UNIFESSPA
Universidade Federal do Sul e Sudeste do Pará

Universidade Federal do Sul e Sudeste do Pará

Sistemas Distribuídos

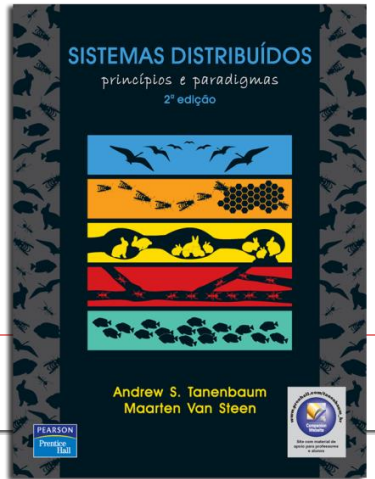
Prof.: Warley Junior
wmvj@unifesspa.edu.br

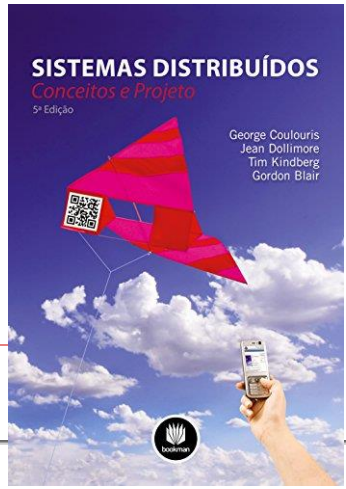
Agenda

- AULA 3
- Processos distribuídos
  - Threads distribuídas
  - Processos clientes
  - Processos servidores
  - Virtualização
  - Organização de Clientes e Servidores
  - Migração de código

Leitura Prévia

- COULOURIS, George. Sistemas distribuídos: conceitos e projetos. 5ª ed. Porto Alegre: Bookman, 2013.
  - Capítulo 3.
- TANENBAUM, Andrew S. Sistemas distribuídos: princípios e paradigmas. 2ª ed. São Paulo: Pearson Prentice Hall, 2007.
  - Capítulo 3.

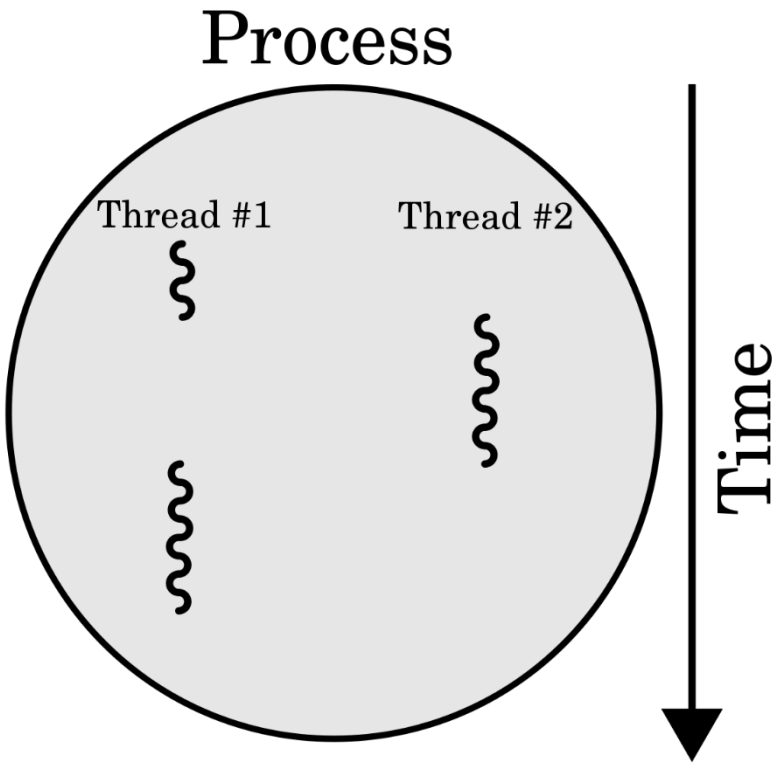




Revisando processos e threads

- Processo
  - São independentes
  - Espaços de endereçamento separados
  - Interage com outros processos por meio de IPCs
- Thread
  - Fluxos de execução dos processos
  - Compartilham o mesmo espaço de endereçamento e alguns dados da tabela de processo

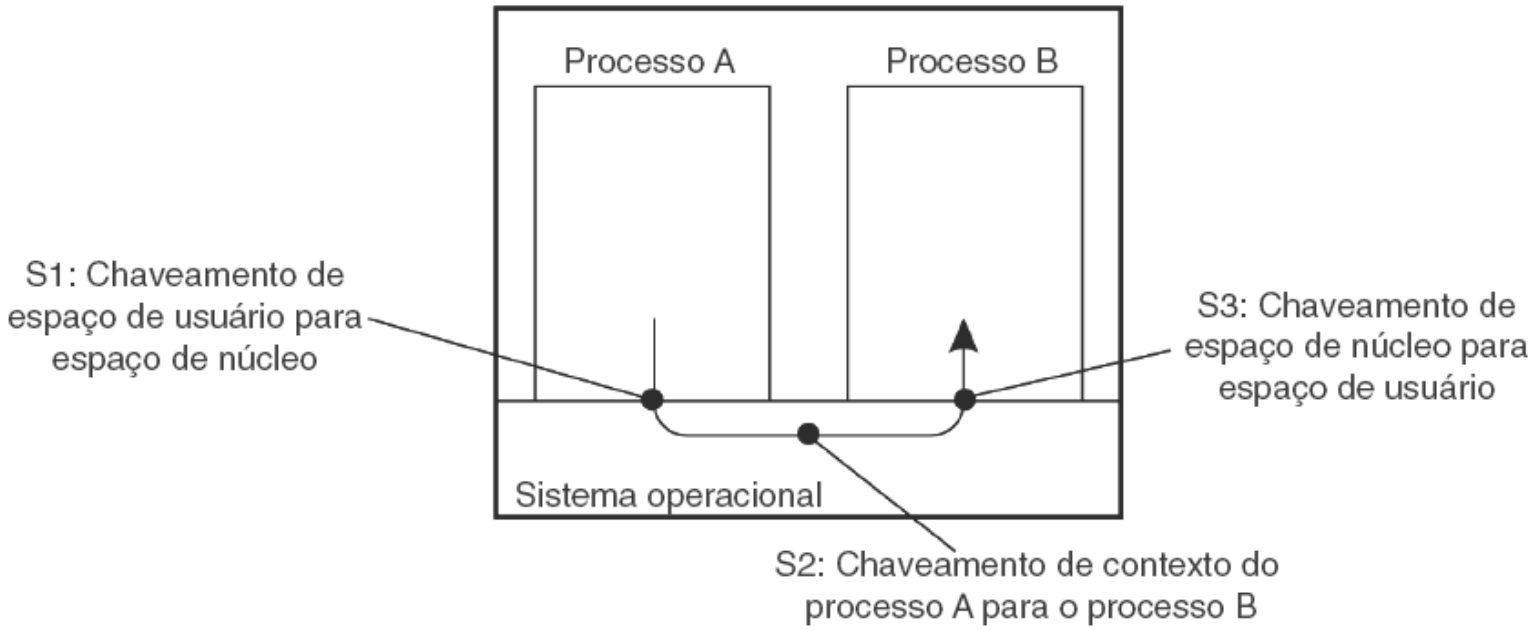
Revisando processos e threads



- Processo servidor de arquivos com um único fluxo faz uma requisição do disco e espera pelo resultado.
- O mesmo servidor com múltiplos fluxos pode atender a solicitações de outros usuários.

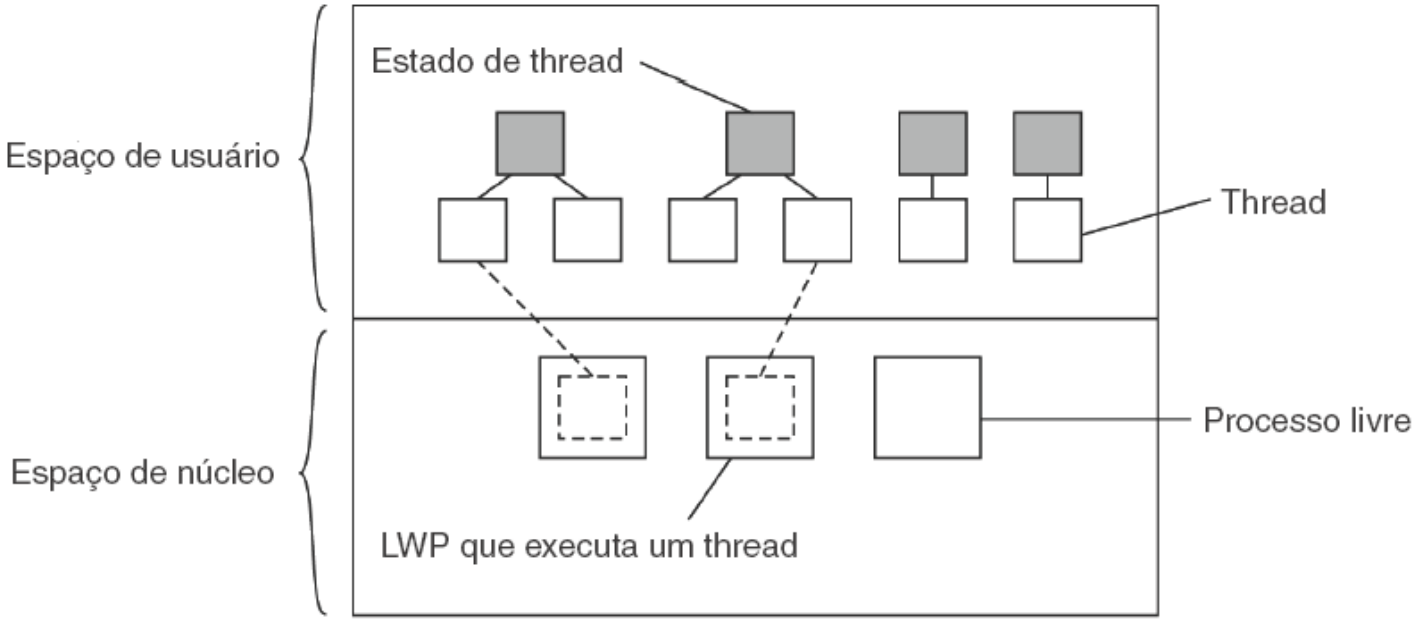
Processos e threads não-distribuídos

- Chaveamento de processo é um processo caro
- Tem que salvar e carregar as informações de contexto



Processos e threads não-distribuídos

- Threads Híbridas
  - *Lightweith Process* (LWP): cada conjunto de threads é mapeada em um número *n* de threads do kernel.



Threads em sistemas distribuídos

- Threads do kernel permitem chamadas bloqueantes sem bloquear todo o processo.
  - Vantagem das threads em SDs:
    - Múltiplas conexões, cada uma implementada por uma thread
    - Oculta a latência da comunicação na rede
    - Inicia a comunicação e realiza outra tarefa

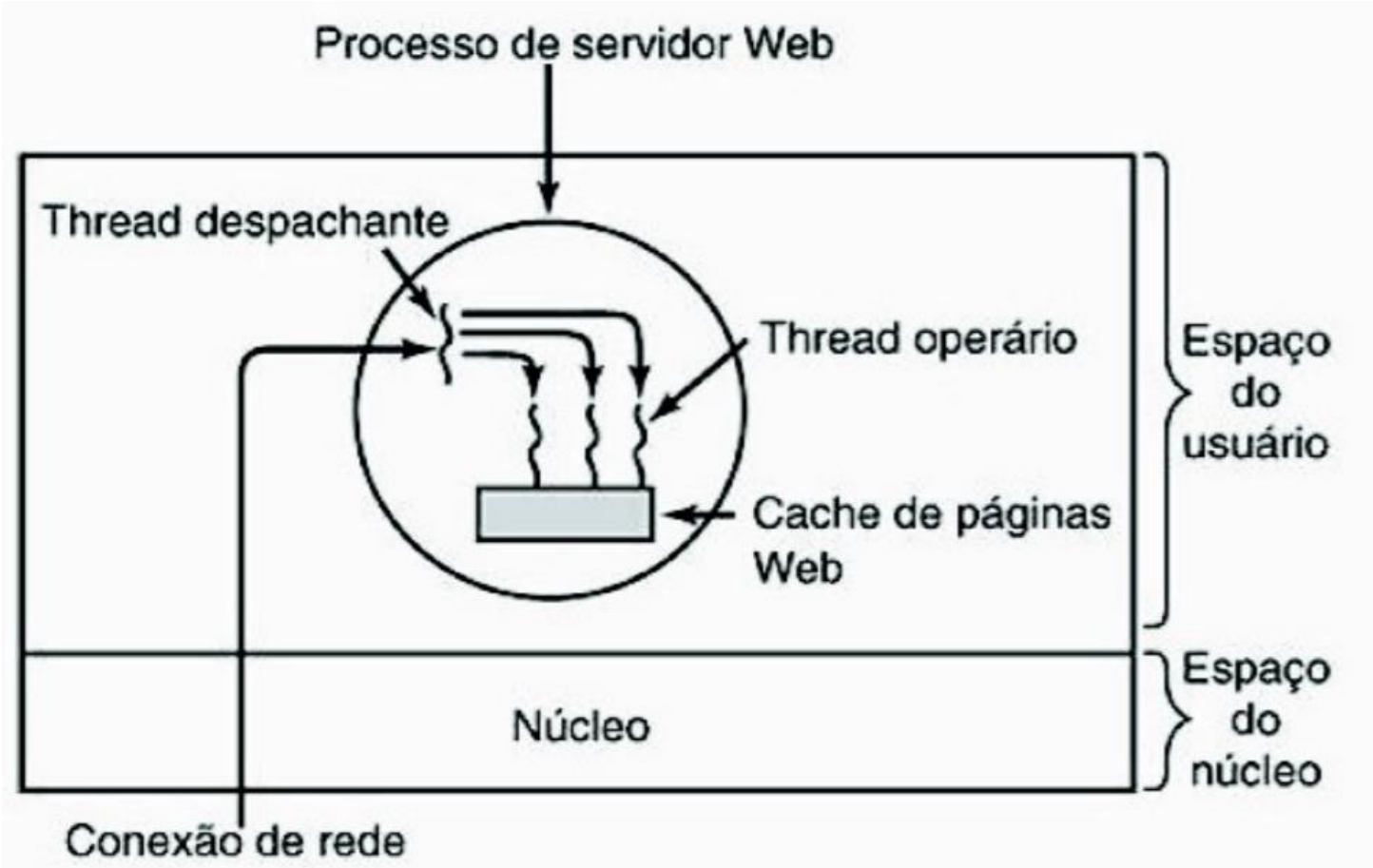
Clientes Multithreads Browsers Web

- Considerando uma conexão persistente com paralelismo
- Requisições são feitas sem que todos os objetos tenham chegados na máquina do cliente
- Cliente pode manipular diversos fluxos em paralelo usando as Threads
- Vantagem: Sem necessidade de esperar até que todos os componentes da página cheguem (transparência)

Servidores Multithreads

- Como threads podem melhorar o desempenho dos servidores?
- Funcionamento genérico dos servidores *multithreads*:
  - Cada requisição que chega passa por uma thread despachante
  - Servidor escolhe um thread operário
  - O thread despachante pode ser selecionado para fazer o trabalho

Servidores Multithreads



Um servidor Web multithread.

Servidores Multithreads

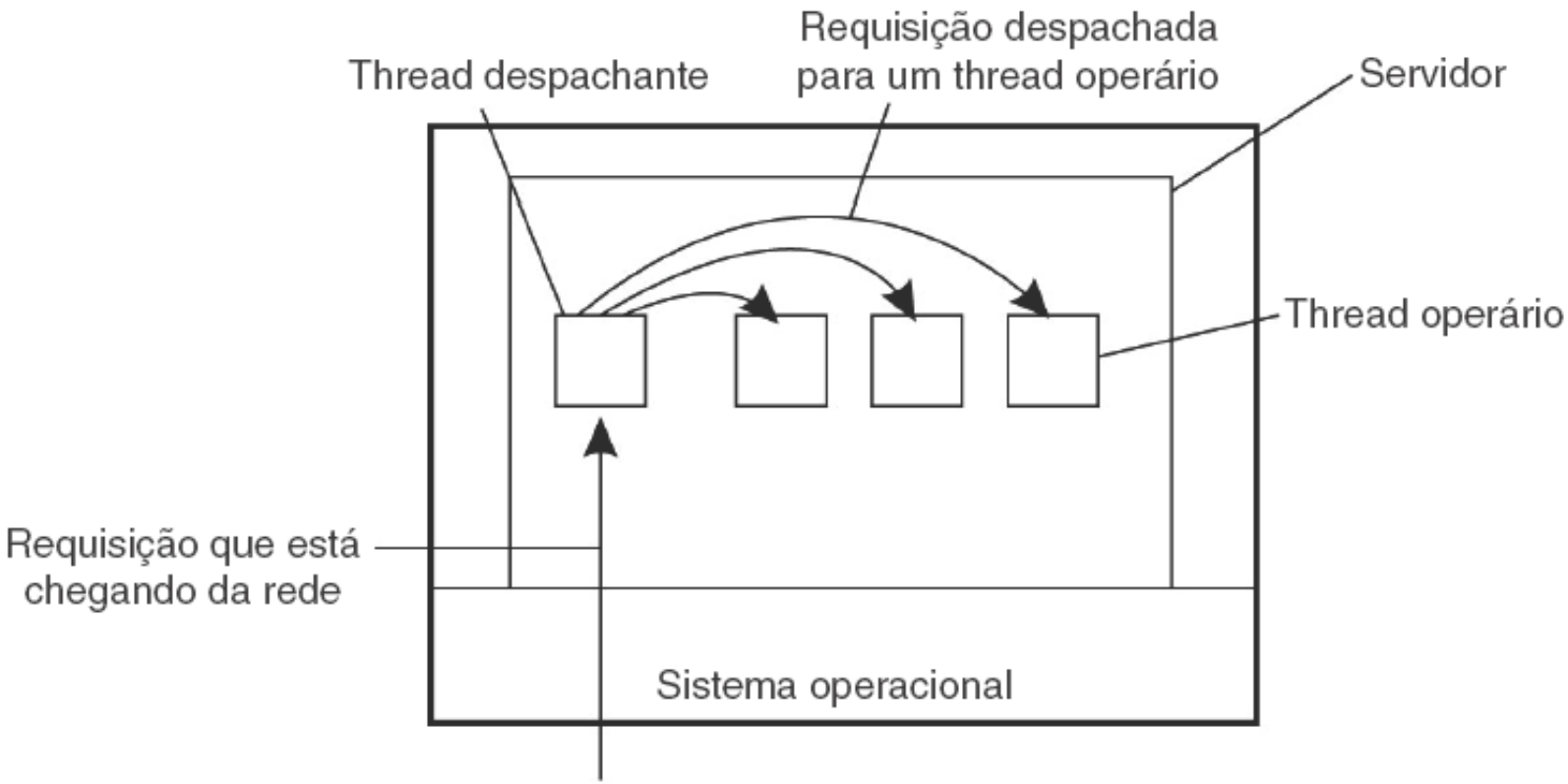
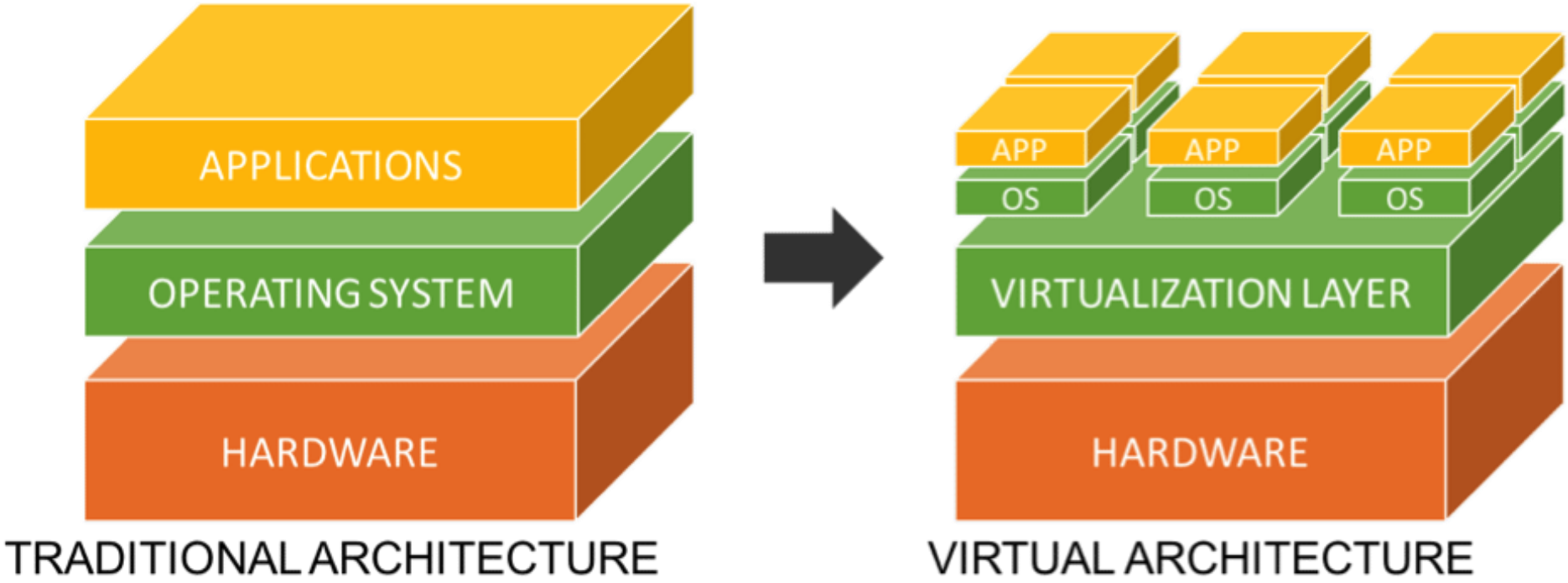


Figura 3.3 Servidor multithread organizado segundo modelo despachante/operário.

Virtualização

- Threads e processos – Dá a ilusão de fazer diversas tarefas ao mesmo tempo.
- Em computadores com uma CPU, a execução simultânea é uma ilusão:
  - Única CPU – somente uma thread ou processo será executada por vez
- Virtualização de recursos: "fingir" que um determinado recurso está replicado no sistema.

Virtualização



Virtualização

- Os sistemas de virtualização:
  - Estende ou substitui uma interface existente para imitar o comportamento de um outro sistema.

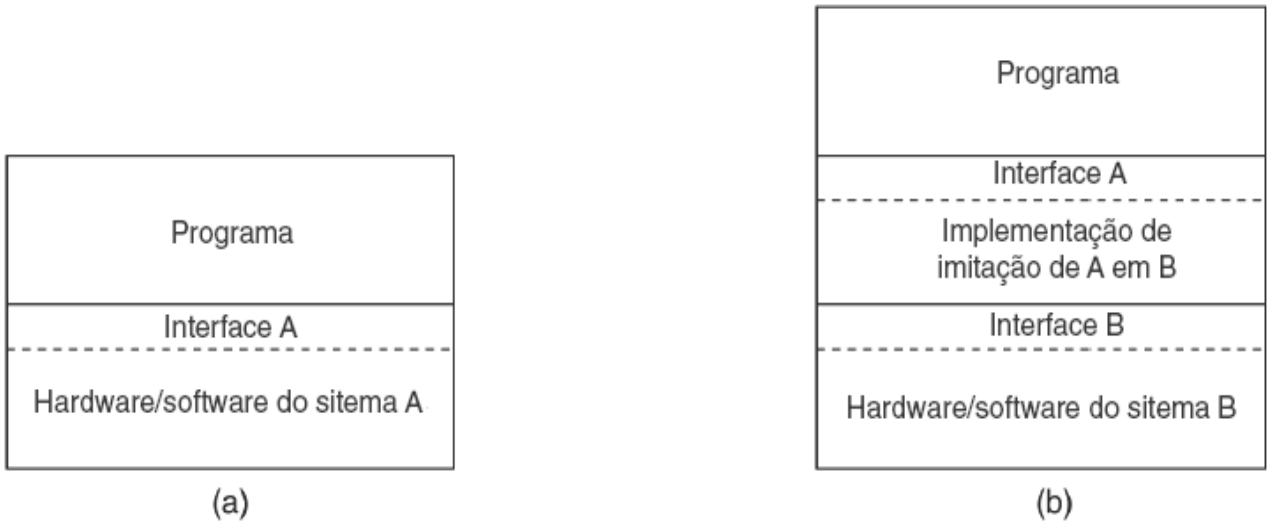


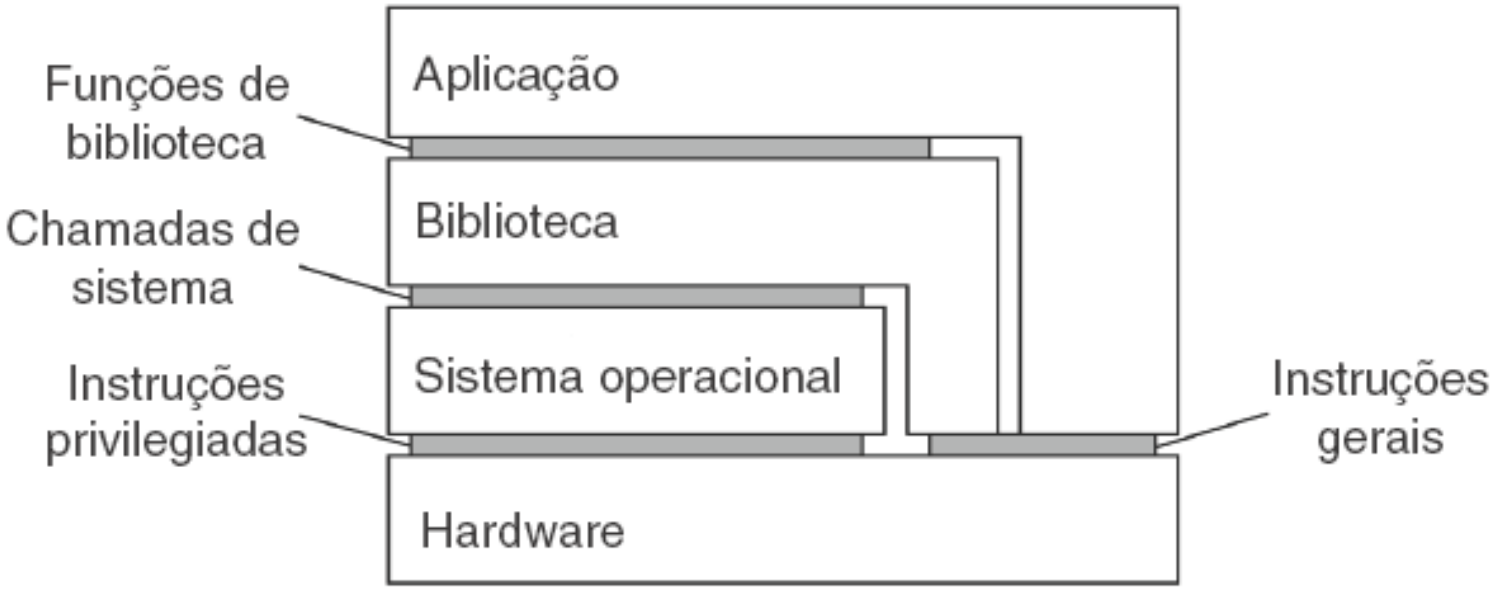
Figura 3.4 (a) Organização geral entre programa, interface e sistema. (b) Organização geral da virtualização do sistema A sobre o sistema B.

Virtualização - Portabilidade

- Softwares em nível mais alto são mais estáveis do que o hardware e sistemas de software de baixo nível.
- Virtualização (middleware) pode ajudar transportando as interfaces de softwares para novas plataformas.
- Novas plataformas são capazes de executar softwares existentes anteriormente.

Arquiteturas de máquinas virtuais

- Existem quatro tipos e níveis diferentes de interfaces



Virtualização

- Em resumo, é imitar o comportamento das interfaces (instruções de máquina, chamadas de sistema)
- Dois tipos:
  1) Máquina virtual de processo
  2) Monitor de máquina virtual

Máquina virtual de processo

- Aplicações desenvolvidas para um SO são executadas em outro SO
- Virtualização feita somente para um único processo
- Exemplo: wine
  - Execução de aplicações Windows no Linux Ubuntu

Monitor de máquina virtual

- Fornece o conjunto de instruções completo do hardware
- Vários SOs diferentes executando independente e concorrentemente na mesma plataforma
- Segurança: isolamento de uma aplicação e seu ambiente
- Exemplos:
  - VMWare e VirtualBox

Virtualização

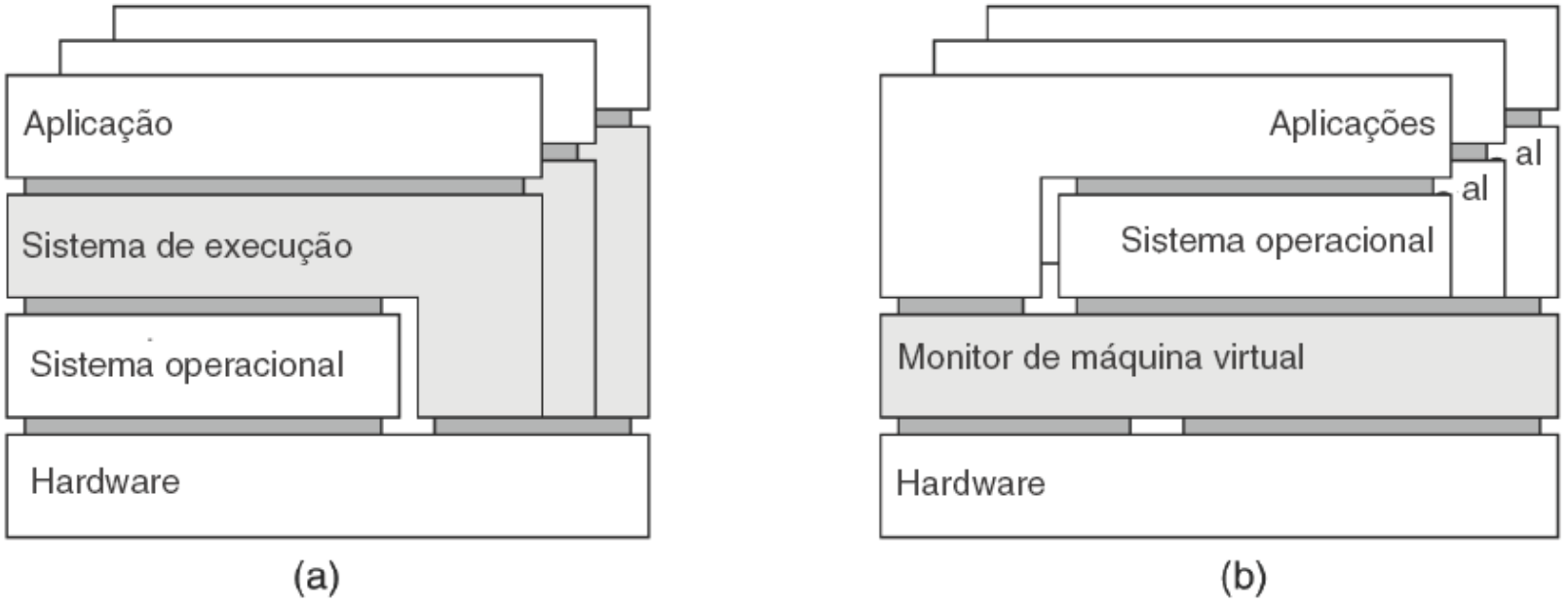
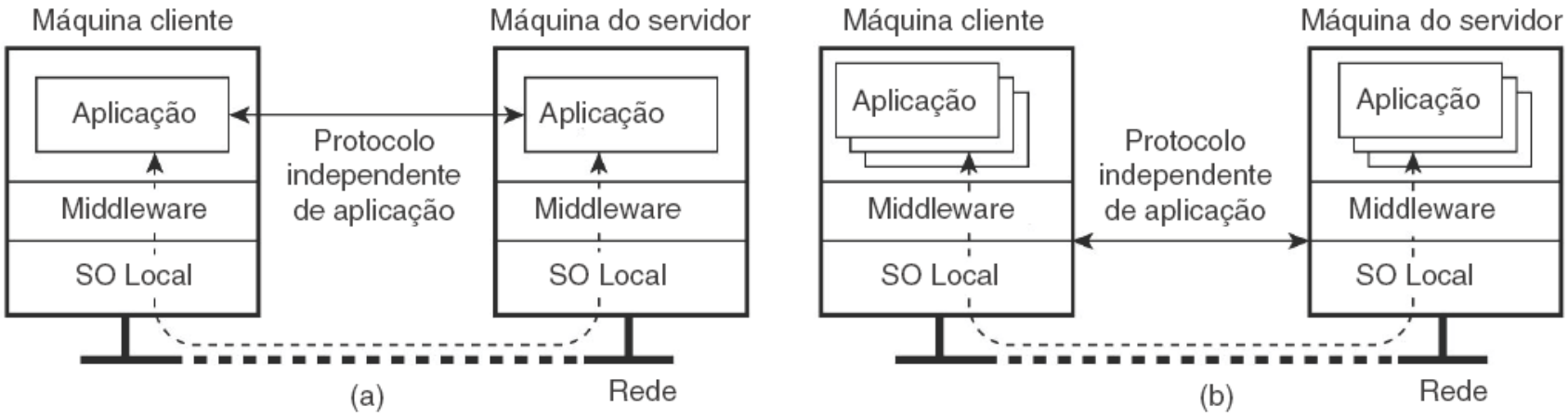


Figura 3.6 *(a) Máquina virtual de processo, com várias instâncias de combinações (aplicação, execução). (b) Monitor de máquina virtual com várias instâncias de combinações (aplicações, sistema operacional).*

Clientes: modos de interação com o servidor

- (a) Protocolo de interação é implementado no nível da aplicação e é específico a uma aplicação.
- (b) Protocolo de interação é implementado no nível do middleware e é genérico.



Clientes: modos de interação com o servidor

- X Window é uma implementação de interface de usuário em rede
- Implementação de um cliente minimizado (*thin client*)

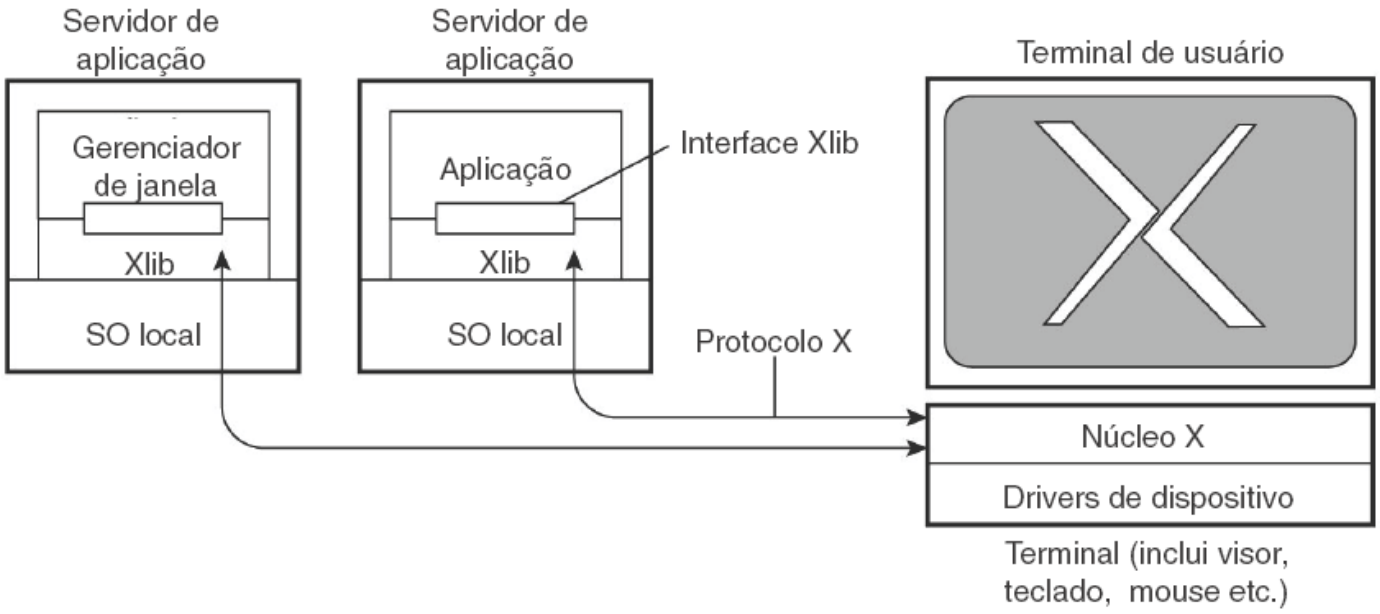


Figura 3.8 Organização básica do sistema X Window.

Clientes - Transparência

- Implementação no lado do cliente:
  - Transparência de Migração: O middleware no cliente oculta do usuário a localização do servidor, caso este migre para outro sítio
  - Transparência a falha: O middleware pode tentar a conexão com um servidor repetidas vezes
    - Uso de cache: falha na conexão com o servidor.

Clientes - Transparência

- Transparência de replicação: várias réplicas de requisição e uma resposta é enviada a aplicação.
- Desempenho?

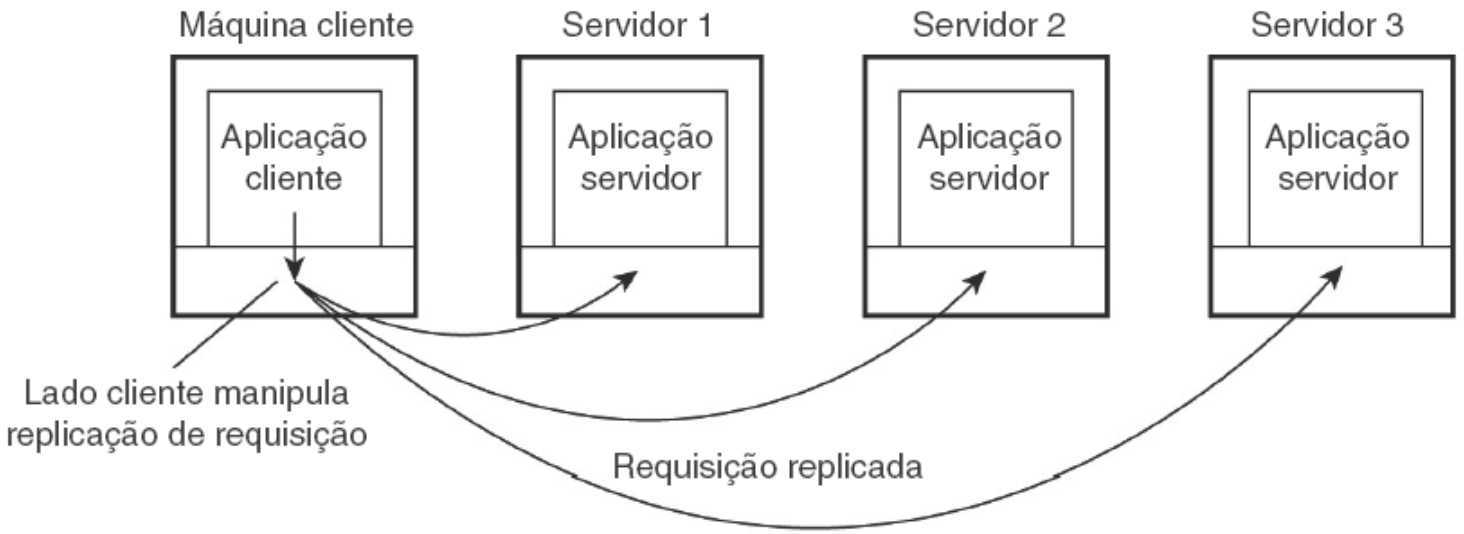


Figura 3.9 Replicação transparente de um servidor usando uma solução do lado cliente.

Servidores

- Como os clientes contatam um servidor?
- Como interromper a comunicação com o servidor?
- Servidor deve guardar estado da comunicação dos clientes?

Servidores: Como cliente contata?

- (a) Um daemon informa o # da porta
- (b) Um superservidor bifurca a chamada de uma porta para o servidor correto

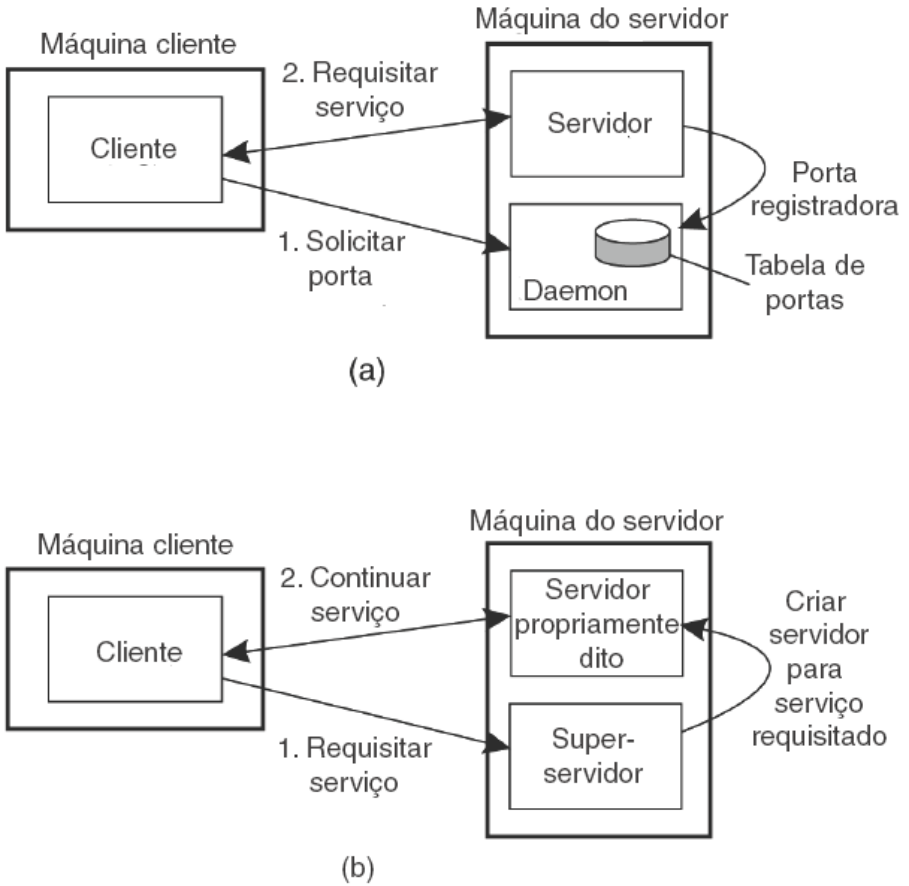


Figura 3.10 (a) Vinculação cliente–a–servidor usando um daemon. (b) Vinculação cliente–a–servidor usando um superservidor.

Servidores: Interrompendo a Comunicação

- Abordagem mais simples: usuário sai abruptamente da aplicação cliente.
- Abordagem mais completa: dados "urgente"
  - Pacotes TCP possuem o campo URG no header
  - Servidor ao receber um pacote com o campo URG setado é interrompido para trata-lo.
  - Ex.: Telnet

Servidores: Manutenção de Estado

- Três implementações:
  - Sem estado
  - Estado flexível
  - Com estado

Clusters de servidores

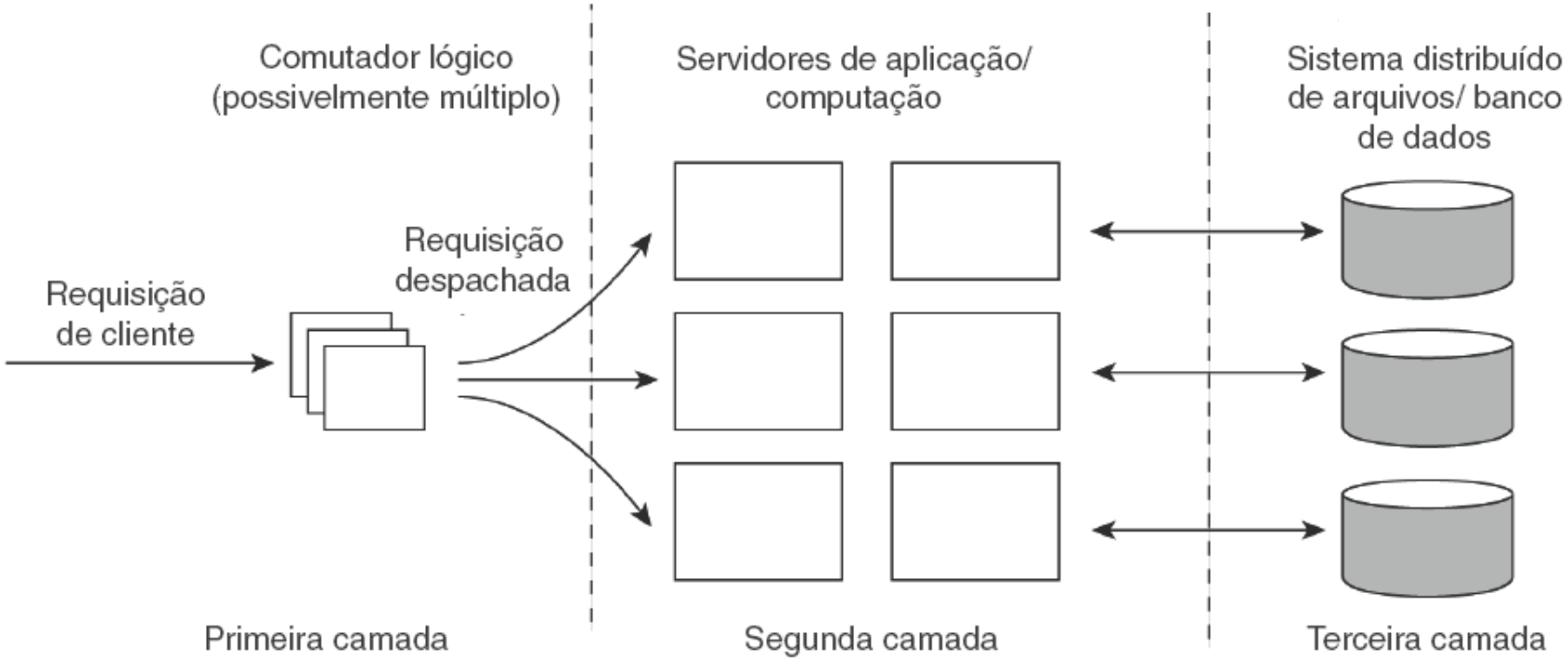


Figura 3.11 Organização geral de um cluster de servidores de três camadas.

Migração de Código

- Migrar códigos de uma máquina para outra
- Principal razão: aumento de desempenho
- Envio de processos para máquinas menos sobrecarregadas
- Evitar grande quantidade de mensagens trocadas entre aplicações cliente-servidor

Modelos para Migração de Código

- Mobilidade fraca: Transfere apenas o segmento de código e alguns dados de inicialização.
- Mobilidade forte: Segmento de execução (e.g. pilha, PC) também pode ser transferido.

Migração em relação ao ponto de início da migração

- Iniciada pelo remetente: A migração é iniciada na máquina em que o código está em execução.

- Iniciada pelo destinatário: A iniciativa da migração de código é tomada pela máquina-alvo.