Naming

Distributed Software Systems

Naming Entities

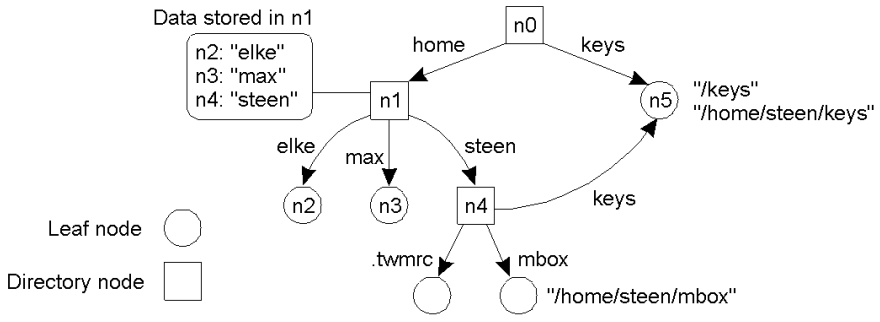
- A name in a distributed system is a string of bits or characters that is used to refer to an entity
- Types of names
  - Address: an access point of an entity
  - Identifiers: a name that uniquely identifies an entity
    - An identifier refers to at most one entity
    - Each entity is referred to by at most one identifier
    - An identifier always refers to the same entity
  - Human-friendly names
  - Location-independent name: a name that is independent from its addresses

Name Spaces and Name Resolution

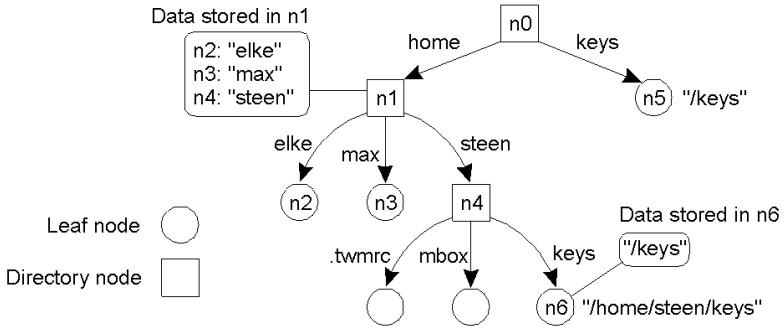
- Names are organized into name spaces
- A name space can be represented as a labeled, directed graph with two types of nodes
  - Leaf nodes and directory nodes
  - Absolute vs relative path names
  - Local names vs global names
- Name Resolution: the process of looking up a name
  - Closure mechanism: knowing where and how to start name resolution

Name Spaces cont'd

A general naming graph with a single root node.



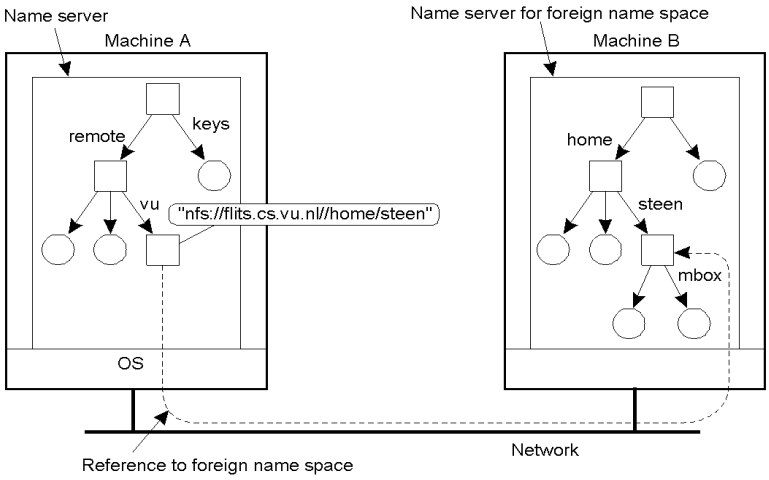
Linking and Mounting



The concept of a symbolic link explained in a naming graph.

5

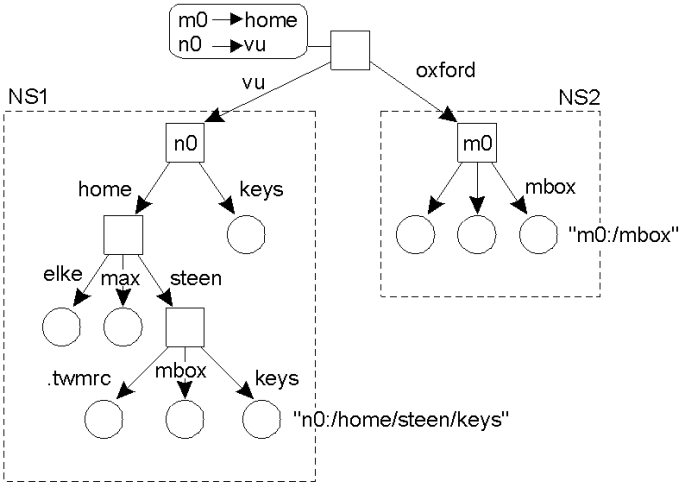
Linking and Mounting



Mounting remote name spaces through a specific process protocol.

6

Merging Name Spaces



Organization of the DEC Global Name Service

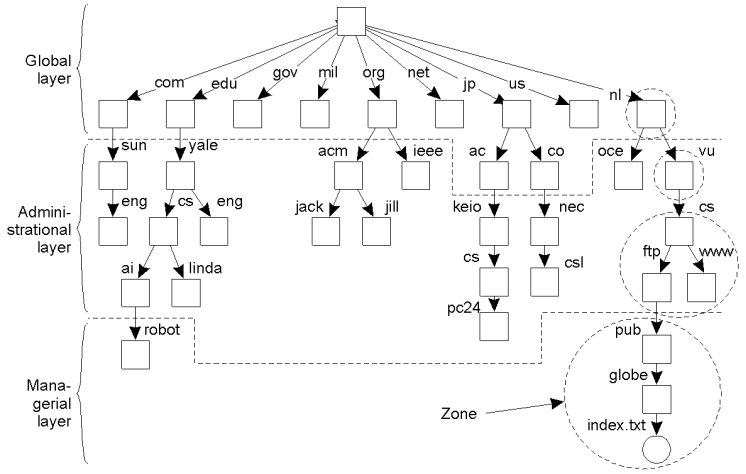
7

Implementing Name Spaces

- Naming service: a service that allows users and processes to add, remove, and lookup names
- Name spaces for large-scale widely distributed systems are typically organized hierarchically
- Three layers used to implement such distributed name spaces
  - Global layer: root node and its children
  - Administrational layer: directory nodes within a single organization
  - Managerial layer

8

Name Space Distribution



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

9

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, as an administrational layer, and a managerial layer.

10

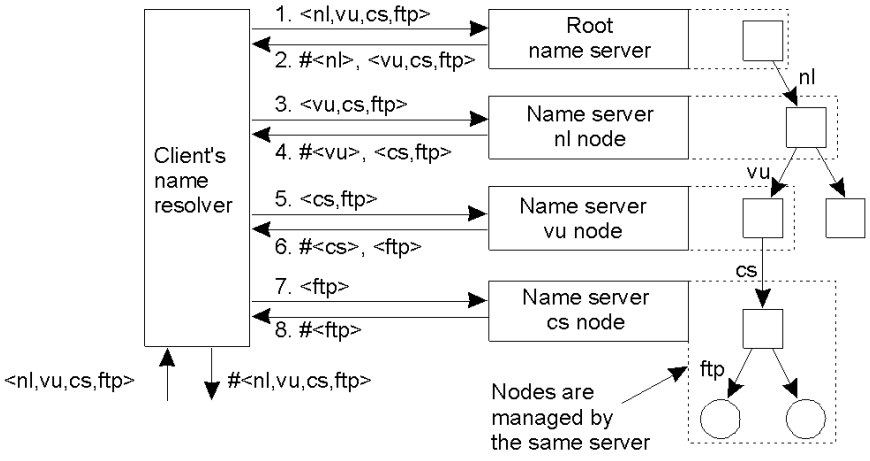
Implementation of Name Resolution

- Iterative vs recursive name resolution
- Recursive name resolution puts a higher performance demand on each name server
  - Too high for global layer name servers
- Advantages of recursive name resolution
  - Caching is more effective
  - Communication costs may be reduced

11

Implementation of Name Resolution (1)

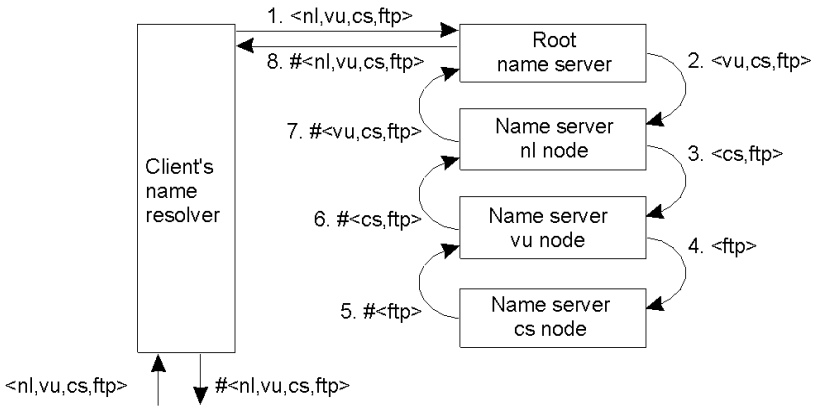
The principle of iterative name resolution.



12

Implementation of Name Resolution (2)

The principle of recursive name resolution.

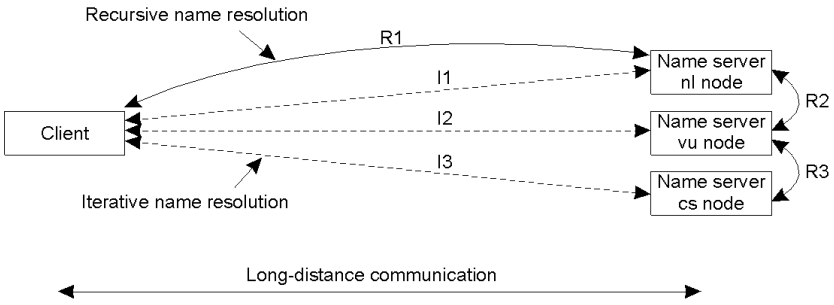


Implementation of Name Resolution (3)

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>
ni	<vu,cs,ftp>	#<vu>	<cs,ftp>	#<cs> #<cs,ftp>	#<vu> #<vu,cs> #<vu,cs,ftp>
root	<ni,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.

Implementation of Name Resolution (4)



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

15

Example System: DNS

- Domain Name System (DNS)
  - Host name to IP address translation
  - Name space organized as a hierarchical rooted tree
    - Name space divided into non-overlapping **zones**
  - Name servers implement the global and administrational layers
    - Managerial layer not part of DNS
    - Each zone has a name server, which is typically replicated
    - Updates take place at the primary name server for a zone
      - Secondary name servers request the primary name server to transfer its content

16

The DNS Name Space

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.

17

DNS Implementation (1)

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

Name	Record type	Record value
cs.vu.nl	SOA	star (1999121502,7200,3600,2419200,86400)
cs.vu.nl	NS	star.cs.vu.nl
cs.vu.nl	NS	top.cs.vu.nl
cs.vu.nl	NS	solo.cs.vu.nl
cs.vu.nl	TXT	"Vrije Universiteit - Math. & Comp. Sc."
cs.vu.nl	MX	1 zephyr.cs.vu.nl
cs.vu.nl	MX	2 tornado.cs.vu.nl
cs.vu.nl	MX	3 star.cs.vu.nl
star.cs.vu.nl	HINFO	Sun Unix
star.cs.vu.nl	MX	1 star.cs.vu.nl
star.cs.vu.nl	MX	10 zephyr.cs.vu.nl
star.cs.vu.nl	A	130.37.24.6
star.cs.vu.nl	A	192.31.231.42
zephyr.cs.vu.nl	HINFO	Sun Unix
zephyr.cs.vu.nl	MX	1 zephyr.cs.vu.nl
zephyr.cs.vu.nl	MX	2 tornado.cs.vu.nl
zephyr.cs.vu.nl	A	192.31.231.66
www.cs.vu.nl	CNAME	soling.cs.vu.nl
ftp.cs.vu.nl	CNAME	soling.cs.vu.nl
soling.cs.vu.nl	HINFO	Sun Unix
soling.cs.vu.nl	MX	1 soling.cs.vu.nl
soling.cs.vu.nl	MX	10 zephyr.cs.vu.nl
soling.cs.vu.nl	A	130.37.24.11
laser.cs.vu.nl	HINFO	PC MS-DOS
laser.cs.vu.nl	A	130.37.30.32
vucs-das.cs.vu.nl	PTR	0.26.37.130.in-addr.arpa
vucs-das.cs.vu.nl	A	130.37.26.0

DNS Implementation (2)

Name	Record type	Record value
cs.vu.nl	NS	solo.cs.vu.nl
solo.cs.vu.nl	A	130.37.21.1

Part of the description for the *vu.nl* domain which contains the *cs.vu.nl* domain.

19

Example System: X.500

- An example of a directory service
  - Analogy: X.500 is to DNS as the yellow pages are to a telephone book
- Each directory entry is made up of a collection of (attribute, value) pairs
  - Attributes can be single-valued or multiple-valued
- Collection of all directory entries is called a Directory Information Base (DIB)
- Each entry has a globally unique name formed by a sequence of naming attributes (Relative Distinguished Names or RDN)
- Lookup operations
  - Read: Read a single record given its pathname in the Directory Information Tree (DIT), I.e. hierarchical name space formed by directory entries
  - List: return the names of all outgoing edges of a given node in the DIT

20

The X.500 Name Space (1)

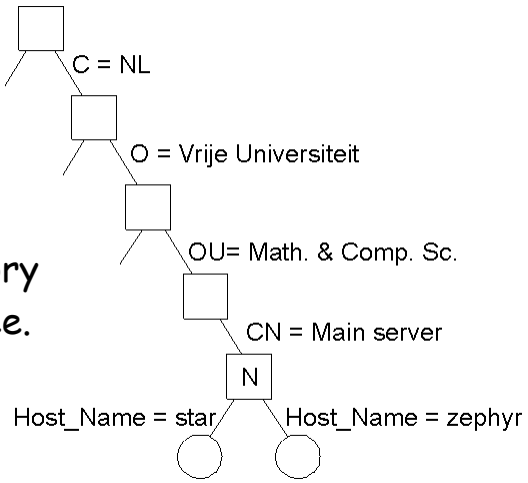
Attribute	Abbr.	Value
Country	C	NL
Locality	L	Amsterdam
Organization	L	Vrije Universiteit
OrganizationalUnit	OU	Math. & Comp. Sc.
CommonName	CN	Main server
Mail_Servers	--	130.37.24.6, 192.31.231,192.31.231.66
FTP_Server	--	130.37.21.11
WWW_Server	--	130.37.21.11

A simple example of a X.500 directory entry using X.500 naming conventions.

21

The X.500 Name Space (2)

Part of the directory information tree.



22

The X.500 Name Space (3)

Two directory entries having *Host_Name* as RDN.

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

23

X.500 implementation

- Similar to DNS
  - The DIT is partitioned and distributed across several servers known as Directory Service Agents (DSA)
  - Clients are represented by name resolvers called Directory User Agents (DUA)
- Differences from DNS
  - Operations for searching through a DIB given a set of criteria that attributes should meet
  - Searching is an expensive operation since several leaf nodes of a DIT will need to be accessed
- Lightweight Directory Access Protocol (LDAP) is an application-level protocol that is a simplified version of X.500
  - Becoming a de facto standard for Internet-based directory services

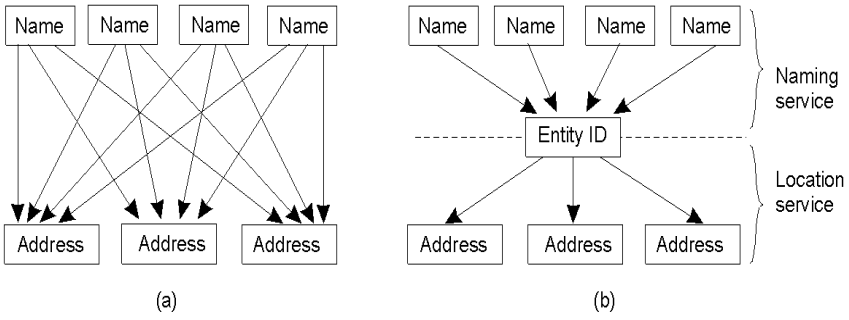
24

Locating Mobile Entities

- Consider an entity that changes its location
  - E.g. ftp.cs.vu.nl moves to another domain
    - Cannot change name
  - Two choices
    - Record the address of the new machine in the DNS database for cs.vu.nl
      - If name changes again, DNS entry will have to be changed again
    - Record the name of the new machine in the database, I.e. use a symbolic link
      - Inefficient lookups
- Traditional naming services such as DNS cannot cope well with mobile entities
  - Problems arise because of the direct mapping between human-friendly names and the address of entities

25

Naming versus Locating Entities



a) Direct, single level mapping between names and addresses.
b) Two-level mapping using identities.

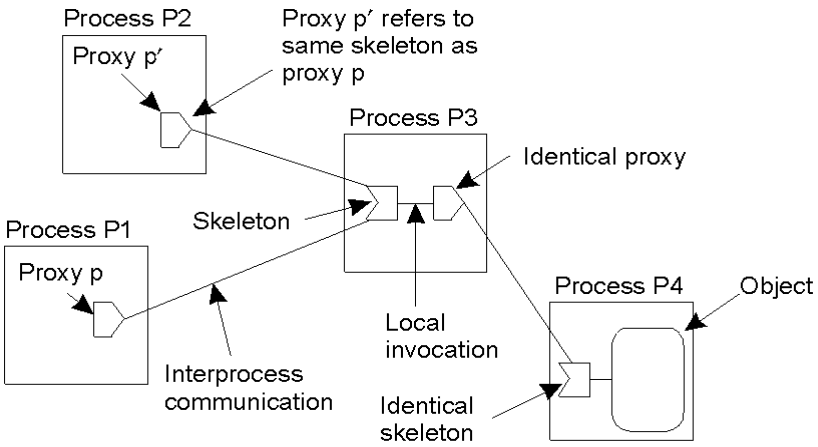
26

Locating Entities

- Simple Solutions that work in a LAN environment
  - Broadcasting & Multicasting
    - Message containing identifier of the entity is broadcast; machine with an access point for the entity replies with the address of the access point
      - ARP protocol for finding the data-link address of a machine given the IP address
  - Forwarding pointers
    - When an entity moves from A to B, it leaves behind a reference to its new location at B
- Home-based Approaches
  - Home agent keeps track of current location of mobile entity
- Hierarchical Approaches

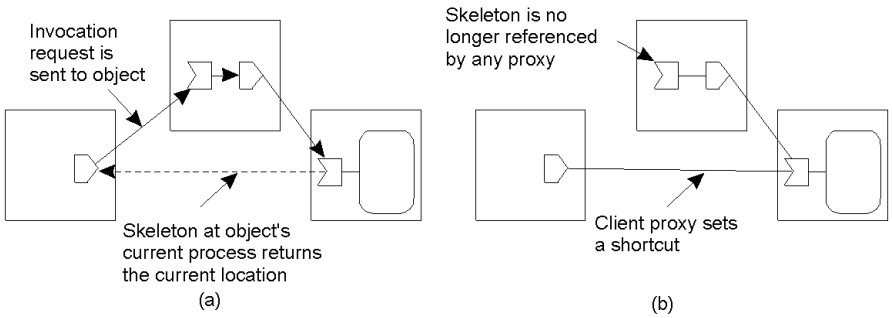
27

Forwarding Pointers (1)



28

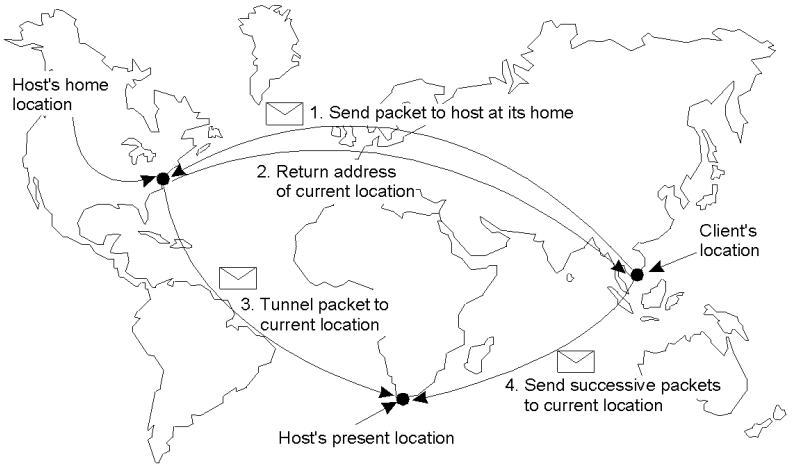
Forwarding Pointers (2)



Redirecting a forwarding pointer, by storing a shortcut in a proxy.

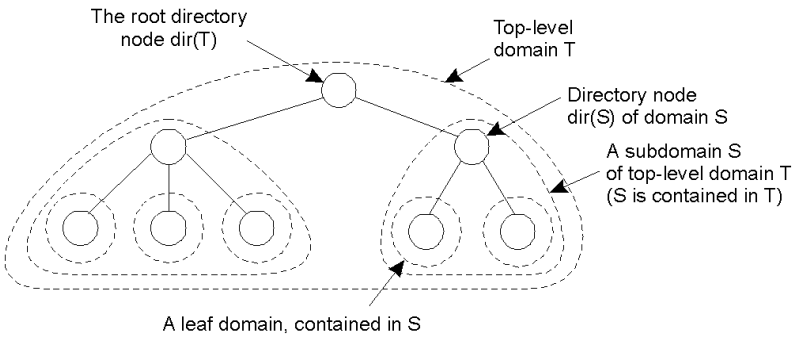
29

Home-Based Approaches



30

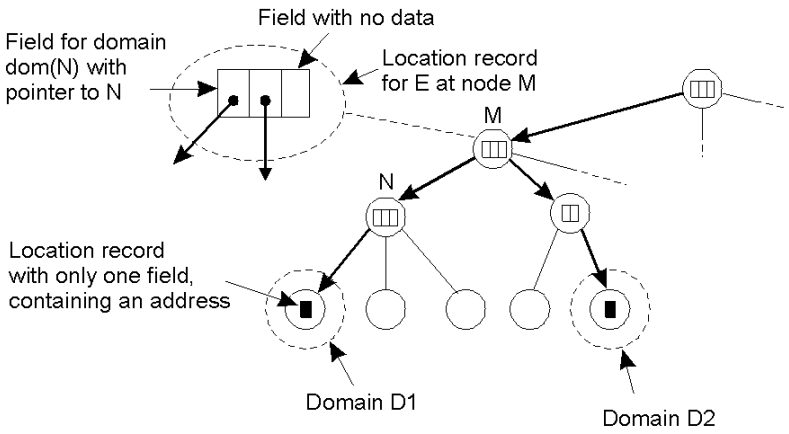
Hierarchical Approaches (1)



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

31

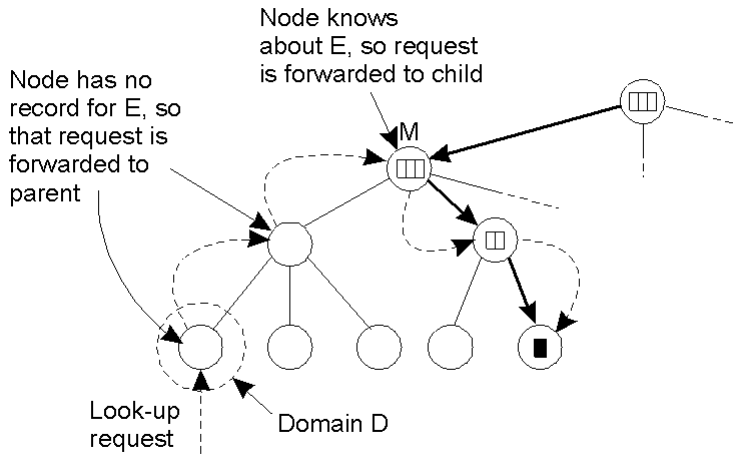
Hierarchical Approaches (2)



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

32

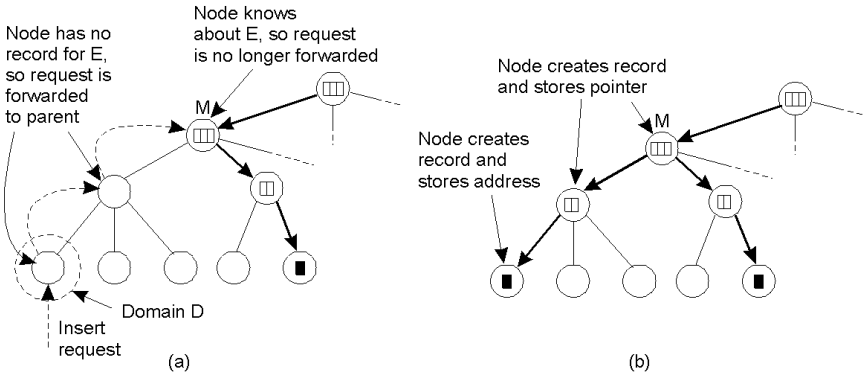
Hierarchical Approaches (3)



Looking up a location in a hierarchically organized location service.

33

Hierarchical Approaches (4)

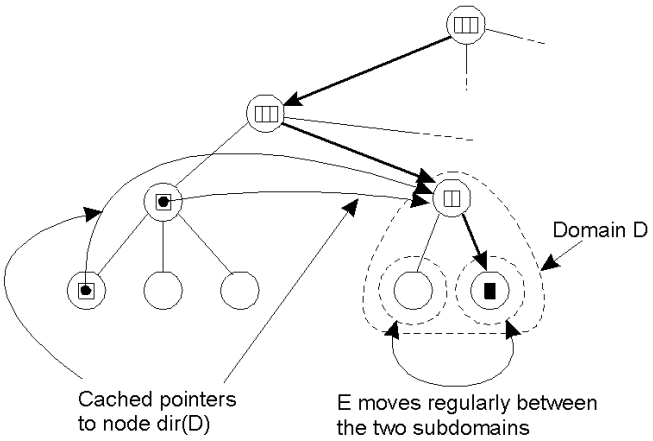


a) An insert request is forwarded to the first node that knows about entity *E*.

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

34

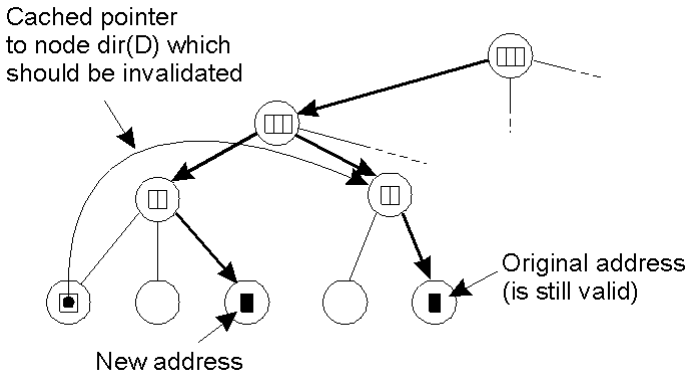
Pointer Caches (1)



Caching a reference to a directory node of the lowest-level domain in which an entity will reside most of the time.

35

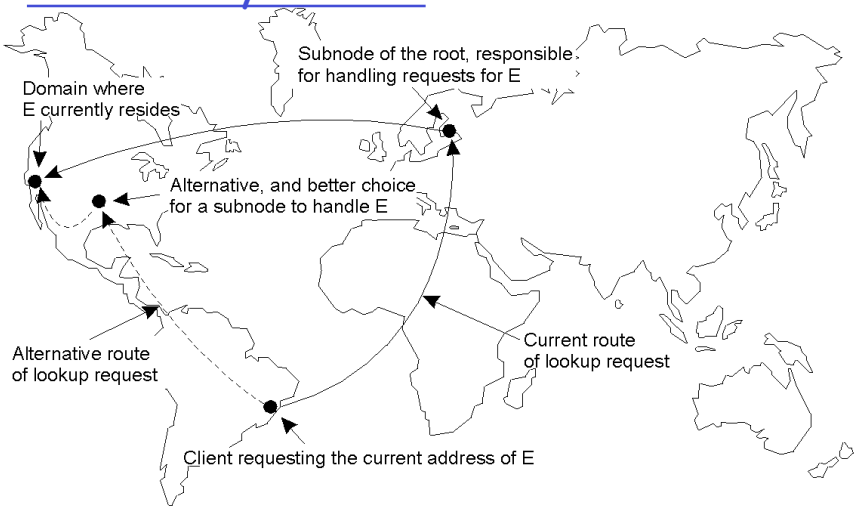
Pointer Caches (2)



A cache entry that needs to be invalidated because it returns a nonlocal address, while such an address is available.

36

Scalability Issues



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

37