5.分布式命名管理

1.基本概念

实体（entity）：系统中的任何对象

- 物理资源型：主机、文件、打印机、磁盘
- 逻辑抽象型：进程、用户、邮箱、新闻组、主页、报文

实体的名字（Name）：一个数字位串或字符串，可唯一地表示一个实体

- 如主机名、文件名、进程名、用户名

命名（Naming）：赋予名字

- 引用（Refer）：如果实体 X 的名字为 A，我们说，名字 A 引用（指向）实体 X。

实体访问点（access point）：用于访问该实体的接口，也是一个实体

- 可以有多个访问点
- 实体可以改变访问点
- 访问点可以赋值给另一个实体

地址（address）：实体访问点的名字，指向（引用）访问点；例如：服务器地址：IP+ 端口号

位置独立性

- 实体的名字包含有地址信息，则是位置相关的
- 实体的名字独立于地址，则是位置独立的

标识符（Identifier）：特殊类型的名字

- 一个标识符最多引用一个实体

- 每个实体被一个标识符所引用
- 一个标识符总是引用同一个实体（不准重用）

机器可读的名字：数字位传，如网卡地址

用户友好的名字（human-friendly name）：有意义的字符串

实体定位：通过名字，找到实体所在的位置，从而找打实体

- 名字解析：将名字映射到地址的操作过程

名称到地址的绑定：二元关系组（名字，地址），举例：域名数据库

固定实体：固定位置，实体的位置不会改变

移动实体（mobile entiny）：具有可变位置，实体的位置可以改变

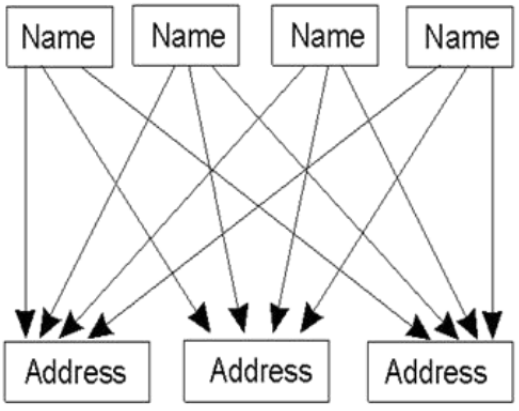
位置更新问题：

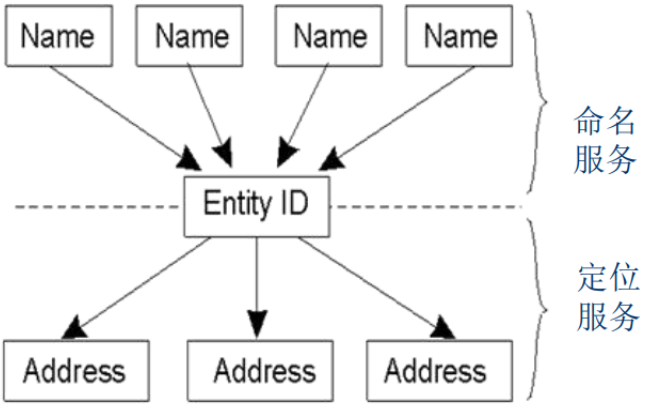
- 位置改变后，需查找 DNS 数据库，进行地址更新
- 地址采用硬链接，如 130.37.21.5，更新效率低
- 地址采用符号链接，ftp.is.vu.nl，查找效率低

传统方法：在名字和地址之间的直接的单级映射

适合移动实体的方法：

- 采用标识符的两级映射
- 将命名服务与定位服务分离开





2.非结构化命名管理

名称中不包含任何访问点的信息。例：72550（工作证号）

2.1 简单的实体定位方法：广播和多播方法

（1）广播方法（broadcast）

例： 地址解析协议（ARP ） ： 由 IP 地址找到它的链路地址

在大规模网络环境中， 效率低

（2）多播方法（multicast）

多播组、 多播地址

定位点到点网络中的实体

定位多个实体： 同一组织

定位实体的副本： 最近的副本

（3）问题：可伸缩性

只适用于小规模网络，如局域网络

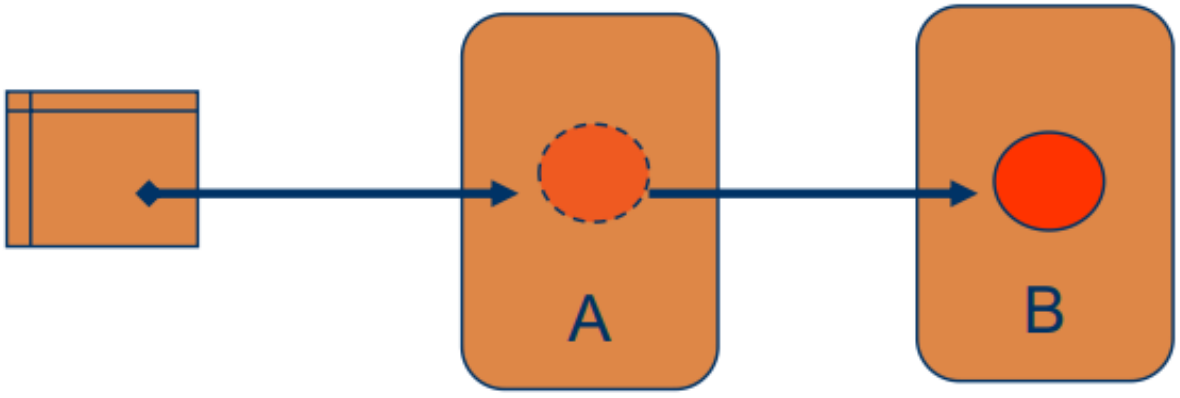
2.2 转发指针方法

当实体从 A 移动到 B 之后，在 A 上设置一个指向 B 的引用

优点：客户可利用传统的命名服务

缺点：间址链可能会很长、链的中间节点需要维护转发信息、链容易断

目标：限制链的长度，保证链的鲁棒性

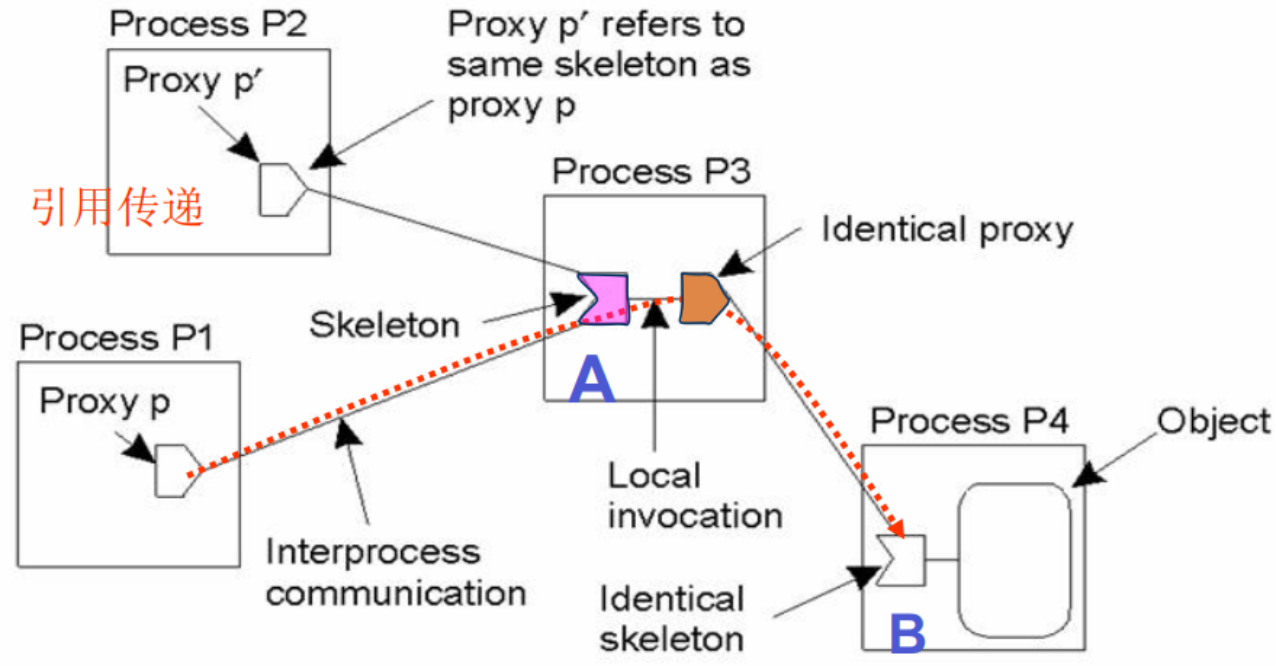


（1）分布式对象迁移原理

SSP 链：（stub 存根，scion 后代） $pairs = (proxy, skeleton)$

- proxy：客户存根
- skeleton：服务器存根

举例：当 O 从 A 移动 B 后，在 A 上保留原来的 proxy，并设立一个代表 O 的 skeleton

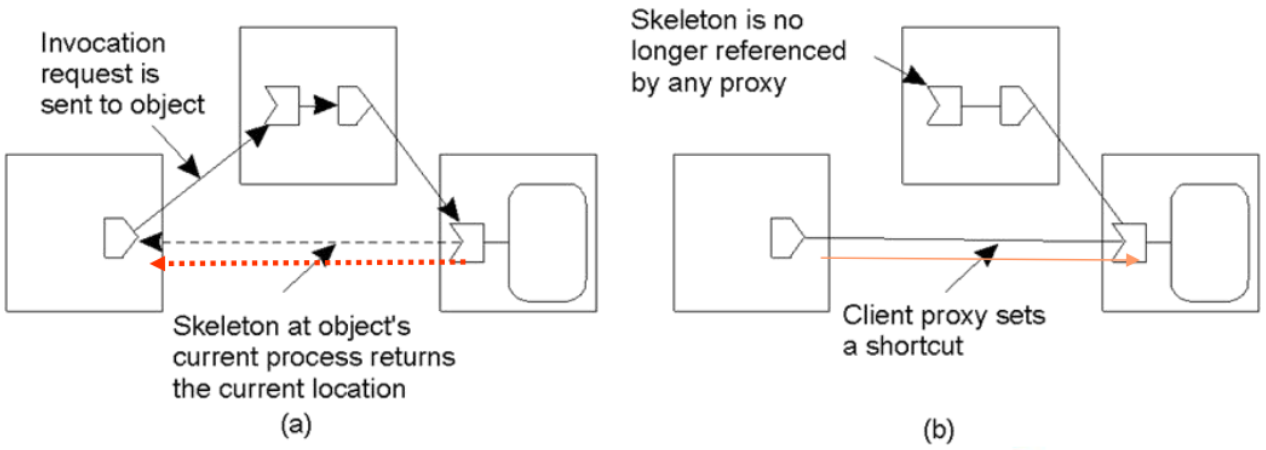


（2）转发指针的优化策略

例：通过在 proxy 中存储一个捷径（shortcut），重定向转发指针

调整策略：

- 单独：仅有请求发起人，即初始 proxy 建立 proxy 短路
- 全部：转发链上的所有 proxy 建立短路



2.3 基于原籍的方法

可用于大规模网络

原籍（Home）位置：

- 实体的创建位置
- 用于跟踪实体的当前位置

举例：移动 IP 原理

- 每一个主机有一个固定 IP 地址
- 固定 IP 地址对应一个 Home 代理
- 主机移动后的临时地址（care-of 转交地址）在它的 Home 代理上登记
- 当 Home 代理收到 packet 后，转发到转交地址

缺点：

- 可能会舍近求远；当前位置比原籍近，增加通信开销
- 固定的原籍位置；无法根据需要改变，如主机已永远移动后

解决方案：将原籍位置记录在传统的名字服务器中

2.4 分布式哈希表（DHT）

（1）概念

散列表（Hash table，也叫哈希表）：根据关键码值（key value）而直接进行访问的数据结构

分布式哈希表（DHT）：

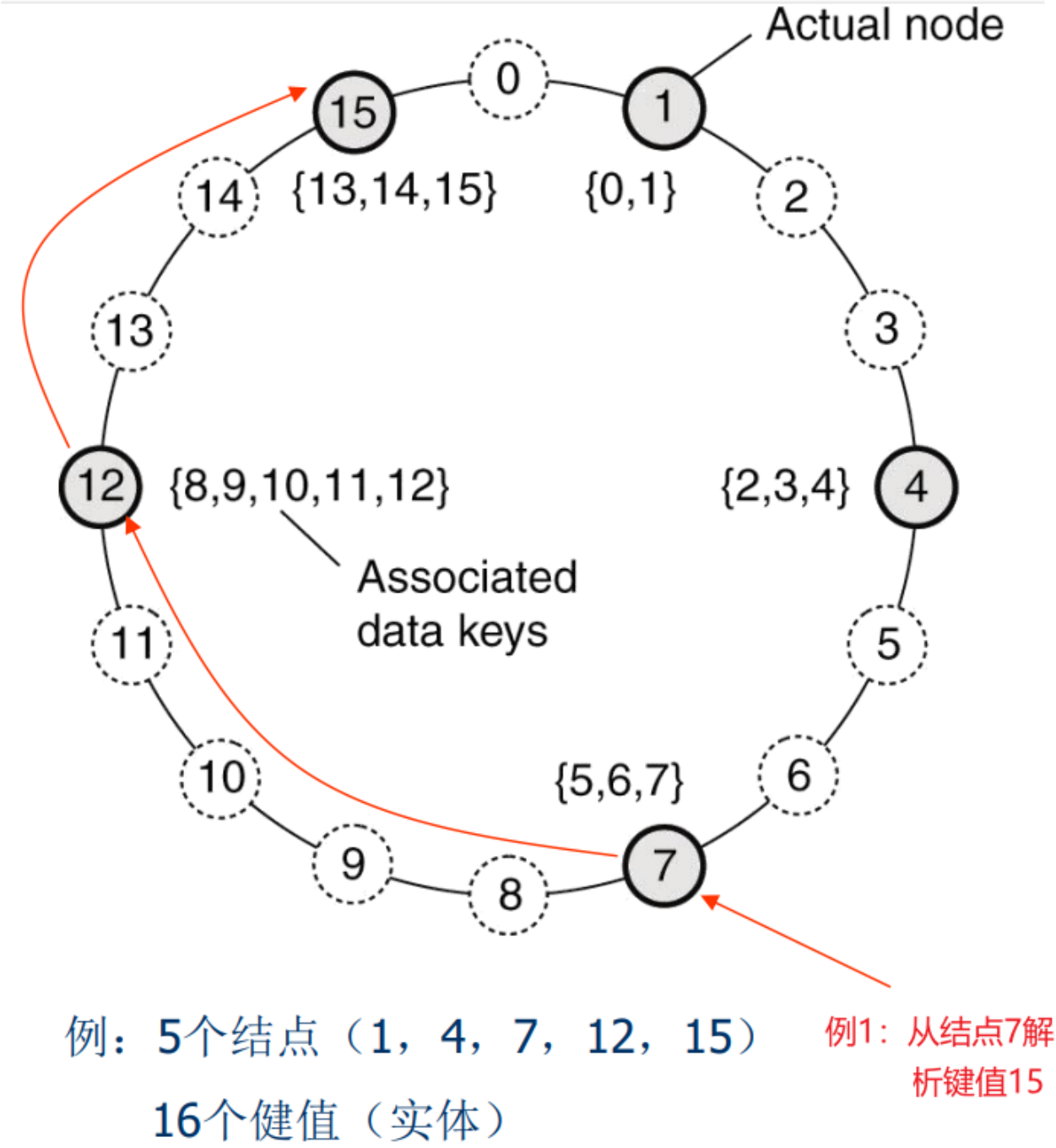
- 广域范围中的巨大哈希表，有若干个分布式结点共同维护
- 散列表被划分成很多块，每个结点被分配一个标识符（id），并称为与 id 对应的散列块的管理者。

散列表入口项：

- 一个实体的名字或关键字，通过散列（hash）函数，被映射为 128 位或 160 位散列值 k
- 键值为 k 的实体位于大于 k 的最小标识符（k≤ id）的节点内

Chord 系统：

- 结点：具有标识符 id
- 实体：具有键值 k
- k 的存储节点 $succ(k)$， $min\ \{id|id \geq k\}$



实体 k 的查找（线性）

- 为结点 p 建立的链表
- $succ(p+1)$：p 的后继节点
- $pred(p)$：p 的前驱结点
- 如果 $pred(p) < k \leq p$，则为结点 p 把自己地址返回给键值为 k 的进程；否则，只需将该请求转发给两个邻接点。
- 时间复杂度：$O(N)$

（2）指示表：

一种路由表

$FT_p[i] = succ(p + 2^{i-1})$，第 i 各实体，指向 p 后 2^{i-1} 个结点。

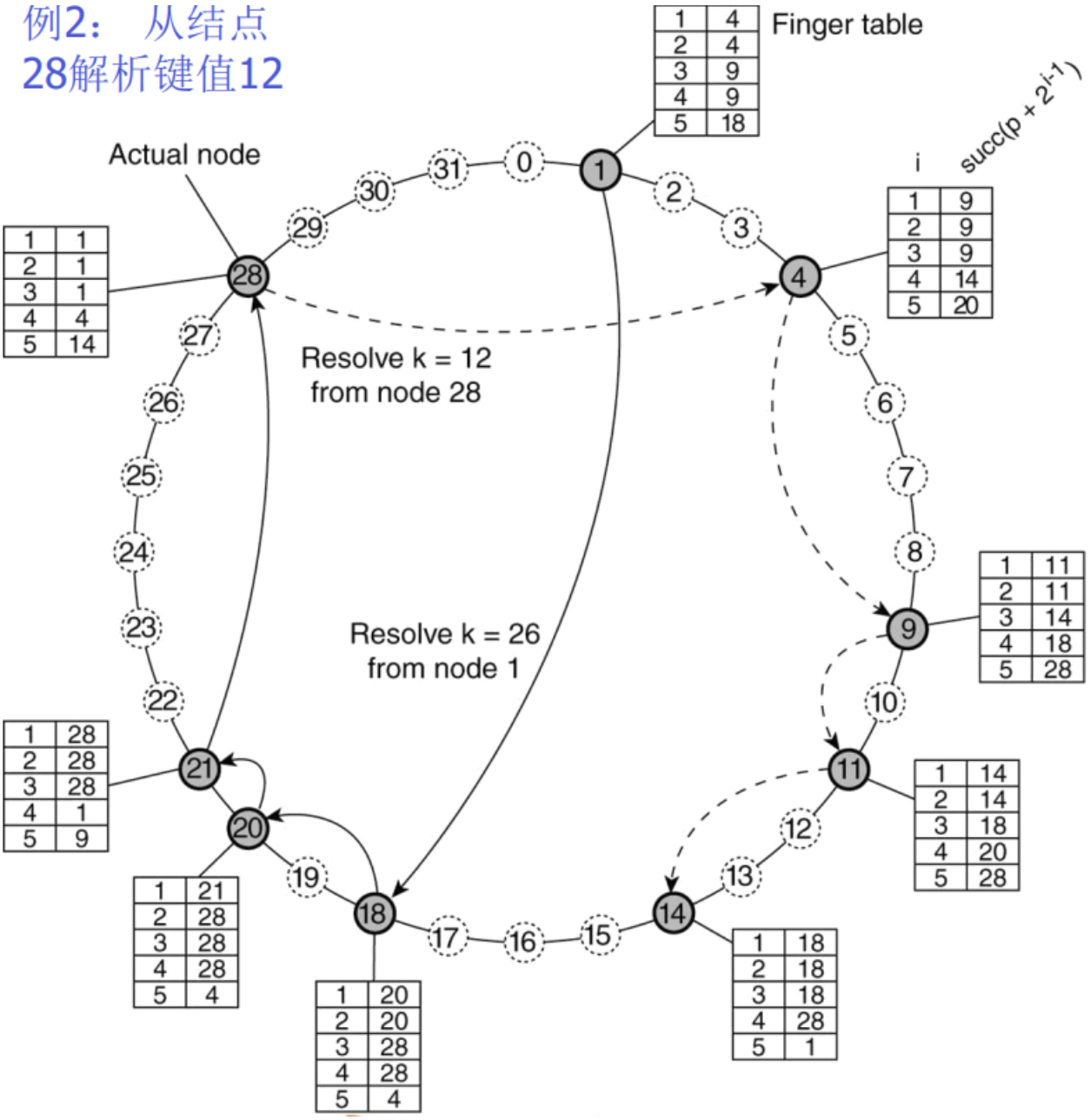
查找键值 k，结点 p 立即把该请求发给 p 的指示标中索引为 j 的结点 q： $q = FT_p[j] \leq k \leq FT_p[j+1]$，j 即为下一跳

时间复杂度： $O(log(N))$

新结点 p 到来：找到 $succ(p+1)$

指示表的更新：

- 检查 $FT_p[1]$ （即 $succ(q+1)$ ）的一致性
- 确定后继节点的前驱节点是否是自己
- 后台进程完成



（3）利用网络邻近的优化策略

跨网路由问题：在解析一个键值时，如通过路径 a–b–c。 而 a–b， b–c 之间是广域通信。 即， 在因特网中产生多次广域消 息传输。

DHT 系统的优化设计：减少因特网中的广域消息传输

考虑底层物理网络的三种设计：

1、基于拓扑的结点表示符赋值

- 两个邻近结点所赋给的标识符是靠近的
- 但不适合 Chord 系统：到因特网映射繁琐、关联失效导致标识符空当

2、邻近路由（proximity routing）

- 每个结点维护一个转发请求的可选列表。例如，每个结点有 r 个后继者
- 当转发请求时，选择离它最近的后继者，减少广域通信
- 单个结点失效时，可使用其他结点

3、邻近邻居结点选择（proximity neighbor selection）

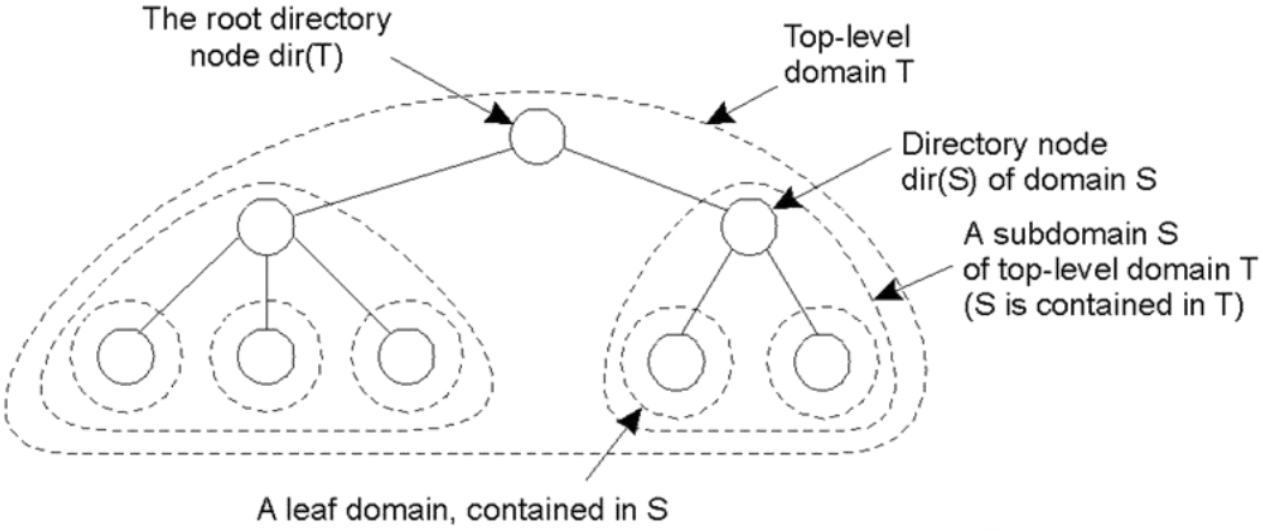
- 优化路由表，选择距离最近的结点为邻居结点
- 对于 Chord，Finger 表中的每个表项包含 r 个后继者时，相当于该方法

2.5 层次方法

将一个定位服务分层组织成域（domain）

- 每个域拥有相关的目录节点
- 根节点：最顶层的目录结点

叶子域对应于局域网（LAN）



定位记录：

- 叶子域目录结点：$<$ 实体ID，当前地址 $>$
- 上层域目录结点：$<$ 实体ID，下层目录结点地址 $>$

根节点：

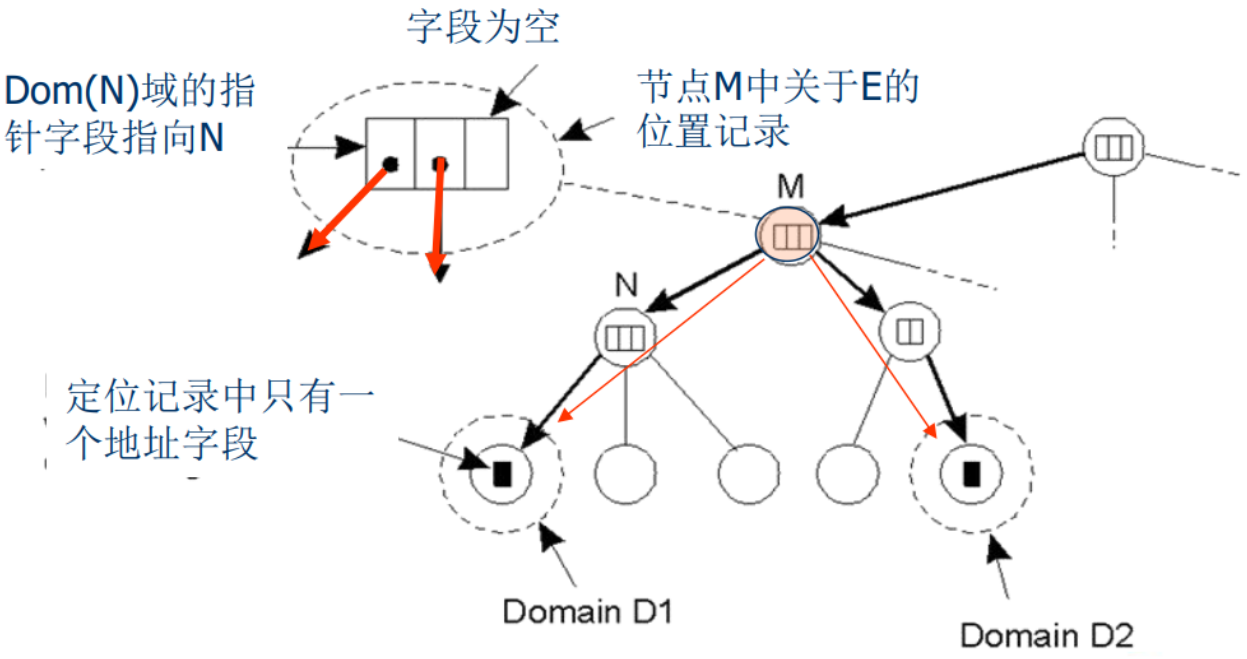
- 包含域中所有的实体的定位记录

一个实体 E 可以拥有 n 个地址 A1，A2

- 每个地址，保存在一个叶子域上，共有 n 个叶子域
- 在 n 个叶子域的最小共同祖先域上，保留 n 个子域指针

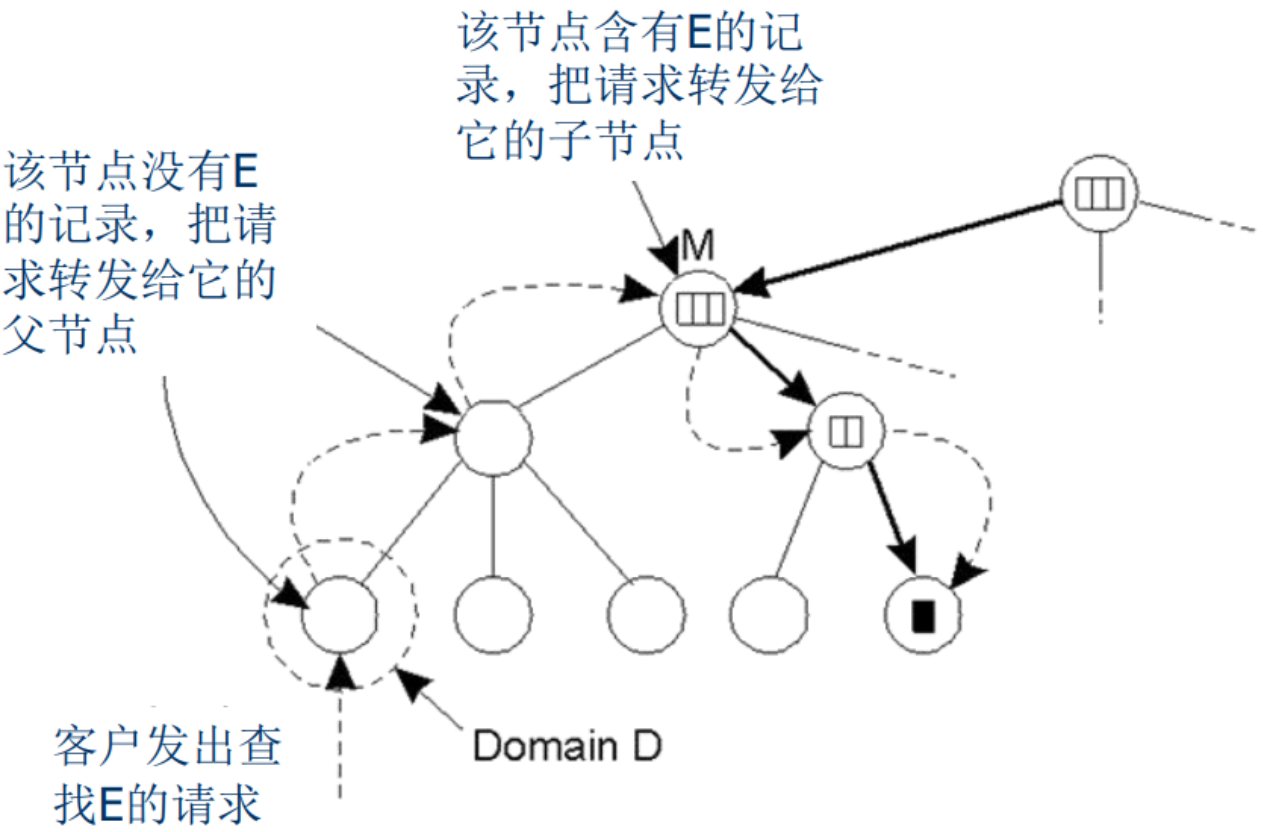
定位记录举例：具有两个地址的实体 E 的目录信息

- D1，D2 叶子域
- M 最小公共祖先

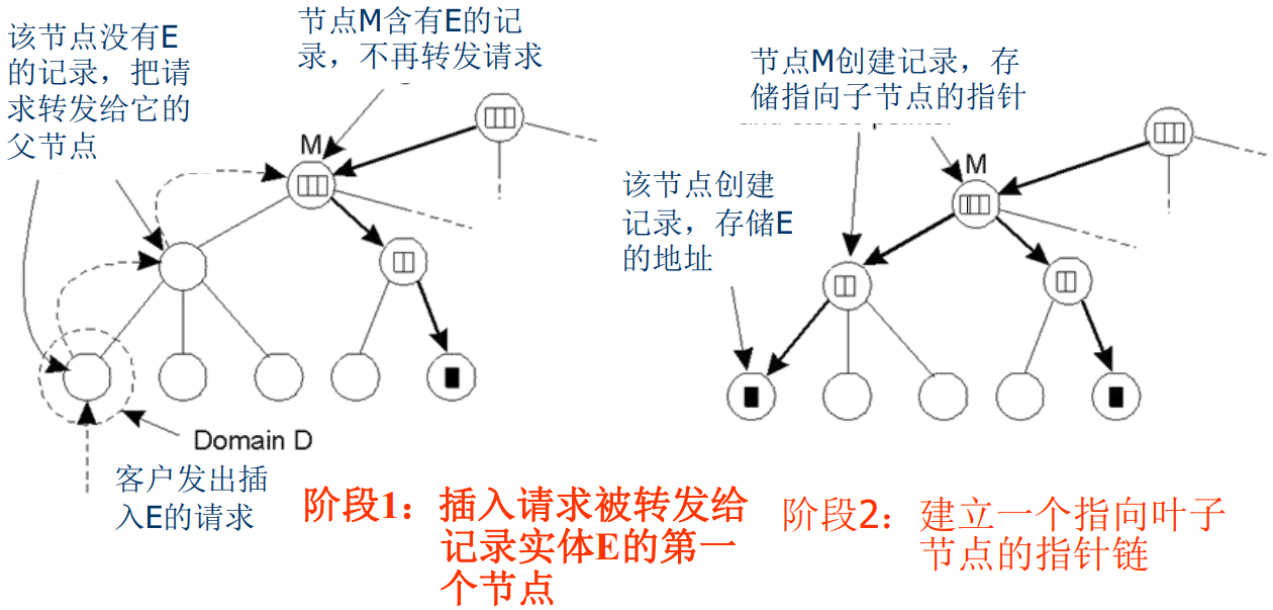


查找操作举例：在层次型定位服务中，查找实体 E 的位置

- 局部性原则：从最小的域开始，自底向上查找



插入操纵举例：自下而上的插入方法



3.结构化命名管理

简单的、可读的名称。例如：ftp.neu.edu.cn（URL 地址）

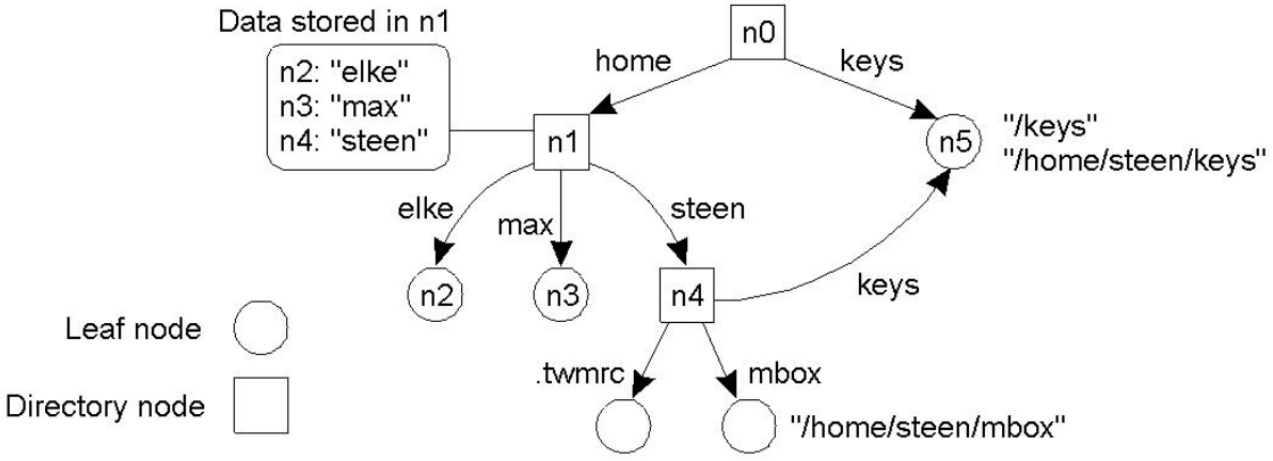
3.1 名字空间

分布式系统中名字集合的组织形式

例：文件目录系统（目录名，文件名）

一个名字空间可表示为带标号的有向图：

- 叶子节点：命名实体的信息（地址、状态）
- 目录结点：＜边标号，结点标识符＞，目录表
- 根节点：只有出边，没有入边。通常只有一个



路径名：对应于边的标号序列

- N：<标号 1，标号 2，....，标号 n>
- 绝对路径名：从根节点开始
- 相对路径名：从非根节点开始

路径的表示：/home/steen/mbox

名字的种类：

- 全局名（global），绝对名，适用于整个系统
- 局部名（local），相对名，与具体目录有关

名字解析（Resolution）

- 找到名字所对应实体， 进而可访问关于实体的信息

闭合机制(Closure)

- 知道如何以及从何处开始名字解析
- 从名字空间中选择开始名字解析的初始节点

- 例： 02483683113

命名系统

- 实现命名和名字解析

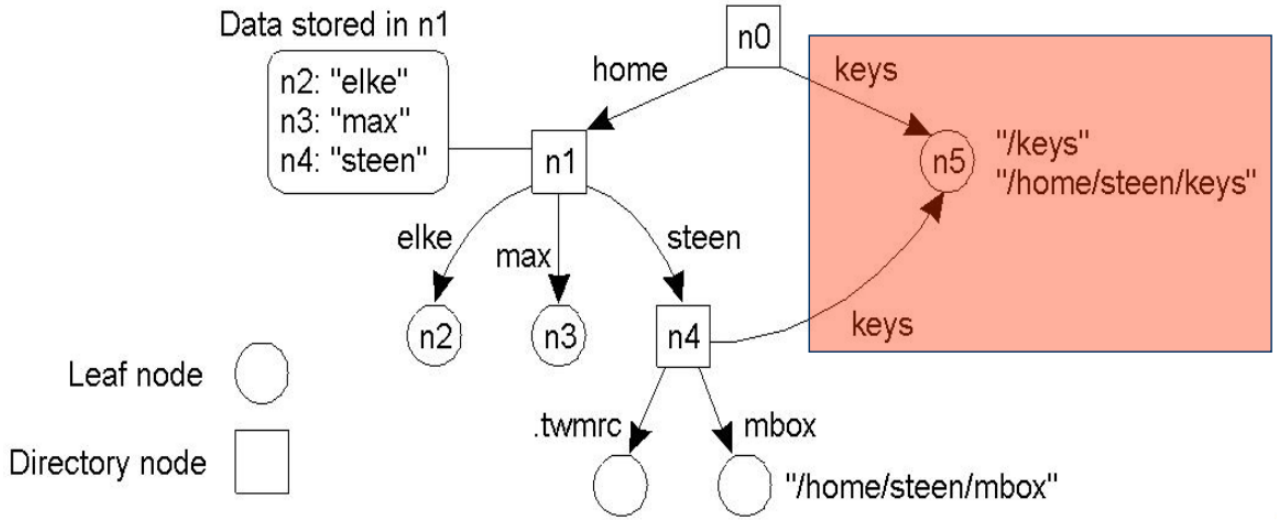
3.2 链接 & 挂接

（1）链接

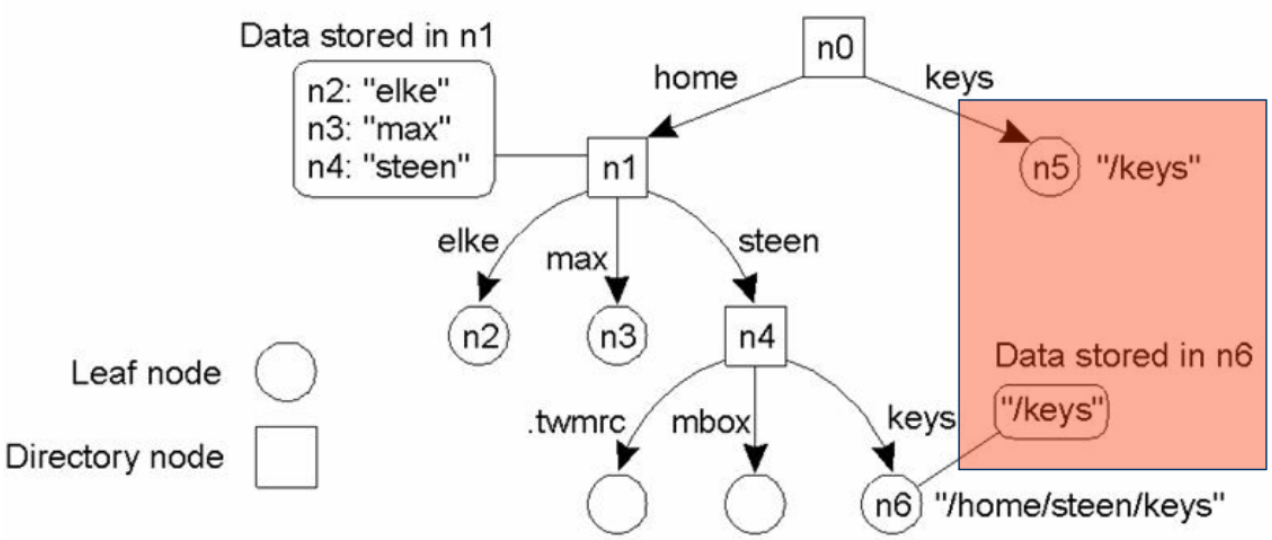
别名（alias）：同一个实体的其他名字

别名的实现方法：链接（link）

- 硬链接(hard)：存储结点标识。用目录结点标识



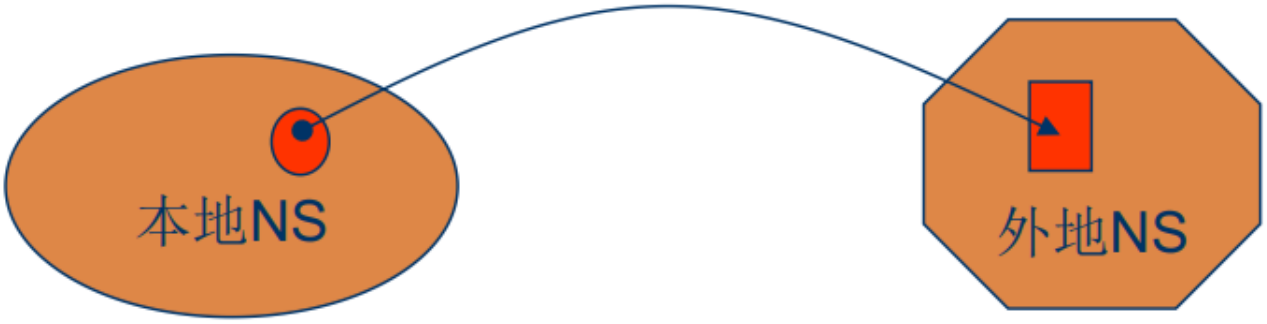
- 符号链接（symbolic）：存储路径别名。用叶子节点表示



（2）挂接

挂接（mount）： 合并两个不同的名字空间

外地名字空间（foreign name space）



挂载托（mount point）：存储外地节点标识符的本地目录节点

挂载件（mounting point）：需安装的外地名字空间的目录节点

（3）挂接方法

分布式系统中挂接的实现

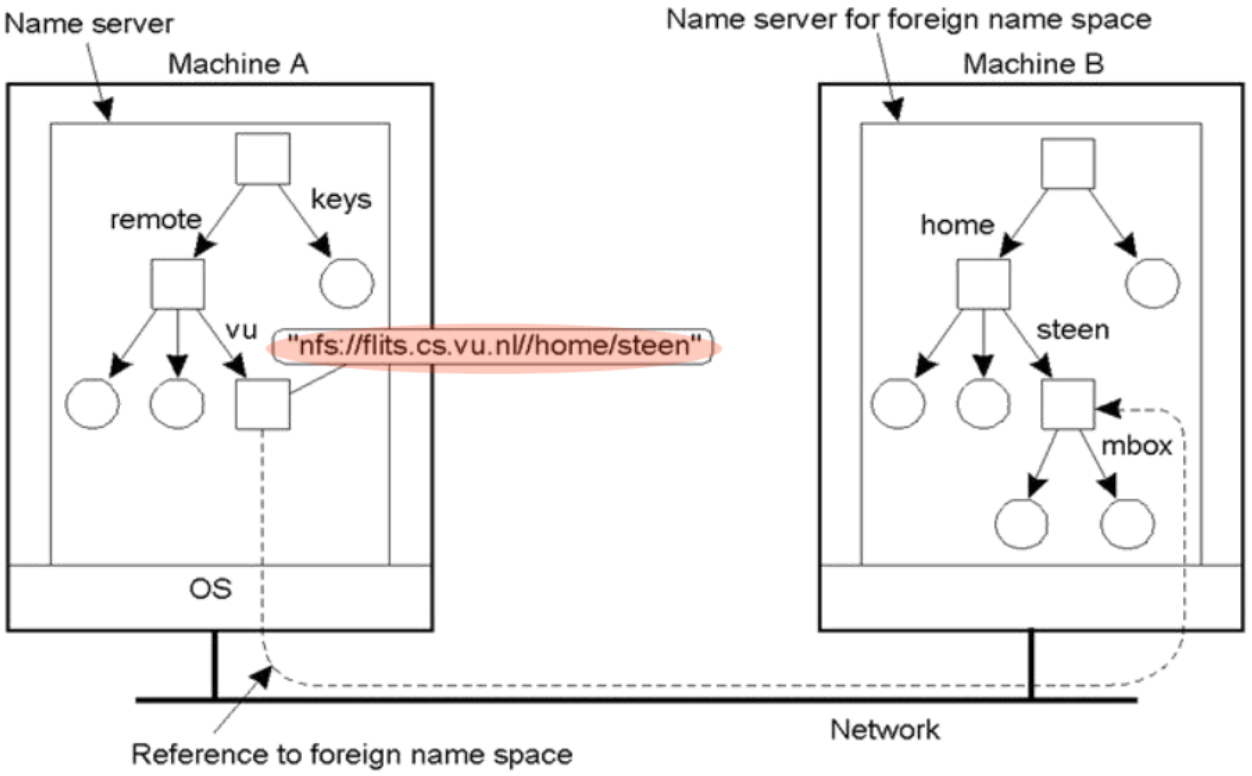
- 访问协议的名字
- 服务器的名字
- 外地名字空间中挂接件的名字

名字的表示

- 例： URL 名， 如 ftp://ftp.neu.edu.cn//home/pub2/yuge
- 访问协议： ftp
- 服务器： ftp.neu.edu.cn
- 安装点： home/pub2/yuge

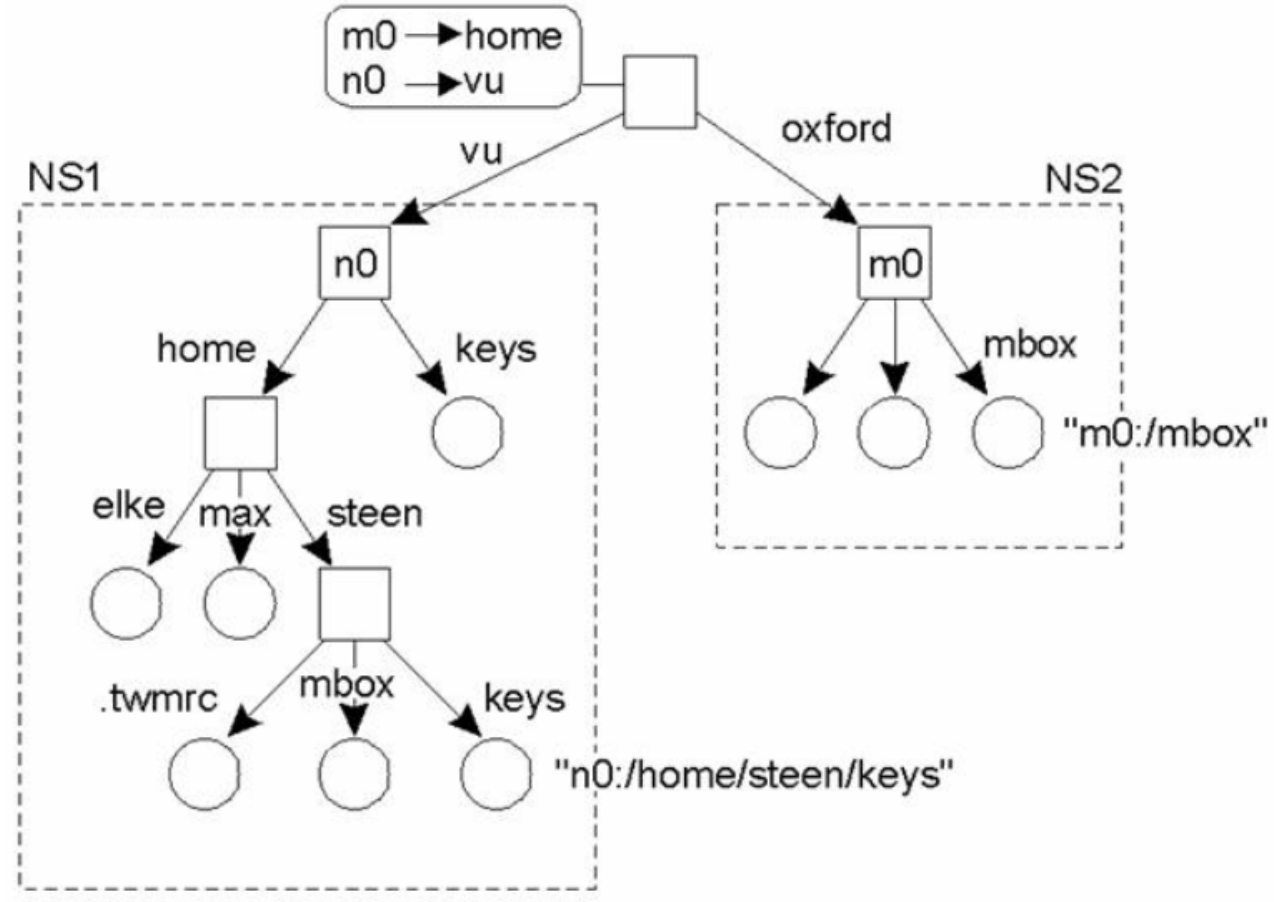
使用专门访问协议， 挂接远程名字空间

- 例： SUN NFS 系统，
- 协议:nfs ； 服务器:flits.cs.vu.nl； 目录:/home/steen



例： DEC Global Name Service 的结构

- 增加一个新根节点， 形成新的名字空间
- 建立映射表（子根节点 ID, 新名字）



3.3 名字空间的实现

名字服务：添加、删除和查找名字

名字服务器：实现名字服务的软件系统

名字空间的分布方式：

- 区域（zone）：水平划分
- 三层结构：垂直划分

（1）名字空间的分布

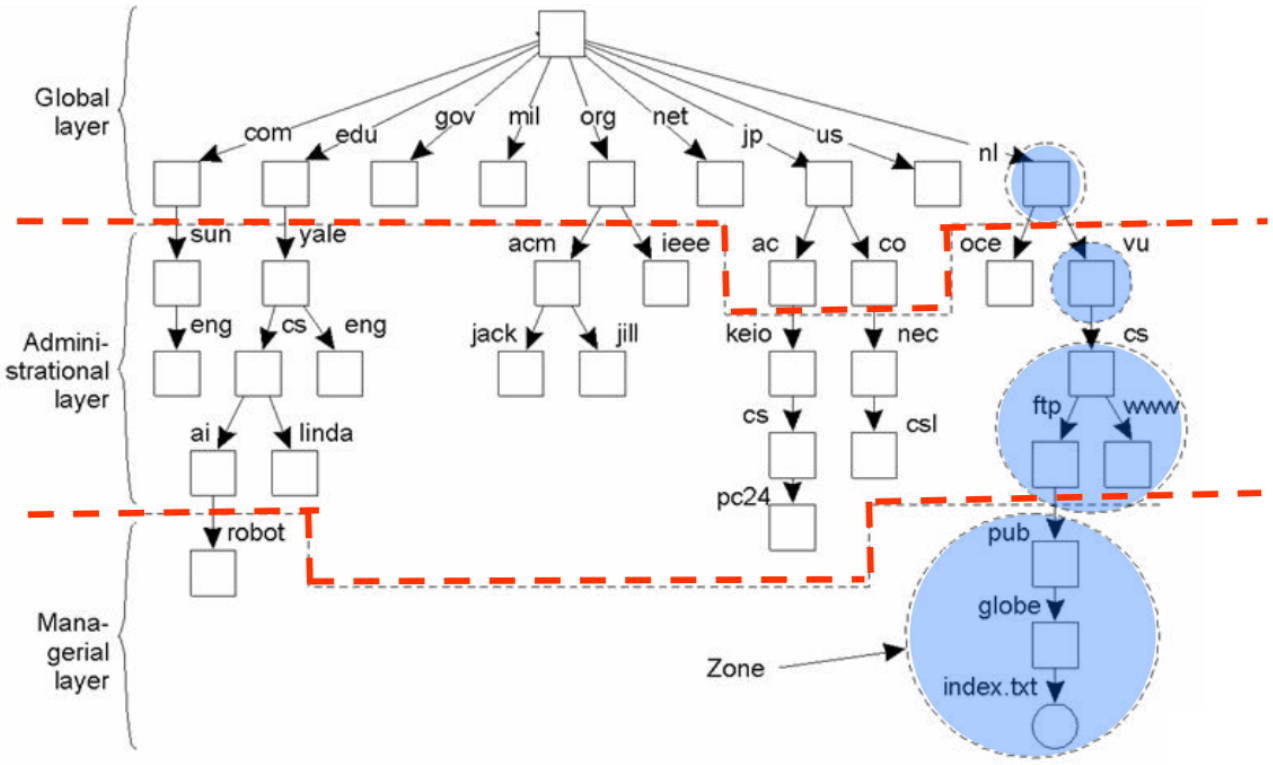
名字空间的层次：

- 全局层（global）：根节点，稳定不变
- 行政层（administrational）：目录结点，很少改变
- 管理层（managerial）：底层节点，可能会经常改变

区域（zone）：

- 不相交的子空间
- 每个区设有一个名字服务器，负责本区的名字服务

DNS 名字空间的三层划分

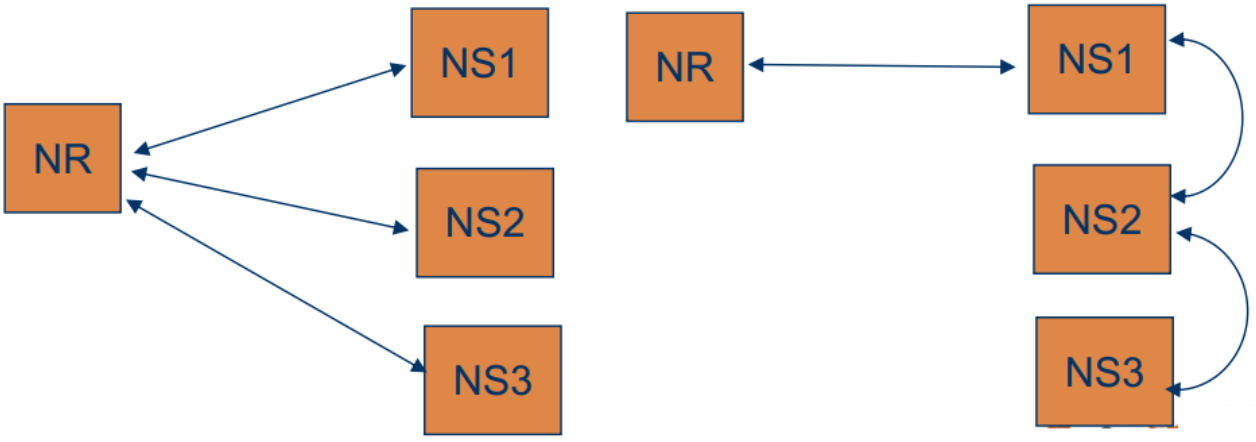


（2）名字解析的实现

名字解析器（NR）：客户端执行名字解析程序

迭代式方法：名字解析器从根名字服务器开始， 逐个与名字服务器交互， 实现名字的解析

递归式方法：名字解析器委托根名字服务器， 由各个名字服务器之间交互， 实现名字的解析



优化：解析中的缓存作用

举例： 递归式名字解析 $< nl, vu, cs, ftp >$

结点服务器	解析	查找	传给子节点	接收和缓存	返回给请求者
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>

3.4 DNS：英特网域名系统

DNS（domain name system）：

- Internet 中查找主机
- 和 email 服务器地址

DNS 名字空间

- 一个带边标号的有向树
- 根节点： 无入边的节点， 用 dot 表示
- 节点： 入边的标识符数为 1， 也可作为该节点的名称
- 域（domain）：一个子树
- 域名： 从根节点开始的路径名
- 区域（zone）： 域的划分， 对应一个名字服务器
- 资源纪录： 节点包含的内容

资源记录

记录类型	管理实体	描述
SOA	Zone	授权起点：保留所表示区域的有关信息
A	Host	地址：包含该节点表示的主机的IP地址
MX	Domain	邮件交换：指向用于处理发给该节点的邮件的邮件服务器
SRV	Domain	服务器：指向处理专门服务的服务器
NS	Zone	名称服务器：指向实现所表示区域的名字服务器
CNAME	Node	规范名称：所表示节点的主名字的符号连接
PTR	Host	指针：主机的规范名字
HINFO	Host	主机信息：该节点表示的主机的信息
TXT	Any kind	文本：实体有用的特别信息

4.基于属性的命名管理

4.1 基本概念

基于属性的命名:

- 一个实体拥有一个相关的属性集
- 例： 一个人， 名字， Alice
- 用（属性， 值） 来描述实体

目录服务:

- 基于属性的命名系统
- 可使用实体的属性查找实体

资源描述框架（RDF） ： 描述资源的统一方法

- <资源， 属性， 属性值>或<主体， 谓语， 客体>
- 例： <Person, name, Alice>

4.2 分层实现方法：LDAP

（1）概念

轻量级目录访问协议（LDAP）

- OSI X.500 目录服务
- 国际电信联盟（ITU） 关于目录服务的建议标准

LDAP 目录项

- 目录项: 由多个记录（属性， 值） 组成。
- 多值属性:用数组或链表表示

目录信息库(DIB)

- 所有目录项的集合。
- 相对区分名（RDN） ： 命名属性
- 每个目录项具有全局唯一的记录名： RDN 值序列

（2）LDAP 目录项

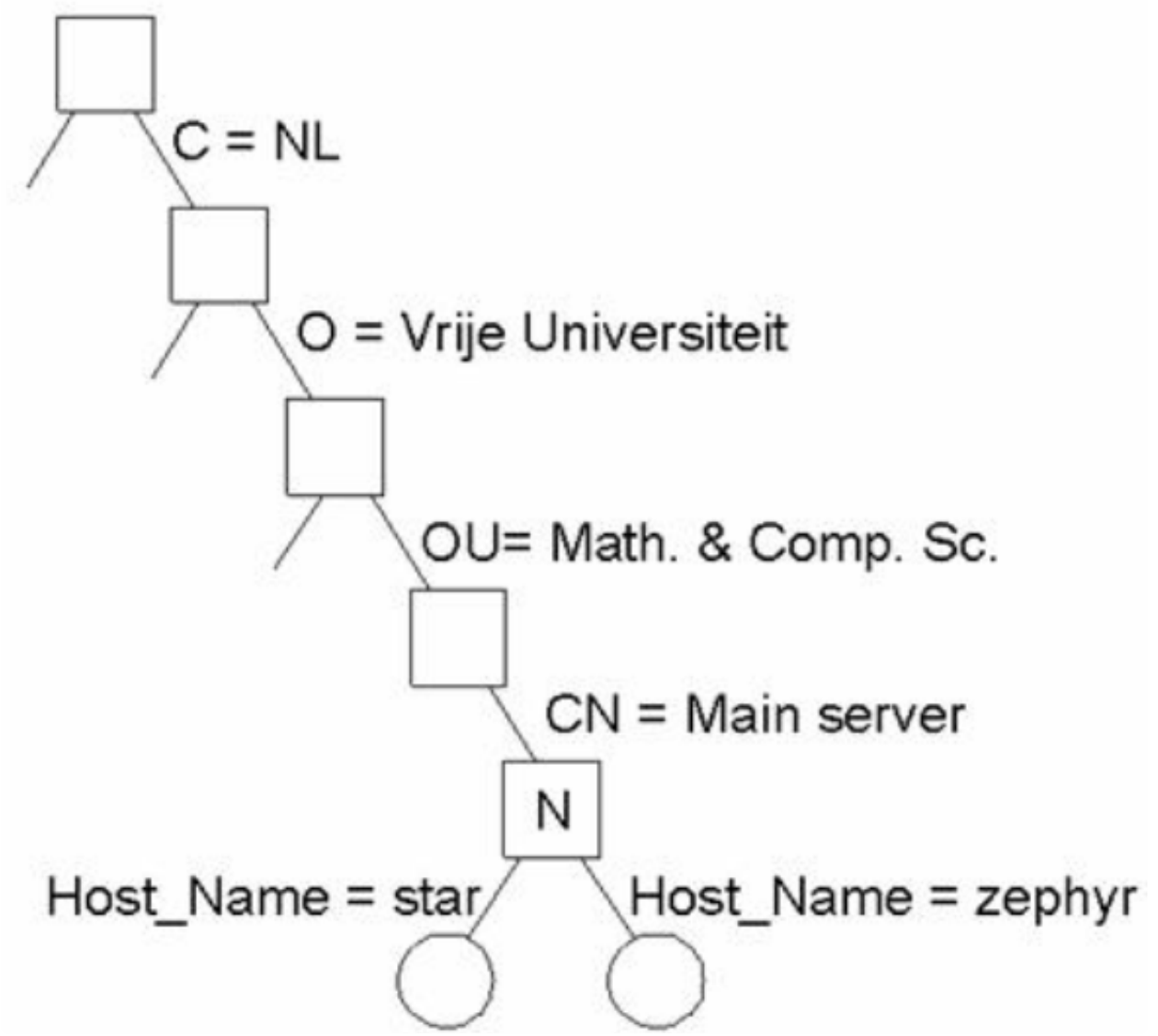
如： `/c=NL/O=Vrije University/OU=Comp.Sc.=nl.vu.cs`

属性	缩写	值
Country	C	NL
Locality	L	Amsterdam
Organization	O	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

（3）目录信息树（DIT）

用于描述目录项集合的层次结构， 命名图

举例： 局部的目录信息树



（3）使用 Host_Name 作为 RDN

目录项1

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

目录项2

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

（4）LDAP 目录服务的实现

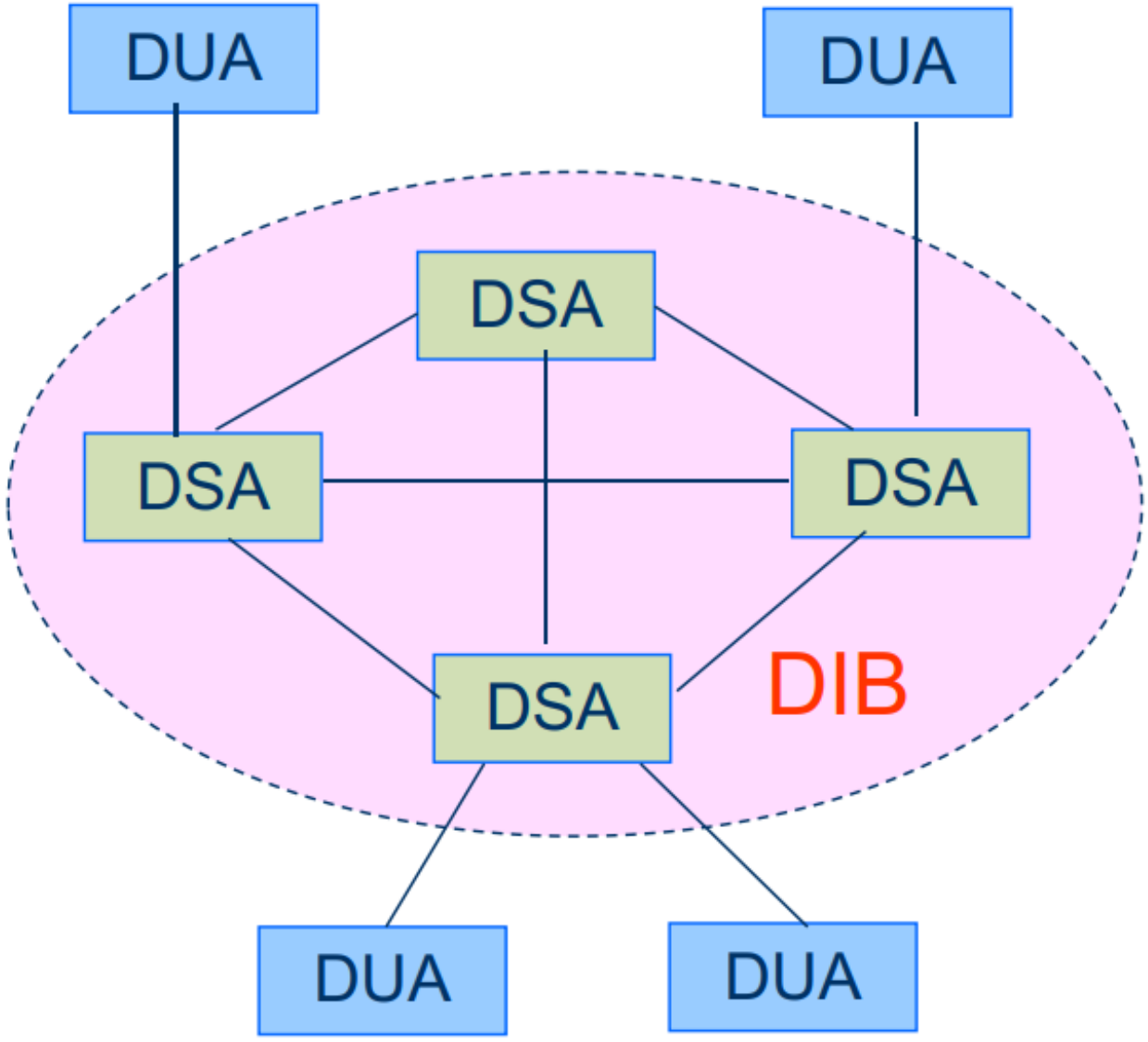
目录服务代理(DSA) ： 管理子 DIB 的服务器

目录用户代理(DUA) ： 代表用户访问目录

目录访问协议(DAP) ： DSA 与 DUA 之间通信

目录系统协议(DSP) ： DSA 之间通信

应用: .Net, Novel, Oracle



（5）LDAP 目录服务的扩展功能

例: 搜索 main server

Answer=search(“&(C=NL)(O=Vrije University)(Ou= *) (CN=Main server)”)

LDAP 森林

- 允许多个目录树共存并相互链接
- 全局索引服务器

LDAP 分层结构

- 根节点采用 DNS 命名系统管理

UDDI：

- Web 服务中的目录服务

4.3 非集中式实现方法