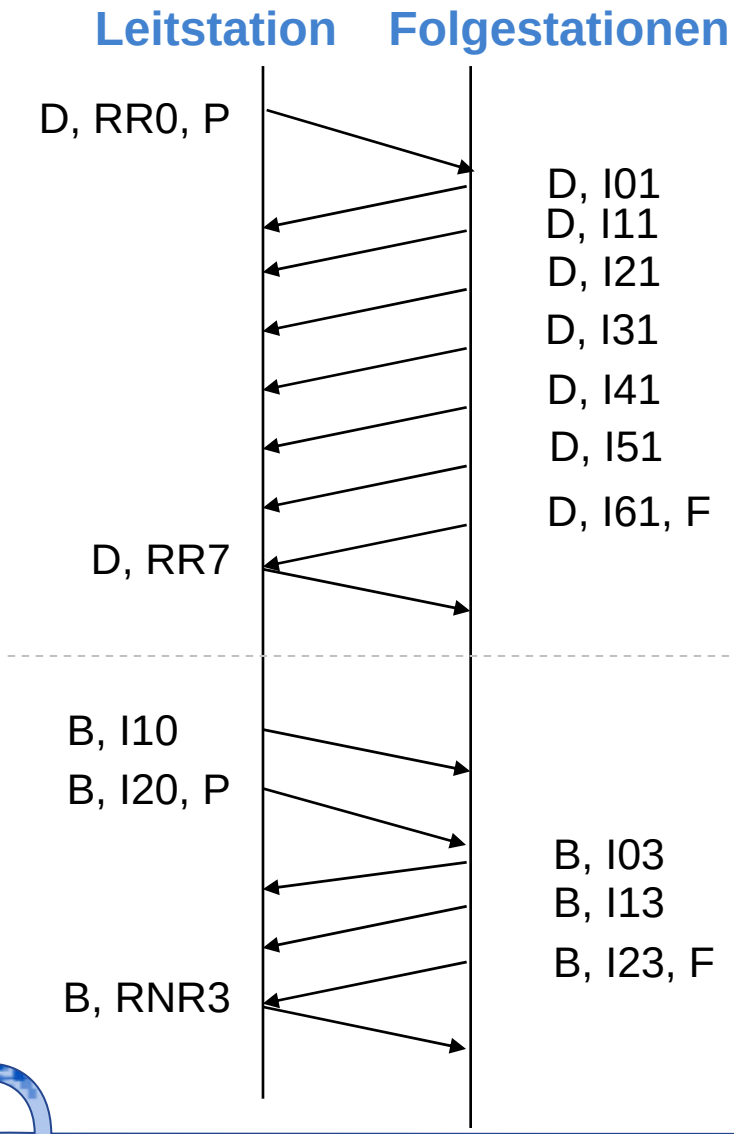
# Fortsetzung Beispiel




- C,RR0,P und C,I00 ... C,I30,F
  - Sende-Aufruf C, Station C überträgt 4 I-Dateneinheiten
- C,RR4
  - Leitstation bestätigt Empfang der 4 I-Dateneinheiten, gibt aber keine Sendeerlaubnis für C

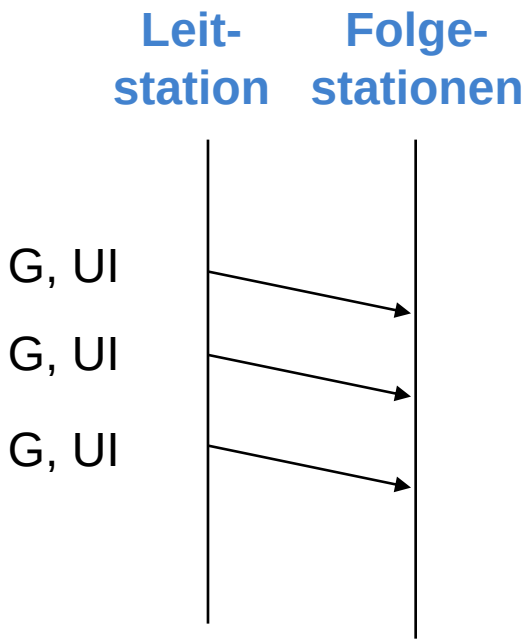
- D,I00 und B,I00
  - Empfangs-Aufruf und Übergabe I-Dateneinheit an D,B

# Fortsetzung Beispiel



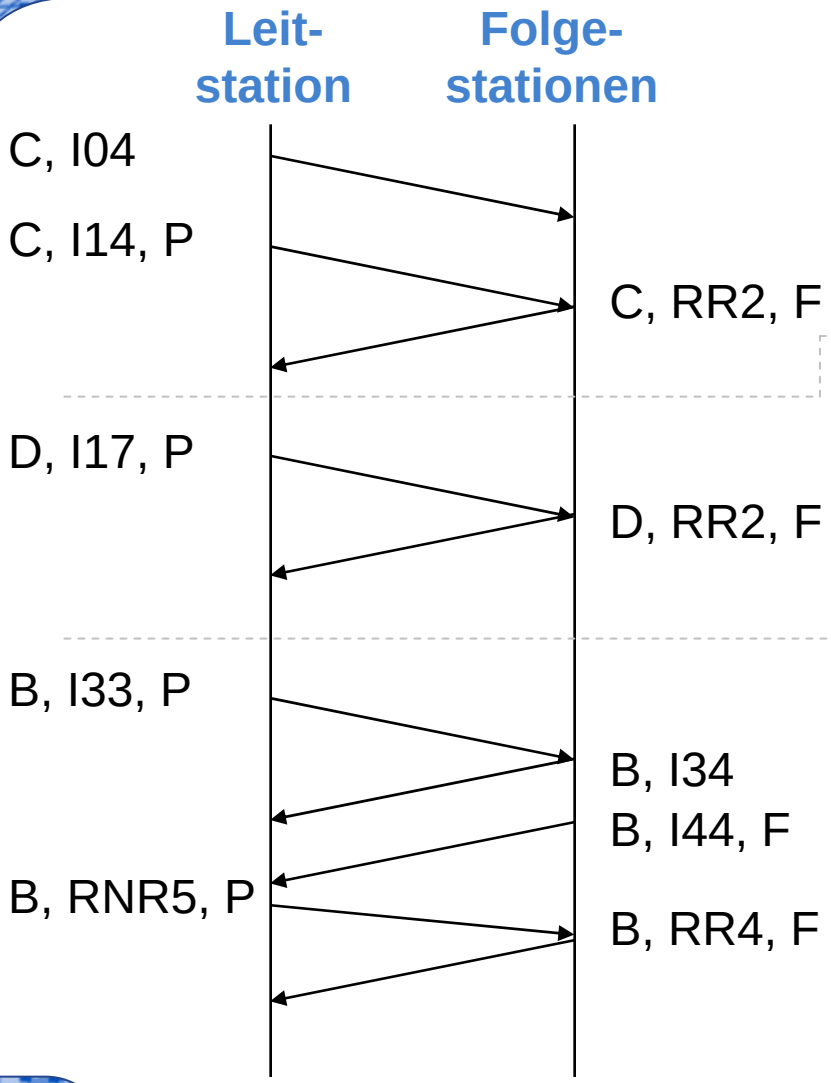

- D,RR0,P
  - Leitstation pollt D
- D,I01 ... D,I61,F
  - D antwortet mit 7 I-Dateneinheiten (Maximalzahl) und quittiert gleichzeitig permanent die I-Dateneinheit für D (siehe vorangegangene Folie)
- D,RR7
  - Bestätigung der 7 I-Dateneinheiten von D
- B,I10 und B,I20,P sowie B,I03 ... B,I23,F
  - Leitstation übermittelt zwei I-Dateneinheiten an B sowie einen Sendeaufruf
  - B antwortet mit 3 I-Dateneinheiten
- B,RNR3
  - Leitstation bestätigt 3 I-Dateneinheiten und erklärt sich als nicht-übernahmebereit für weitere Dateneinheiten von B

# Fortsetzung Beispiel



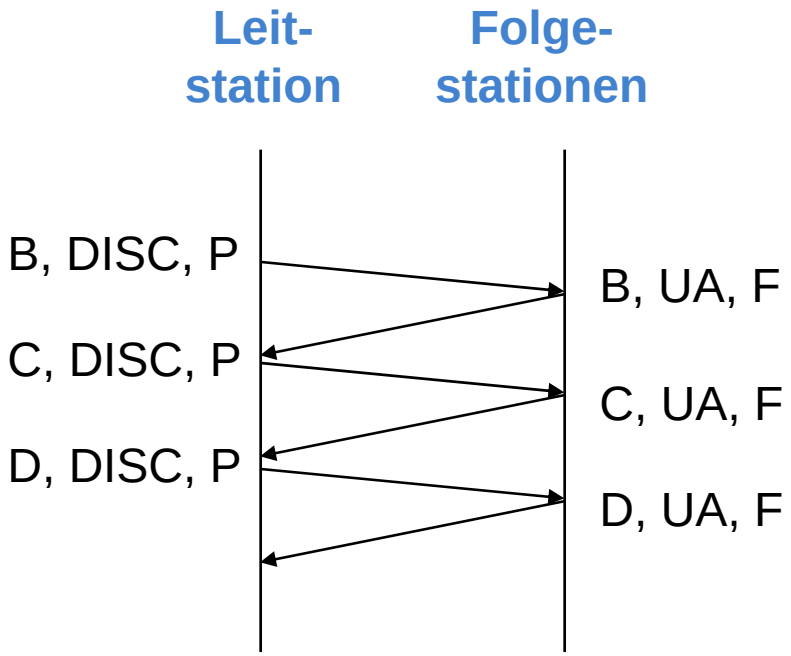

- G,UI
  - Rundrufe mit unnummerierter Information (UI) an alle Stationen über globale Adresse G

# Fortsetzung Beispiel




- C,I04 und C,I14,P und C,RR2,F
  - Zwei I-Dateneinheiten an C
  - Station C bestätigt, lehnt aber Übernahme des Senderechts ab (kein Sendebedarf)
- D,I17,P und D,RR2,F
  - I-Dateneinheit an D
  - Station D bestätigt, lehnt aber Senderecht ab
- B,I33,P und B,I34 und B,I44,F
  - I-Dateneinheit an B
  - B bestätigt während Antwort mit zwei I-Dateneinheiten
- B,RNR5,P und B,RR4,F
  - Leitstation quittiert und hat kein Interesse an weiteren I-Dateneinheiten (RNR) von B

# Fortsetzung Beispiel



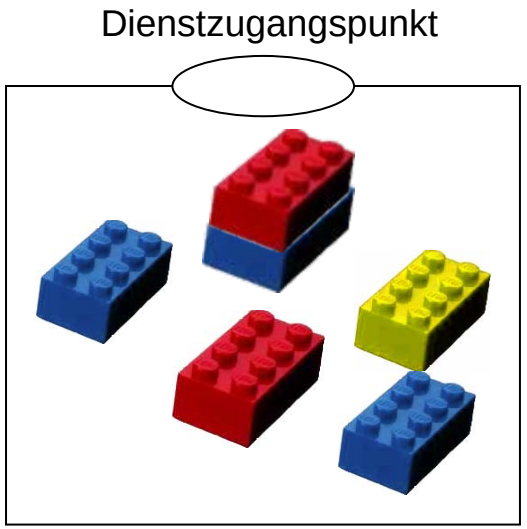

- B,DISC,P und B,UA,F
  - Alle I-Dateneinheiten quittiert
  - Leitstation beendet Verbindungen, setzt *Disconnect-Mode*
  - … ebenso für C, D

# 5.5.5 Varianten von HDLC

- HDLC wird heute in vielen Umgebungen eingesetzt, mit jeweils angepassten Protokollvarianten
- Einige Varianten
  - Link Access Procedure Version B (LAP B)
    - Punkt-zu-Punkt Duplex Übertragungsabschnitte (z.B. X.25)
    - Asynchronous Balanced Mode (ABM) mit kombinierten Leit-/Folgestationen
    - Keine selektiven Rejects (SREJ)
  - Link Access Procedure Version M (LAP M)
    - Für den Einsatz mit Modems konzipiert
    - Akzeptiert asynchrone Daten, überträgt diese dann synchron
  - Link Access Procedure Version D (LAP-D)
    - Zwei Dienste werden angeboten: unzuverlässig und zuverlässig
    - Spezielle Adressinformation
    - Kein Master; Busstruktur
  - PPP-Framing [RFC 1662]
    - Zeichenorientiert, Flag 01111110
    - 16 oder 32-Bit-CRC
    - UI-Dateneinheiten, keine Kontrolldateneinheiten

# Zusammenfassung

- HDLC als konkretes Beispiel eines „einfachen“ Protokolls
  - Bestandteil der Sicherungsschicht
  - Beispiel zur Kombination von Protokollmechanismen als Bausteine eines Protokolls
  - Populäres Protokoll, zu dem viele Varianten existieren
- Zeigt die Komplexität eines scheinbar einfachen Problems



# Übungen

5.1 Erläutern Sie die Aufgaben der Sicherungsschicht.

5.2 Nennen Sie die unterschiedlichen Konfigurationen und Betriebsarten von HDLC, sowie den Ablauf letzterer.

5.3 Wie ist eine HDLC-Dateneinheit aufgebaut?

5.4 Was bedeutet Codetransparenz und wie wird diese Eigenschaft bei HDLC erreicht?

5.5 Welche unterschiedlichen HDLC-Dateneinheiten gibt es und wodurch werden diese identifiziert?

5.6 Was ist das P/F-Bit und wozu dient es?

5.7 Wie läuft im Normal Response Mode eine Datenübertragung samt Verbindungsaufbau und -abbau statt?

5.8 Was versteht man unter „Piggyback ACKs“?

5.9 Erläutern Sie, wie zwei Folgestationen über HDLC miteinander Daten austauschen können.

# Literatur


[Blac91] U. Black; X.25 and Related Protocols; IEEE Computer Society Press, Feb. 1991

- Kapitel 4 behandelt u.a. HDLC und LAPB

[Carl80] D. E. Carlson; Bit-Oriented Data Link Control Procedures; IEEE Transactions on Communications, Vol. 28, No. 4, April 1980

[Gree83] P. Green (Ed.); Computer Network Architectures and Protocols; Plenum Press, 1983

- Kapitel 4: D. E. Carlson; Bit-Oriented Data Link Control

[ISO02] ISO/IEC 13239:2002; Information Technology – Telecommunications and information exchange between systems – High-level data link control (HDLC) procedures; Aug. 2002

- Fasst seit 1996 ältere Standard-Versionen zusammen und löst diese ab
  - Beispielsweise DIN ISO 3309, 4335, 7809

[Stal06] W. Stallings; Data & Computer Communications; Prentice Hall, 2006

- Kapitel 7.3

# Kommunikation und Datenhaltung

## 6. Beschreibungsmethoden für Dienste und Protokolle




**Prof. Dr. Martina Zitterbart**
Dipl.-Inform. Martin Röhricht
[zit | roehricht]@tm.uka.de



# Kapitelübersicht

1. Einführung
2. Physikalische Grundlagen
3. Protokollmechanismen
4. Geschichtete Architekturen
5. Sicherungsschicht: HDLC
6. Beschreibungsmethoden
7. Sicherungsschicht: Lokale Netze
8. Netzkopplung und Vermittlung
9. Die Transportschicht
10. Anwendungssysteme
11. Middleware

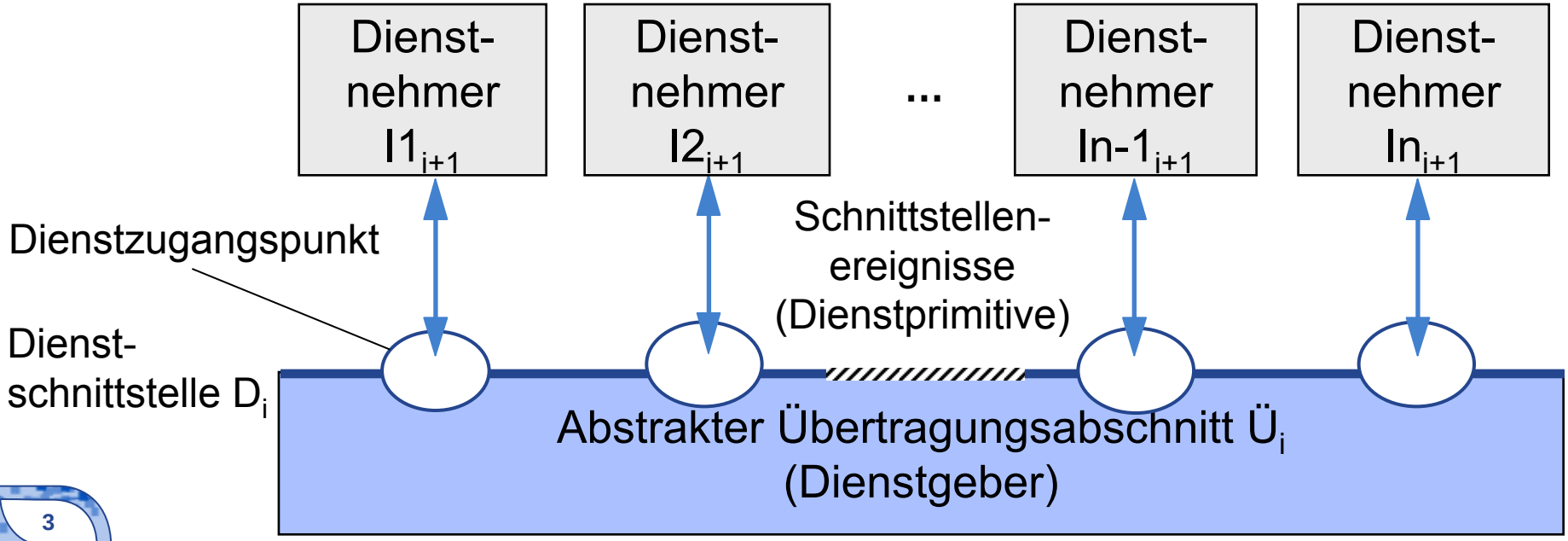
6.1 Dienste
6.2 Weg-Zeit-Diagramme
6.3 Zustandsübergangsdiagramme
6.4 Protokollfunktionalität

# Motivation

- … wo stehen wir in der Vorlesung?
  - … eine Reihe von Bausteinen für die Kommunikation aus Kapitel 3
  - … grundlegende Architekturbetrachtungen aus Kapitel 4
  - … ein Beispielprotokoll (HDLC) aus Kapitel 5
    - Informell beschrieben
    - Formale Beschreibung eines Protokolls?
- Ziel dieses Kapitels
  - Techniken zur formalen Beschreibung von Diensten und Protokollen
    - … bilden Basis für Verifikation

# 6.1 Dienste: Dienstnehmer / Dienstgeber

- Dienstnehmer und Dienstgeber
  - Dienstprimitive zwischen Dienstnehmer und Dienstgeber
    - Entspricht vertikaler Kommunikation zwischen Schichten
  - Inanspruchnahme von Diensten über Dienstzugangspunkte
- Beschreibung eines Dienstes
  - Verhalten an der Dienstschnittstelle
  - Abstrakter Übertragungsabschnitt bildet eine elementare Instanz



# Abstrakte Sicht eines Dienstes

- Dienstschnittstelle
  - Menge räumlich verteilter Dienstzugangspunkte
  - Schnittstellenereignisse treten an Dienstzugangspunkten auf
    - Dargestellt über Dienstprimitive
- Abstrakte Sicht eines Dienstes
  - Entspricht Sicht auf seine Dienstschnittstelle
    - Menge von Dienstzugangspunkten
    - Menge von Dienstprimitiven
      - Zeitpunkt
      - Zugangspunktadresse (d.h. Ort)
      - Typ
    - Ablauffestlegungen

# Dienstprimitive

- Zusammensetzung der Beschreibung von Dienstprimitiven aus drei Bestandteilen
  - Name
    - Z.B. Bitübertragung, Sicherung, Vermittlung, Transport
      - Engl.: Physical, Data Link, Network, Transport
    - Oft in abgekürzter Form (z.B. Ph, DL, N, T)
    - Kann auch entfallen, wenn nur ein Dienst betrachtet wird
  - Name einer Dienstleistung
    - Z.B. Connect, Data, Release, Reset, Abort, Disconnect
    - Oder Abkürzung: Con, Dat, Rel, Res, Abo, Dis
  - Bezeichnung der Grundtypen von Dienstprimitiven
    - Request (Req)
      - Beauftragung (Dienstnehmer → Dienstgeber)
    - Indication (Ind)
      - Benachrichtigung des Partners (Dienstgeber → Dienstnehmer)
    - Response (Rsp)
      - Beantwortung durch Partner (Dienstnehmer → Dienstgeber)
    - Confirmation (Cnf)
      - Benachrichtigung über Abschluss (Dienstgeber → Dienstnehmer)

# Dienstprimitive

- Zusätzlich: Parameter
- Beispiel
  - TConReq(*Adressen*)
    - Verbindungsaufbauanforderung an der Schnittstelle zum Transportdienst
    - Adressen identifizieren Kommunikationspartner
  - HTTPGet[Req](*URL*)
    - Anforderung der HTML-Seite, die durch *URL* identifiziert wird
- Überblick

| Name | **Dienstleistung** | **Grundtypen** | **Parameter** |
|---|---|---|---|
| Physical (Ph)<br>Data Link (DL)<br>Network (N)<br>Transport (T)<br>HTTP<br>FTP<br>... | Connect (Con)<br>Data (Dat)<br>Release (Rel)<br>Abort (Abo)<br>Provider Abort (PAbo)<br>Disconnect (Dis)<br>... | Request (Req)<br>Indication (Ind)<br>Response (Rsp)<br>Confirmation (Cnf) | *Abhängig vom Dienst* |

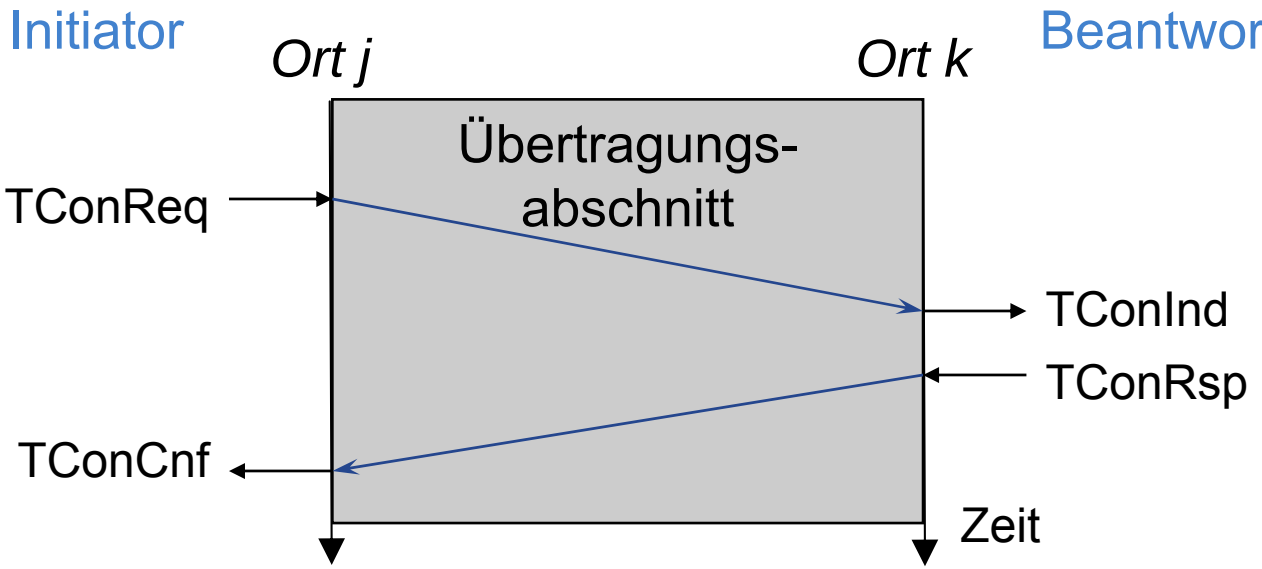
# Beschreibungsmethoden

- Weg-Zeit-Diagramme
  - Zur groben Beschreibung von möglichen Abläufen an Dienstschnittstellen
    - Vgl. Nutzung zur Beschreibung von Abläufen des HDLC-Protokolls in Kapitel 5
- Zustandsübergangsdiagramme
  - Nutzung von Zustandsautomaten
  - Auch genutzt zur Beschreibung von Protokollinstanzen

# 6.2 Weg-Zeit-Diagramme

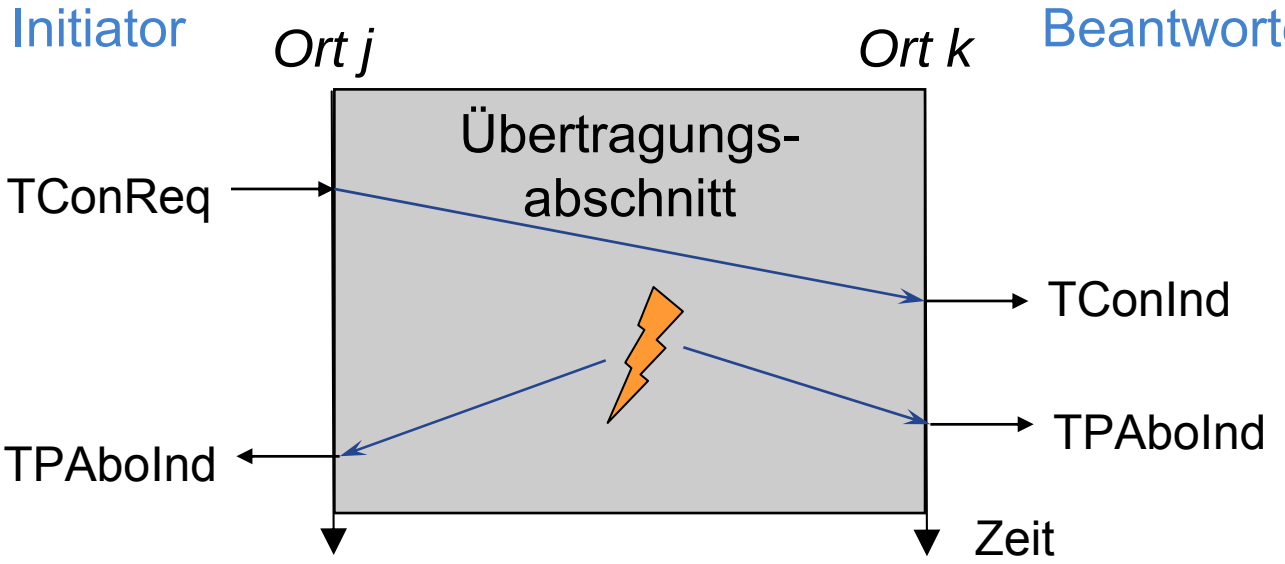
- Aufgabe
  - Darstellung räumlich verteilter Abläufe
  - Grafische Darstellung innerhalb eines 2-Achsen-Koordinatensystems
    - Vertikale Achse: Zeitachse
      - Abstrahiert vom tatsächlichen Zeitpunkt
    - Horizontale Achse: Räumliche Distanz
      - Abstrahiert von der tatsächlichen räumlichen Distanz
- Dargestellter Ablauf
  - Menge von Ereignissen (Dienstprimitiven)
    - Auftragen nach Ort und relativem Zeitpunkt

# Beispiel: bestätigter Dienst




- Regulärer Ablauf der bestätigten Dienstleistung „Verbindungsaufbau“
  - Links: Zugangspunkt des Initiators am Ort j
  - Rechts: Zugangspunkt des Beantworters am Ort k
- Darstellung von Alternativen
  - In Weg-Zeit-Diagrammen unübersichtlich
  - Je Sachverhalt separates Diagramm

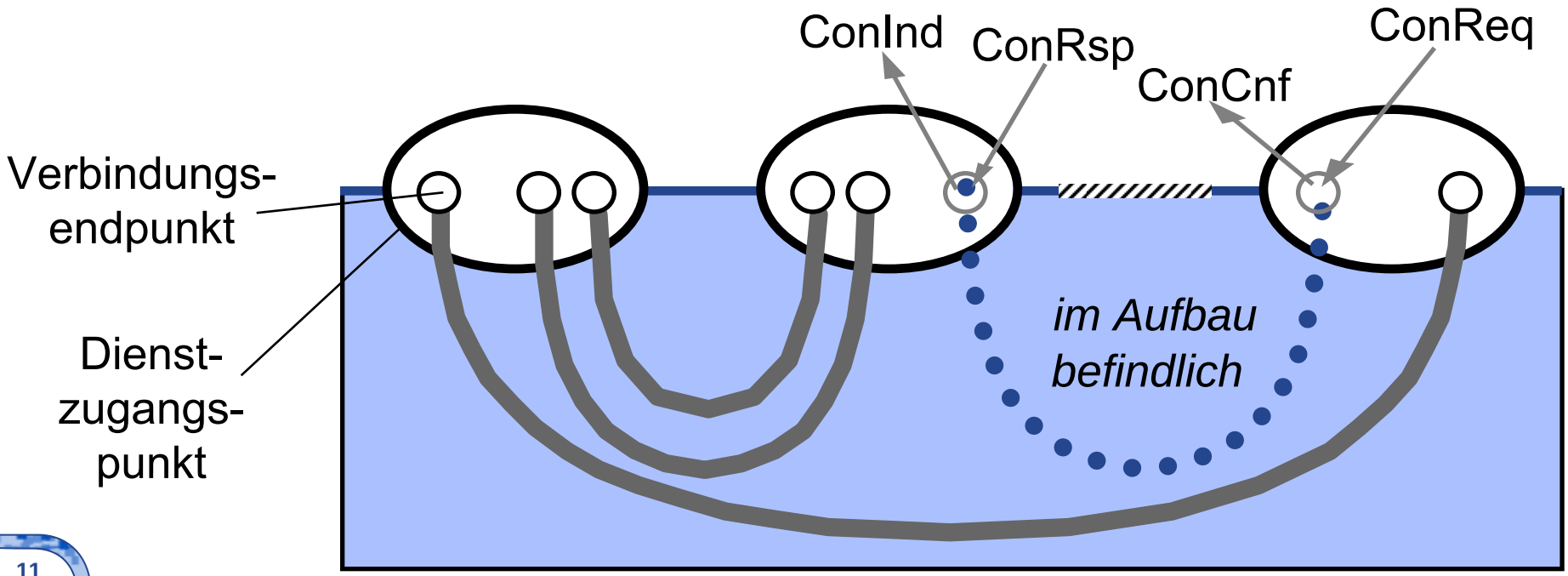
# Beispiel: bestätigter Dienst



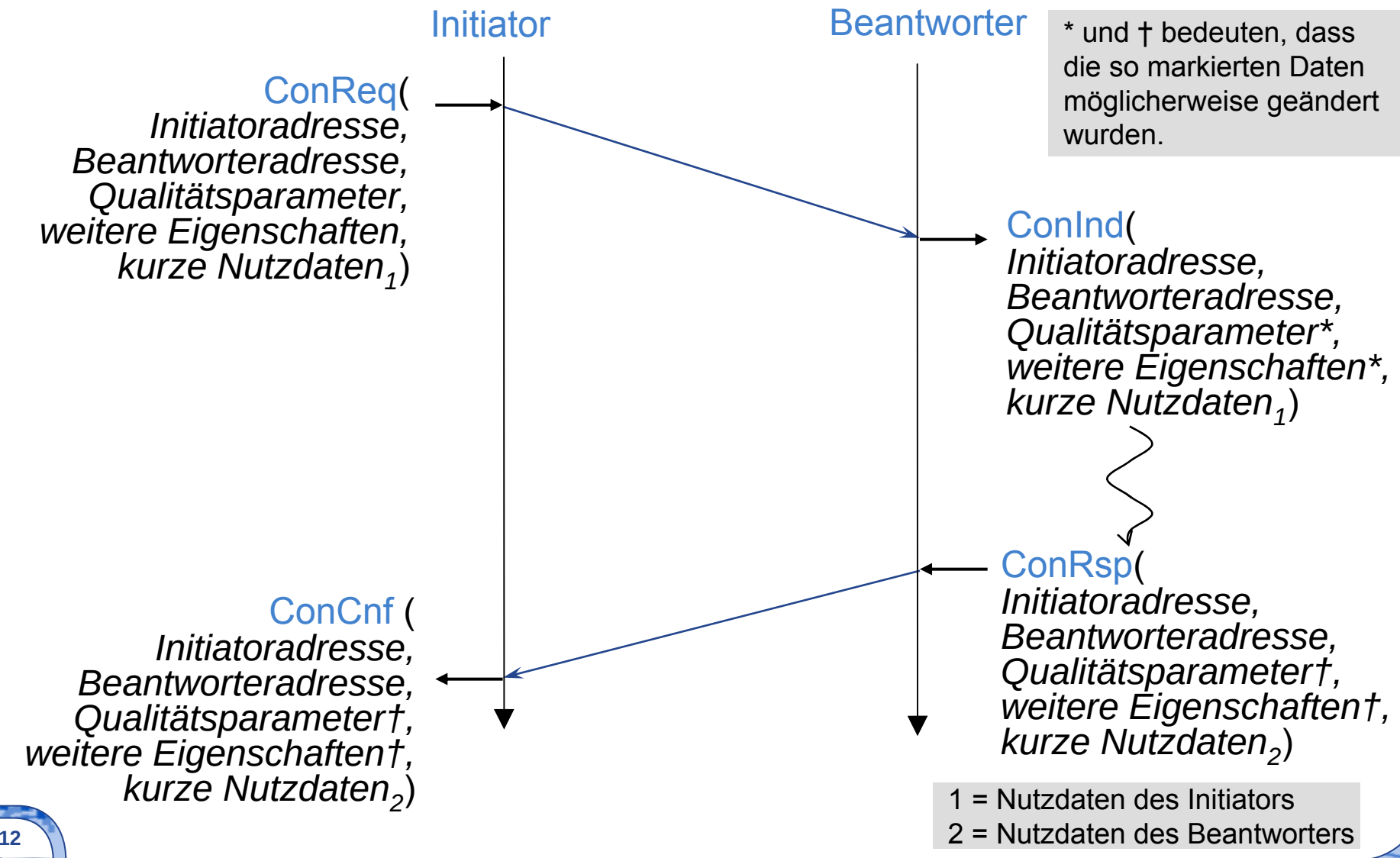

- Alternativer Ablauf
  - Verbindungsaufbau wird an Partner kommuniziert
  - Bearbeitung des Verbindungsaufbaus durch den Dienstgeber (Übertragungsabschnitt) abgebrochen, bevor Partner antworten kann

# Dienstzugangspunkt und Verbindungsendpunkt

- Unterstützung mehrerer Dienstzugangspunkte
  - Dienstzugangspunktadressen identifizieren Benutzer
  - Adressen als Parameter in einigen Dienstprimitiven
- Pro Dienstzugangspunkt mehrere Verbindungen möglich
  - Kennzeichnung durch Verbindungsendpunkte



# Beispiel: Verbindungsaufbau

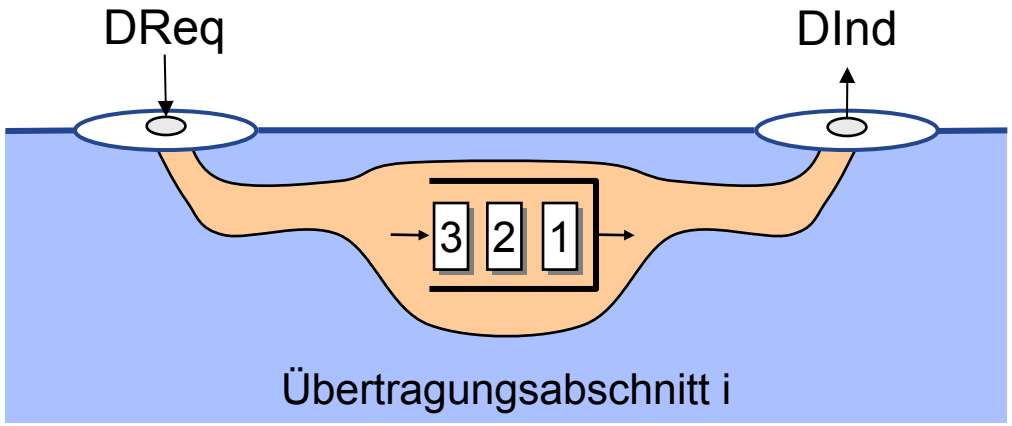



* und † bedeuten, dass die so markierten Daten möglicherweise geändert wurden.

1 = Nutzdaten des Initiators
2 = Nutzdaten des Beantworters

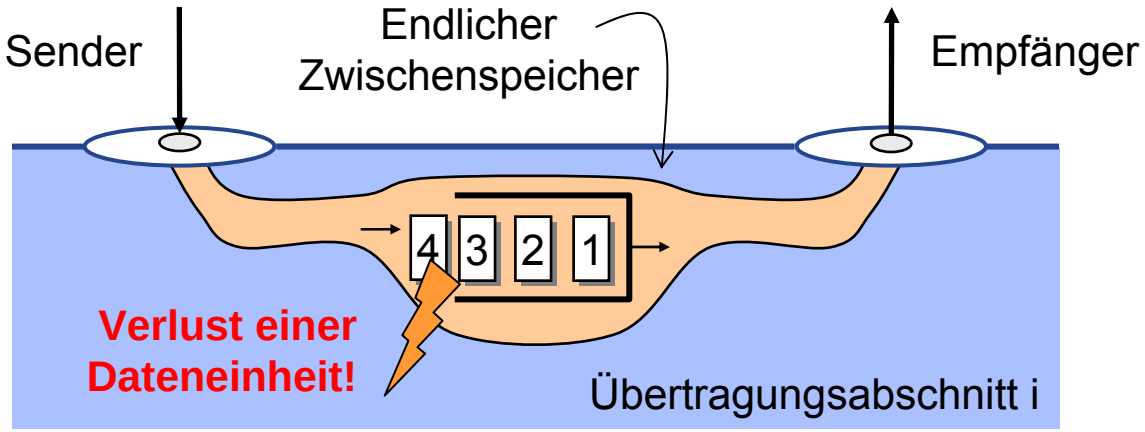
# Datenaustausch: Auslieferungsdisziplin

- Auslieferungsdisziplin
  - Zeitliche Reihenfolge der anstoßenden und daraus resultierenden Dienstprimitive
    - DReq und DInd
  - Varianten
    - treu zur Einlieferungsreihenfolge (FIFO)
    - zufällig
    - FIFO und priorisiert

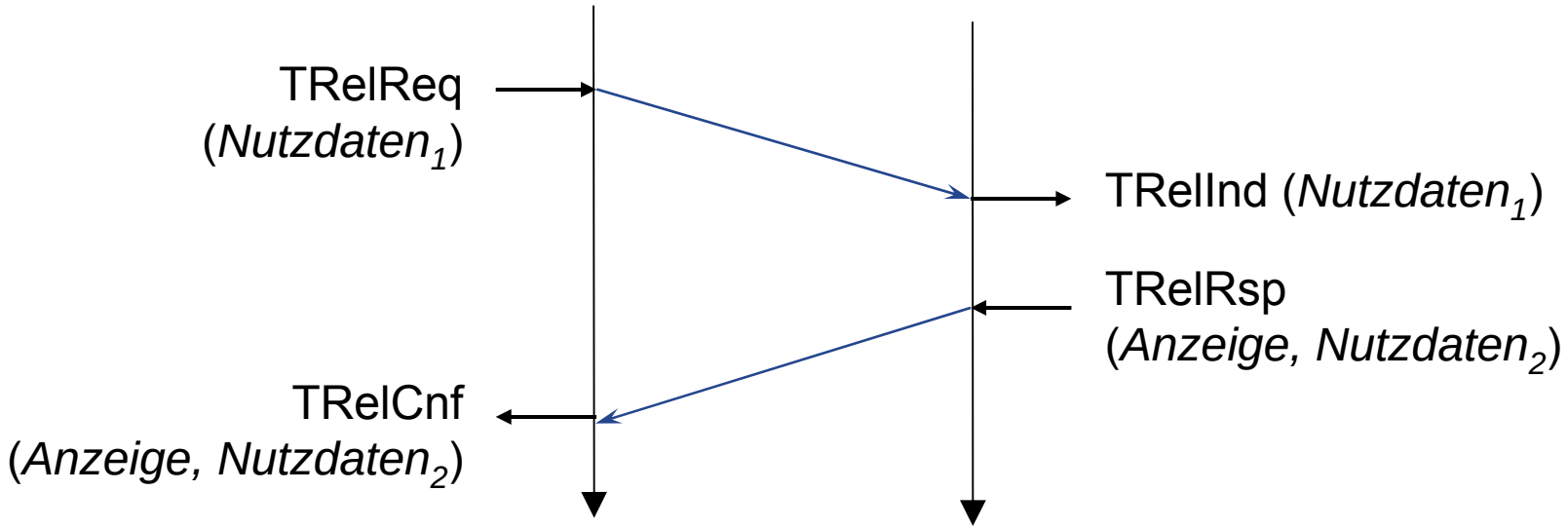


# Datenaustausch: Rückstau

- Rückstau
  - Zwischenpufferung der Daten in zwischenliegenden Systemen des Übertragungsabschnitts
  - Problem: Endlicher Zwischenspeicher
    - Verluste von Dateneinheiten
  - → Staukontrolle löst das Problem



# Beispiel: Verbindungsabbau

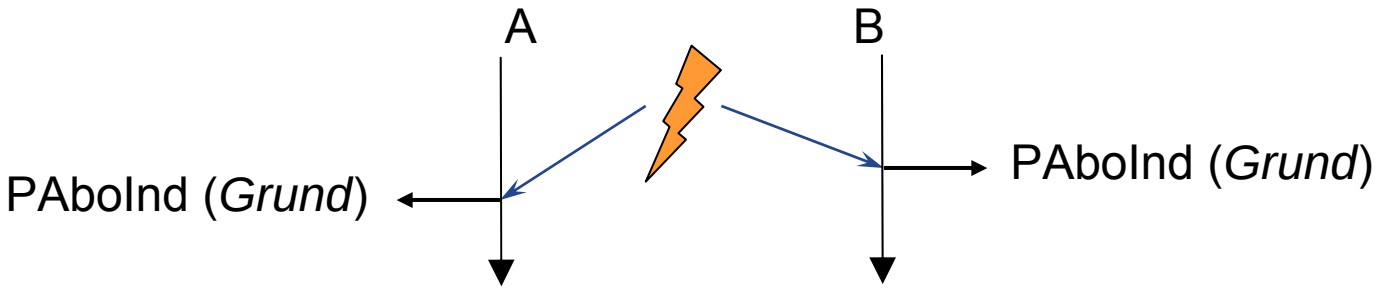



- Geregelter Verbindungsabbau
  - Partner kann Verbindungsabbau zustimmen oder ihn ablehnen

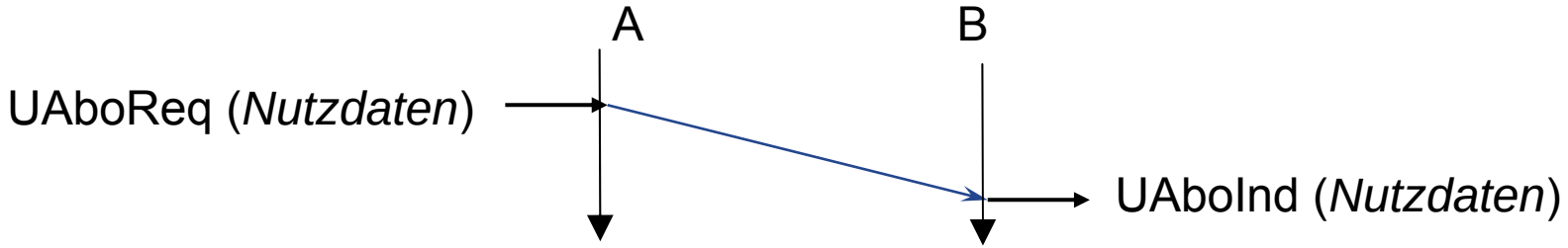
1 = Nutzdaten des Initiators
2 = Nutzdaten des Beantworters

# Beispiel: Verbindungsabbruch

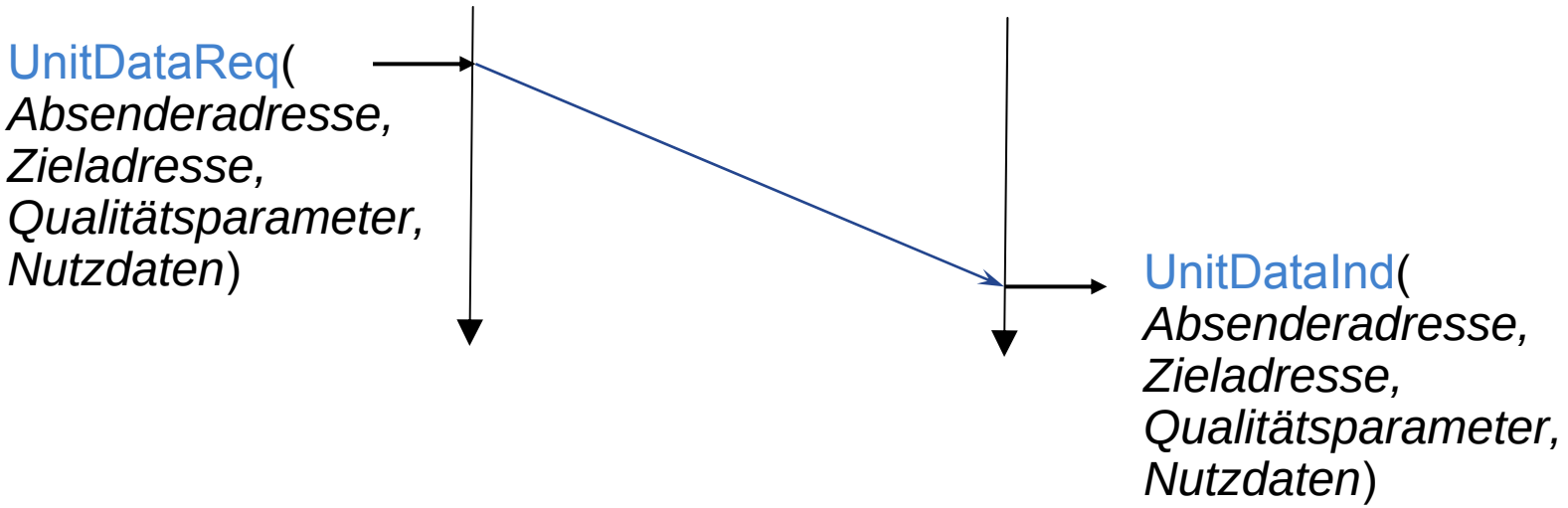
- Außerplanmäßiger Abbruch einer Verbindung
  - Erbringerabbruch (Provider Abort, PAbo)
    - Kann in jedem Zustand einer Verbindung passieren
    - Keine Gleichzeitigkeit / bestimmte Reihenfolge garantiert




  - Nutzerabbruch (User Abort, UAbo)
    - Verbindung gilt für Initiator sofort als abgebrochen



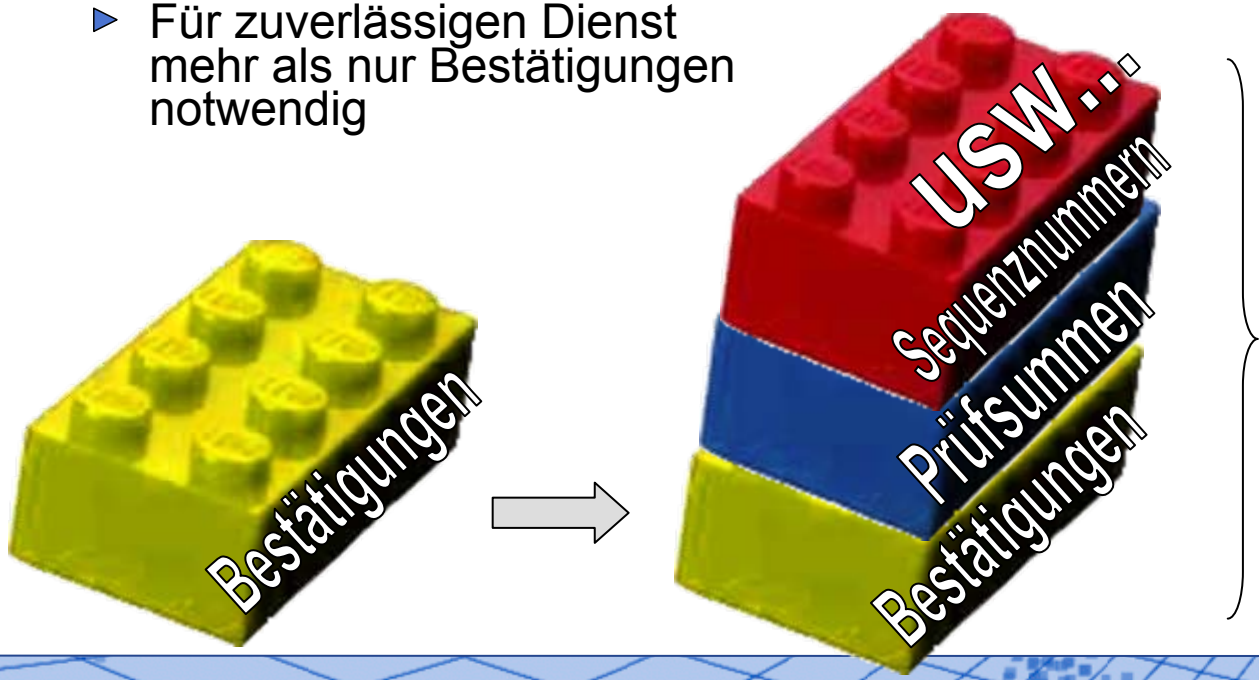
# Beispiel: Datagramm-Dienste




- Keine Verbindungen zwischen Kommunikationspartnern
  - Kein Zusammenhang zwischen verschiedenen Übertragungsleistungen
  - Unterstützt keine Auslieferungsdisziplin
    - z.B. keine Garantie für Reihefolgetreue
- Datagramm-Dienst realisiert unzuverlässige Dienstleistung
  - keine Aushandlung zwischen Kommunikationspartnern

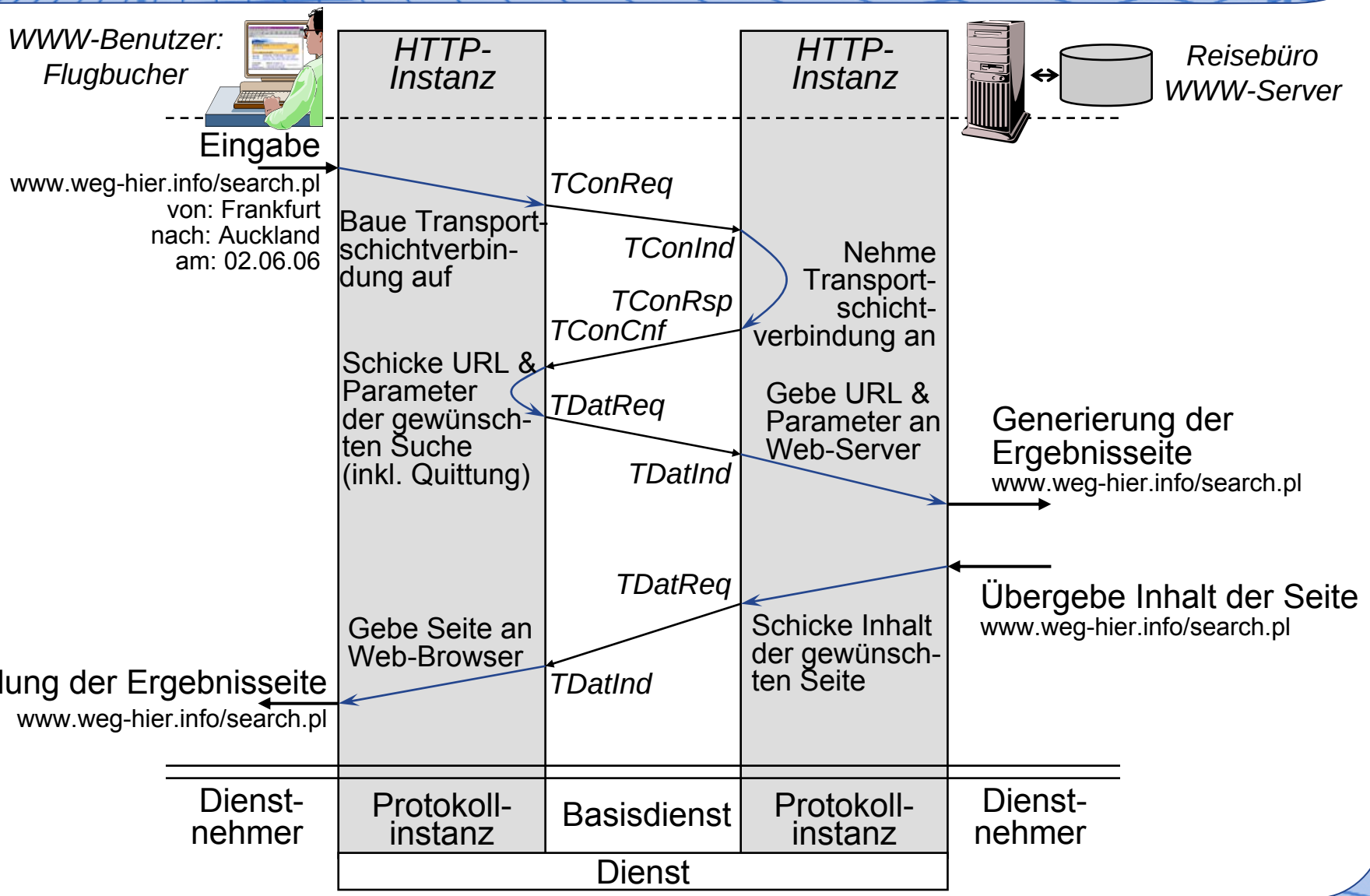
# Einschub Diensteigenschaften: Bestätigt vs. Zuverlässig

- Bestätigt ≠ Zuverlässig
  - Zuverlässiger Dienst stellt sicher, dass alle Daten korrekt übertragen wurden (Wiederholung Kapitel 3)
    - Alle Daten korrekt
    - In der richtigen Reihenfolge
    - Ohne Duplikate
    - Ohne Phantom-Dateneinheiten
  - Bestätigungen nur ein Baustein für zuverlässige Dienste
    - Für zuverlässigen Dienst mehr als nur Bestätigungen notwendig




Auszug aus möglichem Aufbau eines zuverlässigen Dienstes

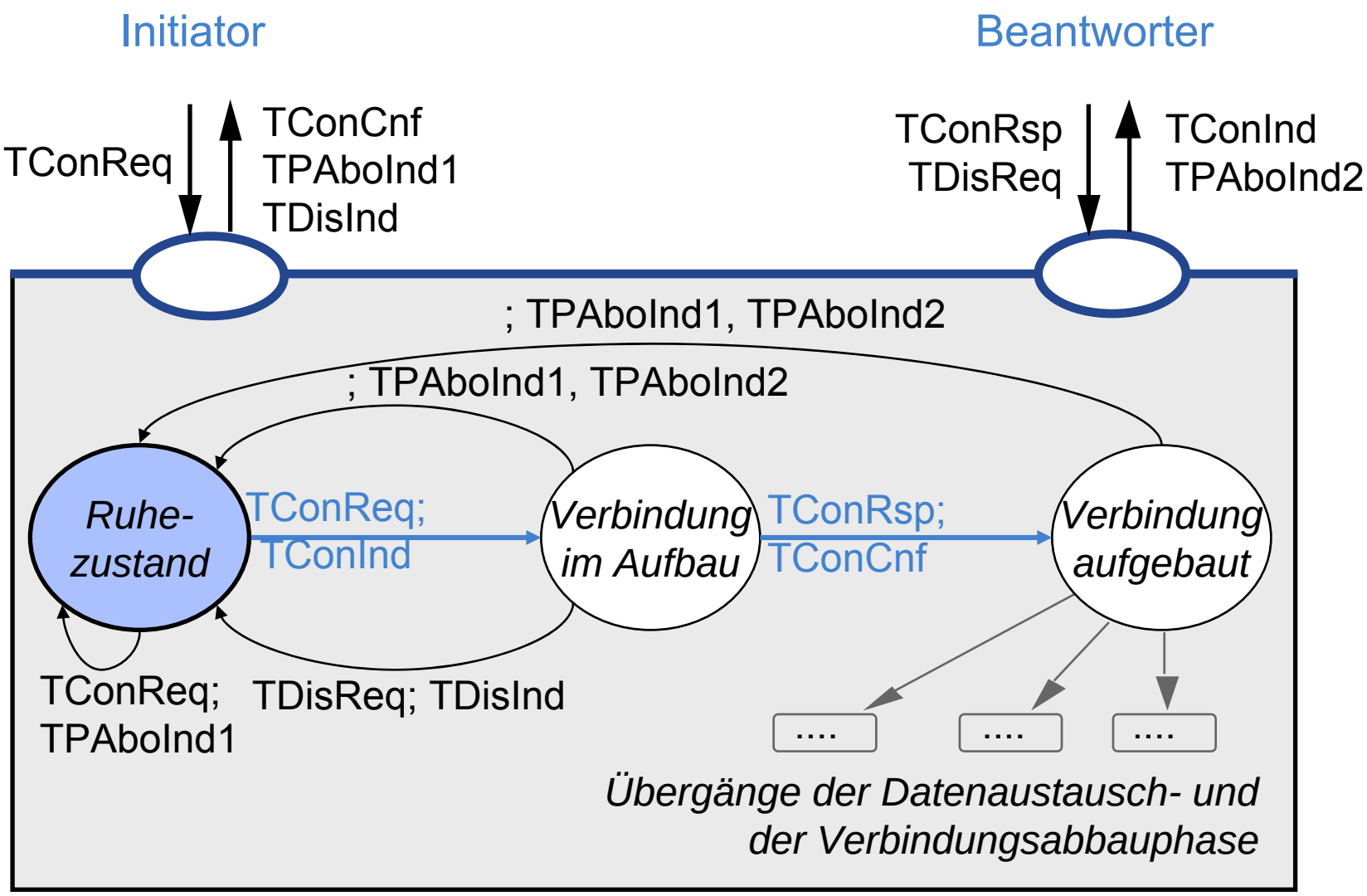
# Anwendungsbeispiel: Fluganfrage im Internet



# 6.3 Zustandsübergangsdiagramme

- Zustandsautomaten
  - Können Verhalten des Übertragungsabschnittes vollständig beschreiben
- Graphisches Hilfsmittel
  - Zustandsübergangsdiagramme
    - Zustand
      - als Kreise notiert
    - Zustandsübergang
      - Pfeil vom aktuellen Zustand in den Nachfolgezustand
      - Beschriftung: Diensttypen Stimulus und Reaktion
        - Z.B. Stimulus; Reaktion1, Reaktion2
      - Spontanübergänge: Keine Angabe eines Stimulus

# Beispiel Verbindungsaufbau



# Beispiel Verbindungsaufbau

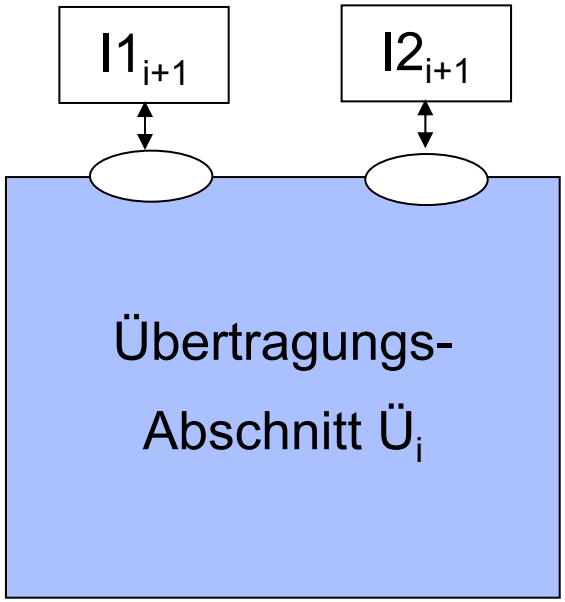
- Dienstprimitive beim Initiator
  - TConReq: Anforderung eines Verbindungsaufbaus
  - TConConf: Verbindungsaufbau erfolgreich
  - TDisInd: Beantworter lehnt Verbindungsaufbau ab
  - TPAboInd1: Abbruch durch Übertragungsabschnitt
- Dienstprimitive beim Beantworter
  - TConRsp: Bestätigung des Verbindungsaufbauwunschs
  - TDisReq: Ablehnung des Verbindungsaufbauwunschs
  - TConInd: Mitteilung des Verbindungsaufbauwunschs
  - TPAboInd2: Abbruch durch Übertragungsabschnitt

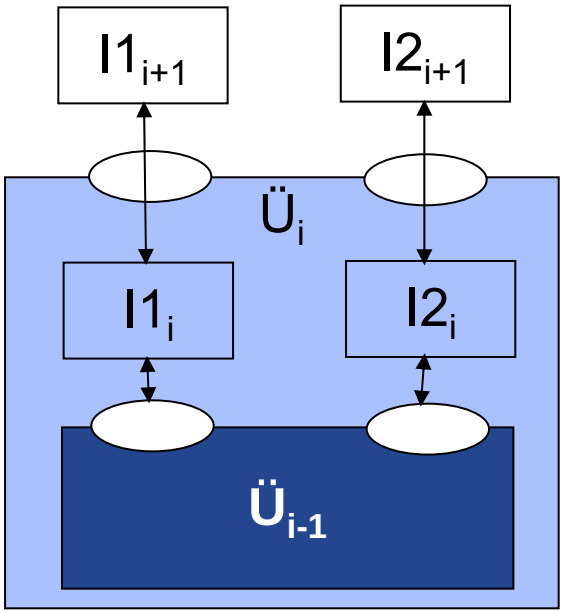
# Beispiel: Verbindungsaufbau

- Verhalten des Übertragungsabschnitts durch Zustandsautomaten repräsentiert
- Drei Zustände
  - Ruhezustand
    - Startzustand
    - Nur TConReq als Stimulus möglich
      - ConInd als Reaktion, falls Medium Wunsch annimmt
      - PAboInd1 als Reaktion, falls Medium Wunsch ablehnt
  - Verbindung im Aufbau
    - ConReq erhalten und mit ConInd reagiert
    - ConRsp erwartet, dann ConConf als Reaktion
  - Verbindung aufgebaut
    - Zustand kann im Beispiel nur durch Spontanübergang verlassen werden

# 6.4 Beschreibung von Protokollfunktionalität

- Abstrakter Übertragungsabschnitt
  - Interne Betrachtung
    - Instanzen, die über untergeordneten Übertragungsabschnitt kommunizieren
- Protokoll
  - Formate und Regeln zur Kooperation der Instanzen einer Schicht
    - Beschreibung der Regeln als Zustandsübergangsdiagramme
    - Formate hier nicht weiter betrachtet



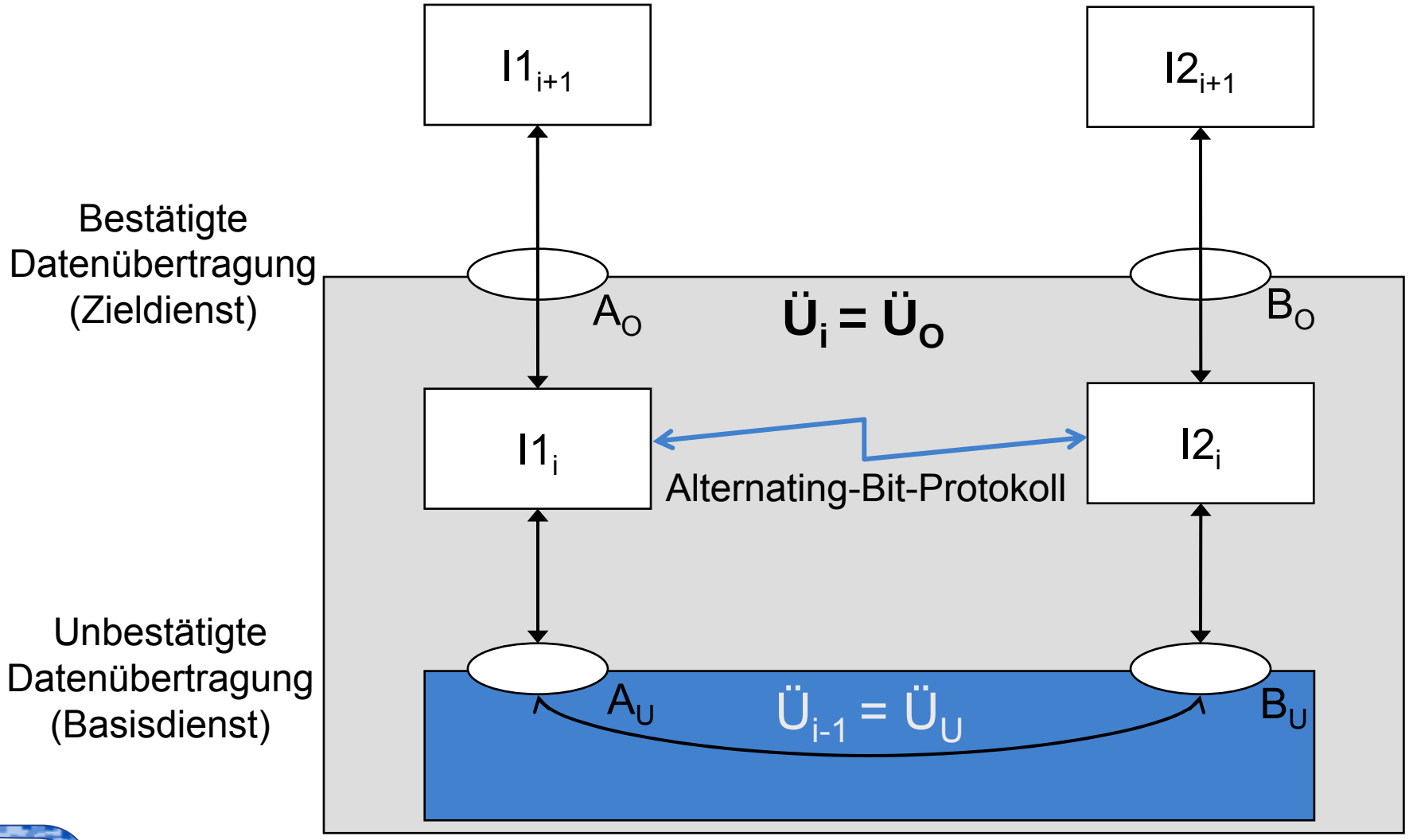
# 6.4.1 Beispiel: Alternating-Bit-Protokoll

- Informelle Beschreibung
  - Nutzung einer unbestätigten Datenübertragung als Basisdienst
  - Bereitstellung einer bestätigten Datenübertragung für den Dienstnutzer
    - Verwendung des ARQ-Verfahrens Stop-and-Wait
      - Senden einer Dateneinheit – Warten auf Quittung
      - Halbduplex-Datenübertragung
        - Mit Quittung können Nutzdaten übertragen werden
      - Sequenznummern alternieren zwischen 0 und 1
        - Unterscheidung zwischen dem Verlust von Dateneinheiten und demjenigen von Quittungen möglich
  - Nachdem eine bestätigte Datenübertragung vollständig beendet ist (Quittung erhalten) geht Instanz in einen Ruhezustand über, bis die nächste Übertragungsanforderung eintrifft
    - Zwei Zustände: Ruhezustand und Warten auf Response
  - Dienstprimitive
    - DtReq, DtInd, DtRsp, DtCnf

# Beispiel: Alternating-Bit-Protokoll

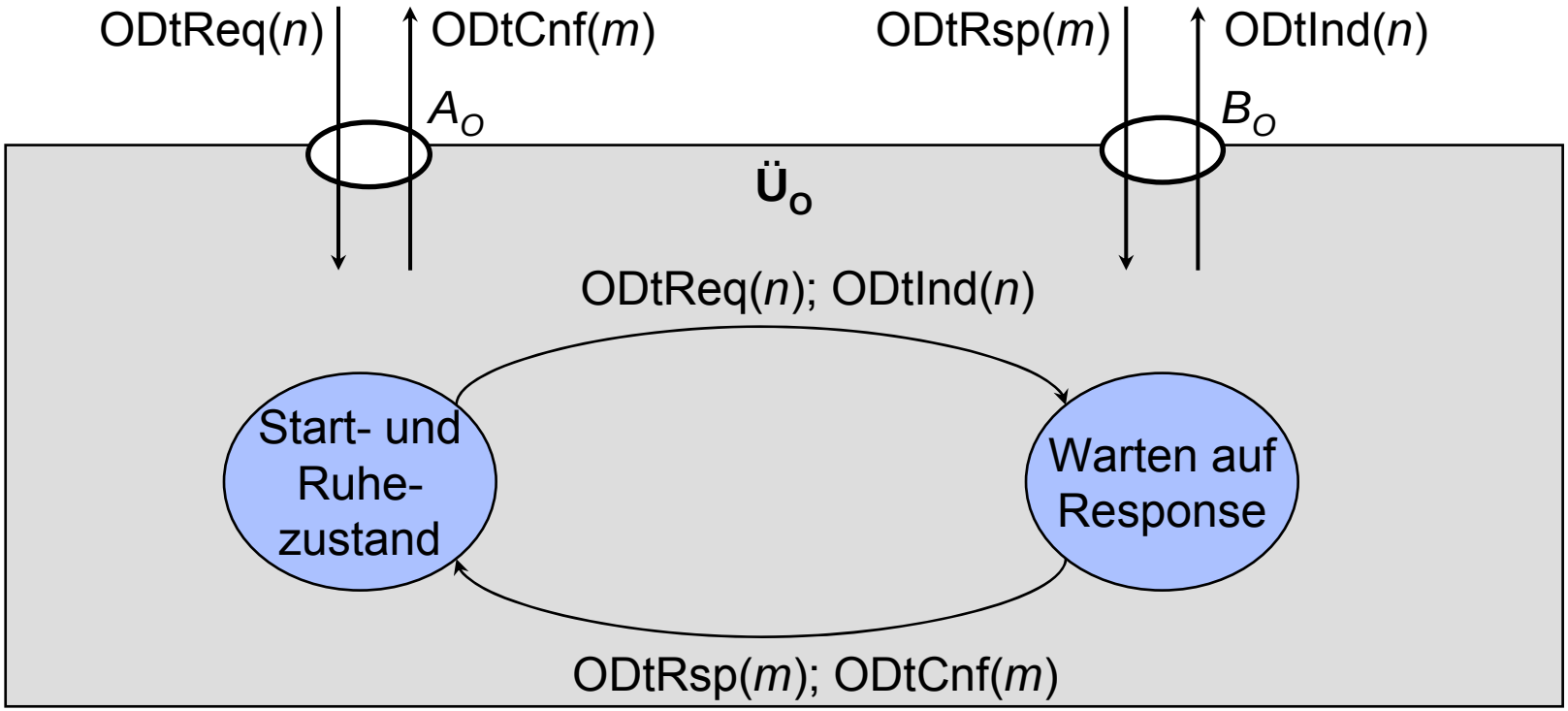
- Vereinfachungen
  - Zieldienst O
    - Es darf sich immer nur eine Dienstleistung in Bearbeitung befinden
      - Eine Dienstleistung entspricht dem Senden einer Dateneinheit
      - Für praktischen Einsatz in vielen Fällen ineffizient (vgl. Kapitel 3)
    - Fehlerfälle (Verfälschungen etc.) nicht betrachtet
      - Für praktischen Einsatz ist dieses Protokoll daher nicht tauglich
  - Basisdienst U
    - Verfälschungen werden dem Dienstnehmer angezeigt
      - In der Praxis besteht immer eine gewisse Wahrscheinlichkeit für unentdeckte Verfälschungen
    - Dienst ist immer verfügbar

# Übersicht

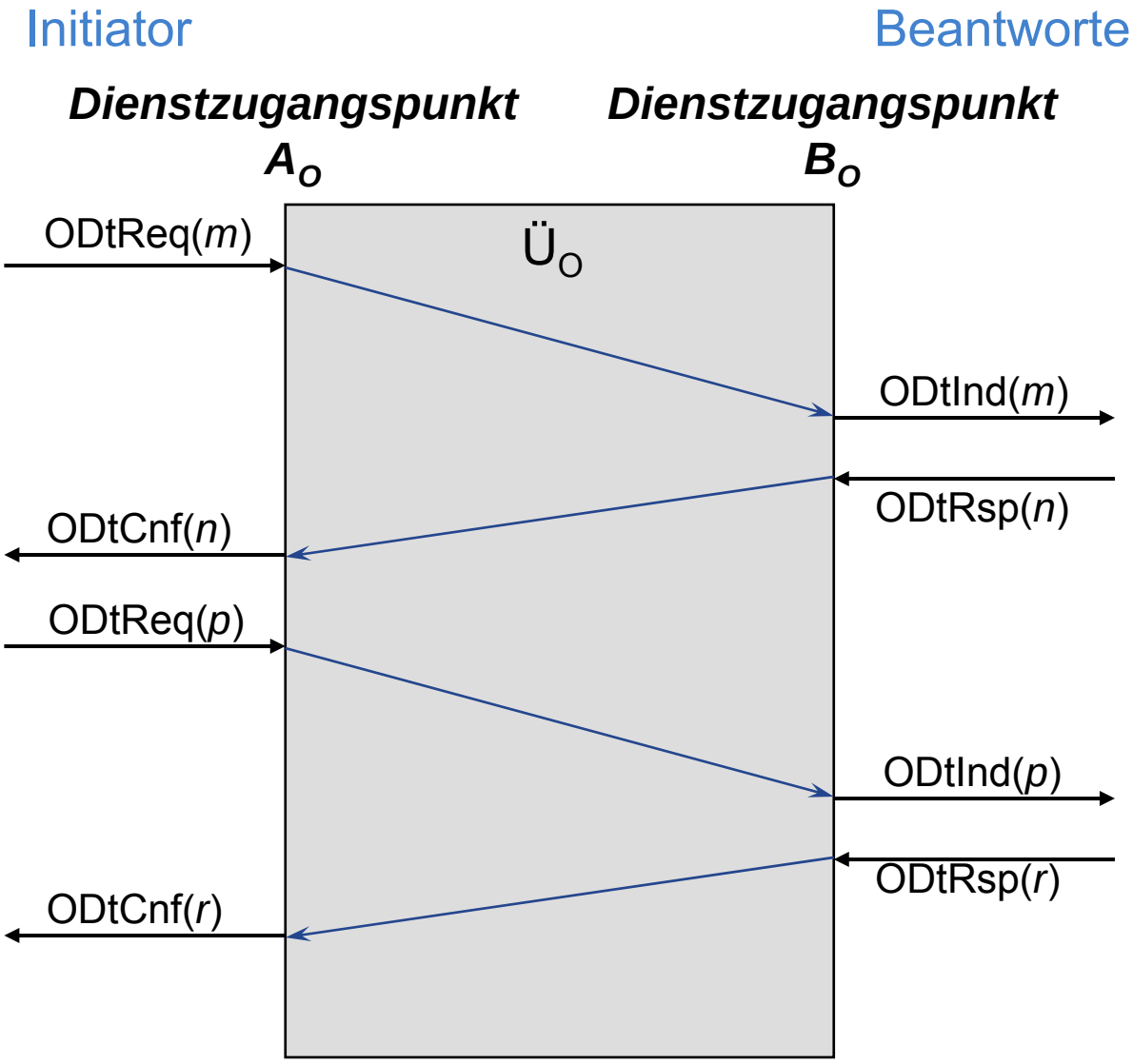

Bestätigte Datenübertragung (Zieldienst)
Unbestätigte Datenübertragung (Basisdienst)
I1i+1
I2i+1
AO
BO
Üi = Üo
I1i
I2i
Alternating-Bit-Protokoll
AU
BU
Üi-1 = ÜU

# Zieldienst O

- Bereitstellung einer bestätigten Datenübertragung
  - Initiator immer am Dienstzugangspunkt $A_O$
  - Beantworter immer am Dienstzugangspunkt $B_O$
  - Parameter n, m: Nutzdaten

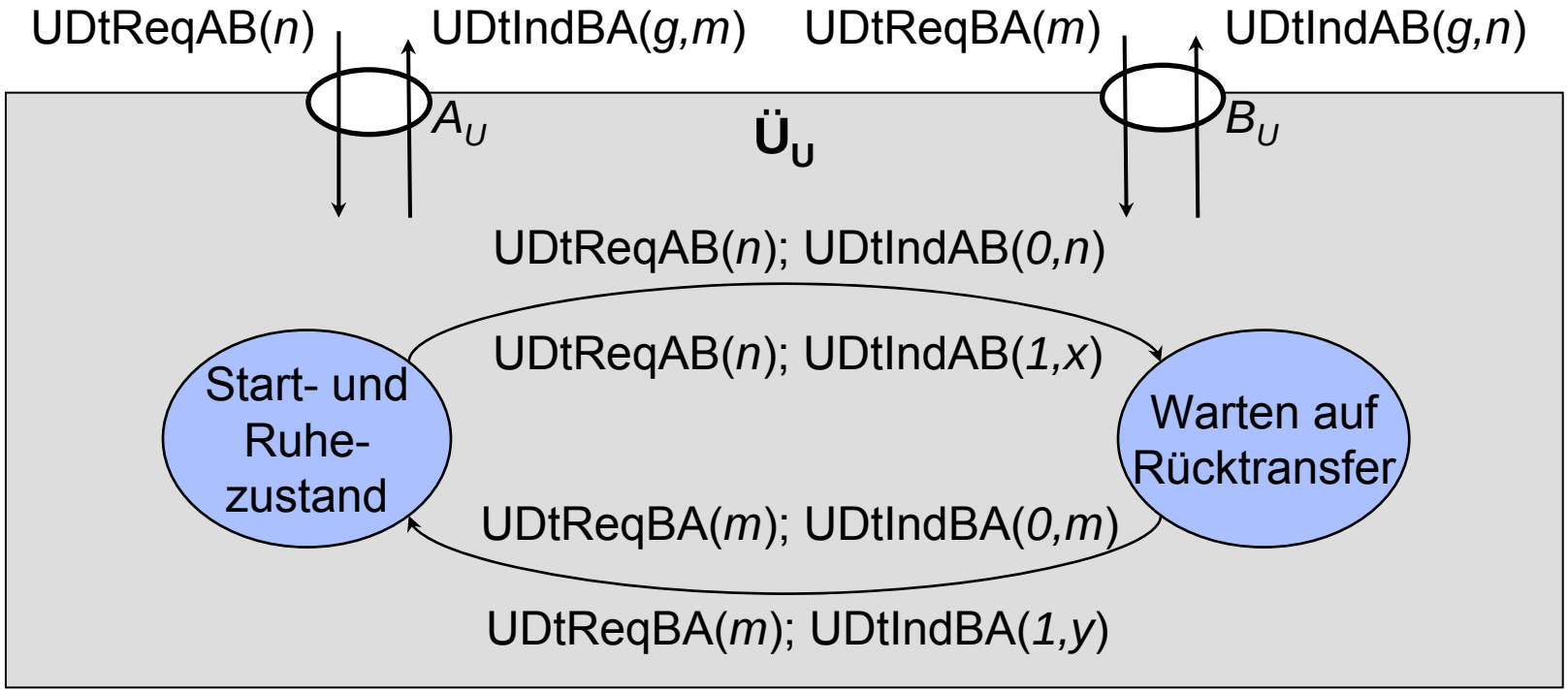


# Dienstablauf in einem Weg-Zeit-Diagramm

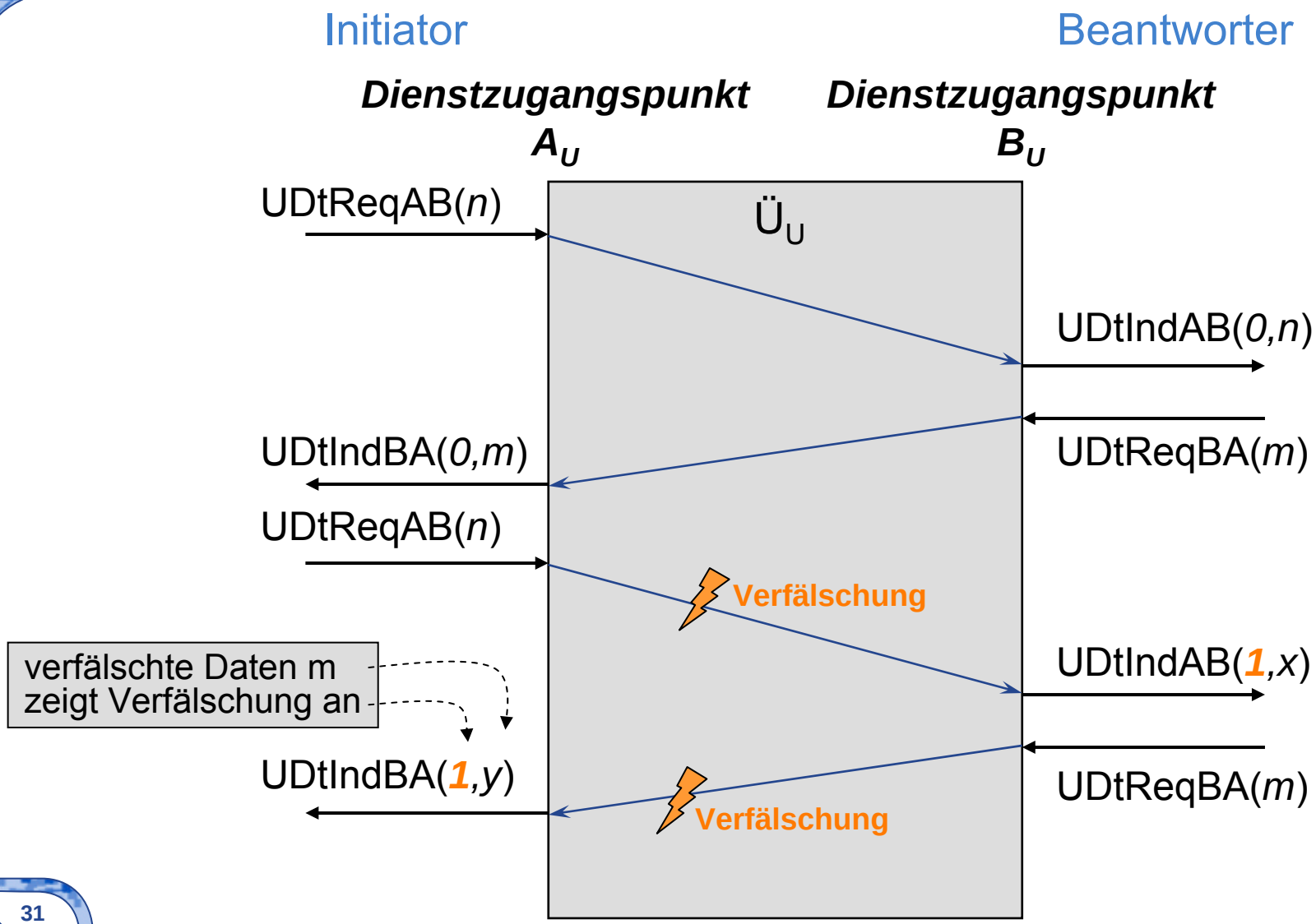


# Basisdienst U

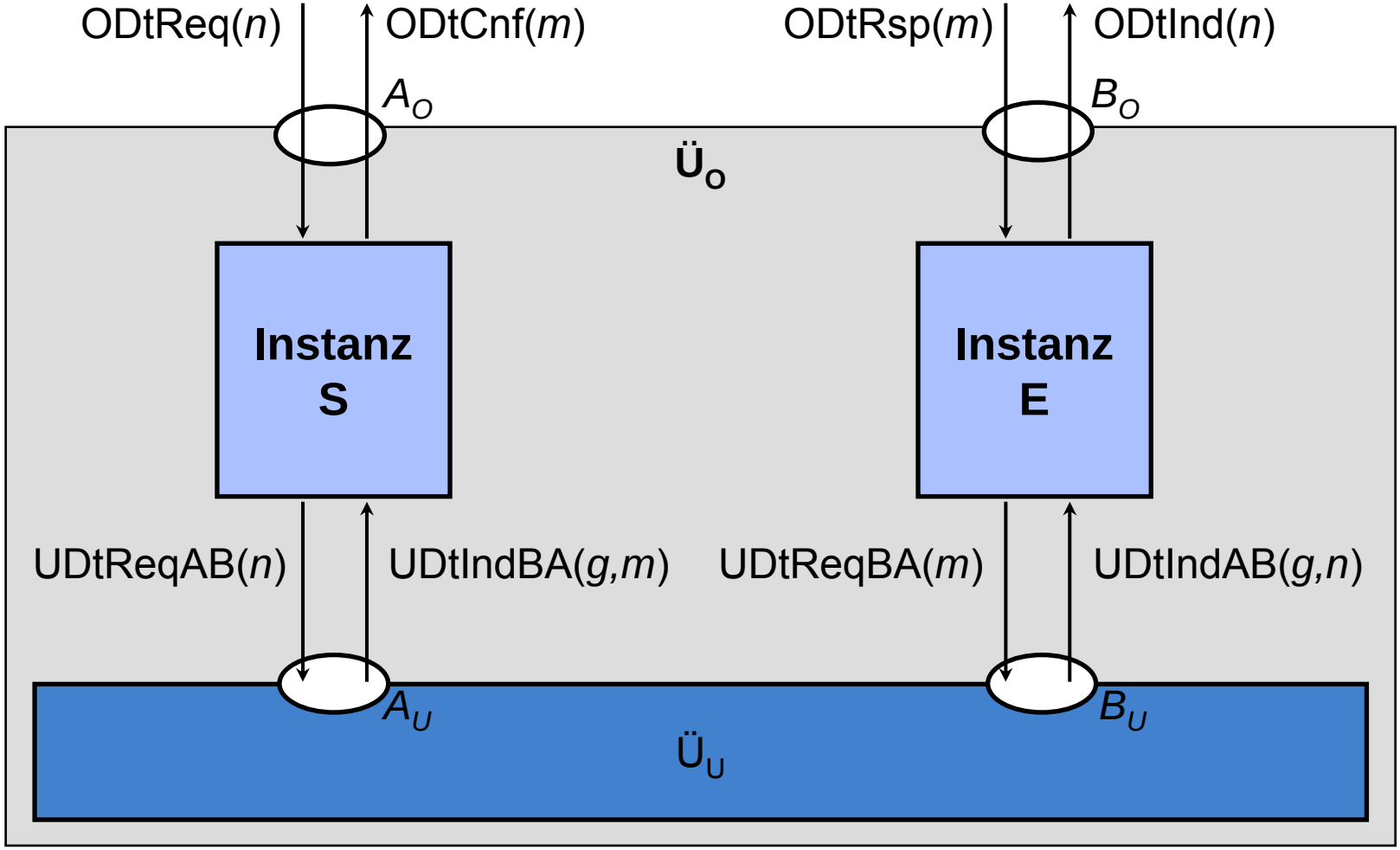
- Bereitstellung einer unbestätigten Datenübertragung
  - Parameter n, m: Nutzdaten; x, y: verfälschte Nutzdaten
  - Parameter g: „0“ – keine Verfälschung, „1“  Verfälschung erkannt
  - Start immer am Dienstzugangspunkt $A_U$
  - Verfälschungen möglich, Verdopplungen und Verlust nicht



# Dienstablauf in einem Weg-Zeit-Diagramm



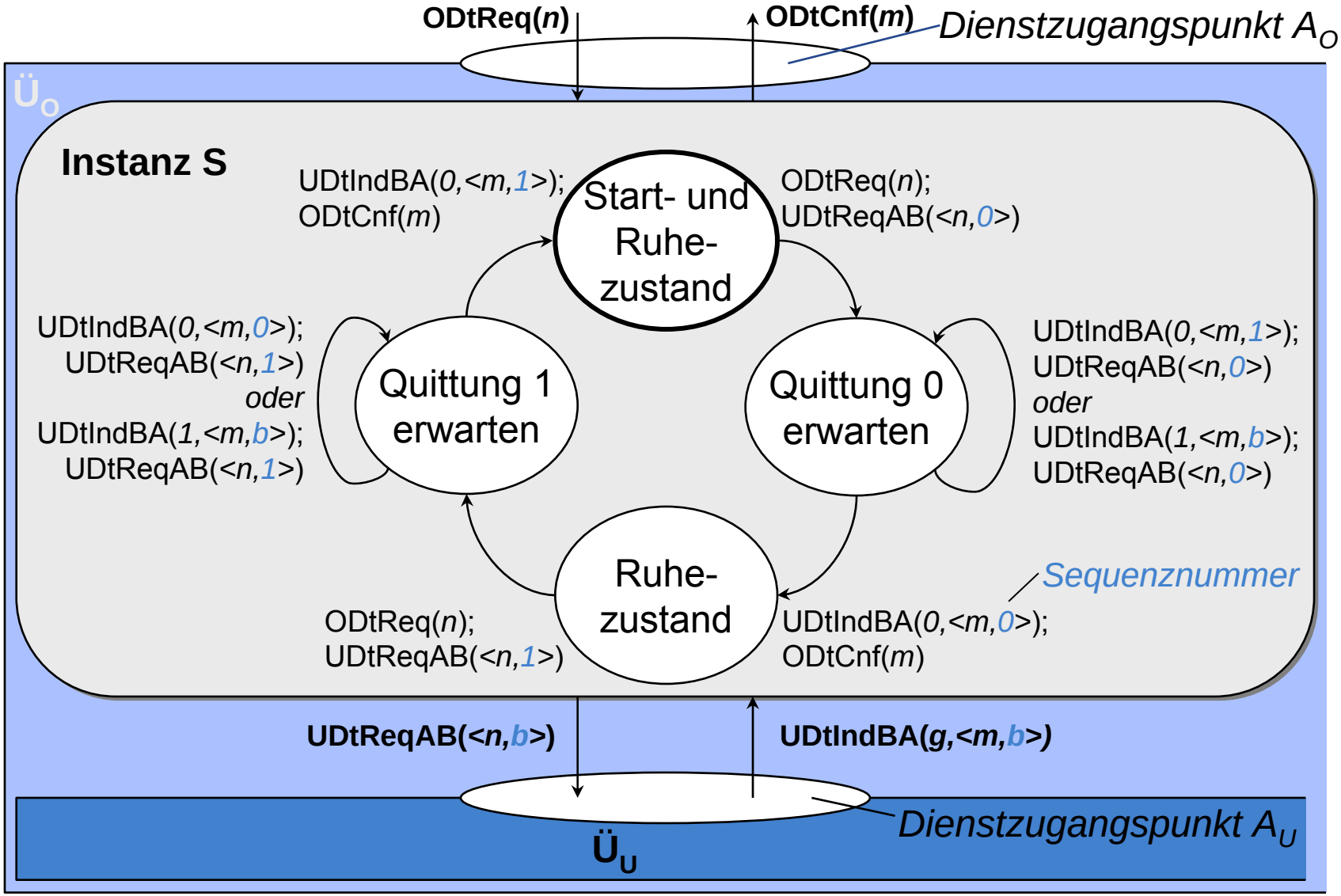
# Zusammenspiel der Schichten O und U



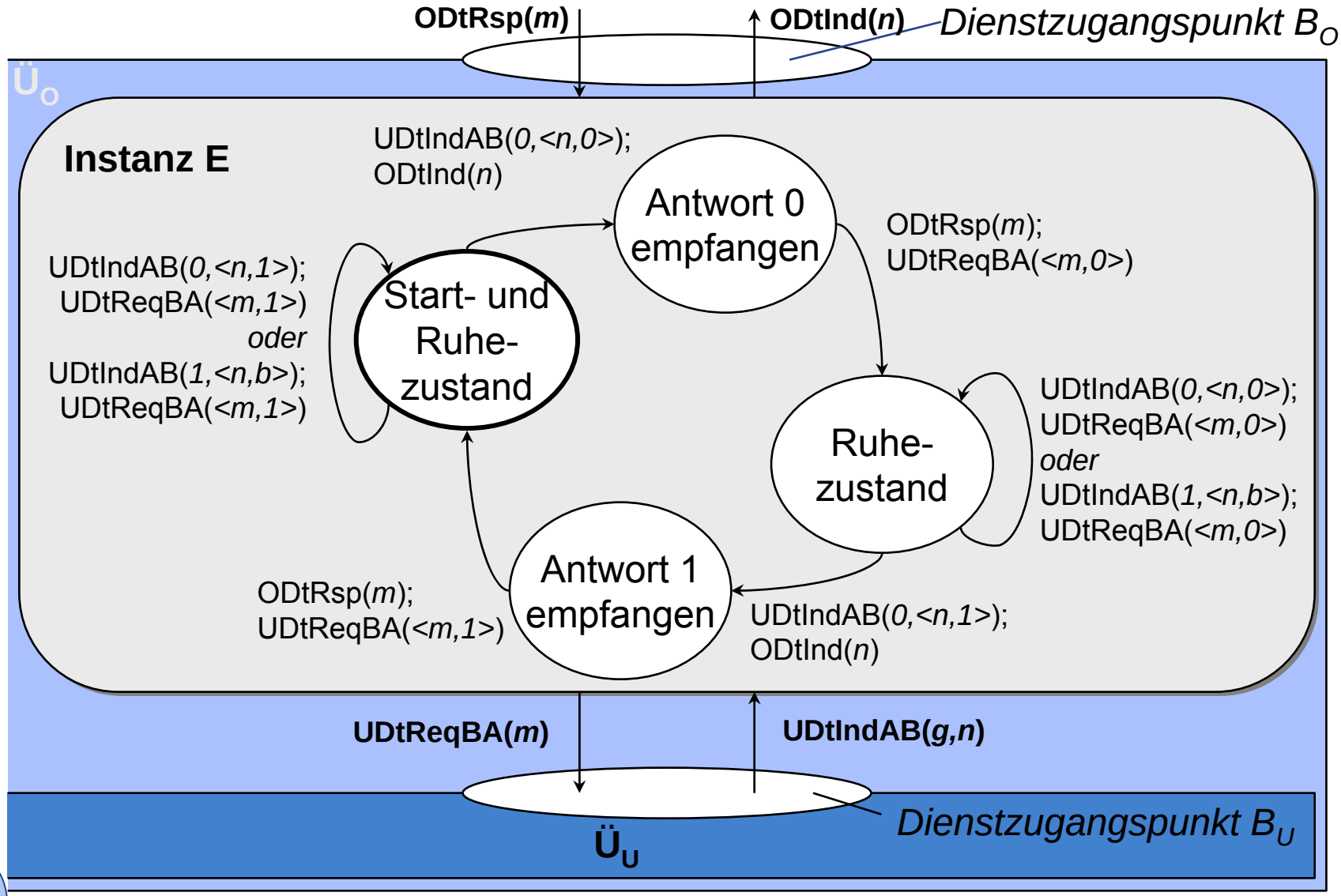
# Protokollinstanz S als endlicher Automat

- Endlicher Automat mit 4 Zuständen
  - Start- und Ruhezustand
    - Durch ODtReq aktiviert
    - Erzeugt UDtReqAB
    - Übergang in Folgezustand „Quittung 0 erwarten“
  - Quittung 0 erwarten
    - Beim Eintreffen von UDtReqBA verlassen
    - Rücksetzen, d.h. UDtIndAB wiederholen
      - Falls g=1
      - Falls falsche Sequenznummer (s=1)
    - Ansonsten: Übergang in Ruhezustand, Ausgabe ODtConf(m)
  - Ruhezustand
    - Mit Start- und Ruhezustand vergleichbar
  - Quittung 1 erwarten
    - Wie „Quittung 0 erwarten“, nur vertauschte Sequenznummern

# Protokollinstanz S als endlicher Automat



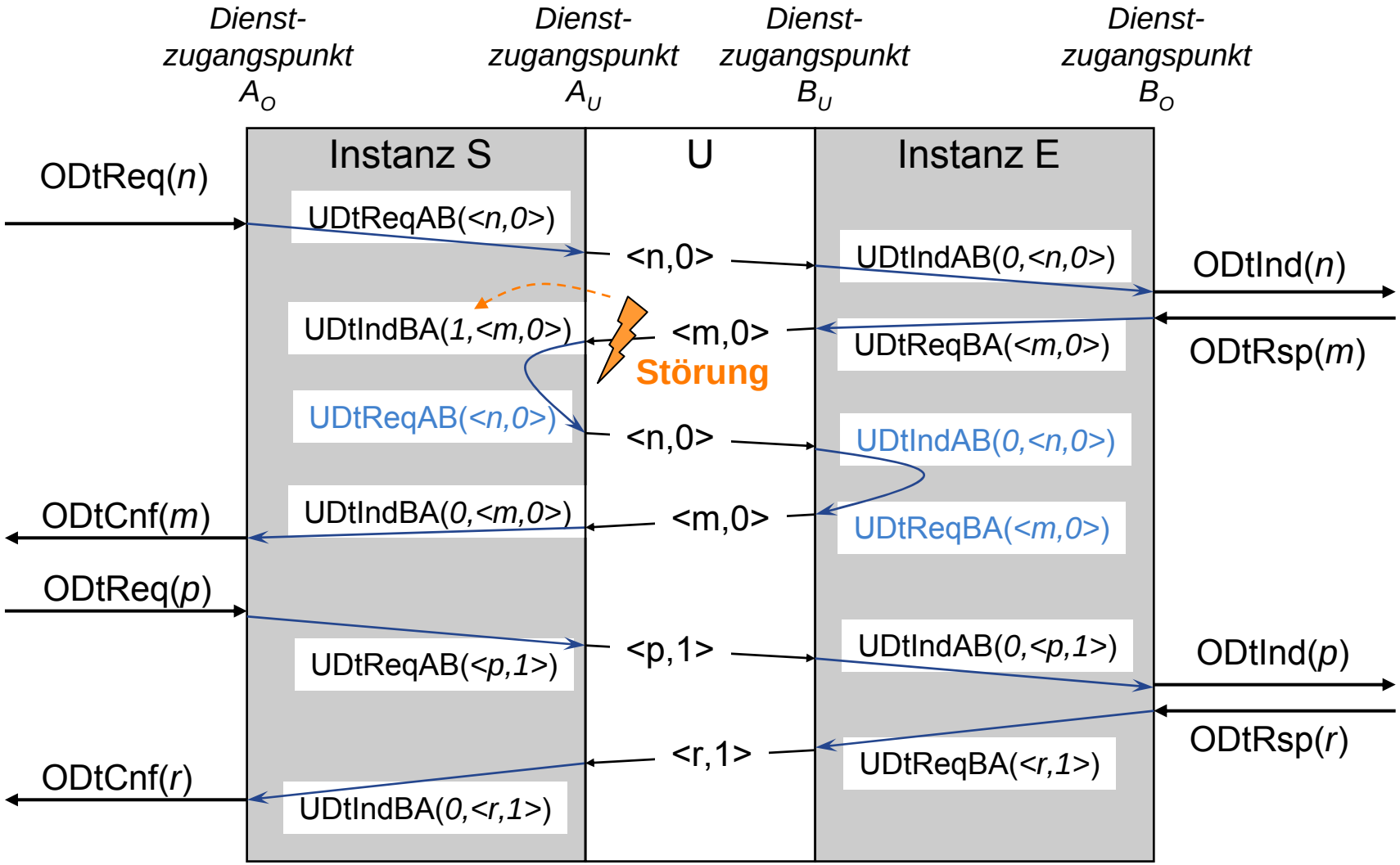
# Protokollinstanz E als endlicher Automat



# Doppelrolle der Sequenznummer

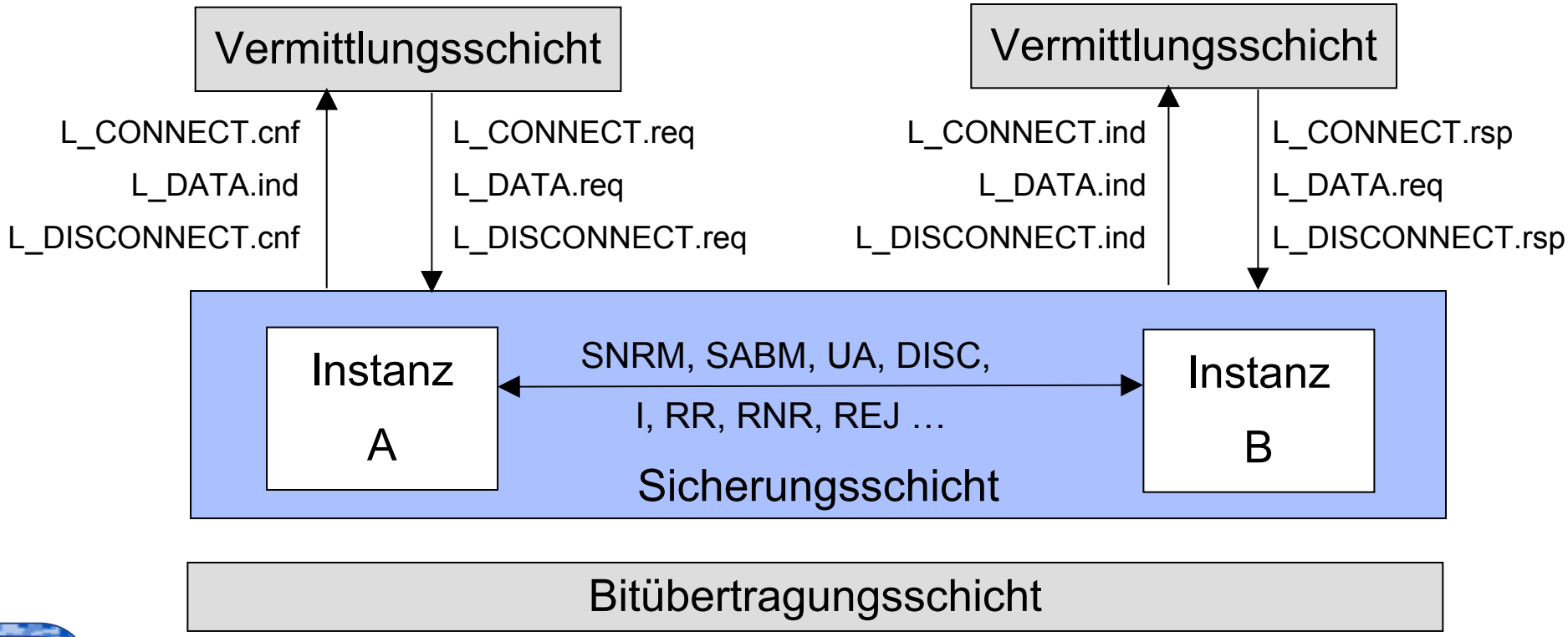
- Sequenznummer mit falschem Wert
  - Wirkt als negative Quittung
  - Aufforderung zur Sendewiederholung
- Sequenznummer mit richtigem Wert
  - Ermöglicht Erkennen von Duplikaten beim Empfänger
- Beispiel auf folgender Folie
  - Dateneinheit mit Sequenznummer 0 wird korrekt übertragen
  - Empfänger antwortet entsprechend
  - Antwort wird verfälscht
  - Sender wiederholt Dateneinheit mit Sequenznummer 0
  - Empfänger erkennt Duplikat
    - Ignoriert Dateneinheit
      - Liefert Daten nicht an den Dienstnehmer (höhere Schicht) ab
    - Wiederholt Antwort

# Vollständiger Beispielablauf

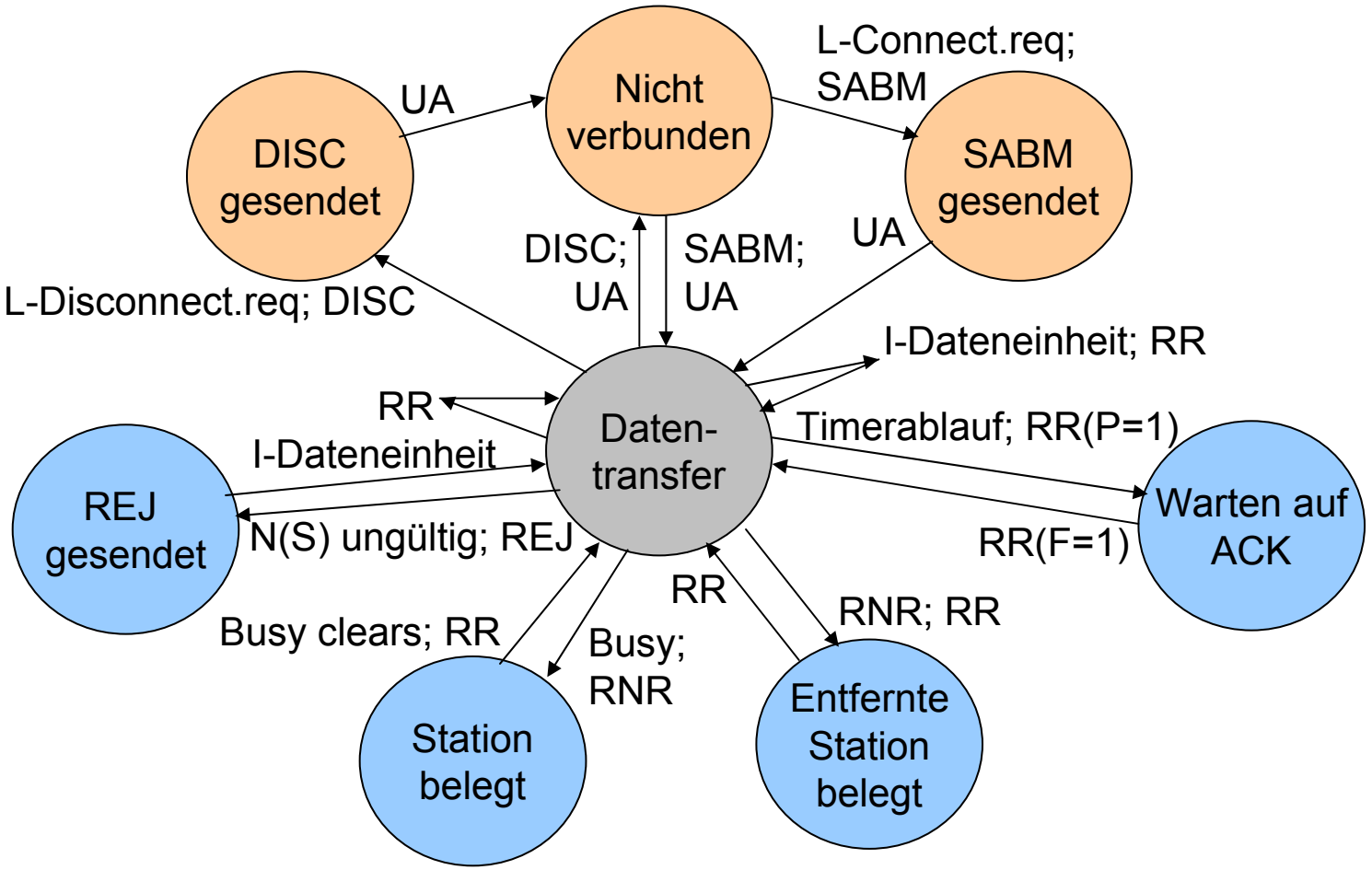


# 6.4.2 Beispiel: HDLC

- Vereinfachter Überblick
  - Dienstprimitive und Dateneinheiten



# Zustandsübergangsdiagramm (ABM)




- Anmerkung
  - Fehlerfälle nicht berücksichtigt

# Zusammenfassung

- Einfache Beschreibungsmethoden von Diensten und Protokollen
  - Weg-Zeit-Diagramme
  - Zustandsübergangsdiagramme auf der Basis endlicher Automaten
- Beispiele
  - Alternating-Bit-Protokoll
  - HDLC
- … so what?
  - Überprüfung der Korrektheit eines Protokolls
  - Basis für interoperable Implementierungen
    - Automatische Erzeugung
    - Händische Programmierung
  - Nicht formale Beschreibung von Protokollen lässt oftmals „Interpretationsspielraum“
    - Ergebnis: nicht interoperable Implementierungen

# Übungen

6.1 Was sind Dienstprimitive und wo treten sie in Erscheinung?

6.2 Wofür steht das Dienstprimitiv TDisConf? Was bedeutet NAboInd?

6.3 Geben Sie das Weg-Zeit-Diagramm eines bestätigten Verbindungsaufbaus auf der Transportschicht an.

6.4 Erläutern Sie den Zusammenhang von Dienstzugangspunkt und Verbindungsendpunkt.

6.5 Wieviele Dateneinheiten können gleichzeitig über das Alternating-Bit-Protokoll zwischen Sender und Empfänger unterwegs sein? Werden hierfür überhaupt Sequenznummern benötigt? Begründen Sie Ihre Antwort.

6.6 Welche Dienstprimitive treten im Alternating-Bit-Protokoll am Dienstzugangspunkt zum „tieferliegenden“ Basisdienst auf? Welche am „oberen“ Dienstzugangspunkt zum Alternating-Bit-Protokoll und wo dort genau?

6.7 Erläutern Sie den Unterschied zwischen dem Zustandsübergangsdiagramm eines Übertragungsabschnittes und dem Automat einer Protokollinstanz.

# Literaturhinweise

[AbLo03] Abeck, Lockemann, Schiller, Seitz; Verteilte Informationssysteme, dpunkt-Verlag, 2003

- Kapitel 5

[Hals05] F. Halsall; Computer Networking and the Internet; Addison-Wesley, 2005

- Kapitel 1.4.7 – Protocol Specification

[LoKr93] Lockemann, Krüger, Krumm; Telekommunikation und Datenhaltung; Hanser-Verlag, 1993

- Kapitel 6

# Kommunikation und Datenhaltung

## 7. Sicherungsschicht: Lokale Netze




**Prof. Dr. Martina Zitterbart**
Dipl.-Inform. Martin Röhricht
[zit | roehricht]@tm.uka.de



# Kapitelübersicht

1. Einführung
2. Physikalische Grundlagen
3. Protokollmechanismen
4. Geschichtete Architekturen
5. Sicherungsschicht: HDLC
6. Beschreibungsmethoden
7. Sicherungsschicht: Lokale Netze
8. Netzkopplung und Vermittlung
9. Die Transportschicht
10. Anwendungssysteme
11. Middleware

7.1 Medienzugriff
- 7.1.1 Zufallsstrategien
- 7.1.2 CSMA/CD
- 7.1.3 Zirkulierendes Senderecht

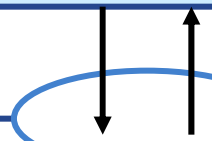
7.2 Strukturierung der Datenübertragung

# Aufgaben der Sicherungsschicht

**2 Sicherungsschicht**

- Verfälschungs- und Verlustsicherung
- Strukturierung des Datenstroms
- Medienzugangskontrolle bei geteilten Medien

⇨ Sicherungsschicht bietet abstraktes Medium „gesicherten Kanal“



**1 Einige Probleme der physikalischen Schicht**

- unerwünscht hohe Störungen der Übertragung
- kaum beeinflussbare Übertragungsqualitäten
- keine Pufferung möglich
- oft keine globale (nur kanalbeschränkte) Adressierung

# Typische Aufgabenteilung bei lokalen Netzen

3 IP und andere

**2b Logical Link Control (LLC)**
(einheitlich für alle Medien)

- Verfälschungssicherung
- Verlustsicherung
- Reihenfolgesicherung
- Flusskontrolle
- Strukturierung der Übertragung

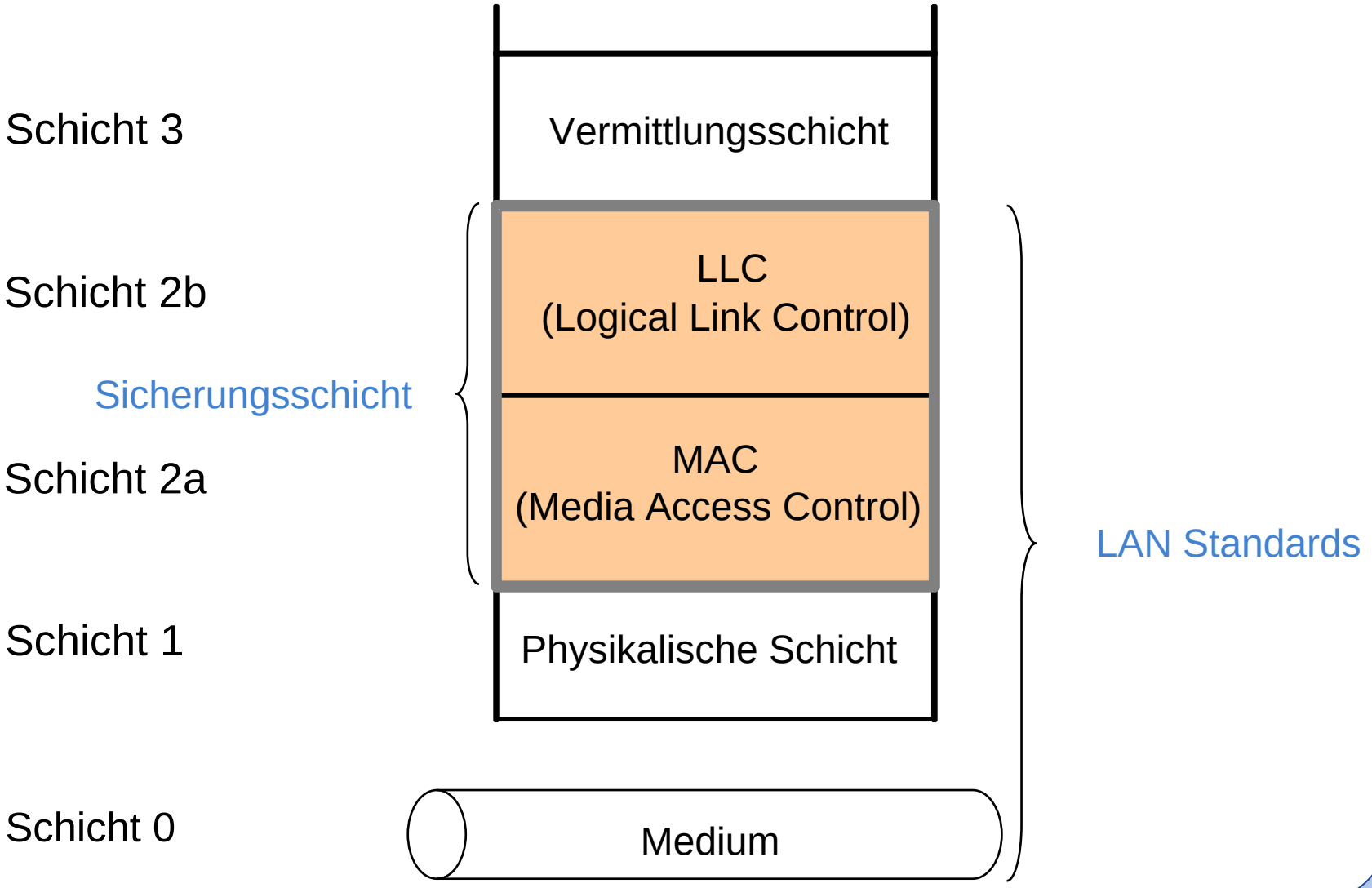
**2a Medium Access Control (MAC)**
(Zugangskontrolle für geteiltes Medium)

- konkurrierender Zugriff
- kontrollierter Zugriff

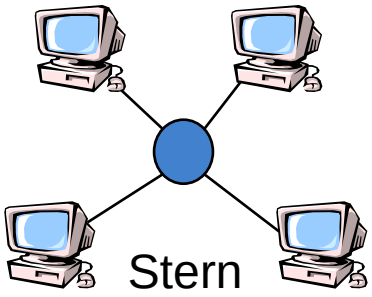
1 Bitübertragungsschicht

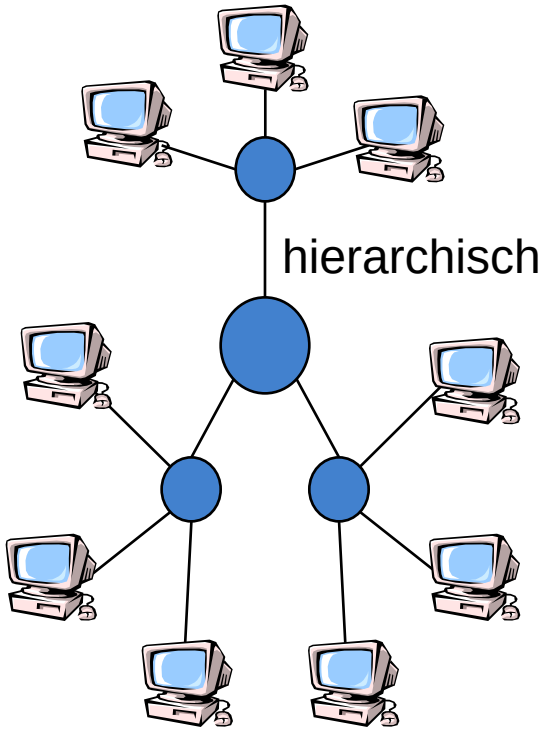
- Medien
  - Verdrillte Kupferadern, Koaxialkabel, Glasfasern, …
- Codierung (Leitungscode)

# Einordnung lokaler Netze

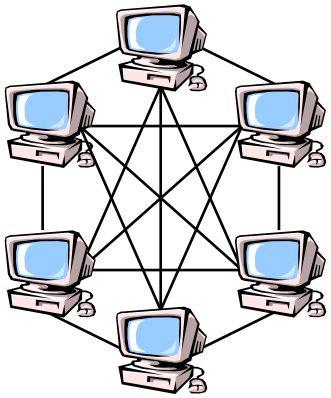


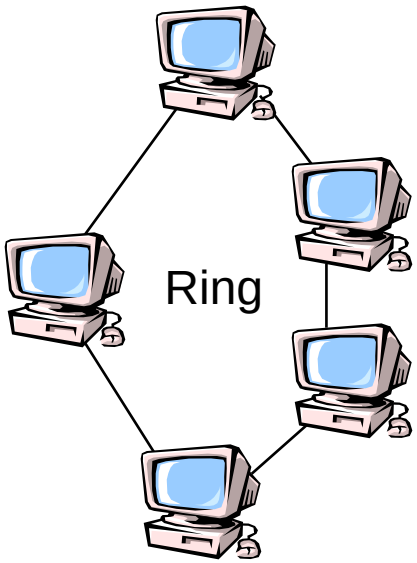
# Topologien



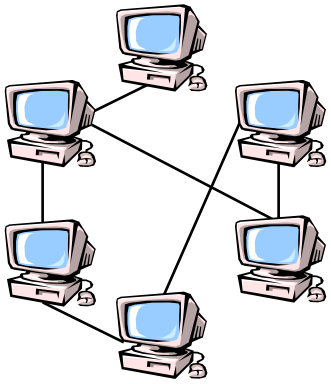



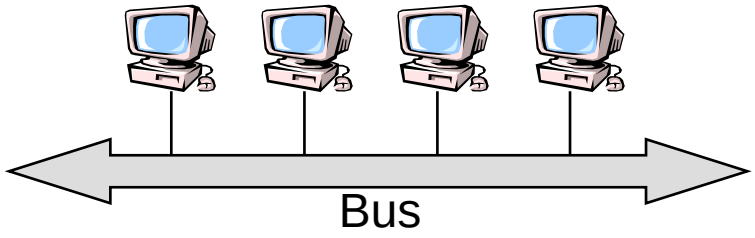

vollvermascht





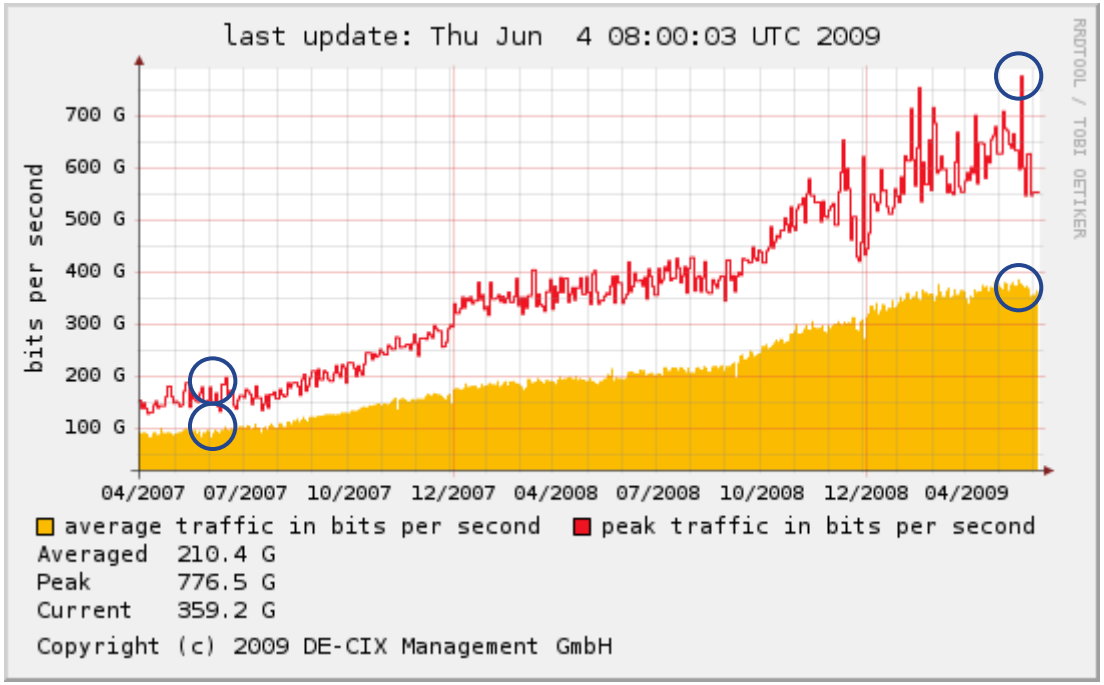

teilvermascht





# Beispiel: Frankfurter DE-CIX-Knoten

- Beispiel für Umstieg auf Stern-Topologie
  - ( … allerdings nicht im lokalen Netz)
  - Ziel: höhere Flexibilität und Ausfallsicherheit
- Ausbau des Frankfurter Internet-Austauschknotens DE-CIX, April 2008
  - Gestiegene Bedeutung für osteuropäischen Breitbandmarkt
  - Fast 4-faches Verkehrsaufkommen 06/2007 – 06/2009
  - Bis zu 776 Gbit/s, 285 angeschlossene autonome Anbieter



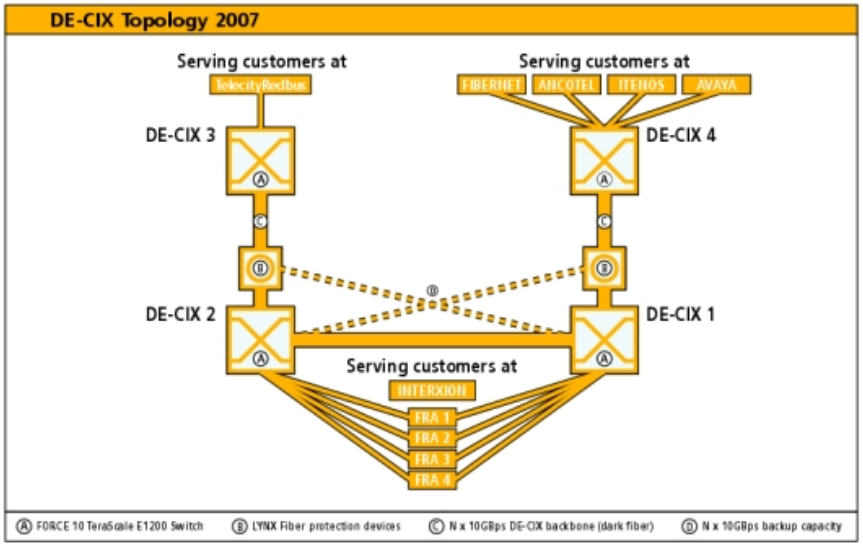

[DECI09]

# Beispiel: Frankfurter DE-CIX-Knoten (2)

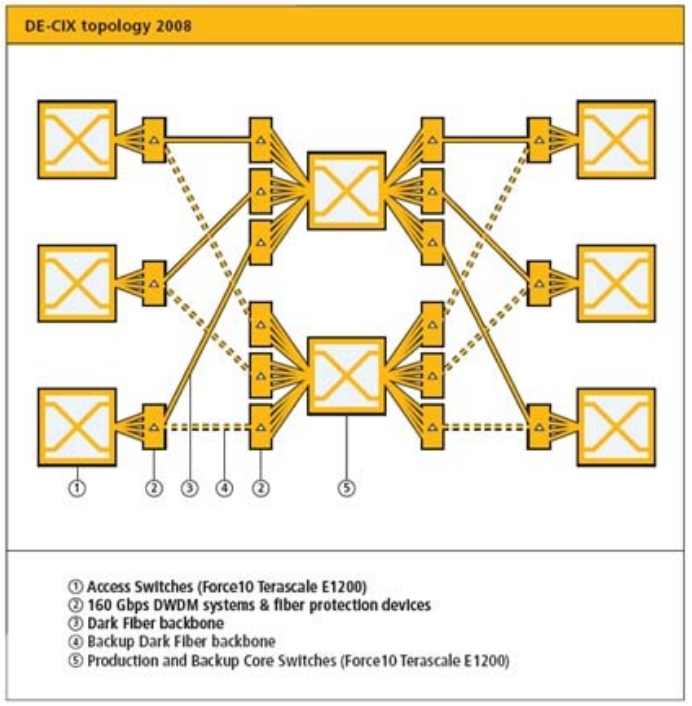
- Umstieg auf Sterntopologie
  - Verbindung mit Kunden-Netzen über 6 Access-Switches
    - Jeweils zwei 160 Gbit/s Glasfasern zur Mitte des Sterns (Core-Switch)
  - Core-Switch: Redundanz durch identisches Gerät im „Hot-Standby“
  - Soll maximalen Datendurchsatz von 1,4 Tbit/s ermöglichen
- Wechsel vom Kunden unbemerkt
  - *"Für uns ist das, als ob man bei einem Auto, das 200 km/h fährt, die Reifen wechselt – und keiner merkt es."*

[Heis08, DECI09]

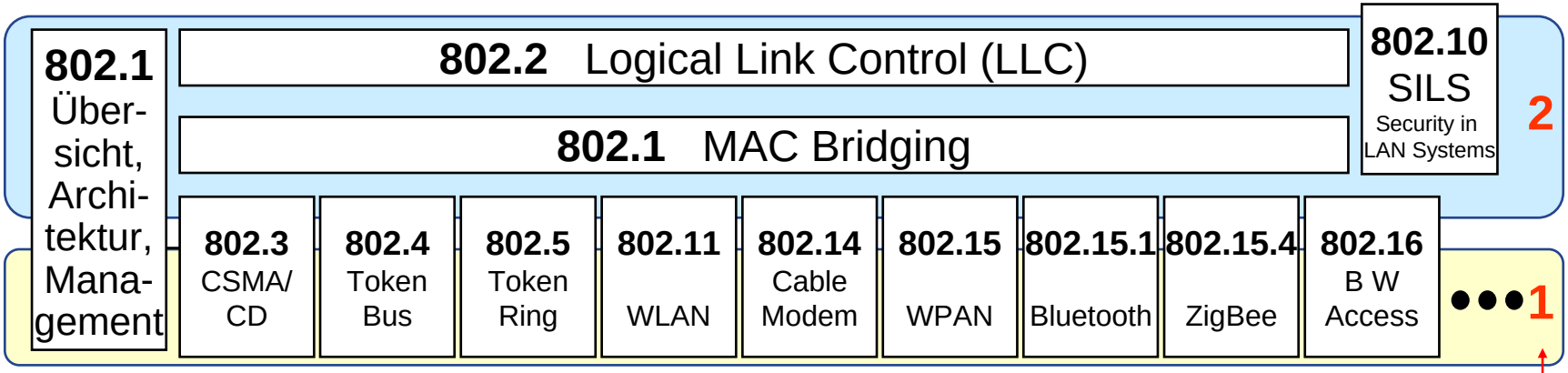
**2007**




**2008**



# LAN-Standardisierung nach IEEE 802

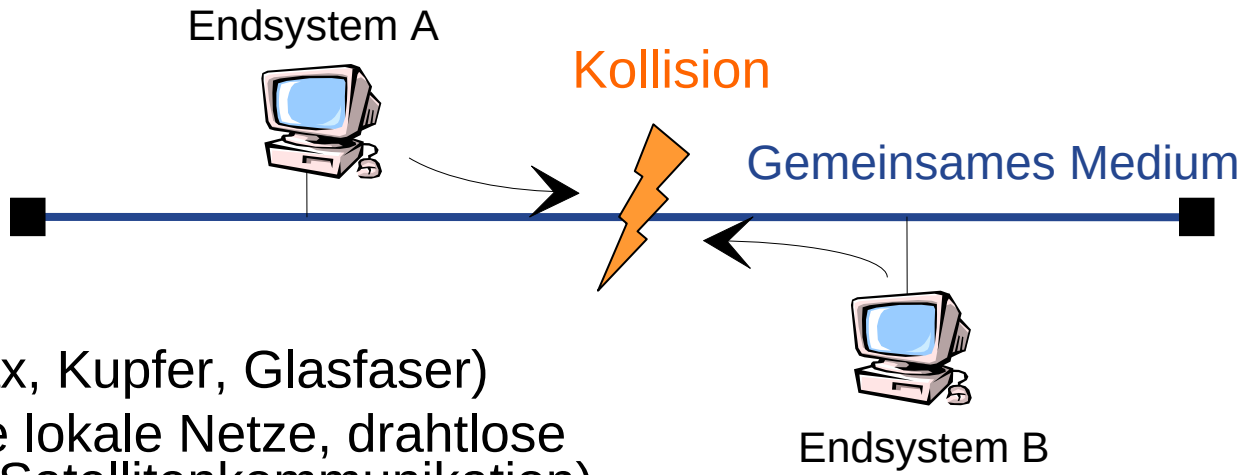



*Themen:*
802.1: Zusammenhang der Standards und MAC Bridging
802.2: Logical-Link-Control-Dienste/Protokolle (LLC)
802.3: CSMA/CD-Protokoll auf Bustopologie
802.4: Token-Bus-Protokoll auf Bustopologie
802.5: Token-Ring-Protokoll auf Ringtopologie
802.10: Sicherheitsstruktur für 802-Protokolle
802.11**:** Wireless LANs
802.14: Cable Modem: Datenübertragung über TV-Kabelmodems
802.15**:** Wireless Personal Area Networks: Kurzdistanz-Netze, Bluetooth
802.16**:** Broadband Wireless Access: Drahtloser Breitbandzugriff (WiMAX)

Informationen zu weiteren Standards der 802-Serie:
http://grouper.ieee.org/groups/802/

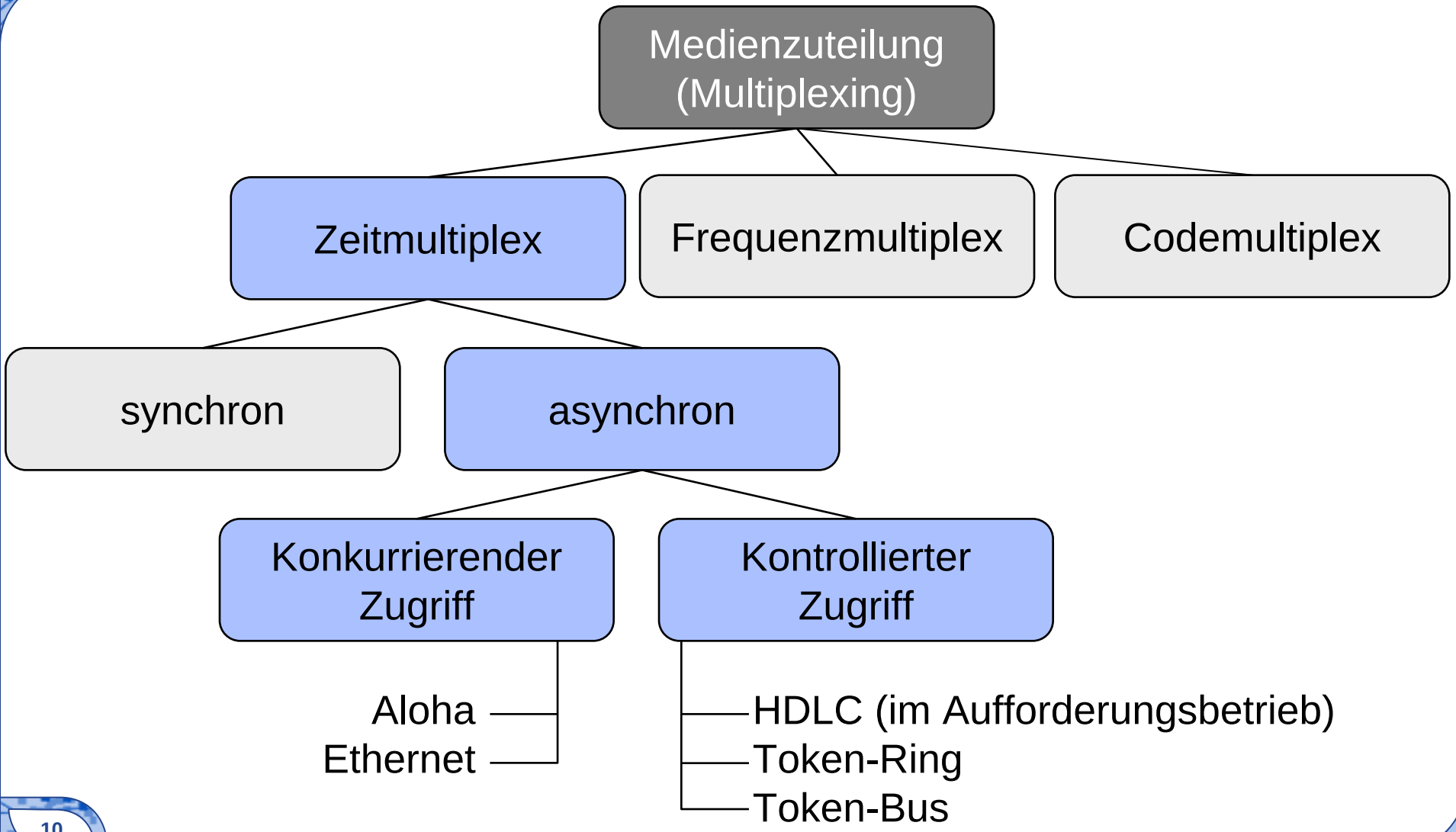
# 7.1 Problemstellung Medienzugriff

- Problem
  - Mehrere Geräte benutzen ein gemeinsames Medium
    - Wie wird der Zugriff auf dieses Medium kontrolliert?
  - Situation



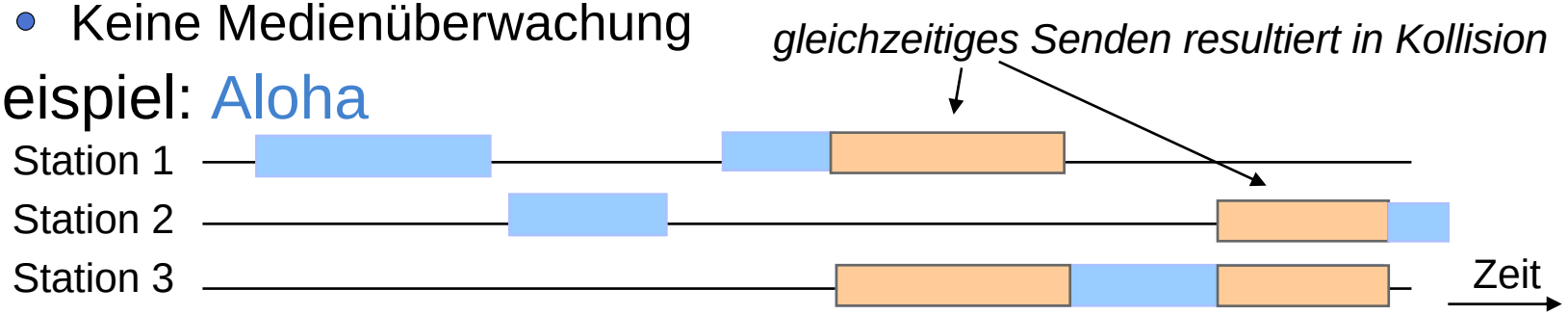

- Medium
  - Kabel (z.B. Koax, Kupfer, Glasfaser)
  - Äther (drahtlose lokale Netze, drahtlose Zugangsnetze, Satellitenkommunikation)
- Wichtige Aspekte
  - Durchsatz
  - Verzögerung
  - Stabilität
  - Fairness

# Medienzuteilungsverfahren (Übersicht)

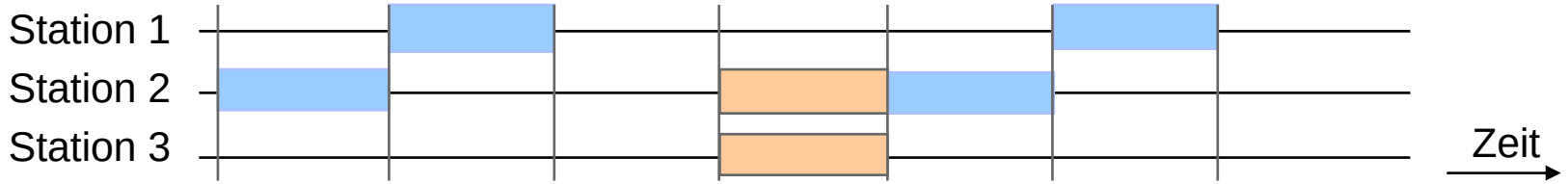


# 7.1.1 Konkurrierender Zugriff: Zufallsstrategien

- Zufälliger Zugriff auf das Kommunikationsmedium
  - Zunächst zur Zugriffskontrolle auf gemeinsamen Satellitenkanal verwendet
  - An der Uni Hawaii eingesetzt, um Rechner auf verschiedenen Inseln zu verbinden
  - Keine Medienüberwachung
- Beispiel: Aloha




  - Funktioniert nur, wenn Sendewünsche zufällig und unabhängig voneinander verteilt sind und die Wahrscheinlichkeit überlappender Sendewünsche gering ist
- Verbesserte Variante: Slotted-Aloha



# 7.1.2 Konkurrierender Zugriff: CSMA/CD

- Carrier Sense Multiple Access with Collision Detection (CSMA/CD)
- Ablauf
  - Listen before Talk*:* Endsystem prüft, ob das Medium frei ist
  - Falls Medium frei ist, kann prinzipiell gesendet werden
  - Beginnen mehrere Endsysteme praktisch gleichzeitig zu senden, so können Kollisionen auftreten:
    - Erkennung von Kollisionen durch Abhören während der Sendung (Listen while Talk)
    - Im Kollisionsfall:
      - Sendeunterbrechung, sowie Senden eines Jamming-Signals
      - Sendung später erneut versuchen
  - Das wiederholte Senden regelt ein Backoff-Algorithmus

# Voraussetzungen für CSMA/CD

- Voraussetzungen für Kollisionserkennung
  - Mindestlänge der 802.3-MAC-Dateneinheiten (siehe Beispielablauf nächste Folie)
    - Das Senden der Dateneinheit darf nach Signallaufzeit durch Medium und zurück noch nicht beendet sein
    - Mindestlänge der Dateneinheiten erforderlich
      - Abhängig von maximaler Ausdehnung des Netzes und Ausbreitungsgeschwindigkeit
        - Bestimmt die Zeit, die ein Bit für die Übertragung vom Sender zum Empfänger benötigt. Wird als Ausbreitungsverzögerung bzw. Signallaufzeit bezeichnet.
      - Wird durch ein so genanntes Stopffeld *PAD(ding)* sichergestellt

# Beispielablauf

$t_L$ Signallaufzeit von A nach B (Propagation Delay)
$2\ t_L$ Signallaufzeit von A nach B und zurück (Round Trip Delay, Round Trip Time)
$e$ Zeitabschnitt zwischen Übertragungsbeginn von B und Kollisionserkennung durch B

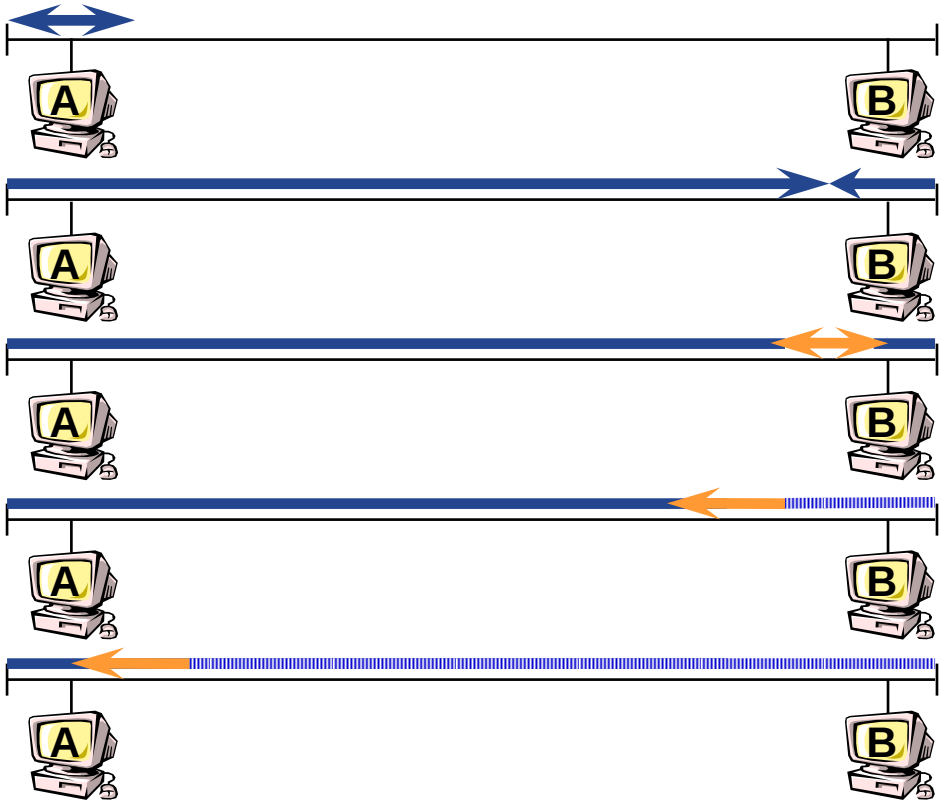
A beginnt Übertragung ($t_0$)

B beginnt Übertragung ($t_0 + t_L - e$)
*(bevor Signal von A eintrifft)*

B entdeckt Kollision, stoppt eigene Übertragung ($t_0 + t_L$)

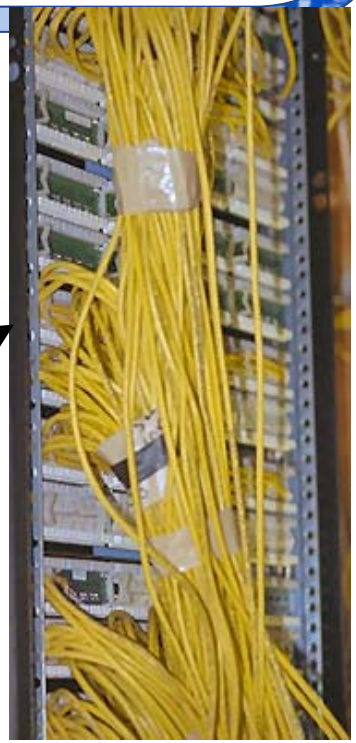
B schickt Jam-Signal, Kollision läuft weiter; B führt Backoff-Algorithmus aus

A entdeckt Kollision ($t_0 + 2t_L - e$)
*(muss dafür noch das Medium abhören,*
*⇨ Grund für Mindestlänge von Dateneinheit)*
A führt Backoff-Algorithmus aus



# Weiterentwicklung: 100Base – Fast Ethernet

- Charakteristika
  - Datenrate 10 bzw. 100 Mbit/s
  - Medienzugriffsverfahren und Format der Dateneinheiten wird von CSMA/CD übernommen
    - „Backward Compatibility“
  - Flexibles Verkablungskonzept (Hierarchie von Hubs)
  - Multiport-Bridges („Switches“) ermöglichen Vollduplexbetrieb
    - kein CSMA/CD am Zugangslink mehr erforderlich
- Varianten



*„patch-panels“ für die Sternverkabelung*



| Bezeichnung | Kabel | Segmentlänge | Vorteile |
|---|---|---|---|
| 100Base-T4 | Vier verdrillte Adernpaare | Max. 100m | Gängige Kabel |
| 100Base-TX | Zwei verdrillte Adernpaare | Max. 100m | Vollduplex bei 100 Mbit/s |
| 100Base-FX | Glasfaser | Max. 2.000m | Vollduplex, lange Strecken |

# Weiterentwicklung: Gigabit Ethernet

- Grundgedanke
  - logischer Schritt von 10 Mbit/s Ethernet über Fast-Ethernet (100 Mbit/s) zu Gigabit-Ethernet (1.000 Mbit/s)
  - Ziel: Beibehaltung des CSMA/CD-Verfahrens
    - „Backward Compatibility“
  - Ideale Ergänzung zu (Fast-) Ethernet zur Fortführung im Backbone-Bereich
  - Einsatz auf Glasfaser und Kupferadern
- Probleme
  - Minimales Länge der Dateneinheiten von (Fast-) Ethernet zu klein, um Kollisionen zu erkennen
    - Kompatibilität zu (Fast-) Ethernet möglich?
  - Strenge Normen zur elektromagnetischen Verträglichkeit
  - Elektro-physikalische Eigenschaften der Kupferadern
    - Übersprechen, Dämpfung, ...

# Gigabit-Ethernet im Detail

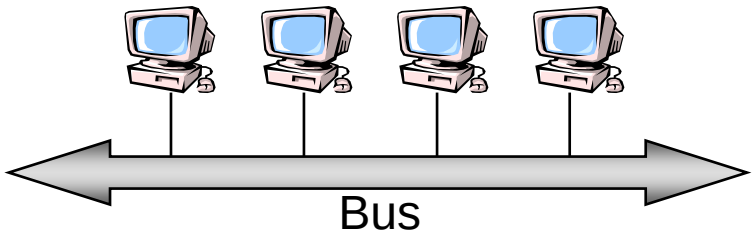
- Duplex-Betrieb und Halbduplex-Betrieb möglich
  - Praktisch alle neueren Komponenten sind vollduplexfähig
- Problem: Beim Halbduplex-Betrieb Kollisionserkennung notwendig
  ⇨ minimale Länge der Dateneinheiten von (Fast-) Ethernet zu klein!
- Lösung
  - Bei Gigabit-Ethernet musste minimale Länge der Dateneinheiten von 64 Byte auf 512 Byte erhöht werden
  - Auffüllen kleinerer Dateneinheiten mit Füllzeichen (Extension Symbols), die an die FCS angehängt werden, jedoch keine Bedeutung haben
    - Weiterhin minimale Länge der „ursprünglichen“ Dateneinheit von 64 Byte
    - Problem: „hoher Verschnitt“
  - Packet Bursting
    - Zusammenfassen mehrerer kleiner Dateneinheiten eines Senders in einer MAC-Dateneinheit
    - Maximale Sendedauer pro einzelnem Sendevorgang

# Gigabit-Ethernet im Detail (2)

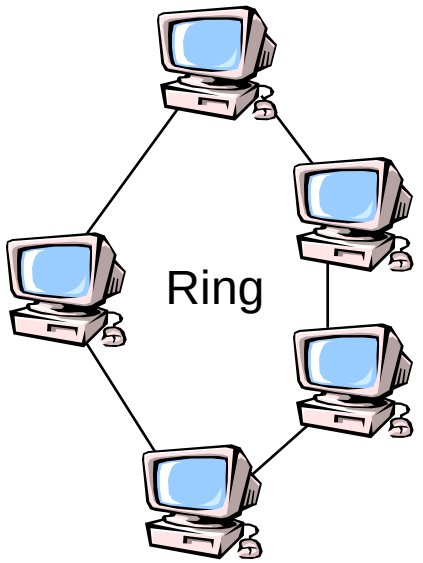
- Burst Limit
  - Beschränkung der max. Sendezeit auf 65.536 bit
- Format der Dateneinheiten entspricht demjenigen von (Fast-) Ethernet auf MAC-Ebene
  - Auf der physikalischen Ebene jedoch leichte Unterschiede, die aber für MAC transparent sind
- Umschalten zwischen 10, 100, 1.000 Mbit/s mittels Autonegotiation möglich

# 7.1.3 Kontrollierter Zugriff: zirkulierendes Senderecht

- Beispiel: Token-Ring (IEEE 802.5)
- Ausgangspunkt bei Ethernet:




- Ausgangspunkt bei Token-Ring:



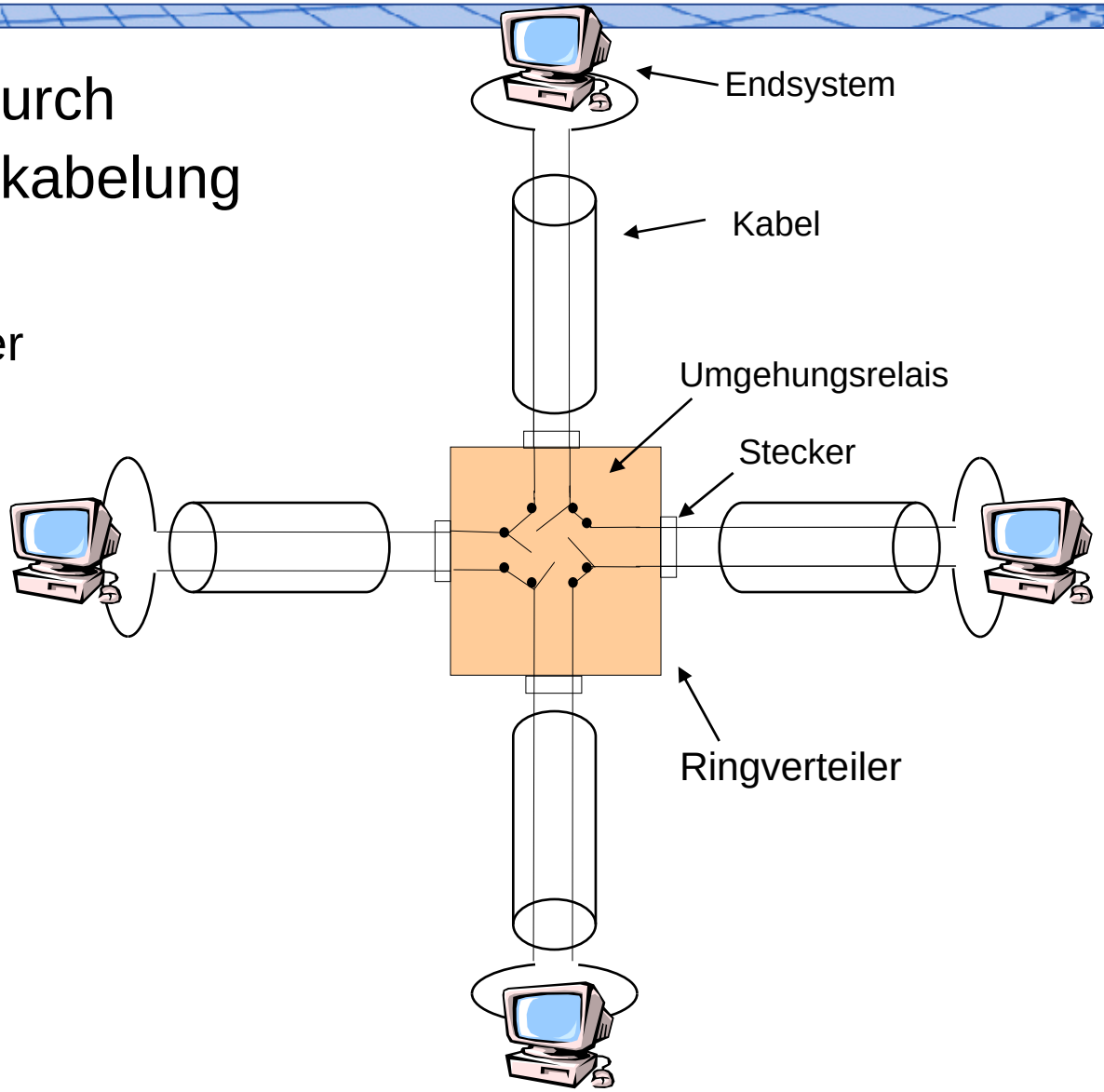
# Kontrollierter Zugriff: zirkulierendes Senderecht

- Beispiel: Token-Ring (IEEE 802.5)
  - Endsysteme sind jeweils Punkt-zu-Punkt zu einem Ring verbunden
  - Jedes Endsystem hat somit Vorgänger und Nachfolger
  - Unidirektionaler Betrieb von Endsystem zu Endsystem
  - Endsysteme sind aktiv an das Medium gekoppelt, d.h. sie können die einkommenden Daten regenerieren und evtl. modifizieren
  - Zuteilung des Senderechts erfolgt durch zirkulierende spezielle Steuerdateneinheit, das so genannte Token
    - Ein Endsystem, das das Token empfängt, hat das Recht, Daten zu verschicken
    - Verfahren wird auch Token-Passing genannt
  - Die verschickten Daten kommen aufgrund der Ringstruktur wieder bei dem sendenden Endsystem an, das diese wieder vom Ring nimmt
  - Danach gibt es das Senderecht, d.h. das Token, an das nachfolgende Endsystem weiter

  ⇨ Es wird ein umfangreiches Token-Management benötigt

# Token-Ring: Schema eines Ringverteilers

- Token-Ring oft durch sternförmige Verkabelung realisiert
  - Mit Ringverteiler



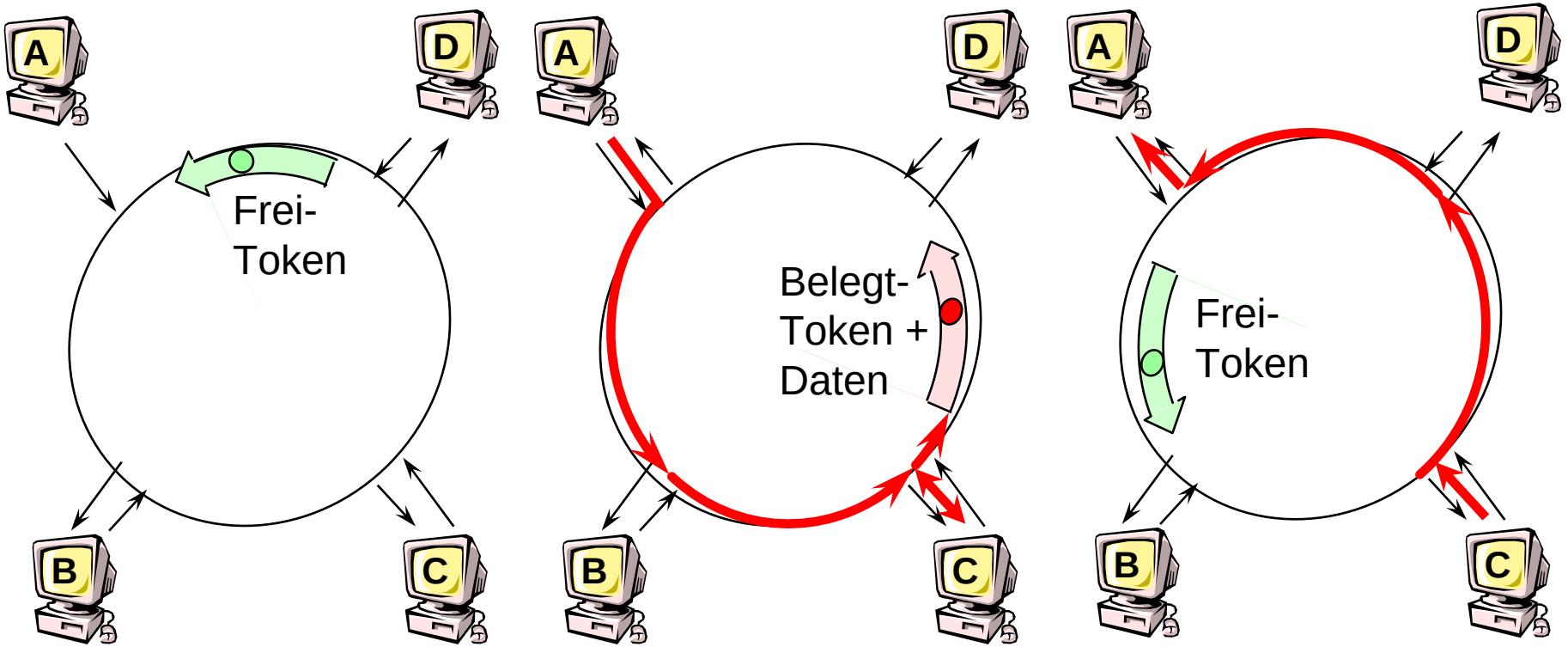
# Allgemeines zum Ablauf

- Protokollablauf
  - Senden ist nur erlaubt, wenn Endsystem ein freies Token „besitzt“
  - Maximale Sendezeit ist durch die Token Holding Time (THT) begrenzt
    - Es können in dieser Zeit mehrere Dateneinheiten gesendet werden
  - Kontrollinformation kann per piggybacking zum Sender gesendet werden
    - Z.B. Quittungen
  - Prioritätsmechanismus steht zur Verfügung
    - 8 Prioritäten

# Monitor

- Monitor
  - „Zentrales“ Endsystem zur Überwachung der Funktionsfähigkeit des Rings
    - Z. B. Token-Verlust, Token-Verdopplung, Ringbruch
    - Jedes Endsystem kann diese Aufgabe übernehmen
  - Einfügen einer minimalen Verzögerung, falls der Ring zu klein ist
  - Generell: Fehlermanagement recht komplex

# Medienzugriff




- Frei-Token kreist
- A hat Sende-wunsch

- A hat Token belegt
- A sendet an C
- C kopiert
- C leitet weiter

- A nimmt Daten vom Ring
  - Weshalb?
- C kopiert und setzt Quittungsbits
- Token wird von A auf frei gesetzt

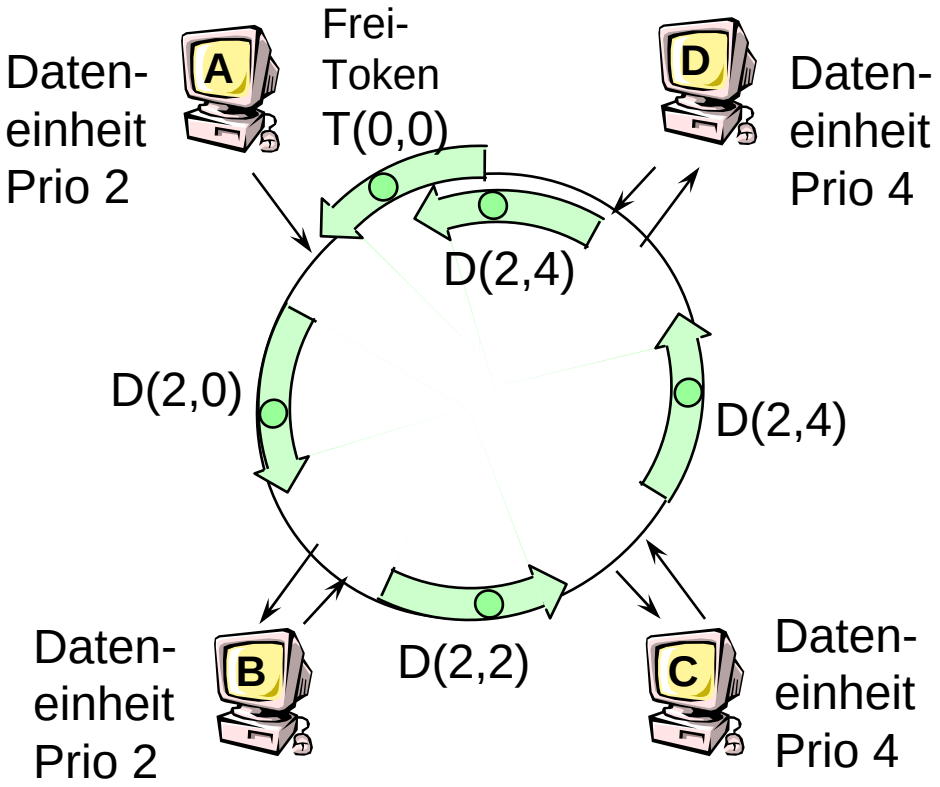
# Prioritäten

- Daten mit hoher Priorität werden zuerst gesendet
  - Gleichberechtigter Zugriff innerhalb einer Prioritätsebene
- Erhöht ein Endsystem die Priorität am Ring, so muss dieses Endsystem die Priorität auch wieder zurücksetzen
- Jedes Endsystem verwaltet hierzu die folgenden Variablen
  - Prioritätsregister
    - Für die höchste Priorität der wartenden Dateneinheit
    - Für die Priorität im Zugriffskontrollfeld
    - Für die Reservierung im Zugriffskontrollfeld
  - Keller
    - Für die Speicherung der alten Priorität
    - Für die Speicherung der neuen Priorität
    - Erlaubt mehrfaches Erhöhen der Priorität

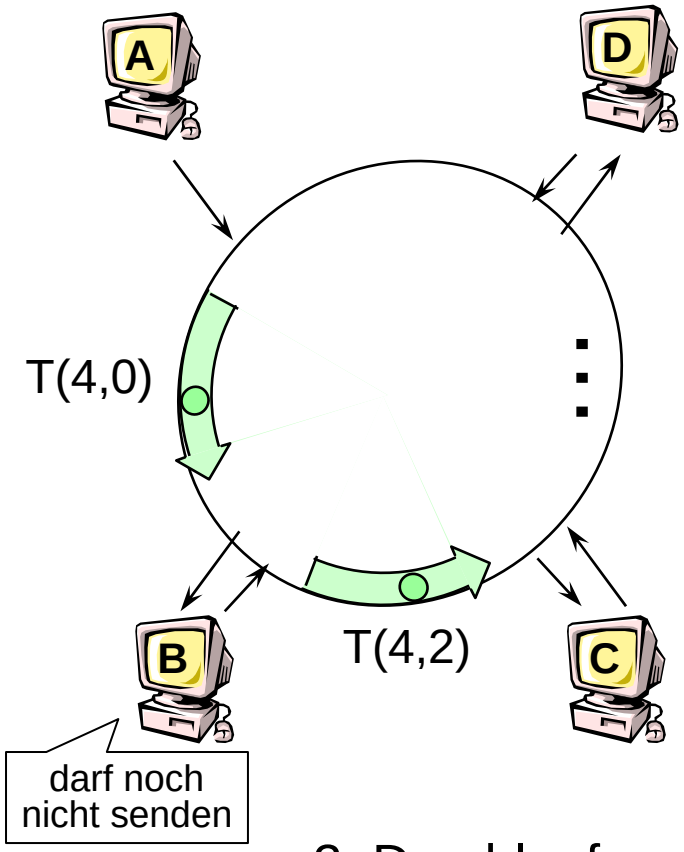
# Beispiel zu Prioritäten

Schreibweise: T(P,R) bzw. D(P,R); P=Priorität, R=Reservierung

Token Daten

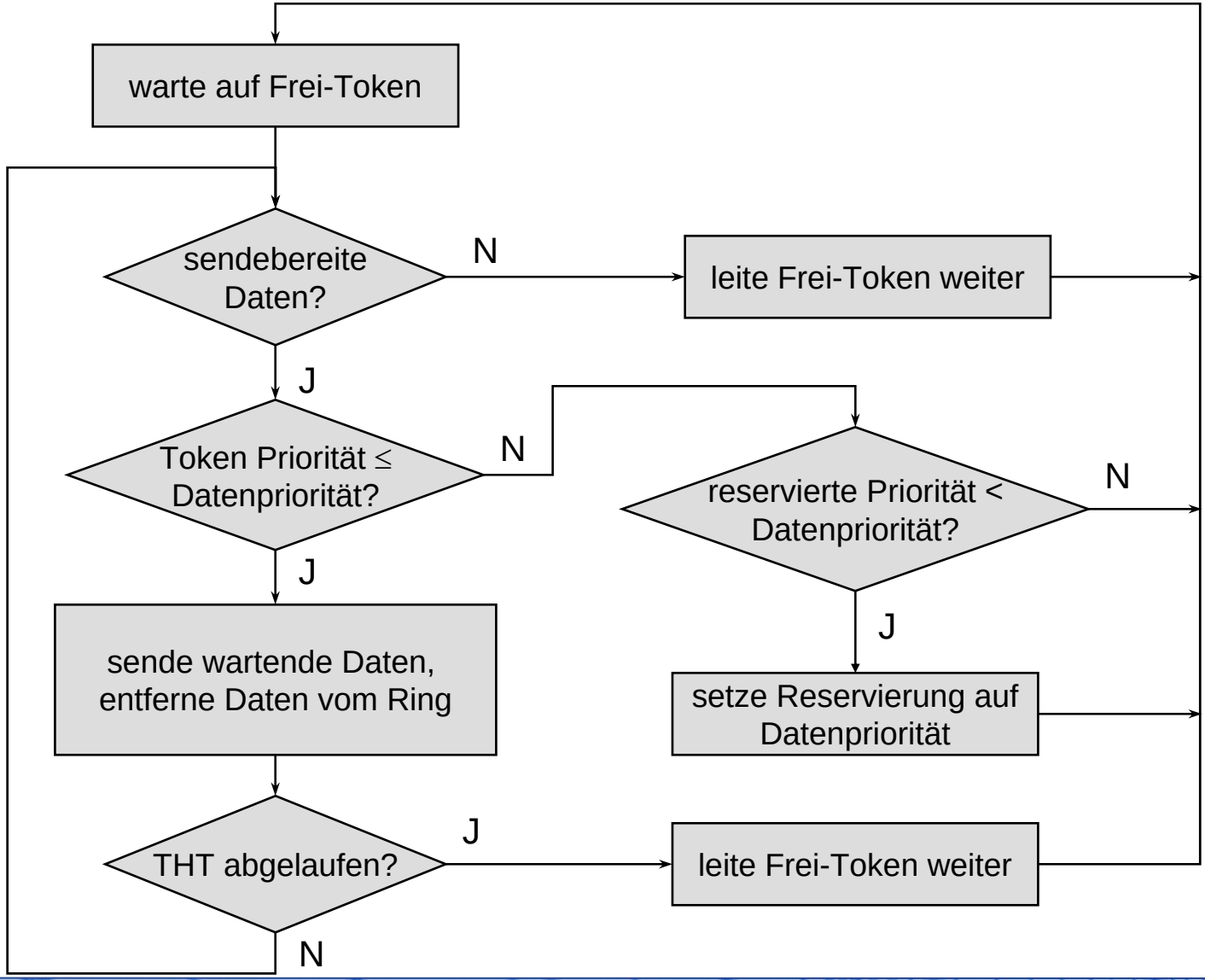



1. Durchlauf




2. Durchlauf

# Sendealgorithmus



# Funktionen zur Fehlerkontrolle und -korrektur

- Genau ein aktiver Monitor pro Ring zur Fehlerkorrektur bei
  - Zirkulierenden Daten
  - Zirkulierendem Token mit hoher Priorität
  - Verlorengegangenem Token
  - Vorhandensein mehrerer aktiver Monitore
- In jedem anderen Endsystem ist ein Monitor in Bereitschaft, um größere Zuverlässigkeit und Verfügbarkeit zu erreichen
- Die Fehlerkorrektur benutzt spezielle Managementdateneinheiten
  - Claim Token: benutzt, um neuen aktiven Monitor zu bestimmen
  - Duplicate Address Token: während Initialisierungsphase
  - Active Monitor Present: regelmäßig vom aktiven Monitor versendet
  - Standby Monitor Present: während Initialisierungsphase, um Adresse der Nachbarn zu bestimmen
  - Beacon: beim Beaconing-Prozess (Fehlersituation)
  - Purge: benutzt von neuem aktiven Monitor, um alle Endsysteme zurückzusetzen

# Zirkulierende Daten oder Token mit hoher Priorität

- Sender erzeugt Daten oder Token mit Monitorbit M=0
- Der aktive Monitor setzt das Monitorbit M auf 1, wenn die Daten bzw. das Token ihn durchlaufen
- Falls der aktive Monitor Daten oder Token mit M=1 sieht, löscht er alle Daten auf dem Ring
  - Wie kann es zu solch einer Situation kommen?
- Nachdem das Löschen beendet ist, erzeugt der aktive Monitor ein neues Token

# Verlorenes Token

- Der aktive Monitor benutzt einen Timer, um den Verlust eines Tokens oder den Verlust von Daten zu entdecken
- Wann wird dieser Timer gestartet? Wie muss er dimensioniert sein?
- Falls der Timer abläuft, löscht der aktive Monitor alle Daten auf dem Ring und erzeugt ein neues Token

# Einfügen eines Endsystems in den Ring

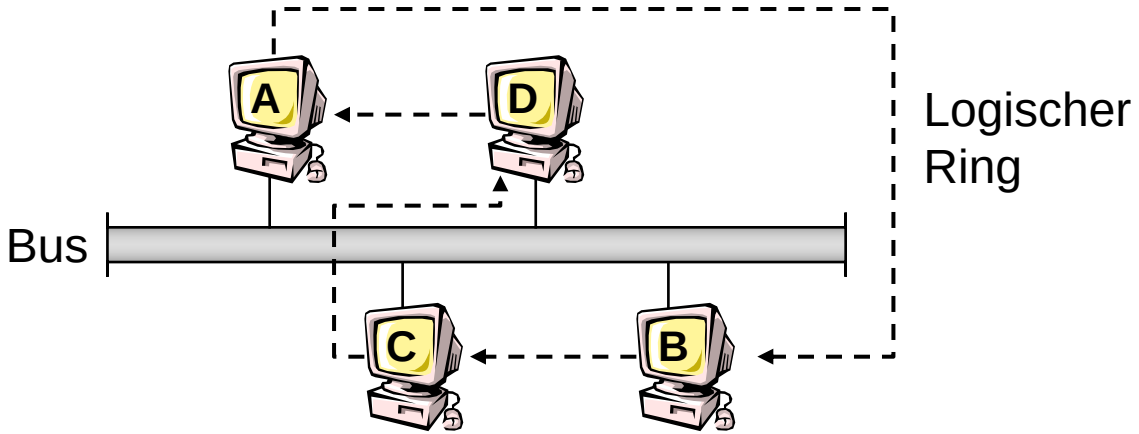
- Verfahren läuft in vier Schritten ab:
  - Selbsttest des neuen Endsystems vor dem Einfügen
  - Einfügen in den Ring und prüfen, ob ein aktiver Monitor vorhanden ist
    - Wieso ist das Überprüfen des aktiven Monitors erforderlich?
  - Test, ob die Adresse im Ring eindeutig ist
  - Teilnahme an der „Neighboring Notification“ Prozedur, um sich bei den Nachbarn bekannt zu machen

# Eigenschaften des Token-Ring

- Erlaubt eine strukturierte Verkabelung von Gebäuden
- Zuverlässigkeit
  - Fehlerhafte Endsysteme können isoliert und aus dem Ring ausgeschlossen werden
- Verteilte Token-Steuerung, d.h. dezentrale Zuteilungsstrategie
- Neue Generierung der Rechteckimpulse in jedem Endsystem
  - Dadurch wenig rauschempfindlich
  - Große Ringe mit vielen Endsystemen möglich
- Flexibilität
  - Konfiguration gemäß den Ansprüchen der Benutzer
  - Ortswechsel von Systemen problemlos
- Natürliche Anwendung für Glasfaser

# Token-Bus

- Token-Bus (IEEE 802.4)
  - Verbindet Vorteile von Ethernet und Token-Ring
    - Einfache und robuste Busverkabelung wie bei Ethernet
      - Ausfall einer Station durchtrennt nicht den Bus
        - kein komplexer Ringverteiler
    - Garantierte Antwortzeiten durch zirkulierendes Senderecht (Token) wie bei Token-Ring
      - Kann Ethernet nicht bieten
  - Aufbau
    - Physikalische Verbindung aller Stationen durch Bus
    - Bildung eines logischen Rings



# Funktionsweise Token-Bus

- Zirkulierendes Senderecht (Token) ähnlich wie bei Token-Ring
  - Übertragung zwischen den einzelnen Teilnehmern über den Bus
    - Broadcast-Medium (im Gegensatz zum Token-Ring)
- Im logischen Ring sind Vorgänger und Nachfolger frei festgelegt
  - Token muss adressiert werden
    - Abweichendes Format des Tokens
  - Nur die Nachfolgestation darf Token entgegennehmen
    - Falls Daten zu senden sind, kann dies dann bis zum definierten Maximum (Token Hold Timer) geschehen
    - Falls nichts zu senden ist, wird neues Token für eigene Nachfolgestation erzeugt
      - Danach wird auf dem Bus auf Aktivität der Nachfolgestation gelauscht, um korrekte Token-Weitergabe sicherzustellen

# Weitere Eigenschaften Token-Bus

- Token-Management unterscheidet sich von Token-Ring
  - Durch Bus-Medium begründet
  - Z.B. Generierung des initialen Tokens oder Behandlung von verlorenen Token unterschiedlich zu Token-Ring
- Prioritätsmechanismus ähnlich Token-Ring
  - Allerdings nur vier unterschiedliche Prioritätsklassen
- Anwendungsgebiete
  - Aufgrund deterministischer Sendezeitgarantien v.a. in der Produktionssteuerung, Fabrikautomatisierung

# Vergleich von CSMA/CD und Token-Ring

- CSMA/CD
  - Vorteile
    - Sehr weit verbreitet, dadurch sehr große Erfahrung
    - Einfaches Protokoll
    - Installation im laufenden Betrieb einfach möglich
    - Passive Kabel
    - Keine Verzögerung bei niedriger Last
  - Nachteile
    - Minimale Größe einer Dateneinheit von 64 Byte, maximal 1.500 Byte
    - Nicht deterministisch, keine Prioritäten
    - Begrenzte Kabellänge
    - Geringe Effizienz durch viele Kollisionen problematisch bei höherer Last
- Token-Ring
  - Vorteile
    - Automatische Erkennung und Elimination von Kabelbruch durch Ringverteiler
    - Prioritäten möglich
    - Kurze Dateneinheiten möglich, Länge nur durch THT begrenzt
    - Sehr guter Durchsatz und hohe Effizienz unter hoher Last
  - Nachteile
    - Zentralisierter Monitor
    - Unnötige Verzögerung unter niedriger Last
    - Fehlerhafter Monitor kann den gesamten Ring in Mitleidenschaft ziehen
    - Hoher Aufwand durch das Ringmanagement

# 7.2 Strukturierung der Datenübertragung

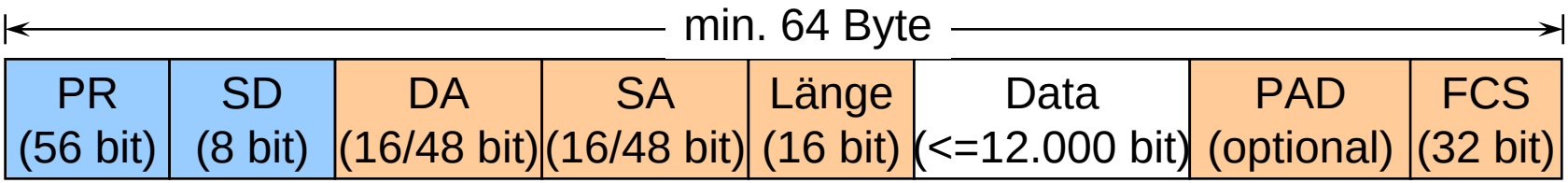
- Ausgangssituation
  - Zu übertragende Daten werden in so genannte Dateneinheiten gegliedert
- Problematik
  - Anfang und Ende der Dateneinheiten muss erkannt werden
  - Dateneinheiten müssen neben den reinen Nutzdaten noch Zusatzinformation transportieren (z.B. Adressen)
  - Grundlegende Gliederung
    - Kopf
    - Nutzdaten
    - Anhang

# Strukturierung der Datenübertragung (2)

- Wichtig: Transparenz
  - Ziel: Übertragung jeglicher Kombination von Daten der darüber liegenden Schicht
- Beispiele für Formate von Dateneinheiten
  - CSMA/CD (Ethernet)
  - Token-Ring

⇨ Die meisten grundlegenden Prinzipien lassen sich auf andere Protokolle übertragen (auch anderer Schichten)

# Format von CSMA/CD-Dateneinheiten




- PR = Präambel zur Synchronisation (1010101010...)
- SD = *Start-of-frame Delimiter* zeigt Beginn an (10101011)
- DA = *Destination Address*, Zieladresse
- SA = *Source Address*, Quelladresse
- Länge = Anzahl der Oktette im Datenfeld
- Data = Datenfeld, das maximal 1.500 Oktette umfassen darf
- PAD = *Padding*, um zu kurze Datenfelder auf die nötige Länge zu ergänzen
- FCS = *Frame Check Sequence*, Polynomdivision mittels CRC32-Polynom zur Fehlererkennung

- Wichtig
  - Einzelne Realisierungen von CSMA/CD (z.B. Ethernet 1.0, Ethernet 2.0 oder IEEE 802.3) verwenden manche Felder in leicht unterschiedlicher Bedeutung

# Format der Token-Ring-Dateneinheiten

## Token:

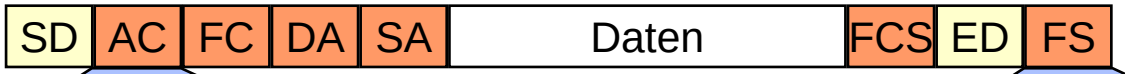
1 1 1 Byte

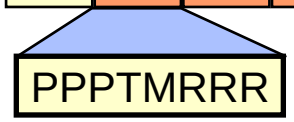



| | | | |
|---|---|---|---|
| SD | Start Delimiter | AC | Access Control |
| FC | Frame Control | DA | Destination Address |
| SA | Source Address | FCS | Frame Check Sequence |
| ED | End Delimiter | FS | Frame Status |

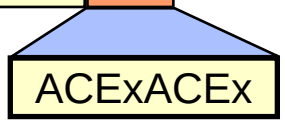
## Dateneinheit mit Nutzdaten:

1 1 1 2/6 2/6 typ. < 5000 4 1 1 Byte



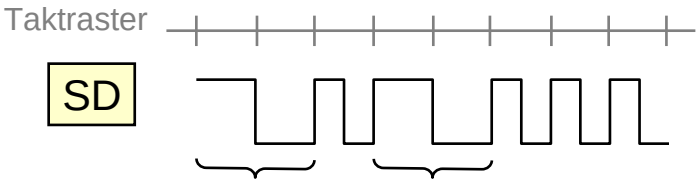




P: Prioritätsbit
T: Token oder Daten
M: Monitorbit
R: Reservierungsbit (für Priorität)




A: Adresse erkannt
C: Daten kopiert
E: Fehler entdeckt

## Transparenz und Rahmenerkennung durch Coderegelverletzung:



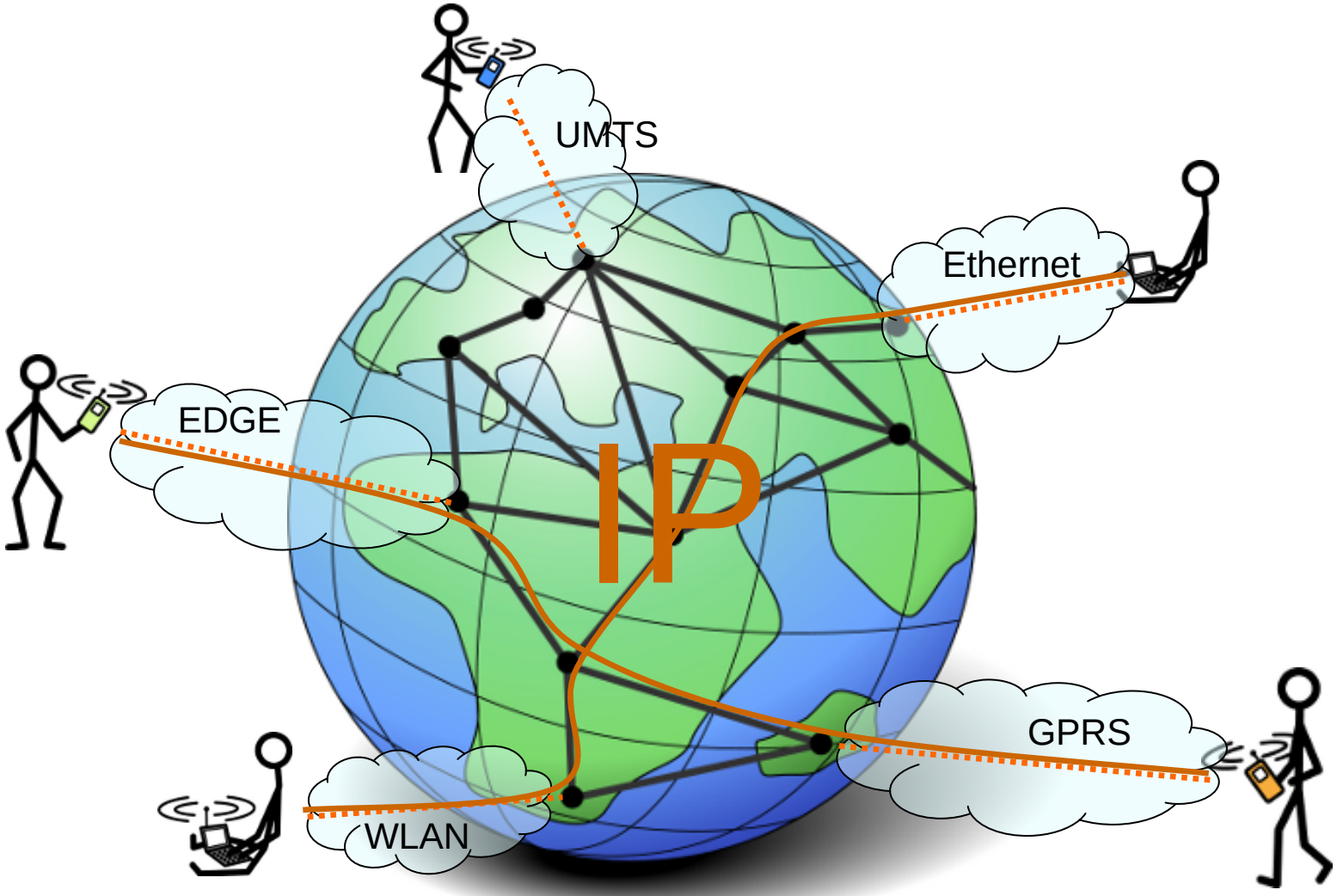

Vorgeschriebener Signalwechsel in der Mitte des Taktrasters fehlt
⇨ Verletzung der differenziellen Manchester-Codierung (ähnlich in ED)

**Folge:** keine Mechanismen wie Stopfen/ Längenfeld nötig, um Transparenz und Erkennung der Rahmengrenzen zu gewährleisten.

# Zusammenfassung

- Medienzugriff (MAC)
  - Konkurrierend
    - ALOHA
    - Slotted ALOHA
    - CSMA/CD
  - Kontrolliert
    - Token-Ring
    - Token-Bus
- Dateneinheiten
  - CSMA/CD-basierte Verfahren
  - Token-Ring
- Ausblick (Mobilkommunikation, Sensornetze)
  - MAC-Verfahren für drahtlose und mobile Szenarien
  - MAC-Verfahren für Sensornetze

# Ausblick: Welt von heterogenen Netzen



# Übungen

7.1 Geben Sie verschiedene Topologien für lokale Netze an.

7.2 Nennen Sie unterschiedliche Medienzuteilungsverfahren und kategorisieren Sie diese.

7.3 Beschreiben Sie die Funktionsweise von CSMA/CD.

7.4 Kann bei CSMA/CD eine Kollision auftreten? Falls ja, wie? Warum existiert bei CSMA/CD eine minimale Länge der Dateneinheiten?

7.5 Beschreiben Sie die Prinzipien der Zuteilungsstrategie durch Tokens und zeigen Sie, wie diese Strategie sowohl in einer Ringtopologie eingesetzt werden kann.

7.6 Können in einem Token-Ring Kollisionen auftreten? Falls ja, wie?

7.7 Auf einem Token-Ring entfernt der Sender obsolete Rahmen. Welche Änderungen müsste man vornehmen, damit der Empfänger den Rahmen entfernt? Was hätten die Änderungen für Konsequenzen?

7.8 Gesetzt den Fall, ein Teilnehmer in einem Token-Ring hat Daten auf den Ring gelegt und wird danach irrtümlich aus dem Ringverteiler ausgesteckt. Wie muss der Ringverteiler reagieren? Wie werden die Daten wieder vom Ring genommen?

7.9 Vergleichen Sie Ethernet und Token-Ring bezüglich

- Medienanschluss
- Quittierung einer Nachricht
- Eignung für den Realzeitbetrieb
- Identifizierung der Länge bzw. des Endes eines Rahmens

# Literatur


[BeGa91] D. Bertsekas, R. Gallager; Data Networks; Prentice-Hall 1991, 2nd Edition
- Kapitel 4 (Schwerpunkt liegt hier auf Leistungsanalyse)

[DECI09] DE-CIX – German Internet Exchange; http://www.de-cix.net

[FrJo99] H. Frazier, H. Johnson; Gigabit Ethernet: From 100 to 1000 Mbit/s; IEEE Internet Computing; January/February, 1999

[Hals05] F. Halsall; Computer Networking and the Internet, 2005, 5th Edition
- Kapitel 3

[Heis08] heise Newsticker; http://www.heise.de/newsticker/meldung/105917/

[Kesh97] S. Keshav; An Engineering Approach to Computer Networking; Addison-Wesley, 1997
- Kapitel 7

[KoBu94] P. Kowalk, M. Burke; Rechnernetze; Teubner-Verlag, 1994
- Kapitel 7

[KuRo07] J. Kurose, K. Ross; Computer Networking; Addison Wesley 2007, 4th Edition
- Kapitel 5

[Stal07] W. Stallings; Data & Computer Communications, Prentice Hall, 2007, 8th Edition
- Kapitel 15 und 16

# Kommunikation und Datenhaltung

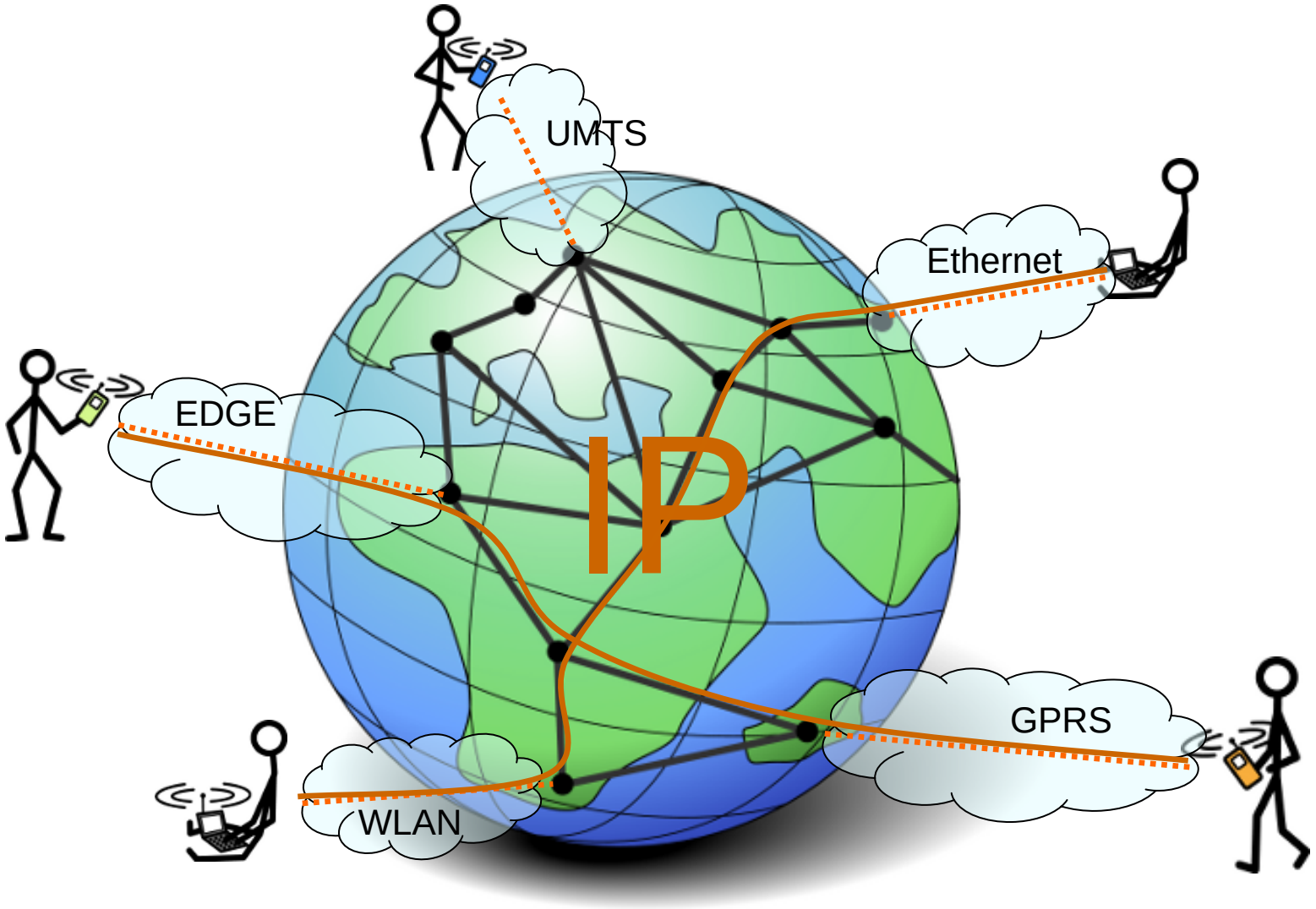
## 8. Netzkopplung und Vermittlung




**Prof. Dr. Martina Zitterbart**
Dipl.-Inform. Martin Röhricht
[zit | roehricht]@tm.uka.de



# Ausblick: Welt von heterogenen Netzen



# Kapitelübersicht

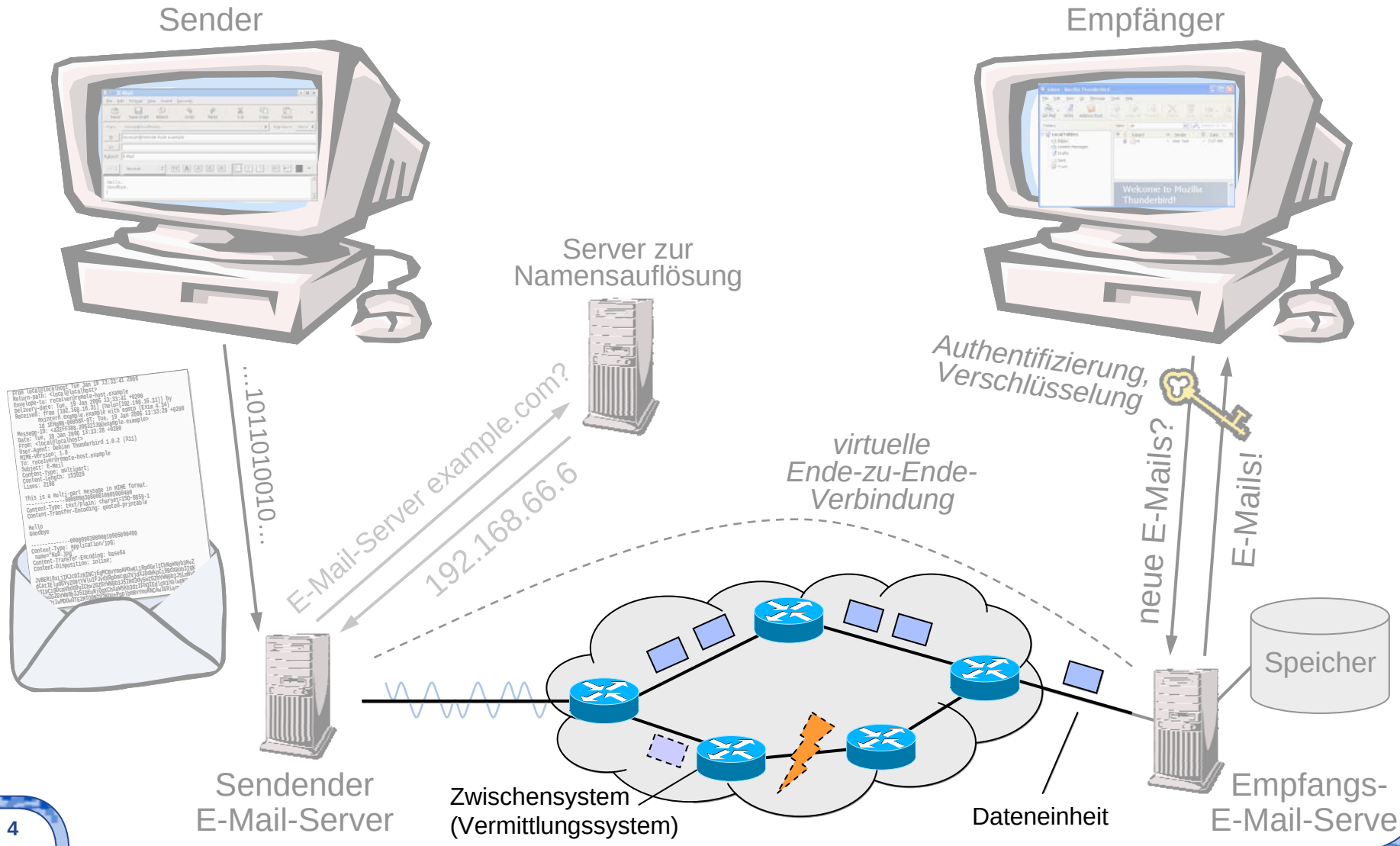
1. Einführung
2. Physikalische Grundlagen
3. Protokollmechanismen
4. Geschichtete Architekturen
5. Sicherungsschicht: HDLC
6. Beschreibungsmethoden
7. Sicherungsschicht: Lokale Netze
8. Netzkopplung und Vermittlung
9. Die Transportschicht
10. Anwendungssysteme
11. Middleware

- 8.1 Allgemeines
- 8.2 Vermittlungstechniken
- 8.3 Netzkopplung auf Schicht 1 & 2
- 8.4 Routing
- 8.5 Vermittlung im Internet

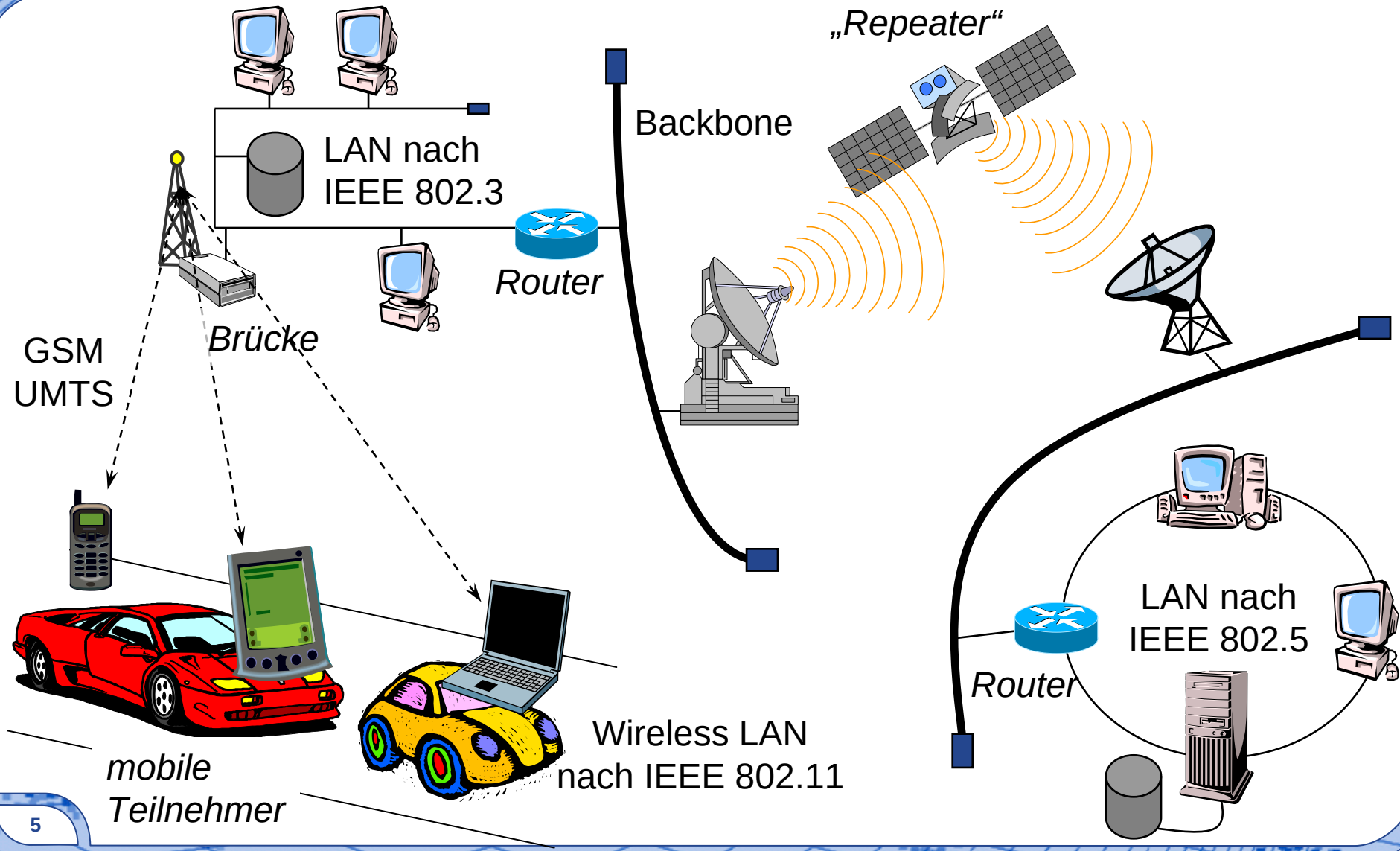
# 8.1 Allgemeines

- Ziel
  - Datenaustausch über große Entfernung und in großen Netzen
- Zentrale Aufgaben
  - Wegewahl
    - erfolgt vor bzw. während einer bestehenden Kommunikationsbeziehung
  - Vermittlung
    - erfolgt für die Nutzdaten während des Datenaustauschs, basierend auf den Ergebnissen der Wegewahl sowie in Abhängigkeit der gewählten Vermittlungstechnik

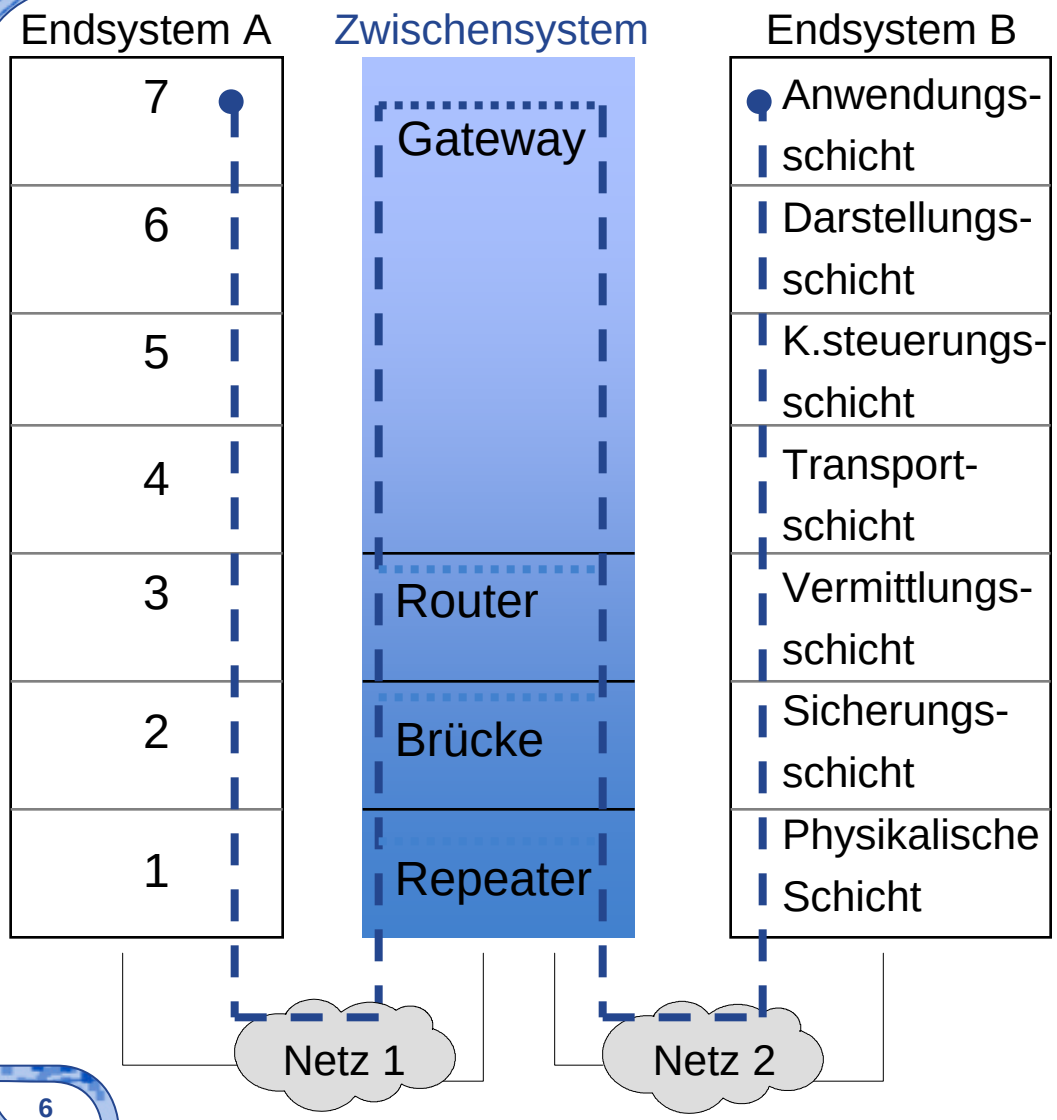
# Netzkopplung und Vermittlung in unserem Beispiel



# Netzkopplung allgemein



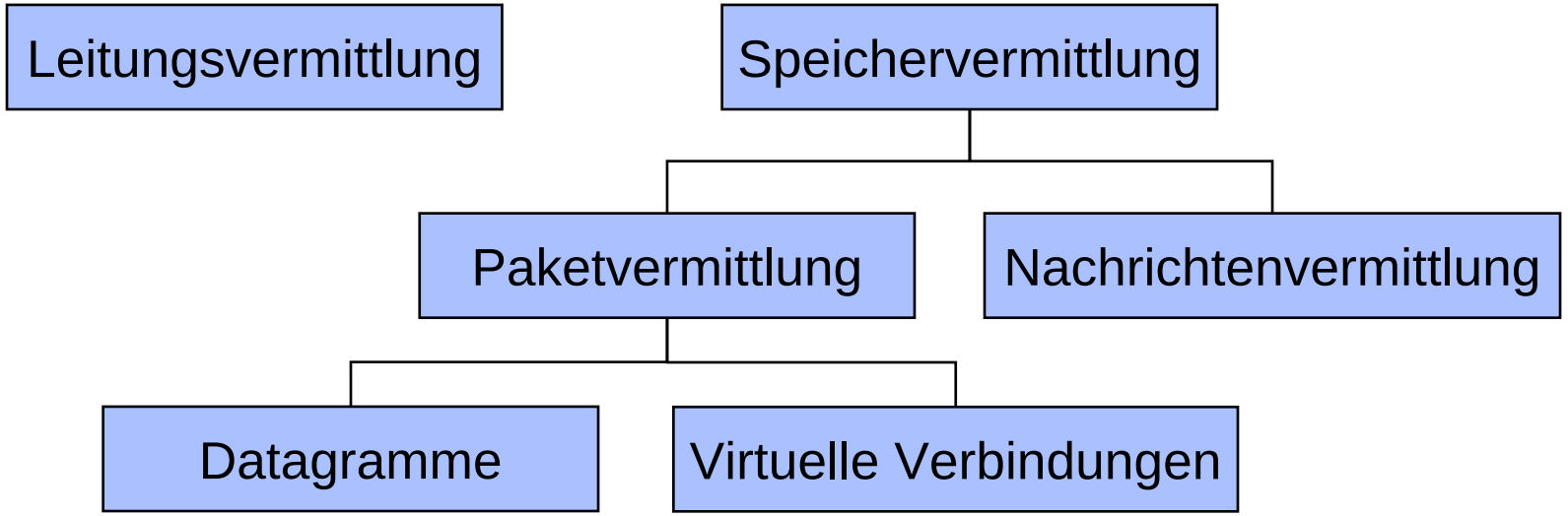
# Typen von Zwischensystemen




- Repeater
  - koppeln auf Schicht 1
  - können Verkehr zwischen Teilnetzen nicht beeinflussen
  - erhöhen physikalische Ausdehnung des Netzes
- Brücken
  - koppeln auf Schicht 2
  - können Verkehr zwischen verschiedenen Teilnetzen separieren
  - erhöhen Anzahl anschließbarer Stationen im Netz
- Router
  - koppeln auf Schicht 3
  - verbinden eigenständiger Teilnetze
  - bessere Sicherheitsvorkehrungen als bei Brücken möglich
- Gateway
  - koppeln auf höheren Schichten

# 8.2 Vermittlungstechniken

- Die folgenden Varianten der Vermittlungstechniken lassen sich unterscheiden
  - Leitungsvermittlung
  - Speichervermittlung
    - Paketvermittlung
      - Virtuelle Verbindungen
      - Datagramme
    - Nachrichtenvermittlung



# Leitungsvermittlung

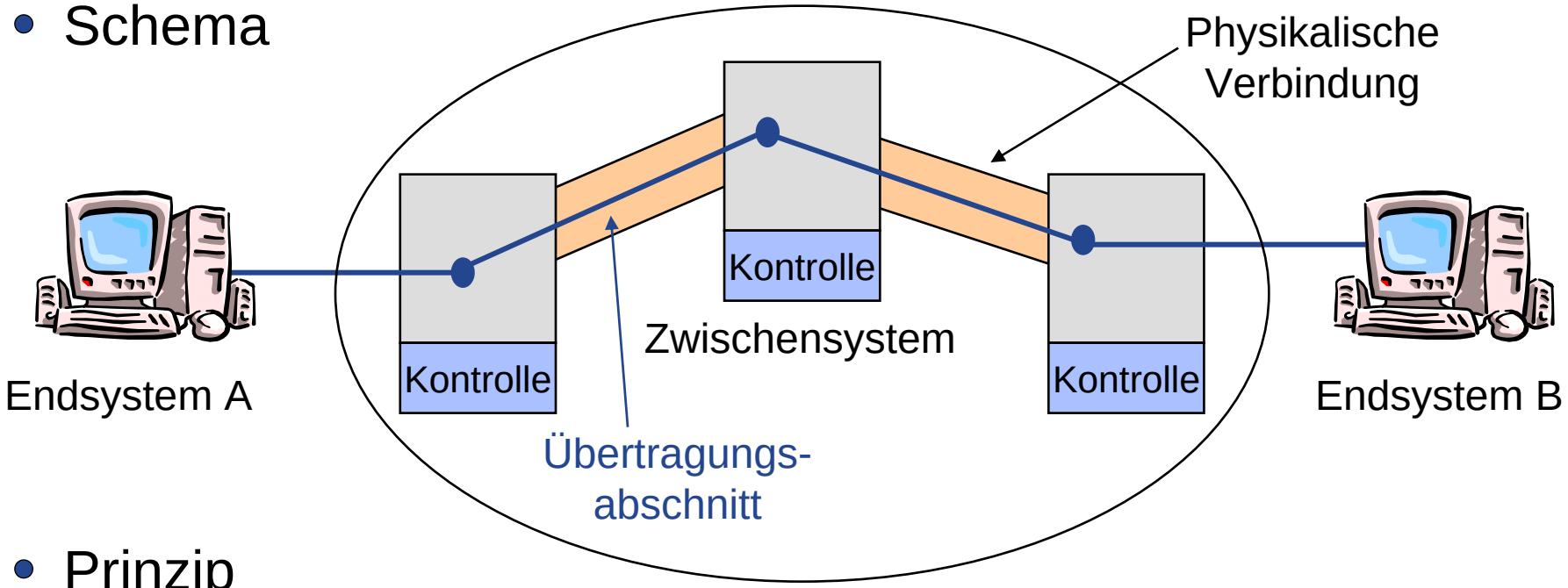
- Eigenschaften
  - Aufbau eines durchgehenden, nicht-speichernden Übertragungskanals (Leitung) zwischen den Endsystemen
  - Übertragungsverzögerungen sind auf physikalisch bedingte signaltechnische Laufzeiten beschränkt
  - Bitfolgen werden reihenfolgetreu übertragen (signalübertragungstechnisch bedingt)
    - Damit wird die Absendereihenfolge beim Empfänger beibehalten (wire-like feature)
  - Vermittlung in den Zwischensystemen erfordert keine zusätzliche Kontrollinformation zur Adressierung
    - z.B. Zieladresse, virtuelle Kanäle

# Leitungsvermittlung

- Einsatz
  - In „klassischen“ Telekommunikationsnetzen
    - z.B. ISDN
- Bemerkung
  - Synonyme Begriffe
    - Leitungsvermittlung
    - Durchschaltvermittlung
    - Circuit Switching
    - Line Switching

# Prinzip der Leitungsvermittlung

- Schema




- Prinzip
  - Verbindungsorientierte Kommunikation
    - Verbindung erhält einen durchgehenden physikalischen Kanal für die ausschließliche Nutzung
      - Hierzu werden aufeinanderfolgende Übertragungsabschnitte miteinander verknüpft
    - Multiplexen mehrerer Verbindungen auf einer physikalischen Leitung mittels starrem Zeitmultiplexen

# Paketvermittlung

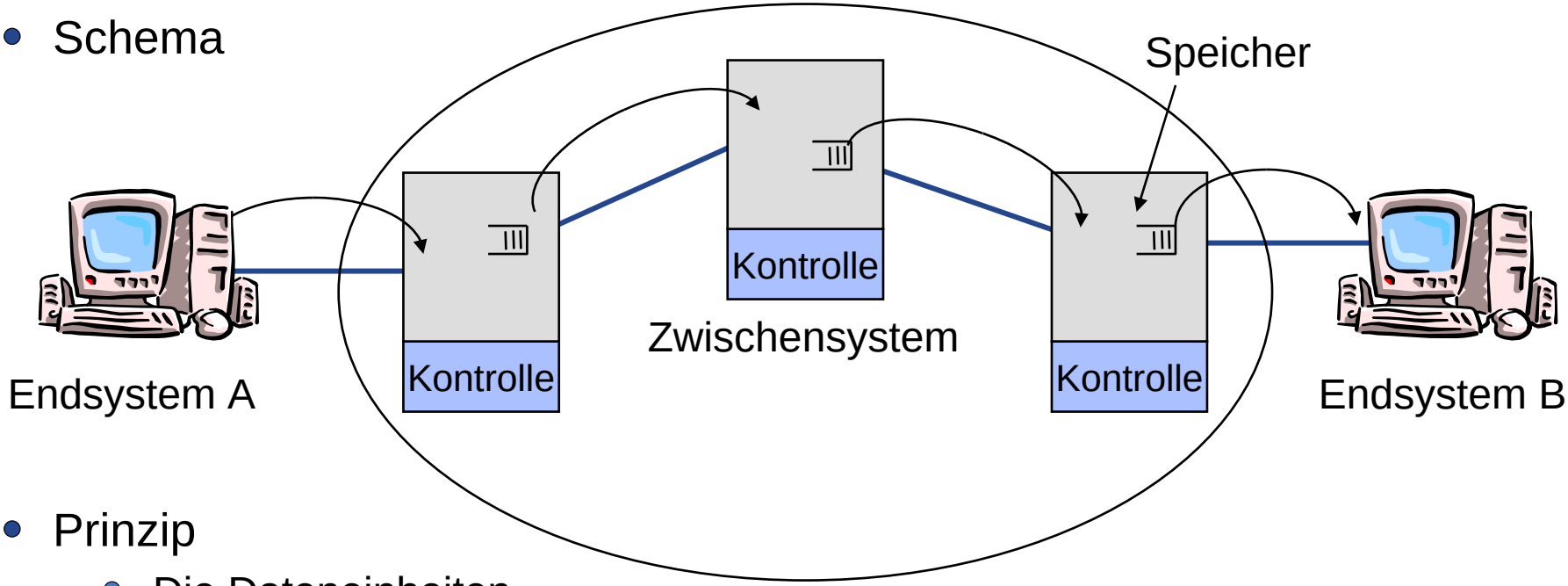
- Charakteristika
  - Vermittlung erfolgt aufgrund von Vermittlungsinformation in den Dateneinheiten
    - Zieladresse (in Datagrammen) oder
    - Lokale Kennung (bei virtuellen Verbindungen)
  - Zwischensysteme verfügen über Speicher
    - Zu vermittelnde Dateneinheiten warten auf das Freiwerden der gewünschten Teilstrecke und werden dazu zwischengespeichert
      - In diesem Sinne gibt es keine Besetzt- oder Belegtfälle
    - Verlust von Dateneinheiten aufgrund begrenzter Speicherkapazität in den Zwischensystemen möglich
    - Wartende Dateneinheiten werden entsprechend der jeweils eingesetzten Scheduling-Verfahren vermittelt
      - Reihenfolgevertauschungen sind hier möglich

# Paketvermittlung

- Charakteristika (Fortsetzung)
  - Es besteht im Allgemeinen keine feste Zeitbeziehung zwischen den einzelnen zu vermittelnden Dateneinheiten
  - Geschwindigkeitsanpassung zwischen unterschiedlich leistungsfähigen Endgeräten ist möglich
- Gliederung
  - Virtuelle Verbindungen
  - Datagramme

# Prinzip der Paketvermittlung

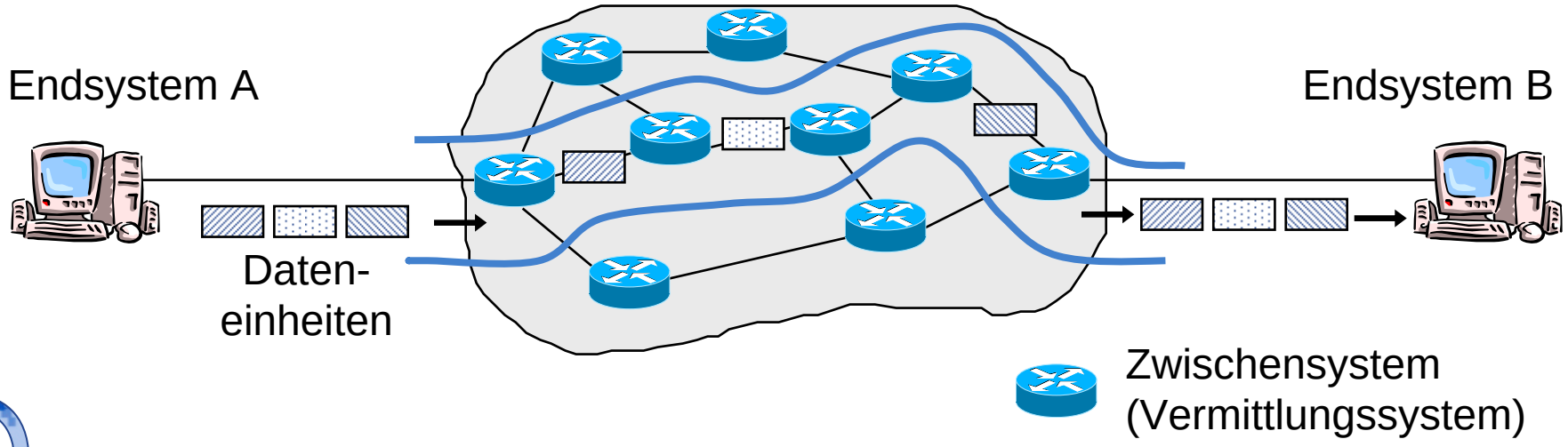
- Schema




- Prinzip
  - Die Dateneinheiten werden abschnittsweise vom Endsystem A an das Endsystem B übermittelt
  - In den Zwischensystemen erfolgt gegebenenfalls eine Zwischenspeicherung (Pufferung)
  - Kommunikation verbindungsorientiert als auch verbindungslos möglich
    - Auswahl des Weges kann entweder beim Verbindungsaufbau oder pro Dateneinheit erfolgen

# Virtuelle Verbindung (Virtual Circuit, VC)

- Eigenschaften
  - Fester Übertragungsweg zwischen zwei Endsystemen
    - Alle Dateneinheiten folgen dem gleichen Weg
    - Reihenfolgeerhaltung gewährleistet
  - Vermittlung basiert auf Dateneinheiten
    - Zusätzlich zu den Nutzdaten muss Vermittlungsinformation transportiert werden
      - häufig als Kennungen bezeichnet
    - Vergleiche Multiprotocol Label Switching (MPLS)
- Schema



# Varianten virtueller Verbindungen

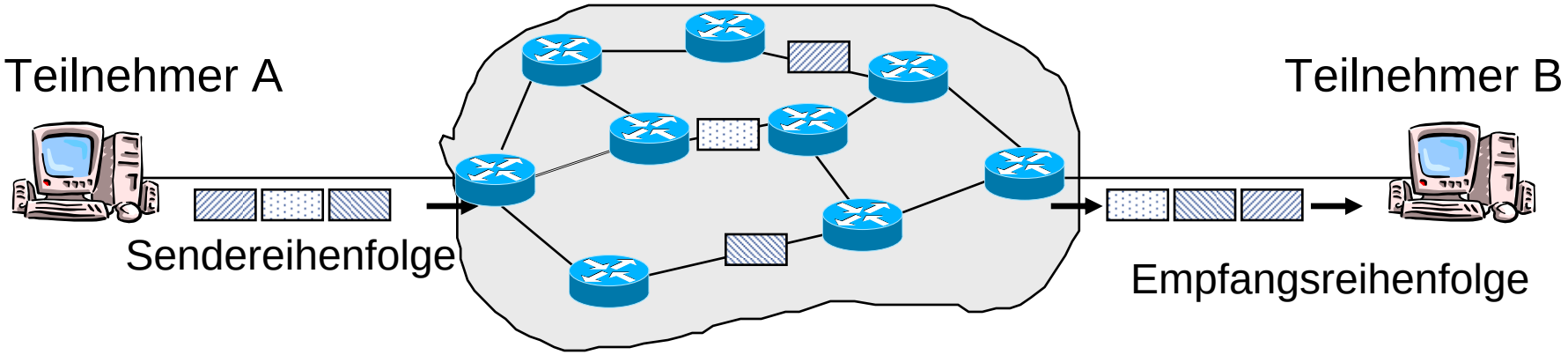
- Varianten
  - Feste virtuelle Verbindung (*Permanent Virtual Circuit*, PVC)
    - Längerfristig eingerichteter Übertragungsweg durch das Netz
      - Aufbau in der Regel durch das Netzmanagement bzw. die Netzadministration
    - Vergleichbar mit einer Standleitung in leitungsvermittelnden Netzen
  - Gewählte virtuelle Verbindung (*Switched Virtual Circuit*, SVC)
    - Virtuelle Verbindung wird bei Bedarf etabliert
      - Hierzu Signalisierungsprotokoll notwendig
    - Datenaustauschphase
    - Abbau der virtuellen Verbindung

# Datagramme

- Dateneinheiten (hier als Datagramme bezeichnet) werden als isolierte Einheiten betrachtet
  - Volle Zieladresse in jedem Datagramm erforderlich
    - Entscheidung über die Wegewahl (d.h. an welchen Ausgang die Dateneinheit vermittelt werden soll) wird pro Dateneinheit getroffen
    - Dateneinheiten können das Netz auf unterschiedlichen Wegen durchlaufen
      - Überholvorgänge möglich
      - Datagramme können ungeordnet beim Empfänger ankommen
- Keine Fehlerüberwachung
- Vorteile
  - Primitiver als virtuelle Verbindungen, daher einfacher zu implementieren
  - Keine Zustandshaltung in Zwischensystemen notwendig
  - Robuster gegenüber Ausfall einzelner Zwischensysteme
  - Keine Verbindungsauf- und -abbau-Phasen
  - Niedriger Mehraufwand für kurzlebige Verbindungen
  - flexibler

# Datagrammvermittlung

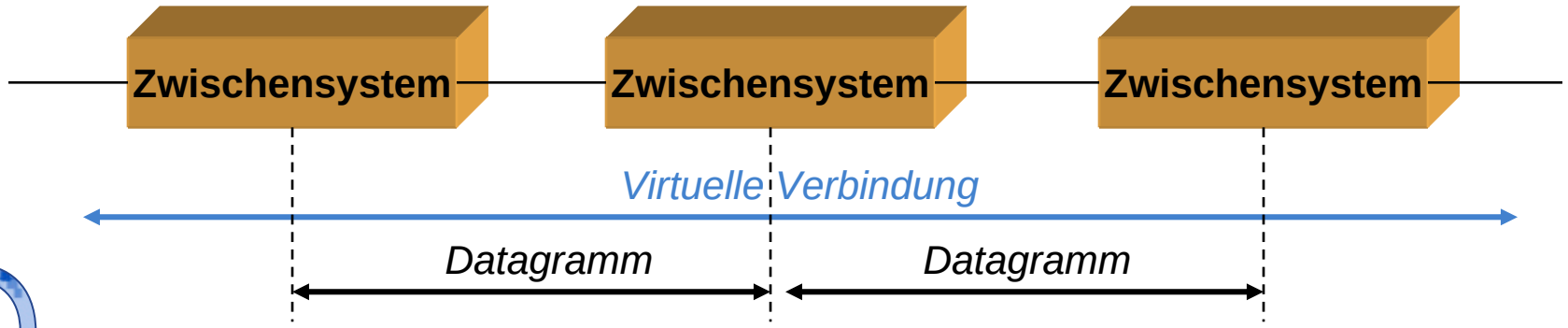
- Schema




Zwischensystem
(Vermittlungssystem)

# Virtuelle Verbindungen vs. Datagramme

- Virtuelle Verbindung
  - Der Weg durch das Netz wird beim Aufbau der virtuellen Verbindung ausgewählt, d.h. für jede neue virtuelle Verbindung findet in jedem Zwischensystem nur einmal eine Wegewahlentscheidung statt
    - Der ganze über diese virtuelle Verbindung fließende Verkehr nimmt denselben Weg
- Datagramm
  - Die Zieladresse bestimmt in jedem Zwischensystem auf dem Weg das ausgehende Interface
    - Für jedes Datagramm wird in jedem Zwischensystem erneut eine Wegewahlentscheidung getroffen
      - Die Datagramme einer Kommunikationsbeziehung können dabei unterschiedliche Wege im Netz durchlaufen



# Virtuelle Verbindungen vs. Datagrammdienst

- Virtuelle Verbindung
  - Zieladresse: nur während des Verbindungsaufbaus benötigt
  - Fehlerbehandlung: für das Endsystem transparent
  - Ende-zu-Ende-Flusssteuerung: ggf. durch das Subnetz bereitgestellt
  - Reihenfolge: Dateneinheiten werden immer in der gesendeten Reihenfolge an das Endsystem übergeben
  - Initialer Aufbau: erforderlich
- Datagramm
  - Zieladresse: in jeder Dateneinheit benötigt
  - Fehlerbehandlung: muss explizit durch das Endsystem erledigt werden
  - Ende-zu-Ende-Flusssteuerung: nicht durch das Subnetz bereitgestellt
  - Reihenfolge: Dateneinheiten werden in der empfangenen Reihenfolge an das Endsystem übergeben
  - Initialer Aufbau: nicht benötigt

# Nachrichtenvermittlung

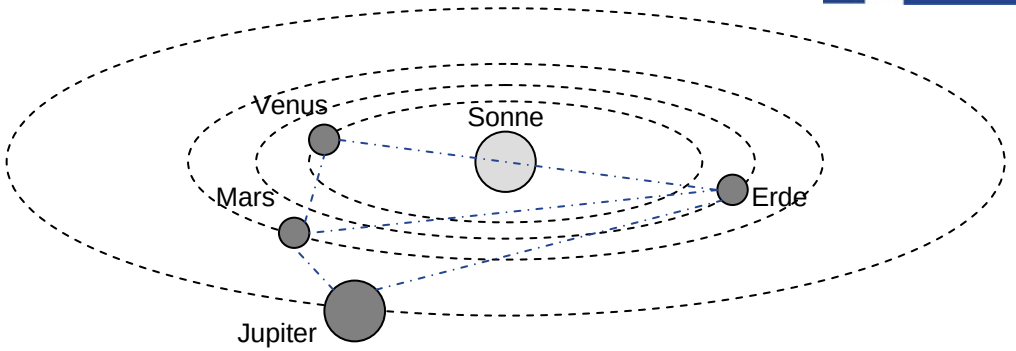
- Charakteristika
  - Einheiten der Vermittlung entsprechen hier anwendungsorientierten Gesichtspunkten und werden oft als Nachricht bezeichnet
  - Eine Nachricht wird typischerweise mittels mehrerer Dateneinheiten vermittelt
    - Segmentierung und Reassemblierung
  - Im Gegensatz zur Paketvermittlung werden Nachrichten in den Zwischensystemen wieder zusammengesetzt (reassembliert)
    - Hieraus folgt, dass alle Dateneinheiten, die zu einer Nachricht gehören, an das gleiche nächste Zwischensystem weitergeleitet werden müssen
    - Die Nachrichten selbst müssen nicht alle gleichem Weg folgen
    - Die Ende-zu-Ende-Verzögerung ist im Vergleich zur Paketvermittlung deutlich höher

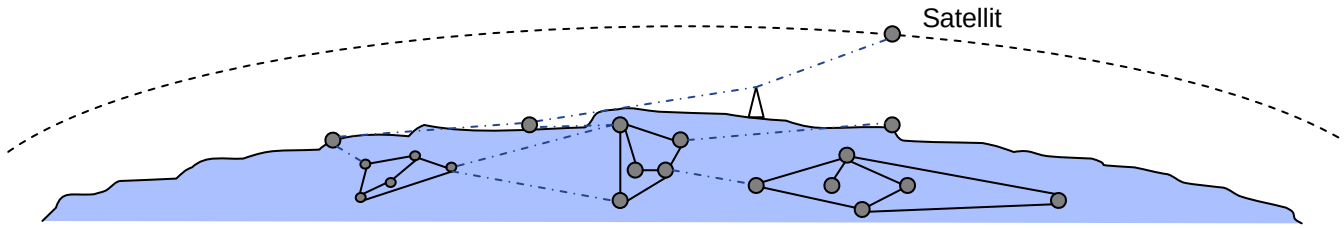
# Anwendungsbeispiel: Delay-Tolerant Networks

- Delay-Tolerant Networks (DTNs)
  - Unterbrochene Verbindungen
    - Große Mobilität einzelner Knoten
    - Gegebenenfalls kein durchgängiger Ende-zu-Ende-Pfad zwischen Quelle und Ziel
  - Lange oder variabel große Verzögerungen
    - Interplanetar bis zu einigen Minuten, trotz Lichtgeschwindigkeit
  - Asymmetrische Datenraten
  - Hohe Fehlerraten
- Einsatzgebiete
  - Interplanetare Netze
  - Mobile Ad-hoc Netze

[Wart03]

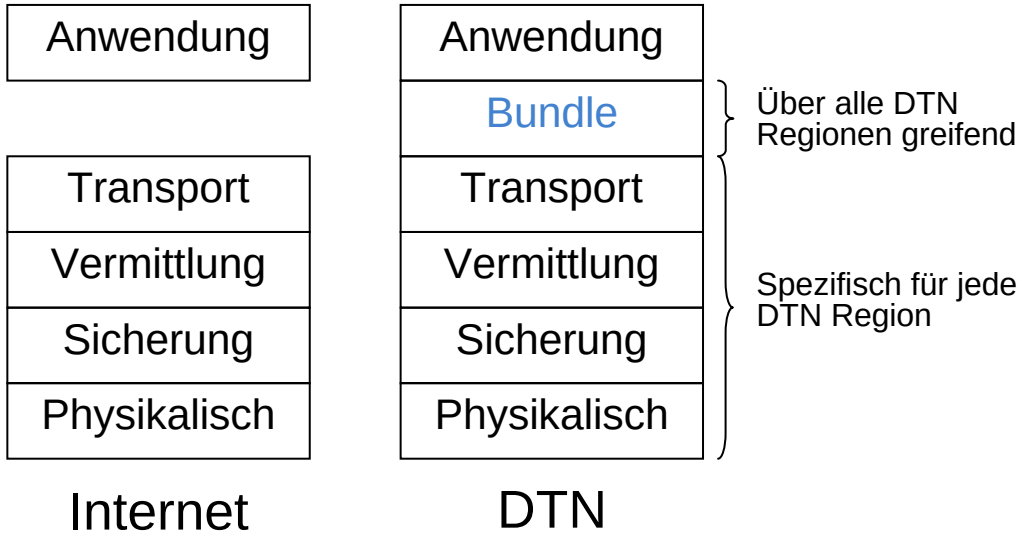
HLK



# Anwendungsbeispiel: Delay-Tolerant Networks (2)

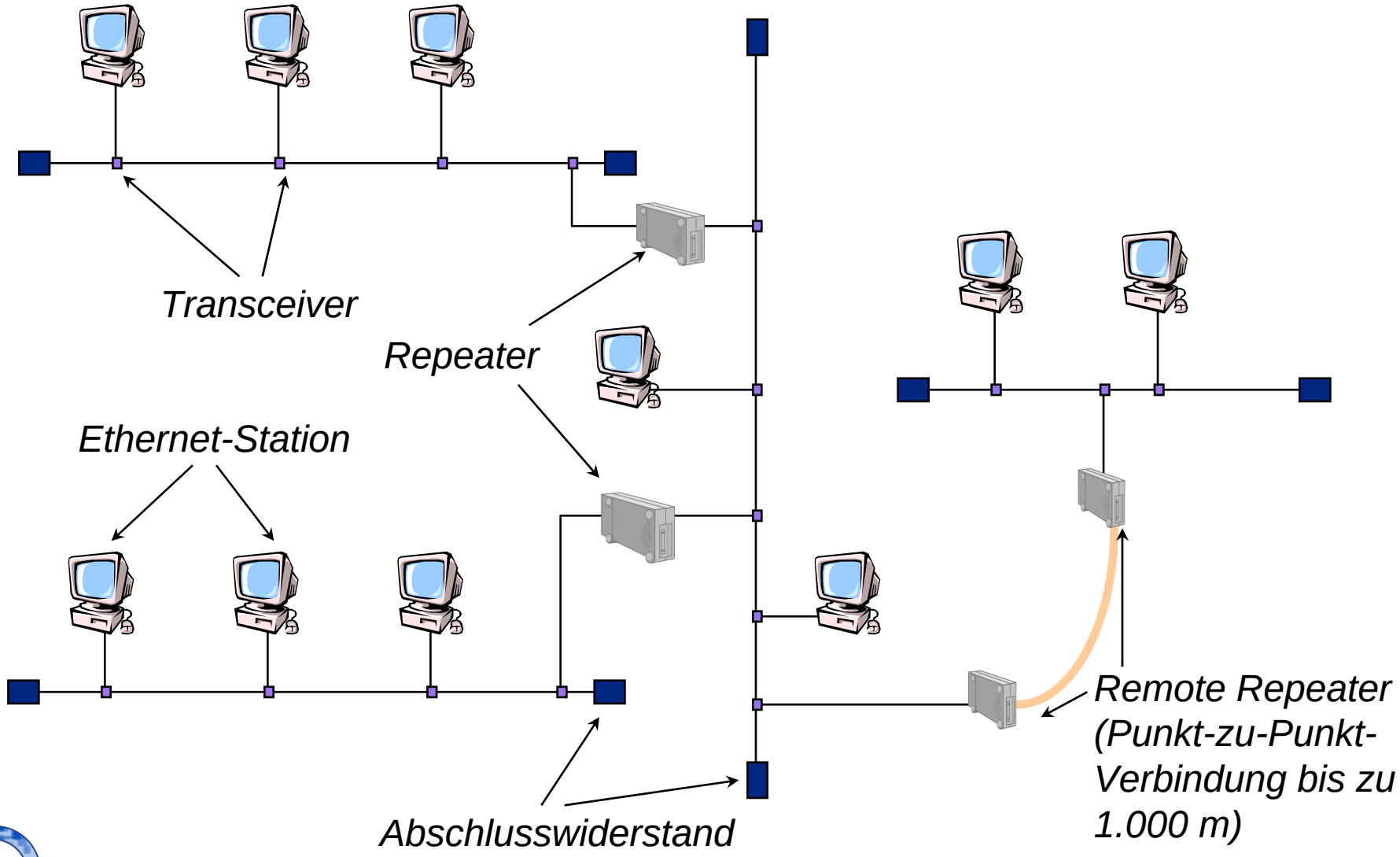
- Speichervermittelte Übertragung
  - Ermöglicht Kommunikation bei Verbindungen, die zwischenzeitlich unterbrochen werden
- Bilden eines Overlays über regionale Netze, inklusive des Internets
  - „Bundle Layer“
- Bundle fasst die regionalspezifischen Schichten zusammen
  - Ermöglicht Kommunikation zwischen verschiedenen Regionen



# Leitungsvermittlung vs. Speichervermittlung

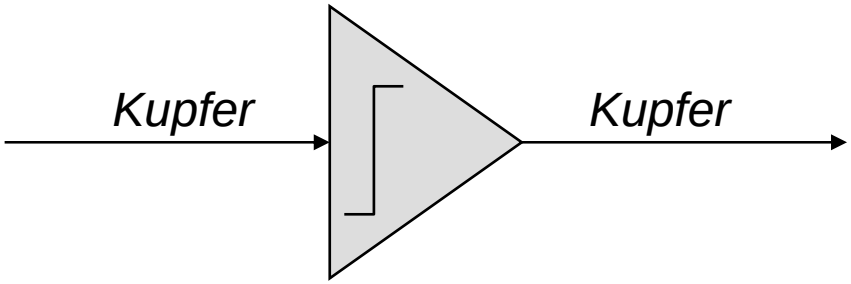
- Leitungsvermittlung
  - starre Leitungszuordnung
  - schlechtes Fehlerverhalten
  - genaue Leistungsvorhersage
  - einfach zu realisieren
  - nur verbindungsorientierter Dienst
  - engl: circuit-switching
  - keine Varianten
  - Beispiele:
    - ISDN
    - GSM
- Speichervermittlung
  - flexible Ressourcenzuordnung
  - flexible Fehlerbehandlung
  - keine Leistungsvorhersage möglich
  - komplexer
  - verbindungsorientierter Dienst und verbindungsloser Dienst („Datagramm“)
  - engl.: store-and-forward-switching
  - Varianten:
    - Nachrichtenvermittlung
    - Paketvermittlung
  - Beispiel:
    - Internet

# 8.3 Netz-Kopplung auf Schicht 1 und 2
# Auf Schicht 1: Beispiel Ethernet

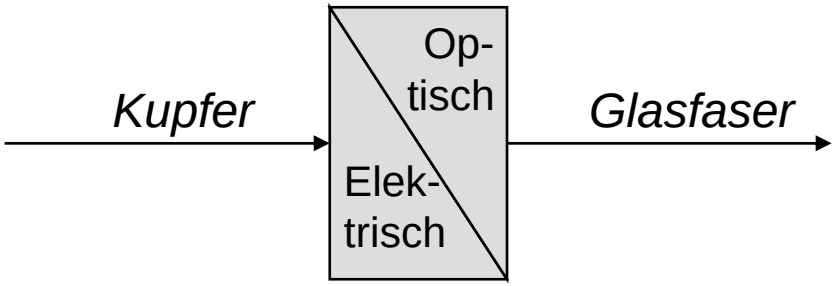


# Repeater

- Kopplung physikalischer Medien
- Keine Zwischenspeicherung
- Keine Bearbeitung der Dateneinheiten
- Medien können unterschiedlich sein, Protokoll auf Schicht 2 muss identisch sein
- Elektrischer Repeater

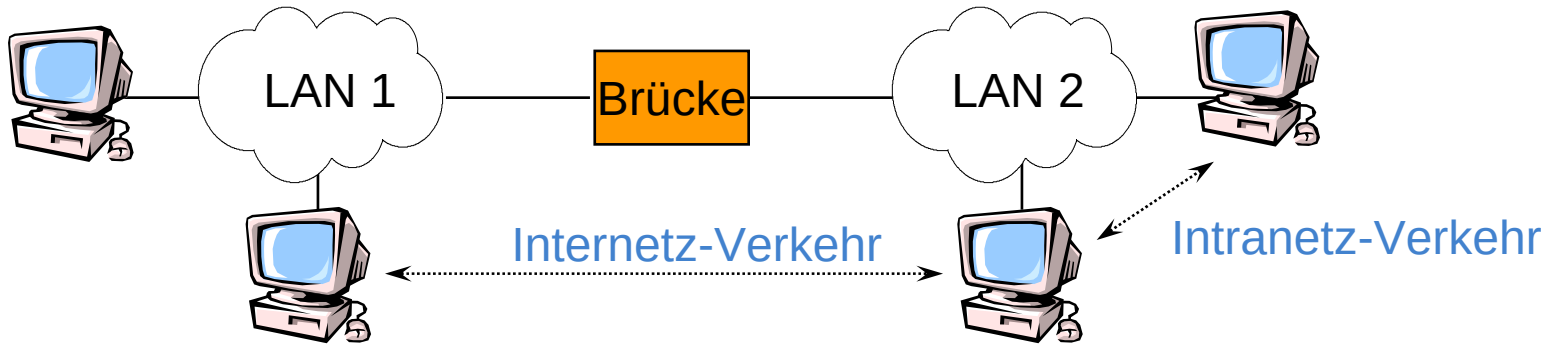



- Elektro-optischer Repeater



# Netz-Kopplung auf Schicht 2: Beispiel Brücken

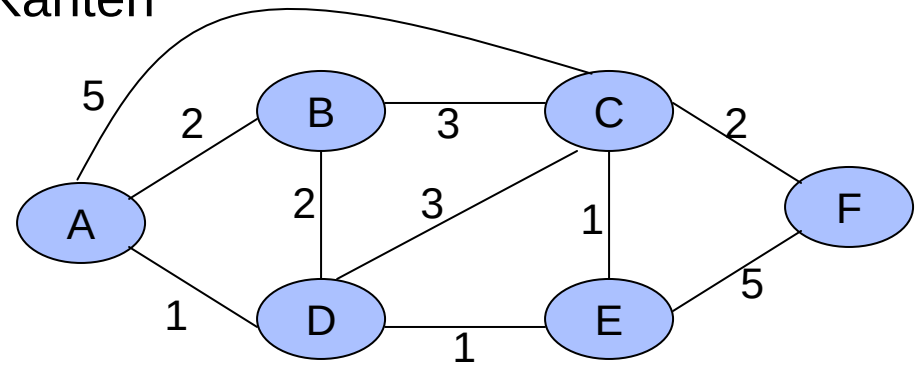
- Ziel
  - Kopplung von lokalen Netzen (LANs) auf Schicht 2
    - homogen: Netzwerke vom gleichen Typ (z.B. 802.x mit 802.x)
    - inhomogen: Netzwerke unterschiedlichen Typs (z.B. 802.x mit 802.y (x≠y))
- Eigenschaften
  - Trennen des Intranetz-Verkehrs in einem LAN von dem Internetz-Verkehr zu anderen LANs (Filterfunktion)
  - Erhöhung der Netzkapazität großer Netze durch Partitionierung
- Schema



# 8.4 Routing

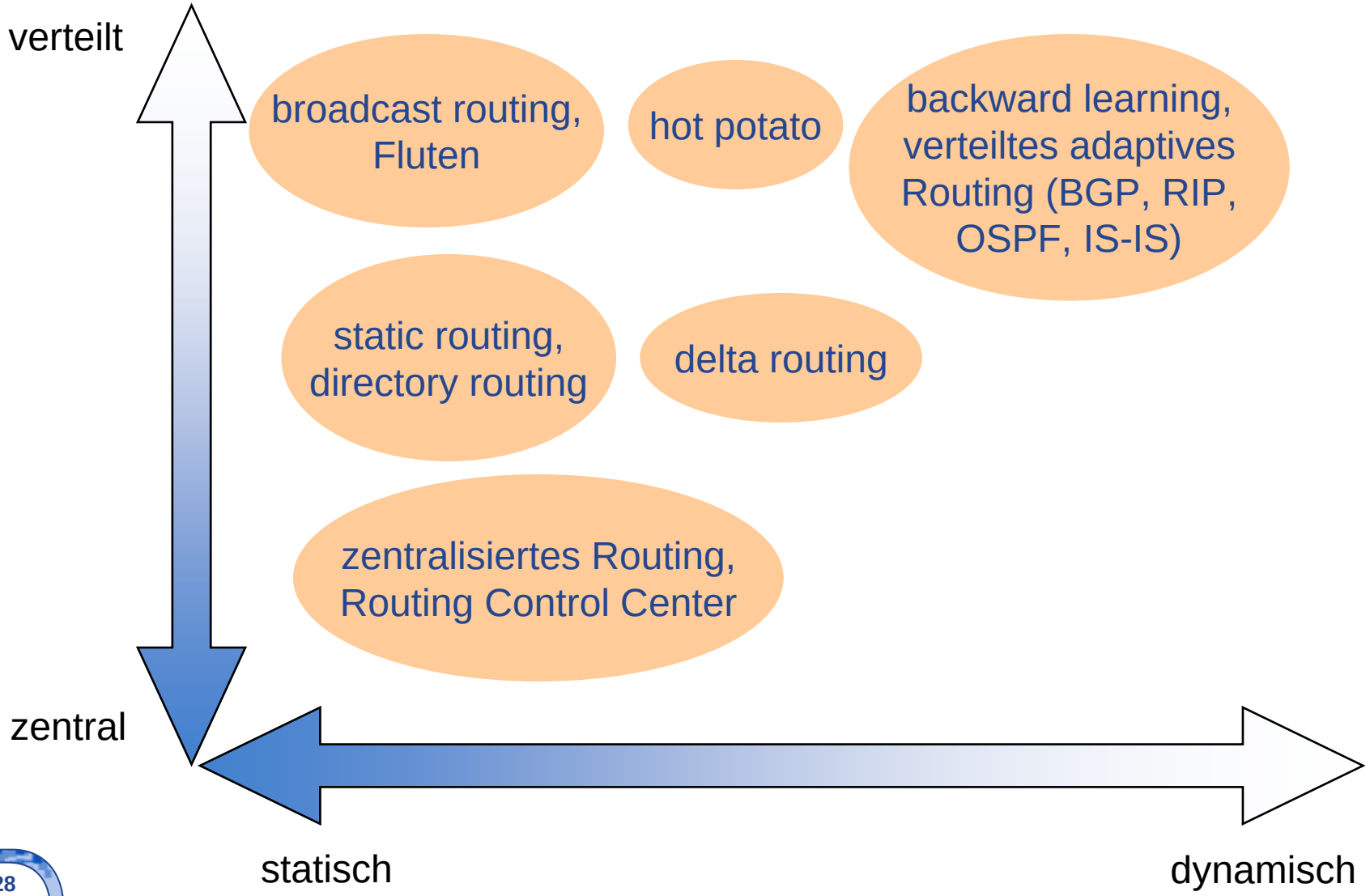
- Ziel: Finden eines „guten“ Wegs
  - Typischerweise ist dies der Pfad mit den geringsten Kosten
  - Andere Metriken sind denkbar
  - Hier: Betrachtung paketvermittelnder Netze
  - ⇨ Aufgabe der Routing-Protokolle

- Modellierung des Kommunikationsnetzes als Graph
  - Router (Zwischensysteme) sind Knoten
  - Übertragungsabschnitte sind Kanten
    - „Kosten“
      - Verzögerung
      - Stausituation
      - Preis ...




- Pfad
  - Folge von Knoten $(n_1, n_2, \ldots, n_k)$, wobei $(n_1, n_2), (n_2, n_3), \ldots (n_{k-1}, n_k)$ Kanten des Graphs sind und kein Knoten mehrfach vorkommt.
  - Pfad mit geringsten Kosten vs. kürzester Pfad

# 8.4.1 Routing-Verfahren im Überblick



# Routing-Verfahren
# Zentralisation und Dynamik

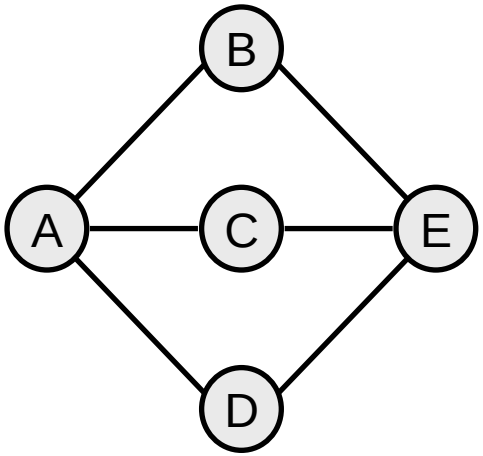
- Zentralisation
  - Wo ist der Routing-Algorithmus lokalisiert?
    - Zentral (in einem Netzkontrollzentrum)
      - Zentrale hat Wissen über das komplette Netz
      - Auch als globales Routing bezeichnet
      - Berechnung optimaler Wege möglich
    - Dezentral (verteilt auf die Vermittlungsknoten)
      - Knoten kennen initial nur ihre Nachbarn
      - Distanz-Vektor-Algorithmen
        - Kein Knoten hat Wissen über das komplette Netz
        - Knoten kennt nie die komplette Route von einer Quelle zu einer Senke
      - Pfad-Vektor-Algorithmen
        - Kein Knoten hat Wissen über das komplette Netz
        - Knoten kennt die vollständige Route bis zur Senke
      - Link-State-Algorithmen
        - Knoten hat Wissen über die gesamte Netztopologie
        - Alle Knoten haben (im stabilen Zustand) die gleiche Sicht auf das Netz

# Statisches Routing – Beispiel

- Beispiel
  - Ziehen einer Zufallszahl x mit $1 > x \geq 0$
    - Falls $x < 0.6$ dann Weiterleiten nach B
    - Falls $0.9 \geq x \geq 0.6$ dann Weiterleiten nach C
    - Sonst Weiterleiten nach D

- Tabelle in Knoten A:

| Ziel | 1. Wahl | | 2. Wahl | | 3. Wahl | |
|---|---|---|---|---|---|---|
| | Kn | Gew | Kn | Gew | Kn | Gew |
| E | B | 0,6 | C | 0,3 | D | 0,1 |
| ⋮ | | | | | | |

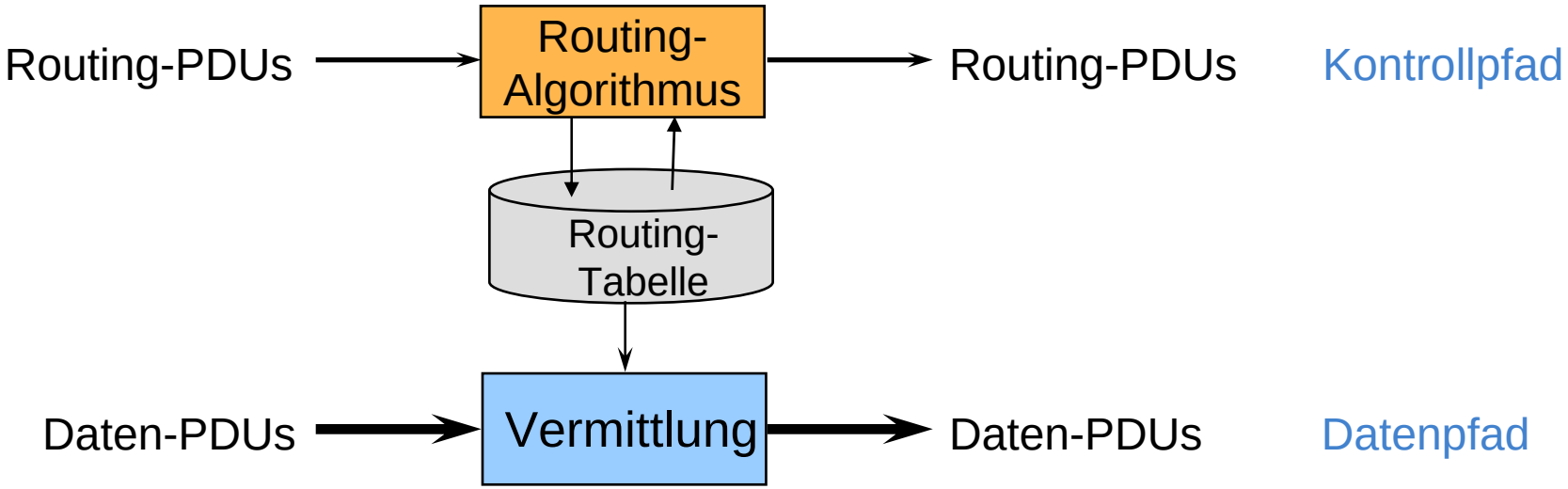


# Routing-Verfahren – Dynamisches Routing

- Wie dynamisch ist das Routing-Verfahren?
  - Nicht adaptiv
    - Routen ändern sich nur sehr selten
      - Oft durch manuelle Änderungen
    - Routenänderungen sind seltener als Verkehrsänderungen
  - Adaptiv
    - Routen ändern sich in Abhängigkeit des Verkehrs bzw. der Netztopologie
      - Aktueller Zustand des Netzes wird damit berücksichtigt
      - Schleifen und Oszillationen in Routen wahrscheinlicher als bei nicht adaptiven Verfahren
    - Können periodisch operieren oder in direkter Reaktion auf Änderungen
    - Zielkonflikt
      - Knoten haben veraltete oder unvollständige Informationen über den Zustand des Netzes
      - Evtl. hohe Belastung durch Austausch von Routing-Informationen

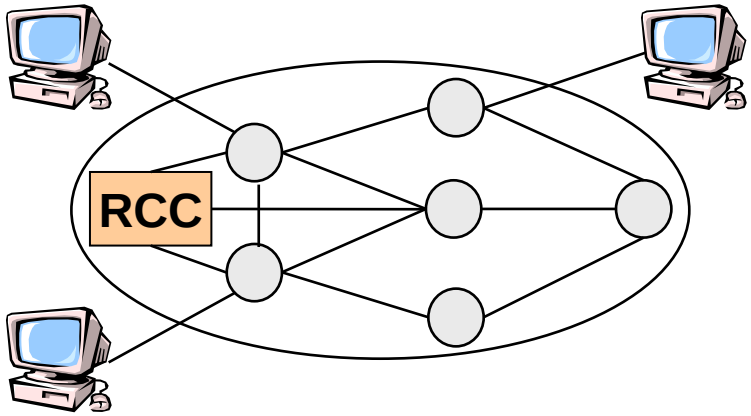
# Router – Kontroll- vs. Datenpfad

- Datenpfad
  - Vermittlung der Daten auf Schicht 3 (Vermittlungsschicht)
- Kontrollpfad
  - Steuert Vermittlung der Daten
  - Routing-Protokolle sind oberhalb der Schicht 3 angesiedelt
    - „Versorgung“ der Routing-Tabelle gemäß dem jeweils eingesetzten Algorithmus
    - Routing-Tabelle enthält Routing-Information, die für das Weiterleiten der Daten ausschlaggebend ist
    - Wegewahl bei der Vermittlung wird anhand der Routing-Information in der Routing-Tabelle durchgeführt



# Zentralisiertes Routing

- Zentralisiertes Routing
  - Adaptives Verfahren
    - Im Netz gibt es ein Routing Control Center (RCC)
    - Jeder Knoten sendet periodisch Zustandsinformationen an das RCC
      - Z.B. Liste aller aktiven Nachbarn
      - Aktuelle Warteschlangenlängen
      - Umfang an Verkehr, der seit dem letzten Bericht abgewickelt wurde
    - Das RCC sammelt diese Zustandsinformationen und berechnet aufgrund dieser Kenntnis über das gesamte Netz die optimalen Wege zwischen allen Knoten (z.B. kürzeste Wege)
    - Jeder Knoten trifft seine Routing-Entscheidungen anhand der ihm zugewiesenen Routing-Tabelle



# Zentralisiertes Routing – Vor- und Nachteile

- Vorteile
  - Das RCC hat theoretisch die vollständige Übersicht und kann perfekte Entscheidungen treffen
  - Knoten müssen keine aufwendigen Routing-Berechnungen durchführen
- Nachteile
  - Für große Netze dauert die Berechnung u.U. sehr lange
  - Ausfall des RCC lähmt das ganze Netz
    - Erfordert daher möglichst robuste Auslegung
      - z.B. durch Redundanz, etwa Backup-Rechner, etc.
  - Globale Inkonsistenzen möglich, da Knoten nahe dem RCC neue Routing-Tabellen wesentlich früher erhalten als die weiter entfernten
  - Starke Belastung des RCC durch die zentrale Funktion

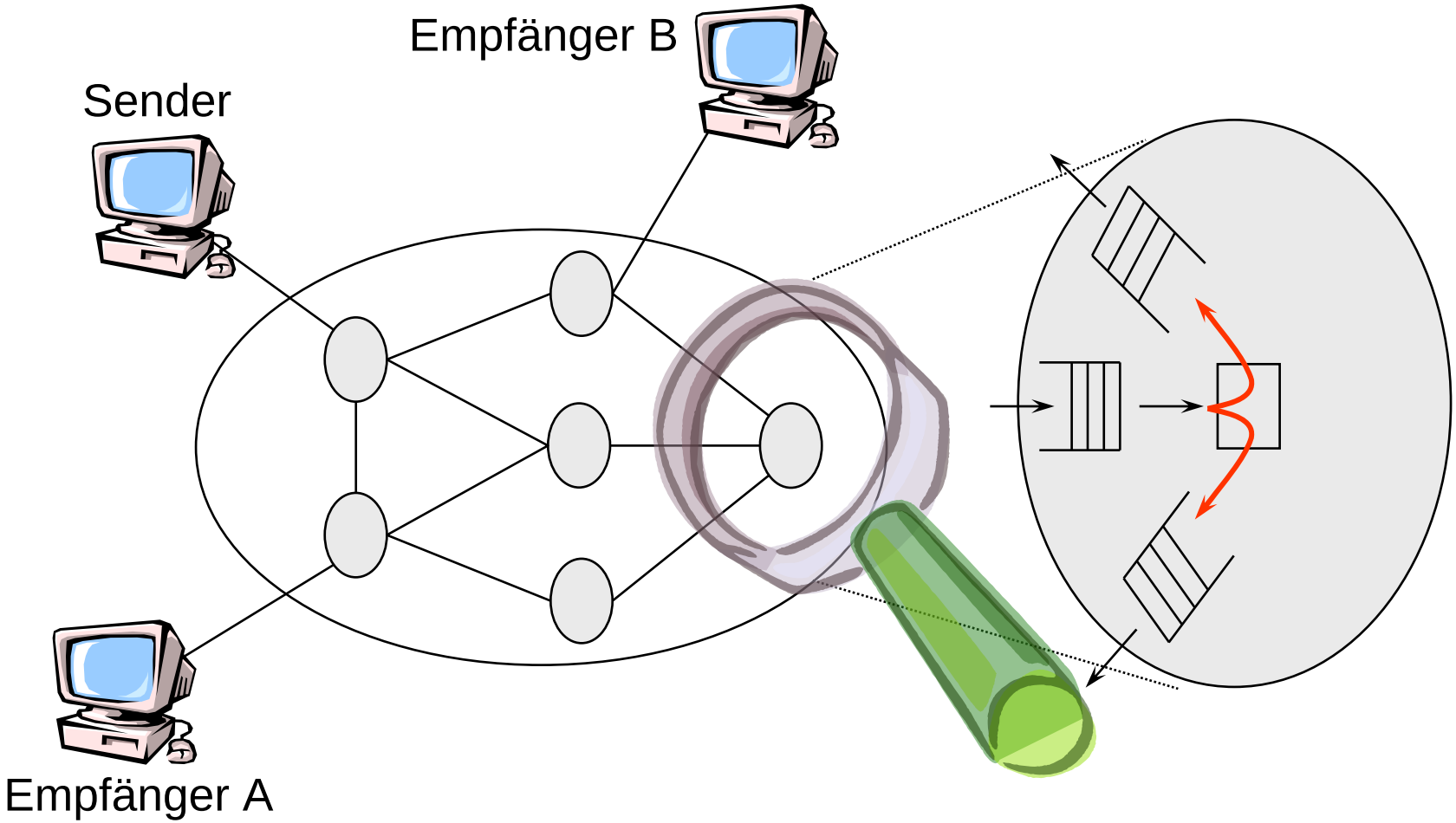
# Isoliertes Routing: Überblick

- Isoliertes Routing (Isolated Routing)
  - Jeder Knoten entscheidet nur aufgrund der Information, die er selbst sammelt
  - Kein Austausch von Routing-Informationen zwischen den Knoten
  - Anpassung an Verkehrs- und Topologieänderungen kann damit nur mit Hilfe beschränkter Informationen erfolgen
  - Unterschiedliche Verfahren
    - Broadcast Routing
    - Hot Potato
    - Backward Learning
    - Delta-Routing (Kombination mit zentralisiertem Routing)

# Broadcast-Routing

- Senden einer Dateneinheit an alle Knoten, z.B. für Umfragen nach bestimmtem Betriebsmittel
- Arten
  - Multidestination Routing
  - Reverse-Path-Forwarding
- Varianten
  - Erstellen einer gesonderten Dateneinheit für jeden Knoten
  - Fluten (Flooding)
    - Einfachstes Verfahren, nicht adaptiv
    - Jede eingehende Dateneinheit wird auf jeder Übertragungsleitung weiter übertragen, außer auf derjenigen, auf der es eintraf

# Broadcast-Routing: Fluten

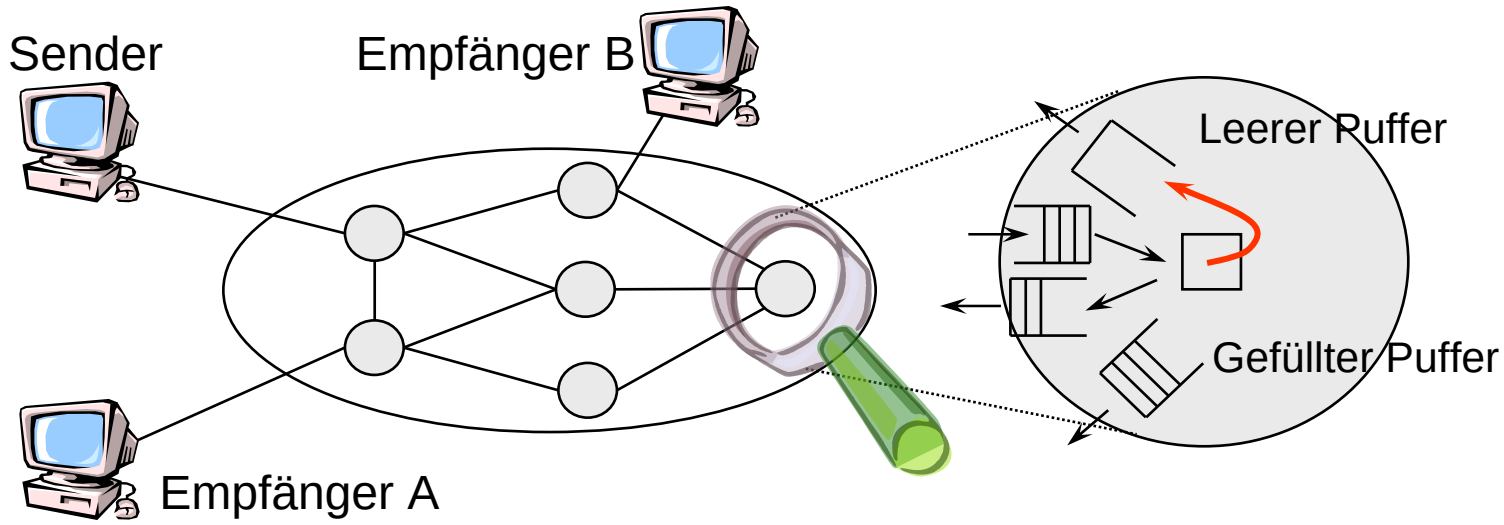


# Broadcast-Routing: Fluten

- Maßnahmen zur Eindämmung der Flut
  - Erkennung von Duplikaten durch Sequenznummern
  - Kontrolle der Lebensdauer einer Dateneinheit durch Zählen der zurückgelegten Teilstrecken (hops)
    - Ein hop-Zähler in der Dateneinheit wird mit der minimalen (idealer Fall) oder der maximaler Weglänge zwischen Quelle und Ziel initialisiert
    - In jedem Knoten wird der Zähler um 1 dekrementiert
    - Falls der Zähler den Wert 0 erreicht, kann die Dateneinheit verworfen werden
- Varianten
  - Selektives Fluten
    - Weiterleitung nicht auf allen, sondern nur auf einigen Übertragungsabschnitten
  - Random Walk
    - Zufällige Auswahl eines Übertragungsabschnittes
- Einsatzbeispiele
  - Mobile Ad-hoc-Netze (MANETs) MK
    - Jedes Gerät ist Endgerät und gleichzeitig Router
  - Drahtlose Sensor-Aktor-Netze DSAN
    - Daten stehen im Mittelpunkt des Interesses

# Hot Potato

- Jeder Knoten versucht, eingehende Dateneinheiten so schnell wie möglich weiterzuleiten
- Dabei wird Übertragungsabschnitt mit der kürzesten Warteschlange ausgewählt
- Varianten
  - Weiterleitung nie auf dem Übertragungsabschnitt, auf dem die Dateneinheit eintraf
  - Kombination mit statischem Routing
    - Auswahl der besten Übertragungsleitung nach statischem Verfahren, solange deren Warteschlangenlänge unter einer bestimmten Schwelle bleibt
    - Auswahl der Übertragungsleitung mit kürzester Warteschlange, falls deren statisches Gewicht nicht zu niedrig ist



# Verteiltes adaptives Routing

- Jeder Knoten tauscht periodisch Routing-Informationen mit jedem seiner Nachbarn aus
- Typischerweise unterhält jeder Knoten eine Routing-Tabelle, die für jeden anderen Knoten im Netz einen Eintrag enthält, hierzu zählen:
  - Bevorzugte Übertragungsleitung für diesen Knoten
  - Schätzung über Zeit oder Entfernung zu diesen Knoten, z.B:
    - Anzahl hops
    - Geschätzte Verzögerung in Millisekunden sowie
    - Geschätzte totale Anzahl von Paketen, die entlang des Weges warten
- Die Schätzungen werden gewonnen aus
  - Zeit oder Entfernung zu den Nachbarn
    - z.B. aus speziellen Echo-Dateneinheiten mit Zeitstempeln
  - Schätzungen der Nachbarn
- Varianten
  - Synchroner Austausch von Routing-Informationen in bestimmten Update-Intervallen
  - Asynchroner Austausch bei signifikanten Änderungen

# 8.4.2 Routing-Protokolle

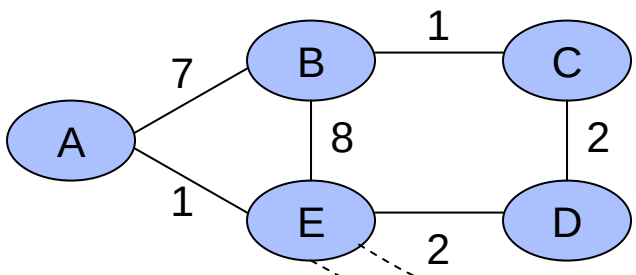
- Im praktischen Einsatz: Verteilte adaptive Routing-Algorithmen
- Dabei werden zwei grundlegende Algorithmen unterschieden
  - Distanz-Vektor-Algorithmen
    - Distanz ist Routing-Metrik
    - jedes System kennt die Distanz zu allen anderen Systemen im Netz
    - hierzu werden die aktuellen Distanzen zwischen den Nachbarn ausgetauscht
    - Problem
      - kürzerer langsamerer Weg wird längerem schnelleren Weg vorgezogen
    - Beispiele
      - Routing Information Protocol (RIP), Distance Vector Routing Protocol (DVRP)
  - Link-State-Algorithmen
    - Unterschiedliche Routing-Metriken möglich
    - berücksichtigt die aktuellen Zustände der Netzanschlüsse
    - jeder Router kennt die komplette Netztopologie und berechnet auf dieser Basis seine Routing-Information
    - Link-State-Algorithmen konvergieren im allgemeinen schneller als Distanz-Vektor-Algorithmen
      - für größere Netze sind sie damit potenziell besser geeignet
    - Beispiele
      - Open Shortest Path First (OSPF), Intra-Domain Intermediate System to Intermediate System Routing Protocol (IS-IS)

# 8.4.2.1 Distanz-Vektor-Routing

- Eigenschaften
  - Verteilt
    - Jeder Knoten erhält Information von seinen direkten Nachbarn, führt eine Berechnung durch und verteilt dann neue Information an seine Nachbarn
  - Iterativ
    - Das Verteilen und Berechnen von Information geht so lange vor sich bis keine Information mehr ausgetauscht wird
  - Asynchron
    - Knoten arbeiten vollständig unabhängig voneinander
- Distanz-Vektor-Tabelle
  - Die grundlegende Datenstruktur für Distanz-Vektor-Algorithmen
    - In jedem Knoten vorhanden
    - Zeile für jedes mögliche Ziel
    - Spalte für jeden direkten Nachbarn
  - Weiterleiten von Daten
    - X will Daten über seinen direkten Nachbarn Z an Y weiterleiten
    - $D^X(Y, Z) = c(X, Z) + \min_w\{D^Z(Y, w)\}$

# Beispiel: Distanz-Vektor-Tabelle

- Beispiel: $D^E(A, D)$
  - Erster Übertragungsabschnitt ist der von E nach D
  - Eintrag in der Tabelle umfasst Kosten von E nach D (2) plus minimale Kosten von D nach A (3)
  - Minimale Kosten von D nach A über Nachbarknoten von D
  - Wie kommt der Wert von 14 für $D^E(A, B)$ zustande?

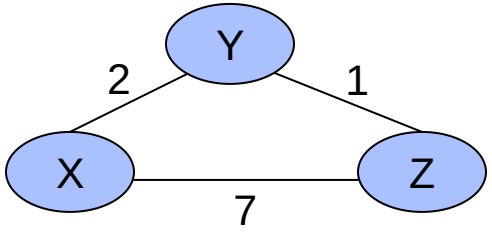



| Ziel \ Next Hop | A | B | D |
|---|---|---|---|
| A | 1 | 14 | 5 |
| B | 7 | 8 | 5 |
| C | 6 | 9 | 4 |
| D | 4 | 11 | 2 |

# Distanz-Vektor-Algorithmus

- Initialisierung
  - Für alle Nachbarn v: $D^X(*,v) = \infty$ , $D^X(v,v) = c(X, v)$
  - Für alle Ziele y: sende $\min_w D(y, w)$ zu jedem Nachbarn, wobei w alle Nachbarn enthält
- Schleife
  - Geänderte Übertragungsabschnittskosten:
    c(X,V) ändert sich um den Wert d (positiv oder negativ)
    - Für alle Ziele y: $D^X(y,V) := D^X(y,V) + d$
  - Update-Nachricht von einem Nachbarn
    - Kürzester Pfad von V zu einem Ziel Y hat sich geändert zu „neuer Wert“
    - $D^X(Y,V) = c(X, V)$ + „neuer Wert“ für dieses Ziel
  - Falls ein neuer $\min_w D^w(Y, w)$ für ein Ziel Y existiert, dann sende diesen Wert zu allen Nachbarn
- Im Folgenden betrachteter Algorithmus ist der Bellman-Ford-Algorithmus

# Beispiel




Annahme:
Knoten arbeiten synchron

| $D^X$ | Y | Z |
|---|---|---|
| Y | 2 | ∞ |
| Z | ∞ | 7 |

| $D^X$ | Y | Z |
|---|---|---|
| Y | 2 | 8 |
| Z | 3 | 7 |

| $D^X$ | Y | Z |
|---|---|---|
| Y | | |
| Z | | |

| $D^Y$ | X | Z |
|---|---|---|
| X | 2 | ∞ |
| Z | ∞ | 1 |

| $D^Y$ | X | Z |
|---|---|---|
| X | 2 | 8 |
| Z | 9 | 1 |

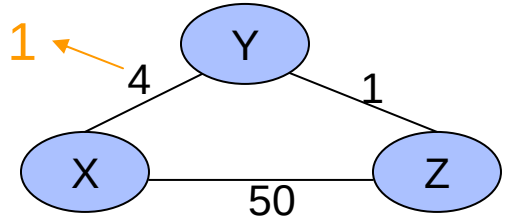
| $D^Y$ | X | Z |
|---|---|---|
| X | | |
| Z | | |

| $D^Z$ | X | Y |
|---|---|---|
| X | 7 | ∞ |
| Y | ∞ | 1 |

| $D^Z$ | X | Y |
|---|---|---|
| X | 7 | 3 |
| Y | 9 | 1 |

| $D^Z$ | X | Y |
|---|---|---|
| X | | |
| Y | | |

# Änderungen der Linkkosten: Good News




| $D^Y$ | X | Z |
|---|---|---|
| X | 4 | 6 |

| $D^Y$ | X | Z |
|---|---|---|
| X | 1 | 6 |

| $D^Y$ | X | Z |
|---|---|---|
| X | 1 | 6 |

| $D^Y$ | X | Z |
|---|---|---|
| X | 1 | 3 |

| $D^Z$ | X | Y |
|---|---|---|
| X | 50 | 5 |

| $D^Z$ | X | Y |
|---|---|---|
| X | 50 | 5 |

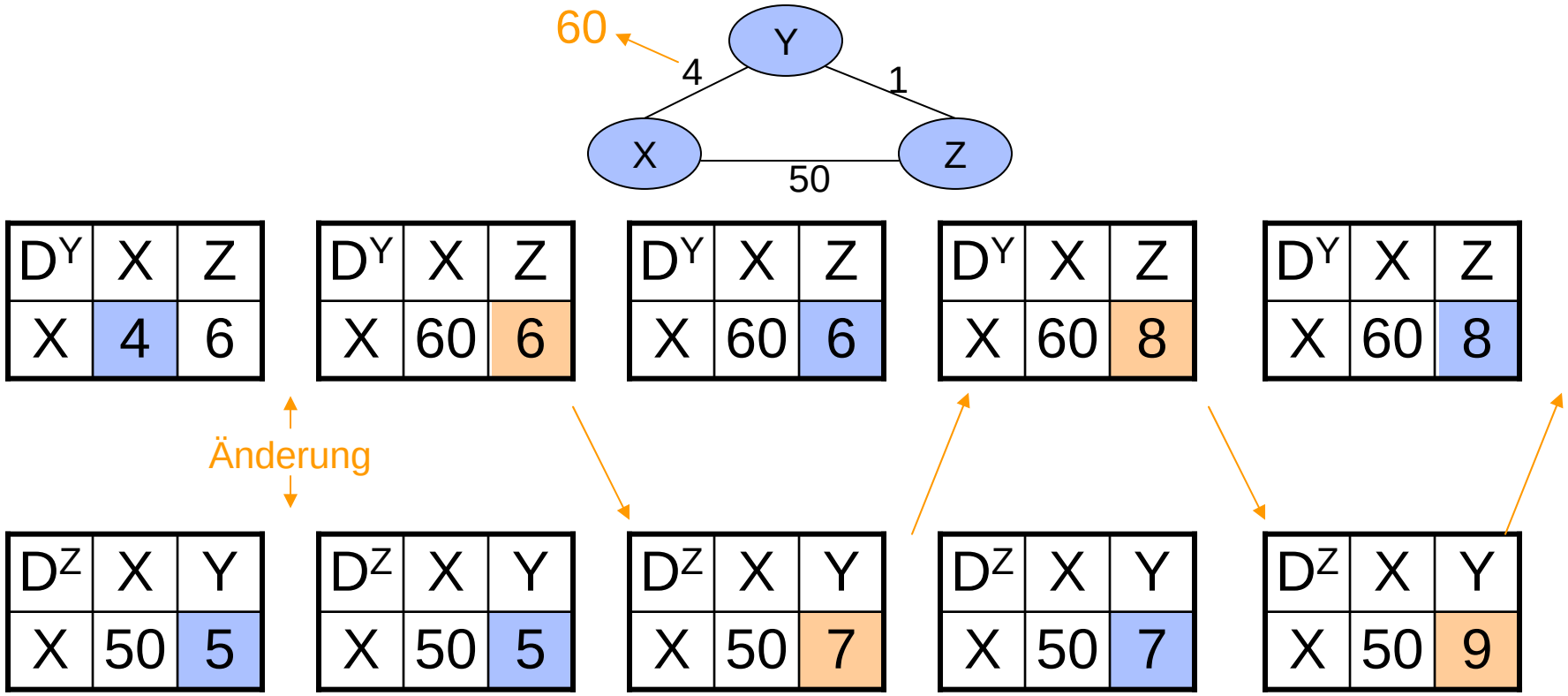
| $D^Z$ | X | Y |
|---|---|---|
| X | 50 | 2 |

| $D^Z$ | X | Y |
|---|---|---|
| X | 50 | 2 |

Änderung

⇨ Die gute Neuigkeit hat sich schnell im Netz ausgebreitet

# Änderung der Linkkosten: Bad News

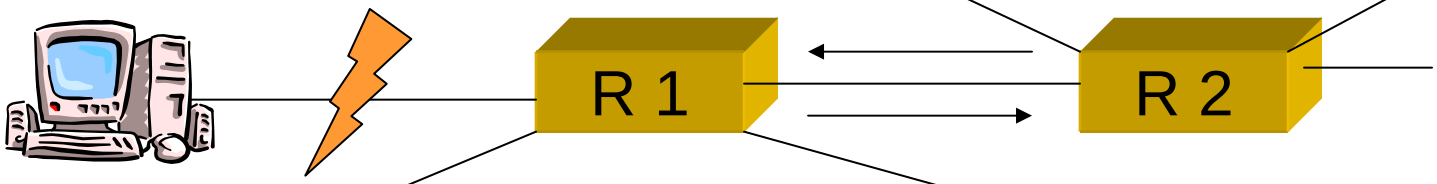



- Schlechte Neuigkeit breitet sich relativ langsam aus und führt u.U. zu Routing-Schleifen
- Schleife hier benötigt 44 Iterationen!

⇨ Count-to-Infinity

# Count-to-Infinity-Problem

| Ziel | Distanz | Nächstes System | Schnittstelle |
|---|---|---|---|
| A | 2 | Router 1 | 1 |
| ... | ... | ... | ... |

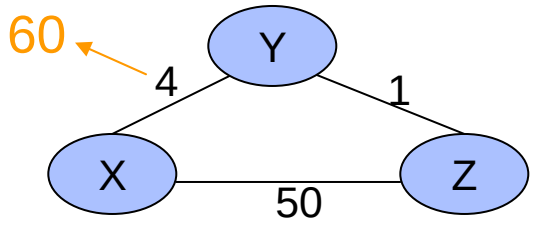



| Ziel | Distanz | Nächstes System | Schnittstelle |
|---|---|---|---|
| A | 3 | Router 2 | 1 |
| ... | ... | ... | ... |

- Zur Vermeidung: Poisonous Reverse

# Poisonous Reverse

- Ziel
  - Vermeidung von den zuvor beschriebenen Routing-Schleifen
- Vorgehensweise
  - Routing-Information wird Y "vorenthalten", wenn Weg über Y kürzer
- Beispiel




| $D^Y$ | X | Z |
|---|---|---|
| X | 4 | ∞ |

| $D^Y$ | X | Z |
|---|---|---|
| X | 60 | ∞ |

| $D^Y$ | X | Z |
|---|---|---|
| X | 60 | ∞ |

| $D^Y$ | X | Z |
|---|---|---|
| X | 60 | 51 |

| $D^Y$ | X | Z |
|---|---|---|
| X | 60 | 51 |

Weg über Z kürzer:
D(X,Y)= ∞

| $D^Z$ | X | Y |
|---|---|---|
| X | 50 | 5 |

| $D^Z$ | X | Y |
|---|---|---|
| X | 50 | 5 |

| $D^Z$ | X | Y |
|---|---|---|
| X | 50 | 61 |

| $D^Z$ | X | Y |
|---|---|---|
| X | 50 | 61 |

| $D^Z$ | X | Y |
|---|---|---|
| X | 50 | ∞ |

# 8.4.2.2 Link-State-Routing

- Grundlegende Vorgehensweise
  - Knoten müssen am Anfang nur ihre direkten Nachbarn kennen
    - Entdecken neuer Nachbarn mit speziellen Dateneinheiten
      - z.B. HELLO
  - Bestimmen der Kosten zu den direkten Nachbarn
  - Link State Broadcast
    - Identität und Kosten zu den direkten Nachbarn werden an alle Knoten im Netz durch Fluten weitergeleitet (Broadcast)
    - Knoten können Topologie lernen durch die Link State Broadcasts der anderen Knoten
    - Ergebnis: Alle Knoten haben identisches Wissen über das Netz

# Link-State-Routing

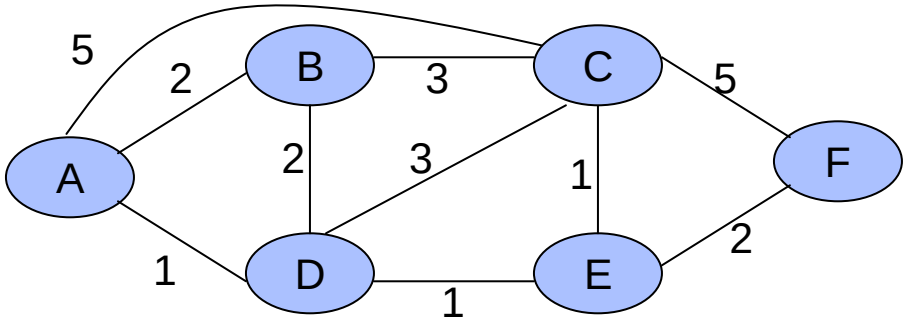
- Berechnung der kürzesten Pfade durch Link-State-Algorithmus
  - Jeder Knoten berechnet die kürzesten Pfade
  - Die berechneten Pfade sind aufgrund der identischen Information gleich
  - Nach Fluten und Berechnung der kürzesten Pfade in jedem Knoten ist das Netz schleifenfrei und in stabilen Zustand konvergiert
  - Dijkstra-Algorithmus zur Berechnung wird im Folgenden betrachtet
    - Berechnet den Pfad mit den geringsten Kosten von einem Knoten zu allen anderen Knoten im Netz

⇨ Link-State Routing wird in den Protokollen OSPF (*Open Shortest Path First*) und IS-IS eingesetzt

# Algorithmus

- Notation
  - $c(i, j)$: Kosten von Knoten $i$ zu Knoten $j$. Es wird angenommen, dass $c(i, j) = c(j, i)$. Falls $i$ und $j$ nicht direkt verbunden sind gilt: $c(i, j) = \infty$
  - $D(v)$: Kosten der Route von der Quelle zur Senke $v$, die momentan die geringsten Kosten besitzt
  - $p(v)$: Vorgänger von $v$ auf dem momentan kürzesten Pfad zu $v$
  - $N$: Menge der Knoten, deren kürzester Pfad von der Quelle bekannt ist
- Algorithmus
  - Initialisierungsphase
    - $N = \{\text{Quelle } A\}$, $D(v) = c(A, v)$ für alle direkten Nachbarn von $A$, $D(v) = \infty$ sonst
  - Schleife
    - Wird entsprechend der Anzahl von Knoten im Netz durchlaufen
      - Finde einen Knoten $w$ mit $w \notin N$ und $D(w)$ ist ein Minimum
      - Füge $w$ zu $N$ hinzu
      - Erneuere $D(v)$ für alle $v \notin N$ und $v$ ist direkter Nachbar von $w$
        - $D(v) = \min(D(v), D(w) + c(w, v))$

# Beispiel



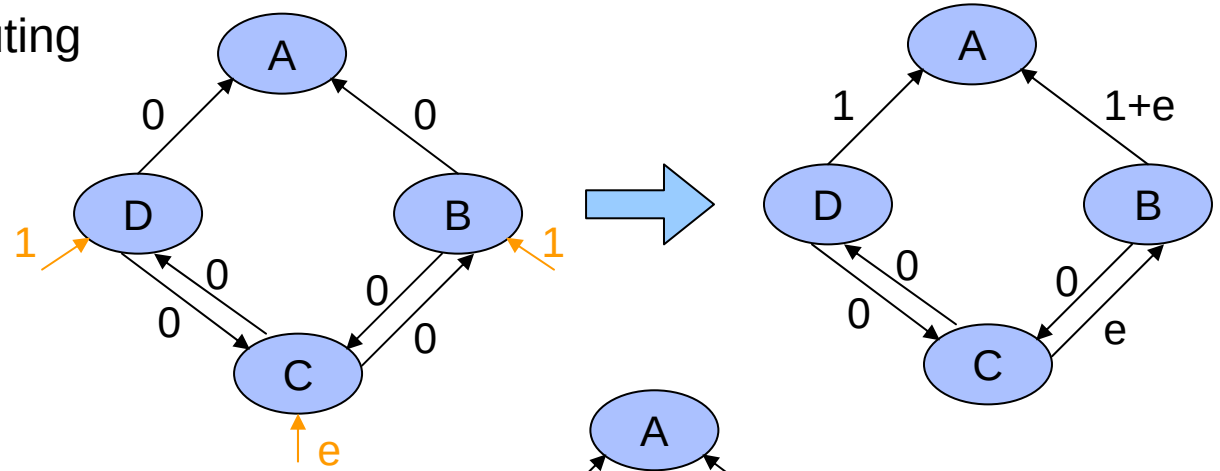

| Schritt | N | D(B), p(B) | D(C), p(C) | D(D), p(D) | D(E), p(E) | D(F), p(F) |
|---|---|---|---|---|---|---|
| 0 | A | 2, A | 5, A | 1, A | ∞ | ∞ |
| 1 | AD | 2, A | 4, D | | 2, D | ∞ |
| 2 | ADE | 2, A | 3, E | | | 4, E |
| 3 | ADEB | | 3, E | | | 4, E |
| 4 | ADEBC | | | | | 4, E |
| 5 | ADEBCF | | | | | |

# Beispiel

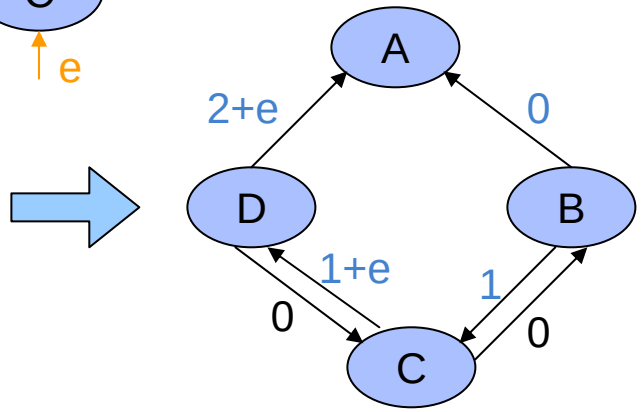
- Initialisierung
  - Die momentan bekannten kürzesten Pfade zu den Nachbarn von $A$ werden gesetzt entsprechend der jeweiligen Link-Kosten (2, 5, 1)
  - Die Pfade zu nicht direkt benachbarten Knoten werden auf $\infty$ gesetzt
- Erste Iteration
  - Knoten mit den geringsten Kosten wird zu $N$ hinzugenommen: $D$
  - $D(v)$ wird für alle Knoten erneuert
    - Kosten zu $C$ und $E$ senken sich
- Zweite Iteration
  - Knoten mit den geringsten Kosten wird zu $N = \{A, D\}$ hinzugenommen: $E$
    - $B$ wäre ebenfalls möglich gewesen
  - $D(v)$ wird für alle Knoten erneuert
    - Kosten zu $C$ senken sich und $F$ ist erstmals mit Kosten geringer als $\infty$ erreichbar
- ...
- Jeder Knoten kennt seinen Vorgänger und somit können die kürzesten Pfade konstruiert werden

# Oszillationen

- Beispielnetz
  - Linkkosten sind äquivalent zur Last auf dem Weiterleitungsabschnitt
  - Die Linkkosten sind hier nicht symmetrisch
  - Die Quellen B, C, und D senden 1, e bzw. 1 „Verkehrseinheiten“
    - Initiales Routing
    - D → A,
      B → A,
      C → B → A

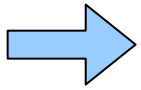


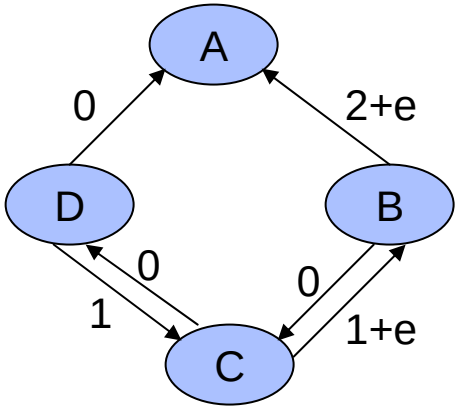

    - Nächste Runde
    - D → A,
      B → C → D → A,
      C → D → A,



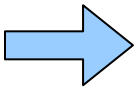
# Oszillationen (2)

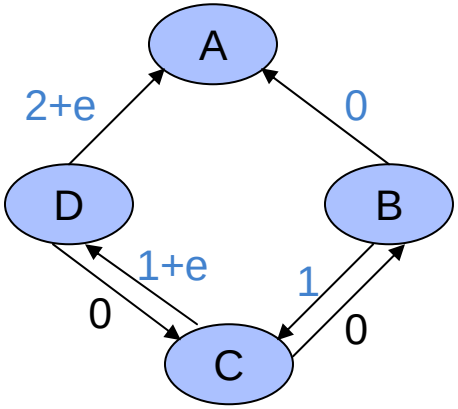
- Nächste Runde
  - Knoten B, C und D entdecken Pfad mit Kosten 0 zu A
- D → C → B → A,
  B → A,
  C → B → A






- ... Und die nächste Runde
- D → A,
  B → C → D → A,
  C → D → A






- Was tun?
  - Router sollten nicht alle zum gleichen Zeitpunkt den Algorithmus berechnen
  - „Self-Synchronization“ sollte möglichst vermieden werden
    - Einführung von Zufälligkeiten („Randomization“)

# Link-State vs. Distanz-Vektor

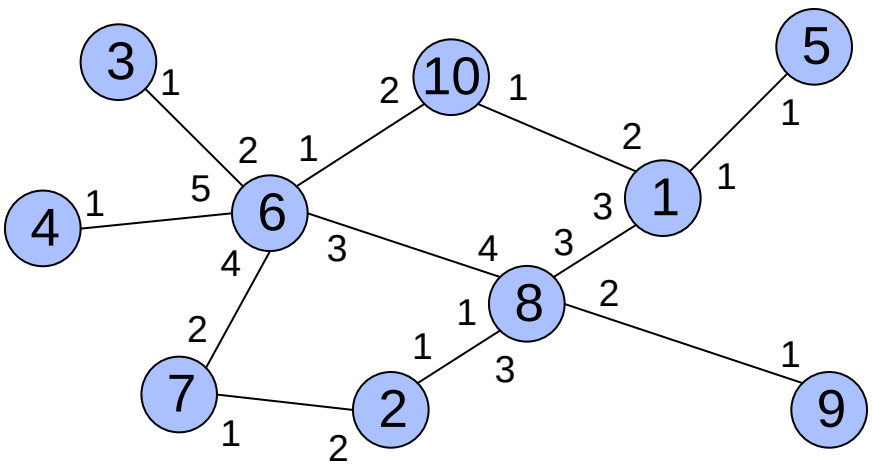
- Komplexität der Kontroll-Dateneinheiten
  - Jeder Knoten muss bei Link-State die Kosten aller Links kennen:
    Bei $n$ Knoten und $E$ Links sind $O(nE)$ Dateneinheiten erforderlich
  - Änderungen müssen bei Link-State an alle Knoten gesendet werden
  - Bei Distanz-Vektor werden Änderungen benachbarten Knoten weitergegeben
- Konvergenzgeschwindigkeit
  - Link-State hat eine Komplexität von $O(n^2)$ und benötigt $O(nE)$ Dateneinheiten
    - Schnelle Konvergenz, danach schleifenfrei
    - Oszillationen sind möglich
  - Distanz-Vektor-Algorithmen können langsam konvergieren und können Routing-Schleifen aufweisen
    - Das Count-to-Infinity-Problem kann ebenfalls entstehen
- Robustheit
  - Routenberechnungen sind bei Link-State separiert und stellen somit eine gewisse Robustheit bereit
  - Bei Distanz-Vektor-Algorithmen kann ein Knoten inkorrekte Pfade zu allen Zielen verbreiten
- Gewinner?
  - Link-State konvergiert schneller und ist robuster
  - Distanz-Vektor einfacher zu implementieren

# 8.4.3 Hierarchisches Routing

- Grundlage
  - Aufteilung großer Netze in „Autonome Systeme“ (AS) oder Regionen
    - Ansonsten Anzahl der Einträge in der Routing-Tabelle und Menge der ausgetauschten Routing-Information nicht skalierbar mit Netzgröße
  - Die Router haben in einem autonomen System i.d.R. nur Routing-Informationen über dieses autonome System
  - In jedem autonomen System gibt es zumindest ein ausgezeichnetes Zwischensystem, das als Schnittstelle zu anderen autonomen Systemen dient
- Vorteil
  - Skalierbarkeit
    - Größe der Routing-Tabellen ist abhängig von der Größe des autonomen Systems
    - Änderungen von Einträgen in den Routing-Tabellen werden nur innerhalb eines autonomen Systems weitergegeben
  - Autonomie
    - Internet = Netz von Netzen
    - Routing kann im eigenen Netz kontrolliert werden
      - Im administrativen System ein einheitliches Routingprotokoll
      - Routing-Protokolle der autonomen Systeme müssen nicht identisch sein

# Routing aus der Sicht eines Algorithmikers*

## Compact Routing




- Gelabelter Verbindungsgraph
  - Feste oder wählbare Adressen $O(\log n)$ Bits
  - Feste oder wählbare „*port numbers*“ $1..\deg(v)$
  - Klassische Routingtabelle: Ziellabel → Port
    - $\Omega(n)$ Bits/Knoten – was erreicht man mit $o(n)$ Bits?

[WaWa07]

Tradeoff Stretch ←→ Lokaler Speicher

Ein Bewertungsmaß: Stretch

- Um welchen Faktor sind Routen zu lang?

$$\max_{Quelle,Ziel}\left(\frac{\text{Länge Routing - Pfad}}{\text{Länge kürzester Pfad}}\right)$$

Aussagen für allgemeine Graphen

- Stretch < 3 nur mit $\Omega(n)$ Bits
- Stretch 3 mit $O(\sqrt{n}\log^3 n/\log\log n)$ Bits lokalem Speicher möglich

* Dank an Dr. Bastian Katz

# Compact Routing für spezielle Graphenklassen

- Bäume
  - Feste *port numbers*: Shortest-Path-Routing mit Adressenlänge und lokalem Speicher in

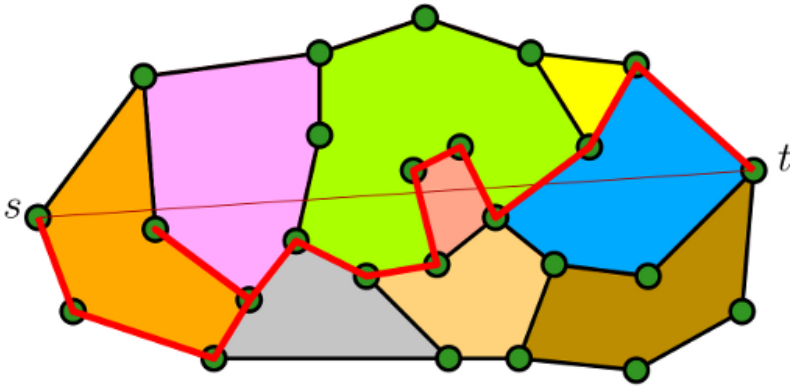
    $O(\log^2 n / \log\log n)$ möglich und optimal
  - Wählbare *port numbers*: je $O(\log n)$ Bits reichen
- Planare Graphen
  - Stretch $(1 + \varepsilon)$ möglich mit Adresslänge und lokalem Speicher in $O((\log^2 n) / \varepsilon)$
- Graphen mit geringer Doubling Dimension $\alpha$
  - Jede $k$-hop-Nachbarschaft lässt sich mit $\alpha$ $k/2$-Nachbarschaften überdecken
    - Routing mit Stretch $(1 + \varepsilon)$ möglich für Adresslänge $O(\alpha \log D)$ Bits und lokalem Speicher $O((1/\varepsilon)^\alpha \alpha \log^2 D)$

# Geometrisches Routing

- Geometrisches Greedy Routing
  - Knoten kennen ihre Positionen, leiten Dateneinheiten an den Nachbarn weiter, der dem Ziel am nächsten ist
  - Kann in lokalem Minimum steckenbleiben



- Facettenrouting
  - Dateneinheiten werden auf Facetten von planarem Teilgraphen geroutet (Planarer Graph verteilt berechenbar)
  - Kann lange Umwege nehmen

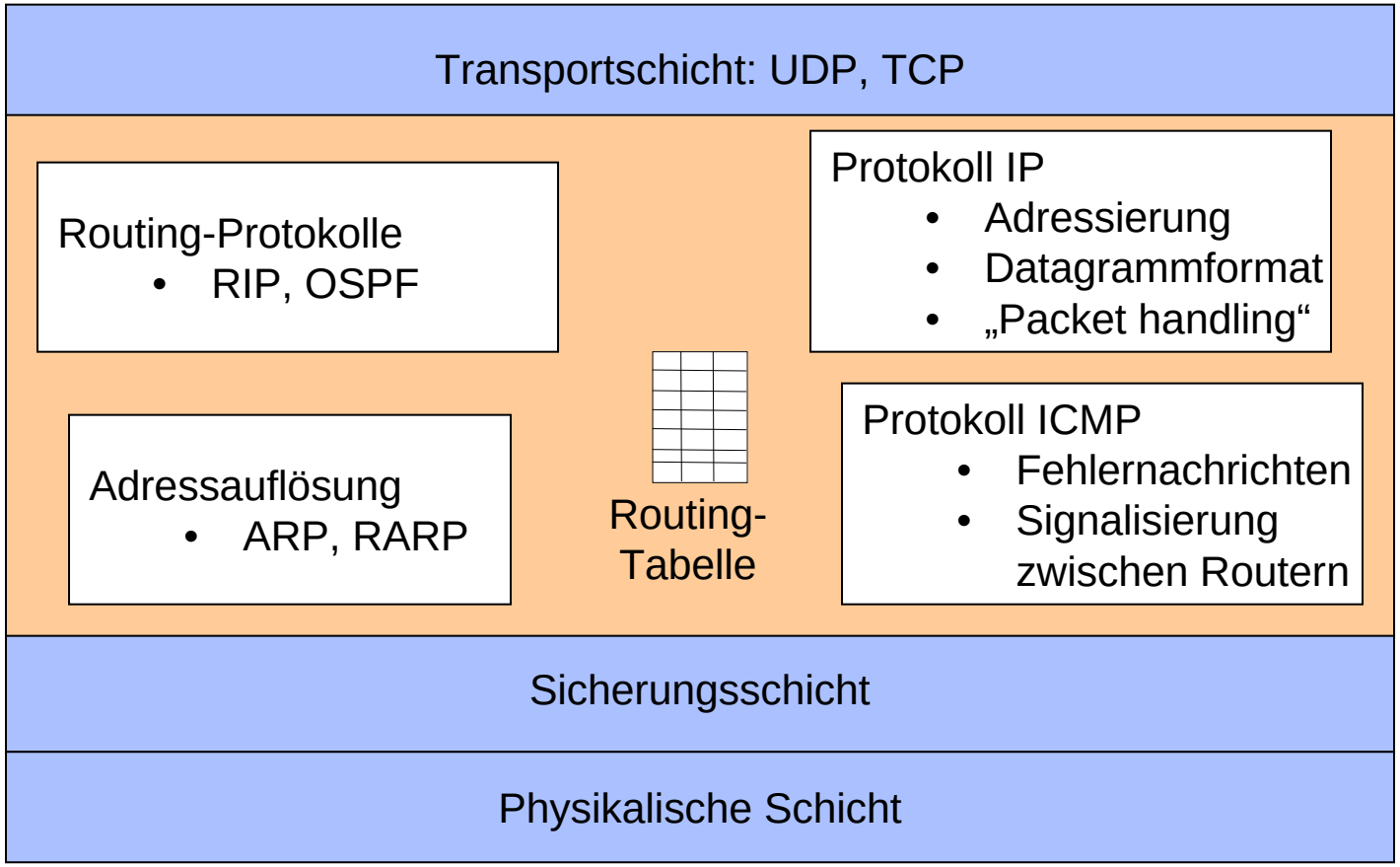



- Geschickte Kombination garantiert Zustellung ohne lokalen Speicher in $O(d^2)$ Schritten, wenn Quelle und Ziel Abstand $d$ haben

# 8.5 Vermittlung im Internet

- Problem
  - Wie werden Daten im Internet weitergeleitet?
- Verfahren
  - Routing-Tabelle liefert Information über den nächsten Hop für die Daten
  - Das Protokoll IP (*Internet Protocol*) führt das Weiterleiten der Daten aus
    - Datagramm-Protokoll
      - Verbindungslos
        - Wie sieht es also mit der Zuverlässigkeit aus?
      - Segmentiert und reassembliert
        - in allen Systemen?
    - Benutzt Internet-Adressierung
      - Unterschied zur MAC-Adressierung?
    - Benutzt weitere Protokolle wie
      - ICMP (Internet Control Message Protocol)
      - ARP (Address Resolution Protocol)
      - IGMP (Internet Group Management Protocol)

# Ein Blick in die Vermittlungsschicht des Internet



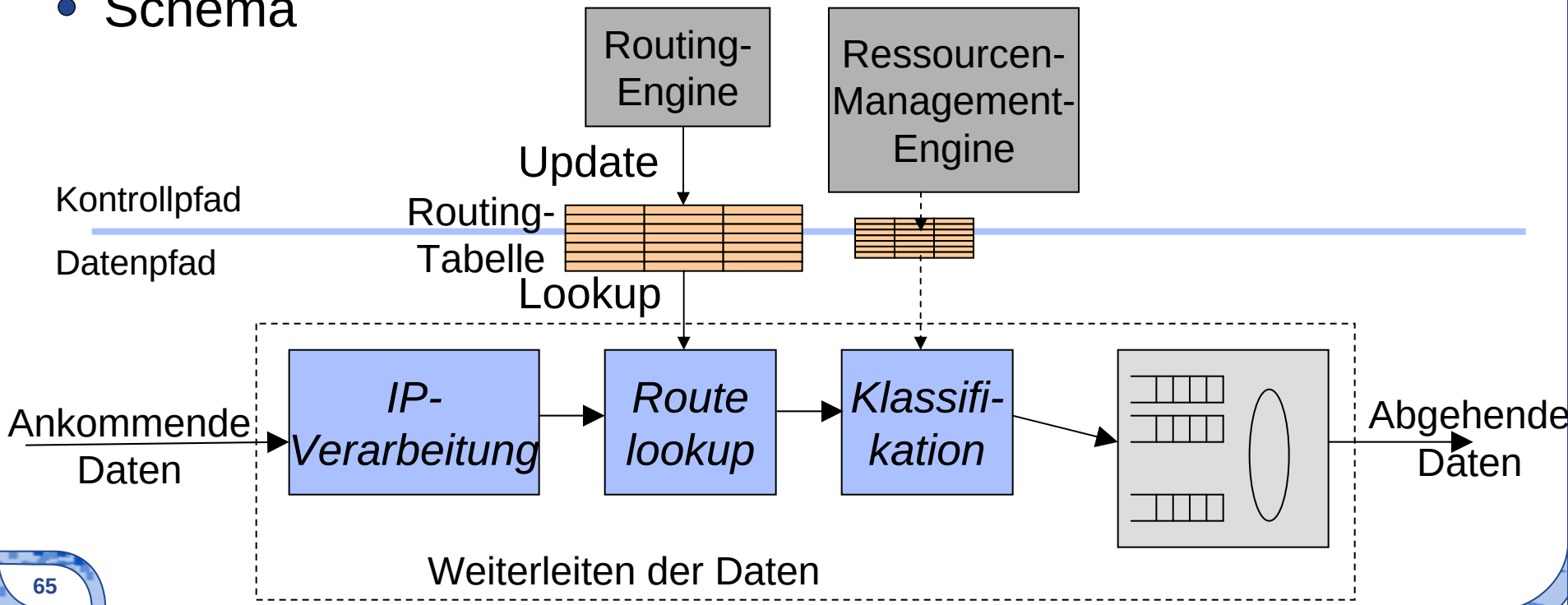
# Aufgaben der transportorientierten Internet-Protokolle

- *TCP* (Transmission Control Protocol):
  Stellt zuverlässigen Transportdienst bereit
- *UDP* (User Datagram Protocol):
  Stellt unzuverlässigen Transportdienst bereit
- *IP* (Internet Protocol):
  Wegewahl und unzuverlässige Übertragung von Datagrammen
- *ICMP* (Internet Control Message Protocol): Unterstützt den Austausch von Kontrollinformationen innerhalb der Vermittlungsschicht
- *IGMP* (Internet Group Management Protocol):
  Unterstützt die Verwaltung von Kommunikationsgruppen
- *ARP* (Address Resolution Protocol): Unterstützt die Zuordnung von IP-Adressen zu den entsprechenden Adressen der Sicherungsschicht
- *RARP* (Reverse Address Resolution Protocol):
  Stellt die Umkehrfunktion von ARP zur Verfügung
- BGP Border Gateway Protocol
- RIP Routing Information Protocol
- OSPF Open Shortest Path First

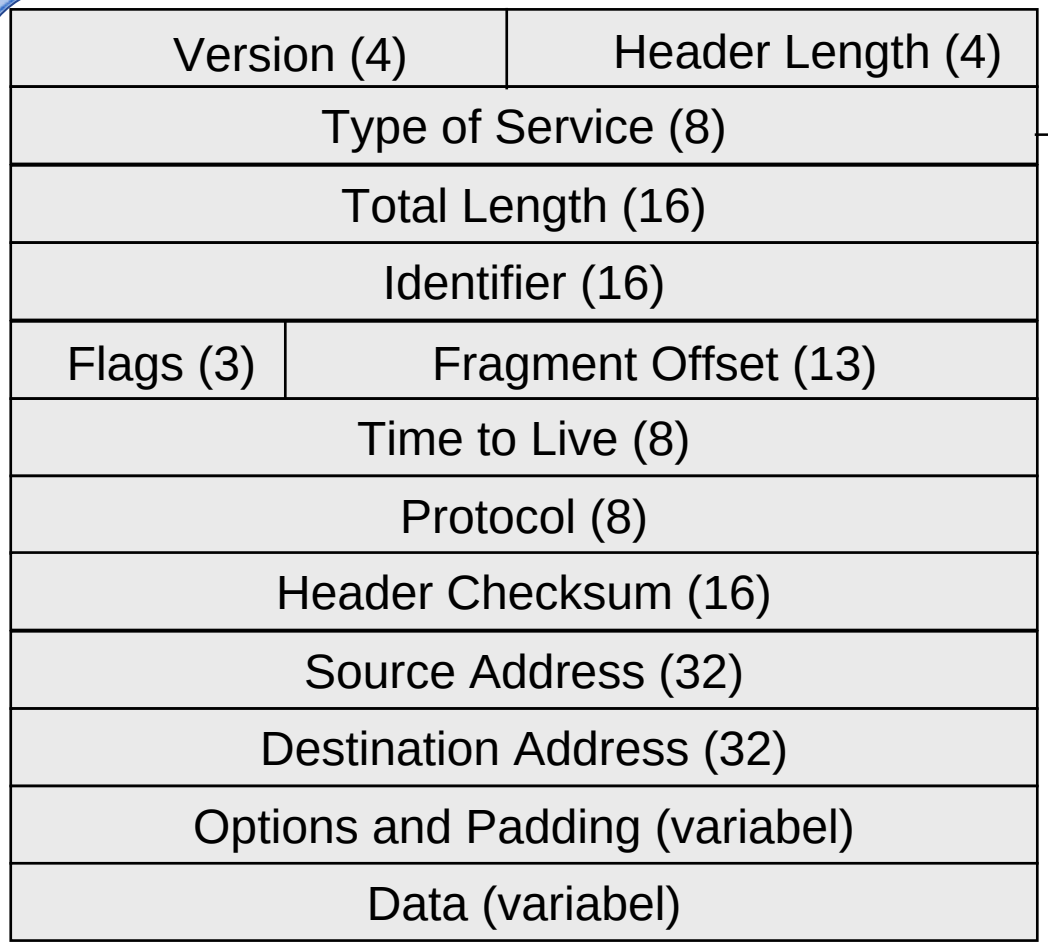
} Routing-Protokolle

# Das Protokoll IP

- IP (*Internet Protocol*):
  „Das“ Protokoll der Vermittlungsschicht im Internet
  - Verbindungslos und unzuverlässig
    - Kein IP-basierter Kontext in End- und Zwischensystemen
  - Verantwortlich für das Weiterleiten von Dateneinheiten im Internet
    - Dateneinheiten werden hier als IP-Datagramme bezeichnet
- Schema



# Format einer IP-Dateneinheit




*reserviert*

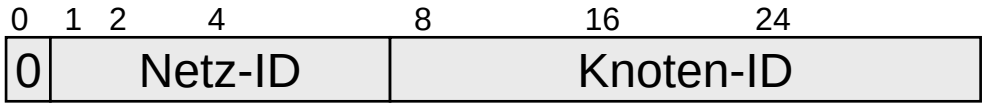
P P P D T R Rs Rs Rs

Precedence
111 Network Control
110 Internetwork Control
101 CEITIC/ECP
011 Flash
010 Immediate
001 Priority
000 Routine

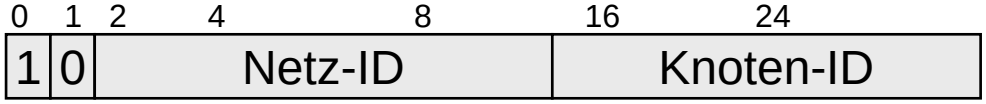
| | |
|---|---|
| Delay: | 0 normal |
| | 1 niedrig |
| Throughput: | 0 normal |
| | 1 hoch |
| Reliability: | 0 normal |
| | 1 hoch |

# IP-Adressierung

- Ursprünglich unterstützte IP fünf verschiedene Adressklassen:
  - Class A für Netze mit mehr als 65.536 Knoten



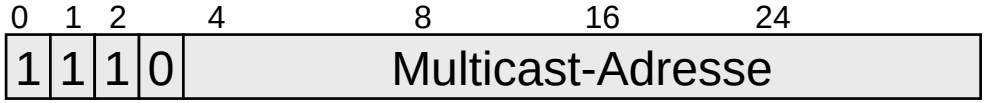

  - Class B für Netze zwischen 256 und 65.536 Knoten



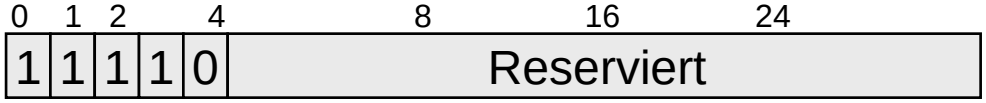

  - Class C für Netze mit weniger als 256 Knoten




  - Class D für Gruppen-kommunikation (Multicast)



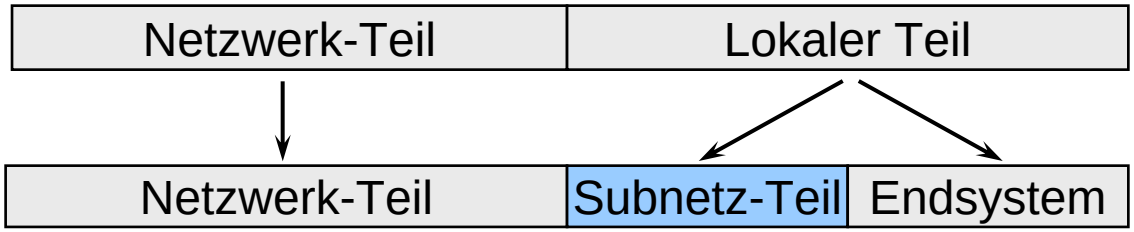

  - Class E, reserviert für zukünftige Anwendungen




- Diese Form der Adressierung wird heute nicht mehr direkt eingesetzt
  - ⇨ *Classless Interdomain Routing* (CIDR, 1993)

# IP-Subnetz-Adressen

- Weitere Strukturierung von IP-Adressen




- Subnetzmasken kennzeichnen den Bereich der IP-Adresse, der das Netzwerk und das Subnetzwerk beschreibt. Dieser Bereich wird dabei durch Einsen („1“) in der binären Form der Subnetzmaske festgestellt
- Beispiel

| | | | | | |
|---|---|---|---|---|---|
| IP-Adresse: | 129. | 13. | 3. | 64 | |
| Subnetzmaske: | 255. | 255. | 255. | 0 | = |
| | 1111 1111 | 1111 1111 | 1111 1111 | 0000 0000 | |
| Netzwerk: | 129. | 13. | | | (Class B) |
| Subnetz: | | | 3. | | |
| Endsystem: | | | | 64 | |

  - Überdeckt die Subnetzmaske nur den Netzwerk-Teil, dann gibt es keinen Subnetz-Teil (z.B. Subnetzmaske 255.255.0.0 bei Class B)
  - Achtung: Systeme, die an mehrere Netze angeschlossen sind (z.B. Router), haben mehrere, netzspezifische IP-Adressen

[RFC950]

# Zuteilung von IP-Adressen

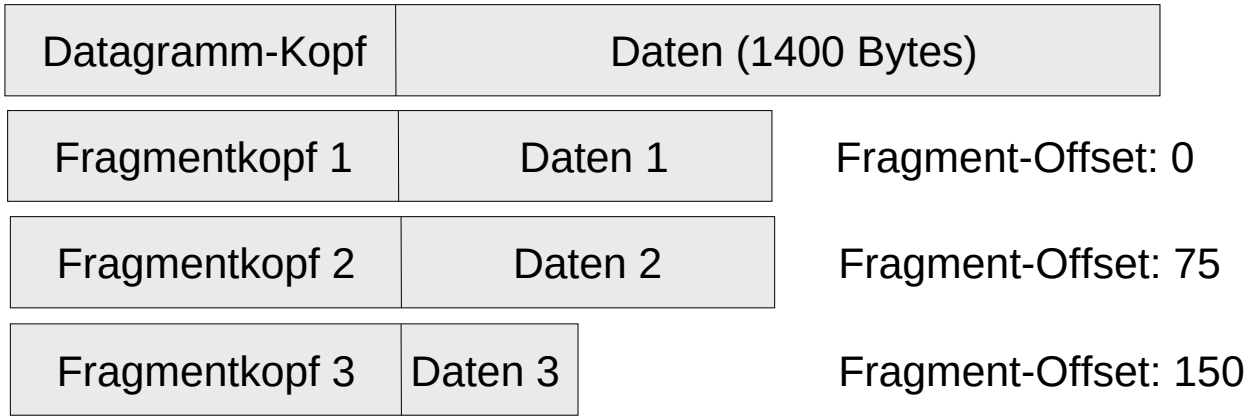
- Es lassen sich die beiden folgenden Varianten unterscheiden
  - Manuelle Konfiguration
  - Dynamische Konfiguration durch das Dynamic Host Configuration Protocol (DHCP)
    - DHCP Server und Client
      - Server liefert bei Anfrage eine IP-Adresse zurück an den Client


[RFC2131]

- Und wie erhält ein Provider seinen Block von Adressen?
  - Verwaltung der IP-Adressen unterliegt der „Internet Assigned Numbers Authority“ (IANA) unterhalb der „Internet Corporation for Assigned Names and Numbers“ (ICANN)
    - Delegation an regionale Registrierungen („Regional Internet Registries“, RIRs)
      - AfriNIC (African Network Information Centre): Afrika
      - APNIC (Asia Pacific Network Information Centre): Asien/Pazifik
      - ARIN (American Registry for Internet Numbers): Nordamerika
      - LACNIC (Regional Latin-American and Caribbean IP Address Registry) – Lateinamerika und einige karibische Inseln
      - RIPE NCC (Réseaux IP Européens) - Europa, Mittlerer Osten und Zentralasien

# Segmentieren und Reassemblieren

- Ziel
  - Anpassung an verschieden lange maximale Längen der Dateneinheiten unterliegender Netze
- Vorgehensweise
  - Flag-Felder des IP-Kopfes werden verwendet:
    - Bit 0: reserviert, muss 0 sein
    - Bit 1: 0 = darf fragmentiert werden<br>1 = darf nicht fragmentiert werden
    - Bit 2: 0 = letztes Fragment<br>1 = es folgen weitere Fragmente
    - Fragment-Offset: Stelle, an der empfangenes Fragment in ursprüngl. Dateneinheit eingesetzt werden muss (Basiseinheit: 8 Bytes)
  - Beispiel

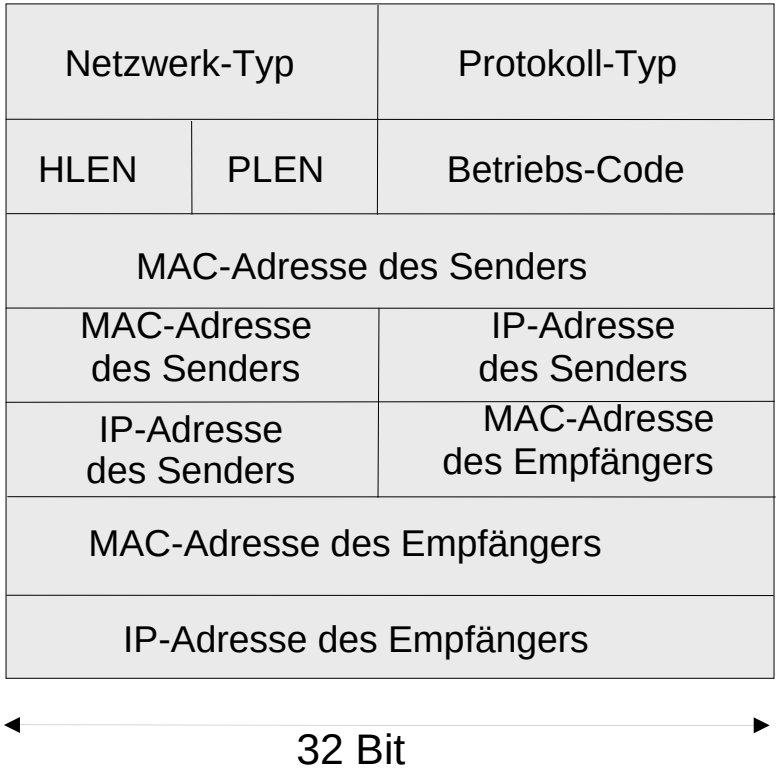


# Empfang einer IP-Dateneinheit

- Folgende Überprüfungen werden durchgeführt
  - Korrekte Länge des Kopfes?
  - IP-Versionsnummer?
  - Korrekte Datagrammlänge?
  - Prüfsumme?
  - Lebenszeit?
  - Protokoll-Identifikation?
  - Adressklasse (Quell- und Zieladresse)?
- Was muss beim erneuten Aussenden auf einem Übertragungsabschnitt berechnet werden?
- Falls ein Fehler erkannt wird:
  - Benachrichtigung von ICMP (*Internet Control Message Protocol*)
    - D.h. ICMP wird hier als Funktion aus IP aufgerufen
    - Reagiert möglicherweise mit dem Aussenden einer ICMP-Dateneinheit

# Einschub: Das Protokoll ARP

- Ziel-System im gleichen IP-Subnetz: Zustellung über Schicht 2
- ARP: Address Resolution Protocol
  - Ist für die Abbildung zwischen IP-Adressen und MAC-Adressen verantwortlich
- Kopf der ARP-Dateneinheit




Netzwerk-Typ: 1 = Ethernet; 6 = IEEE 802.2

Protokoll-Typ: 2048 = IP

HLEN: 2 = 16-Bit MAC-Adresse; 6 = 48-Bit MAC-Adresse

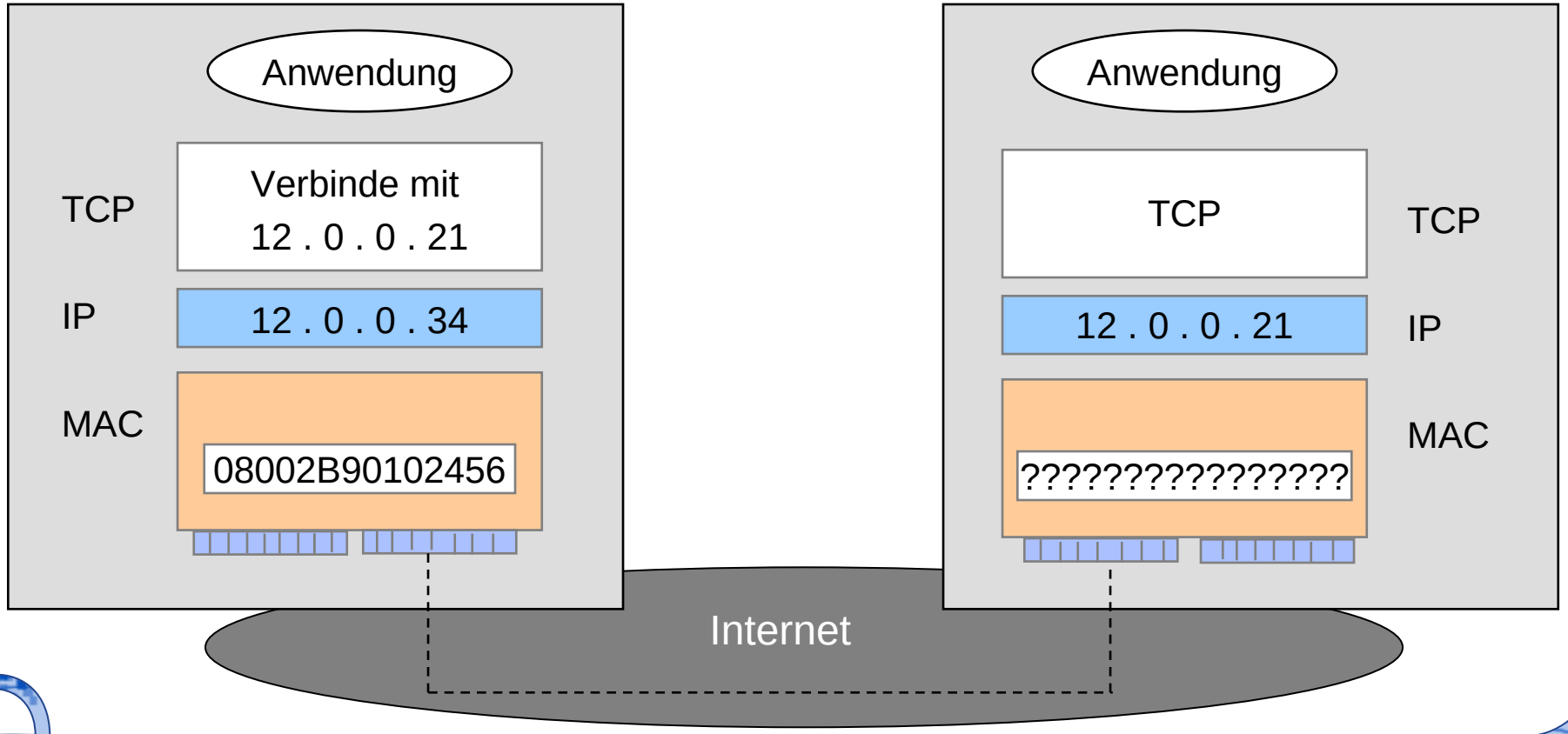
PLEN: 4 = 32-Bit IP-Adresse

Betriebs-Code: 1 = Request; 2 = Reply

HLEN: Header Address Length
PLEN: Protocol Address Length

# Zuordnung von IP- und MAC-Adressen

- Wenn (Ziel-IP-Adresse AND Subnetzmaske) gleich (Eigene IP-Adresse AND Subnetzmaske)
  ⇨ Zielsystem im gleichen IP-Subnetz!
  - Problem: Welche MAC-Adresse hat das nächste System auf dem Weg zum Ziel?
- Schema



# Adressauflösung mit ARP

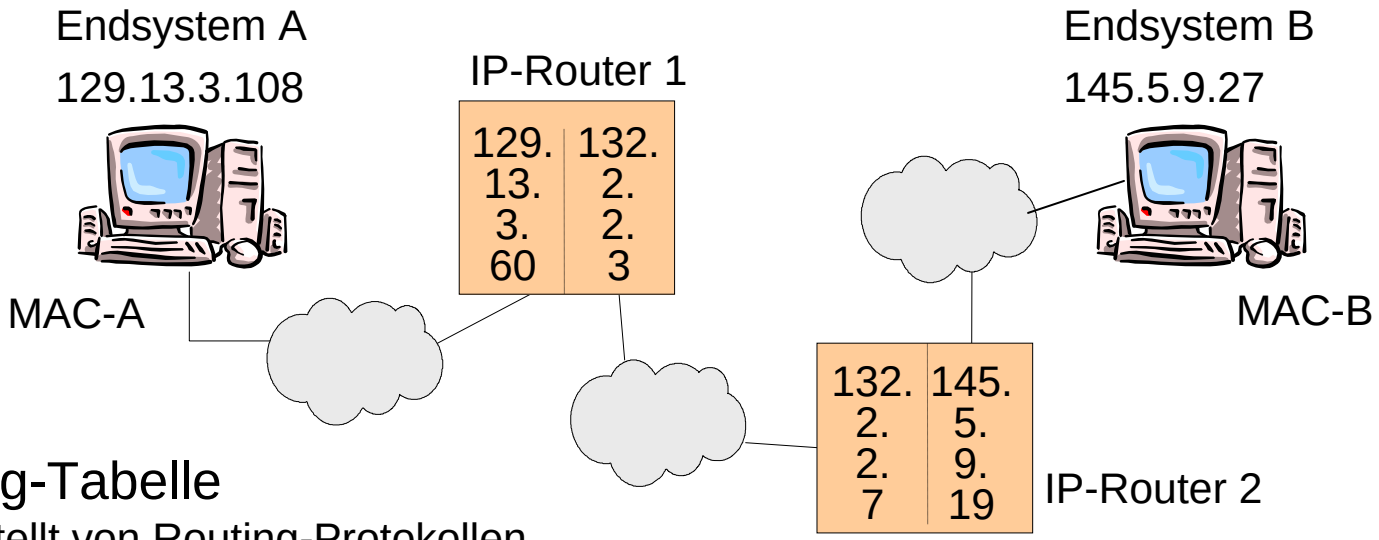
- Dynamisches Lernen von Adresszuordnungen
  - Kleine Tabellen (ARP-Cache bzw. ARP-Table)
    - Maximale Lebensdauer der Einträge (typischerweise 20 Minuten)
  - Hohe Flexibilität
- Nutzt Broadcasting-Fähigkeit der lokalen Netze aus
- Falls kein Eintrag im lokalen ARP-Cache vorhanden ist:
  - Broadcast eines ARP-Request (enthält Ziel-IP-Adresse)
  - Jedes Endsystem liest ARP-Request und überprüft IP-Adresse
    - Falls eigene IP-Adresse, dann ARP-Reply und Eintragen der Information im Hinblick auf das eventuelle Versenden einer Antwort
    - Suchende Instanz trägt Information in ihren ARP-Cache ein
    - Optional: Andere Endsysteme merken sich ebenfalls Adresszuordnung der suchenden Instanz aus der Anfrage

# Wegewahl in IP

- Für ein Endsystem
  - Ist Rechner mit Zieladresse direkt mit dem Endsystem verbunden (Punkt- zu-Punkt oder gleiches lokales Netz), wird das IP-Paket direkt zu diesem Empfänger geschickt
  - Ansonsten wird das Paket an einen voreingestellten Vermittlungsrechner (Default-Router) weitergegeben

- Grundlage für die Wegewahl: Routing-Tabelle mit
  - Angabe der Zieladresse (Endsystem- oder Netzadresse)
  - Angabe des „Next-Hop“ Routers
  - Flags, welche die beiden oberen Angaben genauer klassifizieren
  - Angabe der Netzschnittstelle, auf die ein für die Zieladresse bestimmtes Paket ausgegeben werden soll

# Weiterleiten im IP-Router



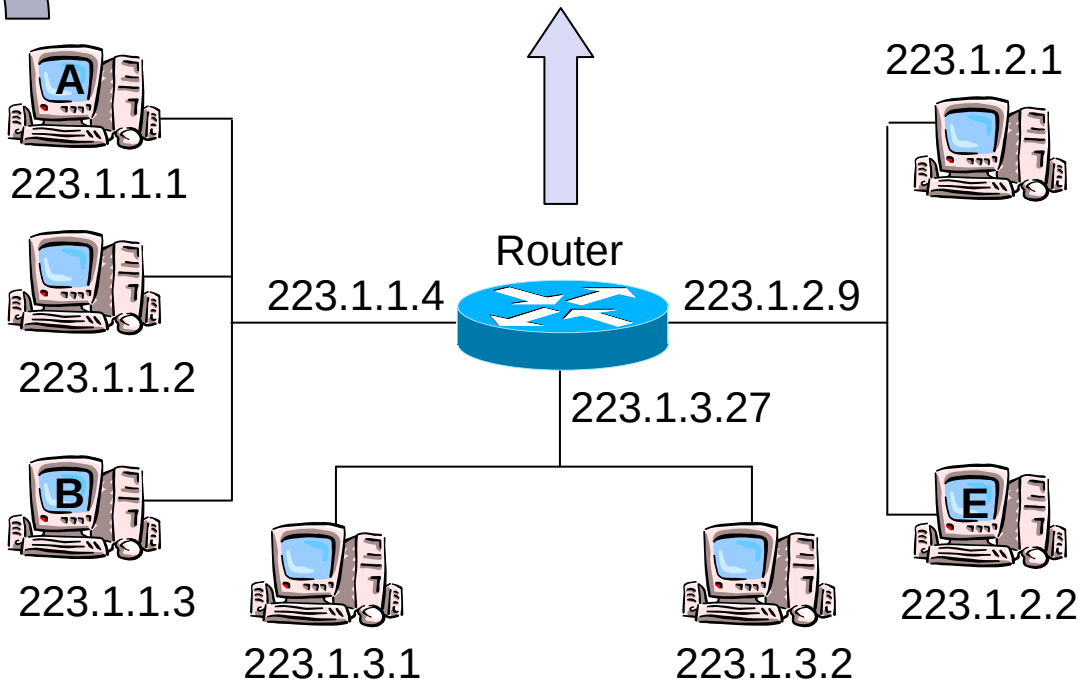

- Routing-Tabelle
  - Erstellt von Routing-Protokollen
  - IP-Adresse des nächsten Systems und Kennung des Ausgangs
- Adressumsetzungstabelle
  - Erstellt von ARP
  - MAC-Adresse des nächsten Systems für IP-Adresse des Endsystems
- Beispiel
  - Ziel: Endsystem B; Quelle: Endsystem A
  - Dateneinheit auf dem Weg von Router 1 und Router 2:
    - MAC-Adressen: MAC-Adresse IP-Router 2 (Ziel) und MAC-Adresse IP-Router 1 (Quelle)
    - IP-Adressen: Endsystem B (Ziel), Endsystem A (Quelle)
      - Beachte: IP-Adresse von IP-Router 2 wird in der Dateneinheit nicht transportiert

# Routing-Tabellen-Lookup

- Aufgabe
  - Identifikation des nächsten Systems auf dem Weg zum Ziel
- Beispielszenario

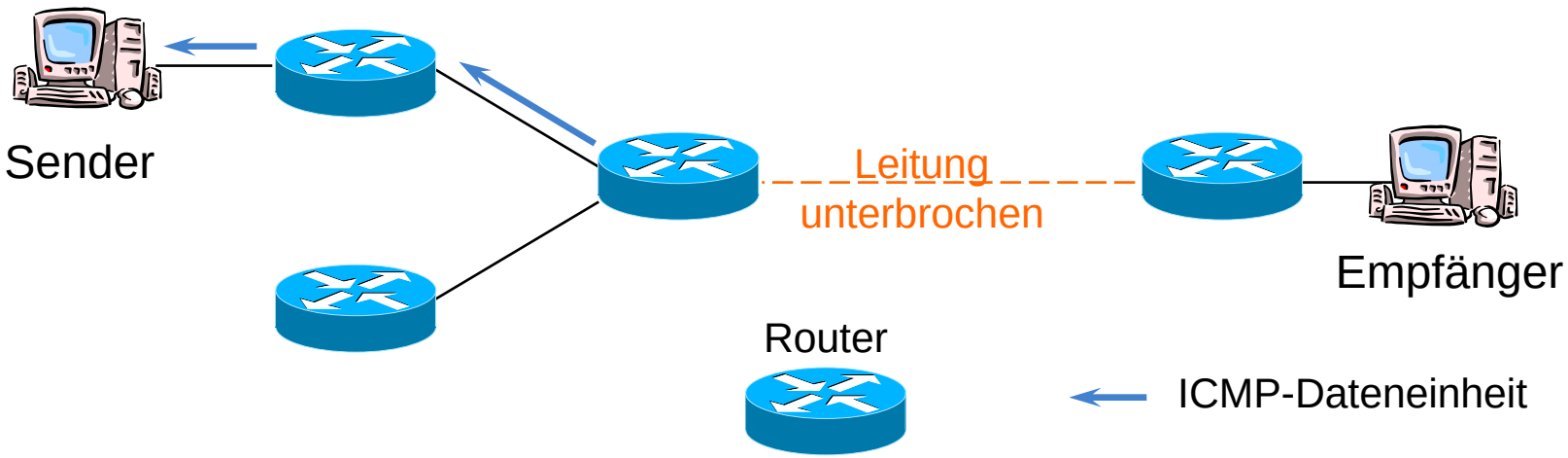
| Ziel-Netz | Nächster Router | Hops | Interface |
|---|---|---|---|
| 223.1.1.0/24 | - | 1 | 223.1.1.4 |
| 223.1.2.0/24 | - | 1 | 223.1.2.9 |
| 223.1.3.0/24 | - | 1 | 223.1.3.27 |

| Ziel-Netz | Nächster Router | Hops |
|---|---|---|
| 223.1.1.0/24 | | 1 |
| 223.1.2.0/24 | 223.1.1.4 | 2 |
| 223.1.3.0/24 | 223.1.1.4 | 2 |



# ICMP – Internet Control Message Protocol

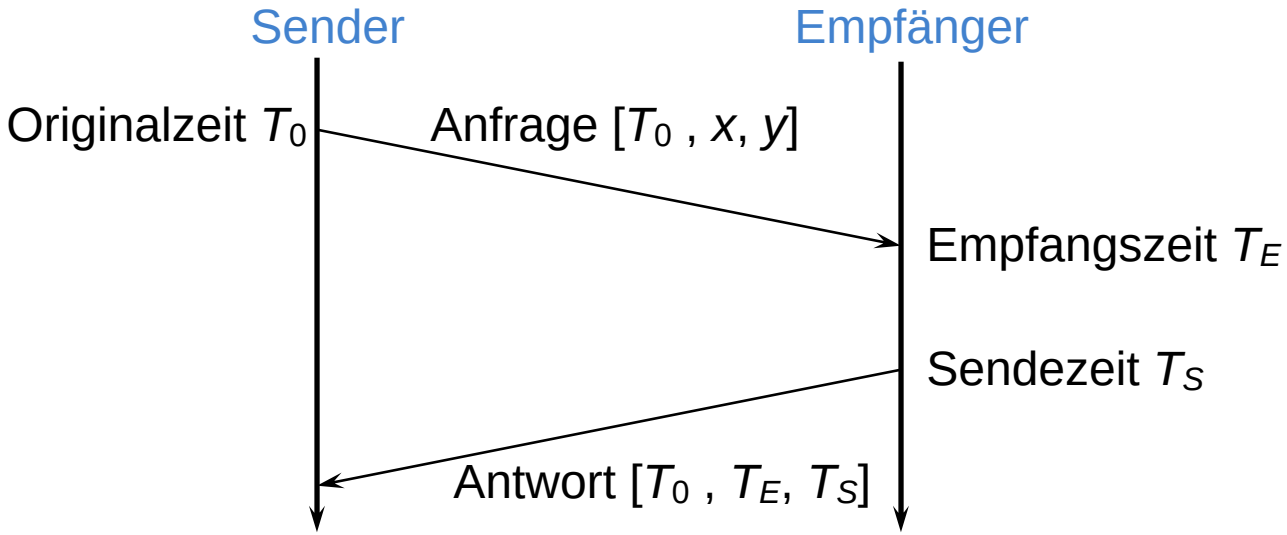
- Einzelne Verluste von Dateneinheiten werden im Normalfall von IP nicht gemeldet (unzuverlässiger Datagrammdienst)
- Schwerwiegende Probleme (z.B. Unterbrechung einer Leitung) werden zur Vermeidung von Folgefehlern mittels ICMP den Kommunikationspartnern mitgeteilt




- ICMP unterstützt den Austausch von Fehlernachrichten, Statusanfragen und Zustandsinformation

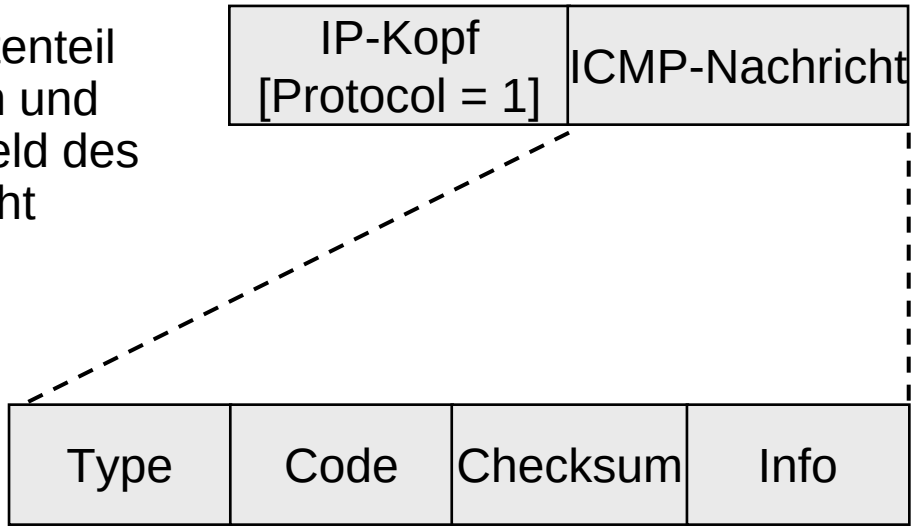
# ICMP-Statusanfragen

- Echo und Echoantwort (echo and echo reply)
  - Dient der Überprüfung der Aktivität von Kommunikationssystemen
  - Der Empfänger einer Echo-Anfrage sendet in der Echo-Antwort die erhaltenen Daten an den Kommunikationspartner zurück
- Zeitstempel und Zeitstempelantwort (timestamp and timestamp reply)
  - Dient der Bestimmung von Umlaufzeiten
  - Die Nachrichten umfassen mehrere Felder zur Aufnahme von Zeitstempeln, anhand derer die Bearbeitungszeiten beim Empfänger und die Verzögerung im Netz bestimmt werden können
- Ablauf



# Format von ICMP-Dateneinheiten

- Übertragung der ICMP-Nachrichten
  - ICMP-Nachrichten werden im Datenteil von IP-Dateneinheiten übertragen und durch den Wert „1“ im Protocol-Feld des IP-Paket-Kopfes kenntlich gemacht




- Format der ICMP-Nachrichten
  - Type
    - Typ der Nachricht (z.B. Type = 3 entspricht „Zieladresse nicht erreichbar“)
  - Code
    - Genaue Beschreibung der Nachricht (z.B. „Netzwerk nicht erreichbar“)
  - Checksum
    - Prüfsumme über die gesamte ICMP-Nachricht
  - Der Inhalt des Info-Teils ist abhängig vom Typ der ICMP-Nachricht
    - z.B. Felder für Zeitstempel bei Nachricht „Zeitstempel und Zeitstempelantwort“

# Beispiel: Routenverfolgung mittels Traceroute

- Keine Garantie, dass aufeinanderfolgende IP-Datagramme von derselben Quelle zum selben Ziel gleiche Route nutzen → Wie Route ermitteln?
- Anwendung Traceroute von Van Jacobsen
  - Nutzt ICMP und TTL-Feld im IP-Kopf
- Empfängt Router Datagramm mit TTL von 0 oder 1 → keine Weiterleitung
  - Router verwirft Dateneinheit und schickt ICMP Nachricht „Time Exceeded“ an ursprüngliches Endsystem zurück
- Funktionsweise von Traceroute
  - IP-Datagramm (unter UNIX mittels UDP) mit TTL = 1 an Zielsystem
    - Erster Router dekrementiert TTL und antwortet mit ICMP Time Exceeded Nachricht
      - Erster Router identifiziert
  - IP-Datagramm mit TTL = 2 an Zielsystem
    - Zweiten Router identifizieren
  - u.s.w.
  - Beim Eintreffen des IP-Datagramms beim tatsächlichen Ziel-Endsystem wird Datagramm nicht verworfen
    - Ungewöhnliche UDP-Portnummer nutzen (> 30.000)
      - Zielsystem antwortet dann mit ICMP „Port Unreachable“ Fehlernachricht
- Für jeden TTL-Wert werden drei Datagramme gesendet
  - Ausgabe Sternchen, falls nicht innerhalb von 5 Sek. Antwort
- Ermittlung der Round-Trip-Time bei jeder zurückkehrenden ICMP-Nachricht

# Beispiel: Routenverfolgung mittels Traceroute (2)

```
$ traceroute ietf.org
traceroute to ietf.org (64.170.98.32), 30 hops max, 40 byte packets
 1  i72poolgate.tm.uni-karlsruhe.de (141.3.70.254)  0.233 ms  0.198 ms  0.166 ms
 2  172.16.4.1 (172.16.4.1)  31.639 ms  31.773 ms  31.995 ms
 3  * * *
 4  172.21.3.9 (172.21.3.9)  30.363 ms  30.440 ms  30.570 ms
 5  Karlsruhe1.belwue.de (129.143.166.129)  30.647 ms  30.730 ms  31.858 ms
 6  Stuttgart1.belwue.de (129.143.1.153)  32.084 ms  31.341 ms  31.400 ms
 7  Stuttgart2.belwue.de (129.143.1.25)  31.773 ms  4.176 ms  2.607 ms
 8  Frankfurt1.belwue.de (129.143.1.130)  8.454 ms  8.447 ms *
 9  ffm-b3-link.telia.net (213.248.84.237)  14.762 ms  14.660 ms  14.639 ms
10  ffm-bb1-link.telia.net (80.91.249.138)  8.364 ms ffm-bb2-link.telia.net (80.91.254.42)
     8.498 ms ffm-bb1-link.telia.net (80.91.249.138)  8.484 ms
11  prs-bb2-link.telia.net (80.91.249.47)  18.436 ms  18.312 ms prs-bb1-link.telia.net
     (80.91.248.69)  18.406 ms
12  nyk-bb2-link.telia.net (80.91.251.100)  97.946 ms  97.677 ms  98.008 ms
13  nyk-b2-link.telia.net (80.91.250.1)  93.273 ms nyk-b2-link.telia.net (80.91.250.5)
     90.499 ms nyk-b2-link.telia.net (80.91.250.1)  93.475 ms
14  ex2-tg4-0.eqnwnj.sbcglobal.net (151.164.249.73)  91.260 ms  91.646 ms  91.376 ms
15  151.164.95.82 (151.164.95.82)  167.339 ms  167.326 ms  167.706 ms
16  AMS-1152322.cust-rtr.swbell.net (75.61.192.10)  169.082 ms  168.639 ms  169.984 ms
[...]
```

Unterschiedliche Routen

# IPv6

- Probleme mit der bisherigen Version von IP führten zur Weiterentwicklung zu IPv6 (IP Version 6):
  - Unterstützung von Milliarden von Endsystemen (16 Byte Adressen)
  - Reduzierung des Umfangs der Routing-Tabellen
  - Vereinfachung des Protokolls zur Effizienzsteigerung
  - Höhere Sicherheit
  - Unterschiedliche Dienstarten, z.B. Echtzeitunterstützung
  - Bessere Multicastunterstützung
  - Integrierte Unterstützung mobiler Teilnehmer
  - Koexistenz der alten und neuen Version
- Doch:
  - Einführung von IPv6 nur schleppend
  - Grund: Zusätze zum alten IP erfüllen bereits die mit IPv6 gesetzten Erwartungen, so dass eigentlich kein direkter Handlungsbedarf besteht!

# Übungen

8.1 Nennen Sie verschiedene Vermittlungstechniken und diskutieren Sie deren Eigenschaften, sowie deren Vor- und Nachteile.

8.2 Erläutern Sie den Unterschied von virtuellen Verbindungen zum Versand von Datagrammen.

8.3 Welche Techniken werden zur Netzkopplung auf den Schichten 1, 2 und 3 eingesetzt und wie funktionieren diese?

8.4 Was sind die Vor- und Nachteile von Brücken gegenüber Repeatern?

8.5 In welche Kategorien lassen sich Routing-Verfahren unterteilen?

8.6 Was ist die Aufgabe eines Routing-Protokolls und wo wird es im Schichtenmodell angesiedelt?

8.7 Wie funktionieren Distanz-Vektor-Routing-Protokolle?

8.8 Worin unterscheidet sich die Verbreitungsgeschwindigkeit von „guten“ bzw. „schlechten“ Änderungen der Linkkosten in Link-State-Protokollen?

8.9 Was versteht man unter dem Count-to-Infinity-Problem und durch welche Technik wird ihm begegnet?

8.10 Wie funktionieren Link-State-Routing-Protokolle?

8.11 Welche Aufgaben hat das Protokoll IP und durch welche Eigenschaften zeichnet es sich besonders aus?

8.12 Geben Sie für die IP-Adresse 129.13.6.34 und die Subnetzmaske 255.255.128.0 an, welche IP-Adressen lokal erreichbar sind.

8.13 Welches Protokoll kommt zum Einsatz, falls eine Zieladresse im eigenen lokalen Netz erreichbar ist, um die Dateneinheit auf Schicht 2 korrekt zustellen zu können und wie funktioniert dieses Protokoll?

8.14 Beschreiben Sie detailliert, wie eine Dateneinheit auf dem Weg zu seiner Zieladresse seinen Weg durch das Internet findet und woher die jeweils benötigten Informationen stammen.

8.15 Wozu dient das Protokoll ICMP und in welchem „Verhältnis“ steht es zu IP?

# Literatur


[Hals05] F. Halsall; Computer Networking and the Internet, 2005, 5th Edition

[KuRo07] J. Kurose, K. Ross; Computer Networking; Addison Wesley 2007, 4th Edition

[RFC950] J. Mogul, J. Postel; Internet Standard Subnetting Procedure (RFC 950); IETF, August 1985

[RFC2131] R. Droms; Dynamic Host Configuration Protocol (RFC 2131); IETF, März 1997

[Stal07] W. Stallings; Data & Computer Communications, Prentice Hall, 2007, 8th Edition

[Wart03] F. Warthman; Delay-Tolerant Networks (DTNs); März 2003

- www.ipnsig.org/reports/DTN_Tutorial11.pdf

[WaWa07] D. Wagner, R. Wattenhofer (Eds.); Algorithms for Sensor and Ad Hoc Networks, Springer, 2007

# Kommunikation und Datenhaltung

## 9. Die Transportschicht




**Prof. Dr. Martina Zitterbart**
Dipl.-Inform. Martin Röhricht
[zit | roehricht]@tm.uka.de



# Kapitelübersicht

1. Einführung
2. Physikalische Grundlagen
3. Protokollmechanismen
4. Geschichtete Architekturen
5. Sicherungsschicht: HDLC
6. Beschreibungsmethoden
7. Sicherungsschicht: Lokale Netze
8. Netzkopplung und Vermittlung
9. Die Transportschicht
10. Anwendungssysteme
11. Middleware

9.1 Einführung
9.2 User Datagram Protocol (UDP)
9.3 Transmission Control Protocol (TCP)

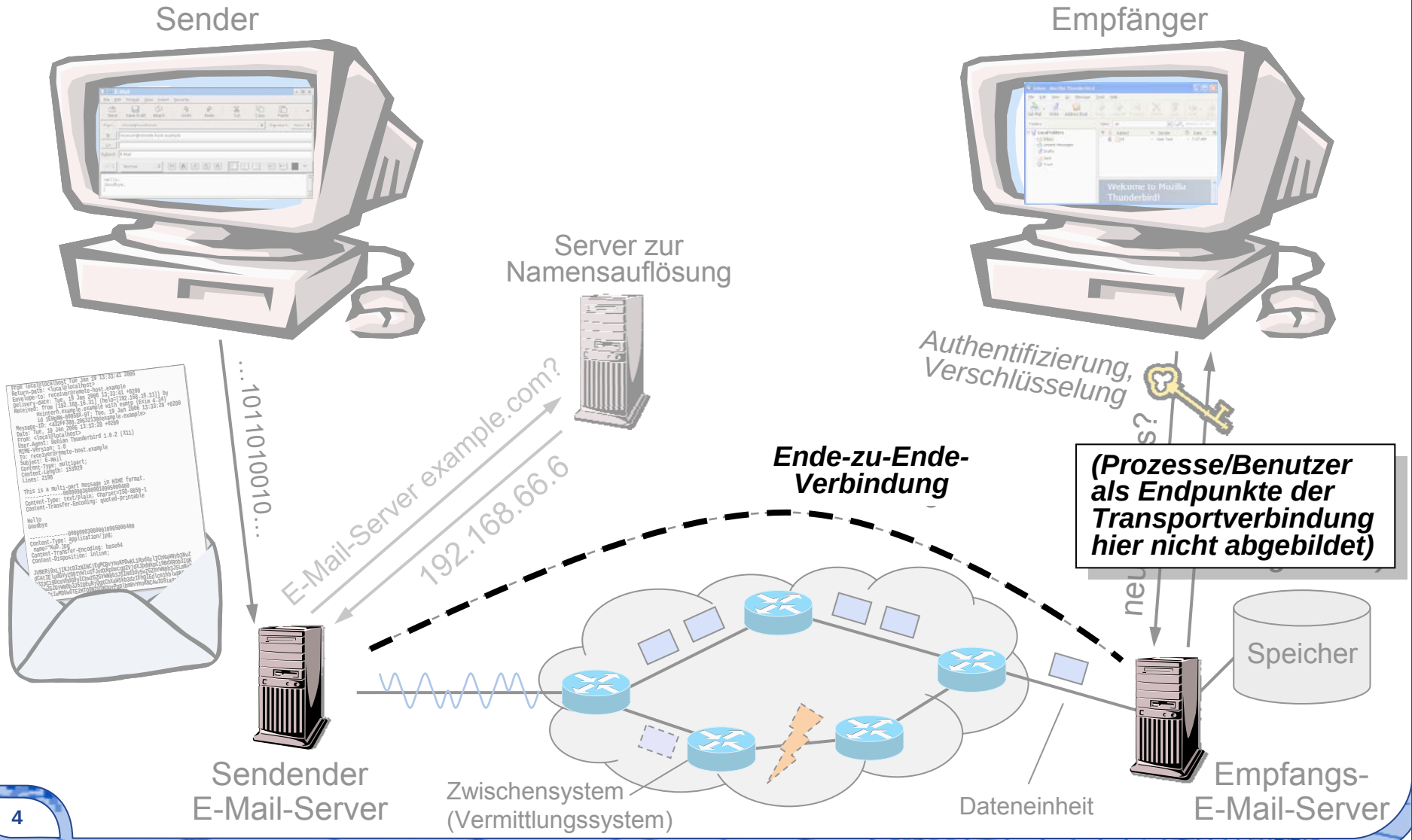
# 9.1 Einführung

- Transportschicht
  - Kommunikation zwischen Prozessen, die potenziell auf unterschiedlichen Rechnern im Netz lokalisiert sind
  - Kann sowohl verbindungslos als auch verbindungsorientiert sein
  - Falls zuverlässiger Dienst gefordert
    - Explizite Phase zum Verbindungsaufbau vor dem Datentransfer erforderlich
    - Explizite Phase zum Verbindungsabbau nach dem Datentransfer erforderlich
- Heterogene Rechner
  - Beim Verbindungsaufbau müssen vorhandene Ressourcen auf Transportebene kommuniziert werden
    - Beispiele für solche Ressourcen?
- Sender kennt nicht alle Teilstrecken einer Ende-zu-Ende-Kommunikationsbeziehung
  - Im Netz können Stausituationen auftreten und müssen behandelt werden

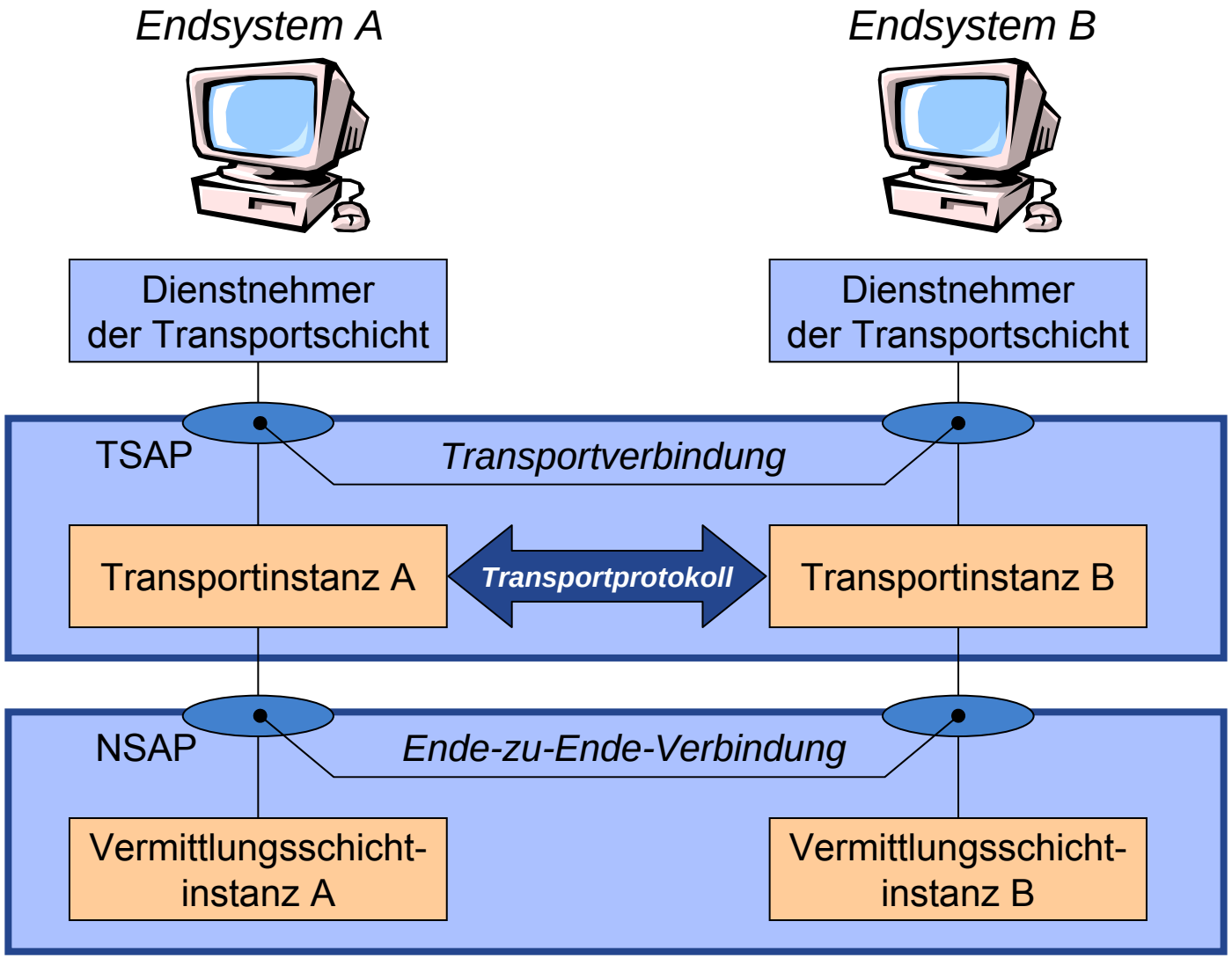
# Aufgaben und Ziel

- Aufgaben
  - Benutzer-zu-Benutzer-Verbindung
    - Statt nur Ende-zu-Ende-Verbindung
  - Isolierung der höheren Schichten von der Technologie, dem Design und den Unvollständigkeiten der verwendeten Netze
  - Teilnehmeradressierung: globaler Adressraum für Teilnehmer
  - Transparente Übertragung der Benutzerdaten
  - Wahlmöglichkeit der Dienstgüte
- Ziel
  - „transport layer […] provides applications with a network independent data interchange service“ [Hals05]

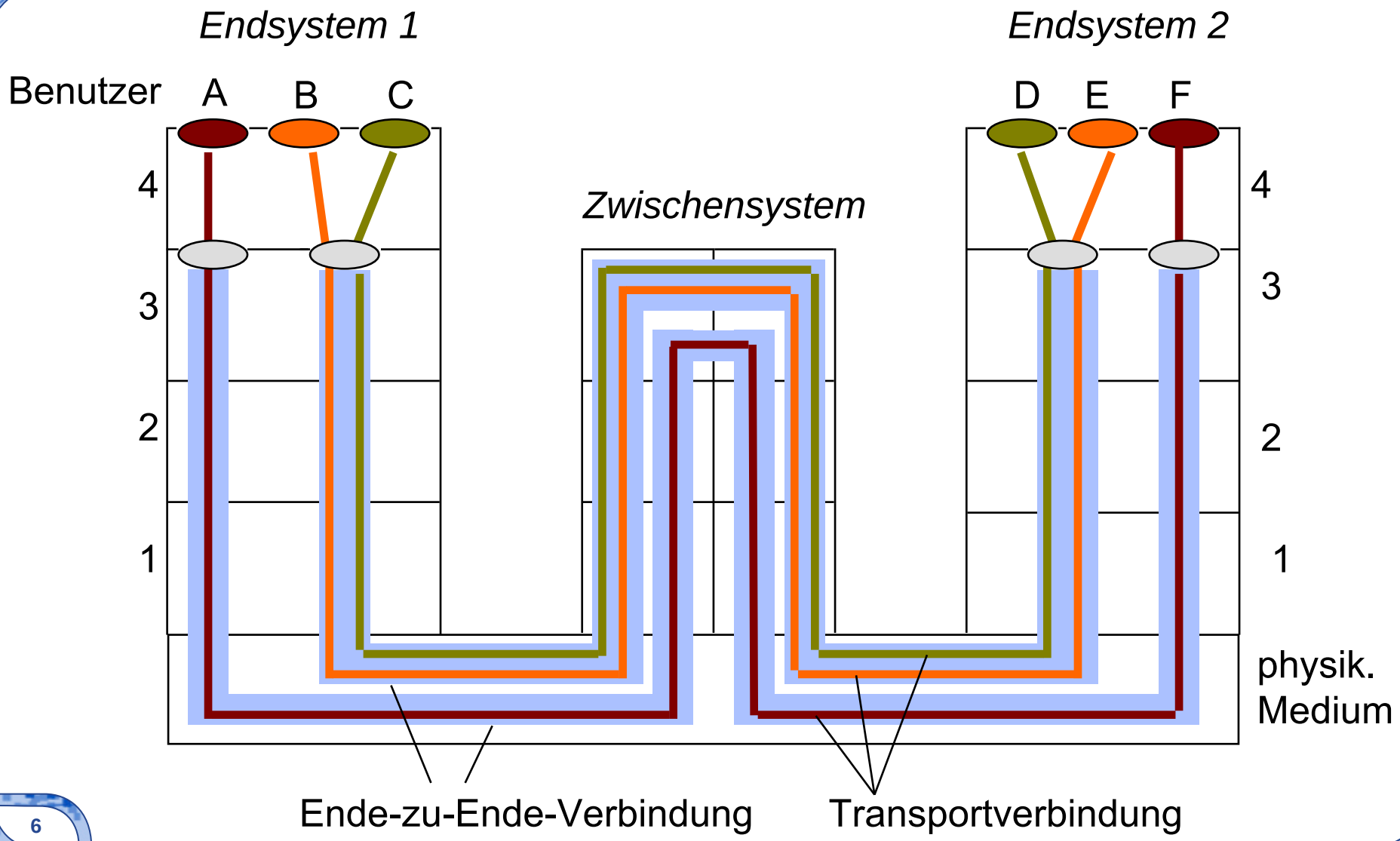
# Transport im Beispiel



# Transport- vs. Ende-zu-Ende-Verbindung



# Einordnung der Transportschicht



# Zuverlässige Transportdienste

- Drei Phasen von Benutzer-zu-Benutzer-Verbindungen
  - Verbindungsaufbau
  - Datenübertragung
  - Verbindungsabbau
- Segmentieren/Reassemblieren von Daten
- Fehlererkennung und -behebung, Sicherstellung der Reihenfolge
  - Quittungen
  - Sequenznummern
- Flusskontrolle, Staukontrolle
- Multiplexen von Transportverbindungen
- ...

# Verbindungsreferenzen

- Dienen zur Identifikation einer Verbindung
- Haben lediglich lokale Signifikanz
  - Worauf bezieht sich lokal?
- Müssen nach Verbindungsabbau „eingefroren“ werden
  - Weshalb?
  - Problem: Abschätzen der Dauer des Einfrierens

# Dienstgüteparameter

- Die Dienstgüte (QoS: Quality-of-Service) bezieht sich auf bestimmte Eigenschaften am Dienstzugangspunkt einer Transportverbindung
- QoS-Parameter werden zwischen den Benutzern und Erbringern eines Transportdienstes ausgehandelt
  - Sie beziehen sich jeweils auf eine Transportverbindung
- Typische Parameter können sein:
  - Verzögerung beim Verbindungsaufbau
  - Fehlerwahrscheinlichkeit beim Verbindungsaufbau
  - Durchsatz während des Datentransfers
  - Übertragungsverzögerung von Dateneinheiten
  - Übertragungsfehlerwahrscheinlichkeit von Dateneinheiten
  - ...

# Transportprotokolle: Ein kurzer Überblick

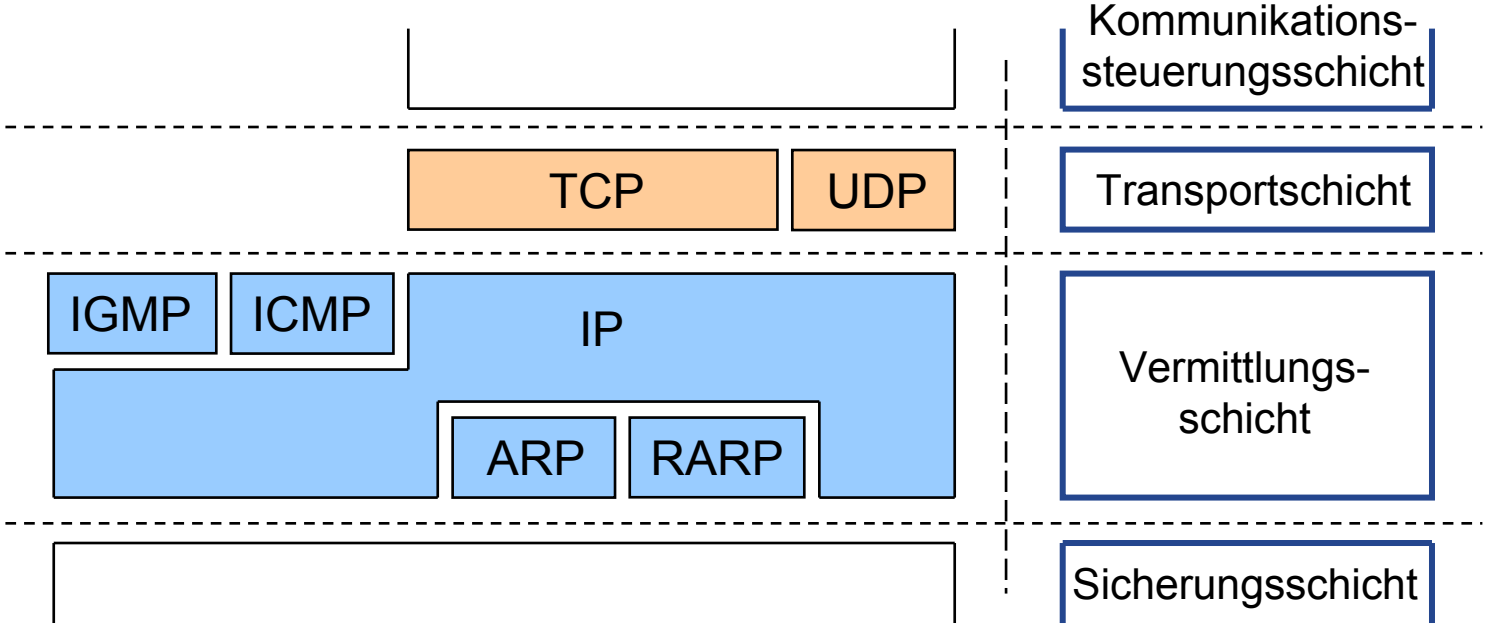
- Transportprotokolle der ISO/OSI-Welt in der Praxis wenig verbreitet
  - 5 verschiedene Transportprotokollklassen: TP0, ..., TP4
  - erforderliche Klasse hängt ab vom Dienst der Vermittlungsschicht sowie vom geforderten Dienst an der Transportdienstschnittstelle
    - Transportprotokoll schließt die „Lücke“ dazwischen
  - im Rahmen der Vorlesung werden ISO-Transportprotokolle nicht weiter betrachtet
    - TP4 ist recht ähnlich zu dem im Internet eingesetzten Transportprotokoll TCP
      - viele Neuerungen wurden dort allerdings nicht mehr nachgeführt (z.B. Staukontrolle)

# Transportprotokolle: Ein kurzer Überblick

- Transportprotokolle der Internet-Welt
  - sehr weit verbreitet, deshalb im Folgenden detaillierter beschrieben
    - UDP (verbindungsloser, unzuverlässiger Transportdienst)
    - TCP (verbindungsorientierter, zuverlässiger Transportdienst)
- Fortgeschrittene Transportprotokolle
  - werden heute vor allem im Forschungsbereich untersucht, beispielsweise zur
    - Unterstützung von QoS-Anforderungen multimedialer Anwendungen
    - Unterstützung von Gruppenkommunikation
    - Unterstützung von Signalisierung
  - werden in der Vorlesung „Next Generation Internet“ näher betrachtet

# Die TCP/IP-Protokollfamilie – Überblick

- Die Bezeichnung TCP/IP wird häufig als Synonym für die gesamte Protokollfamilie verwendet
- Einordnung der Internetprotokolle in das ISO/OSI-Referenzmodell:



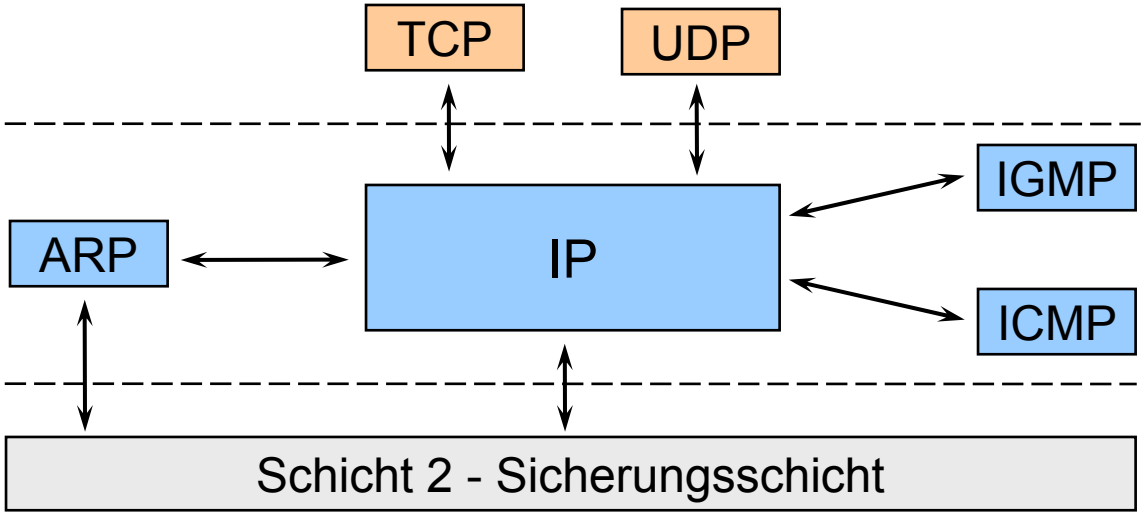

- Obwohl ICMP und IGMP den IP-Dienst nutzen, werden sie dennoch der Vermittlungsschicht zugeordnet
- In den anwendungsbezogenen Schichten 5–7 werden im Internet Protokolle wie z.B. FTP, Telnet oder SMTP eingesetzt (Schichten 5–7 im Internet zusammengefasst zur Anwendungsschicht)

# Die TCP/IP-Protokollfamilie: Protokollaufgaben

- *TCP* (Transmission Control Protocol):
  Stellt zuverlässigen Transportdienst bereit
- *UDP* (User Datagram Protocol):
  Stellt unzuverlässigen Transportdienst bereit
- *IP* (Internet Protocol):
  Wegewahl und unzuverlässige Übertragung von Datagrammen
- *ICMP* (Internet Control Message Protocol): Unterstützt den Austausch von Kontrollinformationen innerhalb der Vermittlungsschicht
- *IGMP* (Internet Group Management Protocol):
  Unterstützt die Verwaltung von Kommunikationsgruppen
- *ARP* (Address Resolution Protocol): Unterstützt die Zuordnung von IP-Adressen zu den entsprechenden Adressen der Sicherungsschicht
- *RARP* (Reverse Address Resolution Protocol):
  Stellt die Umkehrfunktion von ARP zur Verfügung

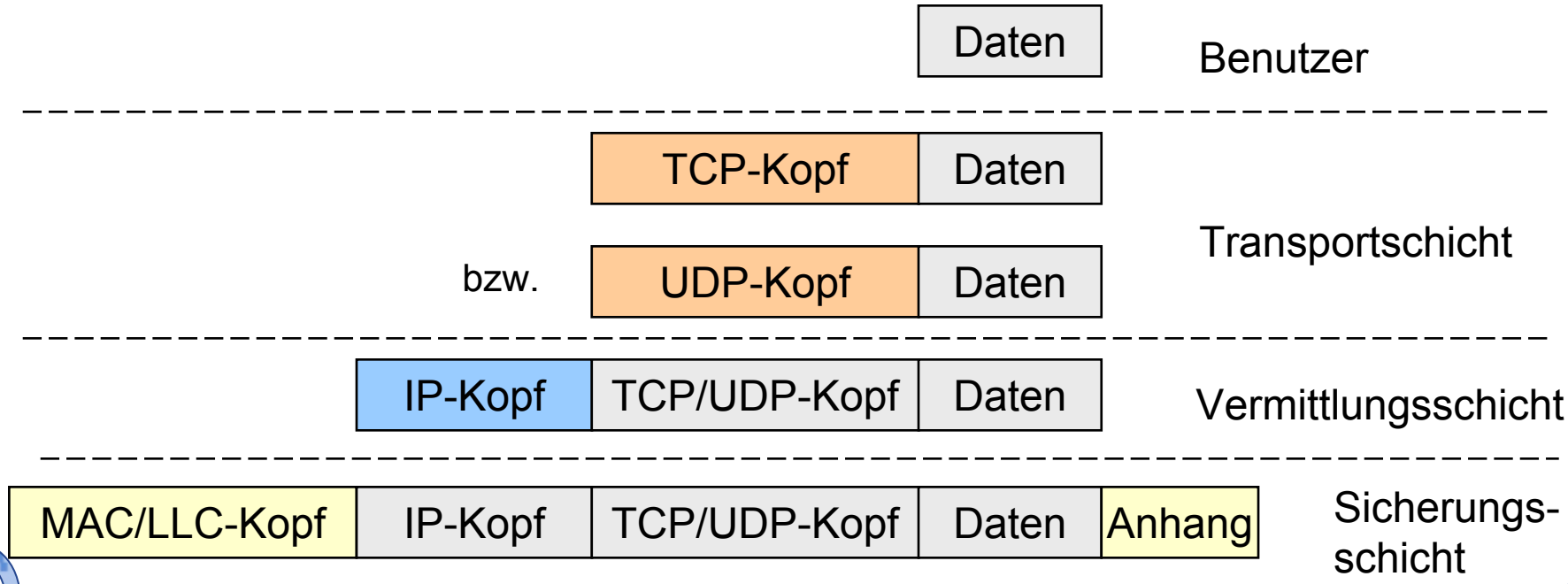
# Zusammenspiel der Protokollinstanzen

- Senden
  - TCP- bzw. UDP-Instanz übergibt Daten mit IP-Adresse des Empfängers zur Übertragung an IP-Instanz
  - IP-Instanz beauftragt ARP-Instanz mit Ermittlung der entsprechenden Schicht-2-Adresse
  - IP-Instanz übergibt Dateneinheiten mit ermittelter Schicht-2-Adresse an Instanz der Sicherungsschicht
- Empfangen
  - Unterscheidung zwischen den verschiedenen Protokollen (TCP, UDP, ICMP, …) anhand des „Protocol“-Feldes im IP-Kopf
  - IP-Instanz reicht empfangene Daten an TCP- bzw. UDP-Instanzen weiter



# Zusammenspiel der Schichten

- IP leitet Dateneinheiten durch das Netz zum Empfänger
- TCP/UDP fügen Anwendungsadressierung zu IP hinzu
  - Ports
- TCP stellt darüberhinaus zuverlässigen Dienst bereit
- Protokolldateneinheiten (PDUs) werden gekapselt



# Internet-Anwendungen und Transportprotokolle

| Anwendung | Anwendungs-protokoll | Verwendetes Transportprotokoll |
|---|---|---|
| E-Mail | SMTP | TCP |
| Remote Terminal Access | Telnet | TCP |
| Web | HTTP | TCP |
| Dateitransfer | FTP | TCP |
| Entfernter Fileserver | NFS | i.d.R. UDP |
| Streaming Multimedia | RTSP, … | i.d.R. UDP |
| Internet-Telefonie | SIP/H.323 & RTP | i.d.R. UDP |
| Netzmanagement | SNMP | i.d.R. UDP |
| Intra-Domain-Routing | RIP | i.d.R. UDP |
| Inter-Domain-Routing | BGP | TCP |
| Namensübersetzung | DNS | i.d.R. UDP |

# Anzahl & Charakteristik geöffneter Verbindungen bei Webseitenaufruf

- Was passiert beim Aufruf einer Webseite?
  - Früher waren Webseiten einfach strukturiert
    - Eine zentrale HTML-Seite mit einigen Grafiken auf demselben Server
- Untersuchung der Verbindungscharakteristika bei Webseitenaufrufen
  - Studie vom Frühjahr 2009 [Char09]
  - Messgegenstände: 100 „beliebteste“ Homepages der USA
  - Messwerte von Interesse
    - Übertragene Datenmenge [Bytes]
    - Anzahl geöffneter TCP-Verbindungen
    - Anzahl unterschiedlicher Server, zu denen Verbindungen aufgebaut werden
    - Anzahl geladener Elemente (nach HTTP GET und POST Requests)
    - Anzahl verschiedener Netzwerkpräfixe (NP) und Autonomer System (AS), die involviert sind
    - Anzahl verschiedener DNS „Second Level Domains“
      - www.google.com, www.uni-karlsruhe.de
      - Können Rückschlüsse auf beteiligte Unternehmen bieten

# Anzahl & Charakteristik geöffneter Verbindungen bei Webseitenaufruf (2)

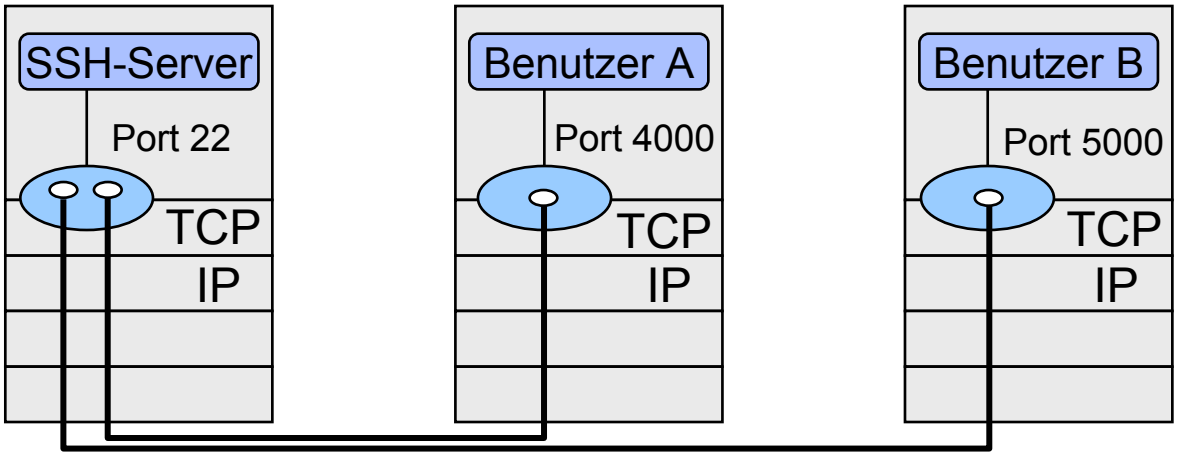
| | Bytes | Verbind. | Server | Elemente | AS | DSN SLDs |
|---|---|---|---|---|---|---|
| Minimum | 943 | 1 | 1 | 2 | 1 | 1 |
| Durchschn. | 600 kB | 22,94 | 8,24 | 62,18 | 5,04 | 5,15 |
| Maximum | 11,7 MB | 107 | 21 | 284 | 13 | 15 |

- Um eine komplette Homepage zu laden müssen durchschnittlich …
  - etwa 23 TCP-Verbindungen geöffnet werden
  - mehr als acht verschiedene Server vom Client kontaktiert werden, die in mehr als fünf autonomen Systemen liegen
    - „Helper Services“ insbesondere von akamai.net, google.com, doubleclick.net
  - mehr als 60 Elemente geladen werden
  - Verbindungen zu Servern mit mehr als fünf verschiedenen Second-Level-Domains aufgebaut werden

# Adressierung

- Adressierung von TCP/UDP-Anwendungen geschieht über *Ports*
- Viele Portnummern kleiner 1024 sind für häufig benutzte Anwendungen reserviert
  - z.B. 21 für FTP, 22 für SSH, 23 für Telnet, 80 für HTTP
- Eine Anwendungsadresse besteht aus der Internetadresse eines Rechners und einem Port
  - Notation: (IP-Adresse:Portnummer) ⇨ Internet-weit eindeutig
- Beispiel: SSH-Server am Telematikinstitut über 141.3.70.4:22 erreichbar

IP-Adresse Port



# Port-Nummern-Konventionen (well-known ports)

- Von einer Anwendung muss der richtige *Port* gewählt werden, um auf der Gegenseite mit der gewünschten Anwendung zu kommunizieren
  - 13: Tageszeit
  - 20: FTP Daten
  - 25: SMTP
    (Simple Mail Transfer Protocol)
  - 53: DNS
    (Domain Name Server)
  - 80: HTTP
    (Hyper Text Transfer Protocol)
  - 119: NNTP
    (Network News Transfer Protocol)

```
> telnet osiris 13
Trying 129.13.3.121...
Connected to osiris.
Escape character is '^]'.
Mon Aug  4 16:57:19 1997
Connection closed by foreign host
```

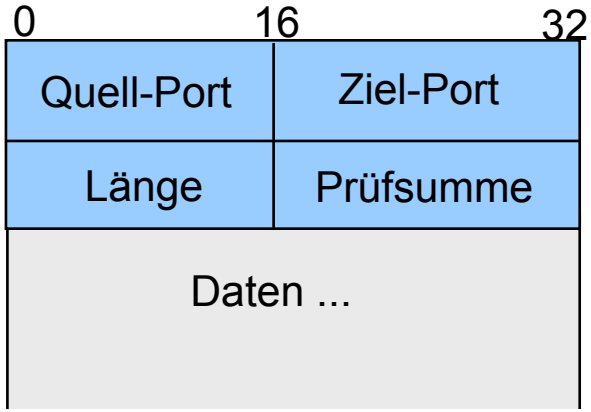
*„well-known“ und traditionell!*

```
> telnet sokrates 25
Trying 129.13.3.161...
Connected to sokrates .
Escape character is '^]'.
220 sokrates ESMTP Sendmail 8.8.5/8.8.5;
Mon, 4 Aug 1997 17:02:51 +0200
HELP
214-This is Sendmail version 8.8.5
214-Topics:
214-    HELO    EHLO    MAIL    RCPT    DATA
214-    RSET    NOOP    QUIT    HELP    VRFY
214-    EXPN    VERB    ETRN    DSN
214-For more info use "HELP <topic>".
...
214 End of HELP info
```

# 9.2 UDP (User Datagram Protocol)

- Sehr einfaches Transportprotokoll
  - RFC 768, August 1980
- Multiplexen / Demultiplexen von Anwendungsprozessen über das Netzwerk
- Einfache Fehlerkontrolle
- Keine Verbindung, keine Zuverlässigkeit
- Simplex
- UDP-Dateneinheit:

[RFC768]

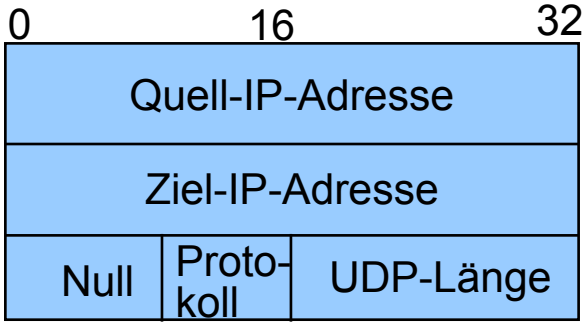


# Eigenschaften von UDP

- Keine Verbindungsaufbauphase
  - Verzögerung bis zum Aufbau einer Verbindung entfällt
  - Daten können sofort gesendet werden
- Kein Verbindungszustand
  - Im Endsystem müssen keine verbindungsrelevanten Informationen gehalten werden
    - z.B. Flusskontrollfenster, Staukontrollfenster, Sequenznummern
  - Server kann mittels UDP typischerweise mehr aktive Clients unterstützen als mit TCP
- Geringer Overhead in der Dateneinheit
  - Lediglich Adressen sowie Längenfeld und Prüfsumme
- Unreguliertes Senden
  - UDP kann Daten so schnell senden wie sie von der Anwendung geliefert werden und wie sie vom Netz abgenommen werden
    - Verluste möglich
- Vorteil
  - Sehr einfaches Protokoll mit äußerst geringem Overhead

# UDP-Prüfsumme

- Berechnung kann deaktiviert werden, wird aber mit IPv6 verpflichtend
  - Gleicher Algorithmus wie bei IP (Einer-Komplement der Summe der Einer-Komplemente von 16-Bit-Worten)
- UDP-Prüfsumme wird berechnet über
  - UDP-Kopf (Prüfsummenfeld ist initial mit 0 zu belegen)
  - UDP-Daten
  - Pseudoheader
- Pseudoheader
  - Teil des IP-Kopfes
    - Quell-IP-Adresse
    - Ziel-IP-Adresse
    - Protokoll
    - Länge der UDP-Dateneinheit (ohne Pseudoheader)
  - Erlaubt Schutz gegen fehlgeleitete UDP-Dateneinheiten (Überprüfung, ob Daten zwischen den beiden korrekten Endpunkten ausgetauscht wurden)
  - Ist Prüfsummenfeld = 0, dann wird keine Prüfsummenberechnung gewünscht
  - Bei berechneter Prüfsumme 0 wird 0xFFFF übertragen



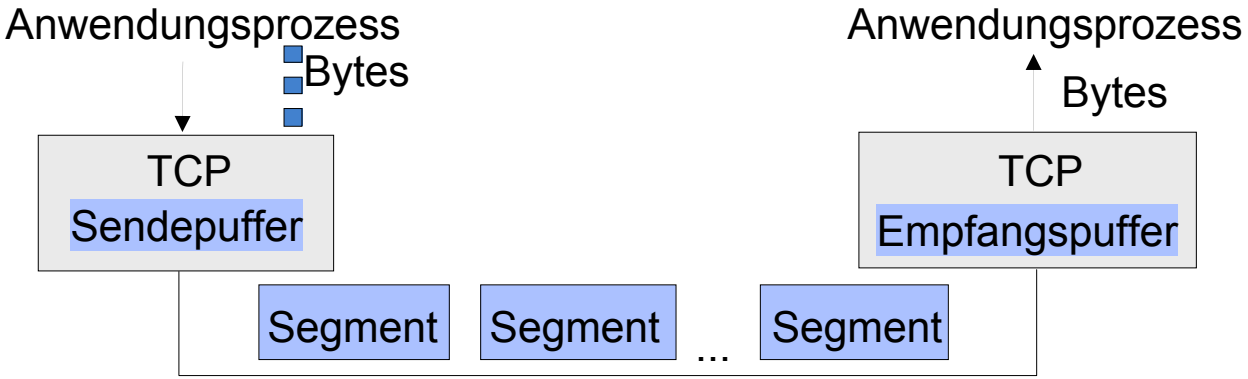
# 9.3 TCP (Transmission Control Protocol)

- Stellt zuverlässigen Transportdienst zur Ende-zu-Ende-Kommunikation zur Verfügung
- Arbeitet an der Dienstschnittstelle mit einem Bytestrom
  - RFC 793, September 1981



[RFC793]

- Die zu IP weitergereichte Dateneinheit wird als TCP-Segment bezeichnet




- Neue / optimierte Funktionen
  - Verbesserung des Timeout-Mechanismus
    - Zeitstempel im Segment-Kopf
  - Erhöhen der Fenstergröße
    - Basiseinheit kann skaliert werden

# Senden einer TCP-Dateneinheit

- Empfang von der Anwendung: Bytestrom
- Senden an IP: TCP-Dateneinheiten (TCP-Segmente)
- Problem
  - Wann wird aus dem Bytestrom eine TCP-Dateneinheit an IP weitergegeben?
- RFC 793 (TCP)
  - TCP should „send that data in segments at its own convenience“  [RFC793]
- Alternativen
  - MSS: Maximum Segment Size
    - Kann sich an der Größe der maximalen Dateneinheit des direkt angeschlossenen Netzes orientieren, an der eventuell ermittelten Maximum Transfer Unit (MTU) oder an der Segmentierungsgröße, mit der IP umgehen können muss
    - Gibt Länge der Anwendungsdaten an, nicht Länge der TCP-Dateneinheit
    - Typische Größen (vermeiden das Segmentieren durch IP)
      - 1460 Byte, 536 Byte, 512 Byte
  - Push (PSH-Flag im Kopf der TCP-Dateneinheit)
    - Sender verlangt hiermit das sofortige Versenden der übergebenen Daten (Vorrangdaten)
    - Wird beispielsweise bei Telnet verwendet
  - Zeitgeber
    - Nach einem gewissen Zeitintervall der Inaktivität werden vorhandene Daten gesendet

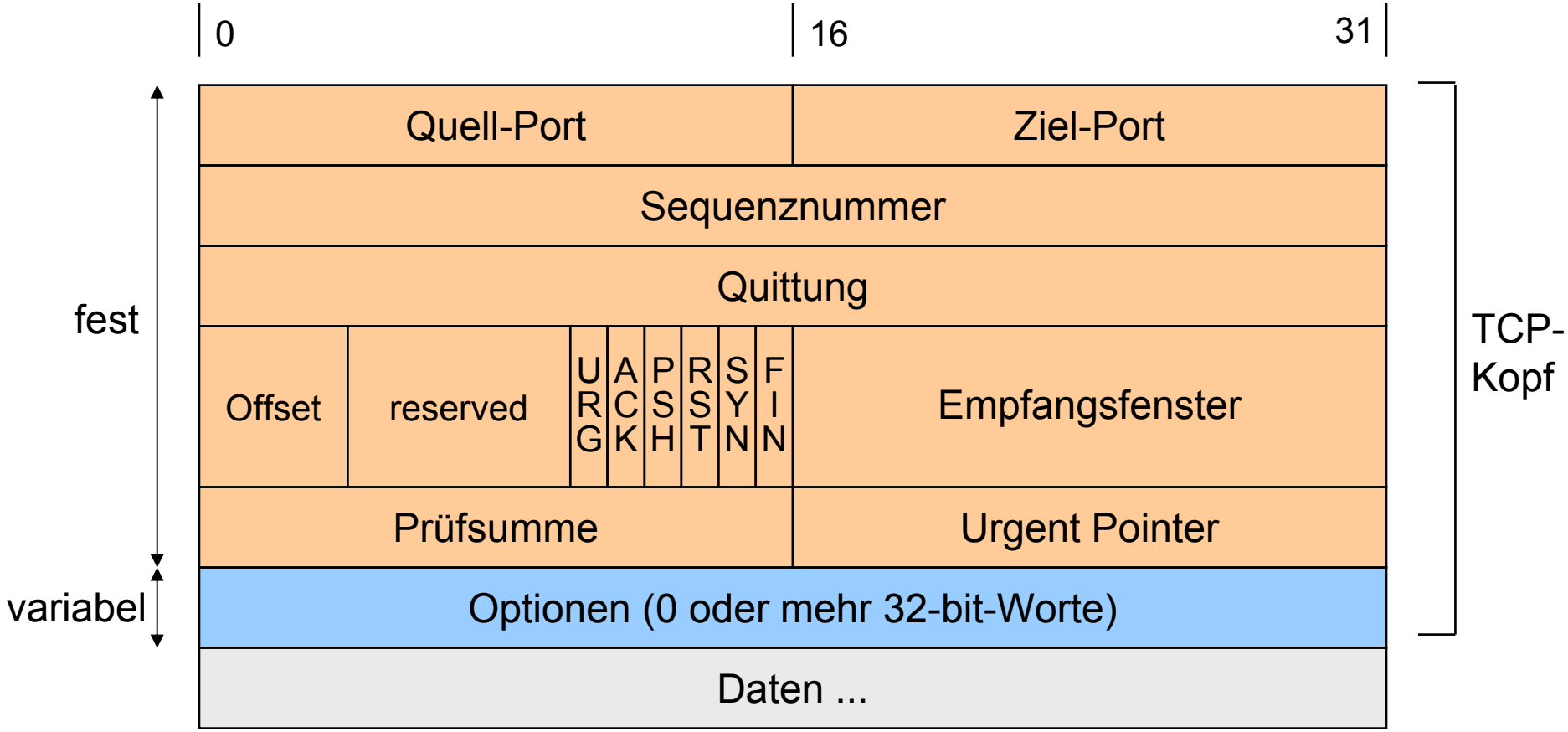
# TCP: Eigenschaften und Dienste

- TCP ermöglicht eine zuverlässige Übertragung eines Bytestroms zwischen zwei Anwendungen über einen unzuverlässigen Vermittlungsdienst
- TCP wurde so entworfen, dass es sich dynamisch an die Eigenschaften des Internet (z.B. heterogene Topologien, schwankende Bandbreiten) anpasst
- Verbindungsverwaltung
  - Verbindungsaufbau zwischen zwei „Sockets“
    - Verbindungsendpunkt innerhalb des Dienstzugangspunkts zur Transportschicht (TSAP)
  - Datentransfer über eine virtuelle Verbindung
  - Gesicherter Verbindungsabbau
    - Alle übertragenen Daten müssen quittiert sein
- Multiplexen
  - Mehrere Verbindungen der Transportschicht können über eine Ende-zu-Ende-Verbindung laufen

# TCP: Eigenschaften und Dienste

- Datenübertragung
  - Duplex
  - Fehlerkontrolle
    - Sequenznummern, Prüfsumme, Quittierung, Übertragungswiederholung im Fehlerfall
  - Reihenfolge, Flusskontrolle (durch Fenstermechanismus) und Staukontrolle
  - Unterstützung von Prioritäten
  - Zeitbehaftete Daten
    - Falls die Auslieferung in bestimmter Zeit nicht möglich ist, wird der Dienstbenutzer informiert
- Fehleranzeige
  - Treten während der Verbindung Störungen auf, wird der Benutzer darüber in Kenntnis gesetzt

# Format einer TCP-Dateneinheit



# Felder einer TCP-Dateneinheit

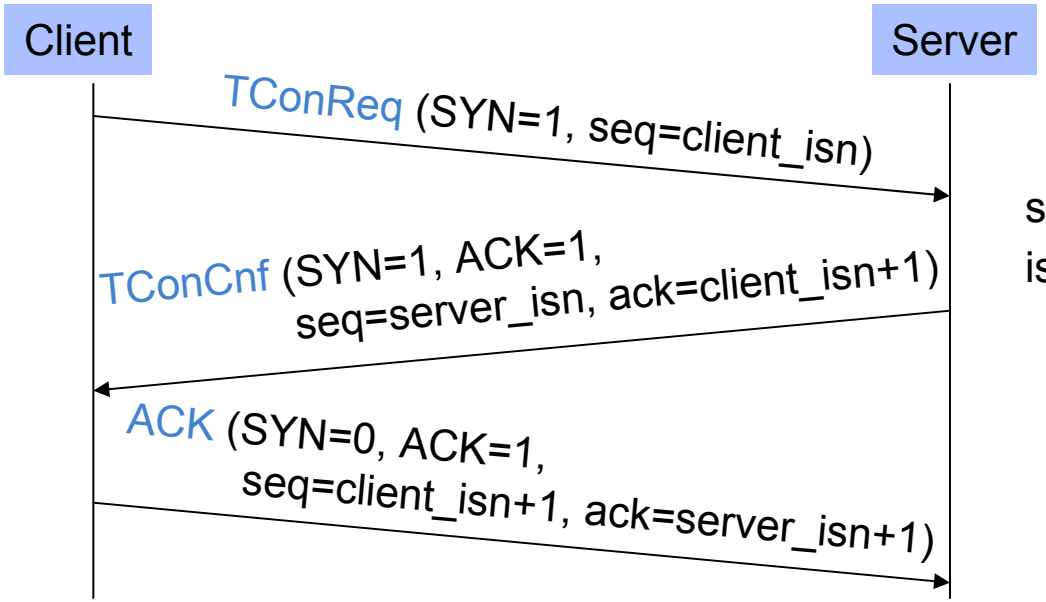
- Quell-Port und Ziel-Port
  - Identifizieren die Endpunkte der Verbindung
    - Wie die Endsysteme ihre Ports zuweisen, bleibt ihnen überlassen
      - Wobei standardisierte Ports auch für zugehörige Dienste verwendet werden sollten
- Sequenznummer
  - Enthält Sequenznummer gemessen in Byte (nicht pro Dateneinheit!)
- Quittung
  - Die nächste vom Empfänger erwartete Sequenznummer
- Offset
  - Anzahl der 32-Bit-Wörter im TCP-Kopf
- URG
  - Wird auf 1 gesetzt, falls der Urgent Pointer verwendet wird
    - Wird in der Regel nicht benutzt
- SYN
  - Wird beim Verbindungsaufbau verwendet, um *Connection Request* (TConReq) bzw. *Connection Confirmation* (TConCnf) anzuzeigen
- ACK
  - Unterscheidet bei gesetztem SYN-Bit eine TConReq-PDU von einer TConCnf-PDU
  - Signalisiert die Gültigkeit des Quittungs-Feldes

# Felder einer TCP-Dateneinheit

- FIN
  - Gibt an, dass der Sender keine Daten mehr senden möchte
- RST
  - Wird benutzt, um eine Verbindung zurückzusetzen
- PSH
  - Signalisiert, dass die übergebenen Daten sofort weitergeleitet werden sollen
  - Gilt sowohl für den Sender als auch für den Empfänger
  - Wird in der Regel nicht benutzt
- Empfangsfenster
  - Dient zur Flusskontrolle
  - Anzahl der Bytes, die nach dem höchsten bestätigten Byte gesendet werden dürfen
- Prüfsumme
  - Enthält die Prüfsumme über TCP-Kopf und Daten (wie bei UDP)
- Urgent-Zeiger
  - Relativer Zeiger auf wichtige Daten
- Das Optionen-Feld
  - kann Optionen variabler Länge aufnehmen (n * 32 Bit)

# TCP-Verbindungsverwaltung

- Verbindungsaufbau (3-Wege)
  - Unterscheidung: Client (initiiert Verbindung) und Server (wartet auf Verbindungswünsche)

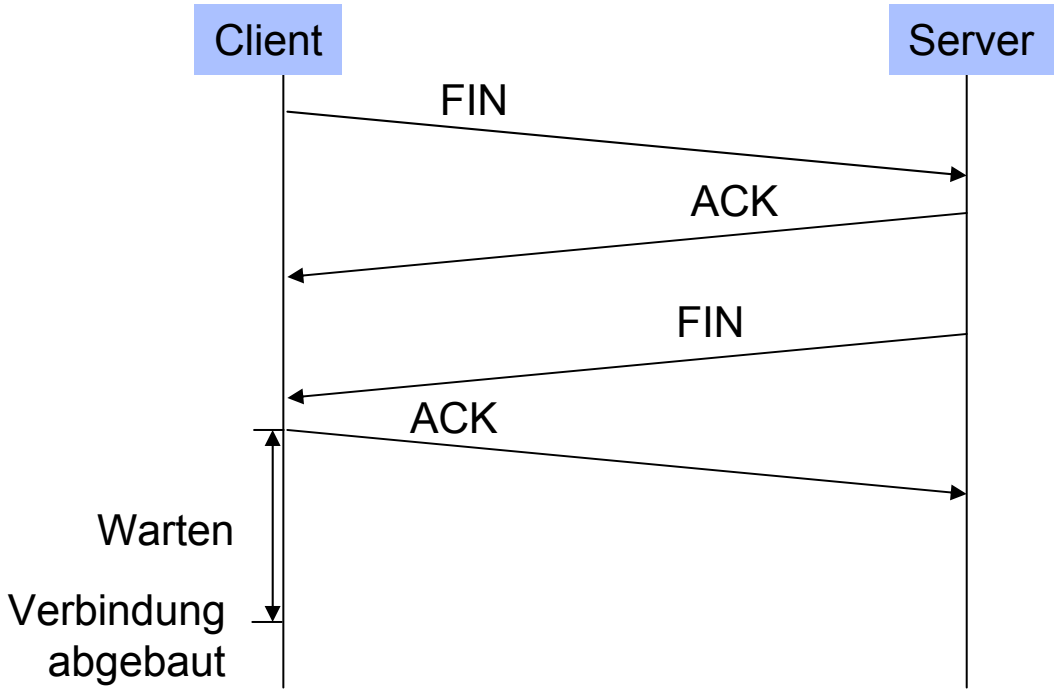



seq: Sequenznummer
isn: Initiale Sequenznummer

  - TConReq und TConCnf führen keine Nutzdaten mit sich
    - Festlegung der initialen Sequenznummern (..._isn)
  - Einige Optionen sind nur beim Verbindungsaufbau erlaubt bzw. ausgehandelt (z.B. WindowScaling)

# TCP-Verbindungsverwaltung

- Verbindungsabbau
  - Kann sowohl vom Client als auch vom Server initiiert werden



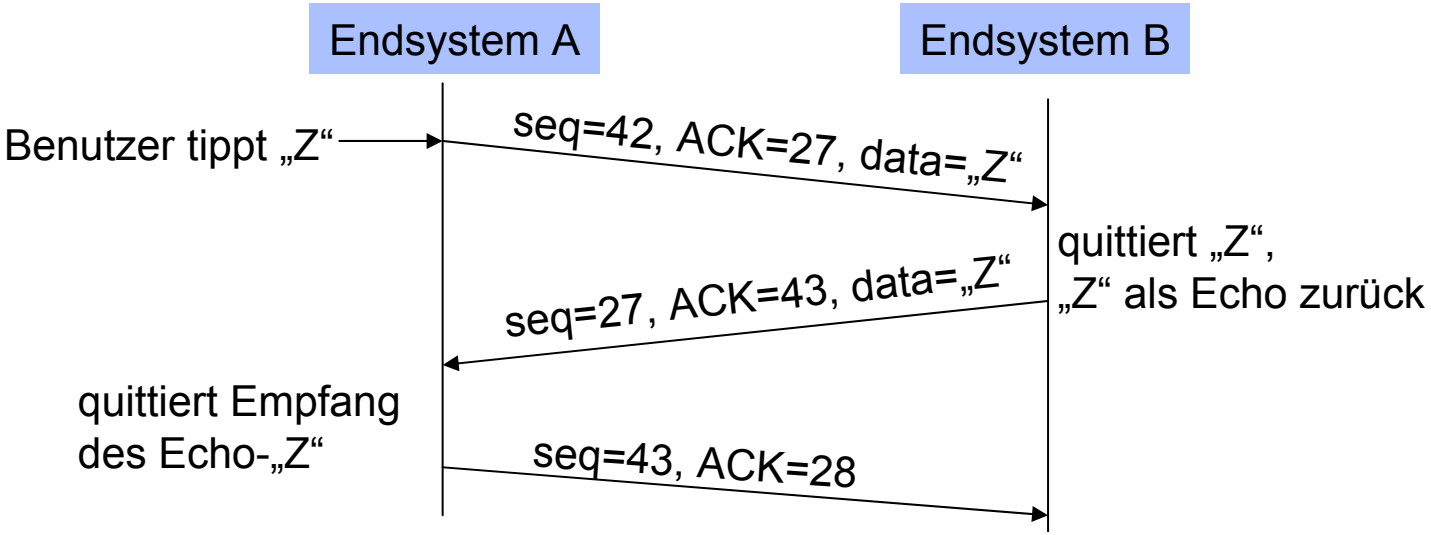

  - Nach letztem ACK wird noch gewartet, bevor lokaler Kontext gelöscht wird
    - z.B. könnte ACK verloren gehen und Server dann sein FIN wiederholen
  - Weitere Varianten für simultanen Verbindungsabbau existieren

# Verbindungsabbau

- Ordnungsgemäßer Abbau
  - Beide Anwendungsprozesse müssen unabhängig voneinander ihre „Hälften“ der Verbindung schließen
    - Wurde lediglich eine Richtung geschlossen, so können in der anderen Richtung noch Daten gesendet werden
      - Damit ist noch ein unidirektionaler Datenfluss möglich
      - In „geschlossener“ Richtung können noch Kontroll-Dateneinheiten gesendet werden
- Abbruch einer Verbindung
  - Verbindung kann mit Reset abgebrochen werden
    - Mittels Dateneinheit mit gesetztem RST-Bit

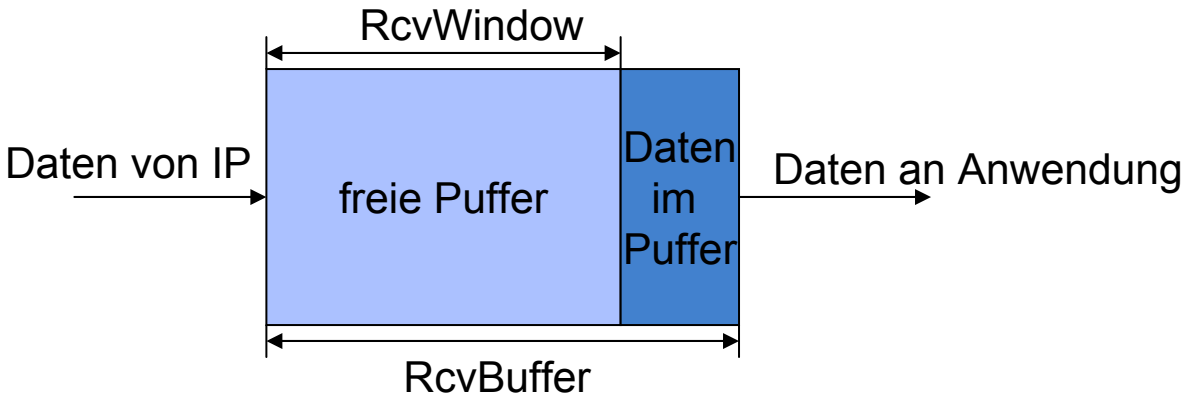
# TCP-Sequenznummern und -Quittungen

- Sequenznummern
  - pro Byte, nicht pro Dateneinheit
  - Initiale Sequenznummer wird von beteiligten Endsystemen zufällig gewählt
  - Nummer des ersten Bytes in der Dateneinheit
- Quittungsverfahren
  - positive kumulative Quittungen
    - Erweiterungen wurden definiert (selektive Quittungen)
  - Sequenznummer des nächsten Bytes, das vom Kommunikationspartner erwartet wird
- Beispiel (Telnet-Szenario):



# TCP-Flusskontrolle

- Empfangsfenster
  - Verwaltung des Pufferplatzes, den der Empfänger für diese Verbindung zur Verfügung stellt (explizite Kreditvergabe)
    - Feld Empfangsfenster im Kopf der TCP-Dateneinheit
    - Variablen in der Empfängerinstanz
      - RcvWindow: momentan noch freier Pufferplatz beim Empfänger
      - RcvBuffer: gesamter Pufferplatz für zu empfangende Daten beim Empfänger kann dynamisch geändert werden
- Schema beim Empfänger:



# TCP-Flusskontrolle

- Szenario
  - Endsystem A schickt große Datei über TCP an Endsystem B
- Variablen beim Empfänger
  - LastByteRead: Letzte Sequenznummer, die von der Anwendung aus dem Empfangspuffer gelesen wurde
  - LastByteRcvd: Letzte Sequenznummer, die über das Netz empfangen und in den Empfangspuffer geschrieben wurde
- Es muss immer gelten

  LastByteRcvd – LastByteRead <= RcvBuffer
- Empfangsfenster

  RcvWindow = RcvBuffer – (LastByteRcvd – LastByteRead)
- Variablen beim Sender
  - LastByteSent: letzte Sequenznummer, die gesendet wurde
  - LastByteAcked: letzte Sequenznummer, die quittiert wurde
- Es muss immer gelten

  LastByteSent – LastByteAcked <= RcvWindow

# TCP-Fluss-/Staukontrolle

- Flusskontrolle regelt den Datenfluss zwischen den Endsystemen
  - Bei TCP: Fenstermechanismus mit Kreditvergabe
- Staukontrolle befasst sich mit Stausituationen in den Zwischensystemen
  - Bei TCP ist „congestion collapse“ möglich
    - Stau in Zwischensystemen führt oftmals dazu, dass Transportprotokolle nach einer gewissen Zeit (aufgrund von Timeouts) Dateneinheiten wiederholen

    ⇨ Verstärkung der Stausituation!
  - Zusätzlich zum Fenster für die Flusskontrolle Einführung eines Staukontrollfensters (CWnd)
    - Gesendet werden darf maximal bis zum Minimum beider Fenstergrößen, es muss somit immer gelten:
      - LastByteSent – LastByteAcked ≤ min { CWnd, RcvWindow }

# TCP-Staukontrolle

- Ziel
  - Vermeidung von Überlastsituationen im Netz
- Verfahren
  - Nutzung von Timern als Hilfsmittel für Stausignal
  - Bei ausbleibender Quittung wird Stausituation vermutet
    - Daraufhin erfolgt Begrenzung der Datenmenge, die TCP senden darf
    - Sender testet durch langsames Erhöhen der Datenmenge, wie viel er senden kann
  - Hierzu: Zusätzlich zum Staukontrollfenster (CWnd) Einführung einer neuen Variablen
    - Schwellenwert (SSThresh)
      - Variable, die beeinflusst, wie Staukontrollfenster vergrößert wird
- Weitere Details in der Vorlesung Telematik



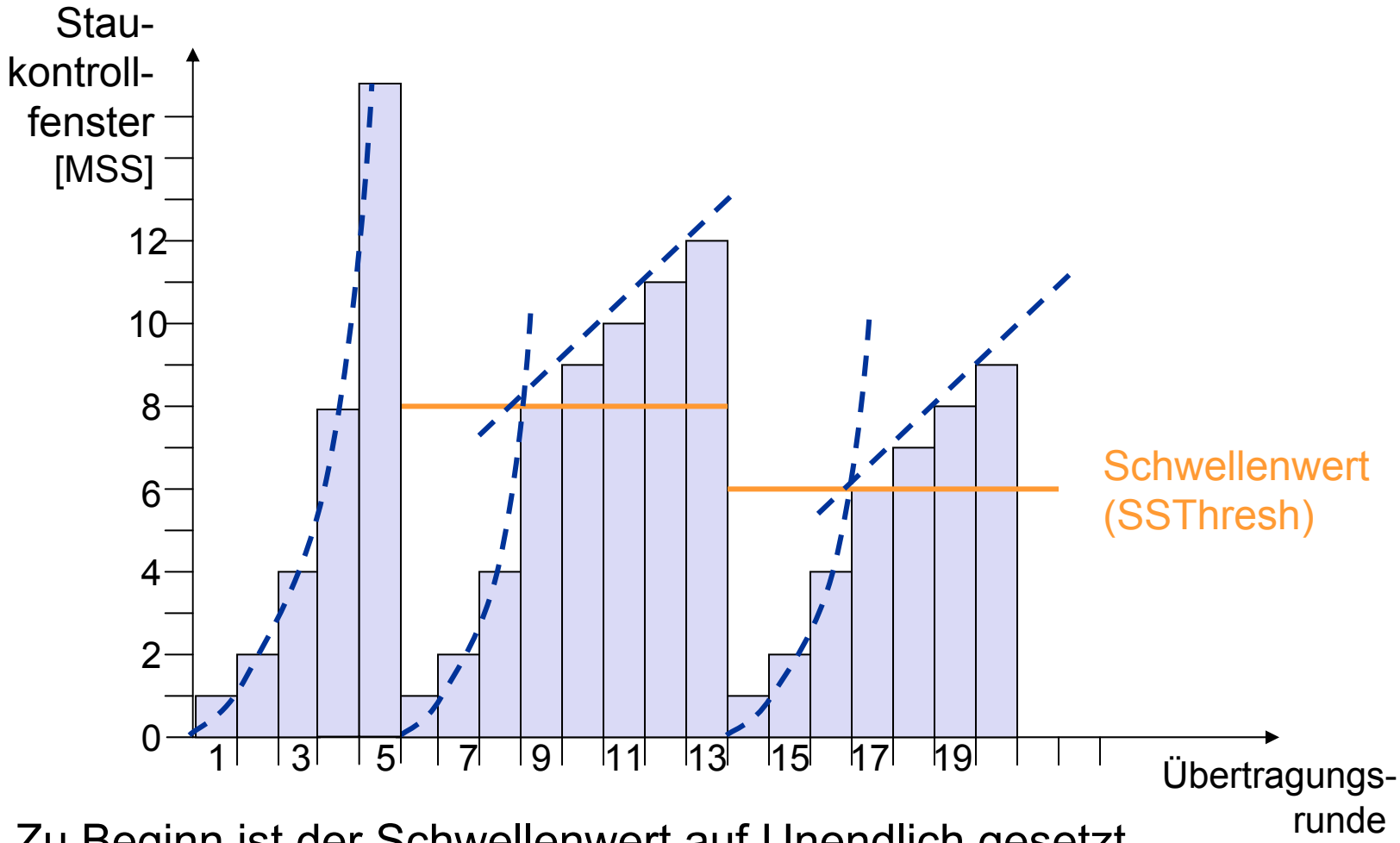
[FlFa99]

# Ablauf TCP-Staukontrolle

- Start: CWnd = 1 MSS [Maximum Segment Size]
- Solange CWnd ≤ SSThresh und Quittungen rechtzeitig empfangen: Slow-Start
  - Exponentielles Erhöhen des Staukontrollfensters
  - Verdopplung des Staukontrollfensters nachdem alle gesendeten Segmente mit ACKs bestätigt wurden (und diese ACKs empfangen wurden)
    - Empfangene Quittung: CWnd + = 1
- CWnd > SSThresh und Quittungen rechtzeitig empfangen: Congestion Avoidance
  - Lineares Erhöhen des Staukontrollfensters
  - Erhöhen des Staukontrollfensters um 1 pro vollständig gesendetem und bestätigtem Sendefenster
    - Empfangene Quittung: CWnd + = 1 / CWnd
- Quittung nicht rechtzeitig empfangen (Timer abgelaufen)
  - Stau vermutet
  - SSTresh auf halbe Größe des Staukontrollfensters setzen: SSThresh = CWnd / 2
  - CWnd zurücksetzen (neue Slow-Start-Phase): CWnd = 1 MSS

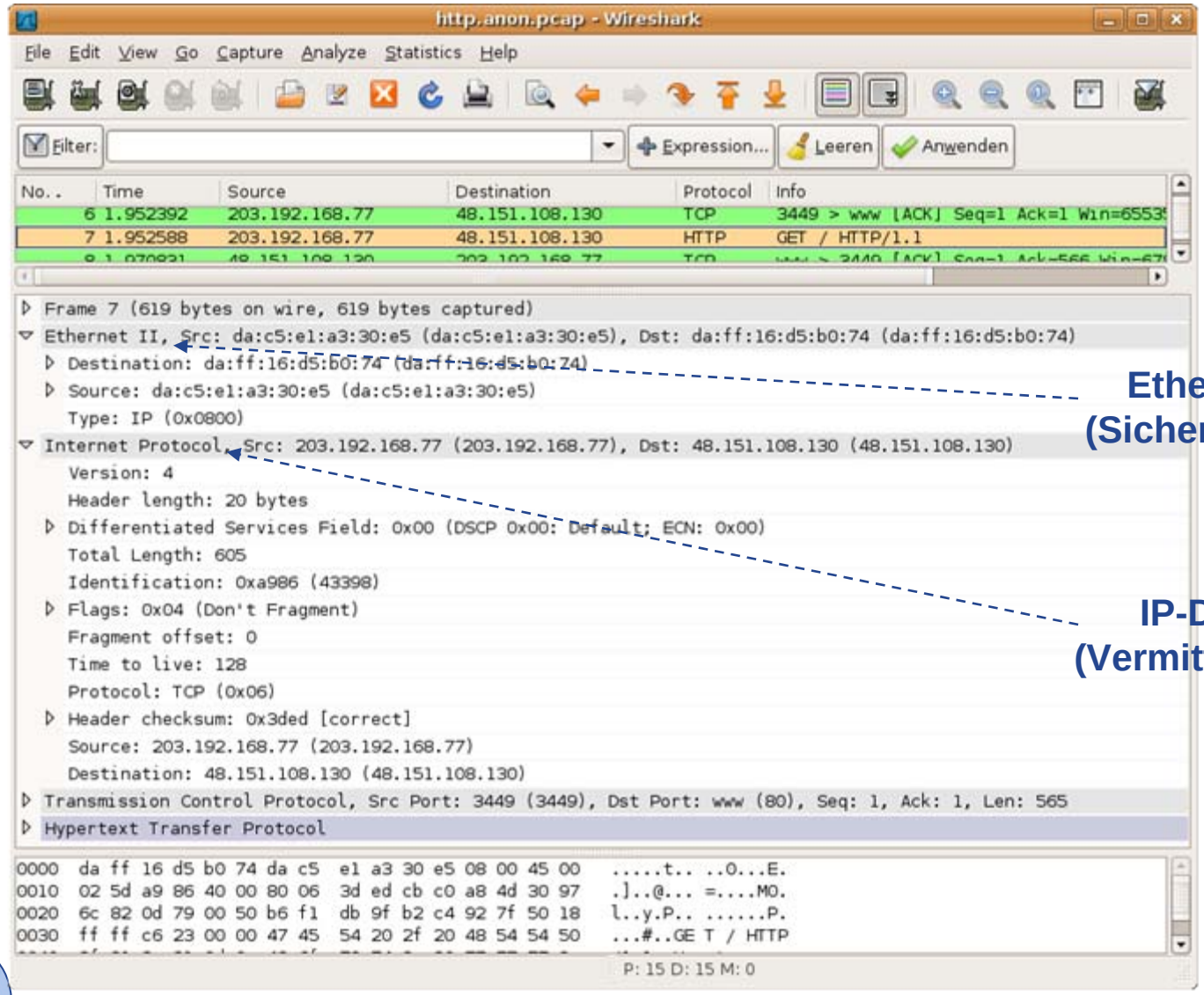
# Entwicklung des Staukontrollfensters

- Beispiel




- Zu Beginn ist der Schwellenwert auf Unendlich gesetzt

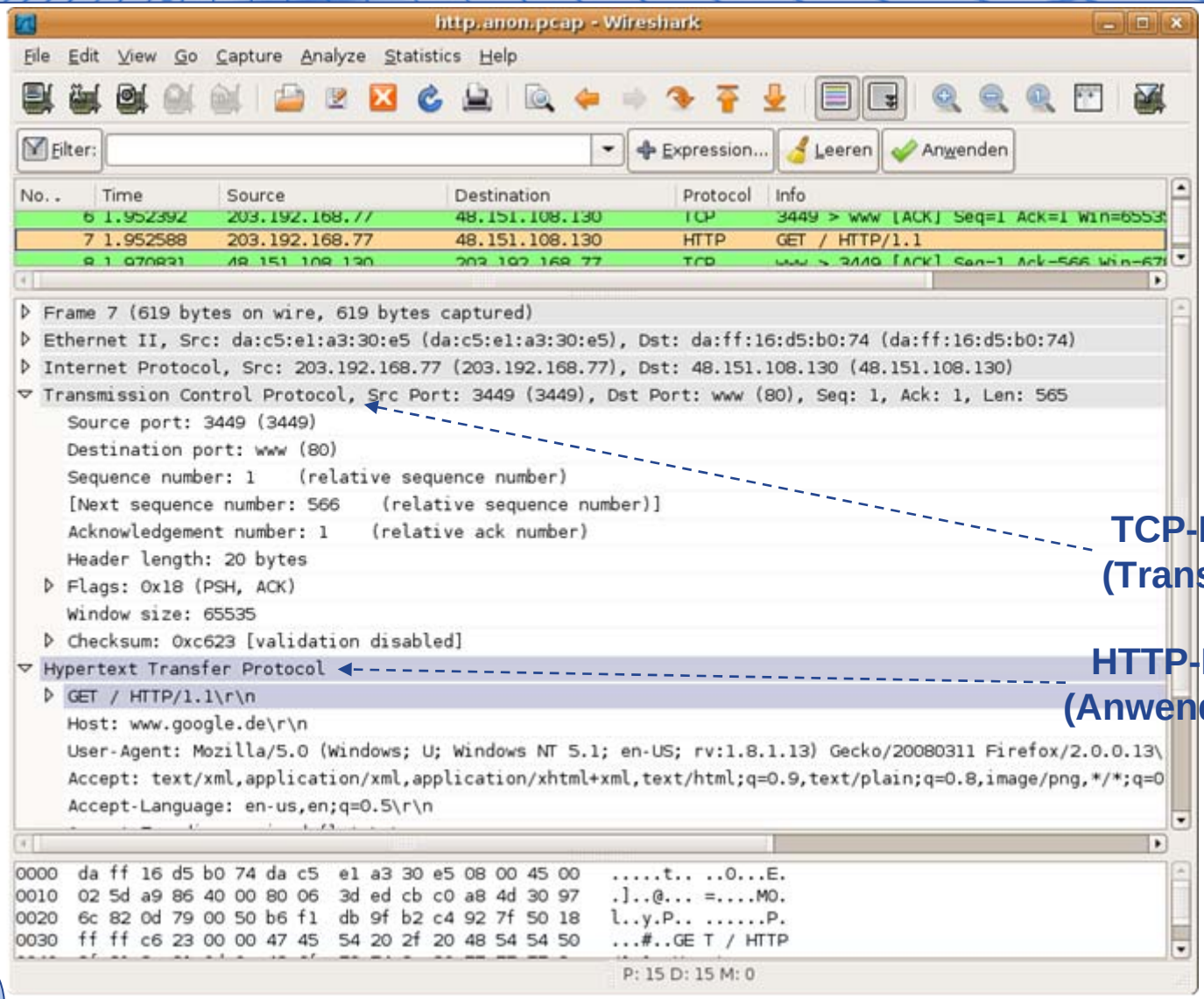
# Anwendungsbeispiel: Wireshark




**Ethernet-Frame (Sicherungsschicht)**

**IP-Datagramm (Vermittlungsschicht)**

# Anwendungsbeispiel: Wireshark (2)

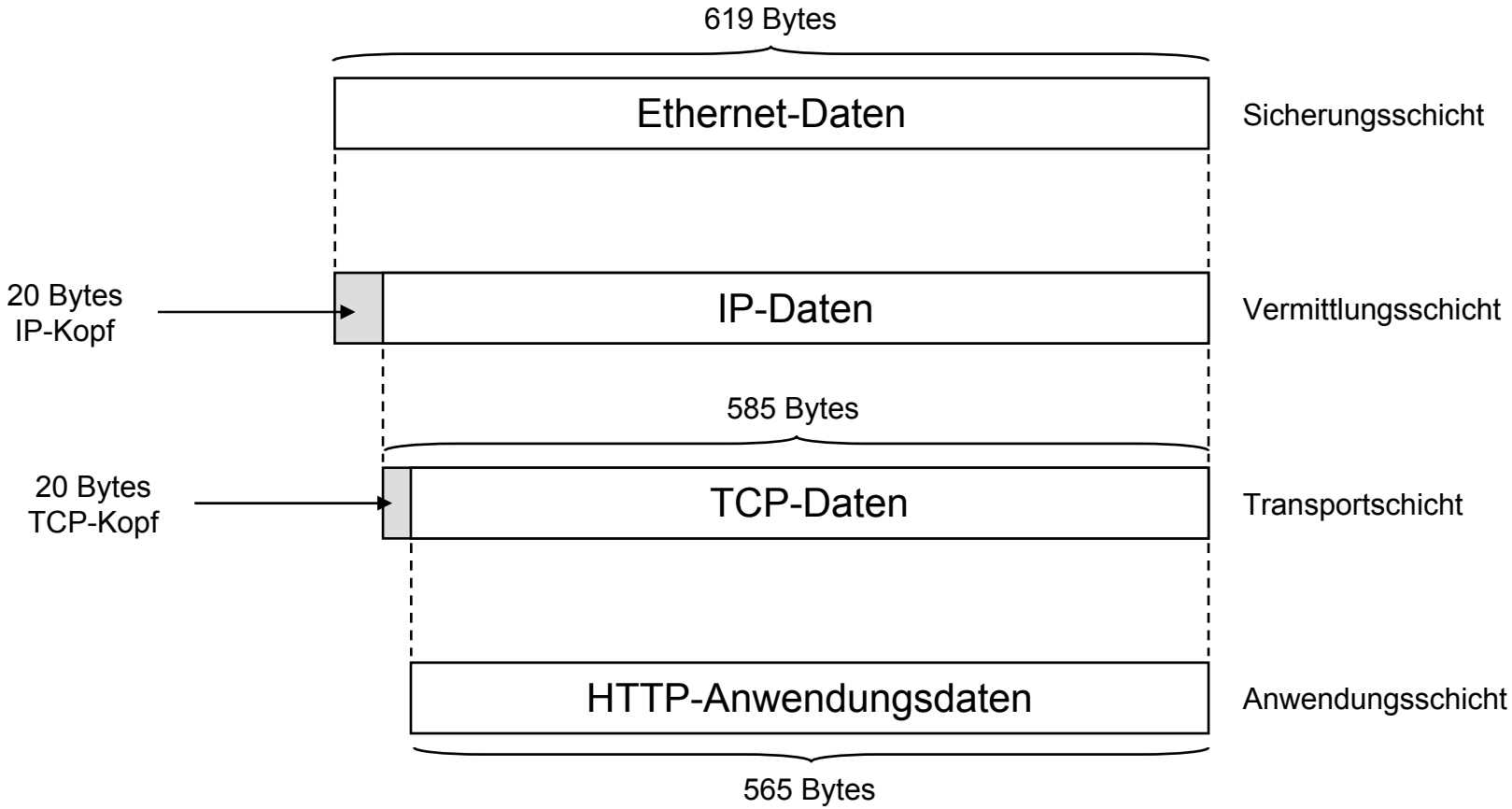



**TCP-Dateneinheit (Transportschicht)**

**HTTP-Dateneinheit (Anwendungsschicht)**

# Anwendungsbeispiel: Wireshark (3)

- Vergleiche Kapitel 4 „Geschichtete Architekturen“



# Ausblick: Protokoll-Engineering

- Detailliertes Protokoll-Engineering am Beispiel TCP in der Vorlesung Telematik
  - Dynamik und TCP
    - „Conservation of Packets“
    - Aktives Warteschlangenmanagement
  - Evaluierung von TCP
    - Periodisches Modell
    - Detailliertes Paketverlustmodell
  - TCP und Fairness
- Spezielle Protokollprobleme
  - Hochleistungskommunikation
  - Drahtlose Sensor-Aktor-Netze

TM

HLK

DSAN

# Aufgaben

9.1 Welche Aufgaben hat die Transportschicht?

9.2 Zwischen was genau transportiert die Transportschicht Daten?

9.3 Wie werden TCP- bzw. UDP-Anwendungen adressiert?

9.4 Durch welche Eigenschaften zeichnet sich das *User Datagram Protocol* aus und welche Vor- bzw. Nachteile besitzt es?

9.5 Erläutern Sie die Eigenschaften des *Transport Control Protocols*.

9.6 Geben Sie einige Beispiele für Anwendungen, bei denen Sie den Einsatz von UDP bzw. TCP bevorzugen würden und begründen Sie Ihre Wahl.

9.7 Welches Verfahren kommt beim TCP-Verbindungsaufbau zum Einsatz?

9.8 Wozu wird für den Dienst, den TCP zur Verfügung stellt, überhaupt eine Verbindung benötigt?

9.9 Welche Art Quittungsverfahren kommt bei TCP zum Einsatz?

9.10 Worin liegt der Unterschied zwischen Fluss- und Staukontrolle? Welche Mechanismen von TCP spielen in diesem Zusammenhang eine Rolle?

# Literatur


[AbLo03] Abeck, Lockemann, Schiller, Seitz; Verteilte Informationssysteme; Dpunkt-Verlag, 2003

- Kapitel 5

[Char09] J. Charzinski; Traffic, Structure and Locality Characteristic of the Web‘s Most Popular Services‘ Home Pages; Kommunikation in Verteilten Systemen (KiVS) 2009, Kassel, Germany.

[Hals05] F. Halsall; Computer Networking and the Internet; Addison-Wesley, 2005, 5th Edition

[FlFa99] S. Floyd, K. Fall; Promoting the Use of End-to-End Congestion Control in the Internet; IEEE/ACM Transactions on Networking, August 1999

[KuRo07] J. Kurose, K. Ross; Computer Networking; Addison Wesley 2007, 4th Edition

[RFC768] J. Postel; User Datagram Protocol; RFC 768, August 1980

[RFC793] J. Postel (Ed.); Transmission Control Protocol; RFC 793, September 1981

[Stev94] W. R. Stevens; TCP/IP Illustrated, Volume 1; Addison-Wesley Professional, 1994

# Kommunikation und Datenhaltung

## 10. Anwendungssysteme




**Prof. Dr. Martina Zitterbart**
Dipl.-Inform. Martin Röhricht
[zit | roehricht]@tm.uka.de



# Kapitelübersicht

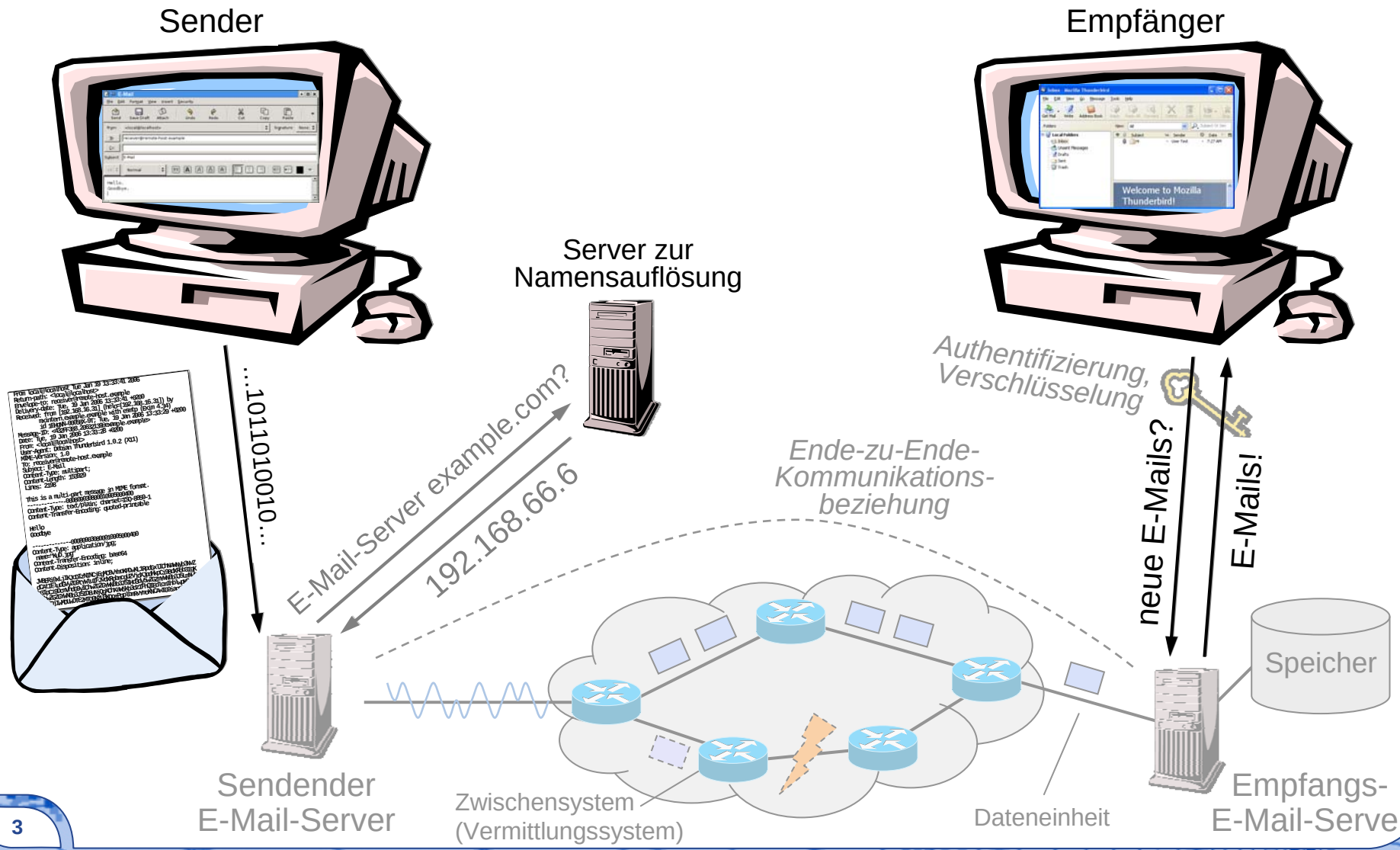
1. Einführung
2. Physikalische Grundlagen
3. Protokollmechanismen
4. Geschichtete Architekturen
5. Sicherungsschicht: HDLC
6. Beschreibungsmethoden
7. Sicherungsschicht: Lokale Netze
8. Netzkopplung und Vermittlung
9. Die Transportschicht
10. Anwendungssysteme
11. Middleware

- 10.1 Das World Wide Web
- 10.2 Namensdienst
- 10.2 Elektronische Post
- 10.3 Dateitransfer im Internet

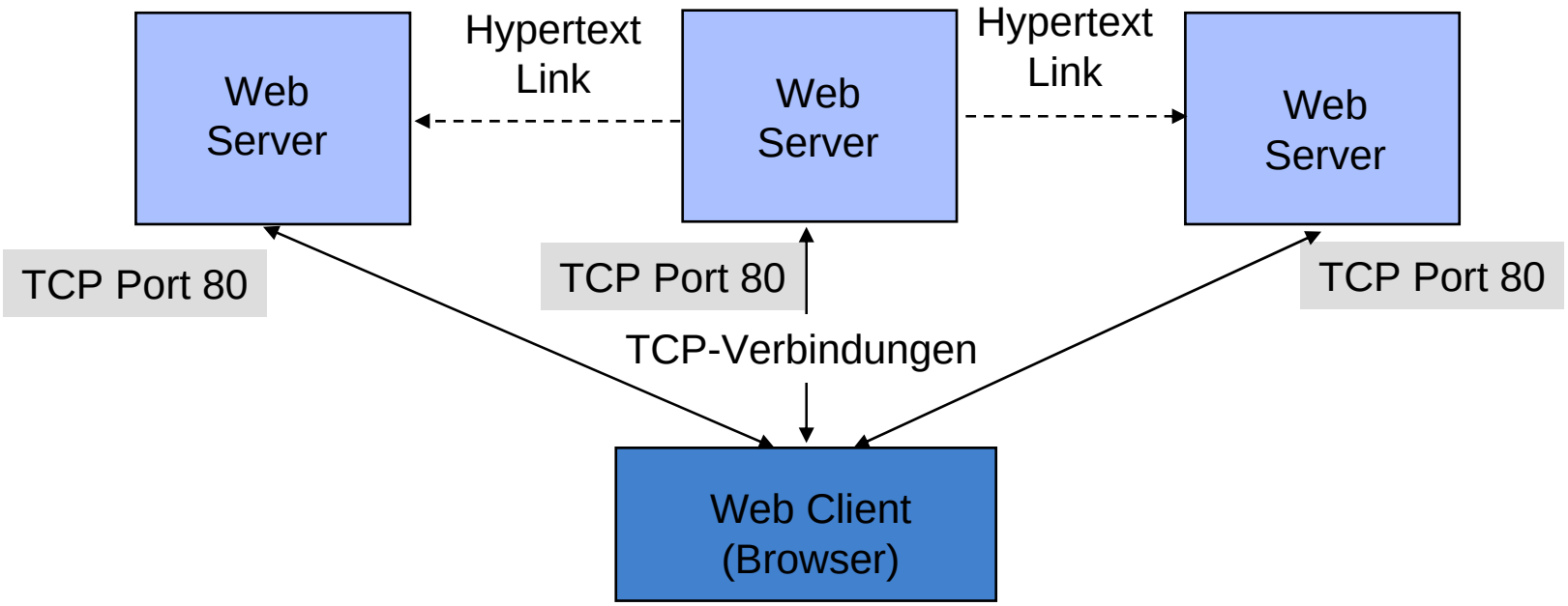
# … endlich Anwendungen

- Die hier behandelten (Standard-)Anwendungen sind allen bekannt
  - Hoffentlich hat sie jeder schon mal benutzt
- Die Anwendungen profitieren von den angebotenen Diensten der darunterliegenden Schichten (z.B. gesicherte Übertragung)
  - Sind daher teilweise relativ einfach aufgebaut
- Allerdings fehlt teilweise noch etwas in der Mitte – die Middleware
- … in diesem Kapitel werden zunächst die Anwendungen vorgestellt
  - … darauf folgend kommt dann die Middleware

# Anwendungen im Beispiel



# 10.1 World Wide Web (WWW)




- Client-Server-Anwendung
- Hypertext
  - Über ein Netz verbundene Objekte
  - Die grundlegende Idee Hypertext ist wesentlich älter als WWW
- Grafischer Browser
  - Nutzung auch durch Nicht-Informatiker
- Die(!) „Killer“-Anwendung im Internet

# Entwicklung des WWW

- Hervorgegangen aus Arbeiten des britischen Informatikers Tim Berners-Lee am europäischen Forschungszentrum CERN (Genf)
  - Ziel: Einfacher weltweiter Austausch von Dokumenten zwischen Wissenschaftlern
- Erster Prototyp Ende 1980
  - Grafisch und zeilenorientiert
- Durchbruch des WWW durch den WWW-Client Mosaic
  - Entwickelt von Marc Andreesen und Eric Bina (University of Illinois)
  - Ursprünglich für X-Windows-Systeme
  - Als Quellcode per FTP kostenlos verfügbar ⇨ schnelle Verbreitung
  - Marc Andreesen und Jim Clark gründeten im April 1994 die Firma Netscape Communication Corporation
  - 1998 wurde Netscape von America Online übernommen
  - Marc Andreesen inzwischen nicht mehr bei Netscape
- Gründung des W3-Konsortiums im Juli 1994
  - Vorrangiges Ziel: Weiterentwicklung des WWW, z.B. durch Standardisierung von HTML
  - Vorsitzender: Tim Berners-Lee
  - Infos unter http://www.w3.org



Berners-Lee



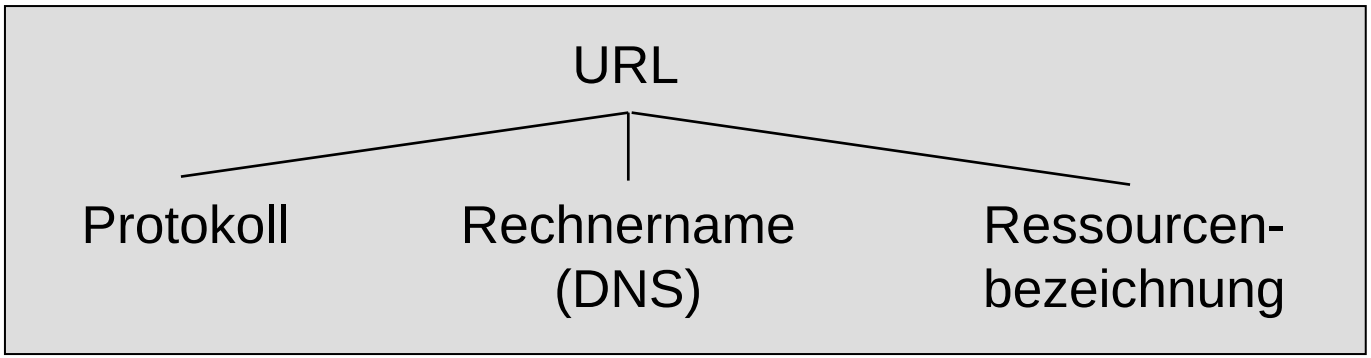
Marc Andreesen

# Eindeutige Adressierung eines Web-Dokuments

- WWW nutzt hierzu den Mechanismus des Uniform Resource Locator / Identifier (URL / URI)


[RFC3986]

  - Repräsentiert eine Ressource und weist der Client-Software den „Weg“ zu ihr
  - Auch für Inhalte anderer Server (USENET, FTP, E-Mail) verwendbar




- Ressourcenbezeichnung identifiziert das Objekt, auf das im jeweiligen Server zugegriffen werden soll, identifiziert
  - Bei WWW: aufgerufene Web-Seite
  - Bei FTP: zu übertragende Datei
  - Bei Mail: Empfänger der Mail (also die Mail-Adresse)

# Komponenten des WWW

- Namen/Namensdienst
  - Universal Resource Locator (URL)
    - kompakte Repräsentation der Lokation und Zugriffsmethode auf Internet-Ressourcen
      - <scheme>:<scheme-specific-part>
      - ftp://<user>:<password>@<host>:<cwd1>..<cwdN>/<name>; type=<typecode>
      - http://<host>:<port>/<path>?<searchpart>
      - mailto:<rfc822-addr-spec>
      - nntp://<host>:<port>/<newsgroup-name>/<article-number>
      - telnet://<user>:<password>@<host>:<port>
      - file://<host>/path
- Hypertext
  - Hypertext Markup Language (HTML) zur Beschreibung von WWW-Seiten
    - Strukturierung der Dokumente in Kopf (*Header*) und Rumpf (*Body*)
    - Tags zur Markierung von Formatierungsanweisungen
      - `<b>Hallo</b>` ⇨ **Hallo**
      - `<i>Hallo</i>` ⇨ *Hallo*
- Austausch von HTML-Seiten
  - Hypertext Transfer Protocol (HTTP)
    - Zustandsloses Protokoll (HTTP-Server hält keine Information über Clients)
    - Oberhalb TCP angeordnet

# 10.1.1 HTTP

- Einfaches ASCII-basiertes Transferprotokoll



[RFC1945]

  - Zwei Typen von Nachichten: Request und Response
  - Zustandslos: Jeder Request wird individuell bearbeitet
  - Setzt auf TCP-Verbindungen auf
    - kurzlebig: HTTP-Server schließt Verbindung nach Beantwortung der Anfrage
  - Default-Port: 80
- Beispiele von Requests
  - GET: Aufruf eines Objekts (mit und ohne Protokollversion)
  - HEAD: Aufruf des Kopfs einer WWW-Seite
  - PUT: Speichern einer WWW-Seite
  - POST: Übermittlung neuer Informationen; z.B. durch interaktive Benutzer ausgefüllter Inhalt von Formularen
  - DELETE: Löschen einer WWW-Seite

# Nicht-persistente Verbindungen

- Ablaufschritte
  - „Anklicken" des Objekts
  - Browser ermittelt URL
  - Browser fragt DNS (Domain Name System) nach IP-Adresse
  - DNS antwortet mit IP-Adresse
  - Browser etabliert TCP-Verbindung zu Port 80 dieser IP-Adresse
  - Browser schickt GET /...
  - Server sendet gewünschtes File
  - TCP-Verbindung wird geschlossen
  - Browser zeigt Text des Files an
  - Browser zeigt Bilder des Files an
- Nicht-persistent
  - TCP-Verbindung wird nach Senden des Objekts geschlossen
  - Pro TCP-Verbindung genau ein Request-Response-Paar
  - Parallele TCP-Verbindungen möglich (ca. 5-10 bei derzeitigen Browsern)
  - Probleme: Hoher Ressourcenverbrauch im Server, Slow-Start, RTT

# Persistente Verbindungen & Pipelining

- Eigenschaft
  - Mehrere Objekte und WWW-Seiten werden über die gleiche TCP-Verbindung ausgeliefert
- Ohne Pipelining
  - Request wird erst nach Erhalt der vorangegangenen Response versendet
  - Verzögerung im Rahmen einer RTT pro Objekt
- Mit Pipelining
  - Requests können direkt hintereinander gesendet werden
  - Reduziert durch Slow-Start und RTT bedingte Verzögerung
  - Default-Modus bei HTTP/1.1



[RFC2616]

# 10.2 Namensdienst

- Namen
  - Verwendung logischer Namen (z.B. www.tm.uka.de)
  - ⇨ Abbildung logischer Namen auf Referenzen (Binding)
    - z.B. Abbildung auf IP-Adresse (DNS)
    - z.B. Abbildung auf interne Objektidentifikatoren
      - (Beispiele im folgenden Kapitel zum Thema Middleware)
    - z.B. Abbildung von Dateiname auf Spur-/Sektor-/Blockadressen
  - ⇨ Analogie zum Telefonbuch: „White Pages“
- Verwaltung von Zuordnungen (Bindings)
  - Entgegennahme neuer Bindings
  - Auffinden der zu einem Objektnamen gehörenden Referenz
  - Löschen von Bindings
  - Erzeugen und Löschen von Namenskontexten
    - Behälter für Bindings

# Namensdienst

- Eigenschaften der Abbildungen
  - Namen können hierarchisch aufgebaut sein
    - Sequenz von Bindungen
  - Bindung ist in seinem Kontext eindeutig
  - Bindung führt zu genau einer Referenz
  - mehrere Bindungen können auf dieselbe Referenz verweisen

- Probleme
  - Ausfall des Namensdienstes
    - *Single point of failure*
      - ⇨ Verteilte bzw. redundant aufgebaute Namensdienste
    - Skalierbarkeit in großen verteilten Systemen
      - ⇨ Verteilte bzw. redundant aufgebaute Namensdienste
    - Wie findet der Dienstnehmer den Namensdienst?
      - Feste Adresse
      - Im Zuge der dynamischen Netzwerkkonfiguration
      - (Mit Hilfe der Middleware, siehe folgendes Kapitel)

# Beispiel: Domain Name System (DNS)

- Problem
  - Internet-Systeme können auf verschiedene Arten identifiziert werden
    - Name oder IP-Adresse
  - Von den menschlichen Benutzern werden Namen bevorzugt
  - Abbildung von Name auf IP-Adresse erforderlich

www.ietf.org

spiegel.de

tm.uka.de

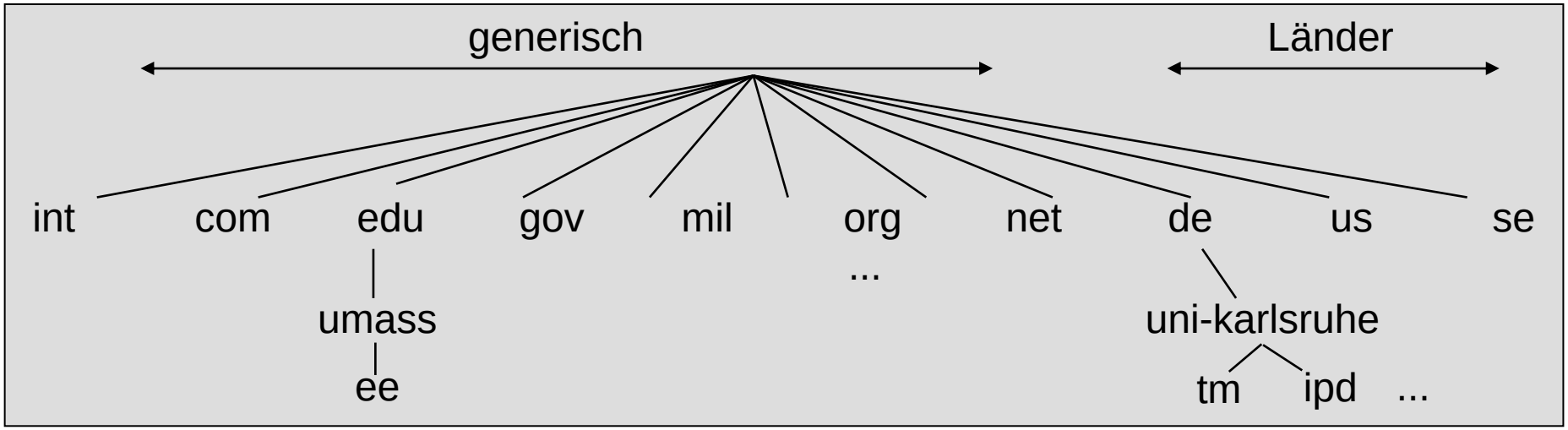


[RFC1034, 10.5]

- Domain Name System
  - Verteilte Datenbank mit einer Hierarchie von Name-Servern (DNS-Servern)
  - Protokoll der Anwendungsschicht, das Kommunikation mit Name-Servern regelt
    - Über UDP realisiert – Port 53
    - Wieso nicht TCP?
  - Arbeitet nach dem Client/Server-Modell
    - Server kann Anfrage an andere Server weiterleiten
  - „Kernfunktion“ im Internet – keine eigentliche Anwendung
    - Komplexität ist am Rande des Netzes lokalisiert ⇨ Internet Design Philosophie!
- Verwendung
  - Wird von anderen Protokollen der Anwendungsschicht eingesetzt: HTTP, SMTP, FTP
- Anfrage nach IP-Adressen, Mail-Server, ...
  - Unter UNIX mit `gethostbyname()`

# Struktur des Internet-Namensraums

- Struktur
  - Domänen, Sub-Domänen usw.
  - Oberste Ebene: Top-Level-Domain (TLD)
    - Generische Domänen
      - .aero, .biz, .cat, .com, .coop, .edu, .gov, .info, .jobs, .mobi, .int, .mil, .museum, .name, .net, .org, .pro, .travel
    - Länder (z.B. .de, .us, .uk,…)
- Repräsentation als Baum




  - Blätter repräsentieren Domänen ohne Sub-Domänen, aber mit IP-Equipment
    - Repräsentiert durch absoluten Namen, z.B. tm.uni-karlsruhe.de

# Zur Info: Zuordnung der Domänen

- com
  - Kommerzielle Unternehmen
- edu
  - Lehranstalten, Universitäten
- gov
  - Regierungsbehörden
- mil
  - Militärische Einheiten
- net
  - Netzbetreiber und -anbieter
- org
  - Organisationen, die nicht in die obigen Kategorien fallen
- arpa
  - Temporäre Arpa-Domäne, die immer noch benutzt wird
- int
  - Internationale Organisationen
- Kurzbezeichnungen der Länder (de, ch, uk, etc.)

# Weitere Dienste von DNS

- Host Alias
  - Endsysteme mit komplizierten Namen können über einen oder mehrere Alias-Namen verfügen
    - Alias-Namen sind typischerweise einfacher zu merken
    - DNS kann für einen Alias-Namen den „eigentlichen“ Namen liefern (kanonischer Name)
- Mail Server Alias
  - E-Mail-Adressen sollten so gestaltet sein, dass sie leicht zu merken sind, z.B. zit@tm.uka.de
  - Aber in vielen Fällen ist der Host-Name des Mail-Servers komplexer gestaltet, z.B. für zit@tm.uka.de der Mail-Server iramx2.ira.uni-karlsruhe.de
- Lastverteilung
  - Bei redundant ausgelegten Servern (z.B. Server-Cluster) kann eine Menge von IP-Adressen mit einem Namen assoziiert sein
    - DNS-Server antwortet mit gesamter Menge der zugehörigen IP-Adressen, verändert aber die Reihenfolge, in der diese genannt werden

# Weshalb verteilte DNS-Server?

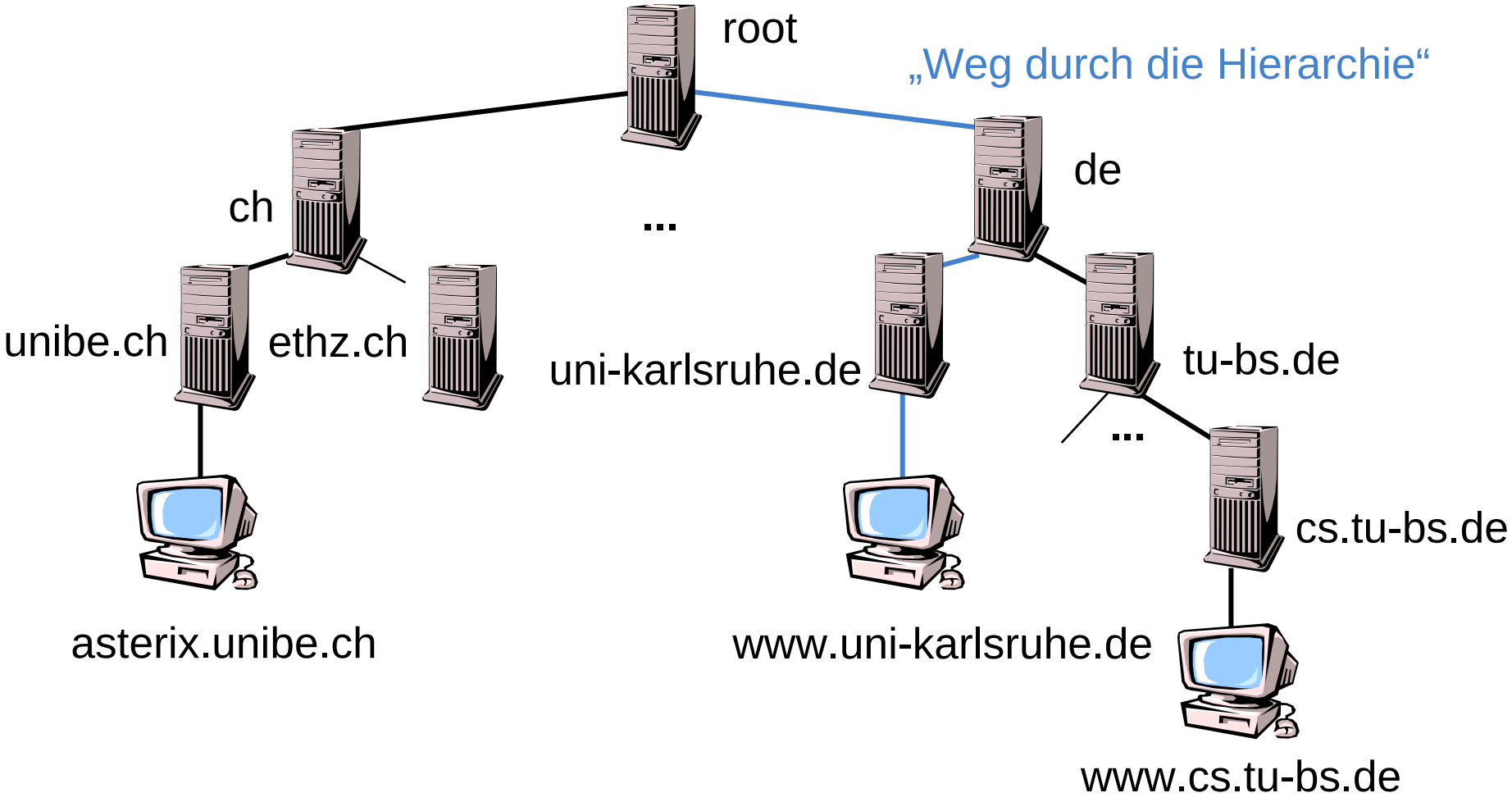
- Probleme bei Zentralisierung
  - Singuläre Fehlerquelle
  - Verkehrsaufkommen an zentralem (weltweit!) Server
  - Entfernung der zentralen Datenbank
  - Verwaltung

  ⇨ Ein zentraler Ansatz skaliert hier nicht!
- Verteilung von DNS-Servern
  - Kein Server kennt alle Abbildungen von Namen auf IP-Adressen
  - Lokale DNS-Server
    - I.A. hat jeder Netz-Provider, jede Firma einen lokalen DNS-Server
      - Am Institut für Telematik (DNS der ATIS): iraun1.ira.uka.de (129.13.10.90)
    - Erste Nachfrage geht immer zu diesem lokalen Server (Default-Server)
  - Autoritativer DNS-Server
    - Enthält autoritative Abbildungen
      - Liefert maßgebliche Antworten

# Name-Server

- Typen von Name-Servern hierarchisch gegliedert
  - Root-Server
    - Weltweit einige wenige solcher Server (bzw. Server-Cluster)
  - Top-Level Domain (TLD) Server
    - Verantwortlicher Server für die Top-Level Domains
  - Autoritative Name-Server
    - Jeder Host ist bei einem solchen Server registriert
    - Häufig gleichzeitig lokaler Name-Server in „seinem“ Netz
  - Lokale Name-Server
    - z.B. Provider, Universität, Firma etc. betreiben lokalen Name-Server
- Für jede Hierarchie-Ebene (= Domäne) gilt
  - Die Adresse eines Root-Servers ist bekannt
  - Kennt Name-Server, die für die nächst tiefere Ebene zuständig sind
  - Jeder Nameserver enthält die Einträge, für die er verantwortlich ist (authoritative)
    - Besitzt Autorität zur Namensvergabe innerhalb dieser Domäne
  - Darüber hinaus werden Ergebnisse von vorhergehenden Anfragen zwischengespeichert
    - Timeout legt fest wie lange Einträge zwischengespeichert werden
      - Typischerweise mehrere Tage

# DNS-Hierarchie: Beispiel



# DNS Root-Server

- Lokation der DNS Root-Server
  - http://www.root-servers.org
- Aufgaben
  - Kontaktiert autoritative Name-Server, falls Abbildung unbekannt
  - Gibt Abbildung dem lokalen Name-Server bekannt
- Weltweit gibt es 13 Root-Server
- Root-Server enthalten nur Einträge für TLDs
  - Generische und Länder-TLDs




[KuRo07]

# Verteilter Betrieb von Root-Servern

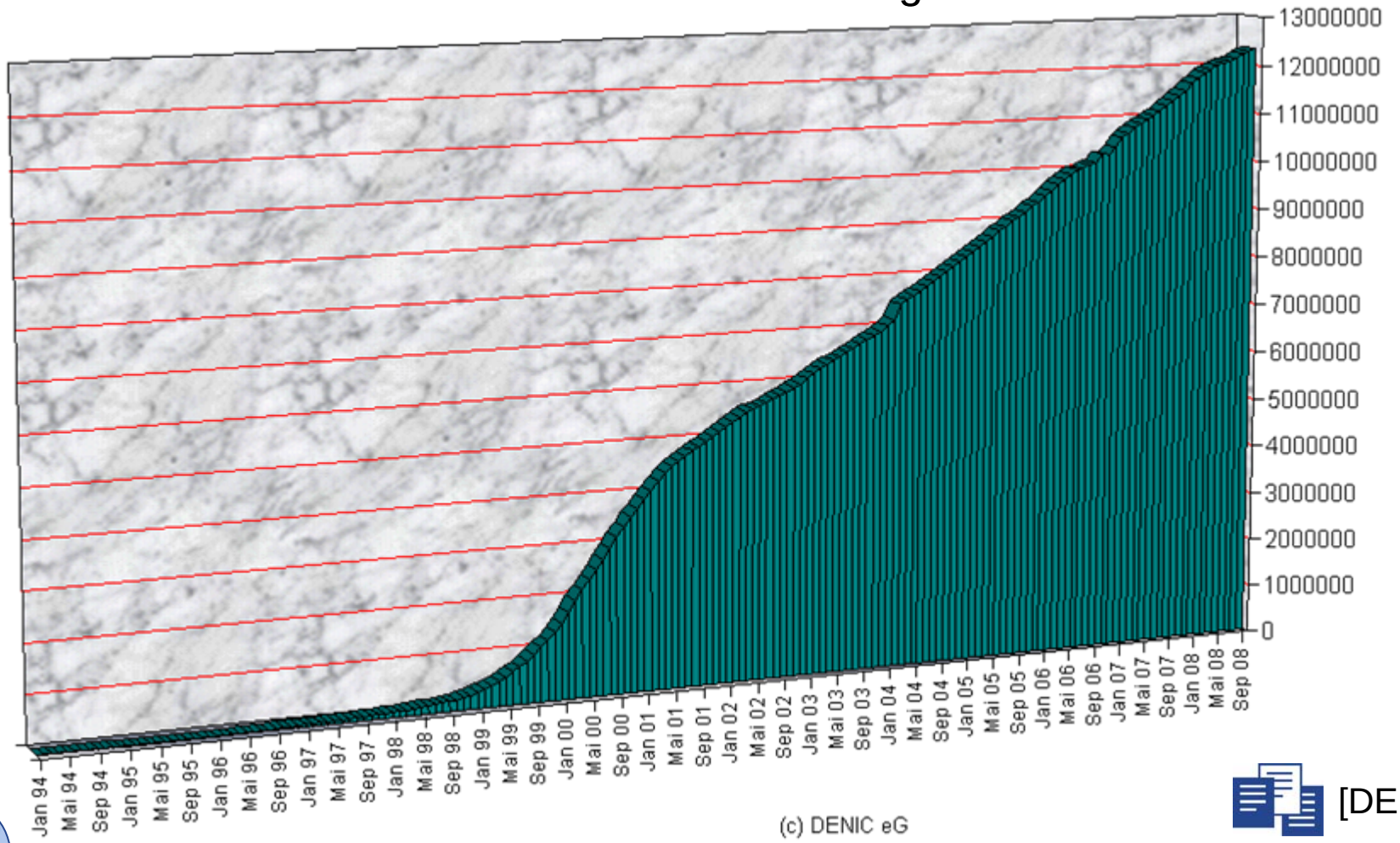
- Root-Server können verteilt betrieben werden
  - z.B. K-Root-Server Mirror Instances




[RIPE09]

# TLD-Name-Server für .de: DENIC

- Zeitlicher Verlauf der Anzahl an registrierten .de-Domains
  - 14.04.2008 12-millionste .de-Domain registriert

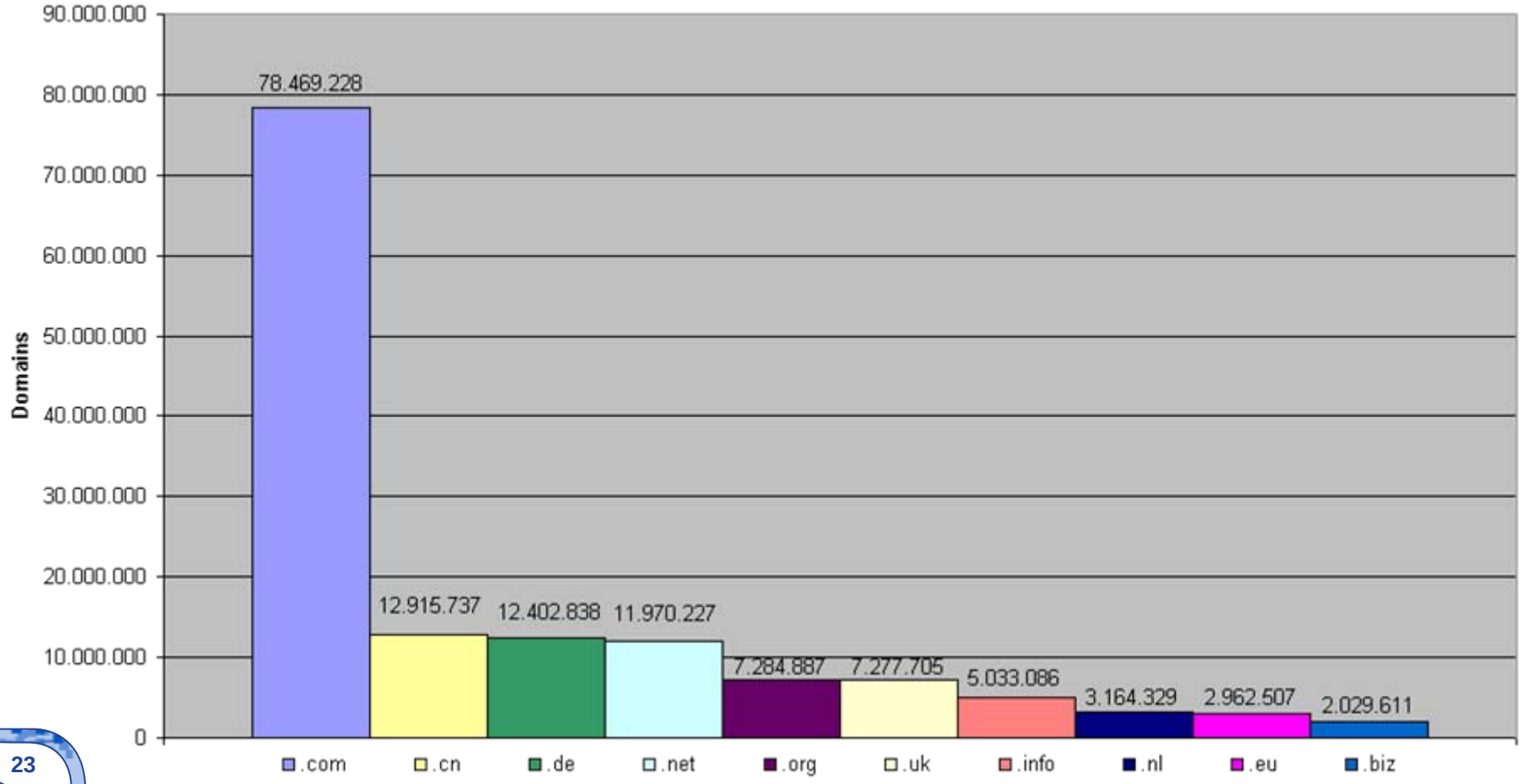



[DENI09]

# Die größten TLDs

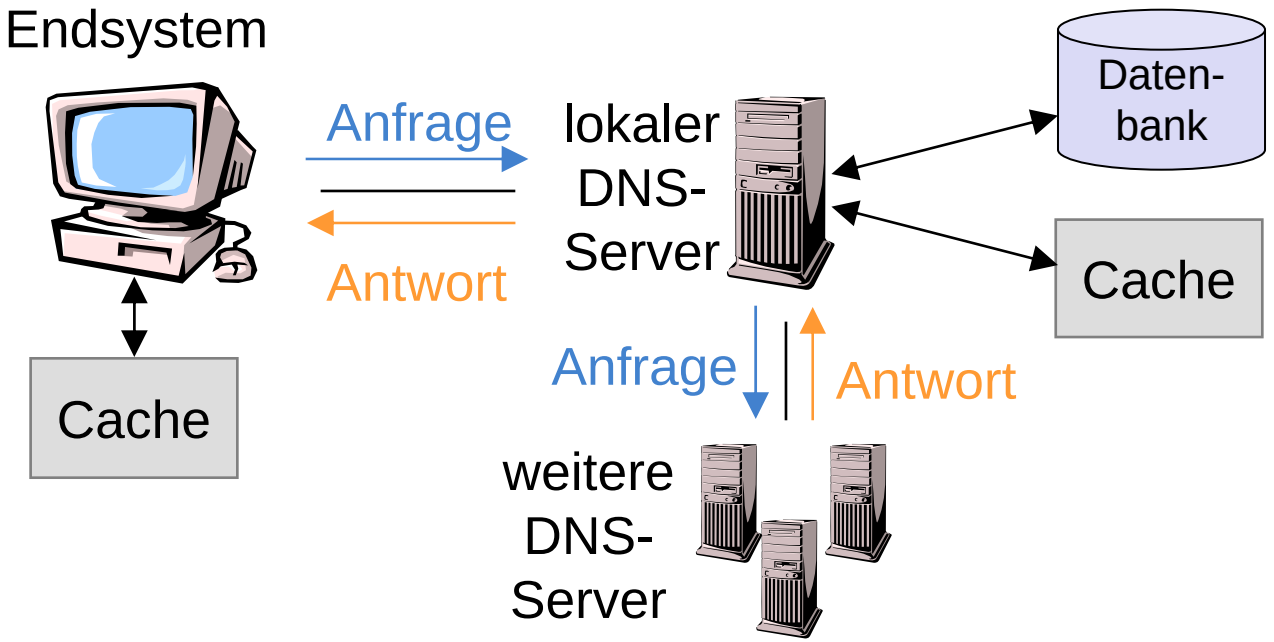
- Die Top Level Domains mit den meisten Einträgen (November 2008)

[DENI09]



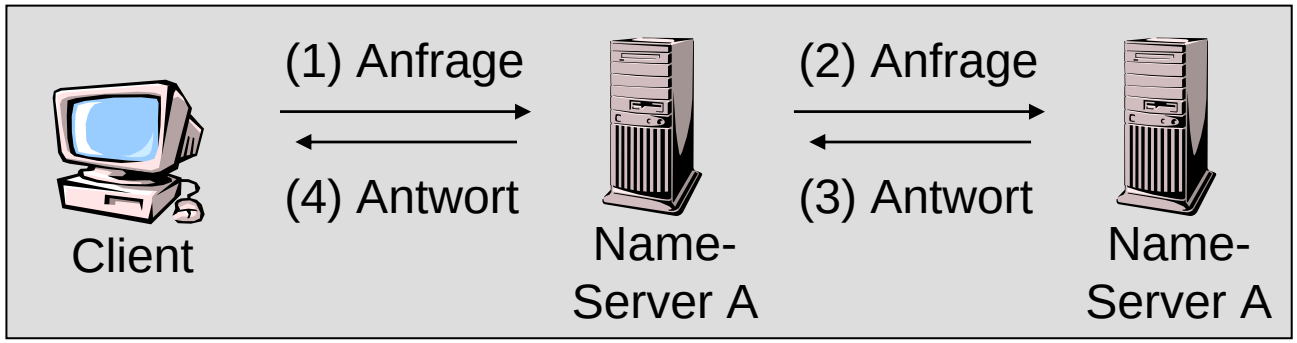
# 10.2.1 DNS-Abfrage

- Schema einer Abfrage
  - Hat Endsystem Antwort nicht im eigenen Cache, wird lokaler DNS-Server befragt
  - Lokaler DNS-Server antwortet entweder
    - aus seiner eigenen Datenbank mit Einträgen (falls autorativ)
    - aus seinem Cache gespeicherter Antworten auf vorangegangene Anfragen
    - nach Befragung anderer DNS-Server, falls er die Antwort nicht liefern kann



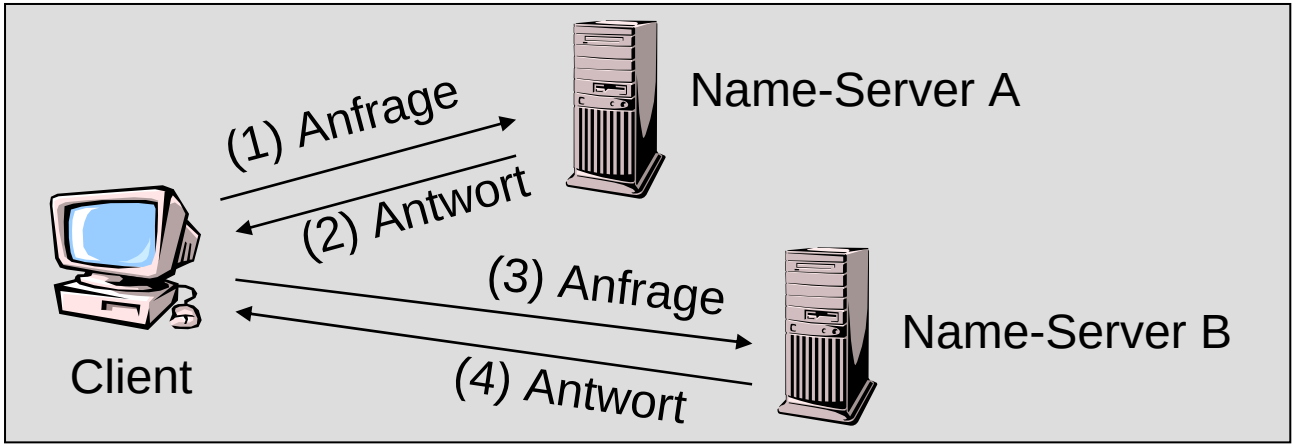
# Typen von DNS-Anfragen

- Rekursive Anfrage
  - Kennt angefragter Server die Antwort nicht, befragt er weitere Server, bis er die Antwort zurückliefern kann
  - Vorteil: Geringer Aufwand für Client, da Arbeit zur vollständigen Namensauflösung dem Server überlassen wird



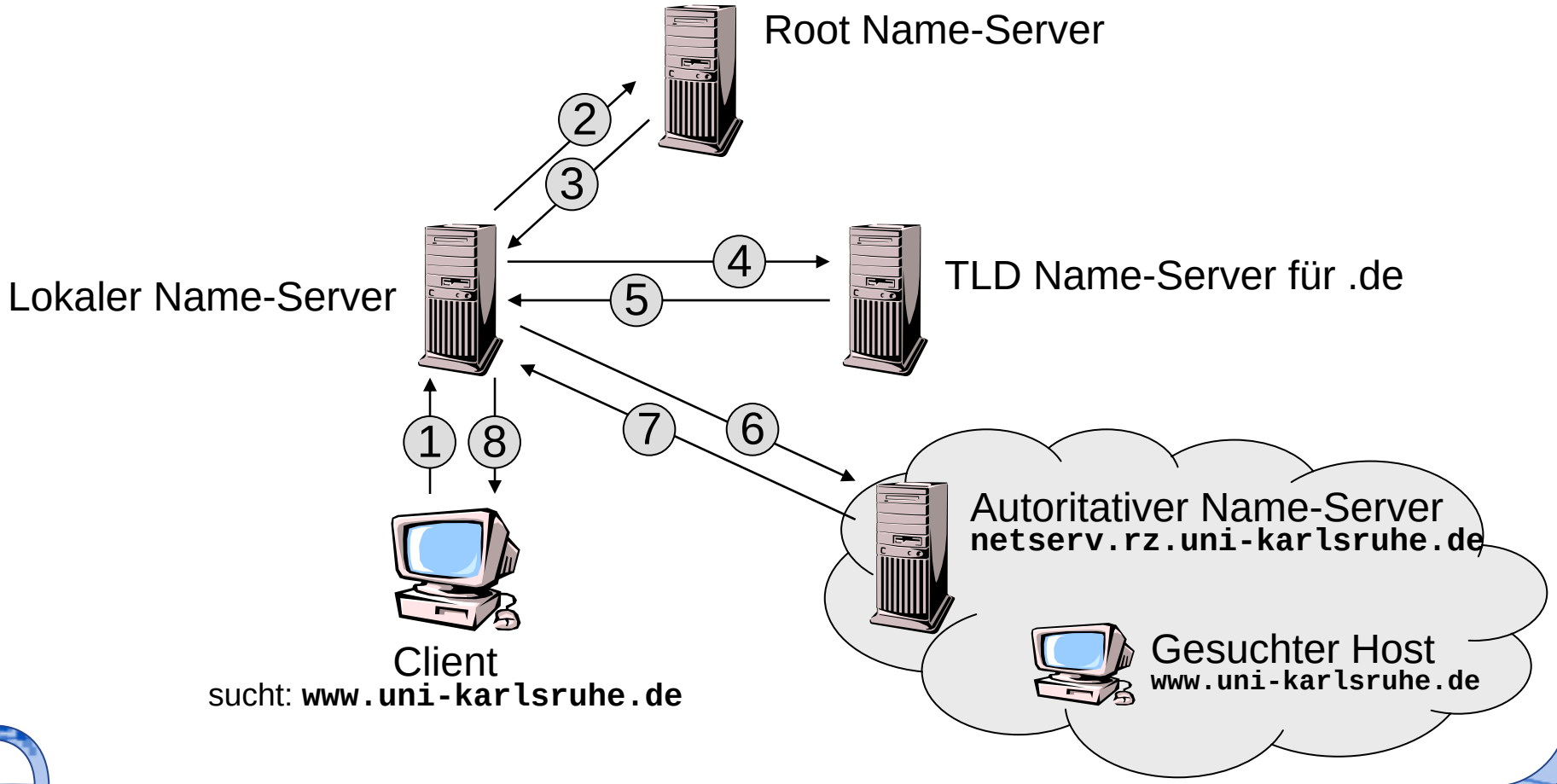
# Typen von DNS-Anfragen

- Iterative Anfrage
  - Kennt angefragter Server die Antwort nicht, kann er mit Verweis auf anderen Server antworten
  - Client befragt dann weitere Server selbst
  - Vorteil: Weniger Aufwand für angefragten Name-Server



# Beispiel

- Typischerweise fragt Client seinen lokalen Name-Server rekursiv
  - Lokaler Name-Server arbeitet dann normalerweise iterativ



# 10.2.2 DNS-Datenbankeinträge

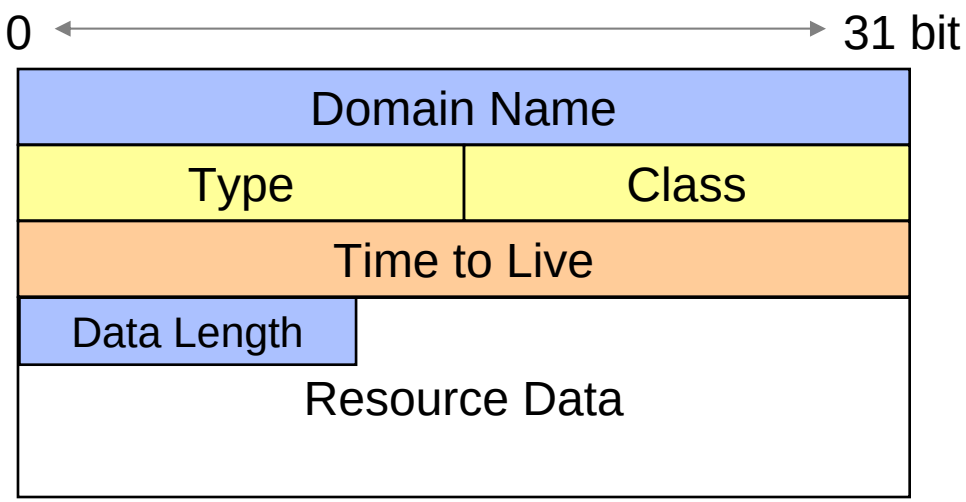
- Resource Records (RR)
  - In der Antwort werden ein oder mehrere solche Resource Records mitgeführt
  - Tupel mit (Name, Wert, Typ, TTL)
    - Typ=A bzw. Typ=AAAA
      - Name=Hostname, Value=IPv4-, bzw. IPv6-Adresse für Hostname
    - Typ=NS
      - Name=Domain, Value=IP-Adresse des autorativen Servers für diese Domäne
        - Dieser Server weiß, wie IP-Adressen für Systeme dieser Domäne bestimmt werden können
    - Typ=CNAME
      - Name=Alias hostname, Value=Canonical hostname
        - Hiermit kann kanonischer Name bestimmt werden
    - Typ=MX
      - Name=Alias hostname, Value=Hostname eines E-Mail-Servers

# Resource Records

- Varianten von Resource Records

| Typ | Beschreibung |
|---|---|
| A bzw. AAAA (Address) | Abbildung Name auf IPv4/IPv6-Adresse |
| MX (Mail Exchange) | E-Mail-Server einer Domäne |
| NS (Nameserver) | Nameserver einer Domäne |
| CNAME (Canonical Name) | „Alias“-Namen für Rechner/Domänen |
| PTR (Pointer) | Abbildung IP-Adresse auf Name |
| HINFO (Host Info) | Zusätzliche Informationen (CPU, …) |

- Format



# IPv6 für DNS-Rootserver

- Seit Februar 2008 AAAA-Records für DNS-Rootserver
  - Namensauflösung von IPv6-Adressen in lesbare Namen und umgekehrt
  - Unterstützung in sechs der insgesamt 13 Rootserver



[Heis08, RFC3596]

- Zuvor keine ausschließlichen IPv6-basierten DNS-Abfragen auf Root-Ebene möglich
- Viele *Toplevel-Domains* bereits mit IPv6-fähigen Nameservern ausgestattet
  - Z.B. DeNIC für TLD „.de“ seit 2004
- Seit 5. Juni 2008 fünf weitere k.root-servers.net Server IPv6-fähig
  - Budapest, Mailand, Helsinki, Genf, Athen
  - Somit Gesamtzahl derzeit sieben

# Beispiel: DNS-Anfrage nach AAAA-Records

```
> dig www.tm.uka.de AAAA

; <<>> DiG 9.4.1-P1 <<>> www.tm.uka.de AAAA
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 12076
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 3, AUTHORITY: 4, ADDITIONAL: 4

;; QUESTION SECTION:
;www.tm.uka.de.                 IN      AAAA

;; ANSWER SECTION:
tm.uka.de.              86400   IN      DNAME   tm.uni-karlsruhe.de.
www.tm.uka.de.          0       IN      CNAME   www.tm.uni-karlsruhe.de.
www.tm.uni-karlsruhe.de. 86400  IN      AAAA    2001:638:204::42

;; AUTHORITY SECTION:
tm.uni-karlsruhe.de.    86400   IN      NS      deneb.dfn.de.
tm.uni-karlsruhe.de.    86400   IN      NS      iraun1.ira.uni-karlsruhe.de.
tm.uni-karlsruhe.de.    86400   IN      NS      iraun2.ira.uni-karlsruhe.de.
tm.uni-karlsruhe.de.    86400   IN      NS      netserv.rz.uni-karlsruhe.de.

;; ADDITIONAL SECTION:
deneb.dfn.de.           22190   IN      A       192.76.176.9
iraun1.ira.uni-karlsruhe.de. 13885 IN   A       141.3.10.90
iraun2.ira.uni-karlsruhe.de. 13885 IN   A       141.3.10.91
netserv.rz.uni-karlsruhe.de. 13887 IN   A       129.13.64.5

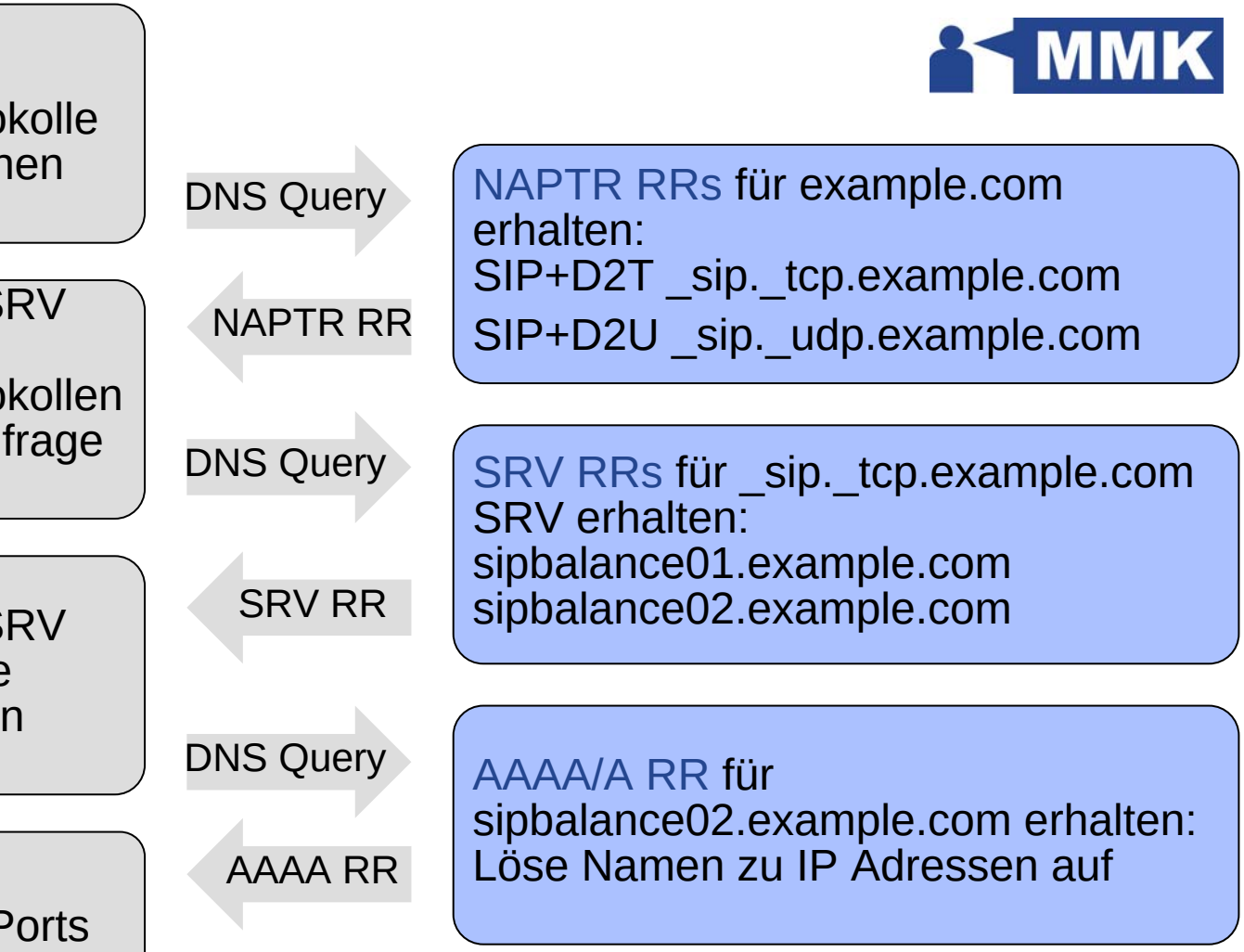
;; Query time: 2 msec
;; SERVER: 2001:638:204::11#53(2001:638:204::11)
;; WHEN: Thu Mar 27 06:20:31 2008
;; MSG SIZE  rcvd: 269
```

DNS-Anfrage via IPv6!

# Namensauflösung bei VoIP per SIP

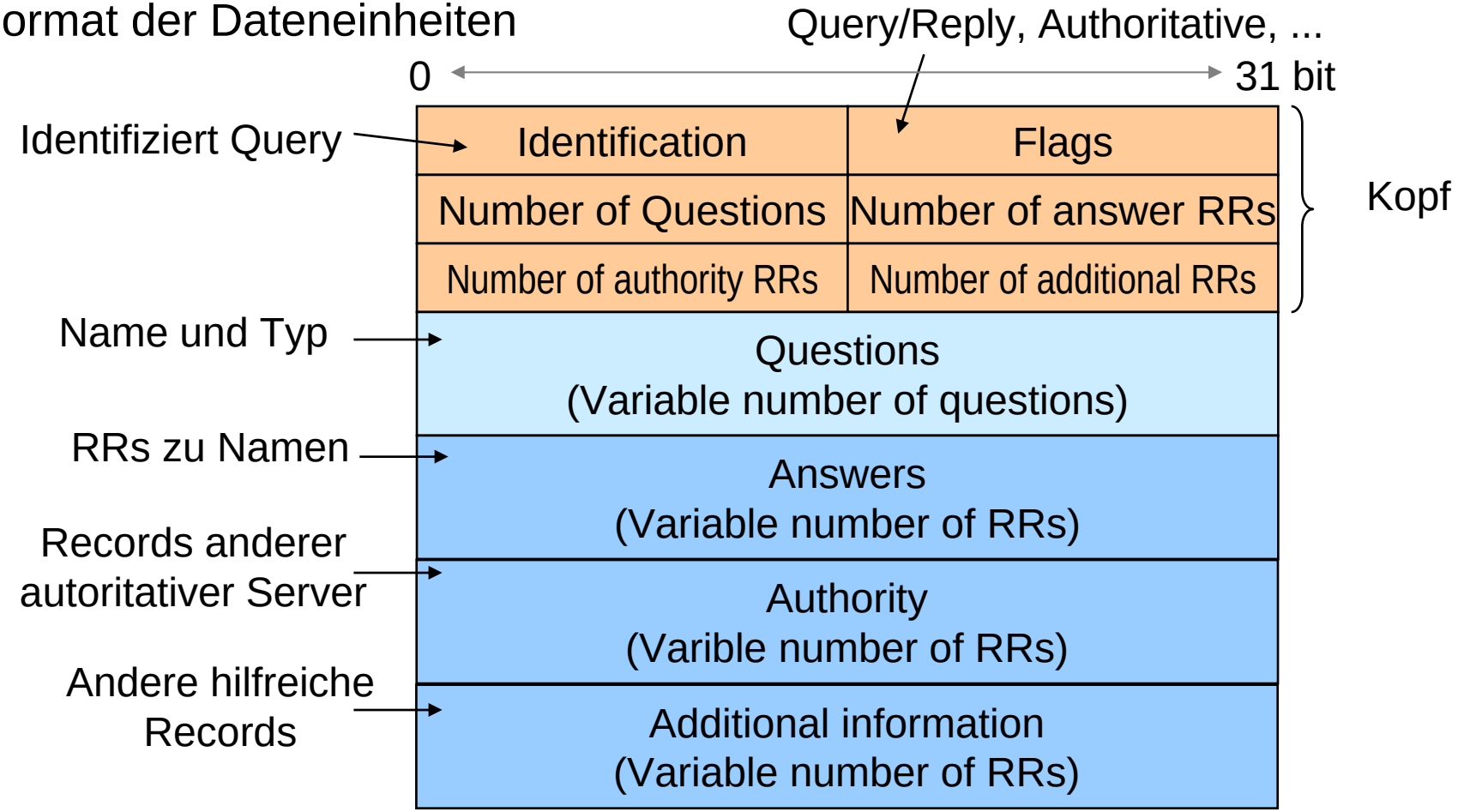
- Lokalisierung des SIP-Servers nur aus Domänennamen (sip:bless@tm.uka.de) nicht einfach
- Problem während Sitzungsaufbaus:
  - Ermitteln von IP Adresse, Port and Transport Protokoll des nächsten Hops
- SIP kann verschiedene Transportprotokolle nutzen
  - TCP, UDP, SCTP, TLS
- Endsysteme müssen herausfinden, welche Transportprotokolle zur Verfügung stehen
- Endsystem kann anderen Server probieren, falls einer ausfällt
- DDDS/NAPTR ermöglicht grobgranulares kapazitätsbasiertes „Load Balancing“

# Namensauflösung bei VoIP per SIP

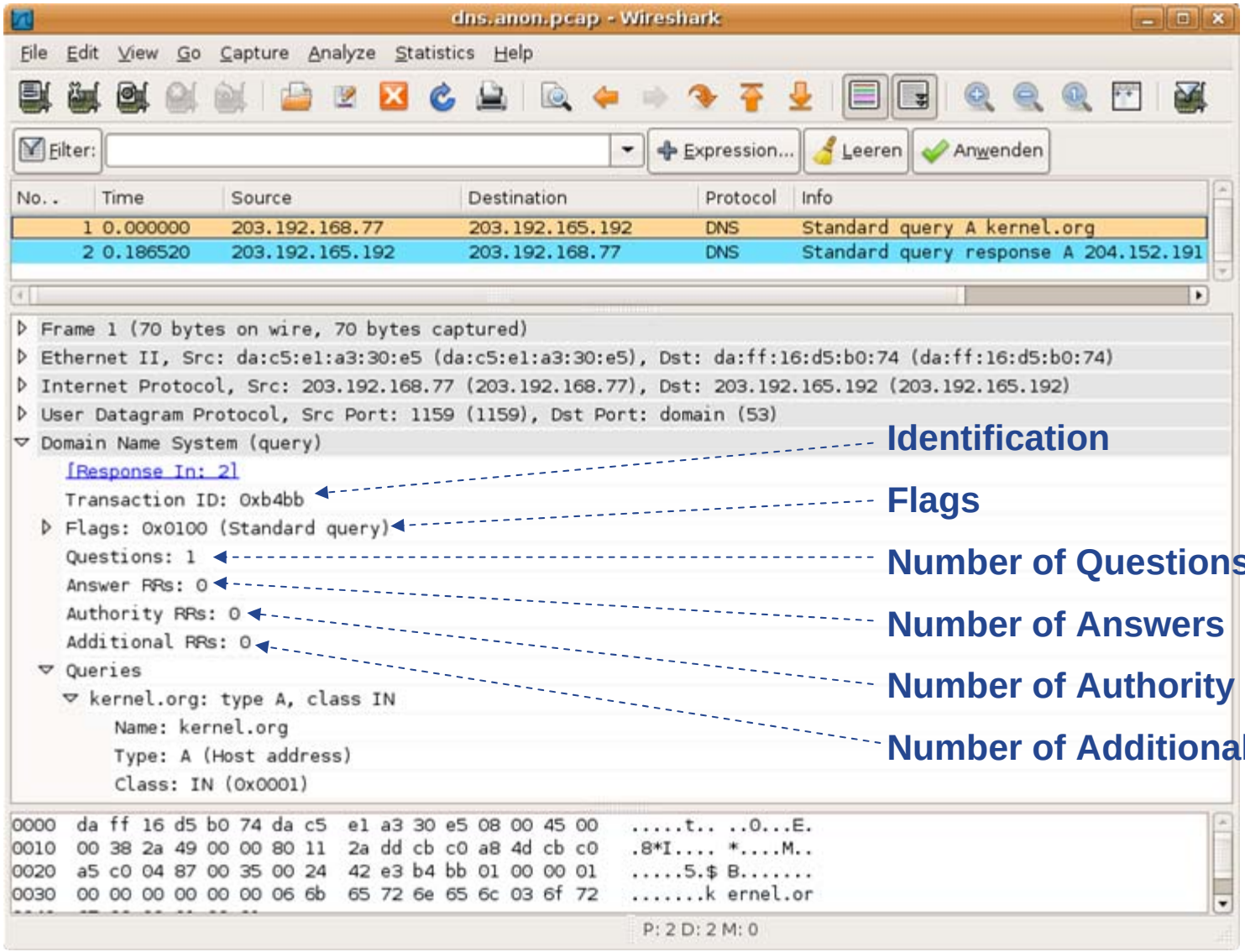


# 10.2.3 DNS-Dateneinheiten

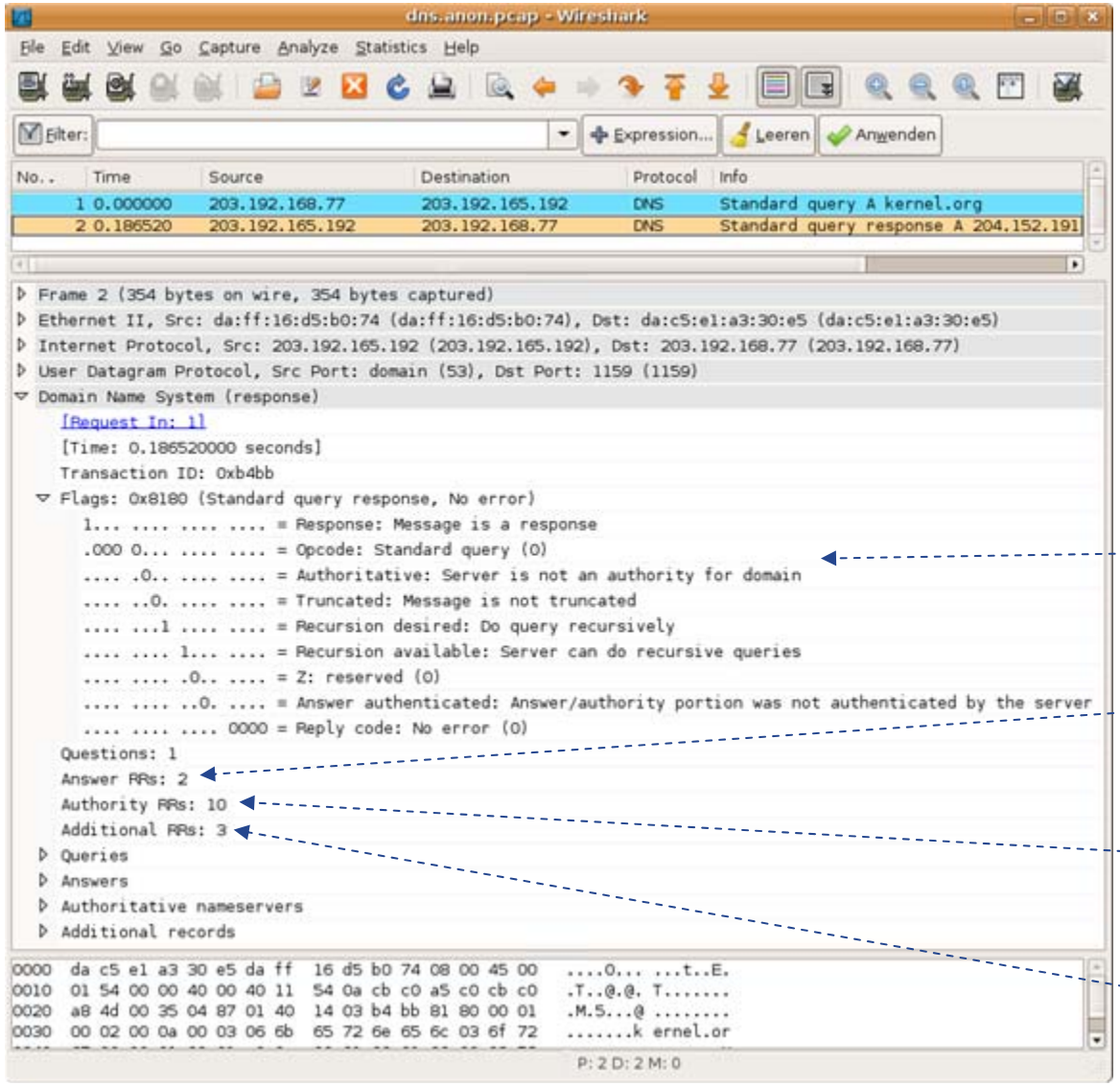
- Typen von Dateneinheiten
  - Query
  - Reply
- Format der Dateneinheiten



# DNS-Query – Wireshark



# DNS-Reply – Wireshark




**Flags im Reply aufgeschlüsselt**

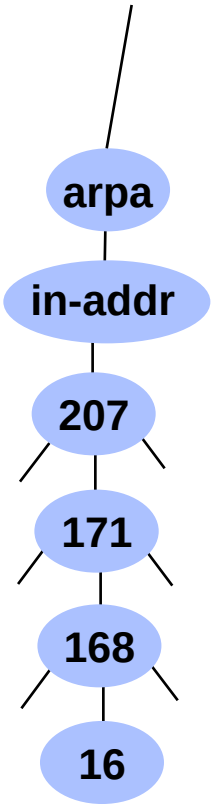
**2 RRs zu Namen (Answers)**

**10 RRs anderer autoritativer Server**

**3 zusätzliche Angaben**

# 10.2.4 Reverse Lookup

- Aufgabe
  - Bestimmung des logischen Namens zu gegebener IP-Adresse
- Vorgehensweise
  - Gesuchte IP-Adresse wird in Zeichenkette umgeformt
  - Hierzu: Spezieller Teilbaum des DNS „in-addr.arpa“
    - Dient der Zuordnung von IP-Adresse ⇨ Name
    - Jede IP-Adresse entspricht Eintrag unterhalb in-addr.arpa
  - IP-Adresse wird dabei invertiert (siehe Beispiel)
- Beispiel
  - IP-Adresse: 207.171.168.16
    - DNS-Name in Anfrage: 16.168.171.207.in-addr.arpa
    - Ergebnis (Antwort RR): Resource Data = www.amazon.de



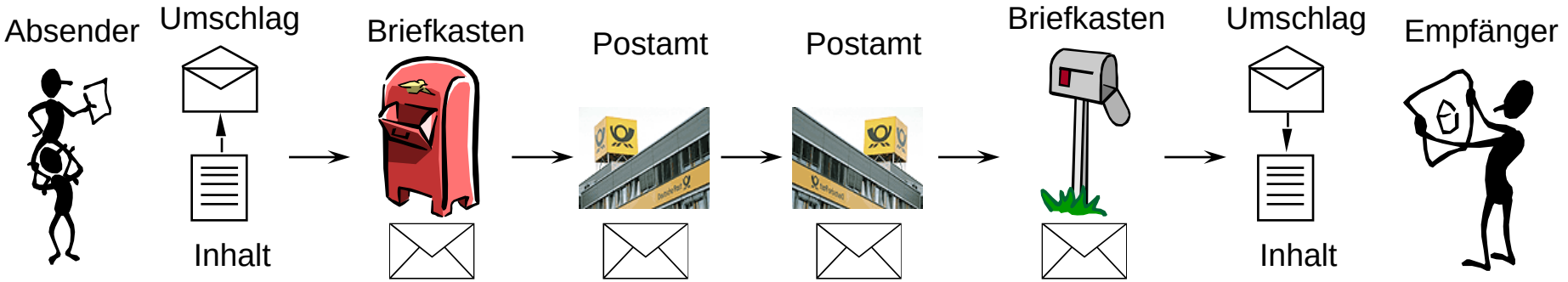
# 10.2.5 DNS-Beispiel: Akamai

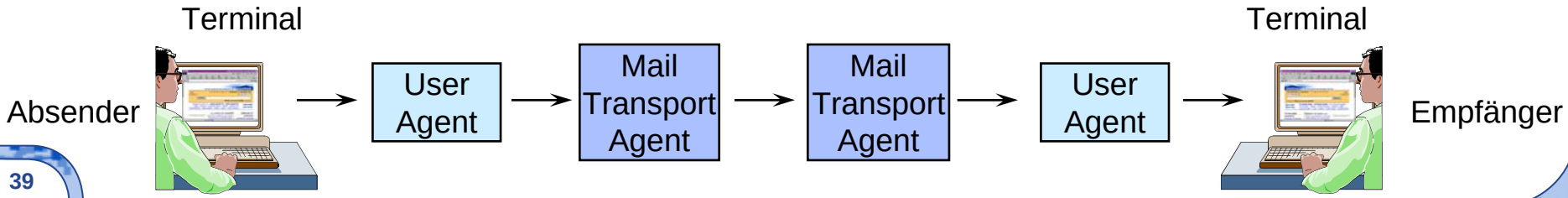
- Ziel
  - Schnellere Auslieferung von Webseiten
  - Auslieferung des Contents durch Proxys
    - sog. Akamai-Server „nahe“ beim User (ISP)
- Anpassung der Webseiten notwendig (Preprocessing)
  - Eingebettete URLs werden zu ARLs (URLs die auf Akamai zeigen)
- Proxy-Auswahl
  - User Lokation + Proxy, der Objekte vorhält
    - nicht alle Objekte liegen auf allen Proxys
- Zweistufiger Abbildungsprozess der ARLs
  - HLDNS (High Level DNS, TTL 30 min)
  - LLDNS (Low Level DNS, TTL 30 sec)
  - Berücksichtigt Zustand des Internets, User-Anforderungen, Verfügbarkeit der Proxys
- Text wird von Originalseite geladen, Objekte von Akamai-Servern
- Probleme
  - Aufwendigere Namensauflösung ⇨ Erhöhte DNS-Verzögerung/ -Verkehr
  - Nachbarschaftsbestimmung ungenau
    - Routing-technisch nicht klar, welcher Proxy am „nächsten“ zur IP-Adresse des Users liegt
      - Welche Metrik bestimmt Nachbarschaft?
        - Anzahl Hops? Auslastung? Geschwindigkeit? ⇨ Keine triviale Frage

[Akam09]

# 10.3 Elektronische Post

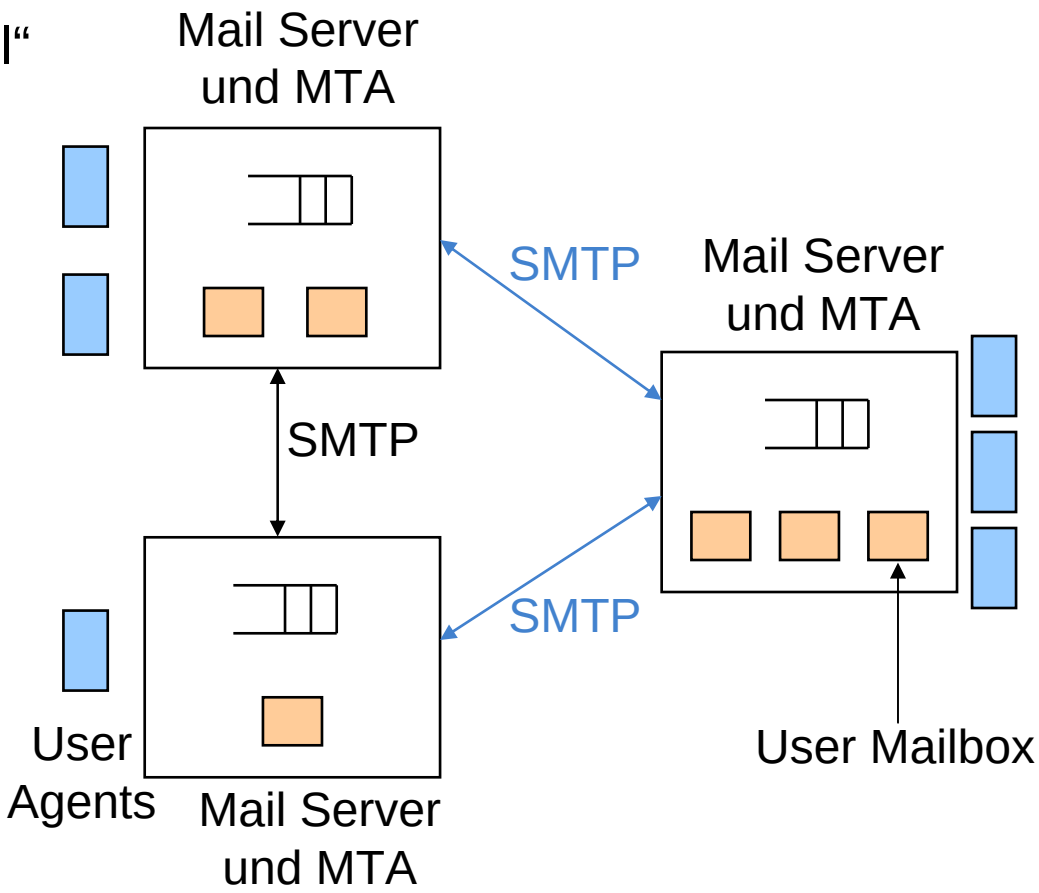
- Hauptziel
  - Internationaler Dienst zum Austausch elektronischer Mitteilungen zwischen Personen oder zwischen Rechnern
- Hauptvorteil gegenüber dem Telefon
  - Asynchron (Nachrichtenvermittlung)
- Vergleich von Briefpost und Elektronischer Post:



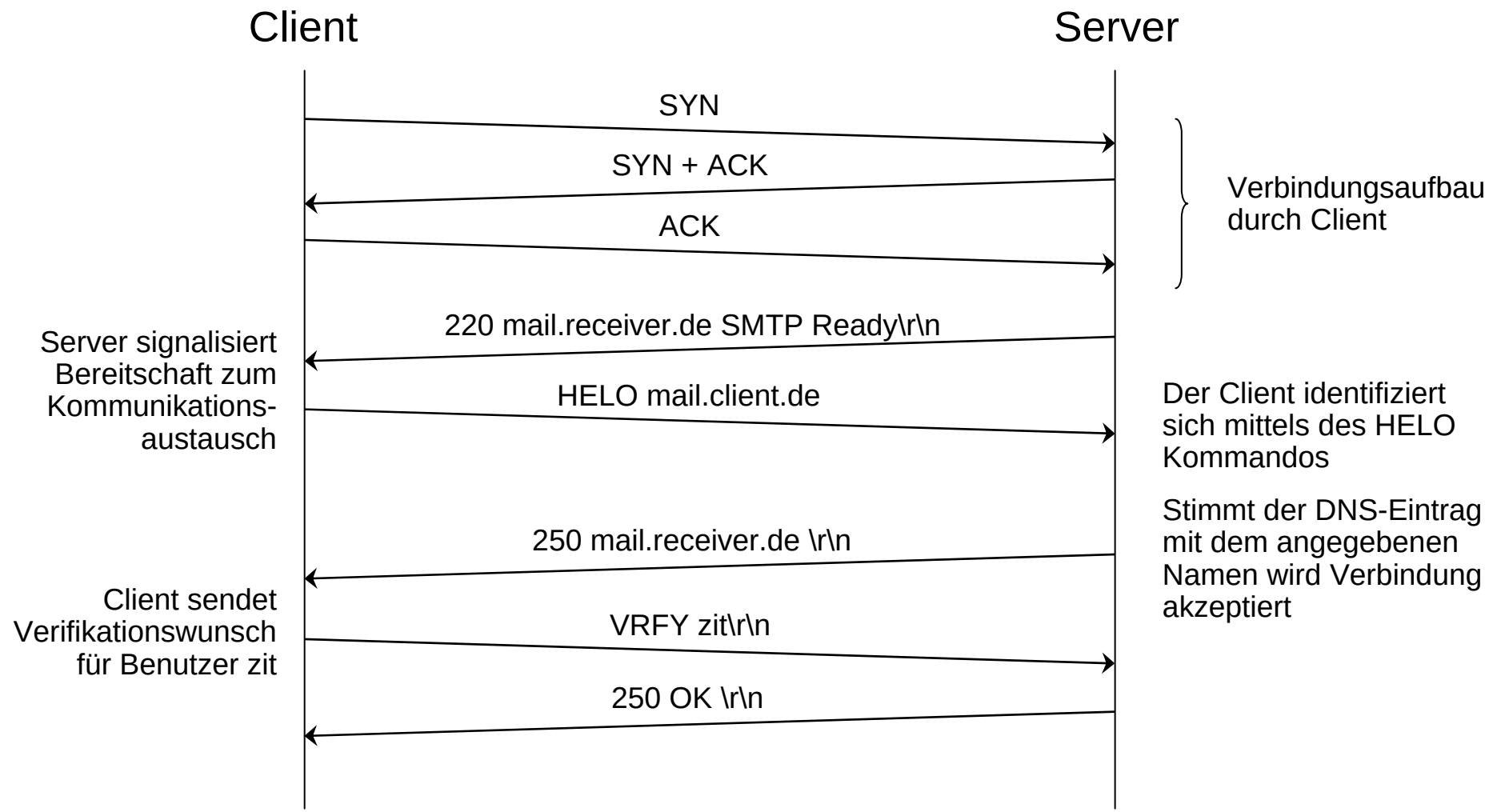
# E-Mail im Internet: SMTP

- E-Mail
  - Vergleichbar mit „Snail Mail“
    - Traditionelle Post
  - Asynchroner Datenaustausch zwischen Benutzern
  - Inhalte
    - Text, Bilder, Audio und Video
- Grundlegende Komponenten
  - User Agent (UA)
  - Mail Server
    - Mail Transport Agents (MTA)
  - Simple Mail Transfer Protocol (SMTP)

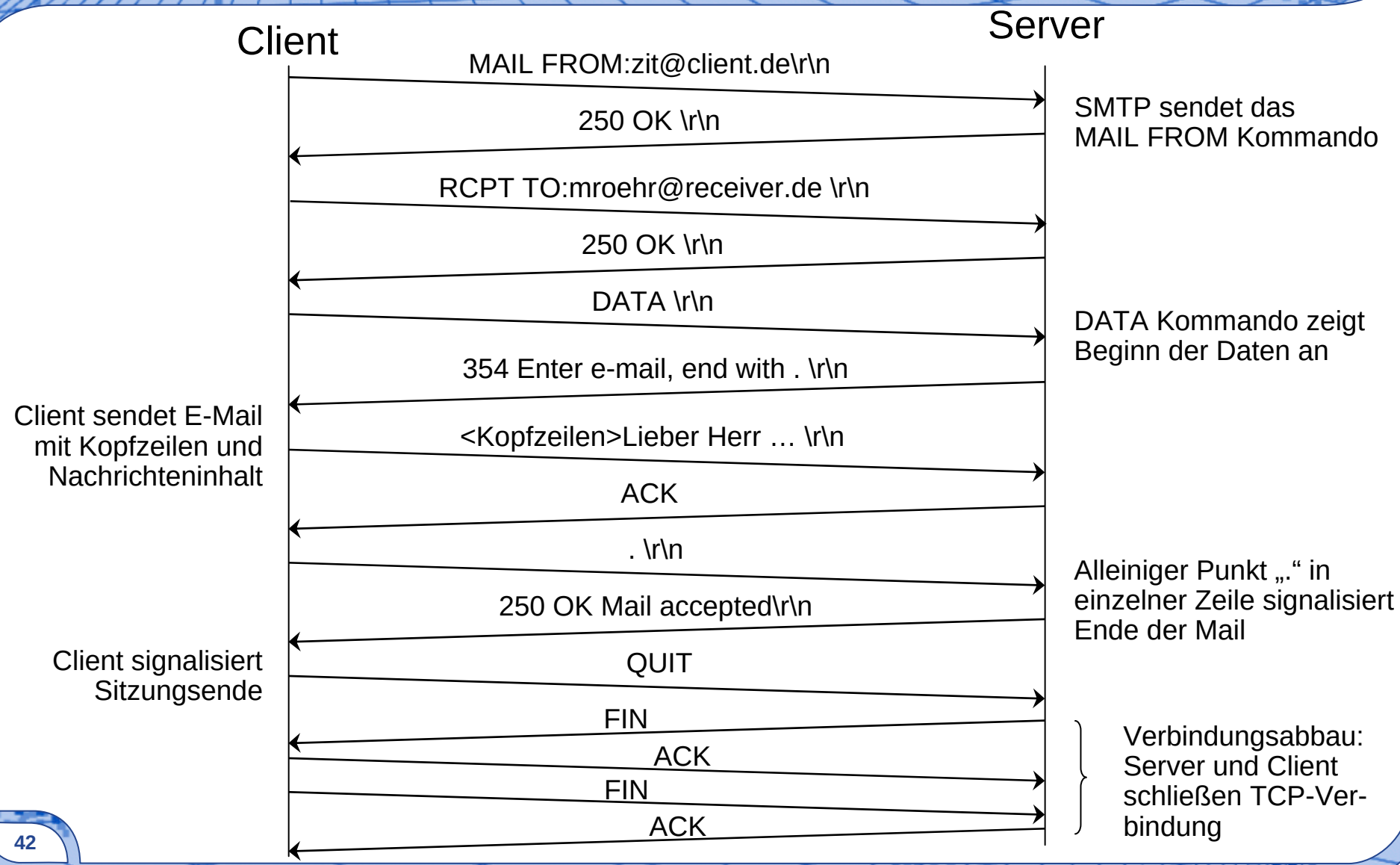



[RFC2821]

# Nachrichtenübertragung mit SMTP (1)



# Nachrichtenübertragung mit SMTP (2)



# E-Mail-Format nach RFC 822

- Beispiel

```
Envelope-to: ruggaber@tm.uka.de
Received: from i72pc108.tm.uni-karlsruhe.de ([141.3.71.162] helo=tm.uka.de)
   by irams1.ira.uka.de with smtp (Exim 3.30 #7 (Debian))
   id 161oxI-0005j9-00
   for <ruggaber@tm.uka.de>; Thu, 08 Nov 2001 14:11:07 +0100
Message-Id: <E161oxI-0005j9-00@irams1.ira.uka.de>
From: ruggaber@tm.uka.de
Date: Thu, 08 Nov 2001 14:11:07 +0100

Hallo!
```

Header

Body

- Weitere Header-Felder:
  - `BCC:` Blind Carbon Copy
  - `Reply-To:` Antwort-Adresse falls nicht an Sender-Adresse zurück geschickt werden soll
  - `In-Reply-To:` Message-ID auf die Bezug genommen wird

- Probleme des RFC-Standards
  - Keine Übertragung von Binärdateien oder ausführbaren Programmen möglich
  - Keine länderspezifischen Zeichen
  - Begrenzte Größe der Mails



[RFC822]

# 10.3.1 MIME – Multipurpose Internet Mail Extensions

- SMTP sieht nur einfache ASCII-Texte als Nachrichten vor
- MIME erweitert den Kopfteil einer Nachricht um Formatinformation



[RFC2045]

  - Content-Type: Definiert den Typ des E-Mail-Inhalts
  - Content-Transfer-Encoding: Definiert die Transfer-Syntax, in der die Daten des Hauptteils übertragen werden
- Weitgehende Kompatibilität zur herkömmlichen Internet-E-Mail
  - Mit der Transfersyntax Base 64 ist es möglich, Binärdaten durch Subnetze zu leiten, die nur die Übertragung von 7-Bit-ASCII-Texten erlauben
  - Die Transfersyntax Quoted Printable erlaubt nationale Sonderzeichen
    - Wird eine solche Mail von einem "normalen Mailer" angezeigt, so werden nur diese Erweiterungen verstümmelt

# MIME

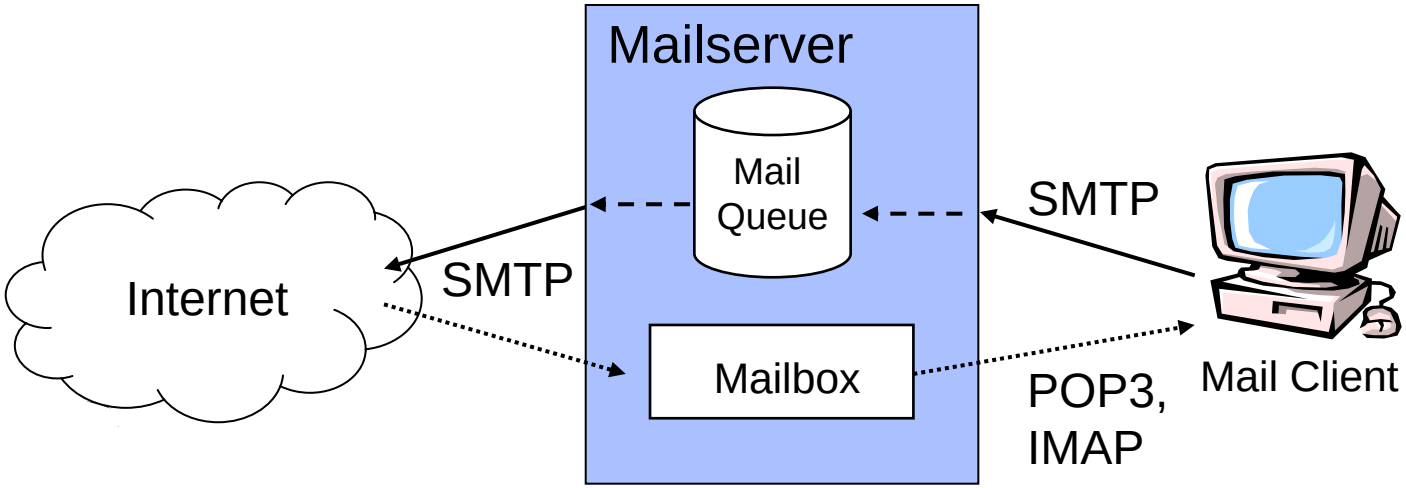
- Content-types für verschiedenartige Inhalte
  - Text: plain, richtext, HTML
  - Multipart: mixed, parallel, alternative, digest
    - Aus mehreren Teilen bestehende E-Mail (z.B. Text und HTML)
  - Message: rfc822, partial, external-body
    - Z.B. für weitergeleitete (Teile von) E-Mails
  - Image: GIF, JPEG
  - Video: MPEG
  - Audio: Basic
  - Application: PostScript, octet-stream
    - Für Daten „bekannter“ Applikationen
- Transfer Encoding
  - 7bit (ASCII Zeichen), 8bit (non-ASCII Zeichen)
  - Binary
  - Quoted-printable
    - Kompromiss zwischen Lesbarkeit und Übertragung von nicht-ASCII-Zeichen
    - ASCII-Zeichen + Hex-Codes
      - z.B. Darstellung von 12 (dezimal) : =0C
  - Base64
    - Abbildung von 6-Bit-Werten auf ASCII-Zeichen



[RFC2046]

# 10.3.2 Mailserver

- E-Mail wird meist zentral über einen Mailserver abgewickelt (Relay-MTA)
- Mittels POP3 (Post Office Protocol 3) holt der Client die vom Mailserver empfangenen und in der Mailbox gespeicherten Nachrichten ab
  - Einfache Funktionalität
- IMAP (Interactive Mail Access Protocol) dient dazu, die Nachrichten zentral auf einem Mailserver zu verwalten
  - IMAP erlaubt erweiterte Kommandos (z.B. Ordnerverwaltung, Filterung)



# 10.3.2.1 POP3

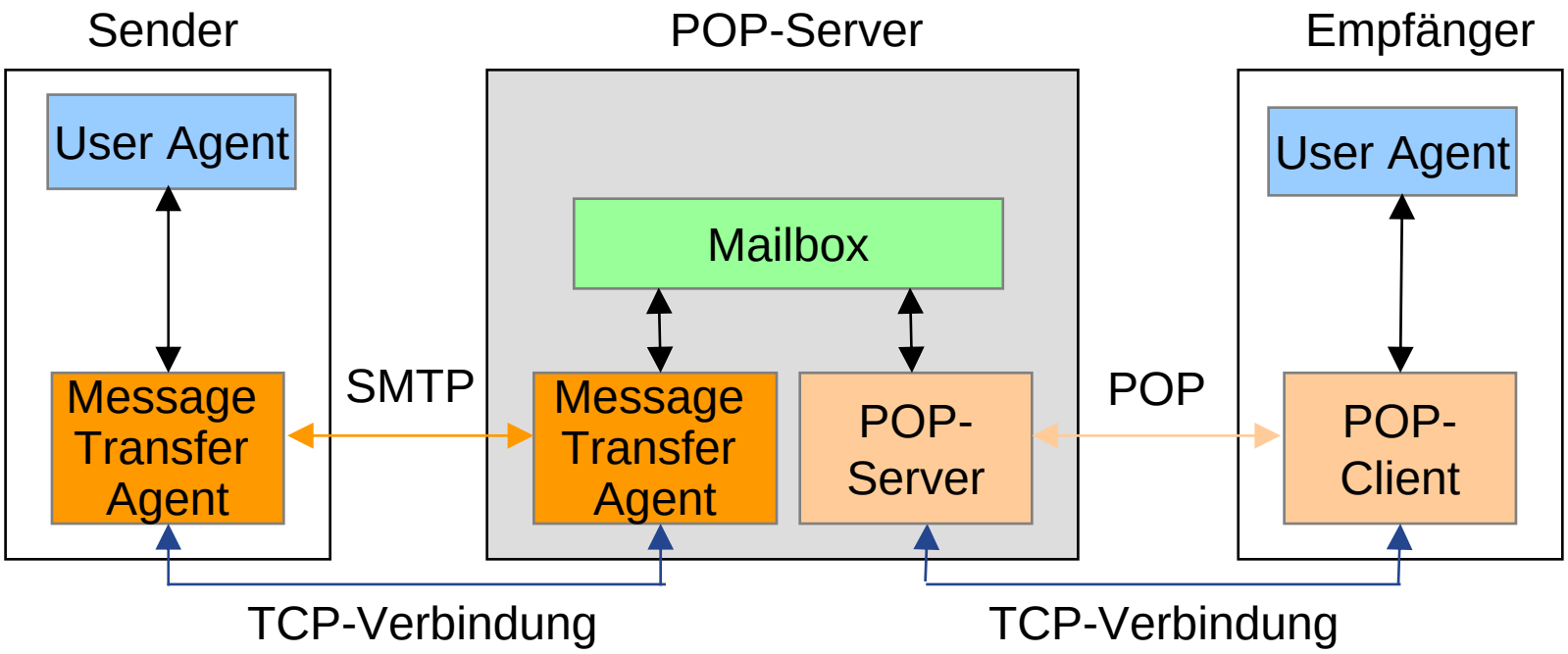
- Ablauf in verschiedenen Phasen
  - User-Agent öffnet TCP-Verbindung zum POP-Server auf Port 110
  - Autorisierung
    - Benutzername und Passwort
  - Transaktion
    - Abrufen von Nachrichten
    - Markieren von Nachrichten (Löschen ...)
  - Update
    - Löschen markierter Nachrichten



[RFC1939]

# POP3

- Schema


Sender
POP-Server
Empfänger
User Agent
Mailbox
User Agent
Message Transfer Agent
SMTP
Message Transfer Agent
POP-Server
POP
POP-Client
TCP-Verbindung
TCP-Verbindung

# 10.3.2.2 IMAP

- Vorteile gegenüber POP3
  - Benutzer kann Nachrichten-Ordner auf dem Server anlegen und verwalten
  - Suchen in Ordnern wird unterstützt
  - IMAP-Server verwaltet Ordner-Hierarchie
  - Benutzer kann Teile von Nachrichten abrufen
    - Wann ist diese Eigenschaft nützlich?
- Ablauf
  - Verbindungsaufbau zwischen Client und IMAP-Server
  - Interaktionen
    - Kommando vom Client, Daten vom Server, Fertigmeldung vom Server
  - Vier Zustände einer IMAP-Sitzung
    - Non-authenticated state
      - Kein Nutzer angemeldet, Einloggen und Abfragen der Server-Fähigkeiten
    - Authenticated state
      - Nutzer angemeldet, Operationen mit Ordnern möglich
    - Selected state
      - Ordner ausgewählt, Operationen mit Nachrichten möglich
    - Logout state
      - Ausloggen, Server sichert Änderungen endgültig



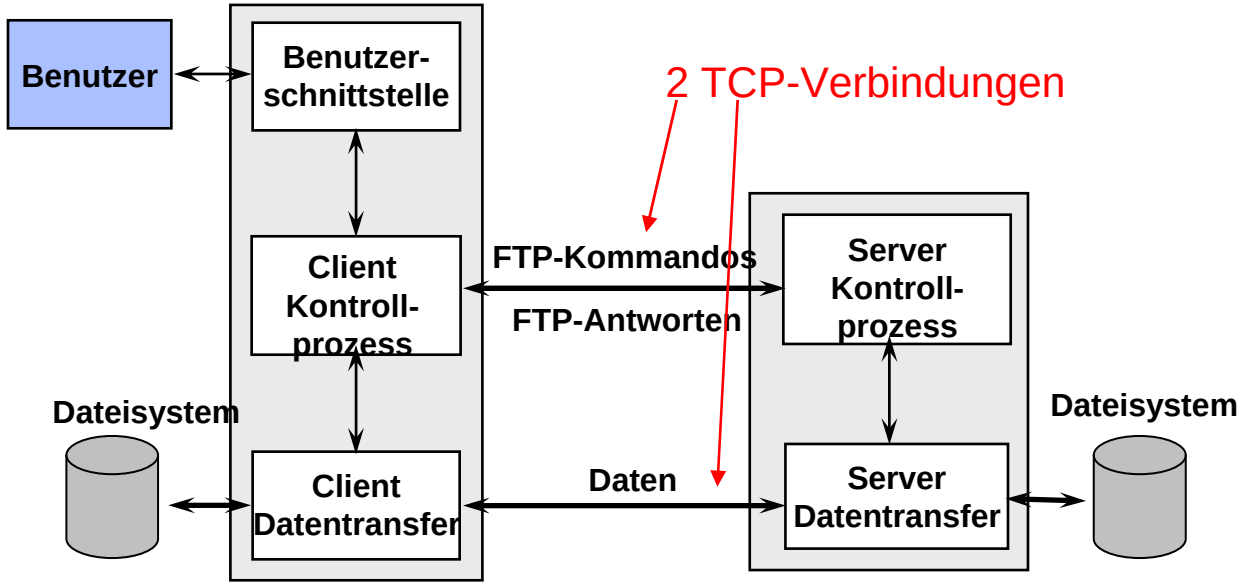
[RFC2060]

# 10.4 Dateitransfer im Internet

- File Transfer Protocol (FTP)
  - Dient der Übertragung von Dateien
  - Basiert auf Client/Server-Modell
- FTP-Dienste
  - Verbindungsaufbau mit Authentifizierung
    - Vorsicht: Authentifizierungsdaten werden im Klartext versendet
    - Auch anonymes FTP möglich
      - Jeder darf FTP-Dienst auf Server nutzen
  - Dateiübertragung von und zum Server
    - z.B. get- und put-Kommandos
  - Operationen auf Dateisystem
    - z.B. cd, dir
  - Hilfefunktionen
    - z.B. Kommando-Auflistung inklusive Parameter
  - Weitere implementierungsabhängige Dienste möglich

# FTP-Prozesse für Dateitransfer

- Nutzung zweier TCP-Verbindungen
  - Kontrollverbindung für Clientbefehle und Serverantworten
    - Server wartet dazu passiv auf Port 21
    - Bleibt bestehen, solange Client mit Server kommuniziert
    - Ermöglicht Kommunikation während eines Dateitransfers
    - IP „Type-of-Service“ sollte wegen Befehlseingabe auf minimale Verzögerung gestellt sein
  - Datenverbindung zur Übermittlung von Daten zwischen Client und Server
    - Nicht-persistent: Nach der Übertragung einer Datei wird die Verbindung geschlossen
    - IP „Type-of-Service“ sollte maximalen Datendurchsatz angeben



# FTP – Datendarstellung


[RFC959]

- Dateityp
  - ASCII-Dateityp, EBCDIC-Dateityp, Binär-Dateityp, lokaler Dateityp
- Formatierungsinformationen
  - Nonprint, Telnet-Format-Control, Fortran-Carriage-Control
- Struktur
  - Dateistruktur, Datensatzstruktur, Seitenstruktur
- Übertragunsmodus
  - Stream-Modus, Block-Modus, Komprimierter Modus

→ Nur die blau markierten Optionen werden typischerweise implementiert

# FTP-Befehle

- Befehle und Antworten erfolgen im ASCII-Format
  - Sequenz Carriage-Return (CR) plus Linefeed (LF) an jedem Zeilenende notwendig
- Befehle bestehen aus 3 oder 4 Bytes mit ASCII-Großbuchstaben
  - Manche mit optionalen Argumenten
  - Gängige der 30 möglichen Befehle siehe Tabelle unten



[Stev04]

| **Befehl** | **Beschreibung** |
|---|---|
| ABOR | Vorherigen FTP-Befehl sowie Datentransfer abbrechen |
| LIST *dateiliste* | Liste von Dateien oder Verzeichnissen ausgeben |
| PASS *passwort* | Passwort für Server |
| PORT $n_1$,$n_2$,$n_3$,$n_4$,$n_5$,$n_6$ | Client IP-Adresse ($n_1$,$n_2$,$n_3$,$n_4$) und Port ($n_5$ x 256 + $n_6$) |
| QUIT | Am Server abmelden |
| RETR *dateiname* | Datei holen (get) |
| STOR *dateiname* | Datei speichern (put) |
| SYST | Server gibt Systemtyp zurück |
| TYPE *type* | Dateityp spezifizieren: A für ASCII, I für Binärdatei |
| USER *benutzername* | Benutzername am Server |

# FTP-Antworten

- Antworten bestehen aus 3-stelligen ASCII-Zahlen mit optionaler Nachricht nach Nummer
- Jede der drei Stellen besitzt andere Bedeutung

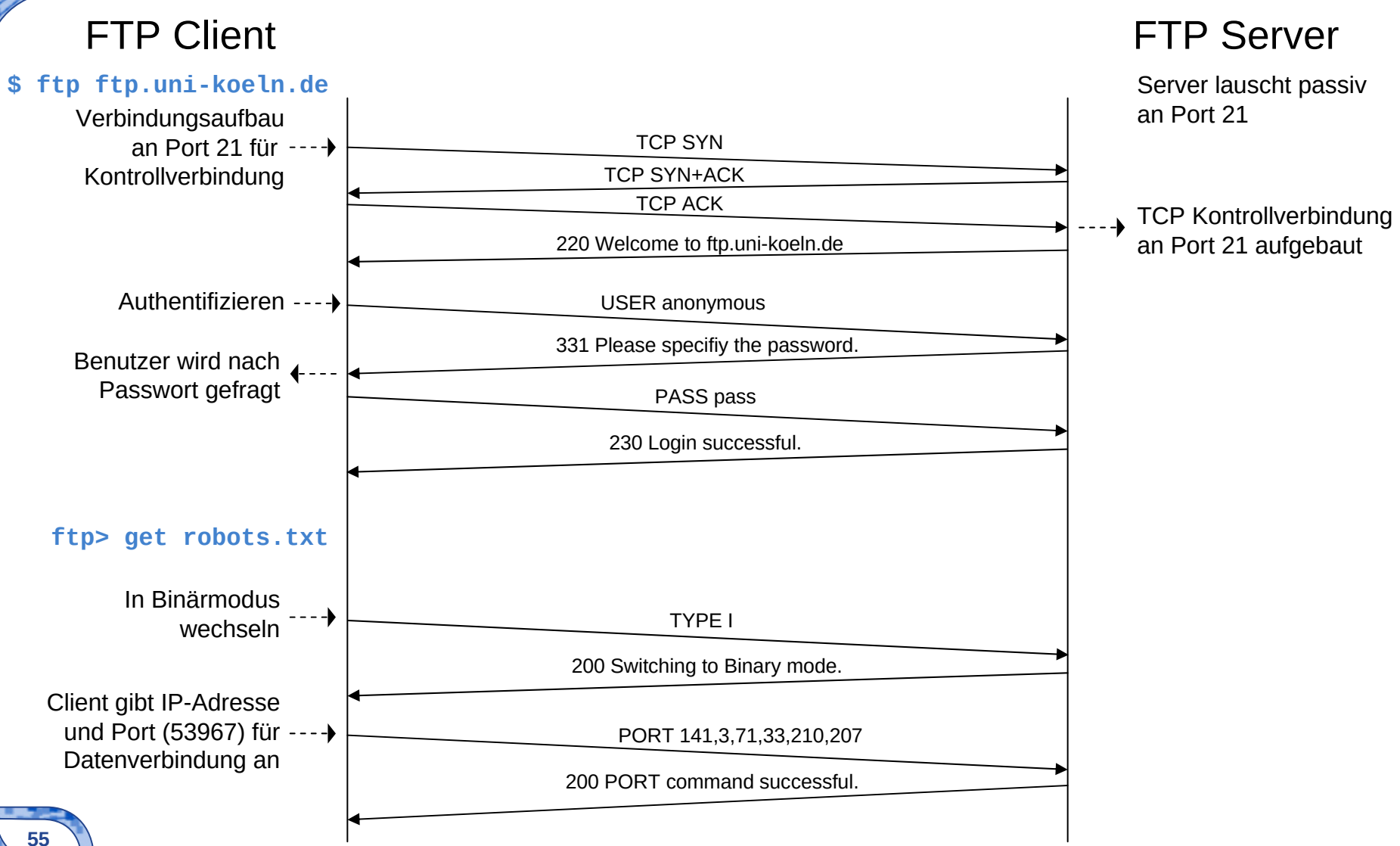

[Stev04]

| Antwort | Bedeutung |
|---|---|
| 1yz | Positive vorläufige Antwort |
| 2yz | Positive abschließende Antwort |
| 3yz | Positive zwischenzeitliche Antwort |
| 4yz | Transiente negative abschließende Antwort |
| 5yz | Permanente negative abschließende Antwort |
| x0z | Syntax |
| x1z | Information |
| x2z | Verbindungen |
| x3z | Authentifizierung und Abrechnung |
| x4z | Nicht spezifiziert |
| x5z | Dateisystem-Status |

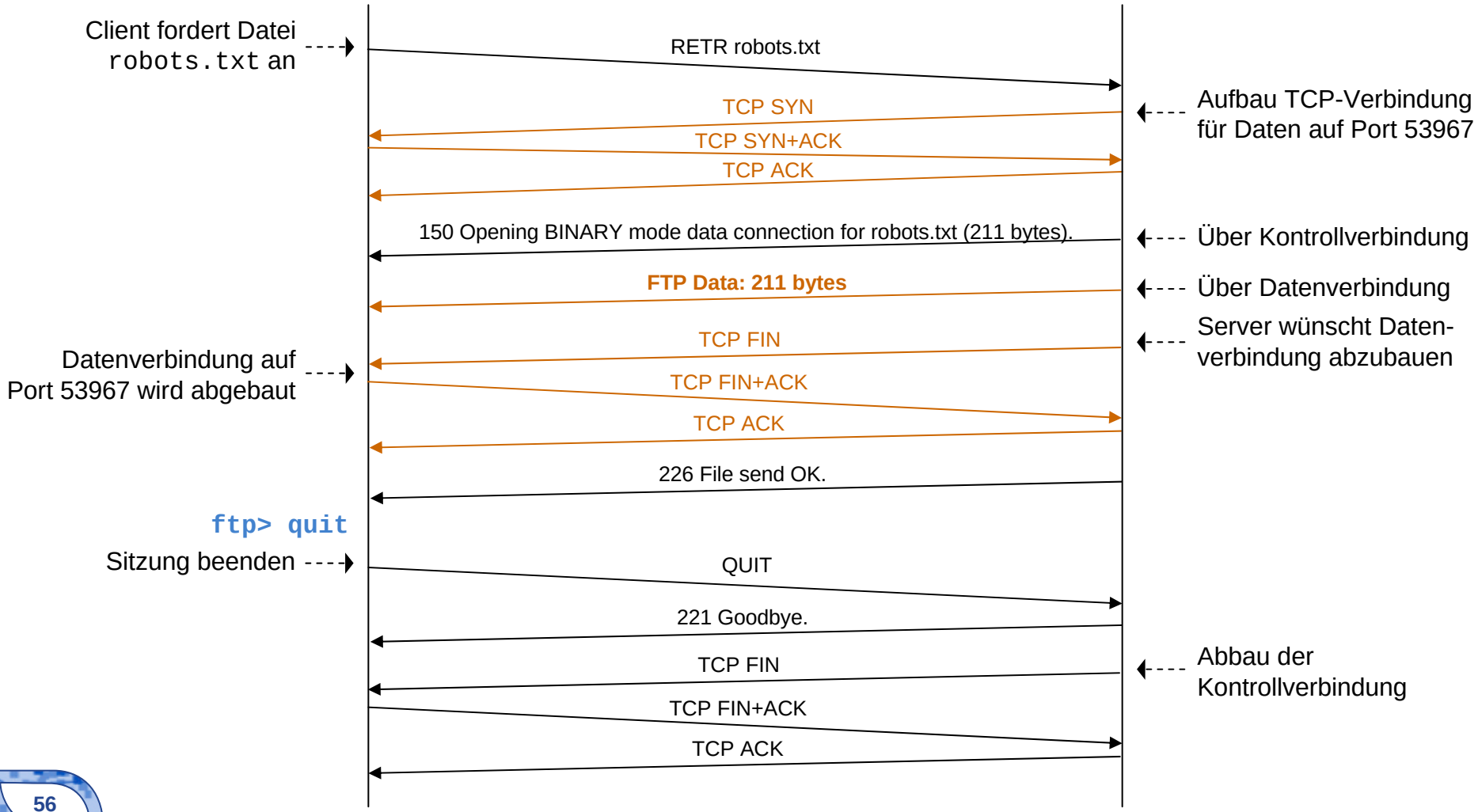
Beispiele:

- 125 Datenverbindung besteht
- 200 Befehl OK
- 214 Hilfemeldung
- 331 Benutzername OK, Passwort erforderlich
- 425 Kann Datenverbindung nicht öffnen
- 500 Syntaxfehler (Befehl unbekannt)
- 501 Syntaxfehler (ungültige Argumente)

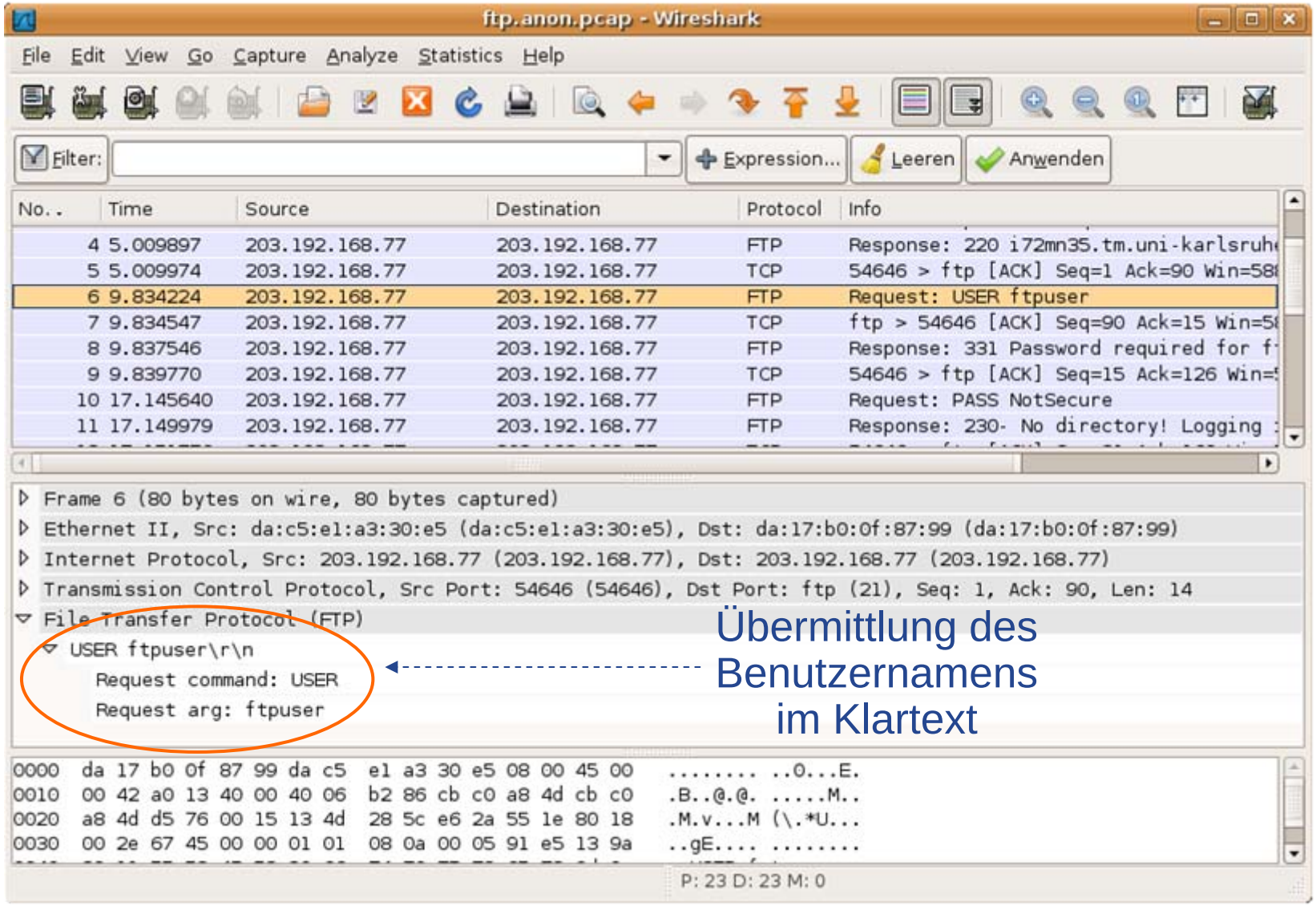
# Beispiel FTP-Nachrichten (1)



# Beispiel FTP-Nachrichten (Forts.)

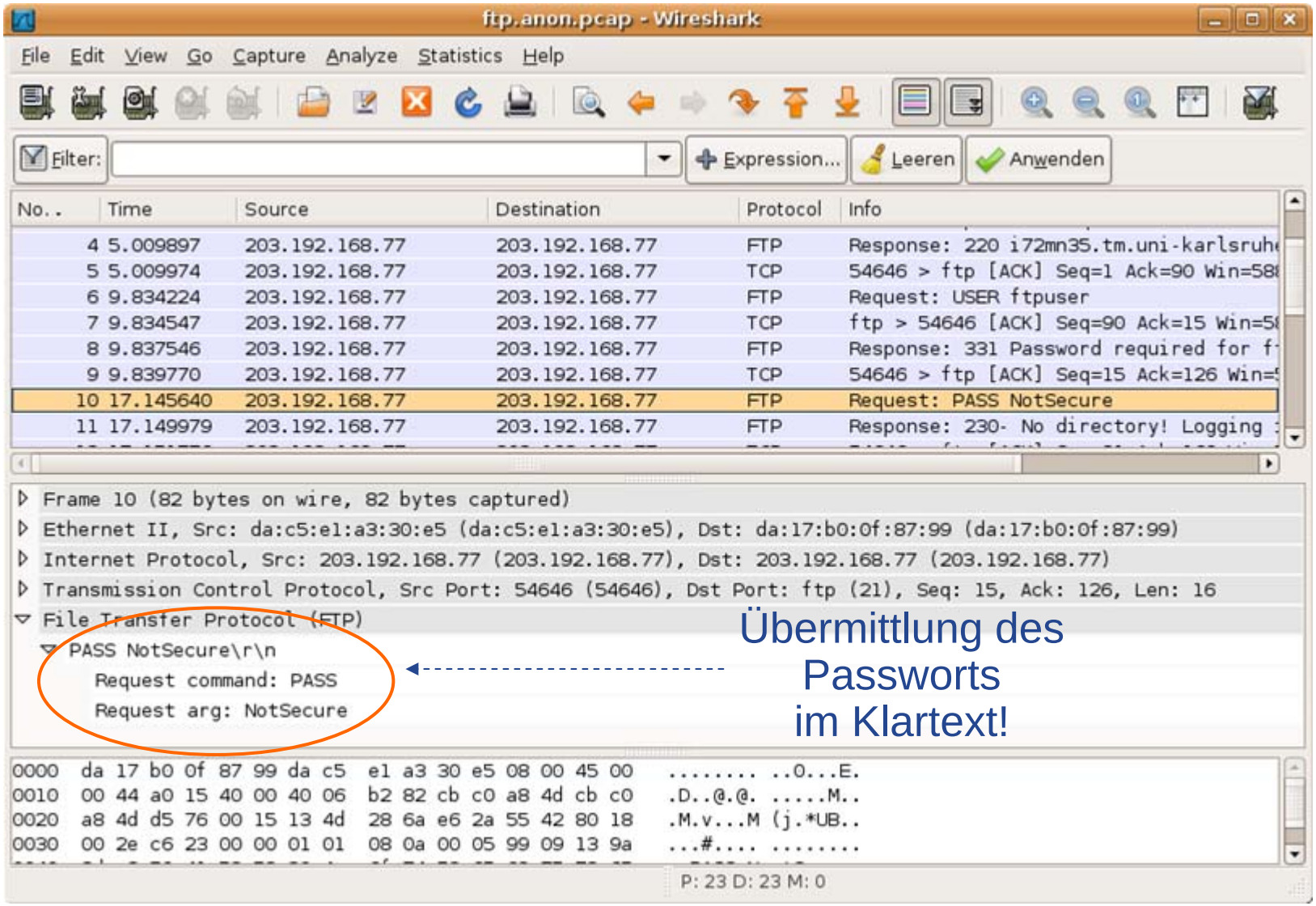


# Beispiel FTP-Authentifizierung



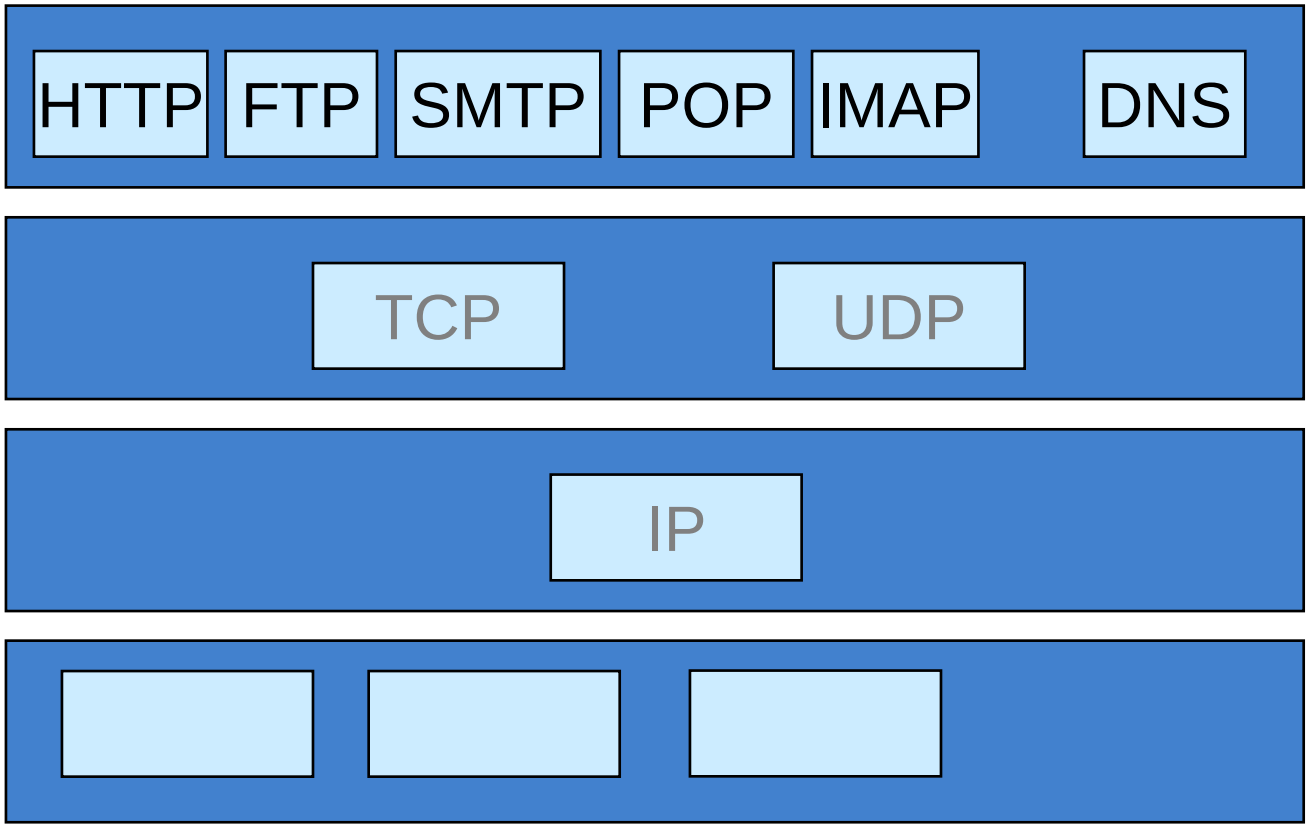

Übermittlung des Benutzernamens im Klartext

# Beispiel FTP-Authentifizierung (2)




Übermittlung des Passworts im Klartext!

# Zusammenfassung



# Aufgaben

10.1 Was ist ein URL, wozu dient er und wie ist er aufgebaut?

10.2 Wie läuft üblicherweise der Abruf einer WWW-Seite über HTTP1.0 ab?

10.3 Beschreiben Sie, wie die Mechanismen *Persistente Verbindungen*, sowie *Pipelining* Verbesserungen bewirken.

10.4 Erläutern Sie die Aufgaben des Domain Name Systems und dessen Aufbau.

10.5 Wie läuft eine Namensauflösung ab und welche unterschiedlichen Anfragetypen können hierbei vorkommen?

10.6 Beschreiben Sie, worin eine DNS-Antwort besteht und geben Sie Beispiele, was sie enthalten kann.

10.7 Wozu dient und wie funktioniert eine rekursive DNS-Abfrage?

10.8 Wie werden E-Mails über das Internet transportiert?

10.9 Erläutern Sie, mit welchen Mitteln über E-Mails mehr als nur Text transportiert werden kann.

10.10 Welche Protokolle spielen beim Abruf von E-Mails aus Mailboxen die Hauptrolle und worin bestehen deren wichtigste Unterschiede?

10.11 Wofür werden beim Dateitransfer über FTP zwei Verbindungen verwendet?

# Literatur


[Akam09] http://www.akamai.com

[DENI09] DENIC eG; http://www.denic.de

[Hals05] F. Halsall; Computer Networking and the Internet; Addison-Wesley, 2005, 5th Edition

[Heis08] R. Kaps; IPv6 für DNS-Rootserver; heise online News, Februar 2008

[KuRo07] J. F. Kurose, K. W. Ross; Computer Networking; 4rd ed., Addison-Wesley, 2007

[RFC822] David H. Crocker (Ed.); Standard for the Format of ARPA Internet Text Messages; RFC 822, August 1982

[RFC959] J. Postel, J. Reynolds; File Transfer Protocol (FTP); RFC 959, October 1985

[RFC1034] P. Mockapetris; Domain Names – Concepts and Facilities; RFC 1034; November 1987

[RFC1035] P. Mockapetris; Domain Names – Implementation and Specification; RFC 1035; November 1987

[RFC1939] J. Myers, M. Rose; Post Office Protocol – Version 3; RFC 1939, May 1996

[RFC1945] T. Berners-Lee, R. Fielding, H. Frystyk; Hypertext Transfer Protocol -- HTTP/1.0; RFC 1945, May 1996

# Literatur


[RFC2045] N. Freed, N. Borenstein; Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies; RFC 2045, November 1996

[RFC2046] N. Freed, N. Borenstein; Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types; RFC 2046, November 1996

[RFC2060] M. Crispin; Internet Message Access Protocol – Version 4rev1; RFC 2060, December 1996

[RFC2616] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, T. Berners-Lee; Hypertext Transfer Protocol – HTTP/1.1; RFC 2616, June 1999

[RFC2821] J. Klensin (Ed.); Simple Mail Transfer Protocol; RFC 2821, April 2001

[RFC3596] S. Thomson, C. Huitema, V. Ksinant, M. Souissi; DNS Extensions to Support IP Version 6; RFC 3596, October 2003

[RFC3986] T. Berners-Lee, R. Fielding, L. Masinter; Uniform Resource Identifier (URI): Generic Syntax; RFC 3986, January 2005

[RIPE09] RIPE NCC; k-root-server; http://k.root-servers.org

[Stev04] W. R. Stevens; TCP/IP; Hüthig-Verlag, 2004

# Kommunikation und Datenhaltung

## 11. Middleware




**Prof. Dr. Martina Zitterbart**
Dipl.-Inform. Martin Röhricht
[zit | roehricht]@tm.uka.de



# Kapitelübersicht

1. Einführung
2. Physikalische Grundlagen
3. Protokollmechanismen
4. Geschichtete Architekturen
5. Sicherungsschicht: HDLC
6. Beschreibungsmethoden
7. Sicherungsschicht: Lokale Netze
8. Netzkopplung und Vermittlung
9. Die Transportschicht
10. Anwendungssysteme
11. Middleware

11.1 Aufgaben und Dienste
11.2 Überwindung von Verteilung
11.3 Überwindung von Heterogenität

# 11.1 Middleware

- Betriebssystem
  - Softwaresystem, das anwendungsnahe Dienste erbringt
    - Abstraktion von der unterliegenden Hardware
    - Schnittstelle zwischen Anwendung und Hardware
  - Ursprünglich lokal auf dem Rechner
  - Steuerung der Vergabe von Betriebsmitteln, Prozessorzuteilung etc.
  - ⇨ Verteilte Betriebssysteme: nicht mehr auf einen Rechner fokussiert
- Middleware
  - Softwaresystem, das Dienste zur Unterstützung verteilter Anwendungen erbringt
    - Sitzt „oberhalb“ des Betriebssystems
  - Rechnerübergreifend, also verteilt
  - Verbirgt Heterogenität von Datenübertragung, Betriebssystemen und Programmiersprachen vor den Dienstnehmern
  - Programmiermodell für verteilte Applikationen

⇨Im Folgenden wird Middleware als „Kitt und Mörtel“ zwischen Dienstnehmer und Dienstgeber betrachtet

- ⇨ Lokale Betriebssysteme sind ebenfalls vorhanden
- ⇨ „Middleware is all about integration“ – „*Middleware is everywhere*“

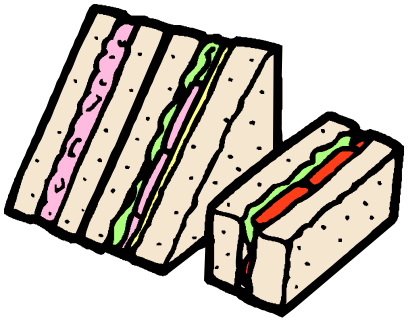


[Vino02]

# Middleware: Aufgaben und Dienste

- Aufgabe
  - Zusammenführung von Dienstnehmern und Dienstgebern
    - Kommunikation
    - Verteilte Transaktionsverarbeitung
    - ...
- Ziele
  - Skalierbarkeit
  - Offenheit
  - Dienstekopplung

- Die Middleware macht den Unterschied aus ...
  - Was wäre ein Sandwich ohne was „dazwischen“?

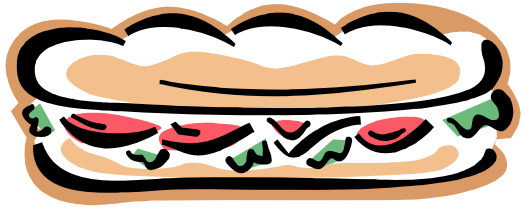


Tomate?

Salat?

Mayonnaise?

... ?



# Aufgaben der Middleware

- Überwindung von Verteilung (Ortstransparenz)
  - Dienstnehmer und Dienstgeber müssen nicht im selben Adressraum ablaufen
- Überwindung von Heterogenität
  - Technische und semantische Interoperabilität
- Vermittlung von Dienstgebern
  - Bevor Dienstgeber in Anspruch genommen werden kann, muss dieser „gefunden“ werden
- Dienstaktivierung und -beendigung
  - Wann wird Dienst aktiviert?
- Lastverteilung
  - Evtl. Verteilung der Anfragen auf unterschiedliche Rechner
- Sicherheit
  - Herstellen gegenseitigen Vertrauens
- Persistenz
  - Dauerhaftigkeit für zustandsbasierte Dienste
- Transaktionen
  - Konfliktserialisierbarkeit

# Herausforderungen der Middleware

- Flexibilität
- Hohe Leistungsfähigkeit
  - Hooks für „Power-User“

} Gegensätzlich!

- Konfiguration
  - Vielzahl an Möglichkeiten
- Übergreifendes Ziel
  - Integration von Anwendungen
    - Über Intranets hinausgehend

⇨ Web Services stellen eine neue wichtige Entwicklung in dieser Richtung dar

# 11.2 Überwindung von Verteilung

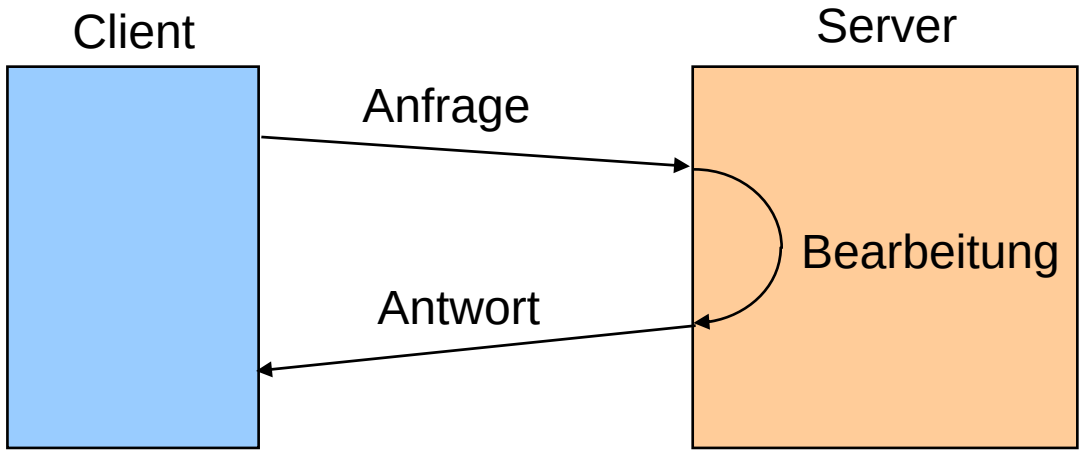
- Verteilte Systeme
  - Dienstnehmer und Dienstgeber laufen nicht unbedingt im selben Adressraum
    - Entfernter Dienstgeber
    - Entfernter Funktionsaufruf (*Remote Procedure Call*, RPC)
    - ...
- „Milderung“ von Verteilungsaspekten
  - Annäherung des entfernten Programmiermodells an das lokale

  ⇨ Für den Client ist der Aufenthaltsort des Servers verdeckt
- Aber: Transparenz ist zwangsläufig unvollkommen
  - Abweichungen bei der Parameterübergabe/-rückgabe
    - Insbesondere bei Zeigerparametern und verzeigerten Strukturen
  - Komplexeres Fehlermodell
  - Auswirkungen auf die Leistung (Verzögerung)

⇨ Programmierer muss sich Zugriffen auf potenziell entfernte Dienste bewusst sein!

# Client-Server-Paradigma

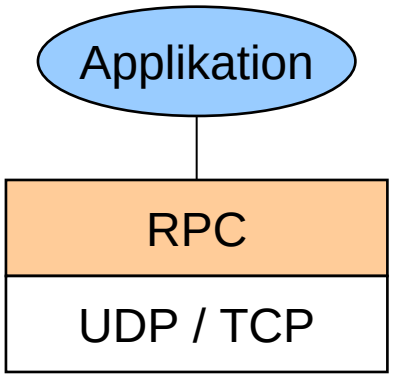
- Grundkonzept
  - Server stellen Dienste zur Verfügung
  - Clients benutzen diese Dienste, indem sie Anfragen an die Server senden
  - Server liefern Ergebnisse der Anfrage an Client zurück
- Unterschiedliche Rollen von Client und Server (Asymmetrie)
- Weiterleitung von Anfragen möglich, Server wird dann zum Client



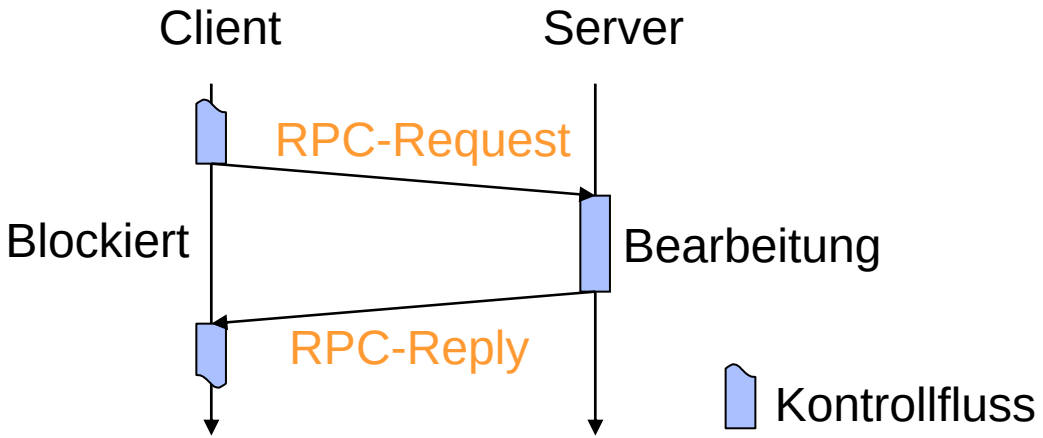

- Im Java-Umfeld
  - *Remote Method Invocation* (RMI)
    - Client-Objekt ruft Methode auf, die ein Server-Objekt auf einem entfernten Rechner bereitstellt

# Remote Procedure Call (RPC)

- Kommunikationsmechanismus für Client/Server-Anwendungen
  - Ermöglicht das Aufrufen von Prozeduren auf entfernten Rechnern
    - Programmiersprachliche Einbettung
  - Für den Programmierer bleibt der Datenaustausch transparent
- RPC ist im Schichtenmodell über UDP oder TCP angesiedelt
  - In der Realisierung meist Bestandteil der Anwendung selbst
- Schema




Ablauf



# Definition und Ablauf

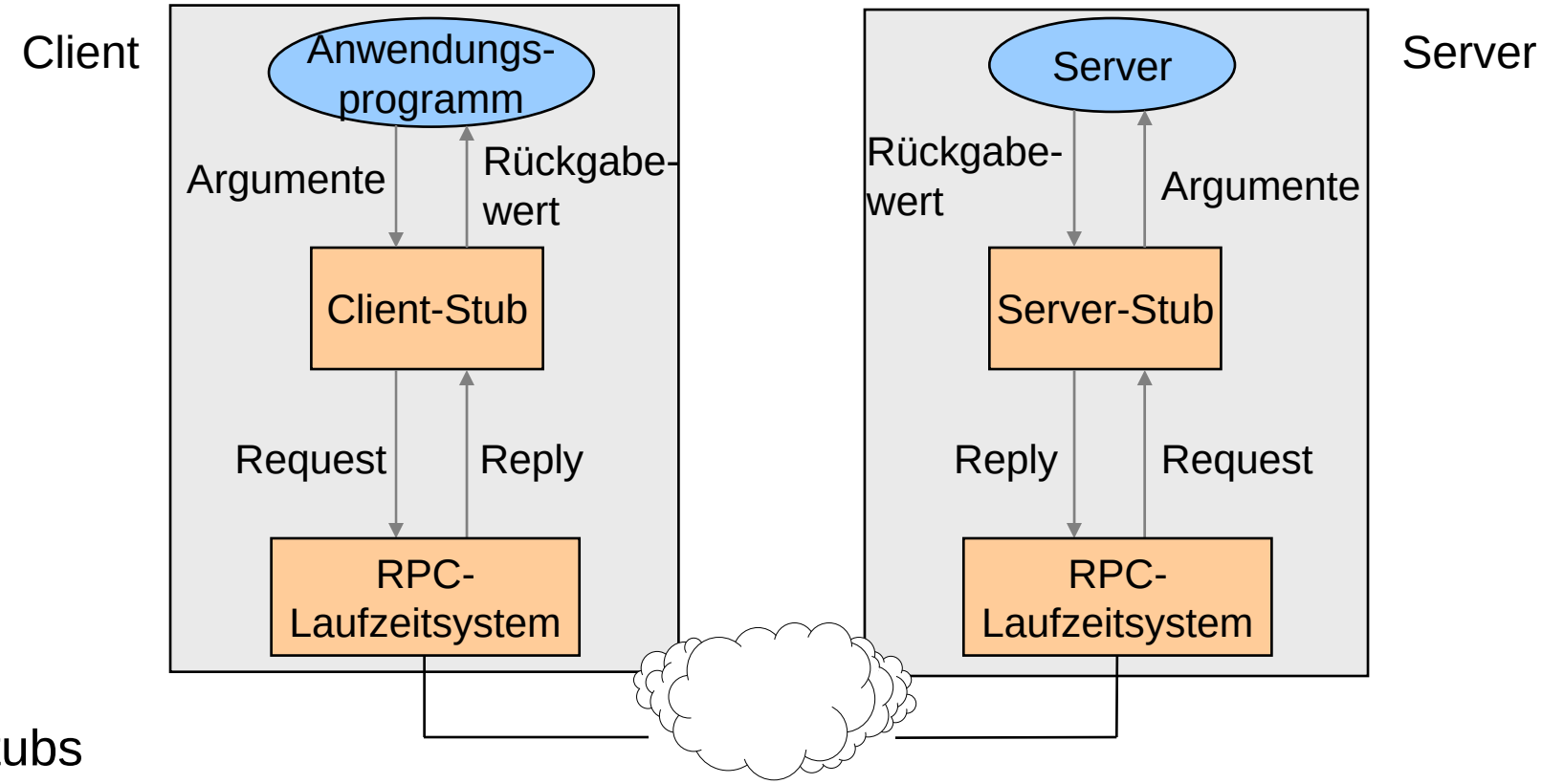
- Erweiterung des Prozeduraufrufs zum entfernten Prozeduraufruf
- Definition (nach Nelson und Birell, 1984)
  - Synchrone Übergabe des Kontrollflusses
    - Aufrufender Client wird blockiert
  - Ebene der Programmiersprache
  - Getrennte Adressräume
    - Keine netzweit eindeutigen Adressen (Zeiger!)
  - Kopplung über schmalen Kanal
    - Weniger leistungsfähig als lokale Interaktionsmechanismen
    - Achtung: Fehler möglich, die im lokalen System nicht auftreten
  - Datenaustausch: Aufrufparameter und Ergebnisse
    - Achtung: unterschiedliche Darstellung von Daten auf heterogenen Maschinen!
- Ablauf
  - Aufruf im Wartezustand
  - Parameter- und Aufrufübertragung ins Zielsystem
  - Prozedurausführung
  - Rückmeldung
  - Fortsetzung der Programmausführung

# RPC-Semantik

- Bei der Verwendung entfernter Prozeduraufrufe können eine Reihe von Fehlern auftreten
  - Anfragen oder Antworten können bei der Datenübertragung verloren bzw. verfälscht werden
  - Client oder Server können im Verlauf eines RPCs (unabhängig voneinander) abstürzen
- Bzgl. der Fehlerbehandlung werden folgende Fehlersemantik-Klassen unterschieden

| Fehlersemantik | Anfrage einer Wiederholung | Filterung Duplikate | Wiederausführung/ Wiederholung Antw. |
|---|---|---|---|
| Maybe | Nein | Nein | Nein |
| At-least-once | Ja | Nein | Wiederausführung |
| At-most-once | Ja | Ja | Wiederholung Antw. |
| Exactly-once | Ja | Ja | Nein |

# Modell des RPC




- Stubs
  - Aufrufkodierung / Ergebnisdekodierung
    - Transformation der Daten in eine flache Repräsentation
    - Nutzt Spezifikation der Prozedurschnittstelle
- RPC-Laufzeitsystem
  - Aufrufübertragung / Ergebnis- bzw. Aufrufempfang / Ergebnisübertragung

# Ablauf

- Anwendungsprogramm ruft entfernte Prozedur auf (die der Client-Stub zur Verfügung stellt) und wird in den Ruhezustand versetzt
- Aufrufkodierung durch den Client-Stub (Marshalling)
- Senden der erzeugten Nachrichten an das entfernte System mit Hilfe des RPC-Laufzeitsystems
- RPC-Laufzeitsystem nutzt Transportprotokoll zum Senden der Nachrichten
- RPC-Laufzeitsystem leitet empfangene Nachrichten an Server-Stub weiter
- Server-Stub dekodiert die Nachrichten (Unmarshalling)
- …

# Ablauf

- …
- Server-Stub führt lokalen Prozeduraufruf durch
  - Übergibt Parameter an tatsächliche Server-Prozedur
- Server-Prozedur übergibt nach Bearbeitung Ergebnis an Server-Stub
- Server-Stub kodiert das Ergebnis
- Server-Stub sendet mittels RPC-Laufzeitsystem die erzeugte Ergebnisnachricht an Client
- RPC-Laufzeitsystem leitet Daten an Client-Stub weiter
- Client-Stub dekodiert Daten, aktiviert das Anwendungsprogramm und reicht Ergebnis zum Anwendungsprogramm weiter

# Schnittstellenbeschreibung

- Für den entfernten Prozeduraufruf wird eine einheitliche Beschreibung der Prozeduren und Parameter benötigt
  - Schnittstellenbeschreibungssprachen
    - Formale Sprache zur Beschreibung von Prozeduren und Parametern
      - Abstrahieren von unterschiedlichen Programmiersprachen (ähnlicher Art)
    - Dienen ausschließlich zur Schnittstellendefinition
  - ⇨ Strikte Trennung von Schnittstelle und Implementierung
  - Auf der Basis der Schnittstellenbeschreibungssprache können Stubs automatisch erzeugt werden
- Beispiele
  - ASN.1 (Abstract Syntax Notation One)
  - XDR (eXternal Data Representation)
  - IDL (Interface Definition Language)

# 11.3 Überwindung von Heterogenität

- Voraussetzung für Inanspruchnahme eines Dienstes
  - Einigkeit über Dienstschnittstelle und Protokoll
- Probleme in verteilten Systemen können entstehen durch
  - unterschiedliche Programmiersprachen
  - unterschiedliche Typsysteme und Aufrufkonventionen
- Ziel
  - Reduzierung der Vielfalt
    - Nicht für jede Kombination aus unterschiedlichen Komponenten soll ein eigenes Verfahren definiert werden ⇨ Komplexität

# Probleme durch Heterogenität

- Unterschiedliche Wertebereiche
  - z.B. Typ **`long`**
    - 4 Bytes auf i386, PowerPC, Sparc
    - 8 Bytes auf Alpha, IA64, Sparc64, x86_64
- Speicherausrichtung
  - Byte-Padding von Compiler
    - GCC ermöglicht bspw. Angabe von **`__attribute__((packed))`**
- Nicht unterstützte Datentypen
  - Bsp.: Typ **`unsigned int`** unter Java nicht vorhanden
- Byteorder: Little oder Big Endian
  - Bsp.: Datei test.c

```
struct foo {
    char s[6];
    int i;
};

struct foo bar = {"FooBar", 0x12345678};
```

  - Objektcode erzeugen mittels
    - **`gcc -c -o test.o test.c`**
  - Assembler-Ausgabe erzeugen mittels
    - **`gcc -S -o test.s test.c`**

# Endianess und Speicherausrichtung

```
$ hexdump -C test.o
00000000  fe ed fa ce 00 00 00 12  00 00 00 00 00 00 00 01  |................|
00000010  00 00 00 03 00 00 01 6c  00 00 20 00 00 00 00 01  |.......l.. .....|
00000020  00 00 01 04 00 00 00 00  00 00 00 00 00 00 00 00  |................|
[...]
00000130  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 50  |...............P|
00000140  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
*
00000180  00 00 00 00 00 00 00 00  46 6f 6f 42 61 72 00 00  |........FooBar..|
00000190  12 34 56 78                                       |.4Vx|
00000194
```

- test.s auf PowerPC (Big Endian)

```
   .section __TEXT,__text,[...]
   .section __TEXT,[...]
   .machine ppc
   .globl _bar
   .data
   .align 2_
bar:
   .ascii "FooBar"
   .space 2
   .long 305419896
   .subsections_via_symbols
```

```
$ hexdump -C test.o
00000000  7f 45 4c 46 01 01 01 00  00 00 00 00 00 00 00 00  |.ELF............|
00000010  01 00 03 00 01 00 00 00  00 00 00 00 00 00 00 00  |................|
00000020  b8 00 00 00 00 00 00 00  34 00 00 00 00 00 28 00  |........4.....(.|
00000030  07 00 06 00 46 6f 6f 42  61 72 00 00 78 56 34 12  |....FooBar..xV4.|
00000040  00 47 43 43 3a 20 28 47  4e 55 29 20 34 2e 31 2e  |.GCC: (GNU) 4.1.|
00000050  33 20 32 30 30 37 30 39  32 39 20 28 70 72 65 72  |3 20070929 (prer|
[...]
000001c0  00 00 00 00 00 00 00 00  01 00 00 00 00 00 00 00  |................|
000001d0
```

- test.s auf Intel x86 (Little Endian)

```
   .file "test.c"
.globl bar
   .data
   .align 4
   .type  bar, @object
   .size  bar, 12
bar:
   .ascii "FooBar"
   .zero  2
   .long 305419896
   .ident "GCC: (GNU) 4.1.3 [...]
```

# Beispiel zur Heterogenität



System 1



System 2

```
TYPE Intf =
  RECORD
    index: INTEGER
    beschr: ARRAY
      [1..20] OF CHAR
    aktiv: BOOLEAN;
END
```

lokale Darstellung

```
WLAN-Interface ::= {
  index          0,
  beschreibung   „Intel Wireless Pro“,
  aktiv          true
}
```

Konkretes Beispiel

```
typedef struct {
  int index;
  char *beschr;
  int aktiv;
} intf;
```

lokale Darstellung

```
Interface ::= SEQUENCE {
    index INTEGER;
    beschreibung IA5STRING;
    aktiv BOOLEAN
}
```

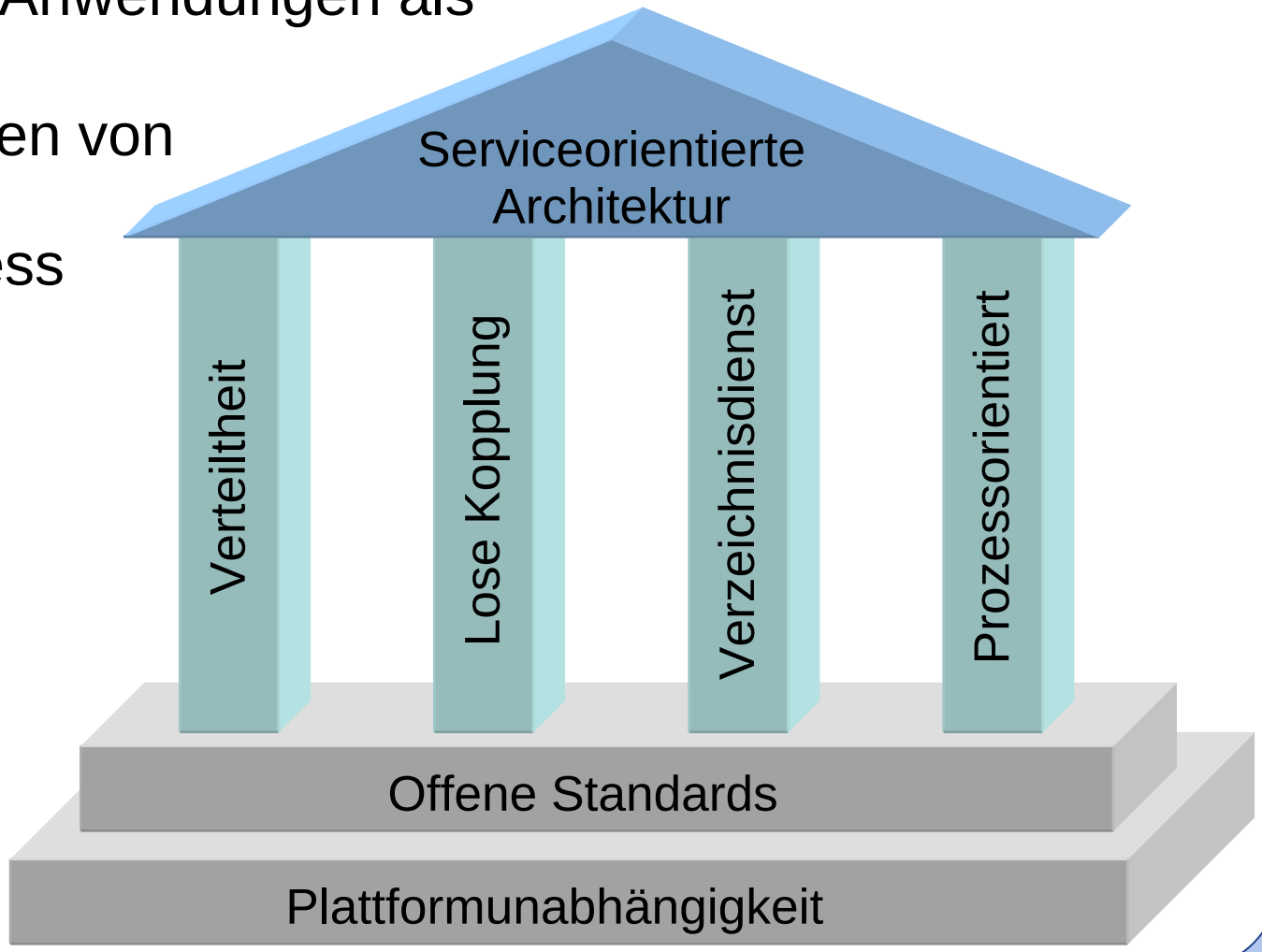
Abstrakte Syntax, z.B. ASN.1

Kodierung/De-Kodierung, z.B. BER

Transfersyntax

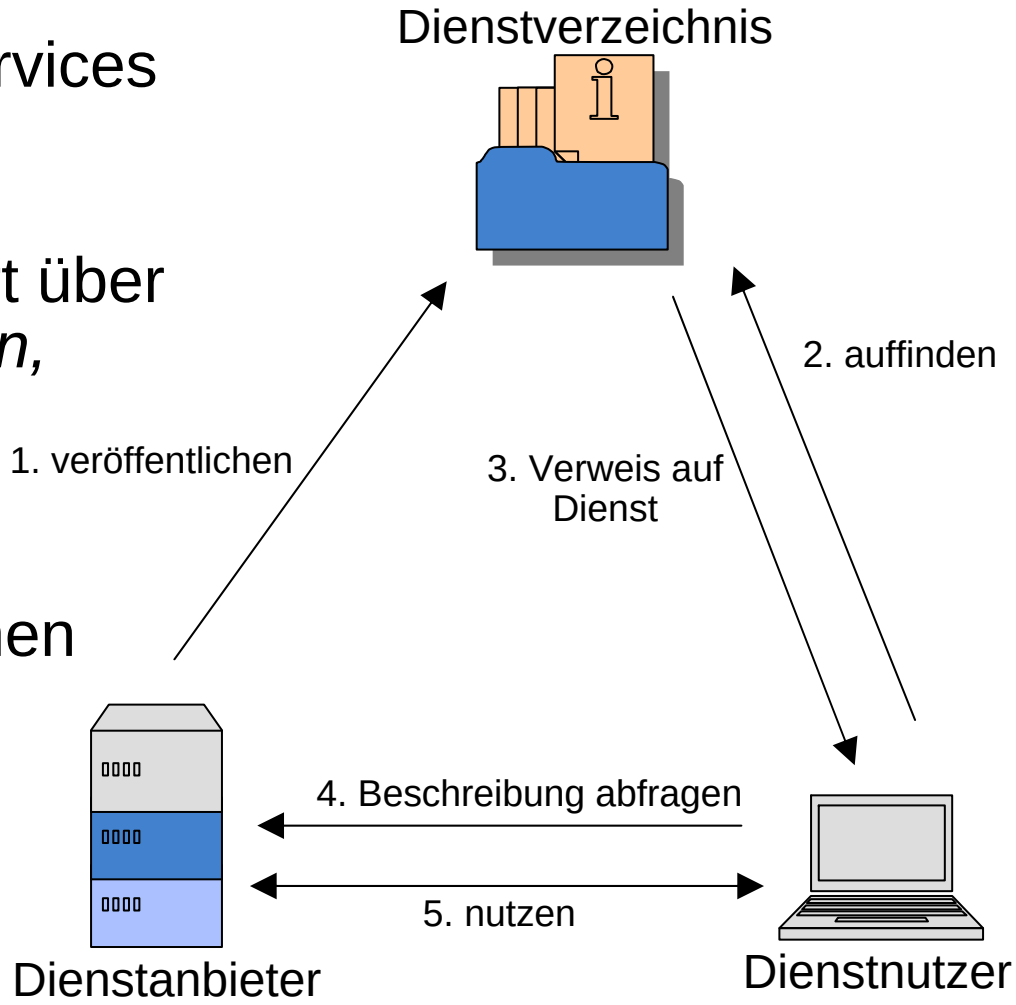
# Serviceorientierte Architektur

- Kapselung von Anwendungen als Dienst
- Zusammenfassen von Diensten als Geschäftsprozess
- Kostensenkung
- Wieder-verwendbarkeit
- Auslagern von Diensten



# SOA mit Web-Services

- Beschreibung von Web-Services durch WSDL (*Web-Service Description Language*)
- Verzeichnis Dienst realisiert über UDDI (*Universal Description, Discovery and Integration*)
- Zugriff auf Web-Services und Kommunikation zwischen Dienstanbieter und -nutzer über SOAP

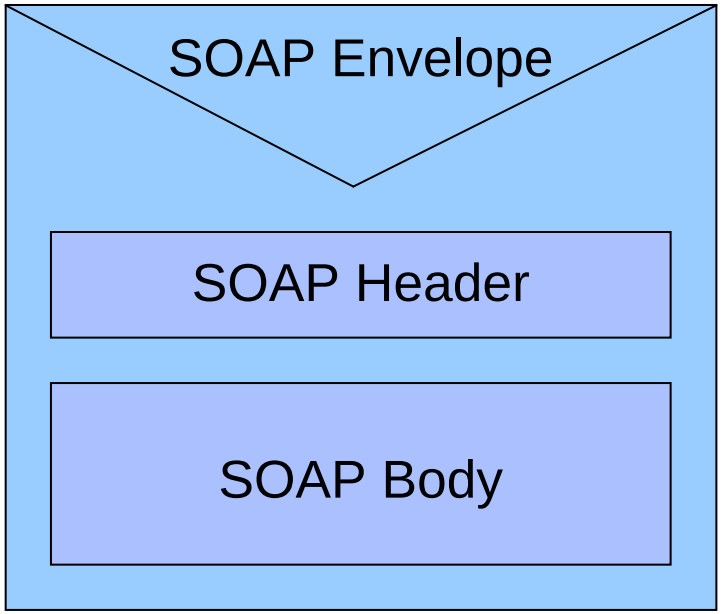


# UDDI-Verzeichnis

- Datenbank von WSDL-Beschreibungen verschiedener Dienste
- Untergliedert in vier Haupttabellen
  - White Pages
    - Auffinden eines Dienstes anhand eines Dienstanbieters/Unternehmens
  - Yellow Pages
    - Dienste nicht nach Anbietern sondern nach Geschäftsfeldern/Branchen geordnet
  - Green Pages
    - Auflisten von Dienstbeschreibungen wenn weder Anbieter noch Branche bekannt sind
  - Service Type Registration
    - Green Pages in maschinenlesbar Form

# SOAP

- SOAP (urspr. Simple Object Access Protocol)
- Transport über HTTP und TCP
  - Andere Kombinationen möglich
- XML Dokument bestehend aus 3 Teilen
  - SOAP Envelope
    - Verwendete SOAP Version
    - XML Wurzelelement
  - SOAP Header
    - Optional einmalig am Anfang einer Nachricht
  - SOAP Body
    - Enthält zu übertragende Information



# WSDL

- WSDL dient der Beschreibung von Web-Services
  - XML-basierte Grammatik zur Beschreibung von Diensten als eine Menge von Endpunkten
- Definition über 6 XML Hauptelemente unterteilt in
  - Abstrakte Definitionen
    - Types
      - Datenaustauschformate
    - Messages
    - PortTypes
      - One-way, Request-response, Solicit-response, Notification
  - Konkrete Definitionen:
    - Bindings
      - Legt verwendetes Protokoll für den Nachrichtenaustausch fest
    - Endpoints
      - Exakte Adresse eines Web-Services z.B.: eine URI
    - Services

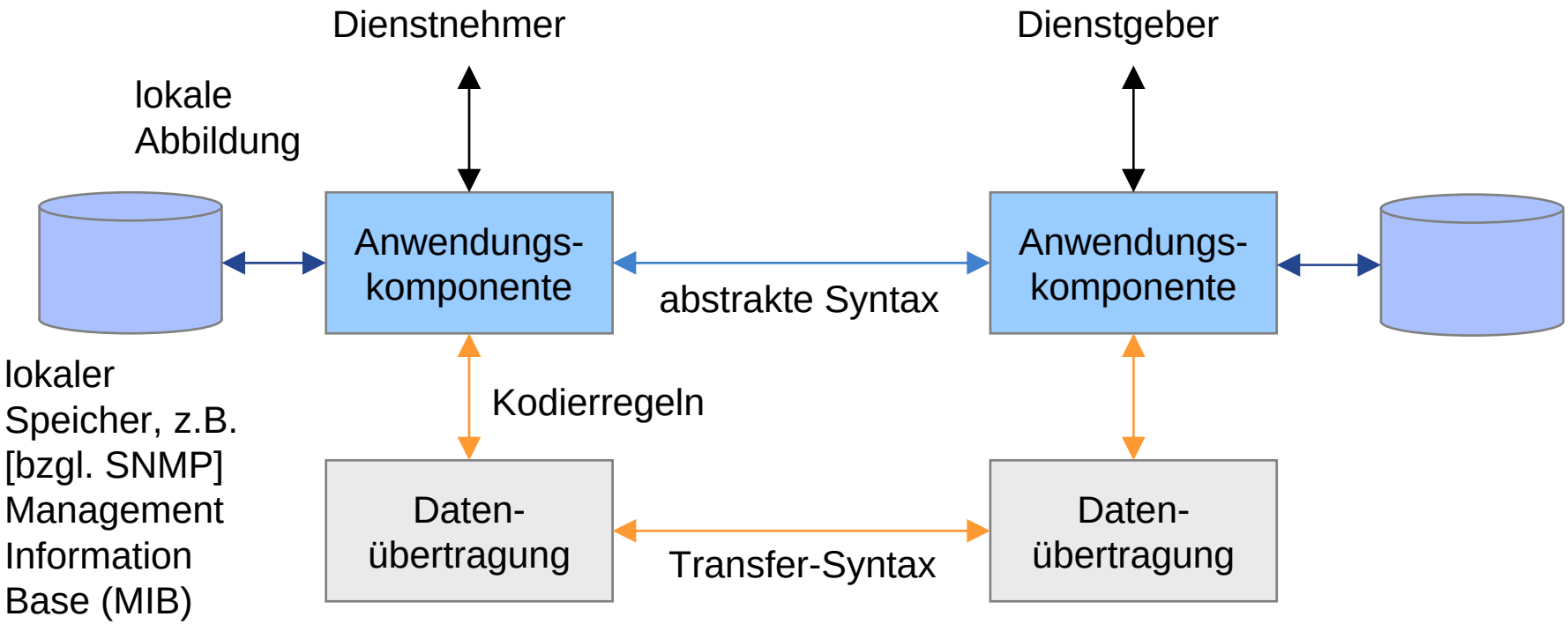
# Abstrakte Syntax und Transfersyntax

- Abstrakte Syntax
  - Menge von Typdefinitionen für Datenobjekte, die in einem Anwendungsprotokoll verwendet werden
  - Sie besagt, was dargestellt ist
  - ASN.1 ist eine Notation zur Definition einer abstrakten Syntax
    - Stark an der Programmiersprache C orientiert
    - Menschliche Lesbarkeit war bei der Entwicklung zweitrangig

- Transfersyntax
  - Konkrete Repräsentation der durch eine abstrakte Syntax beschriebenen Daten
  - Kodierregeln definieren Transformation zwischen abstrakter Syntax und Transfersyntax
  - Für eine abstrakte Syntax können mehrere Transfersyntaxen existieren
    - Normale Kodierung
    - Verschlüsselte Kodierung
    - Komprimierte Kodierung
  - Für ASN.1 sind bisher die Basic Encoding Rules (BER) als Kodierregeln standardisiert
    - Ziel: möglichst kompakte Darstellung, um Übertragungsaufwand zu minimieren

# Abstrakte Syntax und Transfersyntax

- Schema



# Abstract Syntax Notation One

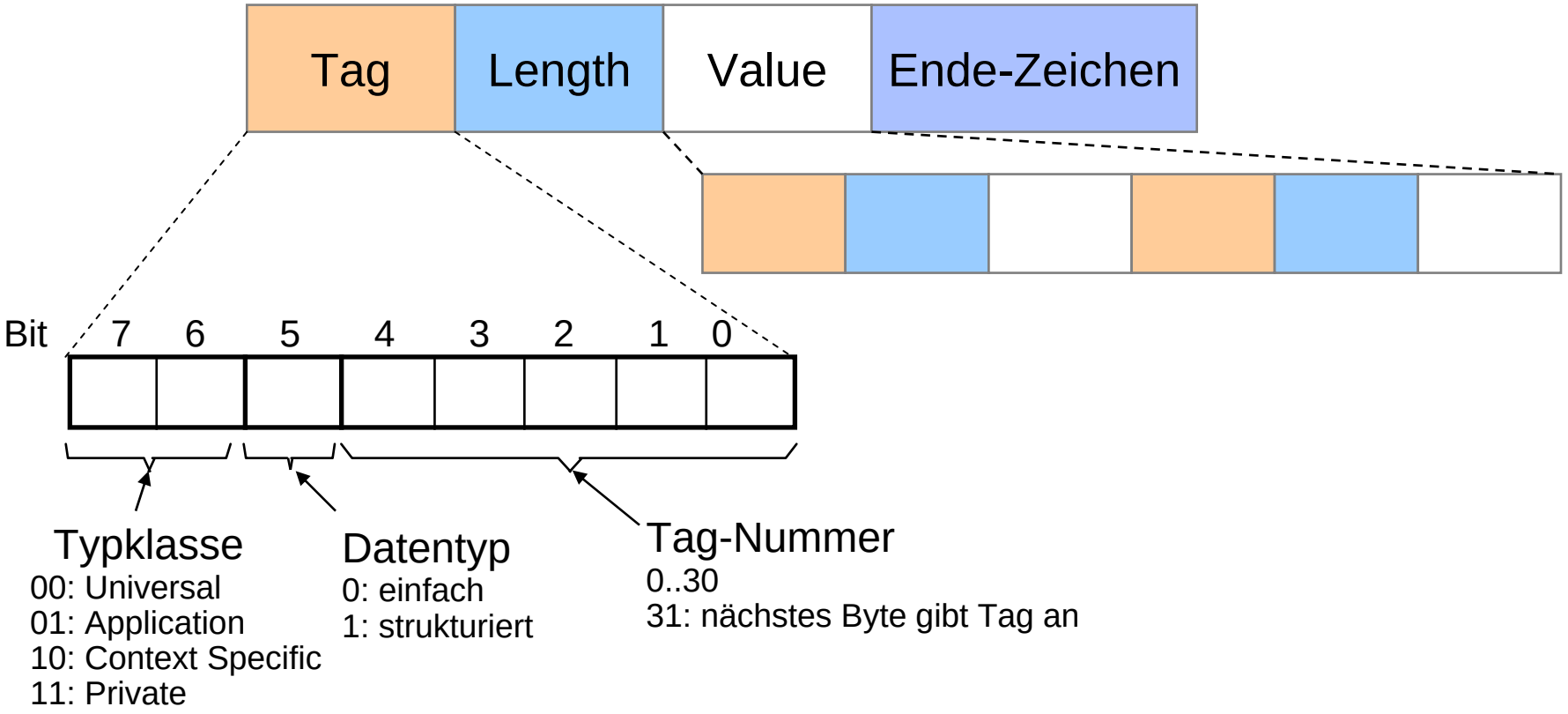
- Abstract Syntax Notation One (ASN.1)
  - Eine Sprache zur Spezifikation der Parameter von Benutzerdaten
  - Erlaubt die Spezifikation von Typen und Werten, ohne die Wertrepräsentation zu bestimmen
  - Angegeben als formale Grammatik in EBNF
- Kodierregeln
  - Spezifizieren die Abbildung der Benutzerdaten in die Form der zu übertragenden Daten
- Basic Encoding Rules (BER) für ASN.1
  - Eine spezielle Menge von Kodierregeln, die international standardisiert sind
- Verwendung unter anderem bei Netzmanagementsystemen (z.B. SNMP: Simple Network Management Protocol)
  - Datenrepräsentation via ASN.1
  - Datenaustausch via UDP/IP
- Trend geht bei neueren Protokollen zur Verwendung von XML als Spezifikations- und Beschreibungssprache

# Basic Encoding Rules (BER)

- Jede Kodierung eines Datenwertes umfasst gemäß BER (Basic Encoding Rules) die folgenden Teile
  - Tag
    - Ein oder mehrere Oktette, welche eine Kodierung für Datentyp-Klasse und Tag-Nummer enthalten
  - Länge
    - Gibt die Länge des Inhalts in Oktetten an oder kennzeichnet, dass ein spezielles Ende-Zeichen vorhanden ist
  - Ende-Zeichen
    - Ein Ende-Zeichen, bestehend aus zwei 0-Oktetten
    - Kann vorhanden sein
  - Inhalt
    - Hier wird der eigentliche Datenwert kodiert. Der Inhalt eines zusammengesetzten Typs wird in naheliegender Weise gebildet, z.B.
      - SequenceType: Der Dateninhalt ist eine Folge von Datenwerten korrespondierend zu den in der Definition aufgeführten Typen.
      - SequenceOfType: Der Dateninhalt besteht aus null oder mehreren Datenwerten zu dem angegebenen Typ.
      - ChoiceType: Der Dateninhalt ist gleich dem Dateninhalt zu dem ausgewählten Typ.
  - Anmerkung: Oktett (engl.: Octet) = Byte mit 8 Bits

# BER-Kodierung

- Grundlegender Aufbau




- Beispiel



# Längenfeld

- Bekannte Länge
  - Kurze Form [1 Byte]
    - Erstes Bit 0
    - Die anderer 7 Bits kodieren die Länge als eine binäre Ganzzahl
  - Lange Form [$n$ Bytes, $n > 1$]
    - Erstes Bit 1
    - Alle anderen Bits des ersten Bytes kodieren die Anzahl der Längenoktette
    - Alle anderen Bits aller folgenden Längenoktette kodieren die Länge als eine binäre Ganzzahl
- Unbekannte Länge
  - Kennung [1 Byte]
    - Erstes Bit 1
    - Alle anderen Bits 0
  - Wird bei der Kodierung zusammengesetzter Datentypen verwendet, bei denen die Länge nicht sofort verfügbar ist
    - Im SNMP nicht zugelassen

# Beispiele

- BOOLEAN: TRUE

| 00 | 0 | 00001 | 00000001 | 11111111 |
|---|---|---|---|---|

$= 1_{16}\ 1_{16}\ FF_{16}$

- INTEGER: 100

| 00 | 0 | 00010 | 00000001 | 01100100 |
|---|---|---|---|---|

$= 2_{16}\ 1_{16}\ 100_{10}$

- INTEGER: 256

| 00 | 0 | 00010 | 00000010 | 00000001 | 00000000 |
|---|---|---|---|---|---|

$= 2_{16}\ 2_{16}\ 256_{10}$

- BITSTRING: "0111110111"

| 00 | 0 | 00011 | 00000010 | 01111101 | 11000000 |
|---|---|---|---|---|---|

$= 3_{16}\ 2_{16}\ 0111110111_{2}$

- OCTET STRING: "abcd"

| 00 | 0 | 00100 | 00000100 | 01100001 | 01100010 | 01100011 | 01100100 |
|---|---|---|---|---|---|---|---|

$= 4_{16}\ \ 4_{16}\ \ 97_{10}\ \ 98_{10}\ 99_{10}\ 100_{10}$

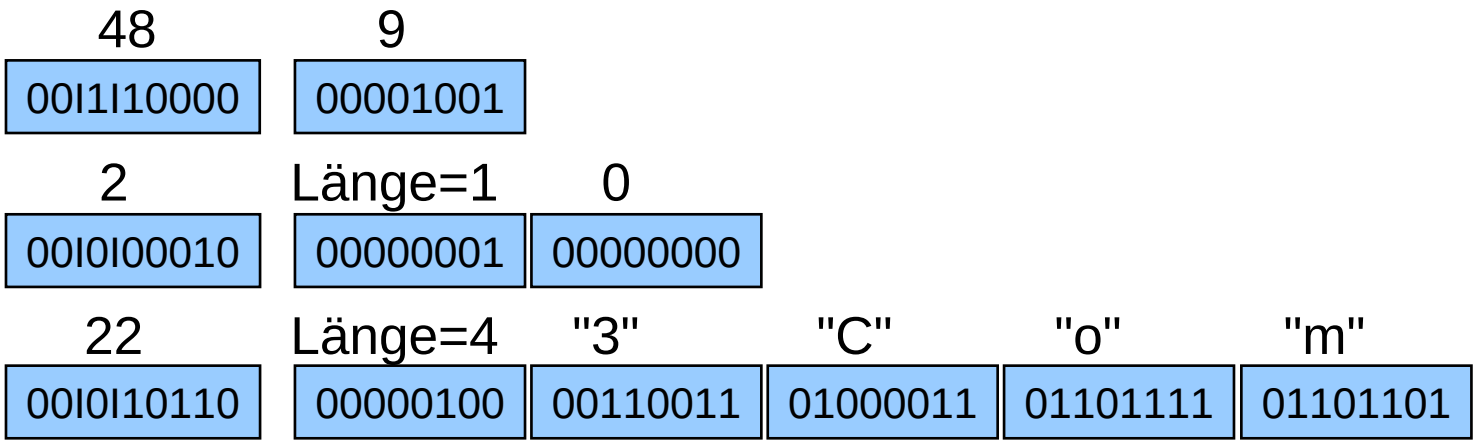
# Beispiele

- ASN.1 Sequence

```
interface :: = SEQUENCE {
                        index           INTEGER,
                        beschreibung    IA5STRING
                    }

LAN-Interface ::= {
                        index           0,
                        beschreibung    "3Com"
                    }
```

- BER Kodierung für `LAN-Interface`



# Zusammenfassung

- Middleware als „Vermittler“ zwischen Dienstnehmer und Dienstgeber
  - hat verschiedene Aufgaben
  - mit dem Ziel der Zusammenführung
- Entfernte Prozeduraufrufe (RPC)
  - Ablauf und Problembehandlung (Fehlersemantiken)
- Überwindung von Plattform-Heterogenität
  - Abstrakte Syntax abstrahiert von konkreter Programmiersprache
  - Transfersyntax definiert Regeln zur Übertragung der in abstrakter Syntax angegebenen Daten
  - Kodierregeln zur Transformation zwischen abstrakter Syntax und Transfersyntax

# Literatur

[Vino02] S. Vinoski; Where is Middleware?; IEEE Internet Computing; März/April 2002

[Come08] D. Comer; Computer Networks and Internets with Internet Applications; 5th edition; Prentice-Hall, 2008