Curso de Redes de Computadores

Vinicius F. Caridá

UEMG
FRUTAL

Capítulo 2: Camada de Aplicação

Metas do capítulo:

- O que?
  - Aprender aspectos conceituais e de implementação de protocolos de **aplicação** em redes.
  - Modelo cliente servidor.
  - Modelos de serviço.
- Como?
  - Estudo de **protocolos populares** da camada da aplicação.

Conhecer...

- Protocolos específicos:
  - *http*
  - *FTP*
  - *SMTP*
  - *POP3*
  - *DNS*
  - *P2P*
- Como é feita a programação de aplicações de rede.
  - Programação usando *sockets*.

UEMG FRUTAL

Conceitos

Clientes, servidores, processos, *sockets* e outros bichos…

UEMG FRUTAL

Princípios de aplicação de rede

- Exemplos de aplicações de rede: correio eletrônico, a Web, mensagem instantânea, login em computador remoto como Telnet e SSH, compartilhamento de arquivos P2P, transferência de arquivos, etc.

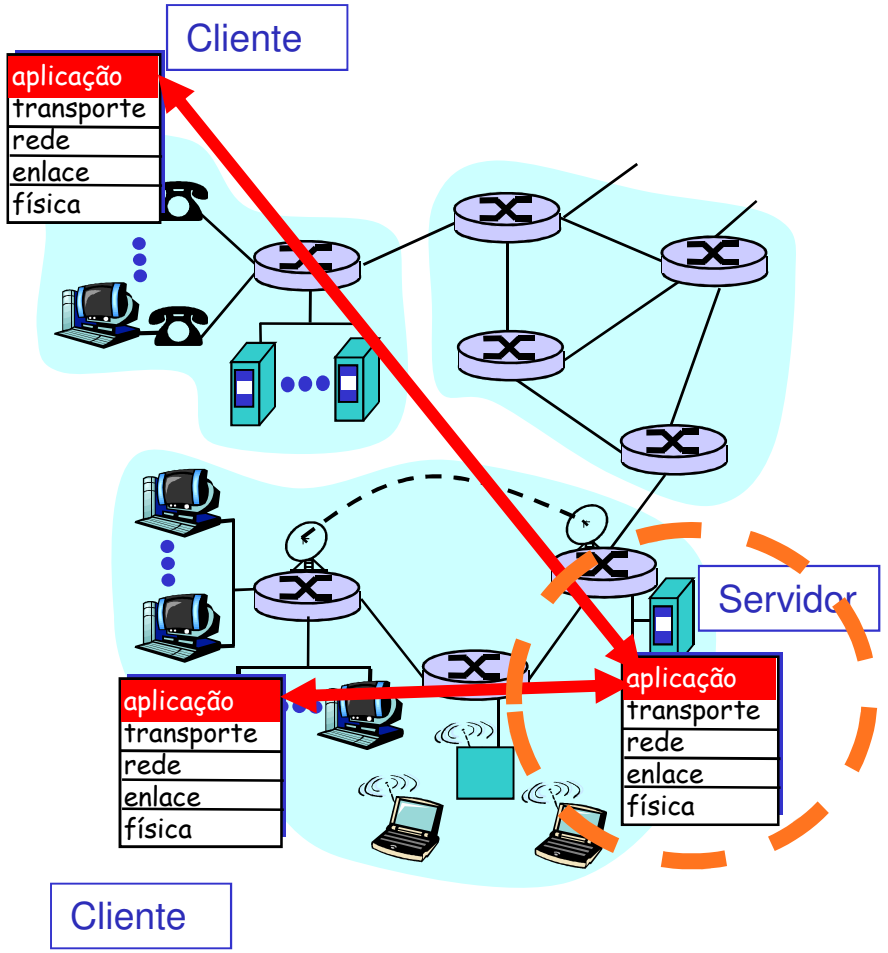
Princípios de aplicação de rede

- O cerne do desenvolvimento de aplicação de rede é escrever programas que rodem em sistemas finais diferentes e se comuniquem entre si pela rede. Por exemplo, a na Web há dois programas distintos que se comunicam, o programa do browser (roda na máquina do ususário) e o programa servidor Web (roda na máquina do servidor Web).

Aplicações e protocolos da camada de aplicação

Aplicação: processos distribuídos em comunicação

- Rodam em hosts.
  - Sistemas finais.
- **Trocam mensagens** para implementar aplicação.
- Exemplo:
  - Correio eletrônico, transferência de arquivos, WWW, login remoto, VoIP, etc...



Protocolos de aplicação

- **Protocolo** da camada de aplicação:
  - **Não é a aplicação.**
  - É APENAS **uma parte da aplicação**.
    - Define **mensagens trocadas** por aplicações, e **ações** as tomadas em sua resposta.
  - **Usam serviços** providos por protocolos de camadas inferiores.

Os processos em dois sistemas de extremidade diferentes comunicam-se logicamente entre si, trocando **mensagens** através da rede de computadores.

- Um processo de **emissão** cria e emite mensagens na rede;
- um processo de **recepção** recebe estas mensagens
  - e responde possivelmente emitindo mensagens de volta.

UEMG FRUTAL

Arquitetura

- A arquitetura da aplicação determina como a aplicação é organizada nos vários sistemas finais.

Arquitetura

- Em uma arquitetura cliente-servidor há um hospedeiro sempre em funcionamento, denominado servidor, que atende a requisições de muitos outros hospedeiros, denominados clientes, estes podem estar em funcionamento às vezes ou sempre. Os clientes não se comunicam diretamente uns com os outros. O servidor tem um endereço fixo, denominado endereço de IP, o cliente sempre pode contatá-lo, enviando um pacote ao endereço do servidor.

Arquitetura

- Em aplicações cliente-servidor, muitas vezes acontece de um único hospedeiro servidor ser incapaz de atender a todas as requisições de seus clientes, por essa razão, muitas vezes são utilizados conjuntos de hospedeiros (server farm) para criar servidor virtual poderoso em arquiteturas cliente-servidor.

Arquitetura

- Em uma arquitetura P2P pura, não há um servidor sempre funcionando no centro da aplicação, em vez disso pares arbitrários de hospedeiros comunicam-se diretamente entre si. Como os pares se comunicam sem passar por nenhum servidor especial, a arquitetura é denominada peer-to-peer, onde nela nenhuma das máquinas participantes precisa estar sempre em funcionamento. Um de suas características mais fortes é a escalabilidade, onde cada par adicional não apenas aumenta a demanda, mas também a capacidade de serviço.

Arquitetura

- Por outro lado, devido à sua natureza altamente distribuída e descentralizada, pode ser difícil de gerenciar aplicações P2P. Muitas aplicações são organizadas segundo arquiteturas hibridas cliente/servidor/P2P, a Napster era um exemplo disso, no sentido de que era P2P porque arquivos MP3 eram trocados diretamente entre pares, sem passar por servidores dedicados, sempre em funcionamento, mas também era cliente-servidor, já que um par consultava um servidor central para determinar quais pares que estavam em funcionamento tinham um arquivo MP3 desejado.

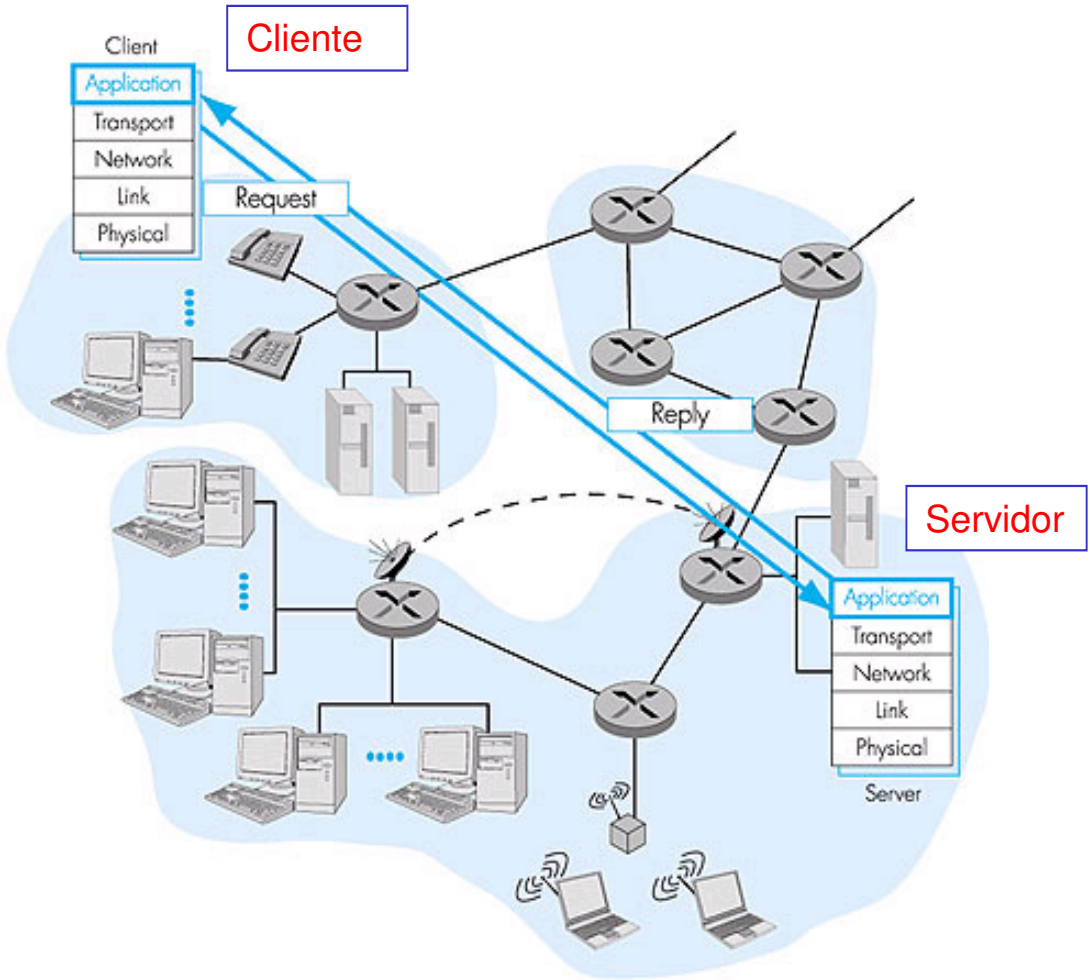
Como os processos trocam mensagens:

- Um protocolo da camada de aplicação **define como os processos** de aplicação **trocam mensagens**.
  - Funcionando em sistemas de extremidade diferentes,
- Um protocolo da **camada de aplicação** define:

- Os **tipos de mensagens trocadas.**
  - Por exemplo, mensagens do pedido e mensagens de resposta.
- A **sintaxe** dos vários tipos de mensagem.
  - Os **campos** na mensagem, e como os campos são delineados.
- A **semântica dos campos.**
  - Isto é, o **significado da informação** nos campos.
- As **regras.**
  - Determinar **quando** e como um processo emite e responde mensagens.

UEMG FRUTAL

Como os processos trocam mensagens



Aplicações de rede: **DEFINIÇÕES**

- Um **processo** é um programa que roda num *host*.
- Dois processos **no mesmo *host*** se comunicam usando **comunicação entre processos (*interprocess comunnication*),** definida pelo sistema operacional (SO).
- Dois **processos em *hosts* distintos** se comunicam usando um **protocolo da camada de transporte**.

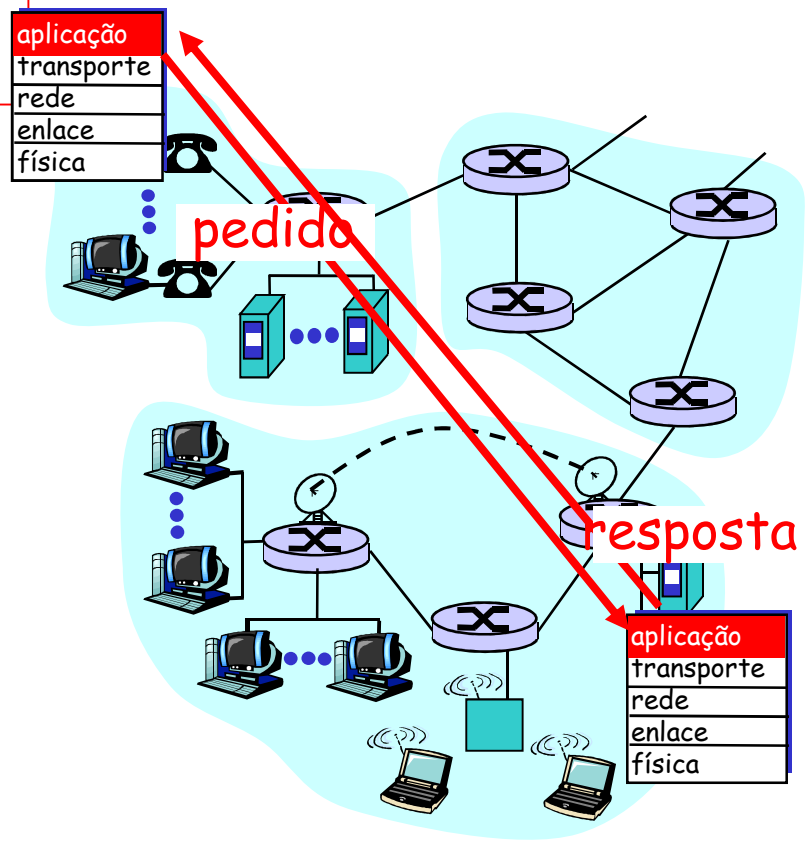
- Um agente de usuário (UA) é uma **interface entre o usuário e a aplicação de rede**.
  - WWW: *browser*.
  - Correio: leitor/compositor de mensagens.
  - *Streaming* A/V: *player* de mídia.

Processos clientes e processos servidores

- Para cada par de processos comunicantes normalmente rotulamos um dos dois processos de cliente(que inicia a comunicação) e o outro de servidor(processo que é contatado para iniciar a sessão).
- Na Web, um processo browser inicia o contato com um processo do servidor Web(cliente) e o processo do servidor Web é o servidor. No compartilhamento de arquivos P2P, quando o ParA solicita ao ParB o envio de um arquivo especifico, o ParA é o cliente enquanto o ParB é o servidor.

Paradigma cliente-servidor

Aplicação de rede típica tem duas partes: ***cliente*** e ***servidor***



Cliente:

- Inicia contato com o servidor
- Define-se como aquele que "chama".
- Solicita serviço do servidor.

Servidor:

- Provê o serviço requisitado ao cliente.

Protocolos da camada de aplicação

API - Aplication Program Interface:

- Interface de programação de aplicações.

- Define a **interface** entre a **aplicação** e **camada de transporte**.
- *APIs* → definidas pelos RFCs.
- *Socket* (= tomada).
  - Dois processos se comunicam enviando dados para um ***socket*** , ou lendo dados de um ***socket.***

... voltaremos mais tarde a este assunto.

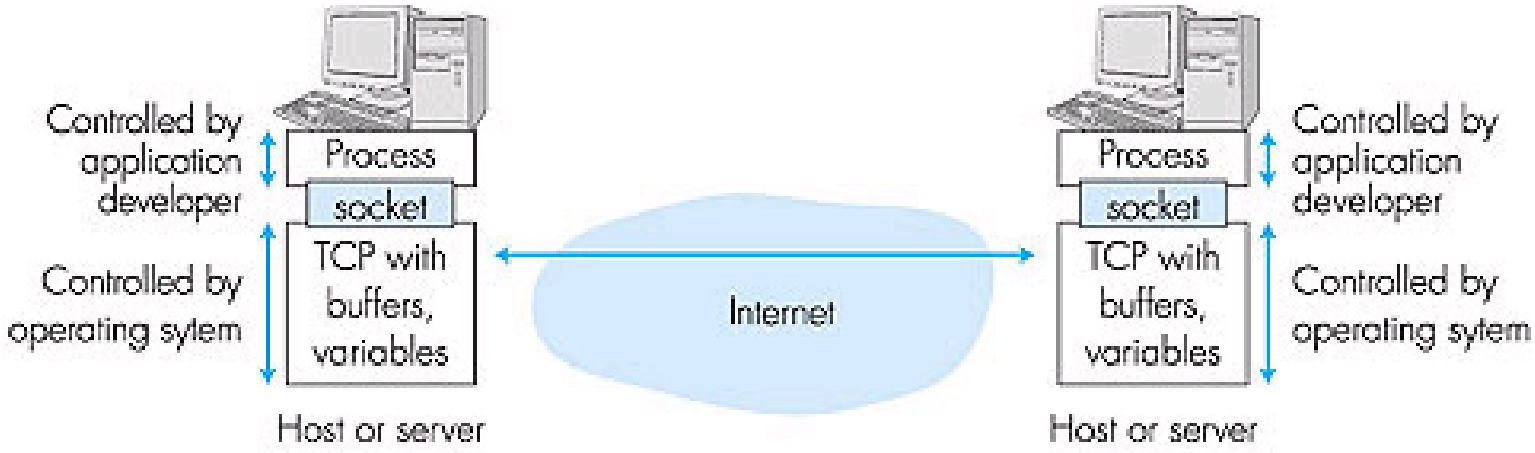
Sockets:

- Um processo envia mensagens para a rede e recebe mensagens dela através de seu **socket**. Um analogia para se entender: Um processo é análogo a uma casa e seu socket à porta da casa, quando um processo quer enviar uma mensagem a um outro processo em outro hospedeiro, ele a empurra porta(socket) afora para dentro da rede, ao chegar ao hospedeiro destinatário, a mensagem passa através da porta(socket) do processo receptor. Um socket é a interface entre a camada de aplicação e a de transporte dentro de uma máquina.

Comunicação pelos *sockets* (1)

- O *socket* pode ser entendido como a **porta** do processo:
  - Um processo **emite e recebe mensagens** da rede, **através de seus *sockets*.**
  - Quando um processo quer **emitir uma mensagem a um outro processo** em um outro host, **empurra a mensagem para o *socket*.**
  - O processo supõe que há um **infra-estrutura de transporte** no outro lado, a qual transportará a mensagem até a porta do processo do destino.

Comunicação pelos *sockets* (2)



– Como um processo identifica outro?

Endereçamento de processos:

- Para que um processo em um hospedeiro envie uma mensagem a um processo em outro, o processo de origem tem de identificar o processo destinatário. Para isso ele precisa de duas informações, o nome ou o endereço da máquina hospedeira e um identificador que especifique o processo destinatário.

Como um processo identifica outro ?

Pergunta: Como um processo pode "identificar"o outro processo com o qual quer se comunicar?

- **Endereço IP** do *host* do outro processo.
  - Por enquanto: o ***IP ADDRESS* é um valor de 32-bits que identifica unicamente o sistema de extremidade**. Mais precisamente: identifica unicamente a **interface** (placa) que conecta esse host à Internet. Ex. `200.145.9.9`
- "**Número de porta**" : permite que o hospedeiro receptor determine para qual processo deve ser entregue a mensagem. Exemplo: Porta 80/TCP.

  Ex. `200.145.9.9:80` (RFC 1700 - Portas *well-known*)

Endereçamento de processos:

- O processo destinatário é identificado por seu **endereço de IP** que é uma quantidade de 32 bits que identifica exclusivamente o sistema final. Além de saber o endereço, o processo de origem tem de identificar o processo que está rodando no outro hospedeiro, um **número de porta** de destino atende a essa finalidade.

- Um protocolo de camada de aplicação define como processos de uma aplicação, que funcionam em sistemas finais diferentes, passam mensagens entre si, em particular ele define os tipos de mensagens trocadas, a sintaxe dos vários tipos de mensagem, a semântica dos campos, regras para determinar quando e como um processo envia e responde mensagens. Muitos protocolos de camada de aplicação são proprietários e não estão disponíveis ao publico. É importante distinguir aplicações de rede de protocolos de camada de aplicação, um protocolo de camada de aplicação é apenas um pedaço de aplicação de rede.

De que serviços uma aplicação necessita?

De que serviços de transporte uma aplicação precisa? (1)

- Quando se desenvolve uma aplicação, deve-se **escolher um dos protocolos disponíveis do transporte**.
- Como se faz esta escolha?
  - Estuda-se os serviços fornecidos pelos protocolos disponíveis do transporte, e escolhe-se o protocolo com os serviços que melhor se adaptem às necessidades de sua aplicação.
  - "Com Conexão" ou "Sem Conexão".

De que serviços de transporte uma aplicação precisa? (2)

Perda de dados:

- Algumas aplicações (por exemplo áudio) podem tolerar algumas perdas.
- Outras (por exemplo, transferência de arquivos, telnet) exigem transferência 100% confiável.

Temporização:

- Algumas aplicações (por exemplo, telefonia em Internet, jogos interativos) requerem baixo retardo para serem "viáveis".

Largura de banda:

• Algumas aplicações (por exemplo , multimídia) requerem quantia mínima de banda para serem "viáveis".
• Outras aplicações ("elásticas") conseguem usar qualquer quantia de banda disponível.

UEMG FRUTAL

Transferência confiável de dados

- Algumas aplicações exigem transferência de dados totalmente confiável, isto é, não pode haver perda de dados (que pode ter consequências devastadoras). Outras aplicações podem tolerar uma certa perda de dados, mais notavelmente aplicações de multimídia.

Largura de banda

- Algumas aplicações têm de transmitir dados a uma certa velocidade para serem efetivas. Se essa largura de banda não estiver disponível, a aplicação precisará codificar a uma taxa diferente ou então desistir, já que receber metade da largura de banda que precisa de nada adianta para tal aplicação sensível à largura de banda. Embora aplicações sensíveis à largura de banda exijam uma dada quantidade de largura de banda, aplicações elásticas(correio eletrônico, transferência de arquivos, etc) podem fazer uso de qualquer quantidade mínima ou máxima que por acaso esteja disponível.

Temporização

- O requisito final de serviço é a temporização. Aplicações interativas em tempo real, exigem limitações estritas de temporização na entrega de dados para serem efetivas.

Serviços providos por protocolos de transporte Internet

Serviço TCP:

- Orientado a conexão: estabelecimento exigido entre cliente e servidor.
- Transporte confiável entre processos emissor e receptor.
- Controle de fluxo: emissor não vai "afogar" receptor.
- Controle de congestionamento: estrangular emissor quando a rede carregada.
- Não oferece: garantias temporais ou de banda mínima (*QoS*).

Serviço UDP:

- Transferência de dados **não confiável** entre processos remetente e receptor.
- **Não provê**: estabelecimento da conexão, confiabilidade, controle de fluxo, controle de congestionamento, garantias temporais ou de banda mínima.

Exercício: Descreva onde e quando é interessante usar UDP, e quando não é.

UEMG FRUTAL

Exercício: Veja quais as principais aplicações Internet e que tipo de protocolos de transporte elas utilizam, e em quais portas operam.

Aplicação	**Protocolo da camada de apl**	**Protocolo de transporte usado**
Correio eletrônico	smtp [RFC 821]	?
Acesso terminal remoto	telnet [RFC 854]	?
WWW	http [RFC 2068]	?
Transf. de arquivos	ftp [RFC 959]	?
streaming multimídia	proprietário (Ex: *RealNetworks*)	?
Servidor de arquivo	NFS	?
VoIP	proprietário (Ex: Skype)	?

UEMG FRUTAL

Vinicius Fernandes Caridá

vfcarida@gmail.com