▶ Modelos de computação distribuída

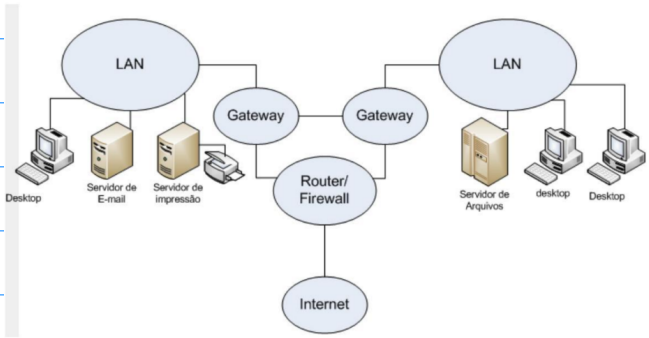
↳ Modelo Físico

- Representação dos elementos de hardware que o constituem e da rede de comunicação.

Exemplos:

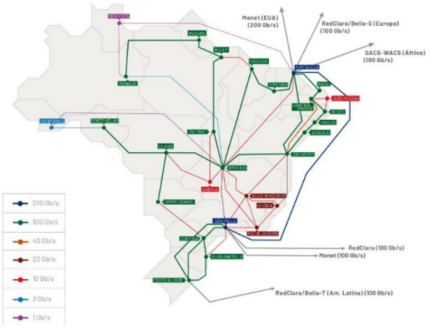
1) Primeiros SDs

- Entre 10 e 100 Nós conectados por uma rede Ethernet local.
- Pequena escala
- Acesso limitado à internet
- Nós clientes homogêneos.



2) Escala Global (Internet)

- Redes locais (LANs) conectadas por backbones
- Grande nº de Nós
- Nós heterogêneos
- Gerenciamento de serviços de QoS



3) SDs Contemporâneos

- Mobilidade
- Virtualização de hosts e de redes
- Microserviços comunicantes nos End-points
- Tecnologias Emergentes (BtC, Nuvem, IoT...)

➡ Arquitetura Cliente/Servidor

1- O cliente invoca/Requisita serviços através do canal de comunicação com o servidor

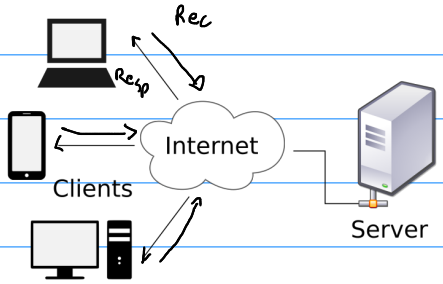
2- O servidor recebe as requisições e executa (provê) serviços. Através do canal

↳ O cliente deve se conectar na mesma porta do processo servidor.

✱ Os "papeis" podem se inverter, ou seja, o cliente provem serviços e o servidor requisitou.

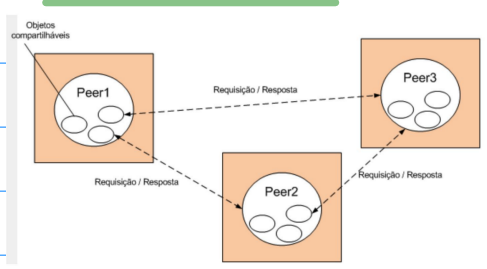
✱ Podem existir vários processos servidores em um único host, porém cada um em uma porta.

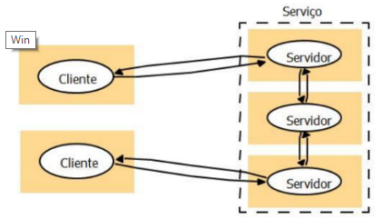
a Clientes



➡ Arquitetura Peer-to-Peer

- Todos os processos atuam como parceiros (peers) sem distinção entre cliente e servidor atuando ora como cliente (Requisitando), ora como servidor (Provê serviços).
- Todos os processos executam o mesmo programa.
- Um nó pode distribuir o conteúdo para outros Nós da rede.
- Muito escalável.





▷ Mapeamento de Serviços

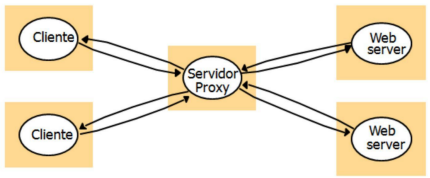
↳ Serviços podem ser implementados como múltiplos processos servidores em hosts separados, interagindo para fornecer serviços para clientes.

Ex.º Web

* CACHE

↳ Local com os dados recentemente acessados, mais próximo do cliente.

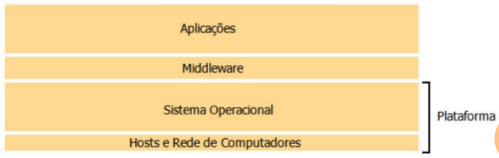
- Novos obj não armazenados na cache
- O serviço de cache fornece uma cópia atualizada e caso não exista é feita uma nova requisição.
- Cache pode ser local ou compartilhada.



▶ Modelos de computação distribuída

○ Arquitetura em camadas

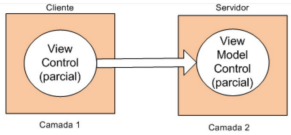
↳ Agrupa os serviços que são requisitados localmente ou remotamente.



- Lógica de apresentação: Visão do usuário
- Lógica de aplicação: Controle, dados específicos da aplicação
- Lógica de dados: Detalhes sobre como os dados são persistidos na base de dados.

▶ Arquitetura em 2 camadas

○ Agrupa a lógica MVC em processos Cliente / servidor, onde a lógica da aplicação fica parte no cliente e parte no servidor para reduzir a latência dos requisições.

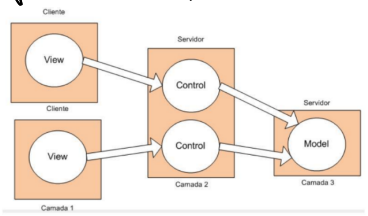


Arquitetura em 3 camadas

- Lógicas da aplicação (control) e armazenagem (model) ficam no servidor.

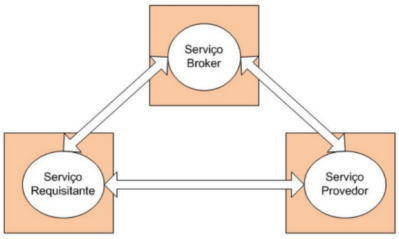
Tem uma melhor distribuição de carga nas camadas internas do serviço

↳ A lógica de aplicação pode ser modificada ou replicada sem interrupção dos serviços dos clientes.



Arquitetura Broker

- Serviço de descoberta e registro de endereços de serviços.
- Servidores → Anunciam seus endereços no Broker
- Clientes → Consultam o Broker para descobrir a localização remota do provedor de serviços
- Muito utilizado em RMI e Web Services



Modelos fundamentais

- definem aspectos comuns em SDs

↳ Modelo de Interação: Lida com a dificuldade de atribuir limites de Tempo de interação entre processos.

↳ Modelo de falha: busca especificar as falhas na comunicação entre processos

↳ Modelo de Segurança: Analisa possíveis vulnerabilidades na comunicação entre processos