Verteilte Systeme

Oktober - November 2023

4. Vorlesung – 26.10.2023

Kurs: TINF21AI1

Dozent: Tobias Schmitt, M.Eng.

Kontakt: d228143@
student.dhbw-mannheim.de

Wiederholungsfragen

- Welche Arten der Kommunikation können Sie unterscheiden?
- Was ist die Idee hinter einem RPC (Remote Procedure Call)?
- Welche Schwierigkeiten finden bei der Kommunikation zwischen verschiedenen Rechnern (Diensten) in heterogenen Umgebungen?
- Nennen Sie Beispiele für flüchtige und persistente Kommunikation.

Wiederholungsfragen

- Was verstehen Sie im Zusammenhang mit Datenstreams unter einer
  - asynchronen Übertragung?
  - synchronen Übertragung?
  - isochronen Übertragung?
- Durch welche Möglichkeiten kann man die limitierte Dienstgüteregulierung durch die Verwendung des IP-Protokolls ausgleichen?

- Warum werden bei Multicast-Kommunikation in Peer-to-Peer-Netzwerken selten optimale Routen verwendet?
- Welche Metriken zur Einschätzung der Qualität eines Multicast-Baumes kennen Sie?
- Was können Sie zur Informationsverbreitung in Peer-to-Peer-Netzwerken basierend auf endemisches Verhalten sagen?

Themenüberblick

- **Benennung und Namenssysteme**
  - Begrifflichkeiten
  - Lineare Benennung
  - Hierarchische Benennung
  - (Attributbasierte Benennung)
- Synchronisierung

Benennung und Namenssysteme Einstiegsfragen

- Was verstehen Sie unter den Begriffen Entität, Adresse und Bezeichner?

- Wie findet man den Rechner zu einer IP-Adresse in lokalen Netzwerken?

- Wie kann das Finden von Entitäten implementieren?
  - Auf einem Rechner?
  - Auf einem Rechner unter Verwendung externer Laufwerke?
  - In verteilten Systemen?
  - Im Internet?

Benennung und Namenssysteme

- **Begrifflichkeiten**
  - Entitäten
    - Bsp.: Hosts, Drucker, Festplatten, Dateien, …
  - Adresse einer Entität
    - Entspricht Zugriffspunkt auf eine Entität

Benennung und Namenssysteme

- **Begrifflichkeiten**
  - Bezeichner einer Entität
    - Unabhängig von Adresse → ortsunabhängig
    - Einfachere und flexiblere Handhabung
    - Echter Bezeichner, wenn
      - Jeder Bezeichner verweist auf höchstens eine Entität.
      - Auf jede Entität verweist höchstens ein Bezeichner.
      - Ein Bezeichner verweist immer auf die gleiche Entität.
  - Benutzerfreundliche Namen
    - Auf die Benutzung von Menschen zugeschnitten

Benennung und Namenssysteme

- Lineare Benennung
  - Bezeichner sind zufällige Bit-Folgen (unstrukturierter / linearer Name)
  - Bezeichner enthält keine Info, wie Zugriffspunkt lokalisiert werden kann
  - Bsp.: MAC-Adresse

Lineare Benennung

- Verwendung von Broadcast
  - Nur für lokale Netzwerke praktikabel
  - Aufforderung an alle, wer den Bezeichner der Entität als Inhalt hat
  - Nur Rechner, die Zugriffspunkt für Entität bieten, melden sich zurück
  - Bsp.: Address Resolution Protocol (ARP)
  - (Welcher Rechner im Netzwerk besitzt eine bestimmte IP-Adresse?)

Lineare Benennung

- Verwendung von Multicast
  - Nur für lokale Netzwerke praktikabel, wenn Multicast-Unterstützung auf Netzwerkschicht (Sicherungsschicht) vorhanden
  - z.B. mobile Host: neben Zuweisung der IP, Eintrag in Multicast-Gruppe

Lineare Benennung

- Annahme:
  - Die Lokalisierung eines Dienstes ist bekannt.

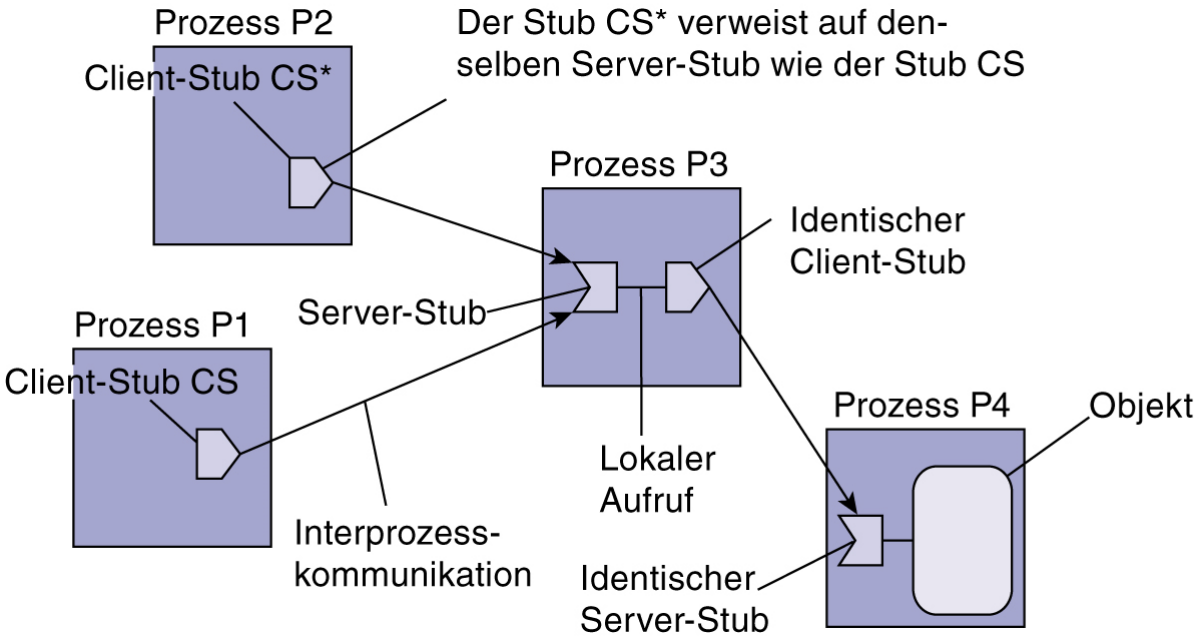
- Wie kann man damit umgehen, wenn dieser Dienst umzieht (ggf. mehrfach)?

Lineare Benennung

- Zeiger zur Weiterleitung
  - Ansatz zur Lokalisierung mobiler Entitäten
  - Bei Verschiebung einer Entität → Hinweis (Zeiger) auf neuen Ort wird hinterlassen
  - Client: Nach Lokalisierung der Entität → Folge den Zeigern

Frage:

Was sind die Nachteile dieses Vorgehens?

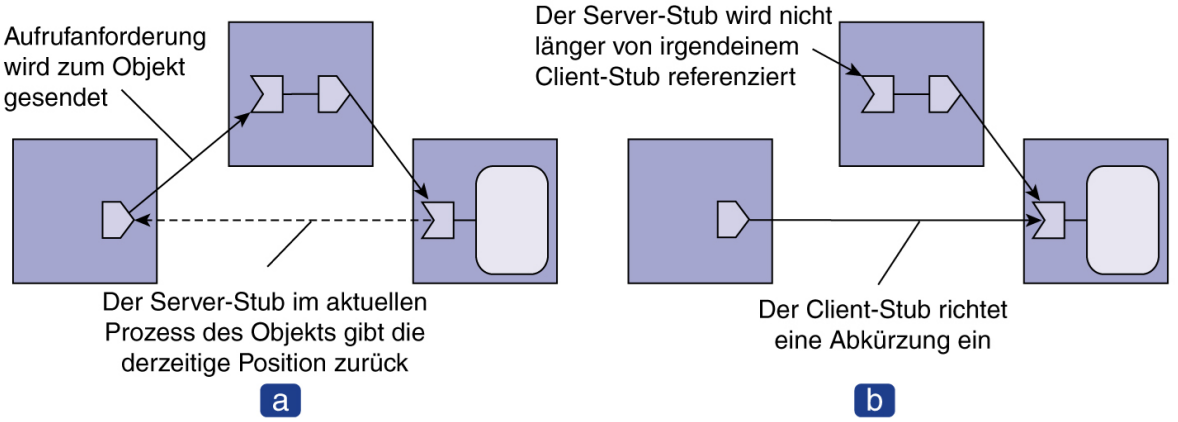


Lineare Benennung

- Zeiger zur Weiterleitung – Teil 2
  - Nachteile:
    - Kette kann sehr lang werden (bei hochgradig mobilen Entitäten)
    - Weiterleitung muss so lange wie möglich gehalten werden
    - Anfälligkeit für gebrochene Verknüpfungen
  - Zielsetzung:
    - Ketten kurz halten und Stabilität sicherstellen

Lineare Benennung

- Zeiger zur Weiterleitung – Teil 3
  - Lösungsmöglichkeit:
  - Neuer Standort wird zurück übermittelt
    - Direkt an auslösenden Client-Stub
    - Zurück über die Kette an den auslösenden Client-Stub
  - Schwierigkeit:
    - Was passiert mit den Weiterleitern?



Lineare Benennung

- Heimatgestützte Ansätze
  - Unterstützung bei mobilen Entitäten
  - Einführung des Heimatstandortes einer Entität
  - Bewegung der Entität
    - Care-of-Adresse wird beim Heimatagenten hinterlegt

Lineare Benennung

- Hintergrund: Peer-to-Peer-Netzwerke

- Wie ließen sich Entitäten in solchen Netzwerken finden?

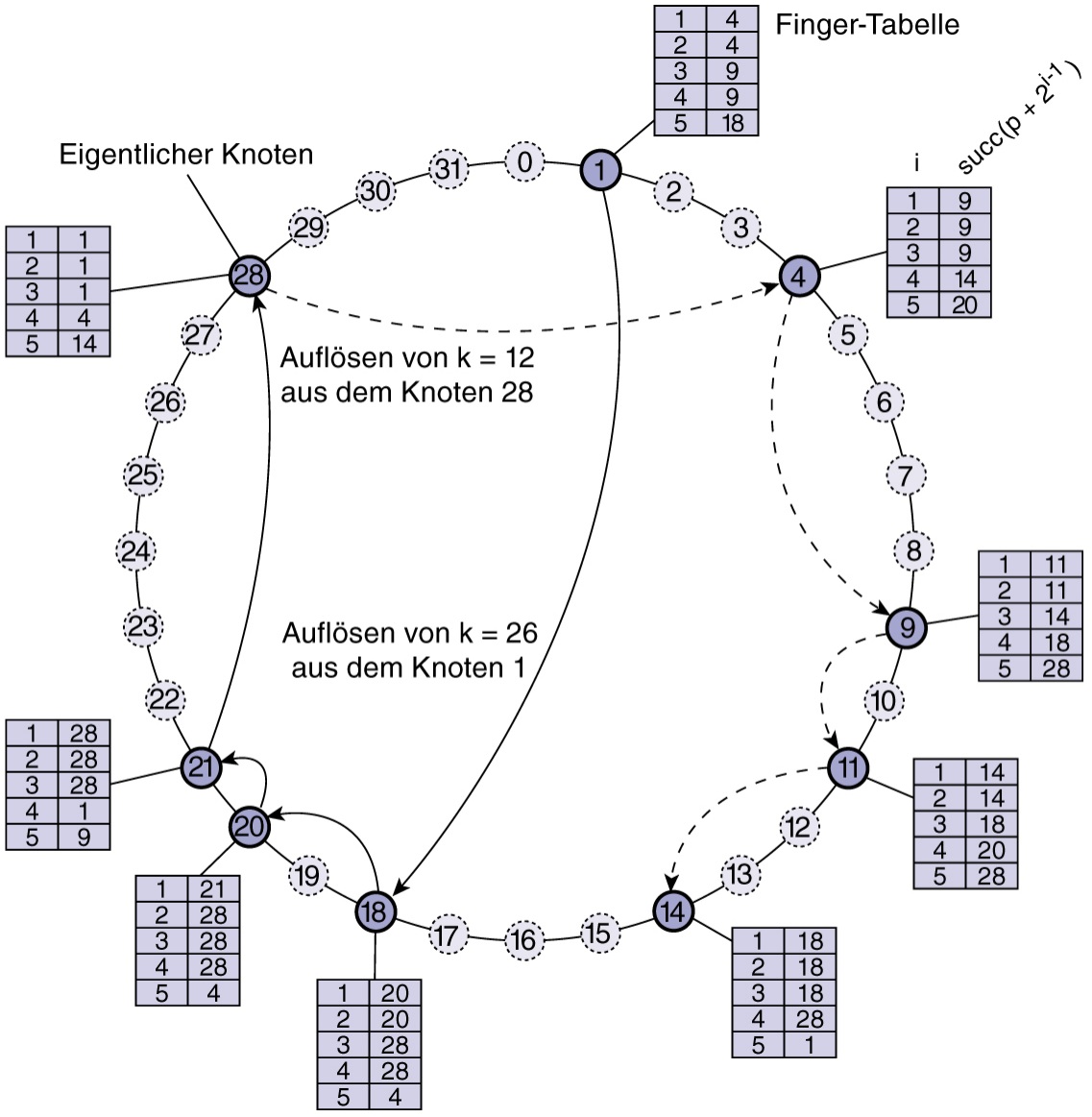
- Was könnte eine Zielsetzung hinsichtlich der Verteilung der Entitäten sein?

Lineare Benennung

- Verteilte Hashtabelle (Distributed Hash Table, DHT)
  - z.B. zwecks Speicherung des Speicherorts von Dateien
  - Prinzip:
    - Zuweisung der Datenobjekte von Schlüsseln (im linearen Wertebereich, 1 Datenobjekt = 1 Schlüssel)
    - Jeder Knoten ist zuständig für Teilbereichs des Schlüsselraums.
    - Zuständigkeiten können sich dynamisch ändern.
    - Routing zu einem zuständigen Knoten über Verantwortlichkeiten der Knoten für Teilbereiche möglich.

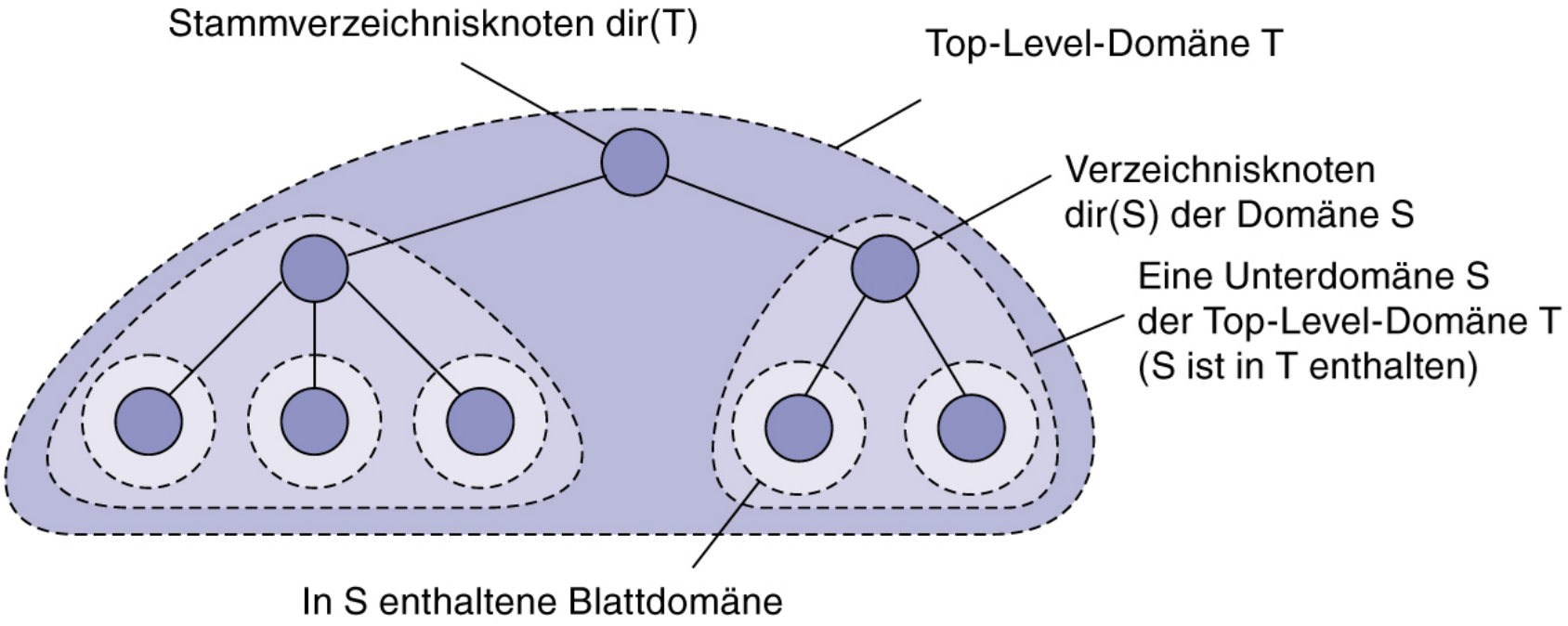
Lineare Benennung

- Beispiel für Verteilte
- Hashtabellen



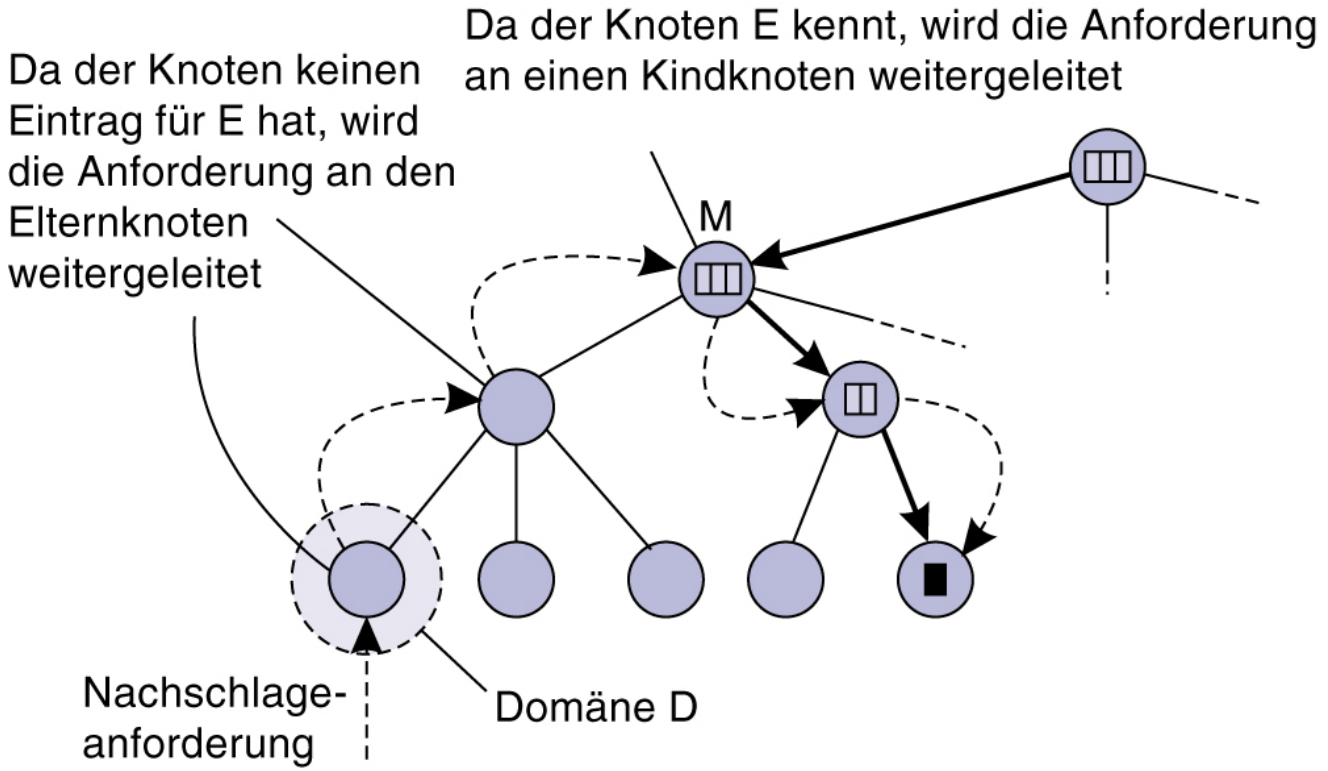
Lineare Benennung

- Hierarchische Ansätze
  - Standorteintrag einer Entität nur auf unterster Domäne
  - Darüber liegende Domänen haben zur Entität nur Referenz auf die tieferliegende Domäne



Lineare Benennung

- Hierarchische Ansätze
  - Nachschlagen eines Standorts



Themenüberblick

- **Benennung und Namenssysteme**
  - Begrifflichkeiten
  - Lineare Benennung
  - **Hierarchische Benennung**
  - (Attributbasierte Benennung)
- Synchronisierung

Hierarchische Benennung

- Nennen Sie mindesten 2 Beispiele für hierarchische Benennungsschemata!

- Was verstehen Sie unter einem Nameserver?
- Welche Anforderungen werden an Nameserver gestellt?

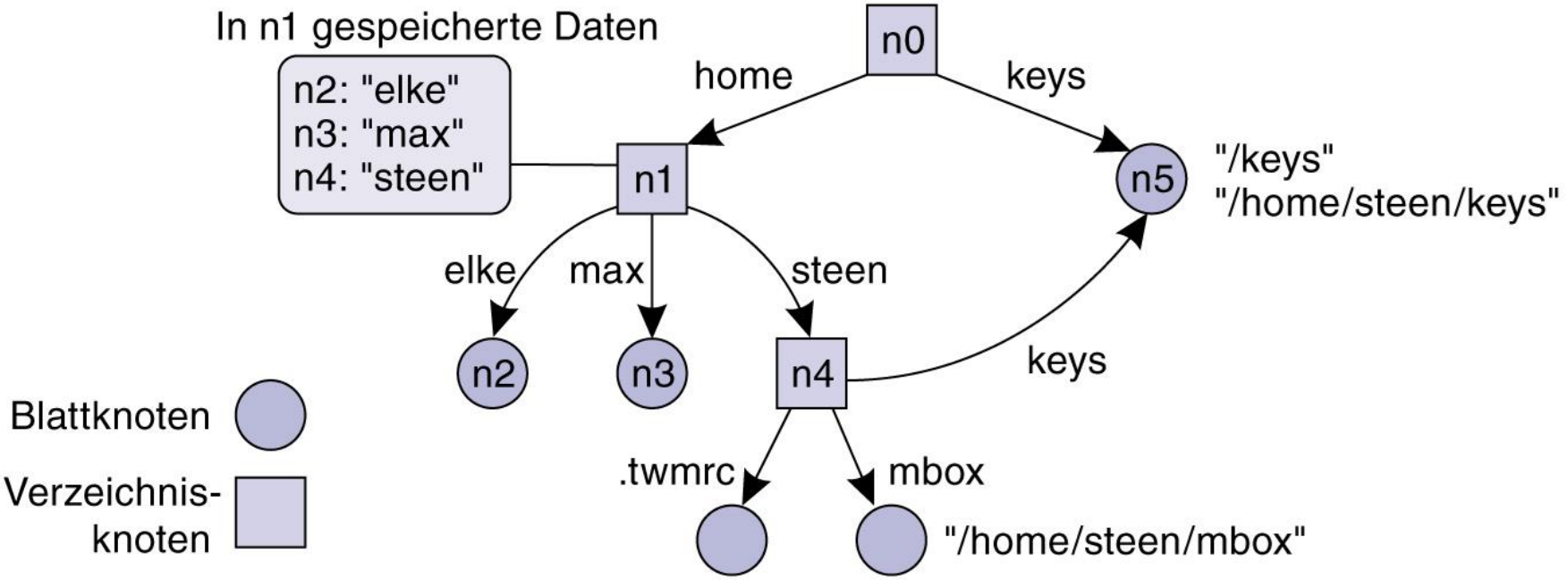
- Auf welche Weise kann man die Namensauflösung mit Hilfe von Nameservern realisieren?

Hierarchische Benennung

- Hierarchische Namen → beschriftete, gerichtete Graphen mit
  - Blattknoten
  - Verzeichnisknoten
- Blattknoten
  - Keine ausgehenden Kanten
  - Steht für eine Entität und enthält allg. Infos (z.B. Adresse)
- Verzeichnisknoten
  - Besitzt gewisse Anzahl ausgehender Kanten mit Namen
    - Ausgehende Kanten in einer Tabelle
  - Verzeichnisknoten ebenso Entität mit Bezeichner

Hierarchische Benennung

- Beispiel eines allgemeinen Namensgraphen mit einem einzelnen Wurzelknoten



Hierarchische Benennung

- Nutzung von Pfadnamen
  - „absoluter Pfad“, wenn erster Knoten im Pfandnamen auf Wurzel referenziert
  - „relativer Pfad“, andernfalls (ausgehend vom aktuellen Ort)
- Pfadnamen als Zeichenkette mit Trennzeichen (z.B. Schrägstrich oder …, z.B. /root/home/….)
  - Aber: Knoten kann durch verschiedene Pfadnamen ausgedrückt werden (siehe Hard Links, Symbolic Links)

Hierarchische Benennung

- Namensauflösung:
- N: < Beschriftung_1 , Beschriftung_2 , … , Beschriftung_n>
  - Start bei Knoten N und Bezeichner für Beschriftung_1 suchen
  - Im Verzeichnis zu „Beschriftung_1“ nach Beschriftung_2 suchen
  - ...

Hierarchische Benennung

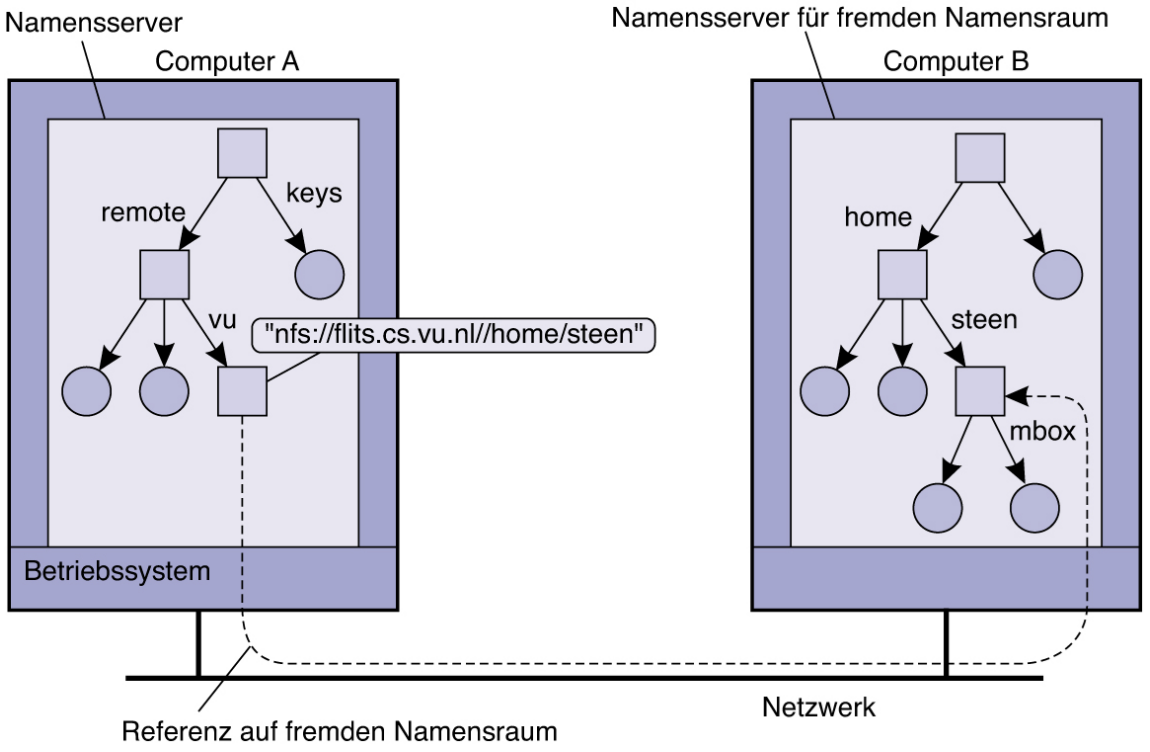
- Mounting unter Unix/Linux
  - Virtuelles Dateisystem (VFS - virtual file system)
  - Systemaufrufe um Unix- und Network-Dateisysteme anzusprechen
  - Idee: Wurzelverzeichnis eines Dateisystems wird an eine Stelle des Virtuellen Dateisystems gemountet
    - z.B. CD-ROM unter /media/cdrom
  - Nutzer -> Standard POSIX Systemaufrufe an das VFS
  - VFS greift über spezielle Funktionen auf jeweiliges Dateisystem zu

Hierarchische Benennung

- Mounting in Verteilten Systemen
  - Network File System (NFS)
  - Notwendigkeit spezieller Infos
    - Name des Zugriffsprotokolls
    - Servername
    - Name des Mountpoints im fremden Namensraum

Hierarchische Benennung

- Beispiel für Mounting in Verteilten Systemen



Hierarchische Benennung

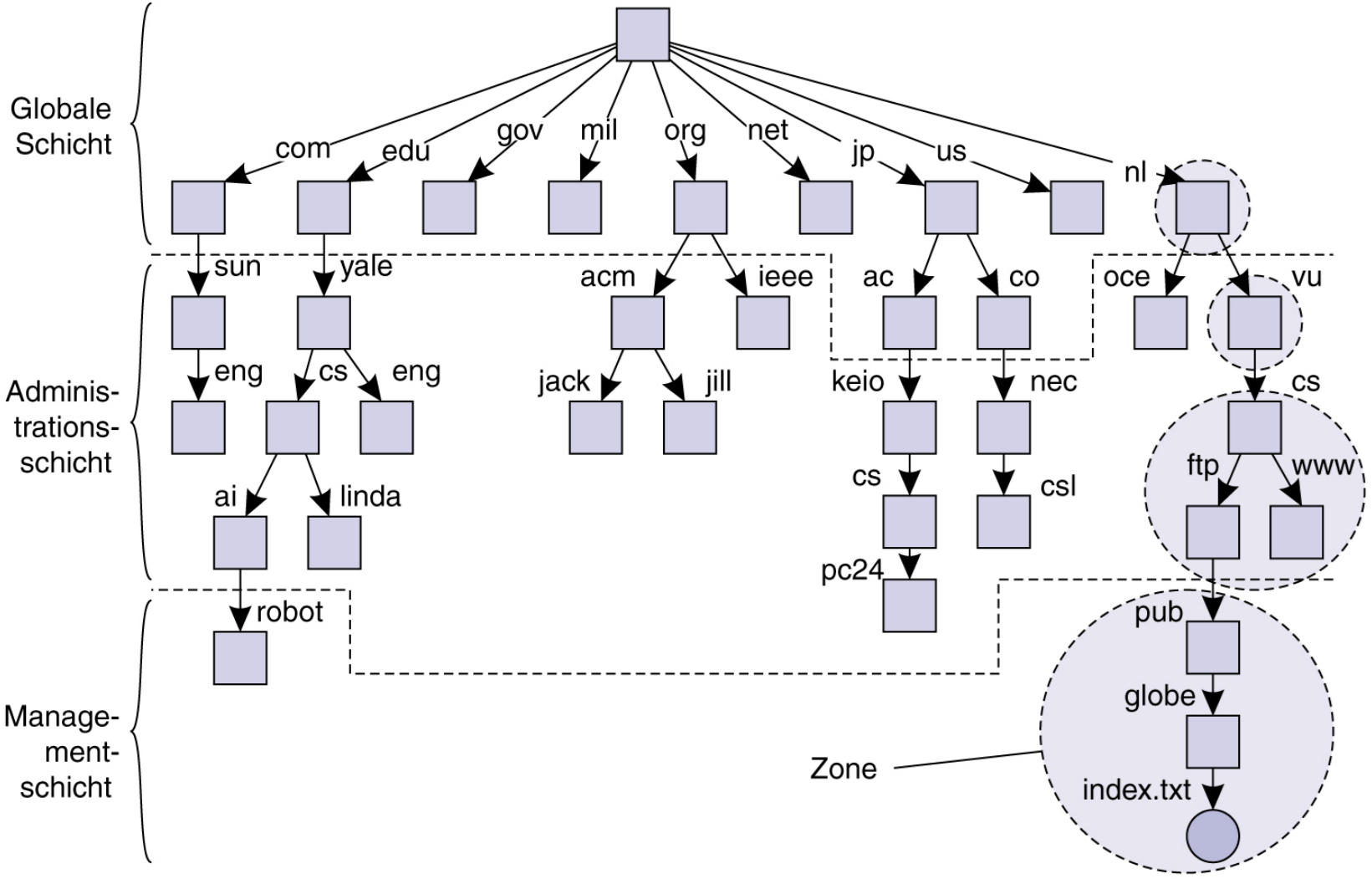
- Implementierung eines Namensraumes
  - Globale Schicht
    - Knoten der höchsten Ebene, Wurzelknoten und seine Kindknoten
    - Verzeichnistabellen ändern sich selten
    - Abbildung von Organisationen oder Gruppen von Organisationen
  - Administratorenschicht
    - Verwaltungsknoten innerhalb einer Organisation
    - Relative Stabilität, aber mehr Änderungen als in globaler Schicht

Hierarchische Benennung

- Implementierung eines Namensraumes
  - Managementschicht
    - Knoten mit regelmäßigen Änderungen
    - Host im lokalen Netzwerk oder gemeinsam genutzte Dateien

Hierarchische Benennung

- Implementierung des DNS-Namensraumes



Hierarchische Benennung

- Anforderungen an die Nameserver

Aspekt	Globale Schicht	Administrations-schicht	Management-schicht
Geografische Ausdehnung des Netzwerkes	Weltweit	Organisation	Abteilung
Knotengesamtzahl	Wenige	Viele	Zahllose
Reaktionsfähigkeit auf Nachschlageanforderungen	Sekunden	Millisekunden	Sofort
Verbreitung von Aktualisierungen	Verzögert	Sofort	Sofort
Anzahl Replikate	Viele	Keine oder wenige	Keine
Caching durch Clients	Ja	Ja	Manchmal

Hierarchische Benennung

- Anforderungen an die Nameserver
  - Globale Schicht
    - Hohe Verfügbarkeit, ansonsten Nichterreichbarkeit eines Teilnetzes
    - Antwortverhalten weniger zeitkritisch, da meistens Caching (bei den Clients)
    - Verwendung vieler Replikate von Servern
      - Verbreitung von Aktualisierungen kann verzögert sein. („Fehlt ein Eintrag, dann frag den Chef.“)

Hierarchische Benennung

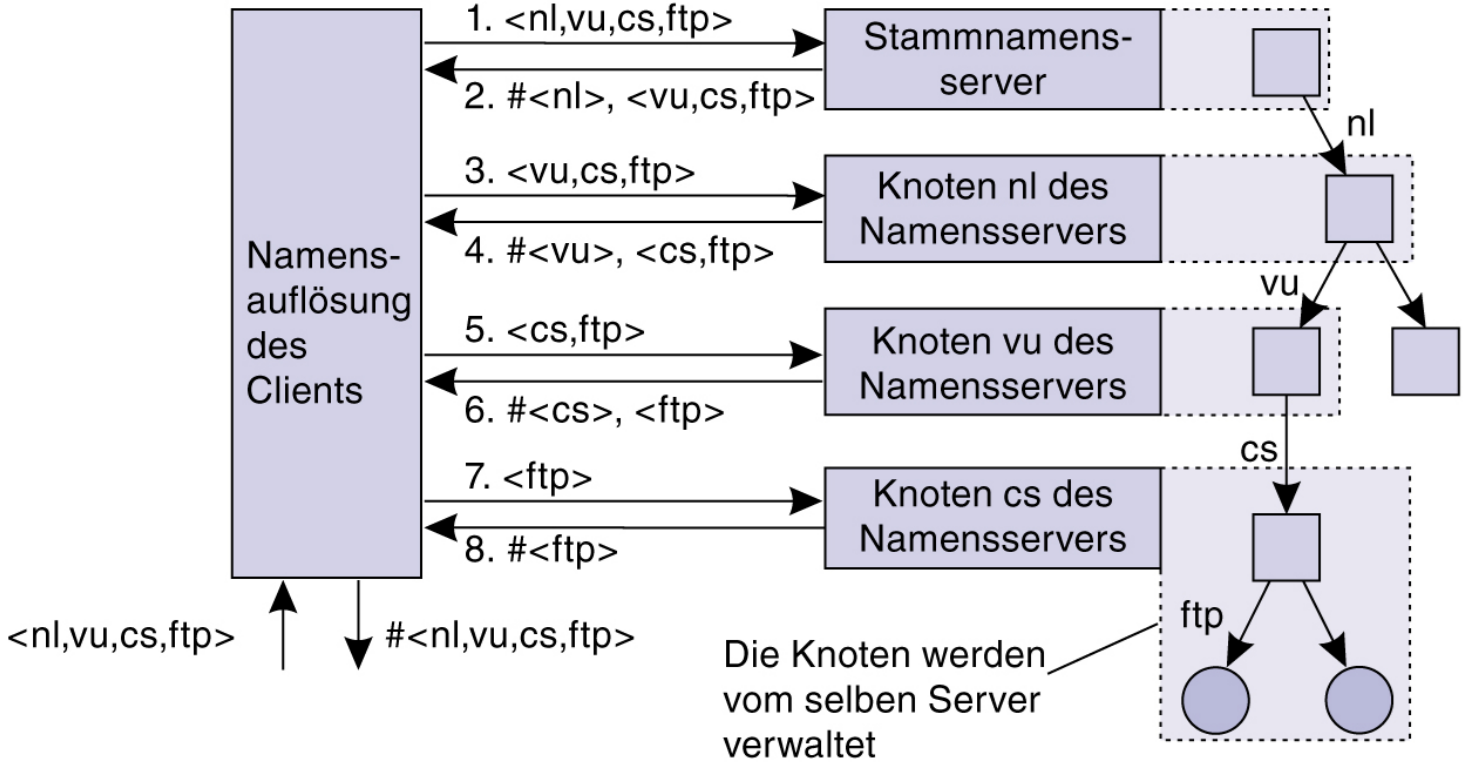
- Anforderungen an die Nameserver
  - Administrationsschicht
    - Verfügbarkeit für Clients der Organisation
    - Schnelle Rückgabe bei Nachschlageergebnisse
    - Verwendung von Caching und Replikaten
  - Managementschicht
    - Verfügbarkeit unkritisch, aber schnelle Antwortzeiten und sofortige Verbreitung von Aktualisierungen

Hierarchische Benennung

- Implementierung der Namensauflösung
  - Annahme: keine Replikate und kein Caching
  - Ansätze:
    - Iterative Namensauflösung
    - Rekursive Namensauflösung

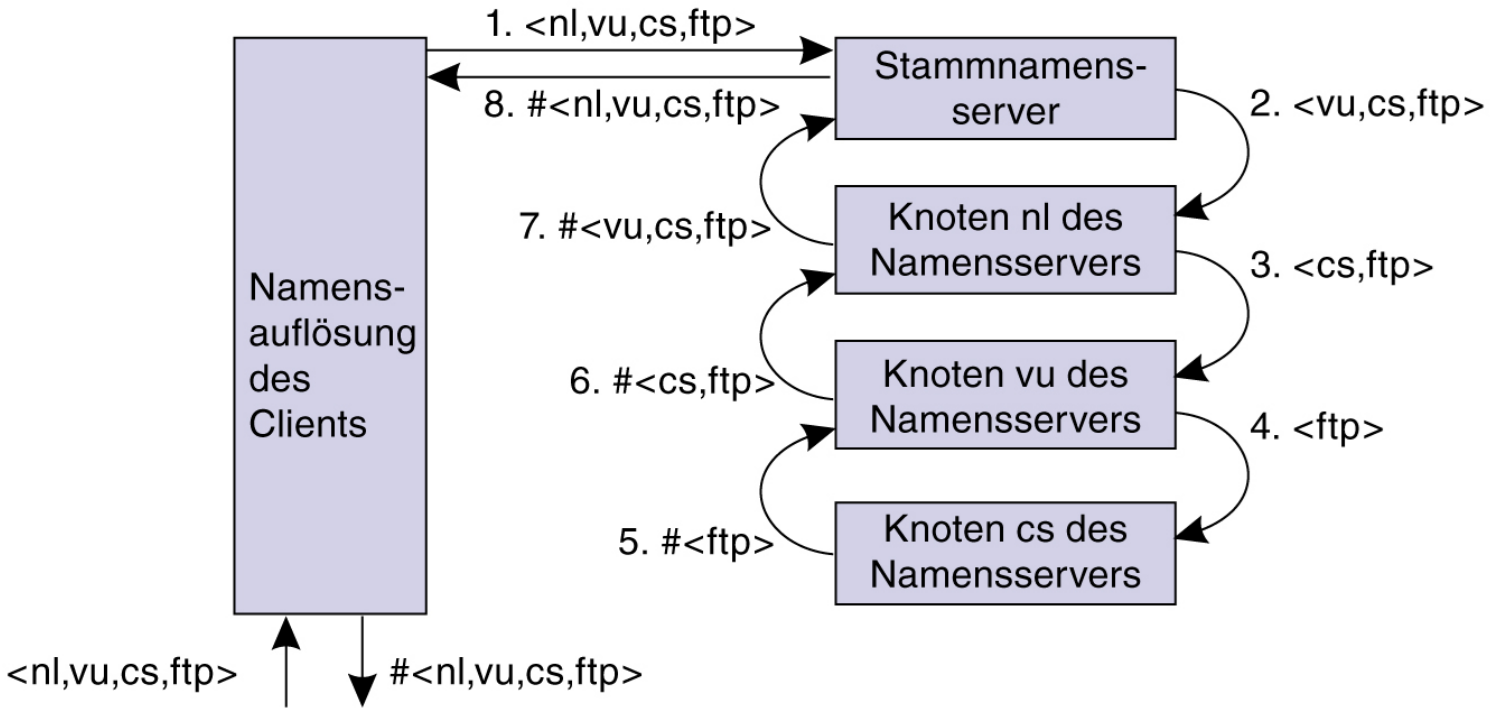
Hierarchische Benennung

- Implementierung der Namensauflösung
  - Iterative Namensauflösung - Beispiel:
    - Auflösung von ftp://ftp.cs.vu.nl/pub/globe/index.html
    - bzw. root:<nl, vu, cs, ftp, pub, globe, index.html>



Hierarchische Benennung

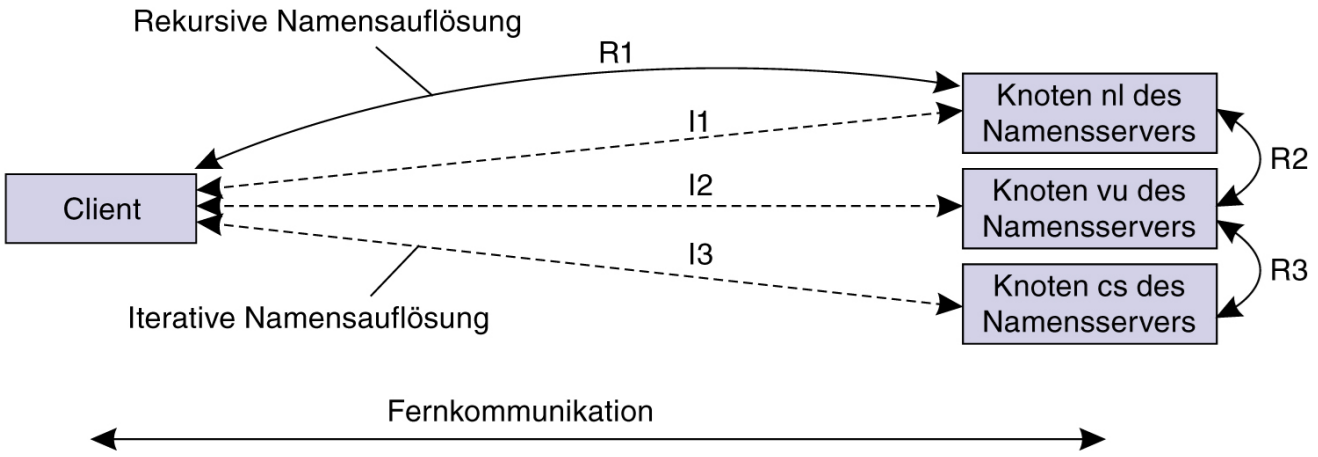
- Implementierung der Namensauflösung
  - Rekursive Namensauflösung - Beispiel:
    - Auflösung von ftp://ftp.cs.vu.nl/pub/globe/index.html
    - bzw. root:<nl, vu, cs, ftp, pub, globe, index.html>



Hierarchische Benennung

- Implementierung der Namensauflösung
  - Nachtteil der rekursiven Namensauflösung
    - Größere Anforderungen an Leistung jedes Nameservers
    - Resultat: Server der globalen Schicht meist nur iterativen Ansatz
  - Vorteile der rekursiven Namensauflösung
    - Leistungssteigerung durch Caching
    - Senkung der Kommunikationskosten
  - Optimierung bei iterativen Namensauflösung
    - Nutzung eines lokalen zwischengeschalteten Nameserver

Hierarchische Benennung



Beispiel hinsichtlich
der Kommunikationskosten:

Themenüberblick

- **Benennung und Namenssysteme**
  - Begrifflichkeiten
  - Lineare Benennung
  - Hierarchische Benennung
  - **(Attributbasierte Benennung)**
- Synchronisierung

Attributbasierte Benennung Konzept

- Kernziele der linearen und hierarchischen Benennung:
  - Standortunabhängigkeit und Benutzerfreundlichkeit

- Attributbasierte Benennung:
  - Suche einer Entität aufgrund Beschreibung
  - Hinterlegung von (Attribut, Wert)-Paaren je Entität
  - Aufgabe
    - Anfrage: Beschreibung eines Benutzers
    - Rückgabe: eine oder mehrere passende Entitäten

Themenüberblick

- Benennung und Namenssysteme
- **Synchronisierung**
  - **Uhrensynchronisierung**
  - Logische Uhren

Uhrensynchronisierung

- Fragen:
  - Warum ist Uhrensynchronisierung wichtig in verteilten Systemen?
  - Wie findet eigentlich Zeitmessung heute statt?
  - Welche Schwierigkeiten könnte es bei der Uhrensynchronisierung in verteilten Systemen geben?
  - Welche Möglichkeiten / Ansätze bestehen Uhren mehrerer Rechner zu synchronisieren?

Uhrensynchronisierung

- Wichtigkeit exakter Zeitmessung
  - Aktienhandel
  - Sicherheitsprüfungen
  - Messungen
  - …
- Wichtigkeit der Uhrensynchronisierung
  - Reihenfolge von Ereignissen, z.B.
    - Einzahlung von 100€, danach Zinsaufschlag von 10%
    - Zinsaufschlag von 10%, danach Einzahlung der 100€

Uhrensynchronisierung

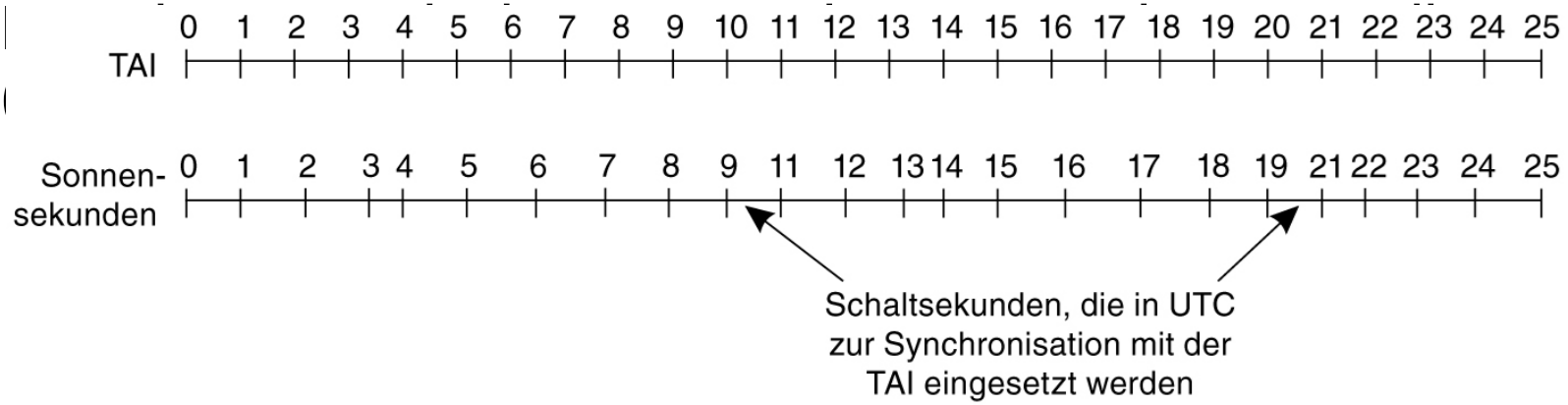
- Problematik der Zeitmessung
  - Mit der Einführung mechanischer Uhren im 17ten Jahrhundert → astronomische Zeitmessung
  - Meriadiandurchgang der Sonne = scheinbar höchste Punkt der Sonne am Himmel
  - Zeitliche Distanz zw. 2 Meridiandurchgängen = 1 Sonnentag
  - => 1 Sonnensekunde = 1/86400 Sonnentag

Uhrensynchronisierung

- Problematik der Zeitmessung
  - Achtung: Der Sonnentag ist nicht konstant!!
    - Vor 300 Mio Jahren -> 1 Jahr entspricht ungefähr 400 Tage
      - Grund: Gezeitenreibung
    - Kurzfriste Schwankungen ebenso möglich
      - Grund: Turbulenzen im Inneren des Erdkerns

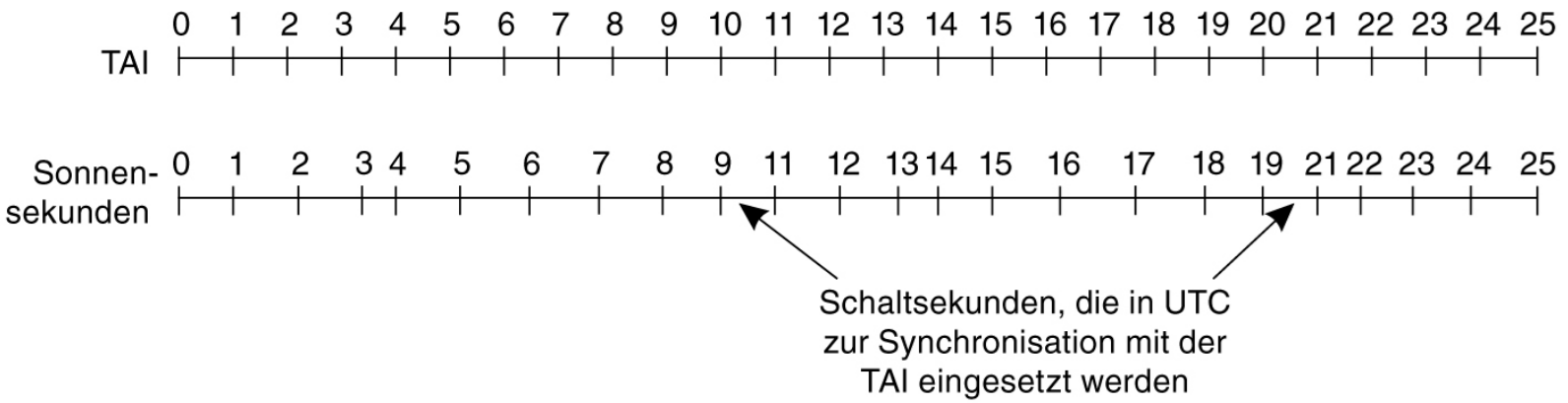
Uhrensynchronisierung

- Zeitmessung
  - Atomuhr seit 1948
    - 9.192.631.770 Übergänge des Caesium-Atoms = mittlere Sonnensekunde im Einführungsjahr
  - Einführung der Internationalen Atomzeit TAI (Temps Atomique International)
  - Problematik: TAI ist konstant, Sonnensekunden nicht
  -



(Für Interessierte bzgl. Schaltsekunden: https://www.timeanddate.de/zeitzonen/schaltsekunden)

Uhrensynchronisierung



(Für Interessierte bzgl. Schaltsekunden: https://www.timeanddate.de/zeitzonen/schaltsekunden)

Uhrensynchronisierung

- Verfügbarkeit der UTC in Deutschland Über Langwellensender der Physikalisch-Technischen Bundesanstalt in Braunschweig (PTB-Zeit, siehe https://www.ptb.de/)

- Problemstellung 1:
  - 1 Rechner besitzt Empfänger für UTC-Zeit
  - Lokale Uhren auf Rechner haben Drift (aufgrund Ungenauigkeiten)
  - Aufgabe: Aller anderen Rechner sollen dazu synchron gehalten werden.

Uhrensynchronisierung

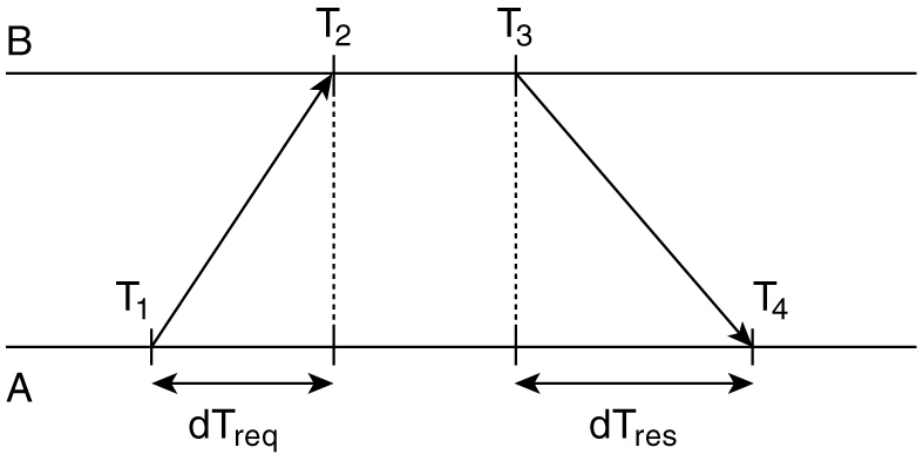
- Verfügbarkeit der UTC in Deutschland Über Langwellensender der Physikalisch-Technischen Bundesanstalt in Braunschweig (PTB-Zeit, siehe https://www.ptb.de/)

- Problemstellung 2:
  - Kein Rechner besitzt Empfänger für UTC-Zeit
  - Lokale Uhren auf Rechner haben Drift (aufgrund Ungenauigkeiten)
  - Aufgabe: Abweichungen kompensieren, Zeitdrift entgegenwirken

Uhrensynchronisierung

- **Network Time Protocol (NTP)**
  - A sendet Zeit an B
  - B antwortet und übermittelt
  - Zeiten
  - Übertragungszeiten

    $\delta = (dT_{req} + dT_{res}) / 2$



  - Abweichungsberechnung: $\theta = T_3 + \delta - T_4$
  - Messung: 8 Wertepaare (θ,δ)
    - kleinster Wert für δ als Schätzung für Verzögerung
    - Zugehöriges θ als zuverlässigste Schätzung der Abweichung

Uhrensynchronisierung

- **Network Time Protocol (NTP)** – Teil 2
  - Probleme 1: $\theta < 0$ (Uhr von A geht zu schnell)
    - Zurücksetzung der Zeit nicht erlaubt (Konsistenzprobleme auf A)
    - „Verlangsamen“ der Uhr auf A (z.B. statt je Timeinterrupt 10ms hinzuzufügen, nun nur +9ms)
      - Hinweis: Analoges Vorgehen, wenn A zu langsam geht

Uhrensynchronisierung

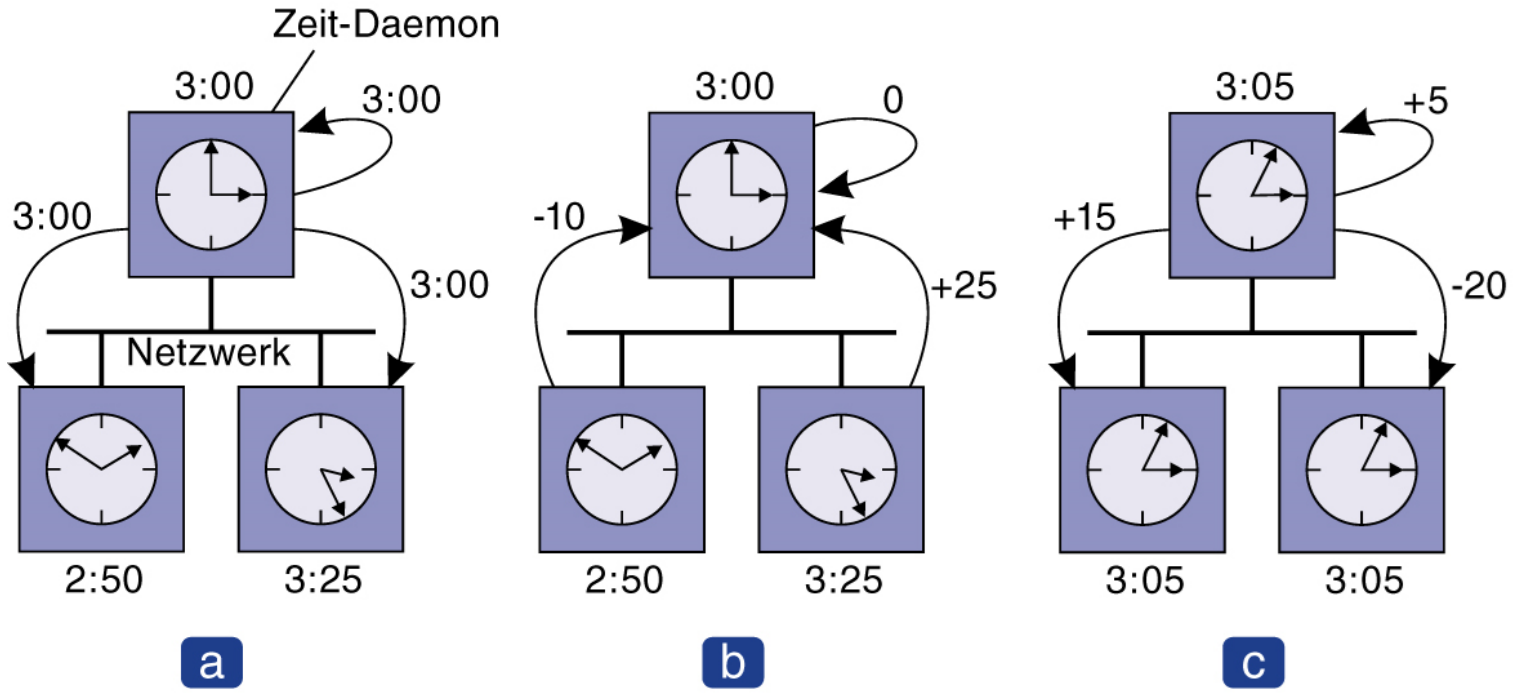
- **Network Time Protocol (NTP)** – Teil 2
  - Probleme 2: Welche Uhr ist genauer?
    - Einteilung der Server in Ebenen (Strata)
    - Stratum 0 ist Refernzuhr (z.B. PTB-Zeit)
    - Stratum 1 ist Empfänger der UTC-Zeit
    - Server, der mit Stratum-n-Server synchronisiert, wird Stratum-(n+1)-Server

Uhrensynchronisierung

- **Berkley-Algorithmus**
  - Aktiver Zeitserver (Zeit-Daemon) ohne UTC-Empfang
  - Anfrage vom Zeitserver an alle Rechner nach deren Systemzeit
  - Aus Antworten: Berechnung einer durchschnittlichen Zeit
  - Mitteilung an jeden Rechner hinsichtlich Verlangsamung oder Beschleunigung zwecks Synchronisierung
  - Ziel: Einigung aller Rechner auf selbe Zeit

Uhrensynchronisierung

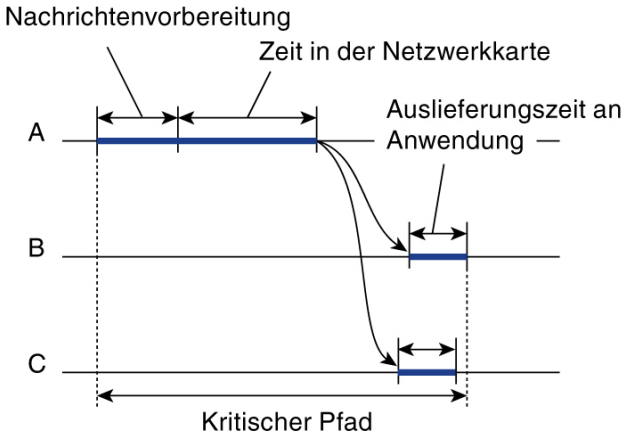
- **Berkley-Algorithmus** – Teil 2



(a) Der Zeit-Daemon fragt alle anderen Rechner nach ihren Uhrzeiten.
(b) Die Rechner antworten.
(c) Der Zeit-Daemon teilt allen mit, wie sie ihre Uhren einzustellen haben.

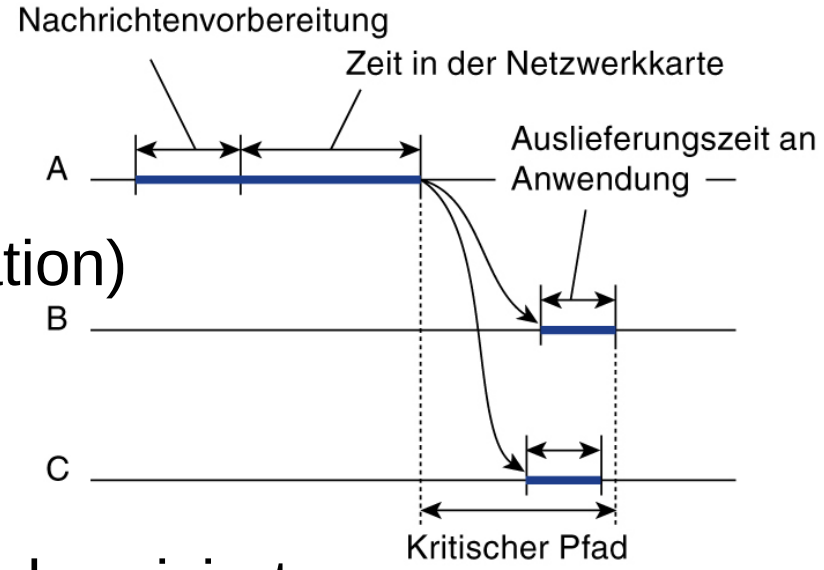
Uhrensynchronisierung

- Uhrzeitsynchronisierung in drahtlosen Netzwerken
  - Bsp.: Sensornetzwerk
  - Problematik: Verwendung von Konkurrenzprotokollen
  - Zeitzusammensetzung zwecks Austausch
    - Nachrichtenvorbereitung
    - Zeit in der Netzwerkkarte
    - Übertragungszeit
    - Auslieferungszeit an die Anwendung



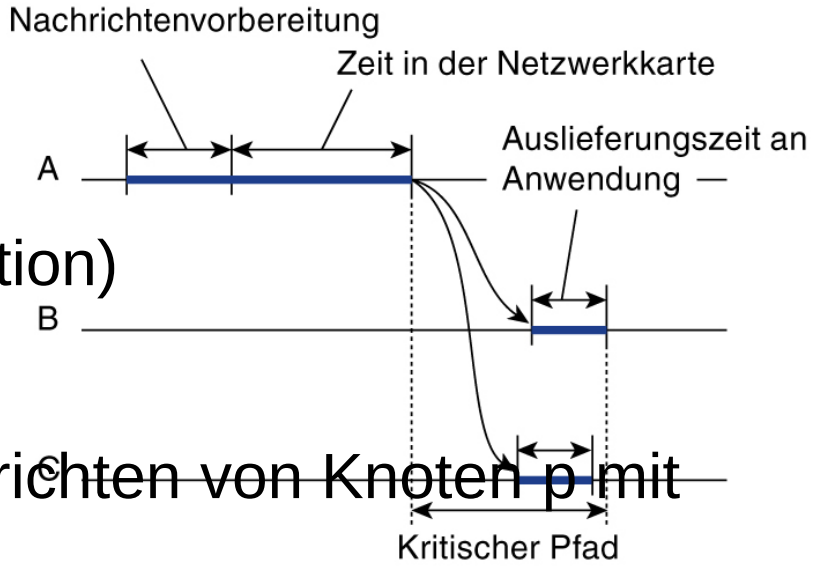
  - Hinweise:
    - Übertragungszeit annähernd konstant (wenn keine Hops)
    - Zeit in Netzwerkkarte mit größten Schwankungen

Uhrensynchronisierung



- RBS (Reference Broadcast: Synchronization)
  - Anwendung für Synchronisierung
  - in drahtlosen Netzwerken
  - Besonderheit: Sender wird nicht synchronisiert
  - Nachricht an Empfänger, wobei Zeitmessung beim Verlassen der Netzwerkkarte beginnt
  - Unsicherheitsfaktoren seitens des Senders beseitigt

Uhrensynchronisierung



- RBS (Reference Broadcast: Synchronization)
  - Idee:
    - Aussenden von m-Referenznachrichten von Knoten p mit Zeit $T_{p,m}$
    - Empfang am Konten q und Vergleich der Zeiten $T_{p,k}$-$T_{q,k}$
    - Mittelung der Abweichung → Abspeichern der Abweichung anstatt der Anpassung der Uhrengeschwindigkeit
    - Verbesserung unter Berücksichtigung der Drift:

      $$Offset_{p,q}(t) = a\ t + b$$

    - (lineare Approximation der Drift)

Themenüberblick

- Benennung und Namenssysteme
- **Synchronisierung**
  - Uhrensynchronisierung
  - **Logische Uhren**

Logische Uhren

- Beobachtung
  - Uhrzeitsynchronisierung zwar möglich, aber kein absolutes Muss
  - Bei nicht-wechselwirkenden Prozessen: fehlende Synchronisierung nicht beobachtbar und kein Resultat von Fehlern
  - Aber wichtig: Einigung auf eine Reihenfolge von Ereignissen

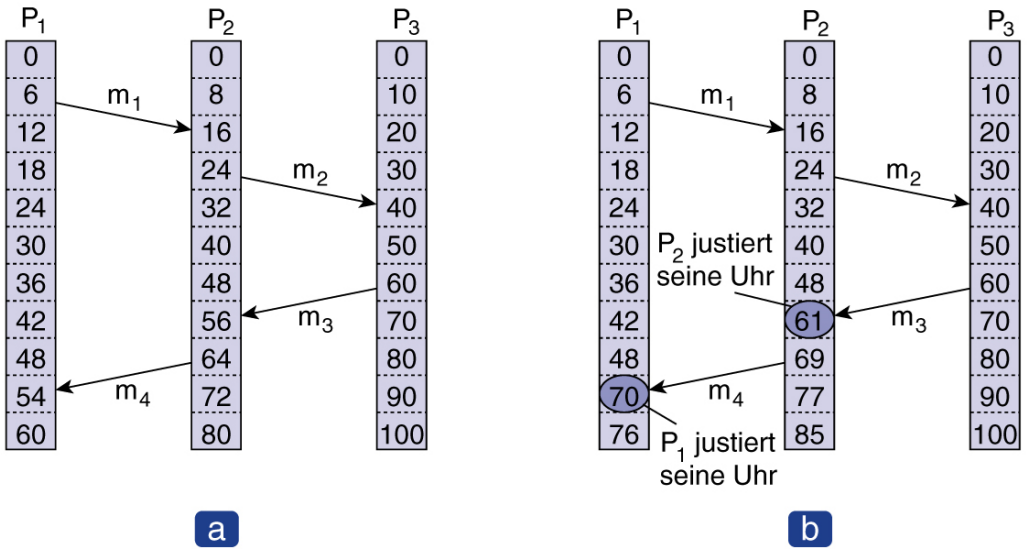
- Frage: Wie kann man die Reihenfolge von Ereignissen in verteilten Systemen sicherstellen?

Logische Uhren

- **Lamport-Zeit / Logische Uhr von Lamport**
  - Bezeichnung: a→b
  - Bedeutung: „a passiert vor b“ („*happens-before*“)
  - Beobachtung:
    - Im selben Prozess passiert a vor b, dann a→b
    - a entspricht Senden einer Nachricht, b entspricht Empfang einer Nachricht, dann a→b
  - Hinweis: Happens-before-Relation ist transitiv
  - Wenn a→b und b→c, dann gilt auch a→c
  - Einführung eines Zeitwertes C(a),
  - wenn a→b, dann C(a) < C(b)

Logische Uhren

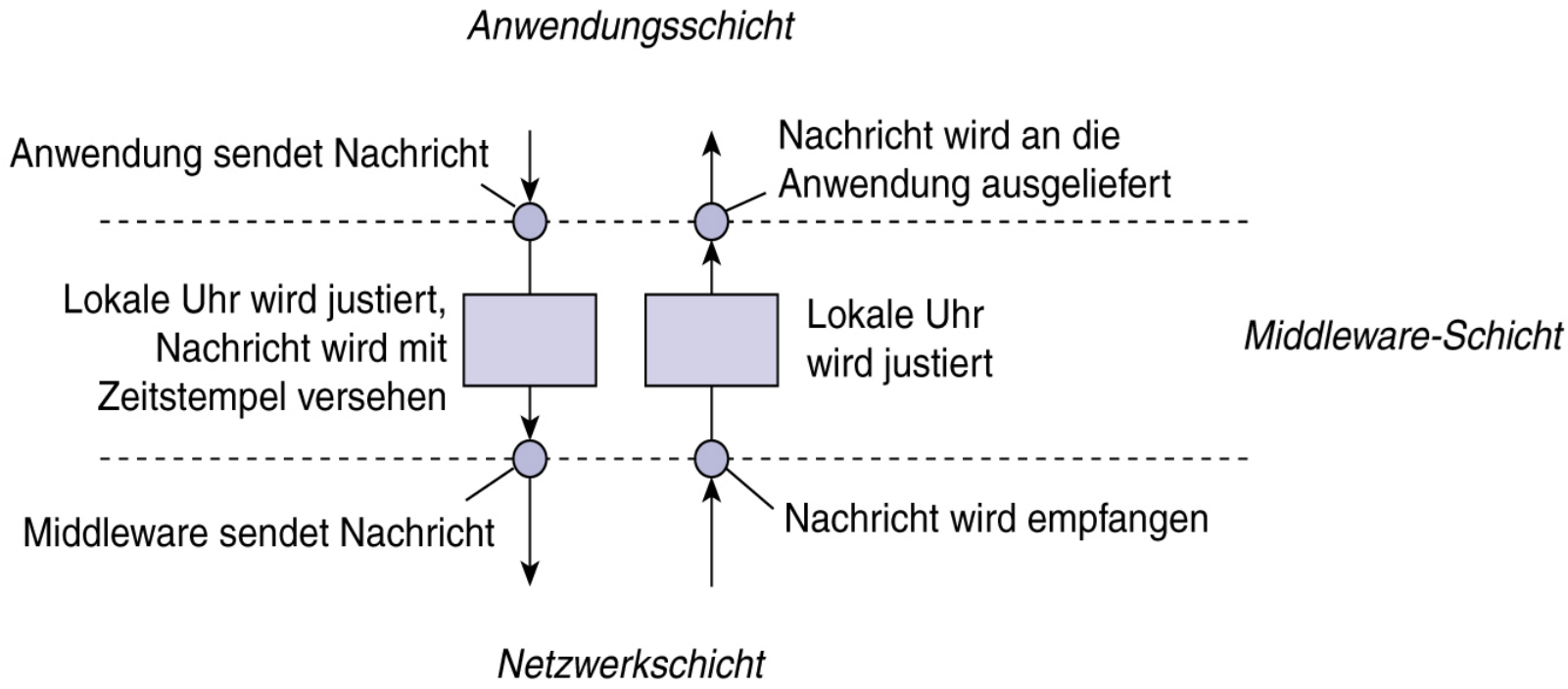
- **Lamport-Zeit / Logische Uhr von Lamport** – Teil 2
  - Bedingungen: Ist a ein Ereignis in Prozess P_i und b ein Ereignis in Prozess P_j, dann gilt a→b wenn
  - (i) $C_i(a) < C_j(b)$ oder
  - (ii) $C_i(a) = C_j(b)$ und $P_i < P_j$
  - (Berücksichtigung der
  - Prozessnummern)
  - Prinzip: Anpassung der
  - jeweiligen Uhren



(a) Drei Prozesse, von denen jeder eine eigene Uhr hat. Die Uhren laufen mit unterschiedlichen Geschwindigkeiten.
(b) Der Algorithmus von Lamport korrigiert diese Uhren.

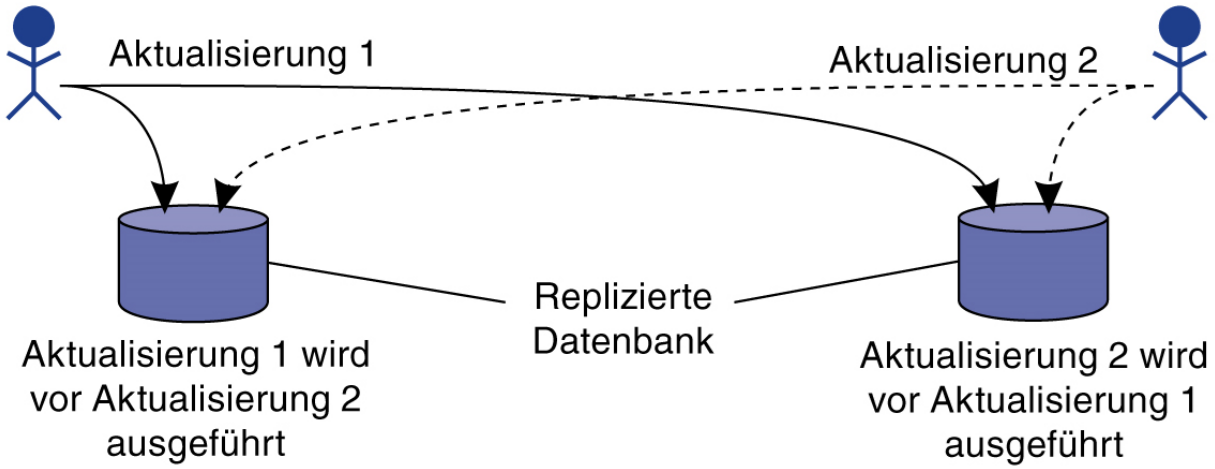
Logische Uhren

- **Lamport-Zeit / Logische Uhr von Lamport** – Teil 3
  - Lokalisierung der logischen Uhren von Lamport in der Middleware-Schicht



Logische Uhren

- Anwendung: **Vollständig geordnetes Multicasting**
- Beispiel: Kontoguthaben 1000€
  - Aktualisierung 1: Einzahlung von 100€
  - Aktualisierung 2: Gutschrift von Zinsen zu 1%
- Problem: Reihenfolge muss auf allen Datenbanken gleich sein



Logische Uhren

- Anwendung: **Vollständig geordnetes Multicasting**
- Annahmen:
  - Keine Nachrichten gehen verloren
  - Eintreffende Nachrichten kommen auch in der Sendereihenfolge eines Sender an

Logische Uhren

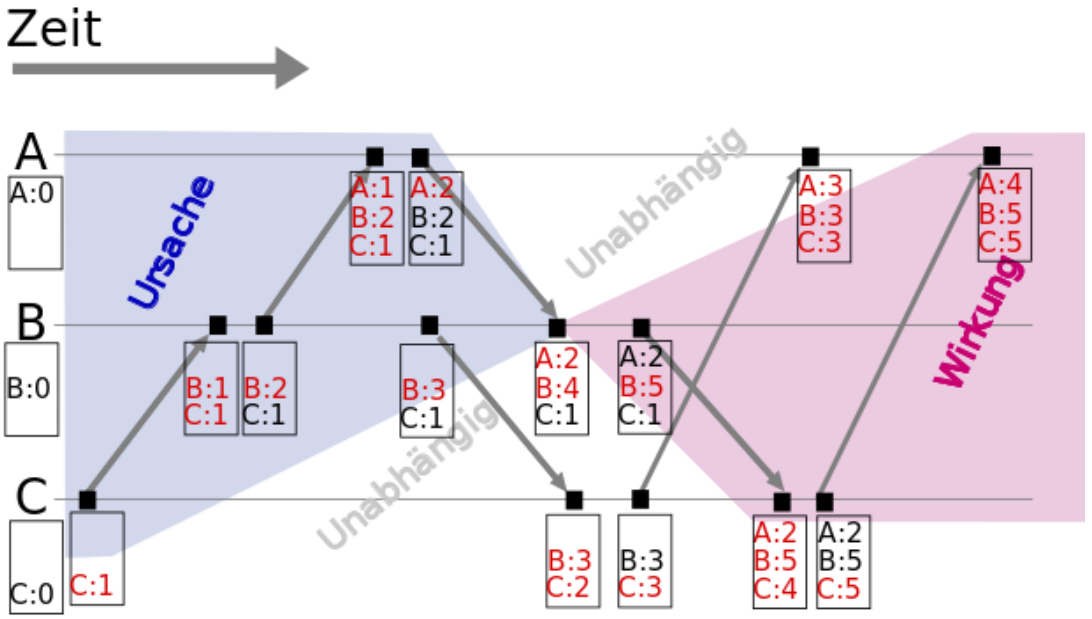
- Anwendung: **Vollständig geordnetes Multicasting**
- Lösung:
  - Jede Nachricht mit aktuellem (logischen) Zeitstempel des Senders versehen an alle Knoten (inklusive sich selber)
  - Empfänger
    - Einsortierung in Warteschlange gemäß dem Zeitstempel
    - Senden einer Bestätigung via Multicast
  - Resultat: Alle Prozesse haben gleiche Kopie der lokalen Warteschlange.

Logische Uhren

- **Vektoruhren**
  - Bei logischer Uhr von Lamport → Vollständige Ordnung (gemäß der Bedingungen)
  - Nachteil bei Lamport: Keine Aussage über Kausalität
  - Ziel:
    - Kausalzuordnung möglich
    - Nebenläufigkeit von Ereignissen ermittelbar

Logische Uhren

- **Vektoruhren** – Teil 2
  - Idee: Vektor mit Zeit für jeden Prozess wird geführt
  - Aktualisierung der eigenen Zeit
  - Versenden des gesamten Vektors
  - Anwendbarkeit:
  - Monitor- und
  - Debugsysteme



(Quelle: https://de.wikipedia.org/wiki/Vektoruhr#/media/Datei:Vektoruhren.svg)