Verteilte Systeme

Organisatorisches

Termine

- 13 Vorlesungen: 12.4. - 5.7.2011, Di, 8:30-10:00
- 10 Übungstermine in voraussichtlich 2 Gruppen:
  - Fr, 8:30-10:00
  - N.N. (voraussichtlich Do 10-12)

    Die Übungstermine in der ersten Woche (15.4.), an Ostern (23.4.) und Himmelfahrt (3.6.) entfallen.
- Klausur: 12.7.2011

Scheinkriterien

- n-1 Übungen Bearbeitet (E(n) = 9)
- n-2 Übungen im wesentlichen korrekt gelöst
- Anwesenheit in den den Tutorien (maximal zwei Fehltage)
- Bestehen der Klausur

Literatur

- George Coulouris, Jean Dollimore, Tim Kindberg
  Distributed Systems - Concepts and Design forth Edition
  Addison Wesley 2005

- Nancy A. Lynch
  Distributed Algorithms
  Morgan Kaufmann 1997

- A.S. Tanenbaum, M. v.Steen
  Distributed Systems: Principles and Paradigms
  Prentice Hall 2006

Verteilte Systeme

Einführung

Was sind Verteilte Systeme?

- A distributed system is one in which components located at networked computers communicate and coordinate their actions only by passing messages. [Coulouris]

- Verteiltes System (distributed system) :
  Prozessoren bzw. Prozesse haben keinen gemeinsamen Speicher und müssen daher über Nachrichten kommunizieren. [Löhr]

Was sind Verteilte Systeme?

- Nichtsequentielle (concurrent) Programmiersprache ohne gemeinsame Variable
- Betriebssystem mit grundsätzlich disjunkten Prozeß-Adreßräumen
- Multiprozessorsystem ohne gemeinsamen Speicher
- Mehrrechnersystem oder Rechnernetz
- Das Internet

Achtung! Abstraktionsebenen

Systeme können auf manchen Abstraktionsebenen Verteilt, auf anderen aber zentralisiert sein.

Beispiele:

- Nichtsequentielle Programmiersprache ohne gemeinsame Variable auf einem Unix-System
- Gemeinsame Objekte auf einem Middelware-System in einem Rechnernetz.

Wozu verteilte Systeme ?

- Parallelverarbeitung
- Client/Server-Betrieb
  statt Teinehmerbetrieb
- Ausfallsicherheit
- Verteilte Anwendungen
- Netzwerkdienste
  - Dateiübertragung
    (file transfer)
  - Fernnutzung
    (remote login)
  - Ressourcenverbund
    (resource sharing)
  - Web, Chat, News,
    Videostreaming, …

Problemfelder

- hochgradige Nichtsequentialität
- unhandlicher Nachrichtenaustausch
- kein Gesamtzustand, der von allen Beteiligten beobachtbar wäre
- Fehlfunktionen von Rechnern und Kommunikationsnetz
- Heterogenität von Rechnern, Betriebssystemen, Teilnetzen
- dynamische Änderung der Systemstruktur
- Sicherheit viel stärker gefährdet als bei zentralisierten Systemen

Abwägung

- Verbergen der schwierigen Problembehandlung durch Bereitstellung geeigneter Abstraktionen für komfortable Anwendungsprogrammierung

versus

- Optimierung der Anwendung und Anpassung von Algorithmen an die konkreten Gegebenheiten im verteilten System

Verteilte Systeme

Klassifizierung von Kommunikationsdiensten

Kommunikationssysteme

- Kommunikationsdienst:

  Operationen zum Senden/Empfangen von Nachrichten

- Kommunikationsprotokoll:

  Vereinbarung zwischen Sender und Empfänger darüber, wie die Daten/Nachrichten übertragen werden

  entspricht in etwas der Implementierung eines Kommunikationsdienstes

Kommunikationssysteme

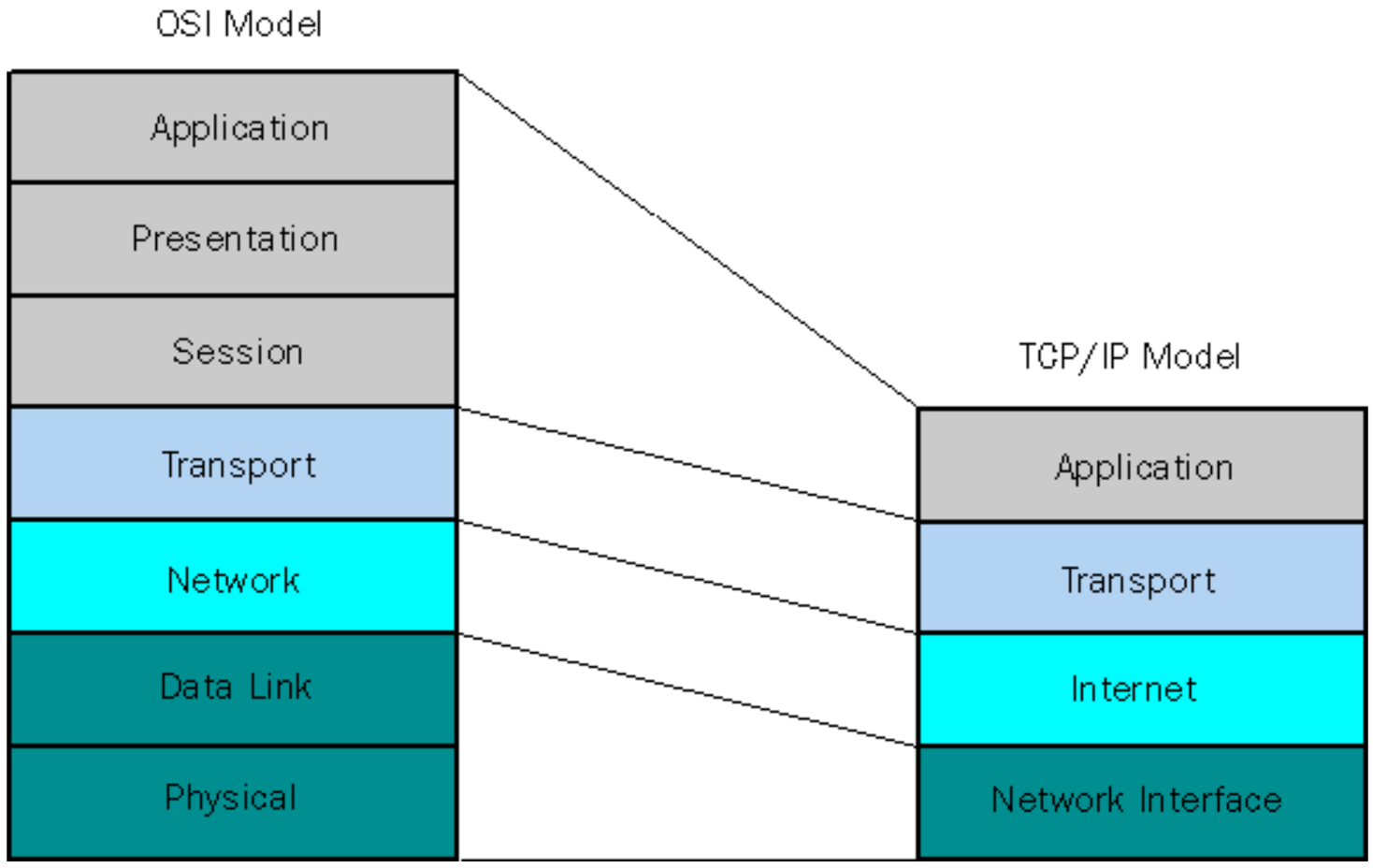
- Kommunikationshardware:

  Hardware, die verwendet wird um die Kommunikations-protokolle auszuführen (Thema von TI-3 / Telematik)

Kommunikationsprotokolle

- Adressierung von Kommunikationspartnern
- Erkennung/Korrektur von Übertragungsfehlern
- Flußsteuerung (Pufferüberlauf, Stau)
- Vermittlung über Umwege (Routing)
- heterogene Datenrepräsentation

Schichtenmodell



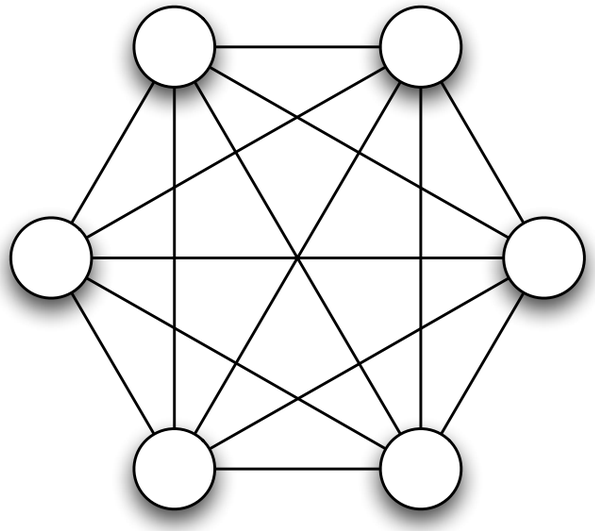
Kommunikationssysteme

- Wichtige Aspekte für Verteilte Systeme:
  - Erkennung/Korrektur von Übertragungsfehlern
  - Topologie
  - Verbindungslose vs. verbindungsorientierte Kommunikation

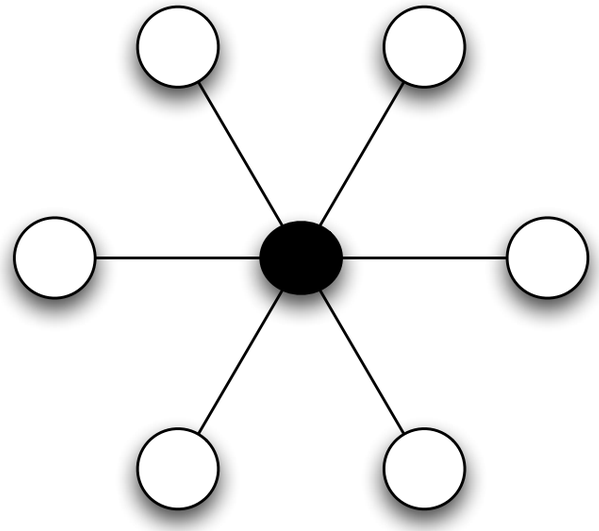
Übertragungsfehler

- Paketverlust
- Paketveränderung
- Paketduplizierung
- Paketreihenfolge

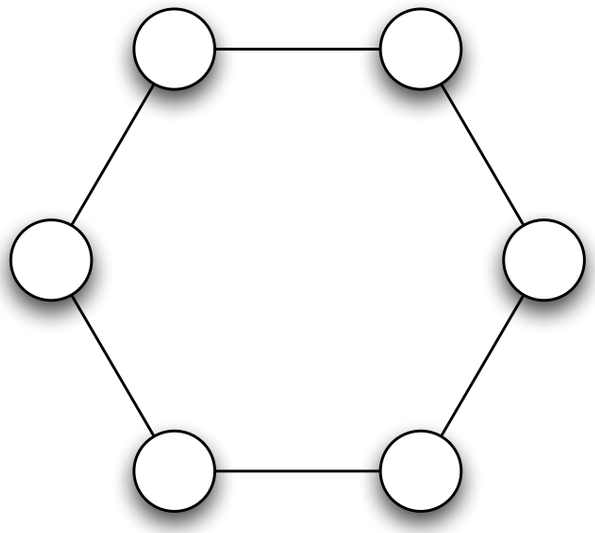
Topologien



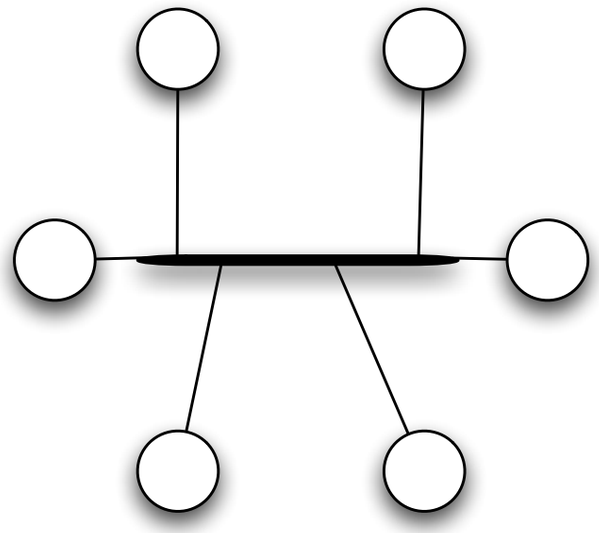
Punkt zu Punkt



Stern

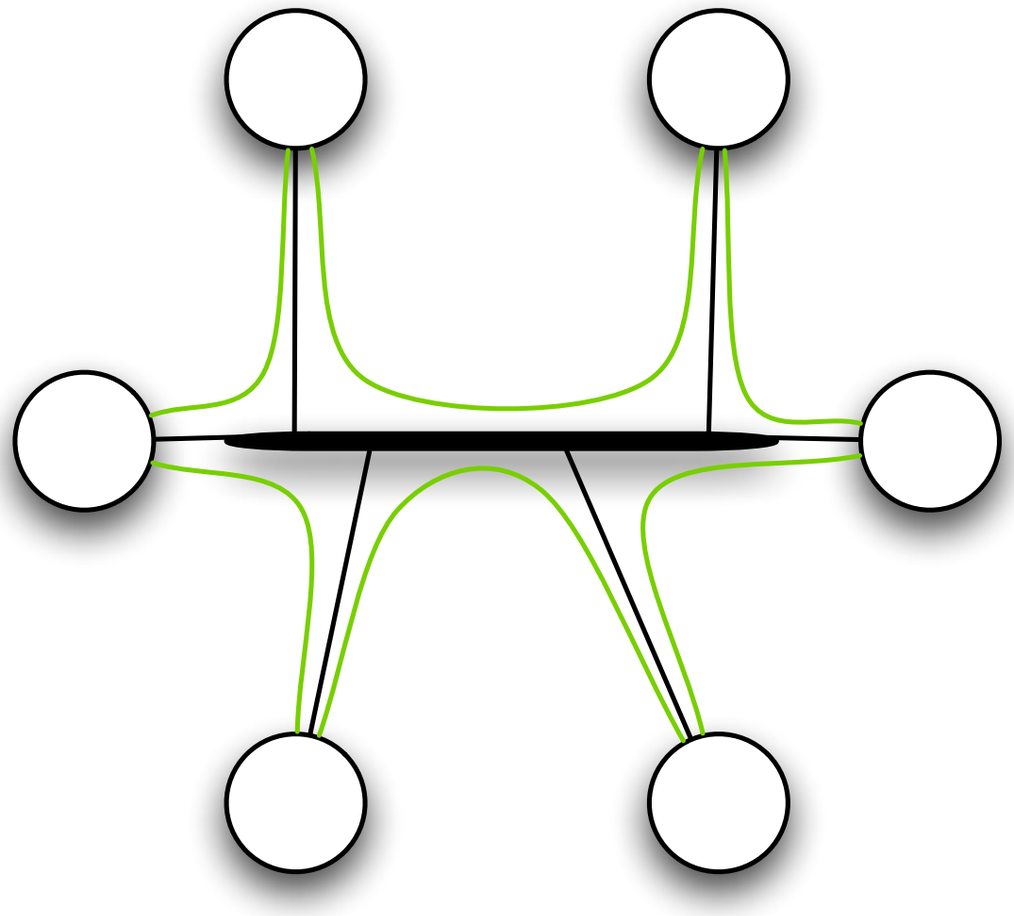


Ring



Bus

Topologien



physisch: Bus logisch: Ring

Verbindungslose vs. verbindungsorientierte Dienste/Protokolle

- verbindungslos (connectionless) (z.B. IP, UDP)
  Nachrichten werden ohne Vorbereitung „bestmöglich“ (best effort), aber ohne jede Zuverlässigkeitsgarantie (betr. Reihenfolge, Verlust, Duplizieren) übertragen.

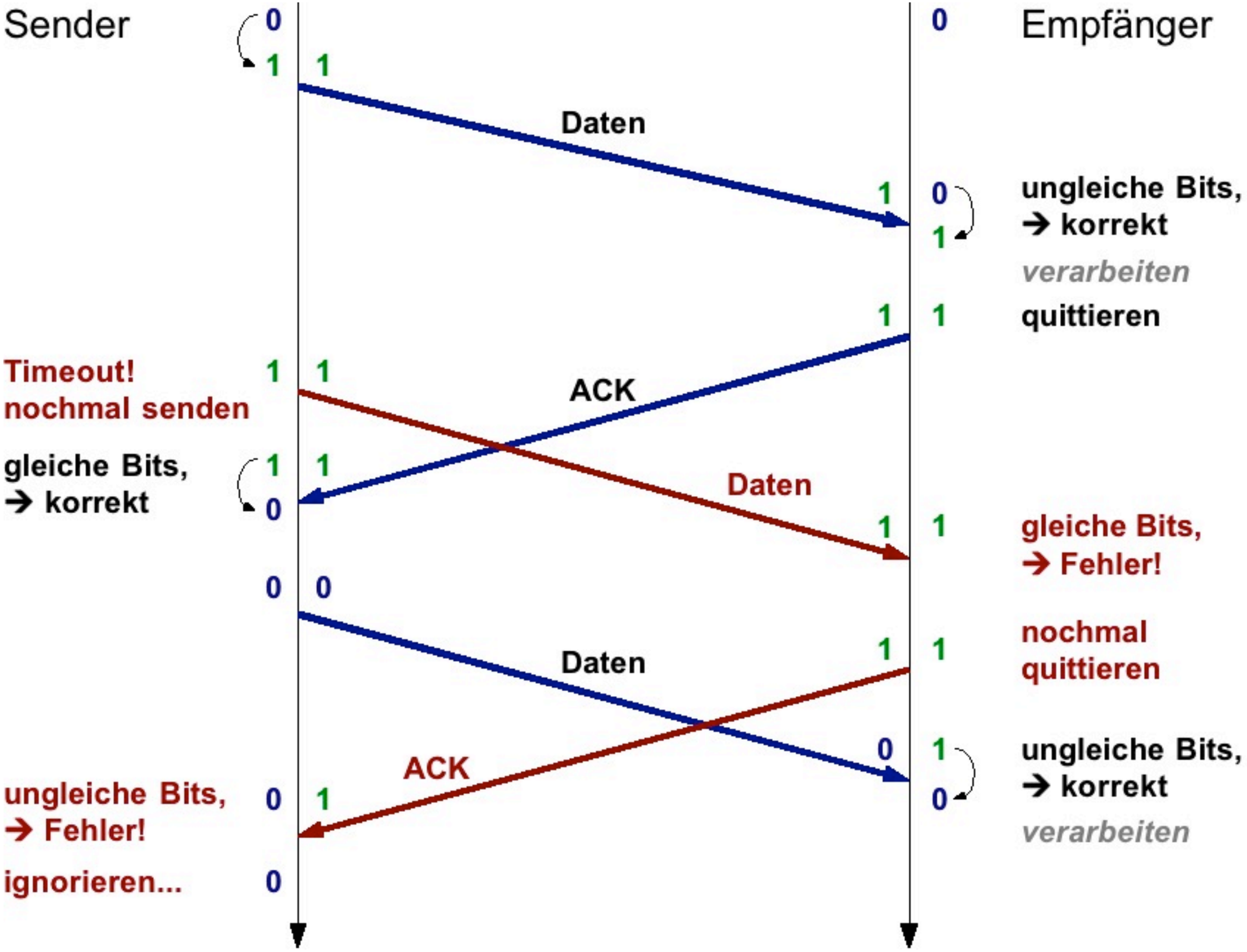
- verbindungsorientiert (connection-oriented) (z.B. TCP, X.25)Sender und Empfänger stellen eine Verbindung zwischen- einander her, d.h. sie etablieren einen virtuellen Kanal, über den ein zuverlässiger Nachrichtenfluß möglich ist:
  Invariante: die Folge der empfangenen Nachrichten ist Präfix der Folge der gesendeten Nachrichten
  Lebendigkeit: jede gesendete Nachricht kann auch irgend- wann empfangen werden.

Beispiel: Alternating Bit Protocol

realisiert zuverlässigen Simplex-Kanal (unidirektional)über unzuverlässigen Duplex-Kanal (bidirektional) mit Flußsteuerung und Fehlerbehandlung.

Idee:

- jede Nachricht einzeln quittieren (acknowledgement, ACK) wenn Quittung ausbleibt (timeout!), Nachricht wiederholen; wenn Nachricht ausbleibt, Quittung wiederholen.
- Nachrichten und Quittungen durchnumerieren.
- Durchnumerieren modulo 2 genügt.


Sender
Empfänger
0
1
1
Daten
1
0
1
ungleiche Bits,
→ korrekt
verarbeiten
1
1
quittieren
Timeout!
nochmal senden
1
1
ACK
gleiche Bits,
→ korrekt
1
1
0
Daten
1
1
gleiche Bits,
→ Fehler!
0
0
1
1
nochmal
quittieren
Daten
0
1
0
ACK
ungleiche Bits,
→ Fehler!
0
1
ignorieren...
0