

**INSTITUTO FEDERAL**  
Norte de Minas Gerais  
Campus Januária

# Redes de Computadores

## *- Camada de Enlace -*

# Camada de Enlace

ORIGEM



DESTINO



PROTOCOLOS

MEIO

## CAMADA DE ENLACE

SINAL



# Arquitetura TCP/IP

## Modelo OSI



## TCP / IP



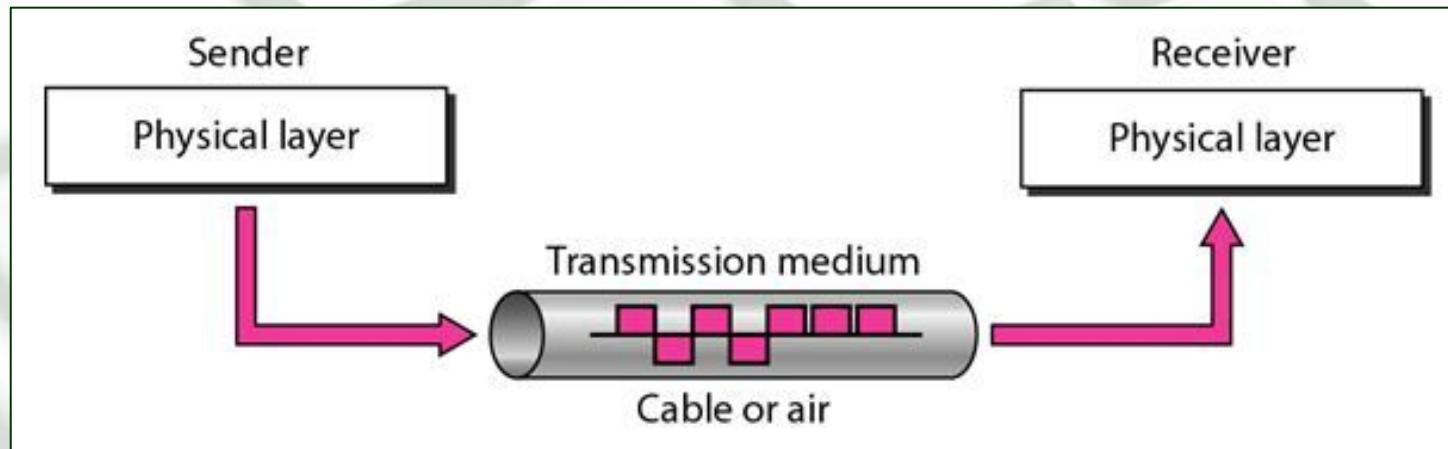
# Meio de Transmissão

- Em Redes, chamamos de **Enlace** ou **Link**, o **meio físico** pelo qual os sinais podem ser propagados.

**Sinal Elétrico?**  
→ *Fios de Cobre*

**Sinal Luminoso?**  
→ *Fibra Óptica*

**Ondas de Rádio?**  
→ *Meio Sem Fio*



# Tipos de Enlaces

## ■ Link Ponto a Ponto

- O enlace (meio de transmissão) é **dedicado** para a comunicação entre dois dispositivos.

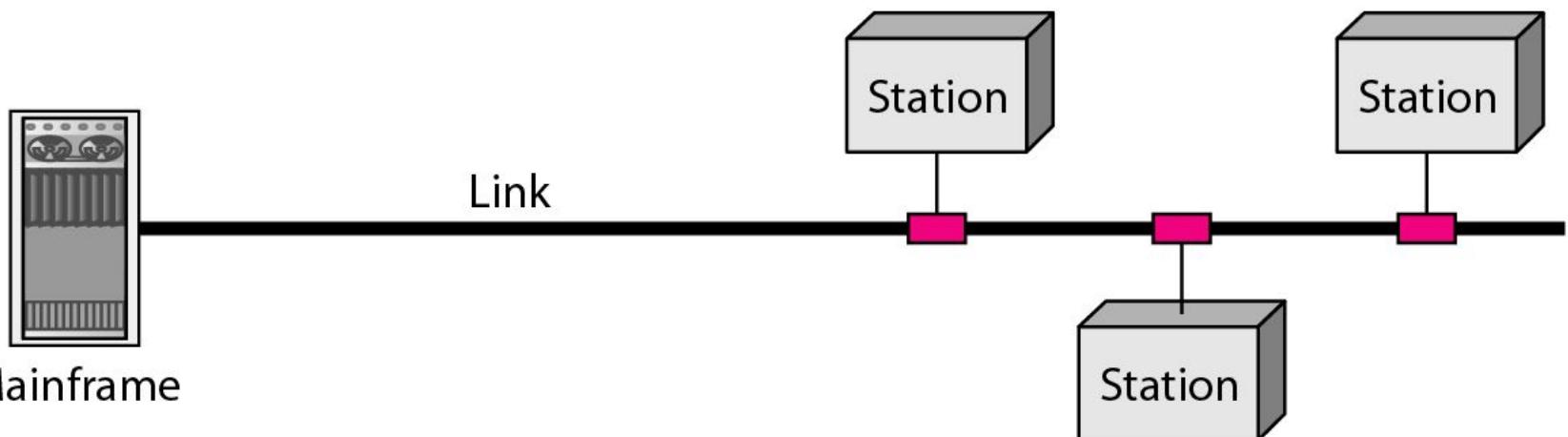
## ■ Link Multiponto

- O enlace é compartilhado entre três ou mais dispositivos.
- Compartilhamento pode ser **espacial, temporal ou probabilístico**.

# Tipos de Enlaces



a. Point-to-point



b. Multipoint

# Camada Enlace

- A principal função da camada de **Enlace de Dados** é transformar um canal bruto de comunicação em uma linha **que pareça livre de erros para as camadas superiores.**
- Em meios compartilhados (multiponto), outra função importante da camada é controlar o **acesso ao meio de transmissão**, minimizando problemas de colisão de dados.

# Camada 2 - Enlace

O objetivo básico é prover a troca de dados (sem erros) entre os nós diretamente conectados pelo meio físico

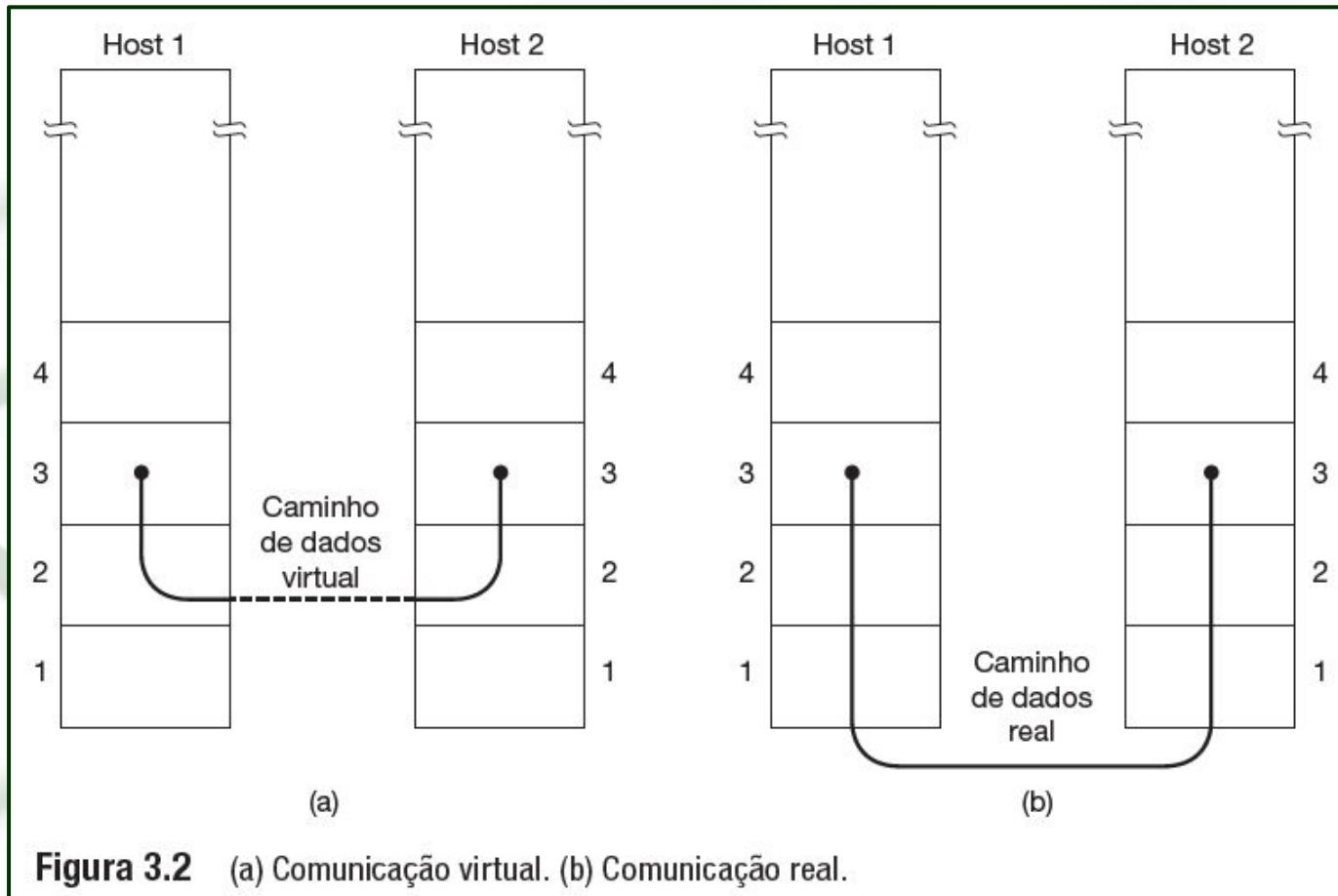
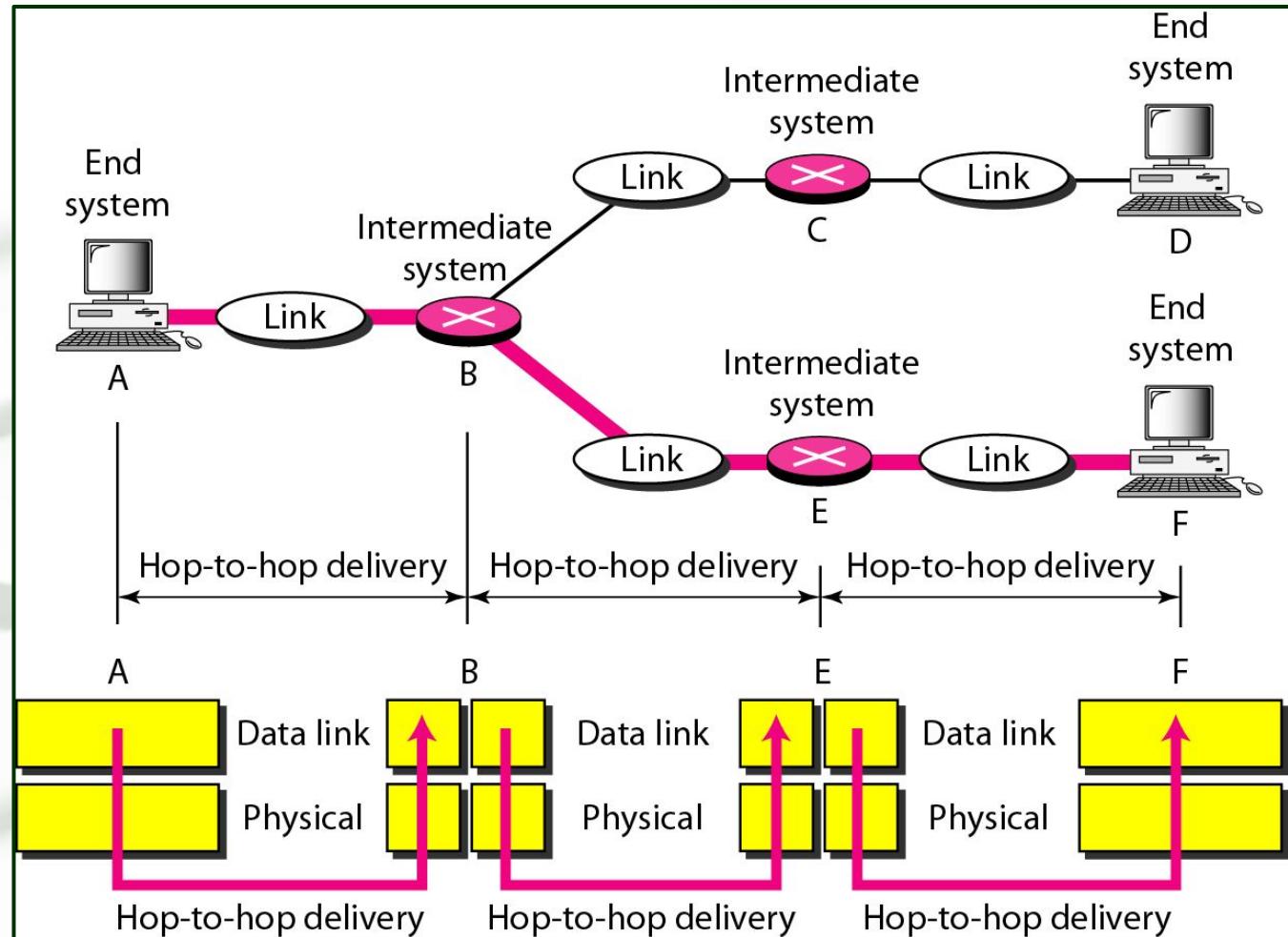


Figura 3.2 (a) Comunicação virtual. (b) Comunicação real.

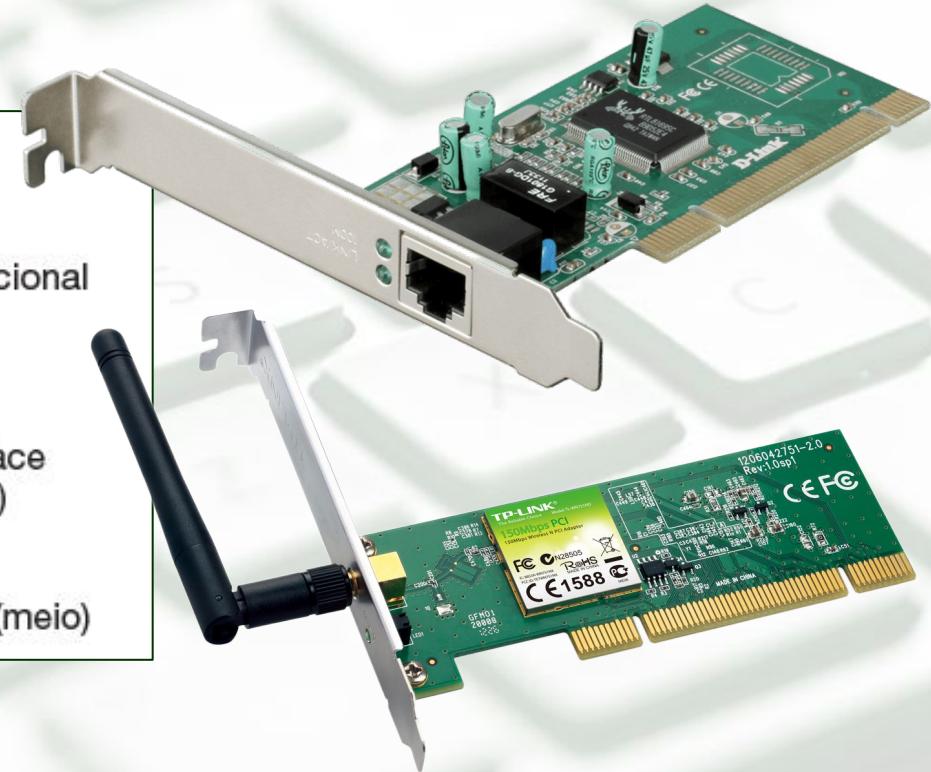
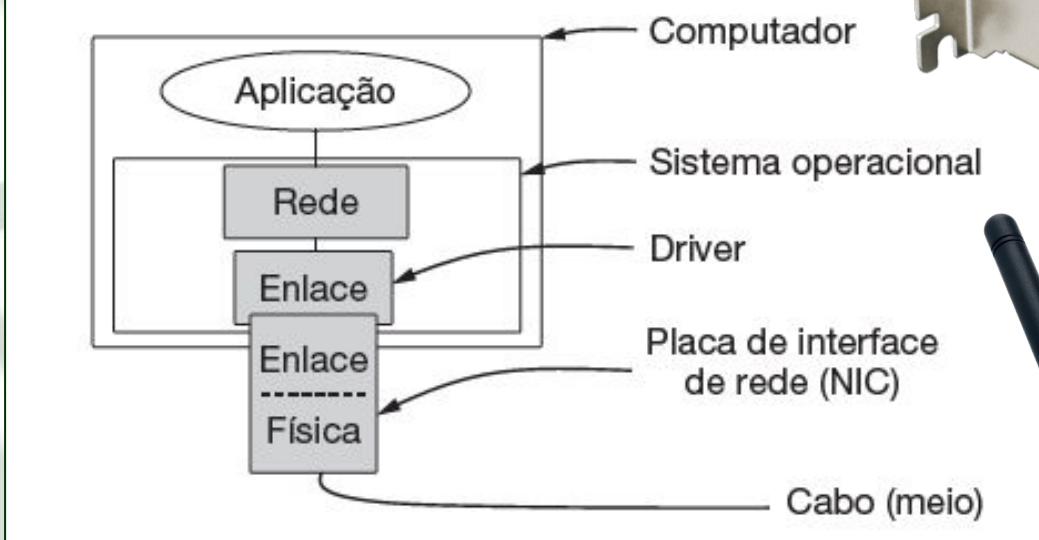
# Camada 2 - Enlace

**O objetivo básico é prover a troca de dados (sem erros) entre os nós diretamente conectados pelo meio físico**



# Camada 2 - Enlace

- Interface de Rede executa as funções de responsabilidade das **camadas Física-Enlace**



## Camada 2 - Enlace

- **Sinais** estão sujeitos a diversos fatores de interferência, gerando erros nos dados transmitidos.
- A **Camada de Enlace** é a primeira barreira protetiva para erros de transmissão.
- *Portanto, uma importante função deste nível é detectar (e quem sabe corrigir?) erros que ocorrem no nível físico.*

# Camada 2 - Enlace

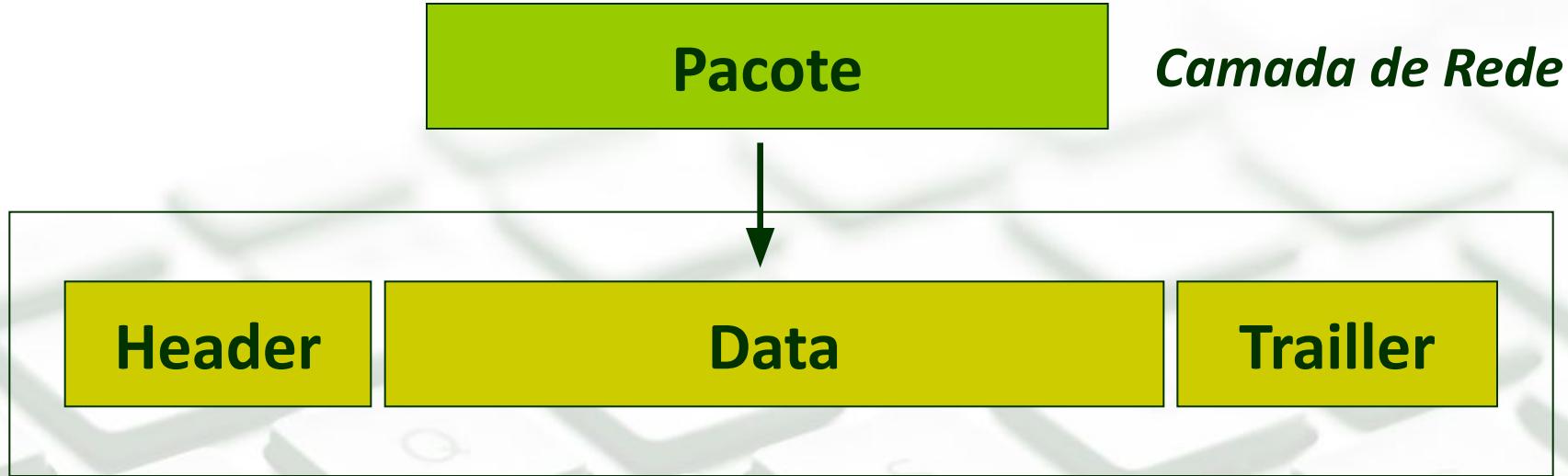
- A camada de enlace oferece portanto um serviço para a camada superior, que é **tornar o canal de transmissão mais “confiável” para o uso dos níveis superiores da pilha.**
- **Mas como detectar erros de transmissão?**
  - É mais prático detectar erros bit a bit?
  - Ou detectar erros em um grupo de bits?

# Camada 2 - Enlace

- A camada de enlace oferece portanto um serviço para a camada superior, que é **tornar o canal de transmissão mais “confiável” para o uso dos níveis superiores da pilha.**
- **Mas como detectar erros de transmissão?**
  - É mais prático detectar erros bit a bit?
  - Ou detectar erros em um grupo de bits?

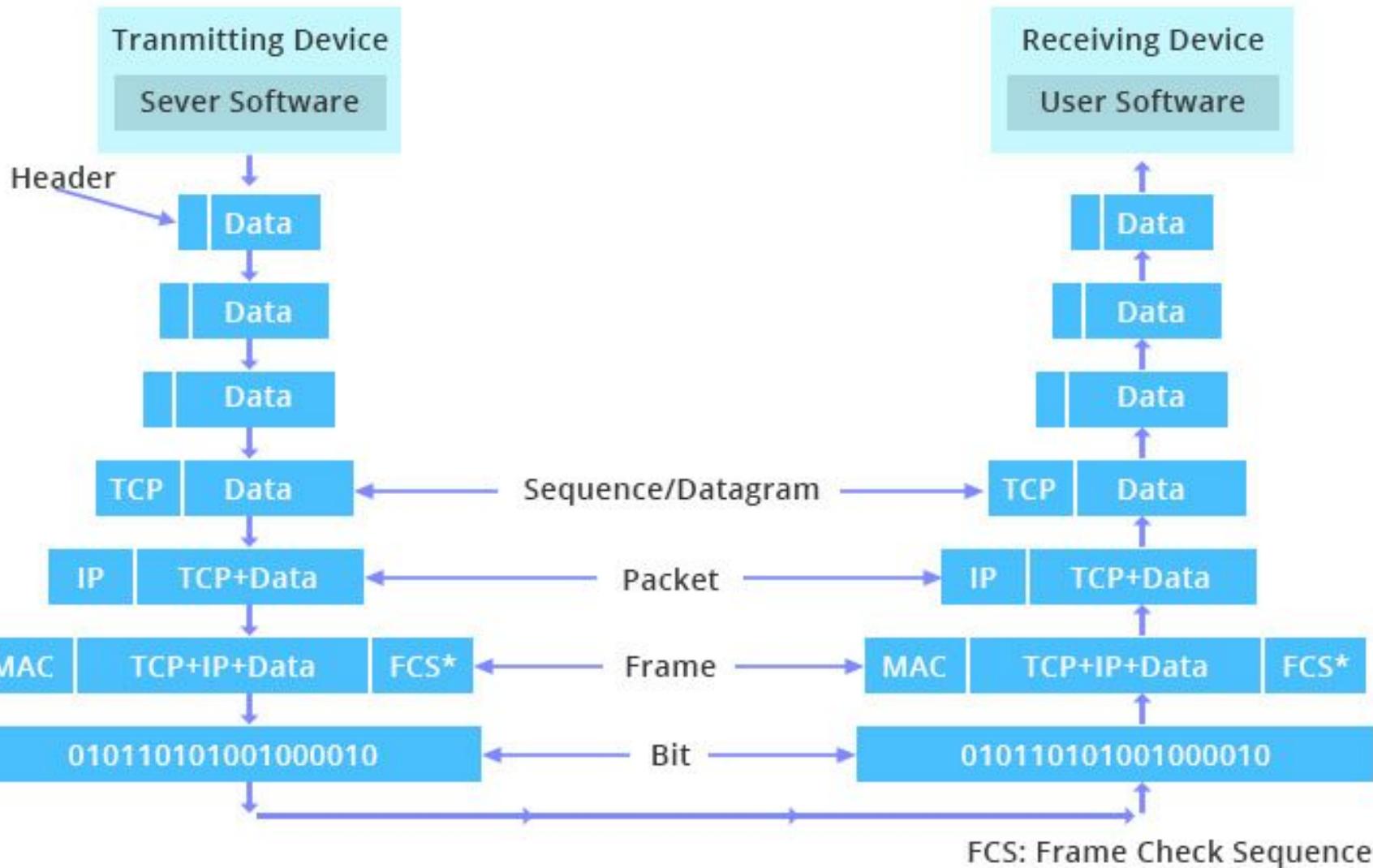
**é necessário agrupar bits => enquadramento**

# Quadro (Frame)



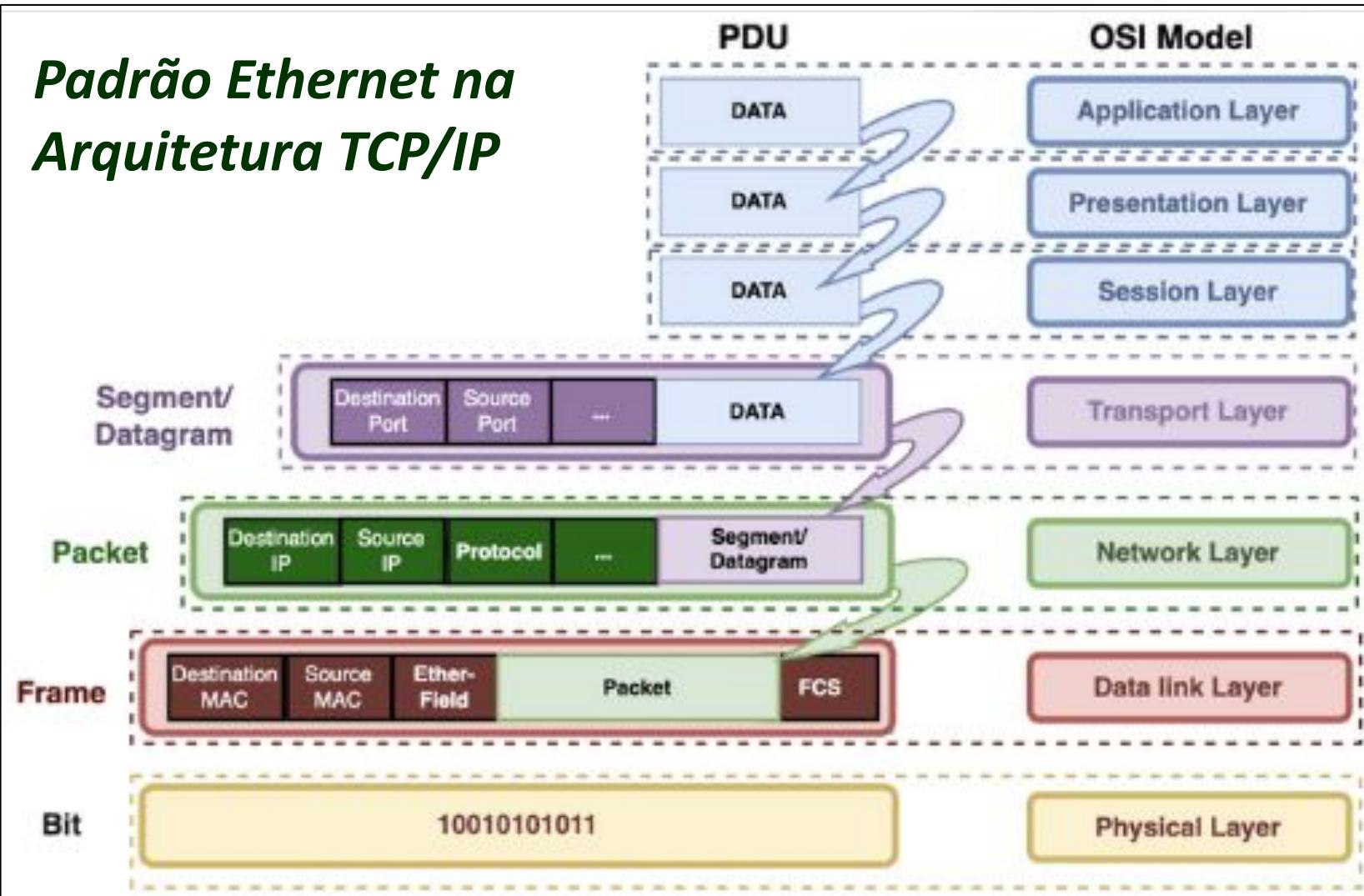
- **Header:** Endereço de Destino e Origem, Contador
- **Data:** Parte útil do quadro
- **Trailler:** Campo de checagem de Erros

# Comunicação TCP/IP



# Comunicação TCP/IP

## *Padrão Ethernet na Arquitetura TCP/IP*



# Funções

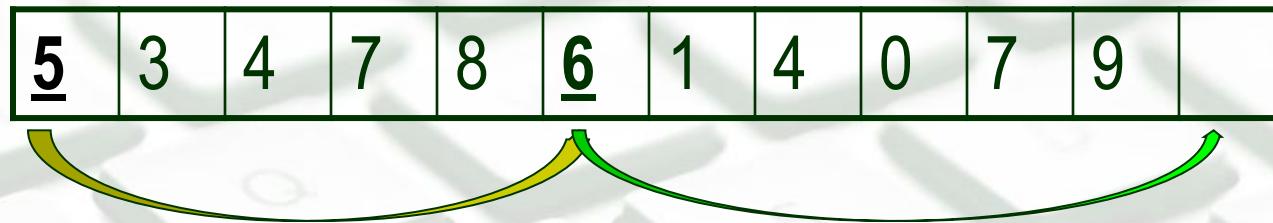
- **Veremos em detalhe os principais protocolos para cada função da camada 2:**
  - 1. Enquadramento
  - 2. Controle de Erros
  - 3. Modelos de Serviço
  - 4. Identificação
  - 5. Controle de Acesso

# Enquadramento

- Para facilitar a detecção de erros, a estratégia da camada de enlace é agrupar uma seqüência de bits, provenientes da camada física, em grupos denominados *quadros/frames*
- Existem 04 métodos básicos para delimitar o tamanho destes quadros...

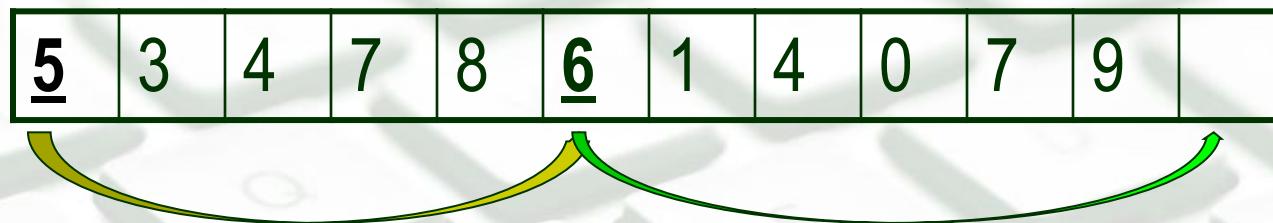
# Contagem de Caracteres

- **Character Count:** Utiliza um campo do cabeçalho do quadro para especificar o número (tamanho) de caracteres deste.

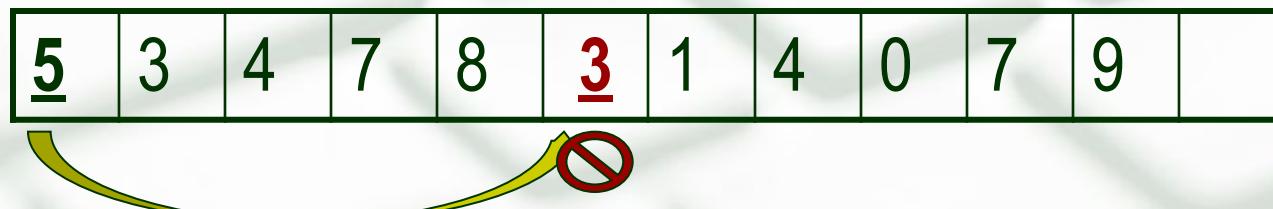


# Contagem de Caracteres

- **Character Count:** Utiliza um campo do cabeçalho do quadro para especificar o número (tamanho) de caracteres deste.



## Problema...



# Inserção de Caracteres

- **Character Stuffing:** Um byte especial (**FLAG**, **SOH** ou **EOT**) para delimitar o início e no fim do quadro.

<b>SOH</b>	Cabeçalho	Carga Útil	Final	<b>EOT</b>
------------	-----------	------------	-------	------------

# Inserção de Caracteres

- **Character Stuffing:** Um byte especial (**FLAG**, **SOH** ou **EOT**) para delimitar o início e no fim do quadro.



## Problema...

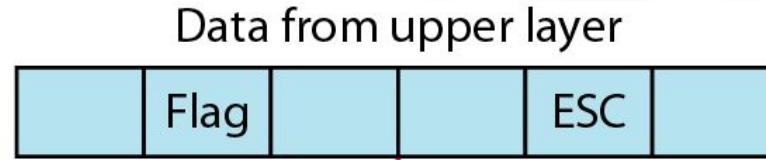


FLAG

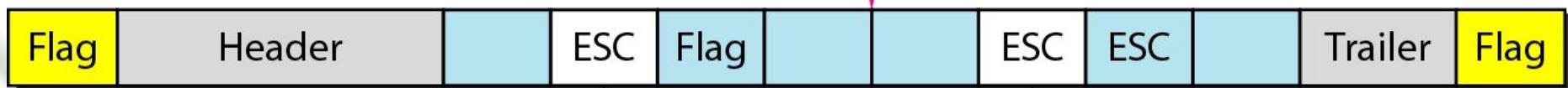
→ **INSERÇÃO DE BYTE ESCAPE  
(OVERHEAD)**

# Inserção de Caracteres

Esta é a solução adotada  
no protocolo PPP



Frame sent

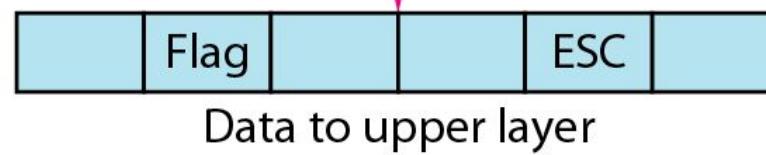


Extra 2 bytes

Frame received



Unstuffed



# Inserção de Bits

- **Bit Stuffing:** Um byte especial (FLAG) **01111110** para delimitar o início e no fim do quadro.

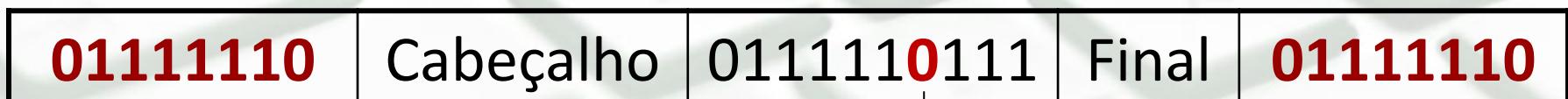


# Inserção de Bits

- **Bit Stuffing:** Um byte especial (FLAG) **01111110** para delimitar o início e no fim do quadro.



## Problema...



Bit  
Stuffing

- MENOS OVERHEAD
- ATUA DIRETAMENTE NA CAMADA FÍSICA

# Inserção de Bits

Esta é a solução adotada no protocolo HDLC

Frame sent



Data from upper layer

00011111001111101000

Stuffed

Frame received



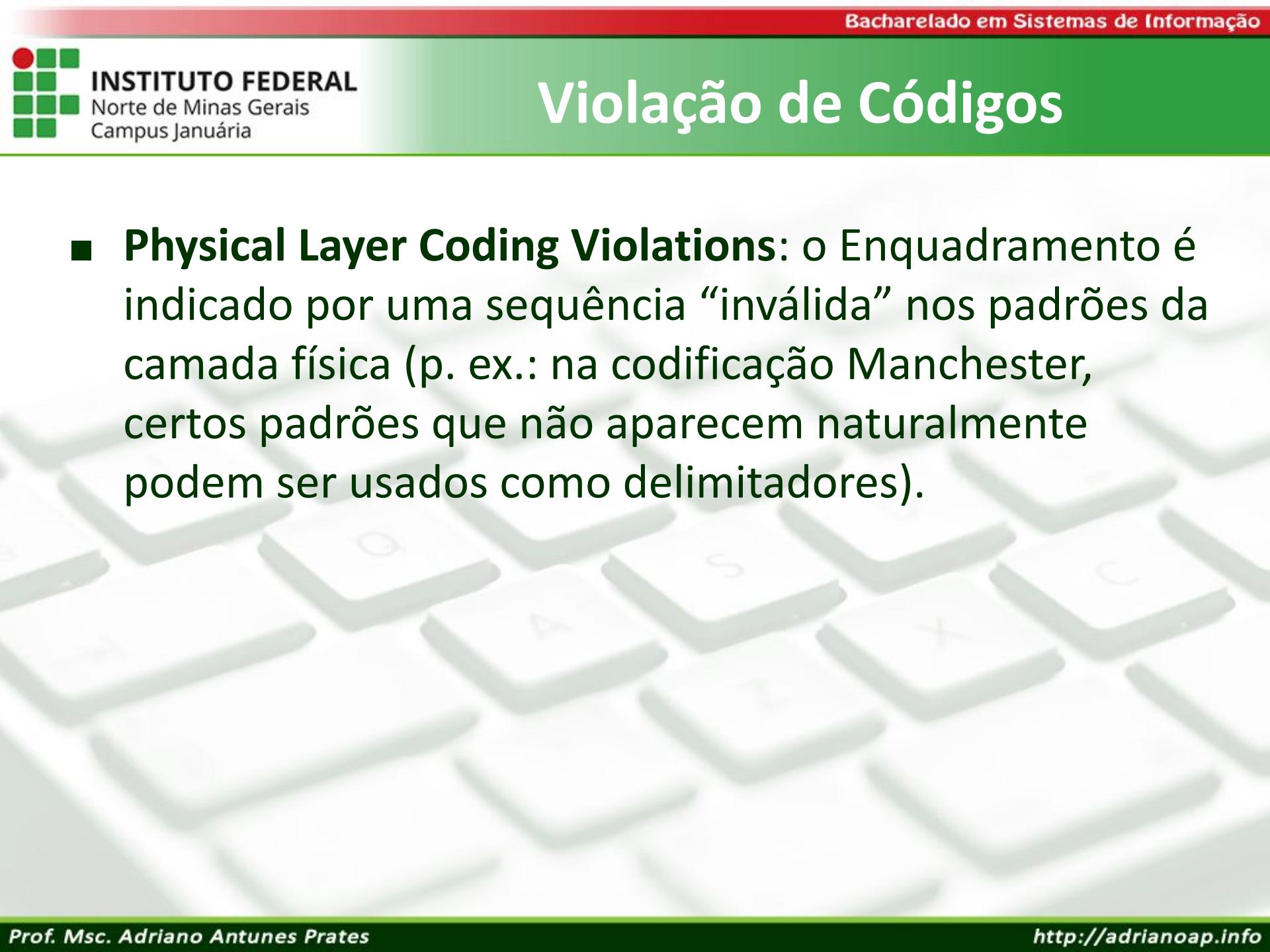
Unstuffed

00011111001111101000

Data to upper layer

# Violação de Códigos

- **Physical Layer Coding Violations:** o Enquadramento é indicado por uma sequência “inválida” nos padrões da camada física (p. ex.: na codificação Manchester, certos padrões que não aparecem naturalmente podem ser usados como delimitadores).



# Exercícios

- Faça o enquadramento das mensagens a seguir...
  - Contagem de Caracteres
    - A S C 3 5
    - 8 4 8 9 5 0 3 3
  - Inserção de Caracteres (Sendo o FLAG = % e ESC = \* )
    - A S % D \* \* K S
    - % J S H % \*
    - S J H % % \* F K L 3 3 %
  - Inserção de Bits
    - 0 1 1 1 1 1 0 0 1 1 1 1 1 1 1 1 0 0 1 0 0 1 1 0
    - 1 1 1 1 1 1 1 1 0 1 1 0 0 0 1 1 0 1 0 0 0 1

# Exercícios

- Retire as mensagens dos quadros a seguir...
  - Contagem de Caracteres
    - 5 6 6 3 7 3 6 5
    - 7 6 6 5 4 1 9 5 6 6 7 2 4 3 4 3
  - Inserção de Caracteres (Sendo o FLAG = % e ESC = \* )
    - % K K \* \* \* % D D H \* % %
    - % \* % S G G T H \* \* \* \* % %
    - % H D 5 6 \* % \* % \* %
  - Inserção de Bits
    - 0 1 1 1 1 1 0 1 1 1 1 0 1 1 0 0 1 0 1 1 1 1 1 1 0
    - 0 1 1 1 1 1 0 1 0 1 1 1 1 0 0 1 0 1 1 1 1 0 0 1 0 1 1 1 1 1 0

# Funções

- **Veremos em detalhe os principais protocolos para cada função da camada 2:**
  - 1. Enquadramento
  - 2. Controle de Erros
  - 3. Modelos de Serviço
  - 4. Identificação
  - 5. Controle de Acesso

# Detecção de Erros

## ■ Há duas estratégias básicas...

- Incluir informações redundantes apenas para permitir a **identificação de erros**.
  - E em caso de detecção, descarta-se o frame.
- Incluir **mais** informações redundantes (gerando **maior overhead**) para permitir a **identificação e a imediata correção** de eventuais de erros.

# Detecção de Erros

## ■ Bit de Paridade

- Um bit extra é adicionado a cada informação, de modo que, a quantidade de bits ‘1’ se torne par (ou ímpar).

- Exemplos:

- Paridade Par

1 0 0 0 1 0 1 1 0

1 0 0 0 1 0 1 0 1

1 1 1 0 1 0 1 0 1

1 1 0 0 1 0 1 0 0

- Paridade Ímpar

1 0 0 0 1 0 1 1 1

1 0 0 0 1 0 1 0 0

1 1 1 0 1 0 1 0 0

1 1 0 0 1 0 1 0 1

# Código de Hamming

- O **Código de Hamming** é um aperfeiçoamento da técnica ‘bit de paridade’, podendo até corrigir erros.
- O código possui inúmeras variantes, contudo é representado da seguinte maneira: **Hamming (7,4)**;  
*Onde...*    7 = Número total de bits total da mensagem – *codeword*;  
                        4 = Número de bits da mensagem original;  
*Portanto neste caso, 3 bits (redundantes) foram inseridos para possibilitar a verificação e correção de erros.*

# Código de Hamming

## ■ Algoritmo:

- Os bits de verificação (bits de Hamming), estarão nas posições  $2^k$  (1, 2, 4, 8, 16, 32...);
- Os bits da mensagem transmitida, estarão nas demais posições (3, 5, 6, 7, 9, 10...);

## ■ Exemplo:

- Deseja-se transmitir a mensagem **1101**. A mensagem enviada será conforme a estrutura a seguir: Hamming (7, 4)

Posição dos Bits	1	2	3	4	5	6	7
Mensagem	H1	H2	1	H3	1	0	1

# Código de Hamming

- Calculando os bits de Hamming...

Posição	1	2	3	4	5	6	7
Mensagem	H1	H2	1	H3	1	0	1

Posição	1	2	3	4	5	6	7
$3 = 1 + 2$	1	1		0			
$5 = 1 + 4$	1	0		1			
$7 = 1 + 2 + 4$	1	1		1			

Mensagem	1	0	1	0	1	0	1
----------	---	---	---	---	---	---	---

# Código de Hamming

*Simulando um erro simples...*

Enviado	1	0	1	0	1	0	1
Recebido	1	0	1	0	0	0	1

*Recalculando Bits de Hamming...*

Posição	1	2	3	4	5	6	7
$3 = 1 + 2$	1	1		0			
$7 = 1 + 2 + 3$	1	1		1			

Calculado	0	0		1			
Recebido	1	0		0			

# Código de Hamming

*Simulando um erro simples...*

Enviado	1	0	1	0	1	0	1
Recebido	1	0	1	0	0	0	1

*Recalculando Bits de Hamming...*

Posição	1	2	3	4	5	6	7
$3 = 1 + 2$	1	1		0			
$7 = 1 + 2 + 3$	1	1		1			

Calculado	0	0		1			
Recebido	1	0		0			
$(1 + 4 = 5)$	1	0		1			

# Exercícios

- Informe qual deverá ser o bit de paridade (Par) nas mensagens a seguir:
  - a) 1 0 0 1 1 0 1
  - b) 1 1 1 0 1 0 1
  - c) 1 1 1 0 1 1 0 1
  - d) 0 0 1 1 0 1 1 1
- Faça o algoritmo de Hamming (7,4) e descubra qual será a ***codeword*** (mensagem enviada) para os seguintes dados:
  - a) 1 0 1 1
  - b) 0 0 1 1
  - c) 1 1 1 1
  - d) 1 0 0 1
- Obtenha a mensagem original a partir das ***codeword's*** abaixo.  
(Em alguns casos poderão ter ocorrido erros simples)
  - a) 0 0 1 0 1 1 1
  - b) 1 0 1 0 0 1 0
  - c) 1 0 1 0 1 1 0
  - d) 1 1 0 1 0 0 0

# Exercícios

- Faça o algoritmo de Hamming (11,7) e descubra qual será a ***codeword*** (mensagem enviada) para os seguintes dados:
  - a) 1 0 0 1 0 0 0
  - b) 1 1 0 0 0 0 1
  - c) 1 1 0 1 1 0 1
  - d) 1 1 0 1 0 0 1
  
- Obtenha a mensagem original a partir das ***codeword's*** abaixo.  
(Em alguns casos poderão ter ocorrido erros simples)
  - a) 1 0 0 1 1 0 0 0 0 0 1
  - b) 1 1 0 1 1 0 0 1 1 0 0
  - c) 0 1 1 0 1 0 1 1 1 0 1
  - d) 1 0 0 1 1 0 0 0 0 0 0

# Custo x Benefício

- Códigos de correção de erros são muito úteis em meios instáveis (como *wireless*), contudo, necessitam de muitos bits redundantes.
- Suponha quadros de 1000 bits. Em uma transmissão de 1 Mb de informação, seria necessário 10.000 bits para corrigir um erro simples de um quadro defeituoso.
- Enquanto que, se cada bloco tivesse apenas um bit de paridade (portanto, mais 1000 bits) , ocorrendo um erro simples, seria necessário apenas retransmitir o quadro defeituoso =  $1000 + 1000$  bits = 2.000 bits.

- **CRC - Código de Redundância Cíclica** é o método mais utilizado para detecção de erros atualmente.
- É capaz de detectar uma grande faixa de erros de transmissão, isolados ou em rajadas.
- Consiste na ideia de acrescentar ao final de um frame, o resto da divisão dos bits do frame por um polinômio gerador.
- **O receptor valida o frame verificando se ele é divisível pelo mesmo polinômio gerador.**

**Exemplo:**

Informação a ser enviada:

**1 0 0 1 1 1 1**

Polinômio Gerador  $G(x)$ :

$X^3 + X^2 + 1 \Rightarrow 1101$

Inicialmente acrescenta-se ao final da mensagem original, a qtde. de bits zero equivalente ao maior grau de  $G(x)$ .

$$\begin{array}{r}
 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\
 \underline{1} & \underline{1} & \underline{0} & \underline{1} & & & & & & & \\
 0 & 1 & 0 & 0 & 1 & & & & & & \\
 \underline{1} & \underline{1} & \underline{0} & \underline{1} & & & & & & & \\
 0 & 1 & 0 & 0 & 1 & & & & & & \\
 \underline{1} & \underline{1} & \underline{0} & \underline{1} & & & & & & & \\
 0 & 1 & 0 & 0 & 1 & & & & & & \\
 \underline{1} & \underline{1} & \underline{0} & \underline{1} & & & & & & & \\
 0 & 1 & 0 & 0 & 0 & 1 & & & & & \\
 \underline{1} & \underline{1} & \underline{0} & \underline{1} & & & & & & & \\
 0 & 1 & 0 & 0 & 0 & 0 & 1 & & & & \\
 \underline{1} & \underline{1} & \underline{0} & \underline{1} & & & & & & & \\
 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & & & \\
 \underline{1} & \underline{1} & \underline{0} & \underline{1} & & & & & & & \\
 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & & \\
 \underline{1} & \underline{1} & \underline{0} & \underline{1} & & & & & & & \\
 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & \\
 \underline{1} & \underline{1} & \underline{0} & \underline{1} & & & & & & & \\
 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
 \underline{1} & \underline{1} & \underline{0} & \underline{1} & & & & & & & \\
 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1
 \end{array}$$

**MENSAGEM ENVIADA = 1 0 0 1 1 1 1 0 1 1**

- O CRC-16 consegue detectar:
  - 100% de erros simples, duplos, e em qtde ímpar.
  - 99,998% de rajadas > 18 bits;
- Já o **CRC-32** se tornou o algoritmo padrão de detecção de erros nas redes locais cabeadas.

## Padrão Internacional - IEEE 802

$$\begin{aligned} \text{CRC}_{32}(x) = & x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + \\ & x^7 + x^5 + x^4 + x^2 + x^1 + 1 \end{aligned}$$

# Exercícios

- Informe qual deverá ser o fluxo de bits transmitidos pela camada de enlace, utilizando o método CRC, sendo o polinômio gerador =  $x^3 + 1$ 
  - a) 1 0 0 1 1 1
  - b) 1 1 1 0 1 0
  - c) 1 0 1 0 1 1
  - d) 1 1 0 0 0 1
  
- Os quadros a seguir foram recebidos pela camada de enlace. Verifique se ocorreu erro durante a transmissão, sabendo que foi utilizado o método CRC, sendo o polinômio gerador:  
 $x^3 + x^1 + 1$ 
  - a) 1 1 1 0 1 1 0 1 1
  - b) 1 0 0 0 1 0 1 1 0
  - c) 1 0 0 1 1 1 0 1 1
  - d) 1 1 1 0 1 1 0 1 1

# Funções

- Veremos em detalhe os principais protocolos para cada função da camada 2:
  - 1. Enquadramento
  - 2. Controle de Erros
  - **3. Modelos de Serviço**
  - 4. Identificação
  - 5. Controle de Acesso

# Modelos de Serviço

- Analisamos algoritmos para checagem de erros em frames recebidos, mas...
  - E se o frame nem chega ao destino?
    - Devido Atenuação ou Colisão
  - E se chegou e foi descartado, como fica o serviço que dependia dele?

Por isso a camada de Enlace oferece diferentes modelos de **Controle de Fluxo** dos frames...

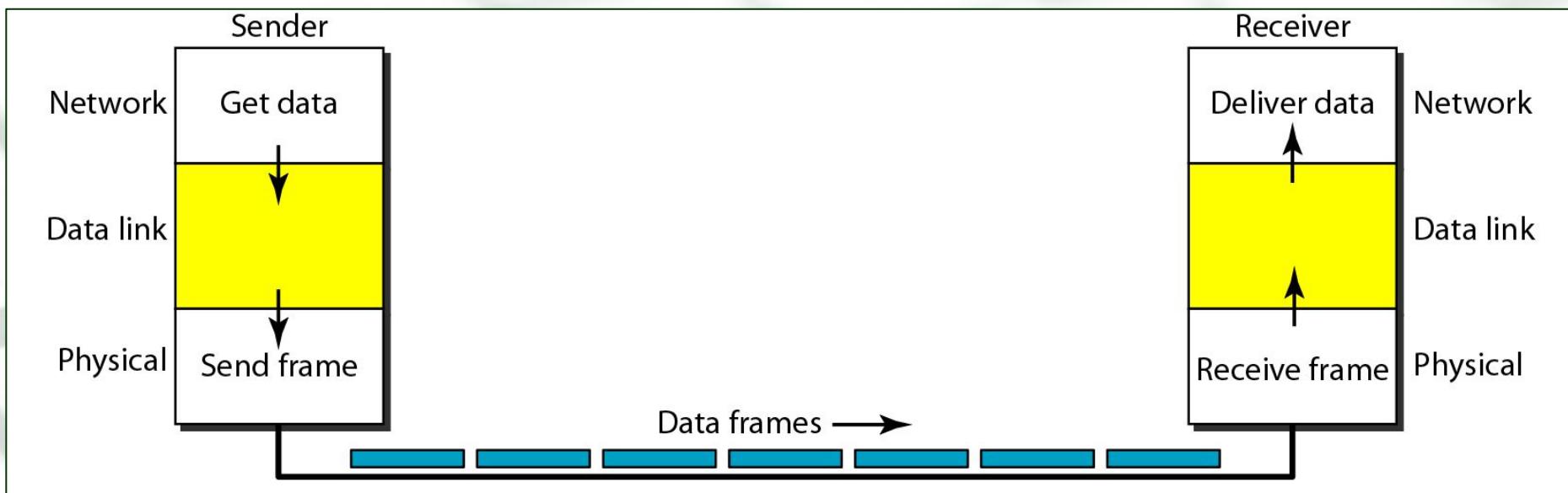
# Modelos de Serviço

## ■ Serviço sem Conexão e sem Confirmação

- O T(x) envia quadros independentes ao R(x), sem o estabelecimento de conexões prévias e sem esperar confirmações de recebimento ou de falhas.
- Não há buffers, retransmissão de frames, nem garantias de entrega, mas é mais simples e rápido.
- Receptor pode ser inundado por frames.
- Controles de fluxo mais refinados são de responsabilidade de camadas superiores.
- É o padrão adotado nas redes **Ethernet**.

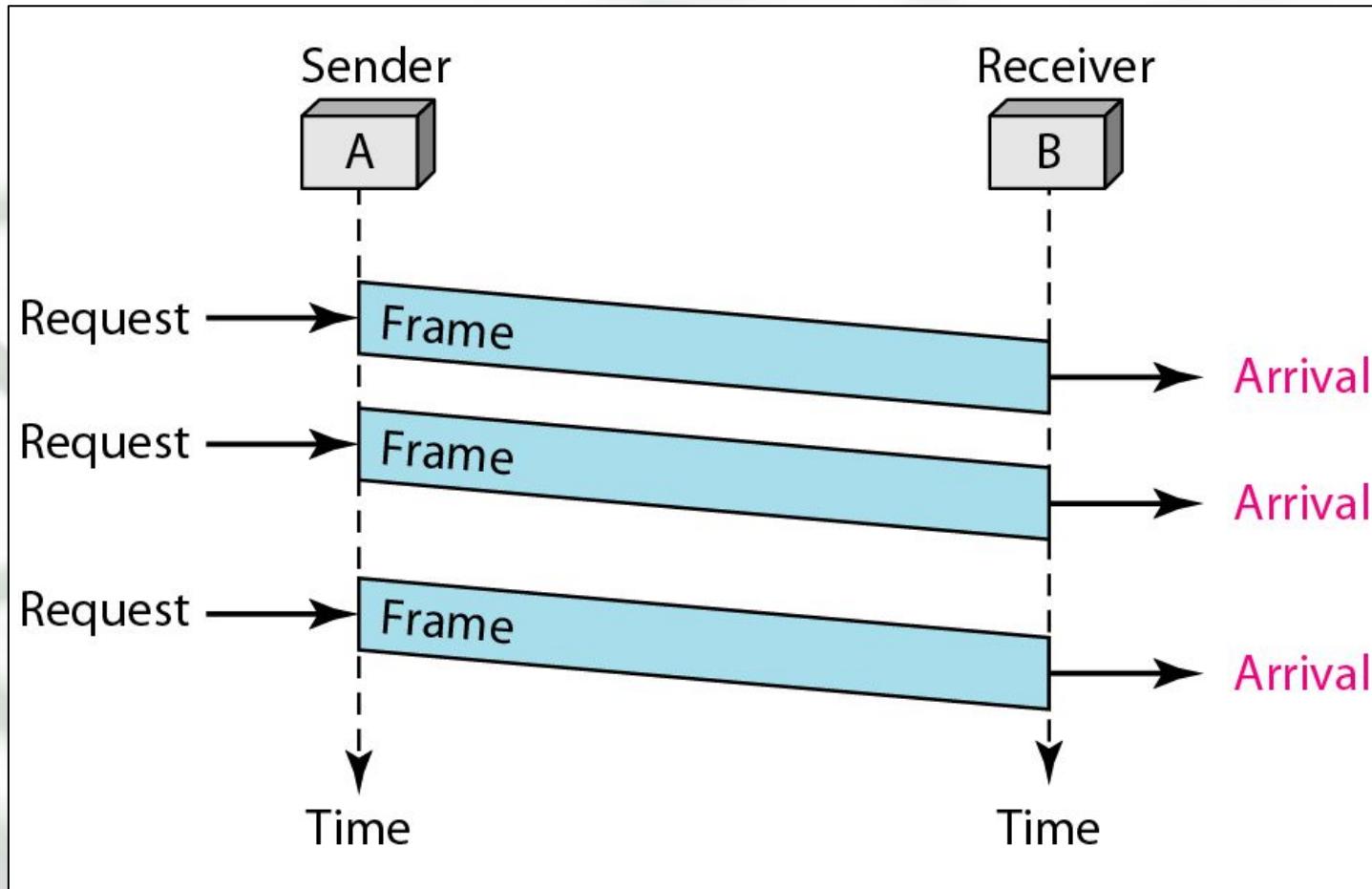
# Modelos de Serviço

## ■ Serviço sem Conexão e sem Confirmação



# Modelos de Serviço

## Serviço sem Conexão e sem Confirmação



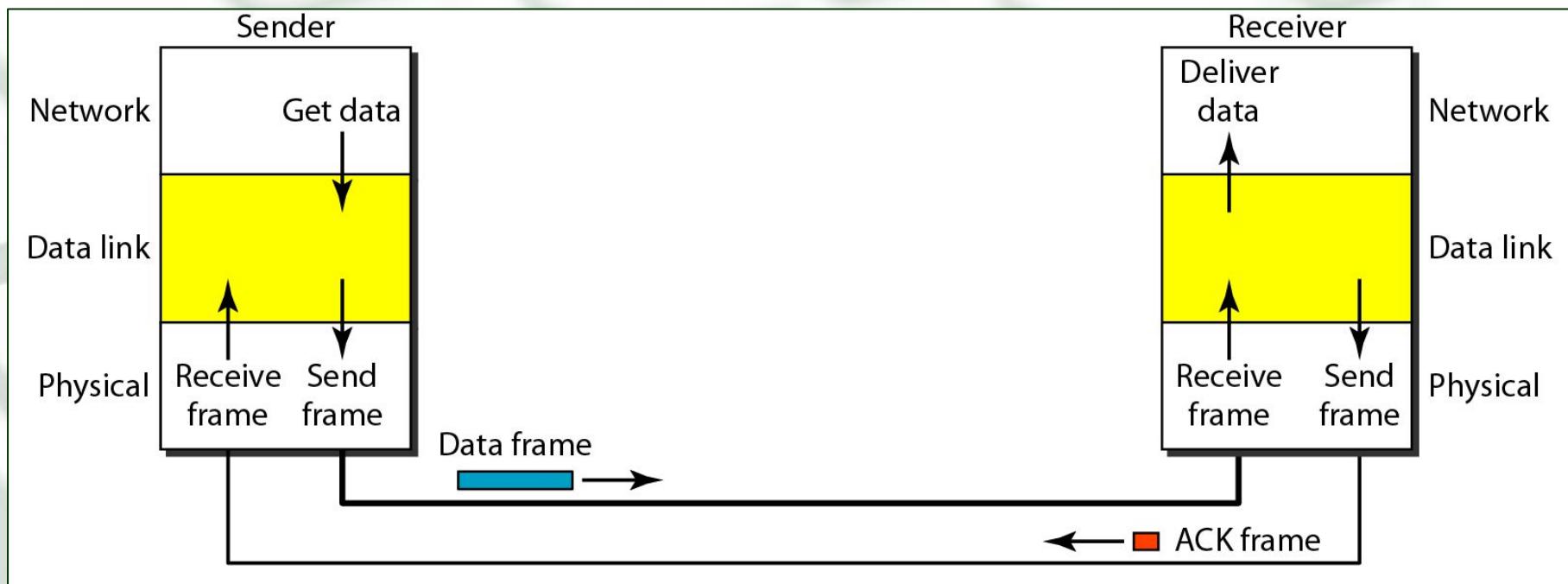
# Modelos de Serviço

## ■ **Serviço sem Conexão, mas com Confirmação**

- Frames devem ser identificados, e mantidos em buffer.
- Novos frames só podem ser enviados mediante confirmação de recebimento (ACK) ou notificação de falha do frame anterior.
- Temporizador é necessário para aguardar pelas notificações.
- Receptor deve estar preparado para eventualmente receber frames duplicados (caso ACK seja perdido).

# Modelos de Serviço

- **Serviço sem Conexão, mas com Confirmação**
  - Algoritmo **STOP and WAIT**

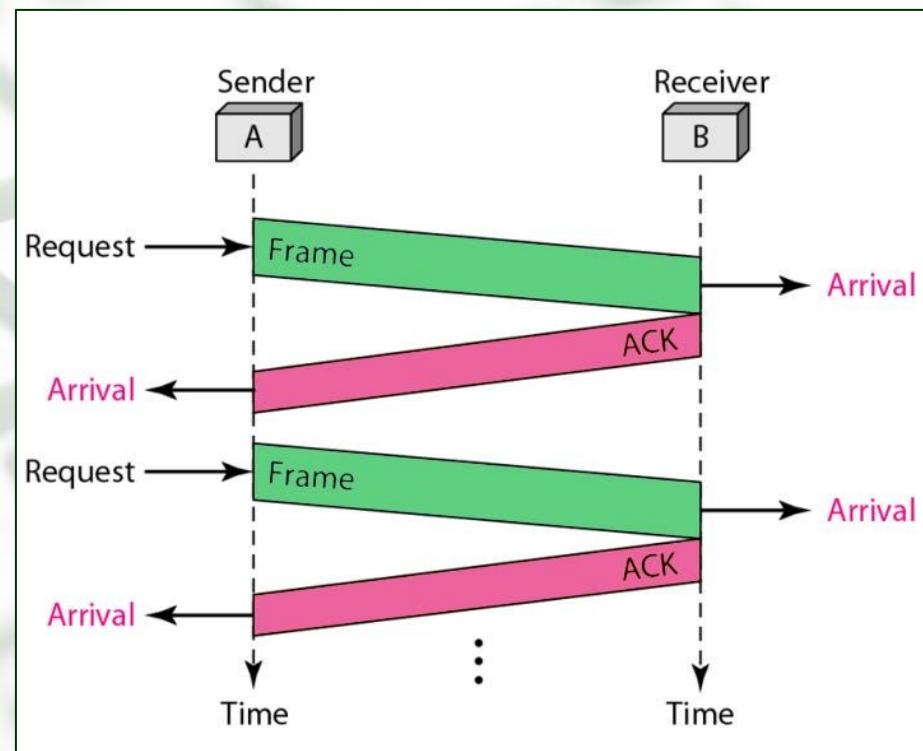


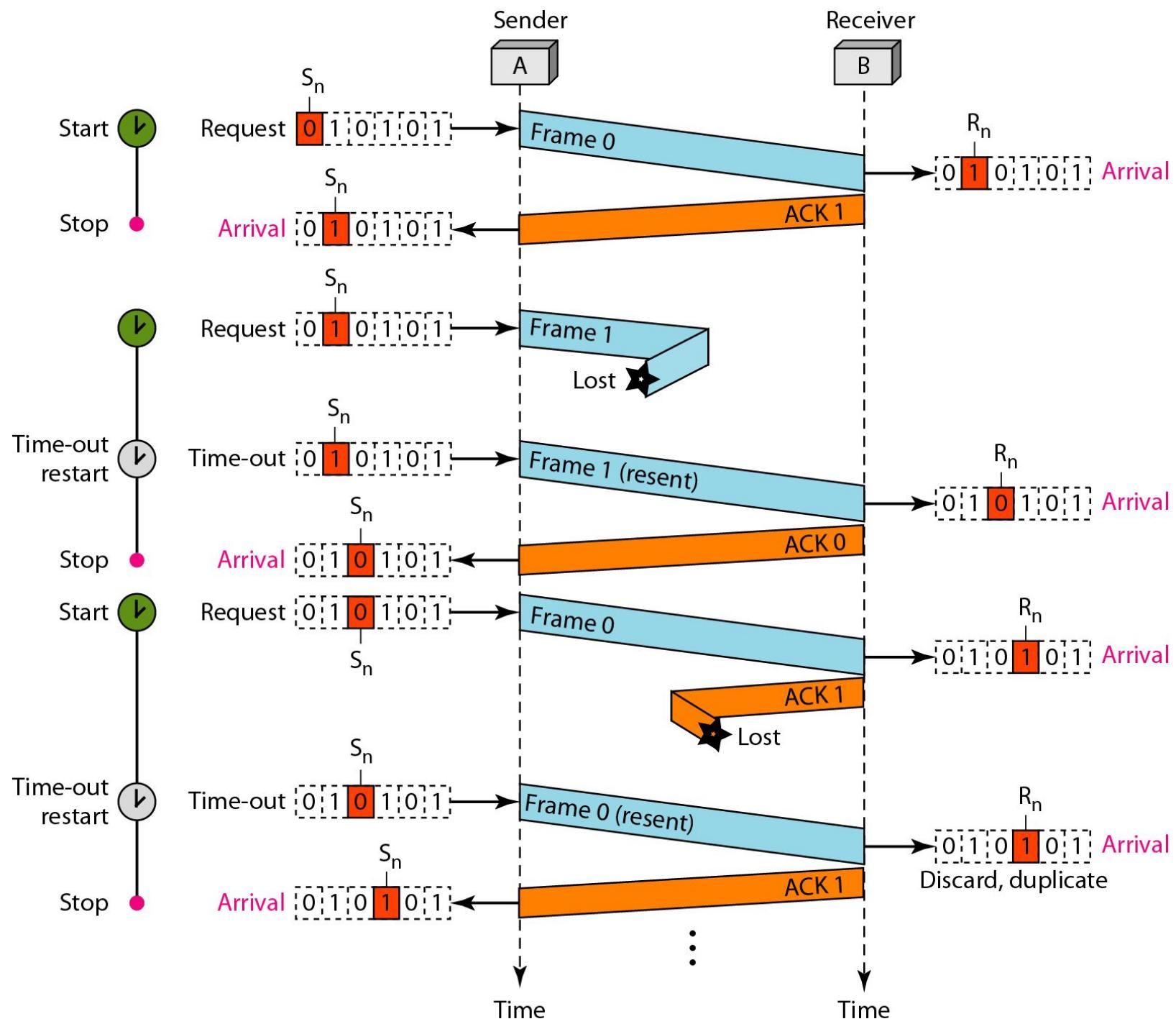
# Modelos de Serviço

- **Serviço sem Conexão, mas com Confirmação**
  - Algoritmo **STOP and WAIT**

*Como  $T(x)$  só envia 1 frame por vez, basta 1 bit para identificar cada frame ...*

Ora é o Frame 0  
Ora é o Frame 1







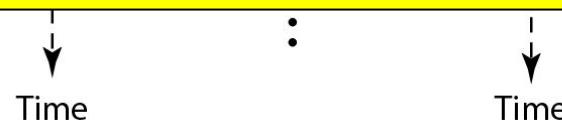
Mas imagine um canal com BW de 1 Mbps e latência de 20ms...

Logo, a capacidade do canal (produto BW \* Latência) é de 20.000 bits.

Imagine que os frames para este protocolo de enlace sejam de 1.000 bits...

Então, nesse modelo de serviço a vazão de dados não é mais determinada pelo Meio de Transmissão, mas sim pelo tamanho máximo do frame.

**EXERCÍCIO:** Calcule qual seria o tempo mínimo dessa transmissão, e o throughput real.



# Modelos de Serviço

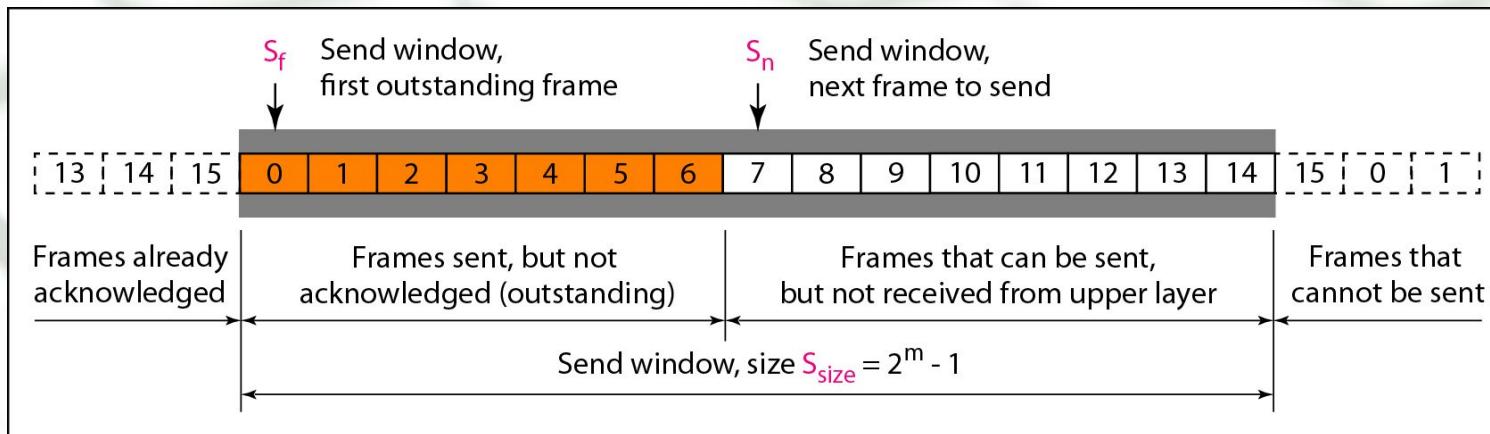
## ■ Serviço Orientado a Conexão

- Nós estabelecem uma conexão antes dos dados serem concretamente transmitidos.
- A conexão é estabelecida, fazendo-se ambos inicializarem recursos como buffers e variáveis de controle.
- O T(x) pode enviar vários quadros mesmo sem ter recebido reconhecimentos dos quadros anteriores.
- Quadros são transmitidos com garantias de entrega, e após o último quadro a conexão é encerrada.

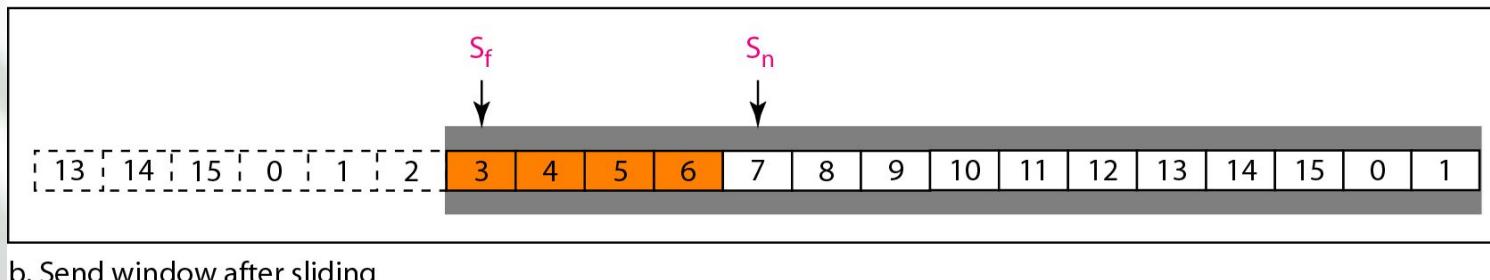
# Modelos de Serviço

## ■ Serviço Orientado a Conexão

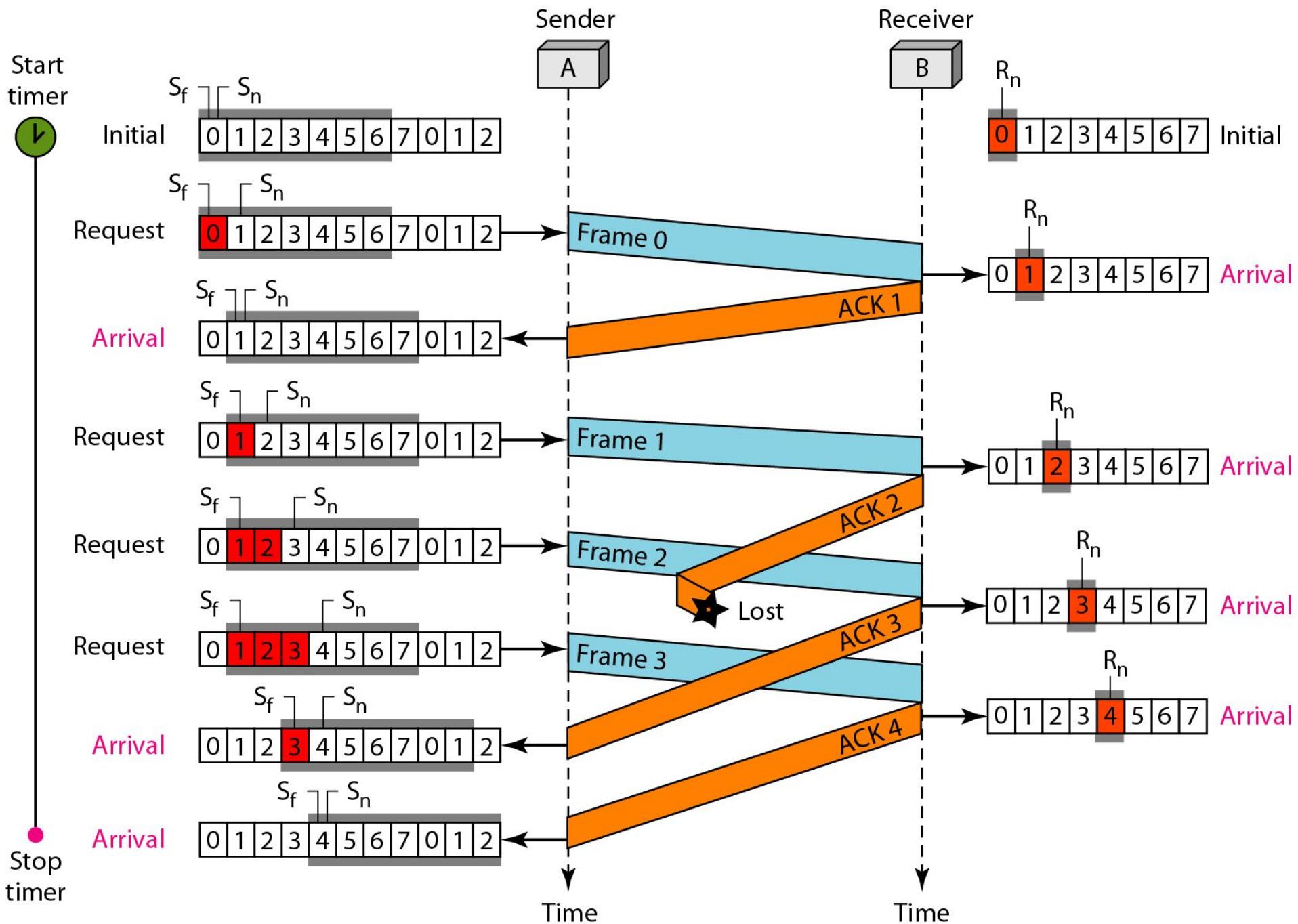
### □ Algoritmo SLIDING WINDOW



a. Send window before sliding



b. Send window after sliding



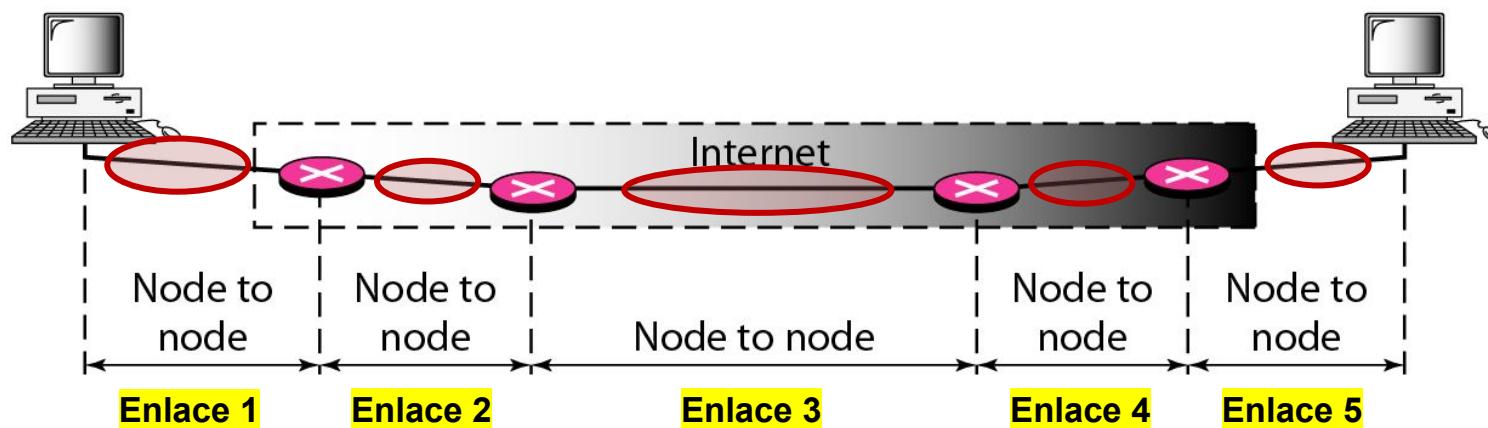
MAS...



# Modelos de Serviço

O modelo de serviço orientado a conexão em nível de enlace é **MUITO RARO** nas redes modernas devido ao custo operacional de manter tantas conexões.

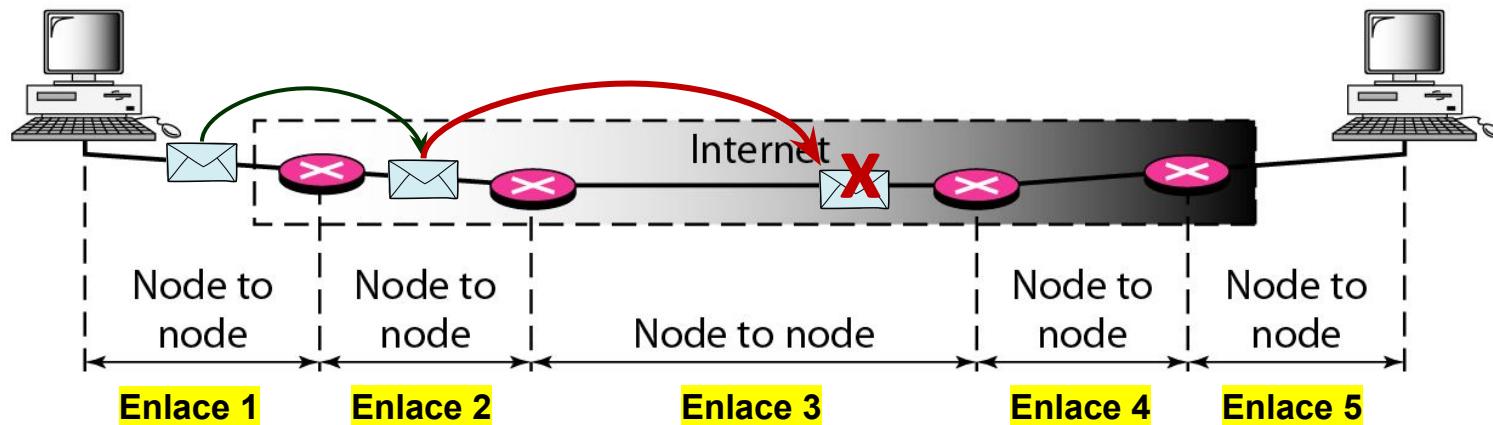
Node to node: Data link layer  
Host to host: Network layer  
Process to process: Transport layer



# Modelos de Serviço

A maioria das tecnologias em nível de enlace, como Ethernet, Wi-Fi, 4G, utilizam modelos de serviço não orientado à conexão, sem garantias de entrega.

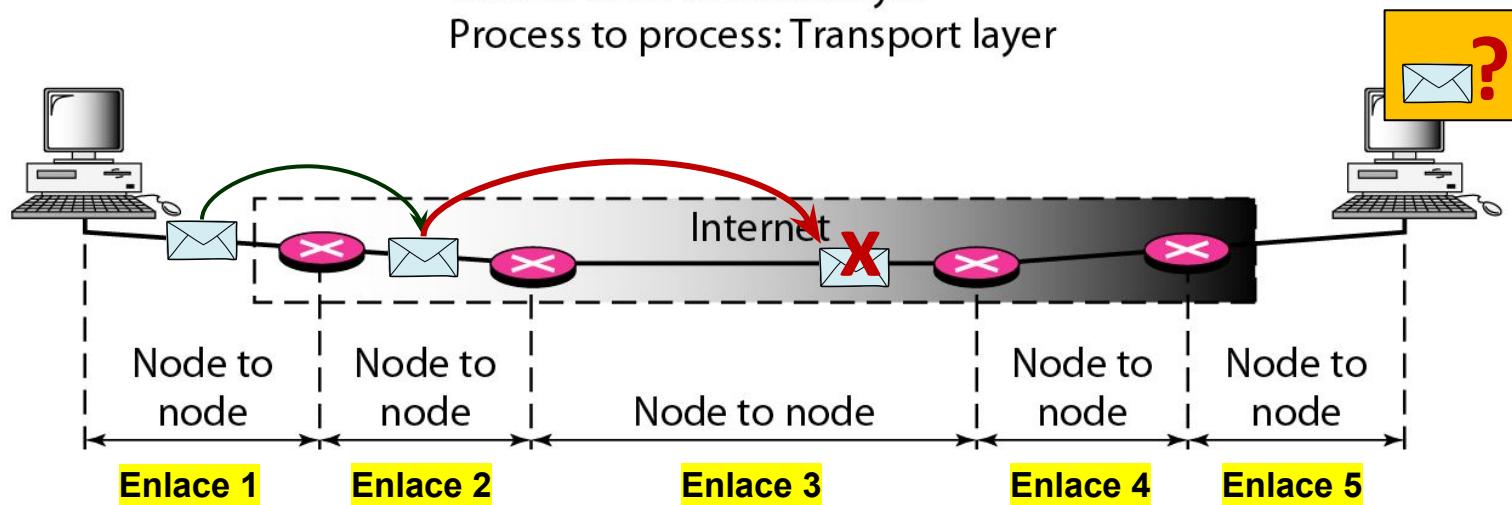
Node to node: Data link layer  
Host to host: Network layer  
Process to process: Transport layer



# Modelos de Serviço

A responsabilidade pela entrega confiável de mensagens, na prática, é atribuída à camada de Transporte, por meio do protocolo TCP, que estudaremos à frente.

Node to node: Data link layer  
Host to host: Network layer  
Process to process: Transport layer

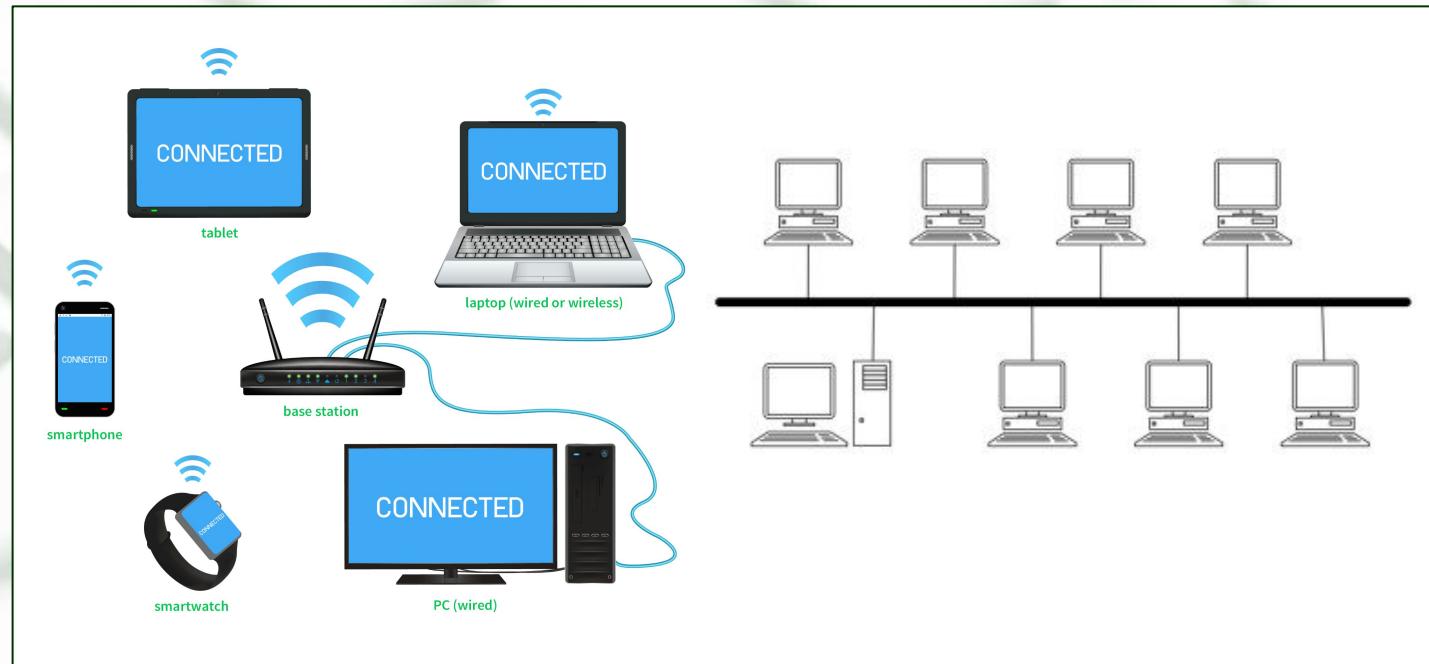
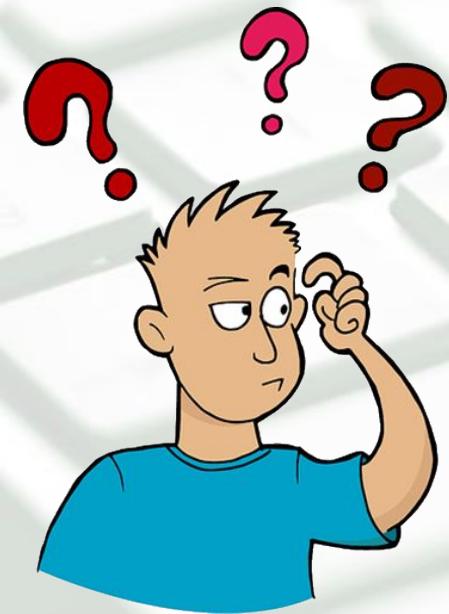


# Funções

- **Veremos em detalhe os principais protocolos para cada função da camada 2:**
  - 1. Enquadramento
  - 2. Controle de Erros
  - 3. Modelos de Serviço
  - 4. Identificação
  - 5. Controle de Acesso

# Pense comigo...

Em um **meio compartilhado**, como identificar corretamente o destinatário de uma mensagem?

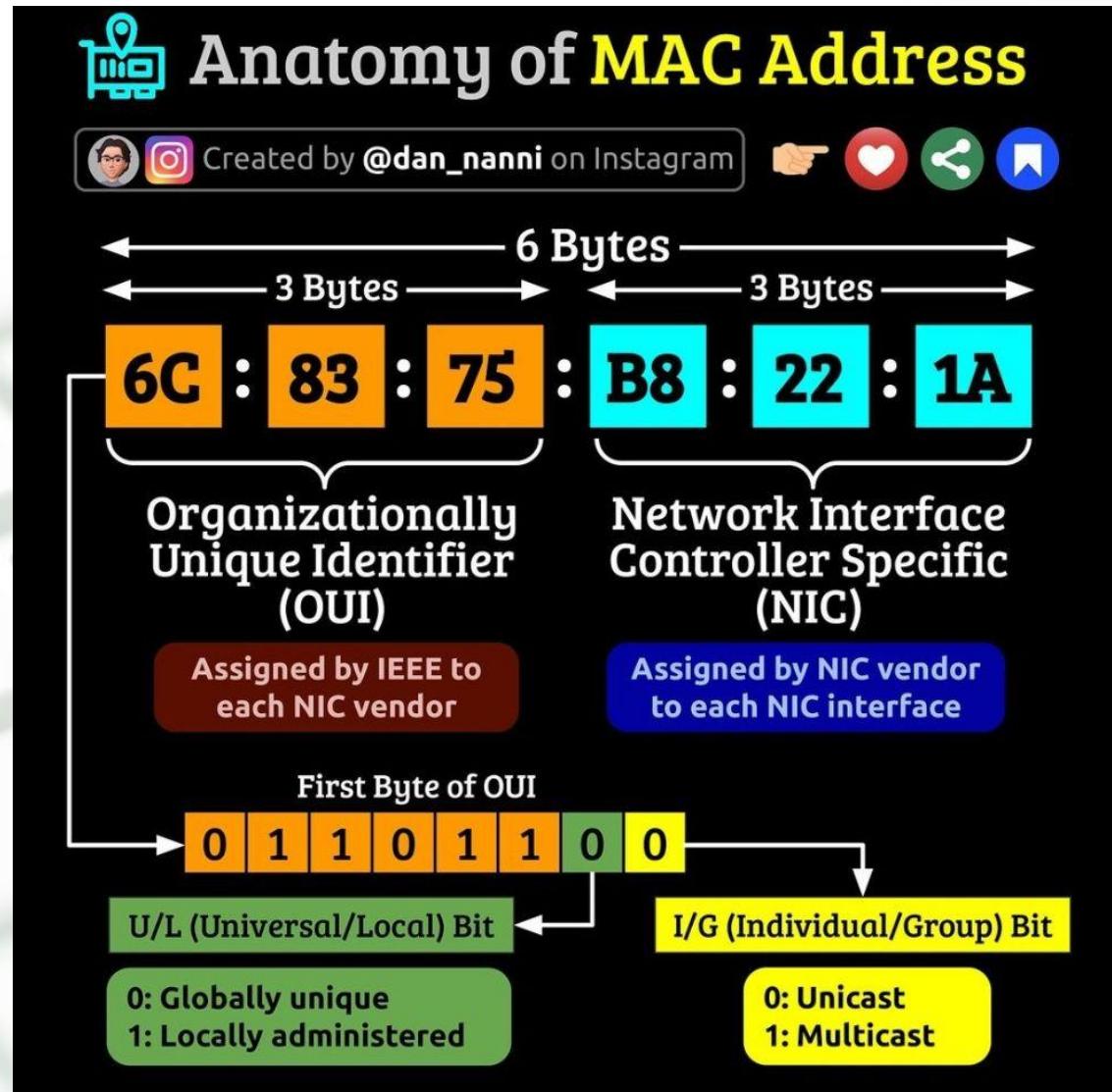


# MAC Address

**MAC Address** é um identificador único atribuído em toda e qualquer interface de rede de um host.

- Também conhecido por **Endereço Físico**, o endereço MAC é um código “único e fixo”, registrado em fábrica em todas as placas de redes dos dispositivos.
- O endereço é composto por **48 bits**, sendo representado em **6 grupos de dois dígitos hexadecimais**.
- Formato: 00-4B-0C-12-34-56  


# MAC Address



# MAC Address

- Site para consulta de fabricantes...



# MAC Address

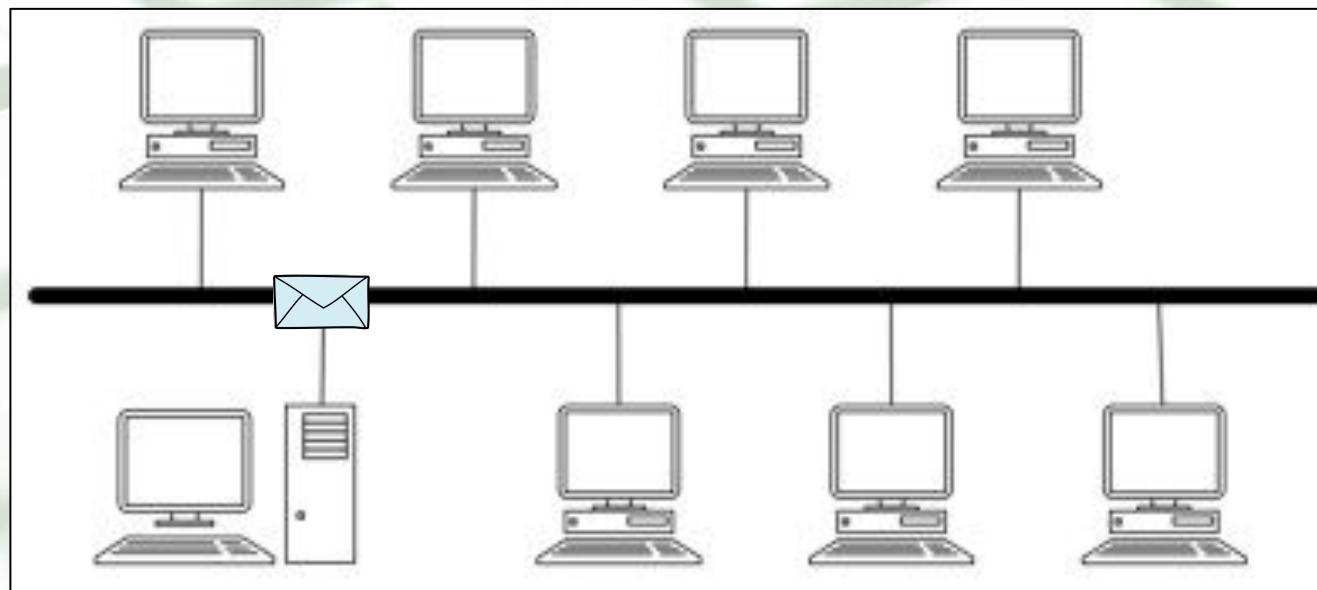


# MAC Address

```
adriano@adriano-pc:~  
Arquivo Editar Ver Pesquisar Terminal Ajuda  
$> ifconfig  
docker0: flags=4099<UP,BROADCAST,MULTICAST> mtu 1500  
    inet 172.17.0.1 netmask 255.255.0.0 broadcast 172.17.255.255  
        ether 02:42:88:75:fb:b5 txqueuelen 0 (Ethernet)  
            RX packets 0 bytes 0 (0.0 B)  
            RX errors 0 dropped 0 overruns 0 frame 0  
            TX packets 0 bytes 0 (0.0 B)  
            TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0  
  
enp3s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500  
    inet 192.168.1.9 netmask 255.255.255.0 broadcast 192.168.1.255  
    inet6 fe80::c3bd:7e3e:f528:f57c prefixlen 64 scopeid 0x20<link>  
        ether 10:c3:7b:c4:69:df txqueuelen 1000 (Ethernet)  
            RX packets 339011 bytes 367634122 (367.6 MB)  
            RX errors 0 dropped 5502 overruns 0 frame 0  
            TX packets 142344 bytes 44759329 (44.7 MB)  
            TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

# MAC Address

- Cada frame sempre carrega consigo:
  - **MAC Address de Origem**
  - **MAC Address de Destino**

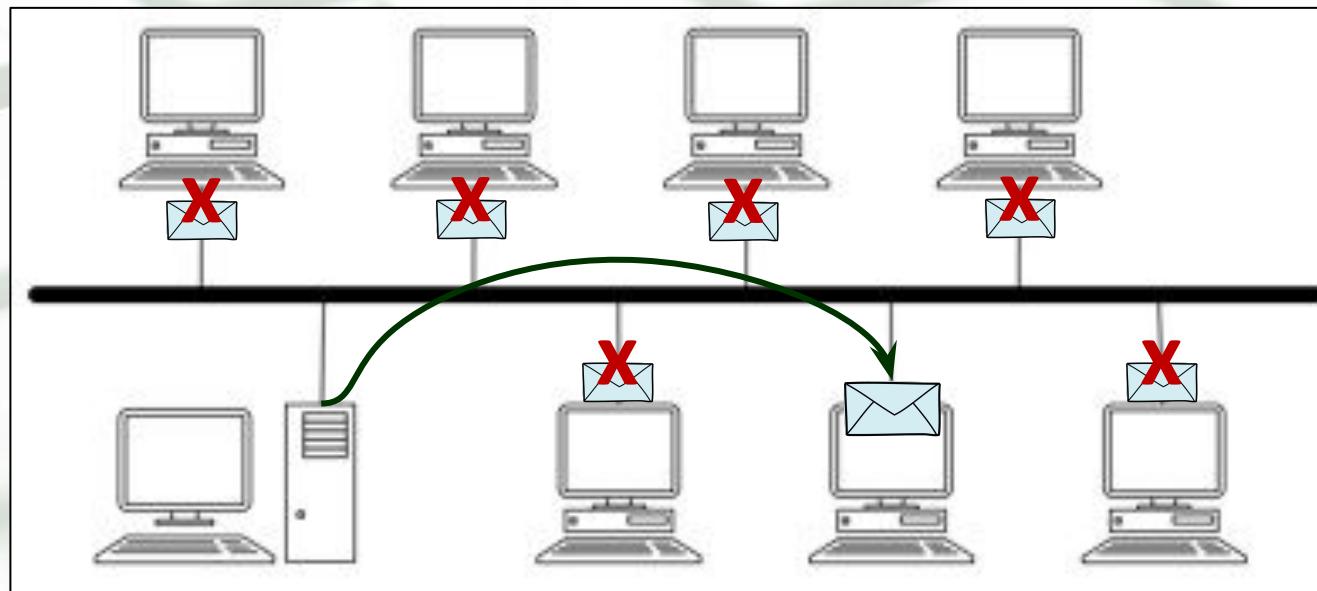


# MAC Address

- Cada frame sempre carrega consigo:

- **MAC Address de Origem**
- **MAC Address de Destino**

É através desse registro que uma placa de rede saberá se deve aceitar ou descartar um frame.

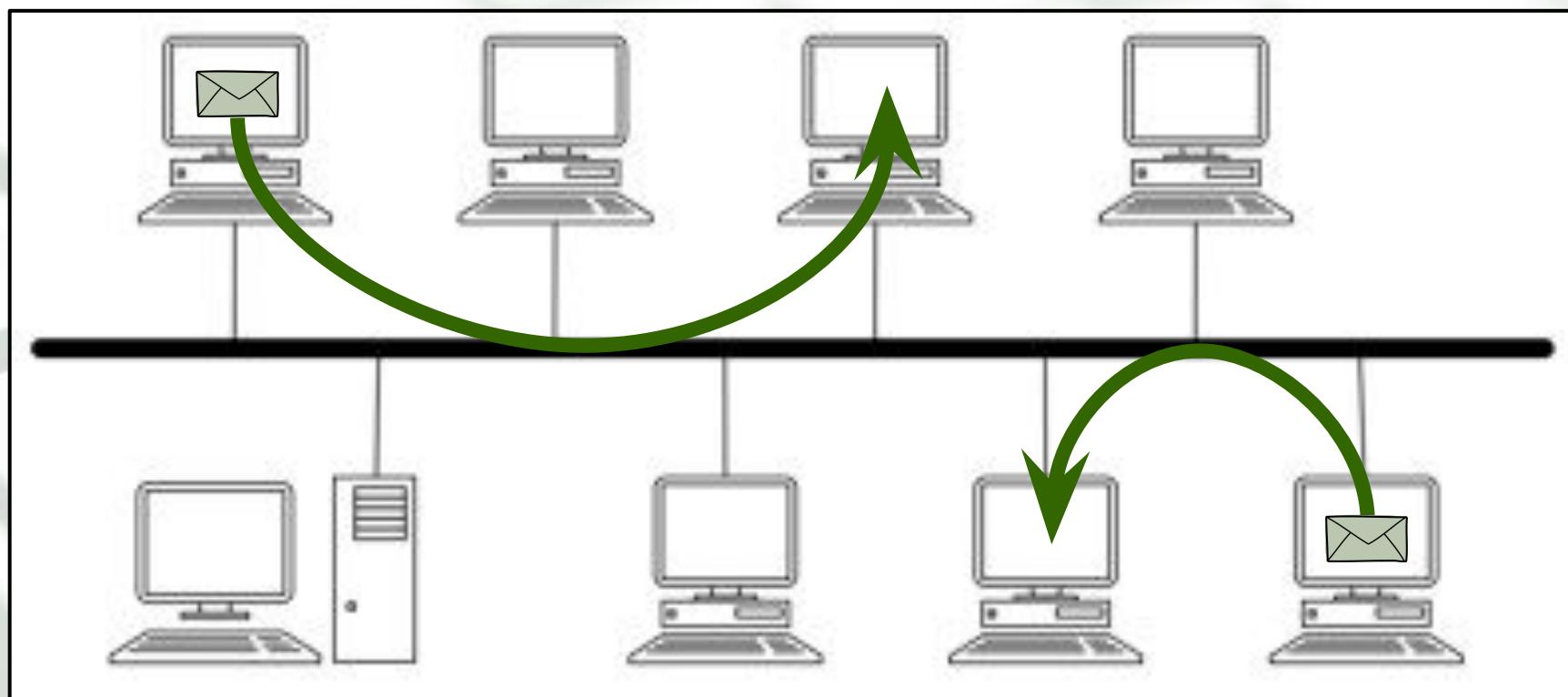


# Funções

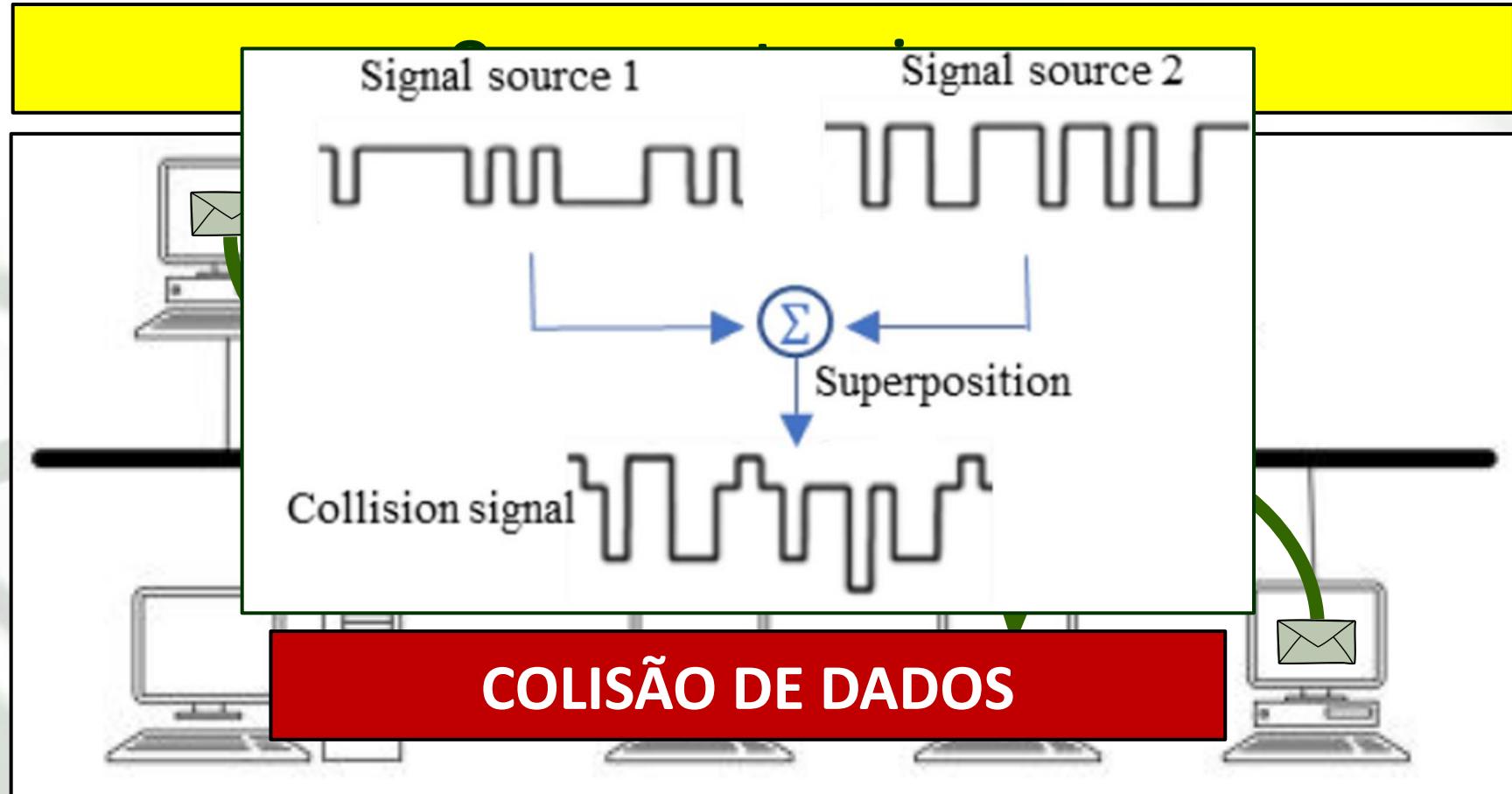
- **Veremos em detalhe os principais protocolos para cada função da camada 2:**
  - 1. Enquadramento
  - 2. Controle de Erros
  - 3. Modelos de Serviço
  - 4. Identificação
  - 5. Controle de Acesso

# Meios Compartilhados

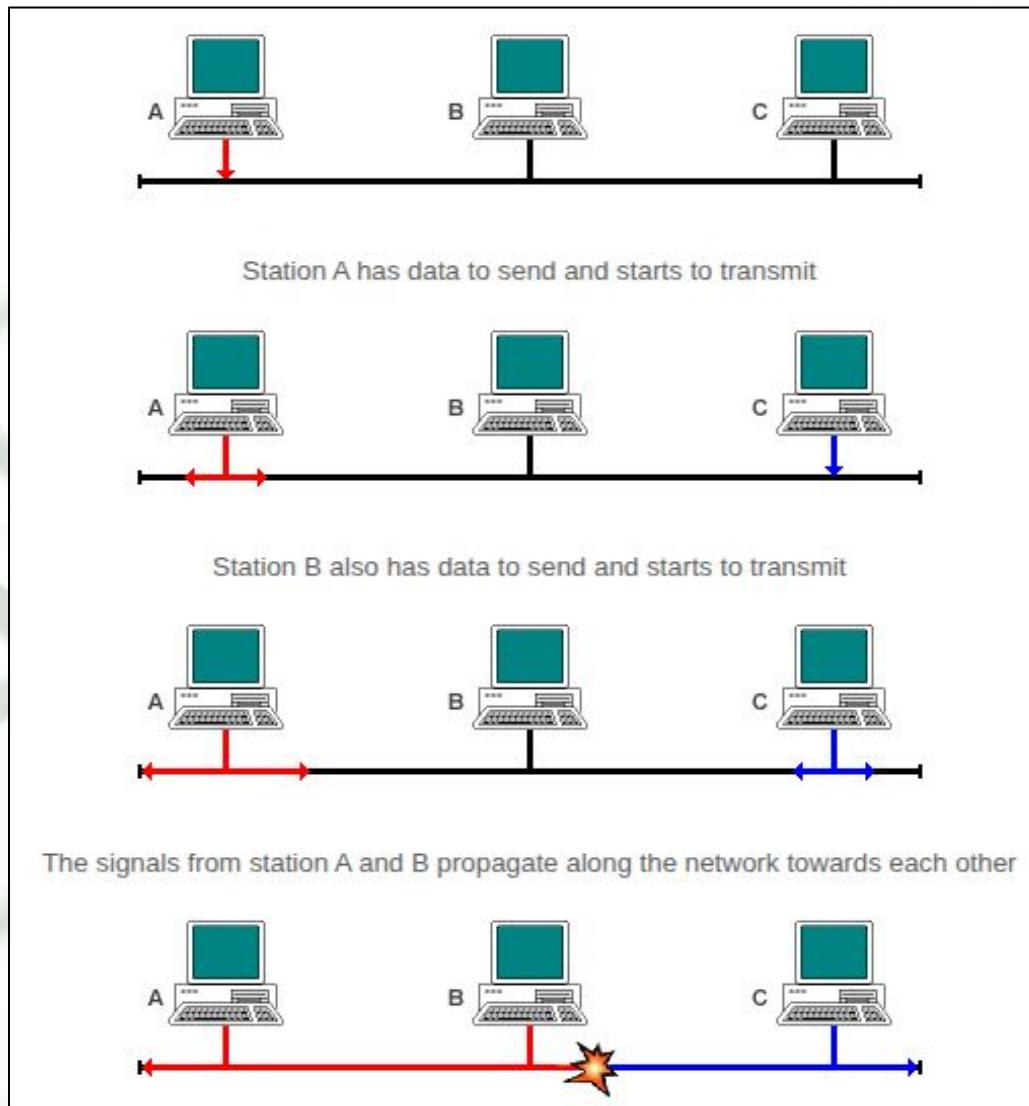
O que aconteceria se...



# Meios Compartilhados

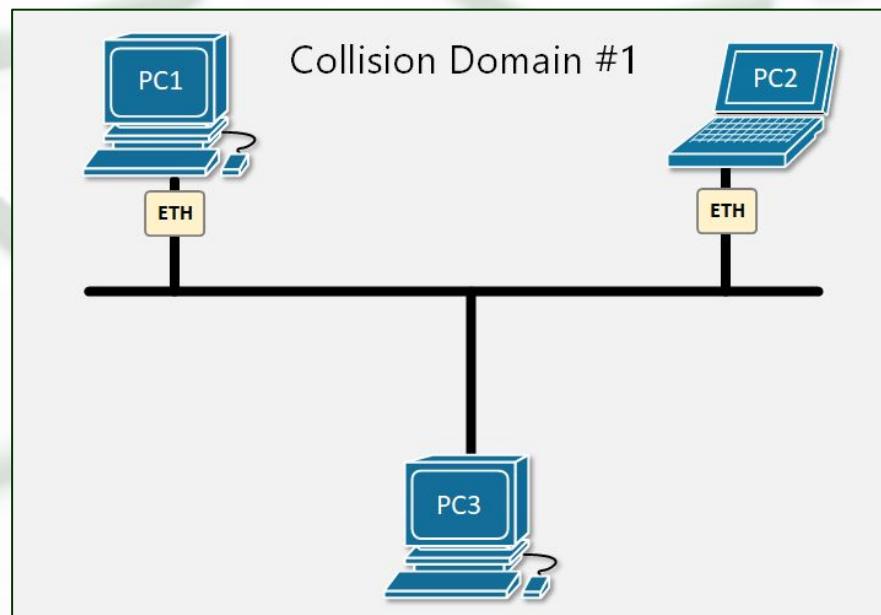


# Colisão de Dados



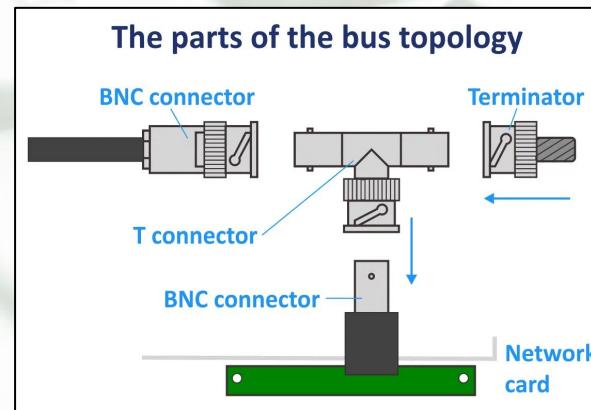
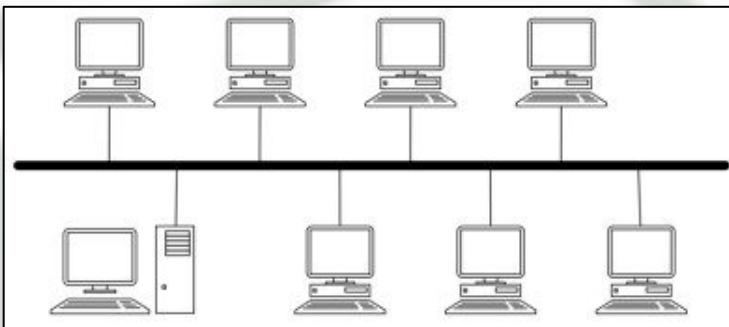
# Domínio de Colisão

- **Domínio de Colisão:** área lógica em uma rede de comunicação **na qual os sinais estão sujeitos à colisão** devido ao compartilhamento do meio.



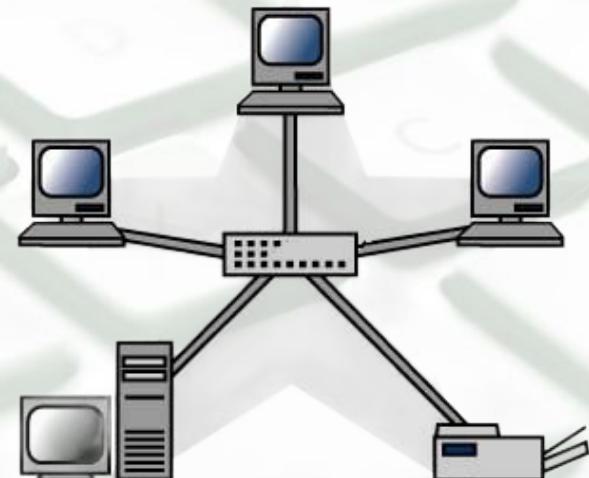
# Meios Compartilhados

- Nos primórdios das Redes de Computadores...
- **Topologia Barramento (Bus)**
  - Todos os dispositivos compartilham um único cabo metálico (geralmente coaxial). Era simples e barato, mas muito ineficiente devido ao **alto grau de perdas por colisão**, além da **dificuldade de expansão da rede**.

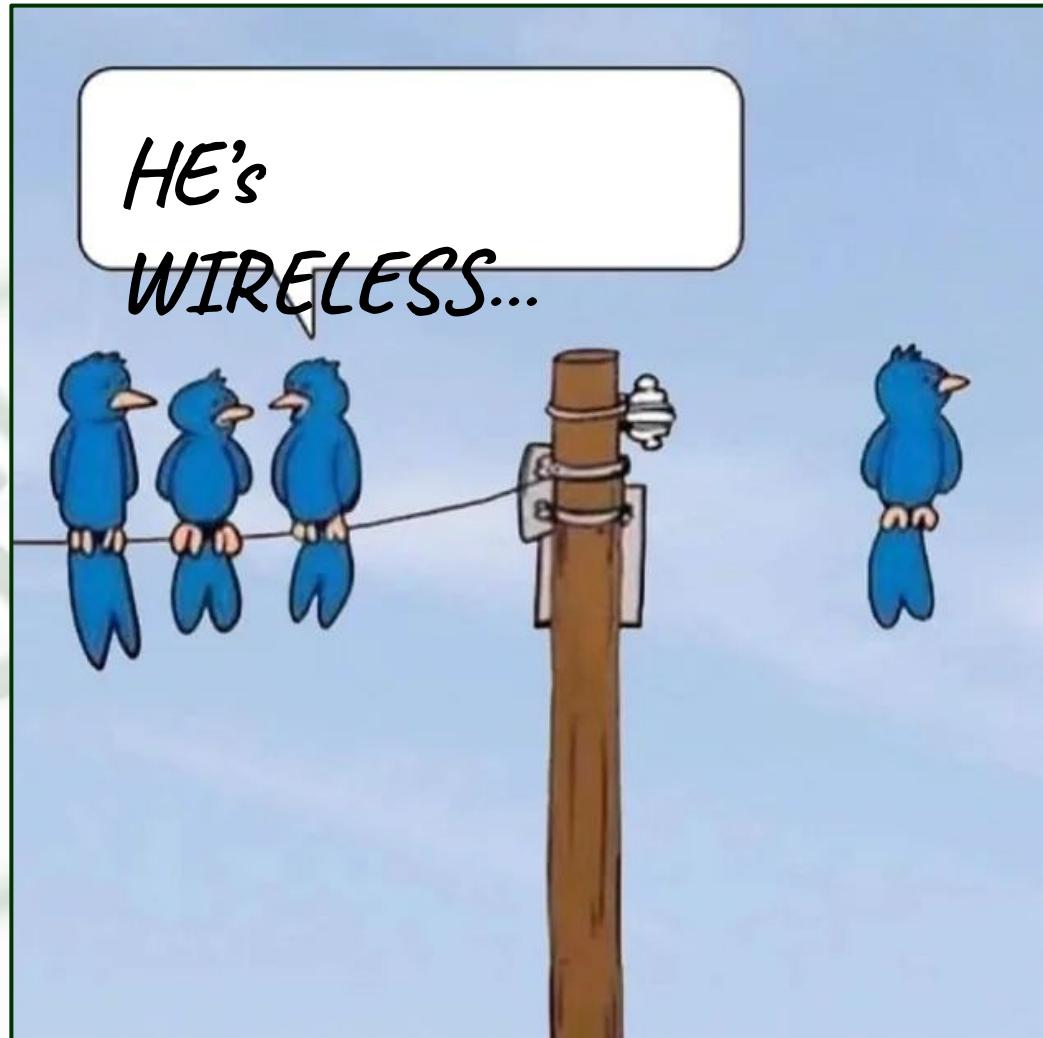


# Meios Compartilhados

- Para simplificar a expansão das redes, foi criada uma nova topologia...
- **Topologia Estrela (Star)**
  - Os dispositivos se conectam a um ponto central (geralmente **HUB** ou **SWITCH**).
  - Tornou-se mais fácil gerenciar e expandir redes, e falhas em um link agora não afetam a rede inteira.

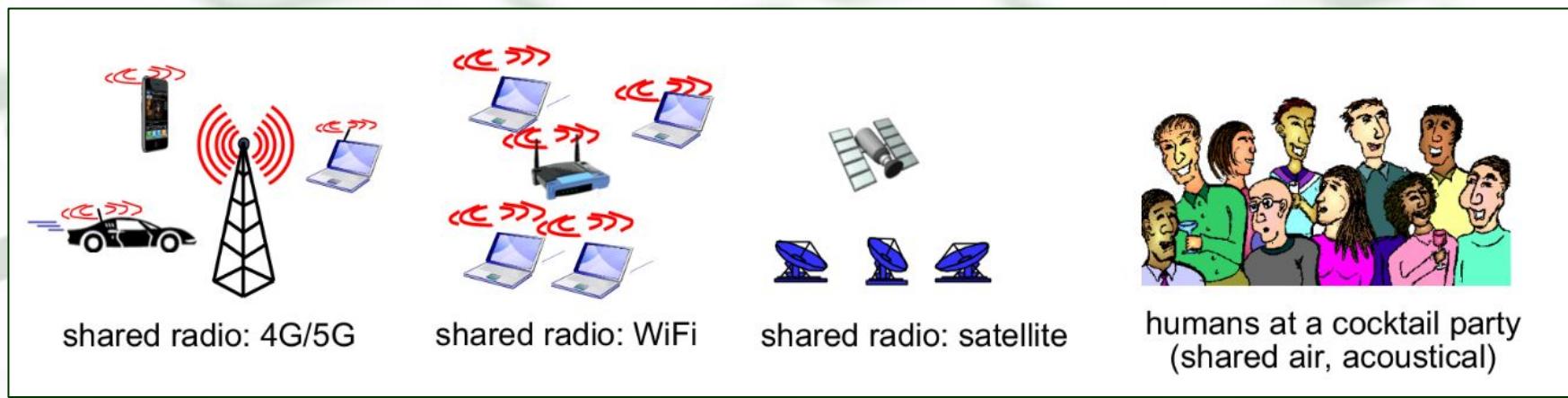


# Meios Compartilhados



# Meios Compartilhados

- Entretanto, o problema de **acesso ao meio** não diz respeito apenas a meios guiados, mas também é um problema natural e crítico em **meios wireless**.

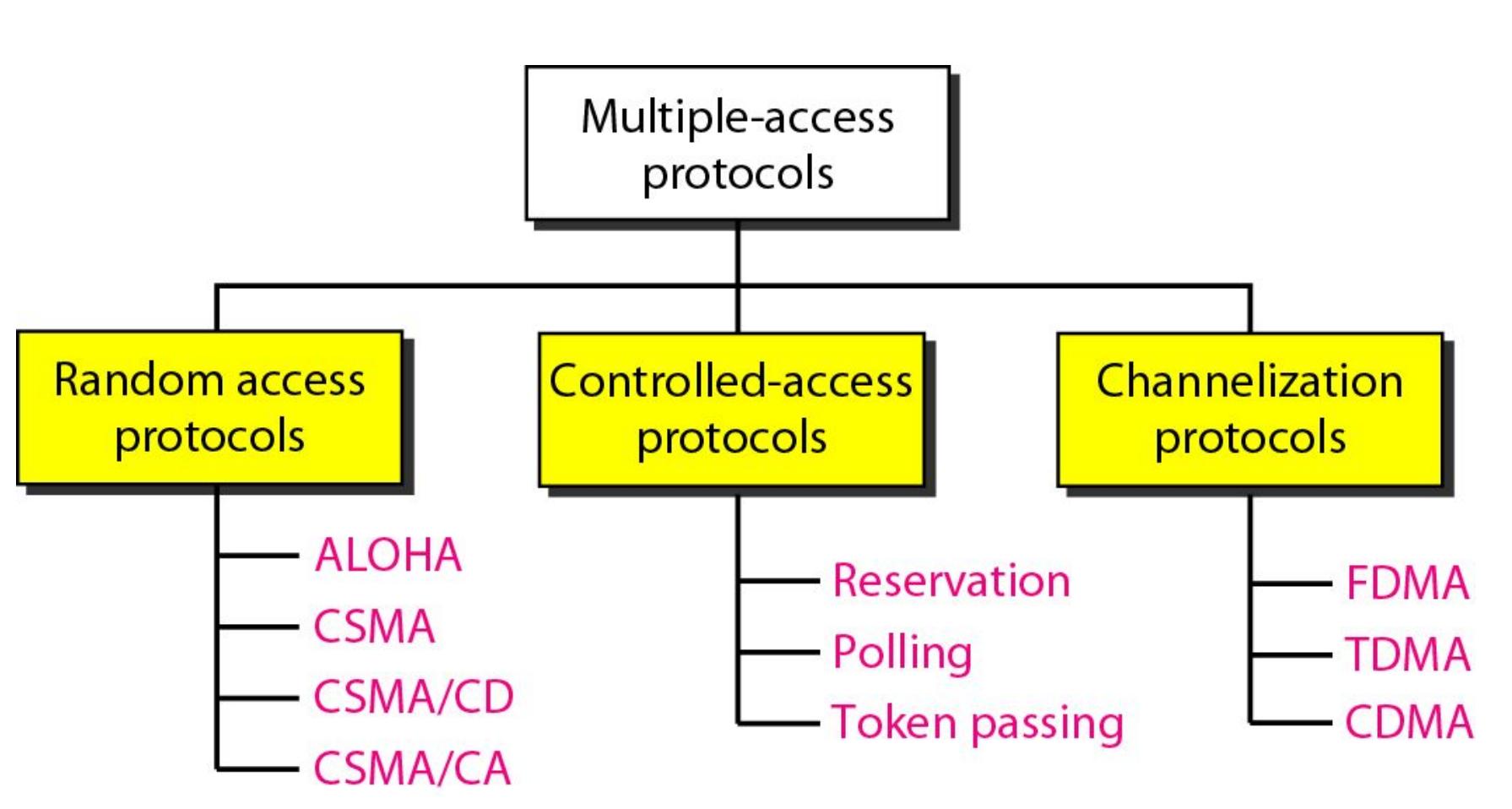


**Acesso eficiente** ao meio compartilhado é essencial para garantir **desempenho, justiça, evitar colisões** e permitir a **comunicação coordenada entre os dispositivos**.

# Acesso ao Meio

- Existem 03 classes de protocolos de acesso ao meio:
  - “Particionamento” do canal, também chamado de **Multiplexação**
  - Protocolos de **Revezamento**
  - Protocolos de **Acesso Aleatório**

# Acesso ao Meio

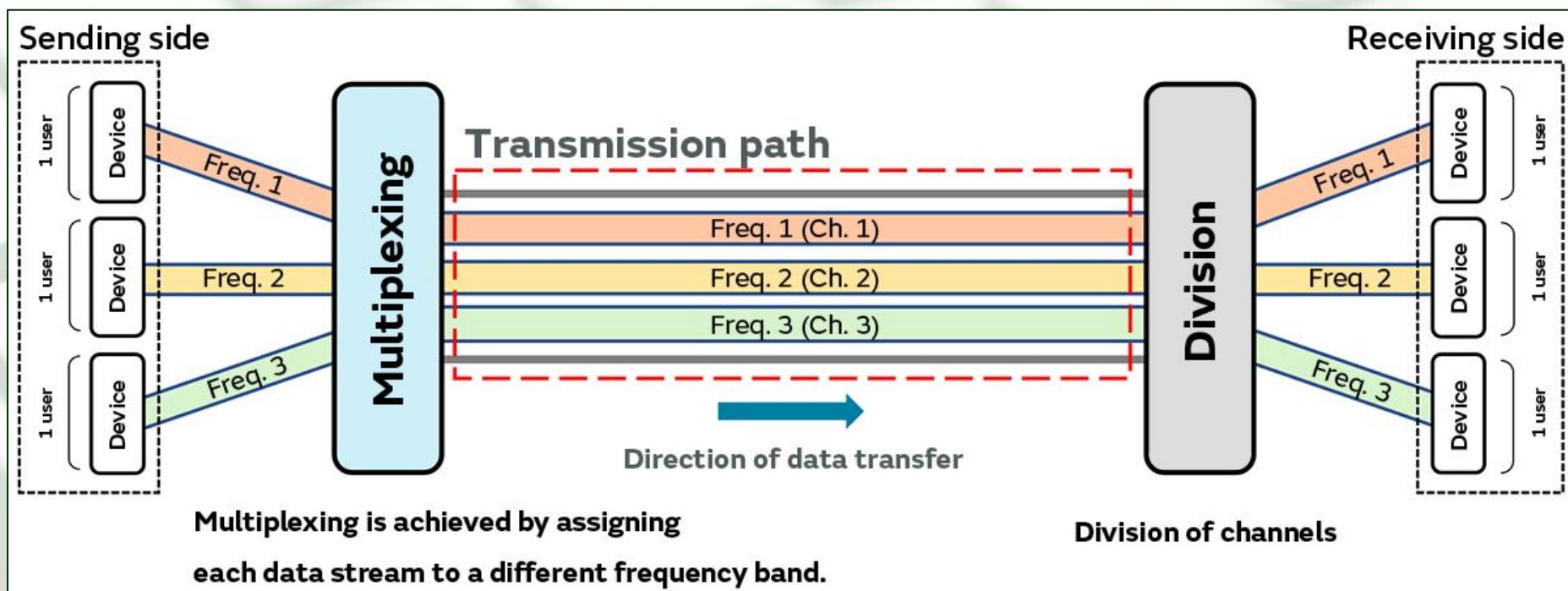


# Multiplexação

- Uma forma de alocar vários usuários concorrentes em um único canal é segmentando o link em diversos **Canais Determinísticos**, através de **Multiplexação**.
- Um **Multiplexador (MUX)** é um dispositivo que combina múltiplas entradas num único canal de saída.
- O **Demultiplexador (DEMUX)** é o dispositivo responsável pela operação inversa.

# Multiplexação FDMA

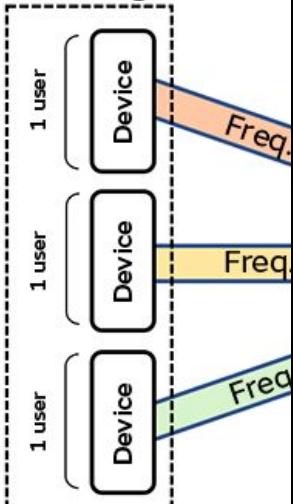
- **FDMA (Frequency Division Multiple Access):**  
Sub-canais utilizando faixa de freqüências distintas.



## ■ FDMA

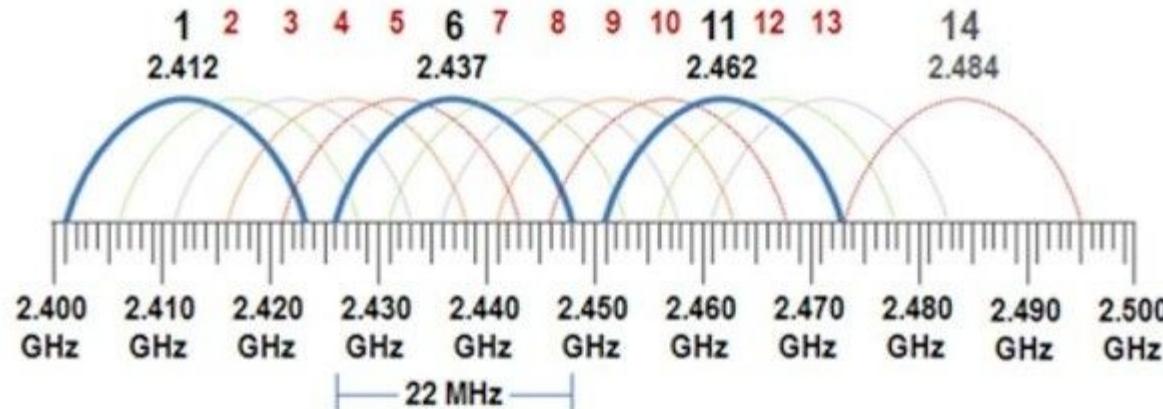
### Sub-carriers

Sending side

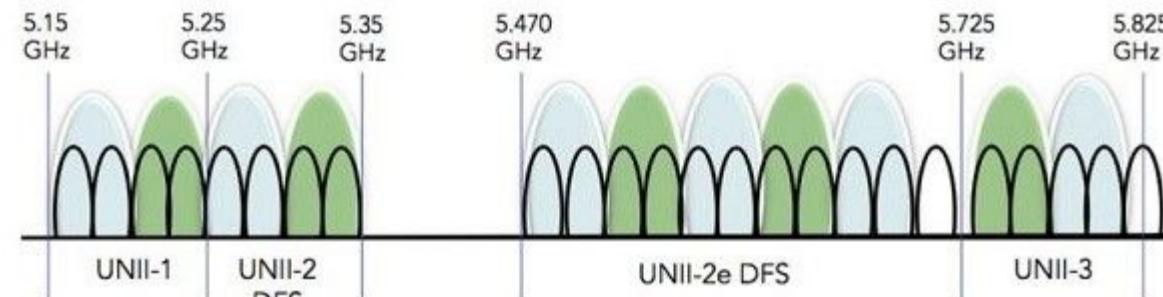


Mult. each

### 2.4 GHz Bands



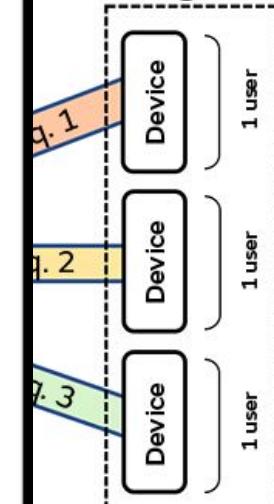
### 5 GHz Bands



### Wi-Fi Channels

intas.

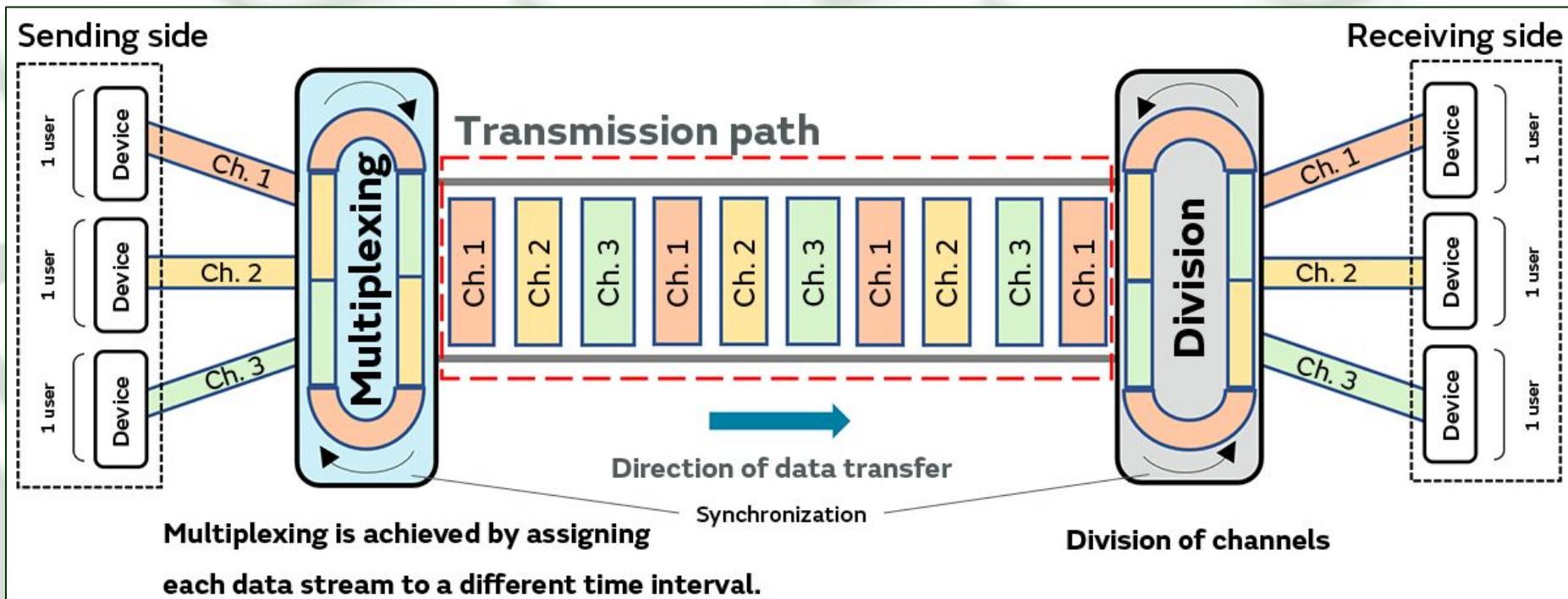
Receiving side



1 user  
1 user  
1 user

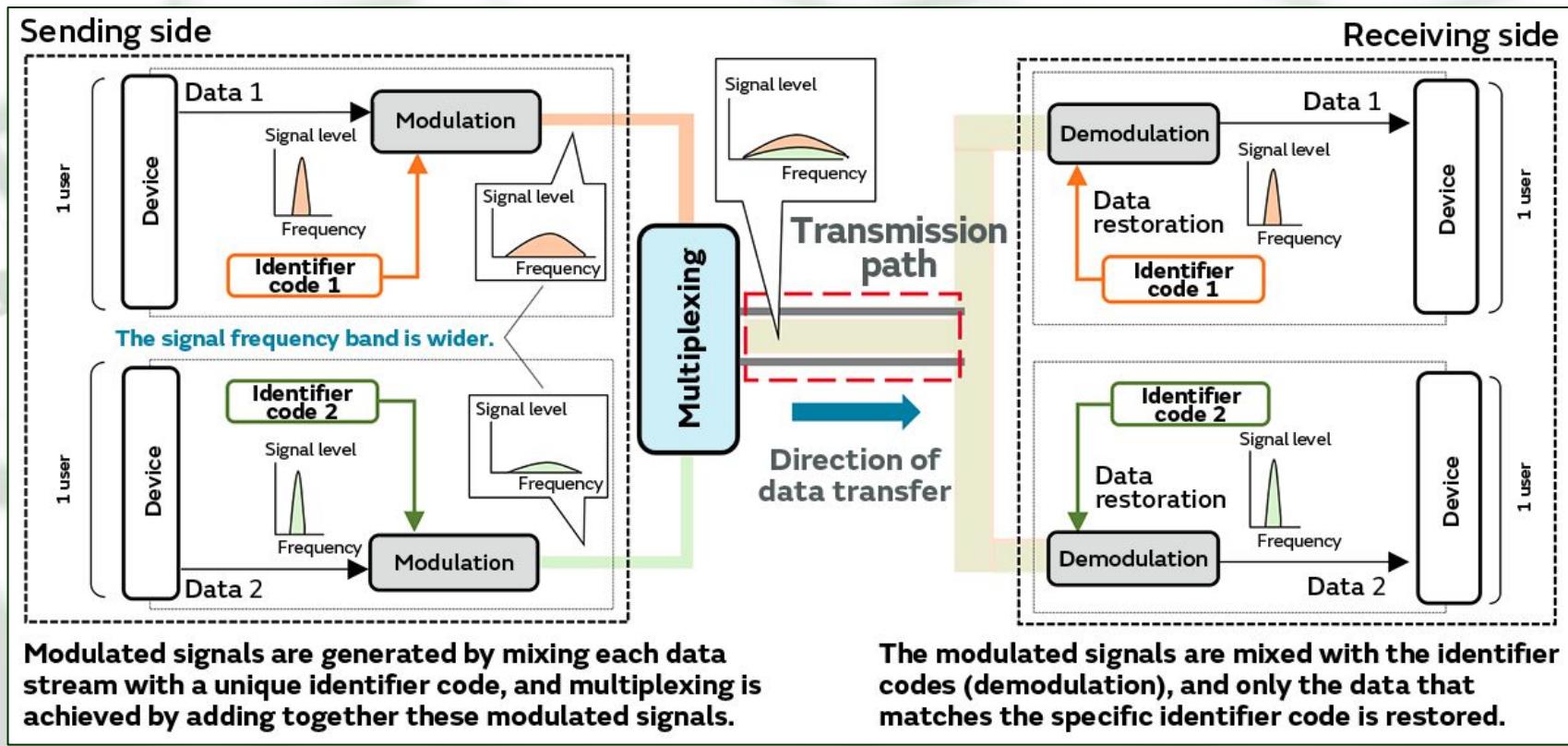
# Multiplexação TDMA

- TDMA (*Time Division Multiple Access*):  
Sub-canais utilizando *slots* de tempo fixo.



# Multiplexação CDMA

- **CDMA (Code Division Multiple Access):**  
Combinação de sinais utilizando codificações distintas.

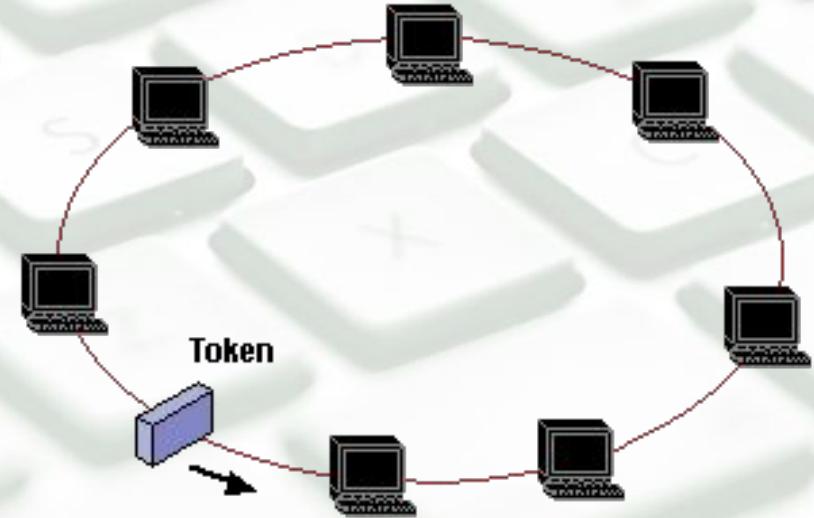


# Multiplexação

- Entretanto, quando o número de usuários é altamente variável (**Redes LAN** por exemplo), **a técnica de multiplexação não se torna viável**, uma vez que grande parte do espectro ou tempo será desperdiçado.
- Na multiplexação, após definido método de divisão do link entre os transmissores, nenhum outro transmissor adicional conseguirá ter acesso ao meio - daí o nome **Alocação Determinística**.

# Revezamento

- Outra classe de Protocolo de Acesso ao Meio é a de **Revezamento**.
- **Protocolo Token Ring (Topologia em Anel)**
  - Padrão IEEE 802.5
- Desenvolvido pela IBM.
- Ineficiente e Problemático.
- **Atualmente é obsoleto.**



# Revezamento

- Outra classe de Protocolo de Acesso ao Meio é a de **Revezamento**.
- **Protocolos de Revezamento**
  - Padrão de mercado
- Desenvolvidos nos anos 70.
- Ineficiente e Problemático.
- **Atualmente é obsoleto.**

**Mas qual tecnologia  
é padrão de mercado  
atualmente?**



- IEEE (i3e)
  - *Institute of Electrical and Electronics Engineers*
- Em **Fev/1980** (802), o IEEE, em conjunto com Indústrias da área, Universidades e Pesquisadores promoveram um grande esforço cujo objetivo era definir **padrões para as redes de computadores**, focando principalmente nas responsabilidades **das camadas física e de enlace**.

# IEEE 802

- Nascia o projeto IEEE 802, que buscava definir **padrões de tecnologias abertas e protocolos confiáveis** e assim garantir a **interoperabilidade** entre equipamentos de **diferentes fabricantes**.
- Foram criados subcomitês (***Working Groups***) para pesquisas e definição destes padrões, cada qual focado em determinadas tecnologias/protocolos.

Alguns “morreram pelo caminho”,  
outros são **padrões de mercado até hoje!**

# IEEE 802

<i>OSI</i>	<i>TCP/IP</i>	<i>IEEE</i>			
Enlace	Interface de Rede	<b>802.2 (LLC)</b>			
Físico		802.3	802.4	802.5	802.11

**802.3 Ethernet (LANs)**

~~802.4 Token Bus~~

~~802.5 Token Ring~~

**802.11 Wi-Fi (WLANs)**

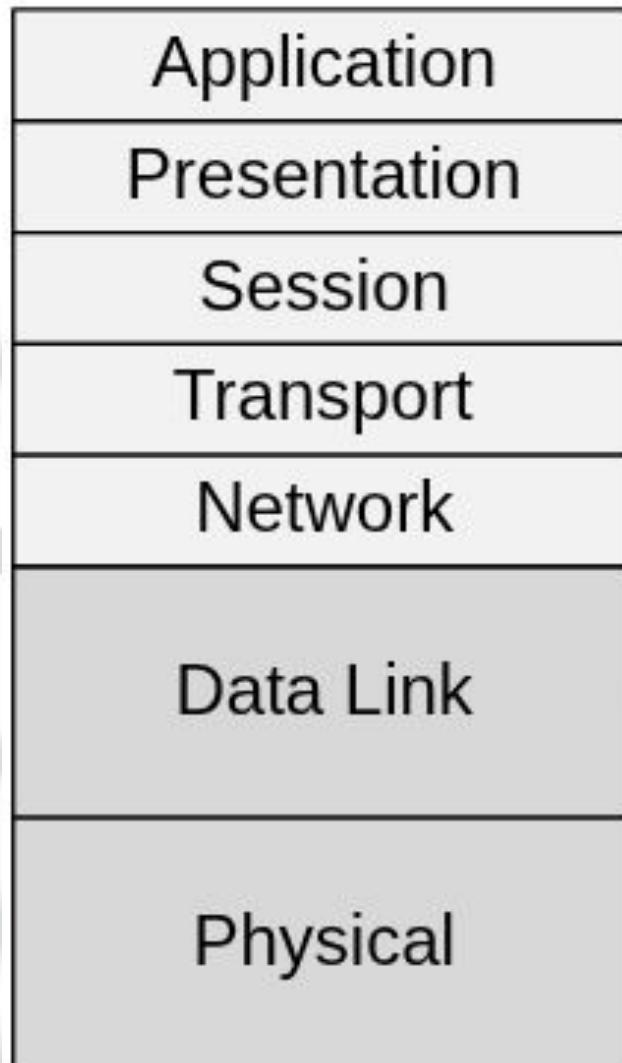
**802.15 Bluetooth (WPANs)**

~~802.16 WiMax~~

Número	Assunto
802.1	Avaliação e arquitetura de LANs
802.2	Controle de enlace lógico
802.3 *	Ethernet
802.4 †	Token bus (barramento de tokens; foi usado por algum tempo em unidades industriais)
802.5 †	Token ring (anel de tokens; a entrada da IBM no mundo das LANs)
802.6 †	Fila dual barramento dual (primeira rede metropolitana)
802.7 †	Grupo técnico consultivo sobre tecnologias de banda larga
802.8 †	Grupo técnico consultivo sobre tecnologias de fibra óptica
802.9 †	LANs isócronas (para aplicações em tempo real)
802.10 †	LANs virtuais e segurança
802.11 *	LANs sem fio (WiFi)
802.12 †	Prioridade de demanda (AnyLAN da Hewlett-Packard)
802.13	Número relacionado à má sorte. Ninguém o quis
802.14 †	Modems a cabo (extinto: um consórcio industrial conseguiu chegar primeiro)
802.15 *	Redes pessoais (Bluetooth, Zigbee)
802.16 †	Banda larga sem fio (WiMAX)
802.17 †	Anel de pacote resiliente
802.18	Grupo técnico consultivo sobre questões de regulamentação de rádio
802.19	Grupo técnico consultivo sobre coexistência de todos esses padrões
802.20	Banda larga móvel sem fio (semelhante ao 802.16e)
802.21	Transferência independente do meio (para tecnologias de roaming)
802.22	Rede regional sem fio

**Figura 1.37** Os grupos de trabalho 802. Os grupos importantes estão marcados com \*. Aqueles marcados com † desistiram e foram dissolvidos.

# IEEE 802



**Padrão (e sub-padrões)**  
**IEEE 802.3 => Ethernet**

**Logical Link Control (LLC)**  
802.2 Standards

**Media Access Control (MAC)**  
802.3 standards

IEEE 802.3 Ethernet  
IEEE 802.3u FastEthernet  
IEEE 802.3z GigabitEthernet  
IEEE 802.3ae TenGigabitEthernet  
IEEE 802.3af Power over Ethernet

Ethernet

# IEEE 802

- Um grande trunfo dos padrões estabelecidos para as Redes Locais (LANs): Ethernet e Wi-Fi, foi a adoção de **uma mesma classe de protocolo de acesso ao meio**, conhecido por **CSMA (*Carrier Sense Multiple Access*)**, com pequenas variações entre si.
- IEEE 802.3 => Ethernet => CSMA/CD
- IEEE 802.11 => Wi-Fi => CSMA/CA

*Isso facilitou a padronização, interoperabilidade e a disseminação dessas tecnologias.*

# *Carrier Sense Multiple Access*

**A estratégia de acesso ao meio do CSMA  
é bastante simplória, mas eficiente...**

- Escute o canal para ver se está ocupado...
  - Caso esteja ocupado, aguarde...
  - Caso esteja ocioso, transmita a informação...

**Essa técnica elimina a ocorrência de colisões?**

# Carrier Sense Multiple Access

- **Não...** O algoritmo por si só não elimina as colisões  
**MAS...**

- *É possível aprimorá-lo com um truque interessante...*

1. E se fosse possível detectar uma colisão em tempo real?
2. E, detectando-se uma colisão, já admitir que a transmissão falhou...
3. E se a transmissão falhou, fazer uma nova tentativa de transmissão em um tempo brevemente aleatório...

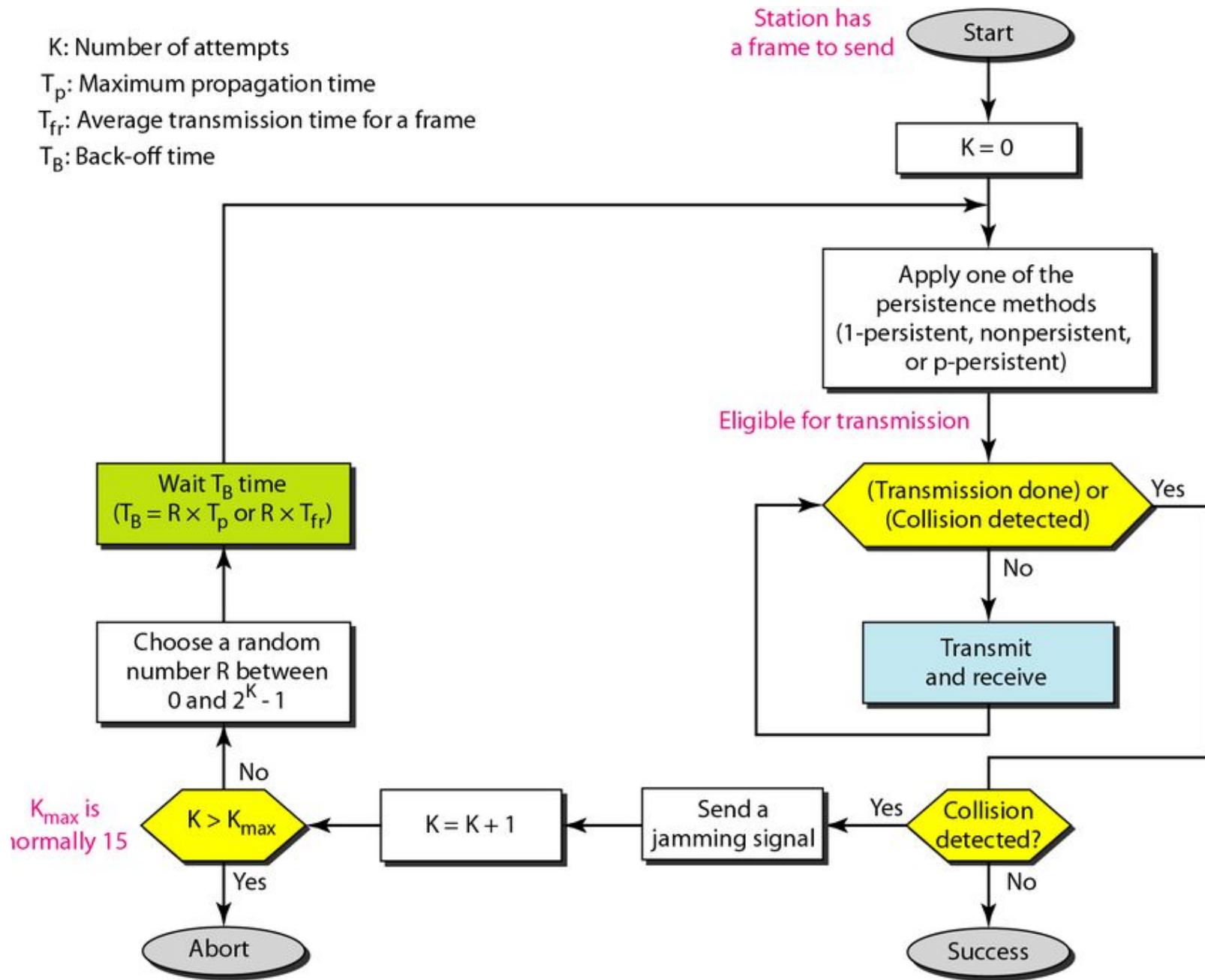
**CSMA/CD => Carrier Sense Multiple Access with Collision Detection**

$K$ : Number of attempts

$T_p$ : Maximum propagation time

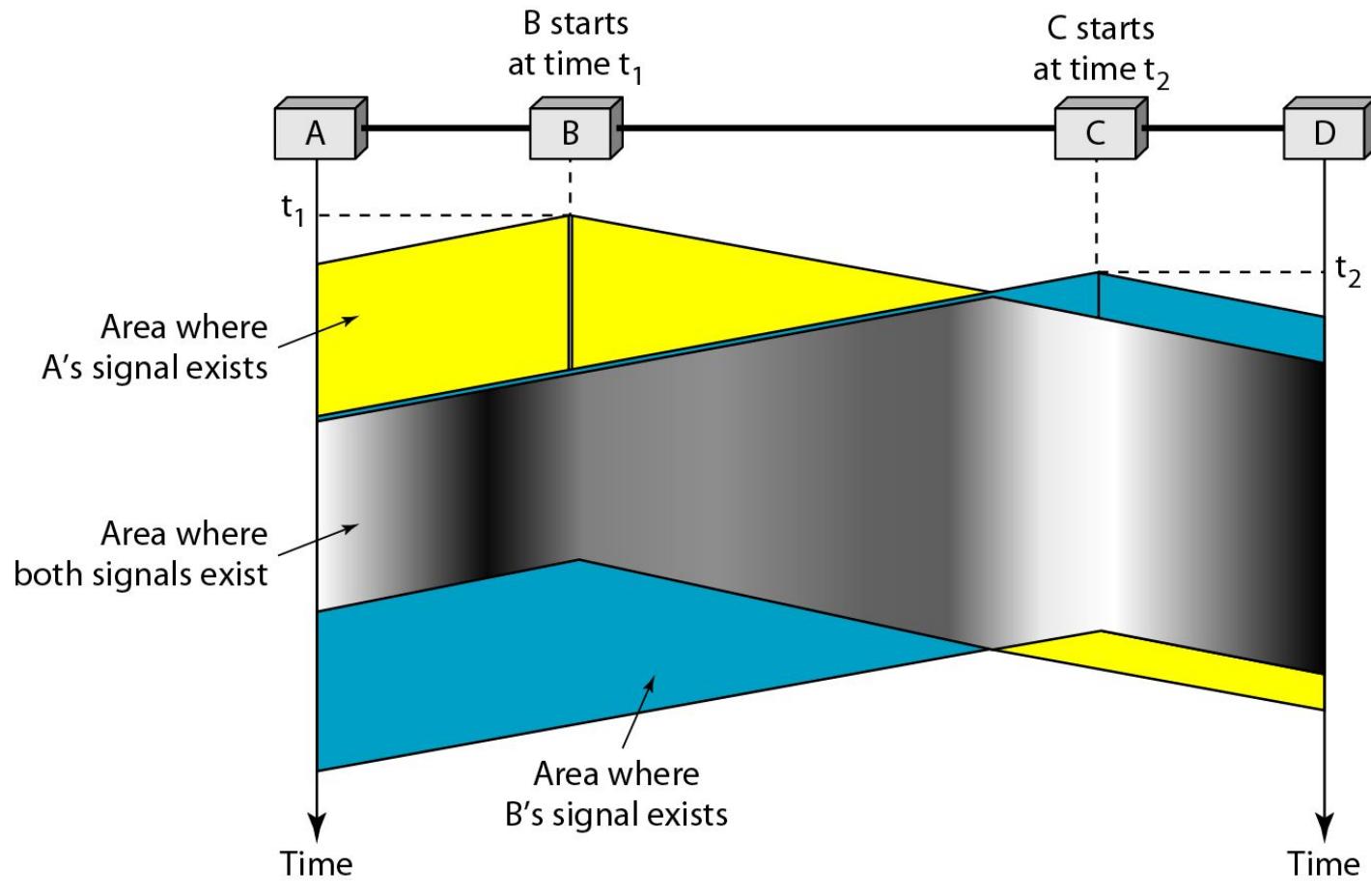
$T_{fr}$ : Average transmission time for a frame

$T_B$ : Back-off time



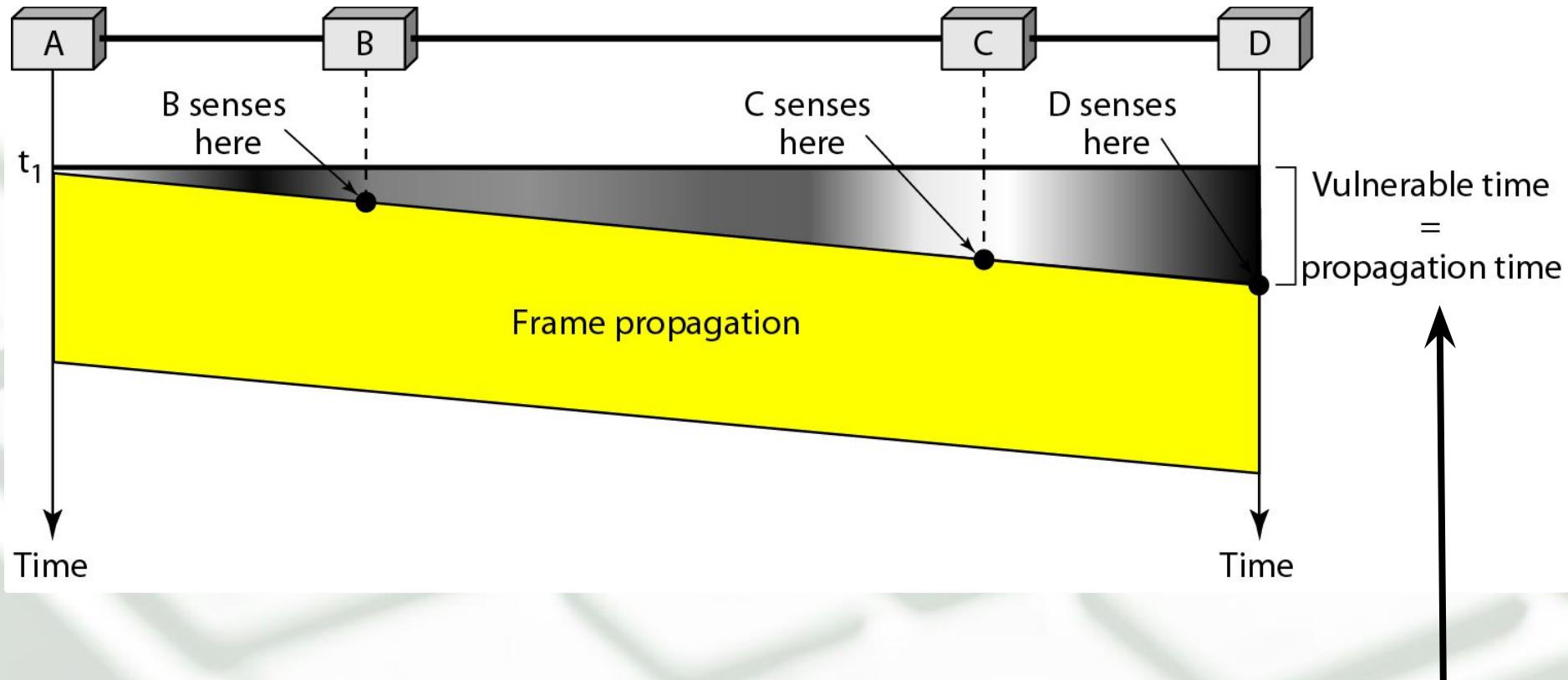
# Carrier Sense Multiple Access

## ■ Mas como DETECTAR UMA COLISÃO?



# Carrier Sense Multiple Access

## ■ Mas como DETECTAR UMA COLISÃO?



# Carrier Sense Multiple Access

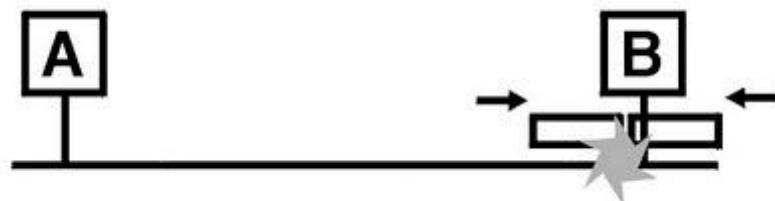
A escuta o meio livre e inicia transmissão...



A encerra transmissão e o frame se propaga pelo meio...



B escuta o meio livre e inicia transmissão. Há colisão no meio.

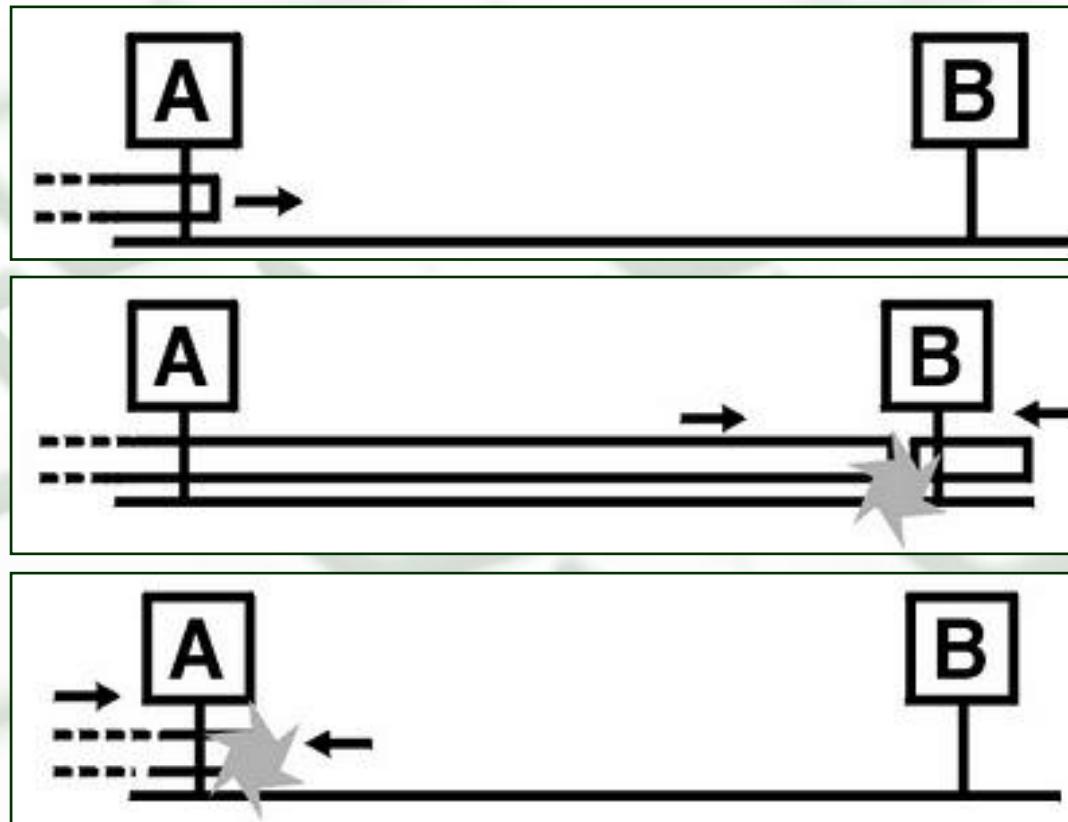


A percebe que houve uma colisão de dados.

**Mas ele precisa retransmitir?**



## ■ A solução...



# CSMA/CD

## ■ A solução...

Todo frame tem que ser grande o suficiente para que o **tempo de sua propagação** dure, no mínimo, o equivalente ao tempo que o sinal gasta para **ir e voltar (RTT)** entre os dois pontos mais distantes do domínio de colisão.

## ■ RTT = ***Round-Trip Time*** (Tempo de Ida e Volta)

- Em outras palavras...

$$\frac{\text{Tempo Propagação}}{\text{Tempo Transmissão}} < 0,5$$

*Se o meio demora 1s para propagar o sinal,  
a transmissão deve durar no mínimo 2s.*

# Exercícios

- Em que o tamanho da janela em um protocolo de sliding window influência no throughput de uma rede?
- **O que é Multiplexação? Qual a finalidade?**
- Quando a Multiplexação não é adequada? Explique Porque!
- **Porque a Multiplexação é dita uma técnica de alocação estática do Meio de Transmissão?**
- Diferencie as técnicas de multiplexação.
- **Dê exemplos de protocolos de acesso ao meio pelo método de acesso aleatório, método por revezamento e método por divisão do canal.**
- O protocolo CSMA é mais eficiente para redes de grande ou pequena extensão? Explique!
- **O que é o IEEE 802?**
- Qual o nome técnico, e qual a utilidade do endereço físico de interface de rede?
- **Qual é o tamanho, e forma de representação do endereçamento em nível de Enlace?**