

## Aula 7

### Professores:

*Anna Dolejsi Santos (UFF)*

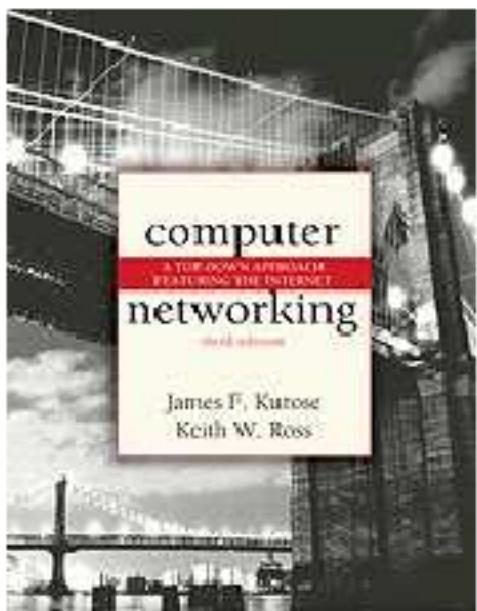
*Célio Vinicius Neves de Albuquerque (UFF)*

## DNS e Redes P2P

### Conteúdo:

- DNS: Domain Name System
- Redes Peer-to-Peer (P2P)

## Livro Texto



### **REDES DE COMPUTADORES E A INTERNET UMA ABORDAGEM TOP-DOWN**

([www.aw.com/kurose\\_br](http://www.aw.com/kurose_br))

**James F. Kurose e Keith W. Ross**

**Copyright: 2006 - 3a. Edição**

**ISBN: 8588639181**

<http://www.pearson.com.br/>

### **Referência Adicional:**

**Redes de Computadores**

**Andrew Tanenbaum**

**Editora Campus, 4a. Edição, 2003**

**ISBN: 8535211853**

Obs: As figuras que não têm referências pertencem ao material disponilizado pelo autor do livro texto ou foram produzidas pelo professor desta disciplina.

## Capítulo 2: Roteiro

- 2.1 Princípios dos protocolos da camada de aplicação
- 2.2 Web e HTTP
- 2.3 FTP
- 2.4 Correio Eletrônico
- SMTP, POP3, IMAP**
- 2.5 DNS**
- 2.6 Compartilhamento de arquivos P2P
- 2.7 Programação de *Sockets* com TCP
- 2.8 Programação de *Sockets* com UDP

## DNS: Domain Name System

Pessoas: muitos identificadores:

- CPF, nome, no. da Identidade

hospedeiros, roteadores Internet :

- endereço IP (32 bit) - usado p/ endereçar datagramas
- "nome", ex., jambo.ic.uff.br - usado por gente

P: como mapear entre nome e endereço IP?

Domain Name System:

- *base de dados distribuída* implementada na hierarquia de muitos *servidores de nomes*
- *protocolo de camada de aplicação* permite que hospedeiros, roteadores, servidores de nomes se comuniquem para *resolver* nomes (tradução endereço/nome)
- nota: função imprescindível da Internet implementada como protocolo de camada de aplicação
- complexidade na borda da rede

## DNS: Domain Name System (cont.)

### Serviços DNS

- Tradução de nome de hospedeiro para IP
- Apelidos para hospedeiros (aliasing)
  - Nomes canônicos e apelidos
- Apelidos para servidores de e-mail
- Distribuição de carga
  - Servidores Web replicados: conjunto de endereços IP para um nome canônico

### Serviços DNS

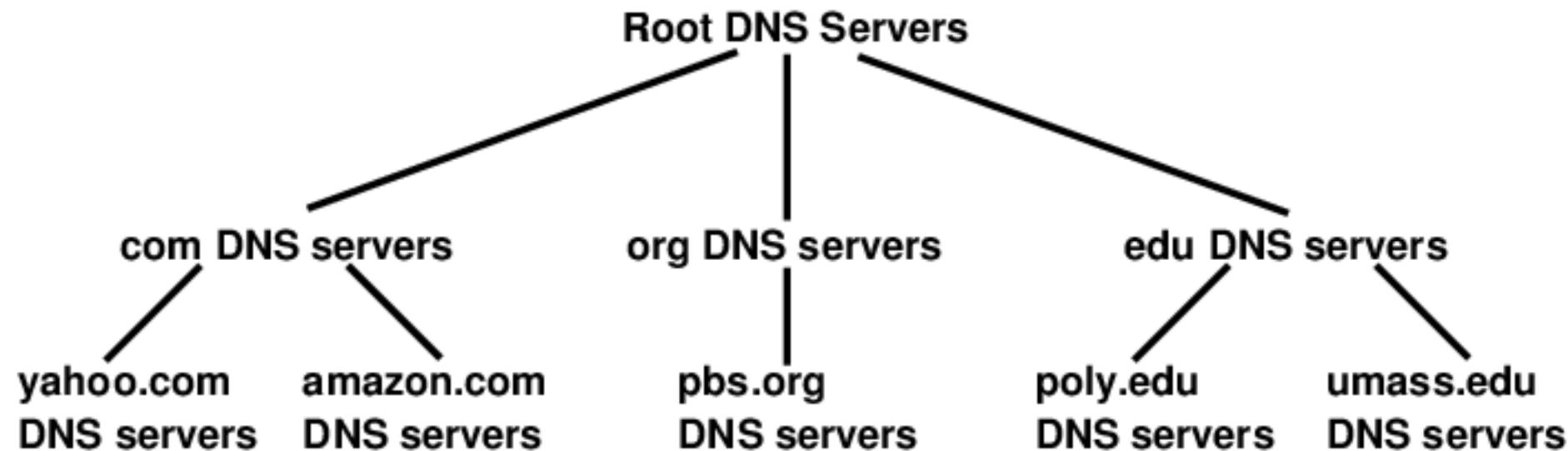
- Roda sobre UDP e usa a porta 53
  - RFCs 1034, 1035
  - Atualizado em outras RFCs

### Por que não centralizar o DNS?

- ponto único de falha
- volume de tráfego
- base de dados centralizada e distante
- manutenção (da BD)

Não é escalável!

## Base de Dados Hierárquica e Distribuída



Cliente quer IP para www.amazon.com; 1a aprox:

- Cliente consulta um servidor raiz para encontrar um servidor DNS .com
- Cliente consulta servidor DNS .com para obter o servidor DNS para o domínio amazon.com
- Cliente consulta servidor DNS do domínio amazon.com para obter endereço IP de www.amazon.com

## DNS: Servidores raiz

- procurado por servidor local que não consegue resolver o nome
- servidor raiz:
  - procura servidor oficial se mapeamento desconhecido
  - obtém tradução
  - devolve mapeamento ao servidor local



## Servidores TLD e Oficiais

- **Servidores *Top-level domain (TLD)*** : servidores DNS responsáveis por domínios com, org, net, edu, etc, e todos os domínios de países como br, uk, fr, ca, jp.
  - Network Solutions mantém servidores para domínio com
  - FAPESP (Registro .br) para domínio br
- **Servidores oficiais:** servidores DNS das organizações, provendo mapeamentos oficiais entre nomes de hospedeiros e endereços IP para os servidores da organização (e.x., Web e correio).
  - Podem ser mantidos pelas organizações ou pelo provedor de acesso

## Servidor de Nomes Local

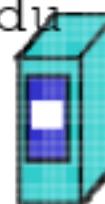
- Não pertence necessariamente à hierarquia
- Cada ISP (ISP residencial, companhia, universidade) possui um.
  - Também chamada do "servidor de nomes default"**
- Quando um hospedeiro faz uma consulta DNS, a mesma é enviada para o seu servidor DNS local
  - Atua como um intermediário, enviando consultas para a hierarquia.**

## Exemplo de DNS

- Hospedeiro em cis.poly.edu quer endereço IP para gaia.cs.umass.edu

*servidor local*

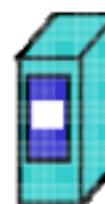
dns.poly.edu



*servidor raiz*



*servidor TLD*



*servidor oficial*

dns.cs.umass.edu



*solicitante*

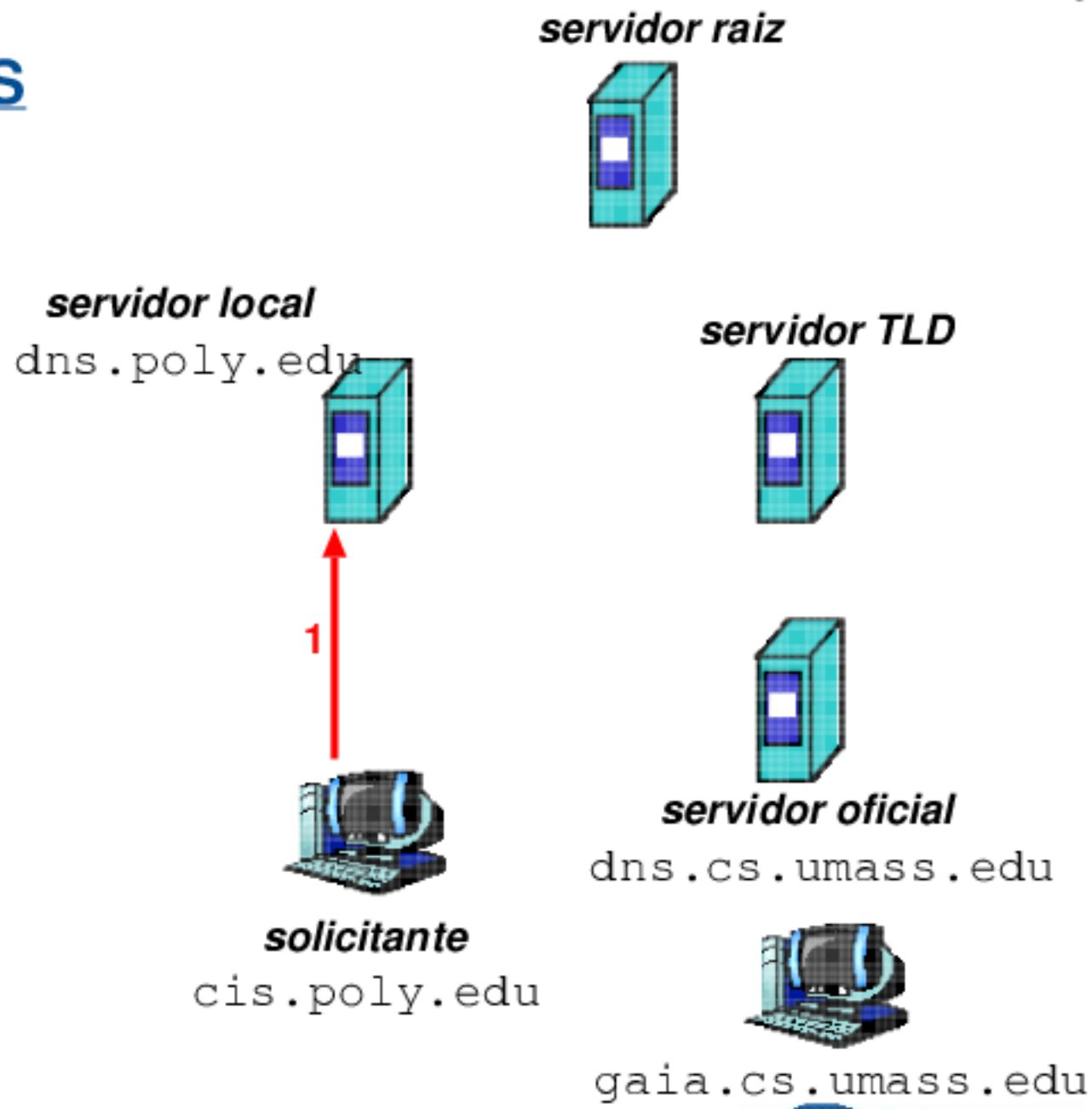
cis.poly.edu



gaia.cs.umass.edu

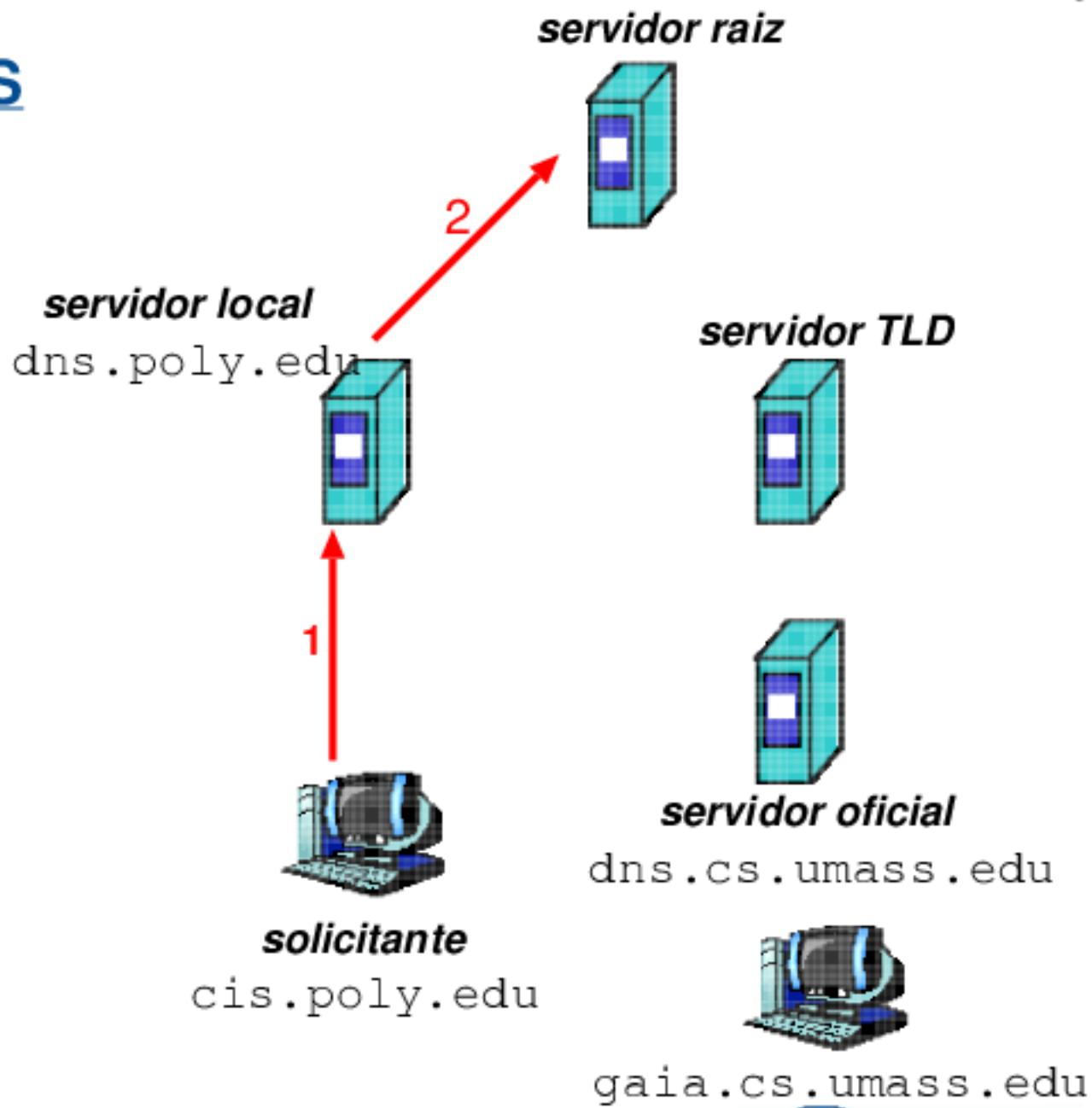
## Exemplo de DNS

- Hospedeiro em cis.poly.edu quer endereço IP para gaia.cs.umass.edu



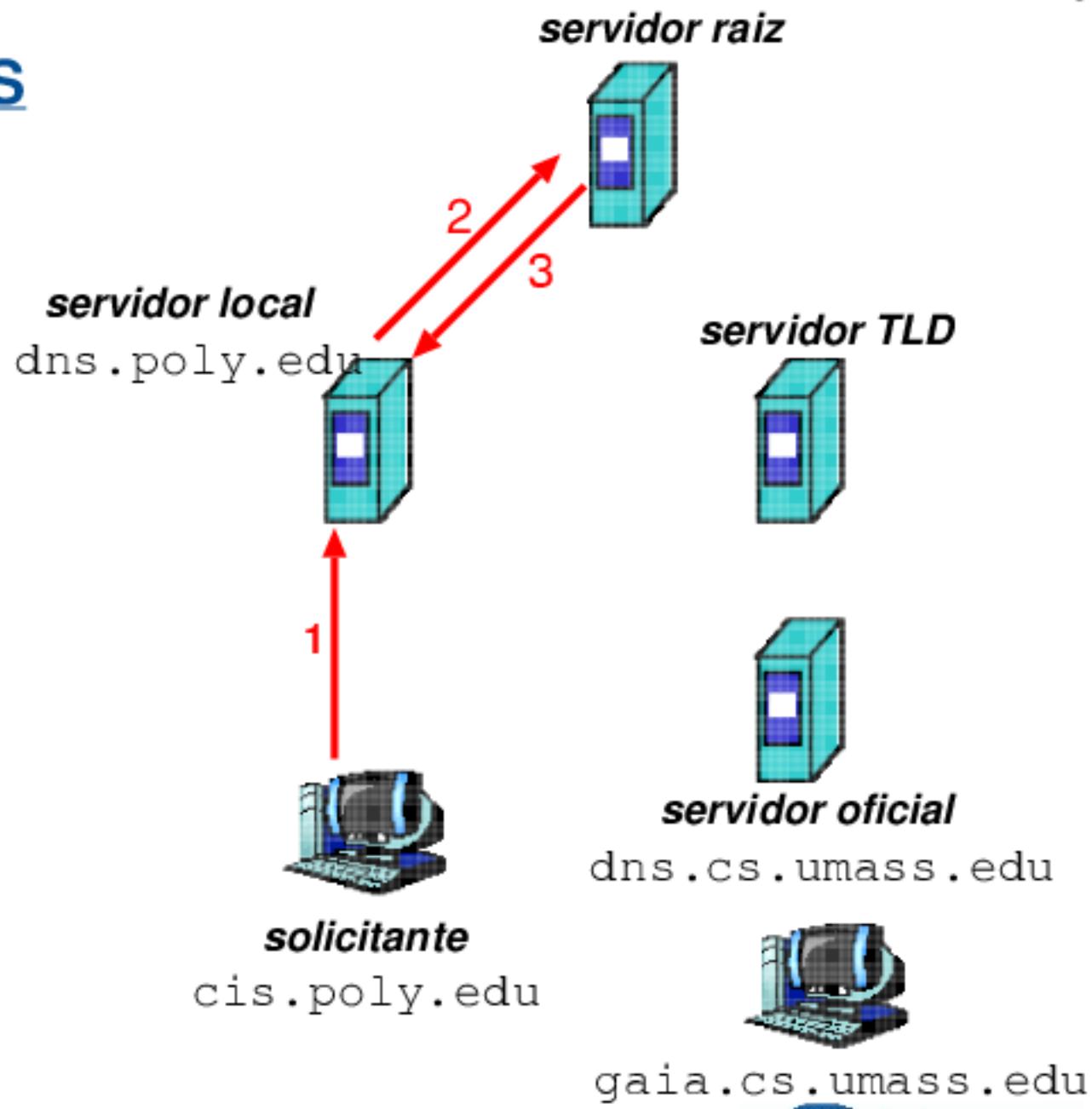
## Exemplo de DNS

- Hospedeiro em cis.poly.edu quer endereço IP para gaia.cs.umass.edu



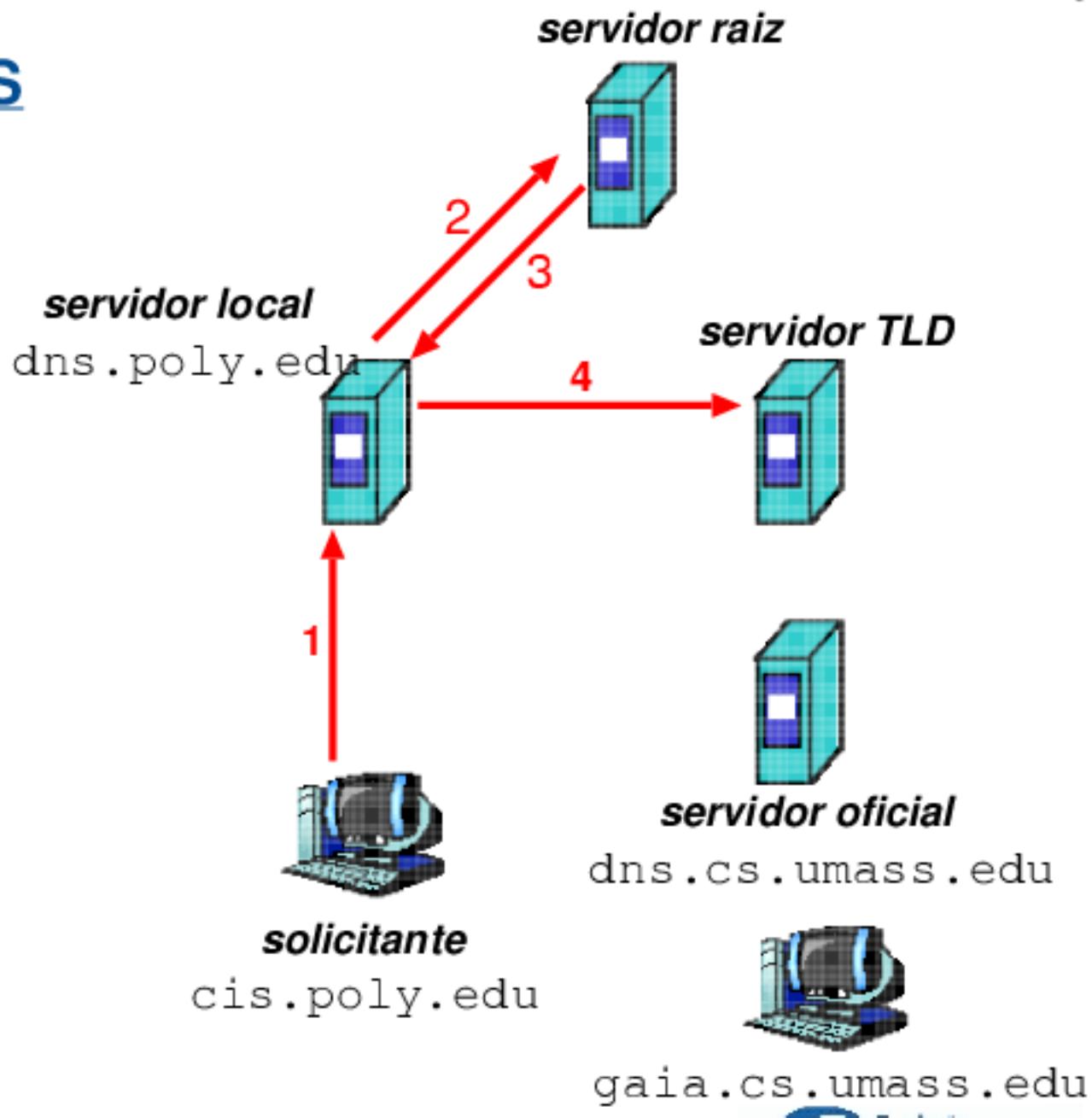
## Exemplo de DNS

- Hospedeiro em cis.poly.edu quer endereço IP para gaia.cs.umass.edu



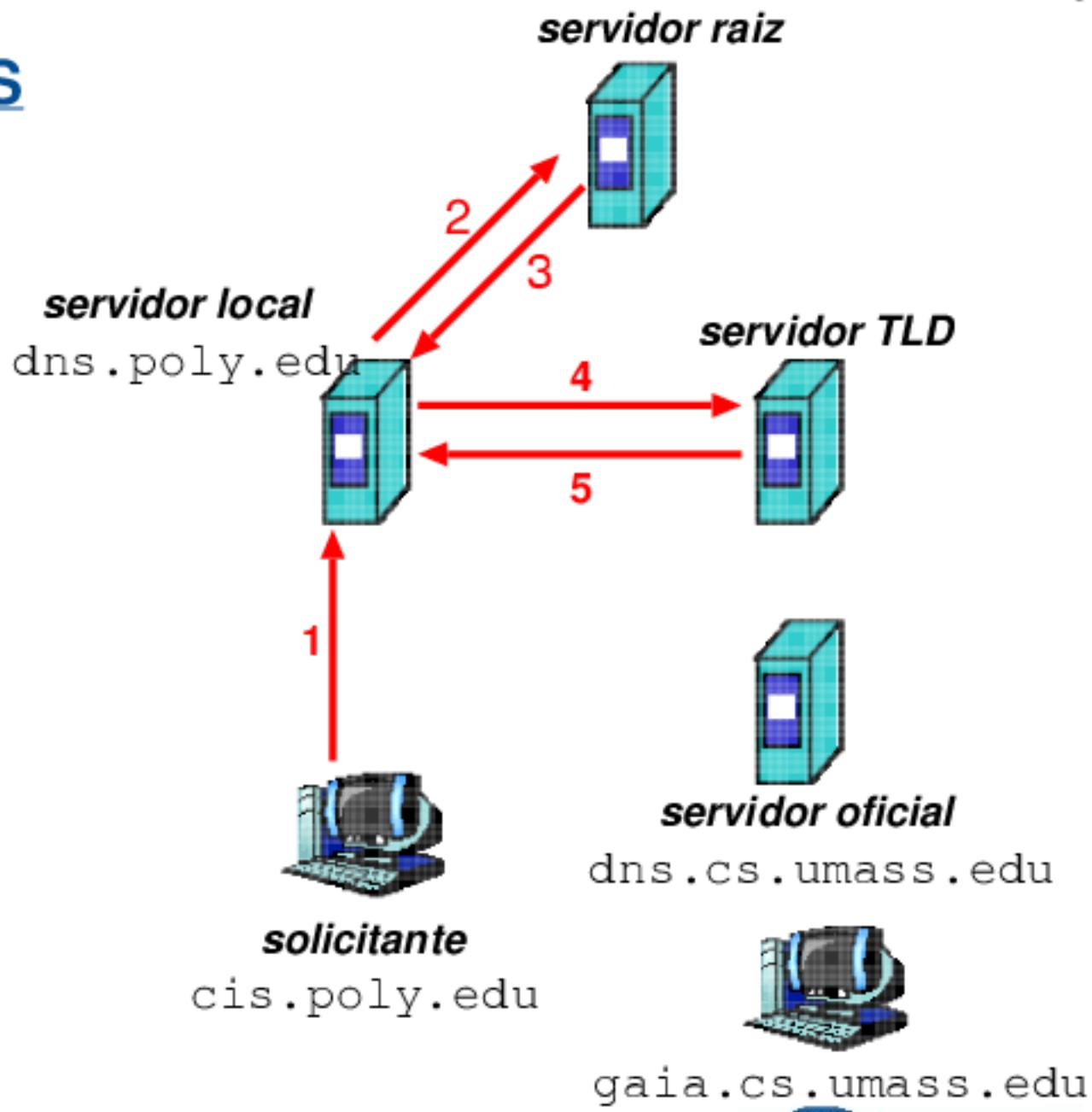
## Exemplo de DNS

- Hospedeiro em cis.poly.edu quer endereço IP para gaia.cs.umass.edu



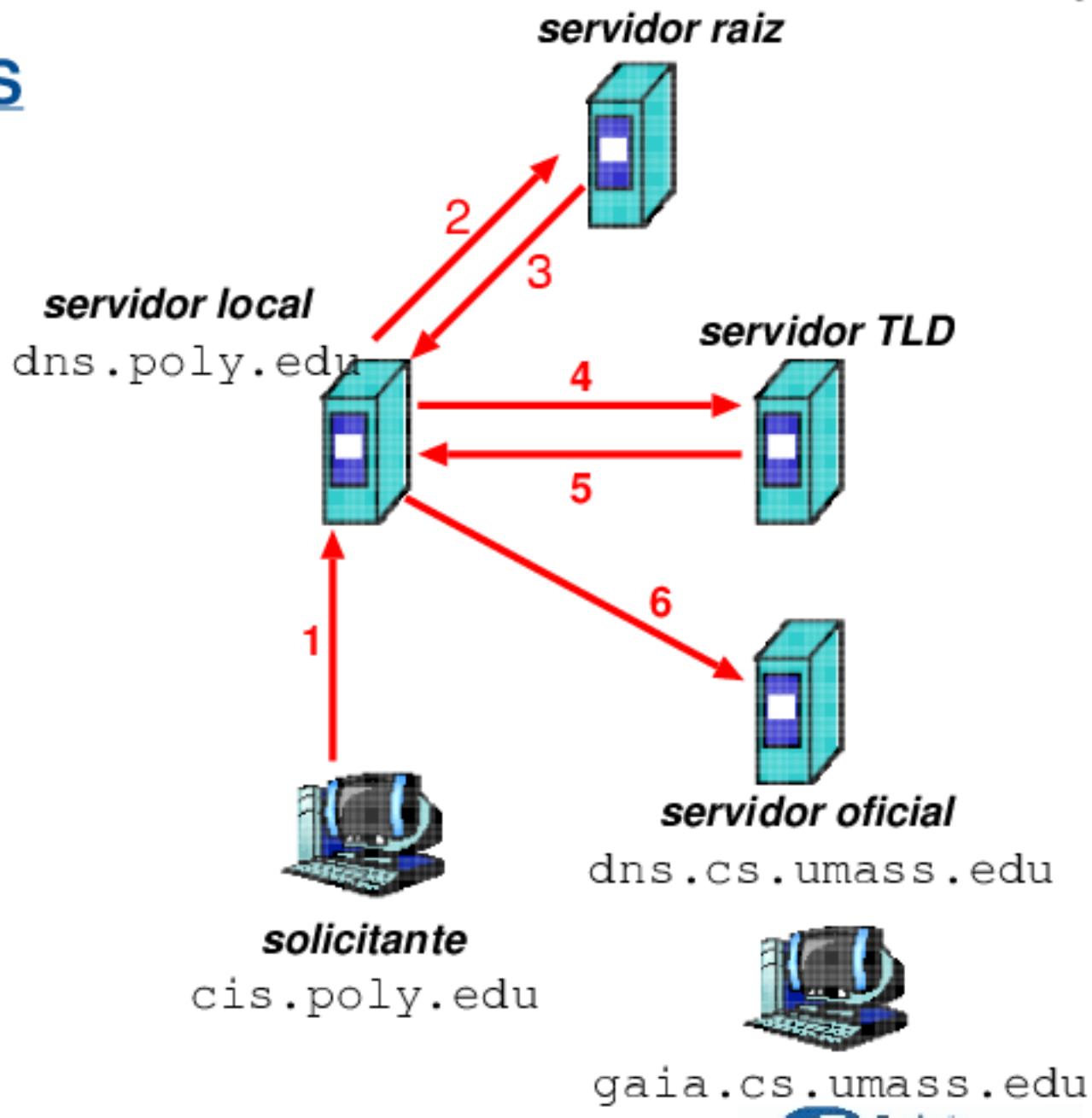
## Exemplo de DNS

- Hospedeiro em cis.poly.edu quer endereço IP para gaia.cs.umass.edu



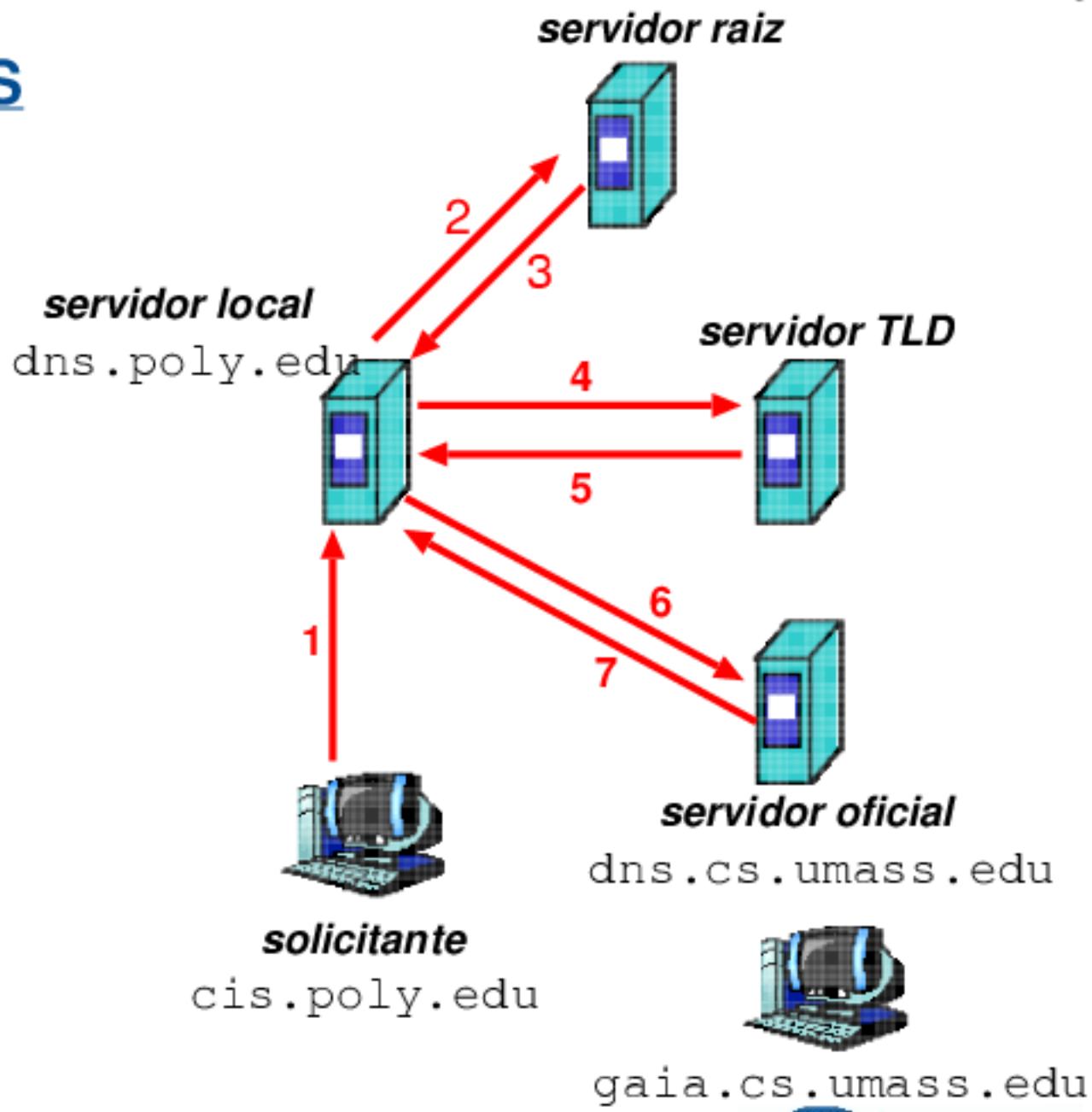
## Exemplo de DNS

- Hospedeiro em cis.poly.edu quer endereço IP para gaia.cs.umass.edu



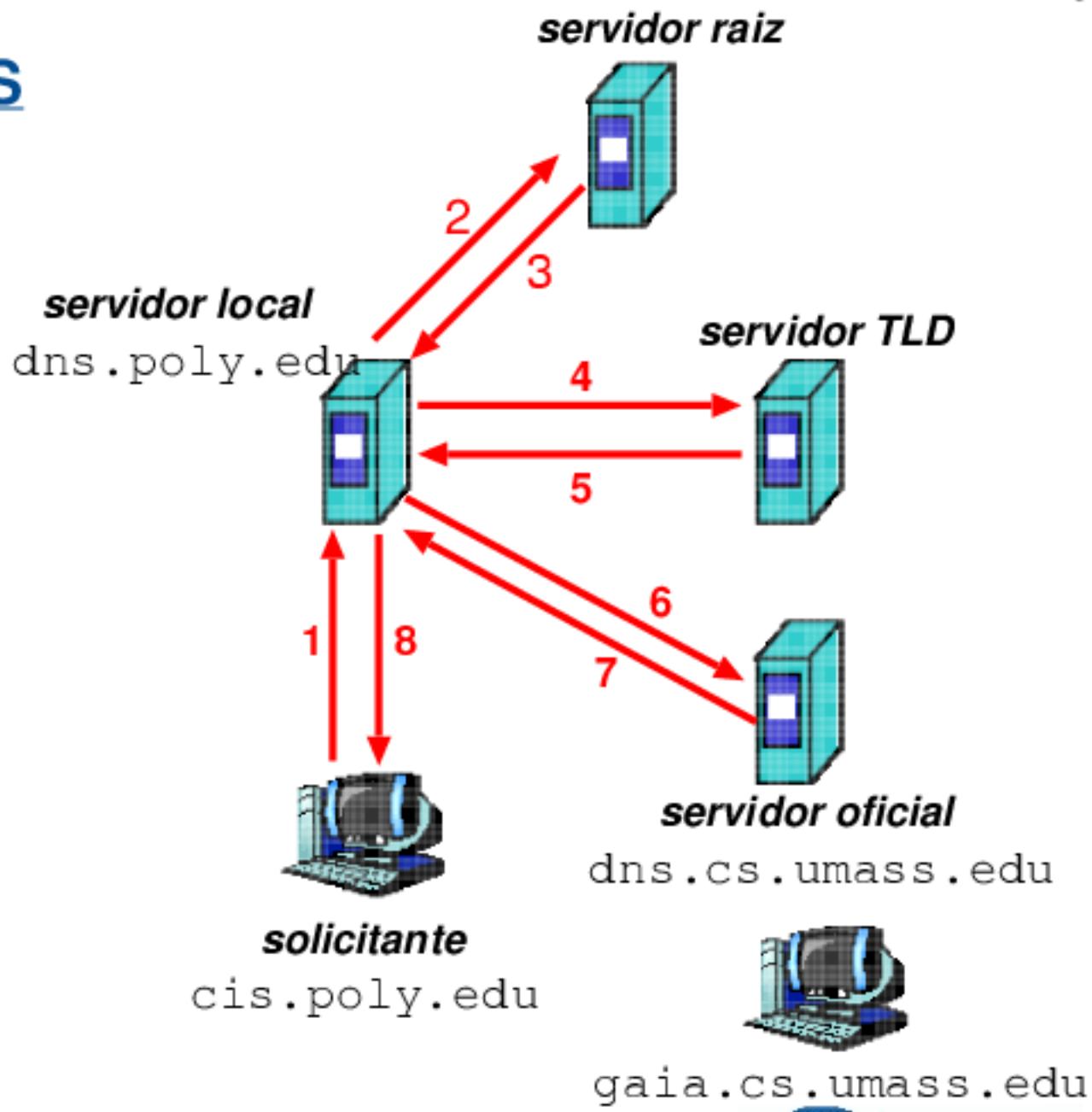
## Exemplo de DNS

- Hospedeiro em cis.poly.edu quer endereço IP para gaia.cs.umass.edu



## Exemplo de DNS

- Hospedeiro em cis.poly.edu quer endereço IP para gaia.cs.umass.edu



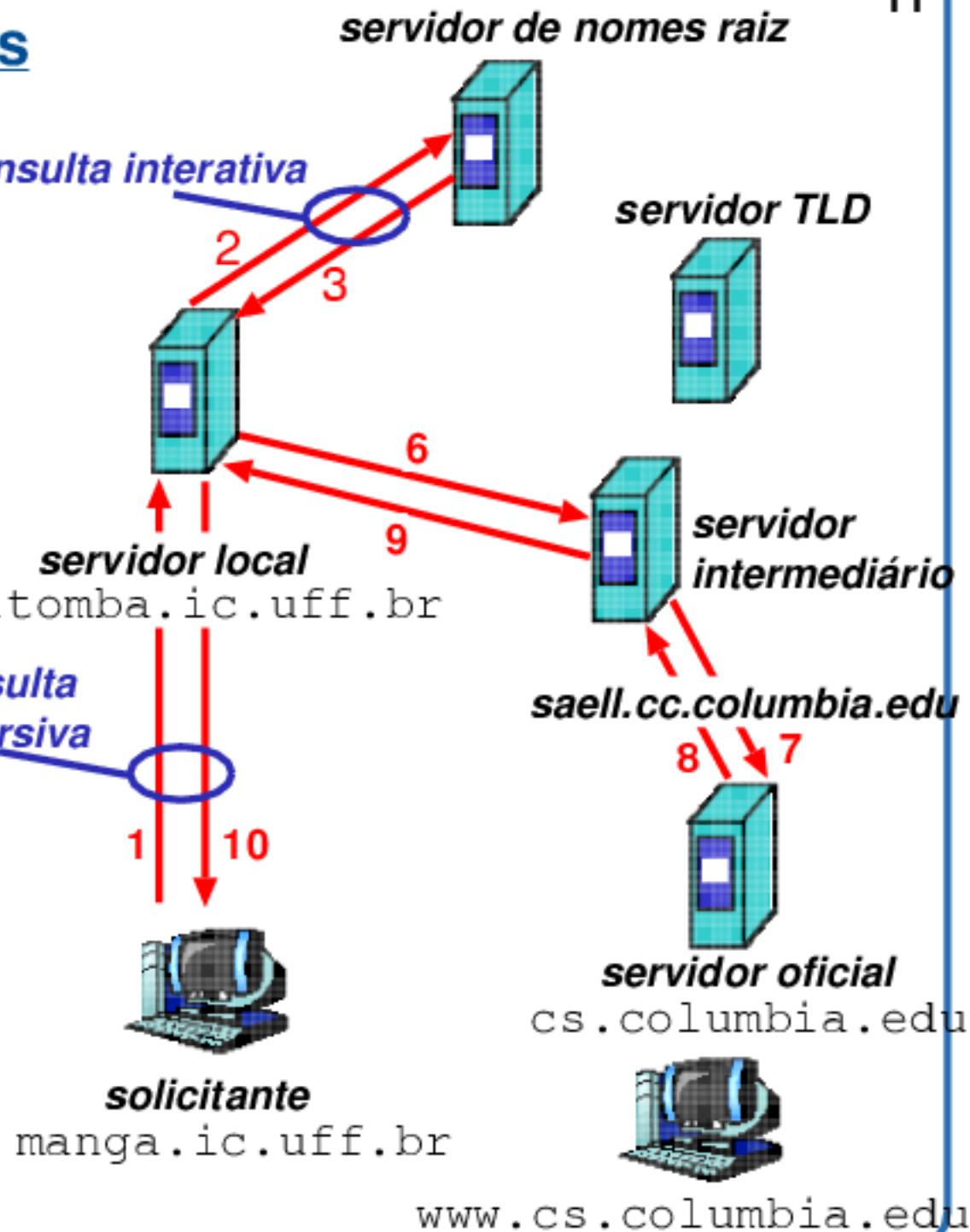
## DNS: tipos de consultas

### consulta recursiva:

- transfere a responsabilidade de resolução do nome para o servidor de nomes contatado
- carga pesada?

### consulta interativa:

- servidor consultado responde com o nome de um servidor de contato
- "Não conheço este nome, mas pergunte para esse servidor"



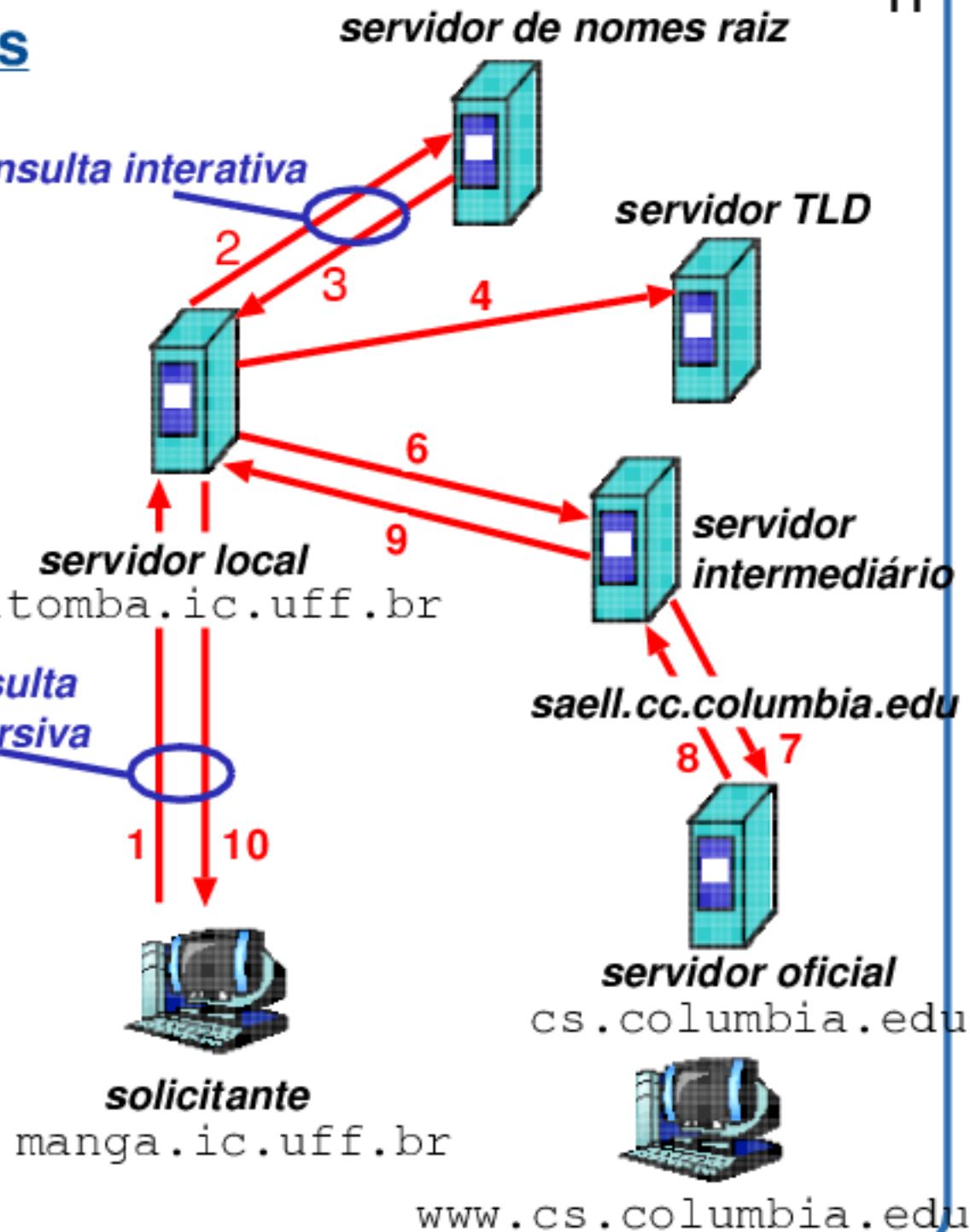
## DNS: tipos de consultas

### consulta recursiva:

- transfere a responsabilidade de resolução do nome para o servidor de nomes contatado
- carga pesada?

### consulta interativa:

- servidor consultado responde com o nome de um servidor de contato
- "Não conheço este nome, mas pergunte para esse servidor"



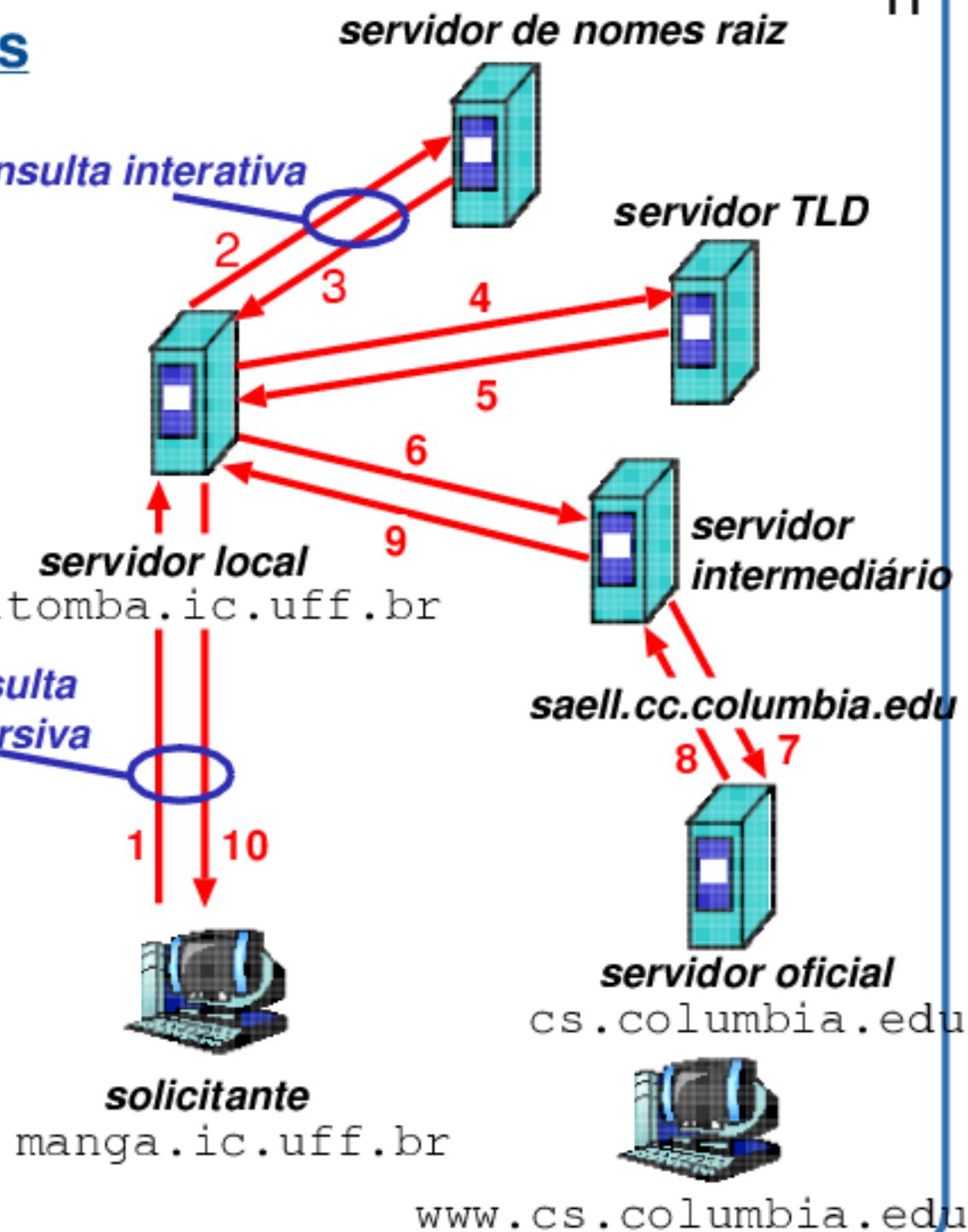
## DNS: tipos de consultas

### consulta recursiva:

- transfere a responsabilidade de resolução do nome para o servidor de nomes contatado
- carga pesada?

### consulta interativa:

- servidor consultado responde com o nome de um servidor de contato
- "Não conheço este nome, mas pergunte para esse servidor"



## DNS: uso de cache, atualização de dados

- uma vez que um servidor qualquer aprende um mapeamento, ele o coloca numa *cache* local
  - entradas na cache são sujeitas a temporização (desaparecem depois de um certo tempo)
  - Servidores TLD tipicamente armazenados no cache dos servidores de nomes locais
    - Servidores raiz acabam não sendo visitados com muita freqüência
- estão sendo projetados pela IETF mecanismos de atualização/notificação dos dados
  - RFC 2136
  - <http://www.ietf.org/html.charters/dnsind-charter.html>

## Registros DNS

DNS: BD distribuído contendo *registros de recursos (RR)*

**formato RR: (nome, valor, tipo, sobrevida)**

- Tipo=A
  - nome é nome de hospedeiro
  - valor é o seu endereço IP
- Tipo=NS
  - nome é domínio (p.ex. foo.com.br)
  - valor é endereço IP de servidor oficial de nomes para este domínio
- Tipo=CNAME
  - nome é nome alternativo (alias) para algum nome "canônico" (verdadeiro)
  - valor é o nome canônico
- Tipo=MX
  - nome é domínio
  - valor é nome do servidor de correio para este domínio

## DNS: protocolo e mensagens

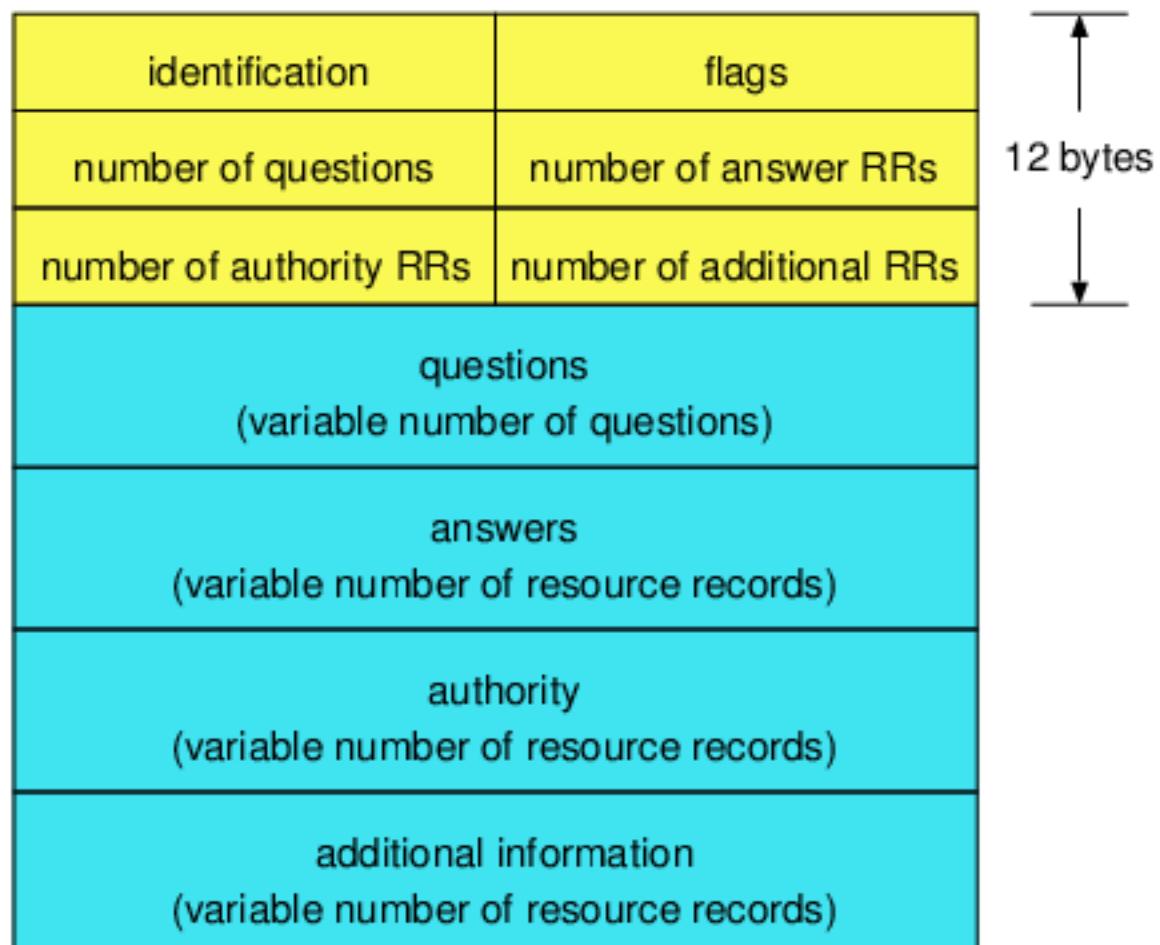
protocolo DNS: mensagens de *pedido* e *resposta*, ambas com o mesmo *formato de mensagem*

cabeçalho de msg

- identificação: ID de 16 bit para pedido, resposta ao pedido usa mesmo ID

flags:

- pedido ou resposta
- recursão desejada
- recursão permitida
- resposta é oficial



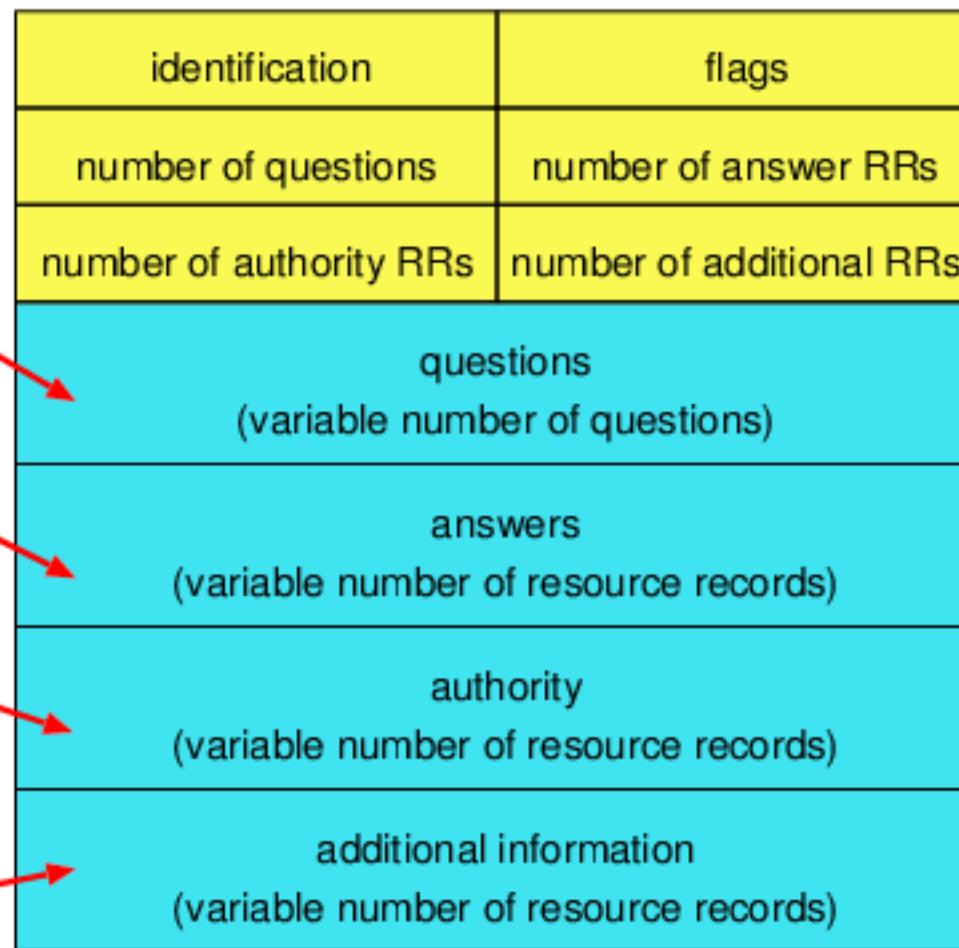
## DNS: protocolo e mensagens

campos de nome, e  
de tipo num pedido

RRs em resposta  
ao pedido

registros para outros  
servidores oficiais

info adicional  
"relevante" que  
pode ser usada



## Inserindo registros no DNS

- Exemplo: acabou de criar a empresa "Network Utopia"
- Registra o nome netutopia.com.br em uma **entidade registradora** (e.x., Registro .br)
  - Tem de prover para a registradora os nomes e endereços IP dos servidores DNS oficiais (primário e secundário)**
  - Registradora insere dois RRs no servidor TLD .br:**  
  
(netutopia.com.br, dns1.netutopia.com.br, NS)  
(dns1.netutopia.com.br, 212.212.212.1, A)
- Põe no servidor oficial um registro do tipo A para www.netutopia.com.br e um registro do tipo MX para netutopia.com.br
- Como as pessoas vão obter o endereço IP do seu site?**

## Capítulo 2: Roteiro

- 2.1 Princípios dos protocolos da camada de aplicação
- 2.2 Web e HTTP
- 2.3 FTP
- 2.4 Correio Eletrônico
- SMTP, POP3, IMAP**
- 2.5 DNS
- 2.6 Compartilhamento de arquivos P2P**
- 2.7 Programação de *Sockets* com TCP
- 2.8 Programação de *Sockets* com UDP

## Compartilhamento de arquivos P2P

### Exemplo:

- Alice executa aplicação cliente P2P no seu *notebook*
- Periodicamente ela se conecta à Internet e recebe um novo endereço IP a cada conexão
- Pede a música "Hey Jude"
- A aplicação apresenta uma lista de outros parceiros que possuem uma cópia de Hey Jude.

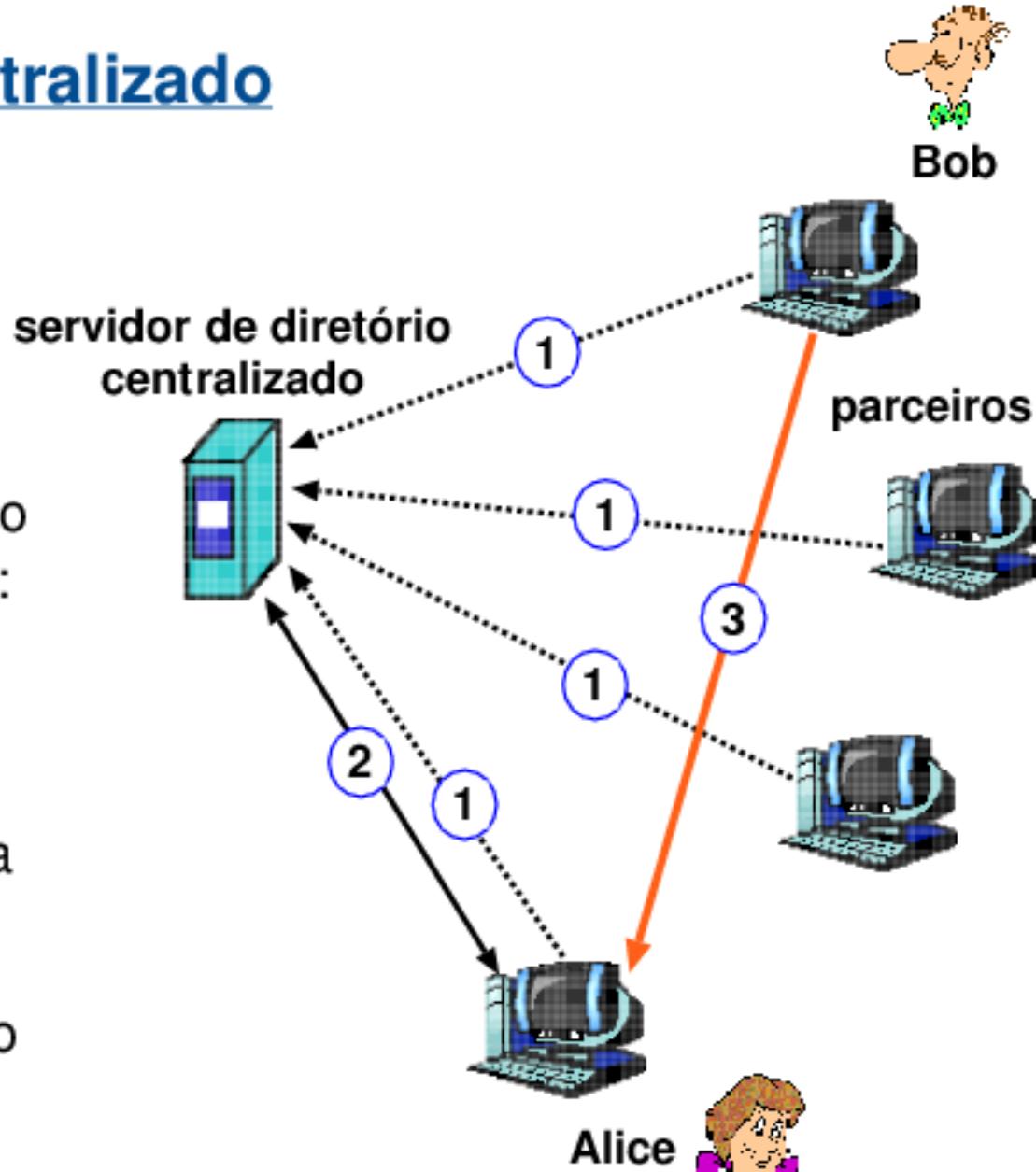
- Alice escolhe um dos parceiros, Bob.
- O arquivo é copiado do PC de Bob para o *notebook* de Alice: HTTP
- Enquanto Alice está baixando a música, outros usuários podem estar pegando arquivos do seu computador.
- O parceiro de Alice é tanto um cliente Web como um servidor Web temporário.

Todos os parceiros são servidores  
= altamente escalável!

## P2P: diretório centralizado

Projeto original do  
Napster

- 1) Quando um parceiro conecta ele informa ao servidor central o seu:
  - endereço IP
  - conteúdo
- 2) Alice consulta sobre a música "Hey Jude"
- 3) Alice solicita o arquivo a Bob



## P2P: problemas com diretório centralizado

- Ponto único de falha
- Gargalo de desempenho
- Violação de Direitos Autorais

a transferência de arquivo é descentralizada, mas a localização do conteúdo é altamente centralizada.

## Inundação de consultas: Gnutella

- Completamente distribuído
- Sem servidor central
- Protocolo de domínio público
- Vários clientes Gnutella implementam o protocolo

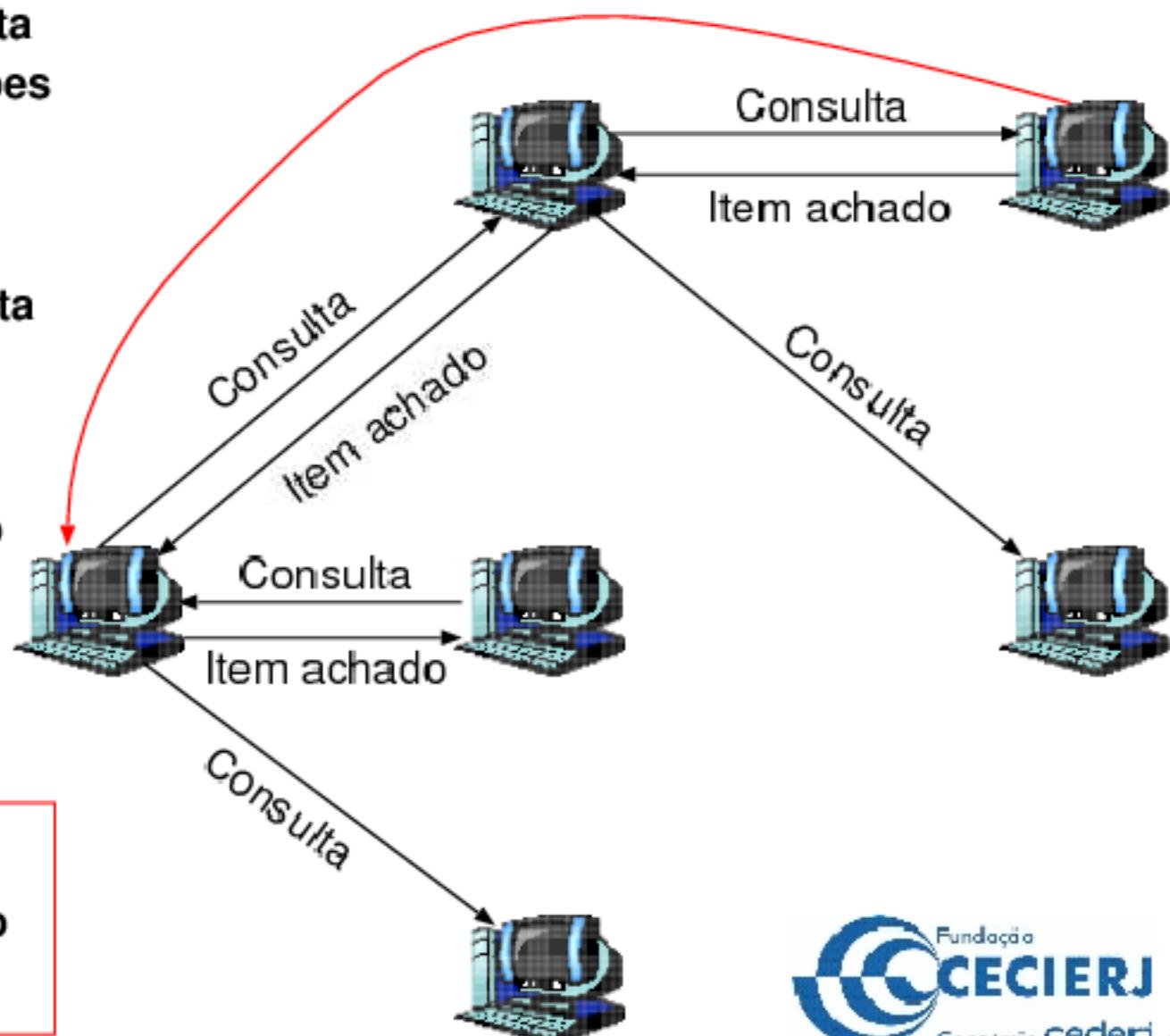
### Rede sobreposta: grafo

- Arco entre pares X e Y se existe uma conexão TCP
- Todos os pares ativos e arcos formam a rede sobreposta
- Arco não é um enlace físico
- Um par vai estar conectado tipicamente com  $< 10$  vizinhos na rede sobreposta

## Gnutella: protocolo

- Mensagem de consulta enviada pelas conexões TCP existentes
- Pares repassam mensagem de consulta
- Resposta sobre item encontrado enviada pelo caminho reverso

Transferência arq:  
HTTP



Escalabilidade:  
Inundação com escopo limitado

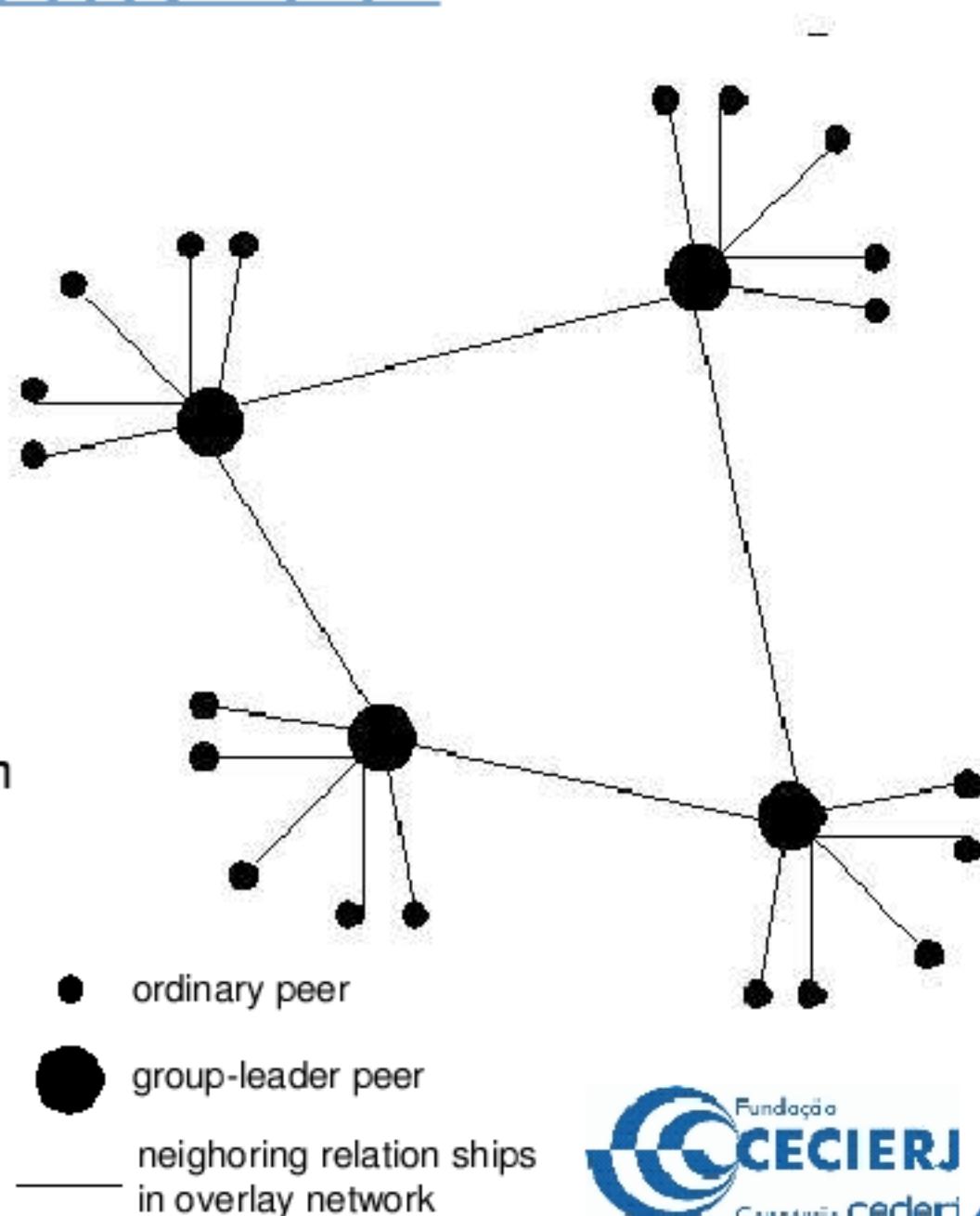
## Gnutella: junção do Par

1. Um par X se juntando deve encontrar algum outro par na rede Gnutella: usa lista de pares candidatos
2. X tenta criar conexões TCP com os pares na lista seqüencialmente até estabelecer conexão com Y
3. X envia mensagem Ping para Y; Y repassa a mensagem Ping
4. Todos os pares recebendo a mensagem Ping respondem com uma mensagem Pong
5. X recebe várias mensagens Pong. Ele pode então estabelecer conexões TCP adicionais

Saída do par pode gerar problemas.

## Explorando heterogeneidade: KaZaA

- ❑ Cada parceiro é um líder de grupo ou está alocado a um líder de grupo
  - Conexão TCP entre cada par e o seu líder de grupo
  - Conexões TCP entre alguns pares de líderes de grupos
- ❑ O líder de um grupo mantém registro sobre o conteúdo de todos os seus filhos



## KaZaA: Consulta

- ❑ Cada arquivo possui um *hash* e um descritor
- ❑ O cliente envia palavras-chave para o seu líder de grupo
- ❑ O líder de grupo responde com os itens encontrados
  - **Para cada item: metadados, hash, endereço IP**
- ❑ Se o líder de grupo repassa a consulta para outros líderes, eles respondem com os itens encontrados
- ❑ O cliente seleciona arquivos para *download*
  - **Requisições HTTP usando hash com identificador são enviadas para os pares que possuem os arquivos desejado**

## Truques do KaZaA

- Limitações na quantidade de *uploads* simultâneos
- Enfileiramento de requisições
- Prioridades para incentivar disponibilização de conteúdo
- Download em paralelo