Sistemas Distribuídos

Aula 5 – Nomeação

DCC/IM/UFRRJ
Marcel William Rocha da Silva

Objetivos da aula

- **Aula anterior**
  - *Middleware* de comunicação
    - Comunicação orientada a mensagem
    - Comunicação orientada a fluxo
  - Multicast em SDs

- **Aula de hoje**
  - Conceitos básicos
  - Nomeação simples
    - Broadcast e multicast
    - Ponteiros repassadores
    - Baseada na localização nativa
    - Tabelas hash distribuídas (DHT)
  - Nomeação estruturada
  - Nomeação baseada em atributo

Nomeação

- Nomes podem ter várias finalidades
  - Compartilhar recursos
  - Identificar entidades de maneira única
  - Se referir a localizações
- Nome deve ser “resolvido” para o recurso/entidade/localização a que se refere
  - **Sistema de nomeação**
- Em SDs → Sistemas de nomeação distribuídos
  - Maneira como a distribuição é feita influencia a eficiência e a escalabilidade

Nomeação

- **Entidades** → qualquer coisa em um sistema distribuído
  - Ex.: hospedeiros, impressoras, discos, arquivos, processos, usuários, filas, grupos de discussão, páginas Web, janelas gráficas, mensagens, conexões de rede, etc

- São acessadas através de **pontos de acesso**
  - **Endereços** → nomes dos **pontos de acesso**
  - Ex.: servidor Web e seu número IP e porta

Nomeação

- **Problema** → entidades podem mudar de ponto de acesso!
  - Ex.: servidor Web que é alocado em outra rede
  - Não é interessante usar endereços para nomear entidades!

- **Solução** → nomes independentes da localização
  - **Identificadores**
  - **Nomes amigáveis a seres humanos**

Identificadores

- **Identificador** → nomeia uma entidade de maneira exclusiva

- Possuem as seguintes propriedades:
  - UM identificador referencia APENAS UMA entidade
  - UMA entidade possui APENAS UM identificador
  - Identificadores NÃO são reutilizáveis

- Normalmente, são uma cadeia de bits
  - Ex.: CPF

Nomes amigáveis

- Objetivo → ser facilmente utilizado por seres humanos
  - Legível

- Normalmente, são uma cadeia de caracteres
  - Pathnames, URLs, nomes de domínio
  - Ex1.: /home/marcel/aula1.pdf
  - Ex2.: http://www.cc.ufrrj.br

Sistemas de nomeação

- Mantém uma vinculação **nome-endereço**
- Na forma mais simples...
  - **Tabela de pares** **`(nome, endereço)`**
  - **Problema** → em sistemas de grande porte, uma tabela centralizada não vai funcionar bem
- Em SDs...
  - Uso de uma **tabela distribuída**
  - Sequência de consultas é necessária para encontrar a entrada desejada (resolução de nome)

Sistemas de nomeação

- Três classes:
  - **Nomeação simples**
  - **Nomeação estruturada**
  - **Nomeação baseada em atributo**

Nomeação simples

- Se aplicam a **identificadores**
  - Nomes são cadeias simples de bits → **nomes simples**
  - Não contém informação sobre como localizar o endereço da entidade associada

- Alguns tipos de sistemas de nomeação simples
  - **Broadcast e multicast**
  - **Ponteiros repassadores**
  - **Baseada na localização nativa**
  - **Tabelas hash distribuídas (DHT)**

Nomeação simples: Broadcast

- Recursos oferecidos em uma **rede local** onde todas as máquinas estão conectadas

- Funcionamento (semelhante ao ARP)
  1. Mensagem contendo o identificador da entidade é enviada em broadcast
  2. Cada máquina verifica se possui esta entidade
  3. Máquina(s) com ponto de acesso para a entidade respondem com o endereço

Nomeação simples: Broadcast

- **Problema** → ineficiente em redes grandes
  - Largura de banda é desperdiçada com o tráfego de mensagens de requisição e resposta
  - Aumenta a probabilidade de mensagens colidirem → redução na vazão
  - Máquinas são interrompidas o tempo todo com requisições que não podem responder

- Solução → **multicast**
  - Enviar mensagens apenas para um grupo restrito de máquinas → possíveis

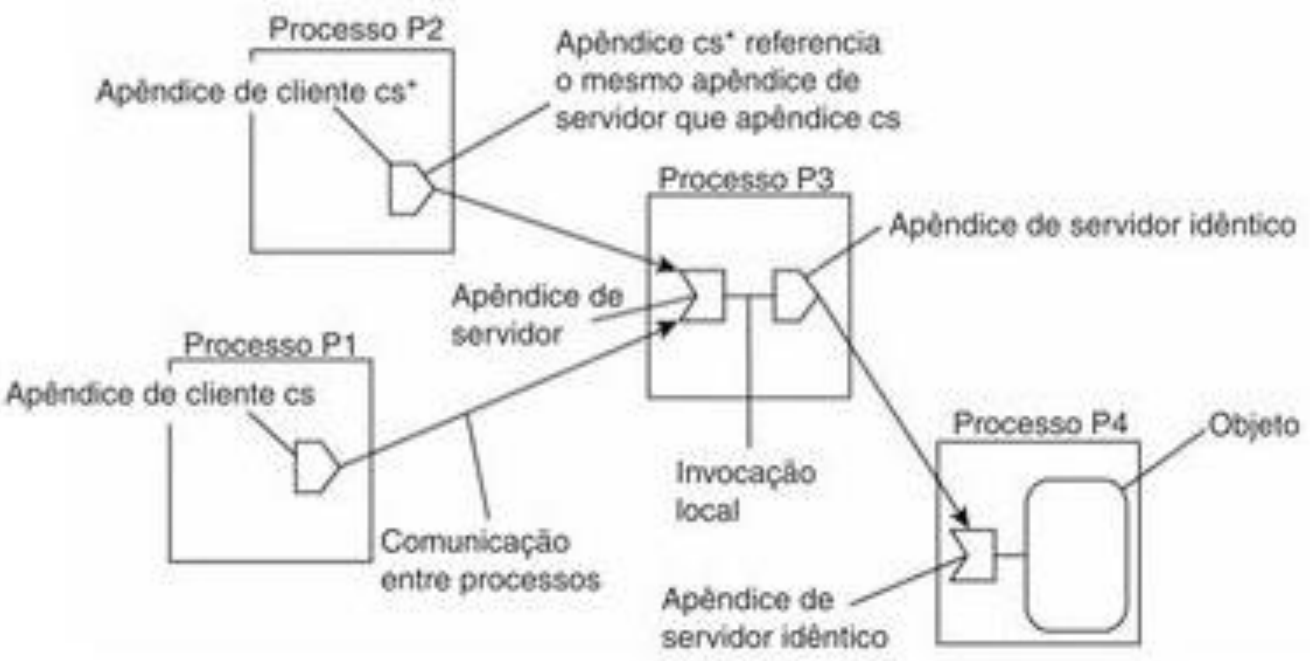
Nomeação simples: ponteiros repassadores

- Princípio → Se uma entidade se move de A para B, deixa para trás em A uma referência para a nova localização em B
  - Endereço atual da entidade pode ser encontrado seguindo a cadeia de ponteiros repassadores
  - Necessário ainda sistema de nomeação para achar o endereço original da entidade!

- **Problemas**
  - Cadeia pode ficar muito grande dependendo da mobilidade
  - Intermediários terão de manter as referências
  - Muito vulnerável a perda de “elos” da cadeia

Nomeação simples: ponteiros repassadores

- Analogia de **objetos** e **chamadas de procedimento remoto**



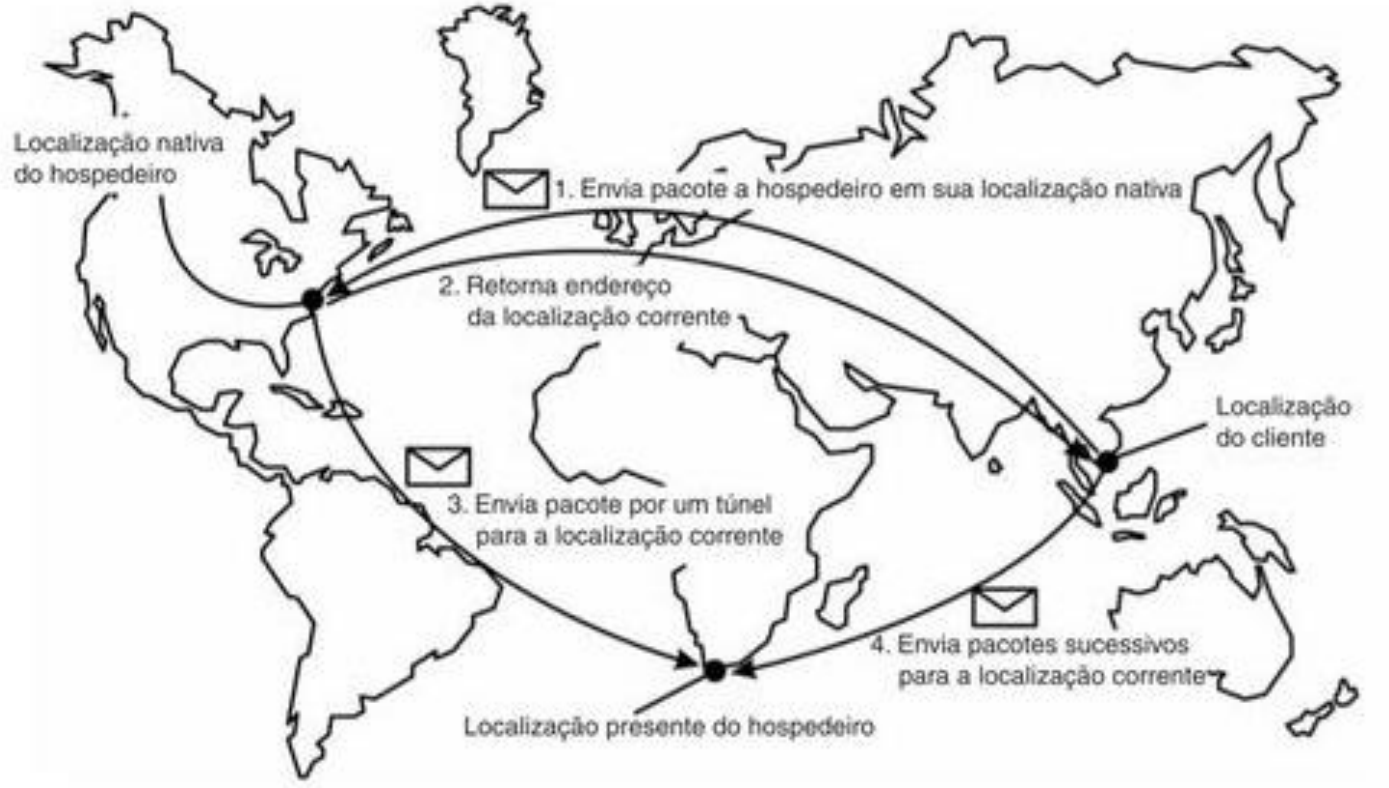
Nomeação simples: Localização nativa

- Utilizar uma localização única (nativa) como endereço da uma entidade
  - Endereço nativo nunca muda
  - Geralmente, endereço onde entidade foi criada
  - Mantém informação sobre a localização atual da entidade → **endereço externo** (*care-of-address* - COA)

- Mensagens para entidade chegam na localização nativa
  - São repassadas para o endereço externo atual
  - Remetente também é redirecionado para o endereço externo

Nomeação simples: Localização nativa

- **Problema** → endereço externo muito distante da localização nativa



Nomeação simples: DHT

- *Distributed hash tables*

- Exemplo → Sistema **Chord**
  - Usa um espaço de identificadores de ***m*** bits para designar nós e entidades específicas (arquivos, processos, etc)
  - Número ***m*** bits é usualmente 128 ou 160
  - Entidade com chave ***k*** está sob a jurisdição do nó que tenha o menor identificador ***id >= k → succ(k)***

Nomeação simples: DHT

- Desafio → Traduzir uma chave ***k*** (identificador) em ***succ(k)*** (endereço)

- Duas abordagens:
  - **Abordagem linear**
  - **Tabela de derivação** (*finger table*)

Nomeação simples: DHT

- **Abordagem linear**
  - Cada nó ***p*** conhece o sucessor ***succ(p+1)*** e o predecessor ***pred(p)***
  - Ao receber uma requisição para a chave ***k, p*** repassa a requisição para os seus vizinhos, a menos que ***pred(p) < k <= p*** → ***p*** retorna o próprio endereço

- Problema → Não escalável!

Nomeação simples: DHT

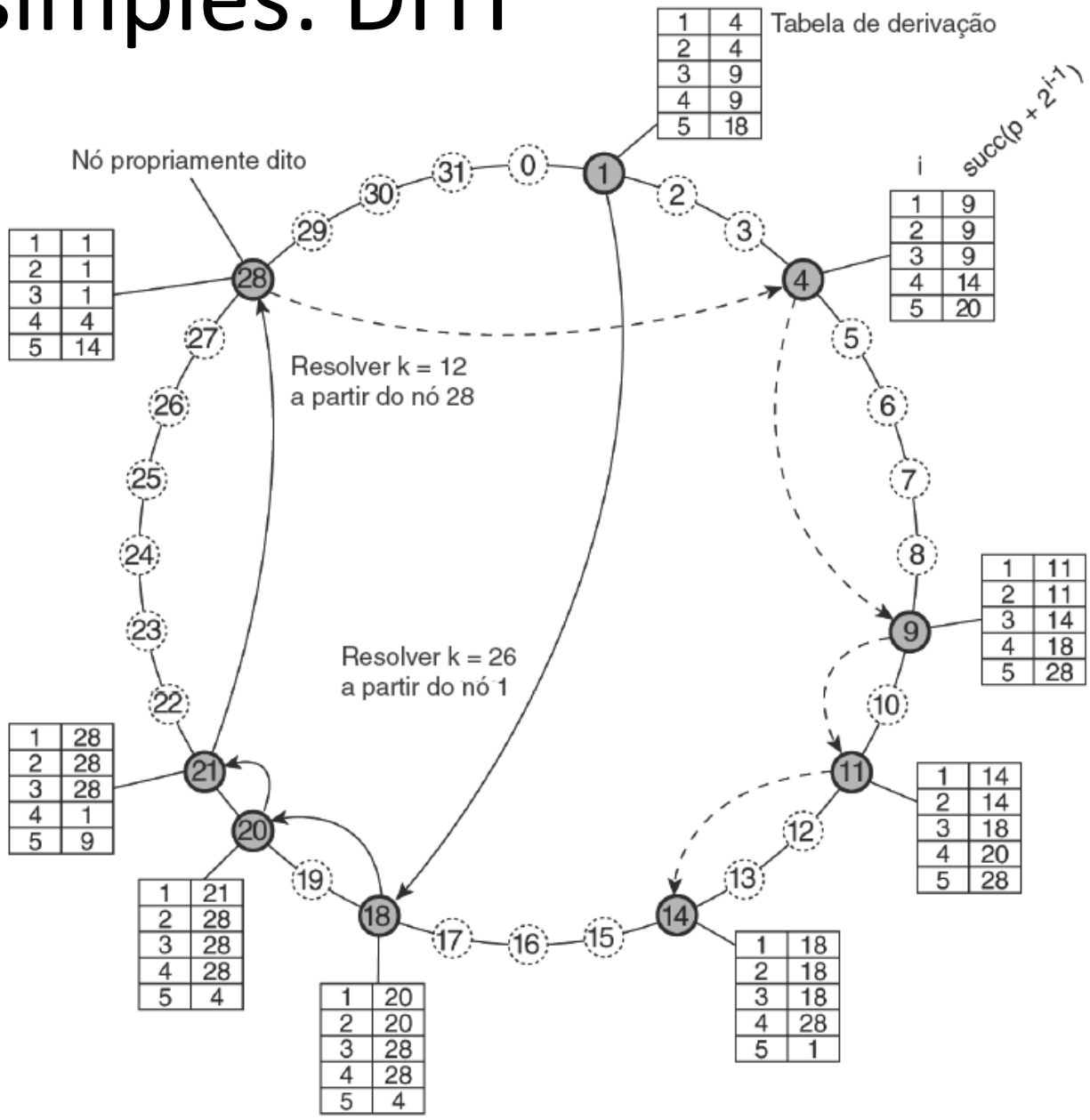
- **Tabela de derivação** (*finger table*)
  - Possui, no máximo, ***m*** entradas
  - Denotando a tabela de derivação de p por Ft_p

$$Ft_p[i] = succ(p + 2^{i-1})$$

  - i-ésima entrada aponta para o primeiro nó que sucede ***p*** por no mínimo 2^{i-1}

Nomeação simples: DHT

- Se ***p = 4*** recebe uma requisição para ***k = 7***
  - ***succ(p+1) = 9***
  - Repassa a requisição ao nó ***9***
- Se ***p = 4*** recebe uma requisição para ***k = 3***
  - ***pred(p) = 1***
  - ***1 < 3 <= 4***
  - Retorna o próprio endereço ***4***



Nomeação simples: DHT

- Como encontrar uma entidade ***k***?
  - Referências na tabela de derivação são **atalhos** para nós existentes no espaço de identificadores
  - Distância do atalho em relação ao nó ***p*** aumenta exponencialmente à medida que o índice na tabela de derivação cresce
  - Para consultar uma chave ***k,*** o nó ***p*** repassará a requisição ao nó ***q*** com índice ***j*** na tabela de derivação de ***p***

$$q = Ft_p[\, j \,] <= k <= Ft_p[j+1]$$

Nomeação simples: DHT

1) Considere a resolução de ***k=12***, a partir do nó ***28***

2) Nó ***28*** consultará ***k=12*** e verificará que:

FT[4] < k < FT[5]

3) Requisição será repassada para o nó ***4***

4) O nó ***4*** selecionará o nó ***9***, porque:

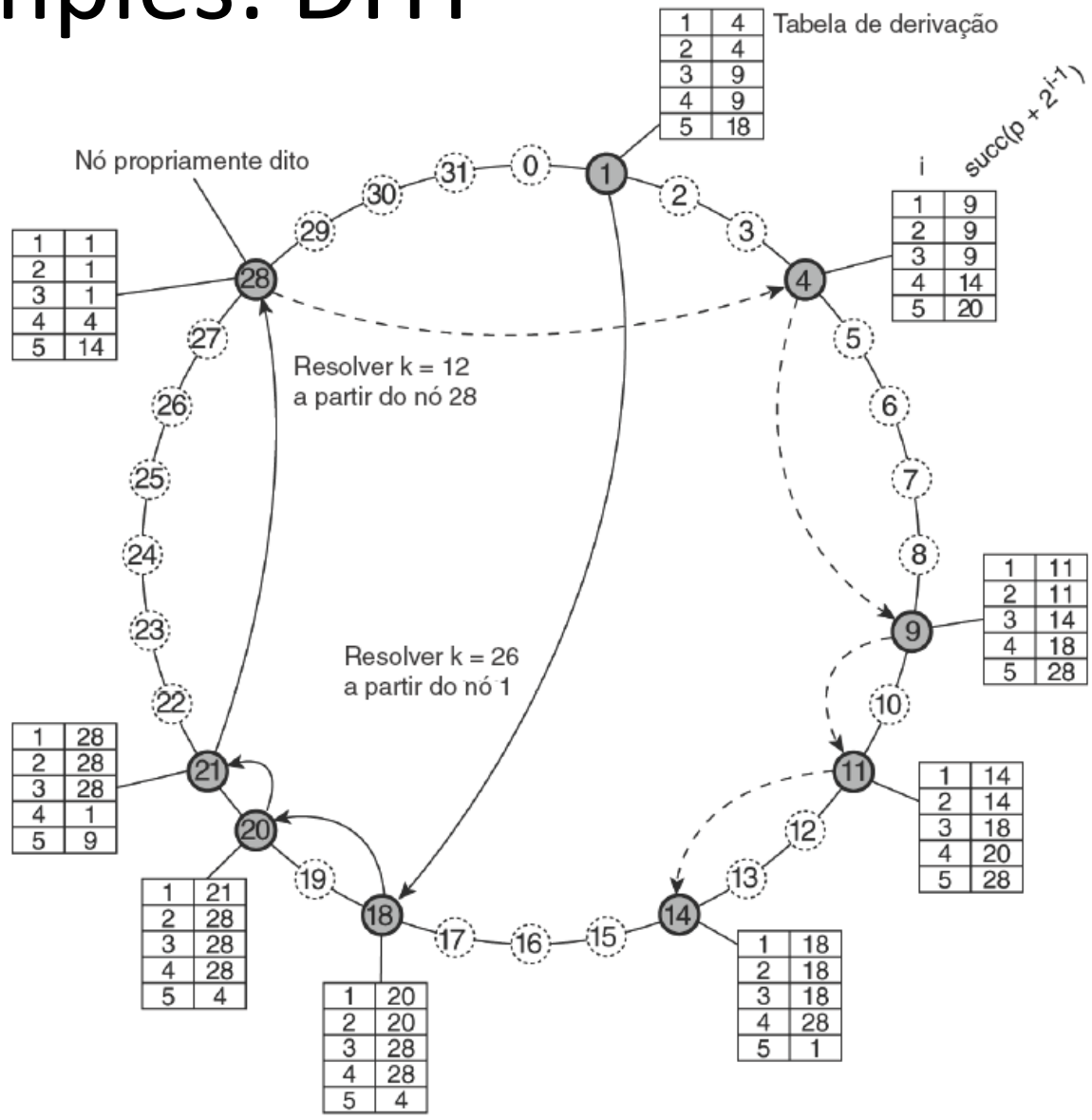
FT[3] <= k < FT[4]

5) O nó ***9*** selecionará o nó ***11***, pois:

FT[2] <= k < FT[3]

6) O nó ***11*** repassará o pedido ao nó ***14***, pois:

FT[1] > k



Nomeação estruturada

- Nomes simples (identificadores) → bons para máquinas, mas não são convenientes para a utilização por seres humanos

- Alternativa → uso de sistemas de **nomeação estruturada**
  - Nomes estruturados
  - Ex.: sistemas de arquivos no Unix, hostnames na Internet

Nomeação estruturada

- Nomes são normalmente organizados em um **espaço de nomes**

- **Espaço de nomes simples** → espaço plano
  - Ex.: Identificadores de m bits (0 até 2^m-1)

- **Espaço de nomes estruturado** → grafo dirigido
  - **Nós-folha**: armazenam as informações (entidades, endereços, etc)
  - **Nós diretório**: armazenam referências para outros nós
    - **Tabela de diretório** com entradas do tipo:

<nome do ramo, nome do nó>

Nomeação estruturada

- **Espaço de nomes estruturado**
  - **Nó raiz**: possui somente ramos de saída (geralmente, uma raiz um por grafo)
  - Caminho no grafo de nomeação é dado por uma sequência de rótulos (*labels*) de arestas:

    `N:<label-1, label-2, ..., label-n>`
  - Caminhos podem ser:
    - **Absolutos** → primeiro nó do caminho é a raiz
    - **Relativos** → primeiro nó diferente da raiz

Nomeação estruturada

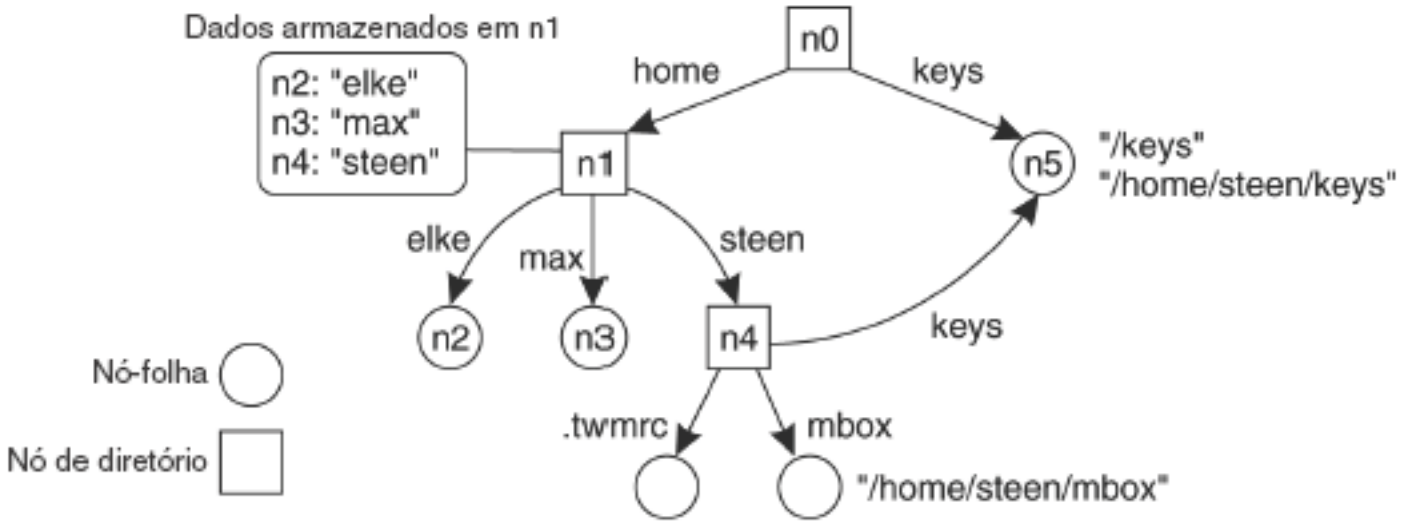


Figura 5.9 Gráfico de nomeação geral com um único nó-raiz.

Nomeação estruturada

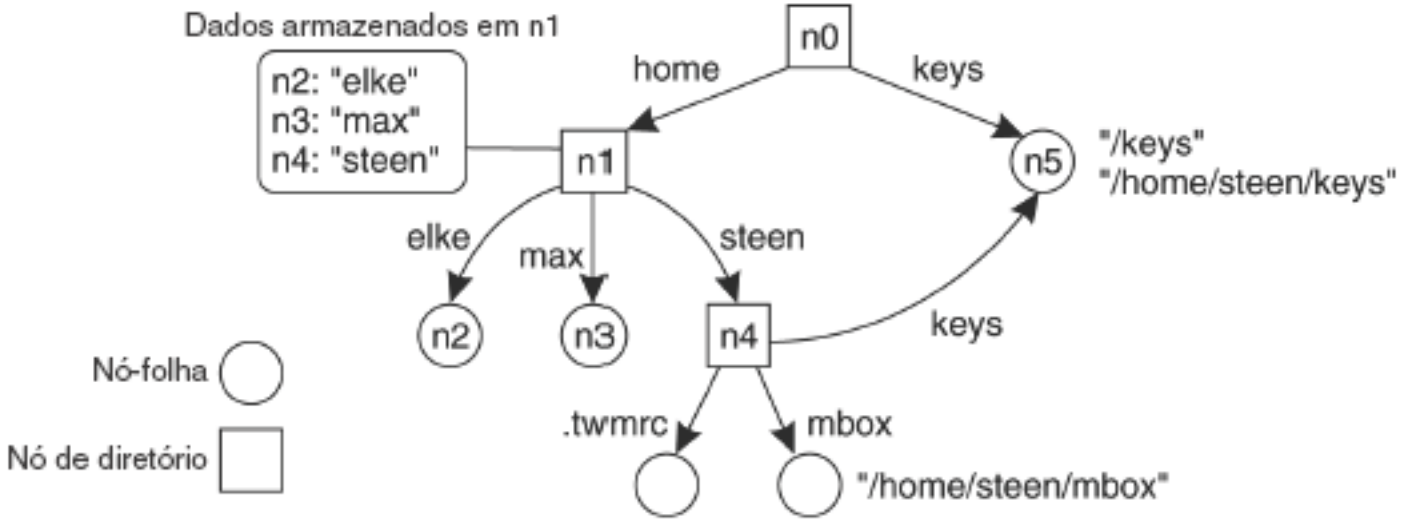
- **Resolução de nomes** → mecanismo usado para buscar uma entidade (ou endereço) referenciada por um nome estruturado
- Dado um **nome de caminho**, deve ser possível consultar qualquer informação armazenada no nó referenciado por aquele nome
  - Basta seguir os ramos do caminho no grafo
- **Problema** → condição inicial
  - Precisamos de uma referência para algum nó a fim de iniciar a resolução de nomes!

Mecanismo de fechamento

- Seleciona um nó inicial do grafo onde deve-se iniciar a resolução de nomes
  - Geralmente, o nó raiz
- Na maioria dos casos são implícitos ao contexto em que se aplicam
  - Ex1.: DNS → todos sabem os endereços dos servidores raiz
  - Ex2.: Sistema de arquivos Unix → primeiro inode do disco lógico

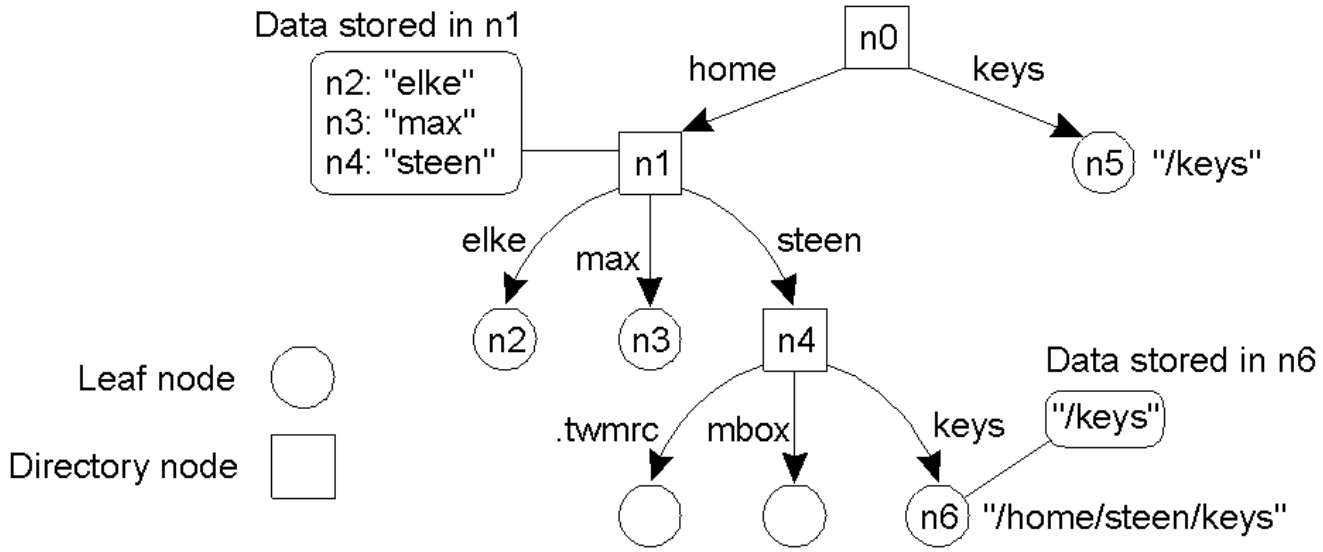
Alias (apelido)

- Nome alternativo para uma entidade
  - **Nomes absolutos**
    - Ex.: **`/home/steen/keys`** e **`/keys`**



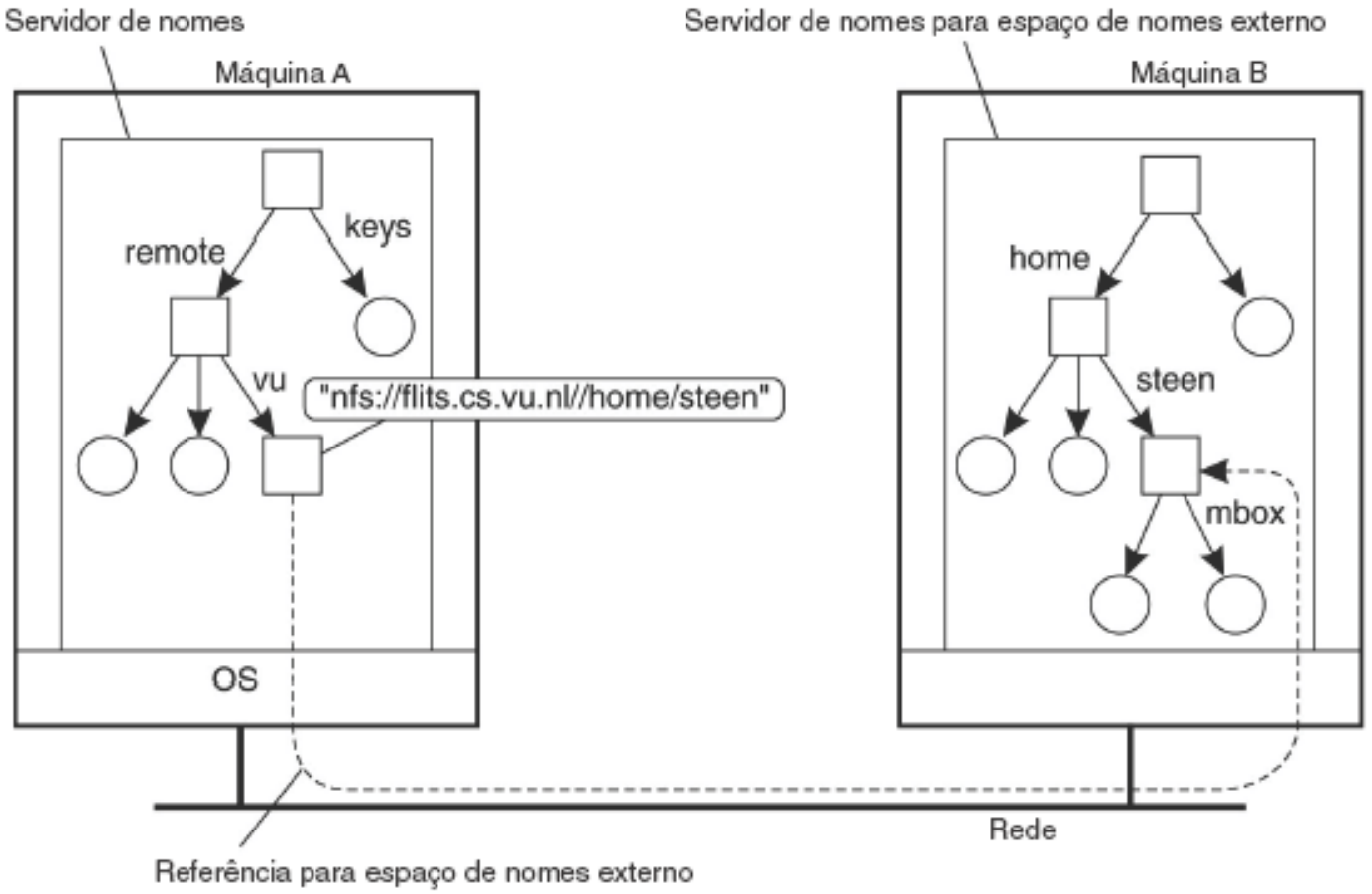
Alias (apelido)

- Nome alternativo para uma entidade
  - **Links simbólicos**
    - Ex.: referência para **`/keys`** armazenada em **`/home/steen/keys`**



Montagem (*mounting*)

- Permite a união de diferentes espaços de nomes de maneira transparente



Implementação de um espaço de nomes

- **Servidores de nomes**
  - Em redes locais → servidor centralizado
  - Em grande escala → vários servidores de nomes distribuídos

- Servidores de nomes distribuídos utilizam a hierarquia do espaço de nomes para a sua estruturação

Implementação de um espaço de nomes

- Hierarquia subdividida em 3 camadas
  - **Camada global**
    - Raiz e seus filhos
    - Principal característica: Estabilidade
    - Podem representar organizações
  - **Camada Administrativa**
    - Nós de diretórios
    - Gerenciados por uma única organização
    - Relativamente estáveis
  - **Camada Gerencial**
    - Nós cujo comportamento típico é a mudança periódica
    - Mantidos por administradores de sistemas e usuários finais

Implementação de um espaço de nomes

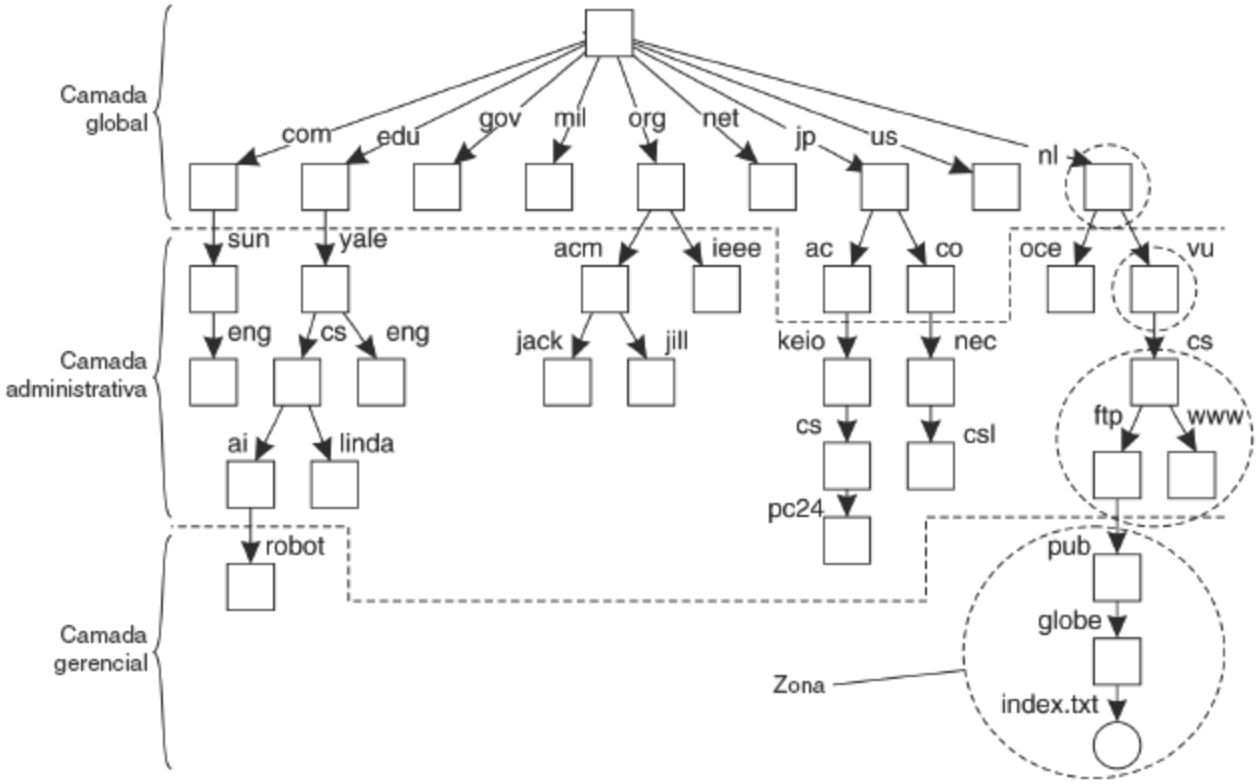


Figura 5.13 Exemplo de repartição do espaço de nomes DNS, incluindo arquivos acessíveis pela Internet, em três camadas.

Implementação da resolução de nomes

- Dois tipos:
  - **Resolução iterativa**
    - Servidor responde somente o que sabe → nome do próximo servidor que deve ser buscado
    - Cliente procura iterativamente os outros servidores
  - **Resolução recursiva**
    - Servidor passa o resultado para o próximo servidor que encontrar
    - Para o cliente, somente existe uma mensagem de retorno: o endereço do nome ou 'não encontrado'

Implementação da resolução de nomes

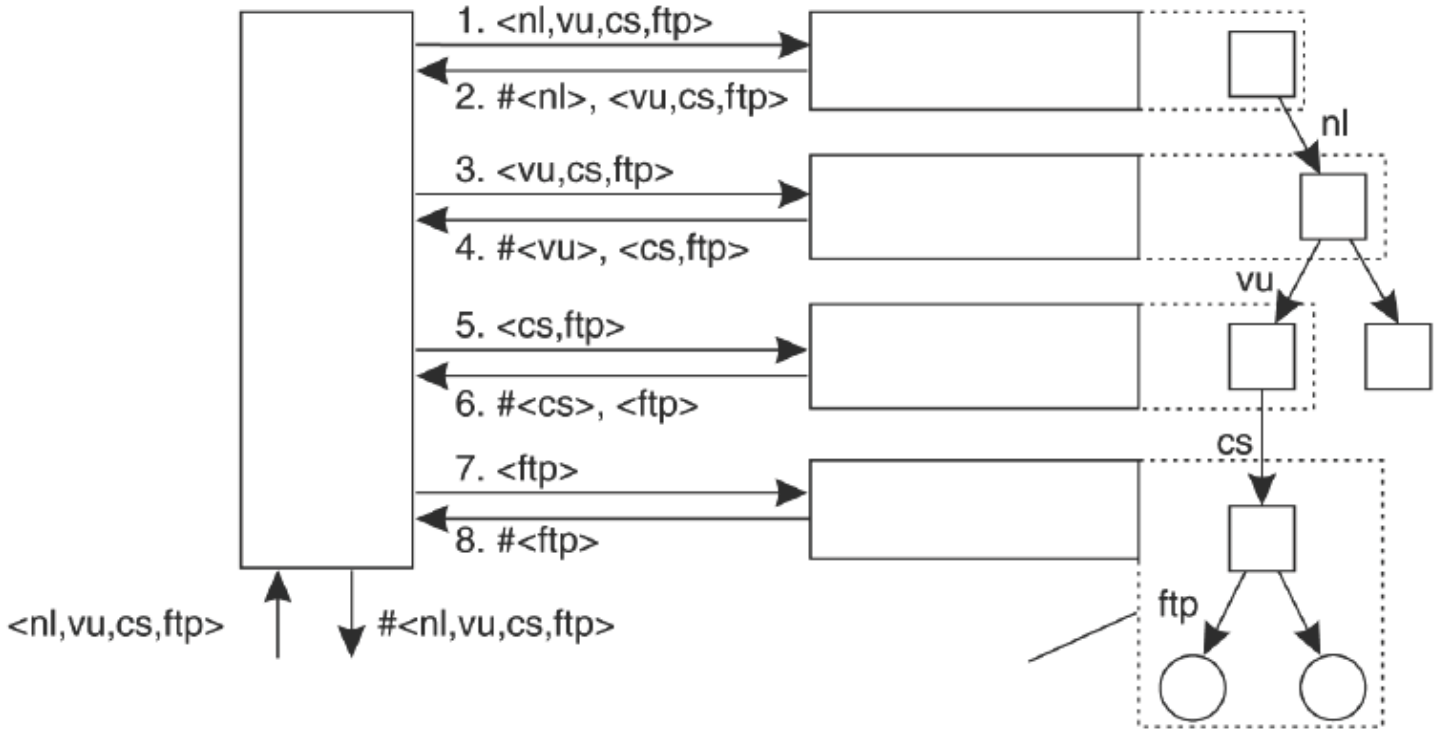


Figura 5.14 Princípio da resolução iterativa de nomes.

Implementação da resolução de nomes

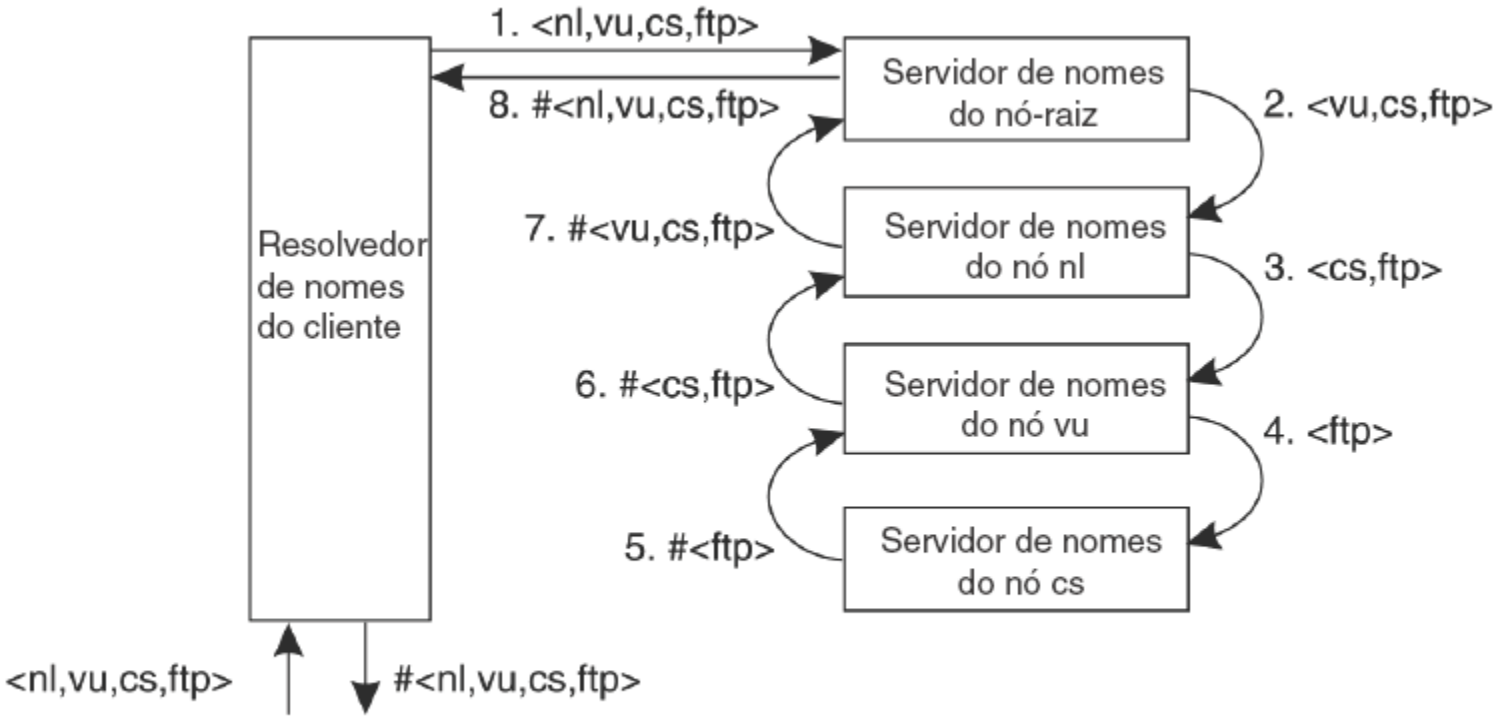


Figura 5.15 *Princípio da resolução recursiva de nomes.*

Iterativa versus recursiva

Tipo	Carga na Camada Global	Cache	Comunicação
Iterativa	Baixa	Menos Eficiente	Custosa
Recursiva	Alta	Mais Eficiente	Menos custosa

Iterativa versus recursiva

- Exemplo:

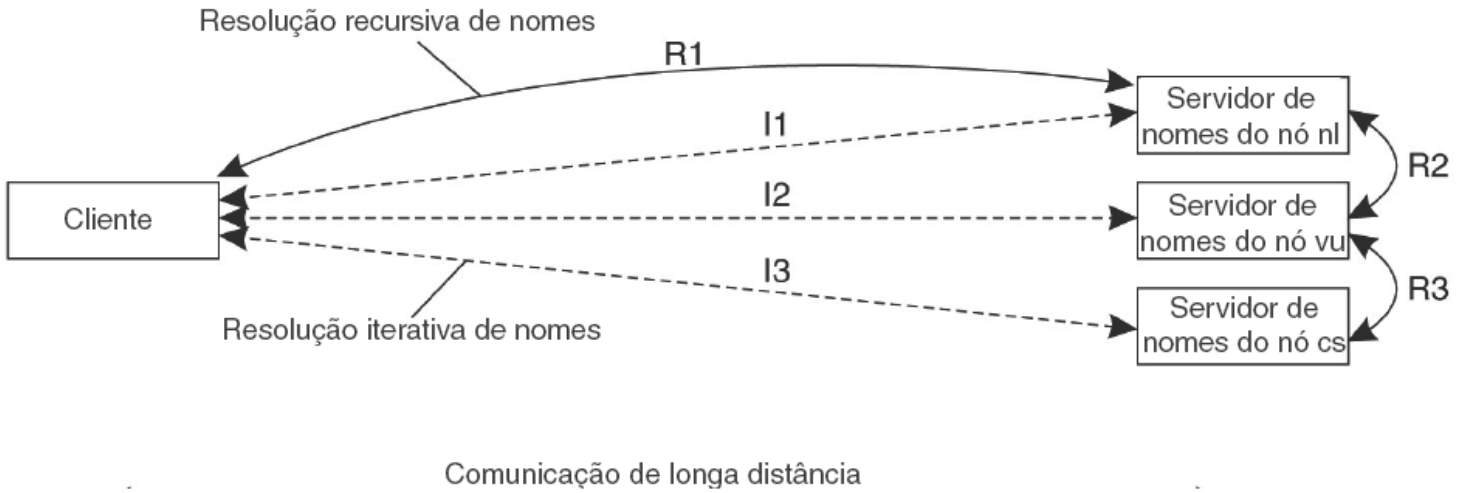
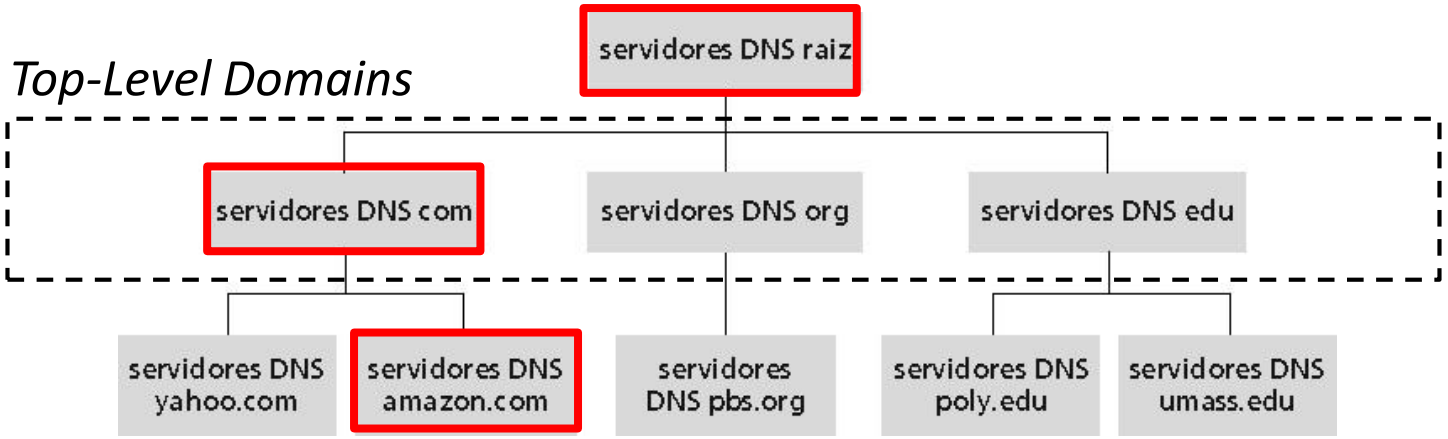


Figura 5.16 Comparação entre resolução recursiva e iterativa de nomes no que diz respeito aos custos de comunicação.

Serviço DNS na Internet



- Cliente deseja o endereço de www.amazon.com
  - Cliente acessa um servidor raiz para obter endereços IP dos servidores TLD .com
  - Cliente acessa um servidor .com para obter o endereço de um servidor com autoridade amazon.com
  - Cliente acessa servidor DNS da autoridade amazon.com para obter o endereço IP do host www.amazon.com

Nomeação baseada em atributo

- Nem sempre o nome da entidade é importante → descrição de características do recurso desejado pode ser mais importante

- Associação de atributos às entidades
  - Pares <atributo, valor>
  - Armazenam informação sobre a entidade
  - Serviços de diretório

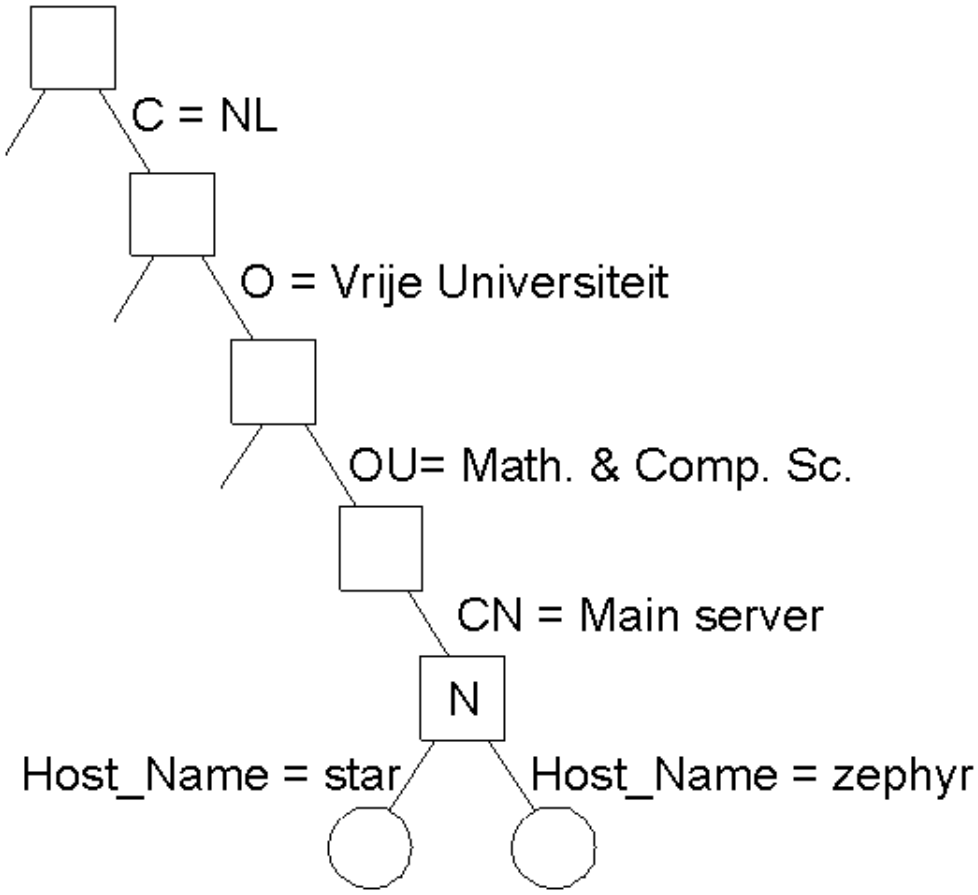
- Não é uma tarefa trivial
  - Qual o melhor conjunto de atributos?
  - Como fazer buscas por valores de atributos?

LDAP

- *Lightweight directory access protocol*
- Possui uma estrutura hierárquica

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

LDAP


C = NL
O = Vrije Universiteit
OU= Math. & Comp. Sc.
CN = Main server
N
Host_Name = star
Host_Name = zephyr

Seminários

- Tópicos e Grupos
  - Java RMI
  - RPC (em qualquer linguagem)
  - WebServices SOAP
  - WebServices RESTfull
  - Globus Toolkit (middleware para grid computing)
  - SDs Baseados em Objetos
  - Sistemas de Arquivos Distribuídos
  - SDs Baseados na Web
  - SDs Baseados em Coordenação

Prova!

- Data: 18/05