Computernetzwerke

Sebastian Bauer

Wintersemester 2022/2023

Outline

Computer Engineering Curriculum

Zeiten

- VL- und Labor wann? Freitags zw. 9:45 und 13:00
- Einteilung nach Bedarf

Labor

Dieses Semester:

- Drei Labore
  - Via virtuellen Maschinen und Mininet
  - Programmieraufgaben
  - Abgabe via HTW-Cloud, GitLab und Moodle
- Jeder Studierende muss diese absolvieren
- Details und Abgabetermine folgen noch
- Bewertung: 0 bis 100%
- Prüfungsvoraussetzung
- Modulnotenanteil: 10%

Vorträge

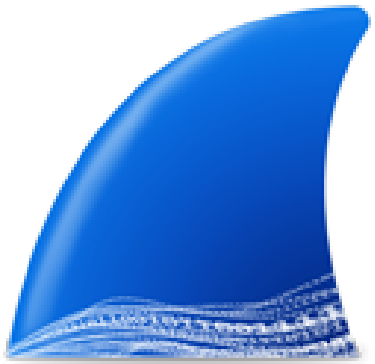
- Teilnehmende wählen ein Thema aus
- Dauer des Vortags: ca. 12 Minuten
- Ab Ende November
- Zusammenfassung und fünf vortragsspezifische Moodle-Quiz-Fragen
- Mehr auf https://gitlab.rz.htw-berlin.de/bauers/ce26-cnw-wise-22/-/blob/main/talks.md
- Prüfungsvoraussetzung
- Modulnotenanteil: 15%

Abschlussklausur

- Termin im 1. PZR: Freitag, 26. Januar 2023
- Termin im 2. PZR: TDB ⇒ lt. LSF
- Voraussetzung: Bestehen der Laborversuche, Vortrag
- Scope: Vorlesung, (Labor-)Übung, Vorträge
- Vermutlich E-Klausur (im Labor oder zu Hause)

Ziele der Veranstaltung

- Grundlegende Begriffe von Computernetzwerke verstehen und anwenden
- Netzwerkschichten
- Netzwerkprotokolle
- Netzwerkprogrammierung
- Kennenlernen von Sniffern wie Wireshark



Voraussetzungen

- Linux-Kenntnisse
- Zahlensysteme (siehe Digitaltechnik und Rechnerorganisation)

Materialien (Slides, Aufgaben, etc.)

URL des GitLab-Projekts

https://gitlab.htw-berlin.de/bauers/ce26-cnw-wise-22

Voraussetzung: Mitgliedschaft im Projekt, ggf. einmalige Registrierung auf https://gitlab.htw-berlin.de nötig → erste Moodle-Aufgabe zum Mitteilen dieses Schrittes

Repo clonen

```
# via Git (Public-Key voher hochladen)
$ git clone git@gitlab.rz.htw-berlin.de:bauers/ce26-cnw-wise-22.git

# Alternative
$ git clone https://gitlab.htw-berlin.de/bauers/ce26-cnw-wise-22.git

# Synchronisieren des lokalen Repos
$ cd ce26-cnw-wise-22
$ git pull origin main
```

Wo finde ich was im Repo?

Labor	`examples/*`, `exercises/practical-ex*.pdf`
Vorlesung	`slides/*.pdf`
Übung	`exercises/ex*.pdf`

Kontakt

- Direkt in der Vorlesung oder im Labor
- Via E-Mail: Sebastian.Bauer@htw-berlin.de
- Raum G 528 oder C 309
- Wenn man mich sieht

Fragen, Kritik, Verbesserungsvorschläge (*nobody is perfect*)
sind jederzeit willkommen!

Inhalt der Vorlesung

- Einführung
- Grundlagen
- Kodierungen
- Physikalische Schnittstellen
- Netzwerkprotokolle
- TCP/IP
- Sicherheit? (Kryptographie, etc.)

Änderungen vorbehalten

Literatur und Ressourcen

- **Tanenbaum, A. S.; Wetherall, D.J.**:
  Computernetzwerke, 2012

- **Baun, C.**, *Computernetze kompakt*, 2018

- https://www.imedita.com/blog/top-10-list-of-network-simulation-tools/

Outline

Computernetzwerke

- Zusammenschluss elektronischer Systeme (Computer, Sensoren, Aktoren) der Kommunikation dieser erlaubt
- Ziel: Ressourcen gemeinsam nutzen

Allgemeiner: Verteiltes System (nach Klaus-Peter Löhr)

Ein *verteiltes System* ist eine Menge interagierender Prozesse (oder Prozessoren), die über keinen gemeinsamen Speicher verfügen und daher über Nachrichten miteinander kommunizieren

Anwendungen von Computernetzwerken

- Business
  - Teilen von Informationen und Ressourcen
  - Web-Applikationen, E-Commerce, ...
- Home
  - E-Mail, Web, Messaging, Social Networks, Wiki, ...
- Mobile
  - GPS, Bezahlung, ...

Klassifikation

Wir unterscheiden Netzwerke anhand:

- Wie Aufgaben verteilt werden,
- der Anzahl der Empfänger einer Nachricht sowie
- der Entfernung der Partner.

Aufgabenverteilung: Client-Server

Client-Rechner verbinden sich mit leistungsfähigen *Server*:

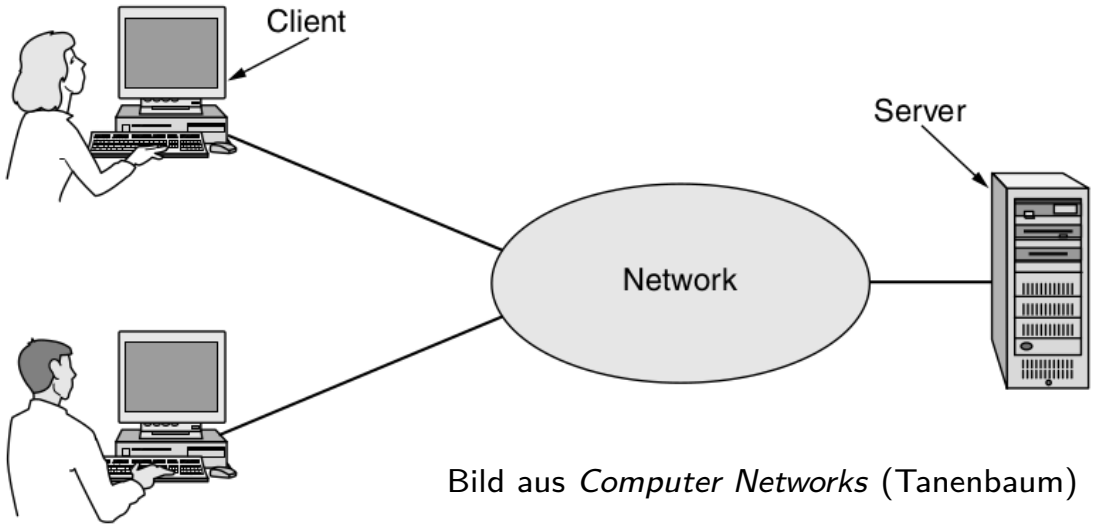
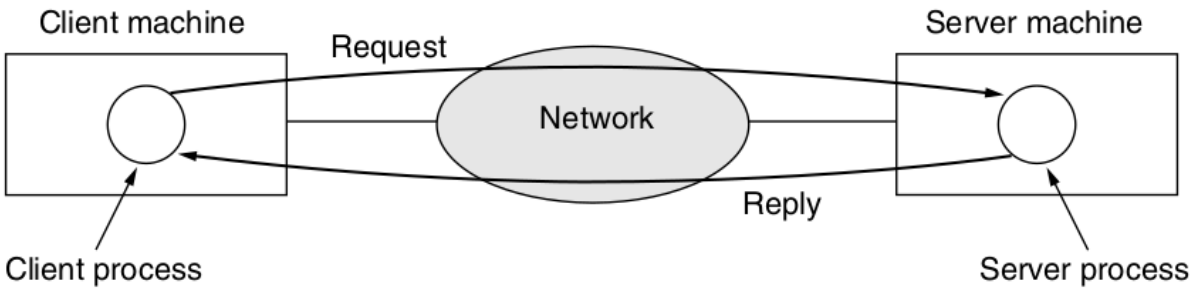


Bild aus *Computer Networks* (Tanenbaum)

Senden Anfragen, die vom Server beantwortet werden:



Aufgabenverteilung: Peer-to-Peer

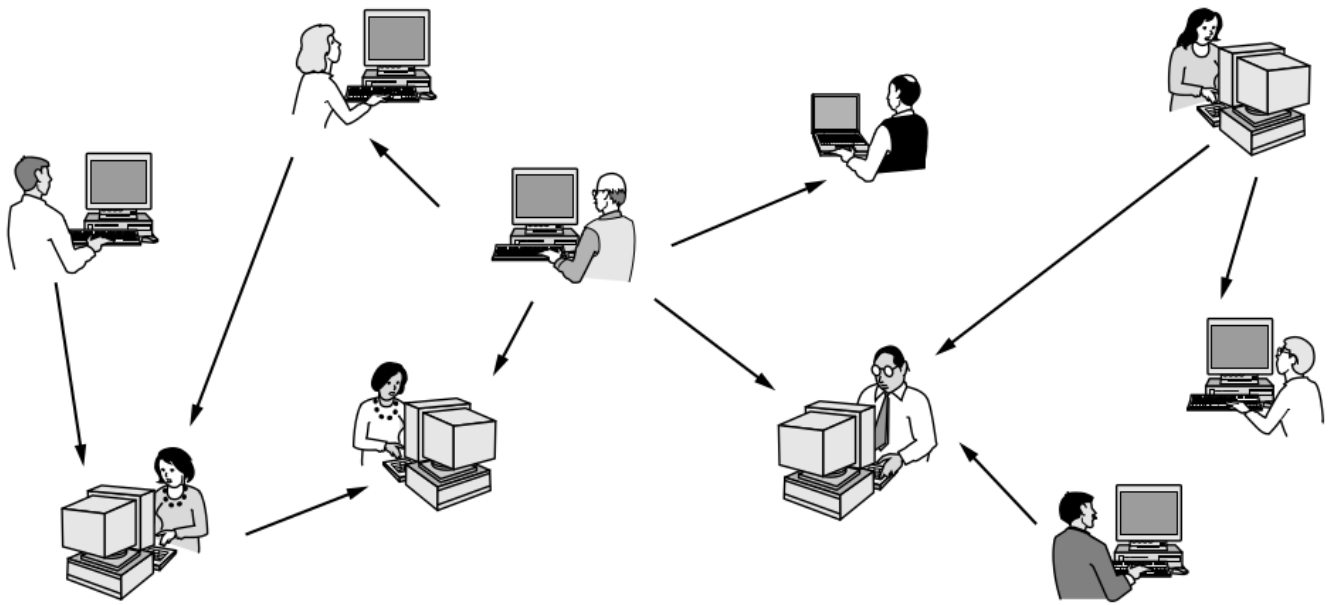
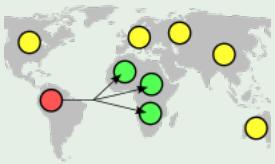
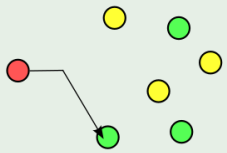
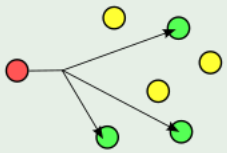
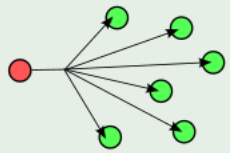
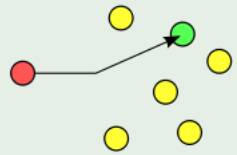


Bild aus *Computer Networks* (Tanenbaum)

Anzahl der Empfänger (Routingschematas)

unicast	zu einem einzelnen Zielrechner
broadcast	zu allen Rechnern im Netzwerk
multicast	zu einer Gruppe von Rechnern im Netzwerk
anycast	zu irgend einem Zielrechner unter einer Menge
geocast	zu einer Gruppe von Rechner basierend auf geographischen Standort

Welche Bilder passen zu welchem Schema?

a)  b)  c)  d)  e) 

Klassifikation anhand der Distanz

Distanz	Partnerknoten	Beispiel
$< 4\,\text{cm}$	Nähe	Near-field Communications (NFC)
$< 1\,\text{m}$	Körper	Body Area Network (BAN)
$1\,\text{m}$	Quatratmeter	Personal Area Network (PAN)
$10\,\text{m}$	Raum	Local Area Network (LAN)
$100\,\text{m}$	Gebäude	
$1\,\text{km}$	Campus	
$10\,\text{km}$	Stadt	Metropolitan Area Network (MAN)
$100\,\text{km}$	Land	Wide Area Network (WAN)
$1000\,\text{km}$	Kontinent	
$10\,000\,\text{km}$	Planet	Internet

Nahfeld (Nearfield)

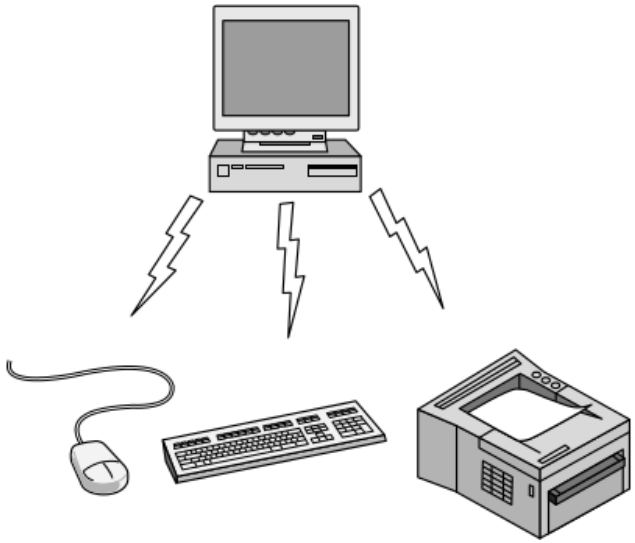
- Drahtlose Kommunikation mit Distanzen bis zu 4cm
- Datenübertragungsrate von derzeit maximal 424 kBit/s
- Anwendungen: Kontaktloses Zahlen, Authentisierung

Body Area Network (BAN)

- Drahtloses Netzwerk tragbarer Geräte am Körper
- Zur Weiterleitung von Daten am Körper angebrachten Sensoren für Vitalwerte
  - Blutdruck, Blutzucker, EKG, Puls, ...

Bild von Wikipedia

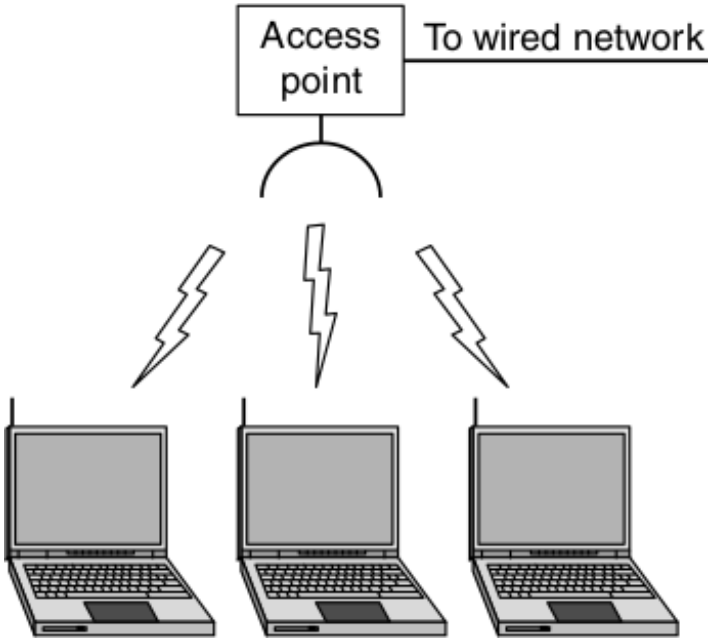
Personal Area Network (PAN)

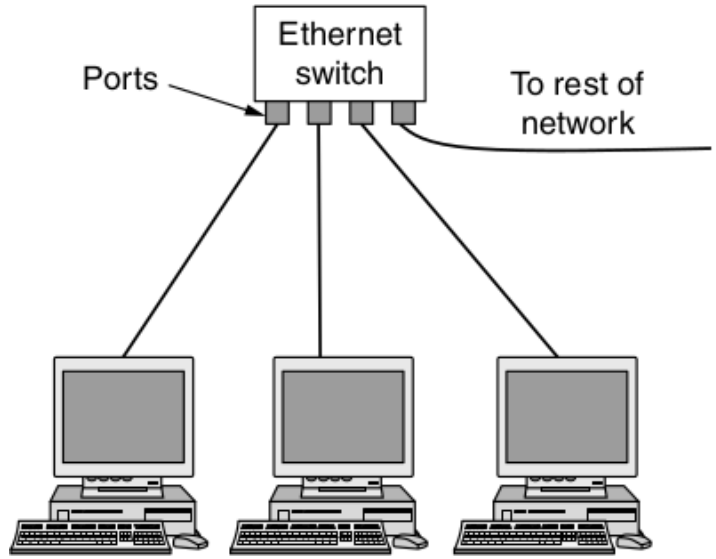


- Alles was am Rechner direkt angeschlossen wird
  - Eingabegeräte
  - Drucker, Monitor
- Drahtlos: IrDA, Bluetooth

Local Area Network (LAN)

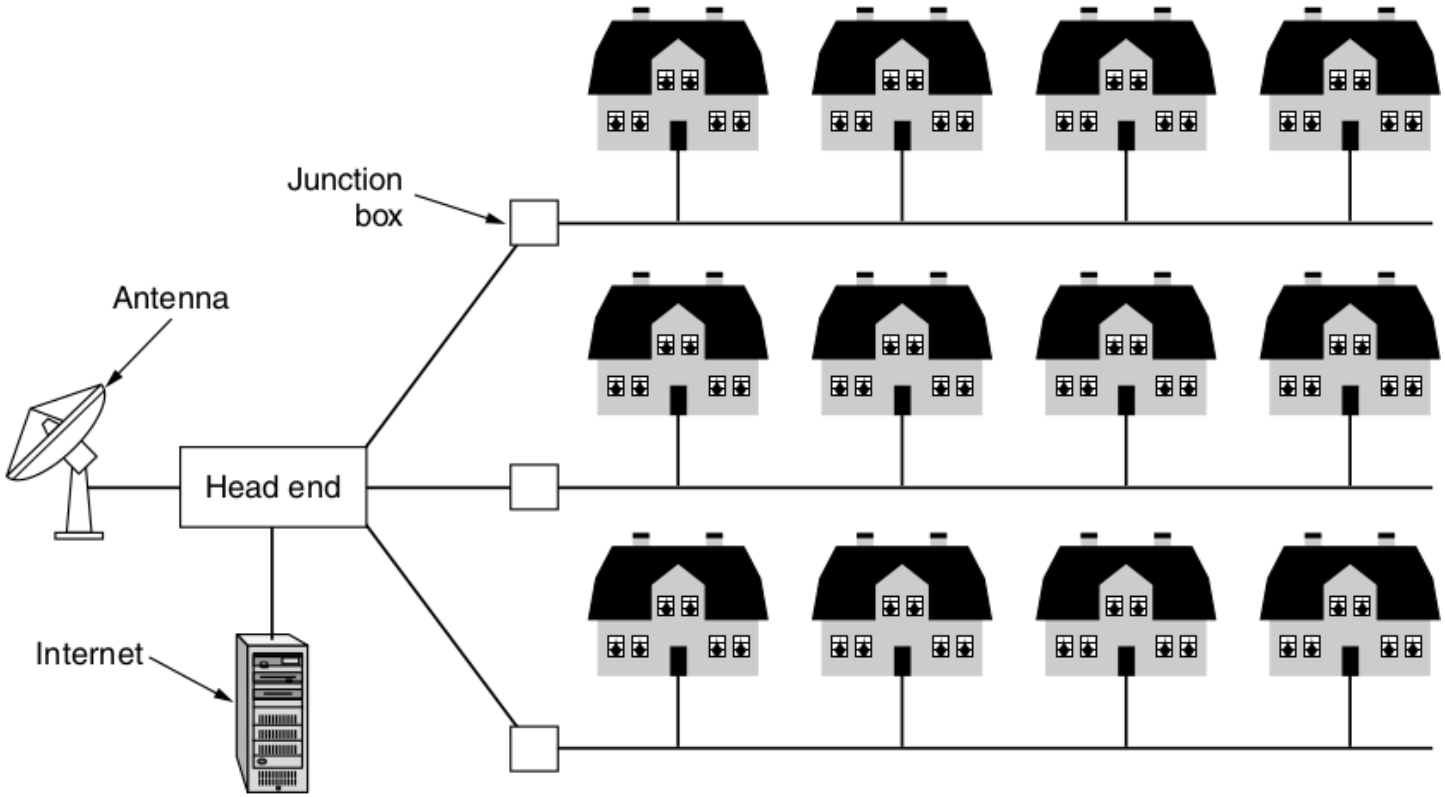
- Für Rechnerverbindung im Raum oder Gebäude
- Drathlos: Wireless LAN (WLAN)



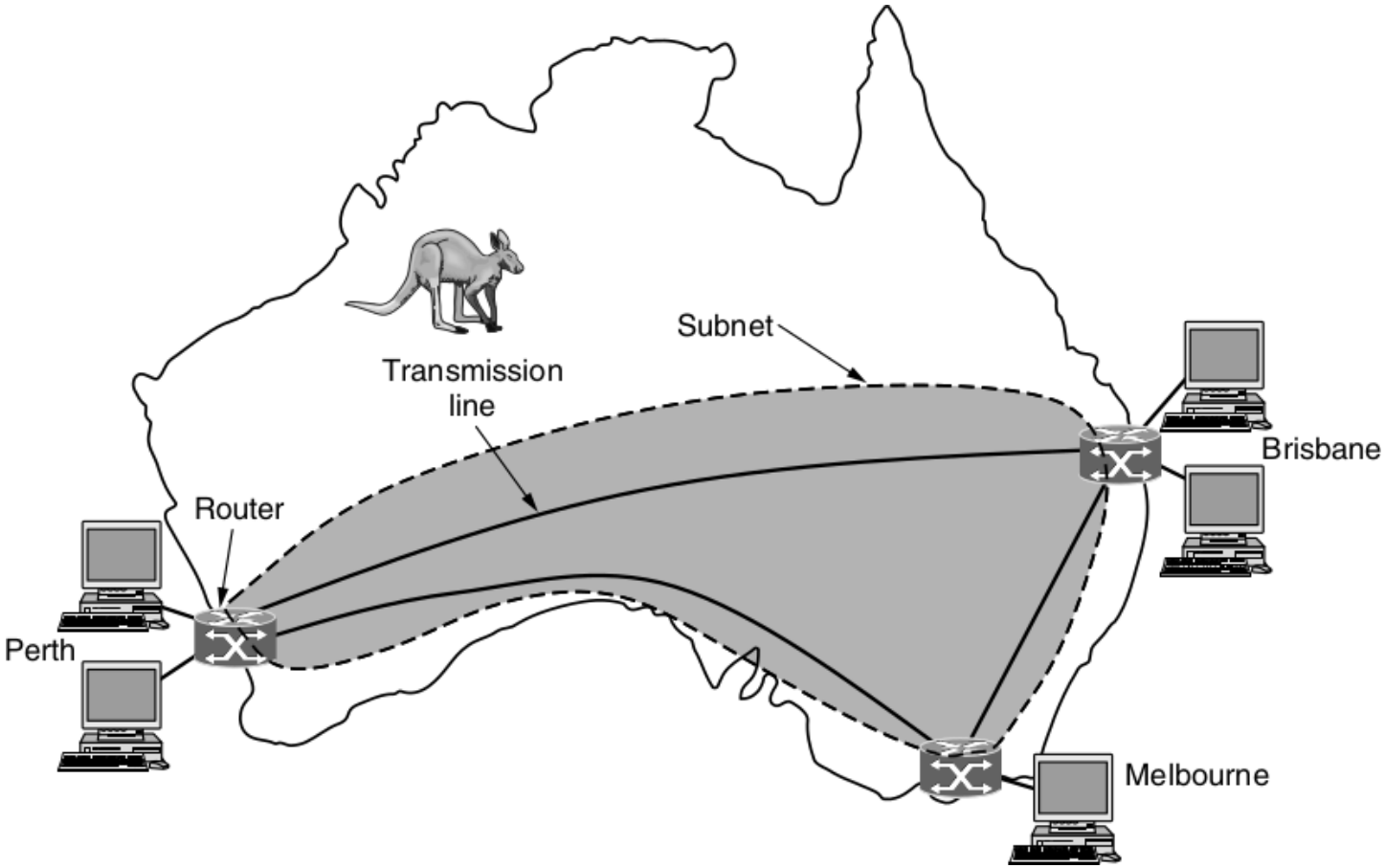


Metropolitan Area Network (MAN)

- Verbindet Geräte innerhalb einer Stadt, z.B. Kabelanschluss
- Drahtlos: WMAN bzw. WiMAX



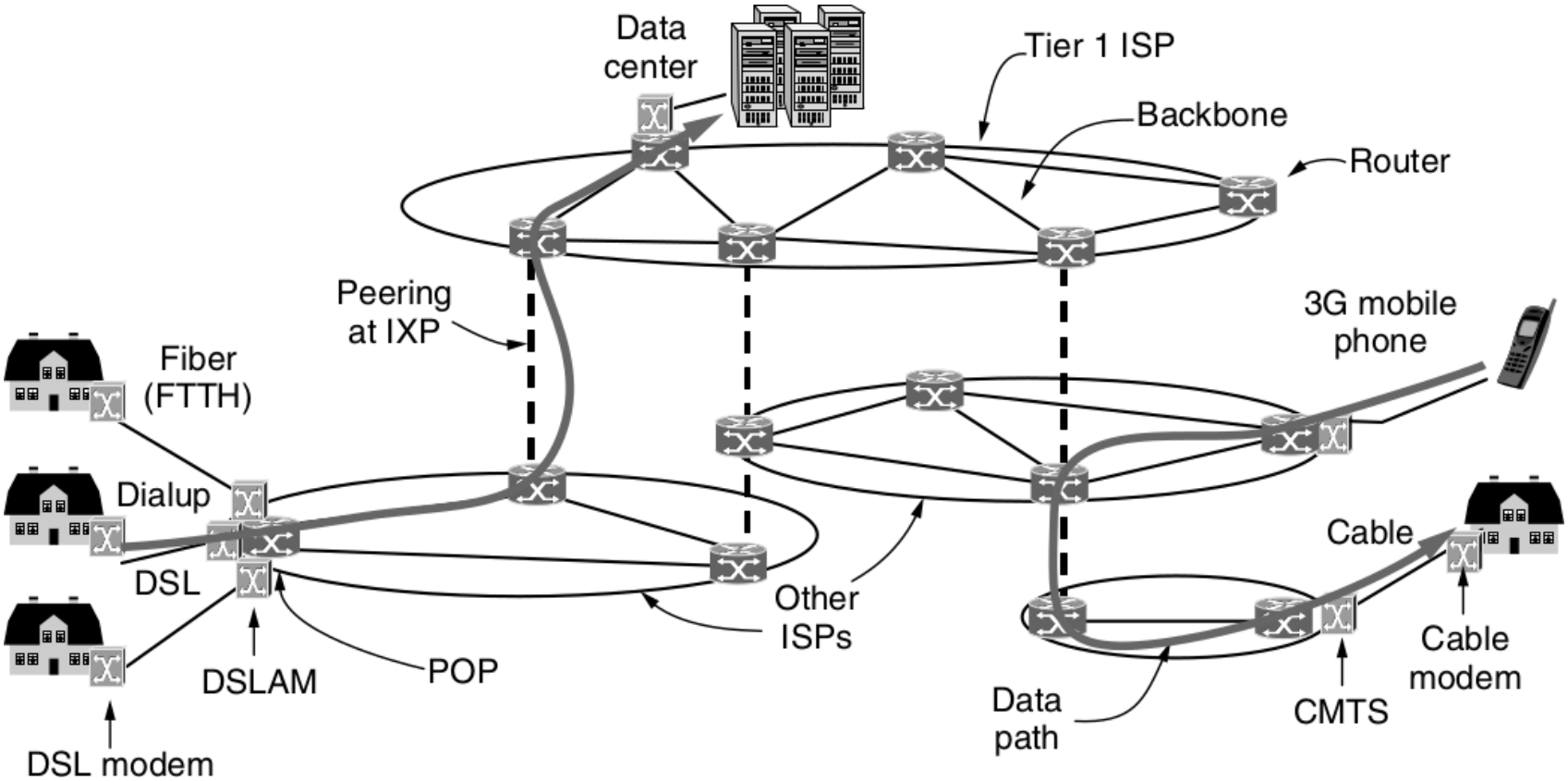
Wide Area Network (WAN)



Internet

- Sammlung von möglicherweise zueinander inkompatiblen Netzwerken
- Das Internet: Verbindet mit Hilfe von *Internetprovidern*
  - Unternehmensnetzwerke (Intranet)
  - Heimnetzwerke
  - Forschungsnetzwerke

Übersicht über die Internetarchitektur



Was passiert, wenn?

Netzwerke sind häufig unzuverlässig ⇒ Klare Regeln für verschiedene Situation während der Kommunikation:

- Wer sendet und an wen?
- Wann fängt eine Nachricht an, wann hört Sie auf?
- Wie erkenne ich fehlerhafte Nachrichten, verlorengegangene Nachrichten?
- Was tue ich überhaupt bei verlorenen Nachrichten?
- Wie kann ich Nachrichten anfordern oder abbestellen?
- Wie werden unterschiedliche Kanäle nachgebildet (Multiplexing)?

Absprachen sind nötig ⇒ Protokolle.

Protokolle

Ein *Protokoll* definiert das Format gültiger Nachrichten (**Syntax**), die genaue Abfolge der Nachrichten (**Grammatik**) sowie das Vokabular gültiger Nachrichten und deren Bedeutung (**Semantik**)

Ein Protokoll für alles wäre zu komplex oder zu einfach (= zu unflexibel) $\Rightarrow$ Schichten

Beispiele

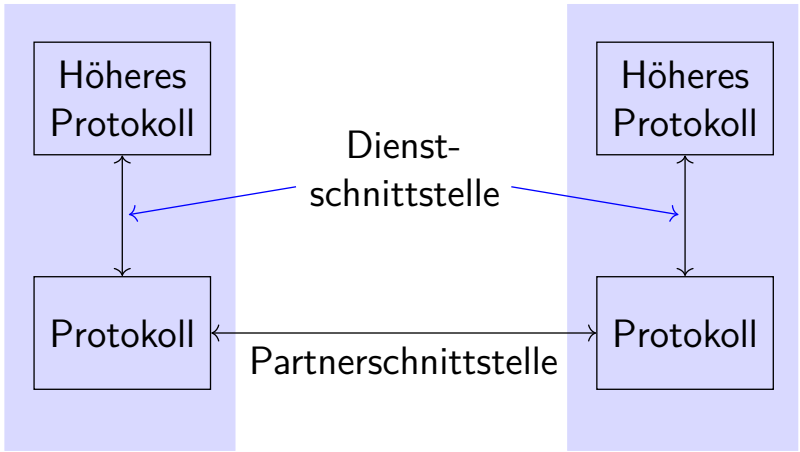
IMAP4 Übertragen von E-Mails zwischen Server und Nutzercomputer (siehe RFC 3501)

IP Weiterleitung von Daten von einem Rechner zu einem anderen (Routing, siehe RFC 791)

Protokollschichten

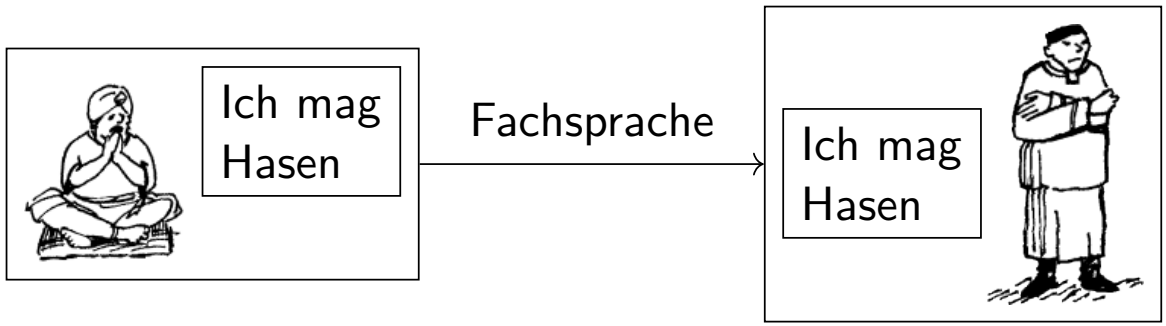
Eine Protokollschicht (engl. *protocol layer*)

- behandelt einen bestimmten Aspekt der Kommunikation,
- kapselt die Daten,
- bietet eine Dienstschnittstelle zur darüber liegenden Schicht (engl. *service interface*) und
- eine Partnerschnittstelle zum Gegenstück auf dem anderen Rechner (engl. *peer-to-peer interface*).



Schnittstelle ≡ eine Menge von Operationen, die zusammen einen Dienst definieren

Kommunikation zwischen zwei Philosophen

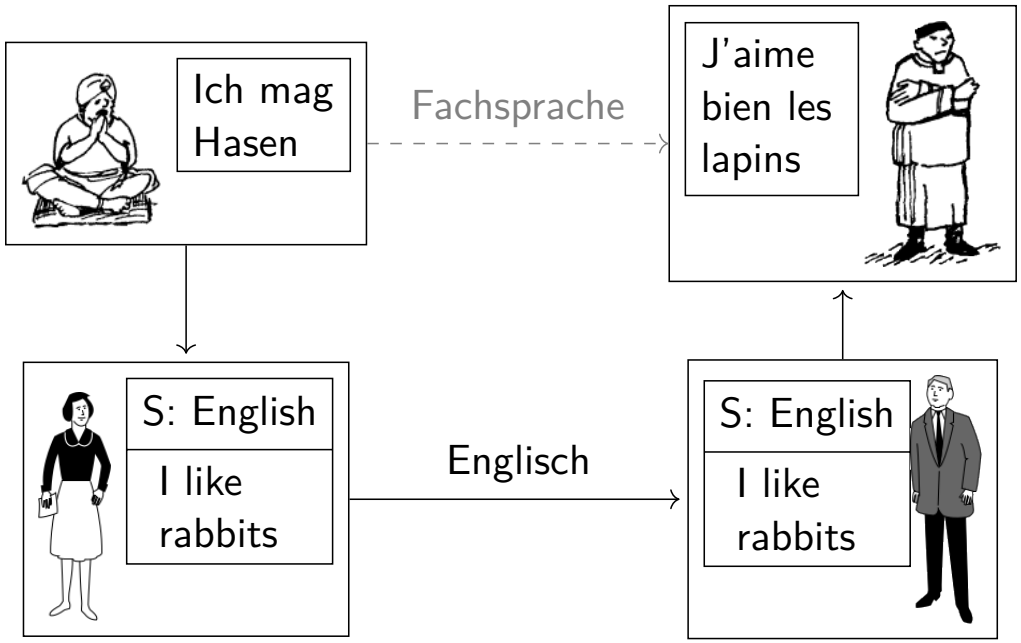


Philosophen verstehen sich, wenn sie in der Nähe sind und dieselbe Sprache sprechen.

Was ist, wenn sie nicht dieselbe Sprache sprechen?

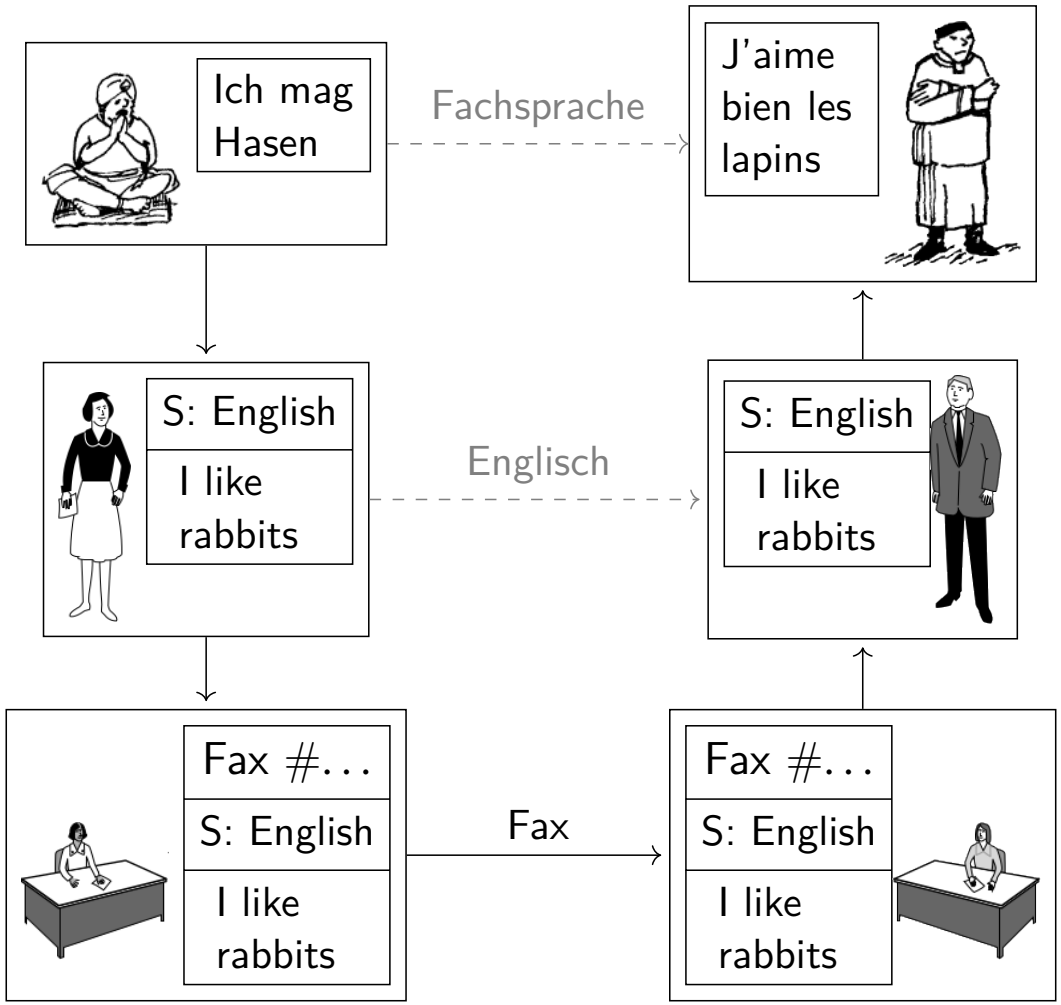
Kommunikation mit zwei Übersetzern

- Philosophen: deutsch oder französisch
- Übersetzer: deutsch/englisch oder englisch/französisch

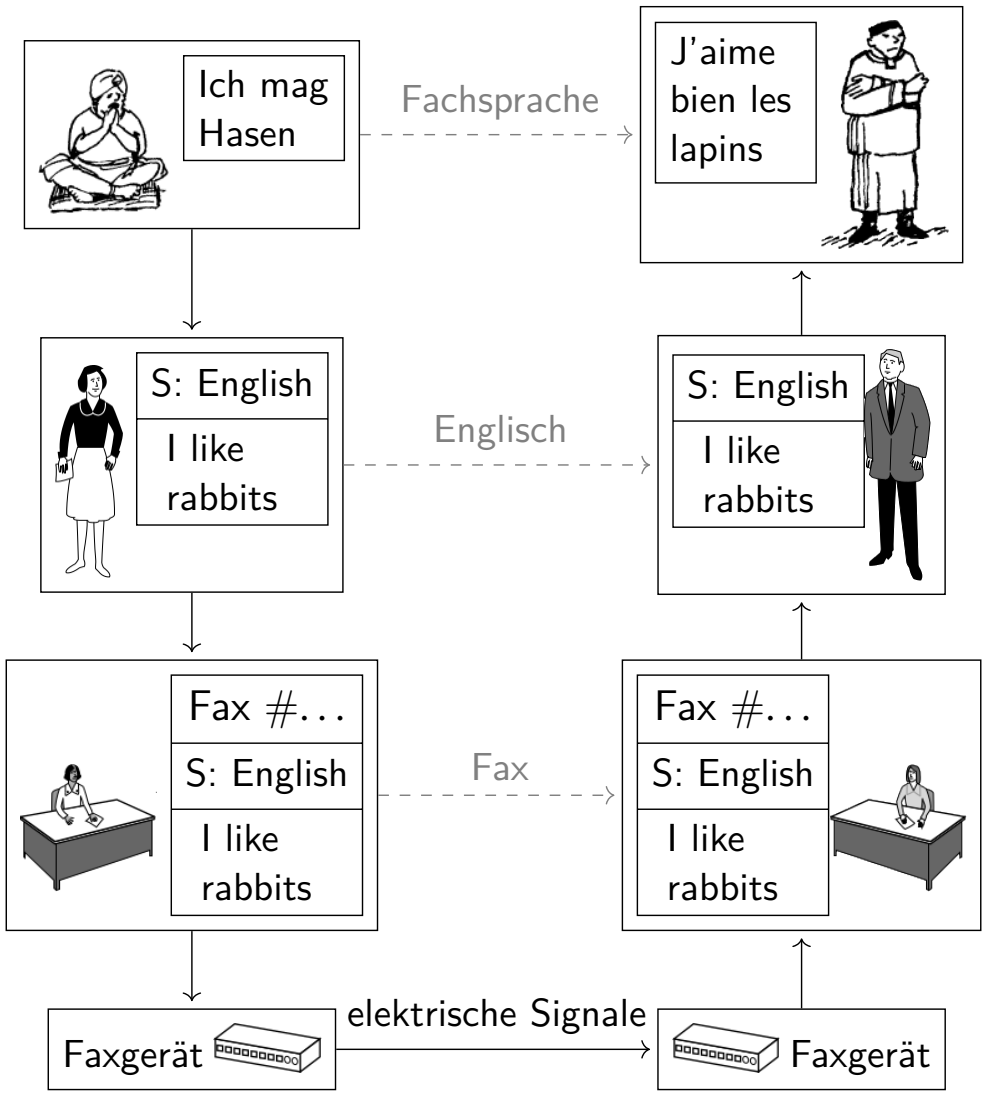


Was ist, wenn sie räumlich getrennt sind?

Kommunikation mit zwei Nachrichtenübermittler



Kommunikation mit zwei Faxgeräten



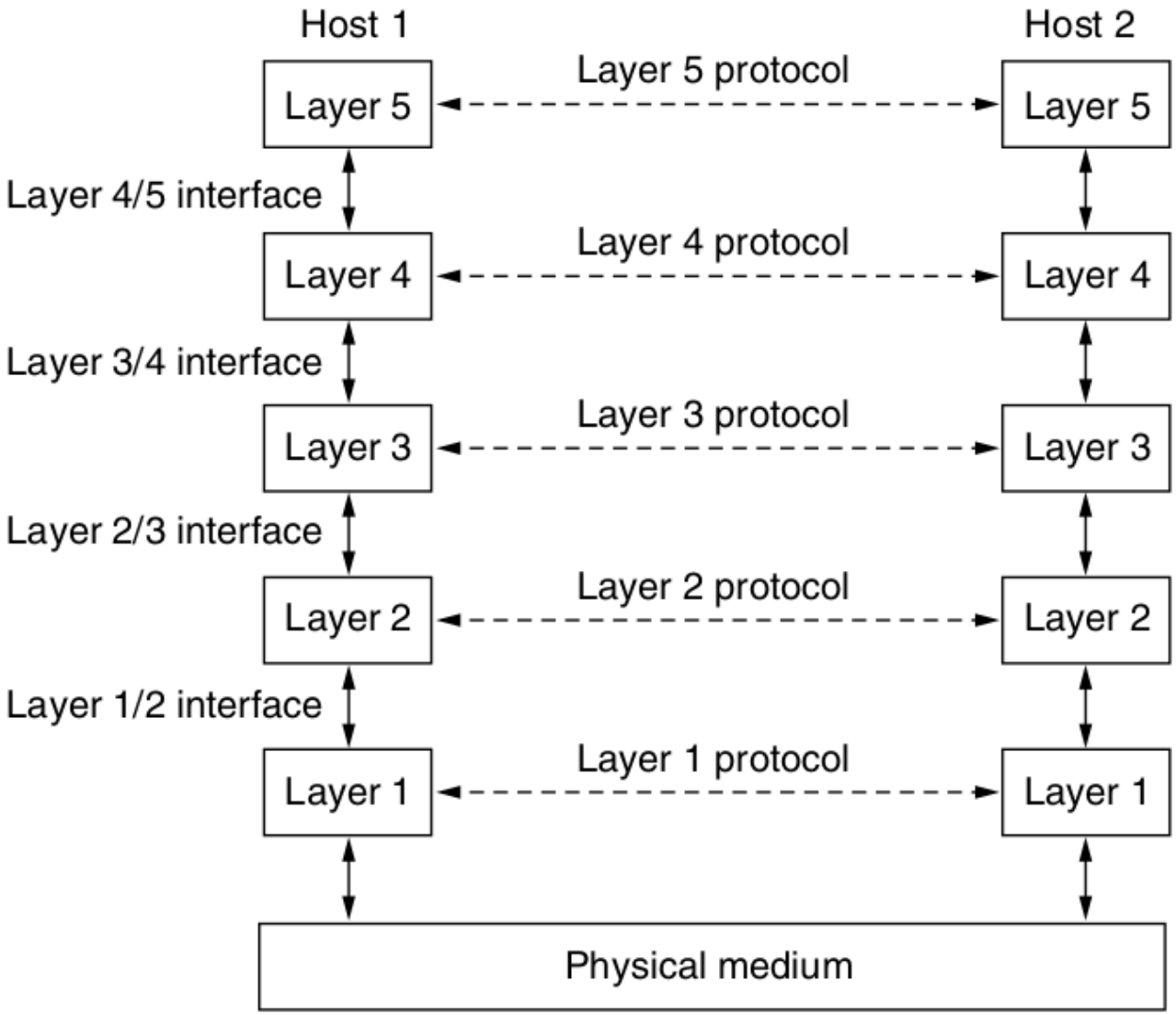
Ziel der Schichtung

Jede Schicht

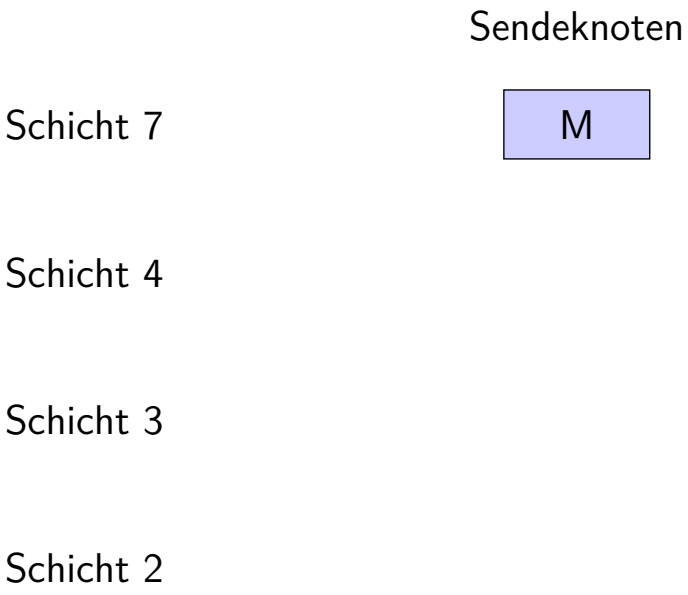
- definiert eine Abstraktionsebene und
- bietet eine definierte Schnittstelle

mit dem Ziel, dass die Implementierung der Schicht *austauschbar* ist.

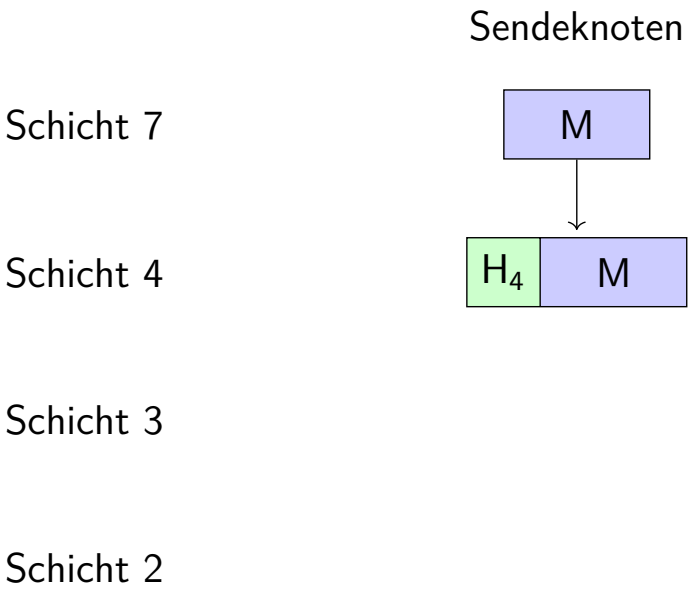
Allgemeines Schichtenmodell



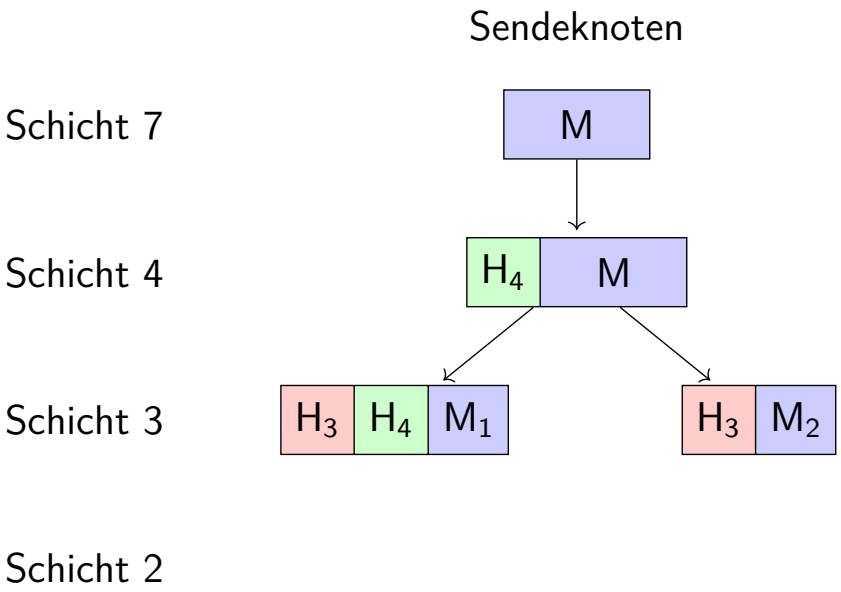
Informationsfluss im Schichtenmodell



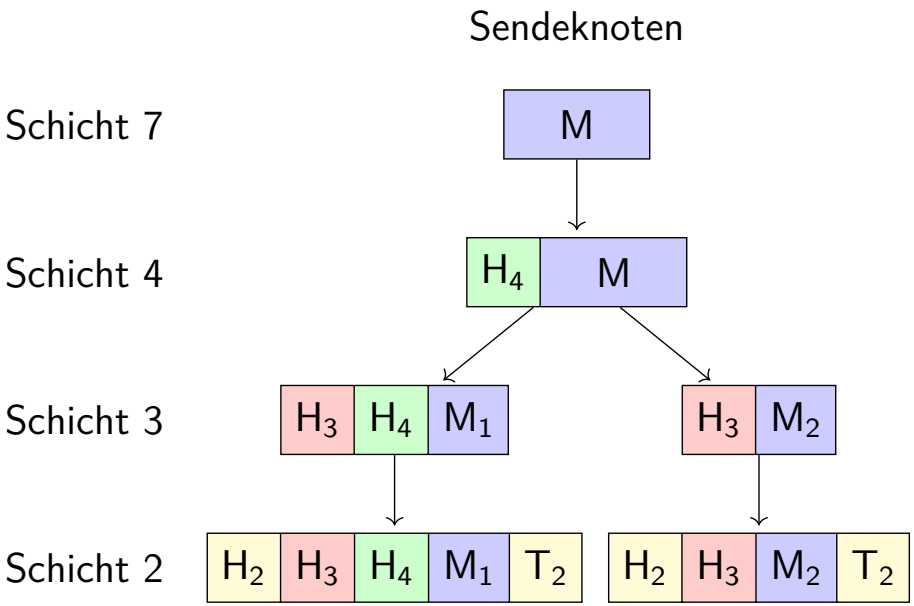
Informationsfluss im Schichtenmodell



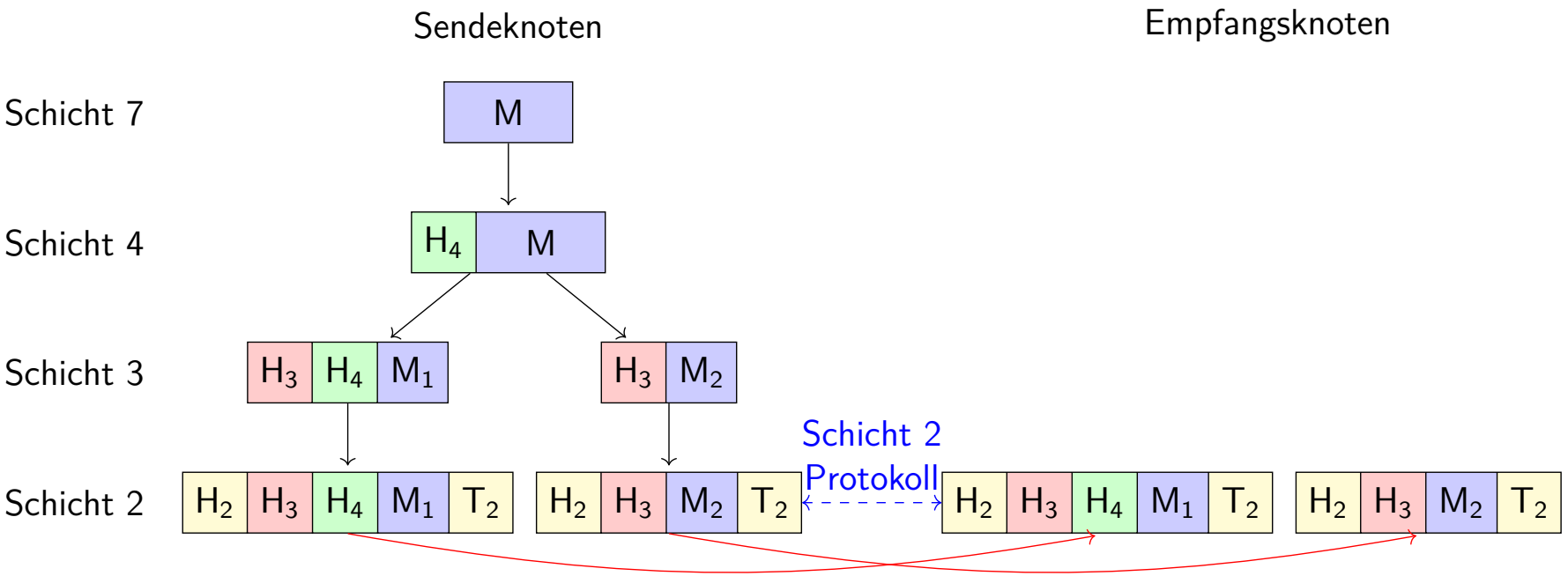
Informationsfluss im Schichtenmodell



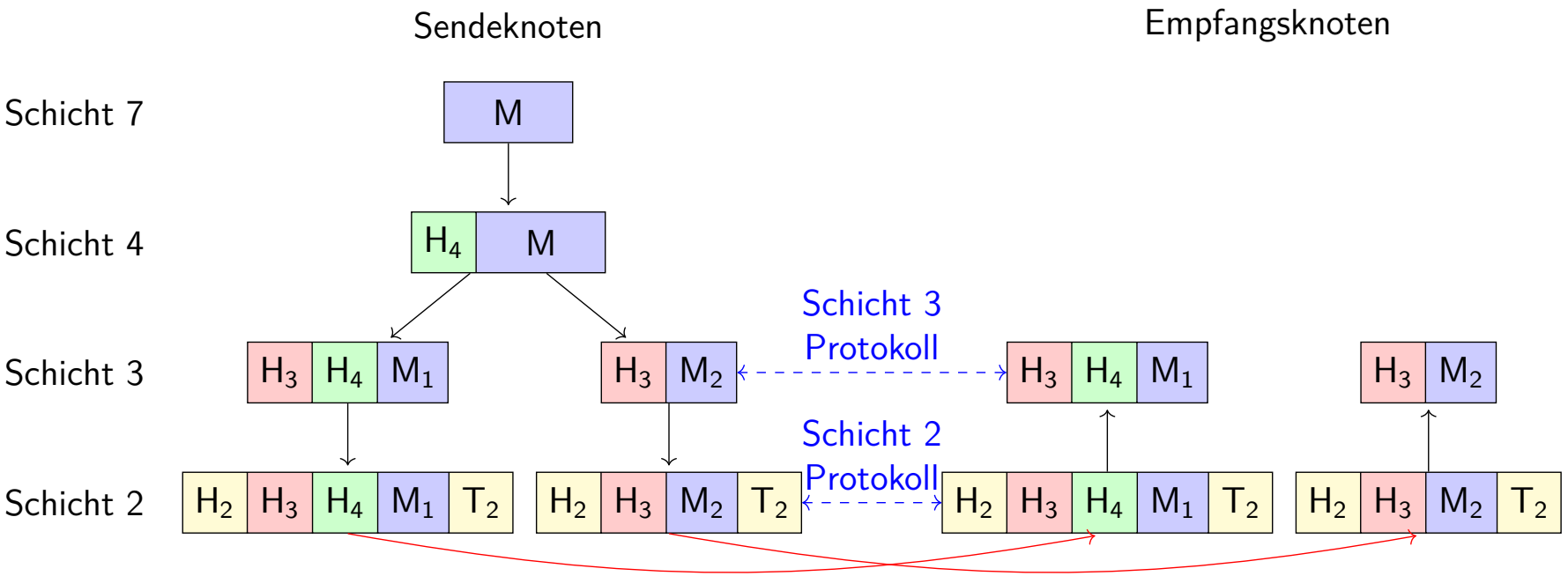
Informationsfluss im Schichtenmodell



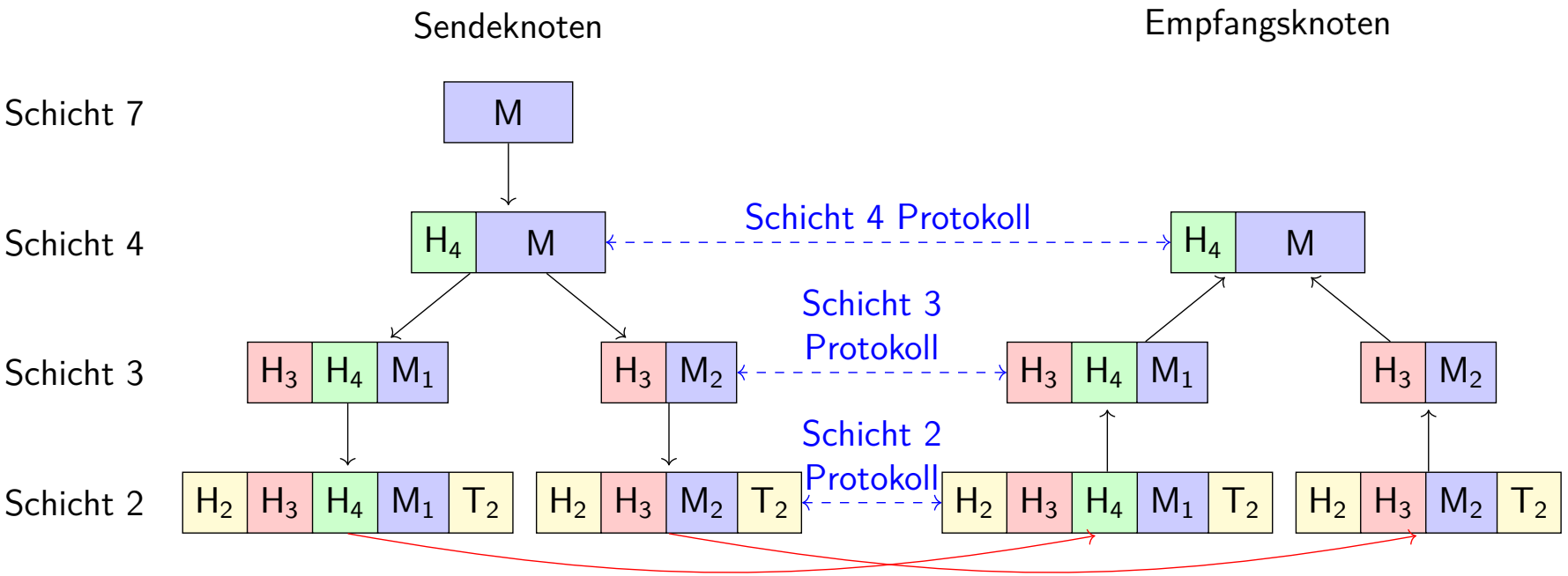
Informationsfluss im Schichtenmodell



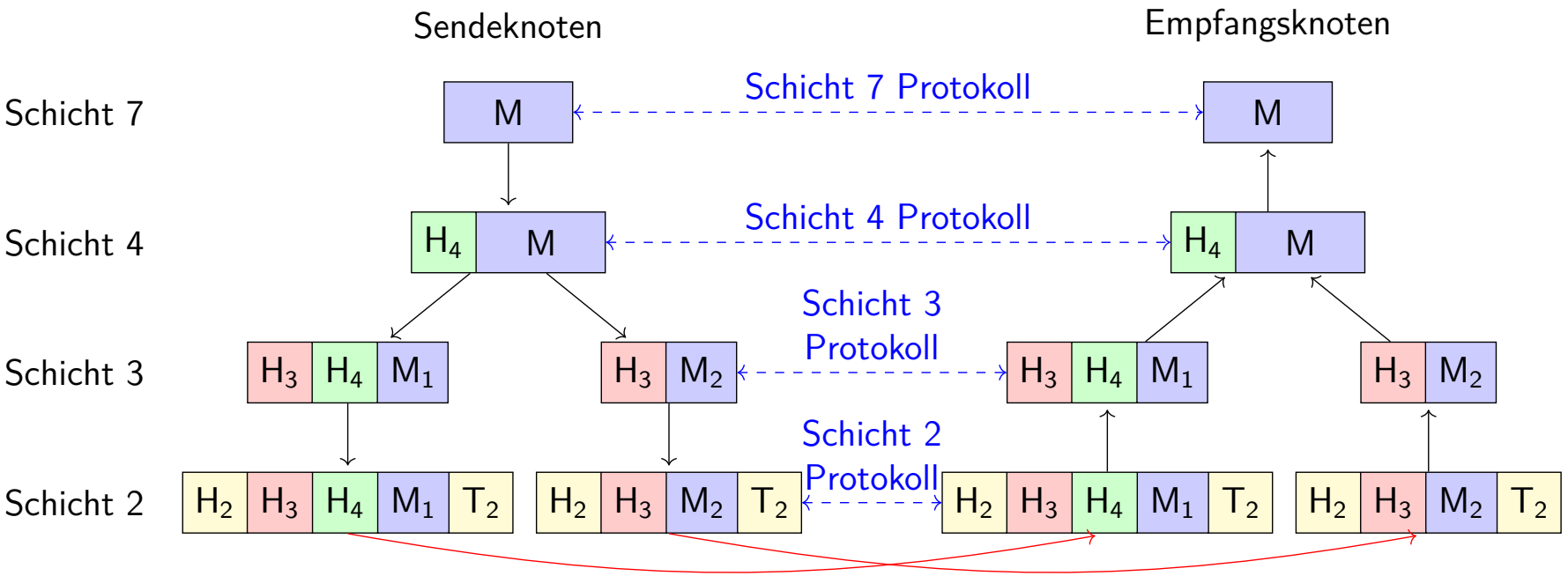
Informationsfluss im Schichtenmodell



Informationsfluss im Schichtenmodell



Informationsfluss im Schichtenmodell

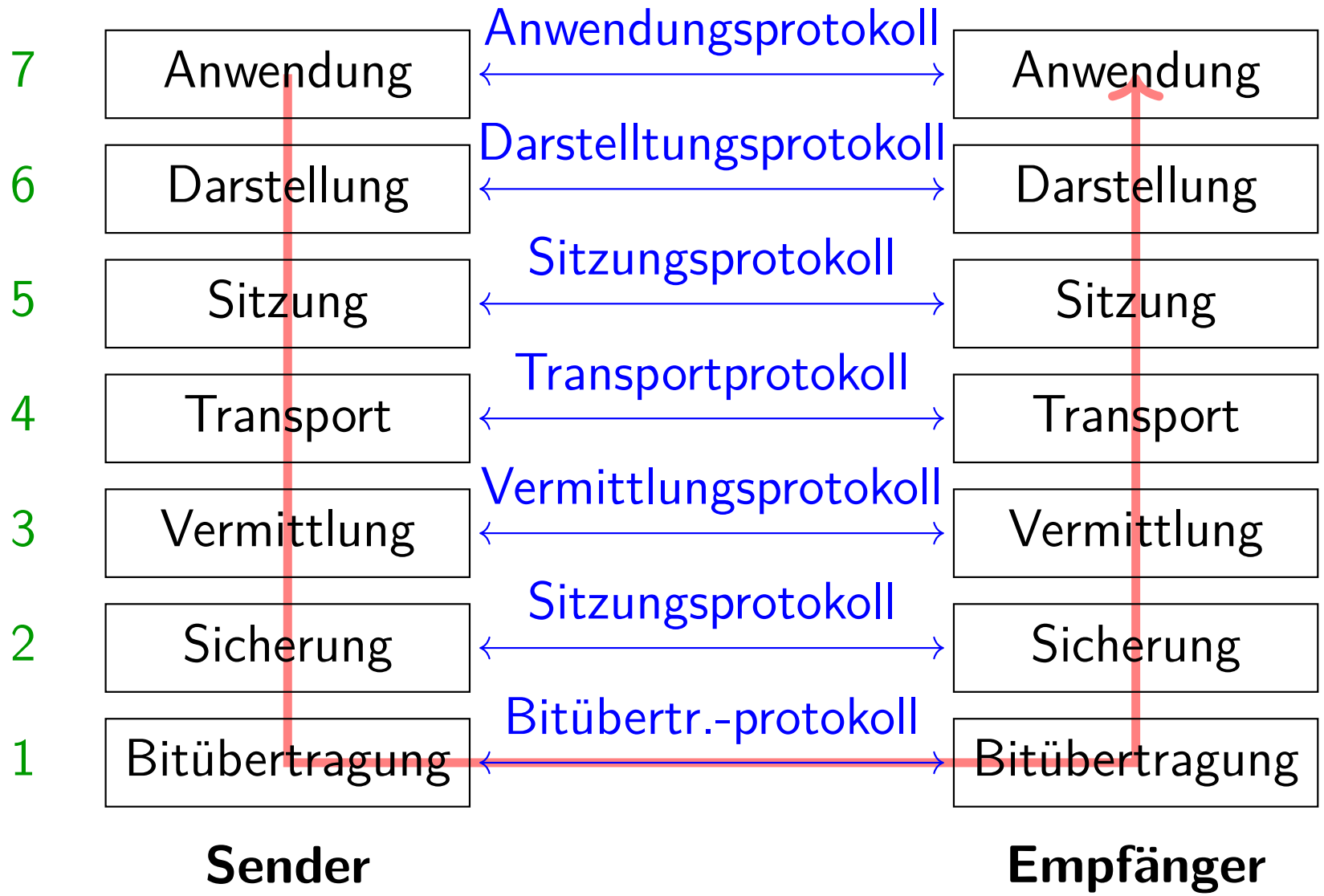


Referenzmodelle

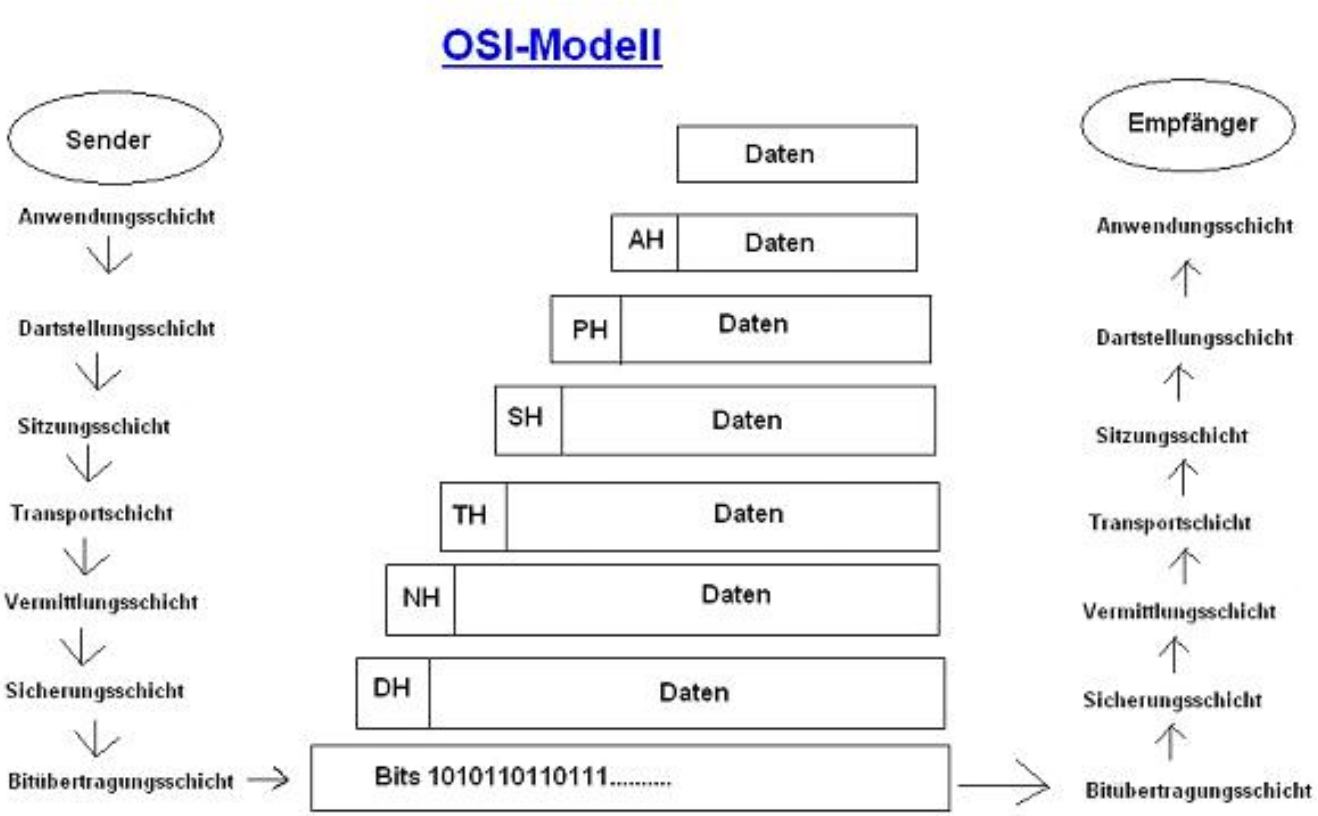
Als Designgrundlage dienen in Computernetzen zwei Modelle:

- Open-Systems-Interconnection-Modell (OSI-Modell)
- TCP/IP-Modell

OSI-Referenzmodell



OSI-Referenzmodell (2)



von https://informatiker.fandom.com/de/wiki/OSI-Referenzmodell

Bitübertragungsschicht (Physical Layer)

- Übertragung einzelner Bits
- Ziel: Wenn Sender 0 sendet, dann versteht Empfänger auch 0 und nicht 1
- Elektrische Spezifikation
  - Medium: Kabel, Glasfaser, Funk, Infrarot, ...
  - Was ist 0, was ist 1? Z.B. welche Spannung, Frequenz, ...
  - Zeitverhalten
  - Codierung und Modulationsverfahren
  - Simplex / Duplex - Übertragung
- Mechanische Spezifikation
  - Form / Art der Stecker und Kabel
  - Anzahl der Pins

Sicherungsschicht (Data Link Layer)

- Maskiert Fehler gegenüber Vermittlungsschicht
- Hierfür werden Daten in *Frames* aufgeteilt, die hunderte bis tausende Bytes umfassen
- Für zuverlässigen Dienste gibt es das Konzept der Bestätigungsframes.
- Zugriffskontrolle (MAC, Media Access Control)
  - physische Addressierung der Kommunikationspartner
  - regelt bei Mehrfachzugriffsverbindungen geordneten Zugriff auf Medium
- Beispiele: Ethernet für Multirechnernetzwerke, Point-to-Point Protocol (PPP) für Zweirechnernetzwerke

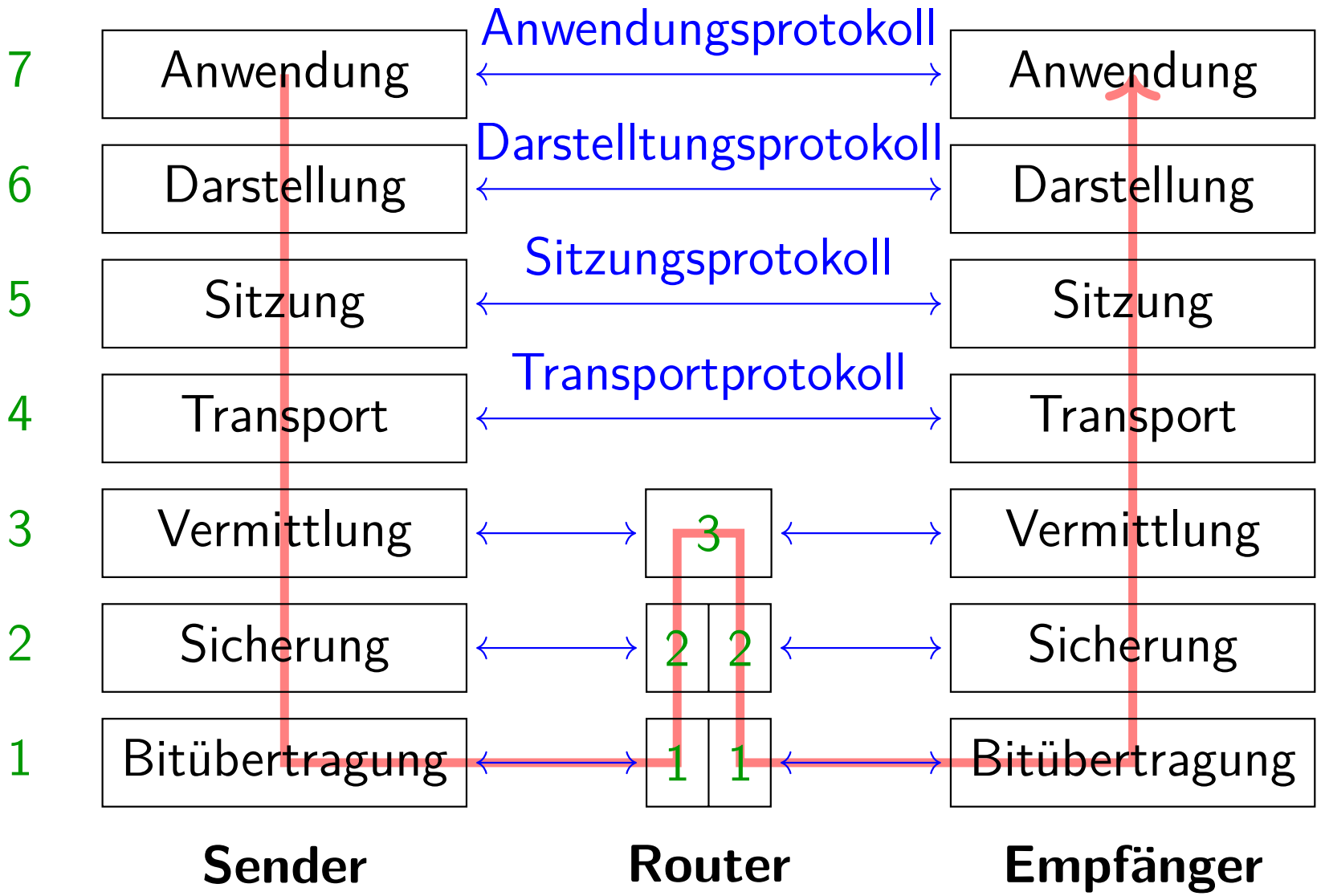
Vermittlungsschicht (Network Layer)

- Weiterleitung von Paketen inklusive des Routings
  - Bestimmung eines Weges von Sender oder Empfänger (statisch, dynamisch)

$\Rightarrow$ Unterste Schicht für Kommunikation zwischen nicht direkt verbundenen Netzwerkknoten

- Oberste Schicht der Netzwerkzwischenknoten (Router)
- Definiert einheitliches Adressierungsschema für alle Teilnehmer
- Beispiel: IP-Protokoll des Internets

OSI-Modell mit Netzwerkzwischenknoten (Router)



Transportschicht (Transport Layer)

- Ende-zu-Ende-Kommunikation zwischen zwei Endpunkten (Prozessen) auf verschiedenen Rechnern
- Stellt verbindungsorientierte Dienste bereit $\Rightarrow$ Partner erhalten Eindruck einer Leitungsvermittlung, selbst bei paketorientierten unteren Schichten
- Definiert Adressierung der zu kontaktierenden Prozessen
- Multiplexing verschiedener Verbindungen (durch Pakete)
- Auf- und Abbau von Verbindungen
- Sichert ggf. auch Datentransport zwischen Endpunkten (Fehlerbehandlung)
- Beispiel: TCP des Internets

Sitzungsschicht (Session Layer)

- Definiert Dienste zur Verwaltung von Sitzungen, z.B.
  - Wer darf wann senden? (Dialogsteuerung, korrekte Reihenfolge der Kommunikation)
  - Ggf. Verwaltung der Verbindungen
  - Atomare Aktionen (alles oder nichts)
  - Synchronisation (Weiterführung eines unterbrochenen Transfers)

In der Praxis befinden sich Teile davon in der Anwendungs- oder Transportschicht.

Darstellungsschicht (Presentation Layer)

- Unterste Schicht, die die Syntax und die Semantik der Daten kennt, z.B. Wert ist eine 32 Bit Ganzzahl
- Konvertiert Datenformate und Datendarstellungen (z.B. nach Big-Endian vom Sender und zurück beim Empfänger)
- Kann auch Kompression und Verschlüsselung enthalten

Anwendungsschicht (Application Layer)

Spezialisierte Dienste für viele Anwendungsbereiche:

- HTTP

Anwendungsschicht (Application Layer)

Spezialisierte Dienste für viele Anwendungsbereiche:

- HTTP (Hypertext Transport Protocol) → Übertragung von Webseiten

Anwendungsschicht (Application Layer)

Spezialisierte Dienste für viele Anwendungsbereiche:

- HTTP (Hypertext Transport Protocol) → Übertragung von Webseiten
- SMTP

Anwendungsschicht (Application Layer)

Spezialisierte Dienste für viele Anwendungsbereiche:

- HTTP (Hypertext Transport Protocol) → Übertragung von Webseiten
- SMTP (Simple Mail Transport Protocol) → Zum Austausch von E-Mails

Anwendungsschicht (Application Layer)

Spezialisierte Dienste für viele Anwendungsbereiche:

- HTTP (Hypertext Transport Protocol) → Übertragung von Webseiten
- SMTP (Simple Mail Transport Protocol) → Zum Austausch von E-Mails
- SMB

Anwendungsschicht (Application Layer)

Spezialisierte Dienste für viele Anwendungsbereiche:

- HTTP (Hypertext Transport Protocol) → Übertragung von Webseiten
- SMTP (Simple Mail Transport Protocol) → Zum Austausch von E-Mails
- SMB (Server Message Block) → Protokoll für Netzwerk-Dateisysteme (hauptsächlich Windows)

Anwendungsschicht (Application Layer)

Spezialisierte Dienste für viele Anwendungsbereiche:

- HTTP (Hypertext Transport Protocol) → Übertragung von Webseiten
- SMTP (Simple Mail Transport Protocol) → Zum Austausch von E-Mails
- SMB (Server Message Block) → Protokoll für Netzwerk-Dateisysteme (hauptsächlich Windows)
- NFS

Anwendungsschicht (Application Layer)

Spezialisierte Dienste für viele Anwendungsbereiche:

- HTTP (Hypertext Transport Protocol) → Übertragung von Webseiten
- SMTP (Simple Mail Transport Protocol) → Zum Austausch von E-Mails
- SMB (Server Message Block) → Protokoll für Netzwerk-Dateisysteme (hauptsächlich Windows)
- NFS (Network File System) → Protokoll für Netzwerk-Dateisysteme (hauptsächlich Unixartige)
- SSH

Anwendungsschicht (Application Layer)

Spezialisierte Dienste für viele Anwendungsbereiche:

- HTTP (Hypertext Transport Protocol) → Übertragung von Webseiten
- SMTP (Simple Mail Transport Protocol) → Zum Austausch von E-Mails
- SMB (Server Message Block) → Protokoll für Netzwerk-Dateisysteme (hauptsächlich Windows)
- NFS (Network File System) → Protokoll für Netzwerk-Dateisysteme (hauptsächlich Unixartige)
- SSH (Secure Shell) → Sicheres Protokoll zur Nutzung entfernter Rechner

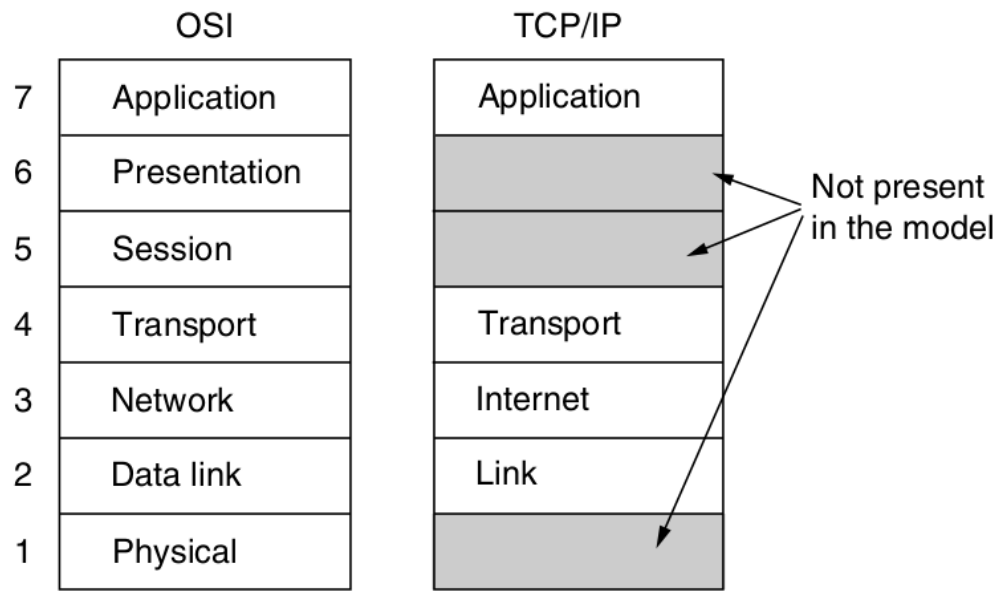
Geräte in Computernetzen

Gerät	**Schichten (OSI)**
Repeater	1
Hubs	1
Bridges[1]	1 – 2
Layer2-Switch	1 – 2
Layer3-Switch (VLAN)	1 – 3
Router	1 – 3
Gateway (Protokollumsetzer)	1 – 7
Firewall	1 – 7
Endgerät	1 – 7

[1] Modems, WLAN-Basisstationen, ...

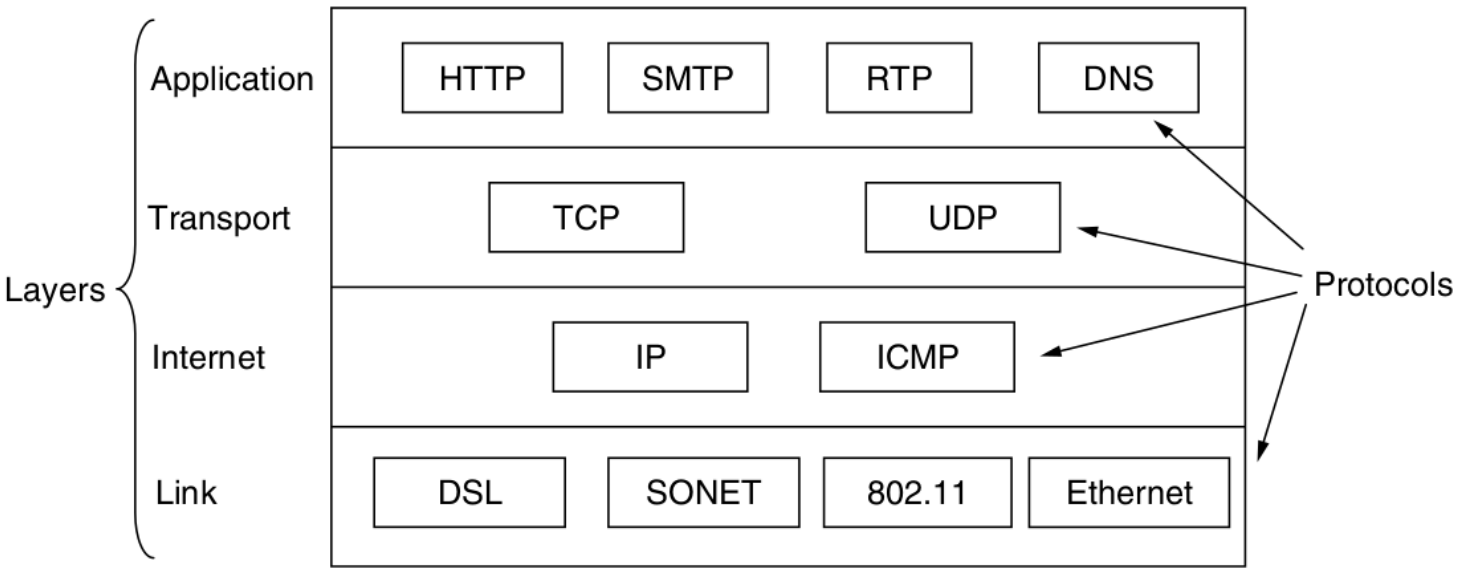
TCP/IP-Referenzmodell

- Verminderte Anzahl von Schichten
- Schichten korrespondieren am ehesten so:



- Folgte aus ARPANET, einem Forschungsnetz

TCP/IP-Protokollfamilie



Protokolle sind formal spezifiziert, z.B. in den Requests for Comments (RFC) der Internet Engineering Task Force (IETF).

Im April 2020 ca. 8750 RFCs.

Flow im TCP/IP-Modell

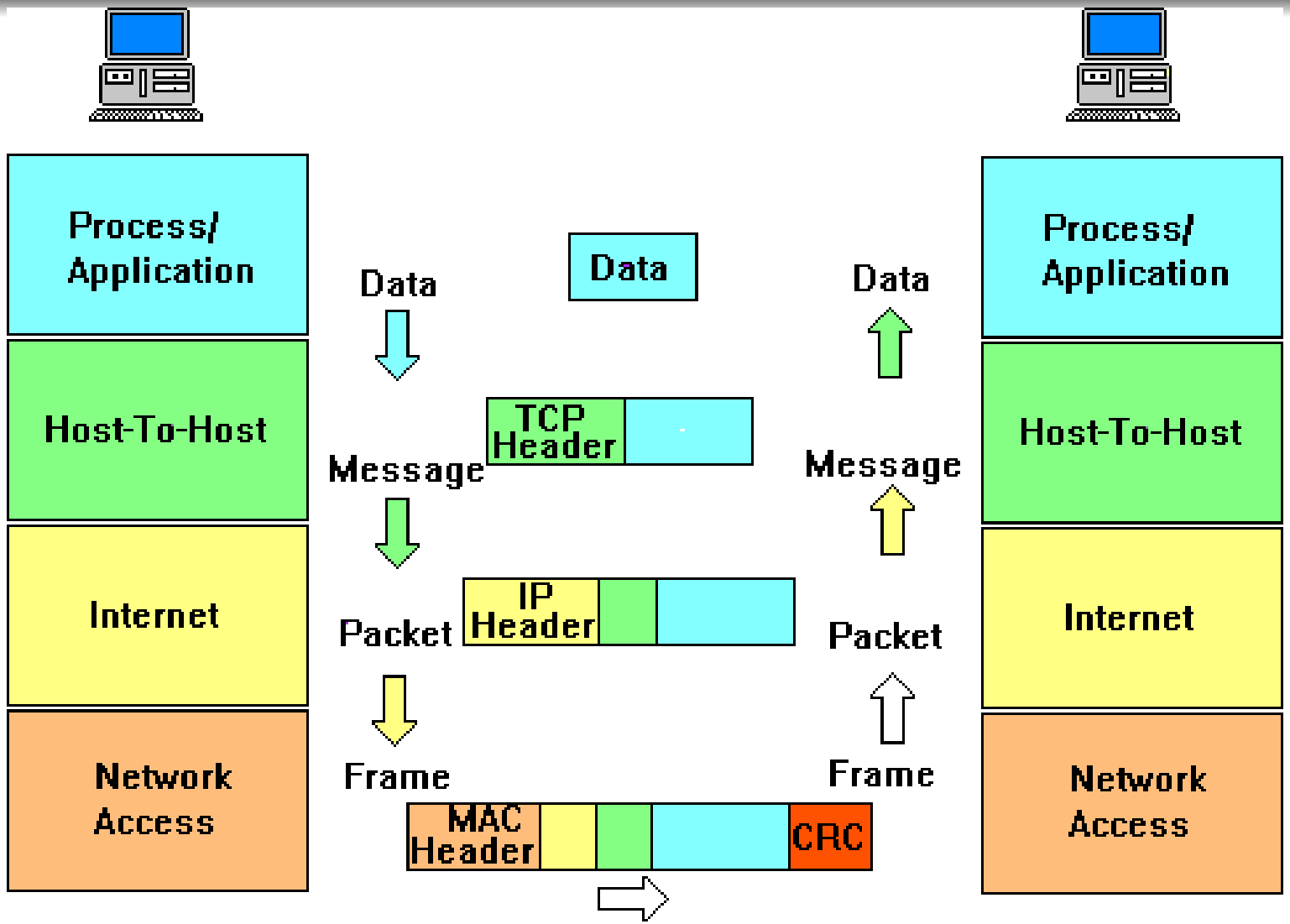
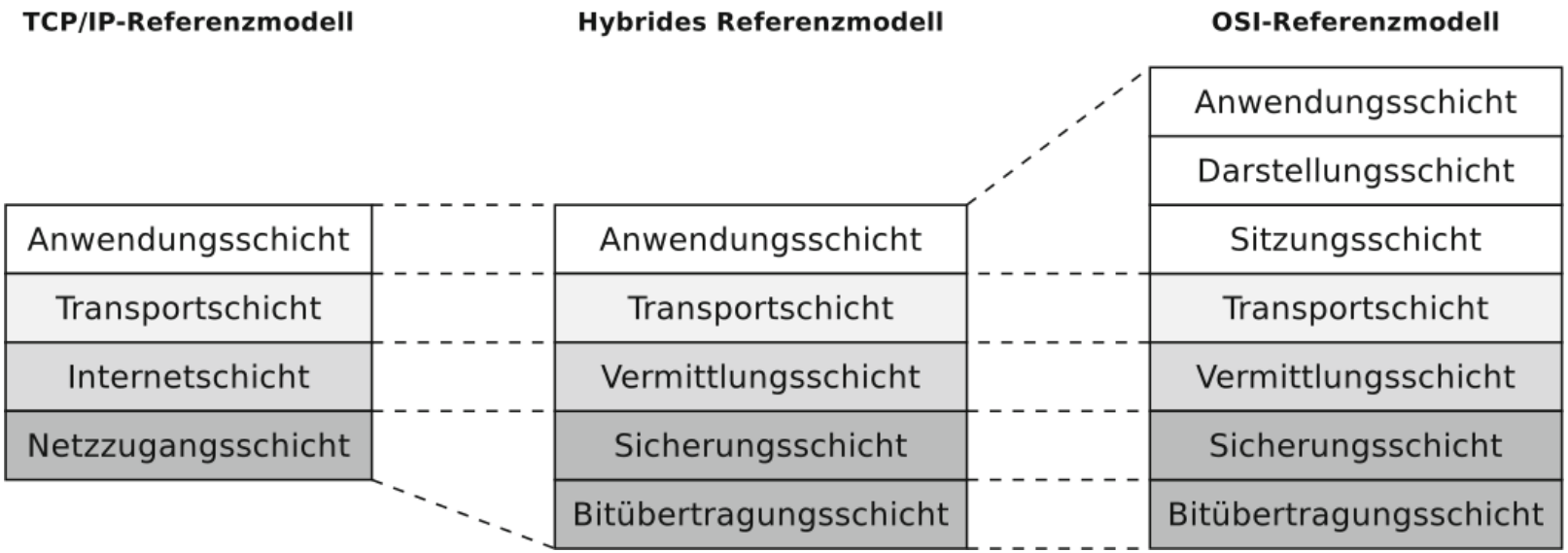


Bild von http://www.netzmafia.de/skripten/netze/netz8.html

Hybrides Referenzmodell

- Trennung zwischen Sicherungssicht und Bitübertragungsschicht sinnvoll
- Separate Darstellungs -und Sitzungsschicht wenig sinnvoll



aus *Computernetze kompakt* (2018)

Datenaustauschsformen

Dienstschnittstellen kommen in zwei verschiedenen Formen:

1. **Verbindungslos**
   - Jede Nachricht (*Datagram*) wird separat geschickt
   - Empfänger wird immer explizit spezifiziert
   - $\approx$ Brief- oder Telegramsendungen

Datenaustauschsformen

Dienstschnittstellen kommen in zwei verschiedenen Formen:

1. **Verbindungslos**
   - Jede Nachricht (*Datagram*) wird separat geschickt
   - Empfänger wird immer explizit spezifiziert
   - ≈ Brief- oder Telegramsendungen
2. **Verbindungsorientiert**
   - Verbindung zu einem Empfänger wird hergestellt
   - Darüber werden Nachrichten an Empfänger gesendet (*Datenstrom*)
   - Empfänger kann über Verbindung antworten
   - Verbindung wird geschlossen
   - ≈ Telefonat

Zuverlässigkeit

- Nachrichten werden mit einer angestrebten Dienstgüte (*quality of service*; QoS) übermittelt, z. B.
  - Alle gesendeten Daten kommen vollständig an
  - Die Daten kommen in der richtigen Reihenfolge an
  - Es gibt keine Duplikate

Zuverlässigkeit

- Nachrichten werden mit einer angestrebten Dienstgüte (*quality of service*; QoS) übermittelt, z. B.
  - Alle gesendeten Daten kommen vollständig an
  - Die Daten kommen in der richtigen Reihenfolge an
  - Es gibt keine Duplikate
- Hierfür nutzt man
  - Sequenznummern in den Nachrichten
  - Empfangsbestätigungen
  - Sendewiederholungen

Beispiele für Datenaustauschformen

Verlässl. Nachrichtenstrom	Textseiten
Verlässl. Bytestrom	Download
Unverlässl. Verbindung	Voice over IP
Unverlässl. Datagram	Spam
Bestätigtes Datagram	Instant-Messaging
Anfrage-Antwort	Datenbankabfrage

Verbindungslos

Verbindungsorient.

Elemente verbindungsorientierter Dienste

LISTEN	Auf eingehende Verbindung warten
CONNECT	Eine Verbindung mit dem entfernten Rechner anfragen
ACCEPT	Eine Verbindung annehmen
RECEIVE	Daten empfangen
SEND	Daten senden
DISCONNECT	Verbindung schießen

Funktionen der BSD-Socket-Schnittstelle, die Zugriff auf diese Elemente erlauben: `listen()`, `connect()`, `accept()`, `recv()`, `send()` und `close()`.

Beispiel einer Client-Server-Interaktion

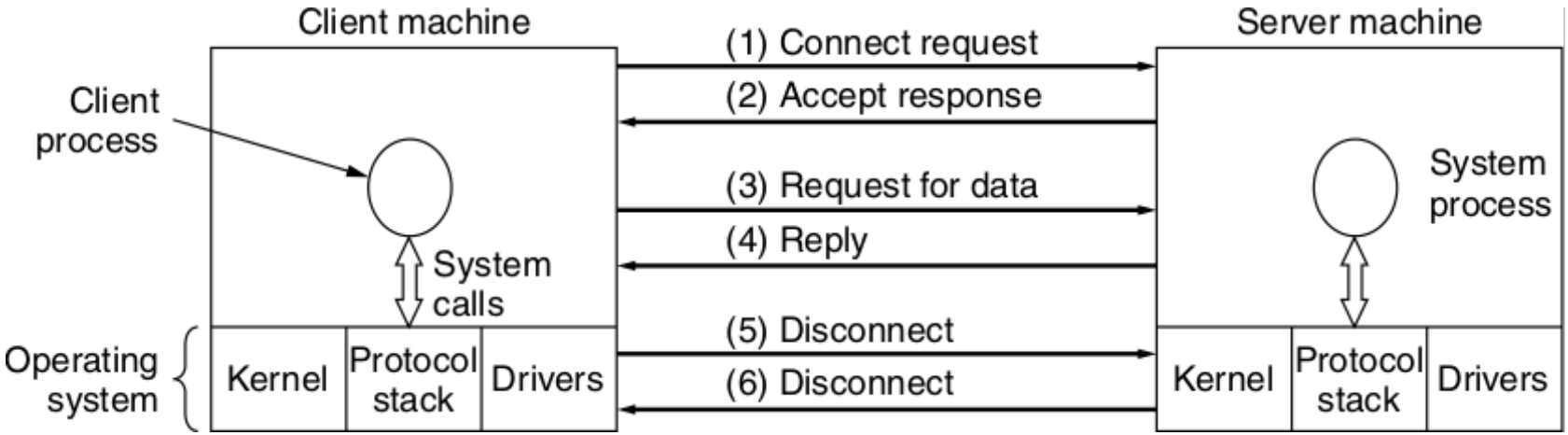


Bild aus *Computer Networks* (Tanenbaum)

Outline

Information

Information ist:

- durch Sinnesorgane vom Menschen empfangbar
- Verringerung von Unsicherheit bzw. Nichtwissen der Nutzer
- kommunikationsabhängig; benötigt Kommunikationskanal
- kopierbar

Beispiele

Texte, Bilder, Töne, ...

Repräsentation von Information

Im klassischen Computer:

- Kleinste Informationseinheit ist 1 Bit mit zwei physikalischen Zuständen (z.B. +3 und -3 V)
- Wir stellen diese Zustände mit Symbolen 0 und 1 dar
- Mit zwei Belegungen kommt man nicht weit $\Rightarrow$ nehmen Bitfolgen der Länge n: $I_n \in \{0,1\}^n$

Himmelsrichtung

Wir benötigen vier Zustände, also 2 Bits und *kodieren* wie folgt: Norden $\rightarrow$ 00, Osten $\rightarrow$ 01, Süden $\rightarrow$ 10, Westen $\rightarrow$ 11.

Kodierung

Kodierung

Eine *Kodierung* über den Mengen A und B ist eine injektive Abbildung $c : A \to B$. Bei uns ist B eine Menge von Bitfolgen.

Aufgabe

1. Erweitere unsere Kodierung der Himmelsrichtung, sodass auch Nebenhimmelsrichtungen erfasst werden können.
2. Wie viele Bits braucht man, um m Symbole als Bitfolge darzustellen?

Textkodierung

- Heute besteht üblicherweise 1 Byte aus 8 Bit (früher auch weniger), 8-Bit-Byte wurde auch Oktett genannt
- Bekannteste Kodierung für Computer ist *American Standard Code for Information Interchange* (ASCII)
- ASCII ist 7 Bit, Bit 8 für Fehlerkorrektur reserviert (als Paritätsbit; praktisch nicht mehr relevant)
- ASCII besteht aus 33 nicht druckbaren sowie 95 druckbaren Zeichen
- Groß- und Kleinbuchstaben unterscheiden sich um ein Bit
- Viele Textkodierungen sind davon abgeleitet

ASCII

Code	...0	...1	...2	...3	...4	...5	...6	...7	...8	...9	...A	...B	...C	...D	...E	...F
0...	*NUL*	*SOH*	*STX*	*ETX*	*EOT*	*ENQ*	*ACK*	*BEL*	*BS*	*HT*	*LF*	*VT*	*FF*	*CR*	*SO*	*SI*
1...	*DLE*	*DC1*	*DC2*	*DC3*	*DC4*	*NAK*	*SYN*	*ETB*	*CAN*	*EM*	*SUB*	*ESC*	*FS*	*GS*	*RS*	*US*
2...	SP	!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
3...	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4...	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5...	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
6...	`	a	b	c	d	e	f	g	h	i	j	k	l	M	n	o
7...	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	DEL

Welches Bit kennzeichnet Groß- und Kleinvarianten?

Keine Umlaute und andere Sonderzeichnen, die für nicht-englische Sprachen nützlich sind!

Nichtdruckbare Werte bei ASCII

- Zeichen, denen normalerweise kein Symbol zugeordnet ist
- Haben Bedeutung in der Steuerung von Ausgabegeräten
- 0x00 - 0x1F sowie 0x7F gehören dazu

0x00 NUL-Byte, kennzeichnet oft Ende of Zeichenketten, `"\0"`

0x09 HT, Horizontaler Tabulatorzeichen, `"\t"`

0x0A LF, Line-feed → Nächste Zeile, `"\n"`

0x0D CR, Carriage-Return → Wagenrücklauf, `"\r"`

0x1B ESC, erlaubt weitere Sonderfunktionen, die über die nächsten Zeichen spezifiziert ist (z.B. Farbe), `"\033"`

LF markiert in Textdateien Zeilenende, manchmal (bei Windows) ist CR + LF dazu nötig.

Beispiel ASCII

```
$ od -t x1z -w8 zauberlehrling.txt
0000000 44 65 72 20 5a 61 75 62  >Der Zaub<
0000010 65 72 6c 65 68 72 6c 69  >erlehrli<
0000020 6e 67 2e 0a 0a 48 61 74  >ng...Hat<
0000030 20 64 65 72 20 61 6c 74  > der alt<
0000040 65 20 48 65 78 65 6e 6d  >e Hexenm<
0000050 65 69 73 74 65 72 2c 0a  >eister,.<
0000060 53 69 63 68 20 64 6f 63  >Sich doc<
0000070 68 20 65 69 6e 6d 61 6c  >h einmal<
0000100 20 77 65 67 62 65 67 65  > wegbege<
0000110 62 65 6e 21 0a 55 6e 64  >ben!.Und<
0000120 20 6e 75 6e 20 73 6f 6c  > nun sol<
...
```

ISO 8859

- Familie von Normen für 8-Bit-Zeichenkodierungen
- Codes 0 - $7F_{16}$ von ASCII übernommen
- Ziel: Aufnahme von Zeichen, die für andere Sprachen benötigt werden (z.B. Umlaute im Deutschen)
- Hierfür Codes $A0_{16}$ bis FF_{16}
- Nur für Sprachen, die auf lateinischen Symbolen basieren (deshalb auch ISO-Latin genannt)

Auszug von ISO 8859-1 (westeuropäische Sprachen)

$C4_{16}$ = Ä, $E4_{16}$ = ä, $D6_{16}$ = Ö, DC_{16}=Ü

Unicode

- Begrenzung auf 1 Byte-Zeichen sehr limitierend
- Unicode ab 1991: Internationaler Standard, um digitalen Code für jedes sinnvolle Schriftzeichen oder Textelement festzulegen
- Ersten 256 Codepunkte identisch zu ISO 8859-1
- Codepunktebereich: U+0000 bis U+10FFFF_{16}
- Davon ca. 144 700 definiert (Version 14.0, September 2021)
- Es erscheinen regelmäßig neue Versionen
- Verschiedene Kodierungen/Transformationen der Codepunkte sind definiert, z.B.: UTF32, UTF16, UTF8

UTF32

- Einfachste Kodierung eines Unicode-Zeichens
- Bildet Codepunkt direkt auf demselben 32 Bit Wert ab
- Dadurch: Leichter wahlfreier Zugriff auf einzelne Codepunkte (aber: manchmal mehrere Codepunkte für ein Zeichen nötig)
- Probleme
  - hoher Speicherbedarf
  - Byte-Reihenfolge (Big-/Little-Endian)
    - UTF-32BE und UTF-32LE sind definiert. Vorgabe: BE
    - Byte-Order-Mark am Beginn kennzeichnet BE oder LE

Aufgabe

Kodiere Zeichenkette *Flöte* nach UTF32BE.

UTF8

- Am häufigsten verwendete Kodierung von Unicode Codepunkten im Web
- Variable Kodierung: Codepunkte werden mit ein bis vier Bytes kodiert
- Definiert in RFC 3629 mit folgenden Schema:

```
0000 0000-0000 007F → 0xxxxxxx
0000 0080-0000 07FF → 110xxxxx 10xxxxxx
0000 0800-0000 FFFF → 1110xxxx 10xxxxxx 10xxxxxx
0001 0000-0010 FFFF → 11110xxx 10xxxxxx 10xxxxxx 10xxxxxx
```

Aufgabe

Kodiere Zeichenkette *Flöte* nach UTF8.

Tools für Textkodierung

`iconv`

- Tool, um Texte von einer Codierung zu einer anderen umzuwandeln

Umwandeln von ISO 8859-15 nach UTF8

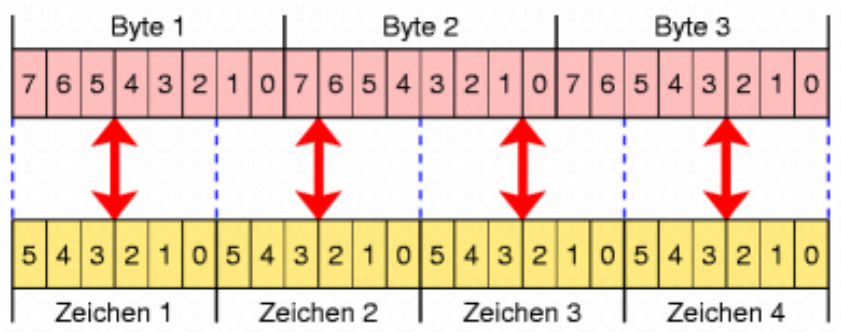
```
iconv -f ISO8859-15 -t UTF8 <input.txt
>output.txt
```

Base64

- Kodierung von 8-Bit-Binärdaten durch 64 druckbare ASCII-Zeichen plus weiteres Zeichen (=) zur Steuerung
- Kodierte Daten lassen sich auf dem Terminal ausgeben
- Ursprünglich für den E-Mail-Verkehr entwickelt, da frühere Mail-Server nicht 8-Bit-Clean
- Aktuelle Version definiert in RFC 4648
- Inzwischen mehr Anwendungen, z. B. Binärdaten in XML oder Skripte, Fingerprints

Wie viele 8-Bit-Zeichen werden ausgegeben, wenn 24-Bit-Binärdaten mit Base64 kodiert werden?

Base64 – Schema



Dez	Bin	Hex	Zeichen	Dez	Bin	Hex	Zeichen
0	000000	0	A	26	011010	1A	a
1	000001	1	B	27	011011	1B	b
2	000001	1	C	28	011100	1C	a
...	...	...	...	...	...	...	...

Komplette Tabelle in RFC 4648

Wie viele Bytes nimmt eine n-Byte große Datei mindestens ein, wenn nachdem sie mit Base64 kodiert wurde?

Base64 – Beispiel: Bilder in HTML einbetten

```
<html>
<body>
<img src="
iVBORw0KGgoAAAANSUhEUgAAAAsAAAALCAMAAACecocUAAAABGdBTUEAALGPC/xhBQAAACBjSFJN
AAB6JgAAgIQAAPoAAACA6AAAdTAAAOpgAAA6mAAAF3CculE8AAAAbFBMVEX////7+0H7+xT78wT7
2xT7yjX7+wT74wD72wD7zgD7vgD7+wAkDAAtAAAAAAD7xgD7rhD7+xwxBAA5AAD71gD7sgD7ogD7
6wD7wgAYCAD77wD78wAxAAD70gz73wDWnlH7+1X70gD7nhj7rgBspddiAAAAAWJLR0QAiAUdSAAA
AAd0SU1FB+YKDBYwIPAsPN8AAABoSURBVAjXJY1BAoIwEAMjandt6iJQVMAV0f//0VbnkrkkAYBd
sz8c8SM0QfQUq5LprGbtBejIfsjFxysCeRNVvRefEudFRFt7wCefl6cWX/Hy5JLzZuO7lL0fzCx+
6mjnlfg/IylbyS9j8AY9crK7BAAAACV0RVh0ZGF0ZTpjcmVhdGUAMjAyMi0xMC0xMlQyMjo0ODoz
MiswMDowMHyBLk4AAAAldEVYdGRhdGU6bW9kaWZ5ADIwMjItMTAtMTJUMjI6NDg6MzIrMDA6MDAN
" />
</body>
</html>
```

Mit dem Kommando base64 können Binärdatein nach Base64 und zurück kodiert werden.

Ende

Ende