DISTRIBUTED SYSTEMS
Principles and Paradigms
Second Edition
ANDREW S. TANENBAUM
MAARTEN VAN STEEN

Chapter 5
Naming

Naming

- Naming and name resolution mechanisms
  - Names, Identifiers, and Addresses
  - Flat Naming
  - Structured Naming
  - Attribute-Based Naming

What's in a name?

- Any entity within a system needs a name – a string of bits/characters referring to an entity.
  - As entities can be operated upon, we need a way of identifying it.
- To operate on an entity, we need an access point – the access point is an address of the entity.
  - Entities may have several access points, and hence several addresses – in just the same way we might have more than one phone number.

Addresses

- The address of an entity may change over time;
  - A new IP address when you move your laptop.
- Addresses however rarely are the same as the name of the entity to which they refer.
  - Machines may be reassigned leading to inappropriate naming.
  - If a machine has more than one access point, which name should be assigned.
- Entity names which are independent of their addresses are easier and more flexible to use – these names are 'location independent'.

Identifiers

- A different type of name is one which uniquely identifies an entity;
  - An identifier refers to at most one entity.
  - An entity is referred to by at most one identifier.
  - And identifier always refers to the same entity.

- Identifiers provide a way of unambiguously referring to an entity.
  - “John Smith” would not be an identifier.
  - A telephone would not be an identifier.

Naming Types

- Flat Naming
  - Systems need to resolve an identifier to the address of its associated entity – an identifier does not contain any information on the associated entity location
- Structured Naming
  - Organized in a name space – represented by a naming graph in which a node represents a named entity and the label on an edge represents the name under which that entity is known
- Attribute-Based Naming
  - Entities are described by a collection of (attribute, value) pairs

Naming Types

How flat names can be resolved?

- Simple Solutions
  - Broadcasting and Multicasting
  - Forwarding Pointers
- Home-based Approaches
- Distributed Has Tables
- Hierarchical Approaches

Forwarding Pointers

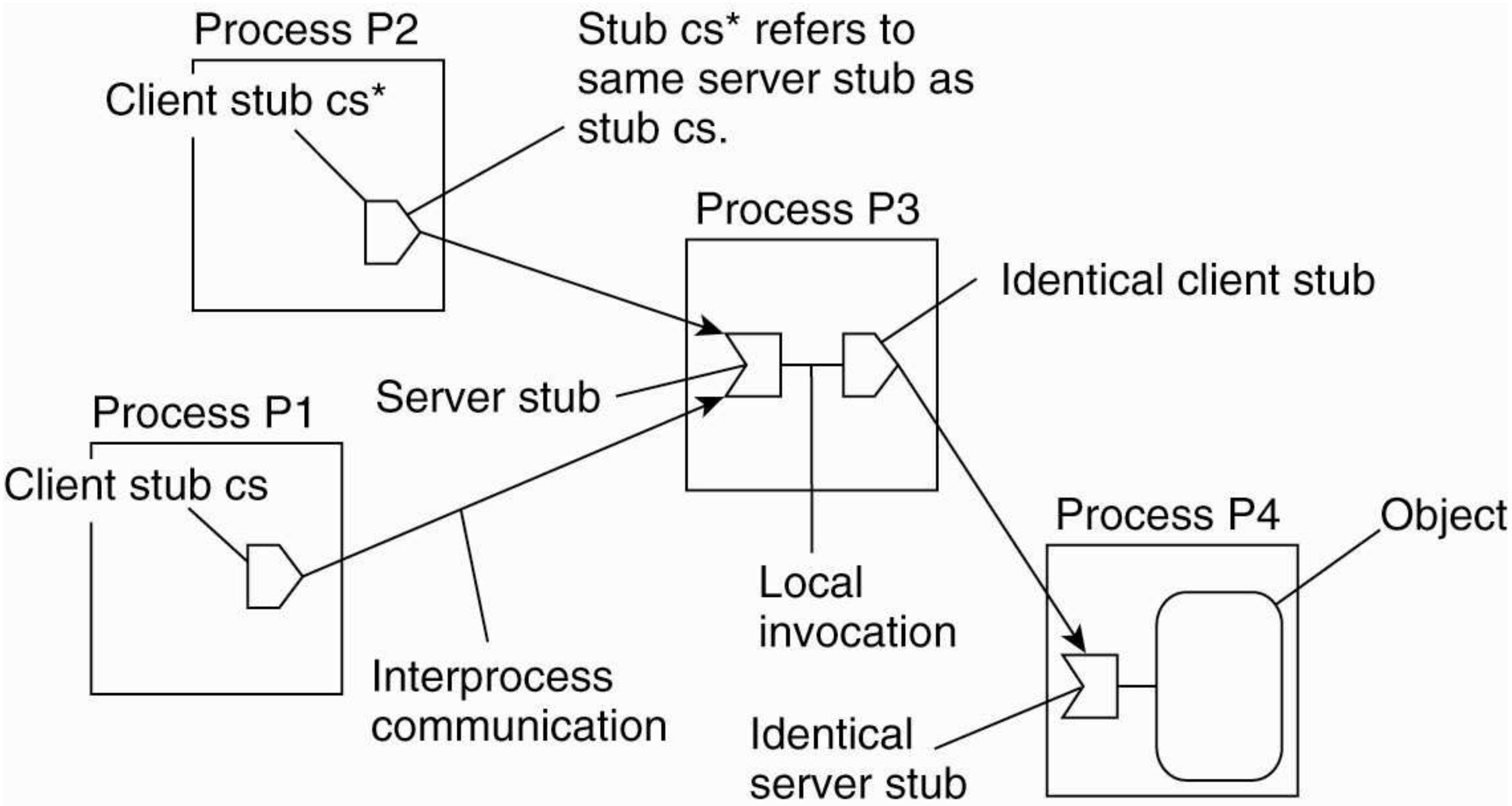


Figure 5-1. The principle of forwarding pointers using (client stub, server stub) pairs.

Forwarding Pointers (cont.)

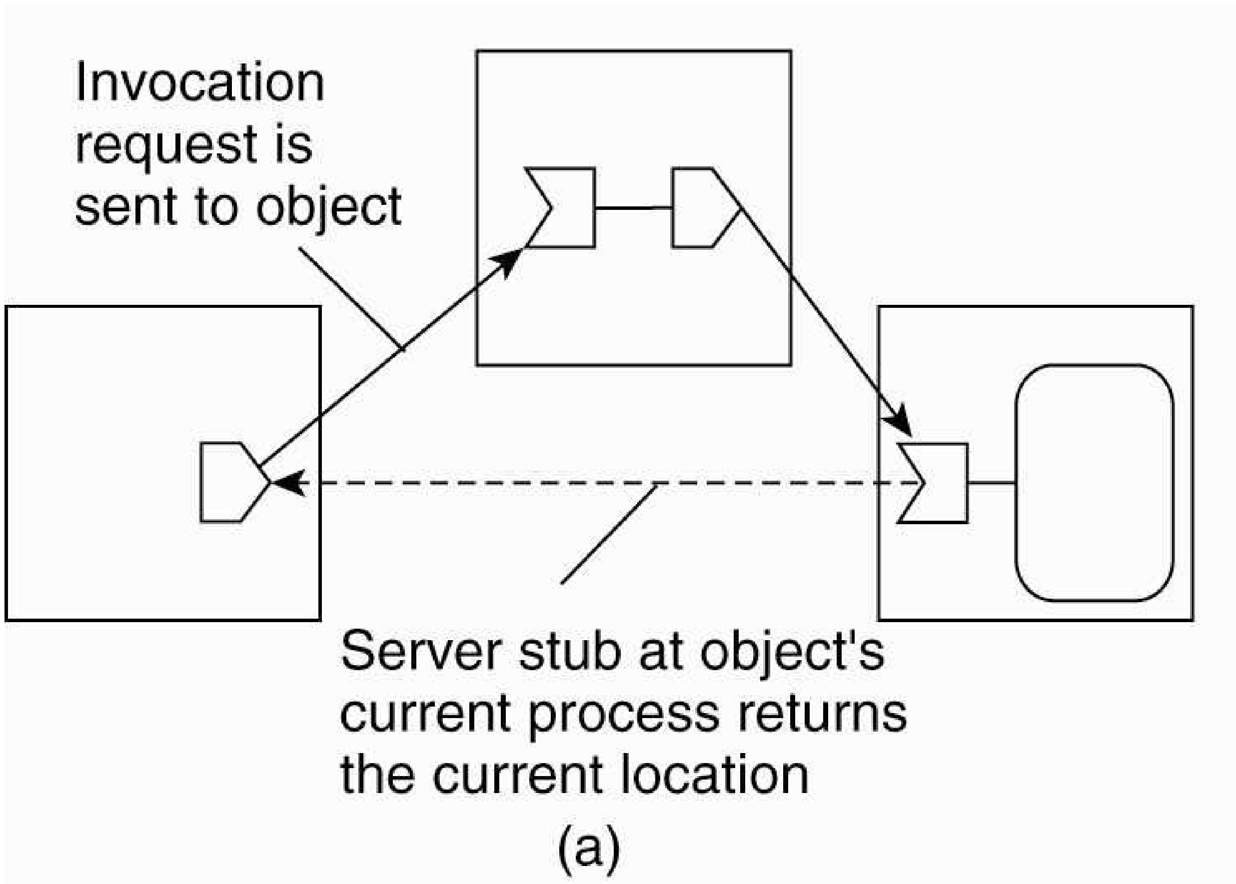


Figure 5-2. Redirecting a forwarding pointer by storing a shortcut in a client stub.

Forwarding Pointers (cont.)

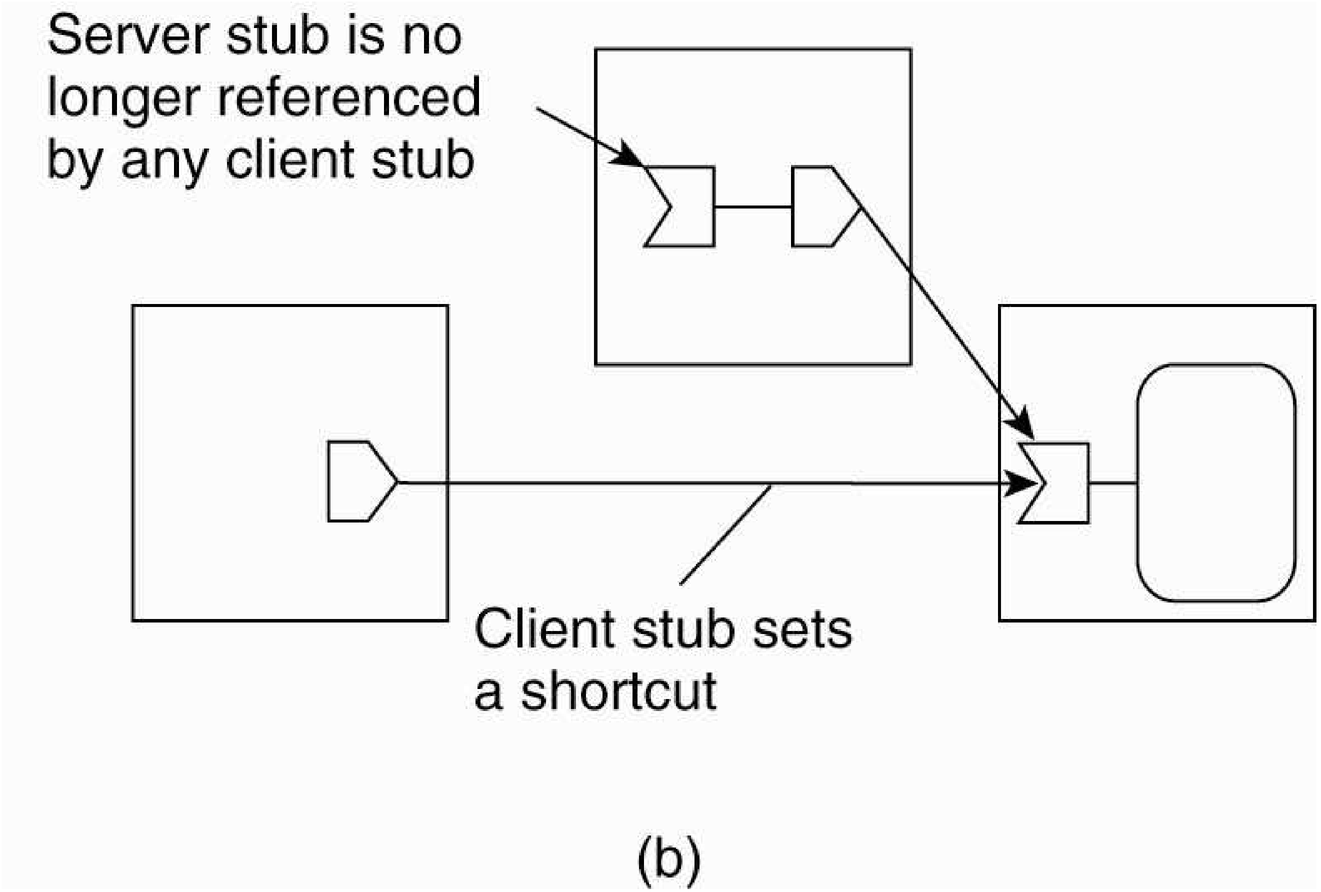


Figure 5-2. Redirecting a forwarding pointer by storing a shortcut in a client stub.

Home-Based Approaches

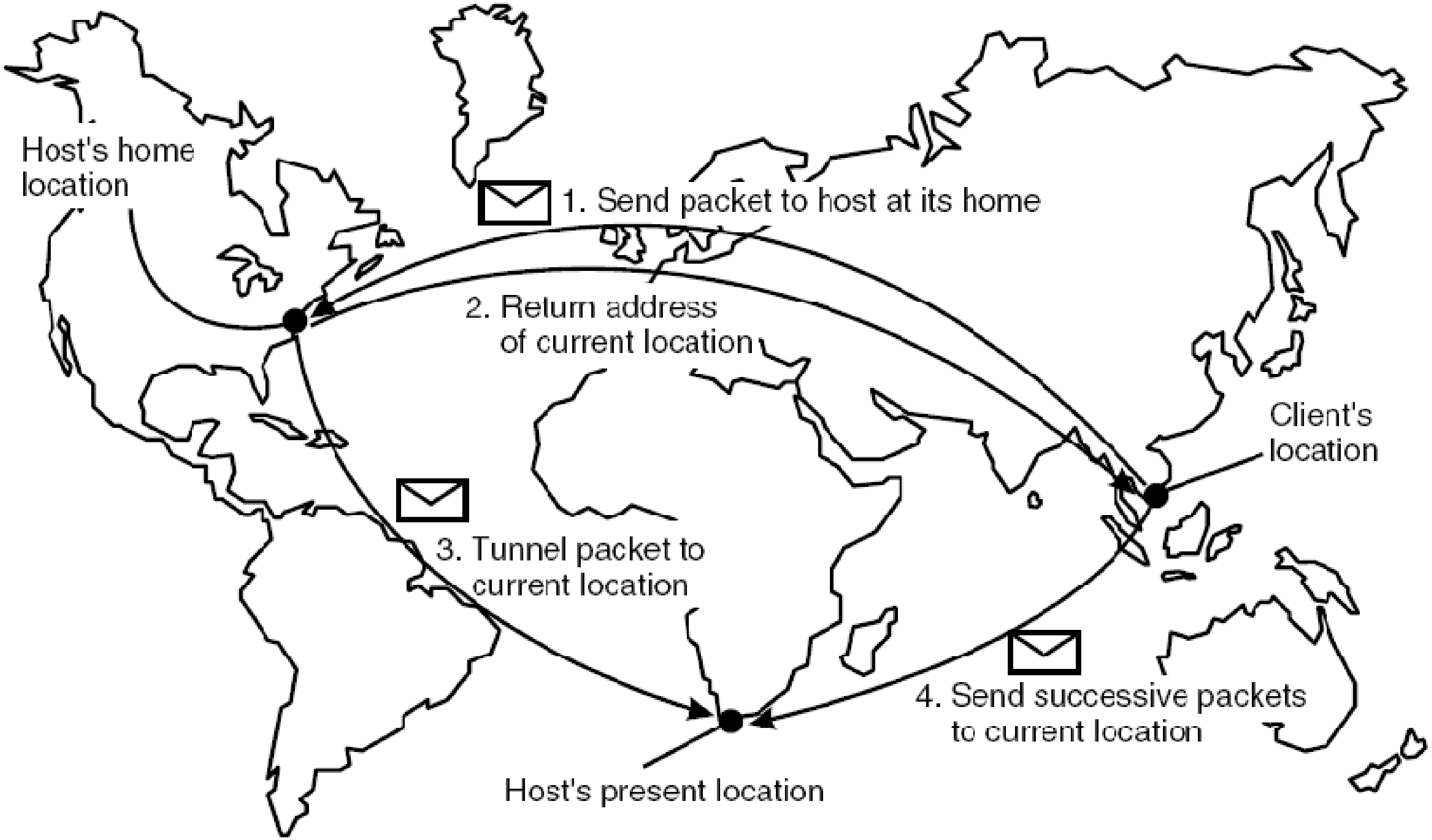


Figure 5-3. The principle of Mobile IP.

Distributed Hash Tables

General Mechanism

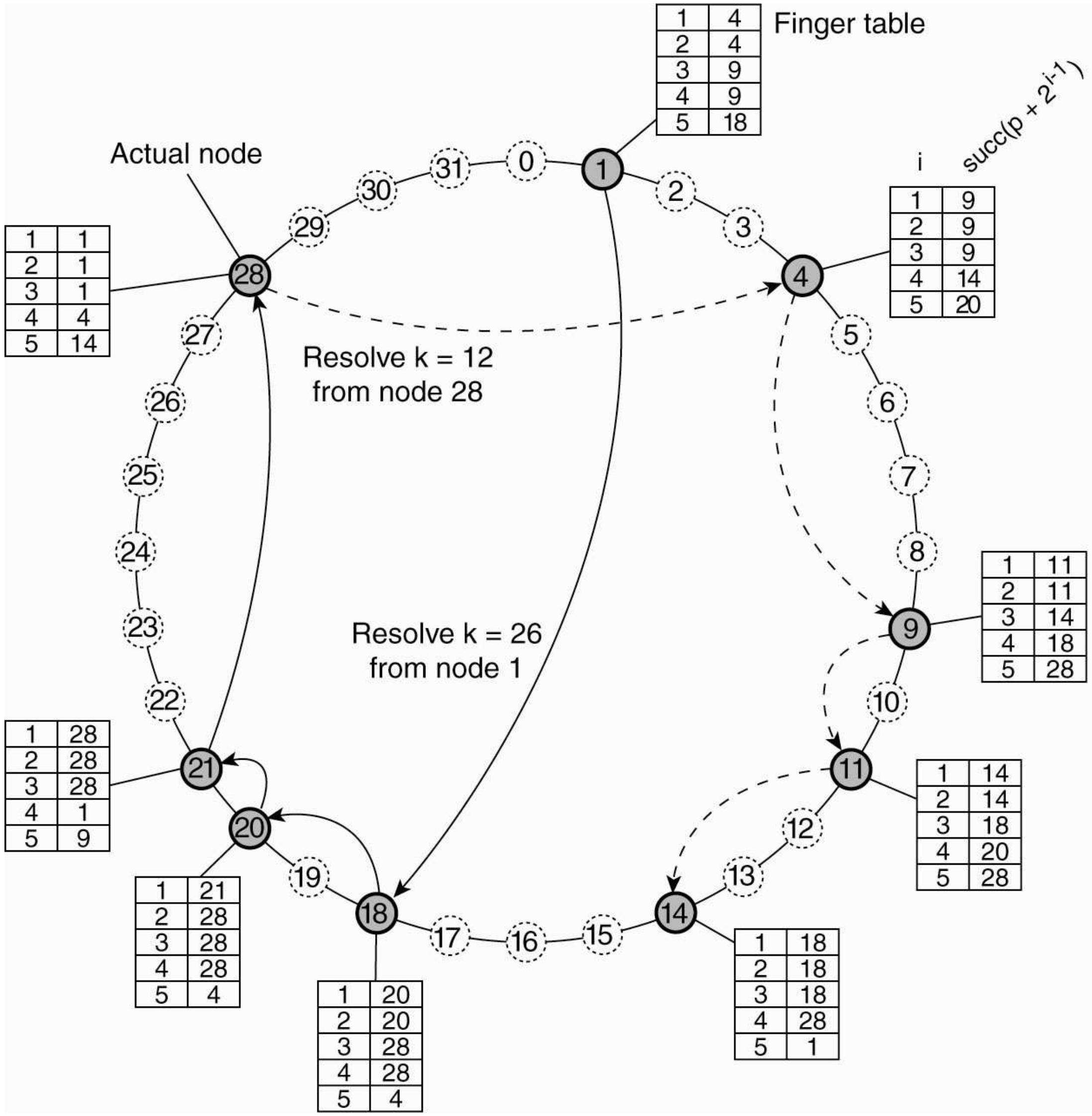
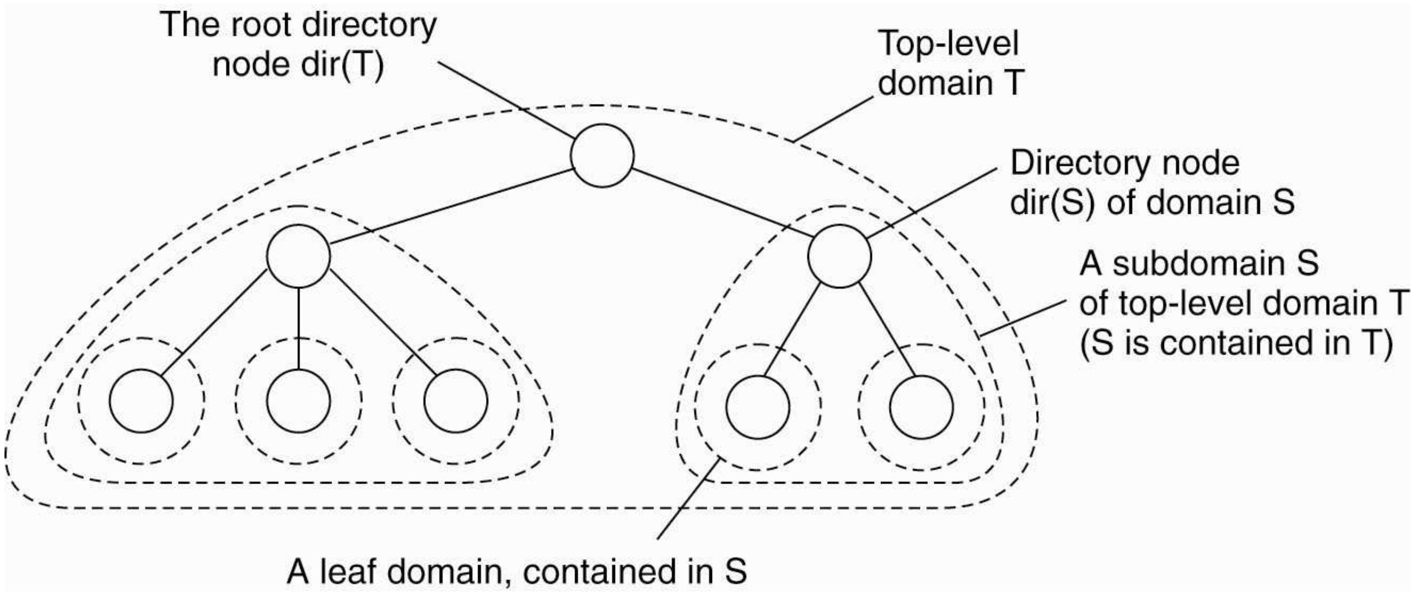


Figure 5-4.
Resolving key 26 from node 1 and key 12 from node 28 in a Chord system.

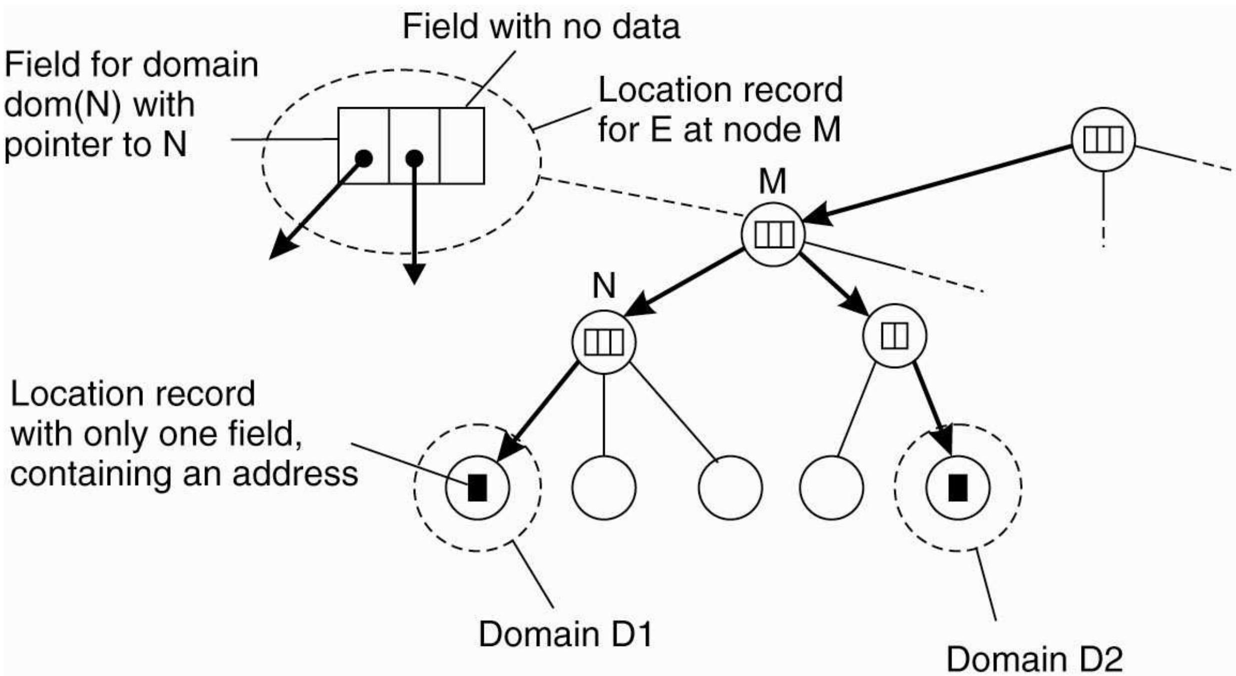
Hierarchical Approaches

- Hierarchical organization of a location service into domains, each having an associated root (directory) node
- Each root node will have a location record for each entity
  - Each record stores a pointer to the directory of the next lower-level sub-domains where that record's associated entity is currently located



Hierarchical Approaches (cont.)

- An entity may have multiple addresses, for example, if it is replicated – smallest domain containing all those sub-domains will have pointers for each sub-domain containing an address
- An example of two addresses in different leaf domains



Hierarchical Approaches (cont.)

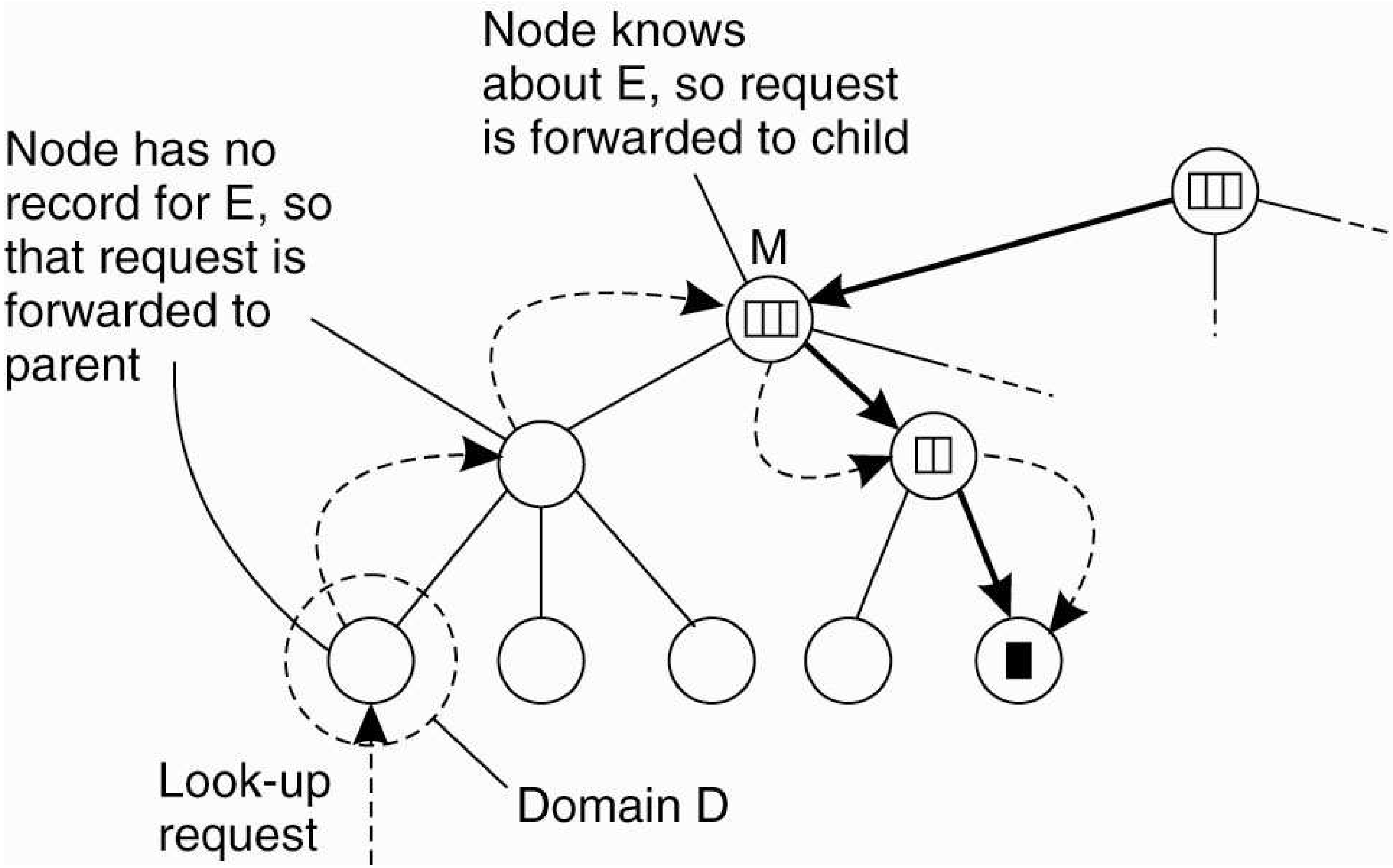


Figure 5-7. Looking up a location in a hierarchically organized location service.

Hierarchical Approaches (cont.)

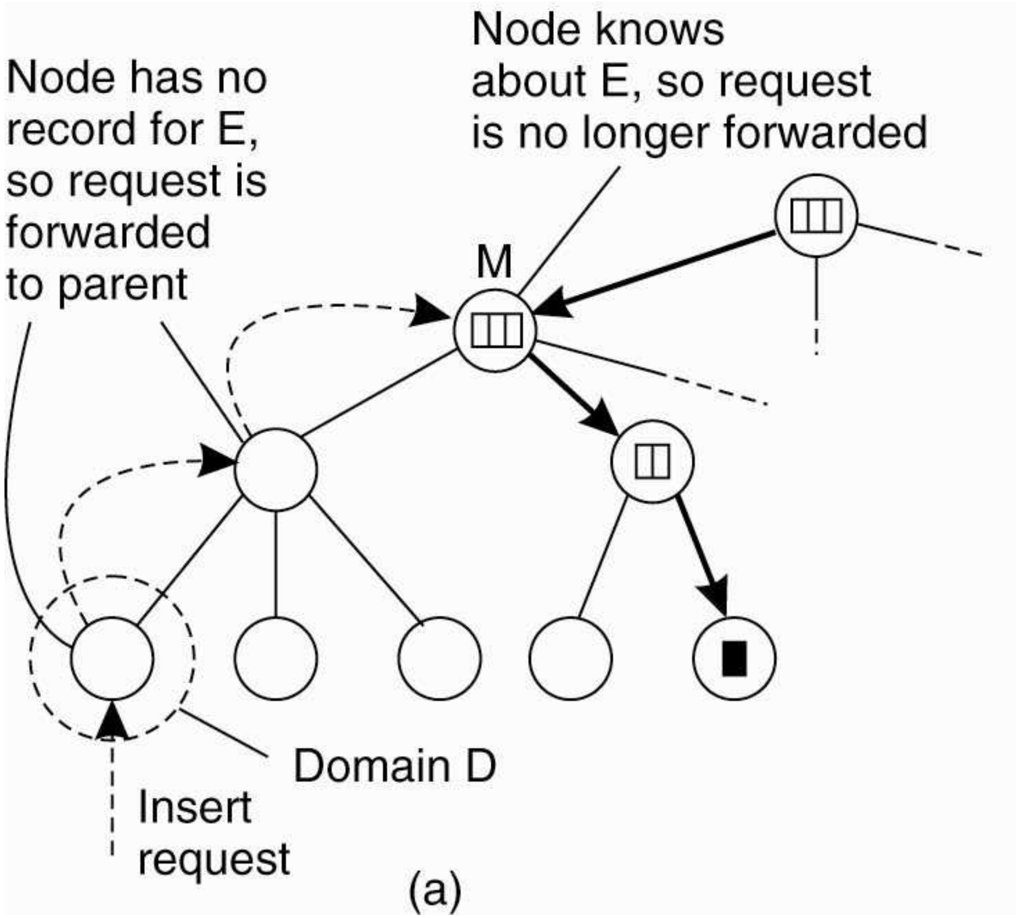


Figure 5-8. (a) An insert request is forwarded to the first node that knows about entity E.

Hierarchical Approaches (cont.)

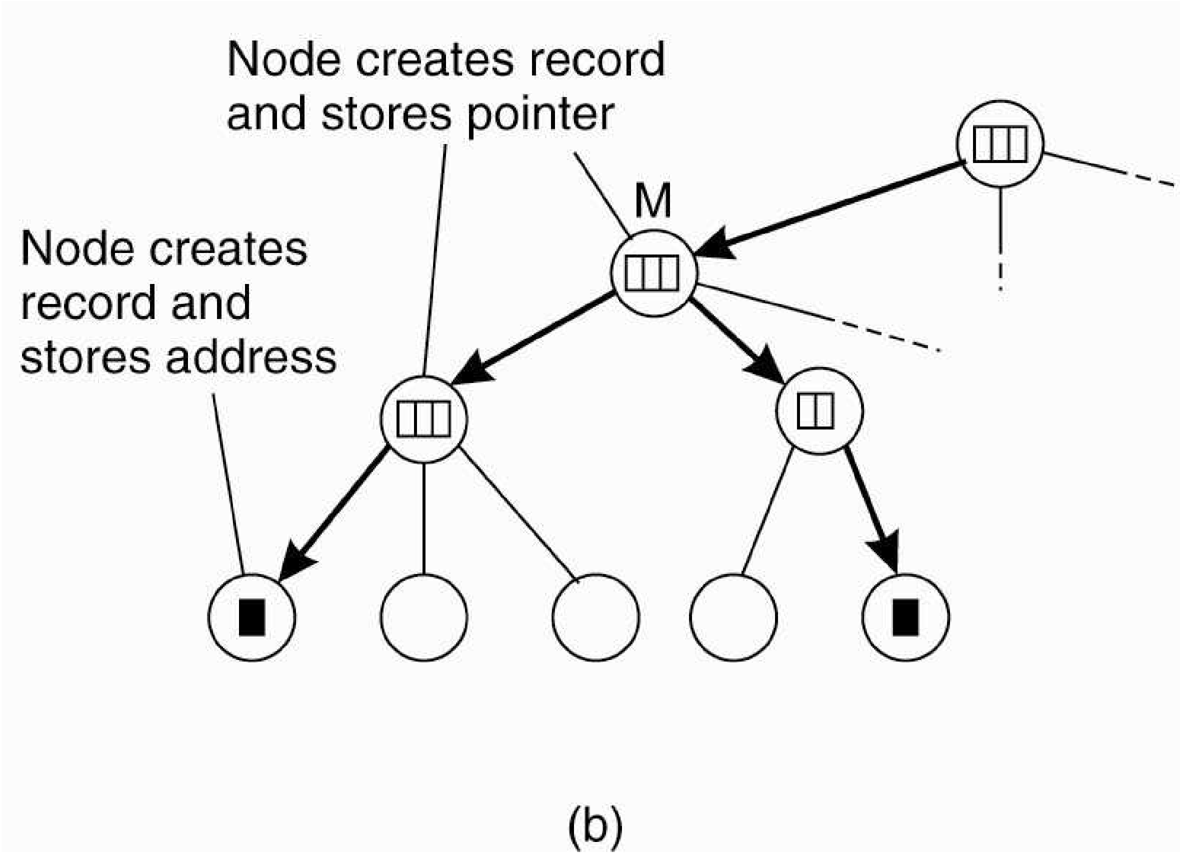


Figure 5-8. (b) A chain of forwarding pointers to the leaf node is created.

Structured Naming

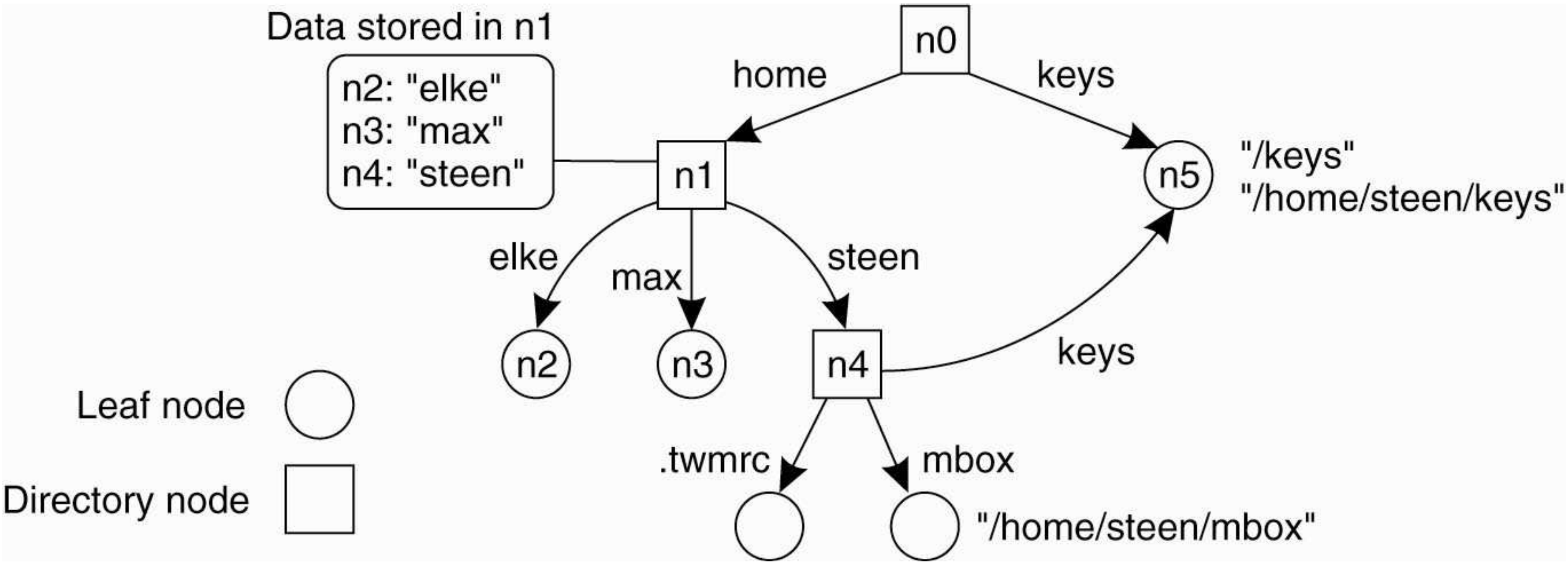
- Structured Naming
  - Organized in a name space – represented by a naming graph in which a node represents a named entity and the label on an edge represents the name under which that entity is known

Namespaces

- A mechanism for storing and retrieving information about
- entities by means of names
  - **Leaf node** – a named entity without any outgoing edge
  - **Directory node** – has one or more outgoing edges labeled with name
  - **Path name** – sequence of labels corresponding to the edges in that path
  - **Absolute path name** – if the first name of the naming graph is root of the naming graph
  - **Relative path name** -otherwise

Name Spaces (cont.)

- A general naming graph with a single root node
- Directed acyclic graph – can have more than one incoming edge, but no cycle



Name Resolution - Looking Up a Name

- Closure mechanism
  - Knowing how and where to start name resolution, specifically deals with finding the initial node in a name space
- Linking - using aliases (another name for the same entity)
  - **Hard links** (in Unix terminology) - allow multiple absolute path names to refer to the same node in the graph (previous diagram)
  - **Symbolic link** – represent an entity by leaf node (next diagram)

Symbolic link

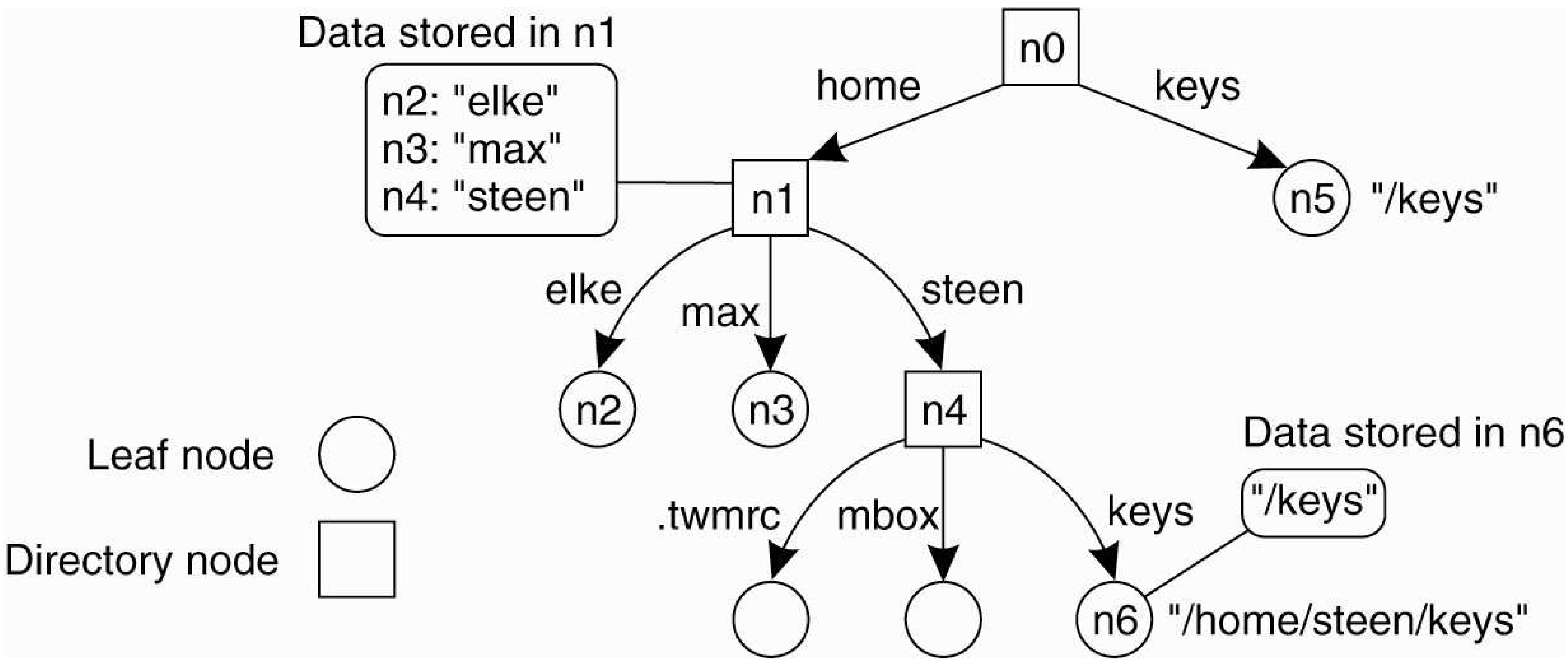


Figure 5-11. The concept of a symbolic link explained in a naming graph.

Mounting

- Mounting
  - Thus far, we have discussed name resolution within a single name space
  - Mounted file system - a directory node stores the identifier of the directory node from a different node space (foreign name space)
  - The stored node identifier is called a mount point, while the directory node in the foreign name space is called a mounting point – usually the root of the foreign name space

Mounting

- Required information for mounting a foreign name space in 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 (cont.)

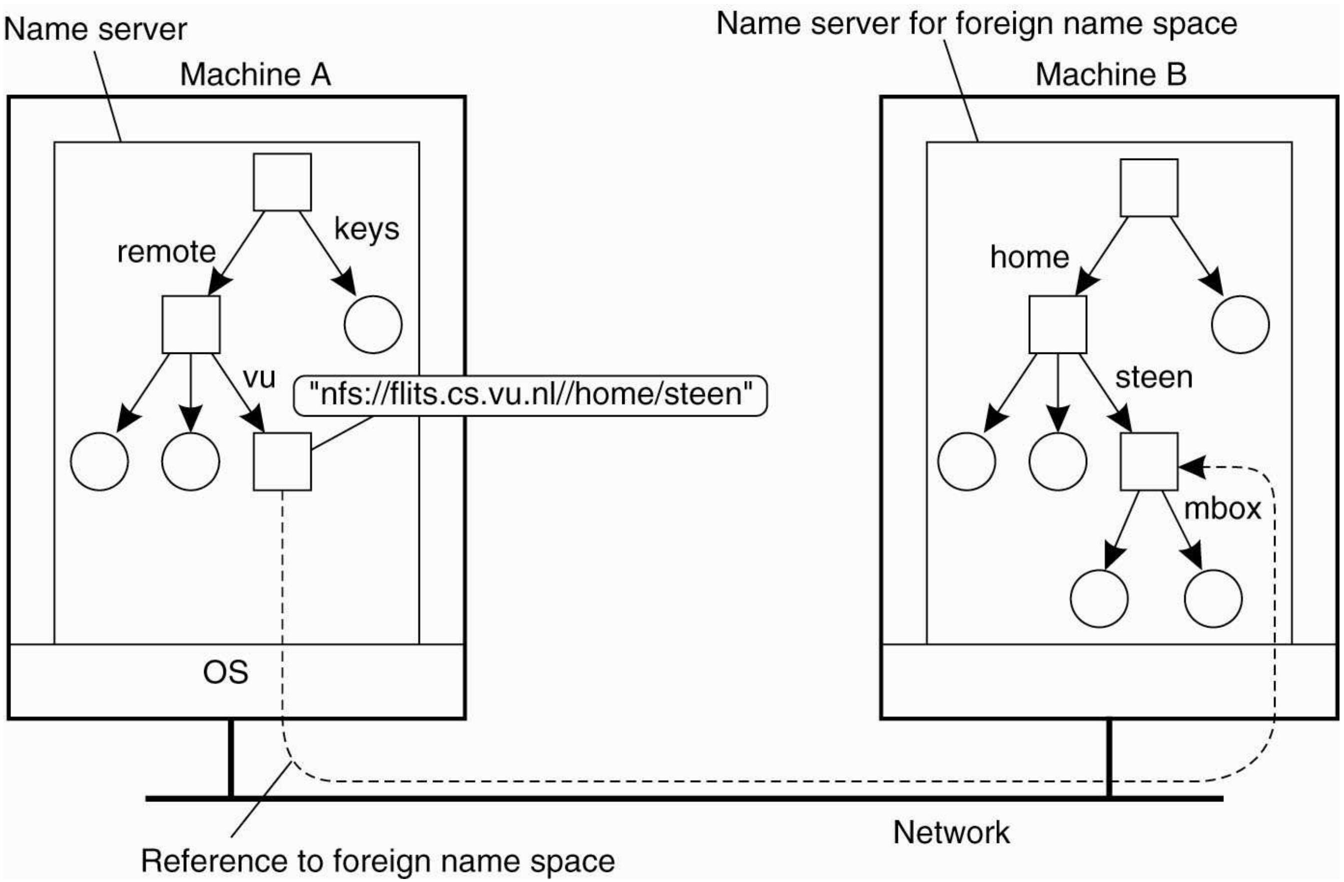


Figure 5-12. Mounting remote name spaces through a specific access protocol.

The Implementation of a Name Space

- Implementation by partitioning into layers
  - Global layer
  - Administrational layer
  - Managerial layer
- Global layer
  - Highest level of nodes (the roots and other directory nodes closed to root)
  - Rarely change– stable
  - May represent organizations, groups of organizations, for which names are stored in the name space

The Implementation of a Name Space

- Administrational layer
  - Formed by directory nodes managed within a single organization
  - Represents group of entities of same organization or administrative unit
  - Less stable than global layer

The Implementation of a Name Space

- Managerial layer
  - Includes nodes representing hosts in local area network are, shared files such as those for libraries and binaries, and user defined directories and files
  - Typically change regularly
  - Maintained not only by the system administrators but also by end users
- Maintained not only by system administrators but also by end users

Name Space Distribution (Example.)

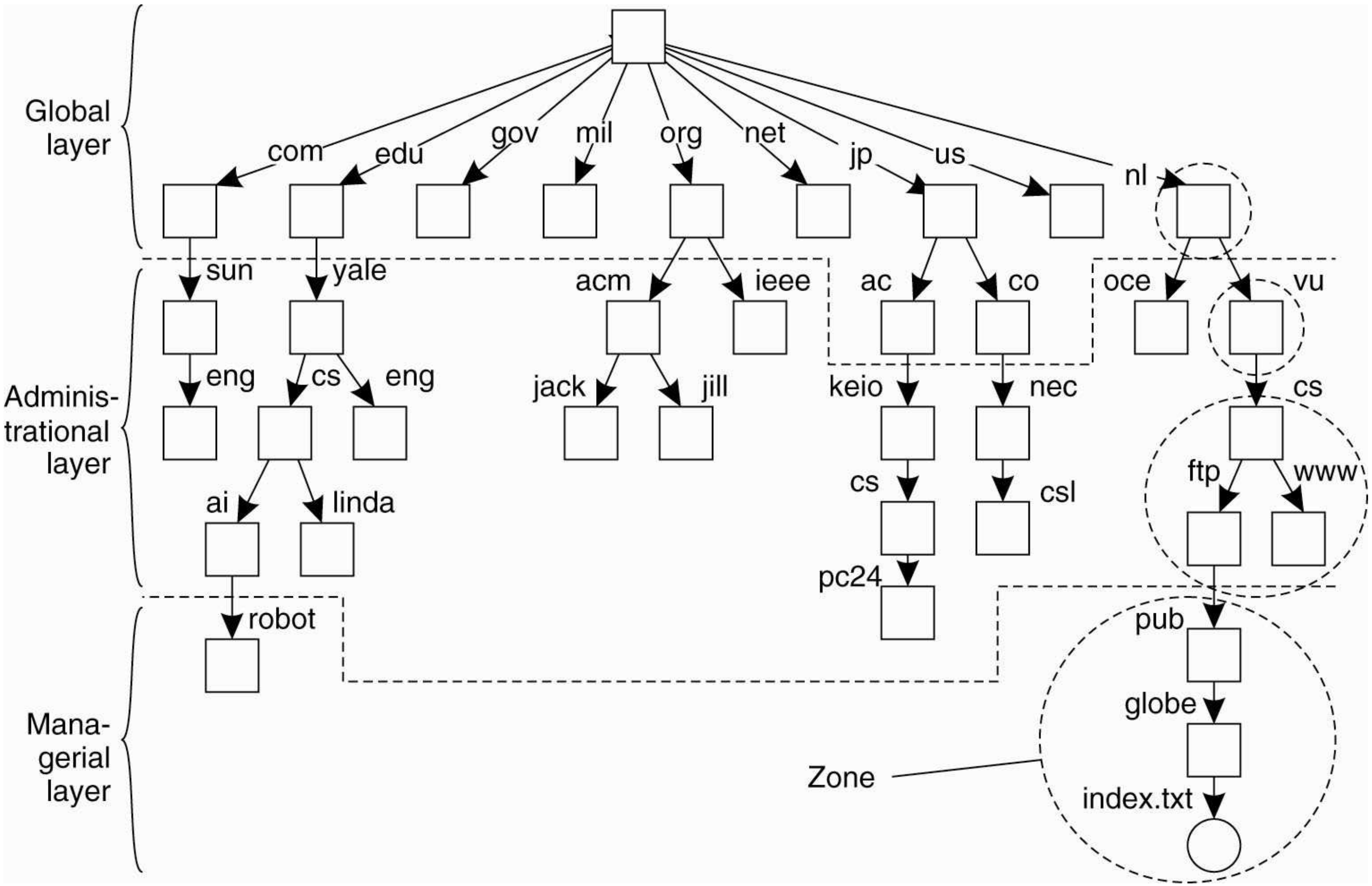


Figure 5-13. An example partitioning of the DNS name space, including Internet-accessible files, into three layers.

Name Space Distribution (cont.)

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

Figure 5-14. 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.

Implementation of Name Resolution

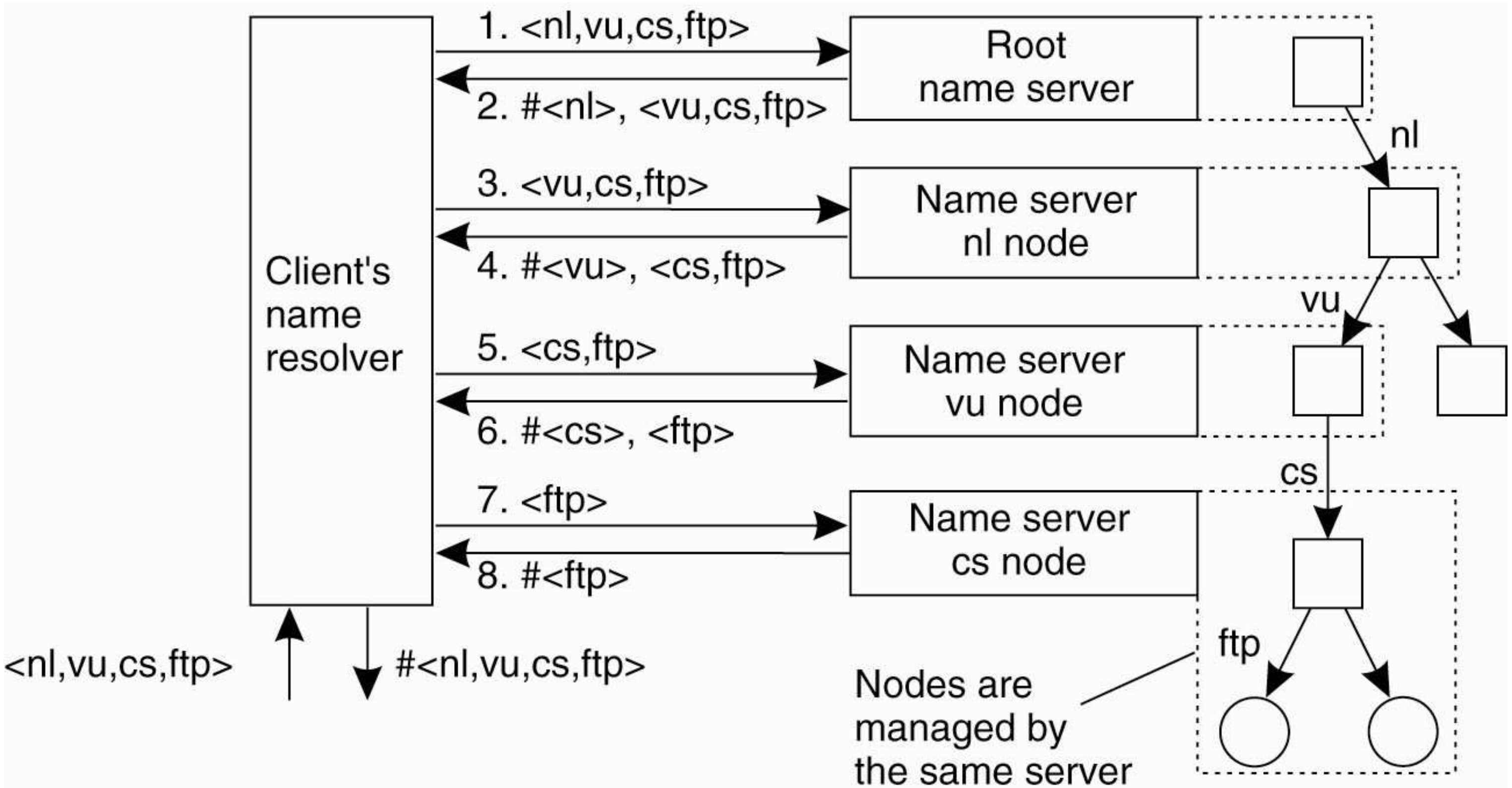


Figure 5-15. The principle of **iterative name resolution**.

Implementation of Name Resolution (cont.)

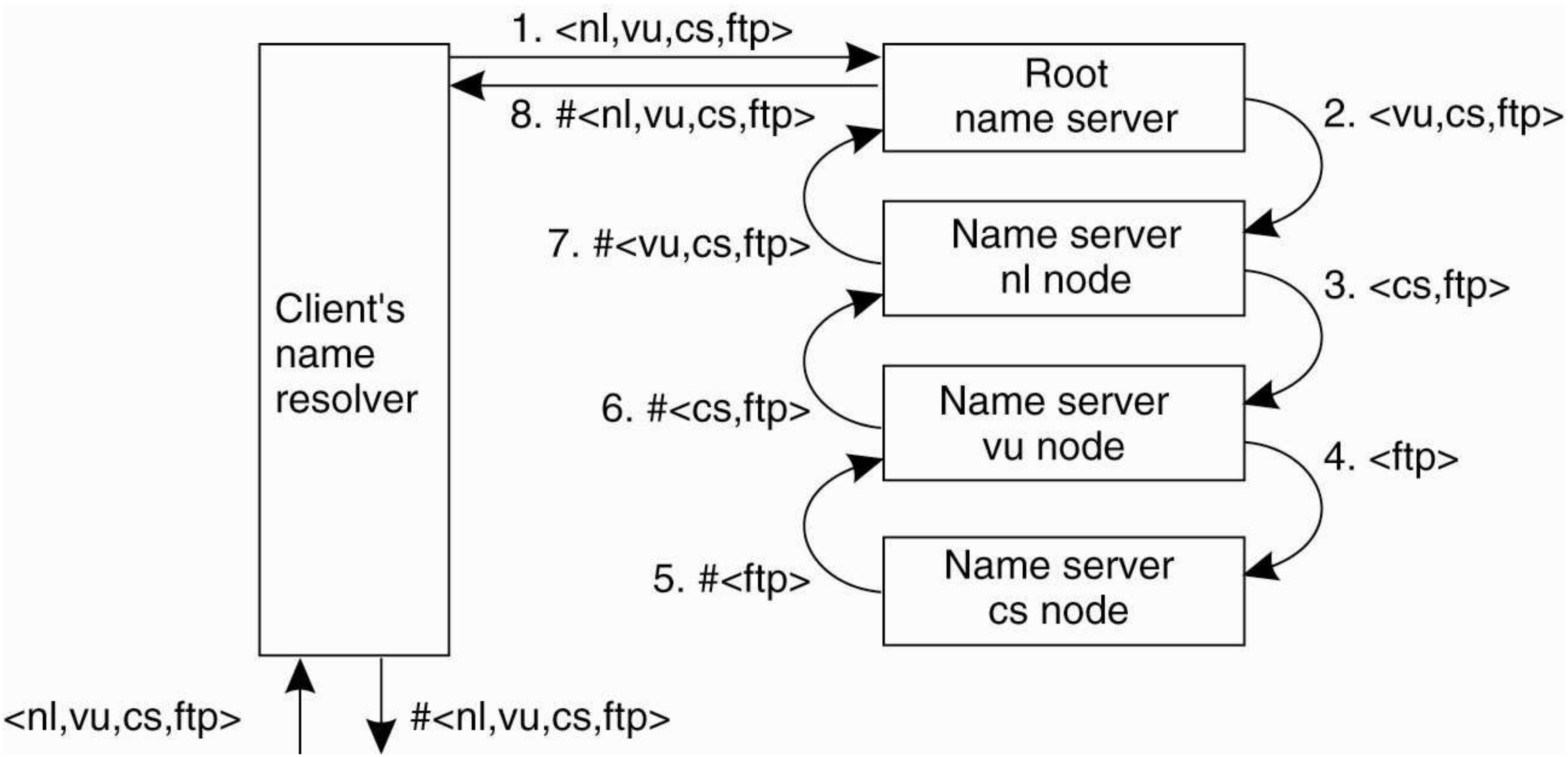


Figure 5-16. The principle of **recursive name resolution**.

Implementation of Name Resolution (cont.)

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>

Figure 5-17. Recursive name resolution of *<nl, vu, cs, ftp>*. Name servers cache intermediate results for subsequent lookups.

Example: The Domain Name System

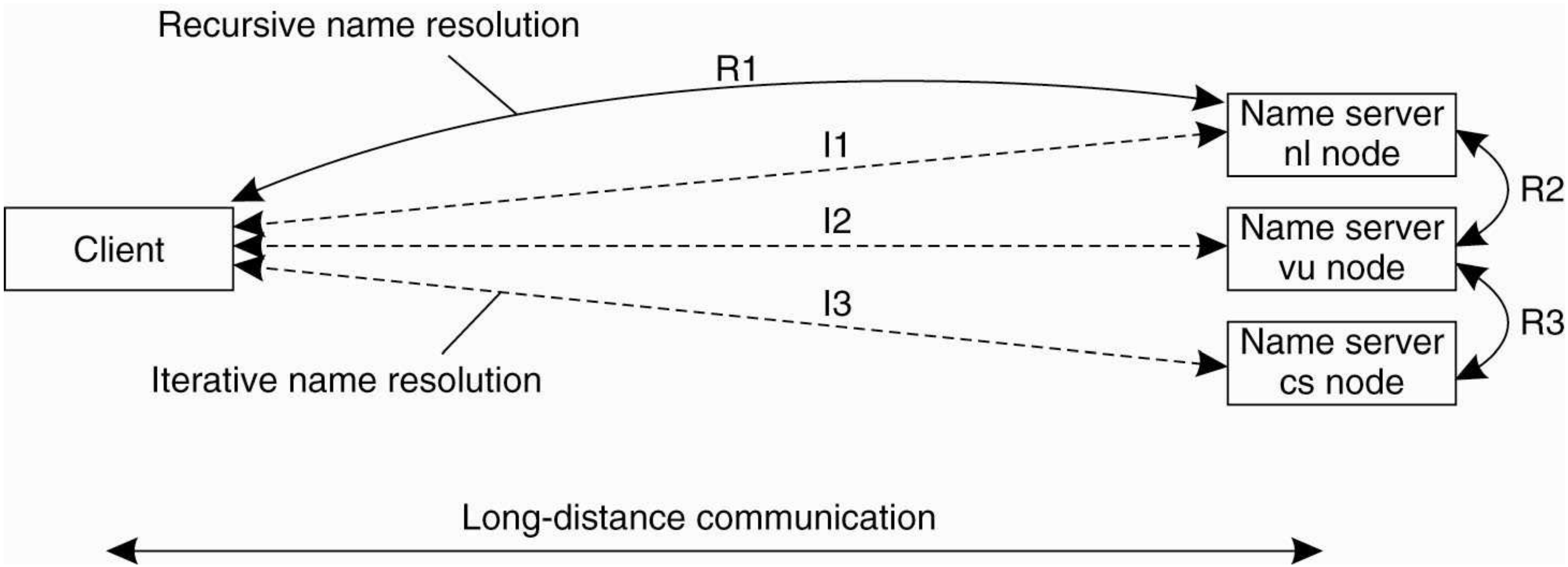


Figure 5-18. The comparison between recursive and iterative name resolution with respect to communication costs.

Attribute-Based Naming

- Flat and structured names have considered mainly location independence and human friendliness of names
- There are scenarios where a user can merely describe (provide attributes) what he/she is looking for - attribute-based naming
  - An entity is described by a collection of (attribute, value) pairs
  - Each attribute describe some aspect of the entity
  - By specifying which values a specific attribute should have a user can essentially constrains the set of entities that the user is interested in
  - The naming system returns one or more entities that matches the user's description

Hierarchical Implementations: LDAP

- Lightweight Directory Access Protocol (LDAP)
  - A simplified protocol to provide directory services in the Internet.
  - Combine structured naming with attribute-based naming
  - Widely adopted in many distributed systems, e.g. Microsoft's Active Directory Service.
  - An application-level protocol that is implemented directly on top of TCP
  - Lookup and update operations can simply be passed as a strings

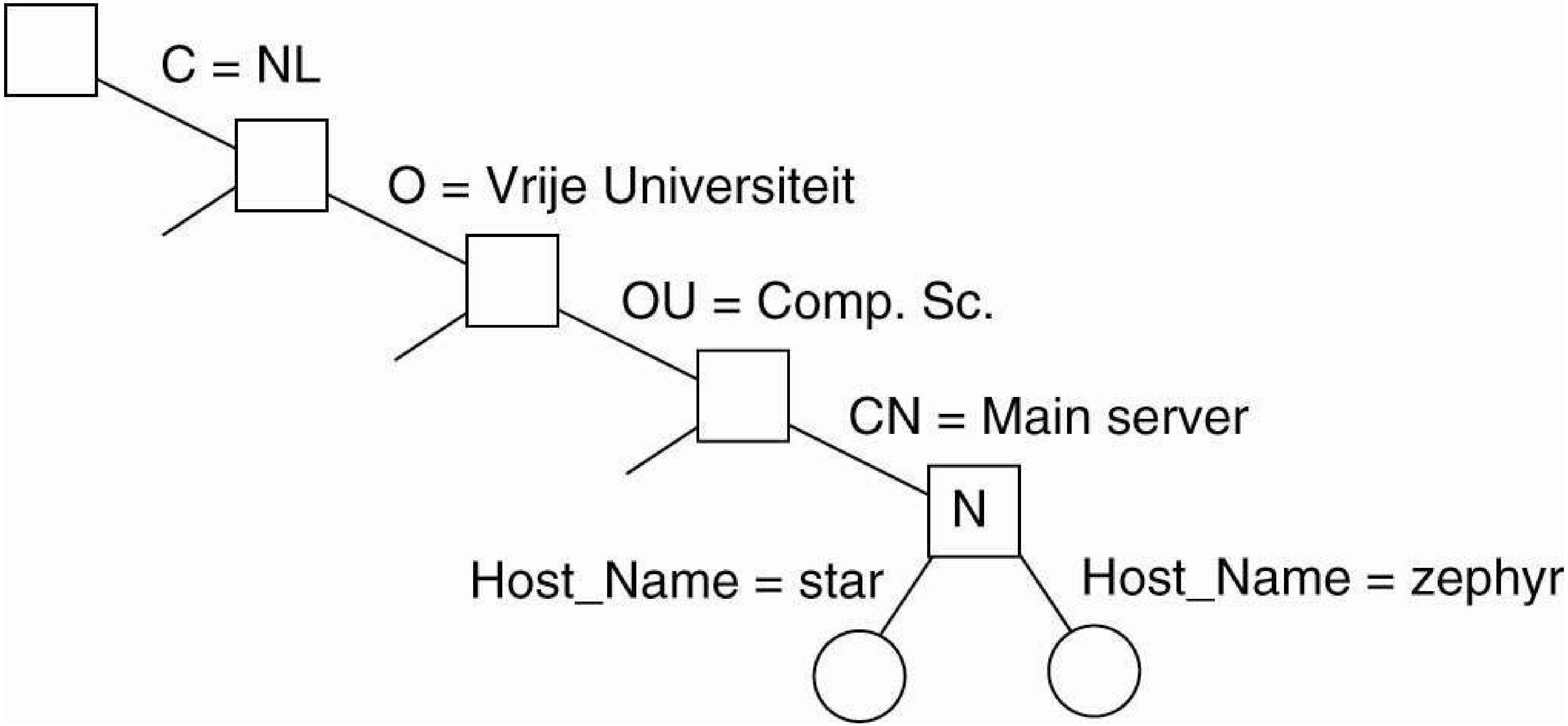
LDAP (cont.)

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

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

LDAP (cont.)

- Part of a directory information tree.



LDAP (cont.)

- Two directory entries having *Host_Name* as RDN.
- Difference between DNS and LDAP implementation – search a directory entry given a set of criteria that attributes of the searched entries should meet

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)

Mapping to Distributed Hash Tables (1)

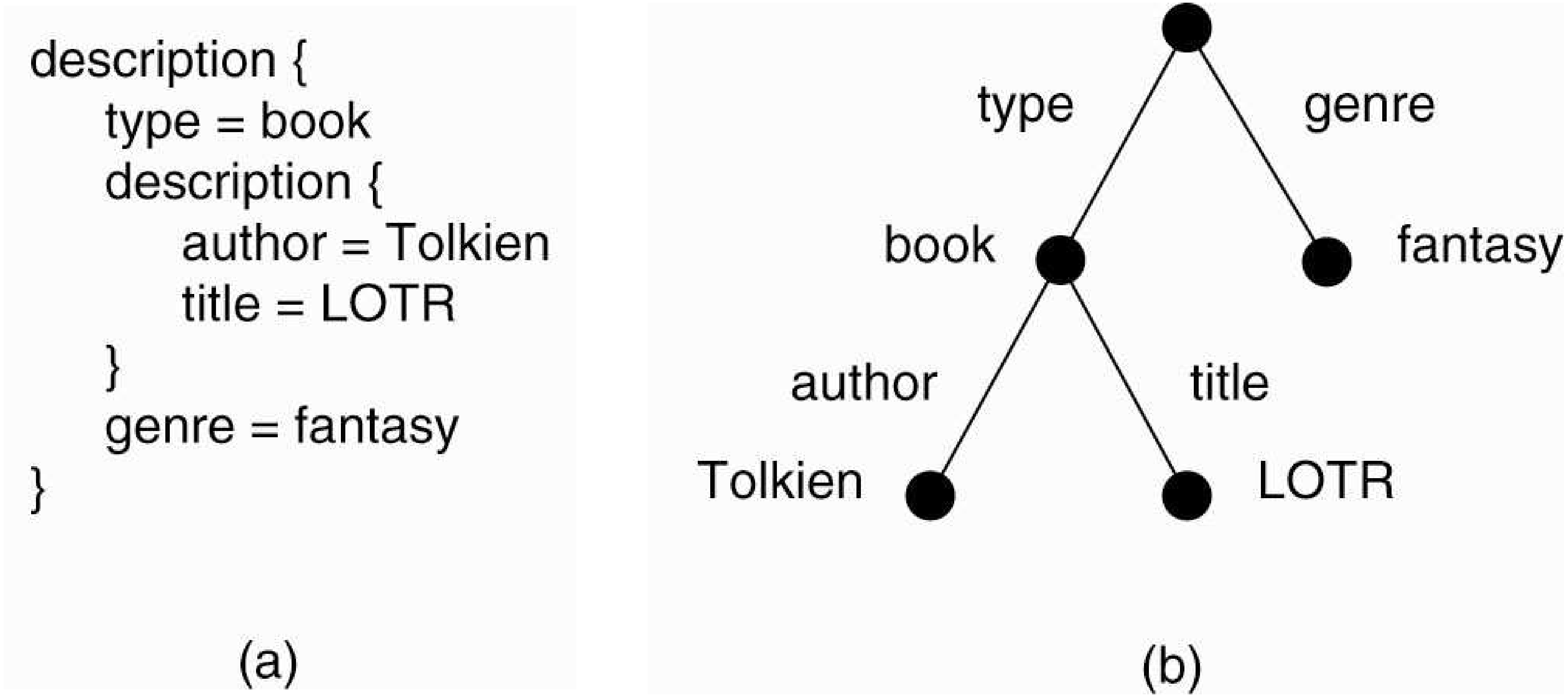


Figure 5-24. (a) A general description of a resource.
(b) Its representation as an AVTree.

Mapping to Distributed Hash Tables (2)

```
description {
    type = book
    description {
        author = Tolkien
        title = *
    }
    genre = *
}
```

(a)

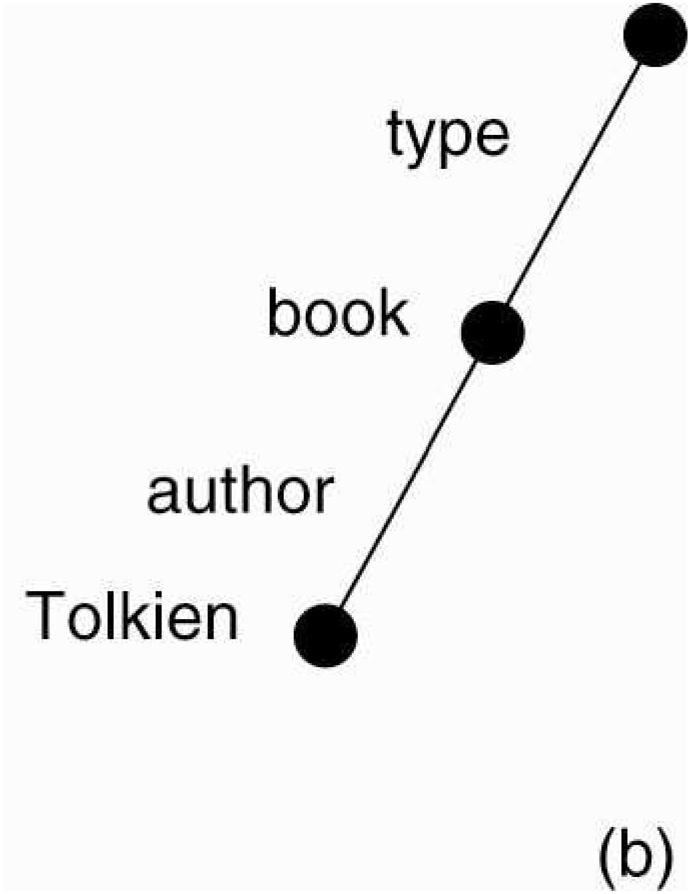


Figure 5-25. (a) The resource description of a query.
(b) Its representation as an AVTree.

Semantic Overlay Networks

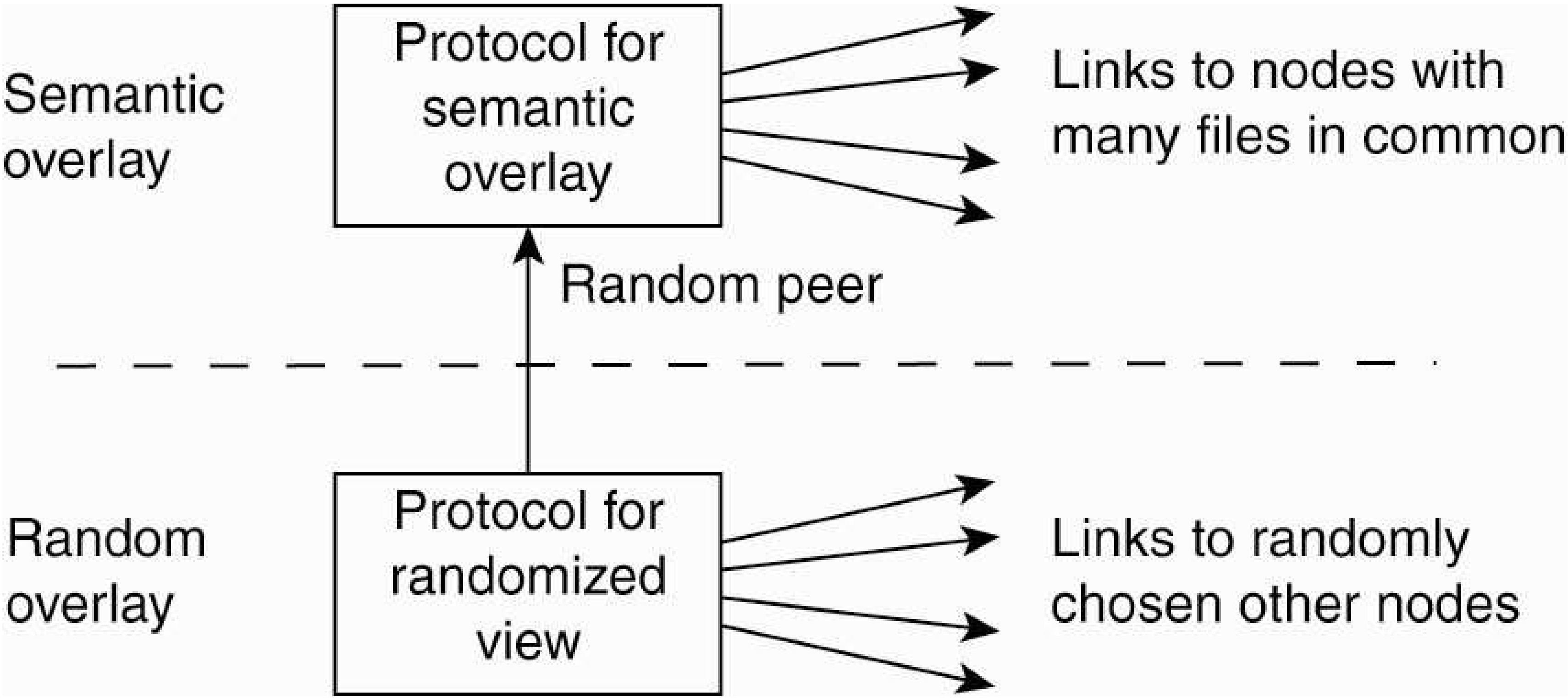


Figure 5-26. Maintaining a semantic overlay through gossiping.