

# **Informationssysteme: Verteilte und Web-Datenbanken**



**Prof. Alfons Kemper, Ph.D.**  
**Lehrstuhl für Informatik III:**  
**Datenbanksysteme**  
**TU München**  
**[www-db.in.tum.de](http://www-db.in.tum.de)**



# Literatur

- A. Kemper und A. Eickler: Datenbanksysteme -- eine Einführung. Oldenbourg Verlag, 9. Auflage, 2013.
  - Grundlagen VDBMS & XML & XQuery & Web Services & Internet-Datenbanken & P2P & MapReduce & Search Engine Basics (PageRank)
- L. Peterson und B. Davie: Computer Networks. Morgan Kaufmann 2012, 5. Aufl. Vertiefend zu Rechnerkommunikation (sehr gut)
- Serge Abiteboul et al: Web Data Management. Cambridge University Press 2011.<http://webdam.inria.fr/Jorge/files/wdm.pdf>
- P. Dadam: Verteilte Datenbanken und Client/Server Systeme. Springer Verlag, 1996
  - Vorlesungsbegleitend
- M.T. Ozsu und P. Valduriez: Principles of Distributed Database Systems. Springer Verlag, 3. Aufl., 2011.
  - Vorlesungsbegleitend
- E. Rahm: Mehrrechner-Datenbanksysteme. Addison Wesley, 1994.
  - Vorlesungsbegleitend
  - Vertiefend zu Parallelen Datenbanken
- S. Ceri und G. Pelagatti: Distributed Databases--Principles and Systems. Mc Graw Hill, 1984. alt aber ganz gut
- S. Abiteboul, P. Buneman und D. Suciu: Data on the Web, ... Morgan Kaufmann, 1999.
- Zusätzlich: Originalliteratur aus Konferenzen und Zeitschriften

# Gliederung

- Terminologie, Einführung
- Grundlagen der Datenkommunikation / Rechnernetze
- Klassische Verteilte Datenbanken
  - Datenbankentwurf
  - Anfragebearbeitung
  - Synchronisation
  - Fehlertoleranz
- P2P Informationssysteme
- Client/Server-Architektur
- Parallele Datenbanksysteme
- Web-Services (SOAP, UDDI, WSDL)
- Internet-Datenbanken
- Web Datenverwaltung (PageRank, MapReduce)

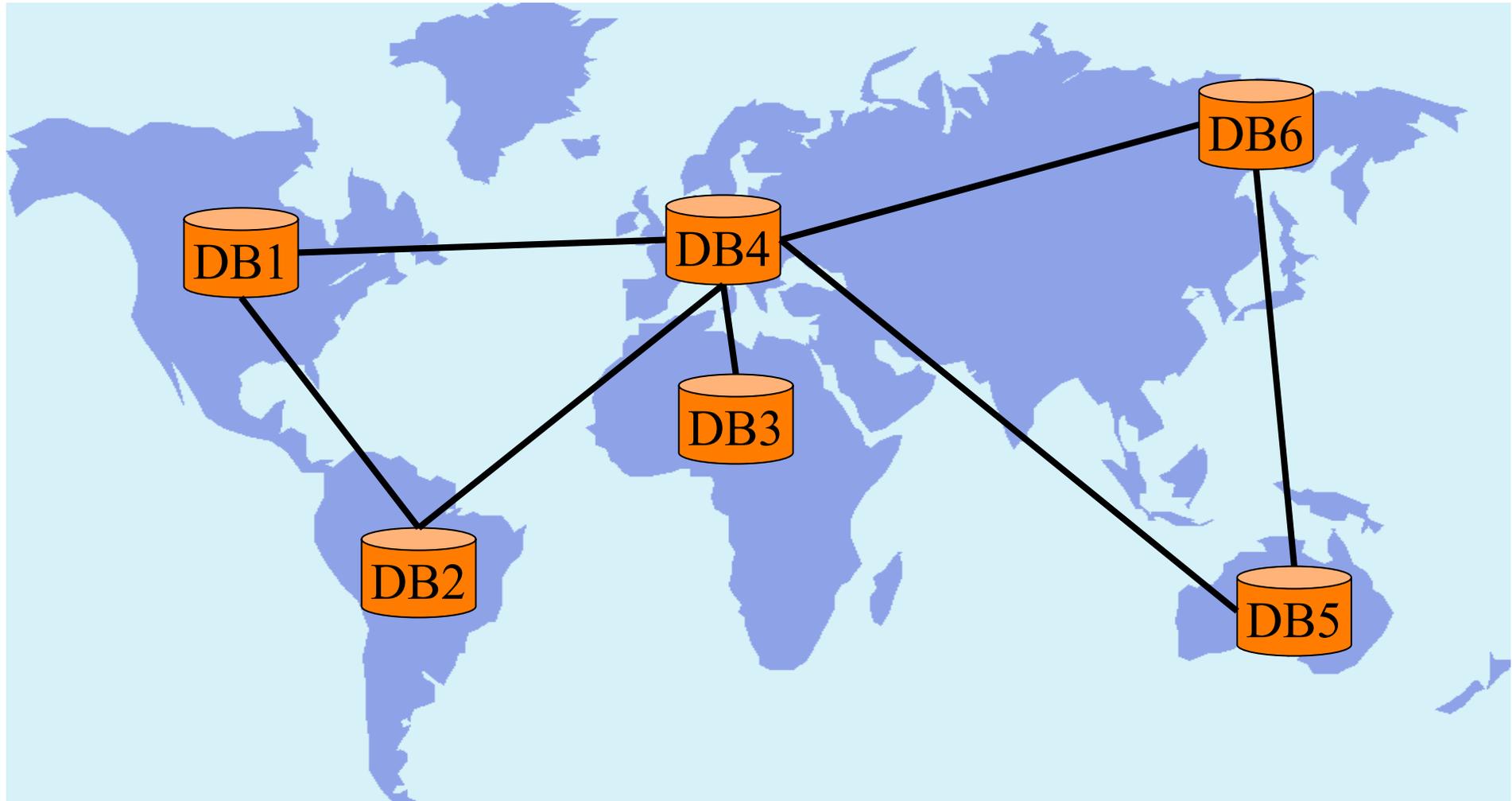
# Gegenläufige Trends

- Zentralisierung
  - ERP (Enterprise Resource Planning)
    - SAP, Baan, PeopleSoft, Oracle Applications
  - Data Warehouses
- Dezentralisierung
  - „Downsizing“ (1000 PCs sind billiger als ein Supercomputer)
  - Parallelisierung der Anwendungen
  - Durchsatzoptimierung
  - Service Oriented Architectures (SOA)
    - Web-Services
- Cloud Computing

# Integration von Datenbanksystemen/ Informationssystemen

- Autonome Einzelsysteme mit existierenden Anwendungen
- Heterogen bezüglich Hard- und Software
  - unterschiedliche Datenmodelle (Netzwerk/Codasyl, relational, hierarchisch)
  - unterschiedliche DBMS (Oracle, DB2 SQL Server, Informix, Sybase, ...)
- Ziel: Single-System-Image für neue globale Anwendungen
- Verfügbarkeit der lokalen Systeme soll erhalten bleiben

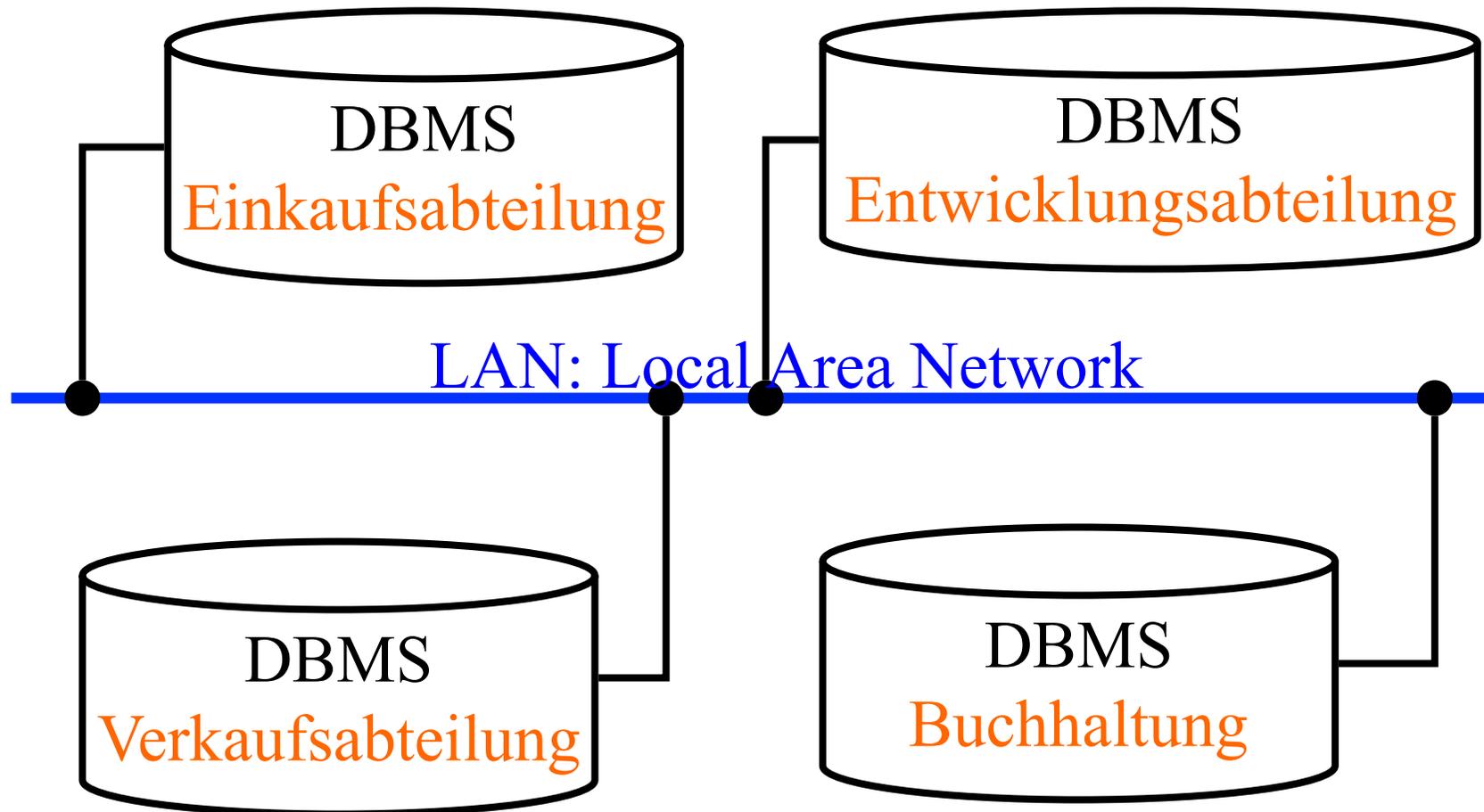
# Weiträumig verteilte Datenbanken



# Problemstellungen von weiträumig verteilten Informationssystemen

- Rechner physisch weit voneinander entfernt (keine Kontrolle)
- Kommunikation über „langsame“ Weitverkehrsnetze
- Dauerhafte Verfügbarkeit der lokalen Stationen nicht immer garantiert
- Ziel: hohe lokale Autonomie

# Lokal verteilte Datenbanken

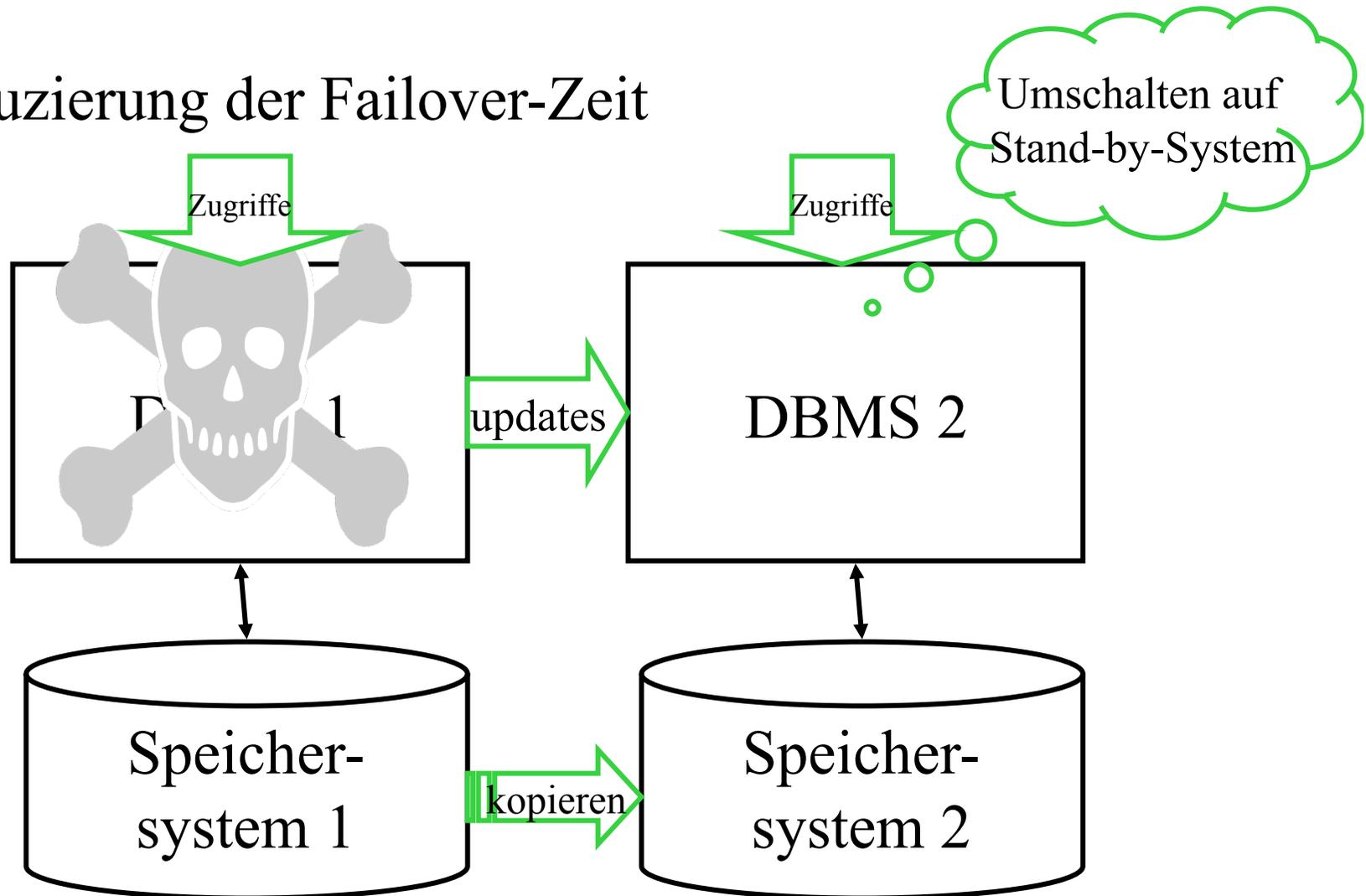


# Lokal Verteilte Datenbanksysteme

- Stationen sind durch ein „schnelles“ Netz (Local Area Network, LAN) miteinander verbunden
- Hoher Grad an Verfügbarkeit des Netzes
- Ausfallsicherheit durch Redundanz/Replikation
- Durchsatzsteigerung durch Parallelisierung
- Mehrrechner-DBMS (Parallele DBMS)
  - spezielle Variante
- ➡ P2P Informationssysteme
  - NoSQL Datenbanken

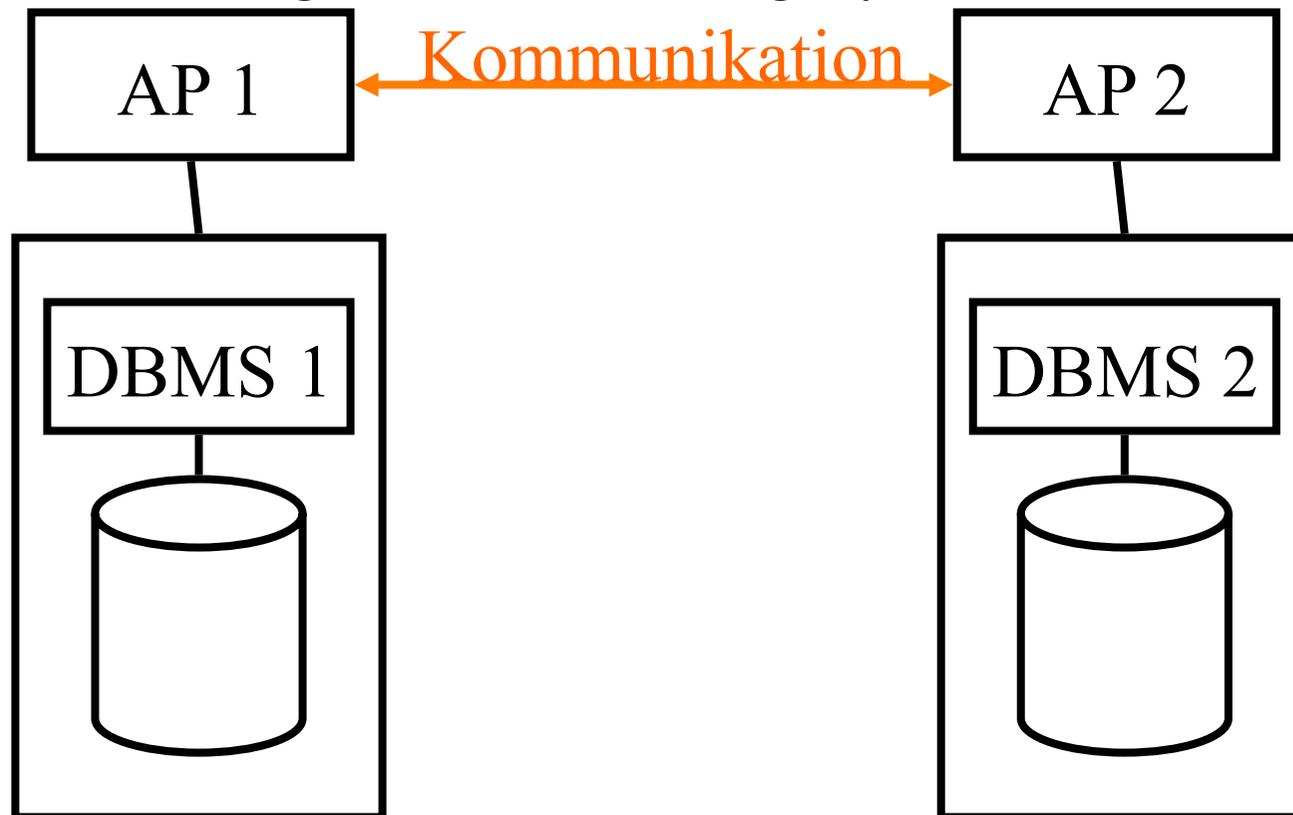
# „Gespiegelte“ Datenbank-Systeme: Ausfallsicherheit

- Reduzierung der Failover-Zeit



# Kommunikation in verteilten Informationssystemen

- Realisierung im Anwendungssystem

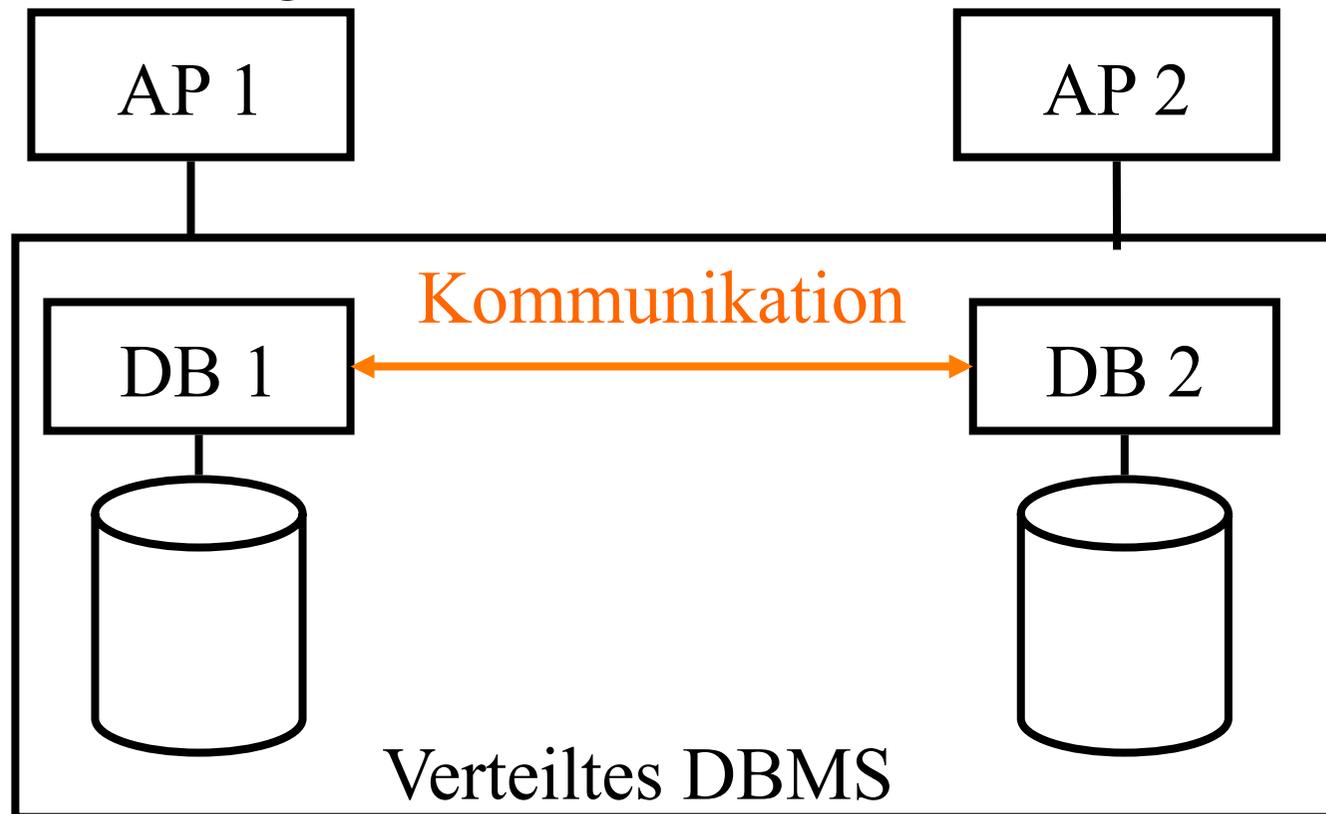


# **Konventionell realisierte verteilte Informationssysteme**

- Verteilung ist in den Anwendungen bekannt (keine Transparenz)
- Direkte Kommunikation zwischen den Anwendungen (Sockets, RPC, SOAP, RMI, o.ä.)
- Kein direkter Zugriff auf Daten des „anderen“ Anwendungssystems
- Fehlertoleranz muss von den Anwendern „programmiert“ werden

# Kommunikation in verteilten Informationssystemen

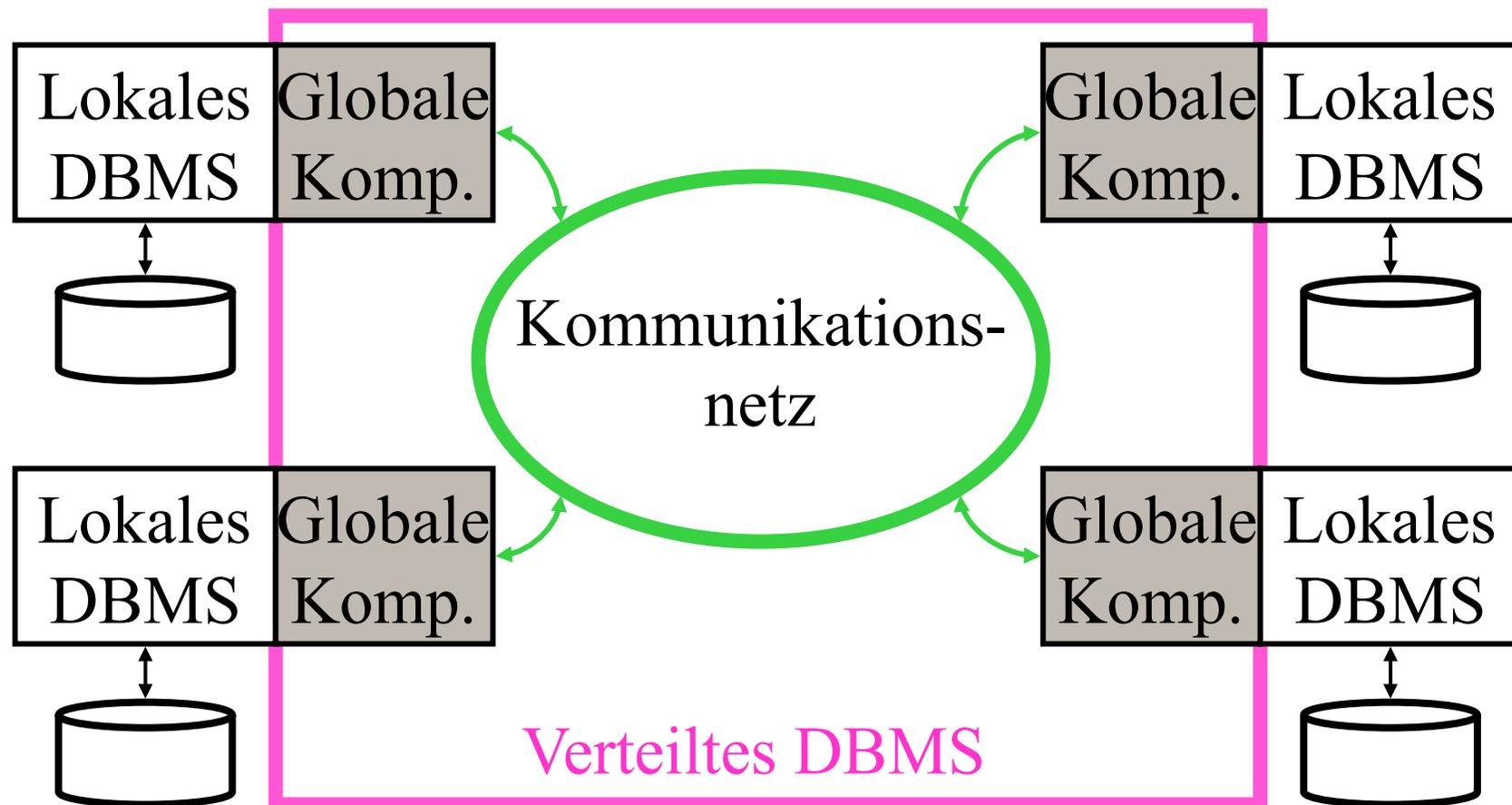
- Realisierung mittel verteiltem DBMS



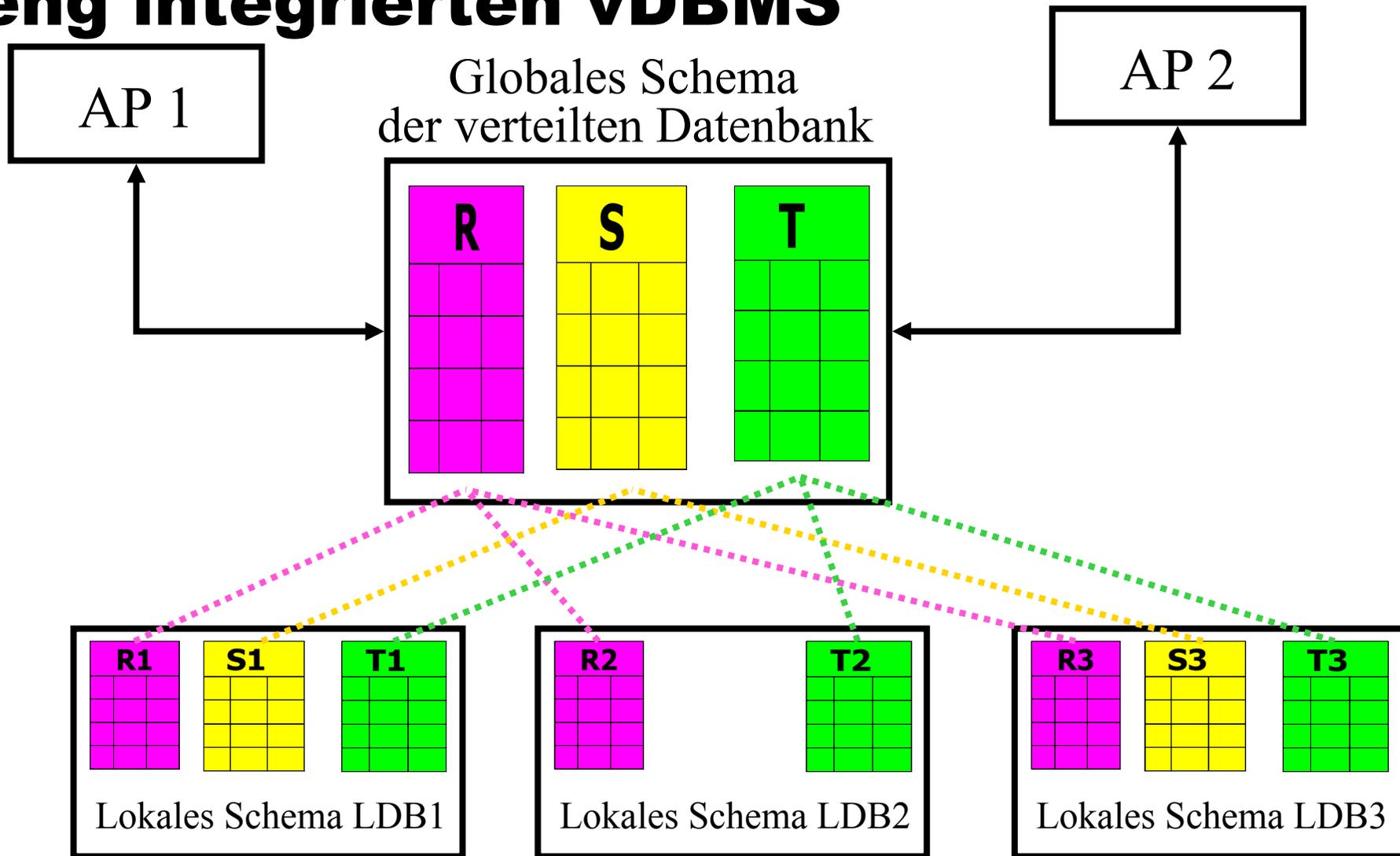
# **Echtes verteiltes Datenbanksystem**

- Verteilung wird durch DBMS realisiert
- Kommunikation ist implizit
- Verteilungstransparenz: Verteilung ist für die Anwendungssysteme nicht sichtbar
- Anwendungsprogramme kommunizieren über gemeinsame Datenbankobjekte

# Grobarchitektur eines verteilten Datenbanksystems



# Schema-Integration bei homogenen, eng integrierten vDBMS



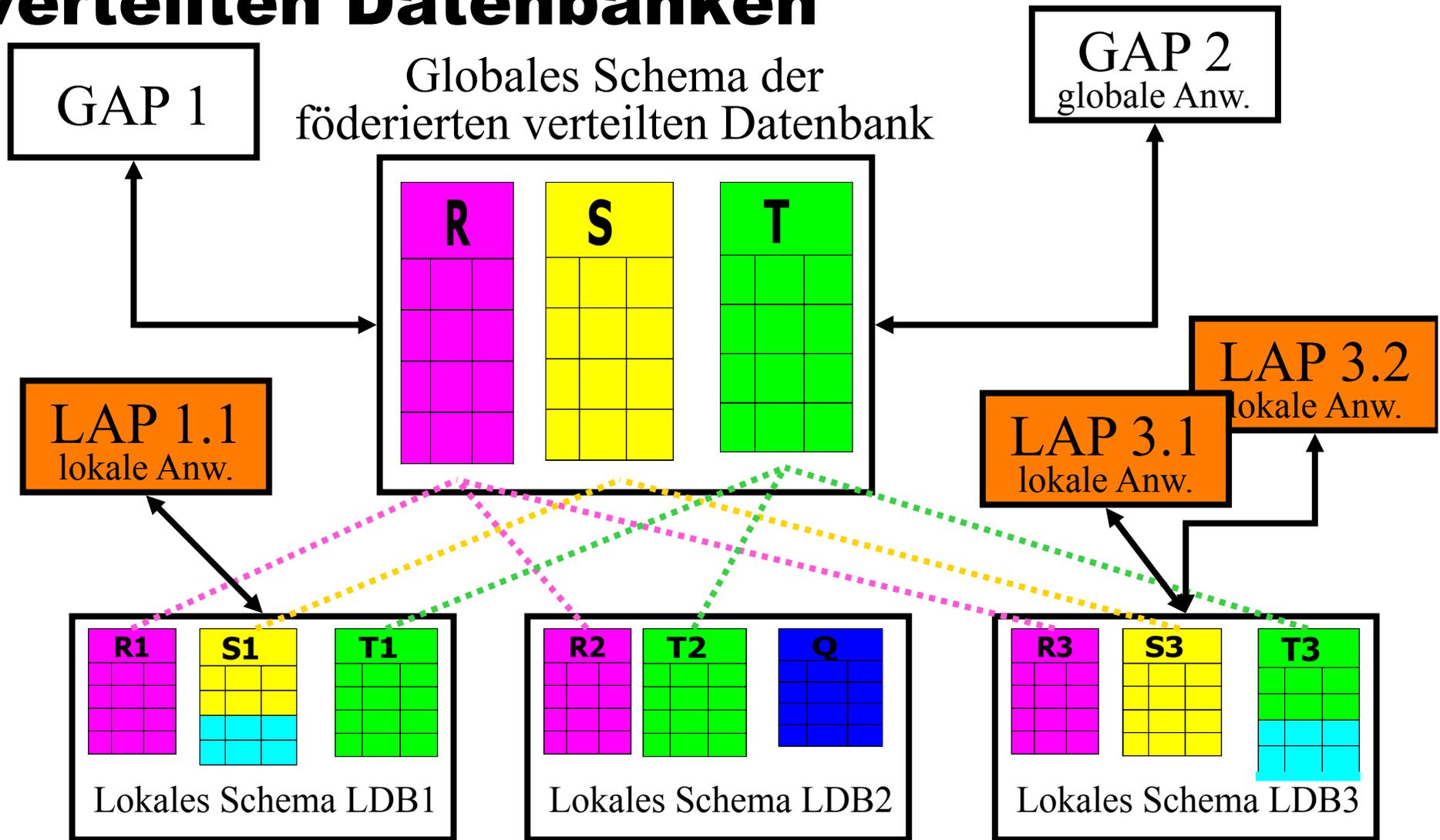
# Charakteristische Eigenschaften: Homogene, eng integr. vDBMS

<b><i>Charakteristische Eigenschaften</i></b>	<b><i>J/N</i></b>
<b>Daten physisch auf mehrere Knoten verteilt</b>	
<b>logische Sicht als eine Datenbank</b>	
<b>Verteilungstransparenz für Benutzer/Anwendungsprogramm</b>	
<b>gemischter DB-Zugang (global/lokal)</b>	
<b>Zerlegung globaler Anfragen durch vDBMS</b>	
<b>lokale Ausführung von Teilanfragen</b>	
<b>globales Transaktionskonzept</b>	
<b>lokale Autonomie beibehalten</b>	

# Charakteristische Eigenschaften: Homogene, eng integrier. vDBMS

<b><i>Charakteristische Eigenschaften</i></b>	<b><i>J/N</i></b>
<b>Daten physisch auf mehrere Knoten verteilt</b>	j
<b>logische Sicht als eine Datenbank</b>	j
<b>Verteilungstransparenz für Benutzer/Anwendungsprogramm</b>	j
<b>gemischter DB-Zugang (global/lokal)</b>	n
<b>Zerlegung globaler Anfragen durch vDBMS</b>	j
<b>lokale Ausführung von Teilanfragen</b>	j
<b>globales Transaktionskonzept</b>	j
<b>lokale Autonomie beibehalten</b>	n

# Schema-Integration bei föderierten verteilten Datenbanken



# Eigenschaften föderierter verteilter Datenbanken

- nicht alle Daten sind global sichtbar
- neben globalen Applikationen (GAP) gibt es auch lokale Applikationen (LAP)
- lokale Datenbanken behalten ihre Autonomie hinsichtlich Schemaerweiterungen und Schemaänderungen (solange keine globalen Schemata betroffen sind)
- man unterscheidet bzgl. homogenen und heterogenen lokalen Datenbanken

# Charakteristische Eigenschaften: Föderierte verteilte DBMS

<b><i>Charakteristische Eigenschaften</i></b>	<b><i>J/N</i></b>
<b>Daten physisch auf mehrere Knoten verteilt</b>	
<b>logische Sicht als eine Datenbank</b>	
<b>Verteilungstransparenz für Benutzer/Anwendungsprogramm</b>	
<b>gemischter DB-Zugang (global/lokal)</b>	
<b>Zerlegung globaler Anfragen durch vDBMS</b>	
<b>lokale Ausführung von Teilanfragen</b>	
<b>globales Transaktionskonzept</b>	
<b>lokale Autonomie beibehalten</b>	

# Charakteristische Eigenschaften: Föderierte verteilte DBMS

<b><i>Charakteristische Eigenschaften</i></b>	<b><i>J/N</i></b>
<b>Daten physisch auf mehrere Knoten verteilt</b>	j
<b>logische Sicht als eine Datenbank</b>	?
<b>Verteilungstransparenz für Benutzer/Anwendungsprogramm</b>	?
<b>gemischter DB-Zugang (global/lokal)</b>	j
<b>Zerlegung globaler Anfragen durch vDBMS</b>	j
<b>lokale Ausführung von Teilanfragen</b>	j
<b>globales Transaktionskonzept</b>	j
<b>lokale Autonomie beibehalten</b>	?

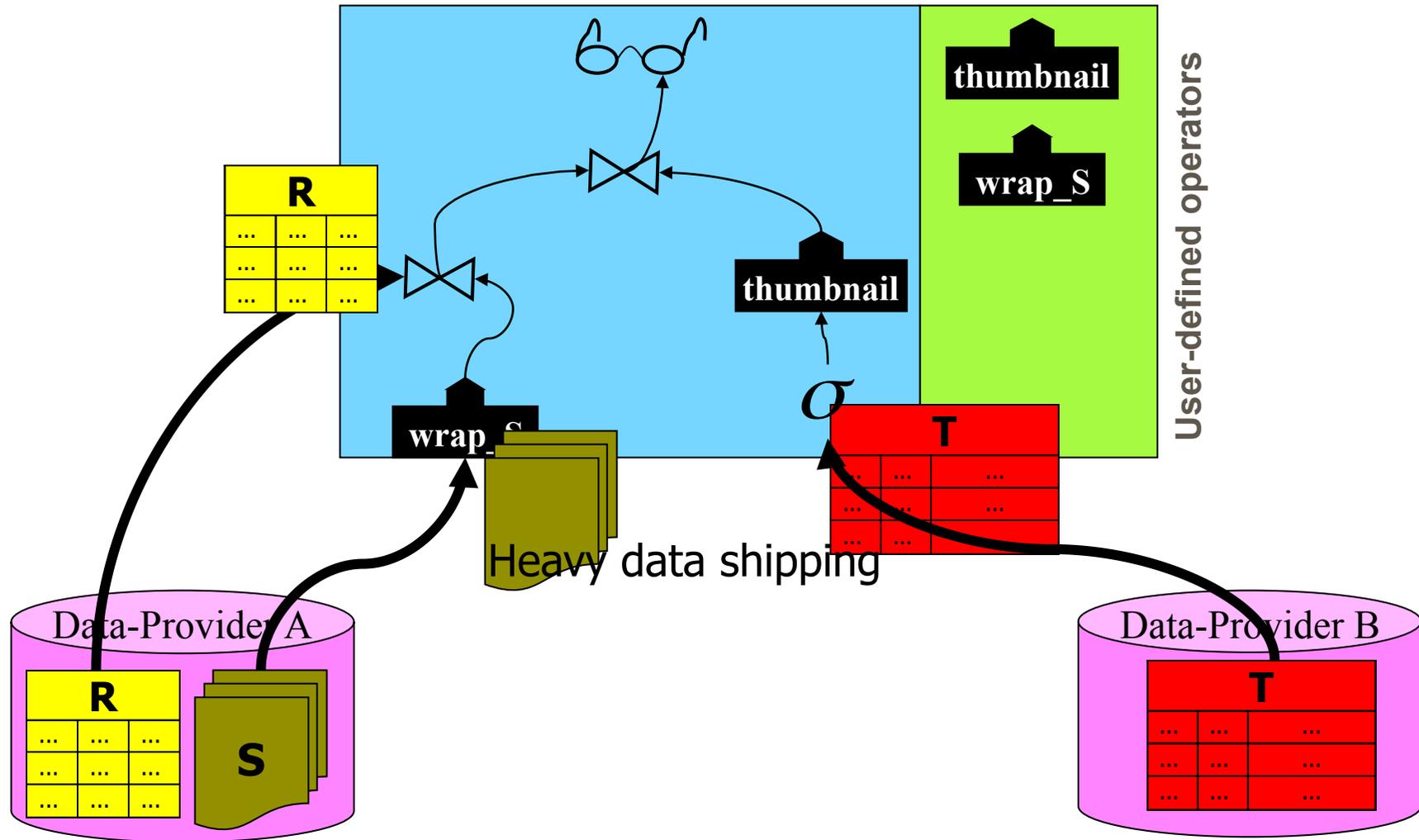
# Offene Multi-Datenbanksysteme

- À posteriori Integration von Datenbanksysteme
- Zugriff auf „fremde“ Datenbanken
  - Hotelreservierungssysteme
  - Flugreservierungssysteme
  - Literatur-Datenbanken
- Wegen der Bedeutung des WWW praktisch sehr relevant

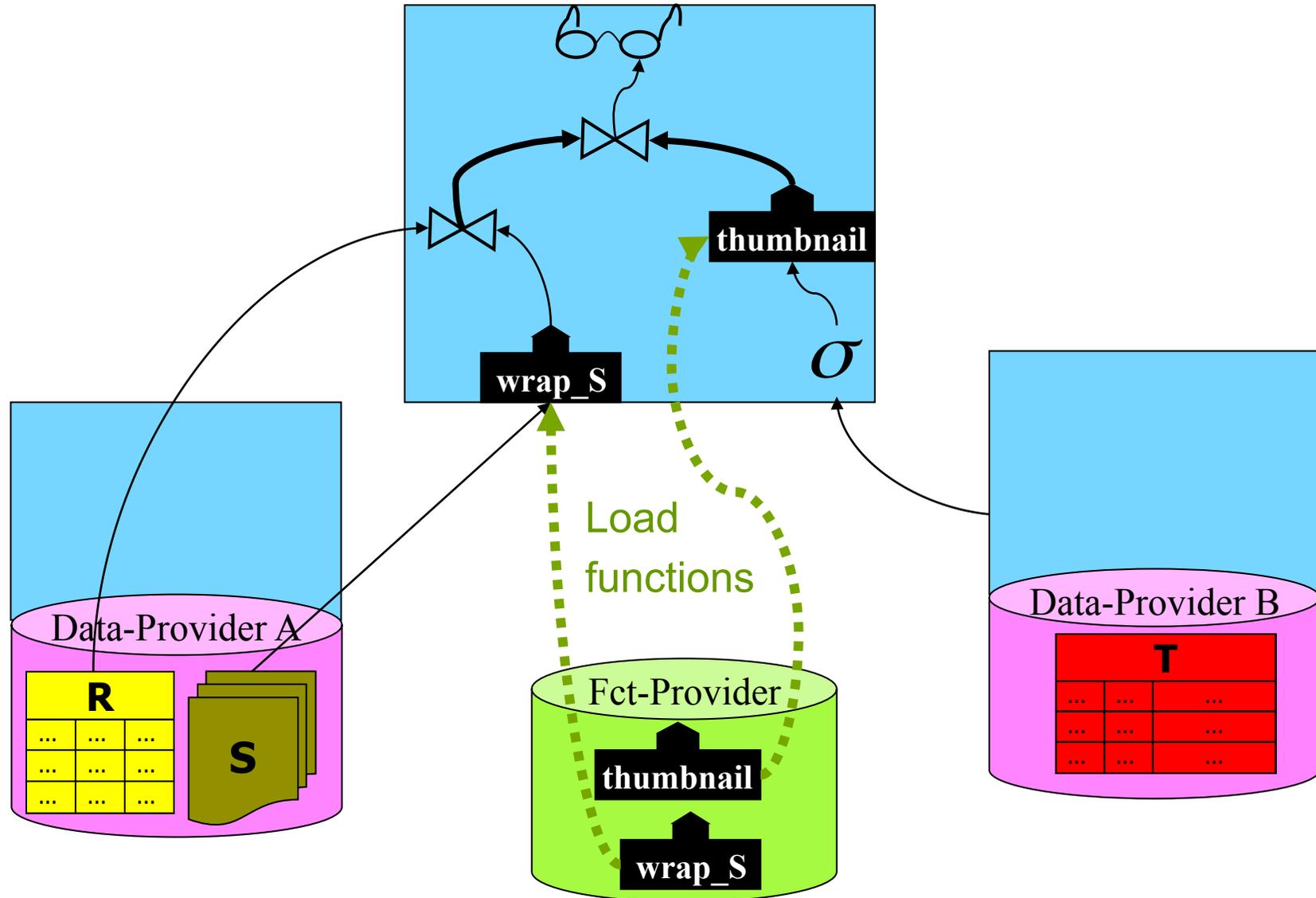
# Offene Multi-Datenbanksysteme (2)

- hoher, vollständiger Grad an Autonomie der Einzelsysteme
- geringe Kooperation der beteiligten Systeme
  - keine globale Transaktionsverwaltung
  - black-box-Systeme
- Atomarität globaler Transaktionen?
  - Kompensation anstelle „Undo“
  - zB Stornierung einer Buchung
- Schema-Integration über sogenannte Middlewaresysteme und Wrapper

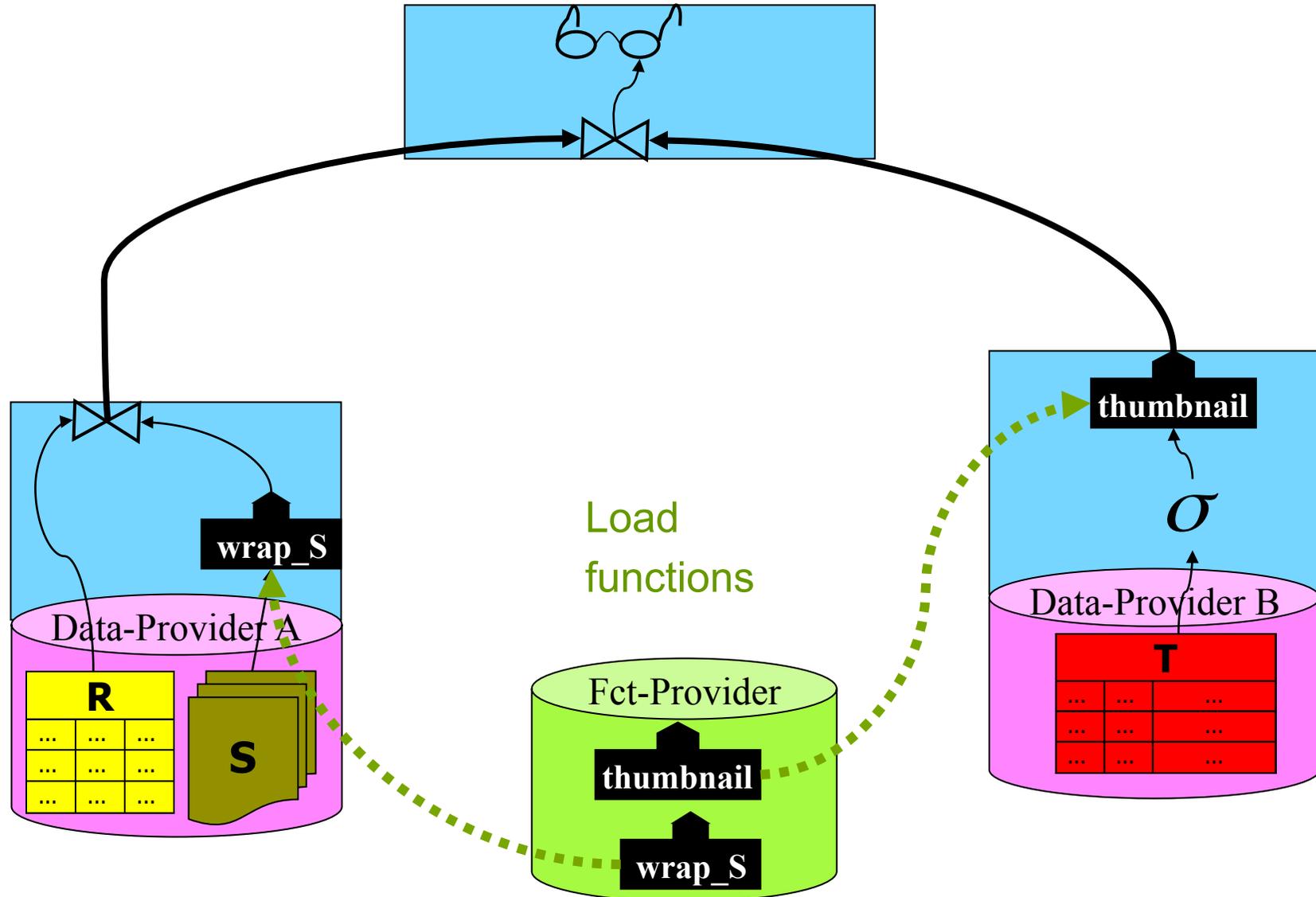
# Anfragebearbeitung im Middleware System



# Goal: Ubiquitous, Open Query Processing Capabilities



# ObjectGlobe: Ubiquitous, Open Query Processing Capabilities



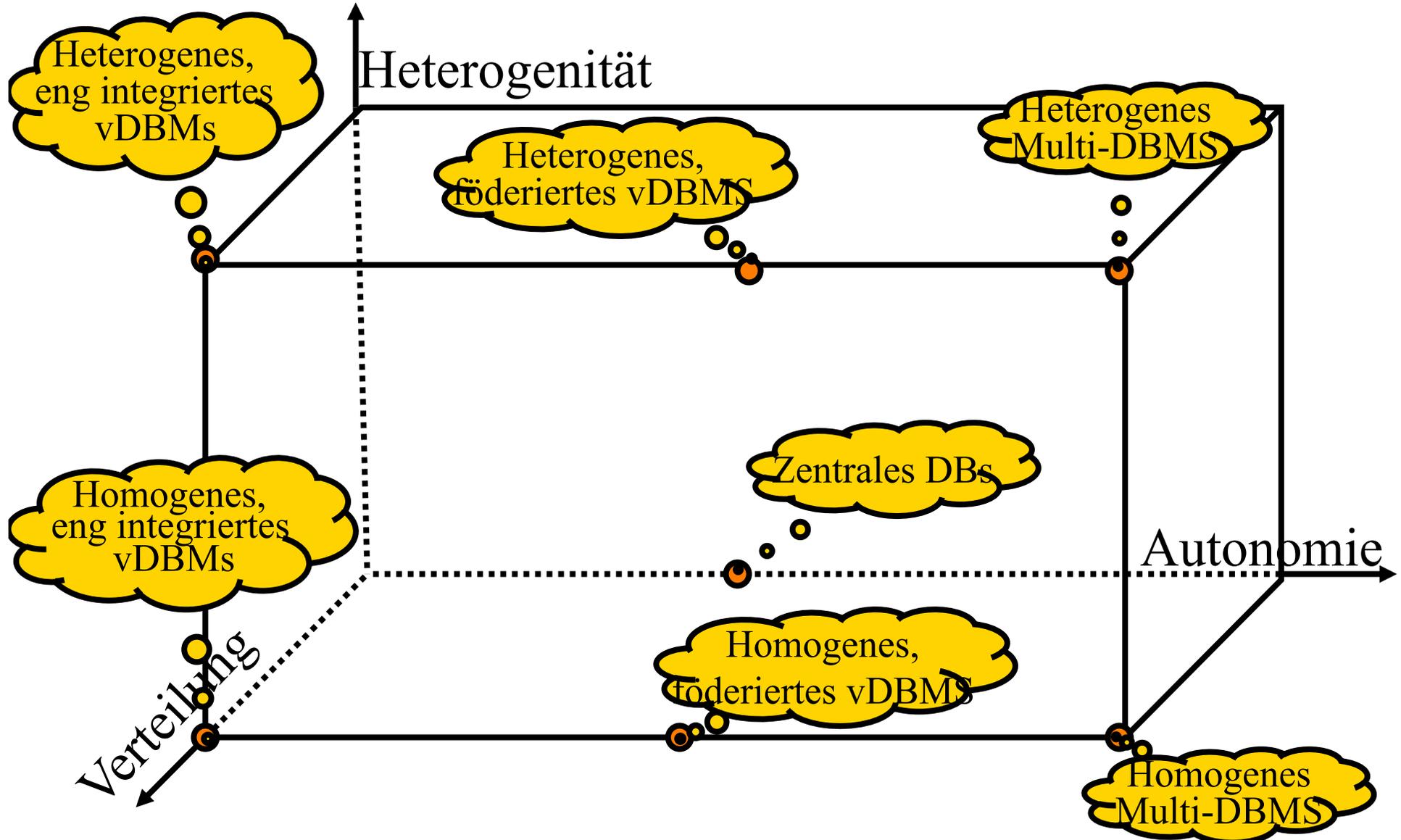
# Charakteristische Eigenschaften: Offene Multi-Datenbanksysteme

<b><i>Charakteristische Eigenschaften</i></b>	<b><i>J/N</i></b>
<b>Daten physisch auf mehrere Knoten verteilt</b>	
<b>logische Sicht als eine Datenbank</b>	
<b>Verteilungstransparenz für Benutzer/Anwendungsprogramm</b>	
<b>gemischter DB-Zugang (global/lokal)</b>	
<b>Zerlegung globaler Anfragen durch vDBMS</b>	
<b>lokale Ausführung von Teilanfragen</b>	
<b>globales Transaktionskonzept</b>	
<b>lokale Autonomie beibehalten</b>	

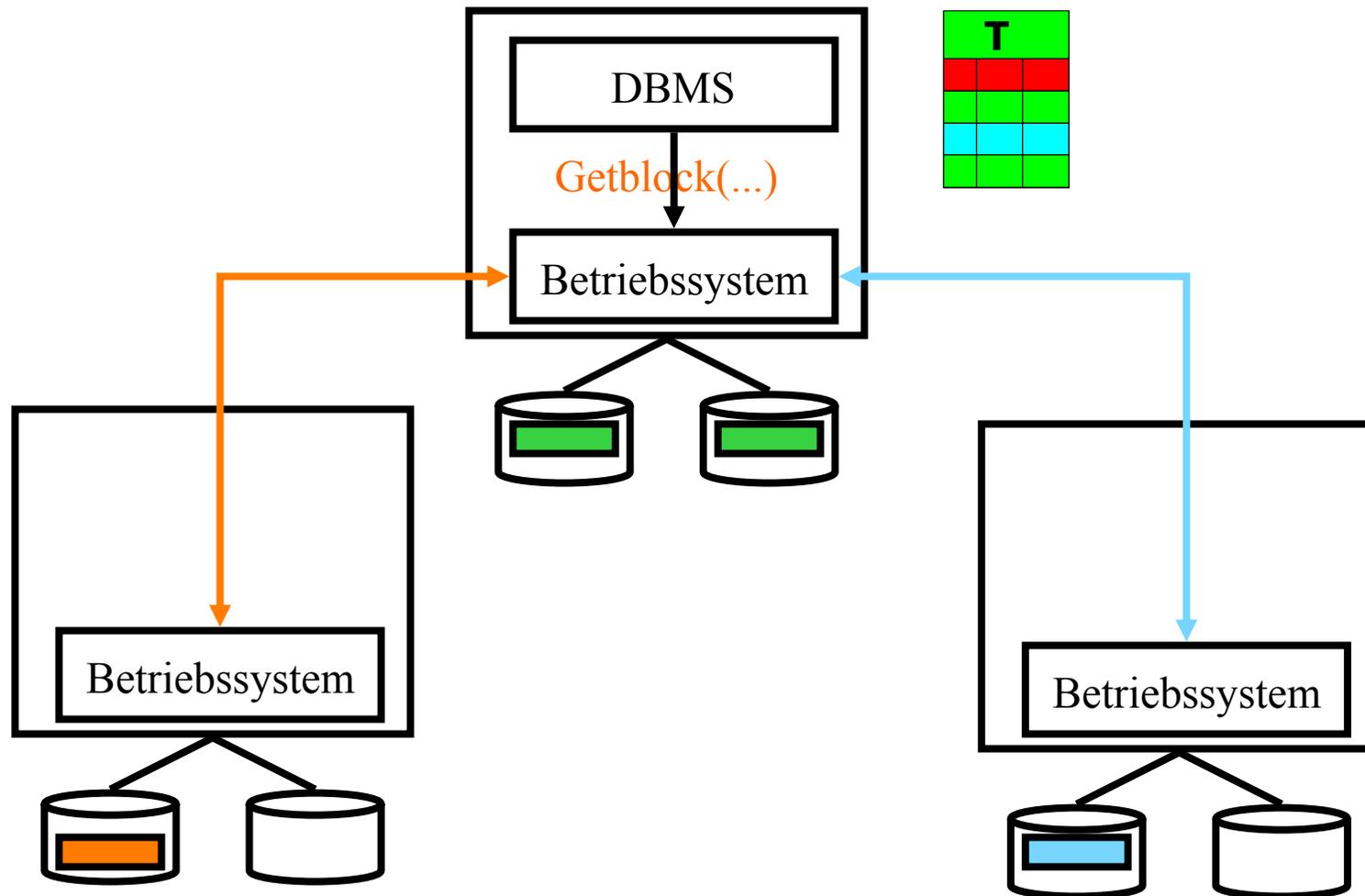
# Charakteristische Eigenschaften: Offene Multi-Datenbanksysteme

<b><i>Charakteristische Eigenschaften</i></b>	<b><i>J/N</i></b>
<b>Daten physisch auf mehrere Knoten verteilt</b>	j
<b>logische Sicht als eine Datenbank</b>	n
<b>Verteilungstransparenz für Benutzer/Anwendungsprogramm</b>	n
<b>gemischter DB-Zugang (global/lokal)</b>	n
<b>Zerlegung globaler Anfragen durch vDBMS</b>	n
<b>lokale Ausführung von Teilanfragen</b>	j
<b>globales Transaktionskonzept</b>	n
<b>lokale Autonomie beibehalten</b>	j

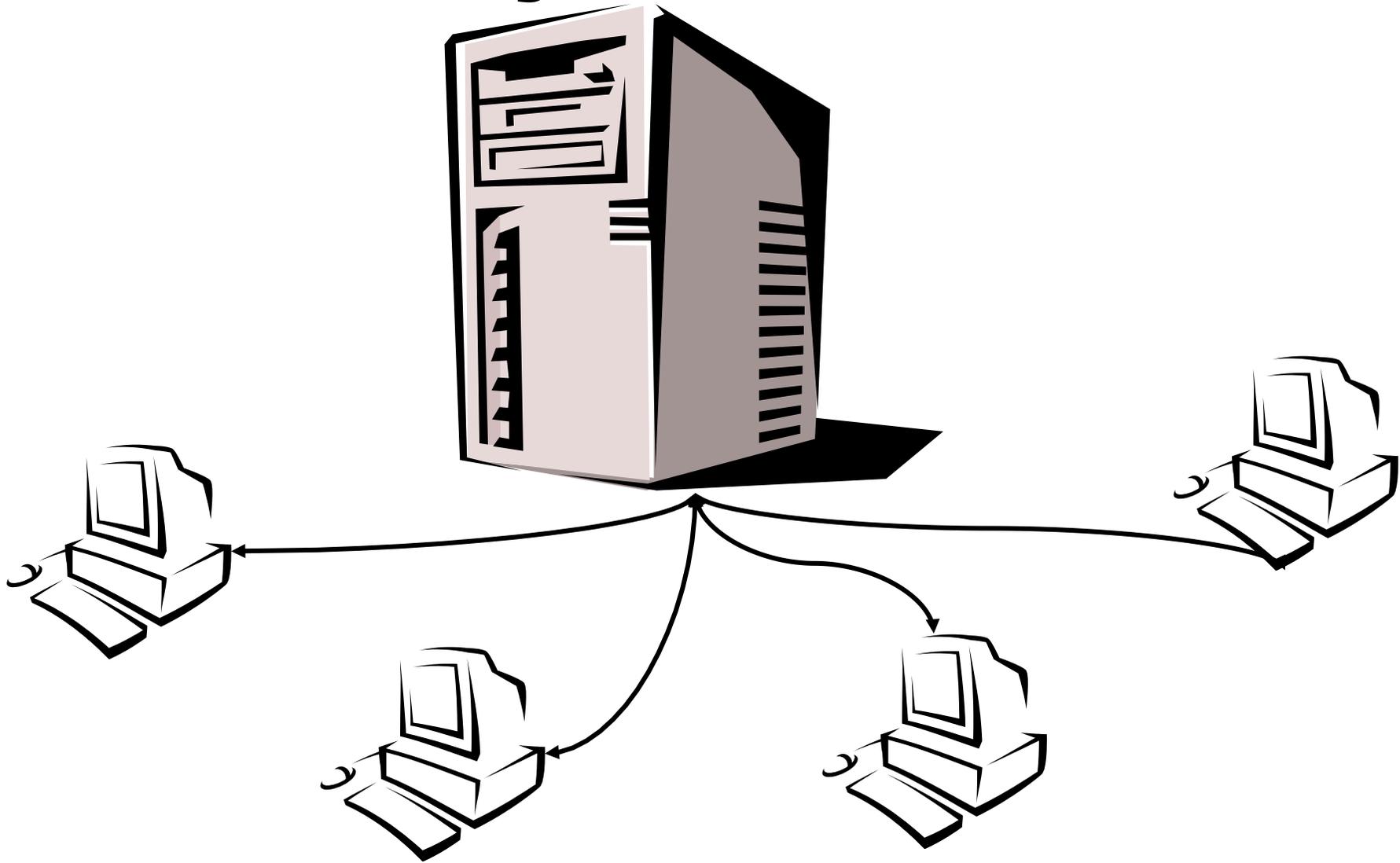
# Klassifikation verteilter DBMS



# Abgrenzung: Verteilte Datenbank - Verteiltes Dateisystem



# Client/Server- Datenbanksysteme



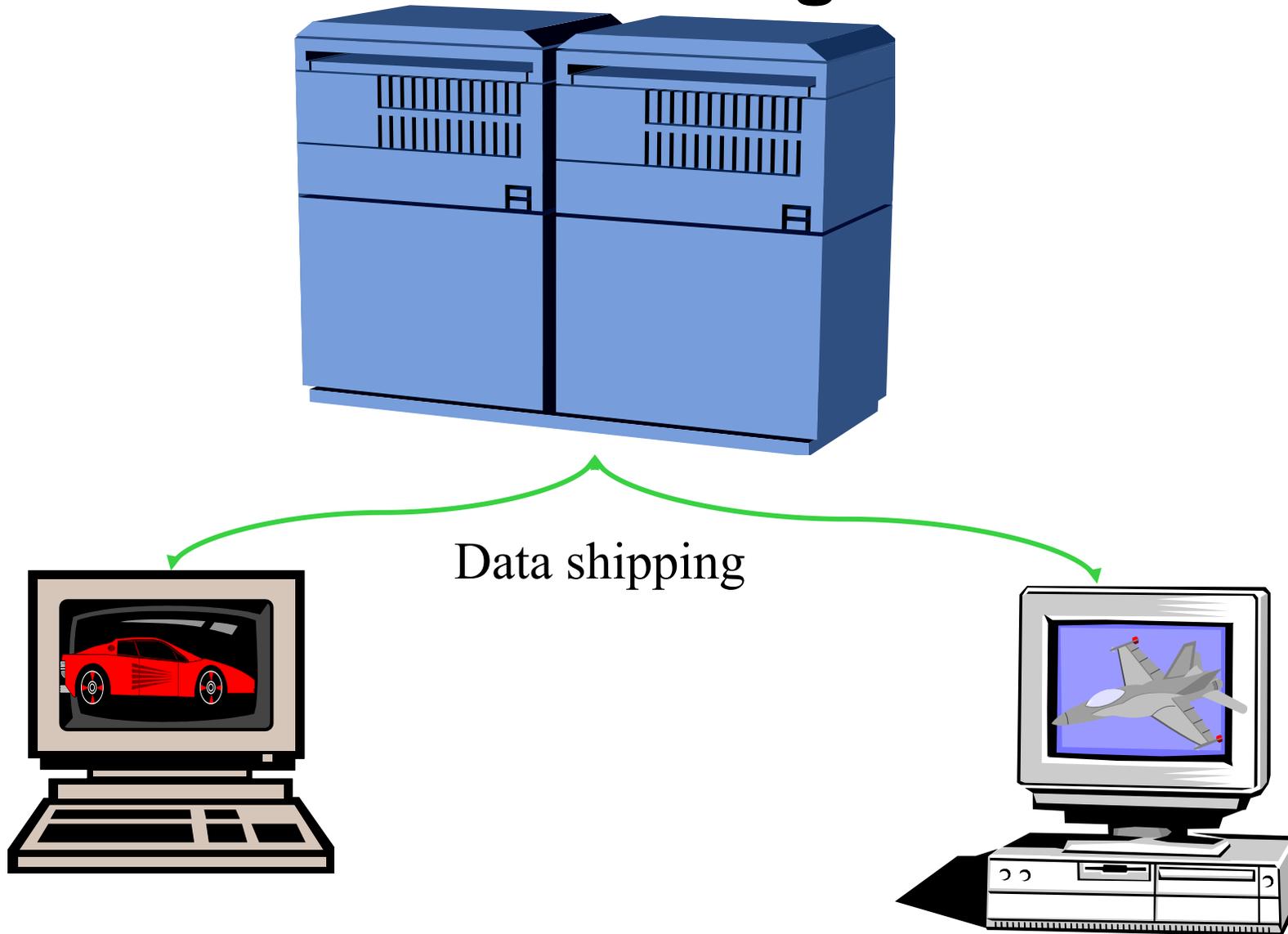
# Unterschiedliche CS-Ausprägungsformen

- Query Shipping
  - Clients haben fast ausschließlich Präsentationsfunktion
  - Server leistet die Arbeit
  - Stored Procedure
  - hauptsächlich in betriebswirtschaftlichen Anwendungen

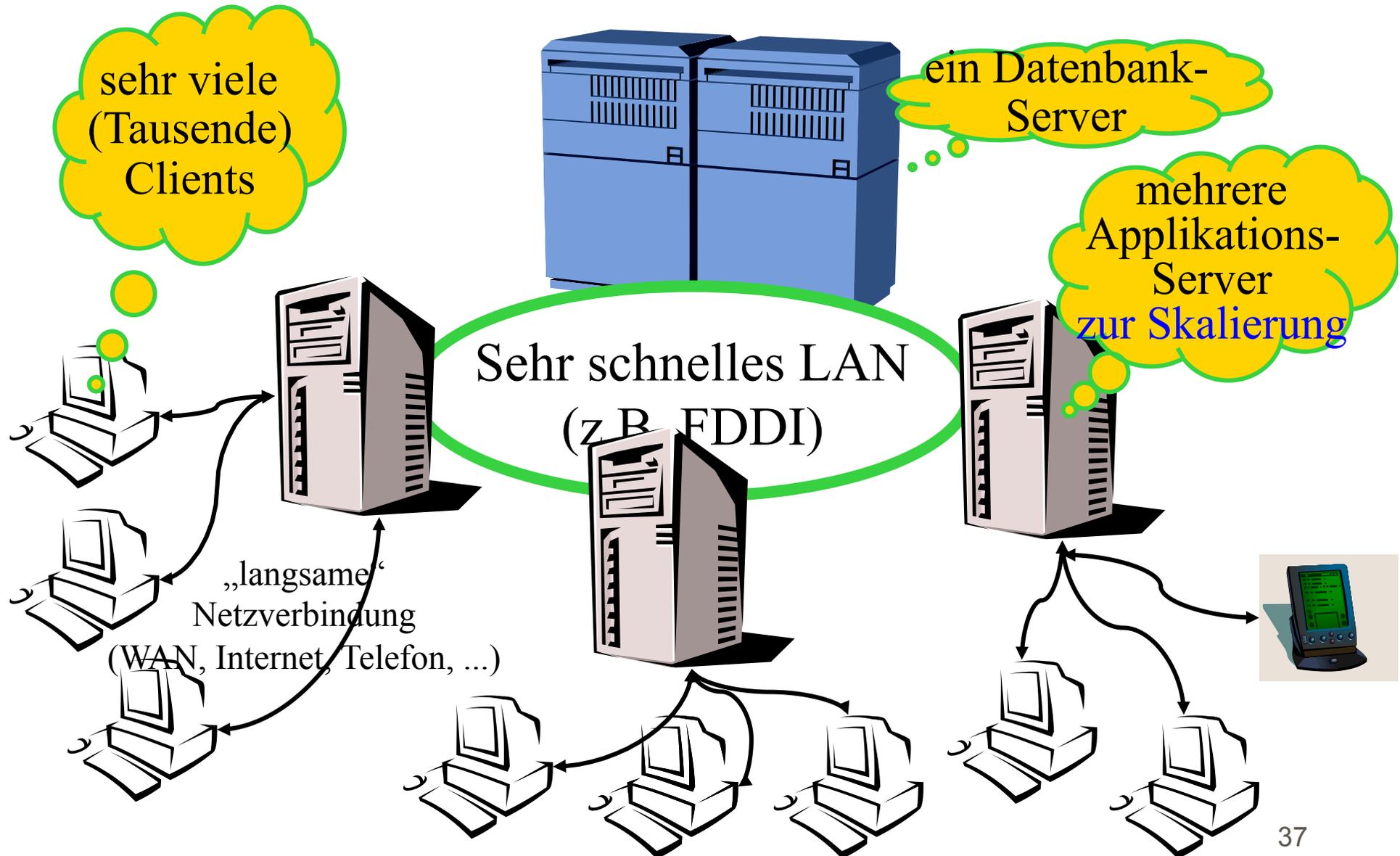
# Unterschiedliche CS- Ausprägungsformen (2)

- Data Shipping
  - Server liefern die Daten
  - Clients verarbeiten die Daten
  - hauptsächlich in ingenieurwissenschaftlichen Anwendungen
  - Clients sind zB CAD-Workstations

# Client/Server: Ingenieurwissenschaftliche Anwendungen



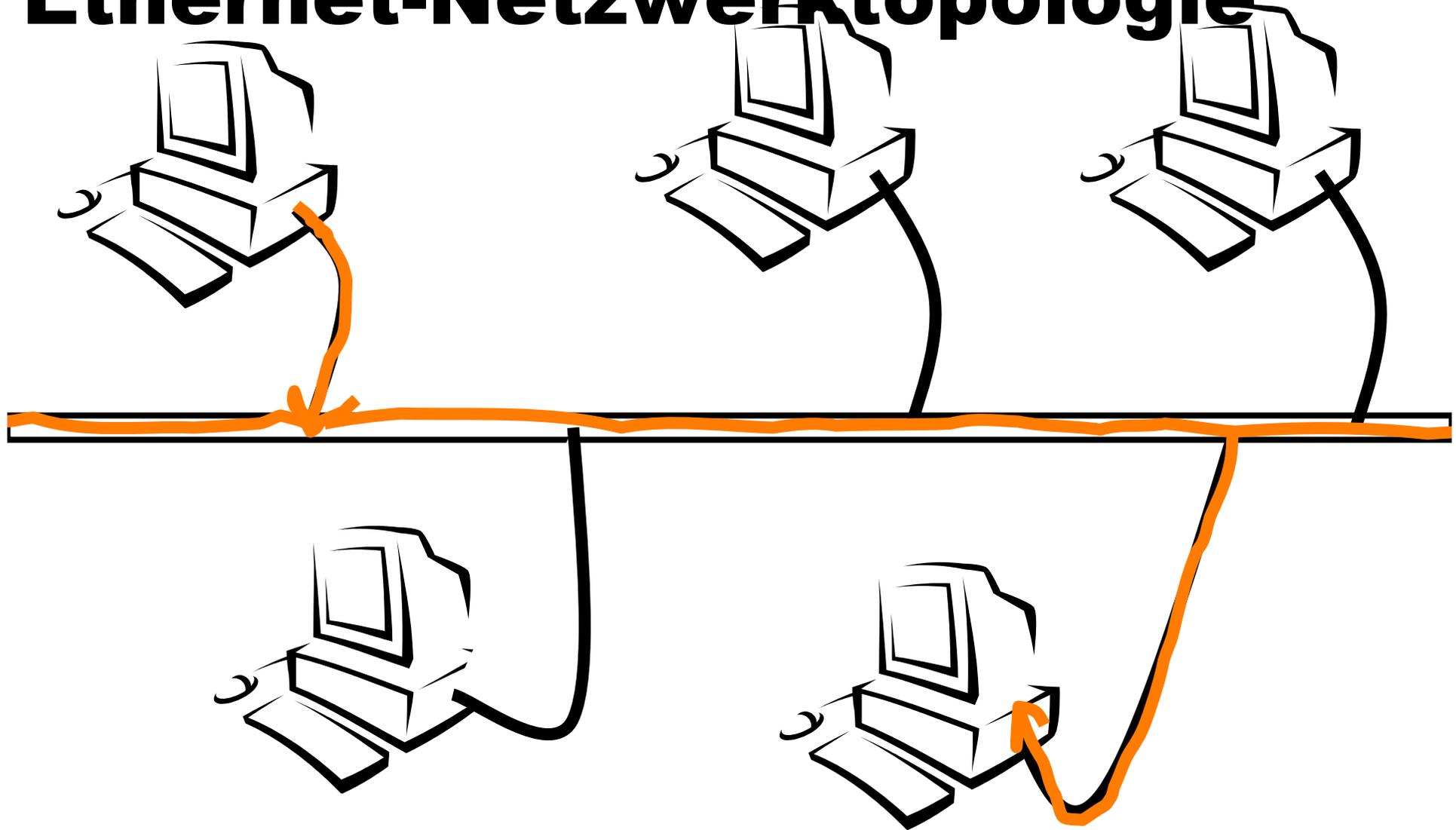
# Dreistufige Client/Server-Architektur (3 Tier, SAP R/3)



# Überblick: Computernetze

- L. Peterson und B. Davie: Computer Networks. Morgan Kaufmann, **5. Auflage 2012** (gibt es aber schon!).
- Local Area Networks (LAN)
  - **Ethernet**
  - Token Ring
  - FDDI
  - ATM
- Internet
  - IP
  - UDP
  - TCP
- Leistungs-Kennzahlen
- Übung: Netzwerk-Programmierung

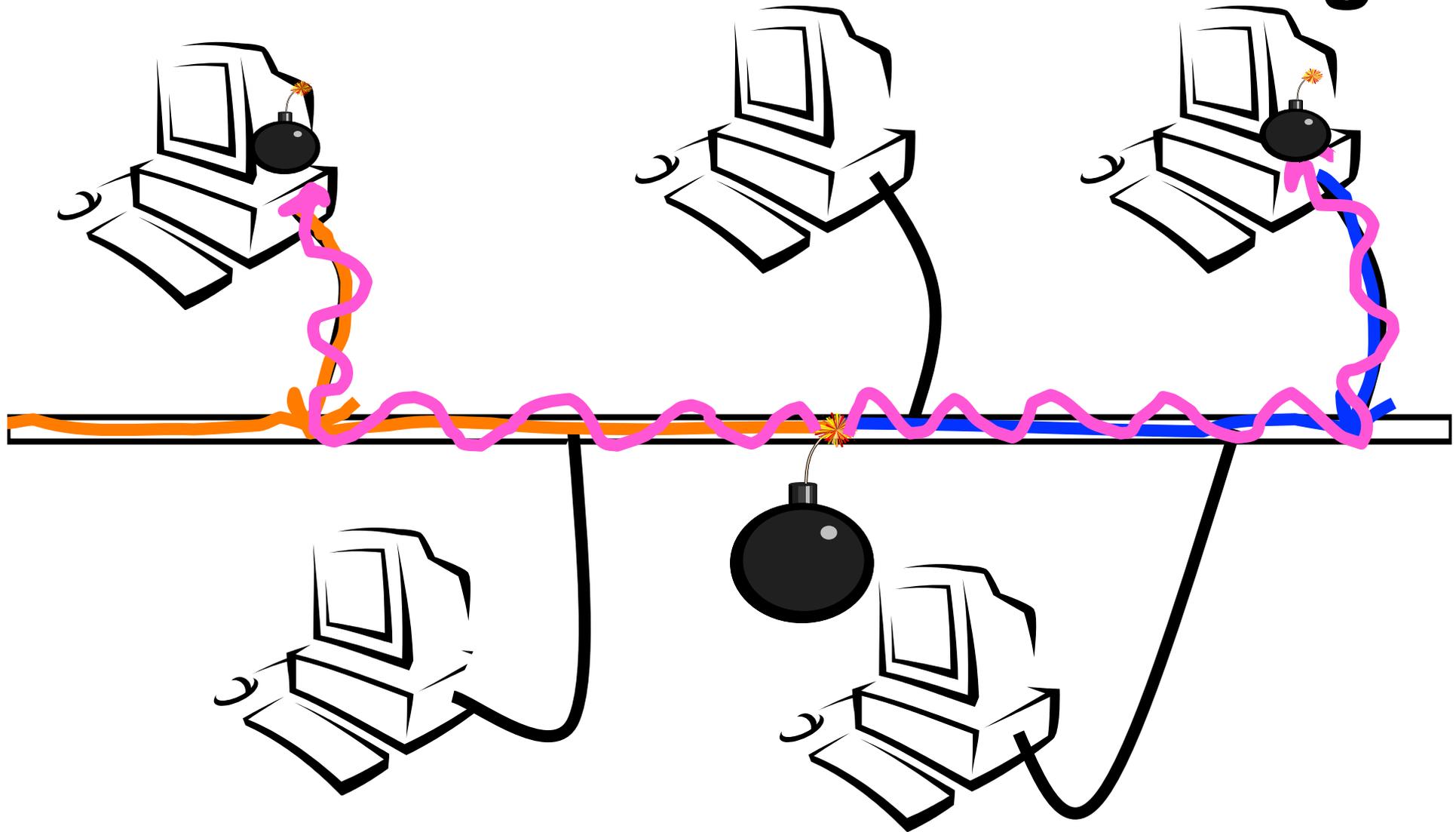
# Ethernet-Netzwerktopologie



# **CSMA/CD: Carrier Sense Multiple Access w/ Collision Detect**

- Gemeinsame Leitung für viele Stationen
- Analogie zu einem Bus
- Carrier Sense: Stationen können erkennen, ob die Leitung benutzt oder frei ist
- Collision Detect bedeutet, daß der Sender mithört und erkennt, falls es zu einer Kollision kommt

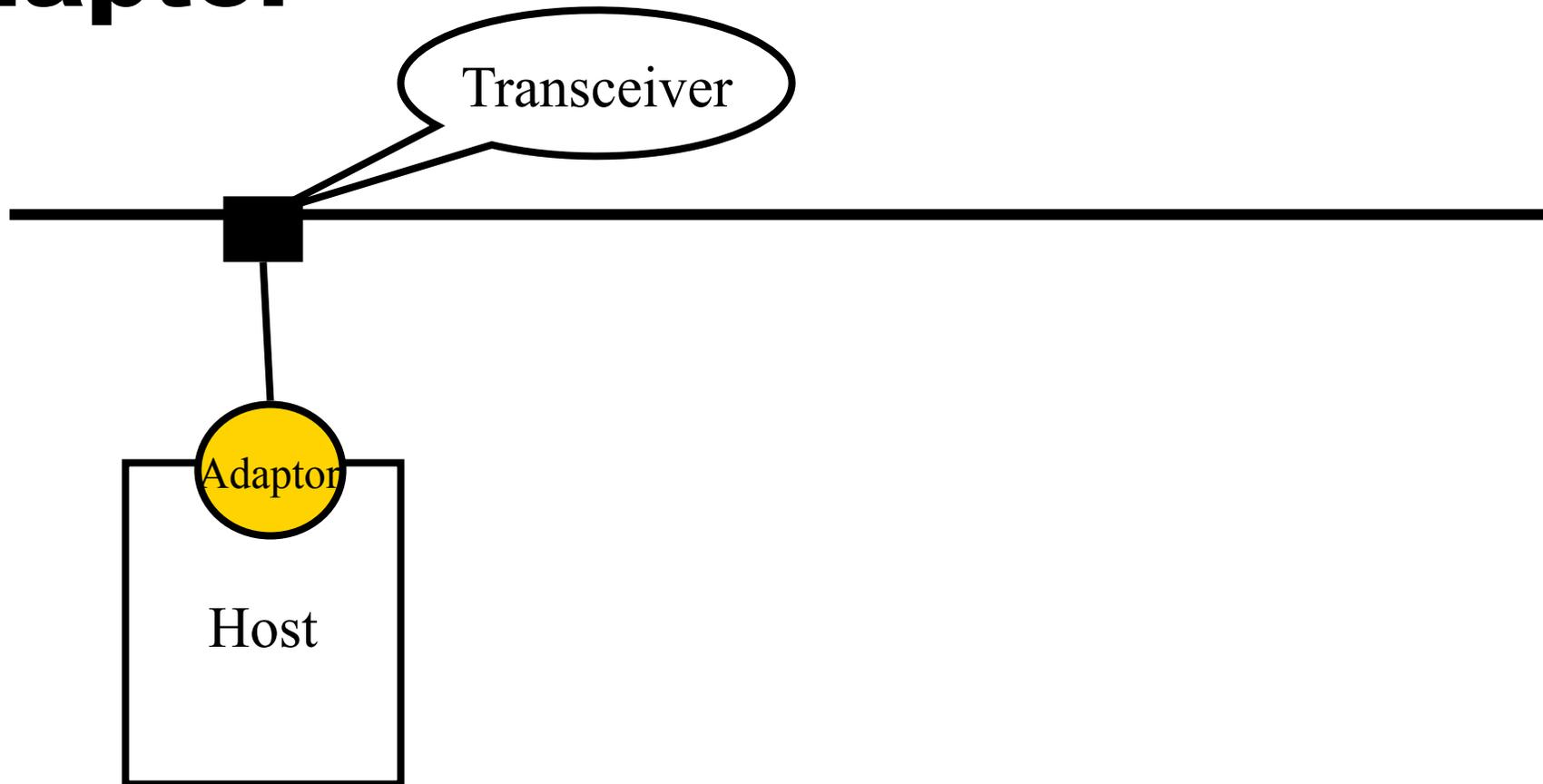
# Ethernet: Kollisionserkennung



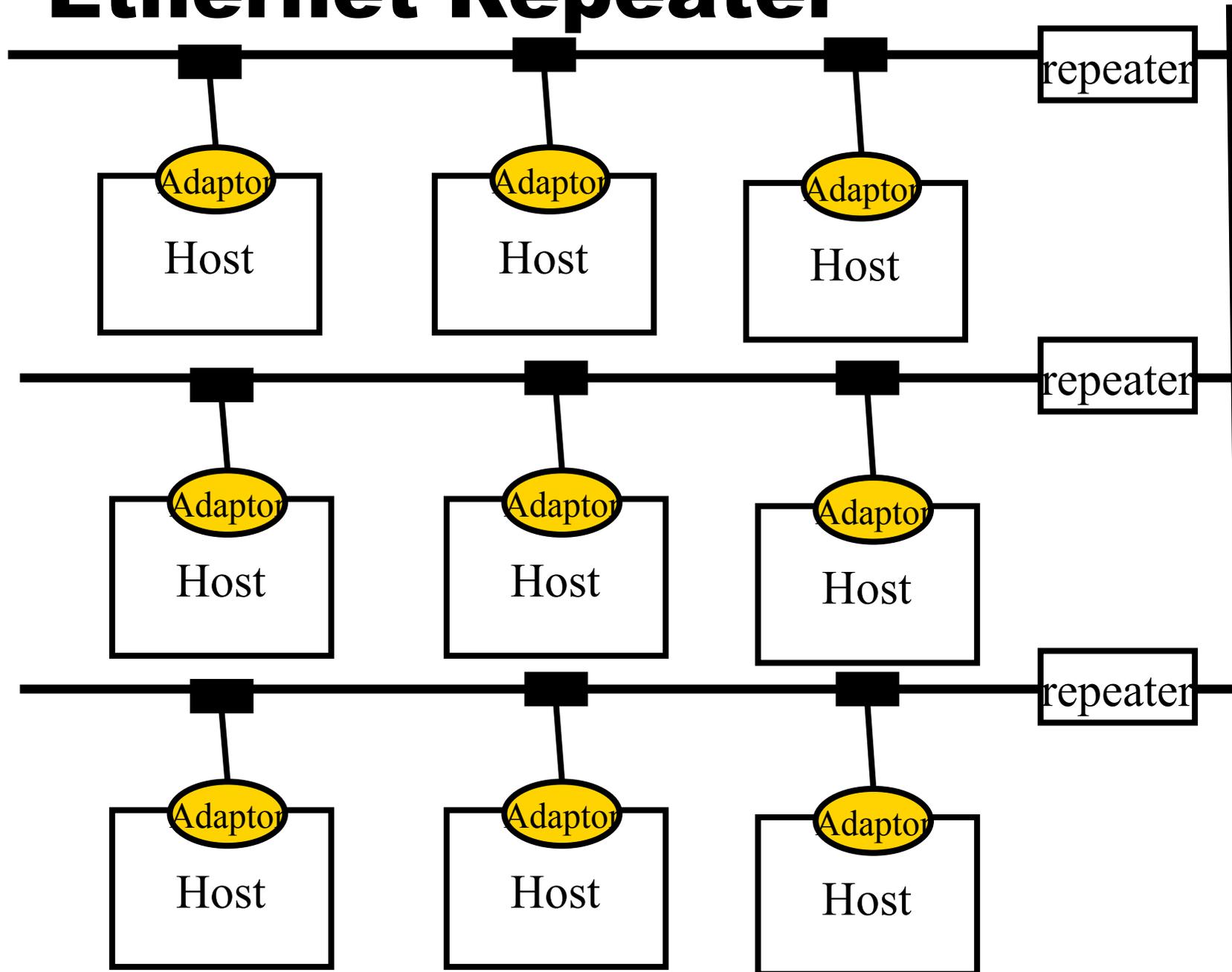
# Nach Kollisionserkennung ...

- Sender wird es noch(mehr)mals versuchen
  - normalerweise bis zu 16 mal
- beim n.-ten Versuch wird der Sender
  - $k * 51.2 \mu\text{s}$  warten
  - $k$  ist eine zufällig gewählte Zahl aus  $\{0,1, \dots, 2^{n-1}\}$
- Also wird beim 3. Versuch 0, 51.2, 102.4, 153.6 oder 204.8  $\mu\text{s}$  gewartet

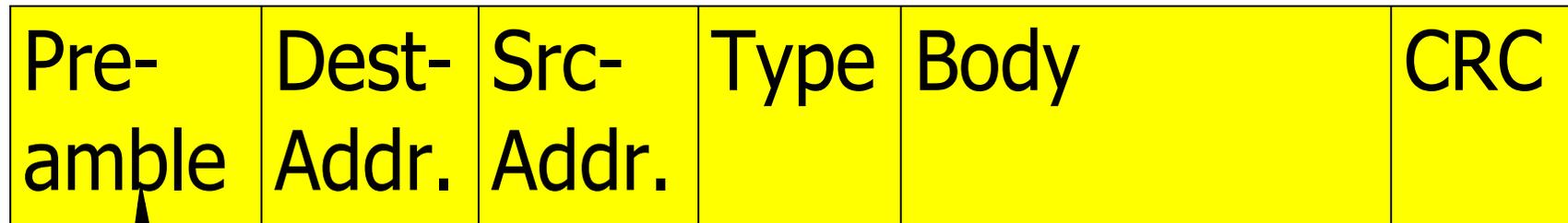
# Ethernet Transceiver und Adaptor



# Ethernet Repeater



# Ethernet Frame



64

48

48

16

variabel,  
minimal 368

32

Zum  
Synchronisieren

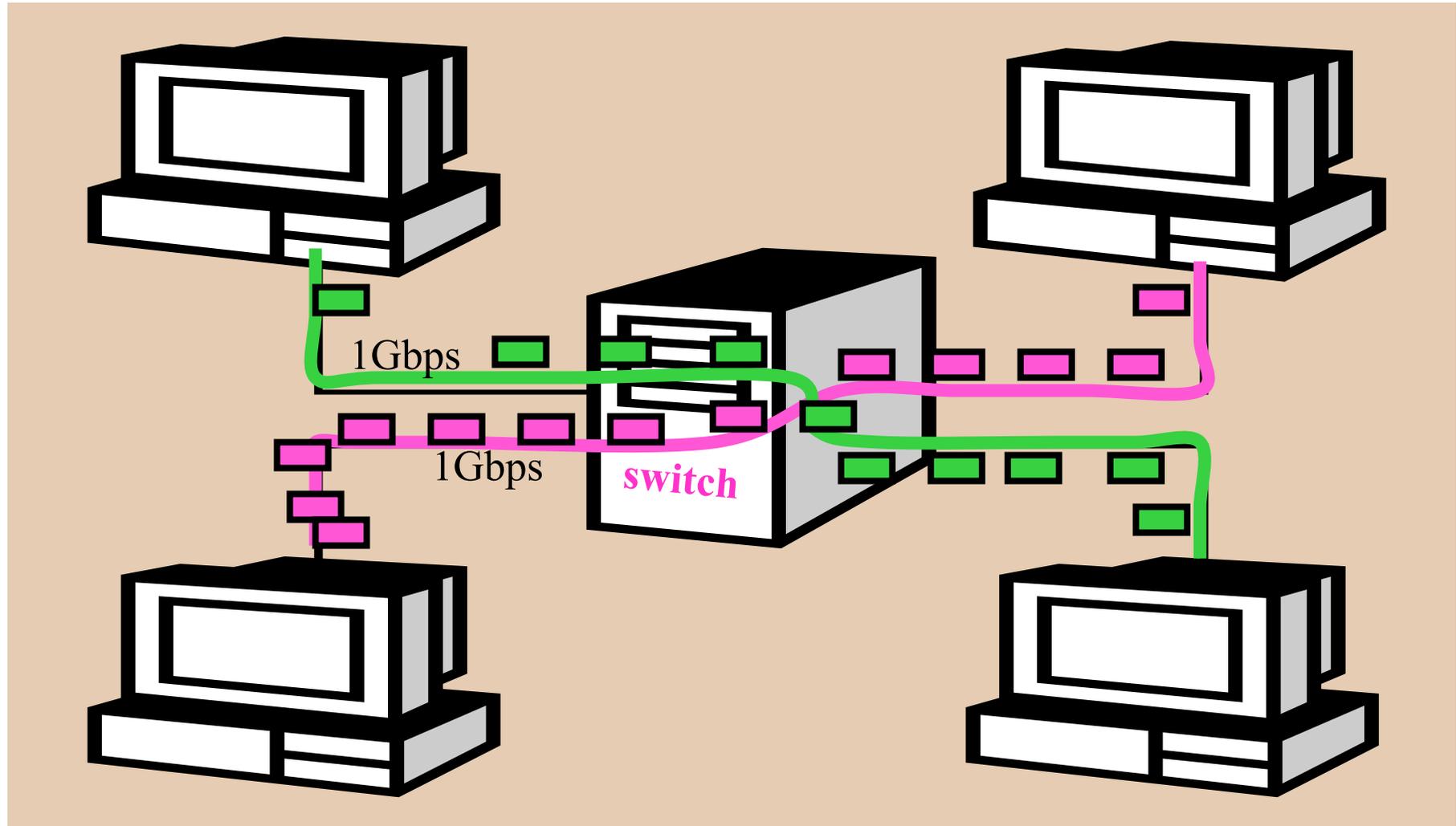
Cyclic Redundancy Code:  
Error-Detecting Code

Alle Ethernet-Adaptoren sind weltweit  
eindeutig „durchnummeriert“  
(Hersteller bekommen Präfix zugeordnet)

# Leistung von Ethernet

- Ursprünglich 10 Mbps
- Busförmige Netze
  - alle Stationen teilen sich das Medium
  - sie können auch alle alles mithören
- Nur bis zu einer Auslastung von ca 30 % geeignet; danach Leistungsdegradierung (wie thrashing)
- keine Realzeit-Tauglichkeit

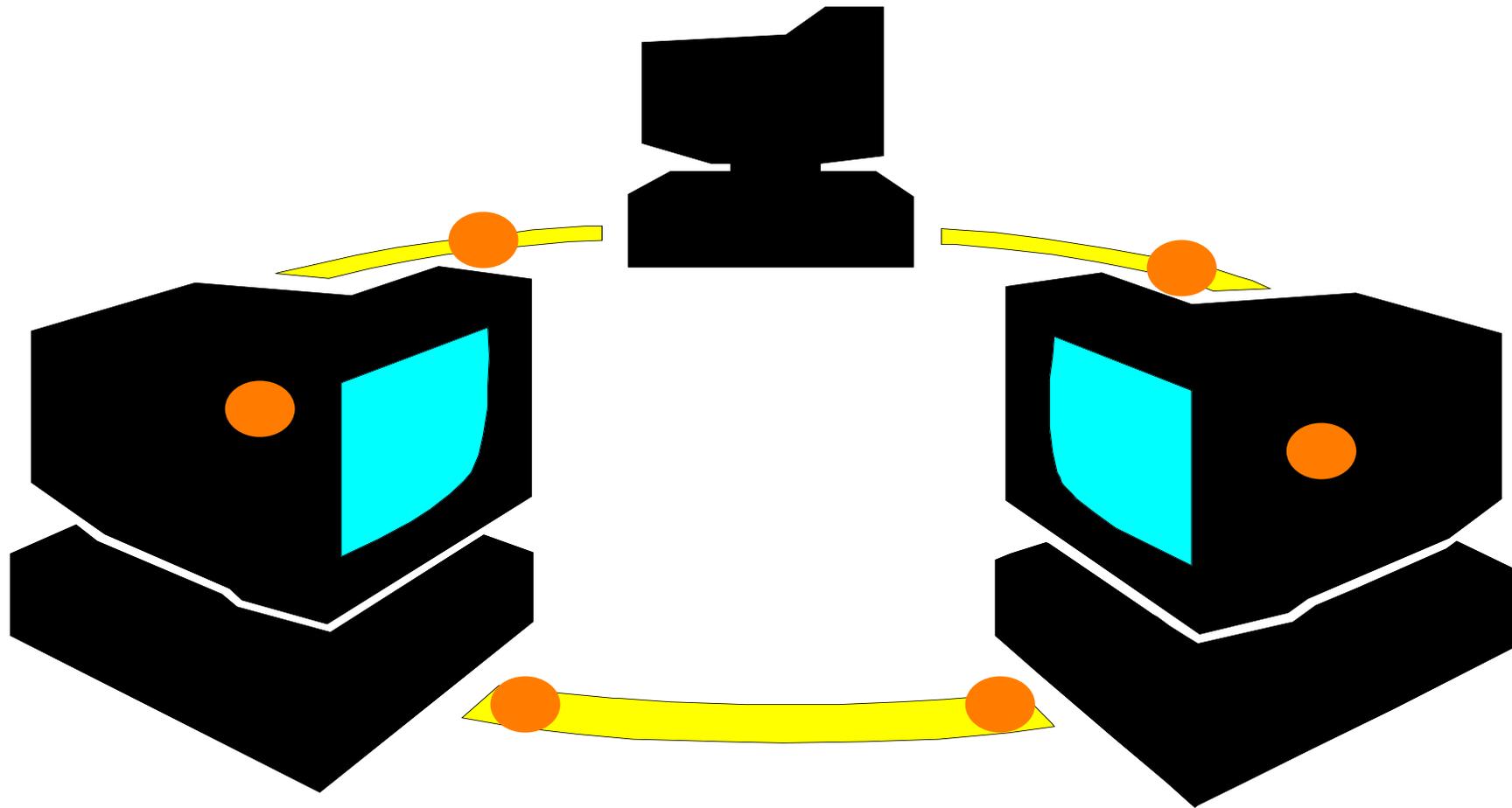
# 1 Gb/s **switched** Ethernet



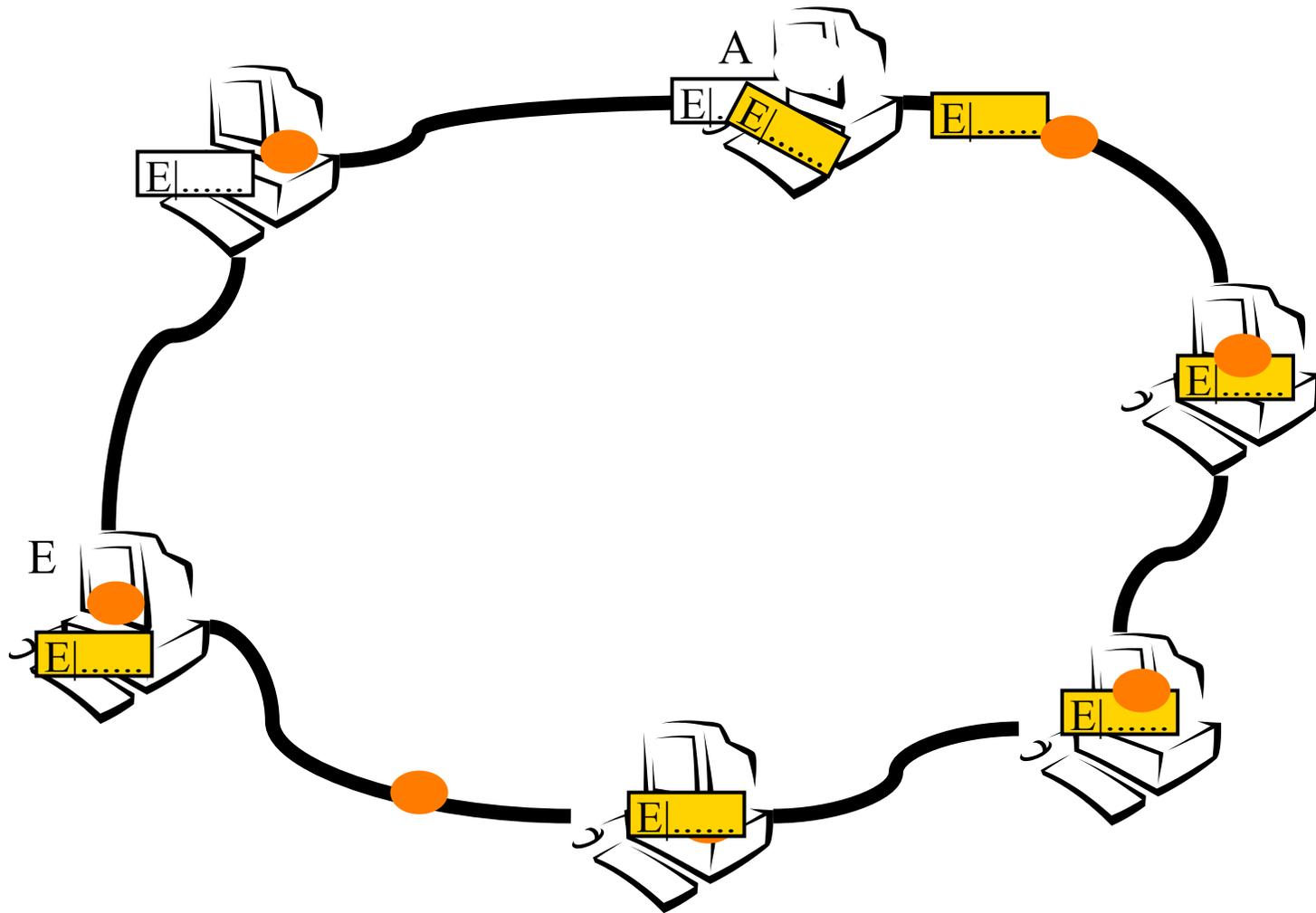
# Stand der Technik

- 1 Gbps Ethernet
  - Switched
- **10 Gbps Ethernet**
  - Verfügbar in neuen Rechnern
- Ethernet wird im LAN vorherrschend bleiben
- ATM konnte Ethernet nicht verdrängen
- Tokenring-Netze werden viel seltener verwendet
  - high-end Bereich (FDDI)
  - Realzeit-Anwendungen

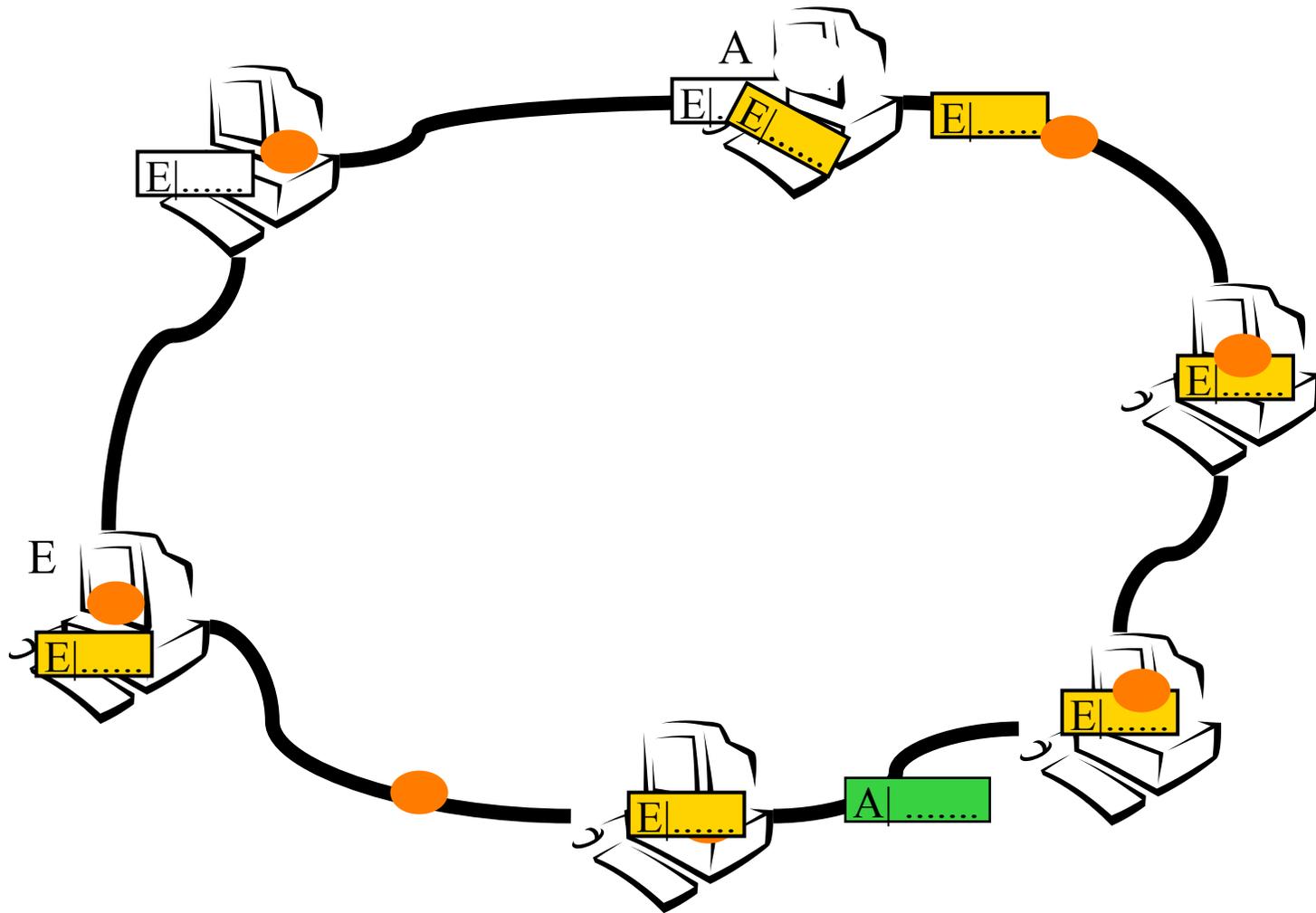
# Tokenring-Netzwerke



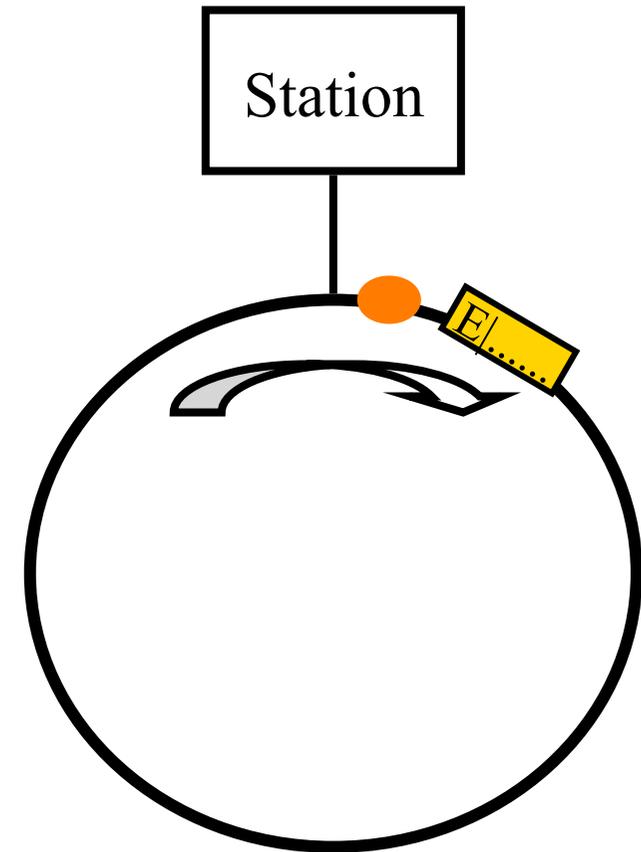
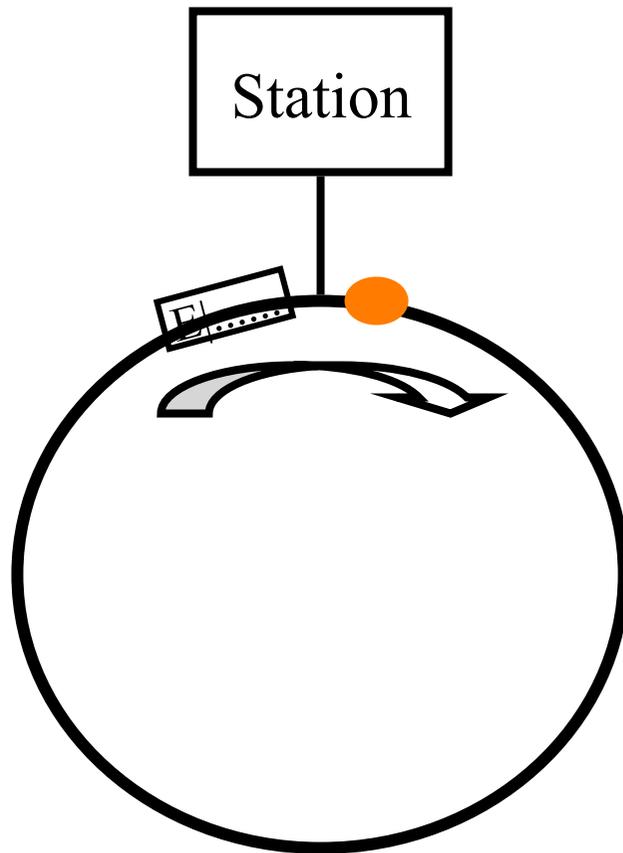
# Nachrichtenaustausch über Tokenring



# Nachrichtenaustausch über Tokenring: **early token release**



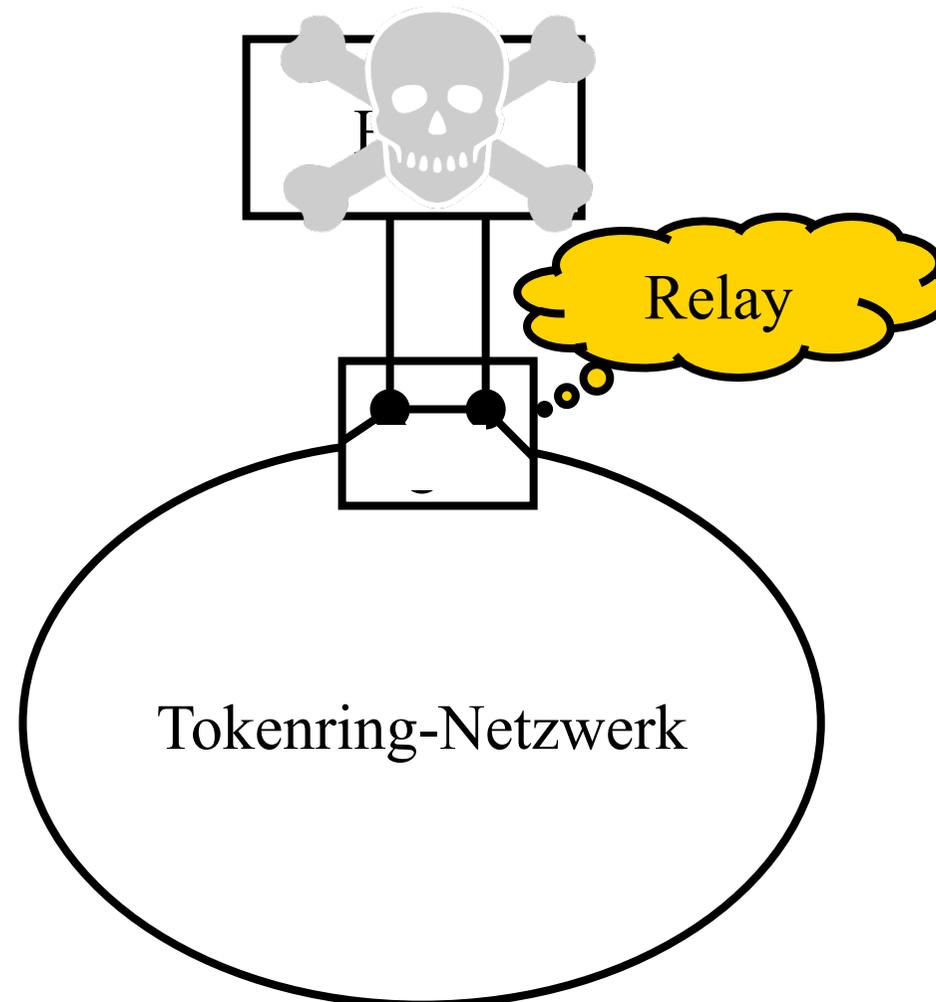
# Token-Freigabe: Delayed versus Early



# Leistungsgarantien

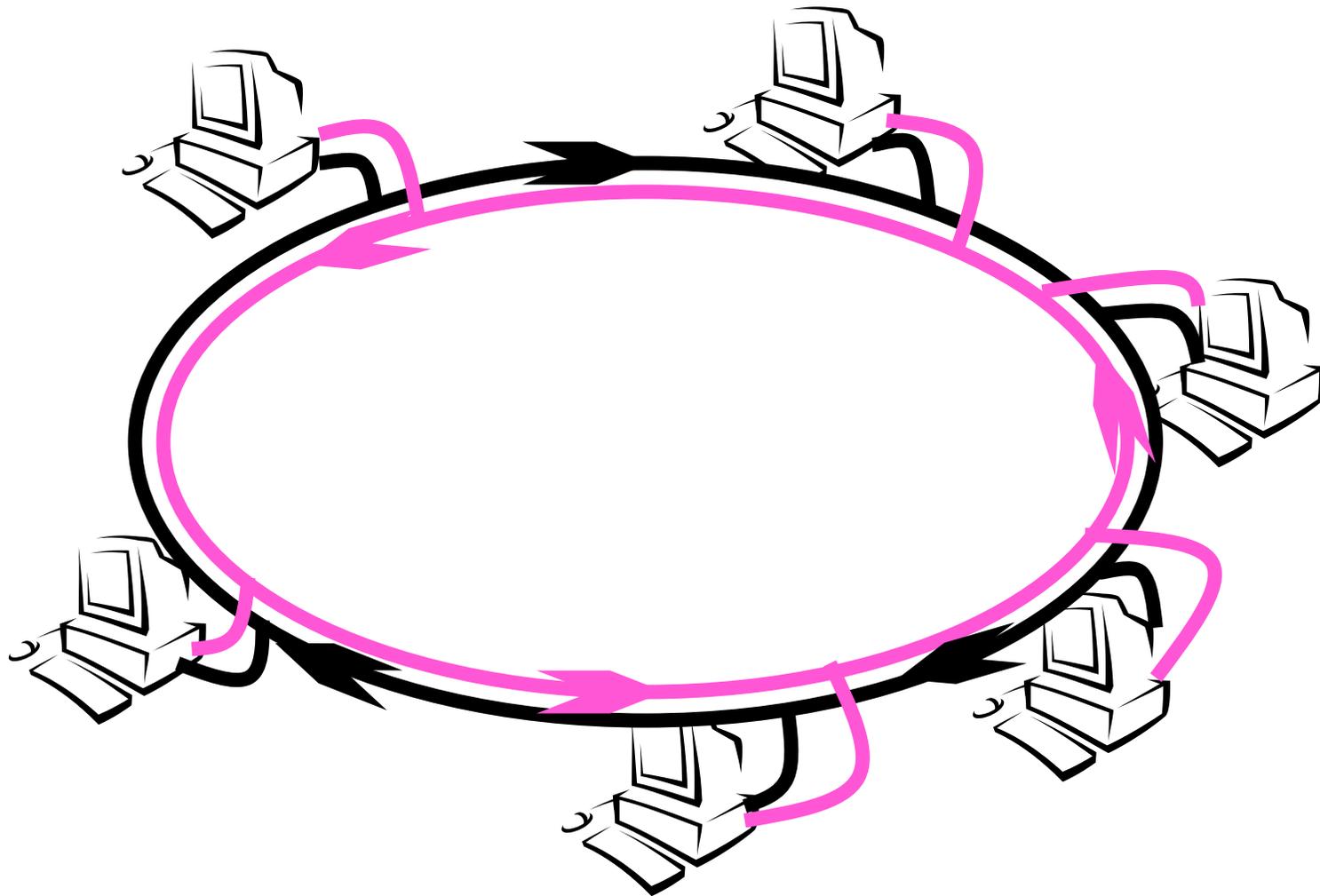
- Eine Station darf das Token nicht beliebig lange „festhalten“
  - auch dann nicht, wenn sie noch (viel) zu senden hat
- nach einer festgesetzten Zeit muss das Token wieder freigegeben werden
- Daraus lassen sich Leistungsgarantien für Realzeitanwendungen herleiten

# Fehlertoleranz gegen Rechnerausfall

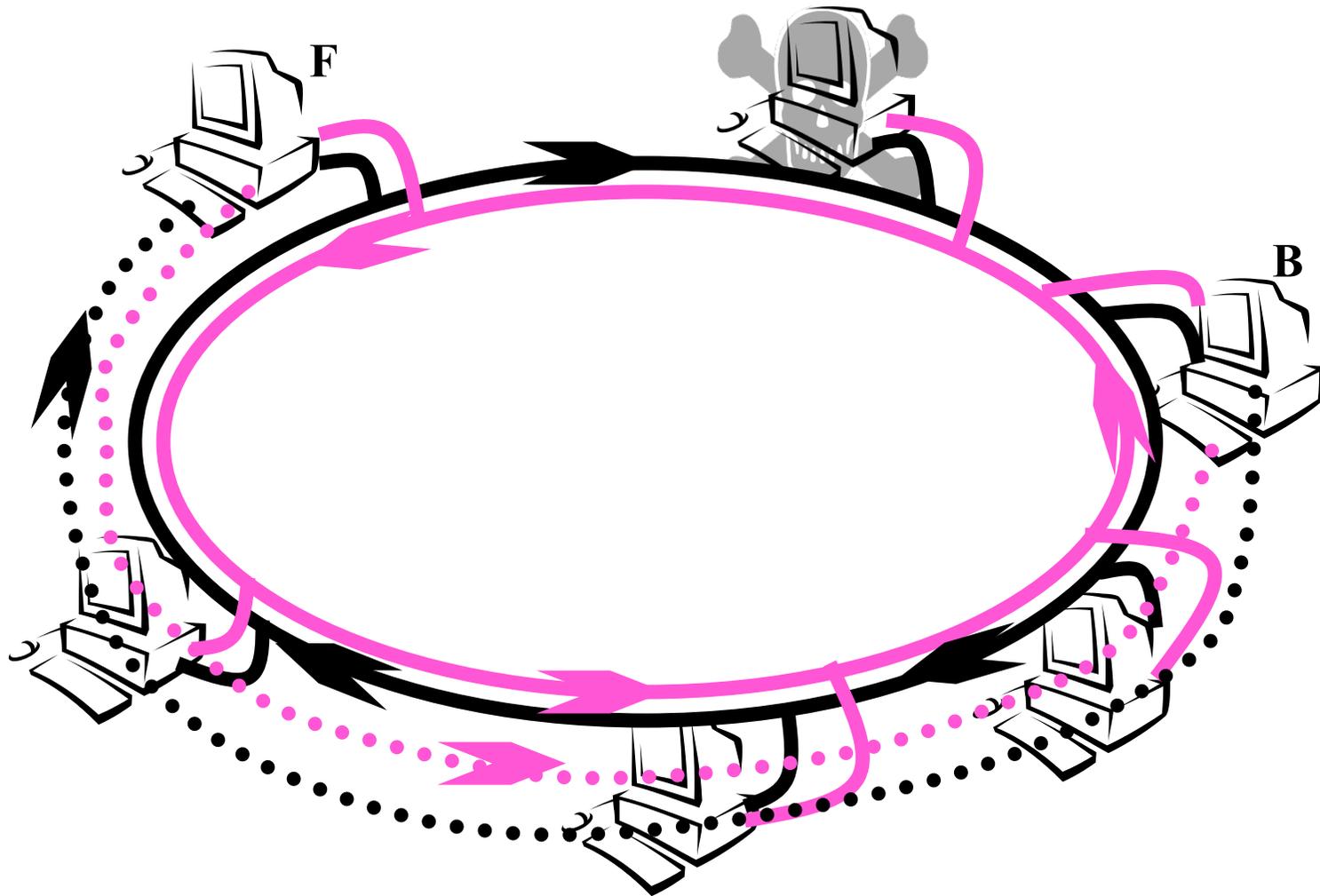


# FDDI Doppelring

## Fiber Distributed Data Interface

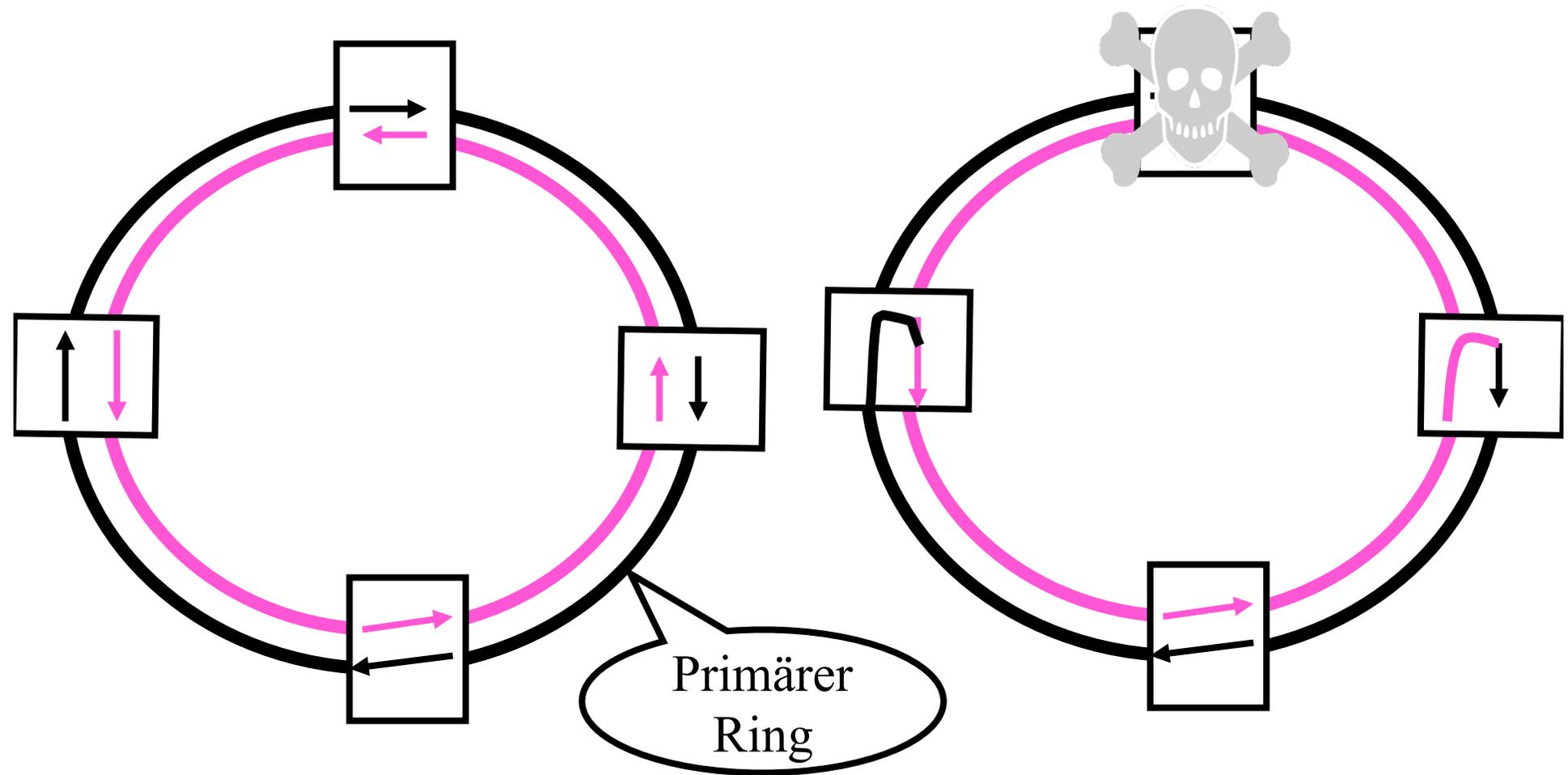


# FDDI: Fehlertoleranz



# Doppelring: normale Op.

# Fehlerbehandlung



# Leistungs-Kennzahlen FDDI

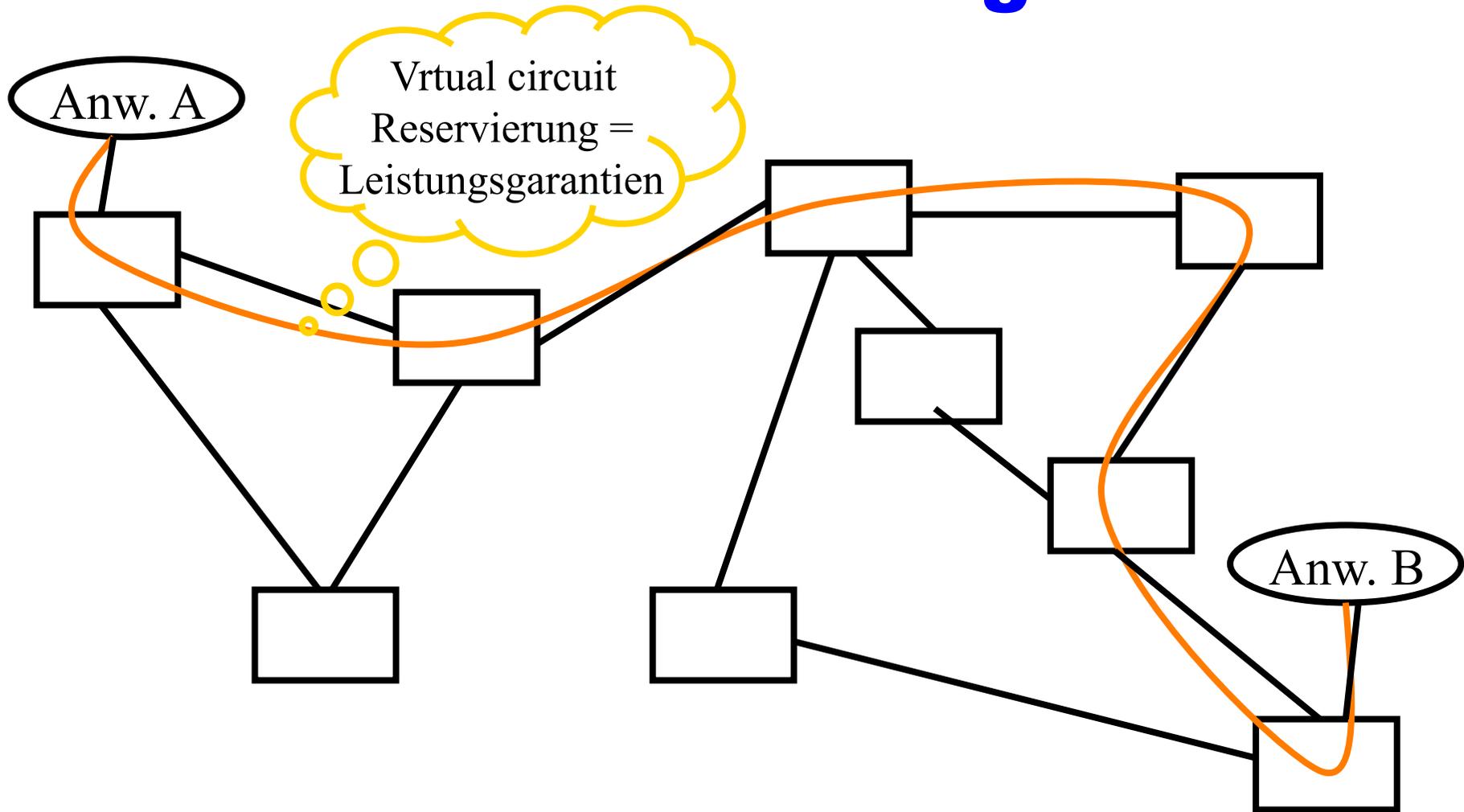
- Maximal ca 500 Stationen
- 100 km Länge
- 100 Mbps
- garantierte TTRT
  - target token rotation time
  - muss von den Stationen kontrolliert/eingehalten werden
- FDDI wird heute im high-end LAN-Bereich eingesetzt
  - u.a. Vernetzung von SAP Application/Database-Server

# ATM: Asynchronous Transfer Mode

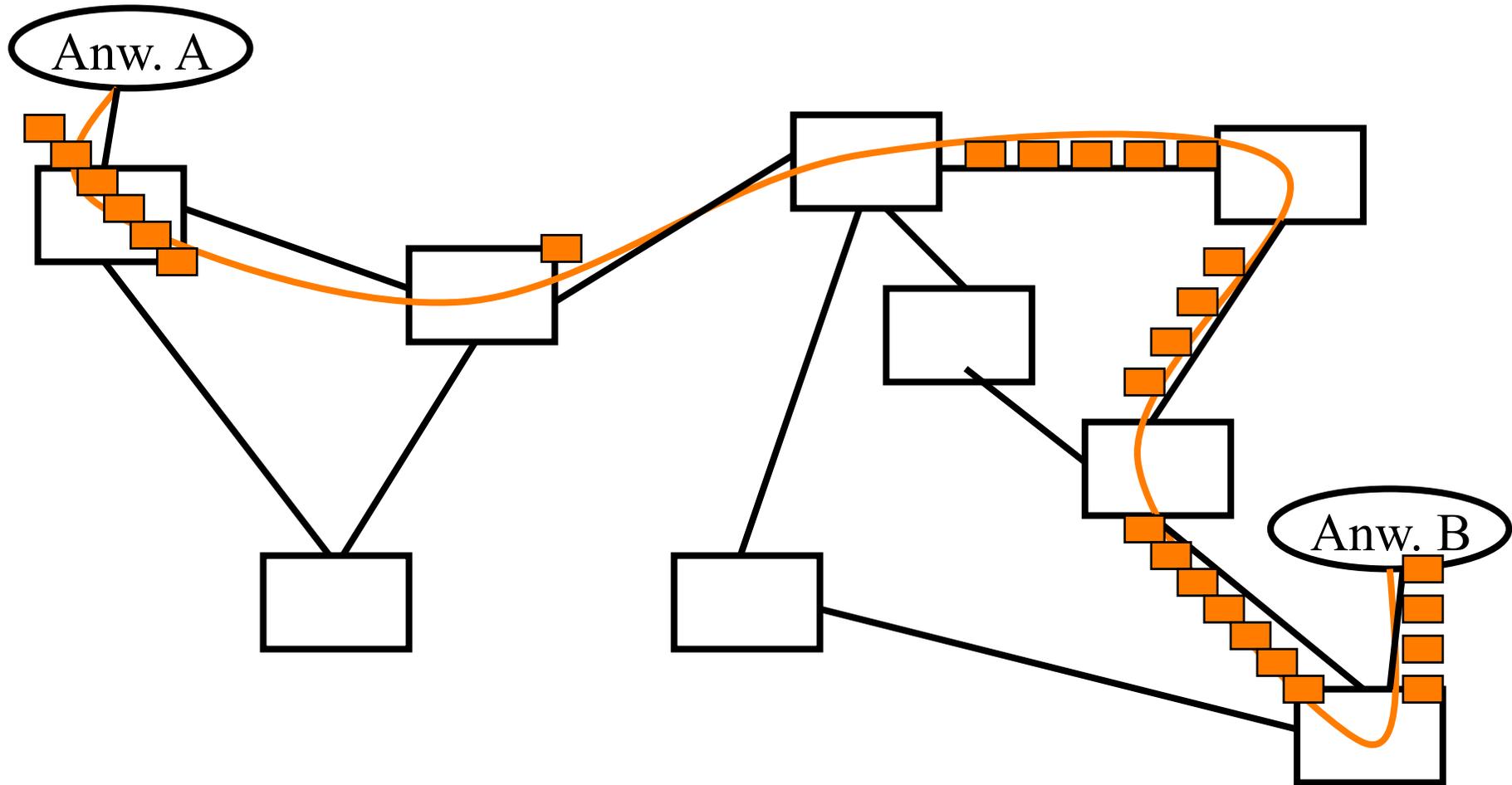
- Wurde Anfang 1980-er entwickelt
- Insbesondere von der Telekommunikations-Industrie propagiert
- virtual circuit ~ vorab reservierte Verbindung durch das Netzwerk
- sehr kleine Pakete/Cells (53 Byte, davon 5 Byte Header, 48 Byte Nutzdaten)
- besonders für die Übertragung von Telefon-daten geeignet (alle 33 ms ein kleines Paket)
- Der ganz große Durchbruch ist ausgeblieben
- Heute als (veraltete?) Backbone-Technologie
  - 0.6 - 2.4 Gbps über Glasfaser

# ATM: Asynchronous Transfer Mode

## Mode: virtueller Leitungsaufbau



# ATM: Asynchronous Transfer Mode: **Cell Switching**

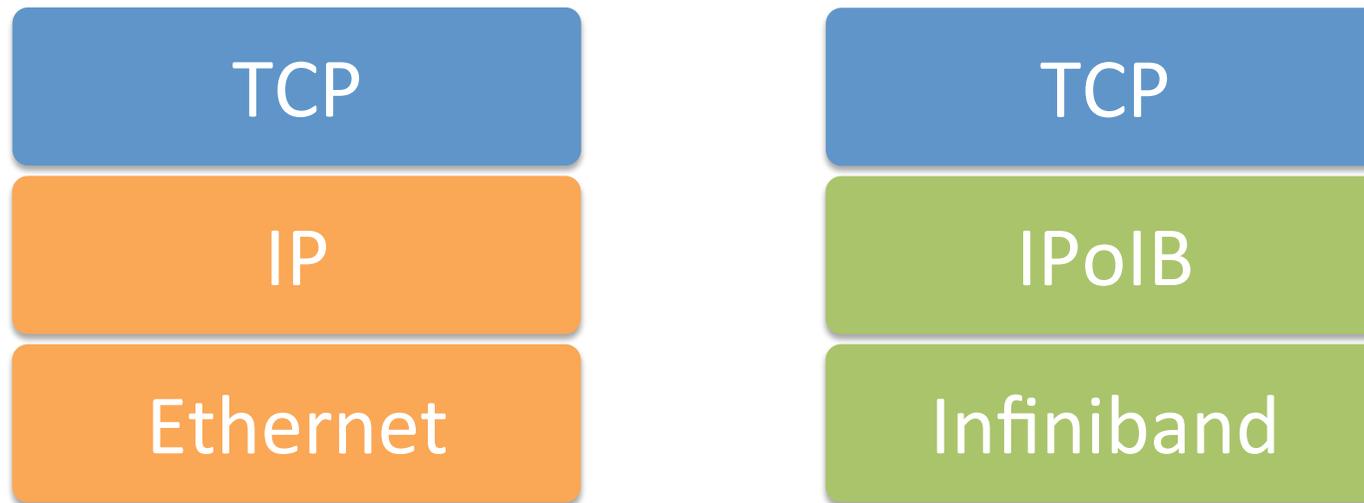


# Infiniband

- Neuer Standard für den “data link layer”
- Alternative zu Ethernet
- Ausgelegt auf sehr hohen Durchsatz von mehreren GB/s
- Erreicht sehr geringe Latenz von nur einer Mikrosekunde

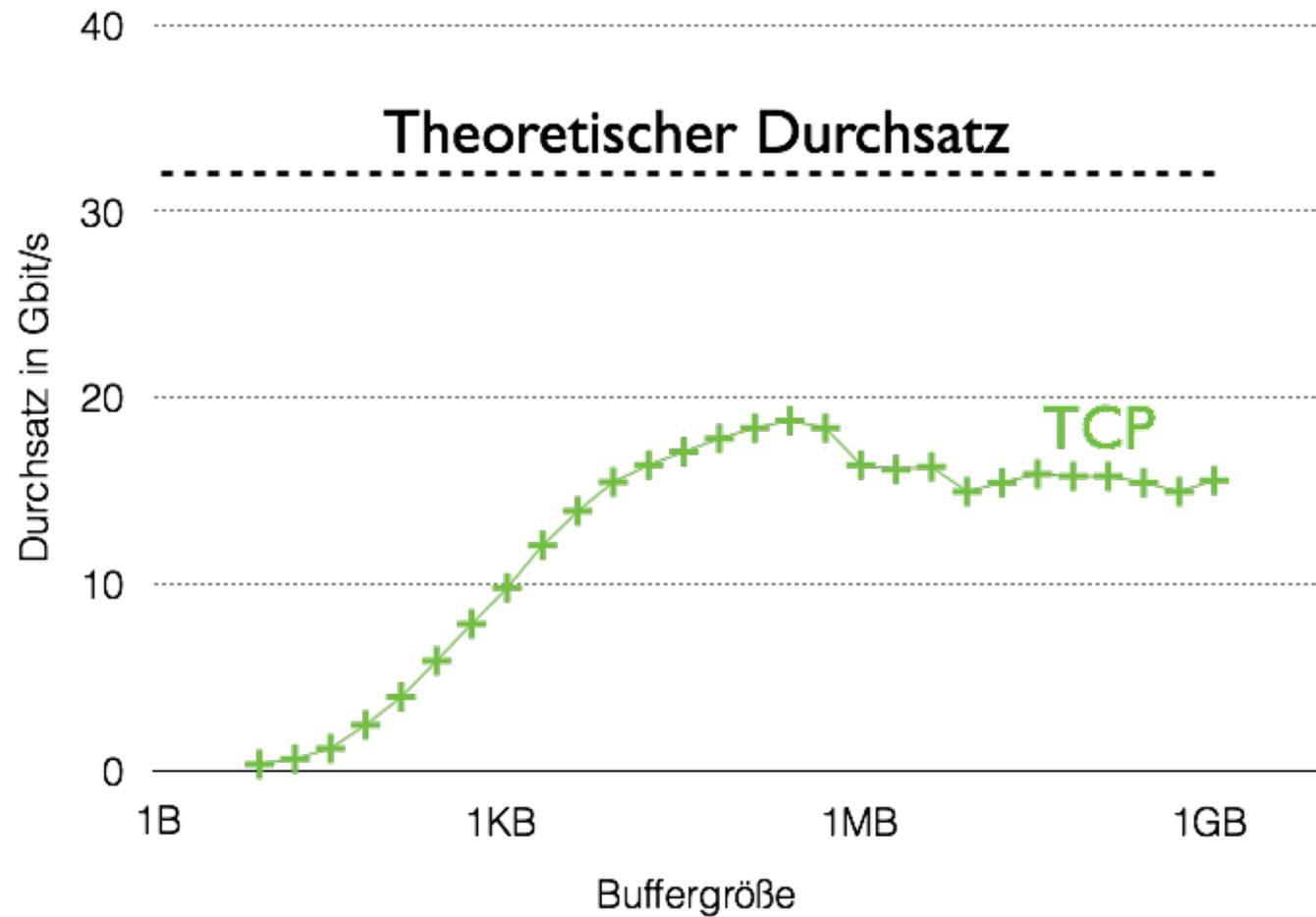
Netzwerk Standard	Theoretischer Durchsatz
Gigabit Ethernet	0,25 GB/s
Infiniband 4xQDR	4 GB/s
Infiniband 4xFDR	7 GB/s

# TCP Kompatibilität mit IPoIB



- IPoIB erlaubt TCP über Infiniband zu benutzen
- **Vorteil:** Anwendungen laufen ohne Anpassung
- **Nachteil:** Geringer Durchsatz, TCP ist Flaschenhals

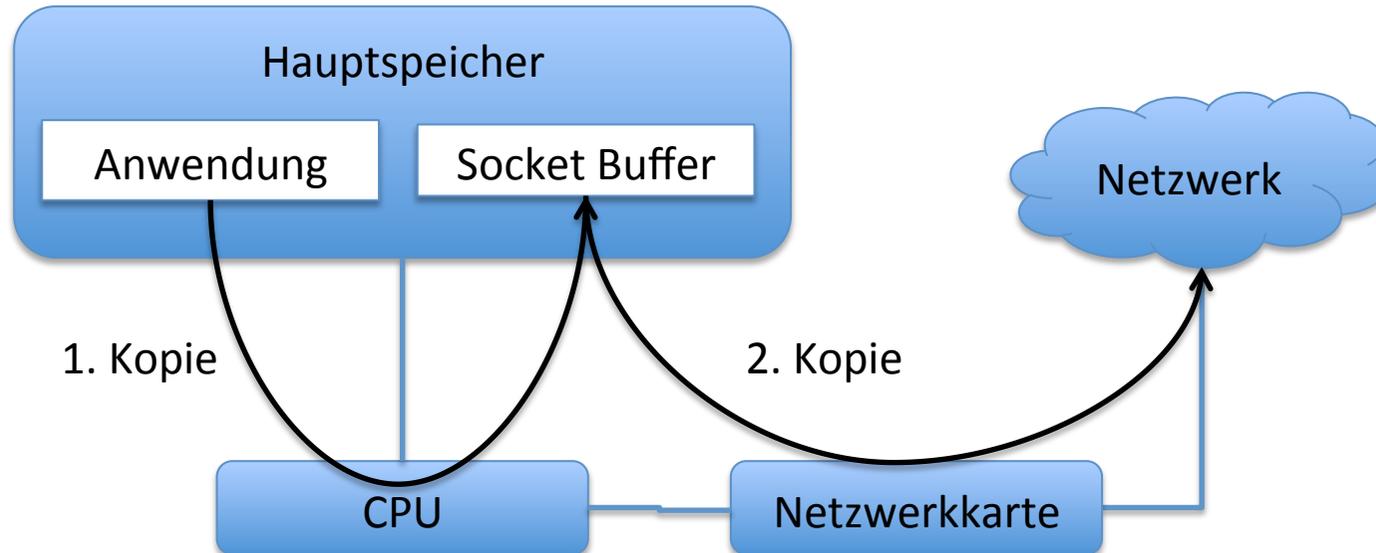
# Durchsatz TCP über Infiniband



# Kosten von TCP

- Systemaufrufe werden benötigt (teuer aufgrund von Kontextwechsel)
- Protokoll wird von der CPU ausgeführt:
  - Zerstückelung großer Nachrichten in Pakete
  - Generierung und Behandlung von ACKs
  - Erneutes Übertragen von verloren gegangenen Paketen
- Daten werden mehrfach über den Speicherbus übertragen

# TCP Datentransfer

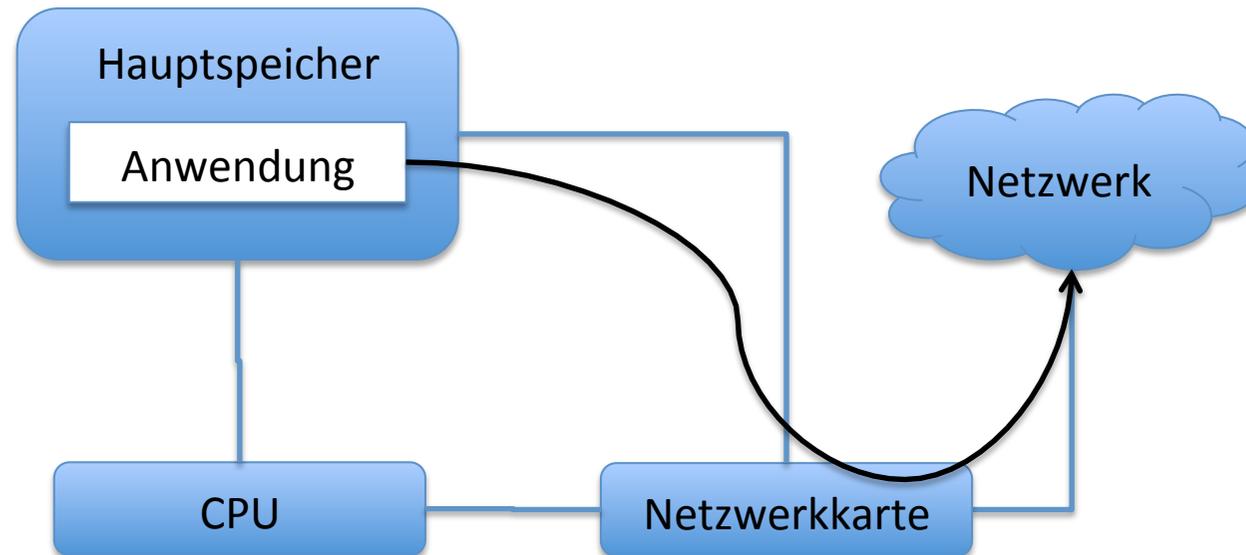


- TCP kopiert die Daten in den Socket Buffer bevor sie verschickt werden

# Lösung: RDMA

- **Remote Direct Memory Access**
- Ermöglicht direkten Zugriff der Netzwerkkarte auf den Hauptspeicher („zero copy“)
- Sehr geringe CPU-Auslastung (keine Systemaufrufe im Datenpfad)
- Protokoll wird komplett asynchron von der Netzwerkkarte ausgeführt
- Erreicht sehr hohen Durchsatz

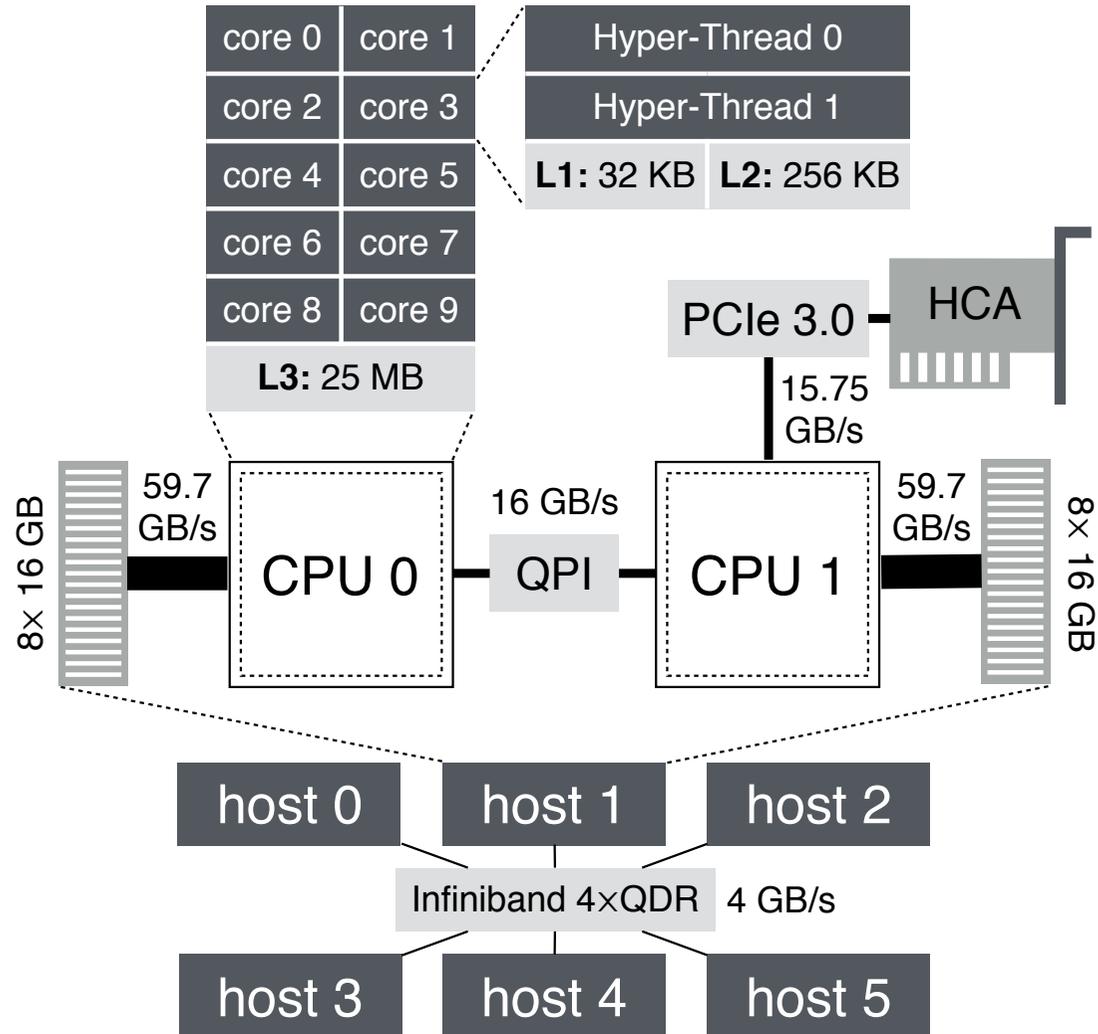
# RDMA Datentransfer



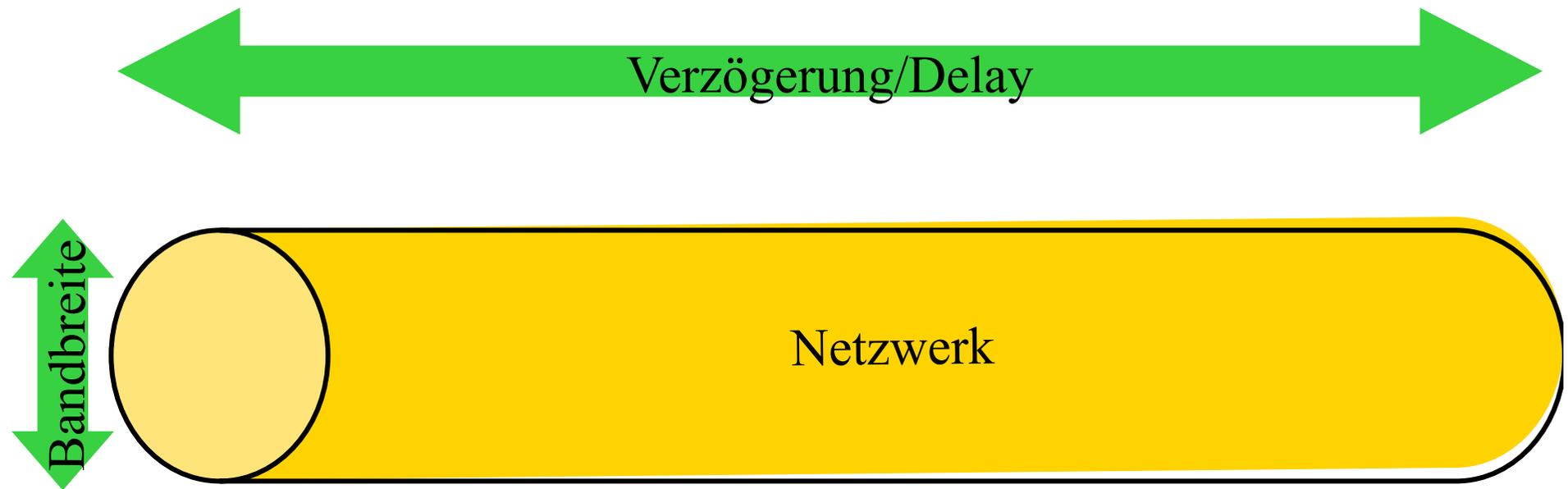
- RDMA-Netzwerkkarten greifen direkt auf den Hauptspeicher zu (ohne Beteiligung der CPU)



# ScyPer Cluster



# Leistungs-Kennzahlen: Bandbreite/Verzögerung



# Leistungskennzahlen eines Netzwerks

- Bandbreite (Throughput, Bandwidth)
  - kann durch Infrastruktur-Investitionen erhöht werden
- Latenz (Latency)
  - $\text{Latency} = \text{PropagationDelay} + \text{Transmit} + \text{Queue}$
  - **PropagationDelay** = Distanz/Lichtgeschwindigkeit
    - „Gott-gegeben“: keine Reduzierung möglich
    - M → HNL: 30.000km / 300.000 km/s → 1/10 s → 100 ms
  - **Transmit** = Nachrichtengröße / Bandbreite
  - **Queue**: Verzögerung durch Warteschlangen in den Routern

# Leistungskennzahlen

- Delay \* Bandwidth
  - intuitiv: Kapazität der Netzwerkverbindung
- Beispiel: Transkontinentalverbindung mit
  - 100 ms RTT (Round Trip Time = 2 \* Delay)
  - 45 Mbps Bandbreite
- Delay \* Bandwidth =  $50 * 10^{-3} * 45 * 10^6$
- =  $2.25 * 10^6$
- = 280 KB
- Ein Sender muss 280 KB senden, bis der Empfänger das erste Bit empfängt

# Leistungskennzahlen

- Delay \* Bandwidth
  - intuitiv: Kapazität der Netzwerkverbindung
- Der Sender sollte  $2 * \text{Delay} * \text{Bandwidth}$  Bit senden, bevor er ein Acknowledgment vom Empfänger erwarten kann
  - $\text{Delay} * 2 = \text{RTT}$  (Round Trip Time)
  - in unserem Bsp also 560 KB
- Wichtig für die richtige Wahl der Fenstergröße bei TCP/IP
  - sollte  $\text{Delay} * \text{Bandwidth} * 2$  betragen, damit die „Röhre“ immer gefüllt ist

# Leistungskennzahlen

- Kennzahlen zwischen München (Europa) und USA im Internet
  - RTT = 300 ms
  - verfügbare Bandbreite = einige Kbps
- RTT im lokalen Netz (LAN)
  - < 1 ms
  - 100 oder 1000 oder 10000Mbps

# Leistungskennzahlen: Kb versus KB, Mb versus MB

- Netzwerk: Bandbreite ergibt sich aus Taktfrequenz
  - 10 MHz =  $10 * 10^{**6}$  Hz ergibt eine Bandbreite von
    - 10 Mbps =  $10 * 10^{**6}$
    - also 1 Mbps =  $10^{**6}$  bits pro sec
    - 1 Kbps =  $10^{**3}$  bps
- Aber: 1 MB =  $(2^{**20}) * 8$  bit
  - 1 KB =  $(2^{**10}) * 8$  bit
- Vereinfachend: 1 MB  $\sim$  10 Mb
  - in „back of the envelope“-Berechnungen

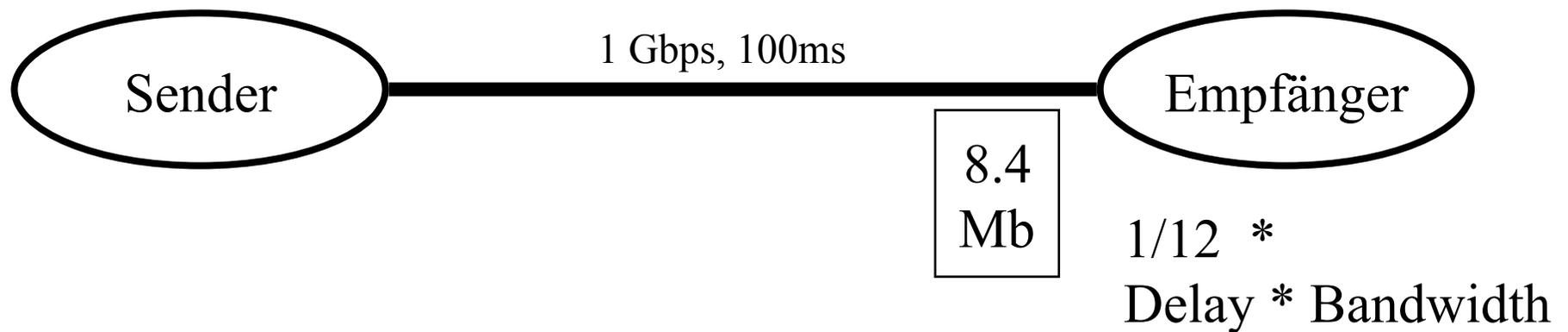
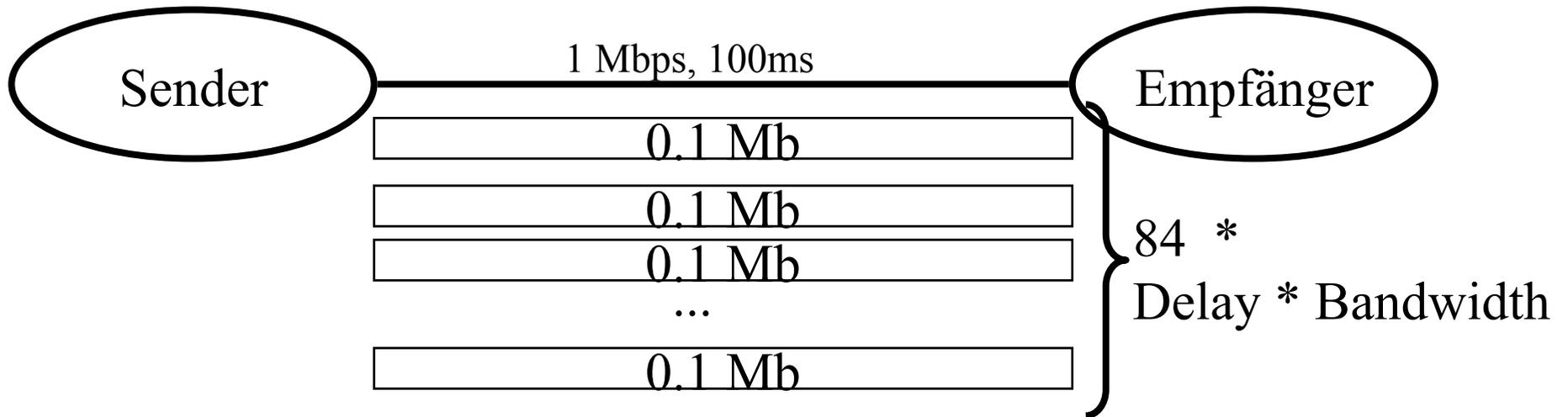
# Leistungskennzahlen

- Transferzeit =  $RTT + 1/\text{Bandbreite} * \text{Größe}$ 
  - RTT: Request + „in-flight-time“ des Nachrichtenansfangs
  - Rest: Übertragung der kompletten Nachricht
- Beispielrechnung
  - RTT = 300 ms
  - Bandbreite = 10 Kbps
  - Größe = 0.1 KB / 1 KB / 10 KB ( $\sim 100 \text{ Kb}$ )
- Transferzeit = 300 ms +  $1/10 \text{ s} = 400 \text{ ms}$
- Transferzeit = 300 ms + 1s = 1100 ms  $\sim 1\text{s}$
- Transferzeit = 300 ms + 10 s = 10300 ms  $\sim 10\text{s}$

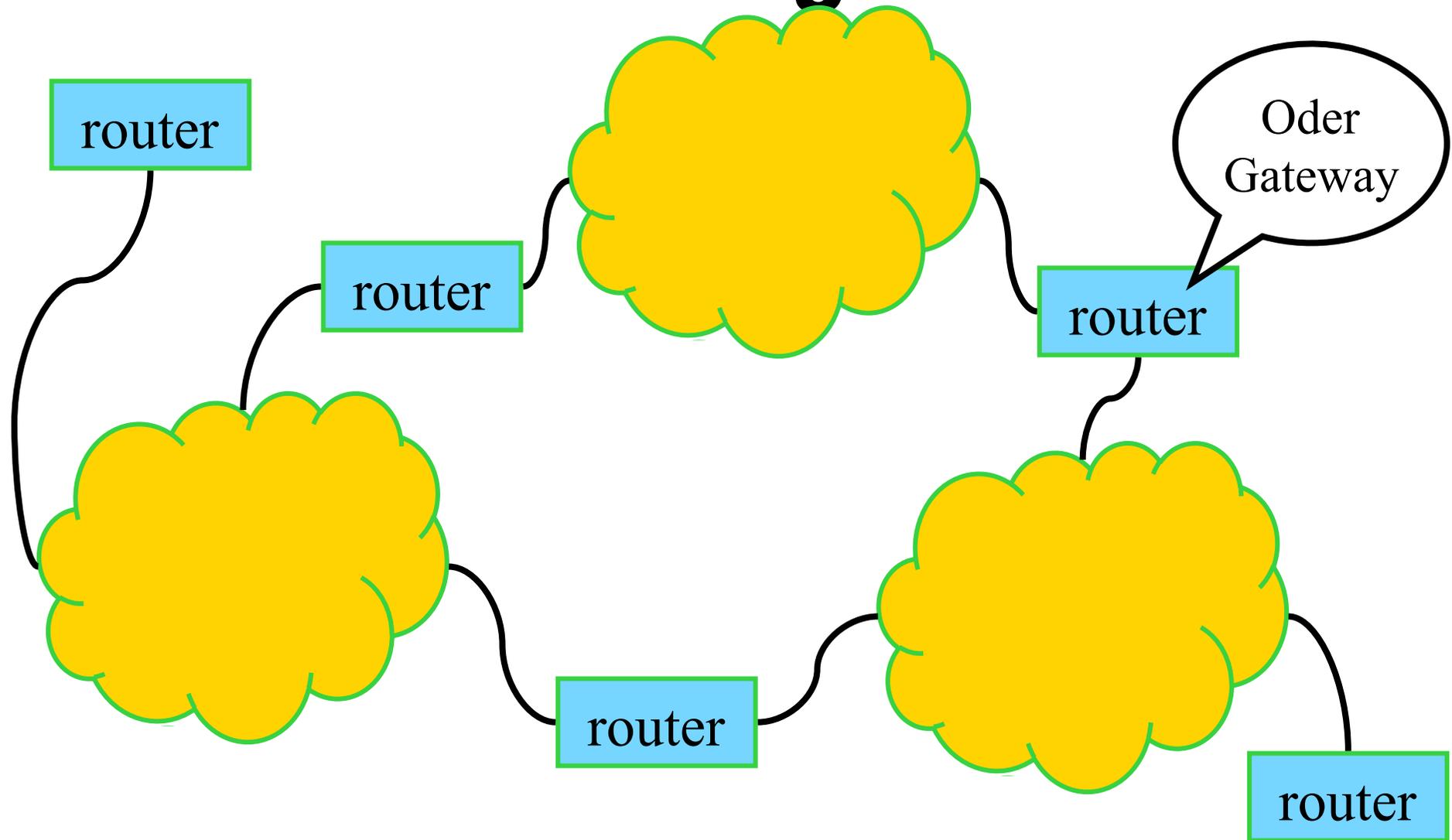
# Leistungskennzahlen: Effekt einer Bandbreitenerhöhung

- Transferzeit =  $RTT + 1/\text{Bandbreite} * \text{Größe}$
- Beispielrechnung
  - $RTT = 300 \text{ ms}$
  - Bandbreite = 1000 Kbps (1 Mbps)
  - Größe = 0.1 KB / 1 KB / 10 KB ( $\sim 100 \text{ Kb}$ )
  - Transferzeit =  $300 \text{ ms} + 1/1000\text{s} = 301 \text{ ms}$
  - Transferzeit =  $300 \text{ ms} + 1/100\text{s} = 310 \text{ ms}$
  - Transferzeit =  $300 \text{ ms} + 1/10 \text{ s} = 400 \text{ ms}$
- Folge: RTT wird immer „dominanter“

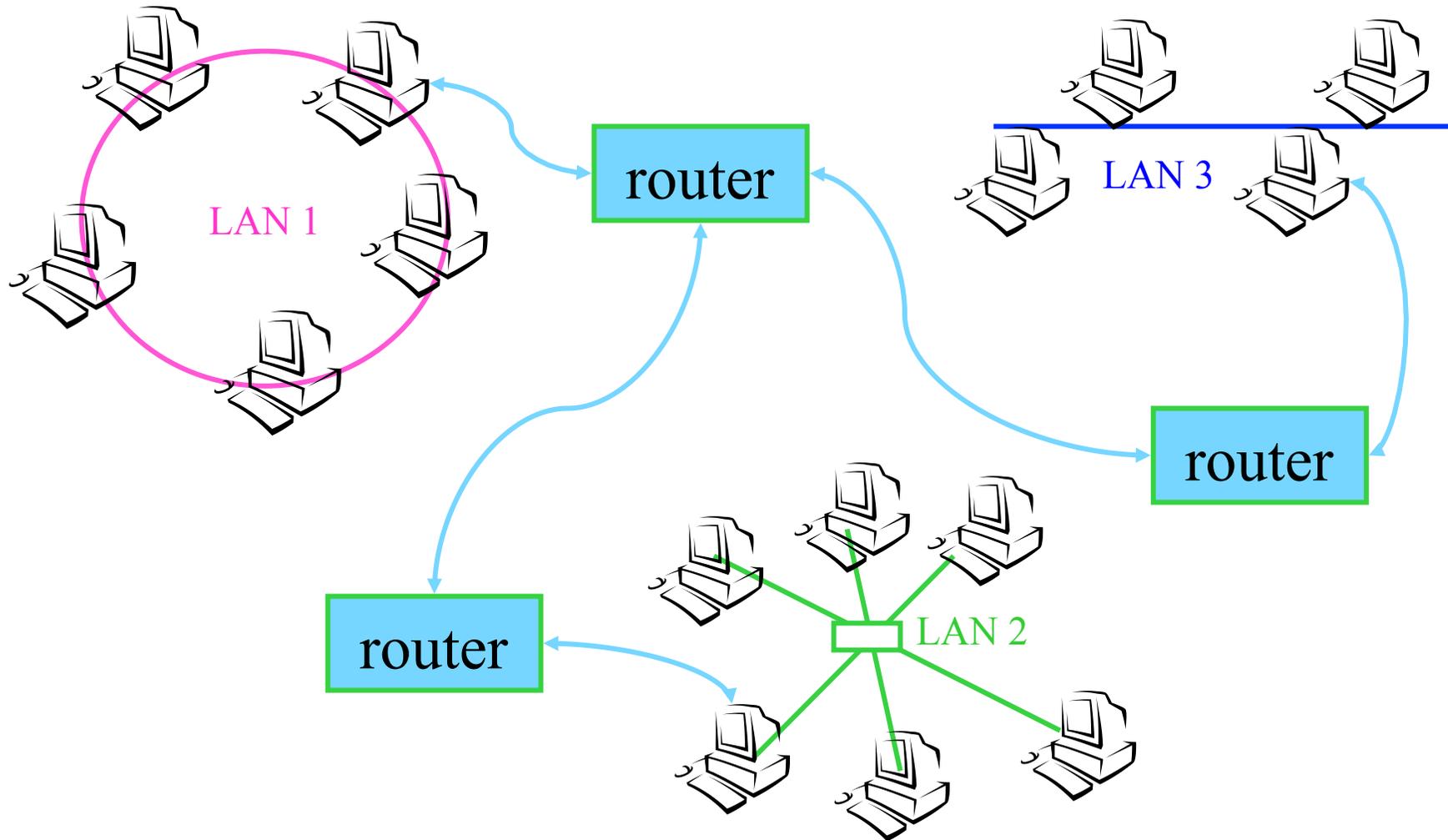
# Zusammenhang zwischen Bandbreite und Latenz



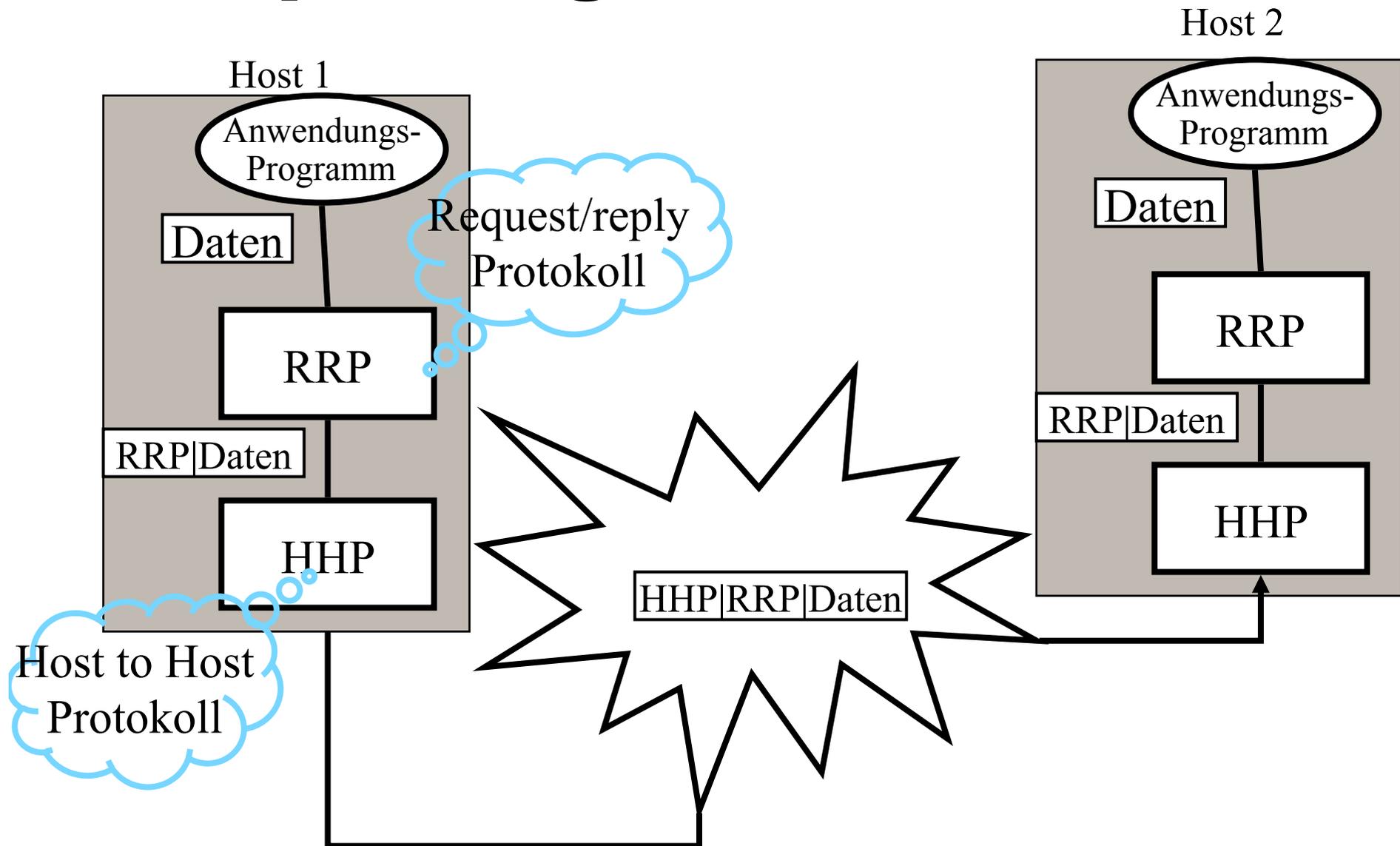
# Verbindung von Netzwerken: Inter-Networking



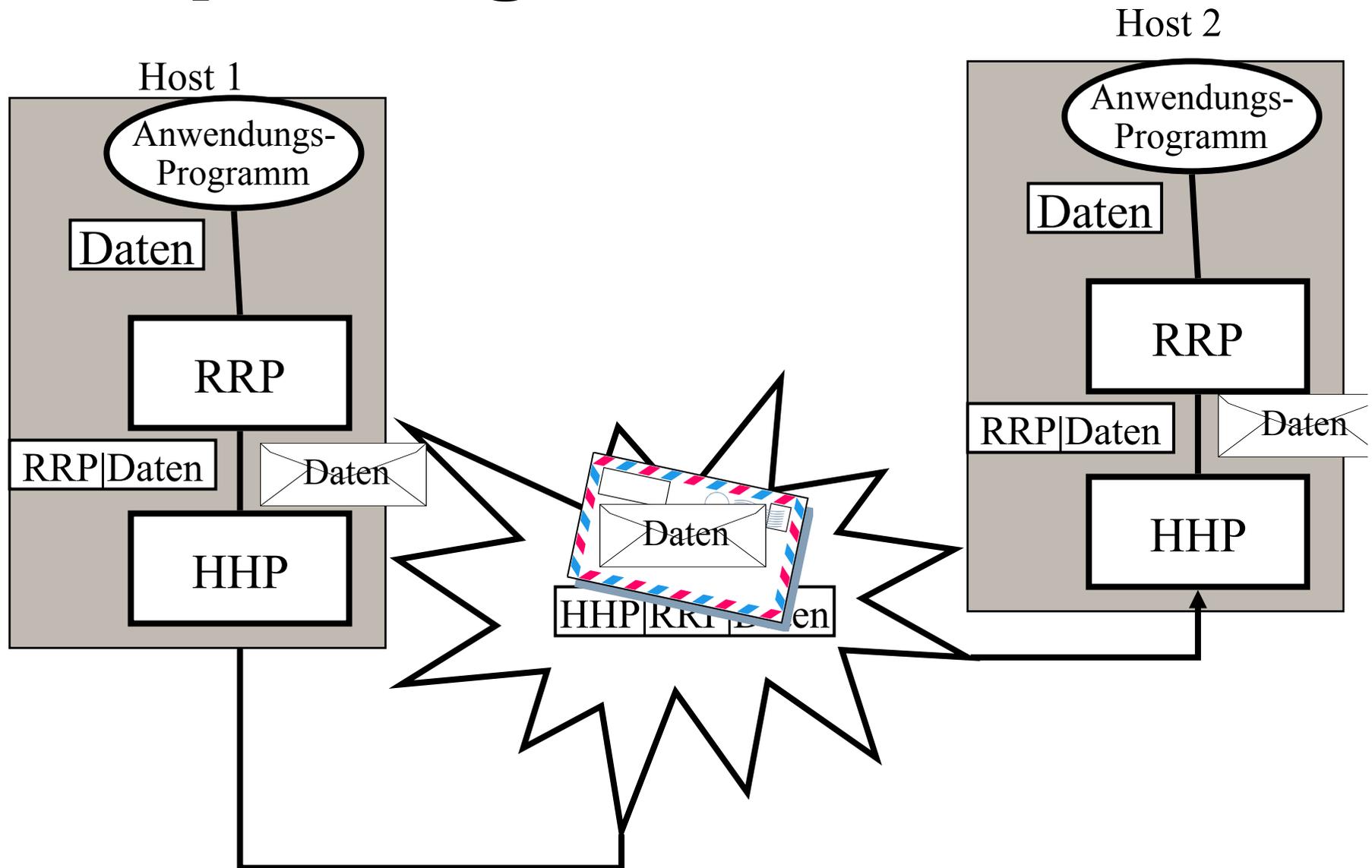
# Inter-Networking



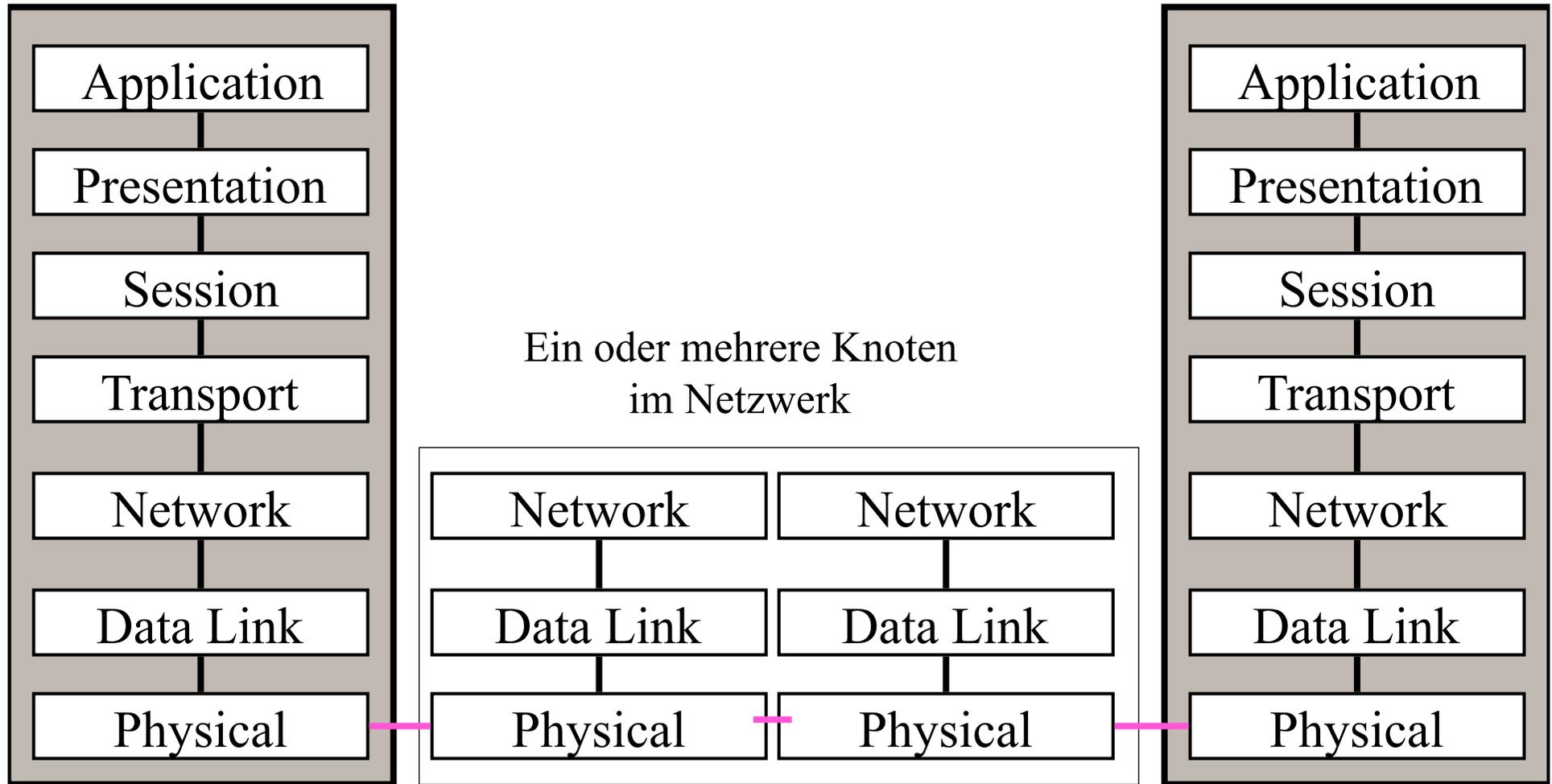
# Verkapselung von Nachrichten



# Verkapselung von Nachrichten



# ISO/OSI Netzwerk-Schichten- Architektur (Open System Interconnection)



# ISO/OSI-Schichtenarchitektur

- Jede Schicht bietet Dienste für die nächst-höhere Schicht an
- logisch gesehen kommuniziert ein Prozess nur mit einem Prozess derselben Schicht auf einem anderen Rechner. Dazu wird ein Protokoll definiert.
- Physisch gesehen kommuniziert ein Prozess nur mit der nächstniedrigeren Schicht auf demselben Rechner
- Ein Protokoll definiert Format und Bedeutung der Nachrichten innerhalb einer Schicht.

# ISO/OSI-Schichtenarchitektur

- Bit-Übertragungsschicht (physical Layer)
  - Bit/Signal-Sequenzen über ein physisches Medium
    - Twisted Pair, Coaxial, Glasfaser, Satellit, ...
  - Synchronisation
  - Medium Access Control
- Sicherungsschicht (Data link Layer)
  - Zusammenfassung von Daten zu Blöcken
  - Blocksynchronisation
  - Fehlererkennung und -Korrektur
    - Checksum- oder Hamming-Codierung
  - Link-Management für verbindungsorientierte Dienste

# ISO/OSI-Schichtenarchitektur

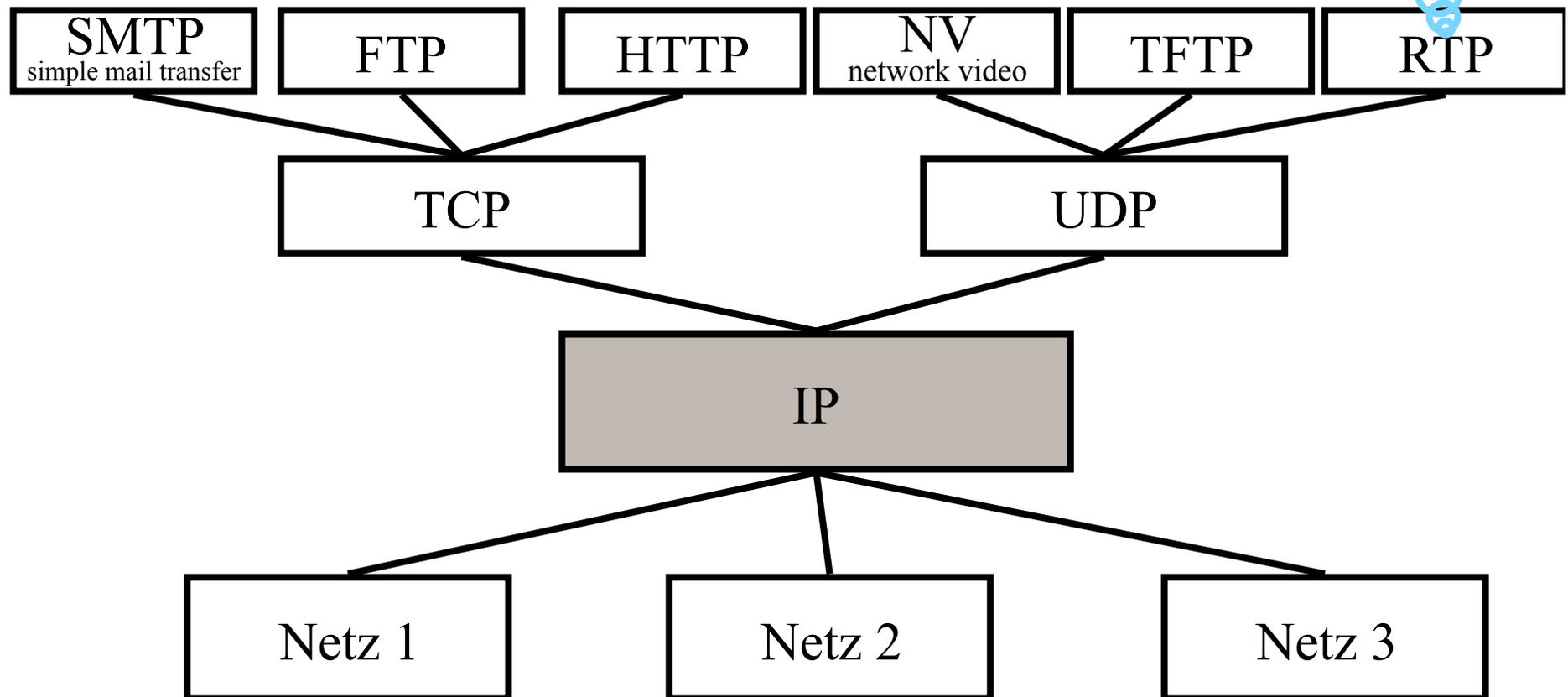
- Vermittlungsschicht (Network Layer)
  - Routing
  - Behandlung von Überlastsituationen innerhalb eines Netzwerks
  - Internetworking
    - Bridges, Gateways zum Übergang zwischen 2 Netzwerken,
    - Anpassung von Differenzen in den Protokollen)
  - Beispielprotokoll: IP
- Transportschicht /Transport Layer)
  - Aufbau und Erhaltung einer virtuellen Verbindung
  - vollkommen netzwerkunabhängiger Transportmechanismus
  - Adressierung
  - Beispielprotokolle: TCP, UDP

# ISO/OSI-Schichtenarchitektur

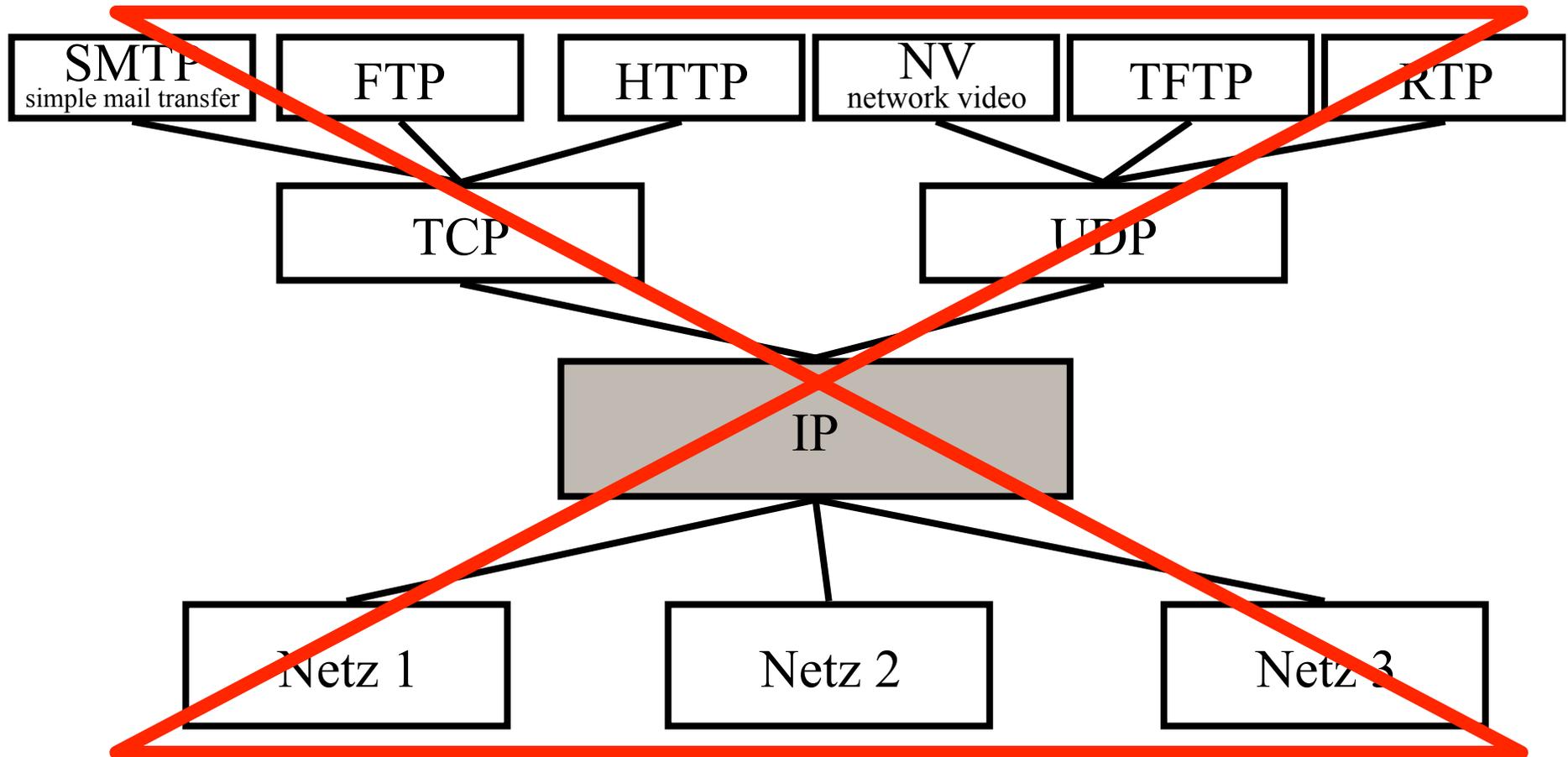
- Kommunikationsschicht (Session Layer)
  - Sitzungen (sessions), die mehrere Transport-Verbindungen umfassen können
  - Beispiele: RPC (Remote Procedure Call), RMI, rlogin
- Darstellungsschicht
  - Kryptographie
    - aber SSL, TLS ist auf Transportschicht
    - und Ipsec ist sogar auf Vermittlungsschicht
  - Komprimierung
- Anwendungsschicht (Application Layer)
  - FTP
  - SMTP (email)
  - HTTP
  - NFS

# Internet Protokoll-Graph

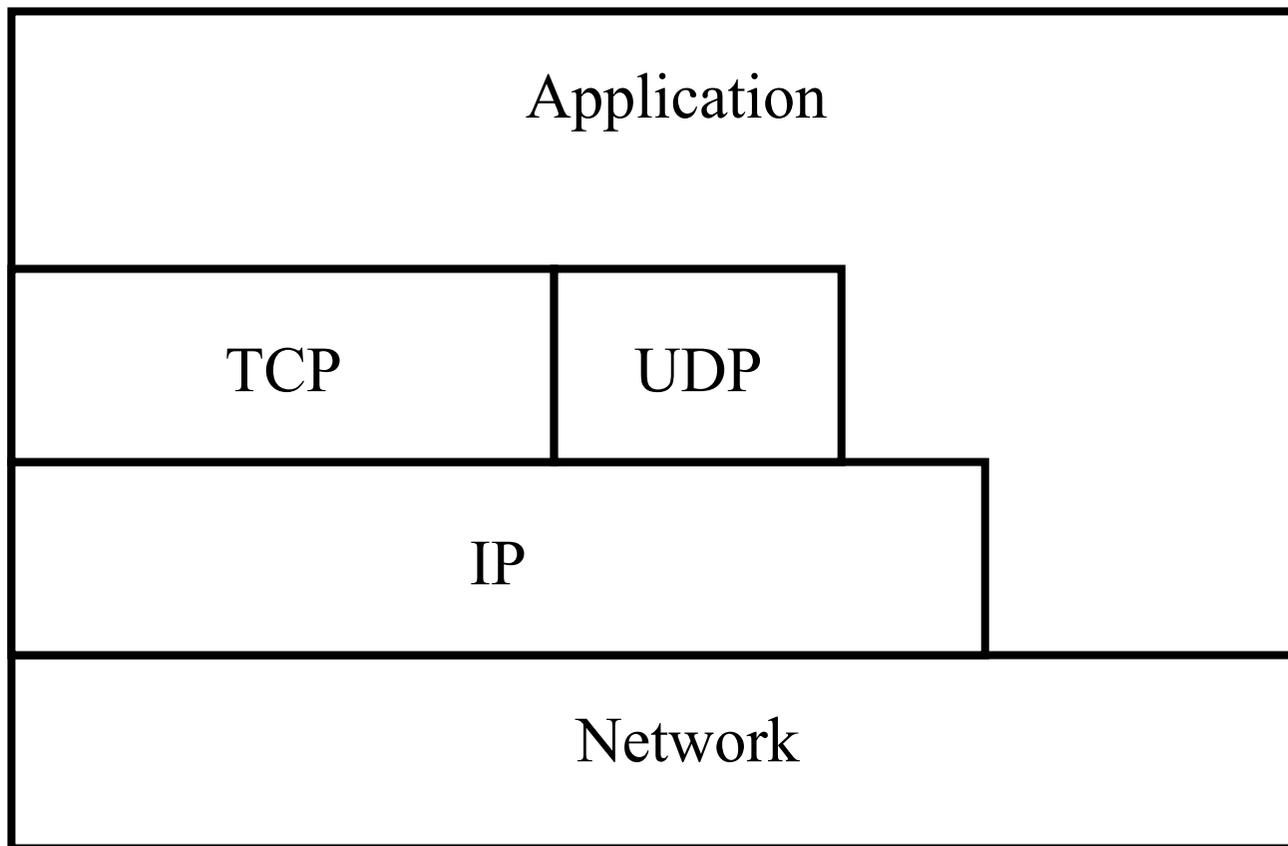
Real-Time  
Transport  
Protocol



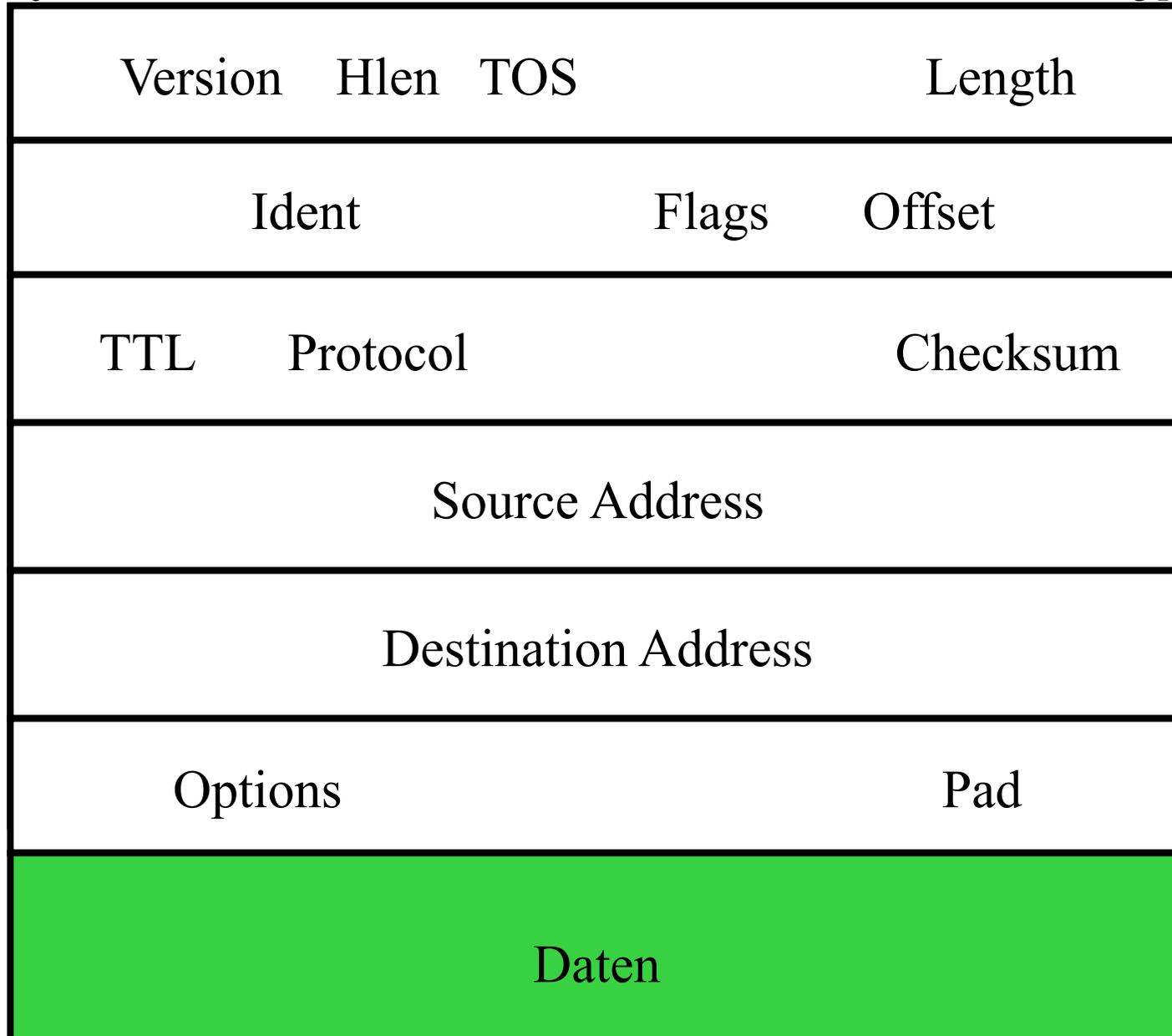
# Sanduhr-Architektur



# Internet-Protokoll



# IP Paket-Header-Format 31



# IP-Adressen (IPv4: 32 Bit)

0	Network	Host
---	---------	------

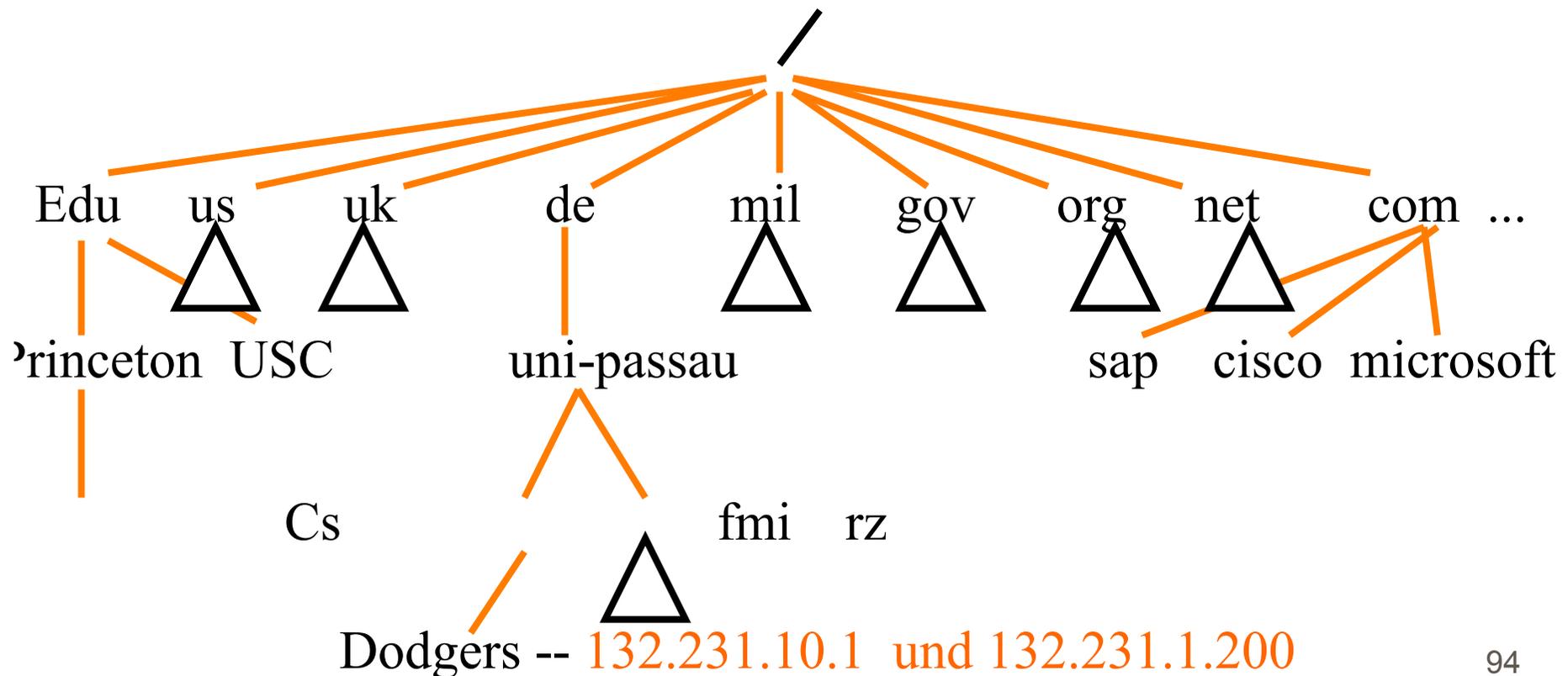
10	Network	Host
----	---------	------

110	Network	Host
-----	---------	------

- Class A: 7 Bit Network, 24 Host
- Class B: 14 Bit Network, 16 Host
- Class C: 21 Bit Network, 8 Host

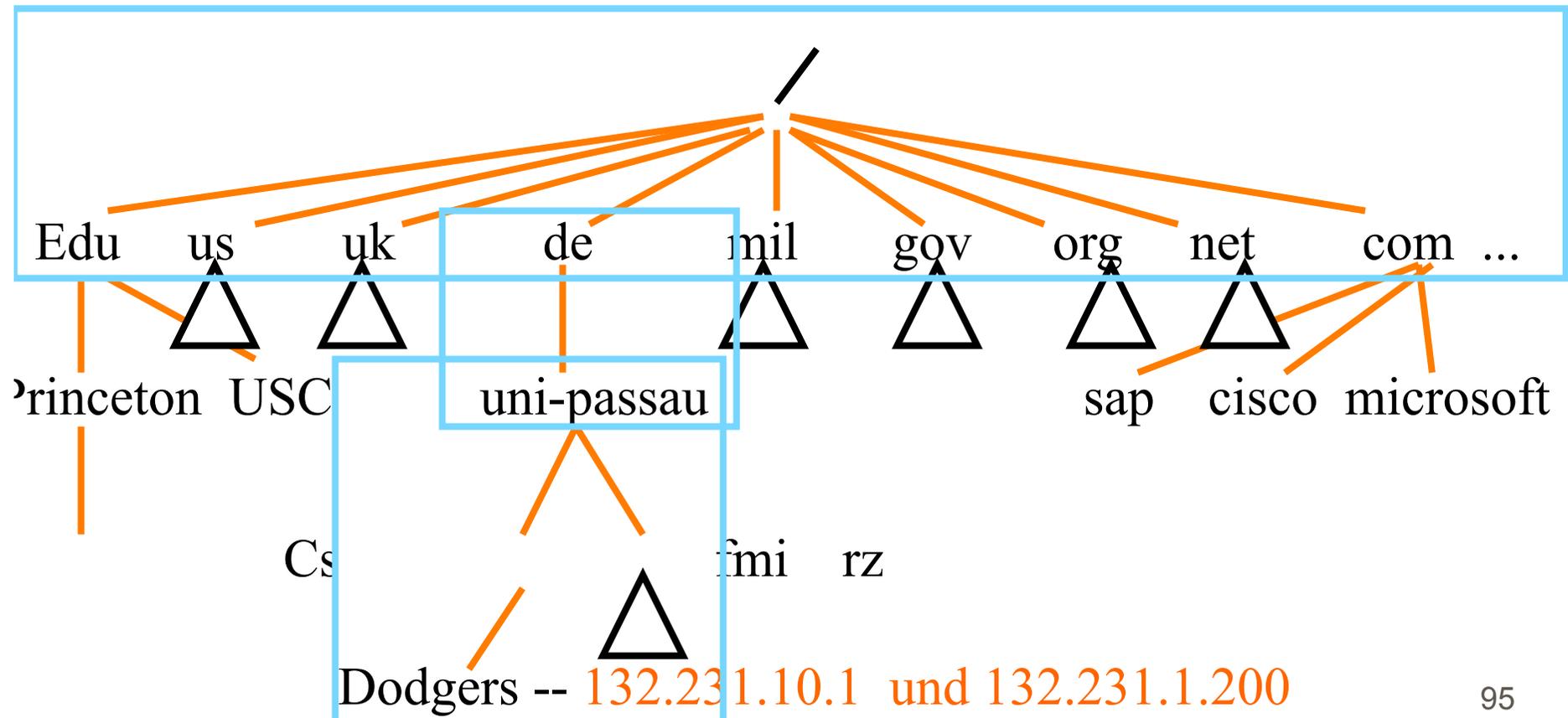
# Domain Namen --- IP-Nr

- ICANN: Internet Corporation for Assigned Names and Numbers



# Domain Namen --- IP-Nr Nameserver DNS

- ICANN: Internet Corporation for Assigned Names and Numbers

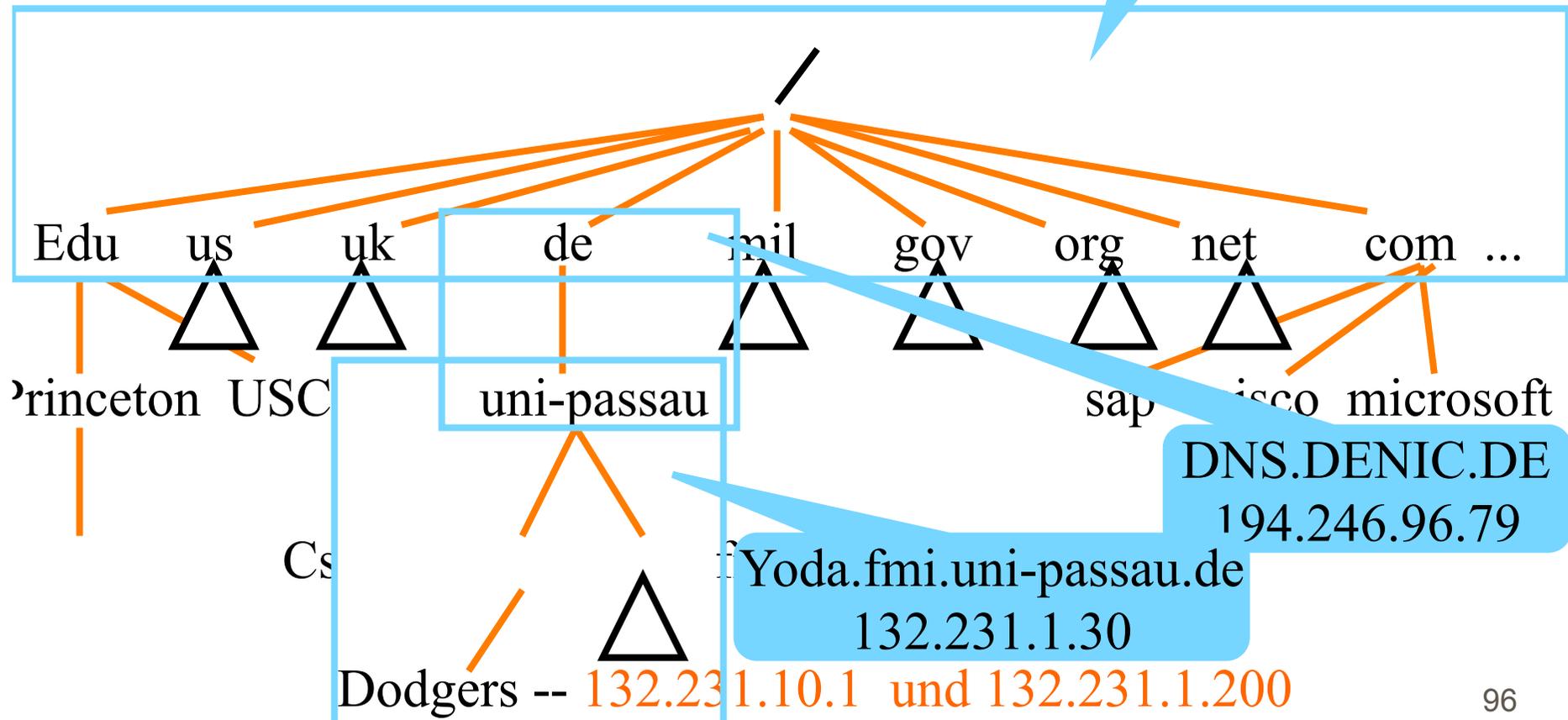


# Domain Namen --- IP-Nr

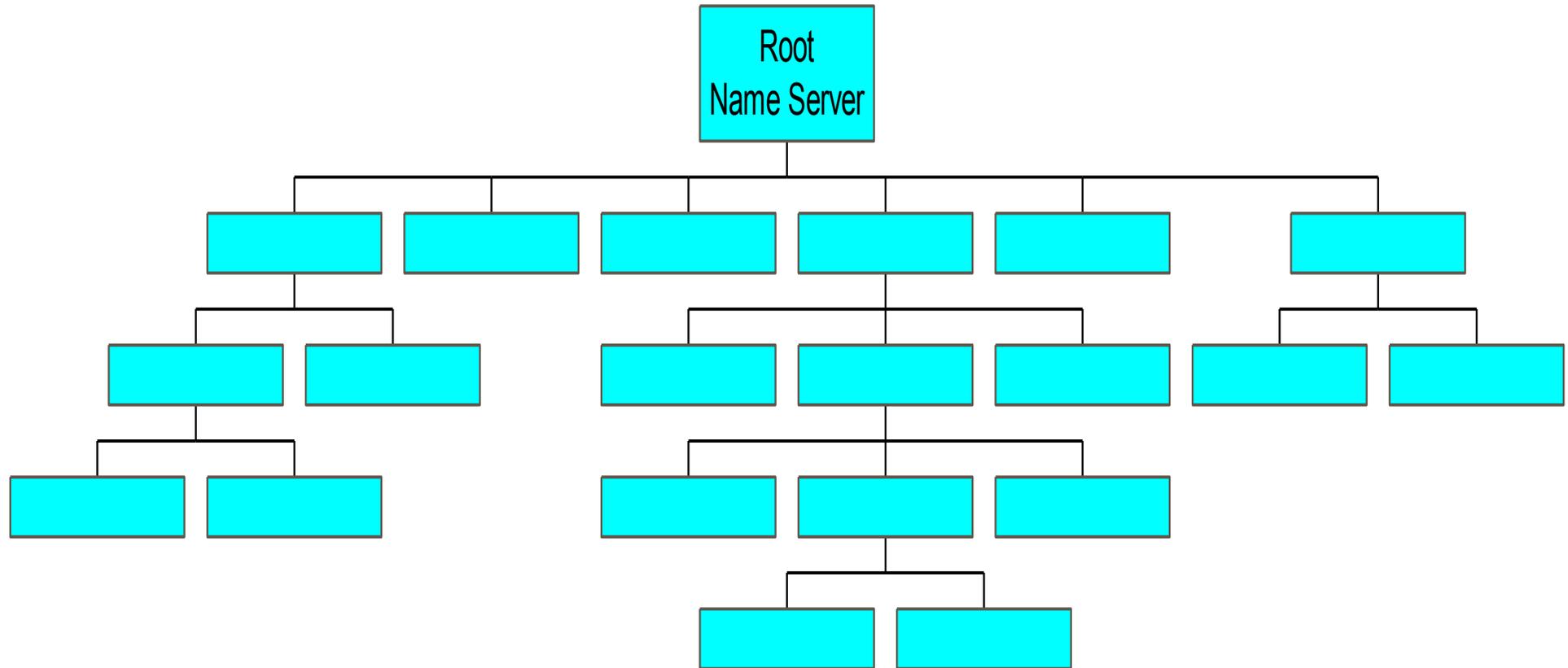
## Nameserver DNS

- ICANN: Internet Corporation for Names and Numbers

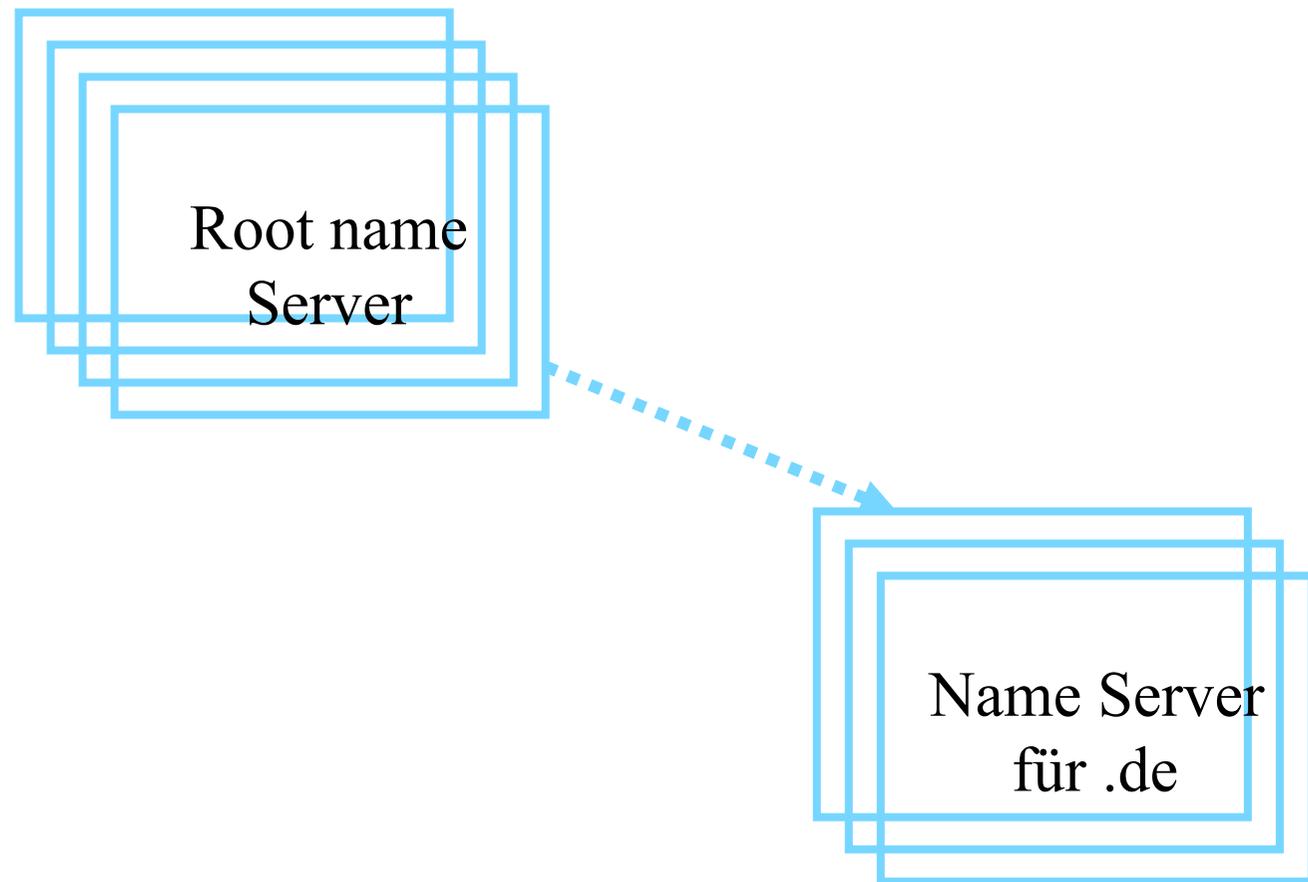
A.root-servers.net  
198.41.0.4



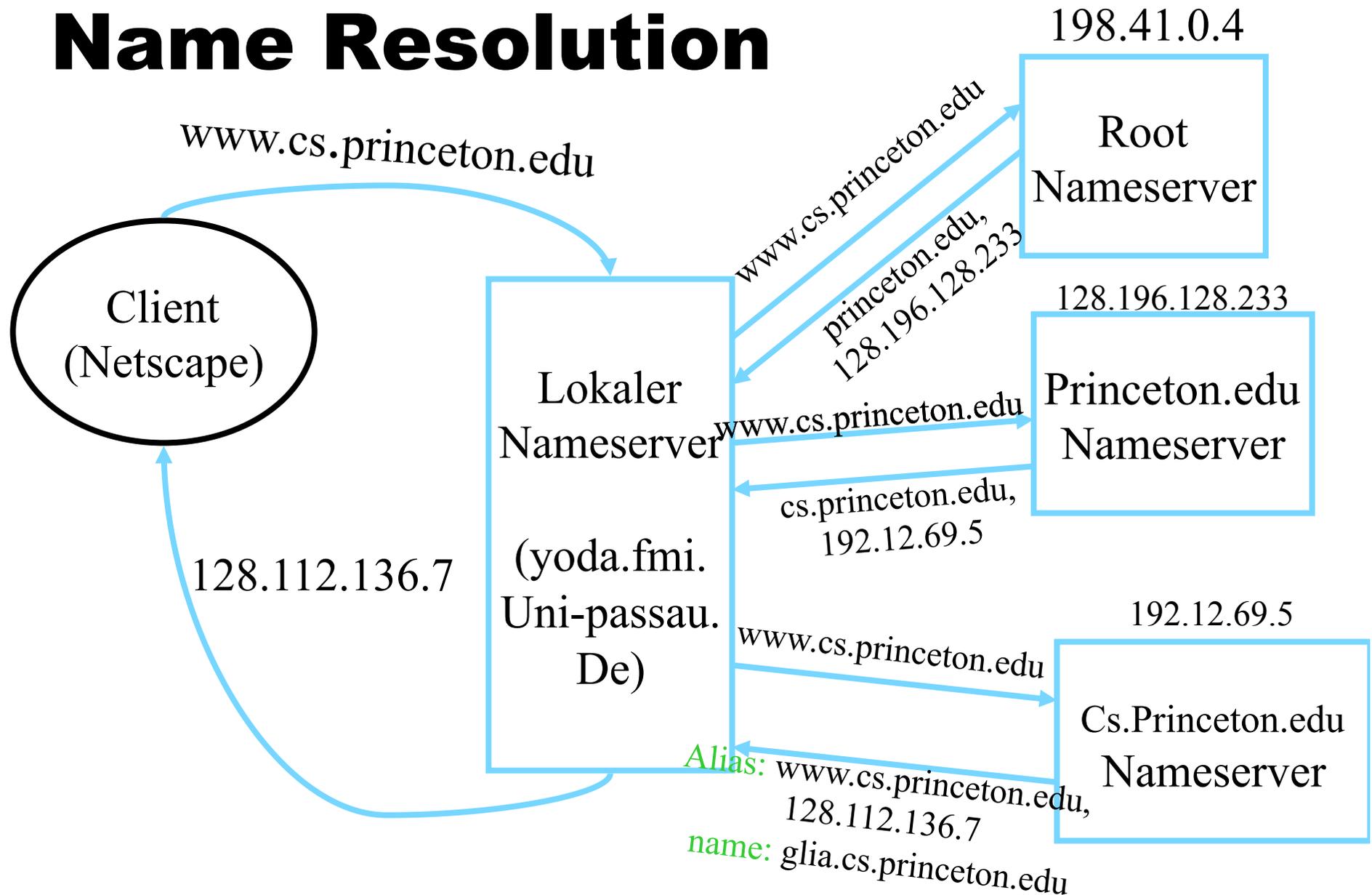
# Name-Server-Hierarchie



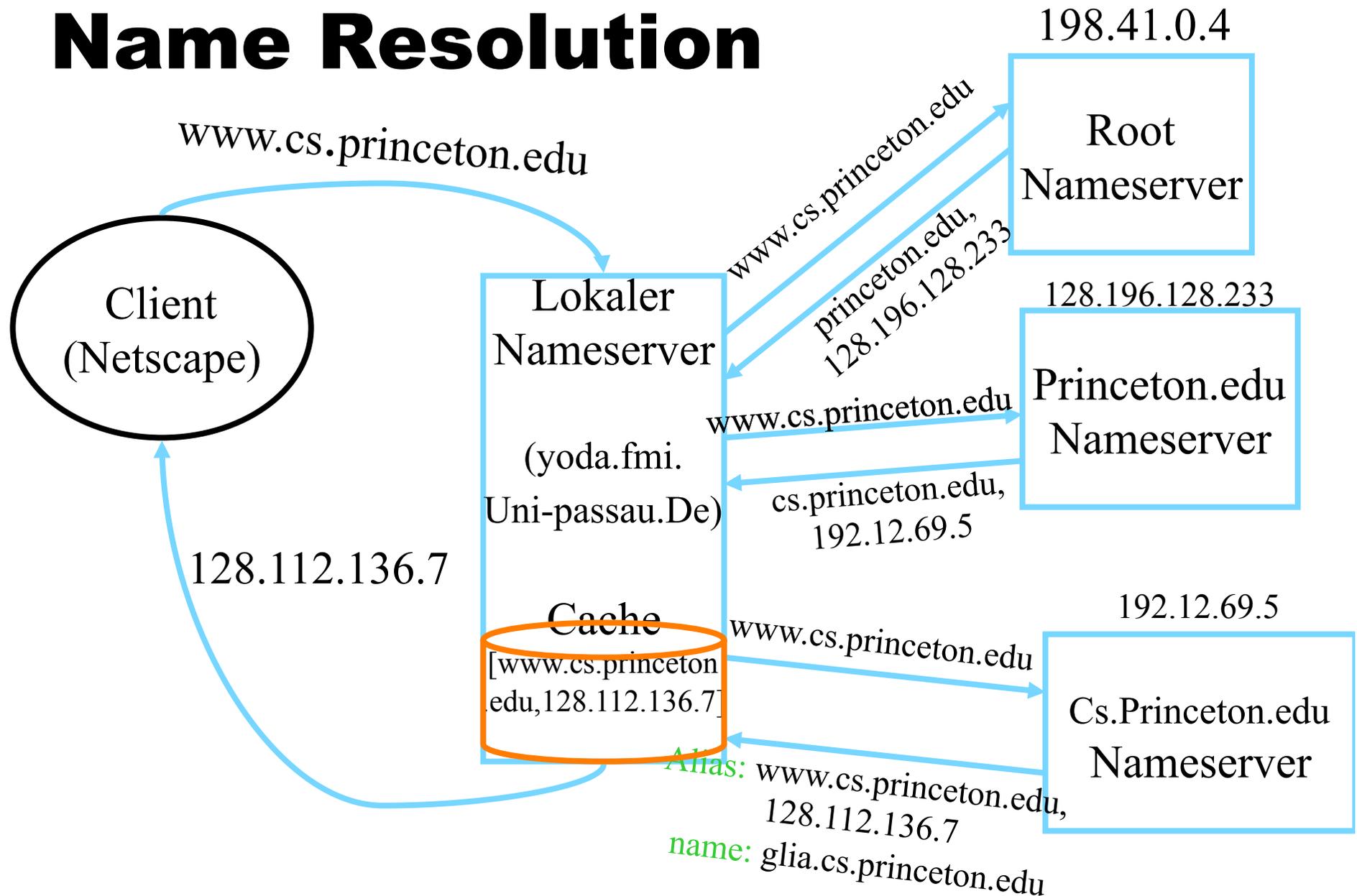
# Fehlertoleranz: Redundanz/ Replikation der DNS



# Namensauflösung Name Resolution



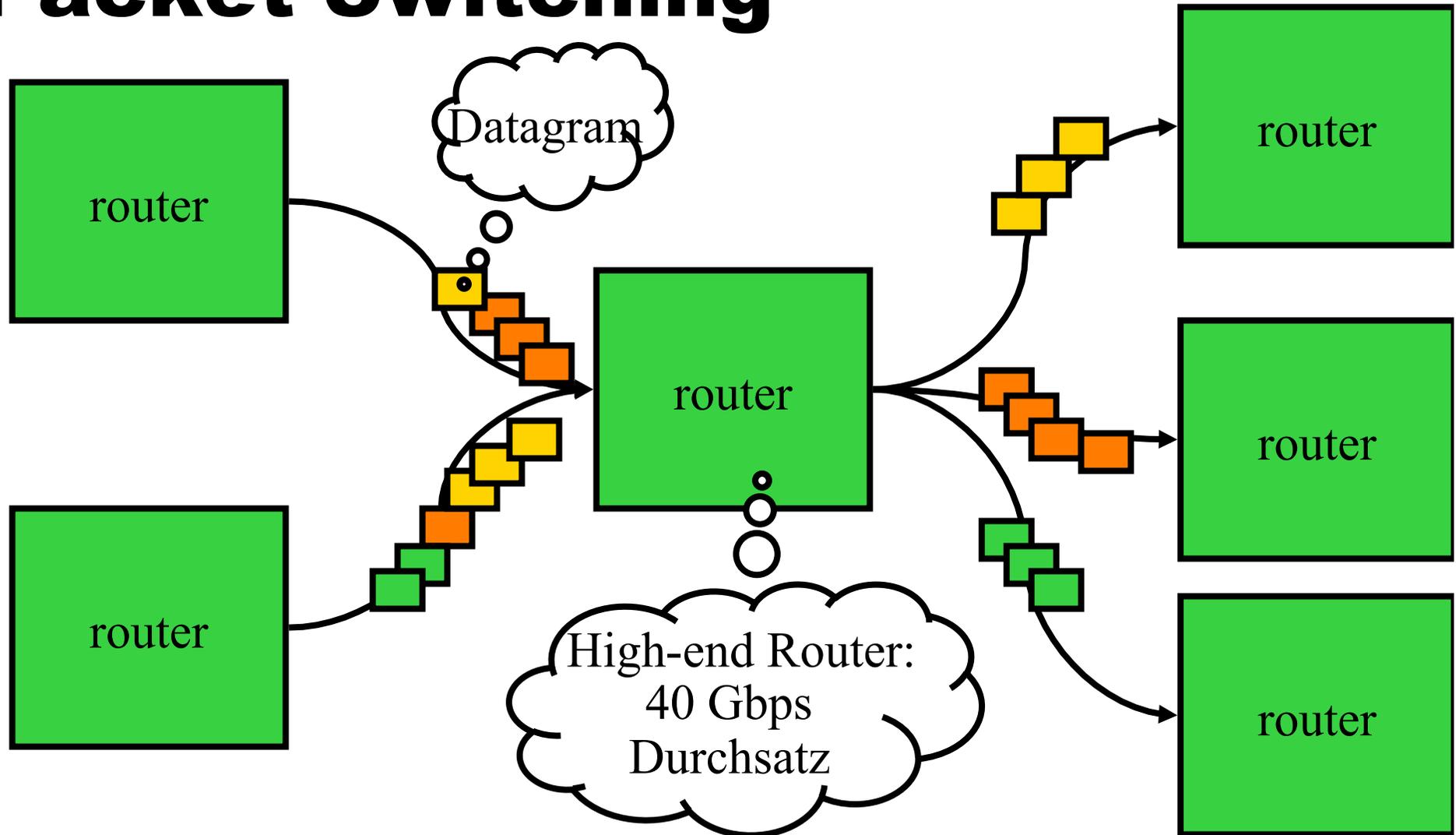
# Namensauflösung Name Resolution



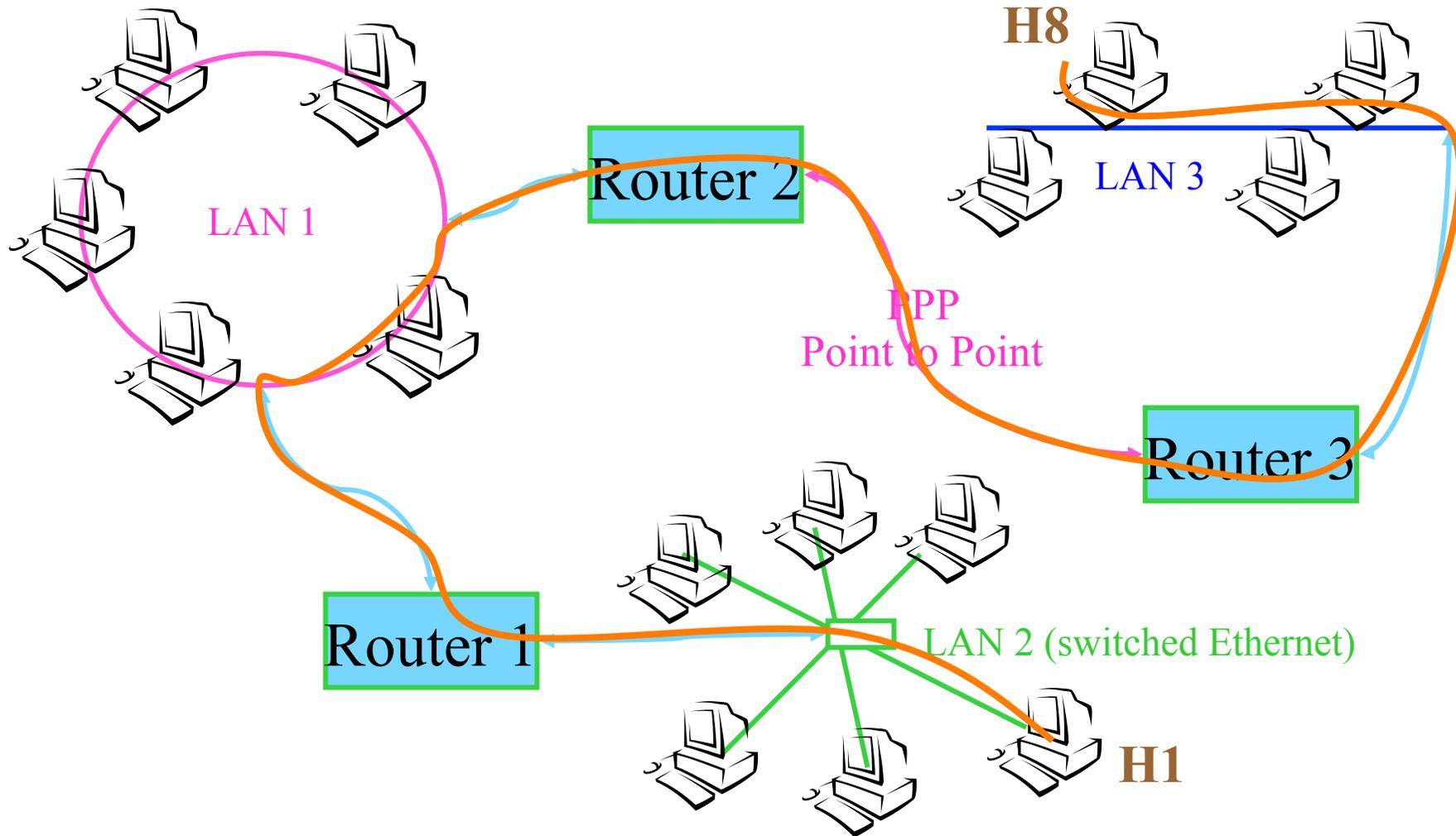
# IP Version 6 (IPv6)

- Neues Header Format
- 128 Bit Adressen
- im Vergleich zu 32 Bit-Adressen in IPv4
  - Adressen werden knapp
  - insgesamt sind theoretisch nur  $2^{32} \sim 4\text{Mrd}$  Rechner adressierbar
  - in Realität viel weniger (weil viele Adressen „vergeudet“ werden)
- Umstellung auf IPv6 im Gange
  - „Internet of Things“ benötigt viele Adressen!

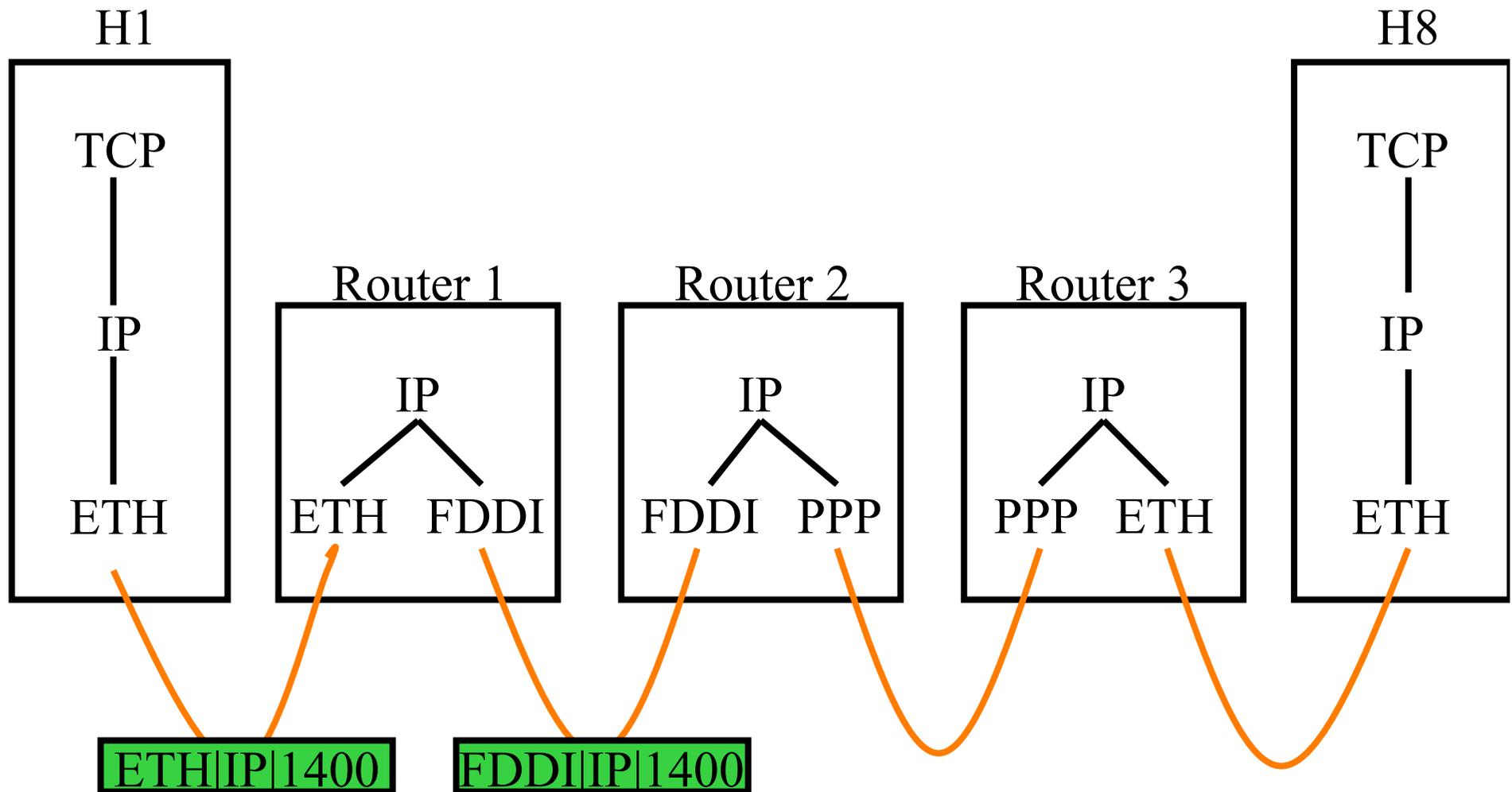
# Packet Switching



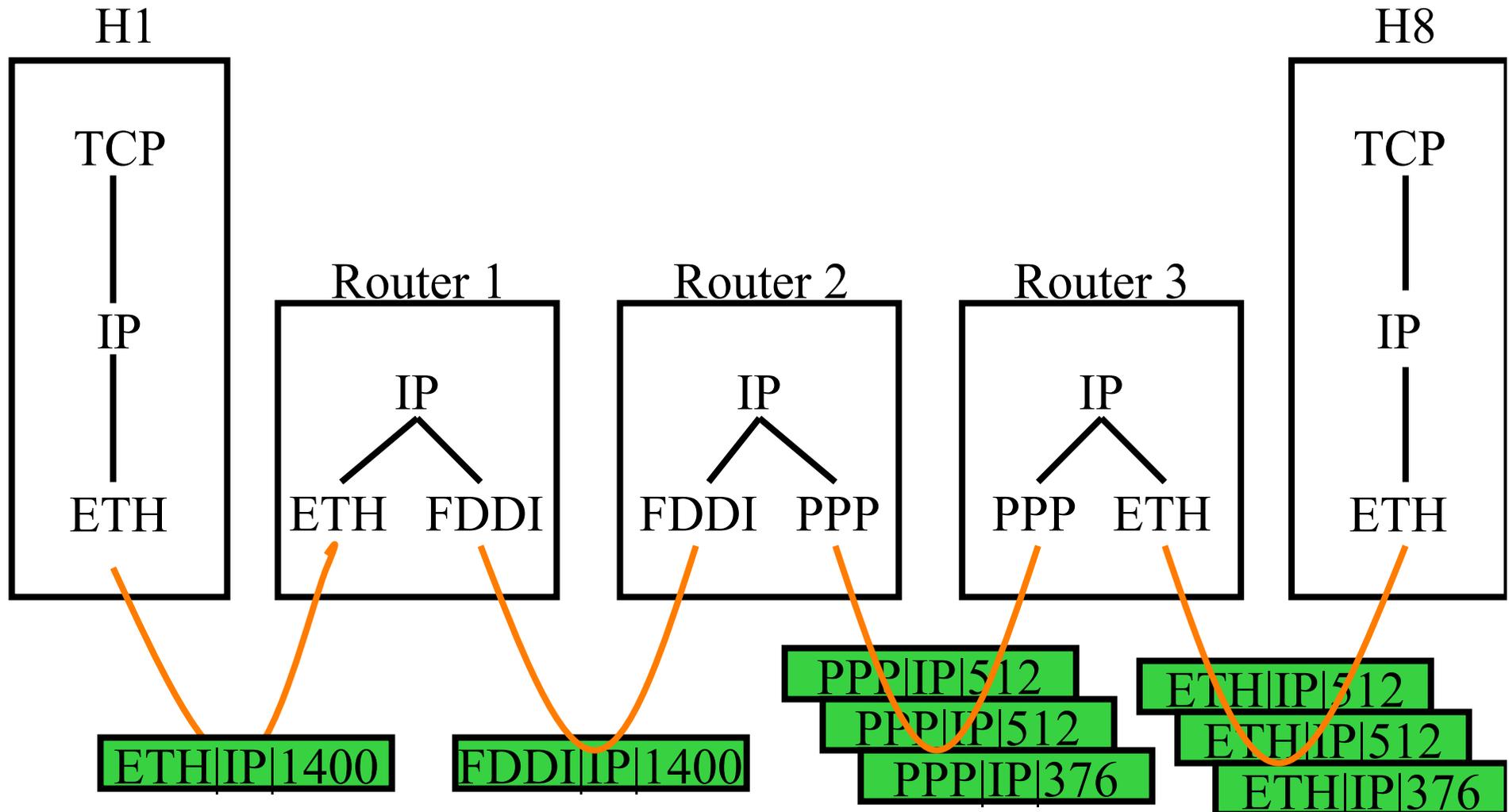
# Inter-Networking



# Transfer von H1 nach H8



# Transfer von H1 nach H8



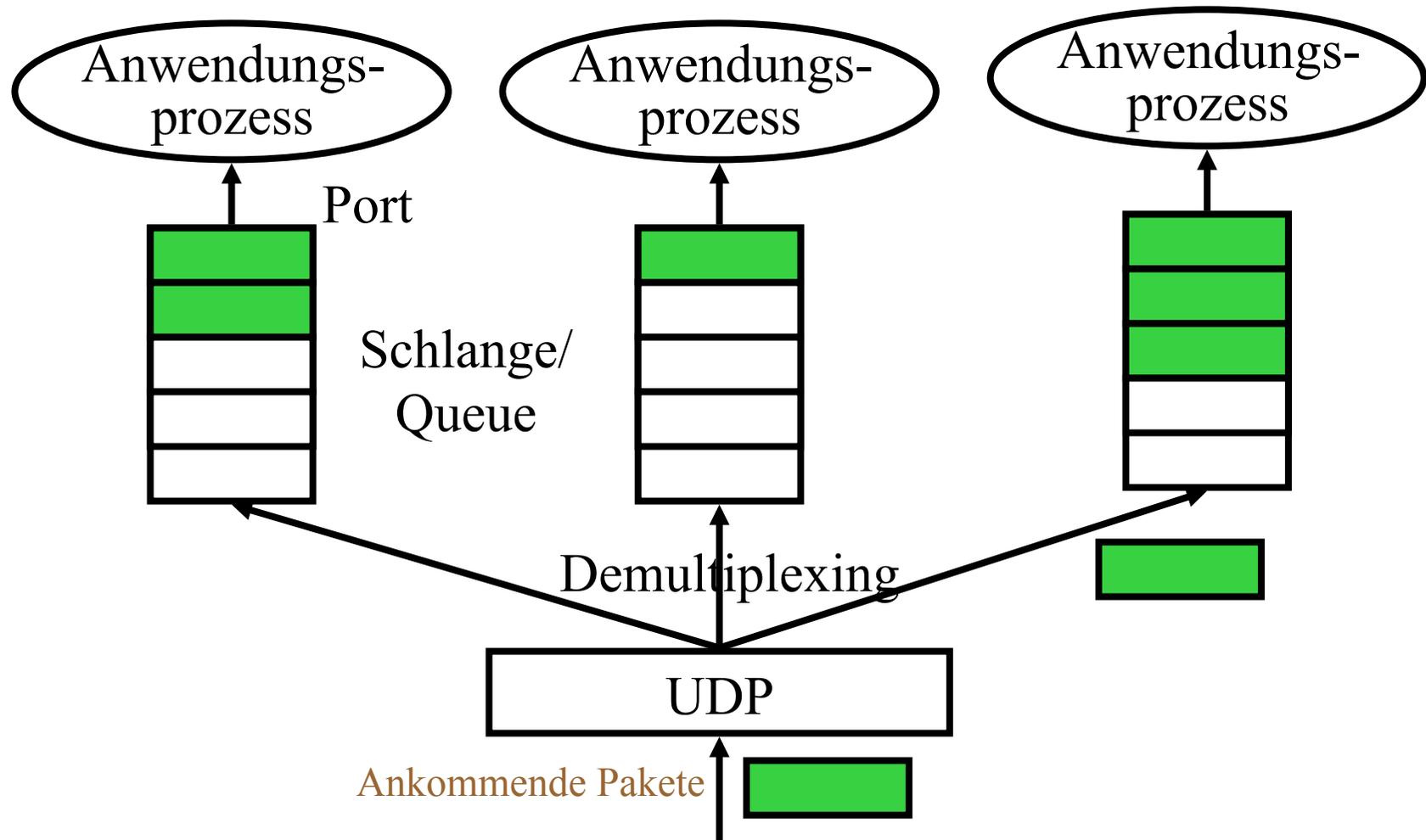
# Best Effort-Prinzip

- IP Pakete können verloren gehen
  - Router ist überlastet und „wirft“ Pakete weg
  - TTL (Time to Live) ist abgelaufen
    - Paket wird dann weggeworfen, damit es nicht ewig „kreist“
    - Hardware/Leitungs-Ausfall (Bagger oder ähnliches)
- Fehlertoleranz muss auf höheren Schichten (TCP) realisiert werden

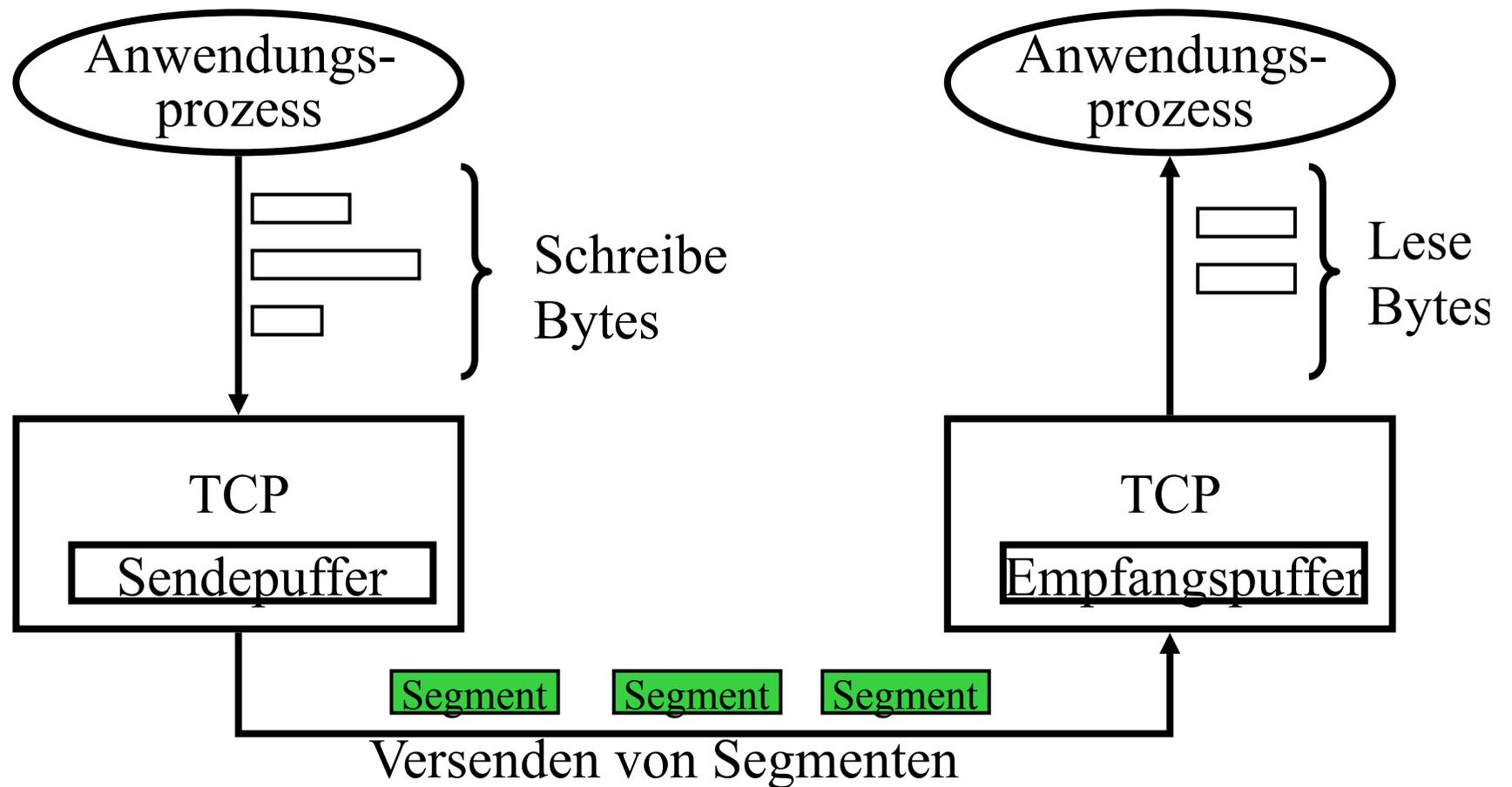
# End to End Protokolle

- UDP
  - IP + Demultiplexing
  - d.h. mehrere parallele Prozesse auf dem gleichen Rechner
- TCP
  - virtuelle Verbindung von Rechner zu Rechner
  - Fehlertolerant

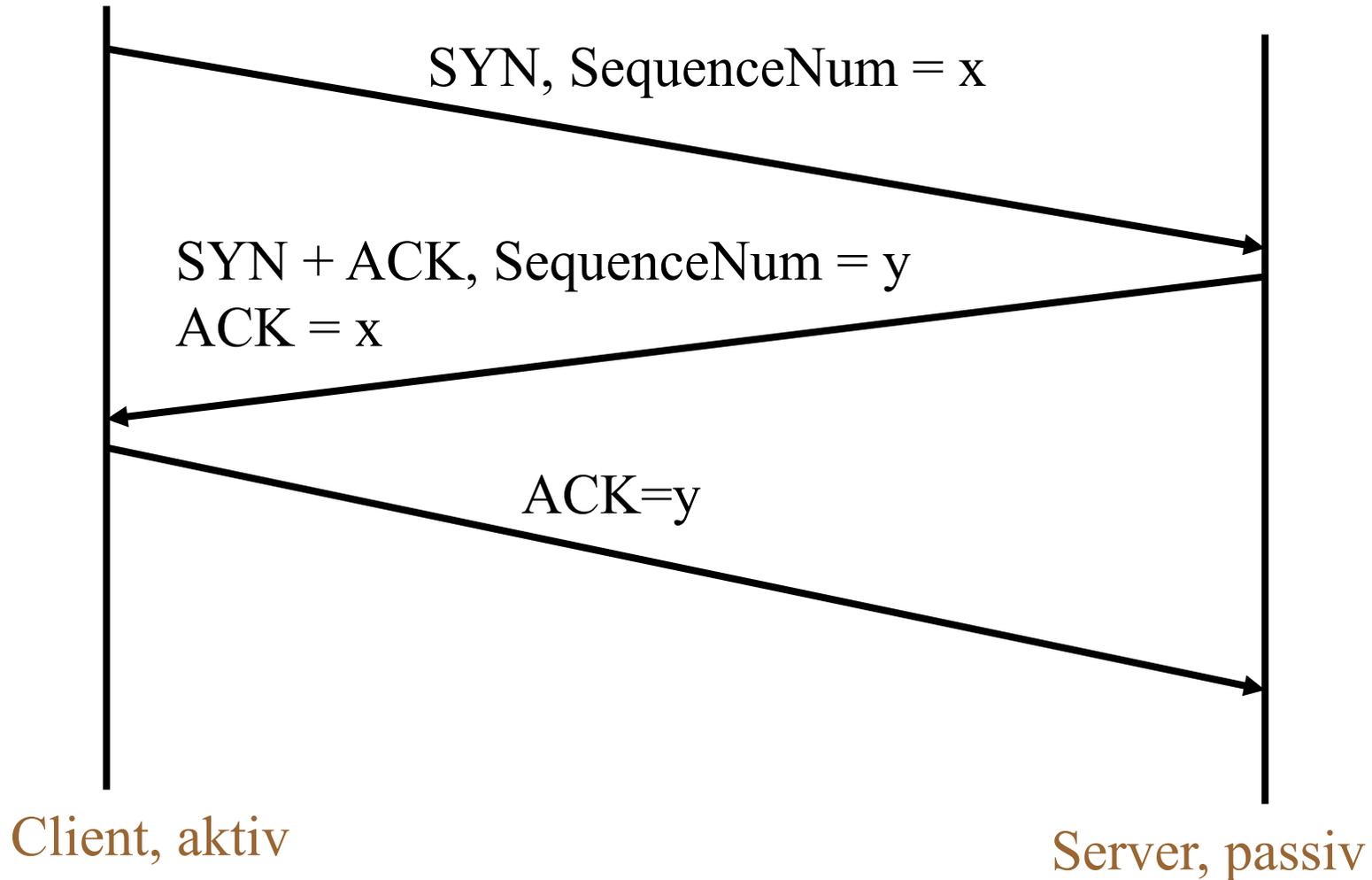
# UDP



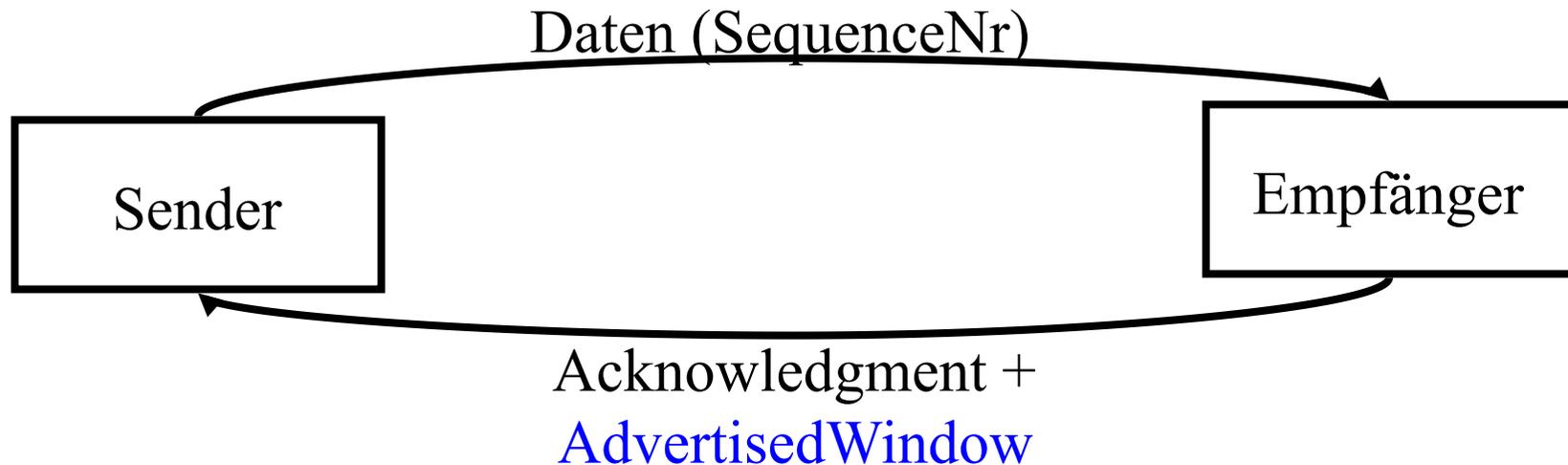
# TCP: Schreiben und Lesen



# Three-Way-Handshake für Verbindungsaufbau

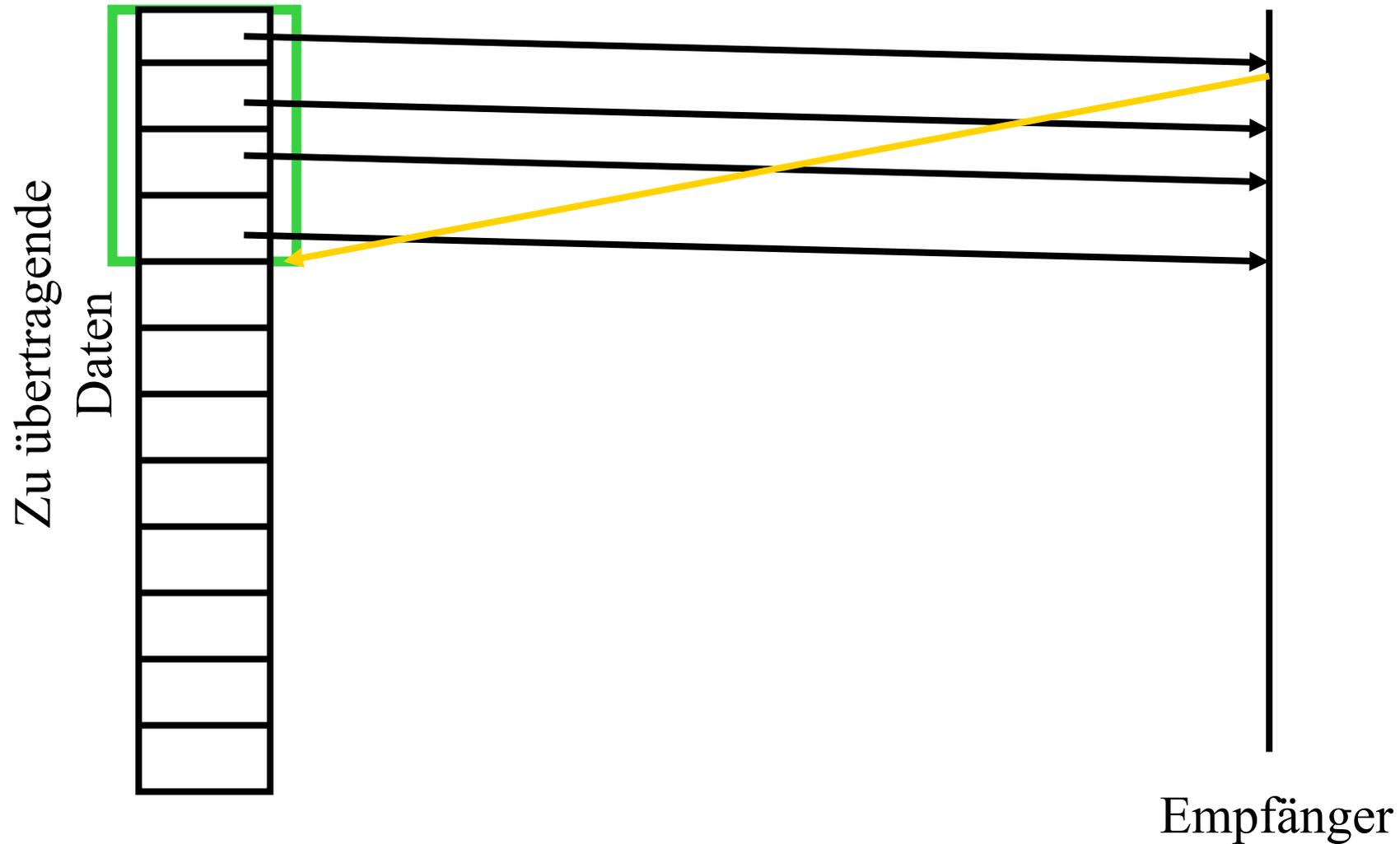


# Kommunikation zwischen Sender/Empfänger

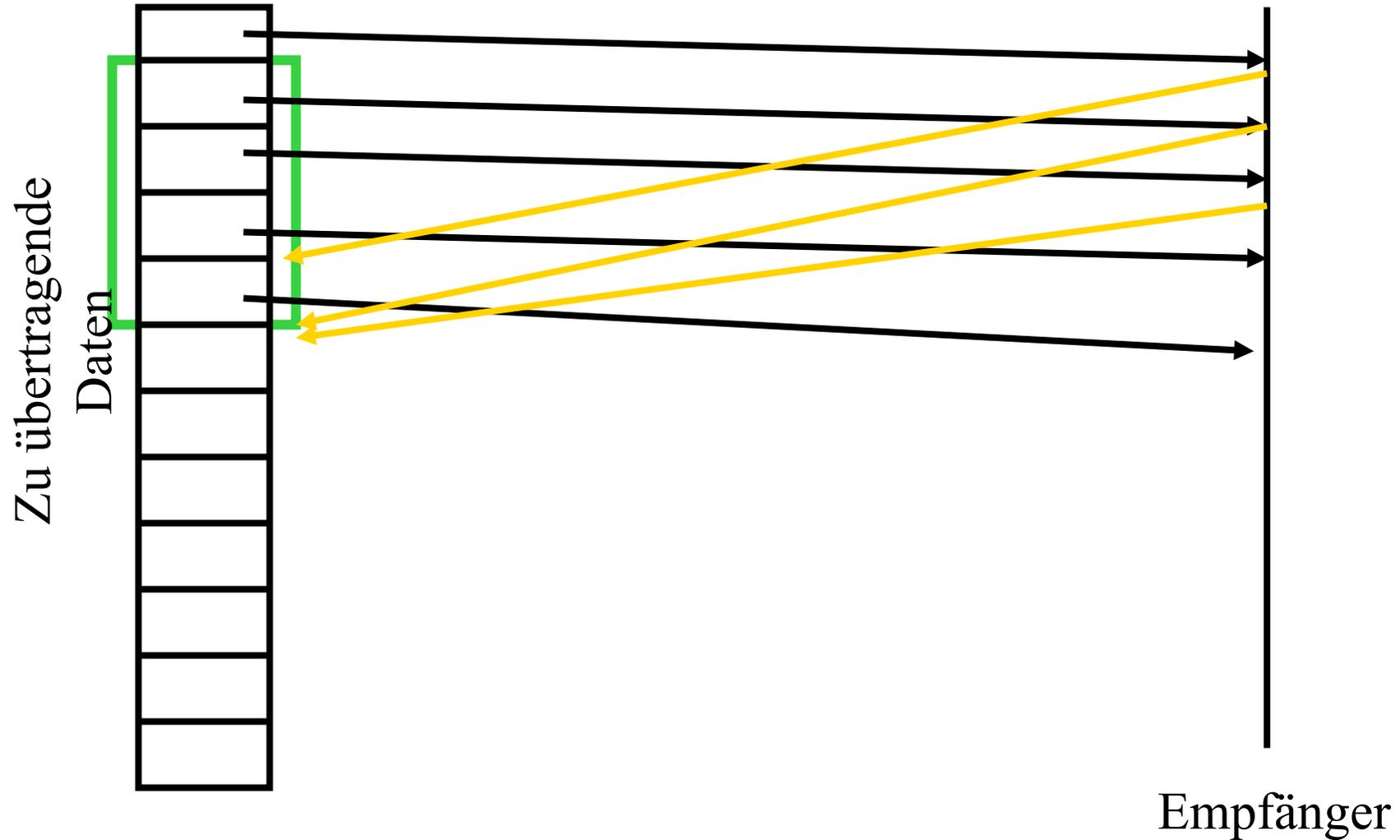


AdvertisedWindow: Verfügbarer Platz im Empfangspuffer  
des Empfängers

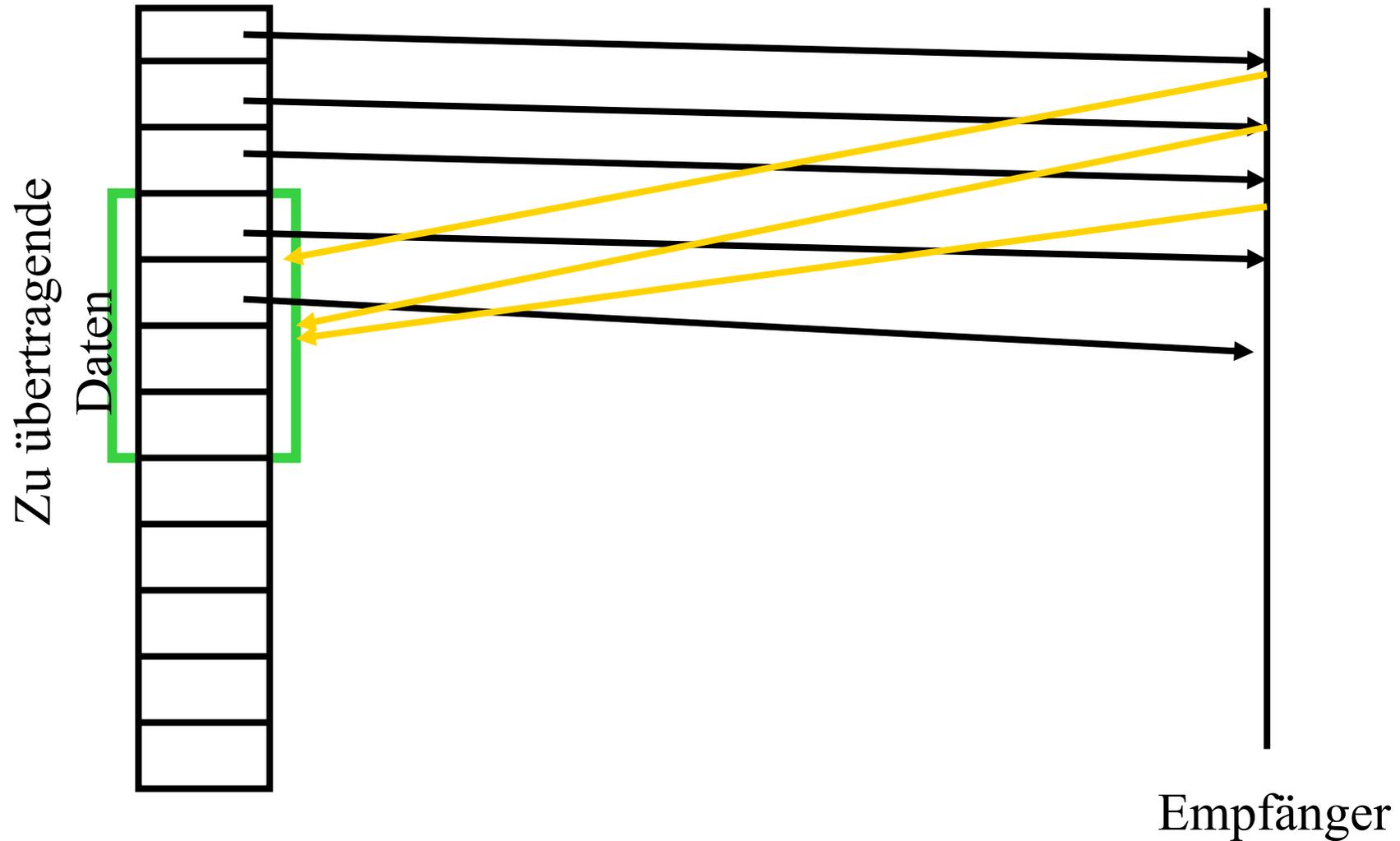
# Sliding Window Technik



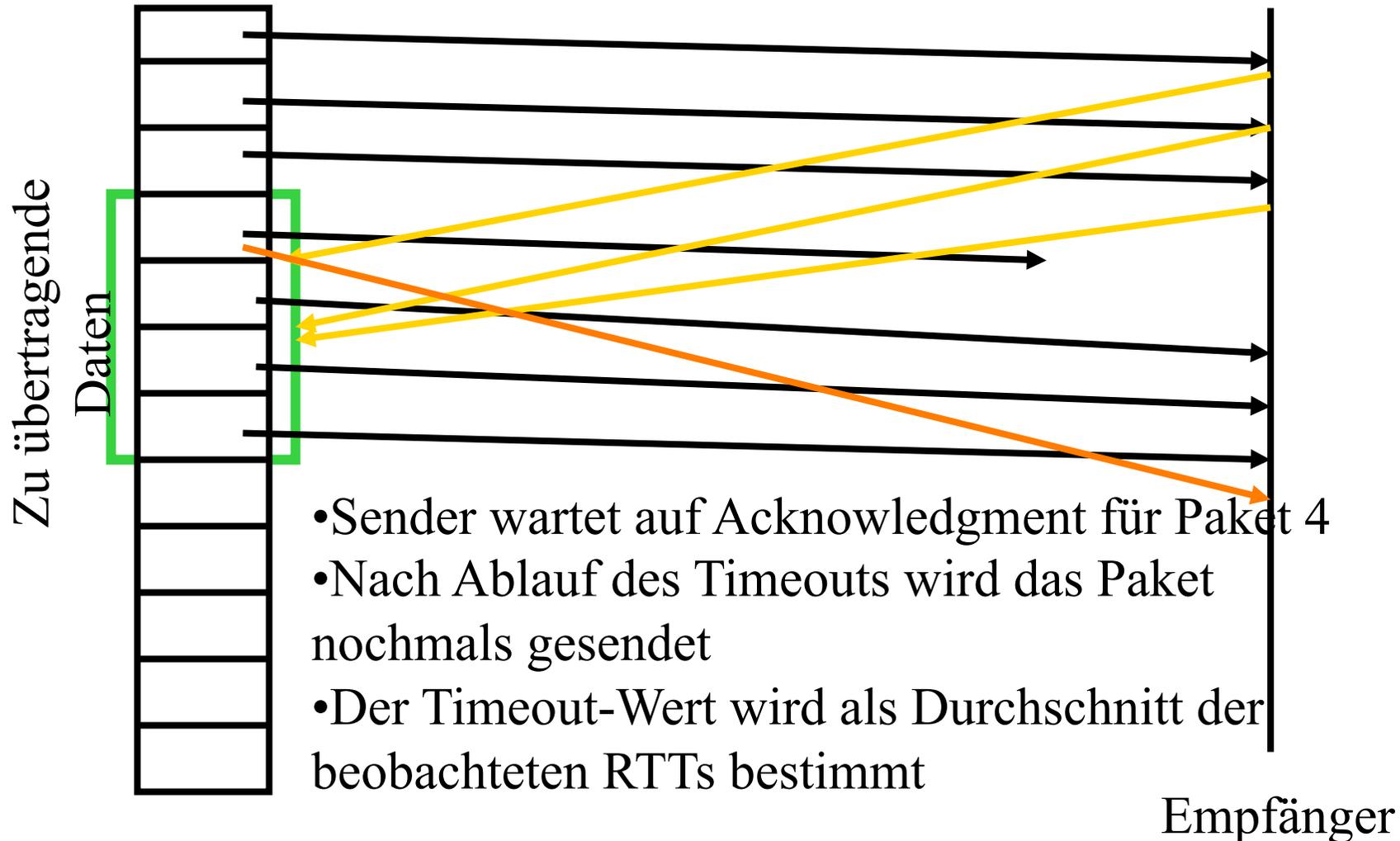
# Sliding Window Technik



# Sliding Window Technik



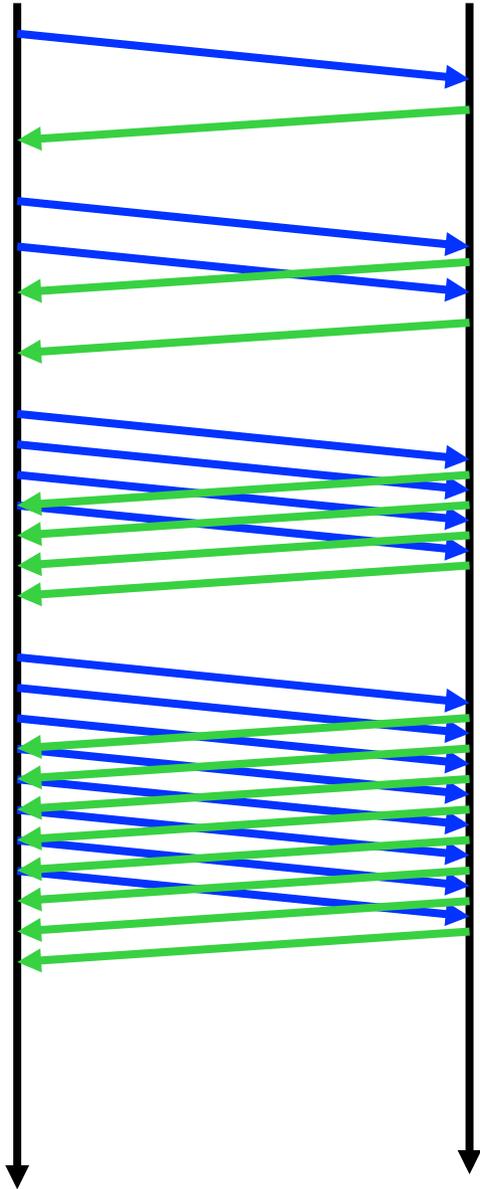
# Sliding Window Technik: Paket 4 gehe verloren



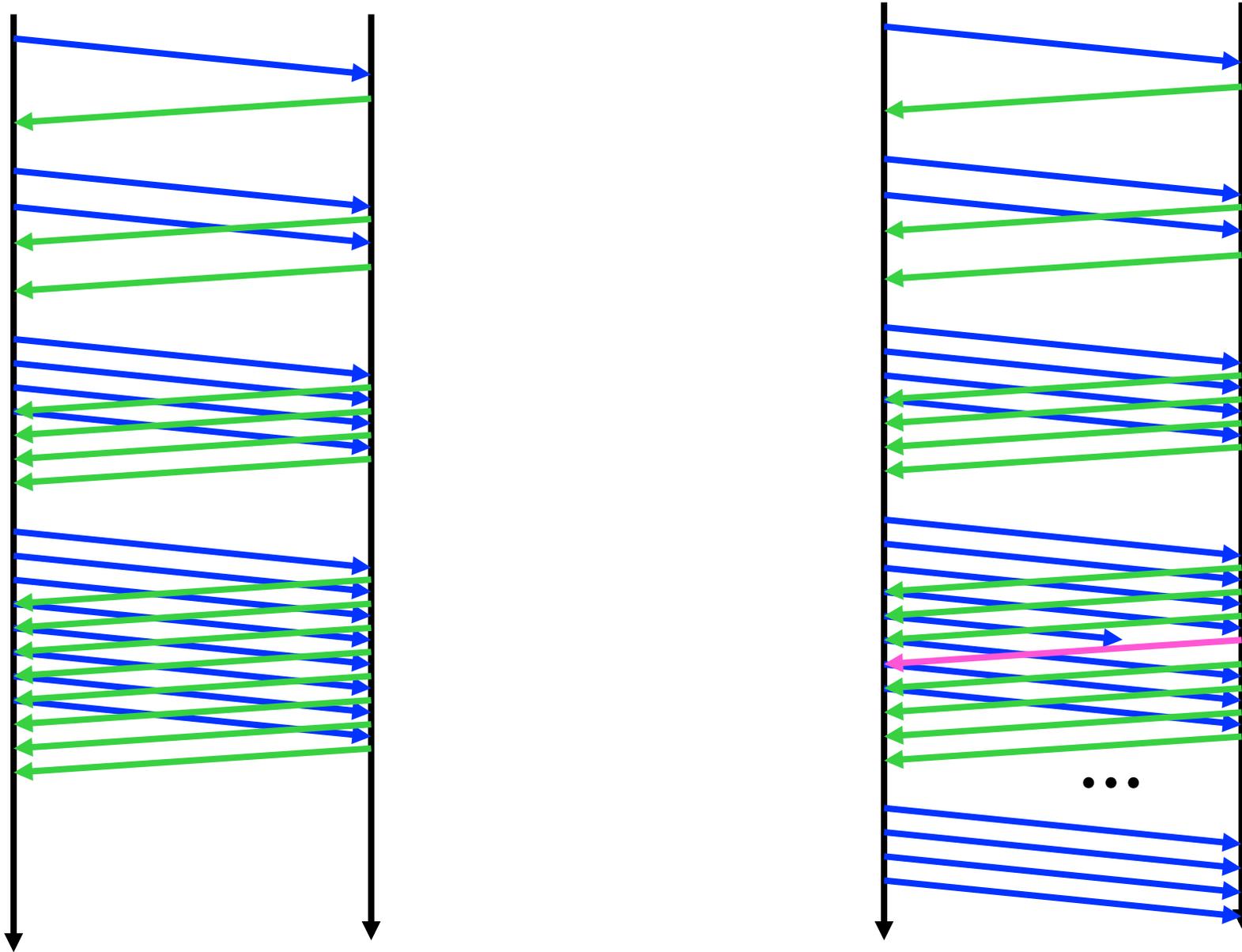
# Richtige Größe des Fensters/ Windows

- Die Größe ergibt sich als das Maximum aus
  - Advertised Window des Empfängers ( $\sim$ Puffer)
  - verfügbare Bandbreite \* RTT
- verfügbare Bandbreite wird durch „vorsichtiges Herantasten“ bestimmt
  - $W = 2, 4, 8, 16, 32, 64, 128, \dots$
  - Wenn Pakete verloren gehen, war man zu optimistisch und muß wieder halbieren
  - danach kann man wieder additiv vergrößern
- Bei jedem Paketverlust wird  $W$  halbiert
  - Congestion Control

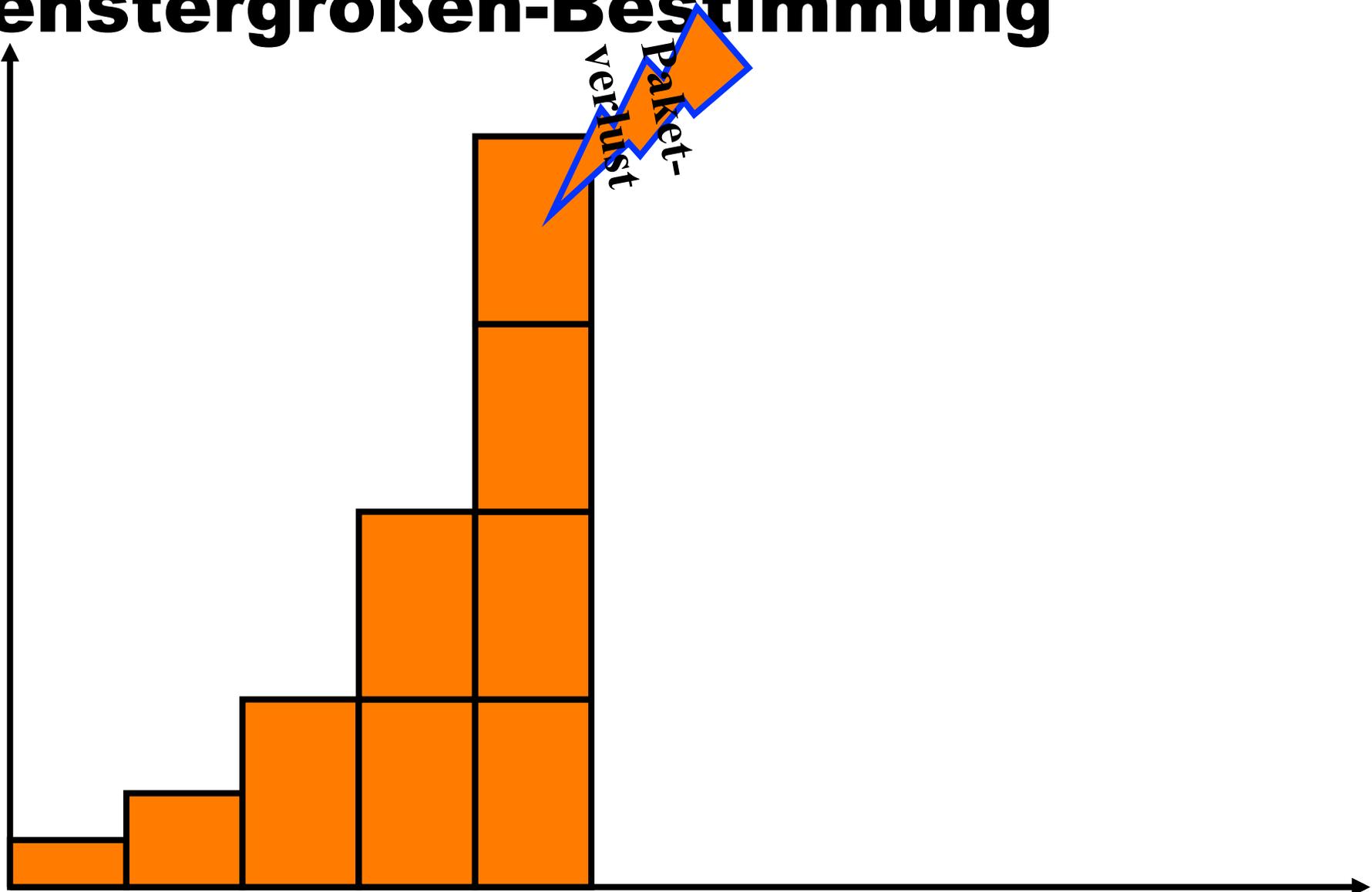
# Fenstergrößen-Bestimmung



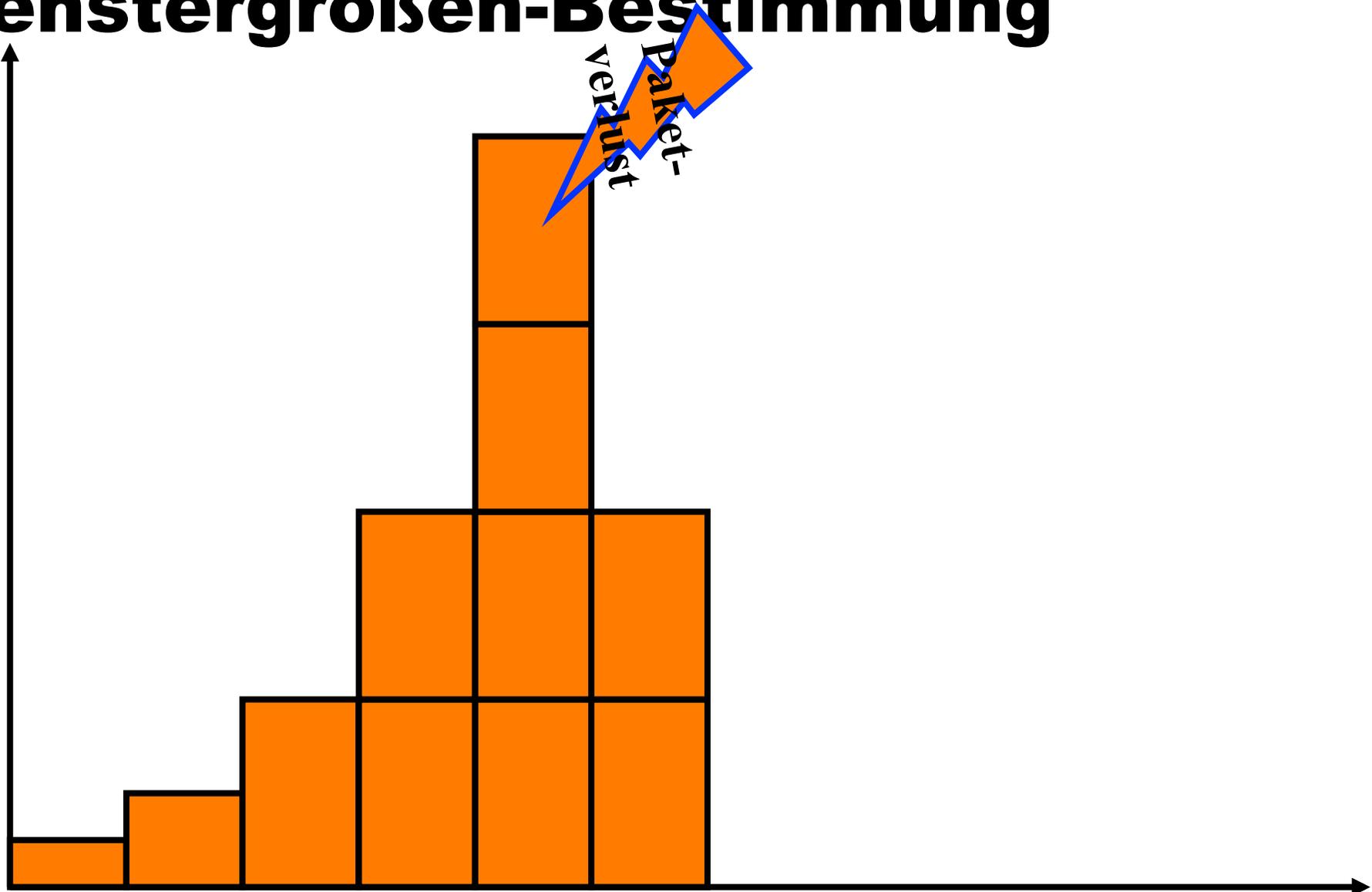
# Fenstergrößen-Bestimmung



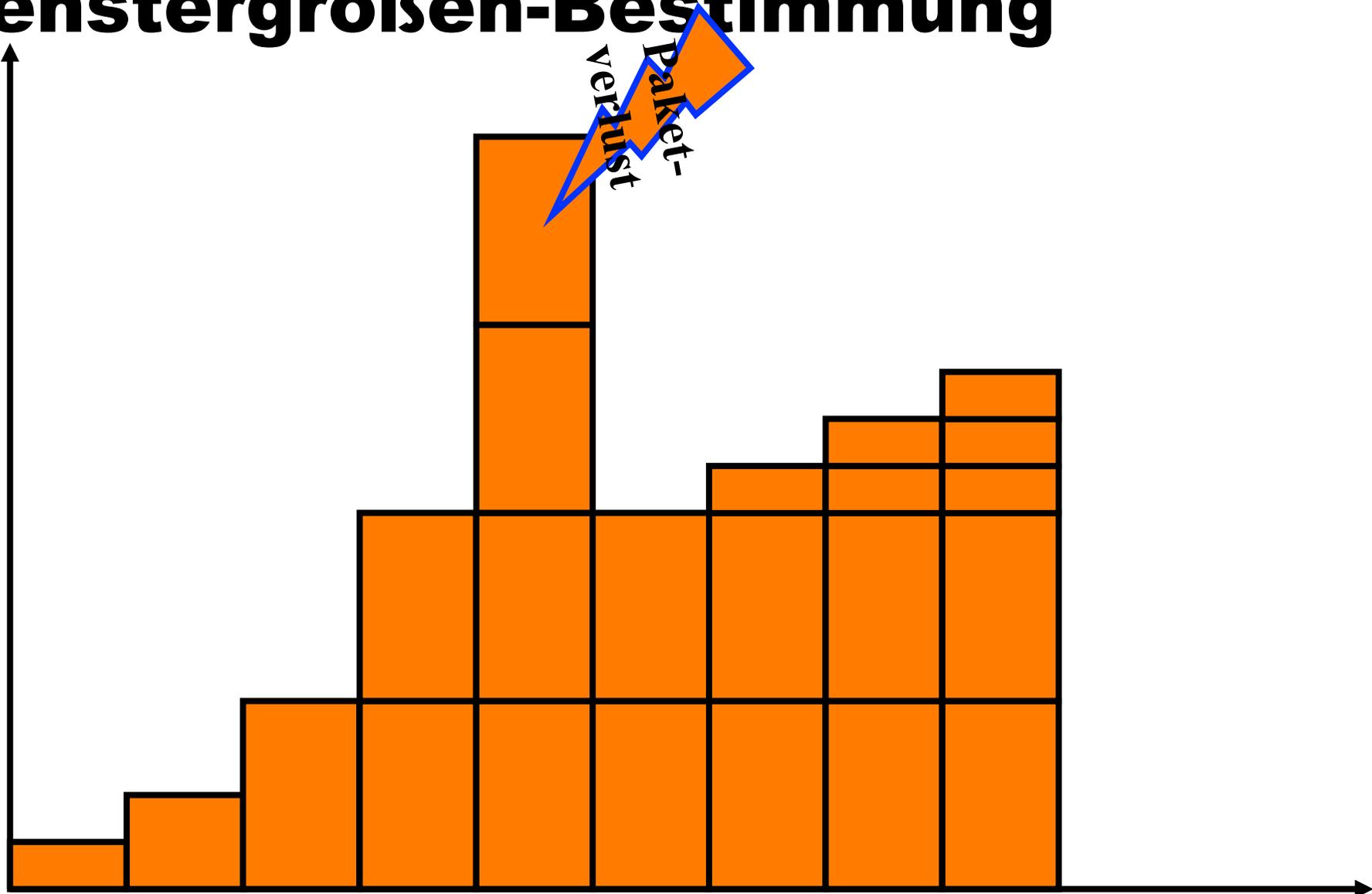
# Vorgehensweise bei der Fenstergrößen-Bestimmung



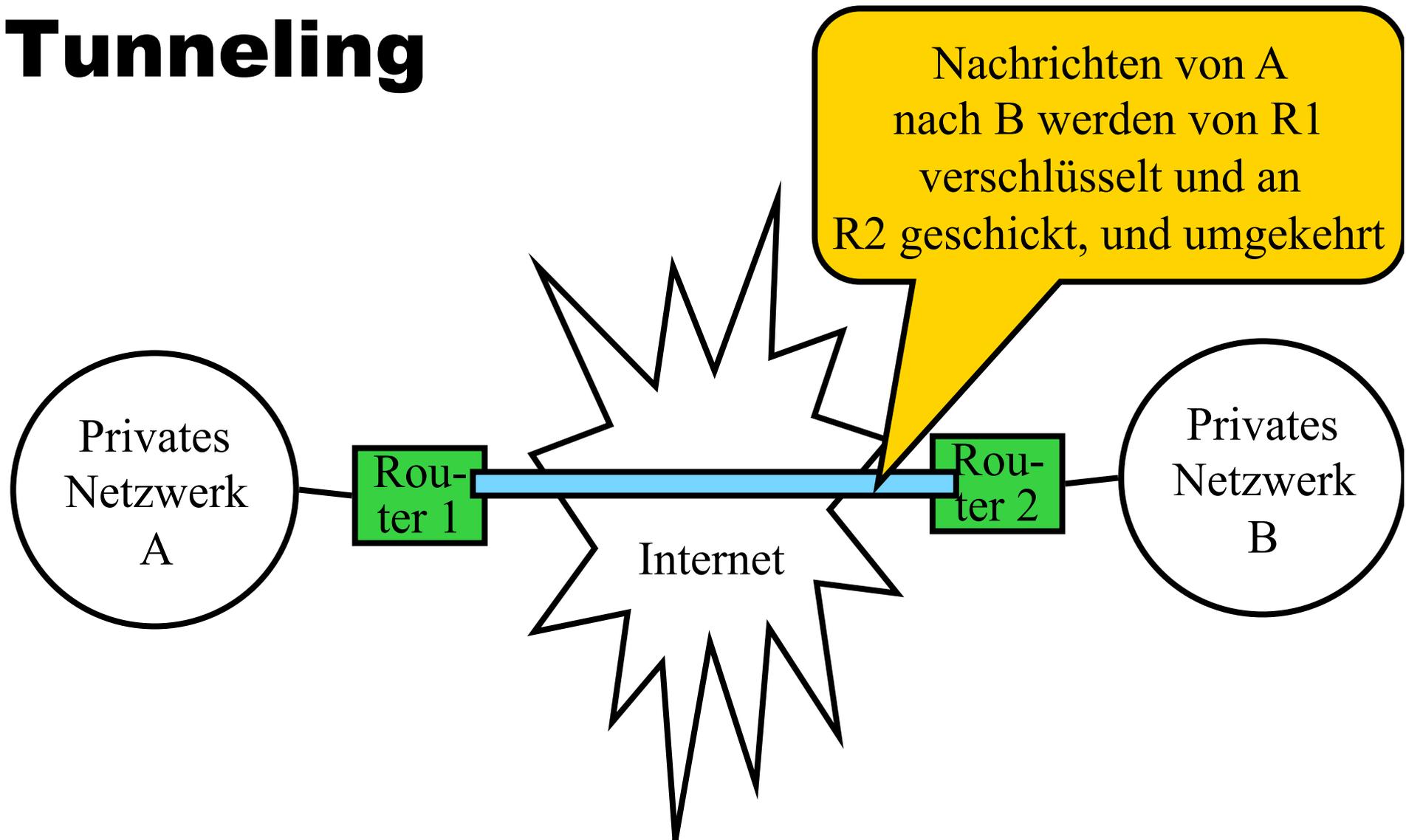
# Vorgehensweise bei der Fenstergrößen-Bestimmung



# Vorgehensweise bei der Fenstergrößen-Bestimmung



# Virtual Private Network/ Tunneling



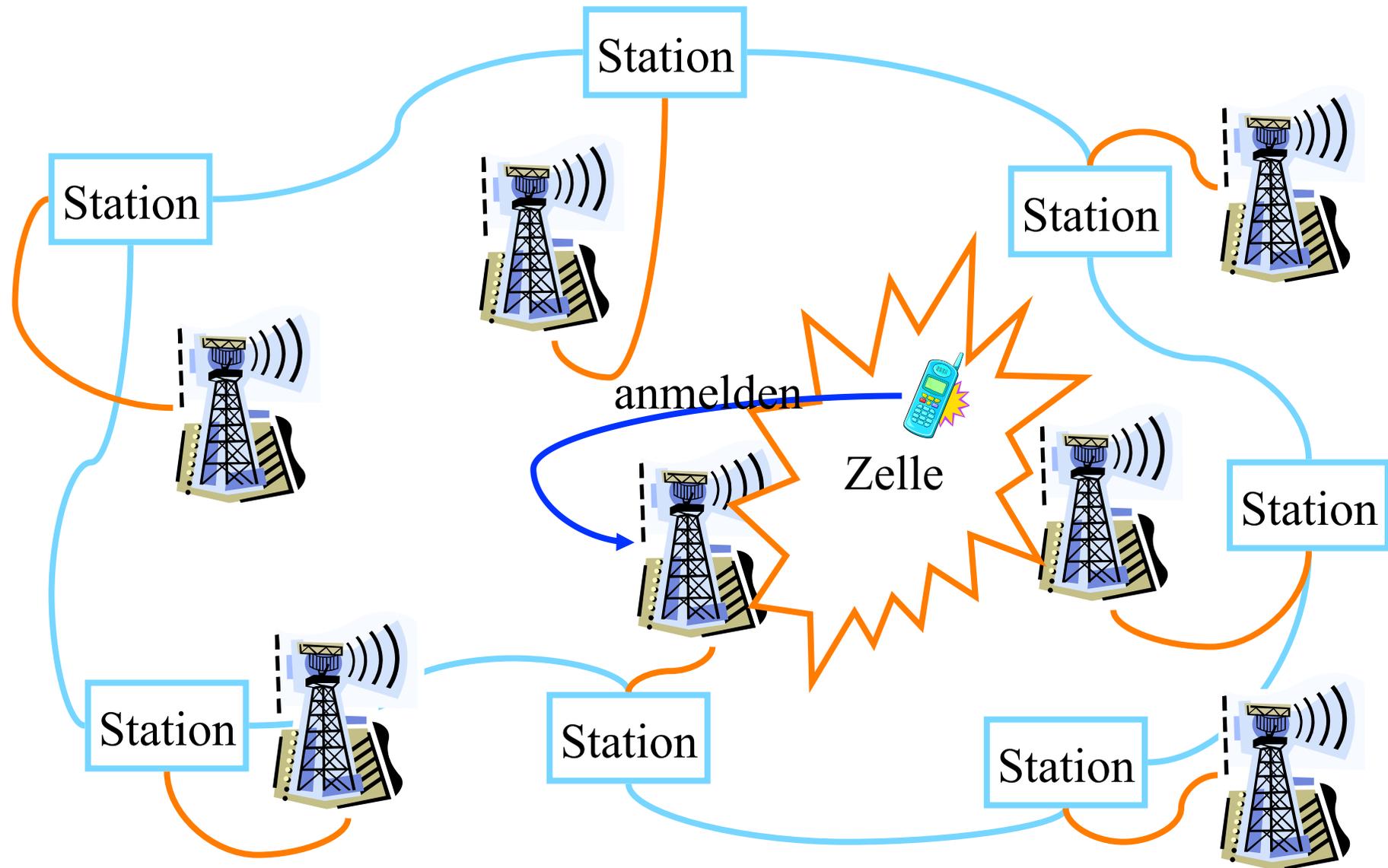
# Wireless Networks

# Kabellose Netzwerke

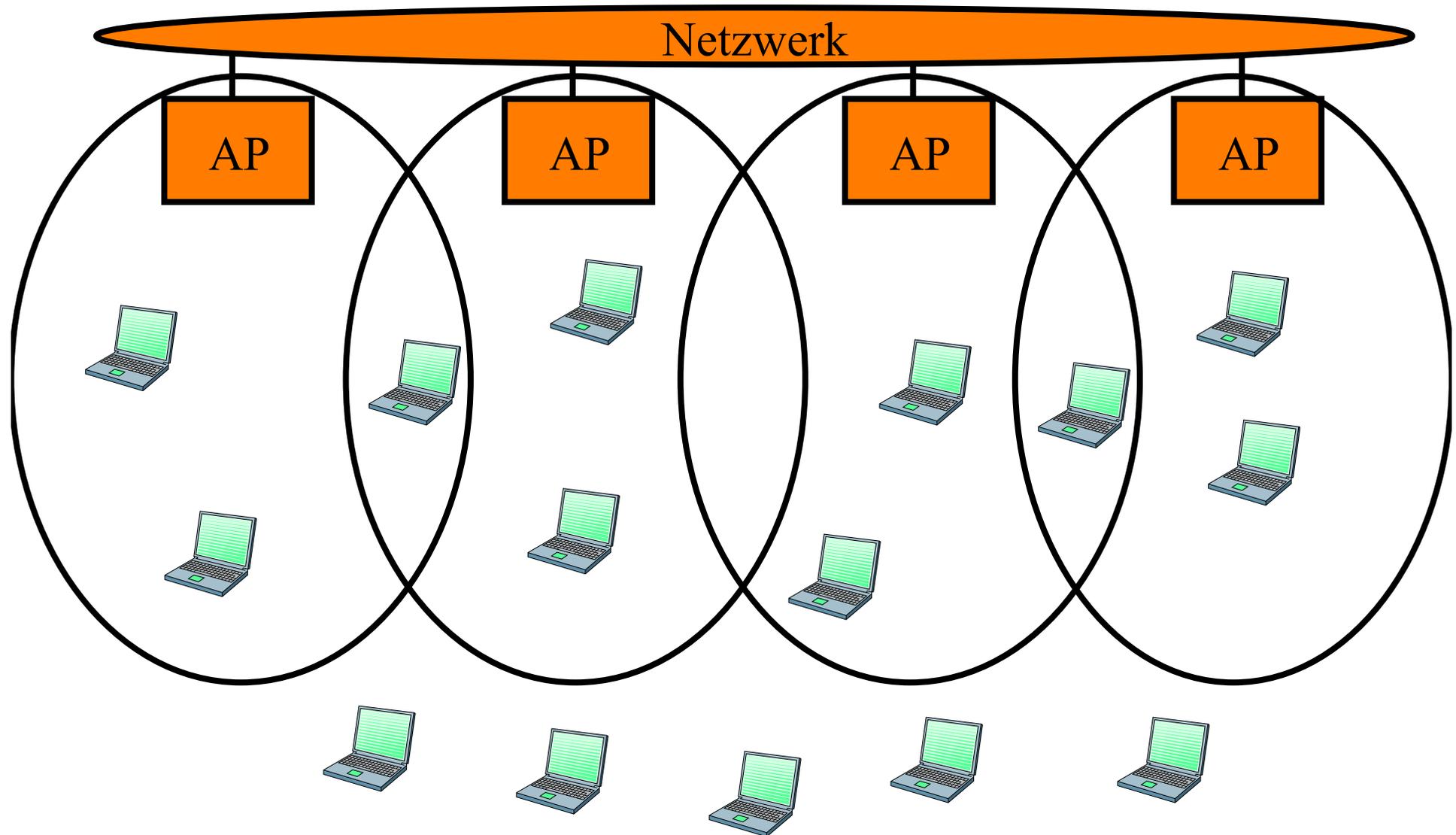
Station

# Wireless Networks

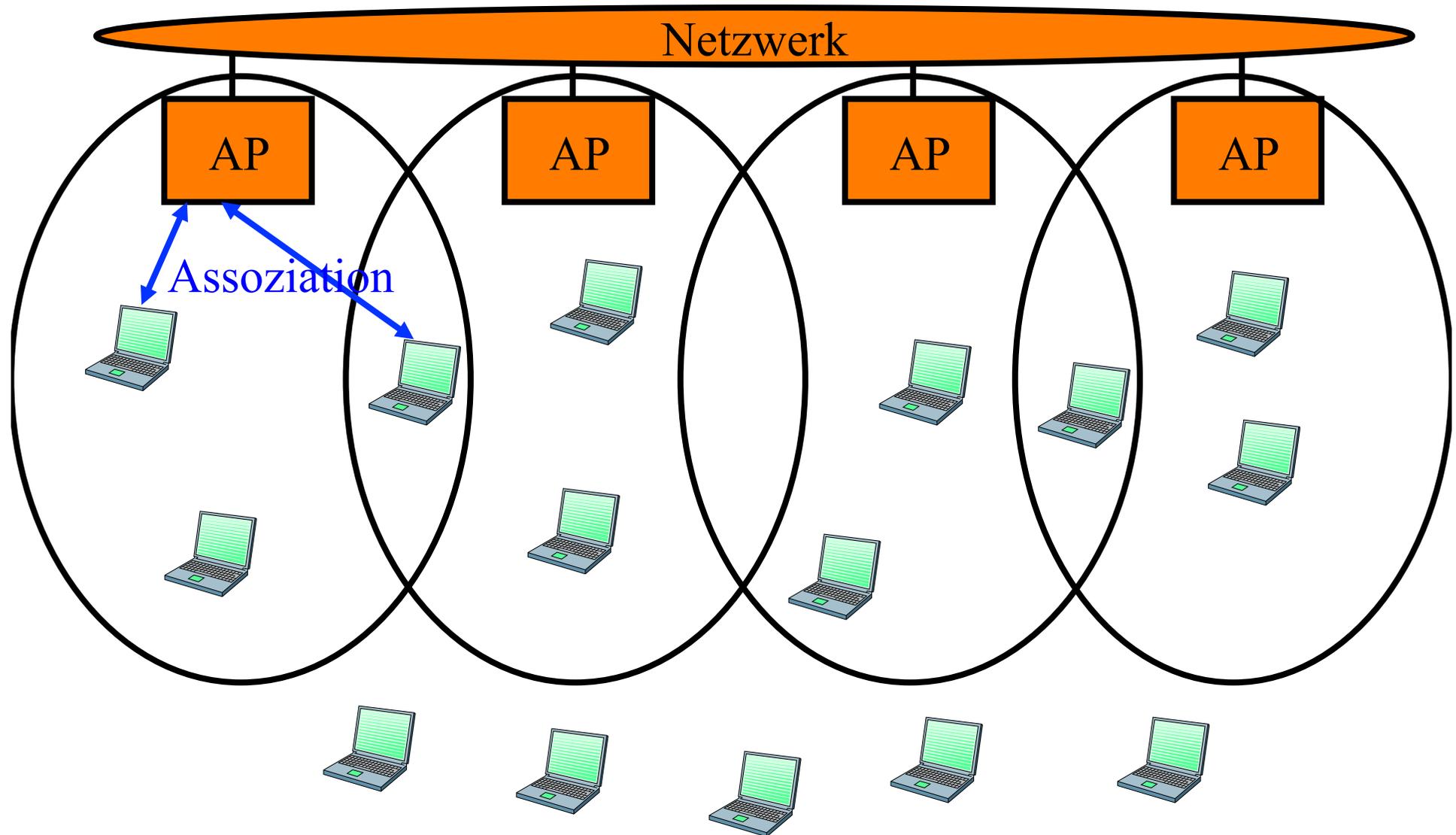
## Kabellose Netzwerke



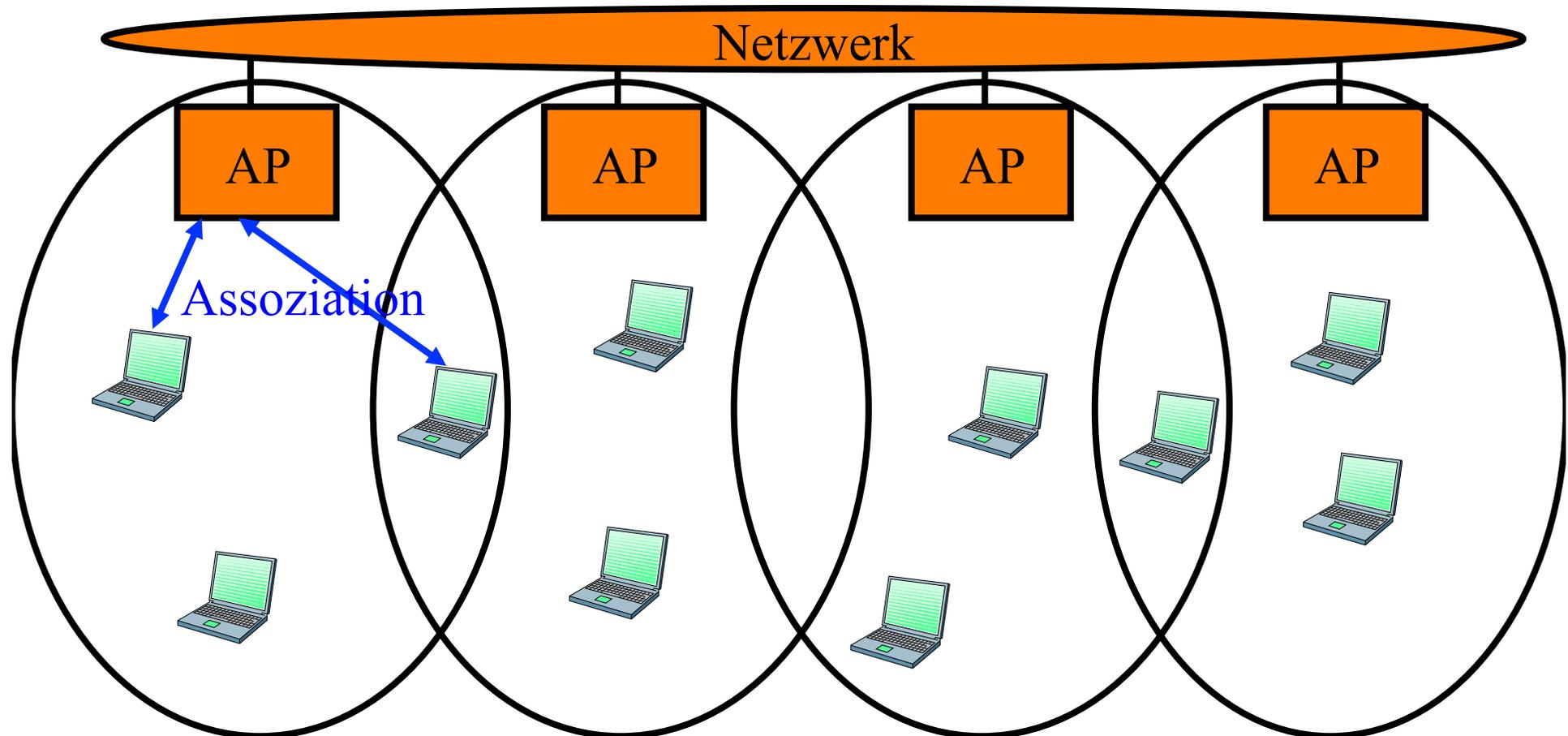
# Wireless LAN / IEEE 802.11



# Wireless LAN / IEEE 802.11



# Wireless LAN / IEEE 802.11



- IEEE 802.11: 1 – 2 Mbps IEEE 802.11b: -- 10 Mbps → 54 Mbps
- APs: Access Points
- Mobile Stationen assoziieren sich mit einem der erreichbaren APs (roaming)

# Wireless LAN / IEEE 802.11

- Samsung hat eine WLAN-Technologie für das 60-GHz-Band entwickelt, die Übertragungsraten von bis zu **4,6 GBit oder 575 MByte** pro Sekunde erlaubt. Die Technik bildet Samsung zufolge die Grundlage für den kommenden WLAN-Standard **802.11ad**
- Markteinführung: Ende 2015/Anfang 2016