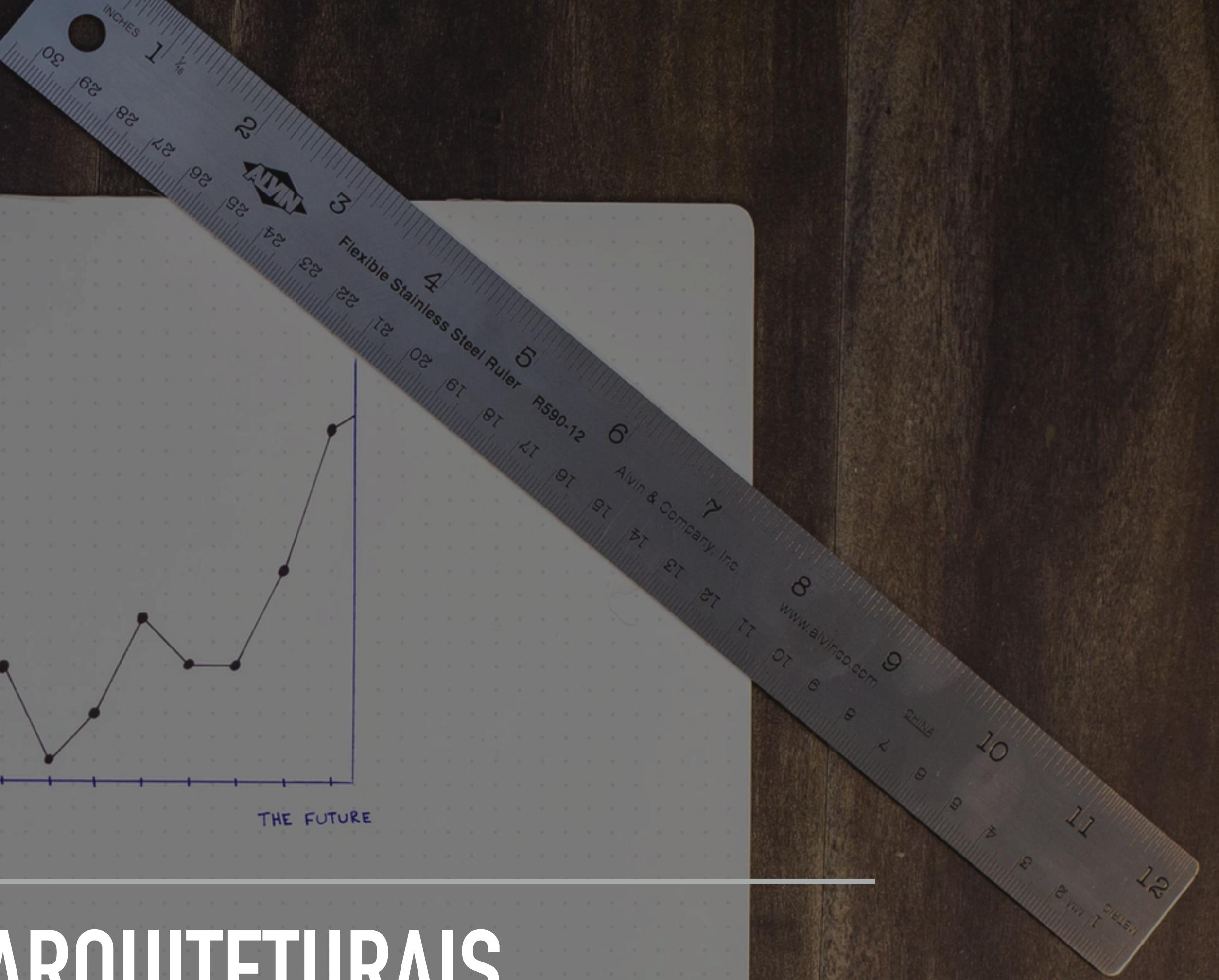
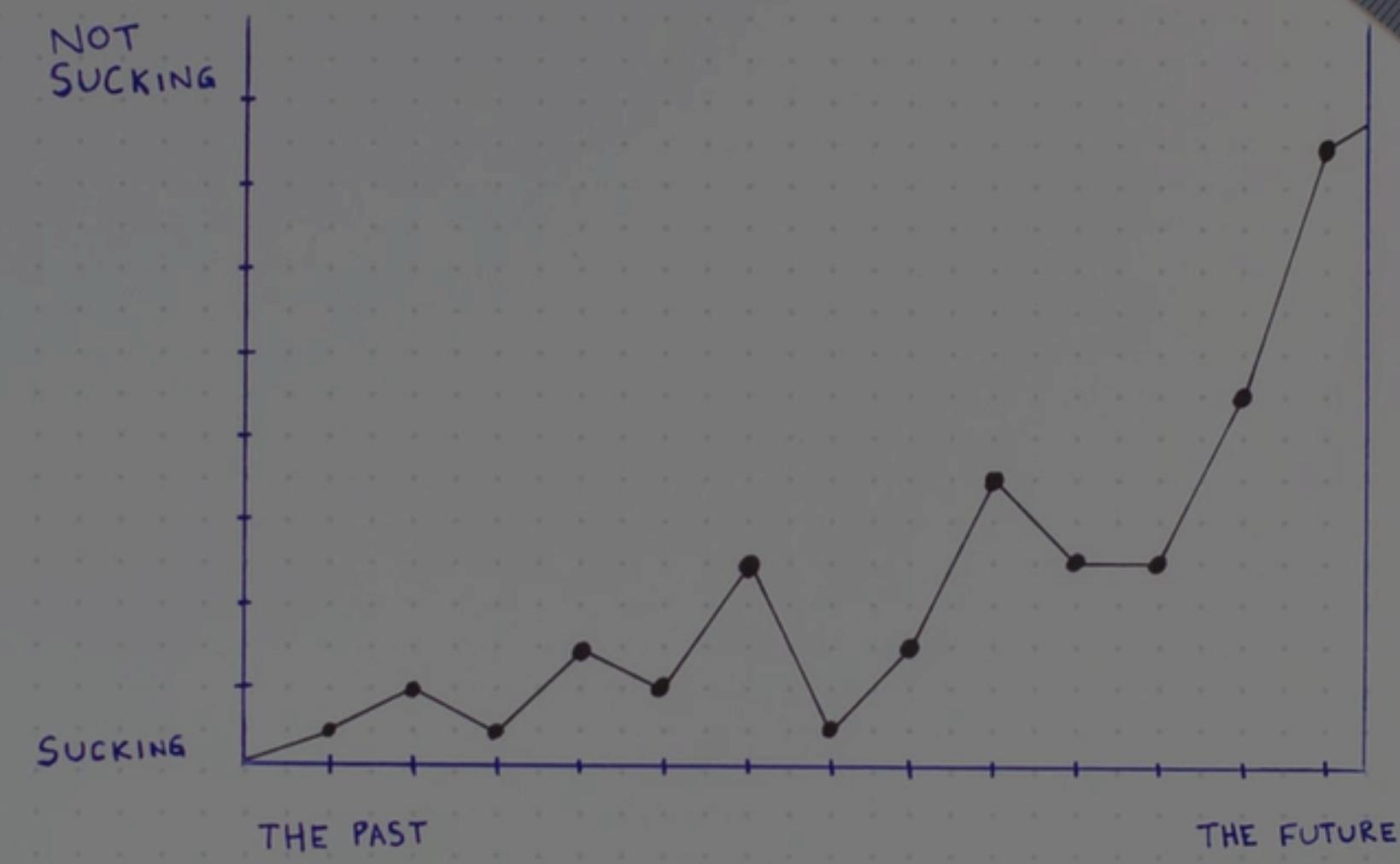


IMD0043

PRINCÍPIOS E OBJETIVOS ARQUITETURAIS



REVISANDO: SEPARAÇÃO DAS PREOCUPAÇÕES

Rede:

- ▶ entrega de pacotes *host-a-host*

Pilha de rede (SO):

- ▶ entrega de pacotes para o socket apropriado (baseado na porta)

Aplicações:

- ▶ enviam e recebem pacotes
- ▶ entendem o conteúdo da carga útil (corpo) dos pacotes

Por que a separação de **preocupações** é importante?

modularidade

MODULARIDADE

- ▶ Se a tarefa de cada componente é bem definida, é possível focar o esforço nesta tarefa
- ▶ E substituir esse componente por qualquer outra implementação que realize essa mesma tarefa,**sem mudar nada mais!**
- ▶ Modularizar é **decompor** programas/sistemas em unidades menores
 - ▶ Uma clara "separação de preocupações"
 - ▶ Essencial na Ciência da Computação, incluindo redes!

MODULARITY BASED ON
ABSTRACTION IS THE WAY TO
GET THINGS DONE

Barbara Liskov

MODULARIDADE EM SISTEMAS COMPUTACIONAIS

Particionando sistemas em módulos

- ▶ Cada módulo tem sua interface bem definida

Interfaces fornecem flexibilidade na implementação

- ▶ Mudanças tem escopo limitado

Exemplos

- ▶ Bibliotecas encapsulam um conjunto de funcionalidades
- ▶ Linguagens de programação abstraem a CPU

O truque é encontrar a modularidade certa

- ▶ Interfaces devem ser duradouras
- ▶ Se as interfaces estão mudando constantemente, a modularidade está incorreta

MODULARIDADE EM REDES DE COMPUTADORES

A necessidade por modularização ainda se aplica

- ▶ E é até mais importante!

Implementações de rede não estão limitadas a linhas de código

- ▶ Modularidade em programas organizam esse código

Implementações de rede está distribuído e em várias máquinas

- ▶ Hosts
- ▶ Equipamentos de rede
- ▶ etc.

PERGUNTAS

Como quebrar o sistema em módulos?

- ▶ Decomposição clássica em tarefas

Onde os módulos são implementados?

- ▶ Hosts? Roteadores? Ambos?

DECISÕES DE MODULARIDADE NAS REDES DE COMPUTADORES

Como quebrar o sistema em módulos?

- ▶ Modelo em camadas (*layering*)

Onde os módulos são implementados?

- ▶ Princípio fim-a-fim (*end-to-end*)

DECISÕES DE MODULARIDADE NAS REDES DE COMPUTADORES

Como quebrar o sistema em módulos?

- ▶ **Modelo em camadas (*layering*)**

Onde os módulos são implementados?

- ▶ Princípio fim-a-fim (end-to-end)

LAYERING: INSPIRAÇÃO

CEO A escreve carta para o CEO B

- ▶ Dobra a carta e entrega para seu secretário administrativo

Secretário administrativo

- ▶ Coloca a carta em um envelope, com o nome completo do CEO B
- ▶ Leva para os Correios

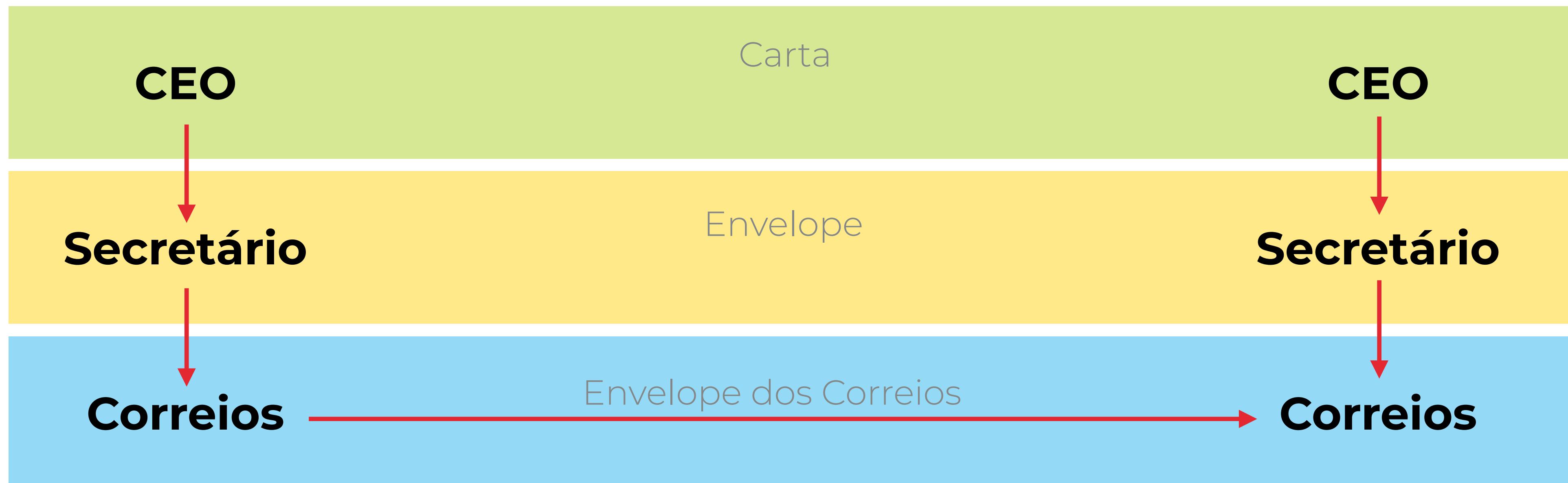
Escritório dos Correios

- ▶ Coloca a carta em um envelope maior
- ▶ Coloca o nome e o endereço no envelope dos Correios
- ▶ Coloca o pacote no caminhão de entrega dos Correios

Correios entrega para a empresa de destino

O CAMINHO DA CARTA

- ▶ Pares entendem as mesma coisas!
- ▶ Ninguém mais precisa entender
- ▶ O nível mais baixo tem o maior “pacote”



LAYERING: QUEBRANDO EM TAREFAS

1. Transmissão de *bits* em um meio
2. Entrega de pacotes para os *hosts* através de uma rede local
3. Entrega de pacotes para os *hosts* através de redes diferentes
4. Entrega confiável (?) de pacotes
5. Fazer algo com os dados

LAYERING: QUEBRANDO EM TAREFAS

1. Transmissão de *bits* em um meio (física)
2. Entrega de pacotes para os hosts através de uma rede local (enlace)
3. Entrega de pacotes para os hosts através de redes diferentes (rede)
4. Entrega confiável (?) de pacotes (transporte)
5. Fazer algo com os dados (aplicação)

CINCO CAMADAS DA INTERNET (TOP-DOWN)

APLICAÇÕES

ENTREGA CONFIÁVEL (OU NÃO CONFIÁVEL) DE DADOS

ENTREGA BEST-EFFORT DE PACOTES GLOBAL

ENTREGA BEST-EFFORT DE PACOTES LOCAL

TRANSFERÊNCIA FÍSICA DE BITS

CINCO CAMADAS DA INTERNET (TOP-DOWN)

CAMADA DE APLICAÇÃO

CAMADA DE TRANSPORTE

CAMADA DE REDE

CAMADA DE ENLACE

CAMADA FÍSICA

CINCO CAMADAS DA INTERNET (TOP-DOWN)

CAMADA DE APLICAÇÃO

construída sobre...

CAMADA DE TRANSPORTE

construída sobre...

CAMADA DE REDE

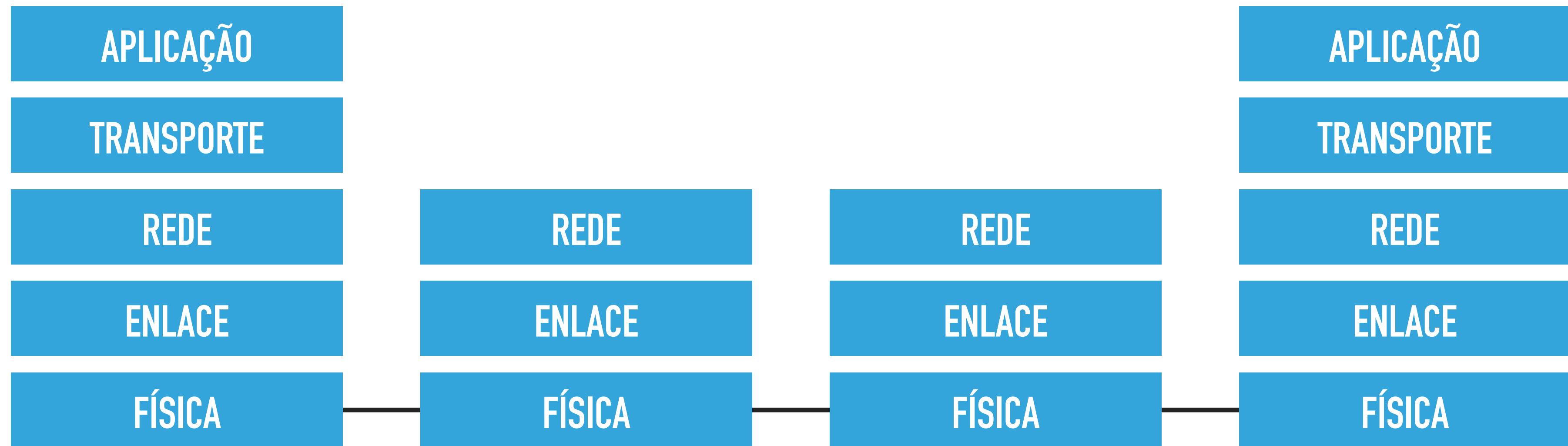
construída sobre...

CAMADA DE ENLACE

construída sobre...

CAMADA FÍSICA

LAYERING



- ▶ Funcionalidades separadas em **camadas**
- ▶ Camada N faz interface somente com a camada $N-1$ e $N+1$
- ▶ Isola complexidade das camadas adjacentes
- ▶ Módulos podem evoluir independentemente

CINCO CAMADAS DA INTERNET (TOP-DOWN)



UM POUCO DE HISTÓRIA... O MODELO DE REFERÊNCIA OSI



- Modelo Open System Interconnect (OSI) inclui duas camadas adicionais que são normalmente implementadas como parte da aplicação

LEMBRANDO: PARES ENTENDEM AS MESMAS COISAS

CEO

Carta

CEO

Secretário

Envelope

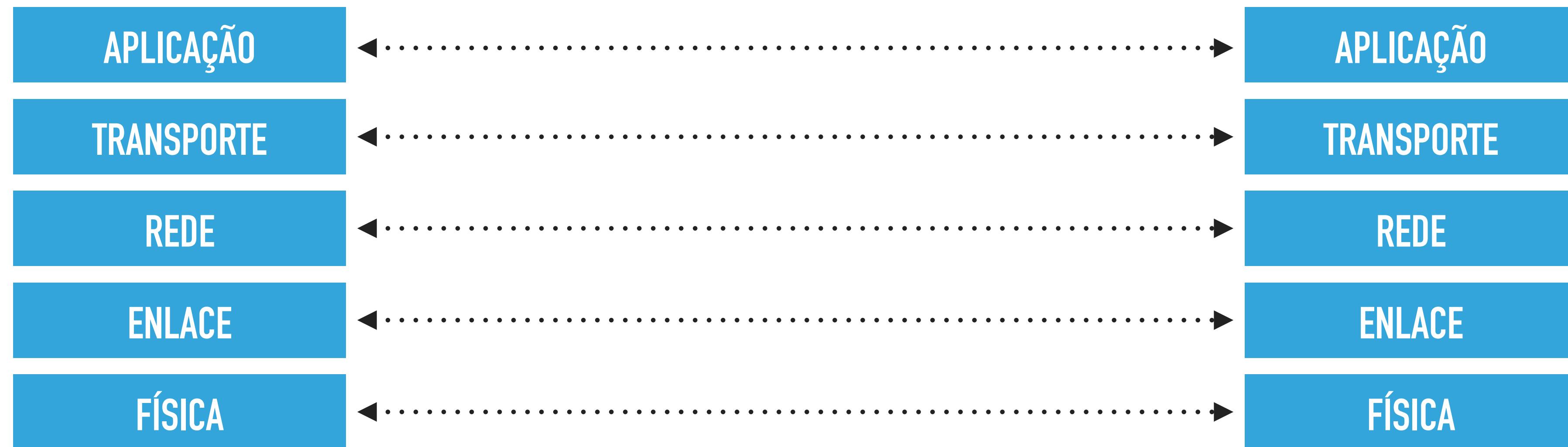
Secretário

Correios

Envelope dos Correios

Correios

CAMADAS E PROTOCOLOS

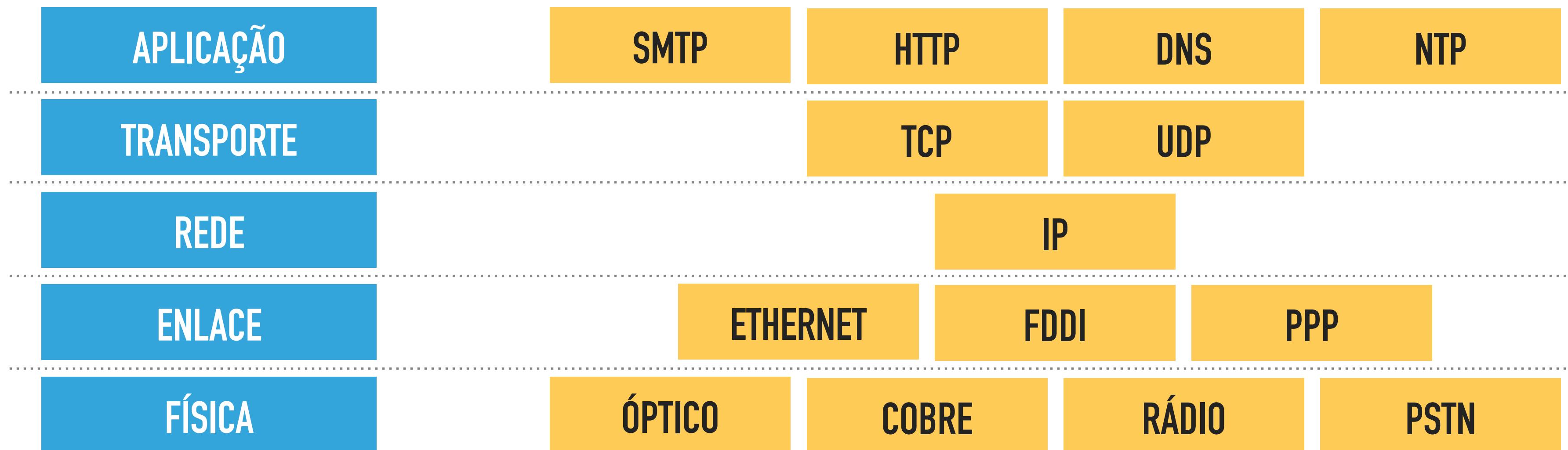


Comunicação entre mesmas camadas em diferentes sistemas é definida pelos **protocolos**

O QUE É UM PROTOCOLO?

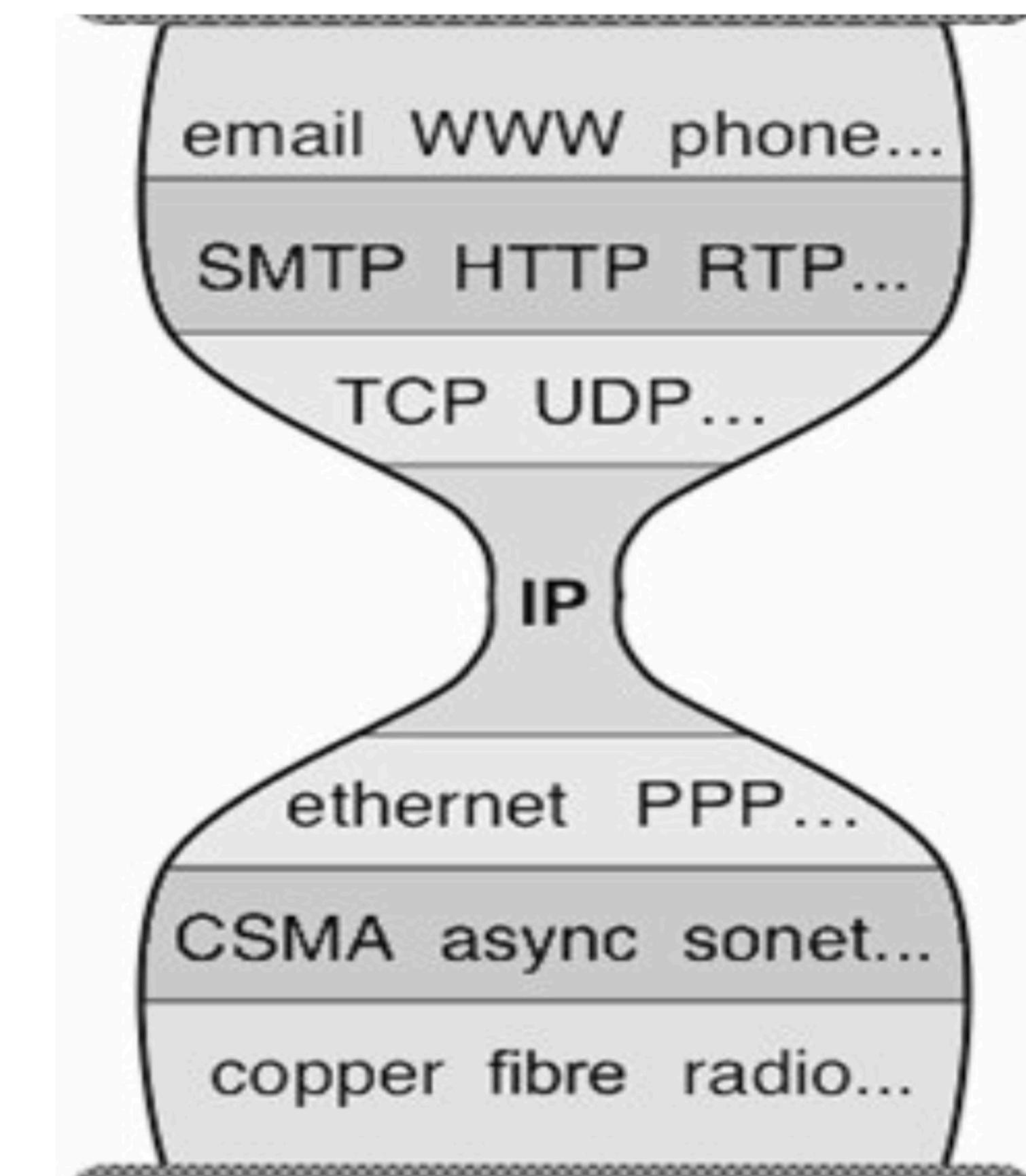
- ▶ Um **acordo** entre as partes que irão se comunicar
- ▶ Definem uma **sintaxe** de comunicação
 - ▶ Cada protocolo define um formato de cabeçalho
 - ▶ Ex: "os primeiros 32 bits irão armazenar o endereço de destino"
- ▶ Definem uma **semântica**
 - ▶ "primeiro um cumprimento, depois uma requisição"
 - ▶ Essencialmente, uma máquina de estados
- ▶ Protocolos existem em muitos níveis, *hardware* e *software*
 - ▶ Normalmente definidos por instituições de **normatização** (IETF, IEEE, ITU)

PROTOCOLOS EM DIFERENTES CAMADAS



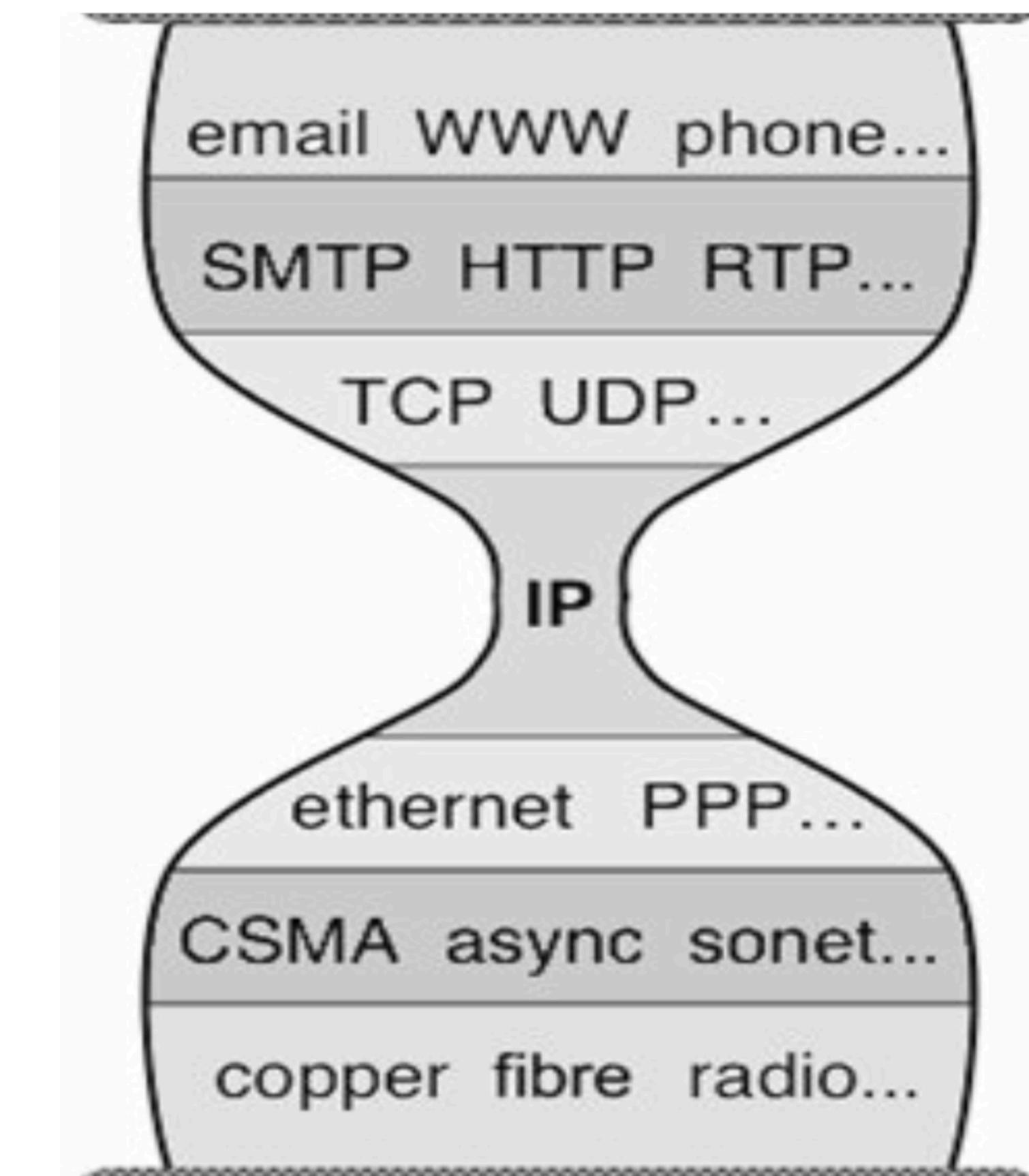
OBSERVAÇÕES

- ▶ Cada camada:
 - ▶ Depende da camada de baixo
 - ▶ Suporta a camada de cima
 - ▶ É independente das demais
- ▶ Múltiplas versões em uma camada
 - ▶ Componentes irão escolher qual protocolo de camada inferior utilizar
- ▶ Mas somente um protocolo de rede: **IP**
 - ▶ Protocolo unificador!



MODELO EM CAMADAS FOI CRUCIAL PARA O SUCESSO DA INTERNET

- ▶ Inovação na maioria dos níveis:
 - ▶ Aplicações (muitas)
 - ▶ Transporte (poucas)
 - ▶ Enlace (poucas)
 - ▶ Física (muitas)
- ▶ Evolução ocorre em paralelo, graças a modularidade
- ▶ Mantida e evoluída por comunidades diferentes



DECISÕES DE MODULARIDADE NAS REDES DE COMPUTADORES

Como quebrar o sistema em módulos?

- ▶ Modelo em camadas (*layering*)

Onde os módulos são implementados?

- ▶ **Princípio fim-a-fim (*end-to-end*)**

DISTRIBUINDO CAMADAS ATRAVÉS DA REDE

- ▶ Camadas são simples em uma única máquina
- ▶ Mas precisando implementar camadas através de dispositivos diferentes e remotos
 - ▶ Hosts
 - ▶ Roteadores e switches
- ▶ O que é implementado e onde?

O QUE É IMPLEMENTADO NO HOST FINAL?

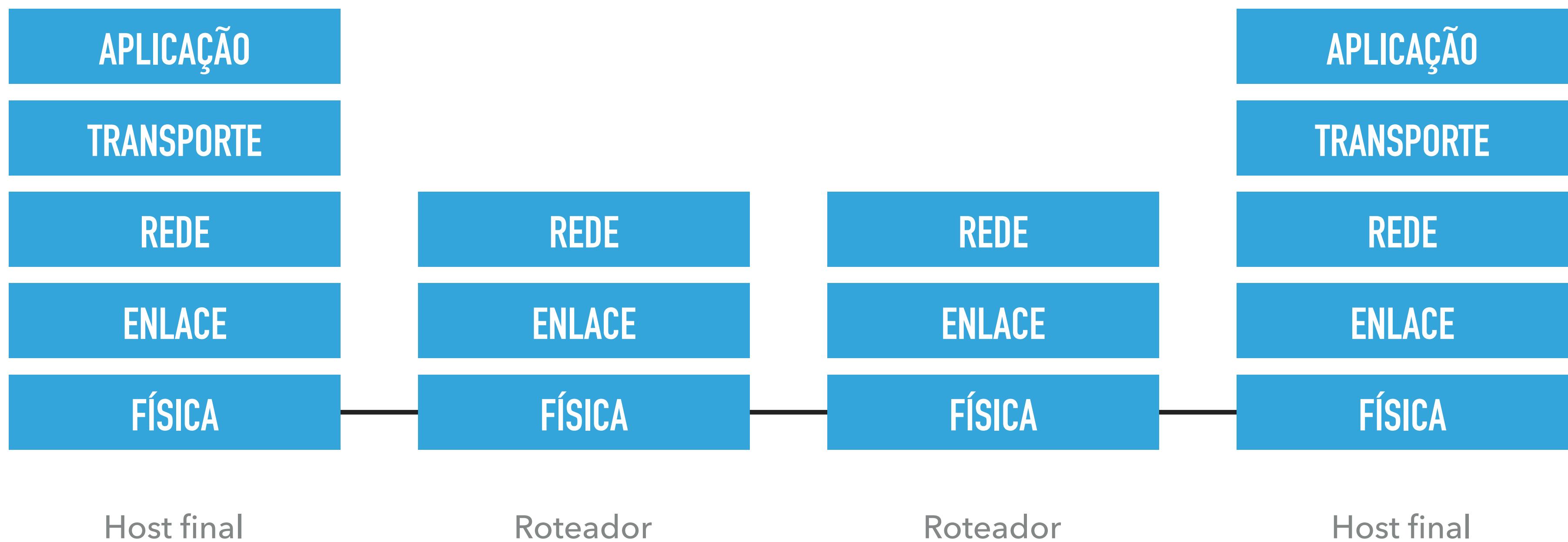
Bits chegam pelo meio físico e devem chegar até a aplicação

Ou seja, todas as camadas devem existir no host!

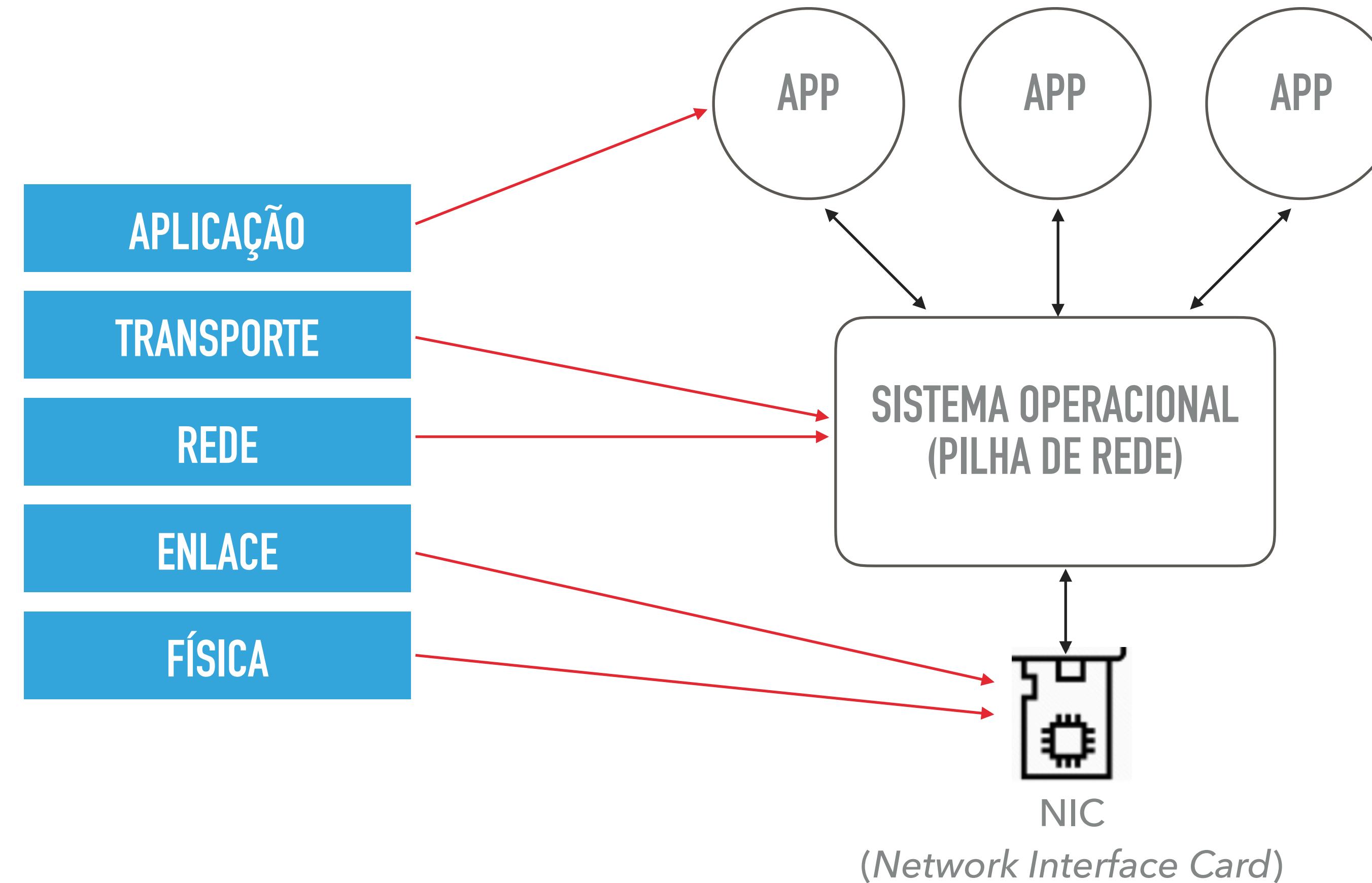
O QUE É IMPLEMENTADO NA REDE?

- ▶ Bits chegam pelo meio físico
 - ▶ **Camada física** é necessária!
- ▶ Pacotes devem ser entregues para o próximo nó da rede
 - ▶ **Camada de enlace** é necessária!
- ▶ Roteadores participam do processo de entrega global
 - ▶ **Camada de rede** é necessária!
- ▶ Roteadores não suportam entrega confiável
 - ▶ Camada de transporte (e acima) não são suportados

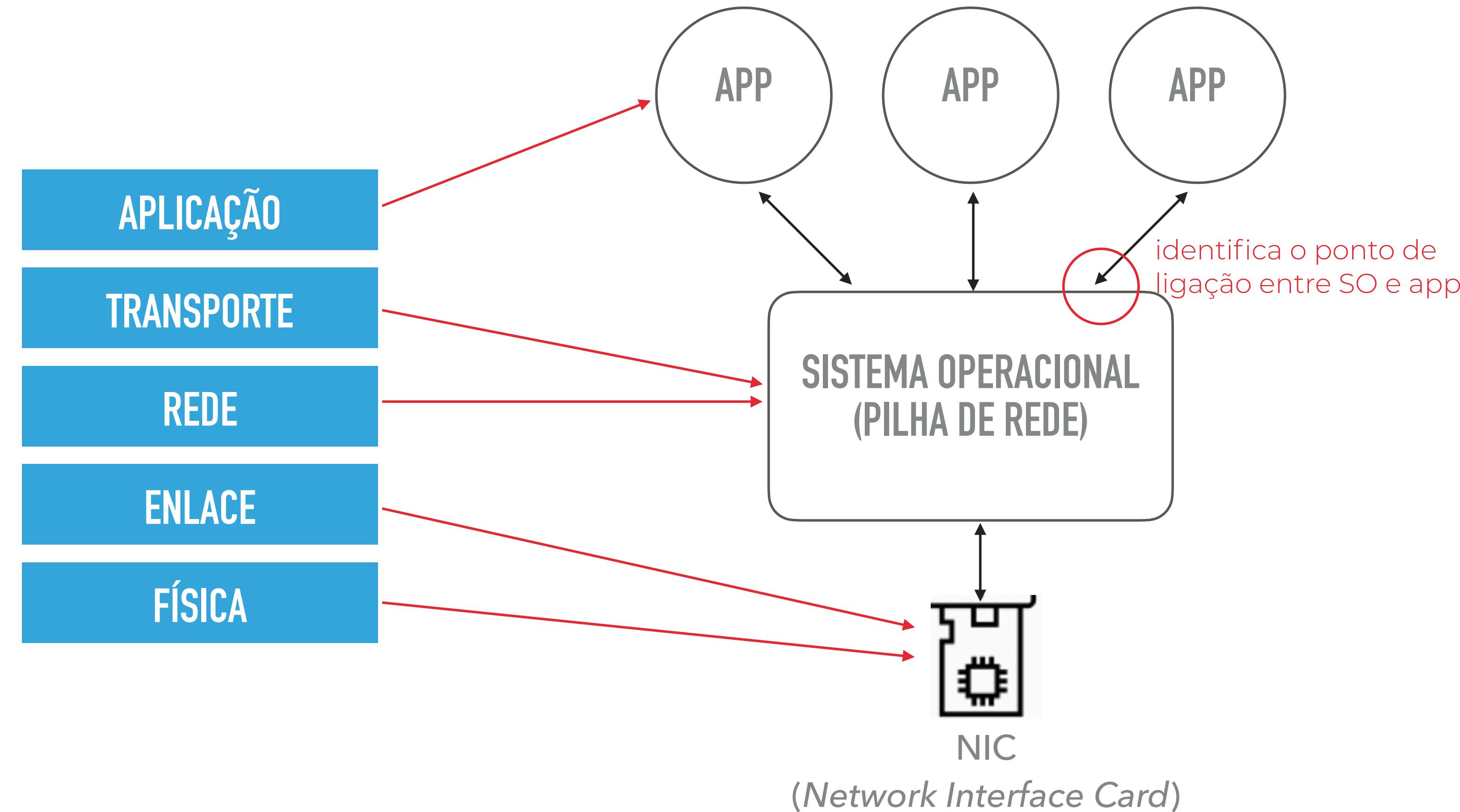
HOSTS E ROTEADORES



UM OLHAR MAIS PRÓXIMO DO DISPOSITIVO FINAL



UM OLHAR MAIS PRÓXIMO DO DISPOSITIVO FINAL



CAMADAS NO DISPOSITIVO FINAL

Camada de aplicação (L7)

- ▶ Parte da aplicação: *browser, cliente de e-mail, etc.*

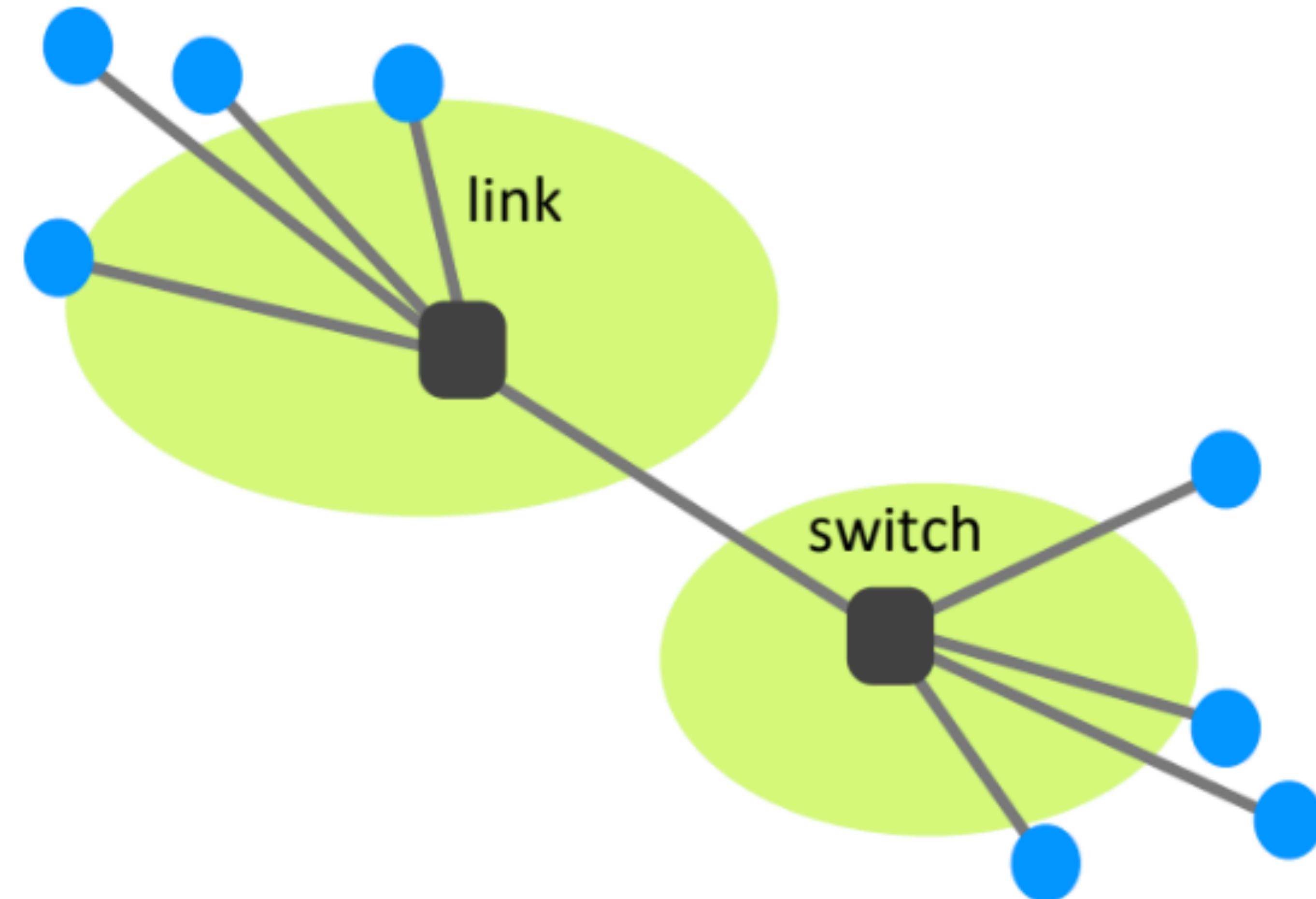
Camada de transporte e de rede (L4 e L3)

- ▶ Tipicamente parte do S.O.

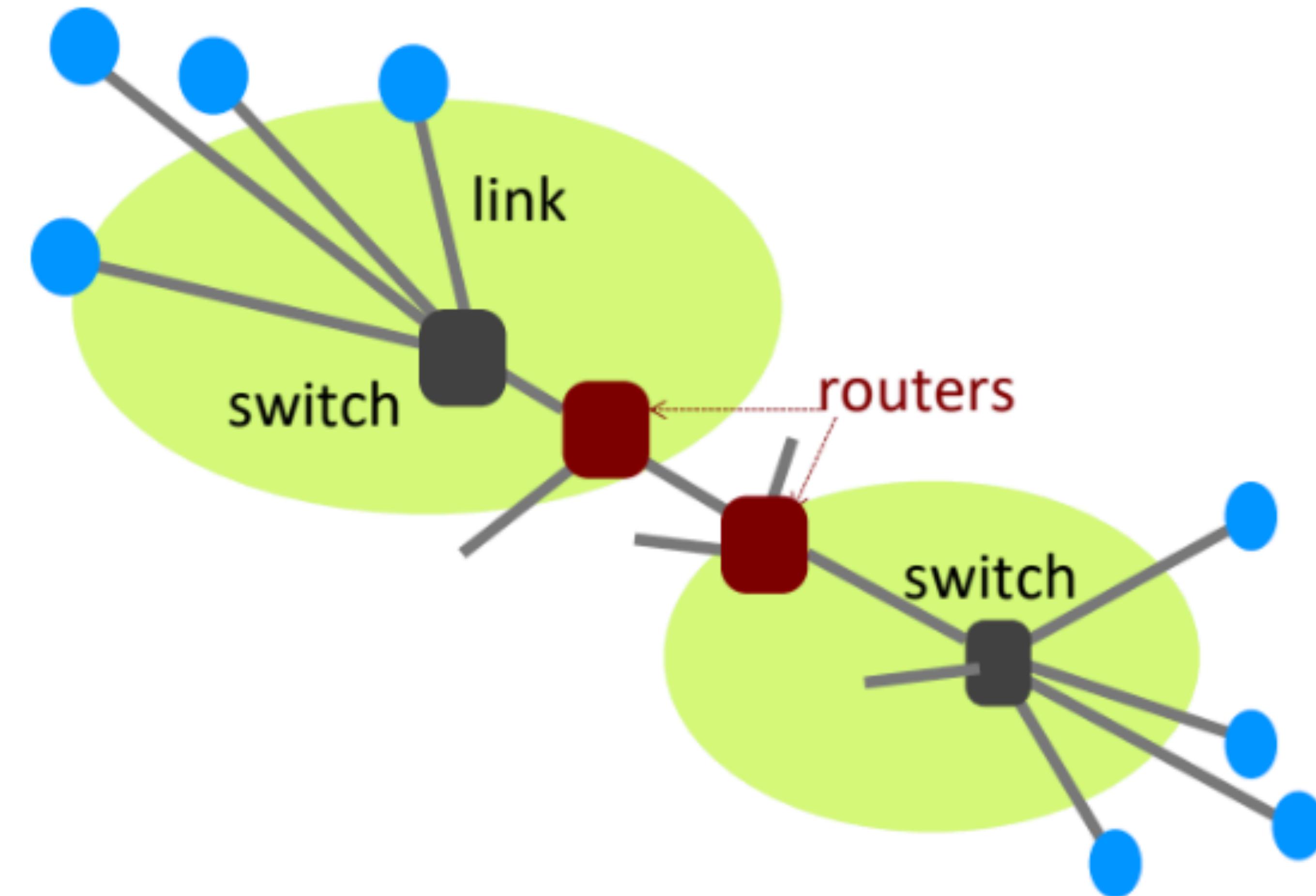
Camada de enlace e física (L2 e L1)

- ▶ *hardware, firmware, drivers*

UM OLHAR MAIS PRÓXIMO NA REDE



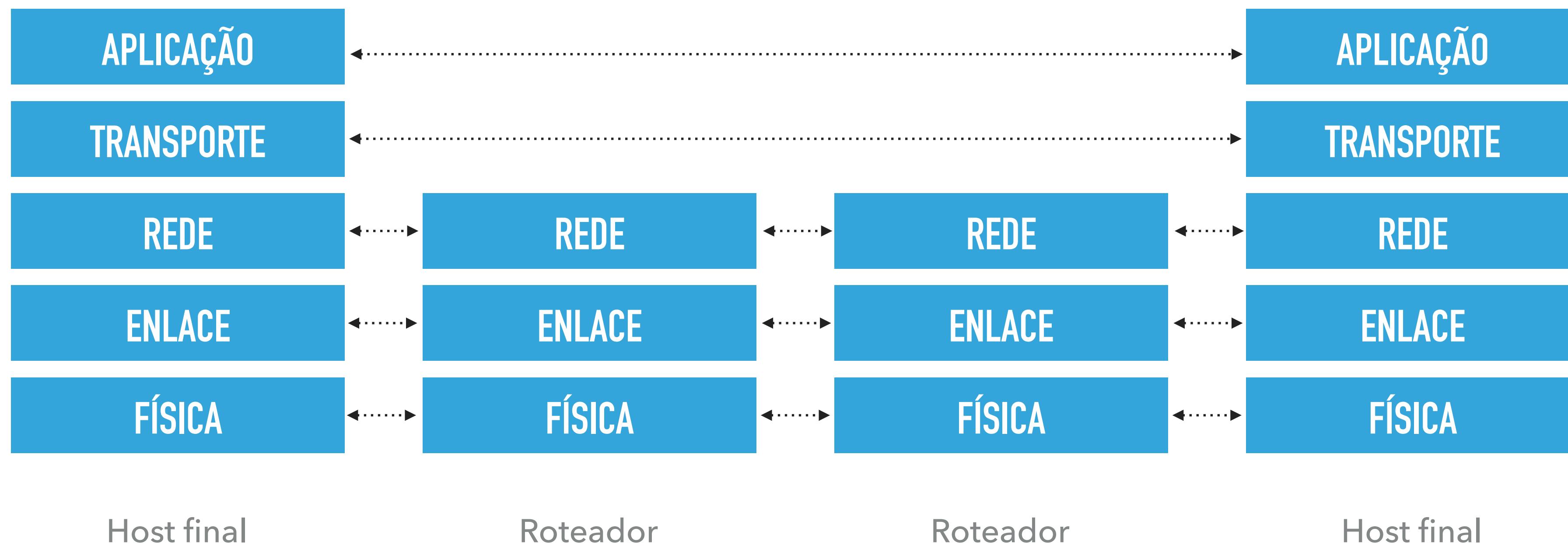
UM OLHAR MAIS PRÓXIMO NA REDE



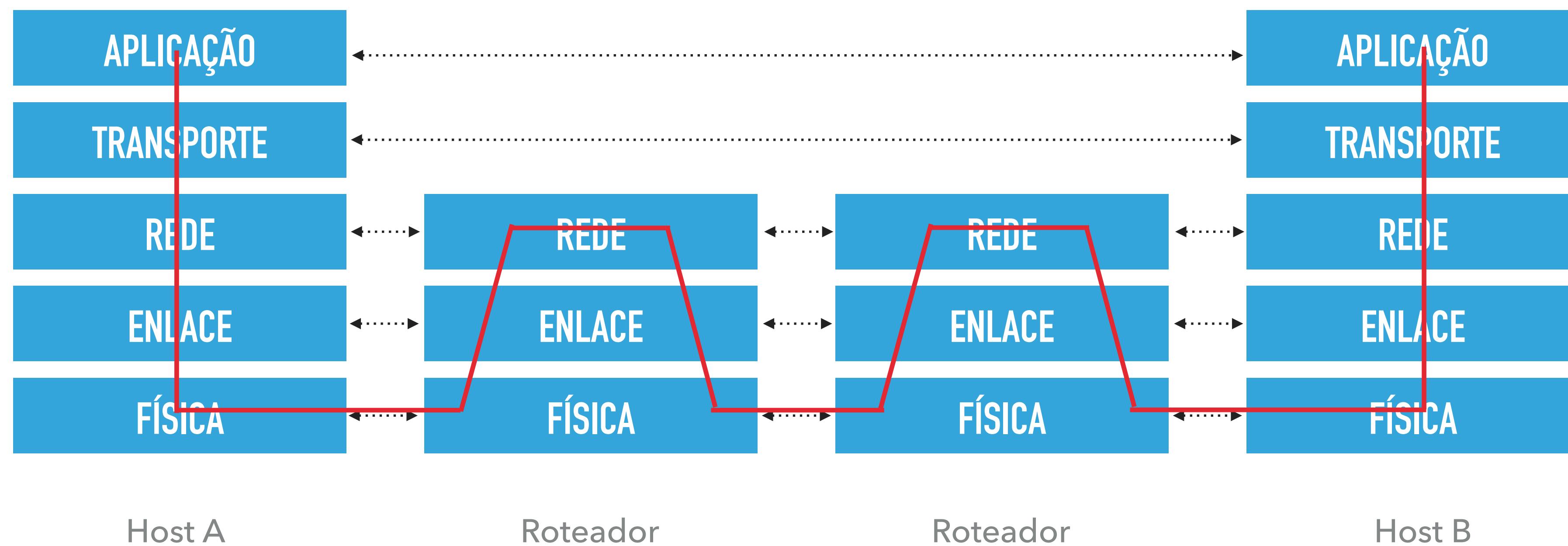
SWITCHES VS. ROTEADORES

- ▶ Switches fazem o que roteadores fazem porém não participam de processo de entrega global, somente local
 - ▶ Switches só precisam suportar L1 e L2
 - ▶ Roteadores precisam suportar L1, L2 e L3
- ▶ Em geral, não vamos focar nessa distinção agora
 - ▶ Na maioria das vezes estaremos falando de roteadores
 - ▶ Quase todos os switches hoje em dia suportam camada de rede

HOSTS E ROTEADORES

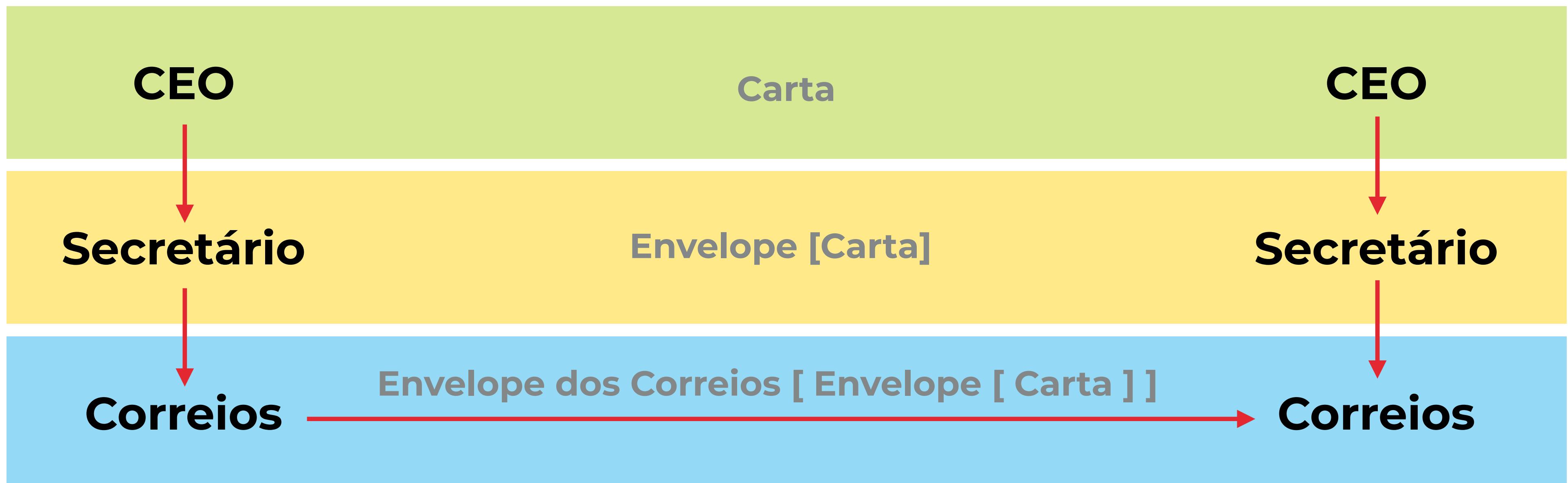


HOSTS E ROTEADORES



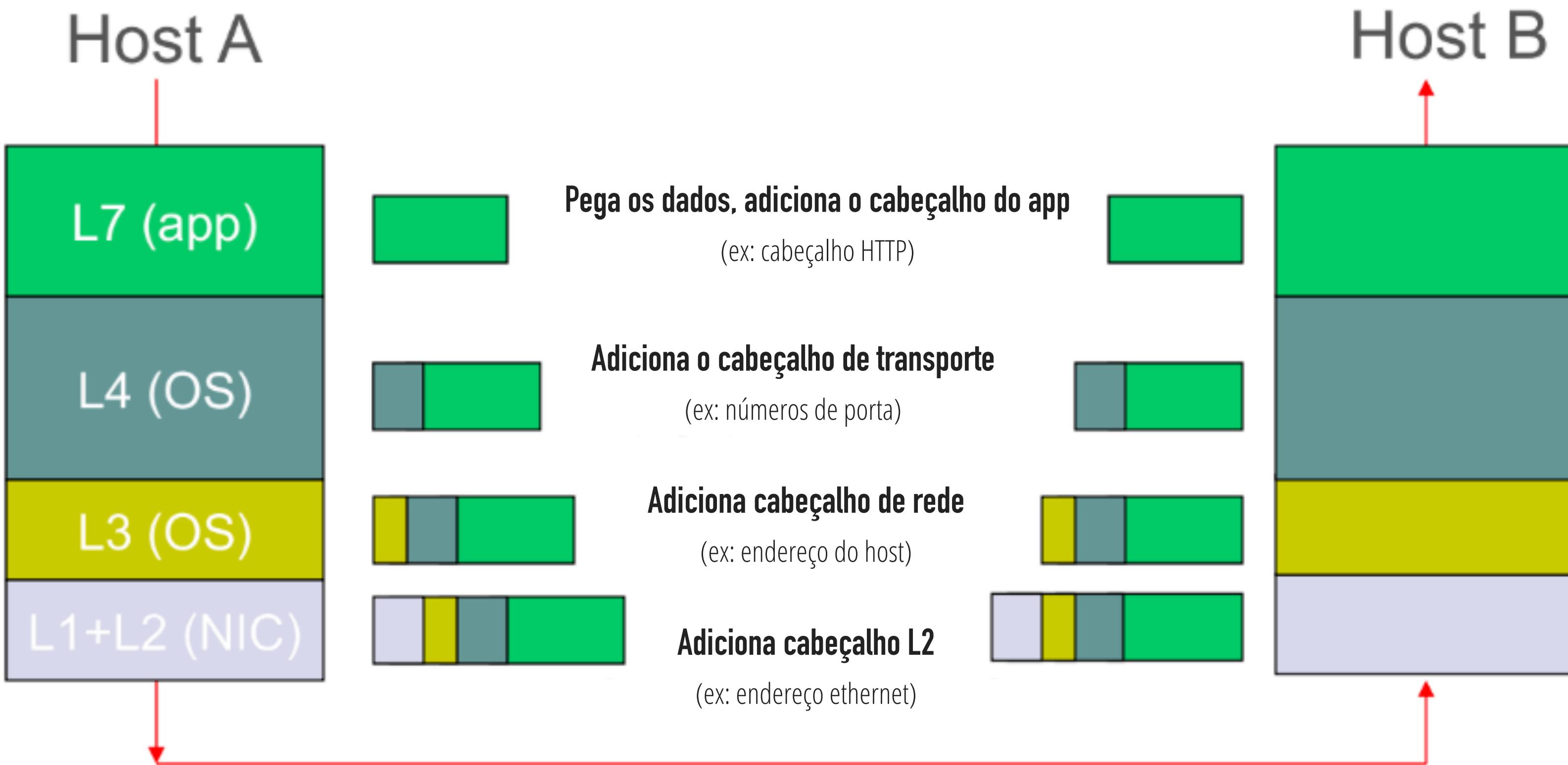
O CAMINHO DA CARTA

- ▶ Comunicação desce até a rede física
- ▶ Depois sobe até a camada relevante
- ▶ Camada mais inferior tem o maior “pacote”



ENCAPSULAMENTO

Pacotes contém múltiplos cabeçalhos!



No meio físico: pacote tem dados + cabeçalhos de todas as camadas

PDUS

PDU = Protocol Data Unit

