Distributed Systems

Naming

Chapter 5

Course/Slides Credits

Note: all course presentations are based on those developed by Andrew S. Tanenbaum and Maarten van Steen. They accompany their "Distributed Systems: Principles and Paradigms" textbook (1st & 2nd editions). http://www.prenhall.com/divisions/esm/app/author_tanenbaum/custom/dist_sys_1e/index.html

And additions made by Paul Barry in course CW046-4: Distributed Systems http://glasnost.itcarlow.ie/~barryp/net4.html

Naming

- Naming systems play an important role in all computer systems, and especially within a distributed environment.
- The three main areas of study:
  1. The organization and implementation of human-friendly naming systems.
  2. Naming as it relates to mobile entities.
  3. Garbage collection – what to do when a name is no longer needed.

Some Definitions

- *Entity* – just about any resource.
- *Name* – a string (often human-friendly) that refers to an entity.
- *Address* – an entities “access-point”.
- A name for an entity that is independent of an address is referred to as “location independent”.
- *Identifier* – a reference to an entity that is often unique and never reused.

Names, Identifiers, and Addresses

- Properties of a true identifier:
  - An identifier refers to at most one entity.
  - Each entity is referred to by at most one identifier.
  - An identifier always refers to the same entity.

Locating Mobile Entities

- Tricky …
- Traditional naming services (DNS, X.500) are not suited to environments where entities change location (i.e., move).
- The assumption is that moves occur rarely at the Global and Administrative layers, and when moves occur at the Managerial layer, the entity stays within the same domain.
- But, what happens if `ftp.cs.vu.nl` moves to `ftp.cs.unisa.edu.au`?

Possible Solutions

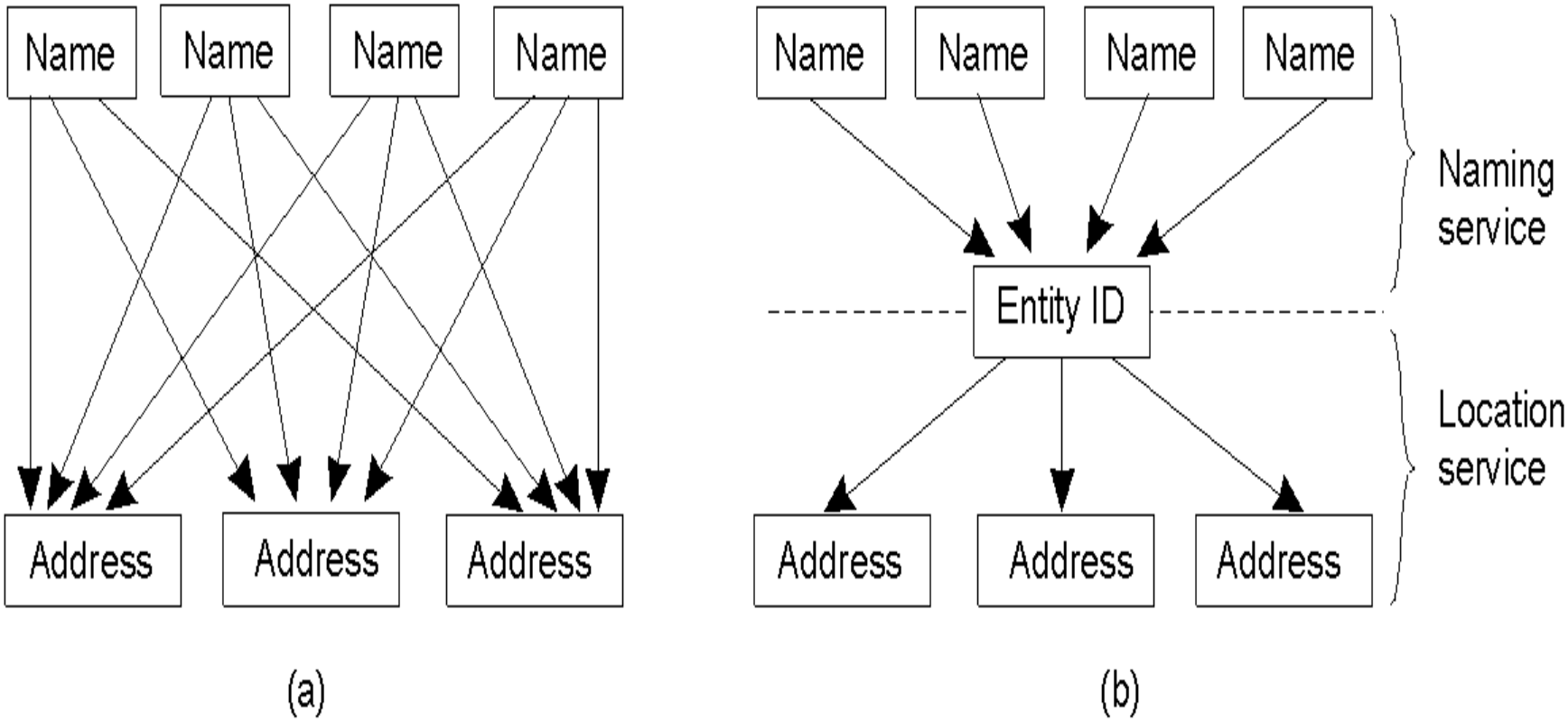
1. A record of the new address of the entity is stored in the cs.vu.nl name server.
2. A record of the name of the new entity is stored in the cs.vu.nl name server (i.e., a *symbolic link* is created).

- Both "solutions" seem OK, until you consider what happens when the entity moves again, then again, then again …
- Consequently, both "solutions" can be shown to be *inefficient* and *unscalable*.

More Location Problems

- Even non-mobile entities that change their name often cause name space problems – consider the DNS within a DHCP environment (currently incompatible).
- So … a different solution is needed.
- What's required is a "Location Service" (or middle-man technology).

Naming vs. Location Services



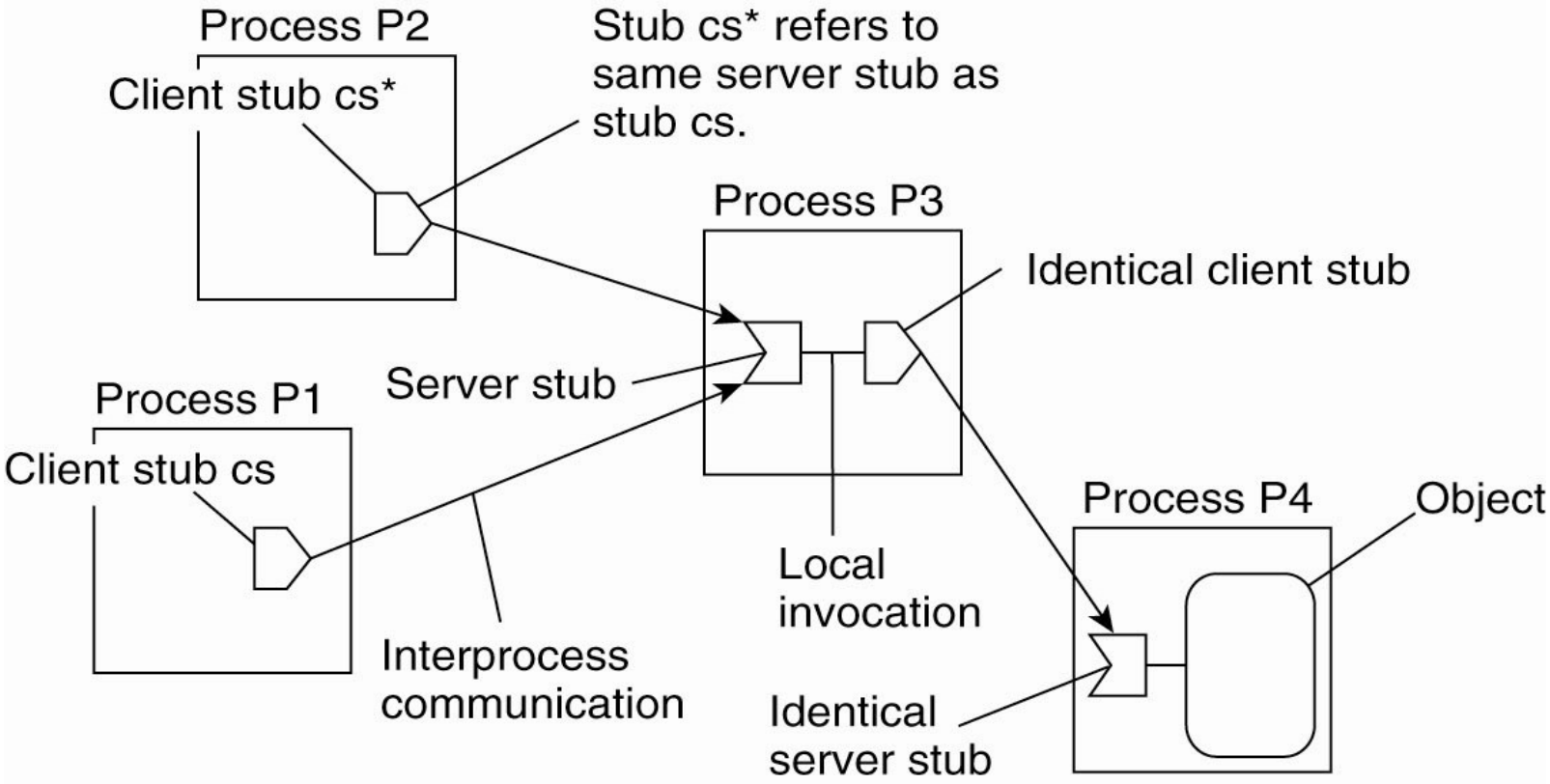
a) Direct, single-level mapping between names and addresses.

b) Two-level mapping using a “location service”.

Simple Solutions Broadcasting

- **Broadcasting** and **Multicasting** technologies.
- Sending out “where are you?” packets …
- *Classic example*: Address Resolution Protocol (ARP) as used by the TCP/IP suite for resolving IP names to underlying networking technology addresses.
- Works well (on LAN’s and other broadcast technologies), but doesn’t scale well.

Forwarding Pointers (1)

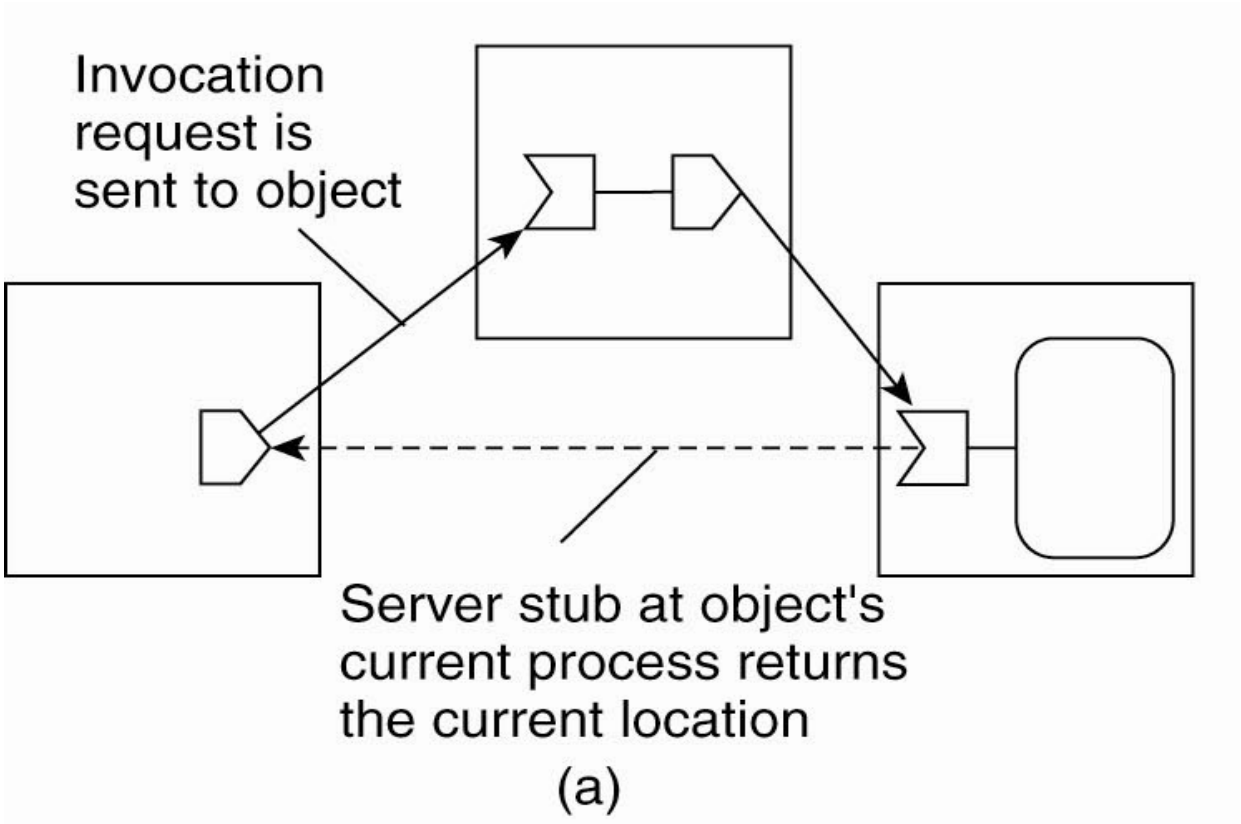


The principle of forwarding pointers using (client stub, server stub) pairs

Disadvantages of Forwarding Pointers

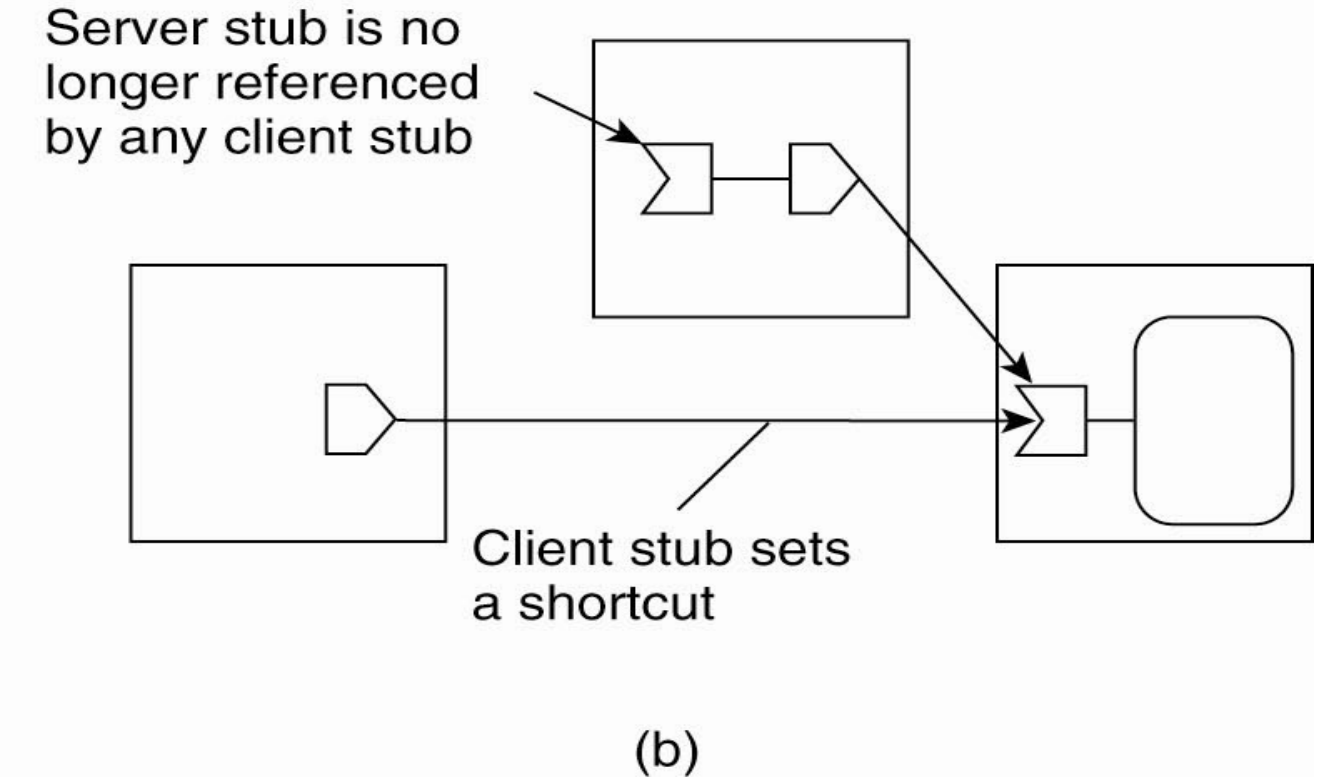
1. A chain can become very long, and the "lookup" eventually becomes prohibitively expensive.
2. All the "intermediate locations" must maintain their chains for "as long as needed" (however long that is).
3. Big vulnerability: *broken links*. Break a link and a forwarded entity is lost …

Forwarding Pointers (2)



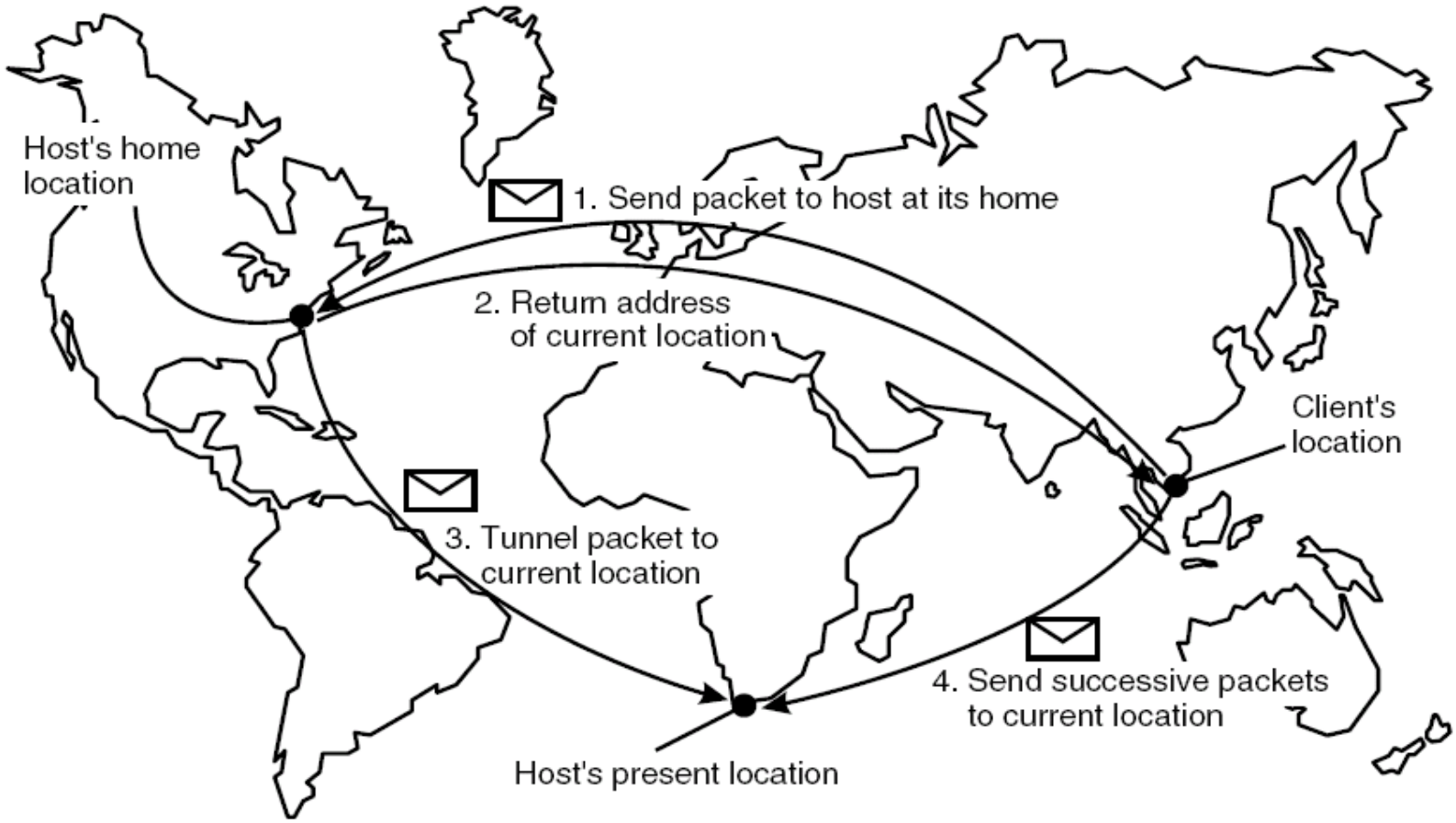
(a) Redirecting a forwarding pointer by storing a shortcut in a client stub

Forwarding Pointers (3)



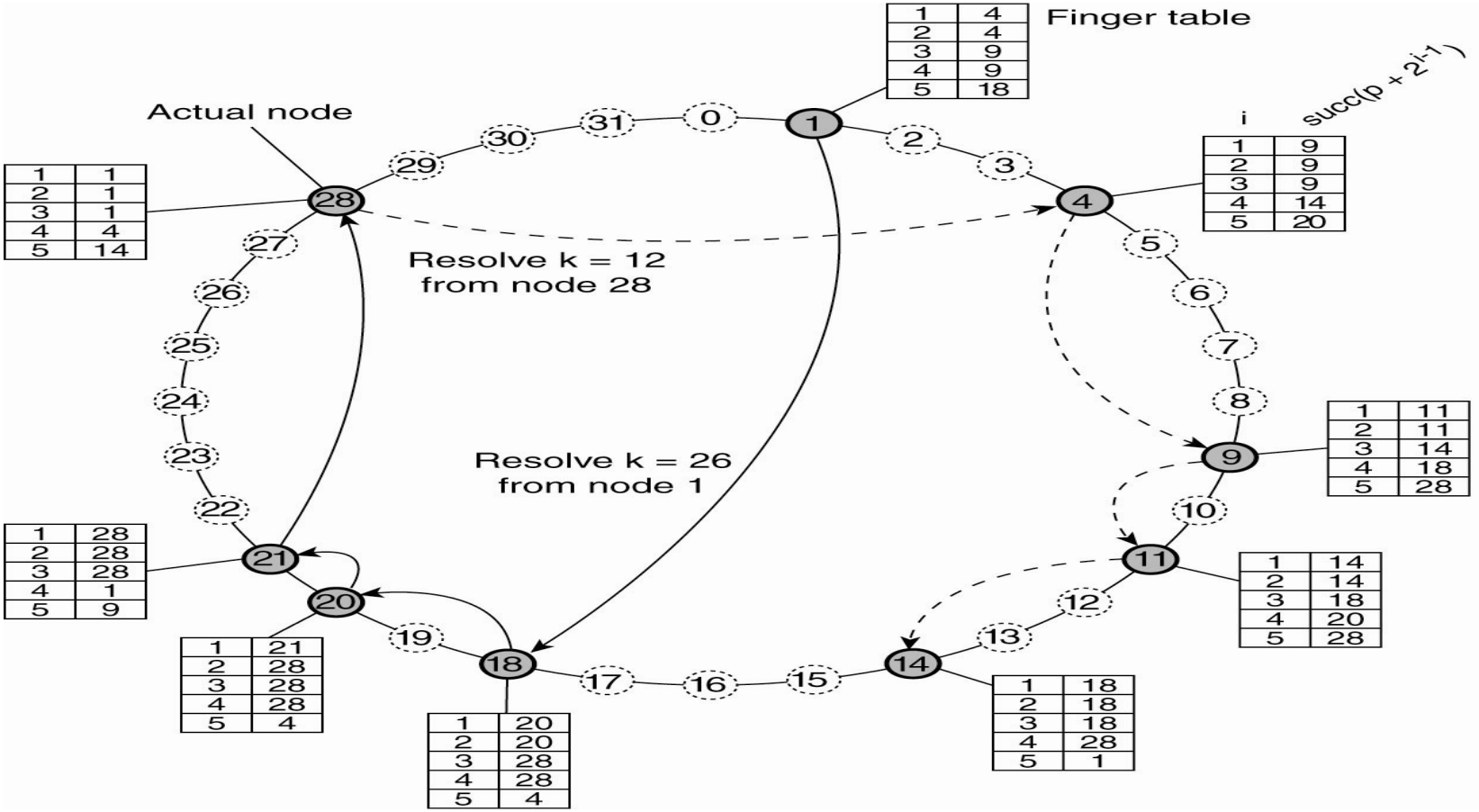
(b) Redirecting a forwarding pointer by storing a shortcut in a client stub

Home-Based Approaches



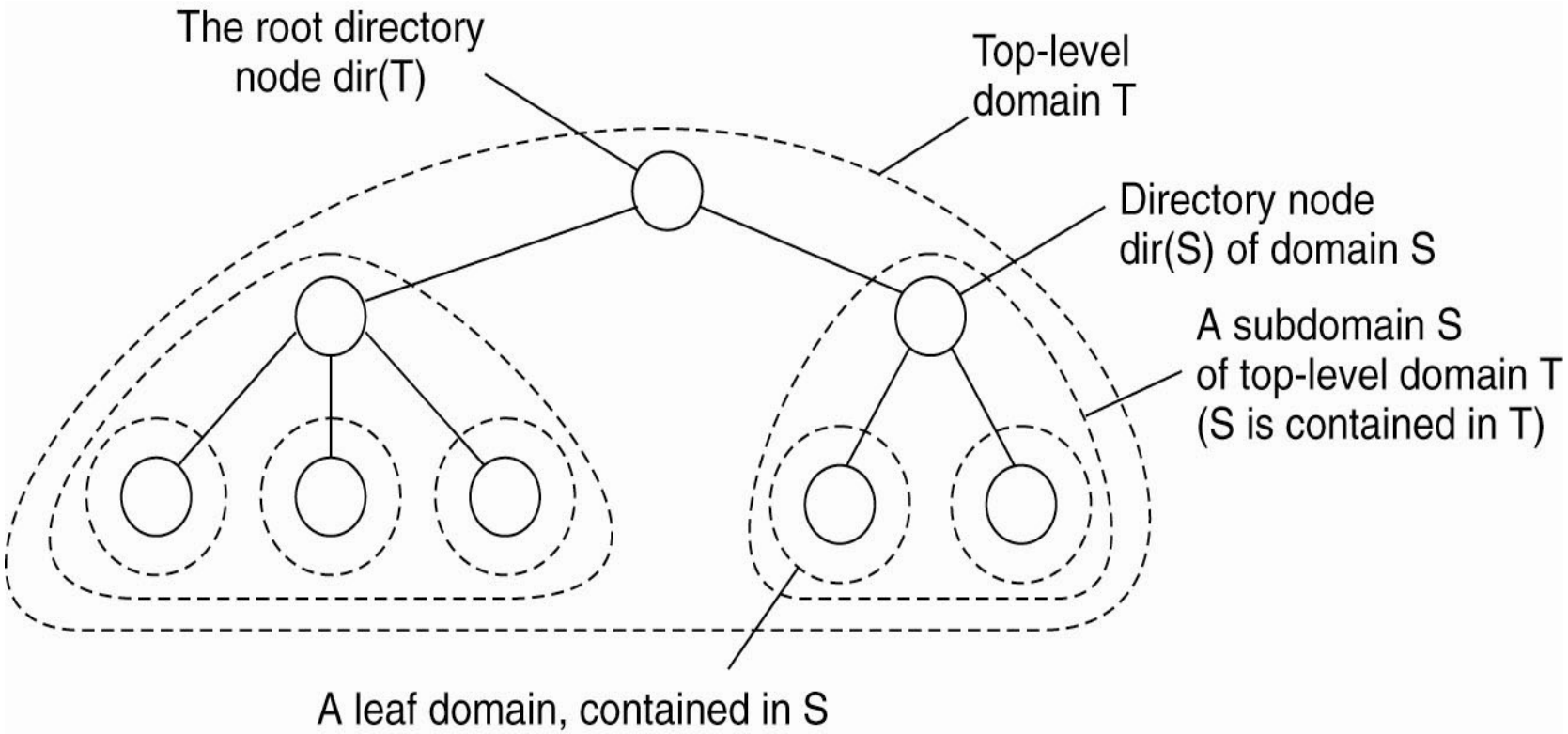
The principle of Mobile IP

Distributed Hash Tables
General Mechanism



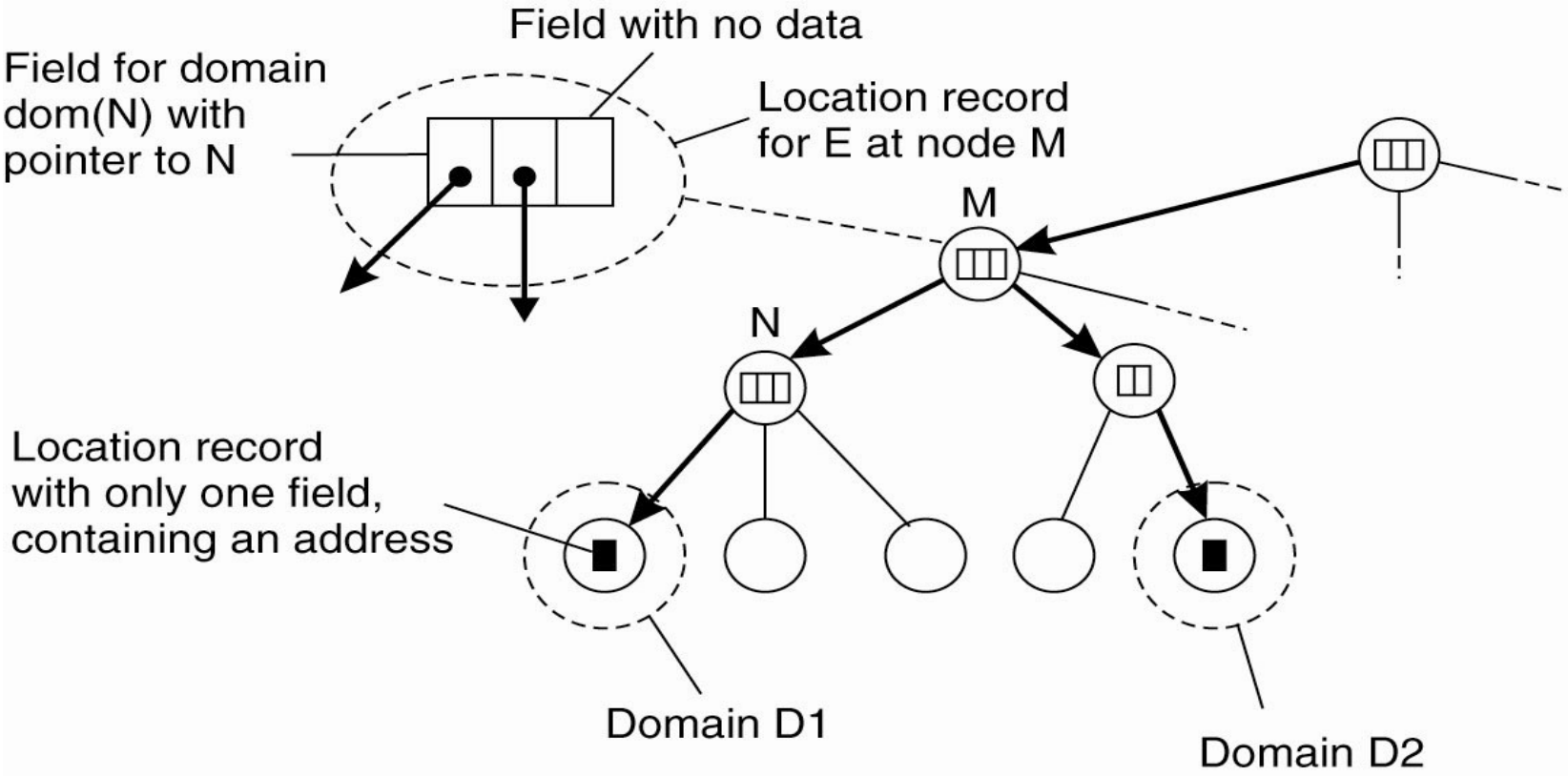
Resolving key 26 from node 1 and key 12 from node 28 in a Chord system

Hierarchical Approaches (1)



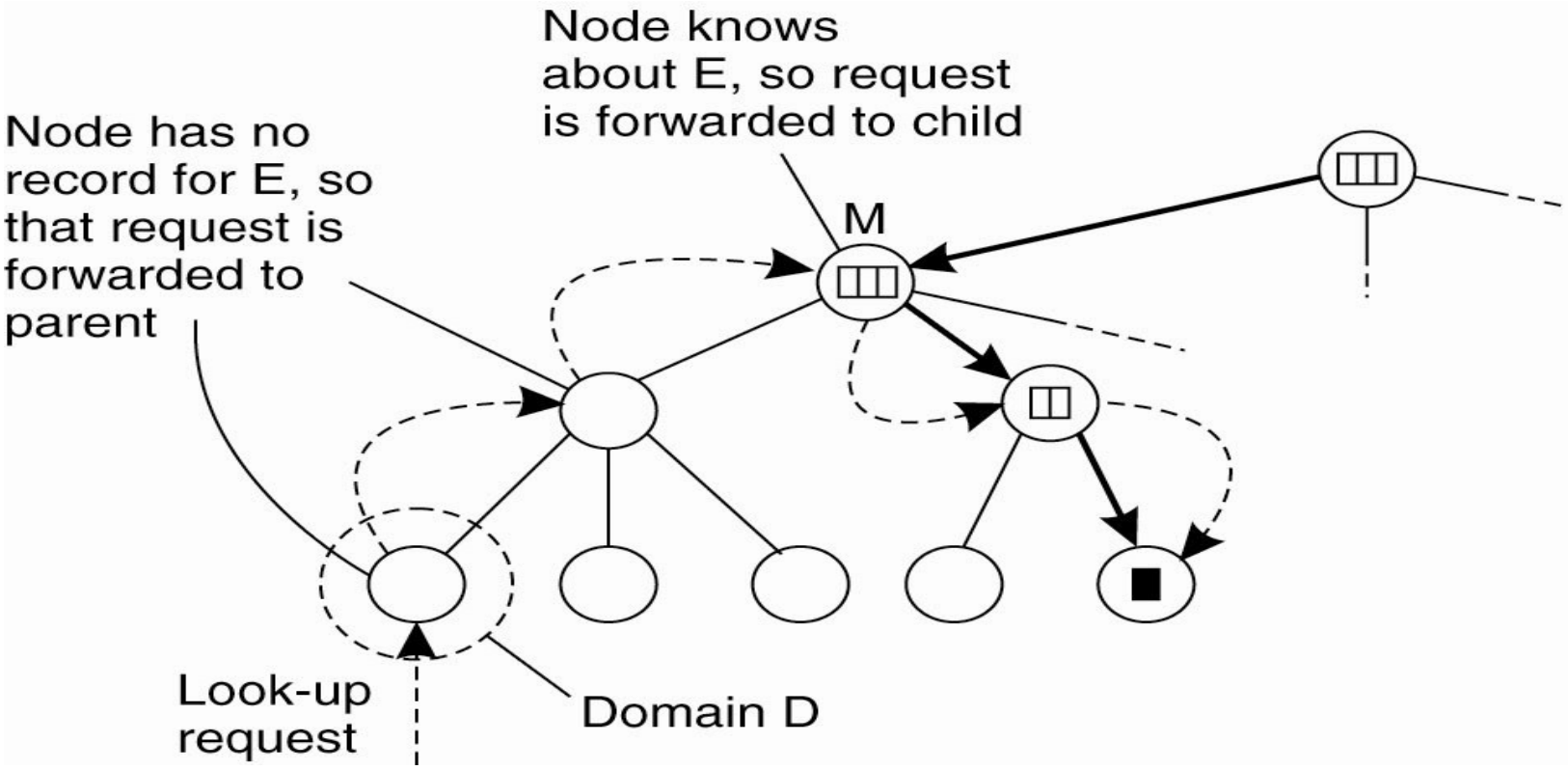
Hierarchical organization of a location service into domains, each having an associated directory node

Hierarchical Approaches (2)



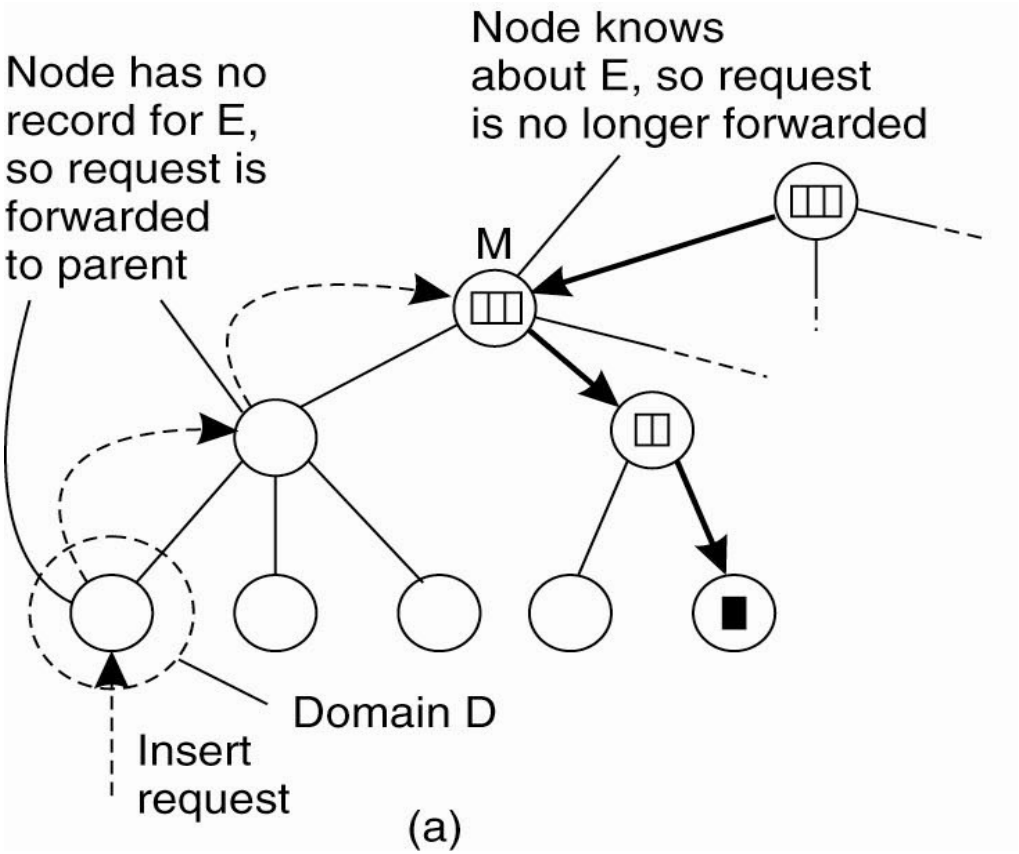
An example of storing information of an entity having two addresses in different leaf domains

Hierarchical Approaches (3)



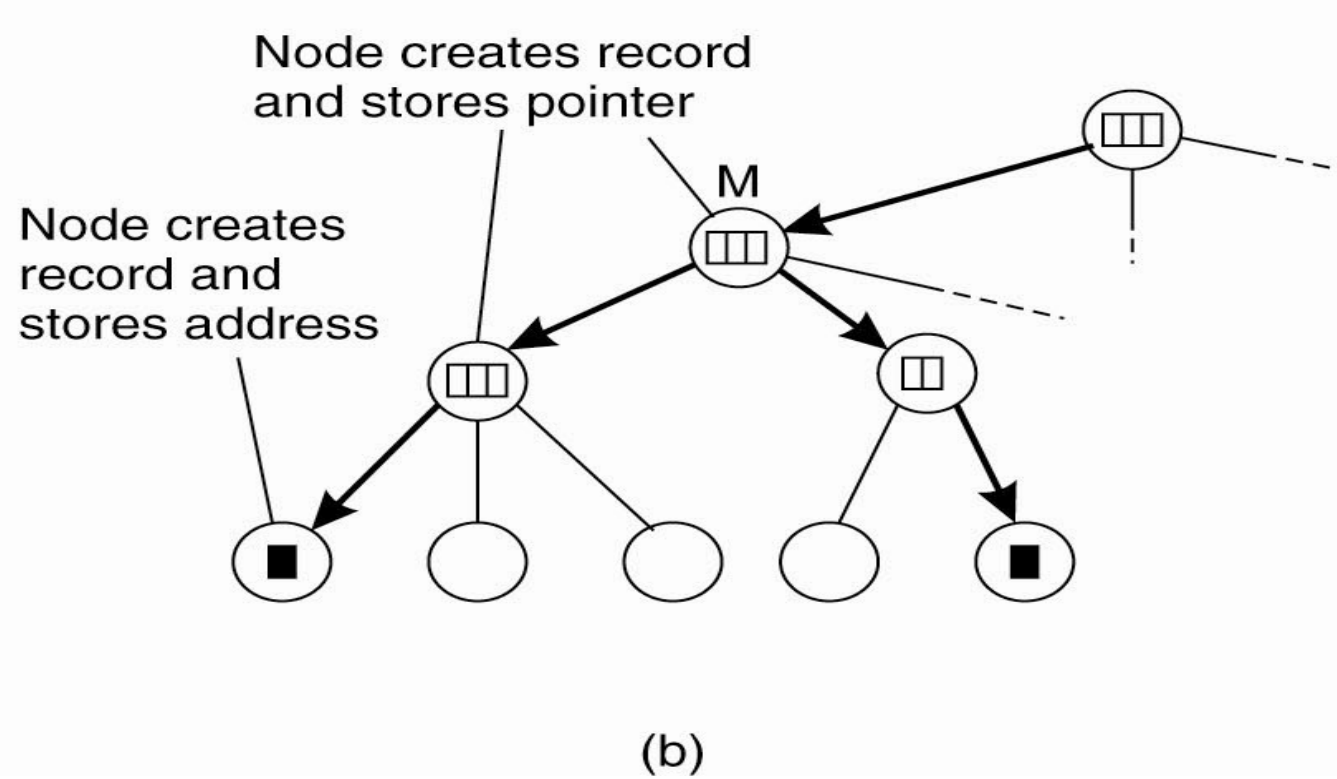
Looking up a location in a hierarchically organized location service

Hierarchical Approaches (4)



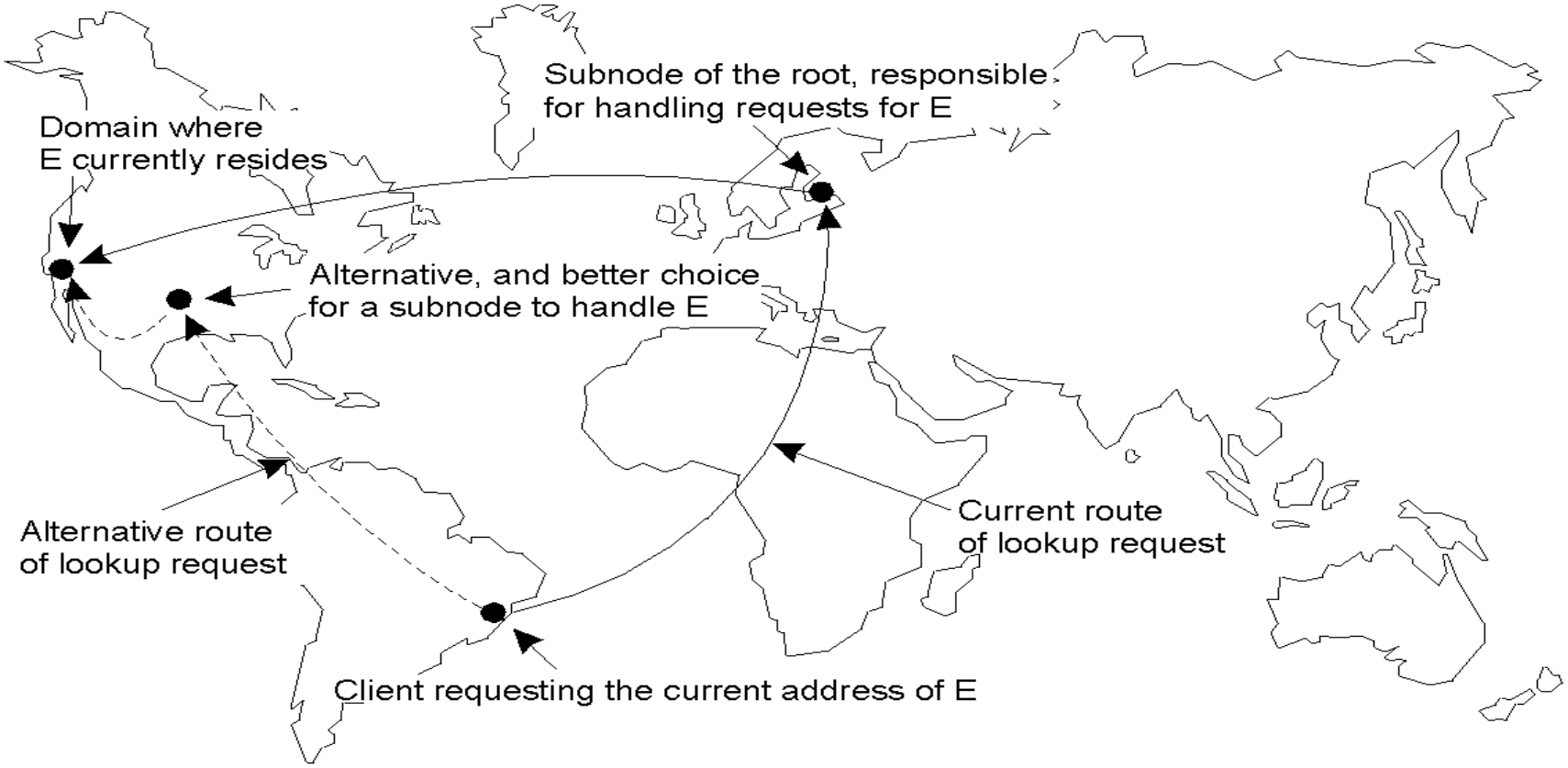
(a) An insert request is forwarded to the first node that knows about entity E

Hierarchical Approaches (5)



(b) A chain of forwarding pointers to the leaf node is created

Scalability Issues with the Hierarchy

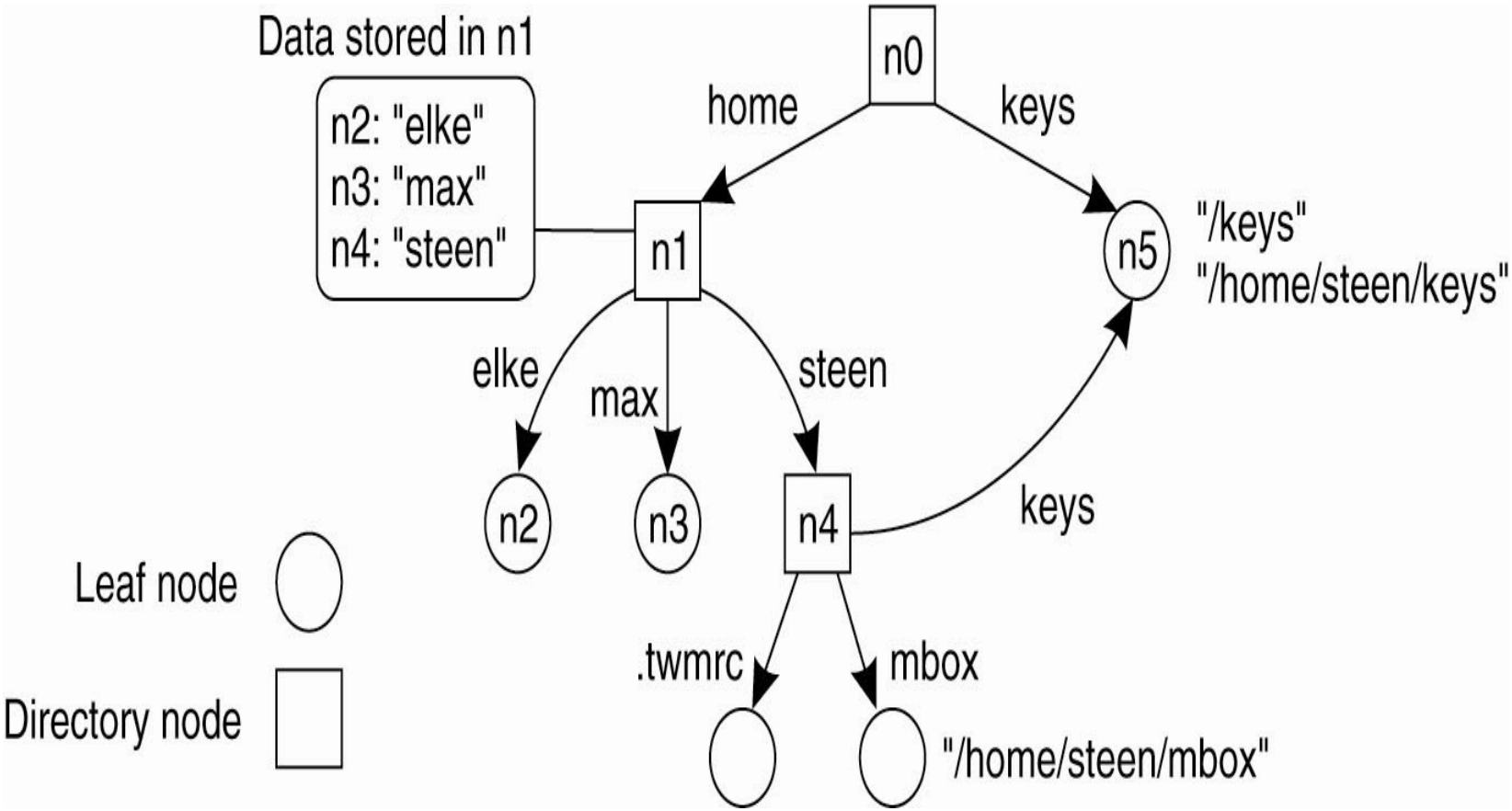


The scalability issues related to uniformly placing subnodes of a partitioned root node across the network covered by a location service

Structured Naming - Name Spaces (1)

- Names are often organized into name spaces.
- Within distributed systems, a name space is represented by a labeled, directed graph with two types of nodes:
  - *leaf nodes*: information on an entity.
  - *directory nodes*: a collection of named outgoing edges (which can lead to any other type of node).
- Each name space has *at least* one **root** node.
- Nodes can be referred to by path names (with absolute or relative).
- File systems are a classic example …

Name Spaces (2)

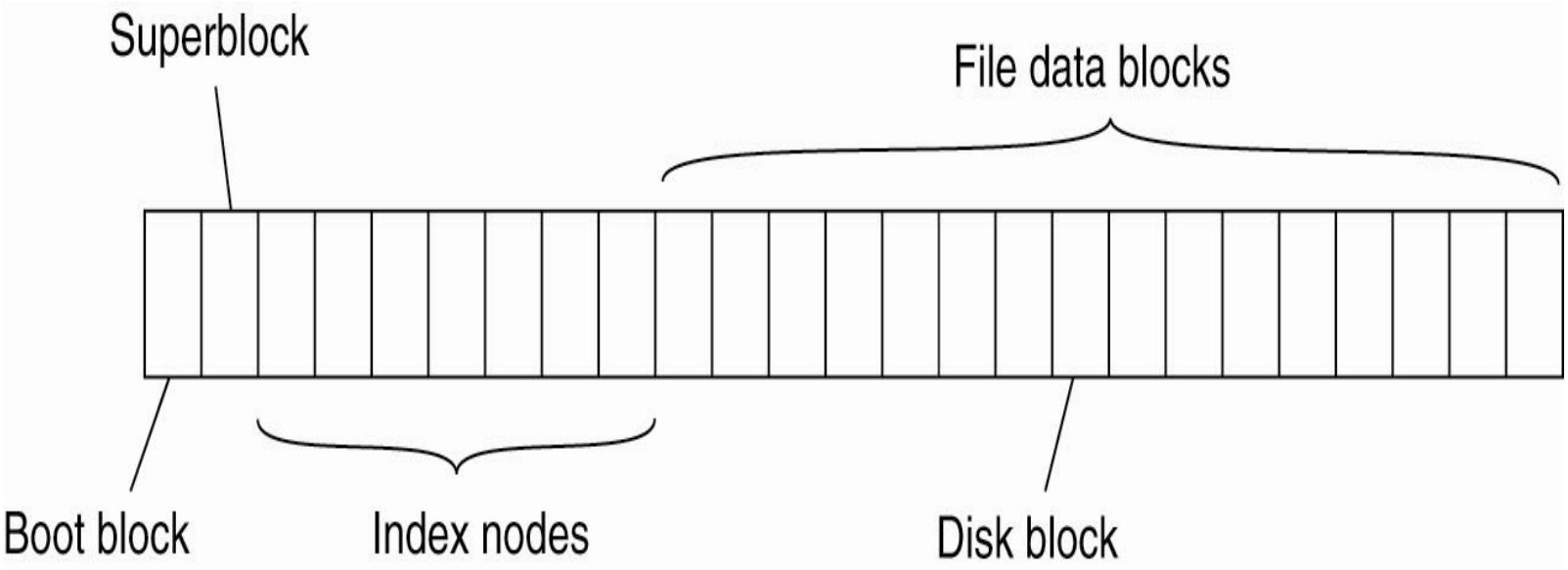


A general naming graph with a single root node

Other Name Space Examples

- UNIX file system implementation (with NFS enhancements to support “remote mounting” of remote file systems).
- SNMP MIB-II (a “sub-namespace” within a much larger name space maintained by the ISO).
- DNS (more on this later).

Unix Name Space

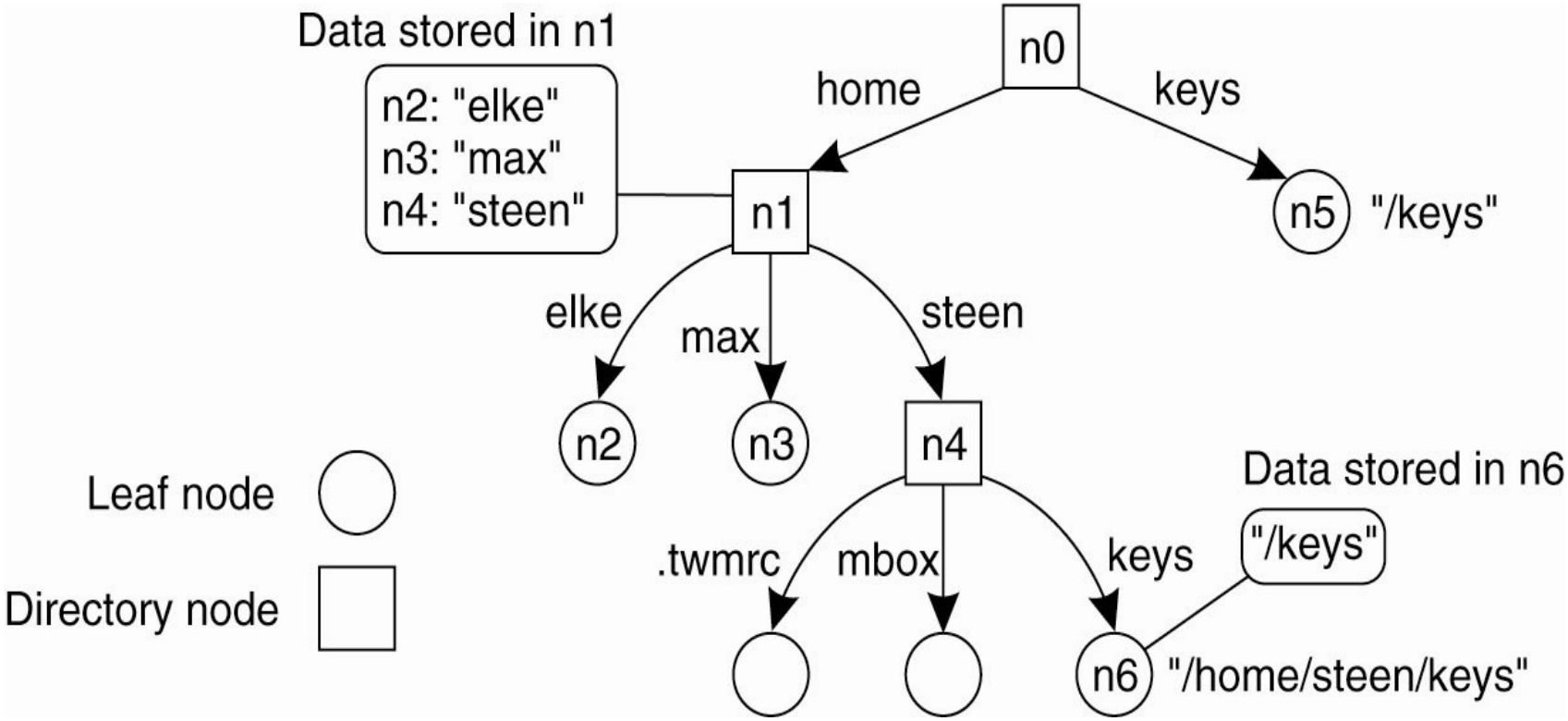


The general organization of the UNIX file system implementation on a logical disk of contiguous disk blocks

Introducing Name Resolution

- The process of looking up information stored in the node given just the path name.
- And assuming, of course, that you know where to start …
- This can be complicated by techniques that have been devised to combine namespaces (such as Sun's NFS mounting and DEC's GNS) …

Linking and Mounting (1)

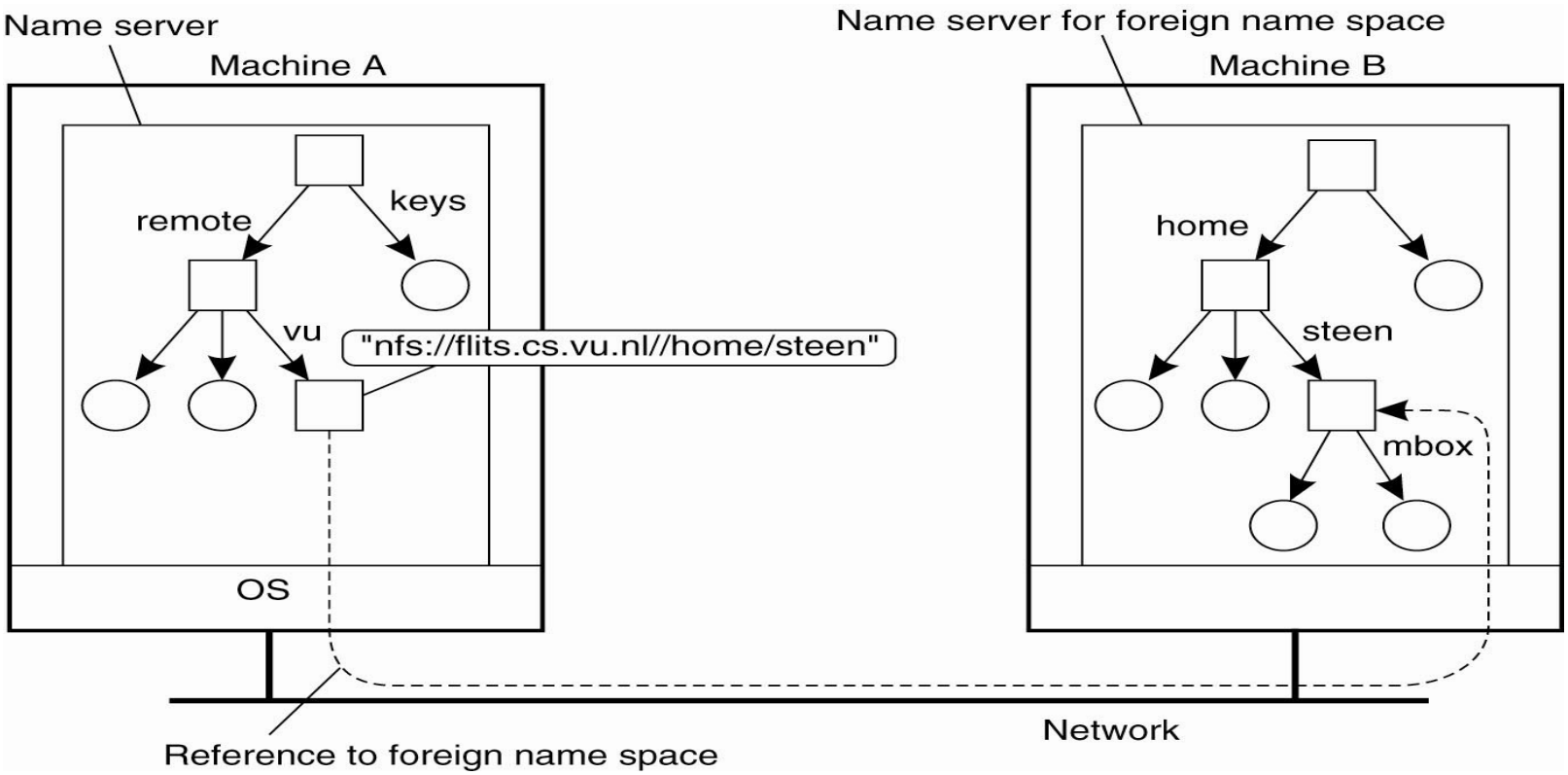


The concept of a symbolic link explained in a naming graph

Linking and Mounting (2)

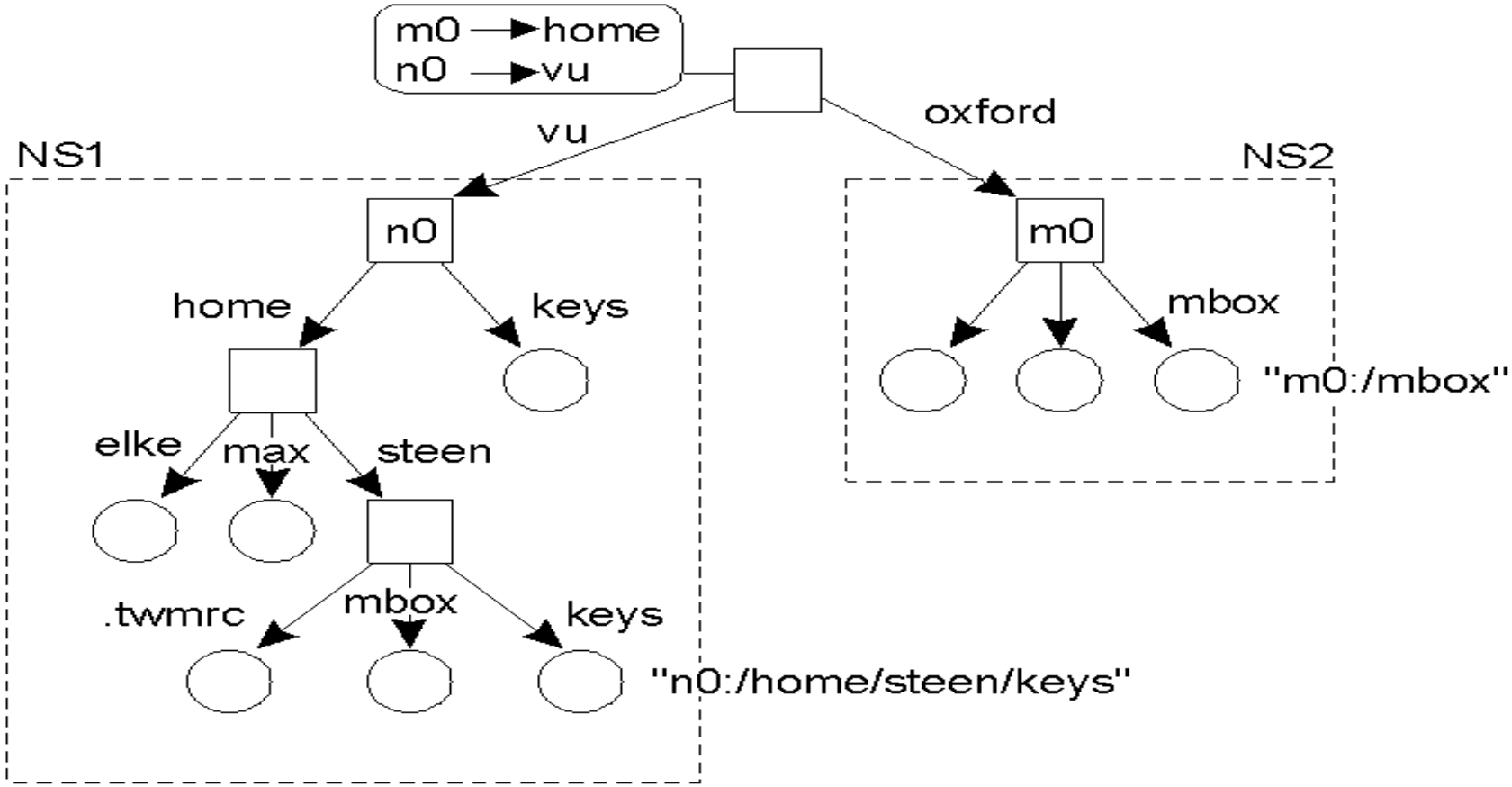
- Information required to mount a foreign name space in a distributed system.
- The name of an access protocol.
- The name of the server.
- The name of the mounting point in the foreign name space.

Linking and Mounting (3)



Mounting remote name spaces through a specific access protocol

Linking and Mounting (4)

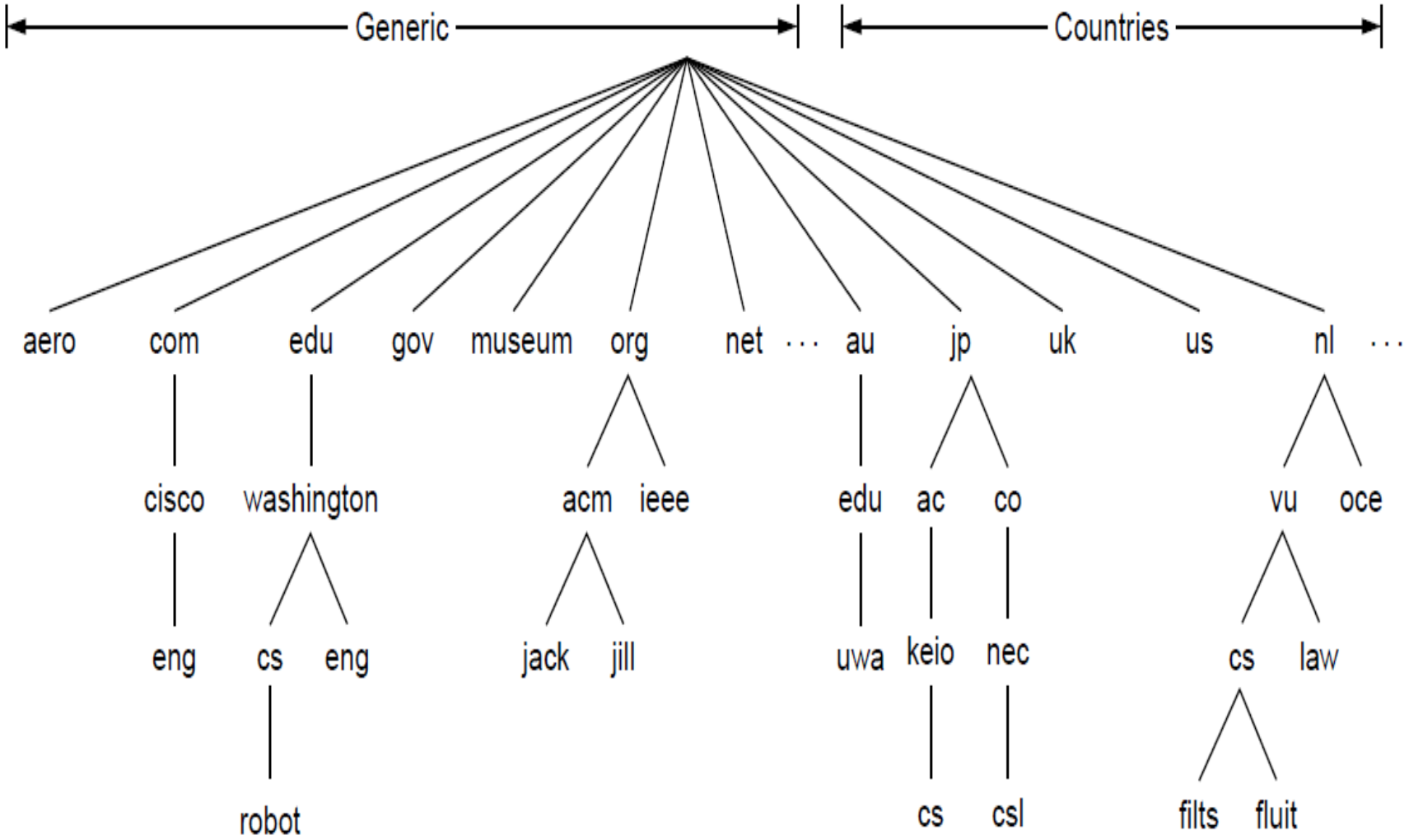


Organization of the DEC "Global Name Service" (adds a new root node and makes existing root nodes its children).

Implementing Name Spaces

- A *Name Service* allows users and processes to add, remove and lookup names.
- Name services are implemented by *Name Servers*.
- On LAN's – a single server usually suffices (think of a 'local' DNS).
- On WAN's – a distributed solution is often more practical (think of the 'global' DNS).
- Often, name spaces (and services) are organized into one of three layers.

A potion of the DNS Name Space



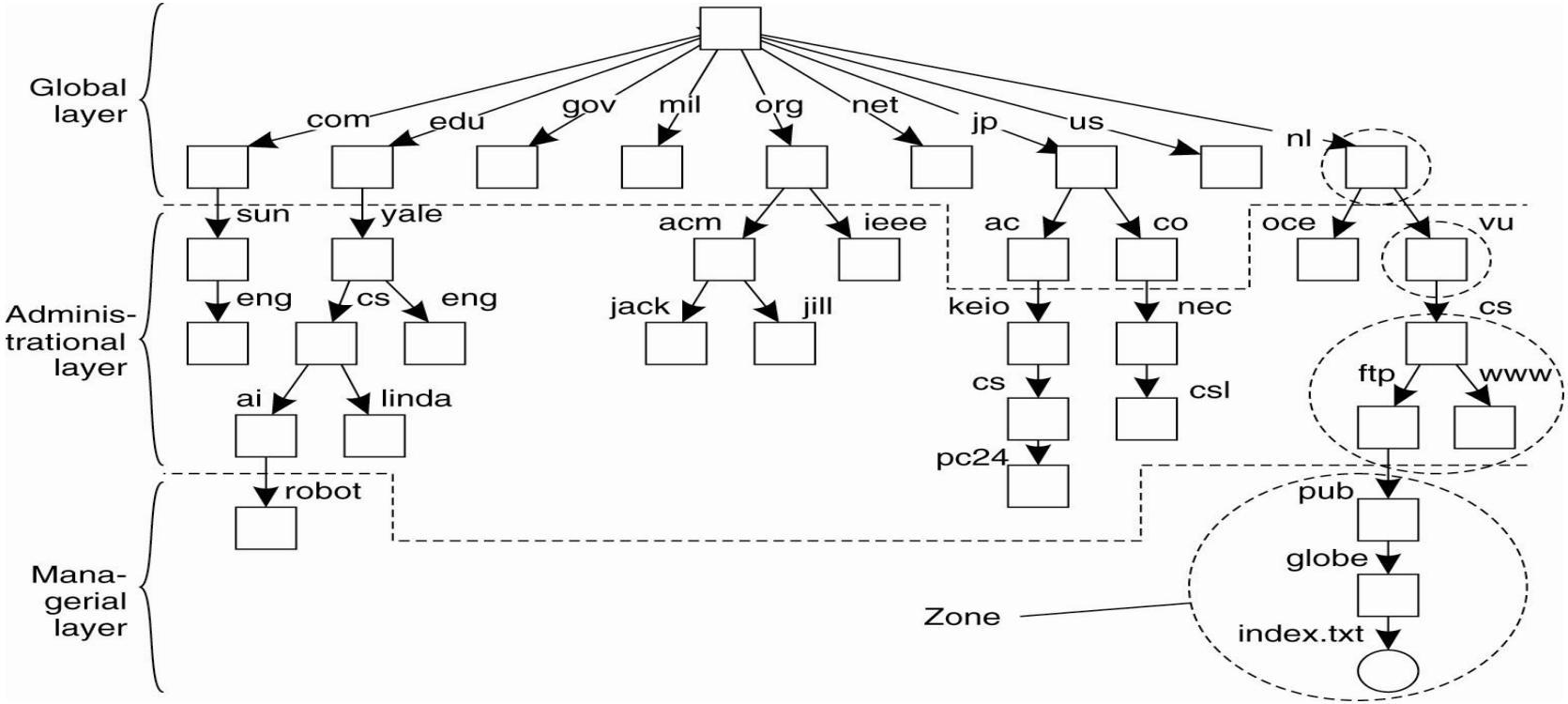
Generic Top-level Domains

Domain	Intended use	Start date	Restricted?
com	Commercial	1985	No
edu	Educational institutions	1985	Yes
gov	Government	1985	Yes
int	International organizations	1988	Yes
mil	Military	1985	Yes
net	Network providers	1985	No
org	Non-profit organizations	1985	No
aero	Air transport	2001	Yes
biz	Businesses	2001	No
coop	Cooperatives	2001	Yes
info	Informational	2002	No
museum	Museums	2002	Yes
name	People	2002	No
pro	Professionals	2002	Yes
cat	Catalan	2005	Yes
jobs	Employment	2005	Yes
mobi	Mobile devices	2005	Yes
tel	Contact details	2005	Yes
travel	Travel industry	2005	Yes
xxx	Sex industry	2010	No

The Three Name Space Layers

- Global Layer: highest level nodes (root); stable; entries change very infrequently.
- Administrational Layer: directory nodes managed by a single organization; relatively stable; although changes can occur more frequently.
- Managerial Layer: nodes change frequently; nodes maintained by users as well as administrators; nodes are the ‘leaf entities’, and can often change.

Name Space Distribution (1)



An example partitioning of the DNS name space, including Internet-accessible files, into three layers

Name Space Distribution (2)

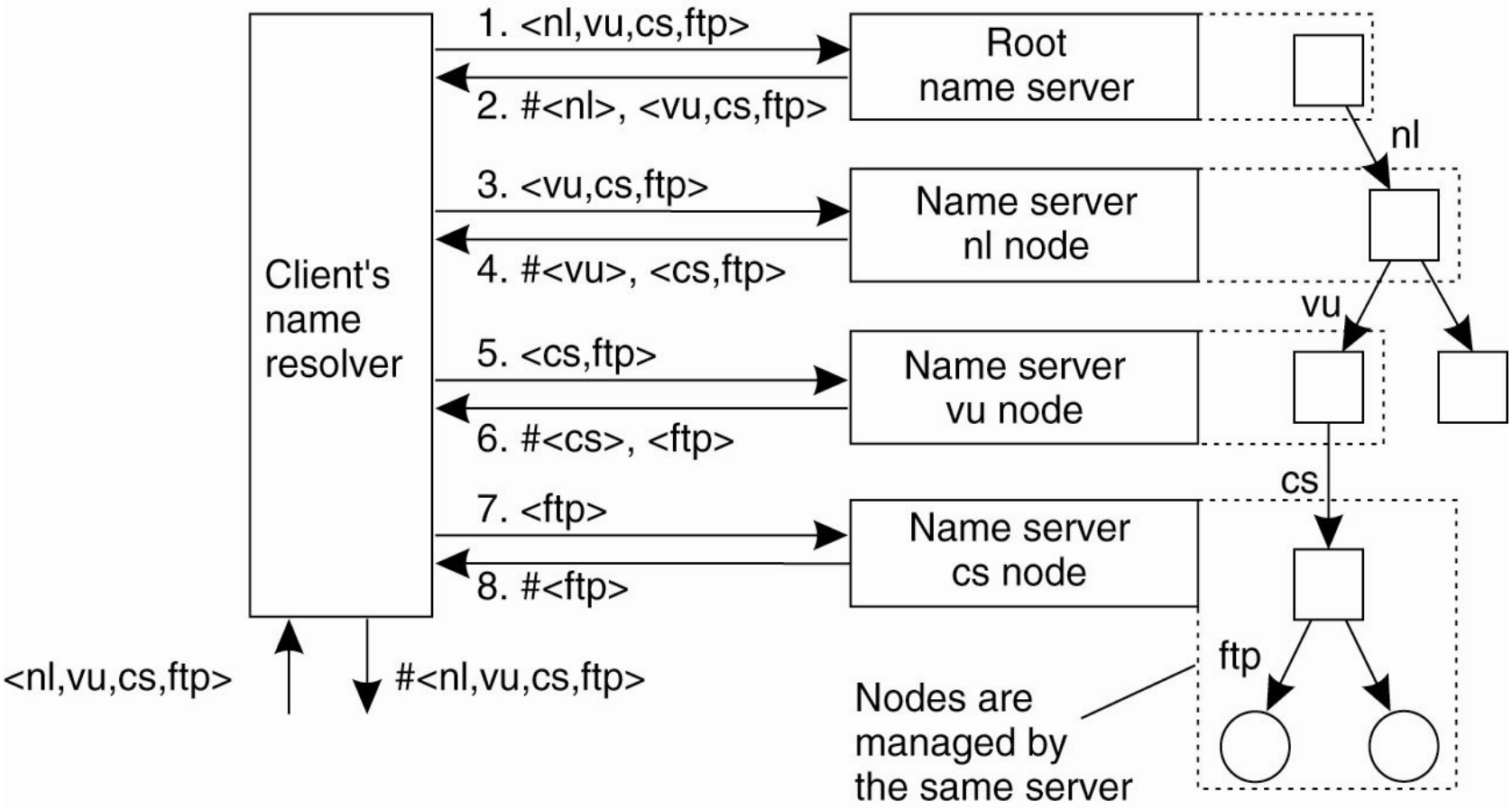
Item	Global	Administrational	Managerial
Geographical scale of network	Worldwide	Organization	Department
Total number of nodes	Few	Many	Vast numbers
Responsiveness to lookups	Seconds	Milliseconds	Immediate
Update propagation	Lazy	Immediate	Immediate
Number of replicas	Many	None or few	None
Is client-side caching applied?	Yes	Yes	Sometimes

A comparison between name servers for implementing nodes from a large-scale name space partitioned into a global layer, an administrational layer, and a managerial layer

More on Name Resolution

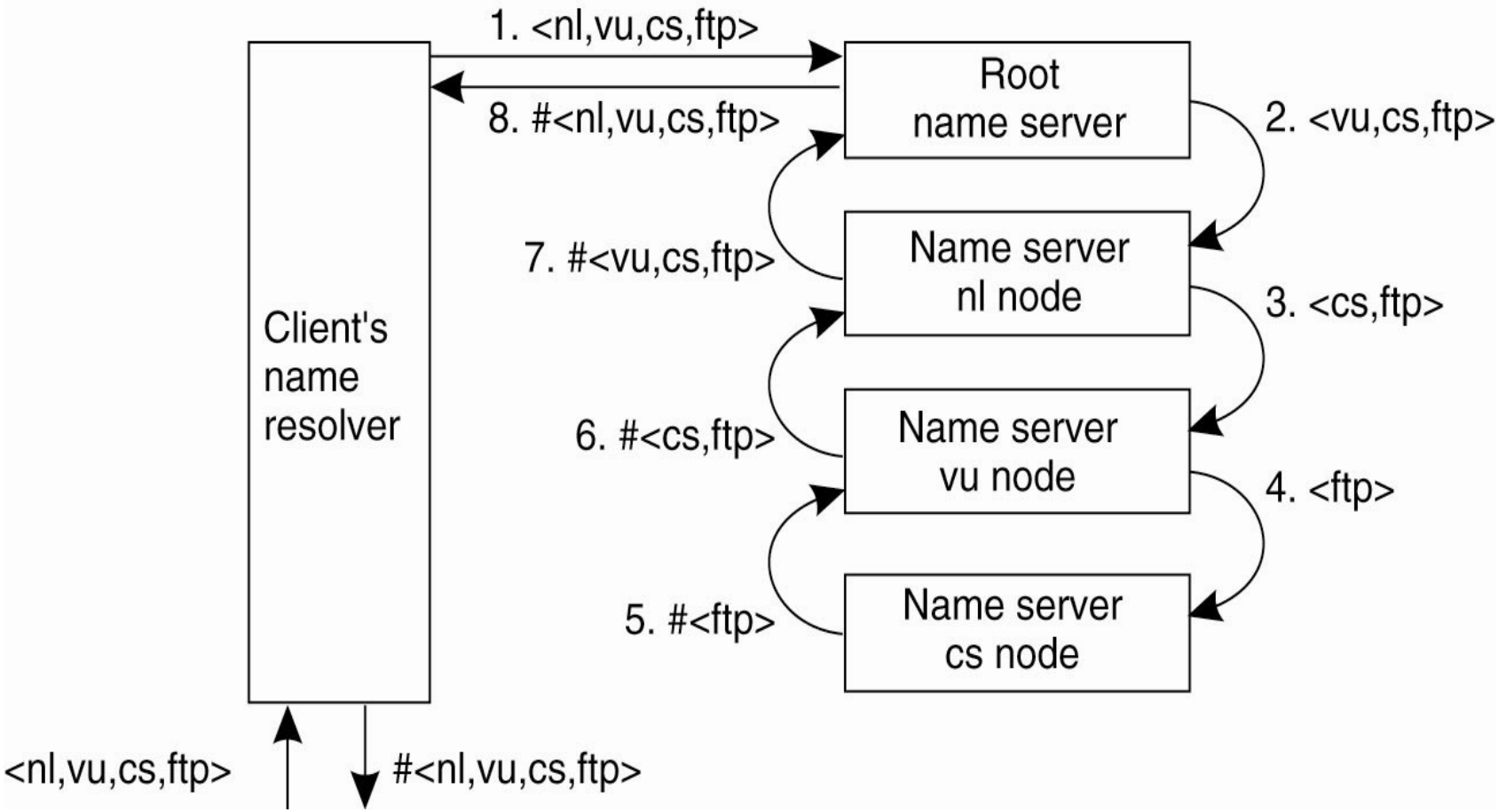
- A “name resolver” provides a local name resolution service to clients – it is responsible for ensuring that the name resolution process is carried out.
- Two Common Approaches:
  1. *Iterative* Name Resolution.
  2. *Recursive* Name Resolution.

Implementation of Name Resolution (1)



The principle of iterative name resolution

Implementation of Name Resolution (2)



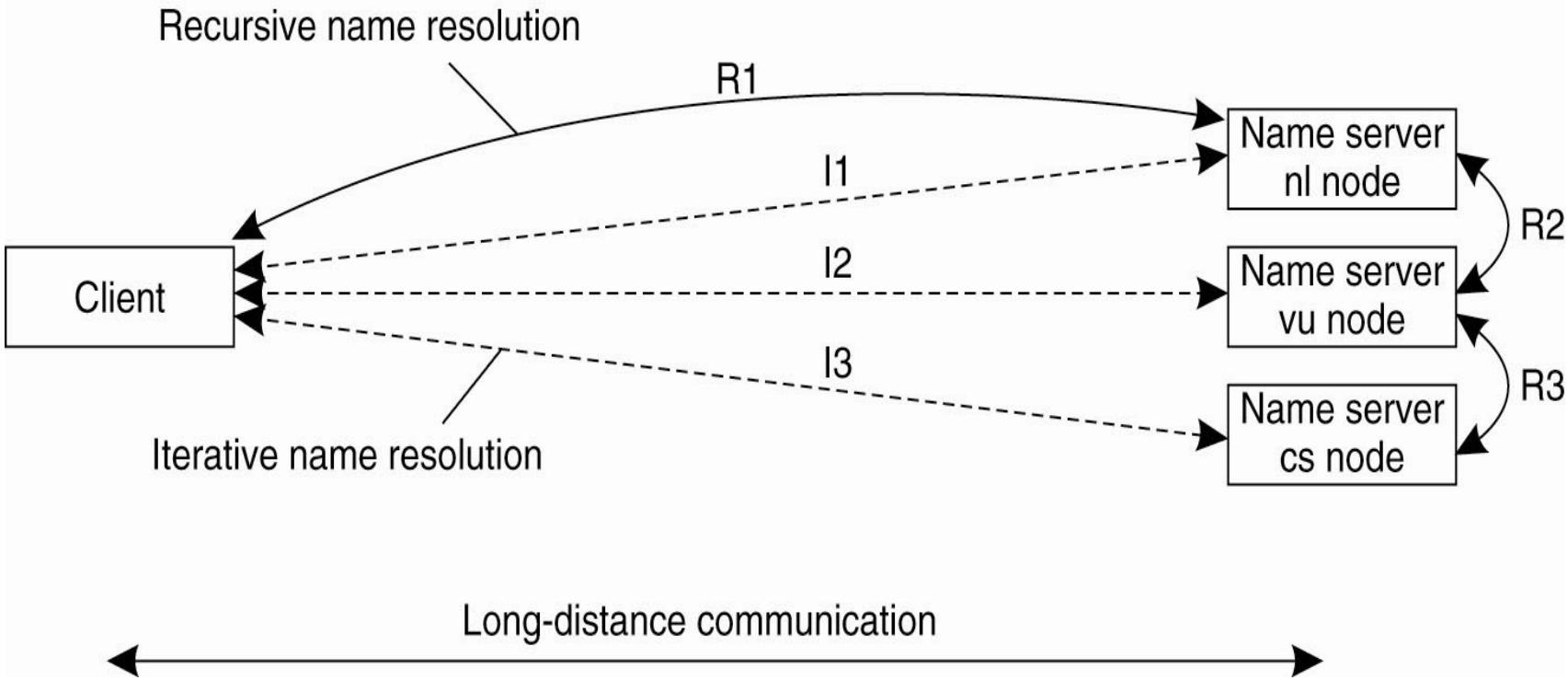
The principle of recursive name resolution

Implementation of Name Resolution (3)

Server for node	Should resolve	Looks up	Passes to child	Receives and caches	Returns to requester
cs	<ftp>	#<ftp>	—	—	#<ftp>
vu	<cs,ftp>	#<cs>	<ftp>	#<ftp>	#<cs> #<cs, ftp>
nl	<vu,cs,ftp>	#<vu>	<cs,ftp>	#<cs> #<cs,ftp>	#<vu> #<vu,cs> #<vu,cs,ftp>
root	<nl,vu,cs,ftp>	#<nl>	<vu,cs,ftp>	#<vu> #<vu,cs> #<vu,cs,ftp>	#<nl> #<nl,vu> #<nl,vu,cs> #<nl,vu,cs,ftp>

Recursive name resolution of *<nl, vu, cs, ftp>*. Name servers cache intermediate results for subsequent lookups.

Geographical Scalability



The comparison between recursive and iterative name resolution with respect to communication costs

Structured Naming - Example: DNS

- “One of the largest distributed naming services in use today.”
- DNS is a classic “rooted tree” naming system.
- Each label (the bit between the ‘.’) must be < 64 chars.
- Each path (the whole thing) must be < 256 chars.
- The root is given the name ‘.’ (although, in practice, the dot is rarely shown nor required).

DNS Names

- A subtree within DNS is referred to as a “domain”.
- A path name is referred to as a “domain name”.
- These can be *relative* or *absolute*.
- DNS server operates at each node (except those at the bottom). Here, the information is organized into “resource records”.

DNS Resource Record Types (1)

Type of record	Associated entity	Description
SOA	Zone	Holds information on the represented zone
A	Host	Contains an IP address of the host this node represents
MX	Domain	Refers to a mail server to handle mail addressed to this node
SRV	Domain	Refers to a server handling a specific service
NS	Zone	Refers to a name server that implements the represented zone
CNAME	Node	Symbolic link with the primary name of the represented node
PTR	Host	Contains the canonical name of a host
HINFO	Host	Holds information on the host this node represents
TXT	Any kind	Contains any entity-specific information considered useful

The most important types of resource records forming the contents of nodes in the DNS name space

DNS Resource Record Types (2)

Type	Meaning	Value
SOA	Start of authority	Parameters for this zone
A	IPv4 address of a host	32-Bit integer
AAAA	IPv6 address of a host	128-Bit integer
MX	Mail exchange	Priority, domain willing to accept email
NS	Name server	Name of a server for this domain
CNAME	Canonical name	Domain name
PTR	Pointer	Alias for an IP address
SPF	Sender policy framework	Text encoding of mail sending policy
SRV	Service	Host that provides it
TXT	Text	Descriptive ASCII text

DNS Implementation (1)

Name	Record type	Record value
cs.vu.nl.	SOA	star.cs.vu.nl. hostmaster.cs.vu.nl. 2005092900 7200 3600 2419200 3600
cs.vu.nl.	TXT	"Vrije Universiteit - Math. & Comp. Sc."
cs.vu.nl.	MX	1 mail.few.vu.nl.
cs.vu.nl.	NS	ns.vu.nl.
cs.vu.nl.	NS	top.cs.vu.nl.
cs.vu.nl.	NS	solo.cs.vu.nl.
cs.vu.nl.	NS	star.cs.vu.nl.
star.cs.vu.nl.	A	130.37.24.6
star.cs.vu.nl.	A	192.31.231.42
star.cs.vu.nl.	MX	1 star.cs.vu.nl.
star.cs.vu.nl.	MX	666 zephyr.cs.vu.nl.
star.cs.vu.nl.	HINFO	"Sun" "Unix"
zephyr.cs.vu.nl.	A	130.37.20.10
zephyr.cs.vu.nl.	MX	1 zephyr.cs.vu.nl.
zephyr.cs.vu.nl.	MX	2 tornado.cs.vu.nl.
zephyr.cs.vu.nl.	HINFO	"Sun" "Unix"

An excerpt from the DNS for the zone *cs.vu.nl*

DNS Implementation (2)

ftp.cs.vu.nl.	CNAME	soling.cs.vu.nl.
www.cs.vu.nl.	CNAME	soling.cs.vu.nl.
soling.cs.vu.nl.	A	130.37.20.20
soling.cs.vu.nl.	MX	1 soling.cs.vu.nl.
soling.cs.vu.nl.	MX	666 zephyr.cs.vu.nl.
soling.cs.vu.nl.	HINFO	"Sun" "Unix"
vucs-das1.cs.vu.nl.	PTR	0.198.37.130.in-addr.arpa.
vucs-das1.cs.vu.nl.	A	130.37.198.0
inkt.cs.vu.nl.	HINFO	"OCE" "Proprietary"
inkt.cs.vu.nl.	A	192.168.4.3
pen.cs.vu.nl.	HINFO	"OCE" "Proprietary"
pen.cs.vu.nl.	A	192.168.4.2
localhost.cs.vu.nl.	A	127.0.0.1

An excerpt from the DNS database for the zone *cs.vu.nl*.

Unstructured Naming
Example: X.500 Naming Service

- A traditional naming service (like DNS) operates much like the Telephone Directory.
  - *Find 'B', then find 'Barry', then find 'Paul', then get the number.*
- With a **directory service**, the client can look for an entity based on a description of its properties instead of its full name. This is more like the Yellow Pages.
  - *Find 'Perl Consultants', obtain the list, search the list, find 'Paul Barry', then get the number.*

More on X.500

- Directory entries in X.500 are roughly equivalent to domain names in DNS.
- The entries are organized as a series of "Attribute/Value Pairings"
- A collection of directory entries is referred to as a Directory Information Base (DIB).

X.500 Attribute/Value Pairings

Attribute	Abbr.	Value
Country	C	NL
Locality	L	Amsterdam
Organization	L	Vrije Universiteit
OrganizationalUnit	OU	Math. & Comp. Sc.
CommonName	CN	Main server
Mail_Servers	--	130.37.24.6, 192.31.231,192.31.231.66
FTP_Server	--	130.37.21.11
WWW_Server	--	130.37.21.11

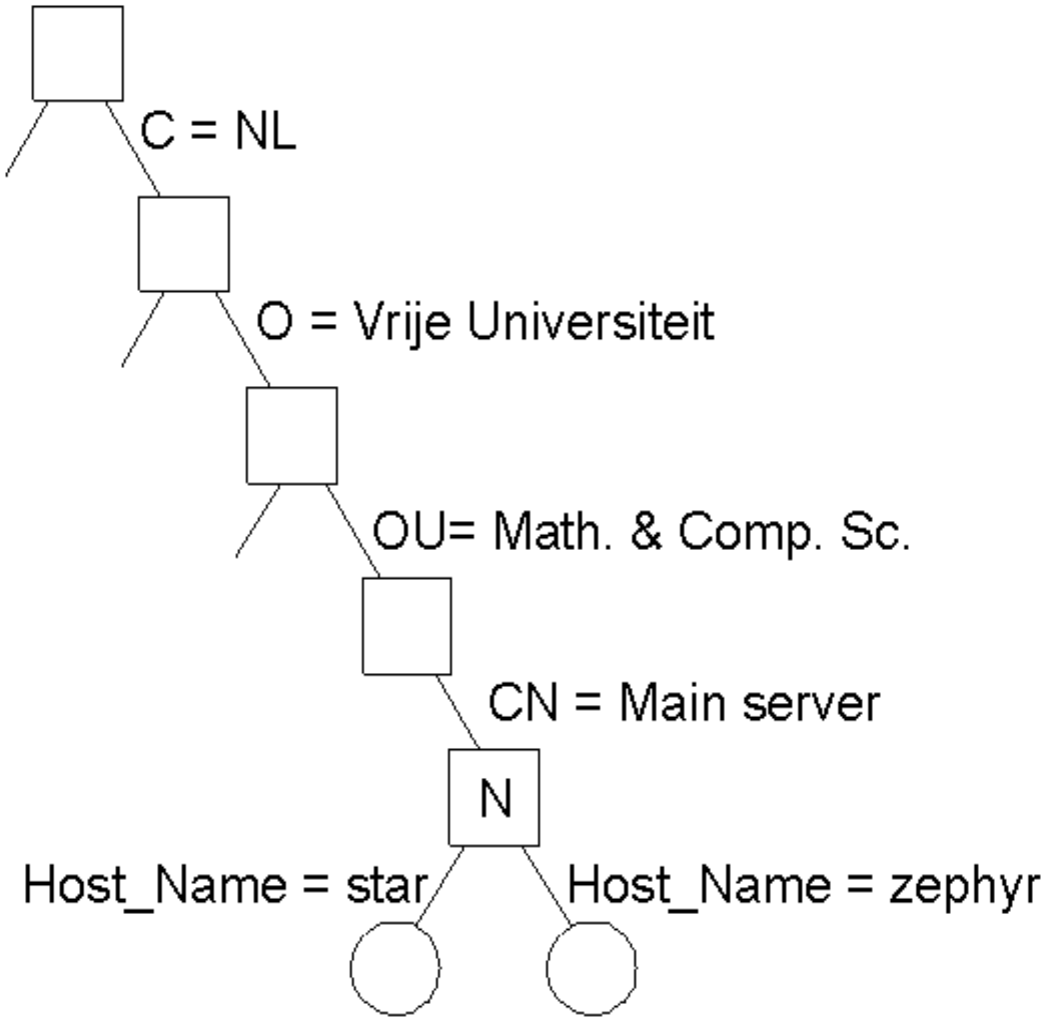
A simple example of a X.500 directory entry using X.500 naming conventions. (Note: both Microsoft and Novell have based their name space technology on the X.500 standard).

X.500 RDN's and DIT's

- A collection of naming attributes is called a Relative Distinguished Name (RDN).
- RDN's can be arranged in sequence into a Directory Information Tree (DIT).
- The DIT is usually partitioned and distributed across several servers (called *Directory Service Agents – DSA*).
- Clients are known as *Directory User Agents – DUA*.

The X.500 DIT

- Part of the X.500 Directory Information Tree (DIT)



The X.500 Name Space

Attribute	Value
Country	NL
Locality	Amsterdam
Organization	Vrije Universiteit
OrganizationalUnit	Math. & Comp. Sc.
CommonName	Main server
Host_Name	star
Host_Address	192.31.231.42

Attribute	Value
Country	NL
Locality	Amsterdam
Organization	Vrije Universiteit
OrganizationalUnit	Math. & Comp. Sc.
CommonName	Main server
Host_Name	zephyr
Host_Address	192.31.231.66

Two directory entries having *Host_Name* as RDN.

X.500 Commentary

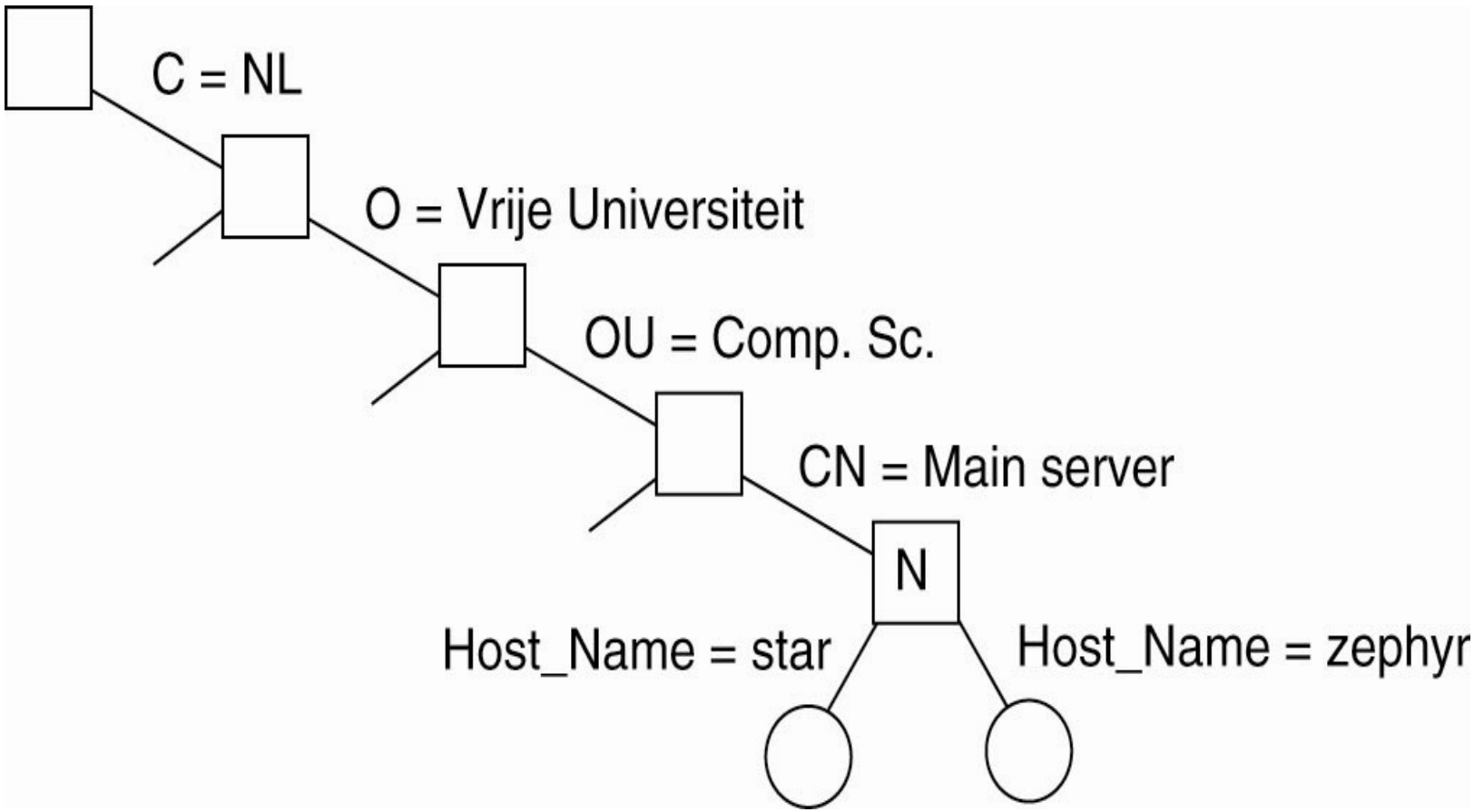
- Searching the DIT is an expensive task.
- Implementing X.500 is not trivial (as is the case with so many ISO standards).
- On the Internet, a similar service is provided by the simpler Lightweight Directory Access Protocol (LDAP), which is regarded as a useful and implementable subset of the X.500 standards.

Unstructured Naming - Example: LDAP (1)

Attribute	Abbr.	Value
Country	C	NL
Locality	L	Amsterdam
Organization	O	Vrije Universiteit
OrganizationalUnit	OU	Comp. Sc.
CommonName	CN	Main server
Mail_Servers	—	137.37.20.3, 130.37.24.6, 137.37.20.10
FTP_Server	—	130.37.20.20
WWW_Server	—	130.37.20.20

A simple example of an LDAP directory entry using LDAP naming conventions

Example: LDAP (2)



(a) Part of a directory information tree

Example: LDAP (3)

Attribute	Value
Country	NL
Locality	Amsterdam
Organization	Vrije Universiteit
OrganizationalUnit	Comp. Sc.
CommonName	Main server
Host_Name	star
Host_Address	192.31.231.42

Attribute	Value
Country	NL
Locality	Amsterdam
Organization	Vrije Universiteit
OrganizationalUnit	Comp. Sc.
CommonName	Main server
Host_Name	zephyr
Host_Address	137.37.20.10

(b)

(b) Two directory entries having *Host_Name* as RDN