Verteilte Systeme

Namensdienste und Internet Standardanwendungen

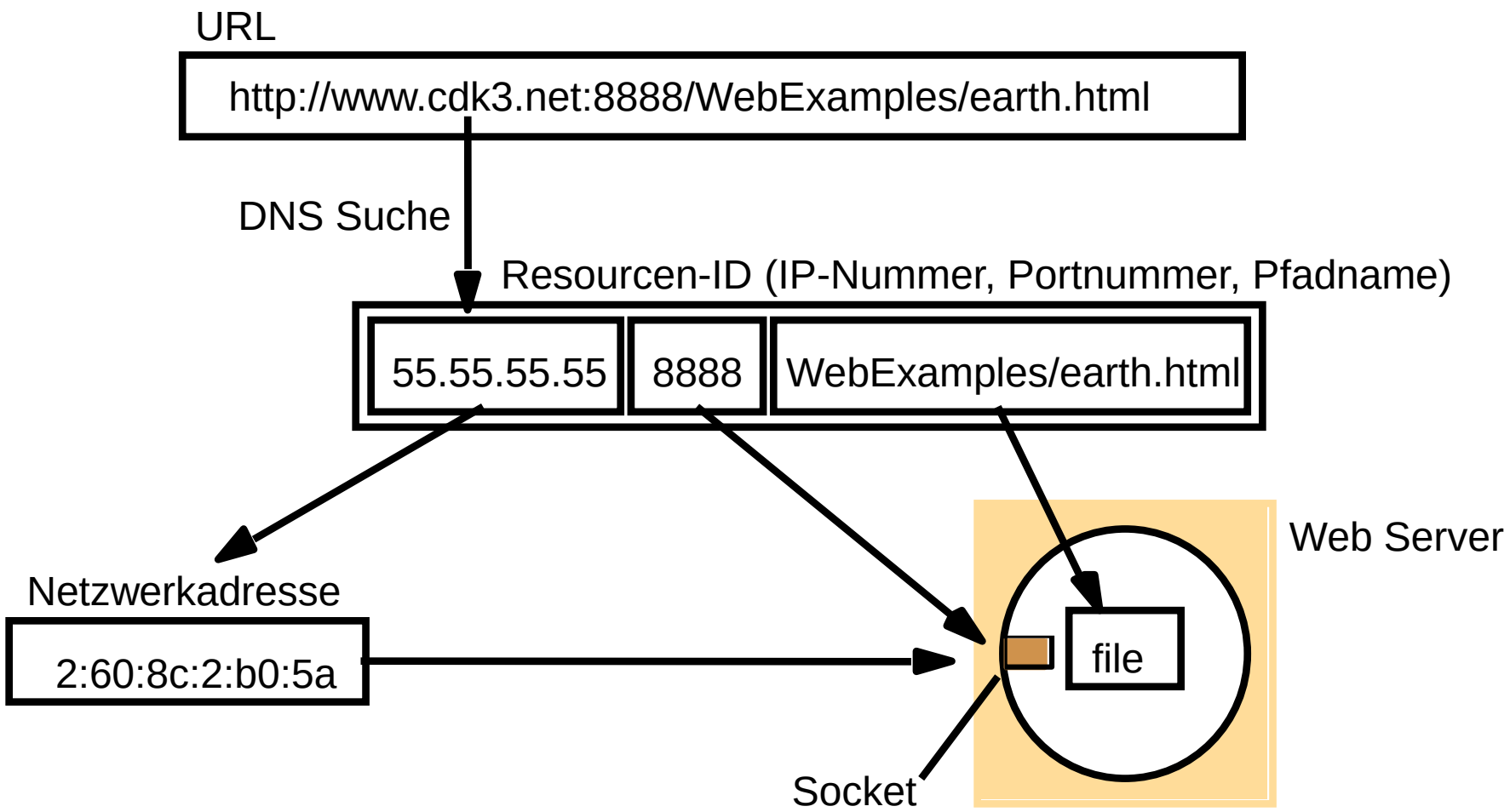
Namen in verteilten Systemen

- Namen dienen der **(eindeutigen) Bezeichnung** von Objekten (Bezeichner)
  - Variablen, Prozeduren, Datentypen, Konstanten, ...
  - Rechner, Dienste, Dateien, Geräte
- Zweck von Namen:
  - **Erklären**: laser88, klauck@informatik.haw-hamburg.de, /etc/passwd/
  - **Identifizieren**: `laser88` statt "der mit „laser88“ bezeichnete Drucker“ oder `10030` als RPC-Programmnummer statt "die mit „HelloWorld“ bezeichnete Prozedur“
  - **Lokalisieren**: Zugriff auf Objekt über den Namen: 131.246.19.42:1633 als Adresse der entfernten Funktion „HelloWorld“
- Für die **Praxis wichtig**:
  - Kenntnisse über Namensauflösung
  - Kenntnisse über Lokalisierung, speziell mobiler Objekte

Namen und Adressen

- Jedes Objekt hat eine Adresse:
  - Speicheradresse
  - Ethernet-Adresse
  - Internet-Adresse
  - Internet-Adresse, Protokoll-Port
- Adressen sind "physische" Namen (Namen unterer Stufe)
  - **Direkte Lokalisierung** eines Objektes
  - in einem **Kontext** (Adressraum, Namensraum) **eindeutig**
- Entkopplung von Namen und Adressen unterstützt **Ortstransparenz**
- **Zuordnung** "Name ---> Adresse" notwendig (Bindung)

Beispiel: URL-Abbildung



Identifikation von Resourcen

- **URI**: Universal Resource Identifier
  - Ein String, der eine Ressource im Netz identifiziert, ohne auf die Zugriffsart einzugehen
  - Spezifiziert in RFC 3986

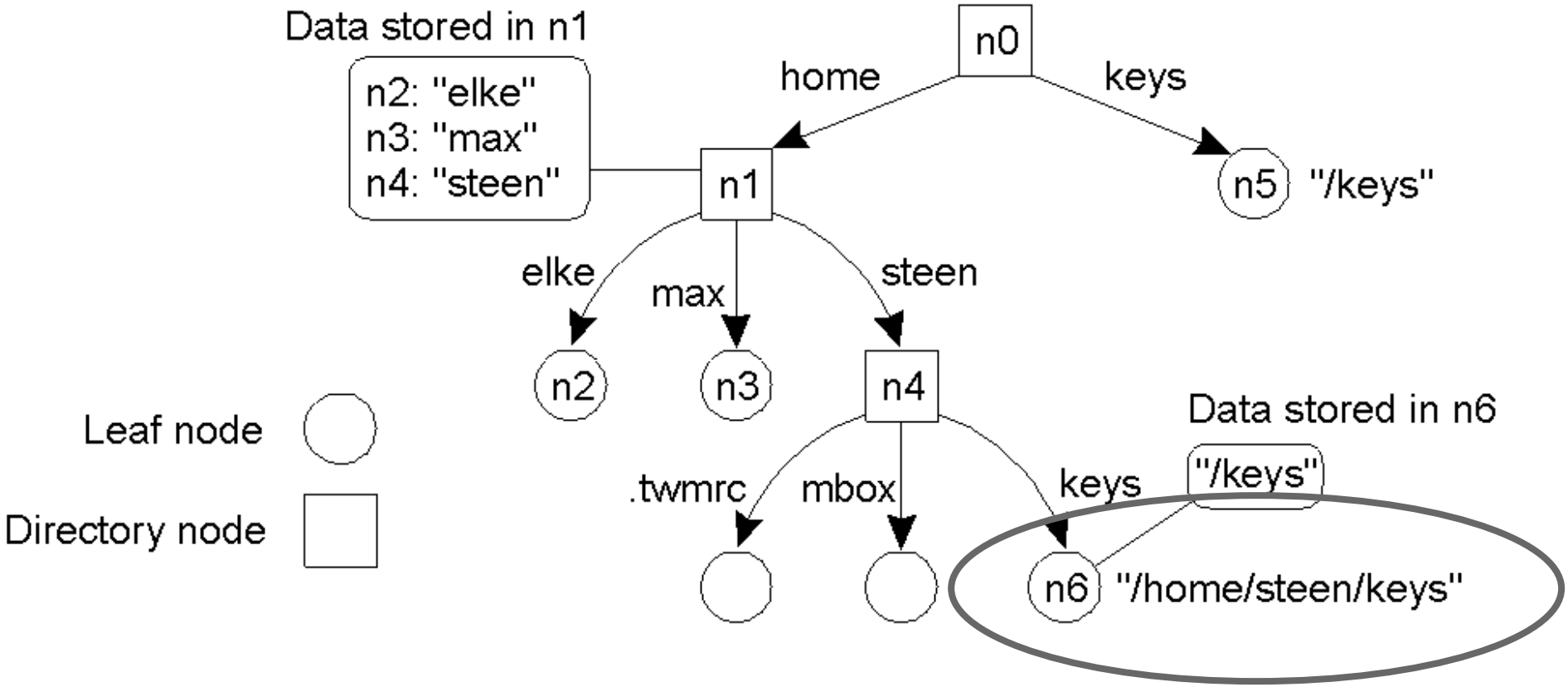
- **URL**: Uniform Resource Locator
  - URLs sind spezielle URIs
  - Eine URL identifiziert eindeutig ein Dokument im Netz, auf das mittels eines Schemes (z.B. HTTP) zugegriffen werden kann
  - URLs haben eine feste Syntax, die das Zugriffsprotokoll und den Ort im Netz identifizieren (DNS-Computername + Pfadname auf dieser Maschine)
  - Definiert zuerst in RFC 1738,

- **URN**: Uniform Resource Name
  - Dauerhafter Name einer (nicht notwendig vernetzten) Resource (historisch)
  - Heute abgedeckt durch URIs

Namen und Binden

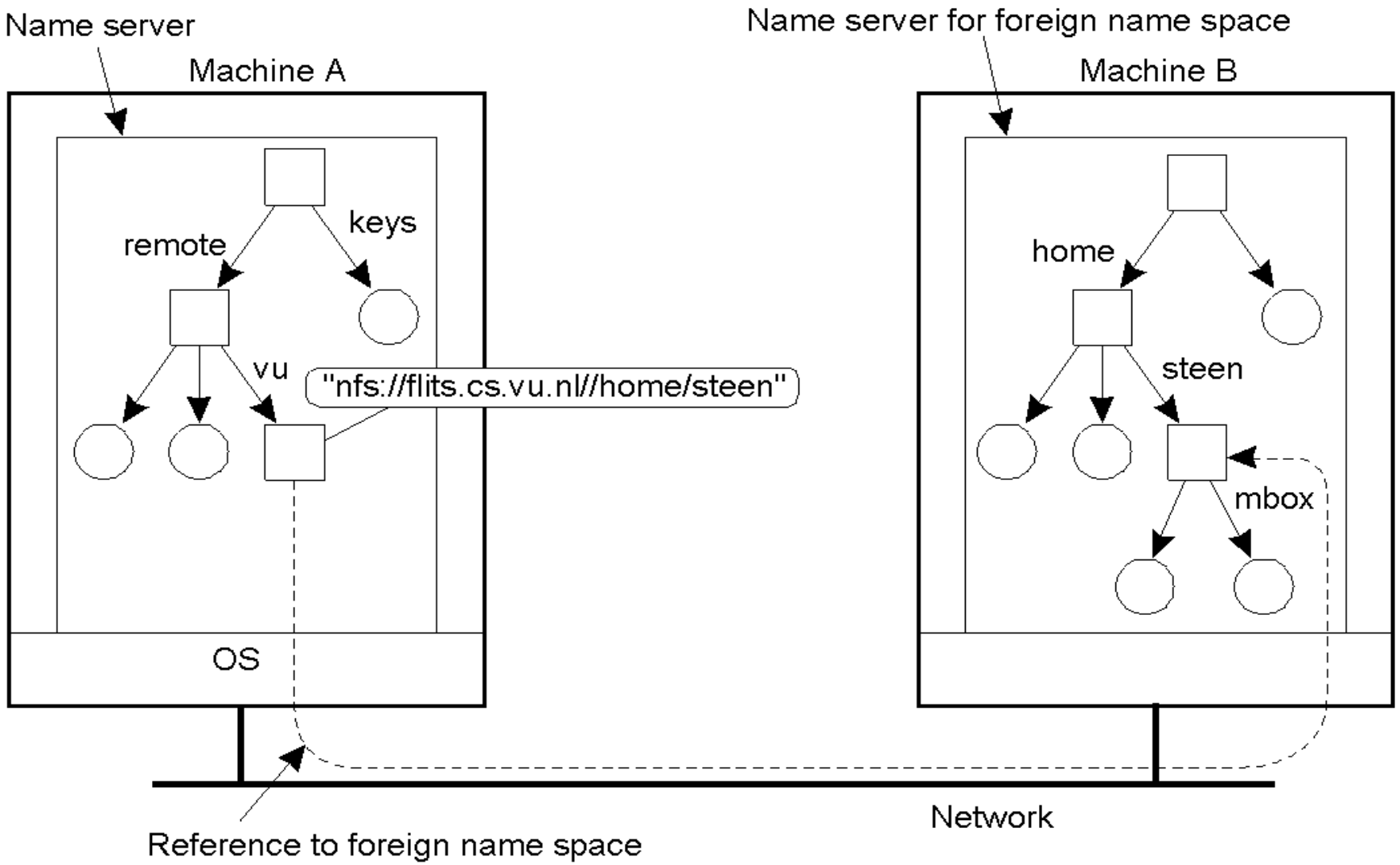
- Zuordnung Name ---> Adresse
- Bindezeitpunkt:
  - beim Übersetzen (**statisches Binden**)
    z.B. bei Programmiersprachen
  - beim Starten (**"halb"-dynamisches Binden**)
    z.B. moderne Binder (SunOS), nach dem Start in der Regel nicht änderbar
  - beim Zugriff (**dynamisches Binden**)
    in verteilten Systemen angebracht:
    - Neue Dienste
    - Verlagerung existierender Dienste

Mounting



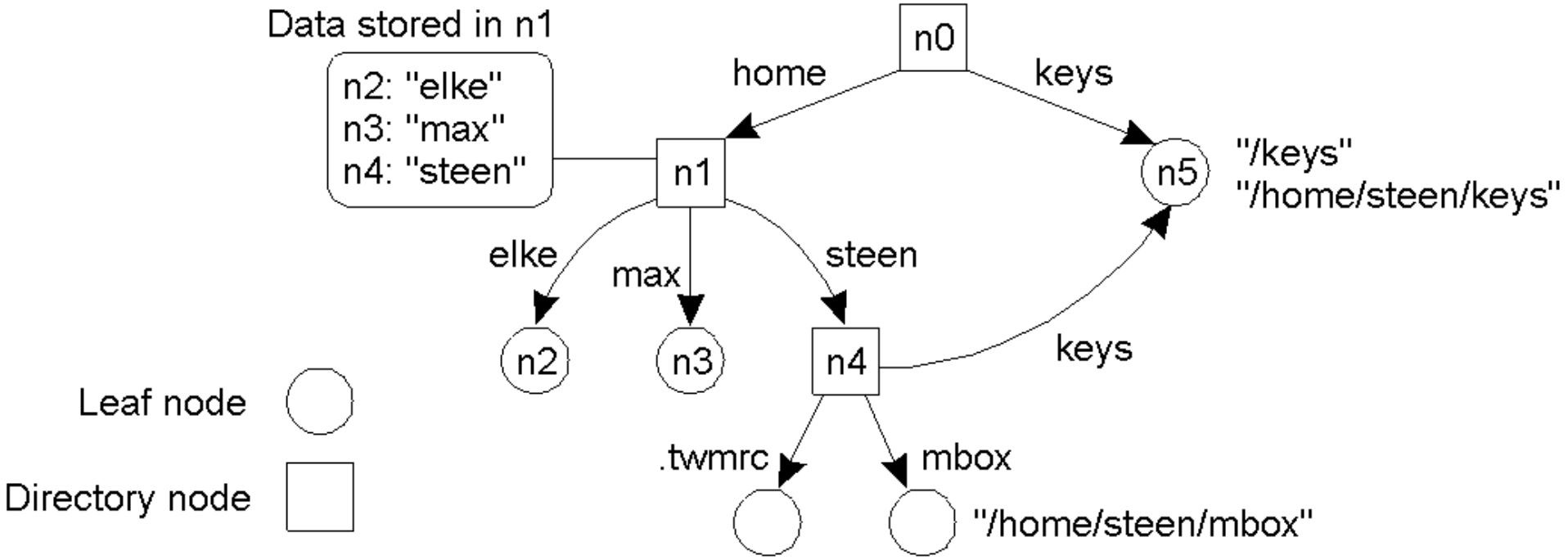
Der Begriff einer symbolischen Verbindung erklärt am Beispiel eines Namensgraphen

Mounting



Mounting eines entfernten Namensraum durch ein spezielles Protokoll

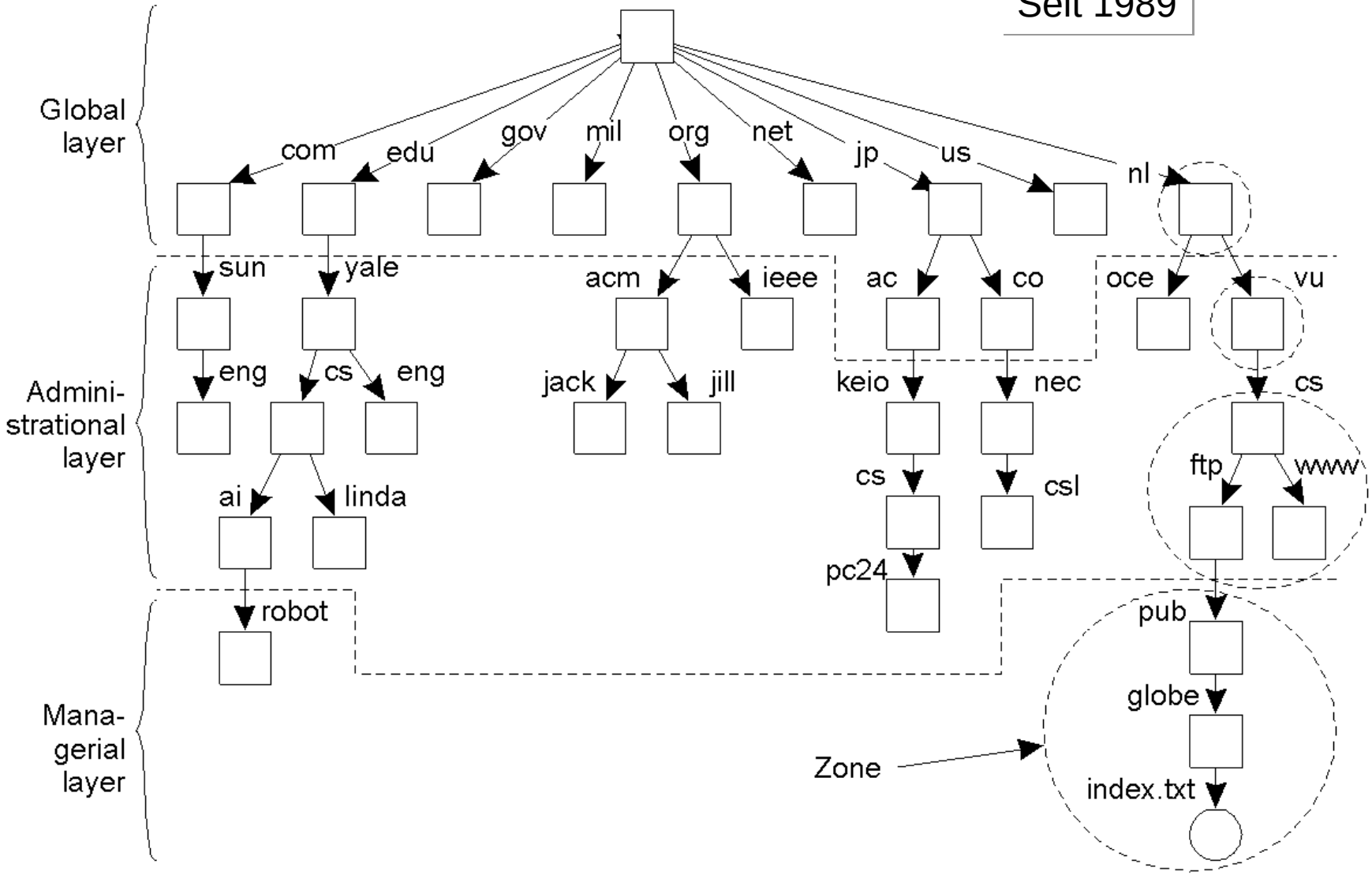
Namen und Namensräume



- Namen nur relativ zu einem Namensraum **eindeutig**
- Existiert nur ein Namensraum --> **flacher Namensraum**
- Namensräume haben selbst wieder Namen --> **hierarchischer Namensraum**
- In hierarchischen Namensräumen bilden Präfixe den Kontext

Logische Schichten

Seit 1989

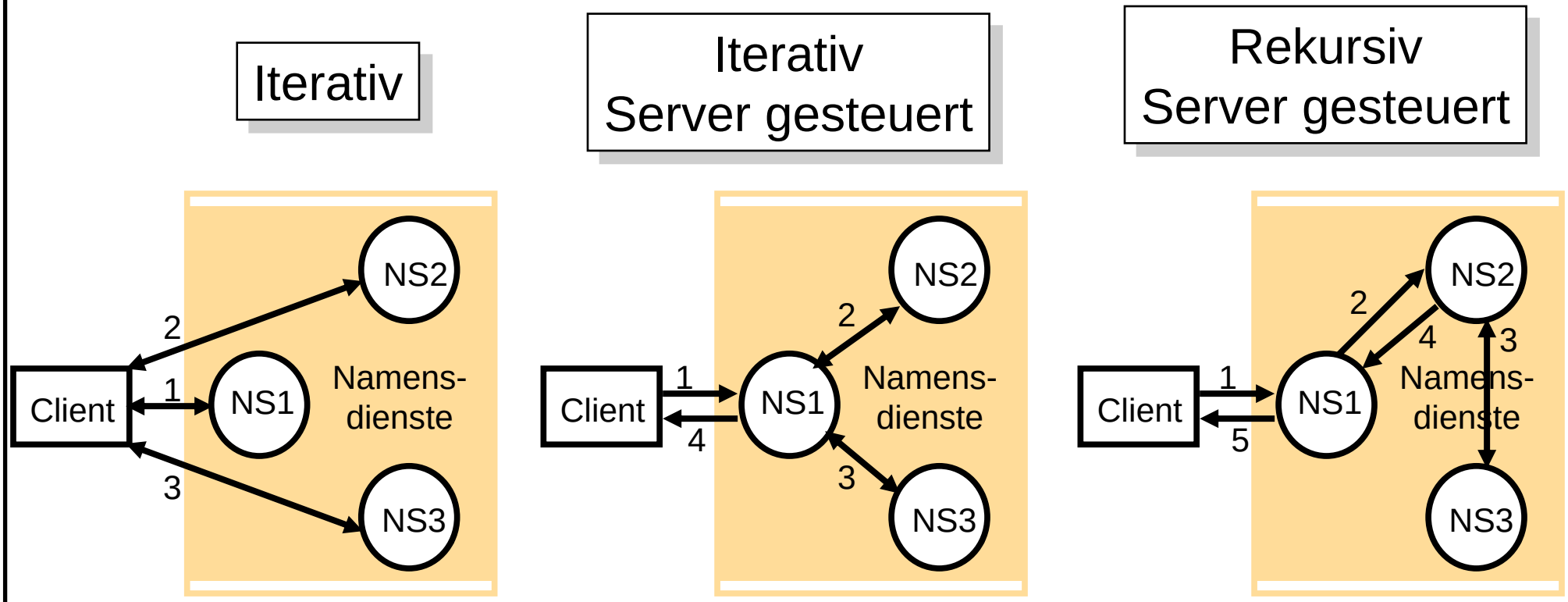


Anforderung an Namensräume

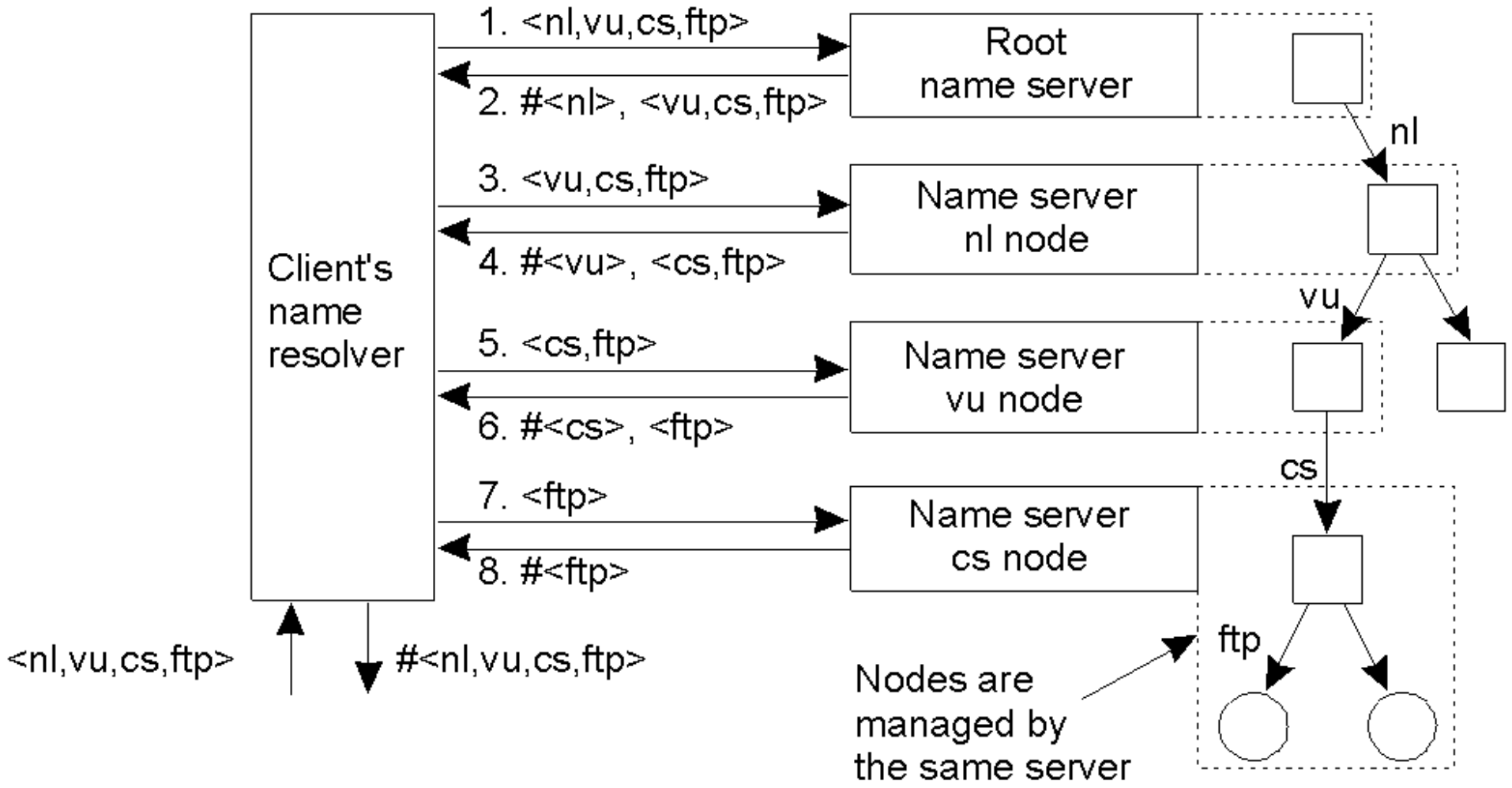
- Einfache, aber **bedeutungsvolle Namen** erlauben
- Potentiell eine **unendliche Anzahl** von Namen zulassen
- **Strukturiert**
  - Um ähnliche Sub-Namen ohne Kollisionen zu erlauben
  - Um verwandte Namen zu gruppieren
- **Re-Strukturierung** von Namensbäumen erlauben: Für einige Arten der Änderung sollten alte Programme weiterhin funktionieren
- **Lange Lebensdauer** des Dienstes (Investitionssicherung)
- **Hohe Verfügbarkeit**
- Management des **Vertrauens**
- **Tolerierung von Misstrauen** (in einem großen System sind nicht alle Komponenten gleich vertrauenswürdig)

Namensauflösung

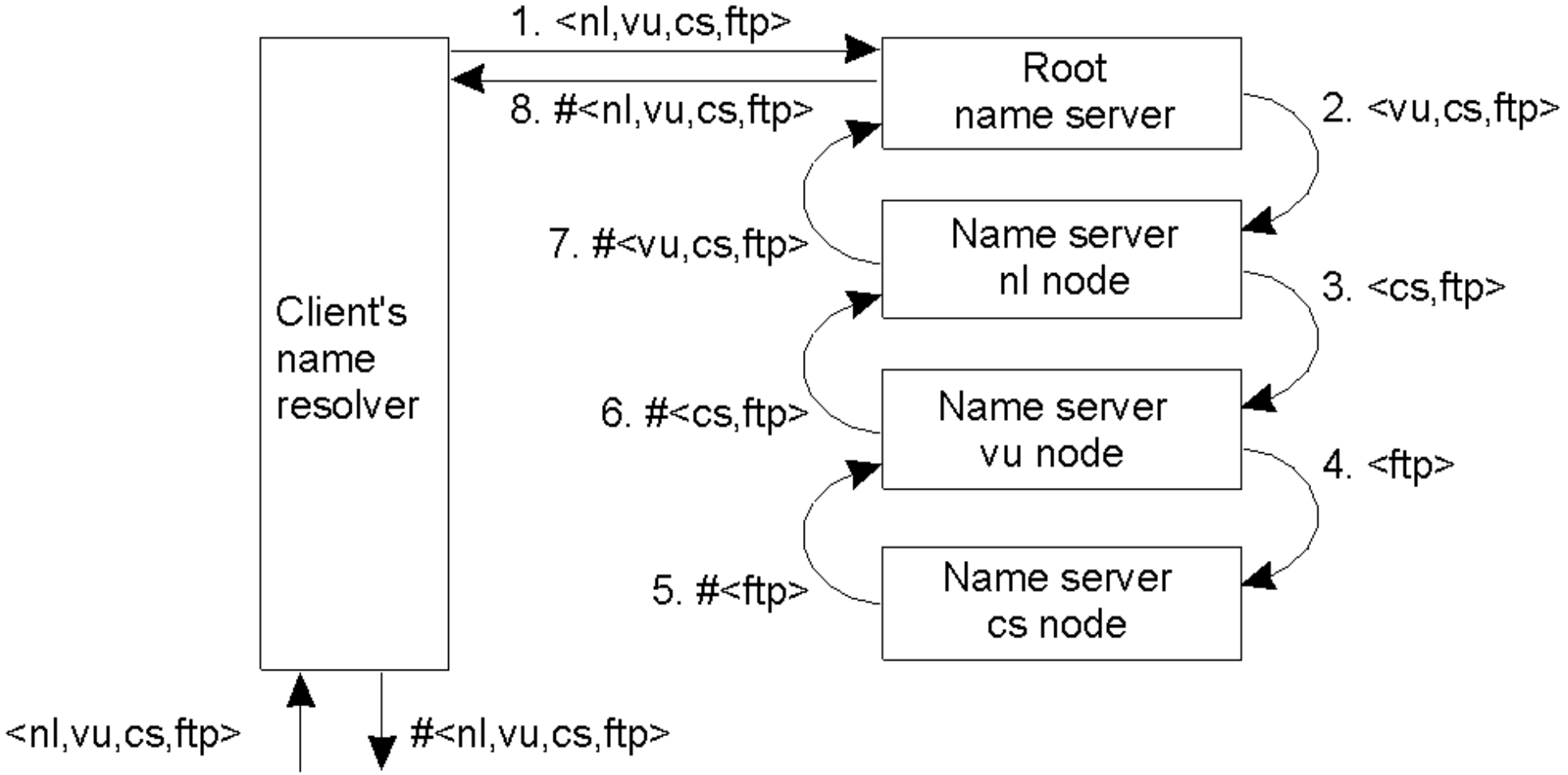
- Namensauflösung: Prozess des Findens der Bindung eines Namens
- Strategien für die Navigation:



Beispiel: Iterative Namensauflösung



Beispiel: Rekursive Namensauflösung



Beispiel: Namensdienste

- Das Telefonbuch
- Die „Gelben Seiten“
- Internet: Domain Name System (DNS)
- JAVA Registry
- X.500 Directory Service (CCITT/ISO) Standard
  verwendet in OSF/DCE, als „Verbesserung“:
  LDAP (Lightweight Directory Access Protocol)
- CORBA NameService
- JNDI (Java Naming and Directory Interface)
- NIS = Network Information Service
- UDDI (Universal Description, Discovery, and Integration)
- ...

Verzeichnisdienste

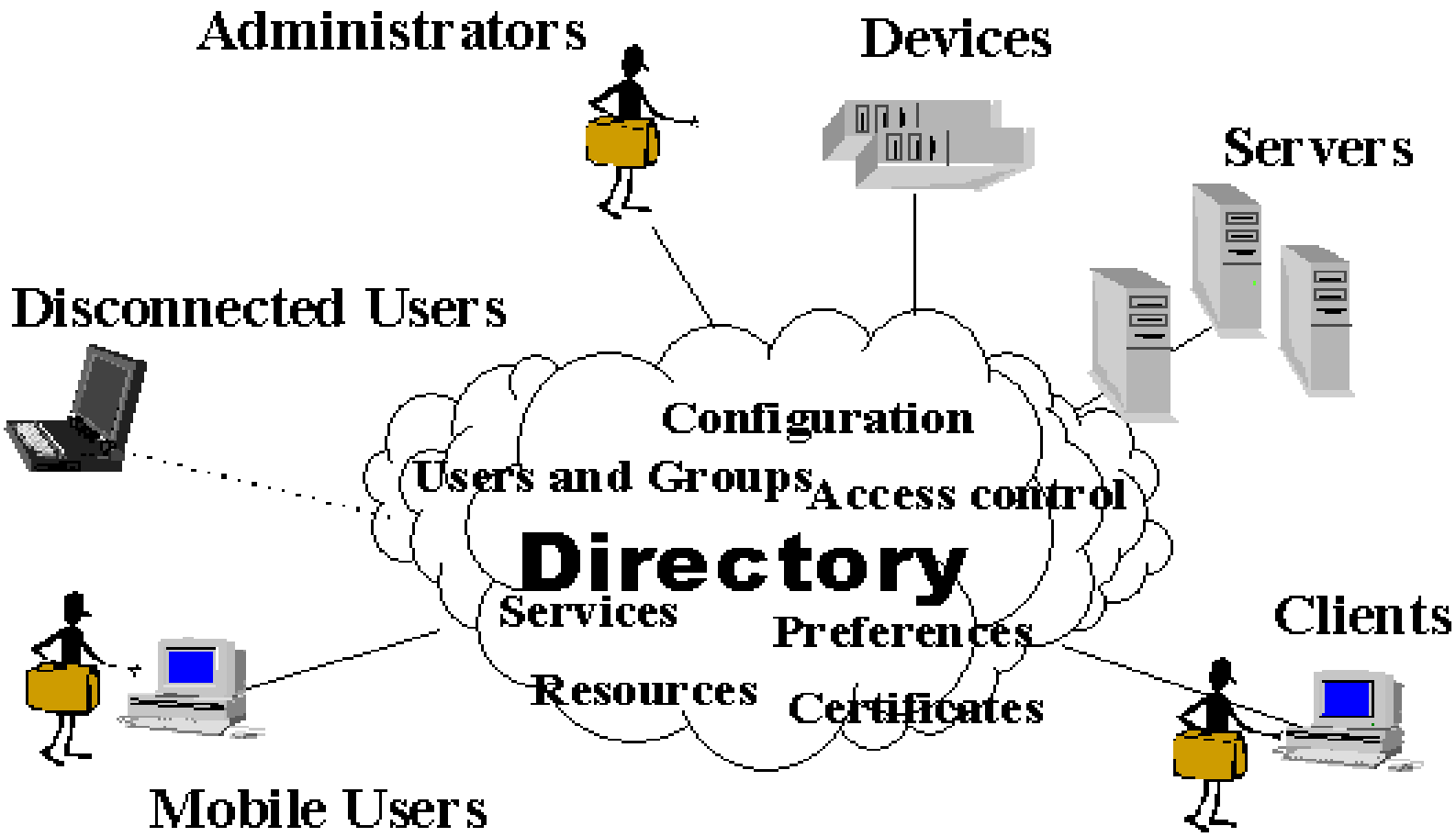
Rechnernetze benötigen in der Praxis weitere, anwendungsspezifische Informationen:

- Login-Daten
- Informationen über netzweite Dienste
- Informationen über Software und Geräte ...

Hierfür sind Verzeichnisdienste gut geeignet:

- Flexibel hierarchisch strukturierbar
- Eigene Zugriffs- und Verteilungsfunktionen
- Beispiele: NIS, NDS, ADS, LDAP

Anwendungen



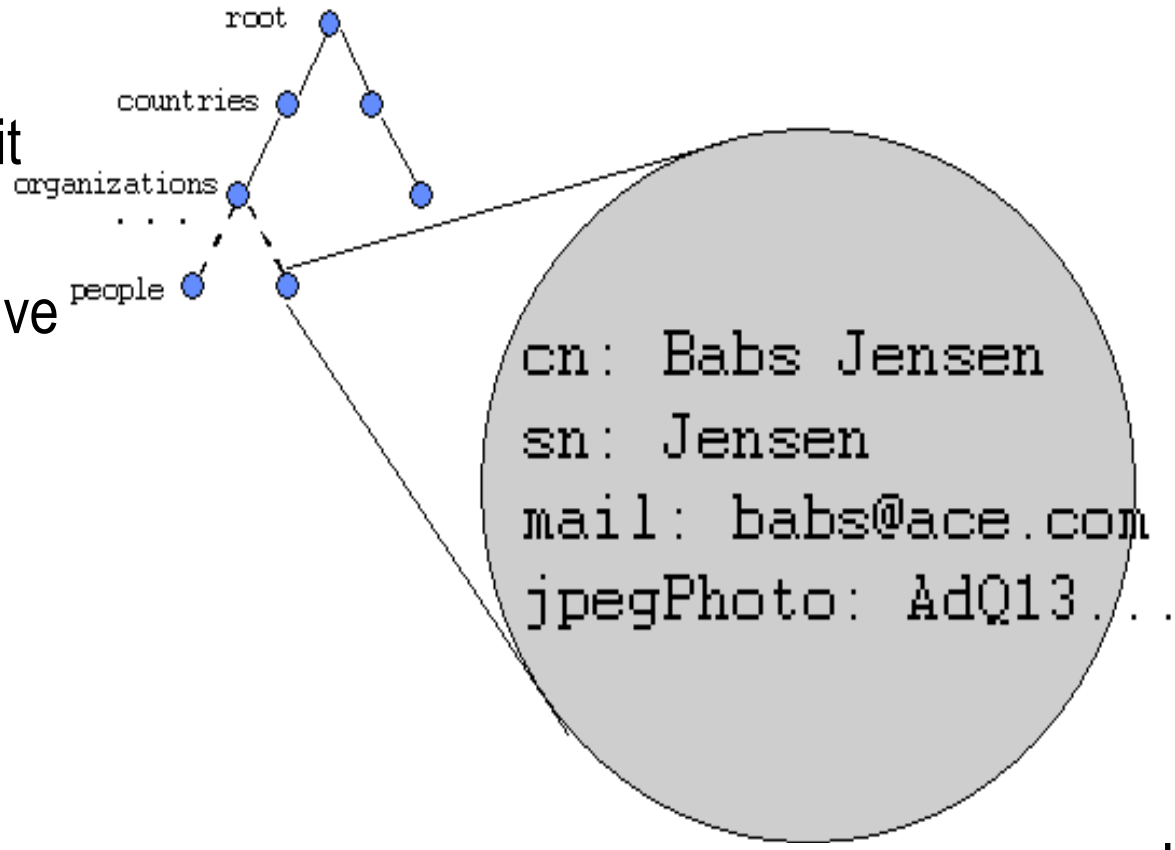
LDAP

Ein LDAP-Baum gliedert sich in Knoten

Jeder Knoten bildet ein Objekt mit zugehörigen Attributen

Jeder Knoten besitzt einen Relative Distinguished Name als Informationsschlüssel

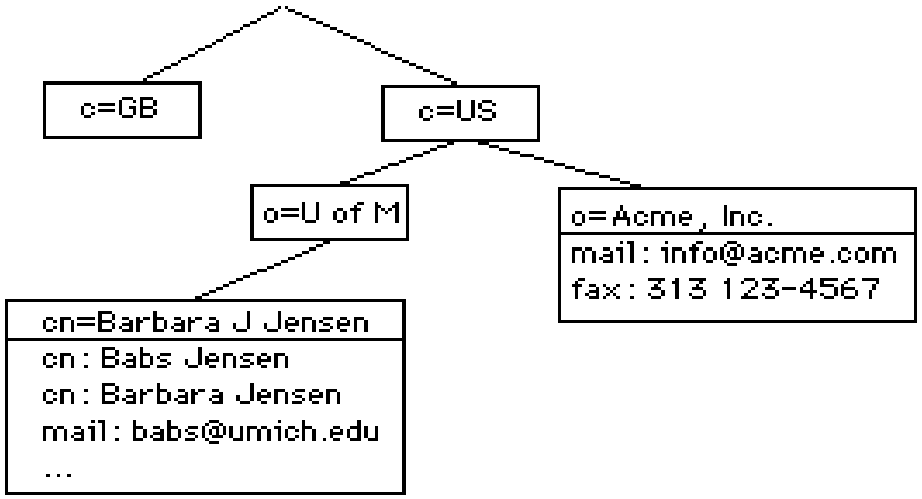
Bsp:



cn= Babs Jensen

LDAP Struktur

- Beinhaltet Informationen über Objekte
- Objekte haben ein oder mehrere Attribute
- Attribute können mehrere Werte annehmen
- Ein Attribut bildet den RDN (cn=Barbara J Jensen)



- Zusammen mit den darrüberliegenden Knoten des Objekts wird der Distinguished Name gebildet:

dn: cn= Barbara J Jensen, o=U of M, c=US

LDAP-Operationen

LDAP Zugriffe bilden drei Gruppen:

Lesen - Schreiben - Verbinden

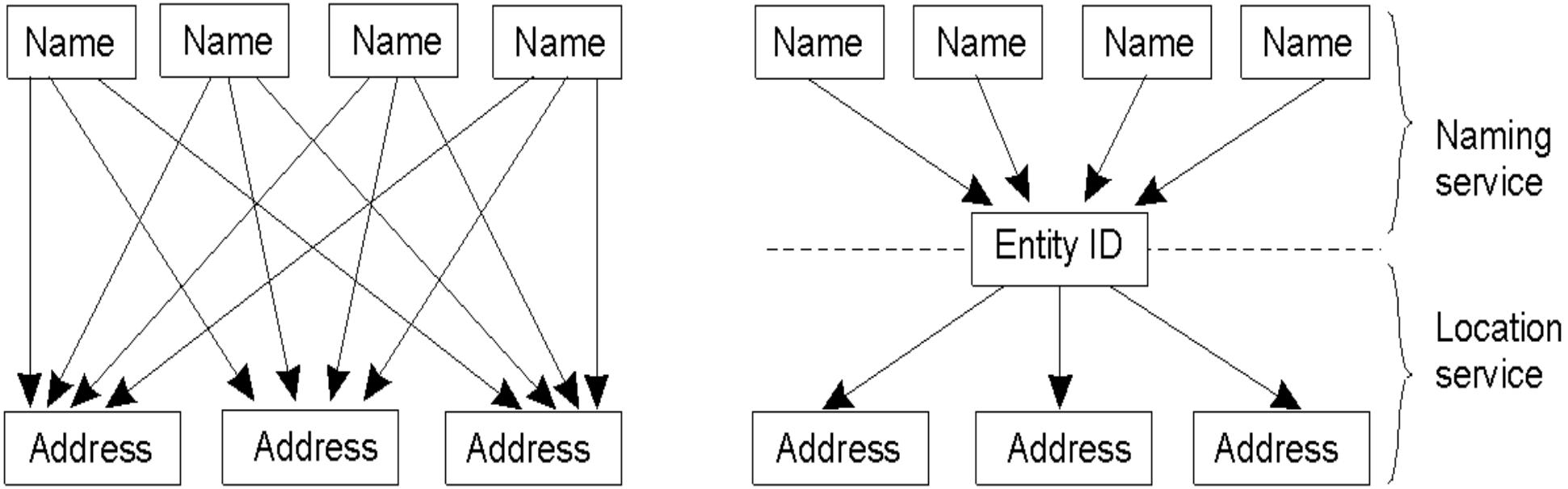
- Bind
- Search
- Compare
- Add
- Delete
- Modify
- ModifyRDN
- Abandon

Bsp: ldapdelete -h ldap.uom.edu -D “cn=LDAP Admin, o=U of M, c=US” -w “secret” “cn= Barbara J Jensen, o=U of M, c=US”

LDAP-Technologien

- LDAP ist ein offener Standard (RFC 3377/4510-4520)
- LDAP hat breite Betriebssystemunterstützung
- LDAP - Server beherrschen Replikation und Delegation (Referral)
- LDAPv3 unterstützt SSL, SASL und Kerberos-Authentisierung
- Die meisten LDAP-Daten sind Textstrings (einfache Kodierung bei Netzübertragung), Binärdaten können verarbeitet werden
- JAVA Zugriff: JNDI

Lokalisierung Mobiler Objekte

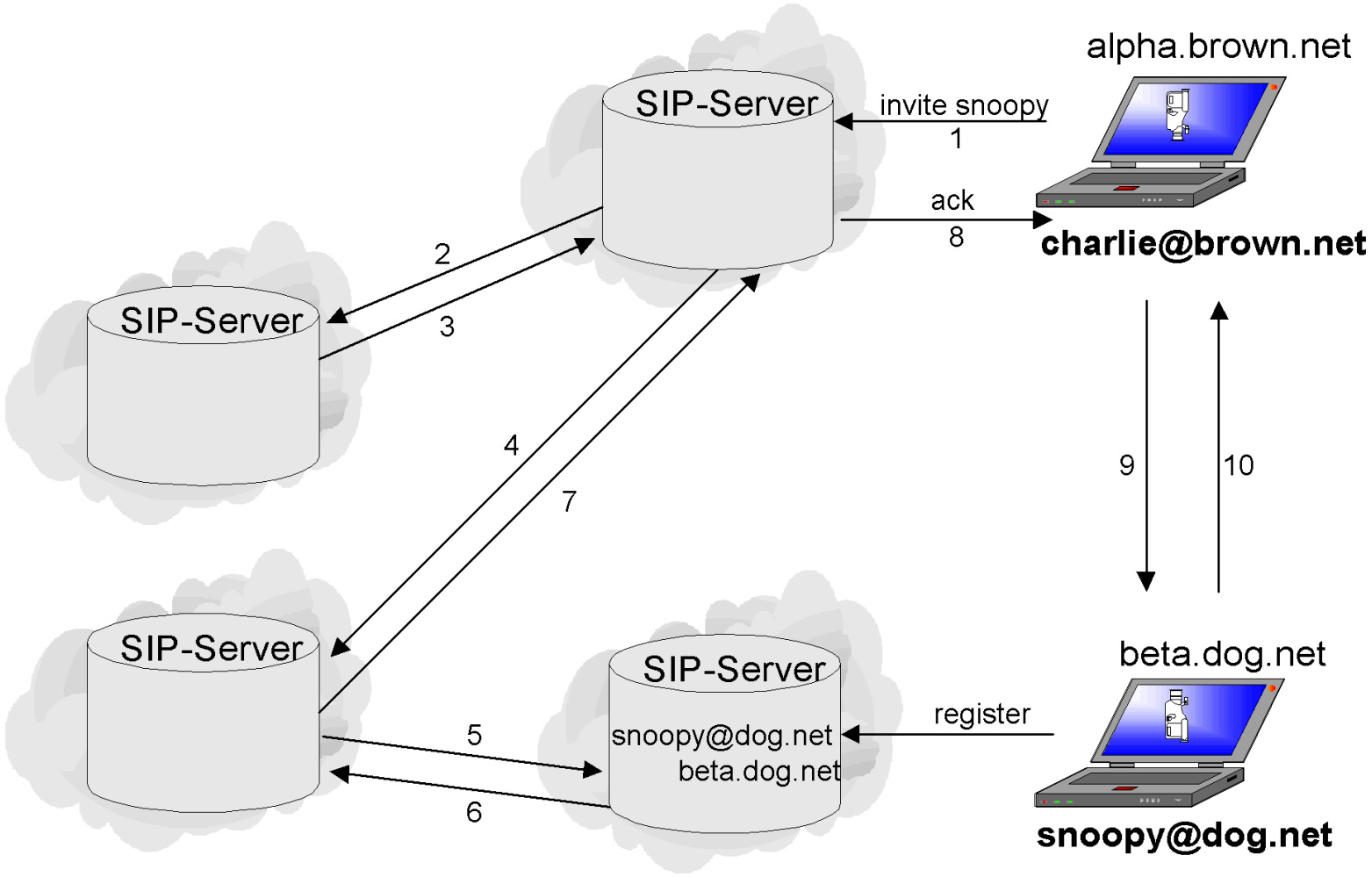


- **Einebenen-Zuordnung:** Direkte Abbildung von Namen auf Adressen. Änderung bei Namens- oder Adressänderungen.
- **Zweiebenen-Zuordnung:** Einführung eines eindeutigen Identifieres, der statisch immer dem selben Objekt zugeordnet ist.

Location Service: Broad-/Multicasting

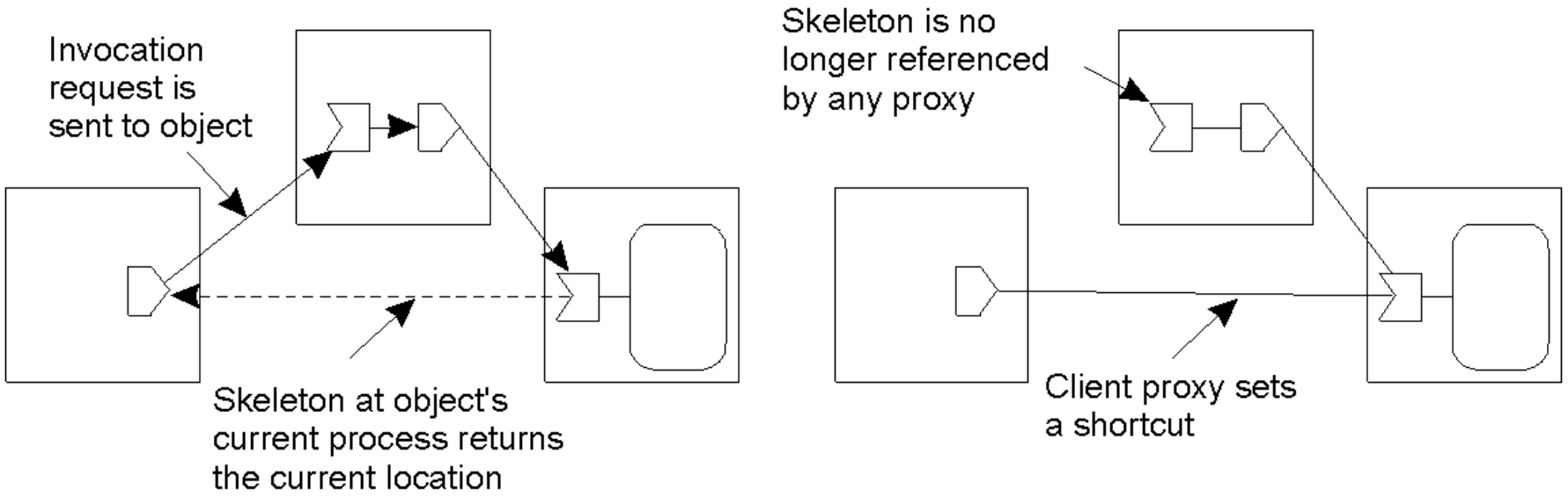
- Nur im LAN mit „überschaubarer Größe“ anzuwenden.
- Ablauf:
  1. Nachricht mit Identifiziere eines Objekts wird per Broadcast im LAN versendet.
  2. Rechner, der Zugangspunkt für das Objekt bereitstellen kann, sendet Antwort mit Adresse des Zugangspunktes
- Ineffizient bei „größeren“ LAN‘s. Einschränkung durch Multicastgruppen möglich.

Unstrukturierter Peer-to-Peer Ansatz: Beispiel SIP Proxies

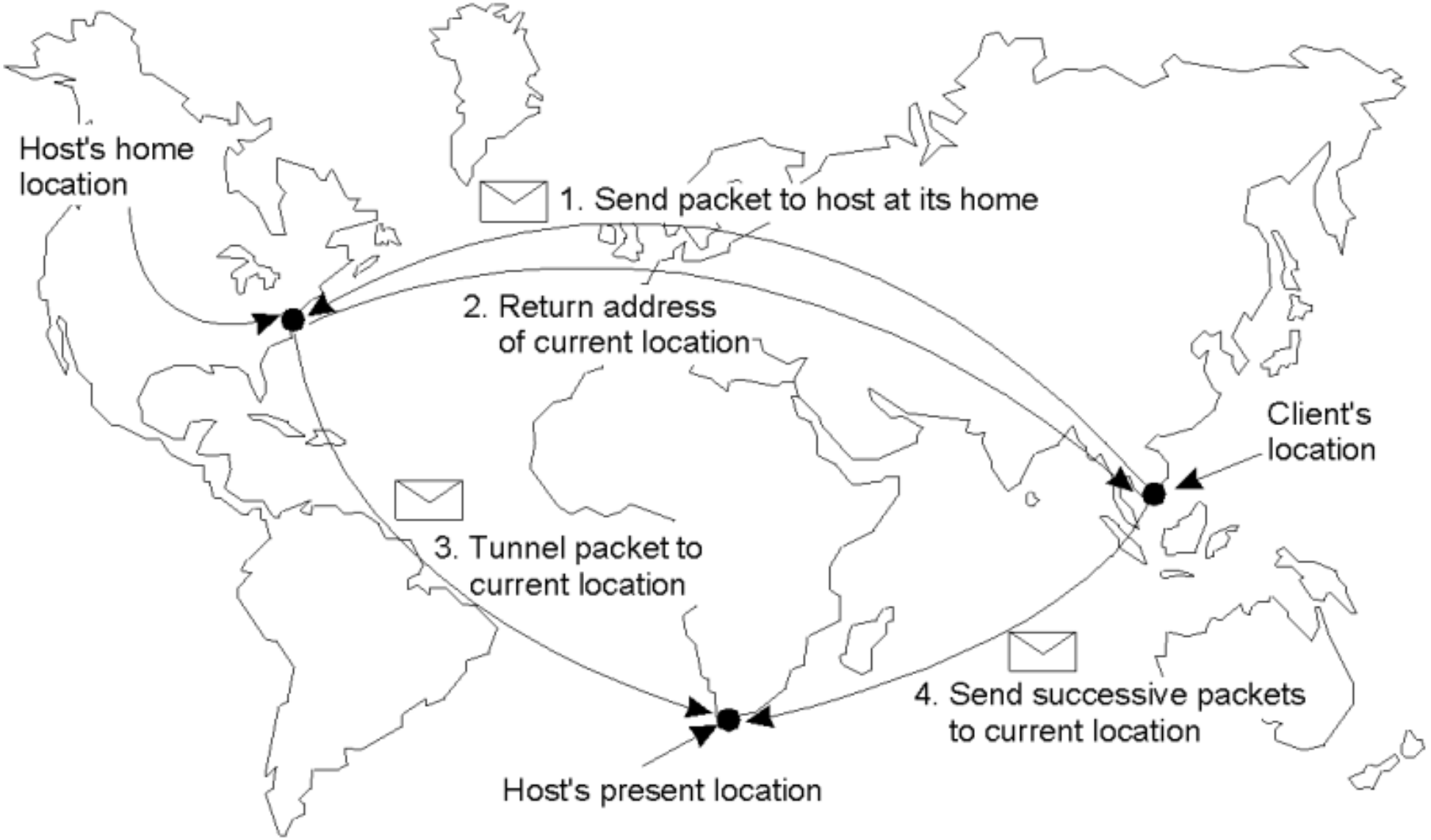


Location Service: Forwarding Pointers

- Nur im LAN mit „überschaubarer Größe“ anzuwenden.
- Ablauf: Bei Migration von Rechner A zu Rechner B wird auf Rechner A eine Referenz auf den neuen Ort B hinterlassen..
- Ineffizient bei „häufiger“ Migration: Es entstehen zu lange Ketten. Einschränkung durch „Kettenreduktion“ möglich.



Location Service: Home Location



Home Location: Mobile IP

- Jeder mobile Host nutzt eine **feste IP-Adresse**
- Vollständige Kommunikation geht an „**Home Agent**“ (an der Home Location im Heimat-Netzwerk)
- Bei **Bewegung** des mobile Host
  - Anforderung einer (temporären) **care-of-Adresse** durch mobilen Host
  - Registrierung der care-of-Adresse beim Home Agent
  - care-of-Adresse nur für Kommunikation
- Feste IP-Adresse ist **eindeutiger, statisch zugeordneter Identifier**

Home Location: Nachteile

- **gesteigerte Kommunikationslatenz** (permanente Nachfrage beim Home Agent)
  - Lösung: Zweischicht-Schema (aus dem Mobilfunk)
    - zuerst eine lokale Registrierung
    - dann (bei Misserfolg) Home-Position kontaktieren

- die Verwendung einer **festen** Home-Position (= IP-Adresse) (Erreichbarkeit ? Entfernung zur tatsächlichen Position ?)
  - Lösung: Verwendung eines Namendienstes
    - Caching der Position möglich
    - Bei Misserfolg Namensdienst abfragen

Kernproblem: Identifier-Locator Split

- Jeder Teilnehmer benötigt zwei Bezeichner (Adressen)
  - Ein **Identifier** (ID) bezeichnet, **wer** er ist
  - Ein **Locator** (Loc) bezeichnet, **wo** er ist
- In statischen Netzen sind beide Bezeichner statisch
  - Können sich deshalb auf eine Adresse reduzieren
  - Im Internet ist die ‚reguläre' IP Adresse ID und Loc
- In mobilen Netzen ändert sich der Locator
  - ID und Loc divergieren: **ID-Loc Split**
  - Netzwerk und Endsysteme müssen Dualität handhaben

IPv6 Mobility: The Answer of the New Internet

Objective:

Application persistence while roaming between IP subnets / providers

Preserve upper layer (L 4+) communication when changing IP subnets

Key Aspects:

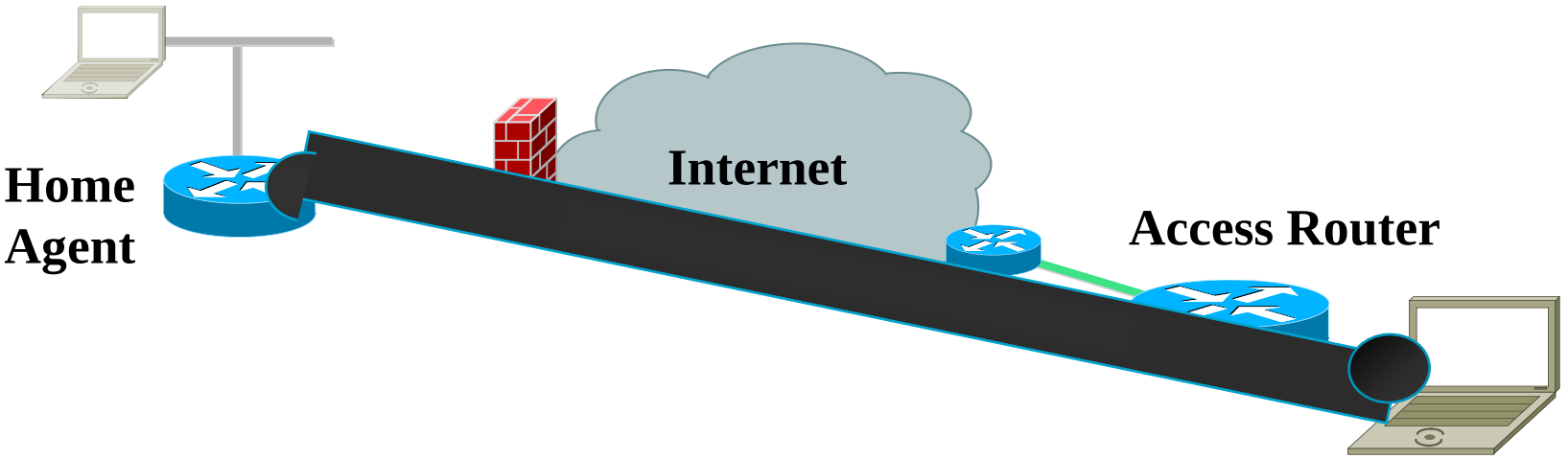
- Mobile Node (MN) globally addressable: fixed Home Address (HoA)
- Home Agent (HA) to permanently represent MN at home network
- Mobile Node locally addressable: changing Care of Address (CoA)
- Sustain partner sessions: update Correspondent Nodes (CN)
- Enable efficient communication (route optimisation)

Approaches:

Mobile IPv4: IP Mobility Support for IPv4 (RFC 3344)

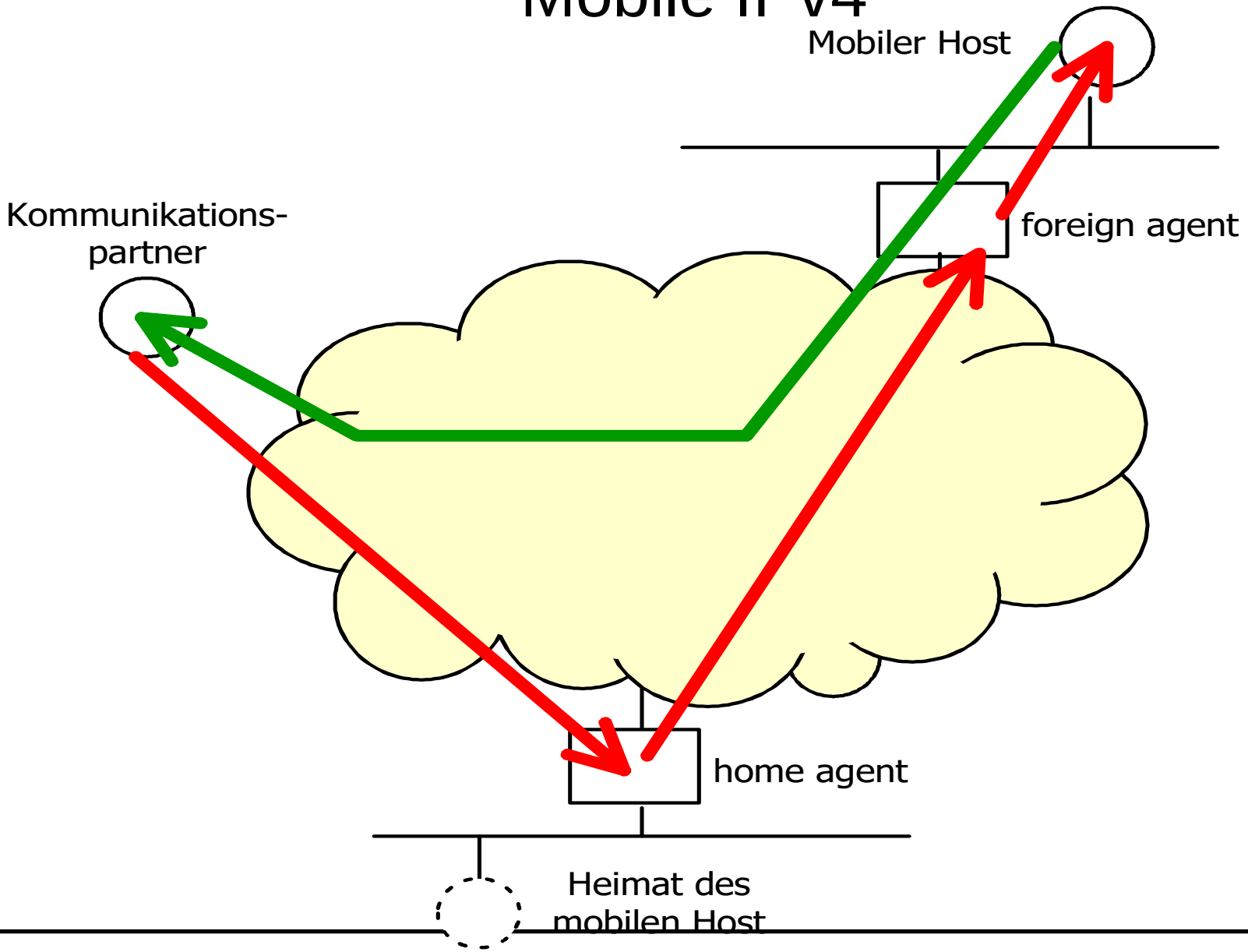
Mobile IPv6: Mobility Support in IPv6 (RFCs 3775/3776, 6275)

Mobile IP

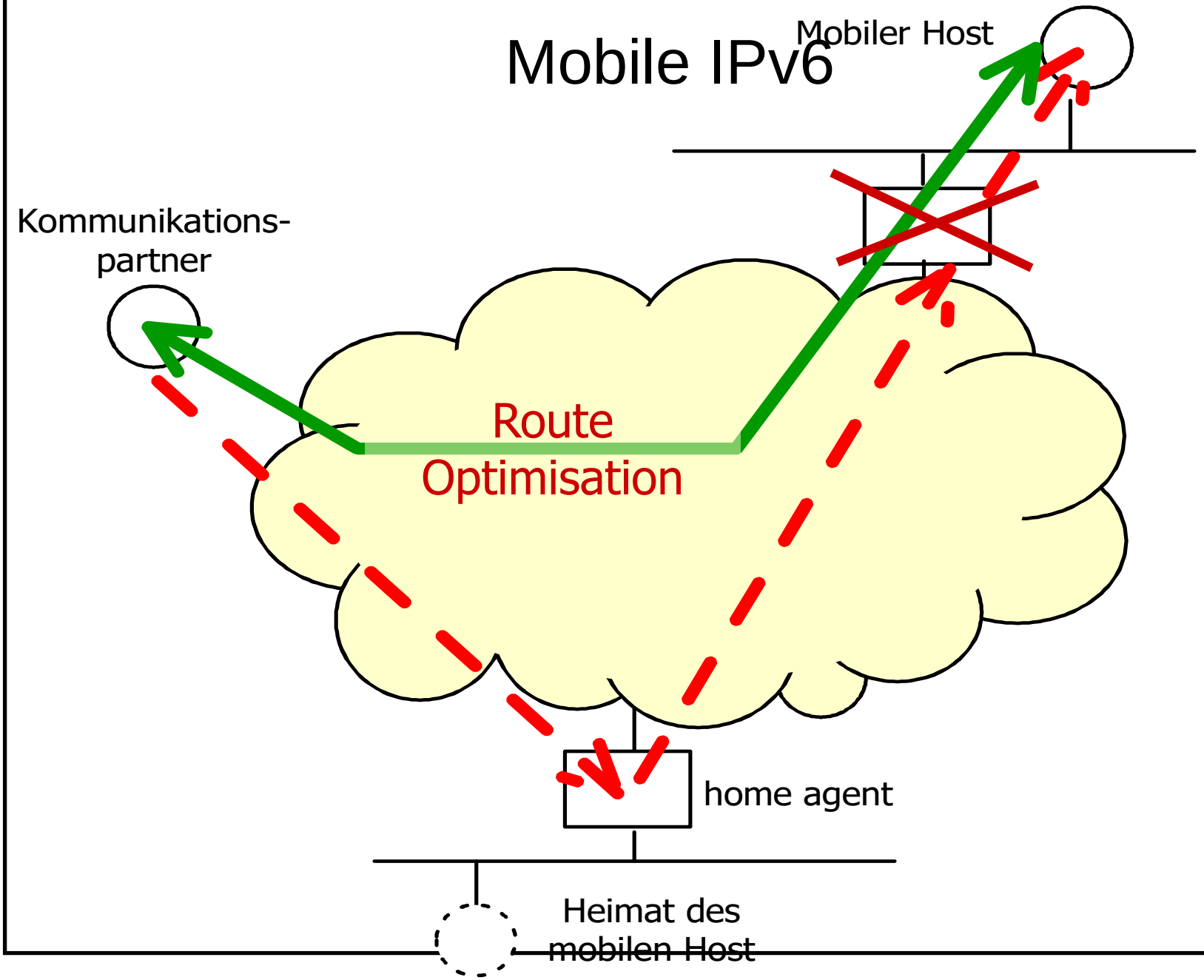


- IPv4‘s Design Stationary (Routing-Updates Slow)
- Implementation of Mobile Services: Tunneling via Home Agent
- IPv6 Potential:
  - Several Addresses (2 for Mobile Node, many for Mobile Networks)
  - Flexible Architecture - no dedicated Access-Services (Agents, DHCP)

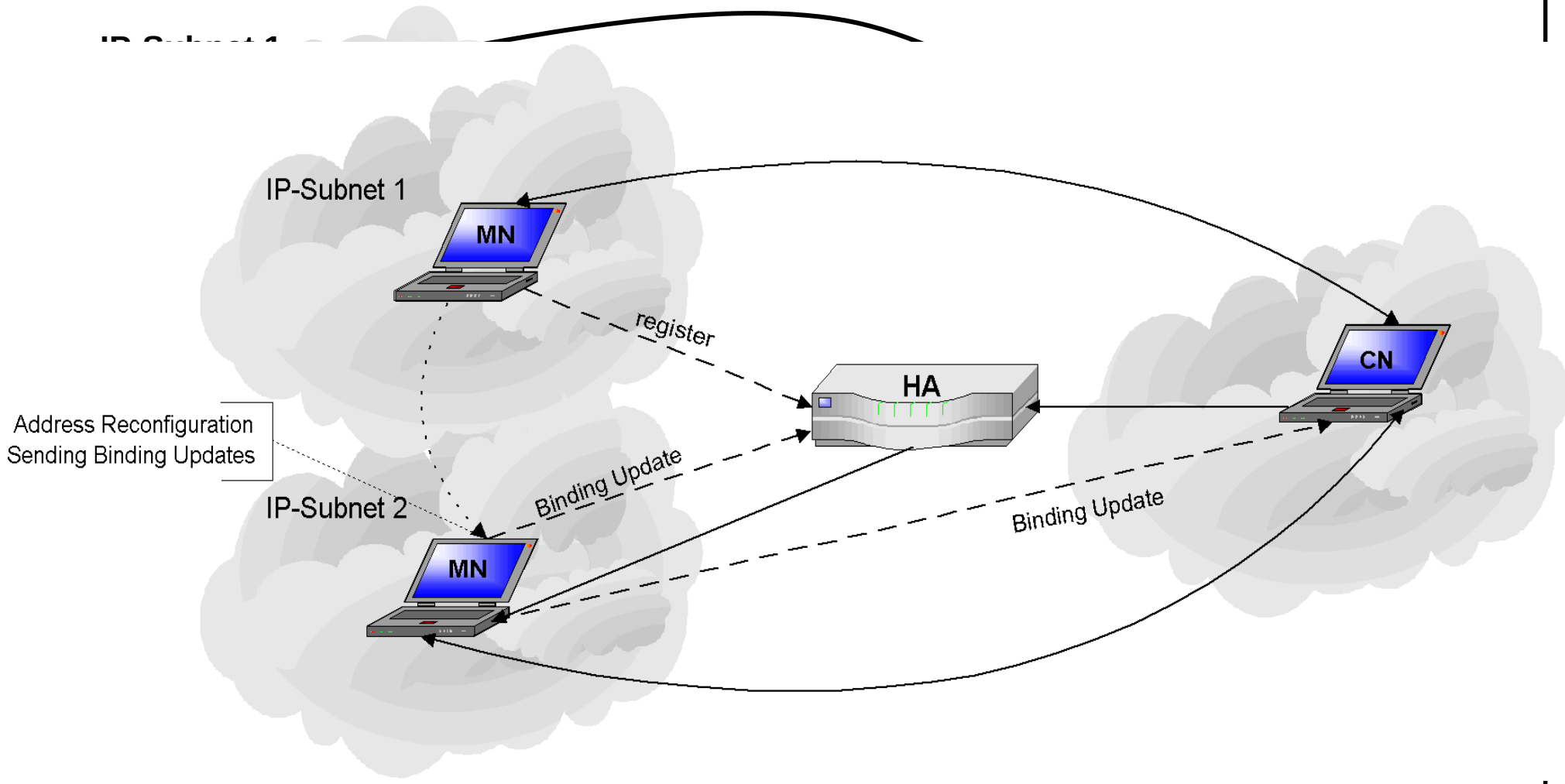
Mobile IPv4


Mobiler Host
Kommunikations-
partner
foreign agent
home agent
Heimat des
mobilen Host

Mobile IPv6


Mobiler Host
Kommunikations-
partner
Route
Optimisation
home agent
Heimat des
mobilen Host

Mobile IPv6


IP-Subnet 1
MN
register
HA
CN
Address Reconfiguration
Sending Binding Updates
IP-Subnet 2
Binding Update
Binding Update
MN

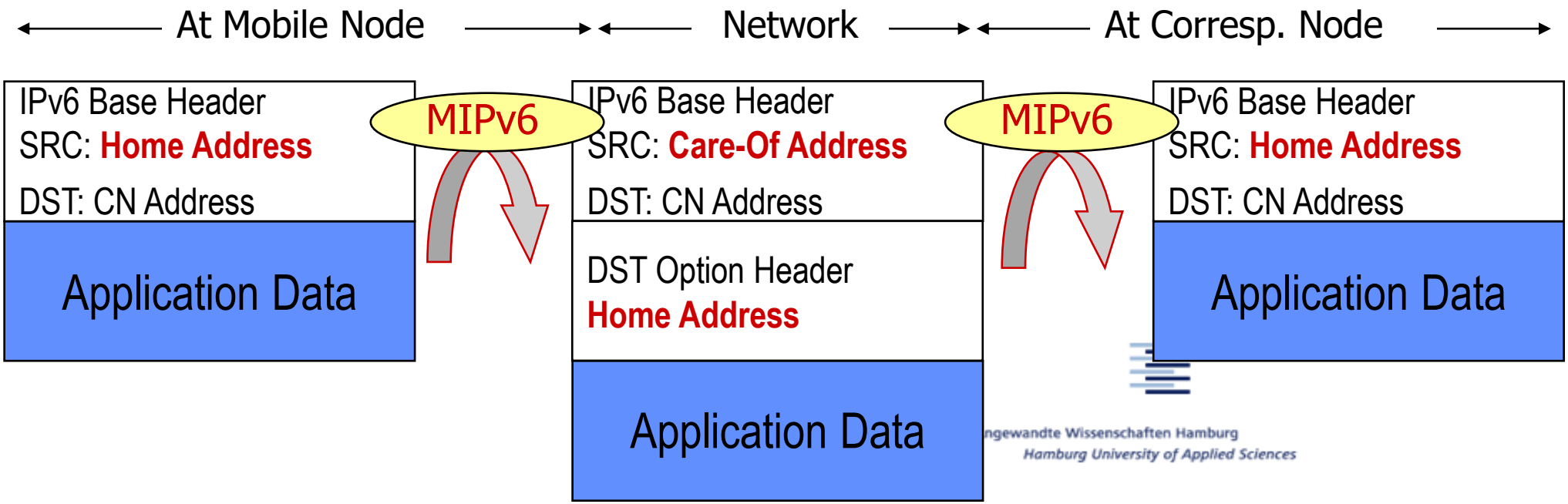
Basic Mobile IPv6

MIPv6 transparantly operates address changes on IP layer by:

- MN‘s stateless configuration of Care of Address in a foreign network and Binding Updates (BUs) with Home Agent (HA) and Correspondent (CNs).
- MN continues to use its original Home Address in a Destination Option Header, thereby hiding different routes to the socket layer.
- CNs continues to use Home Address of the MN, placing current CoA in a Routing Header as Source Route.
- MN, CN & HA keep Binding Cache Tables.
- Home-Agent needed as Address Dispatcher.

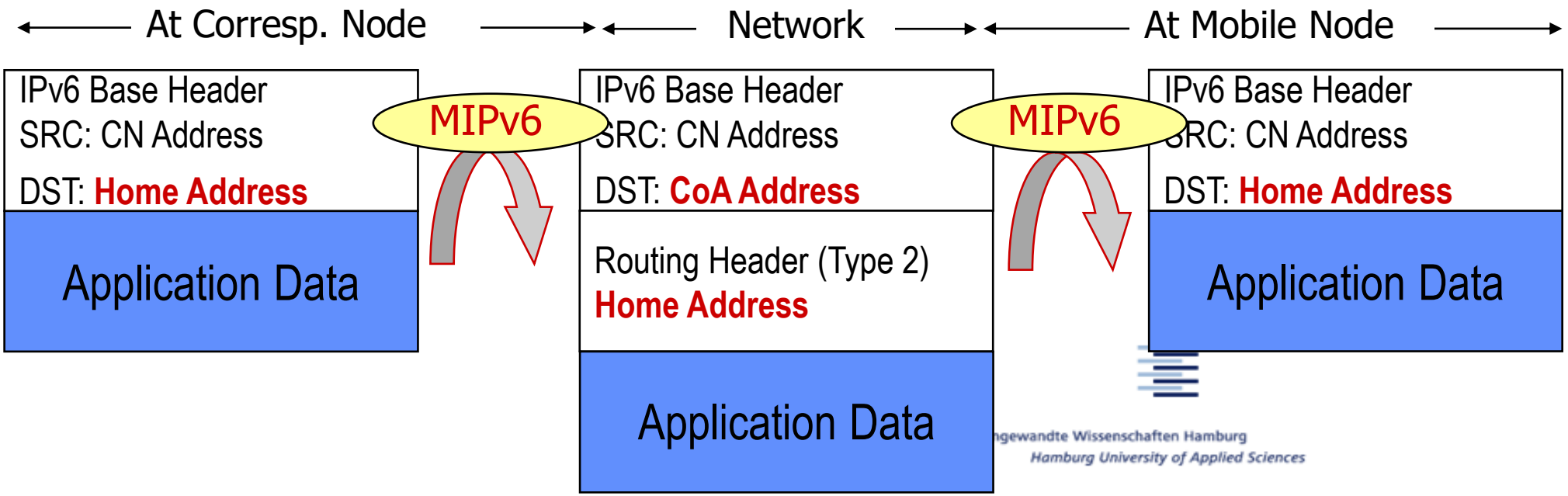
MIPv6 Transparent Communication MN ➔ CN

- Application persistence requires continuous use of HoA
- Infrastructure requires use of topologically correct source address: CoA
- MIPv6 stack moves HoA to Destination Option Header



MIPv6 Transparent Communication CN ➔ MN

- Application persistence requires continuous use of HoA
- Route optimisation operates with CoA
- MIPv6 extracts CoA from Binding Cache and initiates source routing to HoA via CoA

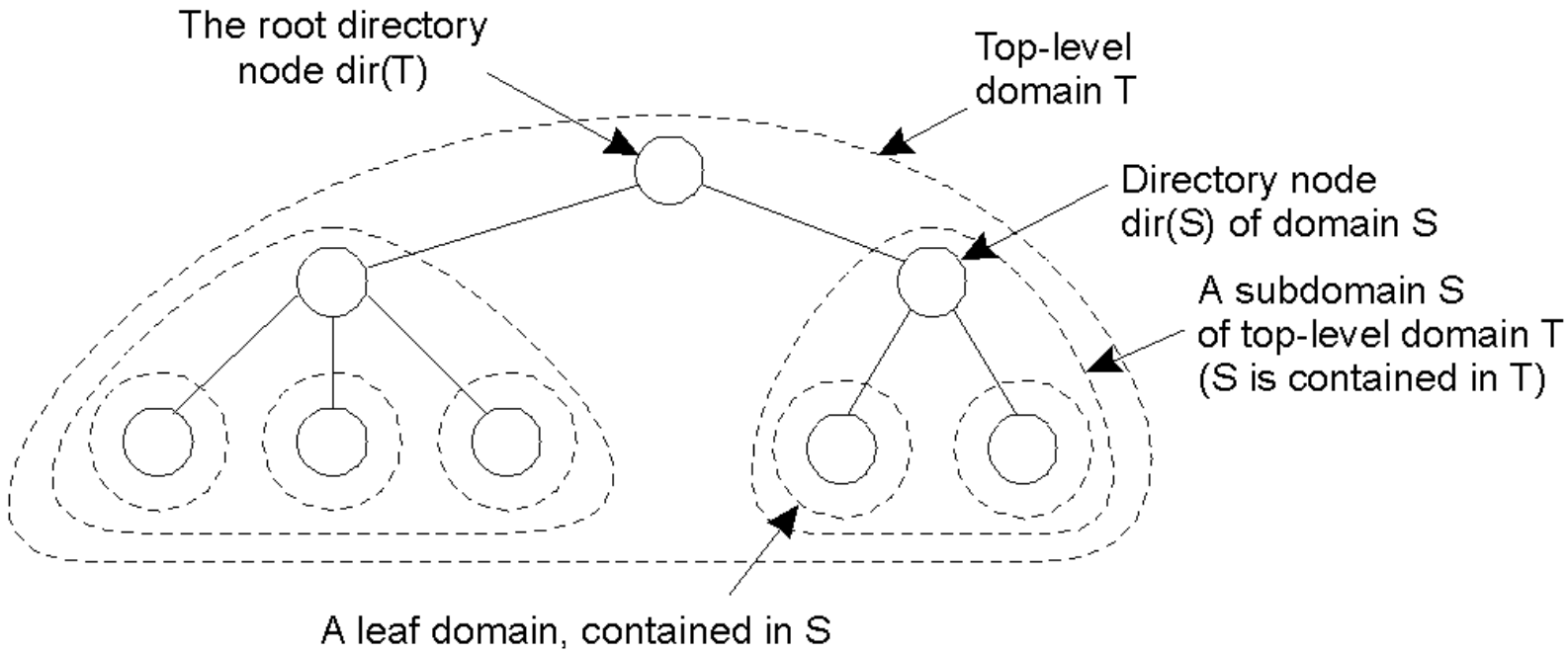


Errungenschaften von MIPv6

- Implementiert **ID-Loc Split auf Netzwerkebene**
  - Splitting und Merging ‚wo sie auftreten‘
  - Modifikation nur an Endgeräten ➨ MIPv6-Stack
  - Router sehen stets korrekte Locator
- Erzielt nahezu optimale Kommunikationsbeziehungen
  - Detour via Home Agent nur für erste Pakete
- Sicherheit noch unbeantwortet
  - Gefahr: Diebstahl von Diensten
  - Elegante, zuverlässige Lösungen ➨ mehr im Master

Location Service: hierarchische Ansätze

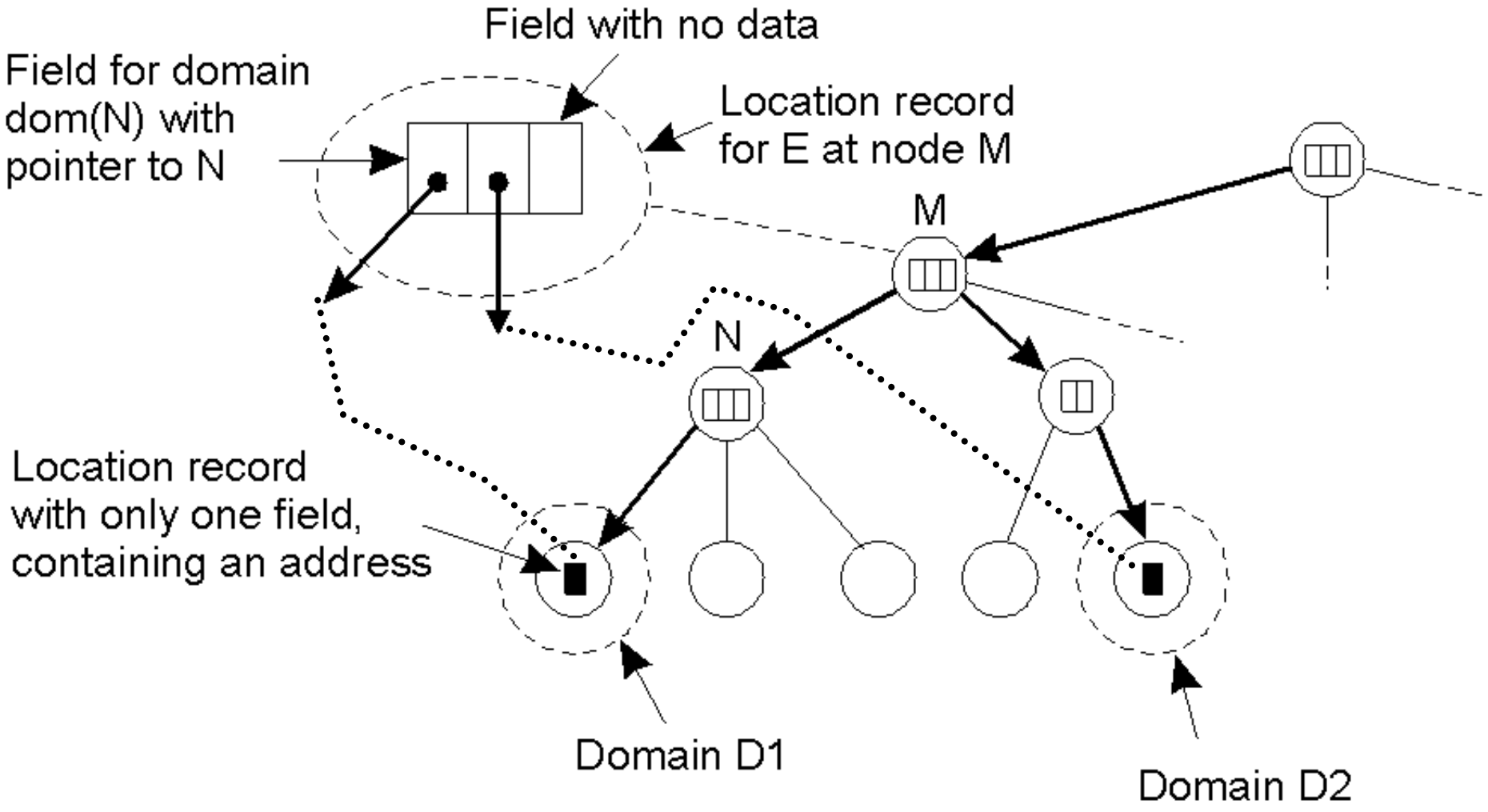
- Erweiterung der zweistufigen Home-Location zu mehreren Stufen
- ist ein (allgemeines) **hierarchisches Suchschema**
- **Ausgangspunkt**:
  - Netzwerk in mehrere Domänen hierarchisch unterteilt (ähnlich DNS)
  - Jeder Domäne ist Verzeichnisknoten dir(D) zugeordnet



Hierarchischer Ansatz

- Jedes Objekt E in Domäne D wird durch **Positionsdatensatz** in dir(D) dargestellt.
  - Positionsdatensatz für Objekt O im Verzeichnisknoten N für Blattdomäne D enthält die **aktuelle Adresse** von O
  - Positionsdatensatz für Objekt O im Verzeichnisknoten N' für nächst höhere Domäne D‘enthält nur **Zeiger auf N**.
  - Positionsdatensatz für Objekt O im Verzeichnisknoten N‘‘ für nächst höhere Domäne D‘‘enthält nur **Zeiger auf N‘**.
  - usw.
- **Wurzel-Knoten** besitzt Positionsdatensatz für **jedes Objekt**
- Bei **mehreren Adressen** (z.B. durch Replikate) existieren **mehrere Zeiger**

Hierarchischer Ansatz: Beispiel



Hierarchischer Ansatz: lookup