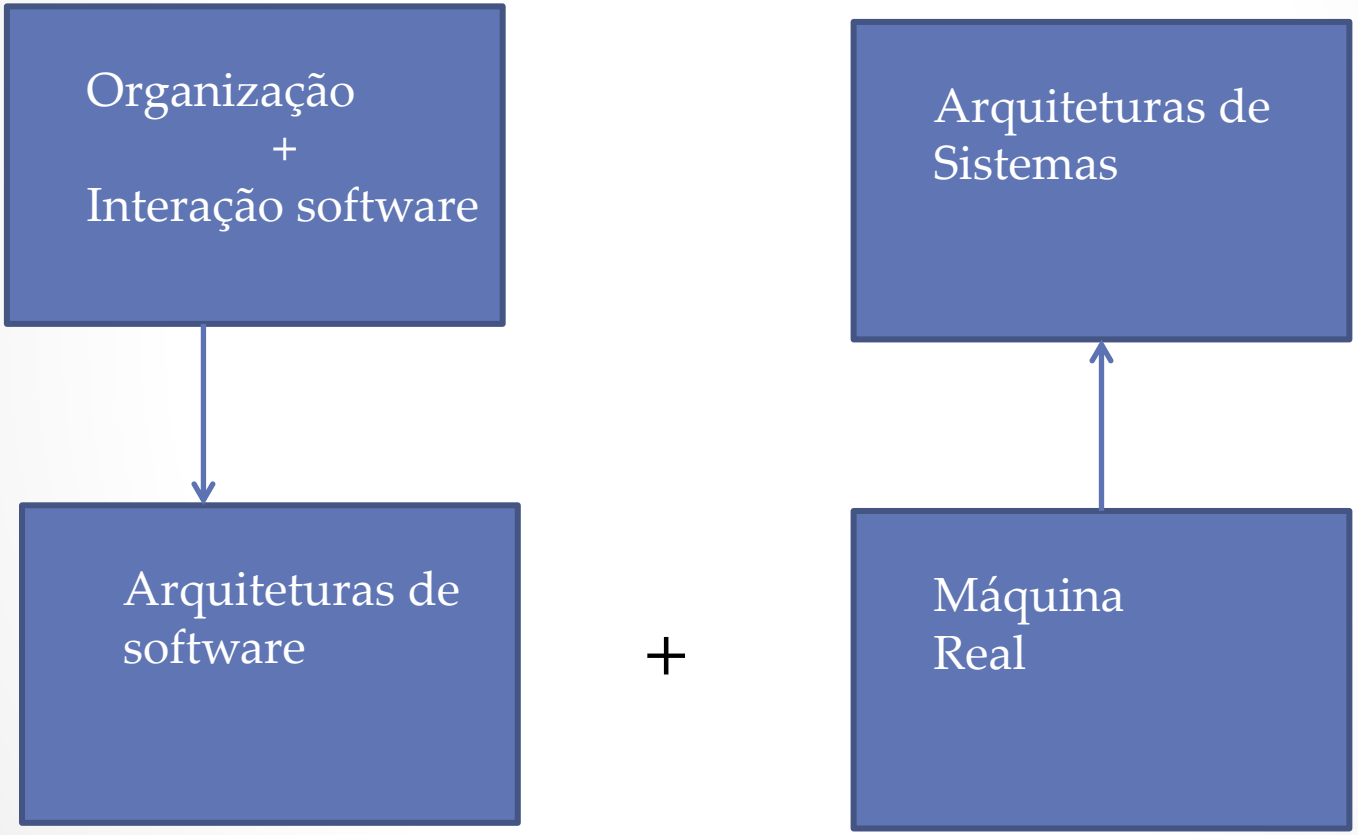
# Universidade Estadual de Mato Grosso do Sul
# Curso de Ciência da Computação
# Disciplina de Algoritmos Paralelos e Distribuídos

Aula 2 – Modelos de Sistemas e Arquiteturas

# Sistemas Distribuídos



# Aspectos de um Sistema Distribuído

- Modelo Físico
  - É o modo mais explícito de se descrever um sistema. É a descrição de *hardware* do sistema.
- Modelo Arquitetônico
  - É a descrição de um sistema em termos de tarefas computacionais e comunicações executadas pelos elementos do sistema.
- Modelo Fundamental
  - É uma perspectiva abstrata cujo objetivo é examinar aspectos individuais do sistema.
- Obs: Os componentes do sistema podem ser substituídos desde que suas interfaces sejam mantidas.

# Estilo Arquitetônico

- Tem como princípio definir:
  - O modo como os componentes estão conectados;
  - O modo como os dados são trocados entre os componentes;
  - O modo como os componentes são configurados em conjunto para formar um sistema.

- Os estilos arquitetônicos podem ser:
  - Arquiteturas em camadas;
  - Arquiteturas baseadas em objetos;
  - Arquiteturas baseadas em dados;
  - Arquiteturas baseadas em eventos;

# Estilo Arquitetônico Arquitetura em Camadas

- Os componentes são organizados em camadas.
- O componente da camada N tem permissão de chamar componentes da camada N-1.
- É um estilo comum em redes de computadores.

# Estilo Arquitetônico
# Arquiteturas Baseada sem Objetos

- Objeto → componente.
- Objetos são conectados por meio de uma chamada de procedimento (remota)
- Amplamente utilizada para sistemas de software de grande porte.

# Estilos Arquitetônicos

# Arquiteturas Centradas em Dados

- Os processos se comunicam por meio de um repositório comum.

- Sistemas distribuídos baseados na Web, em grande parte, são centrados em dados.

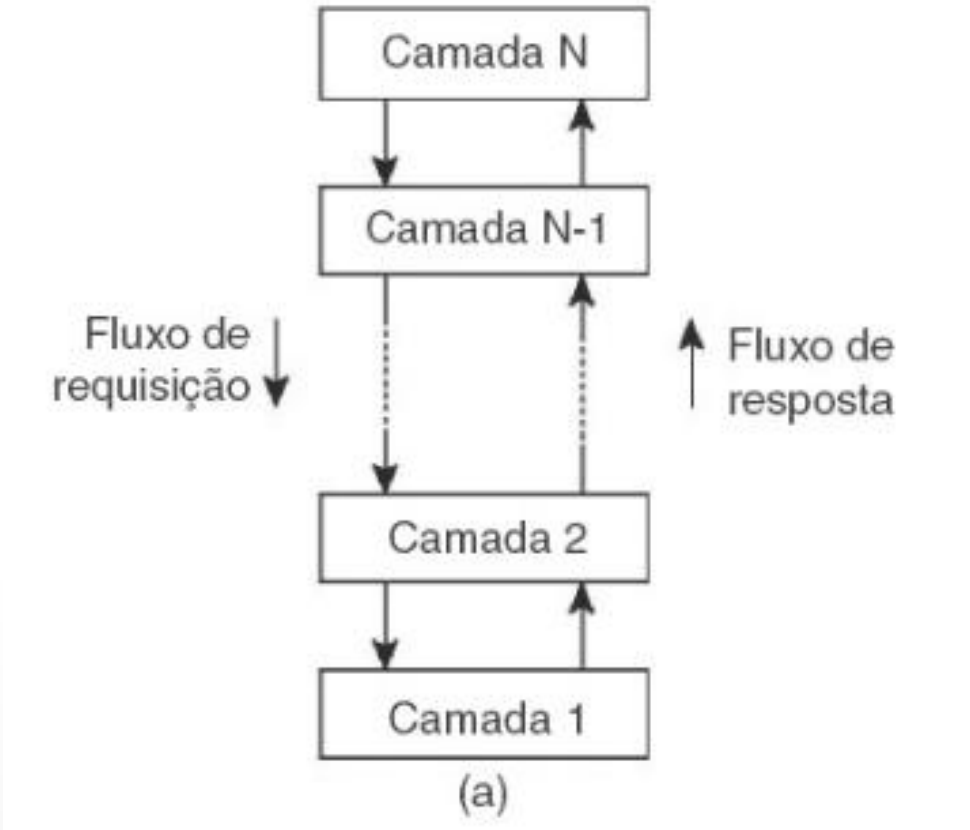
# Estilos Arquitetônicos

# Arquiteturas Baseadas em Eventos

- Sistemas publicar / subscrever
- Processos publicam eventos e o middleware assegura que somente os processos que se subscreveram para esses eventos poderão ter acesso a esses eventos
- Processos fracamente acoplados: processos não se referem explicitamente uns aos outros.

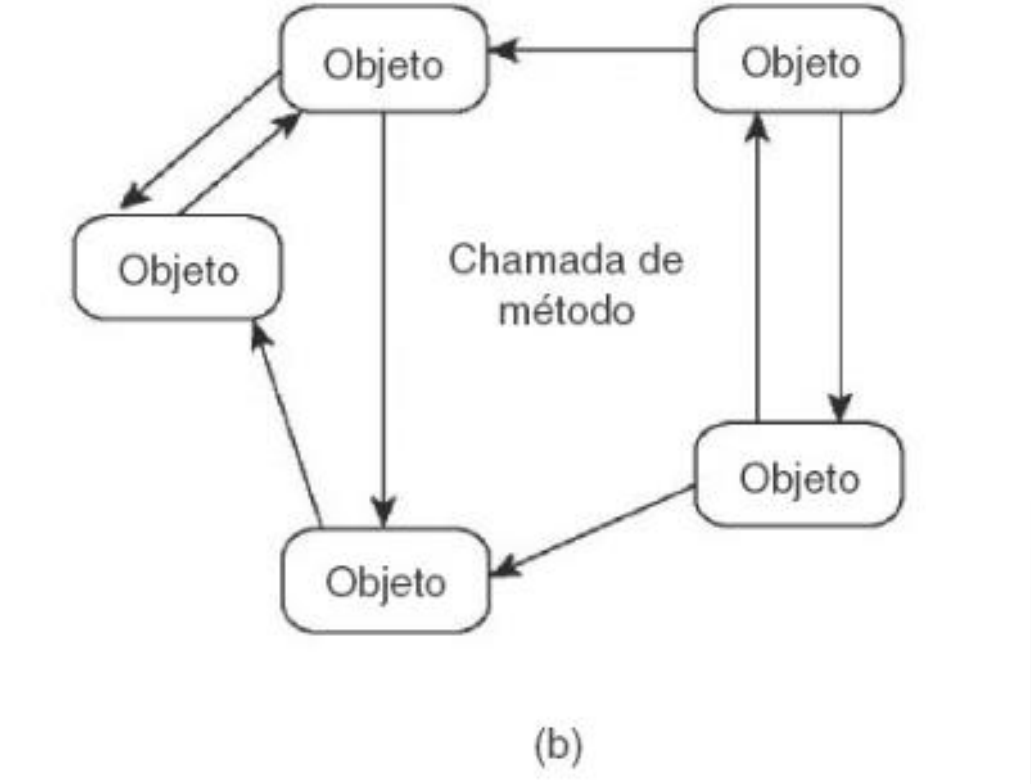
# Estilos Arquitetônicos

- a) estilo arquitetônico em camadas



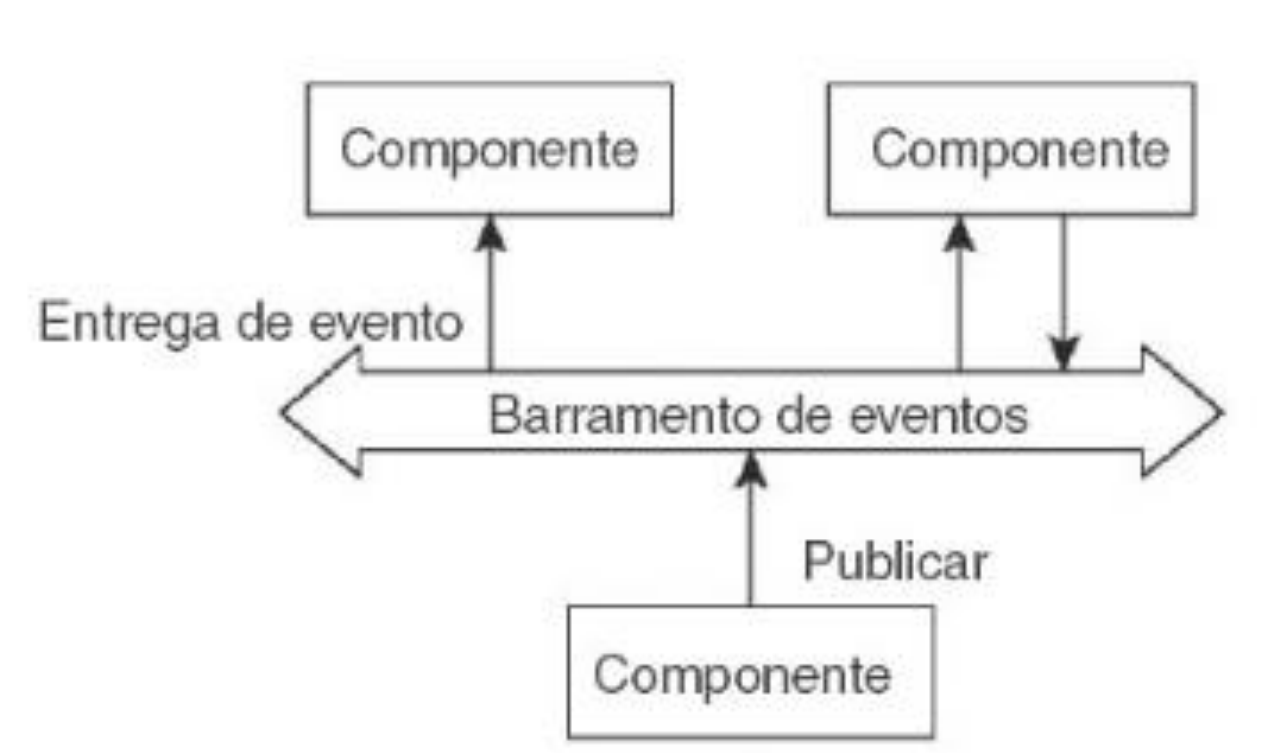
# Estilos Arquitetônicos

- b) estilo arquitetônico baseado em objetos



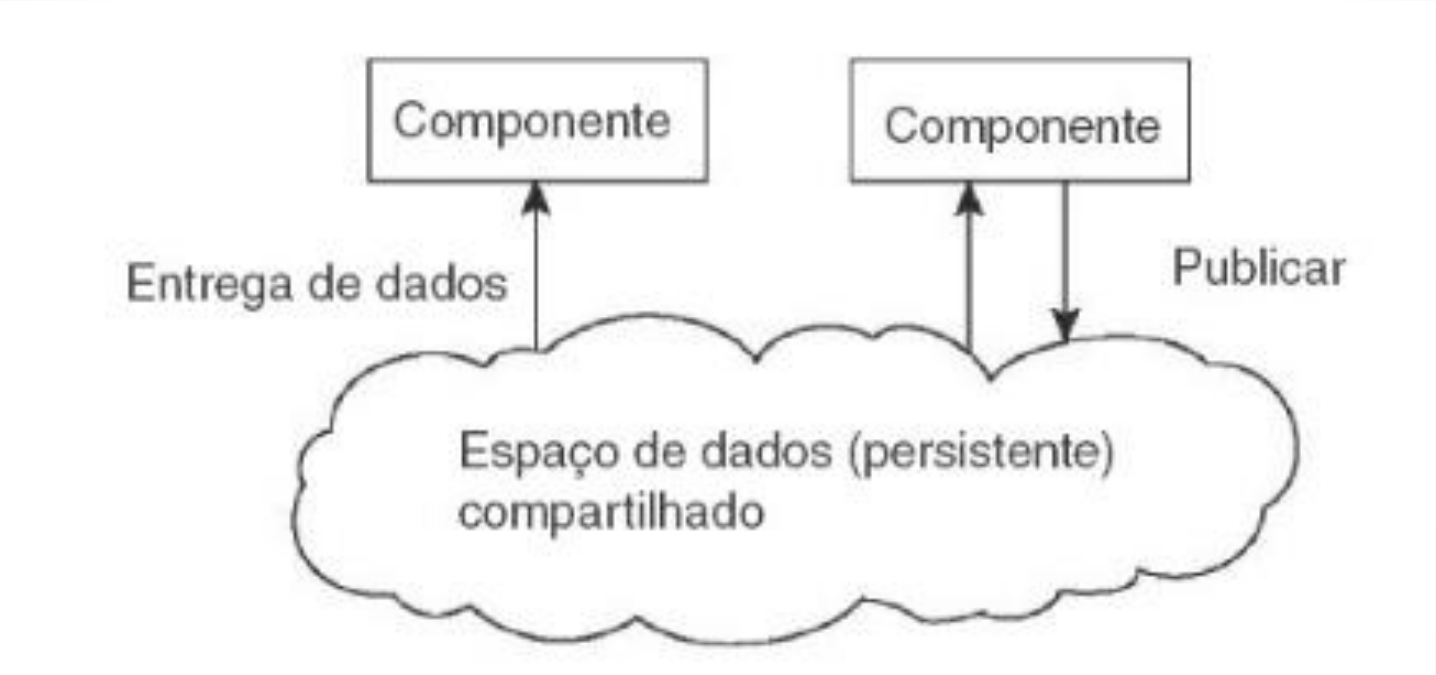
# Estilos Arquitetônicos

- C)estilo arquitetônico baseado em eventos



# Estilos Arquitetônicos

- d) estilo arquitetônico baseado em espaço de dados compartilhados



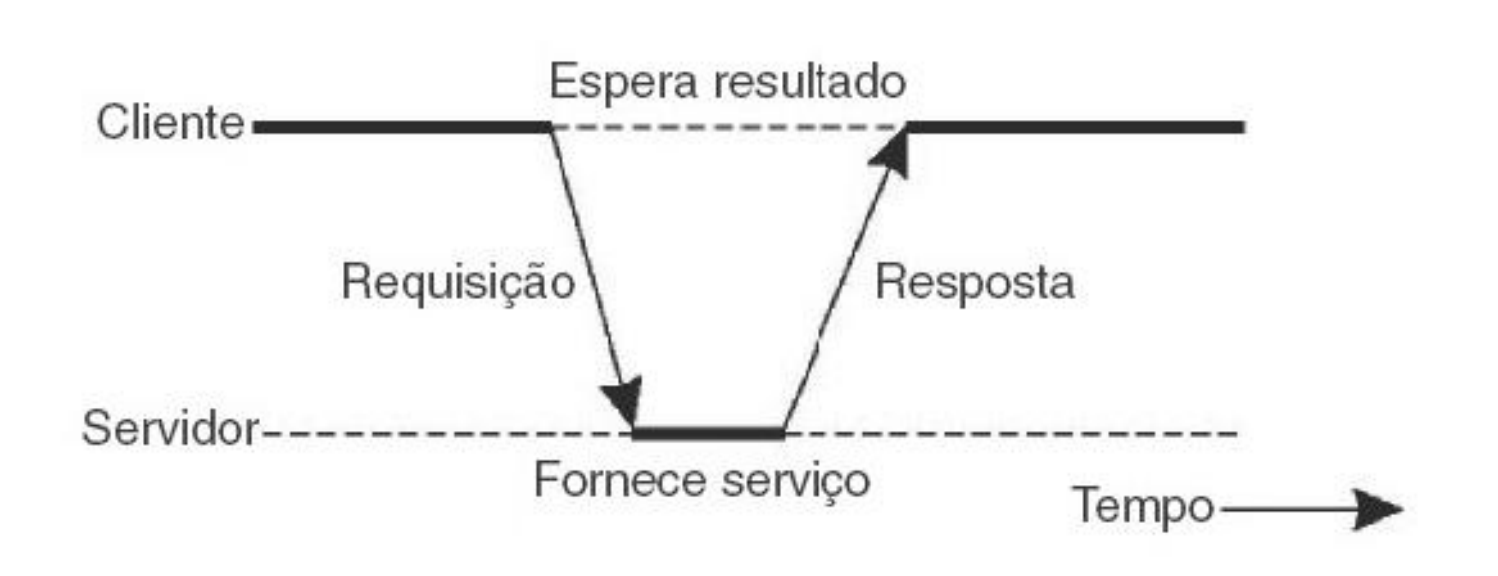
# Arquitetura de Sistemas

- Diz respeito a componentes de software, sua interação e sua colocação em máquinas reais.

- São três os tipos de arquiteturas de sistemas:
  1. Centralizadas
  2. Descentralizadas
  3. Híbridas

# Arquiteturas de Sistemas Centralizadas

- Típico modelo cliente-servidor
- Comportamento de requisição – resposta



# Arquiteturas de Sistemas Centralizada

- Estabelecimento da comunicação:
  1. Protocolo sem conexão
     1. Protocolo simples que funciona em redes locais
     2. Cliente empacota um mensagem para o servidor diretamente.
     3. Este método é eficiente desde que não haja problemas.
     4. Exemplo: falhas em transferências bancárias
     5. As operações podem ser repetidas sem causar danos

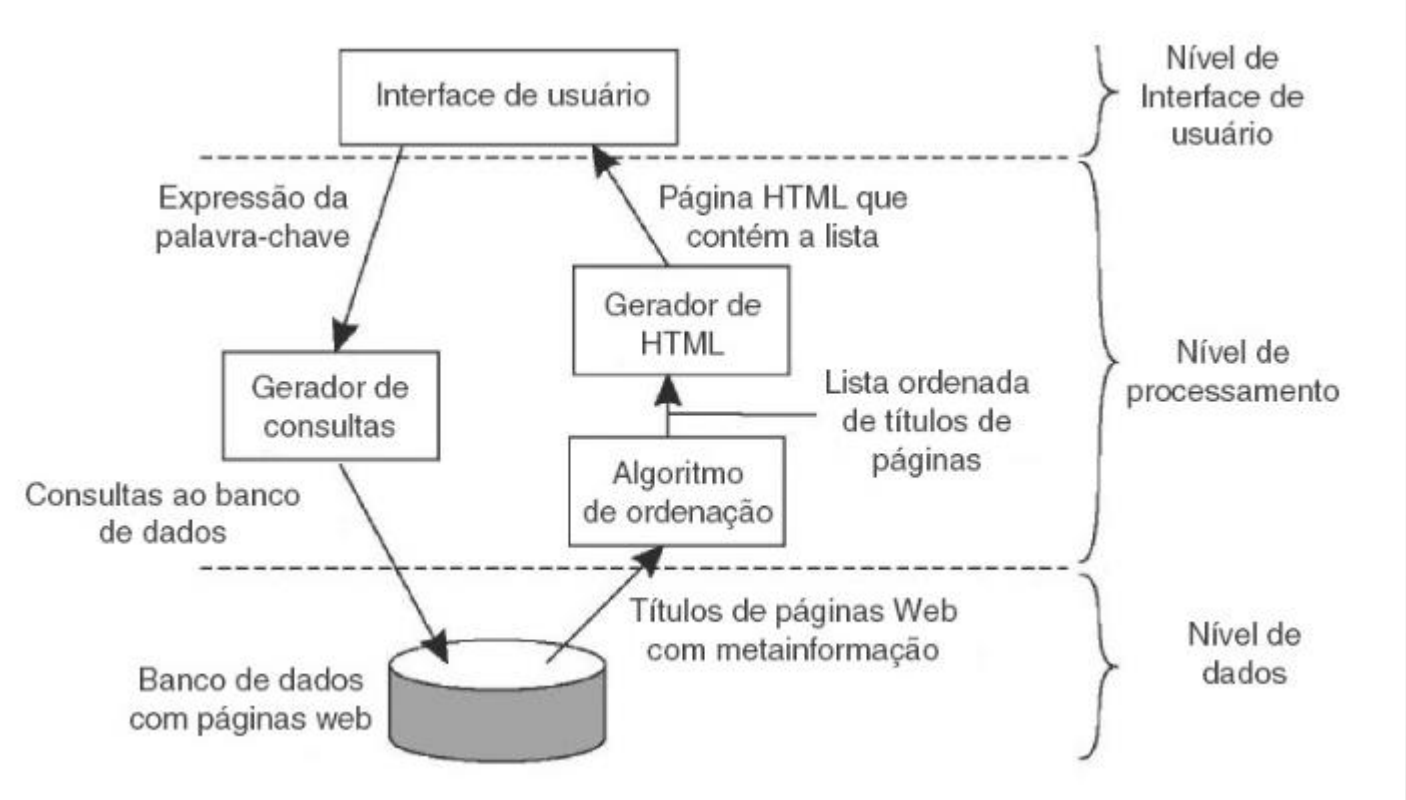
  2. Protocolo orientado a conexão
     1. Solução funciona bem em sistemas de longa distância
     2. Sempre que um cliente requisita um serviço, primeiro se estabelece a conexão com o servidor e depois se envia a requisição.

# Arquitetura de Sistemas Centralizada

- Como distinguir entre cliente e servidor?
  - Exemplo: O servidor de banco de dados distribuído repassa requisições a servidores de arquivos.
- Visando que muitas aplicações consideram dar suporte ao acesso de usuários a banco de dados:
  - Nível de interface do usuário
  - Nível de processamento
  - Nível de dados

# Arquiteturas de Sistemas Centralizadas

- Organização simplificada de um mecanismo de busca da internet em três camadas diferentes.

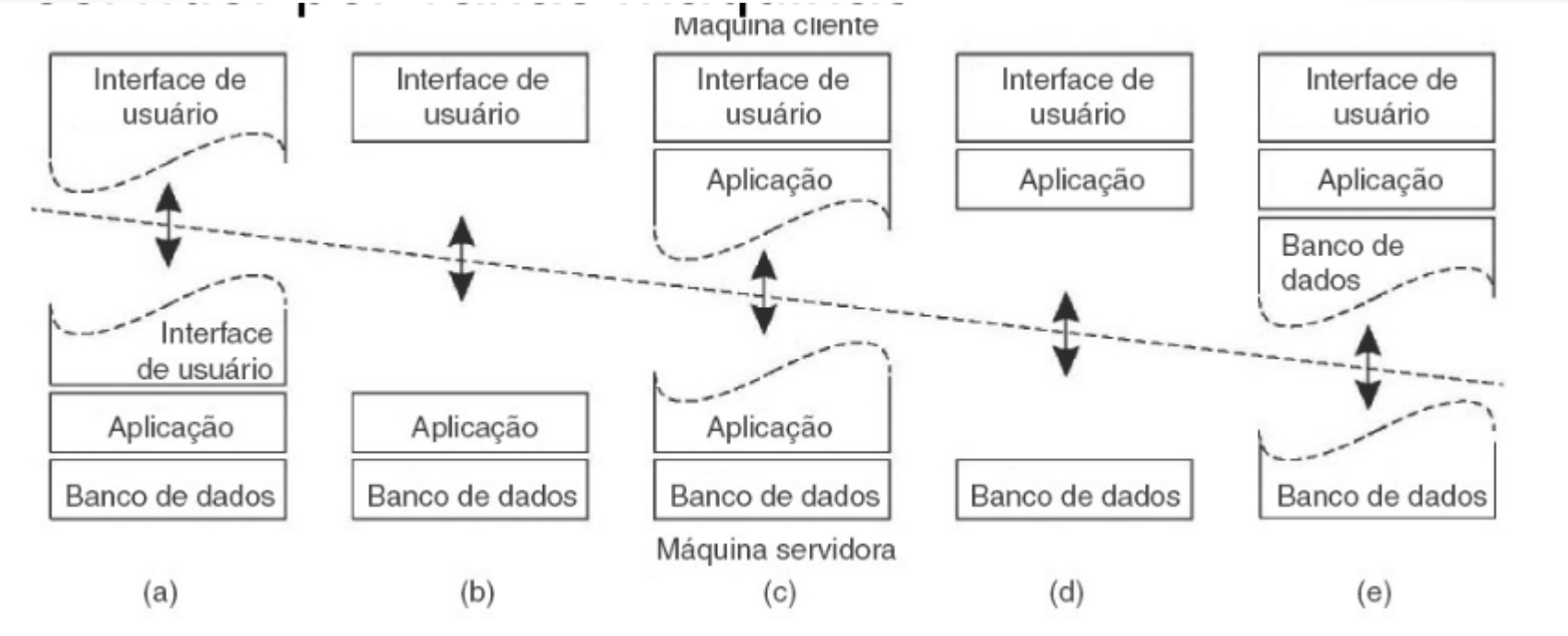


# Arquiteturas de Sistemas Centralizadas

- Nível de Interface do Usuário
  - Consiste em programas que permitam aos usuários finais interagir com aplicações.
  - Pode apresentar diversos níveis de complexidade.
- Nível de Processamento
  - Normalmente contém as aplicações .
  - Exemplo: Análise de dados financeiros que pode exigir métodos e técnicas sofisticadas de estatística.
- Nível de Dados
  - Na sua forma mais simples, consiste em um sistema de arquivos
  - É muito utilizado em banco de dados
  - Geralmente implementado no lado servidor
  - Mantém os dados consistente
  - Os dados costumam ser persistentes

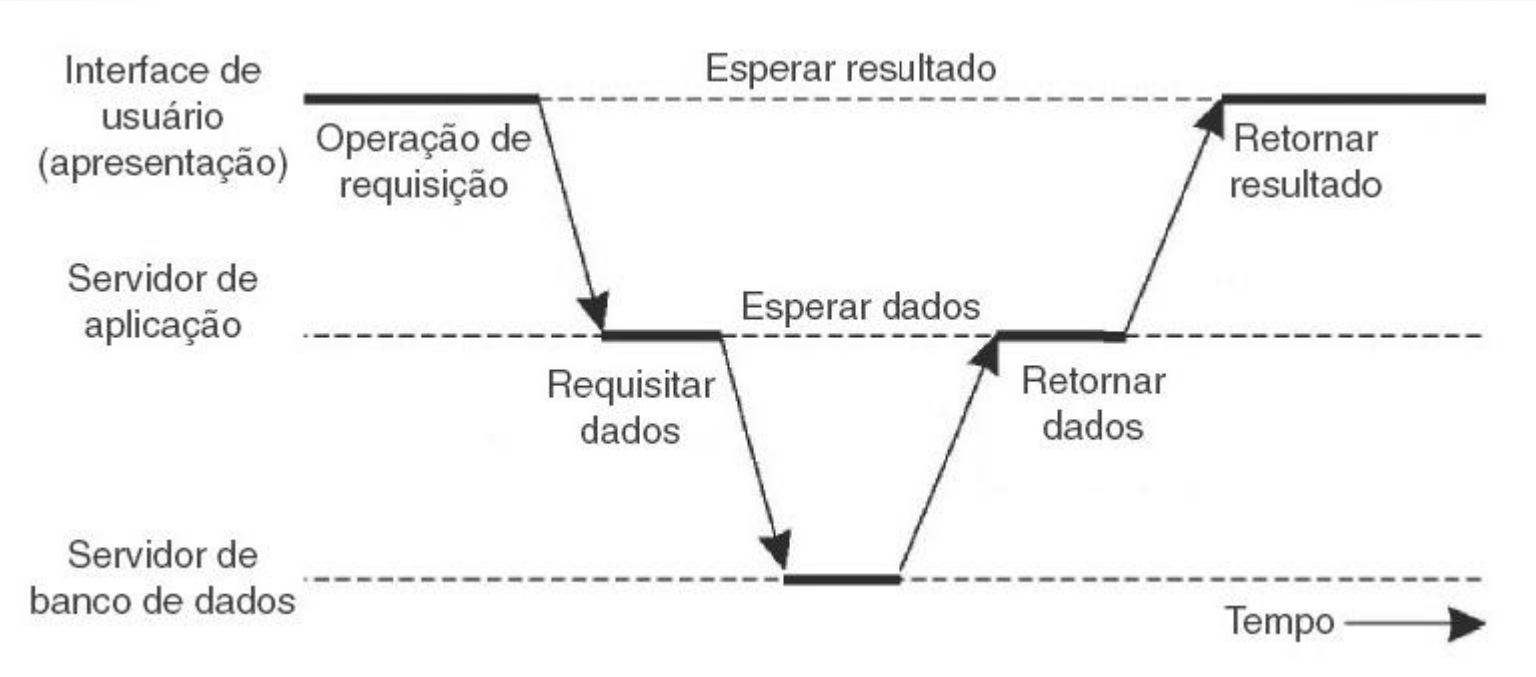
# Arquiteturas de Sistemas
# Arquiteturas Multidivididas

- Apresentam três níveis lógicos com várias possibilidades para a distribuição física de uma aplicação cliente-servidor por várias máquinas.
- Exemplo: alternativa de organização cliente-servidor



# Arquiteturas de Sistemas Arquiteturas Multidivididas

- Exemplo de um servidor que age como cliente



# Arquiteturas de Sistemas Descentralizadas

- Cliente-servidor possuem duas distribuições:
  - Distribuição vertical
    - Componentes logicamente diferentes em máquinas diferentes
    - Cada máquina é projetada para um grupo específico de funções
  - Distribuição horizontal
    - Cliente ou servidor pode ser fisicamente subdividido em partes logicamente equivalentes
    - Porção própria de dados para cada parte.

# Arquiteturas de Sistemas Descentralizadas: Peer toPeer

- Processos são todos iguais
- Grande parte da interação entre processos é simétrica
- Cada processo age como cliente e servidor ao mesmo tempo
- É formado por um conjunto de nós, organizados em um *overlay* ou rede de sobreposição
  - Overlay: rede na qual os nós são os processos e os enlaces representam os canais de comunicação possíveis
- A comunicação não pode ser feita diretamente
- Aplica-se a arquiteturas estruturadas ou não-estruturadas

# Arquiteturas Peer to Peer Estruturadas

- Rede de sobreposição é construída com a utilização de um procedimento deterministico.
- **Tabela de Hash Distribuída**
- Dados e nós recebem uma chave aleatória
- Importante: conseguir implementar um esquema eficiente e determinístico que mapeia a chave de um dado para o identificador de um nó.
  - Ao consultar um determinado item de dado, o endereço de rede do nó com o conteúdo é retornado.
  - A requisição é roteada entre os nós até que o nó com o dado requisitado seja alcançado.

# Arquiteturas Peer to Peer Não-Estruturadas

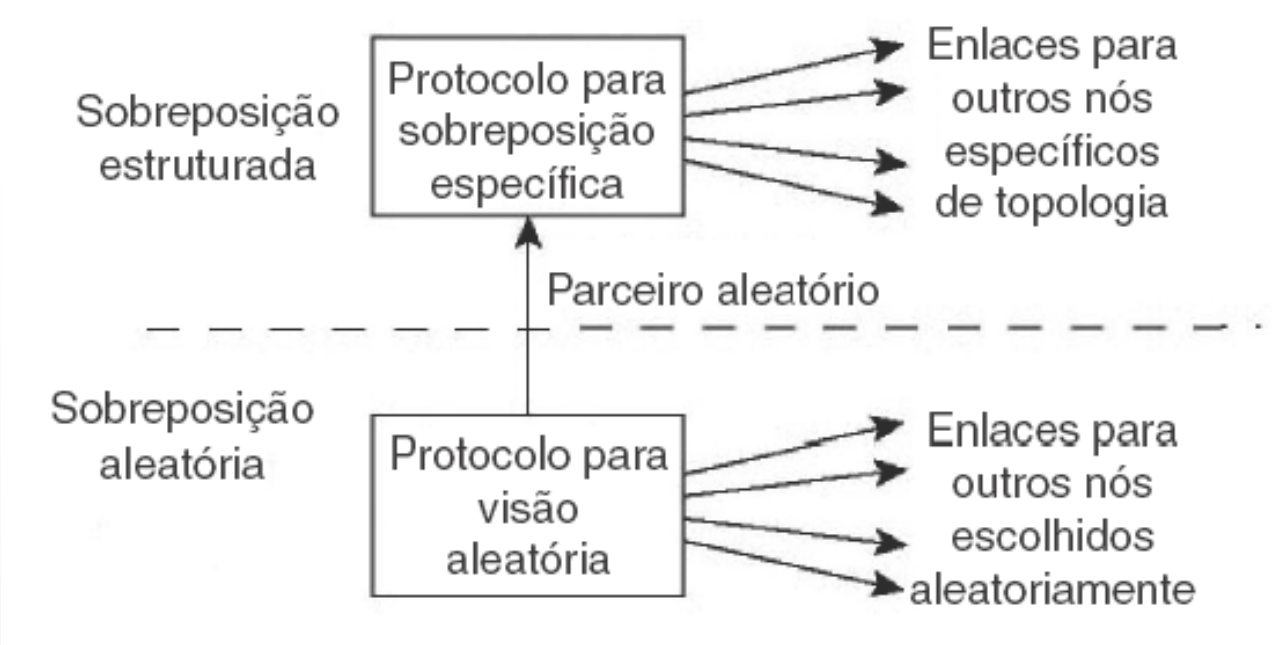
- Algoritmos não aleatórios são usados para construir a rede de sobreposição.
- Cada nó mantém um lista de vizinhos.
- Dados também são espalhados aleatoriamente.
- Como encontrar os dados?
  - Inundar a rede com uma busca (flood).

# Arquiteturas Peer to Peer Não-Estruturadas

- Gerenciamento de associação ao grupo
  - Grafo aleatório
  - Cada nó possui ***n*** vizinhos → sempre uma visão parcial do todo.
  - Nós trocam entradas regularmente de sua visão parcial.
  - Principal objetivo: Atualizar saída de nós, construir uma nova vizinhança de forma dinâmica para alcançar uma característica em específico.
  - Nós trocam as listas de vizinhos em dois modos diferentes: *pull* (puxar) e *push* (empurrar).
  - Protocolos que usam somente *pull* e *push* → grafos não conectados.

# Arquiteturas Peer to Peer
# Gerenciamento de Topologia

- Abordagem de duas camadas para construir e manter topologias específicas de sobreposição usando técnicas de sistemas peer-to-peer não estruturadas.

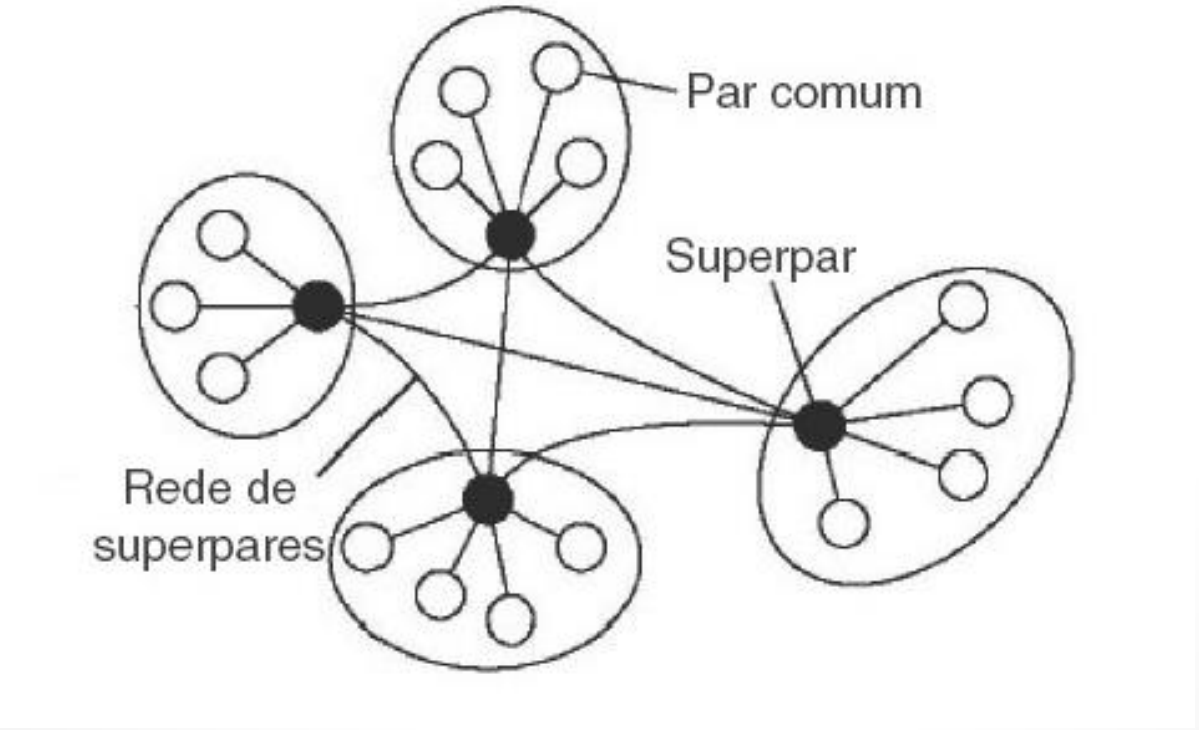


# Arquiteturas Peer to Peer Superpares (superpeers)

- À medida que a rede cresce, localizar itens de dados em sistemas P2P não estruturados pode ser problemático.
- Nós que mantém o índice de dados ou que agem como nós intermediários que possuem dados para disponibilizar os recursos a nós vizinhos.
- Sempre que um nó  comum se junta à rede, ele se liga a um dos superpares.
- Problema: Seleção do lider!

# Arquiteturas Peer to Peer Superpares (superpeers)

- Organização hierárquica de nós em uma rede de superpares.

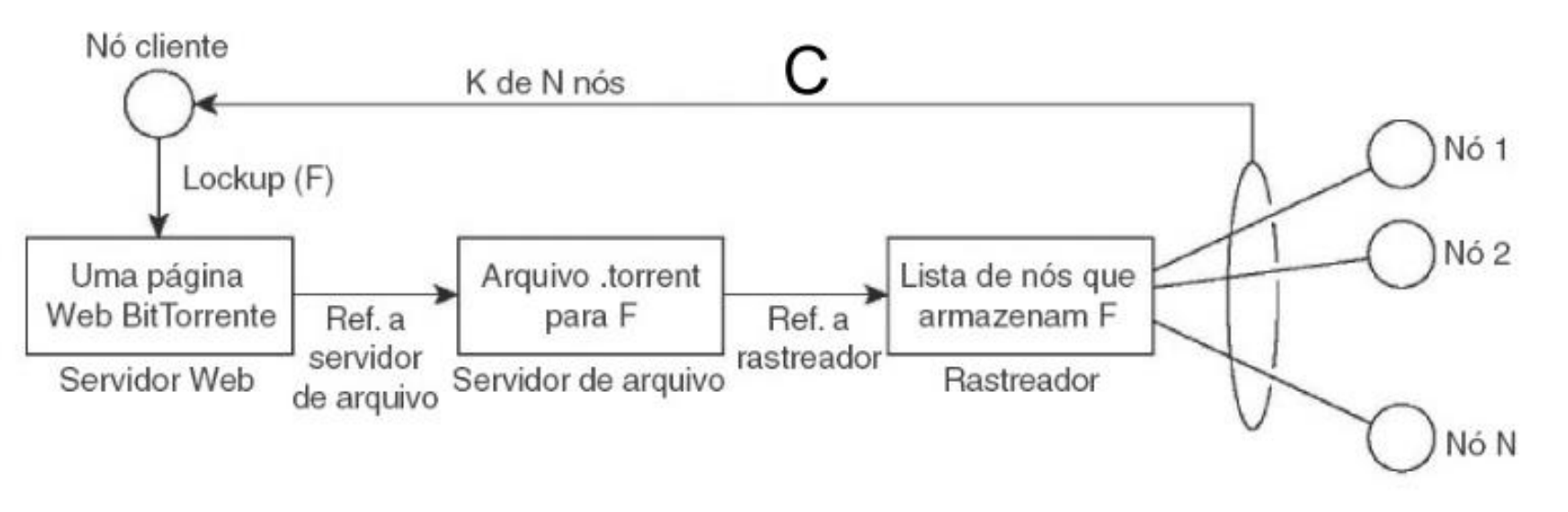


# Arquiteturas Híbridas

- Sistemas híbridos nas quais soluções cliente-servidor são combinadas com arquiteturas descentralizadas.
- Exemplo: Sistemas distribuídos colaborativos.
  - Objetivo principal: iniciar a troca de informações.
  - Após a adição de um nó na rede, a distribuição dos dados é feita de forma descentralizada.
  - BitTorrent

# Arquiteturas Híbridas BitTorrent

- Funcionamento principal do BitTorrent



# Arquiteturas *versus* Middleware

- Sistemas de middleware seguem um estilo arquitetônico específico.

- Muitos middleware adotam sistema arquitetônico baseado em objetos (CORBA, TIB/RENDEZVOUZ).

- Ideia principal:
  - Desenvolver sistemas de middleware que sejam simples de configurar, adaptar e personalizar conforme necessidade da aplicação.
  - Solução:
    - INTERCEPTORES

# Arquiteturas *versus* Middleware

## Interceptores

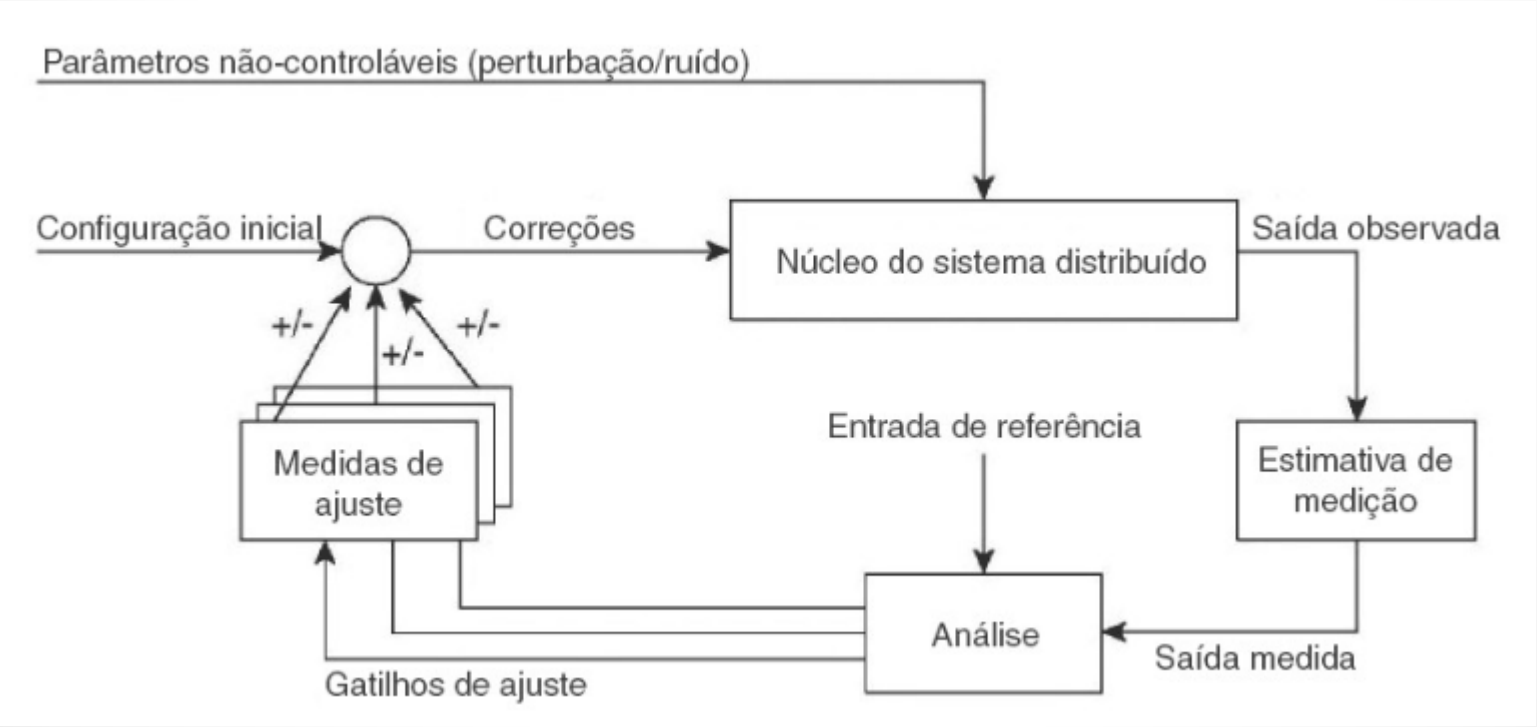
- Software que interromperá o fluxo de controle usual e permitirá que seja executado um outro código.
- Exemplo:
  - O objeto A chama um método do objeto B
  - A chamada original é transformada em uma chamada genérica
  - Se o objeto B é replicado, cada réplica deveria ser explicitamente invocada
  - Interceptor de nível de requisição:
    - replicar as chamadas

# Autogerenciamento em SD

- Ideia → fornecer soluções gerais de blindagem contra aspectos indesejáveis inerentes a redes
- Objetivo → suportar o maior número possível de aplicações
- Solução → SDs adaptativos
- Visão→ Construir sistemas onde seja possível fazer monitoramento e ajustes

# Autogerenciamento em SD

- Organização lógica de um sistema de realimentação de controle



# Exercícios e Bibliografia

- Este material tem sua fonte no livro:
  - Sistemas Distribuídos: Princípios e Paradigmas.
  - Andrew Tanenbaum e Marteen Steen

- Exercícios:

1. Fazer a leitura do capítulo 2(arquiteturas) do livro acima citado.
2. Fazer todos os exercícios do capítulo citado no item 1, com exceção dos exercícios referentes a **tarefas de laboratório**.