Unidade 4

Programação Distribuída

- Introdução de Sistemas Distribuídos
- Modelos de Sistemas Distribuídos
- Comunicação entre Processos

Parte do material é retirado do Livro:
From Coulouris, Dollimore and Kindberg
Distributed Systems:
Concepts and DesignEdition 3, © Addison-Wesley 2001

Sistemas Distribuídos

- O que são?
  - São sistemas compostos por diversas partes cooperantes que são executadas em máquinas diferentes interconectadas por uma rede
  - É um sistema como uma coleção de computadores autônomos conectado por uma rede, com software projetado para produzir um aparato de computação integrado. São implementados em plataformas de hardware que variam em tamanho de algumas estações de trabalho interconectado por uma simples rede até a internet.

2

Sistemas Distribuídos

- "Você sabe que tem um sistema distribuído quando a falha de um computador do qual você nunca ouviu falar faz com que você pare completamente de trabalhar." [Leslie Lamport]

3

Sistemas Distribuídos

- Exemplos de aplicações distribuídas
  - WWW, ICQ, IRC, Morpheus/Kazza, etc.
  - Sistemas bancários
  - Sistemas de gerenciamento de redes de telecomunicações, transmissão de energia, etc.
  - Sistemas de informação de grandes empresas
  - ...

4

Sistemas Distribuídos

- Estrutura Física
  - Máquinas (*hosts*) são conectadas a um provedor (ISP) ou rede local, metropolitana, sem fio, etc.
  - Estas redes são interligadas por redes de longa distância públicas (ex.: Internet) ou privadas

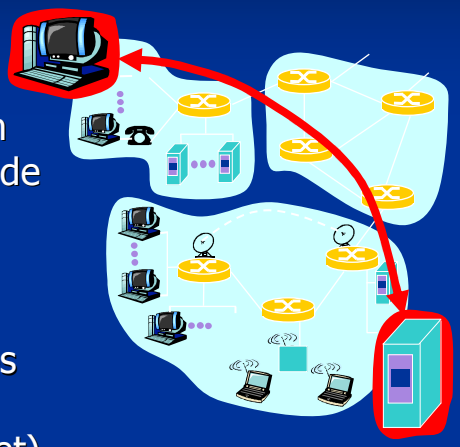


Figura 1.2
Um intranet típica

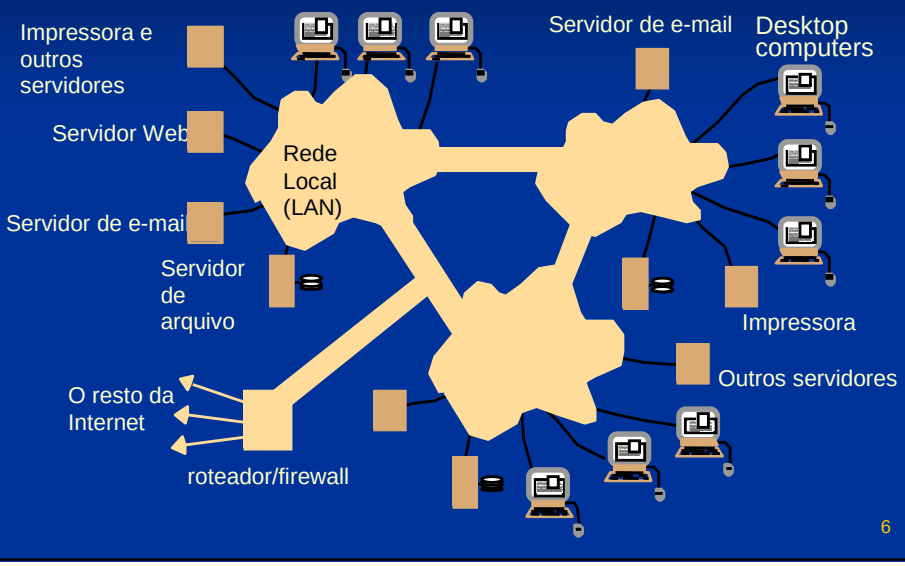
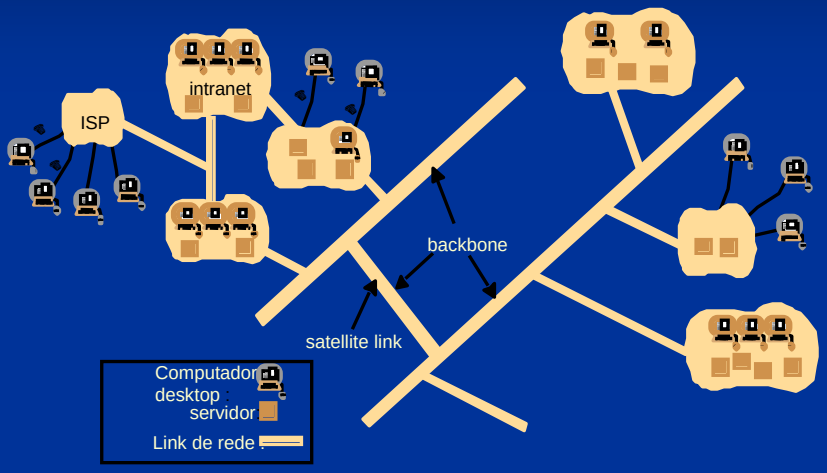
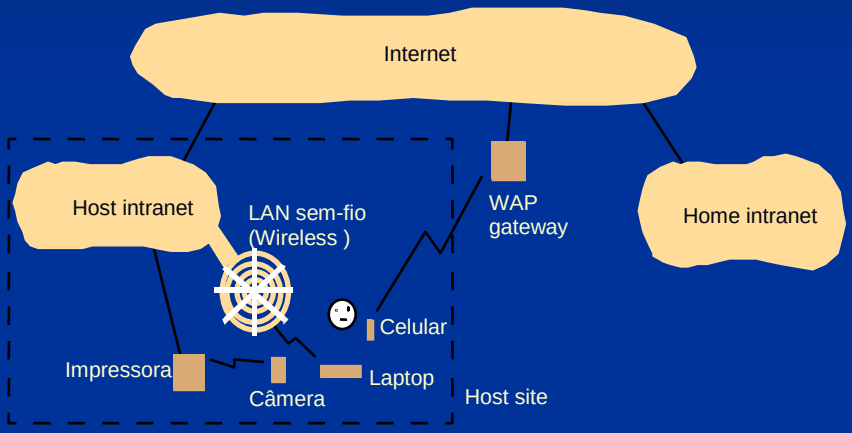


Figura 1.1
Um porção típica da Internet



7

Figura 1.3
Aparelhos portáteis e móveis em sistemas distribuídos



8

Vantagens de Sistemas Distribuídos em relação a Sistemas Centralizados

- **Custo/beneficio**: 10 máquinas de x 200 MHz (sucata) é mais barato que 1 máquina de 2 GHz
- **Capacidade de processamento (Velocidade)**: é possível construir sistemas com valor agregado muito maior
- **Maior domínio de aplicações:** TF, groupware, redes P2P, MSN, ICQ, Skype
- **Distribuição física**: algumas aplicações são essencialmente distribuídas (e.g., correio eletrônico)
- **Confiabilidade**: se uma máquina quebra, outras podem guardar backup
- **Disponibilidade**: se uma máquina sai do ar (ou melhor, do fio), pode-se usar outra

9

Vantagens de Sistemas Distribuídos em relação a Sistemas Centralizados

- Vantagens
  - Usam melhor o poder de processamento por distribuir a carga entre as máquinas
  - Podem apresentar melhor desempenho, maior confiabilidade, e suportar um maior número de usuários
  - Permitem compartilhar dados e recursos e reutilizar serviços já disponíveis
  - ...

10

Vantagens de Sistemas Distribuídos em relação a Sistemas Centralizados

- Dificuldades
  - Acoplamento fraco
    - Máquinas trocam dados pela rede
    - Tempo de comunicação ilimitado → pode comprometer o funcionamento do sistema
    - Como reduzir o tráfego na rede?
  - Não-deterministas
    - Comportamento pouco previsível
    - Como evitar congestionamento/sobrecarga?

11

Vantagens de Sistemas Distribuídos em relação a Sistemas Centralizados

- Dificuldades
  - Sistemas heterogêneos
    - Máquinas, linguagens e S.O.´s diferentes
    - Como tratar estas diferenças?
  - Não há uma base de tempo global
    - Em geral os relógios não estão sincronizados
    - Como ordenar os eventos no sistema?
  - Difícil ter uma visão global
    - Transações envolvem várias máquinas
    - Como manter o estado global do sistema?

12

Vantagens de Sistemas Distribuídos em relação a Sistemas Centralizados

- Dificuldades
  - Acesso concorrente a dados e recursos
    - Dados/recursos em diferentes máquinas
    - Como controlar o acesso concorrente?
    - Como evitar deadlocks?
  - Difícil gerenciar e manter o sistema
    - Máquinas dispersas pela rede
    - Administração pode ser independente

13

Vantagens de Sistemas Distribuídos em relação a Sistemas Centralizados

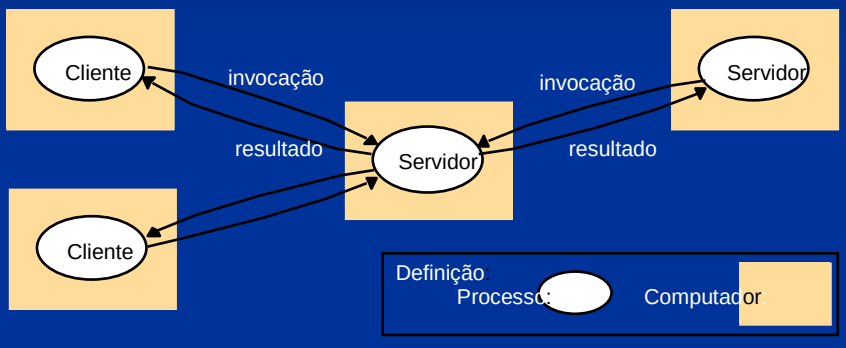
- Dificuldades
  - Sujeito a falhas
    - As máquinas e a rede podem falhar
    - Como evitar que estas falhas comprometam o funcionamento do sistema?
  - Segurança
    - Mais difícil controlar o acesso a dados e recursos em sistemas distribuídos
    - Como garantir a segurança do sistema e o sigilo dos dados trocados pela rede?

14

Figura 2.1
Camada de software e hardware em sistemas distribuídos

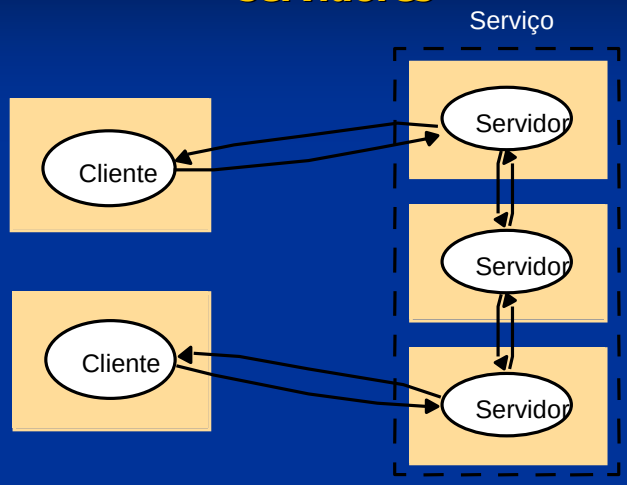
15

Figura 2.2
Clientes invocam servidores individuais



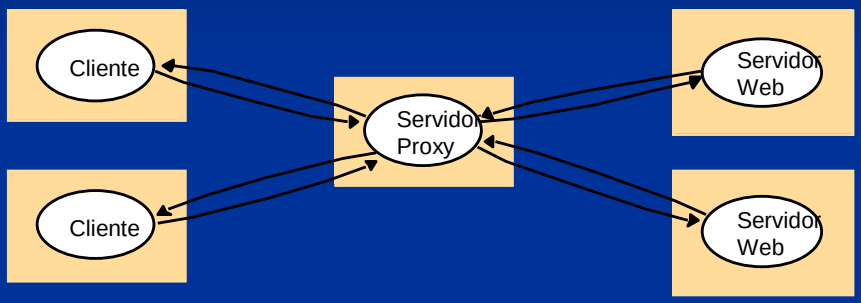
16

Figura 2.3
Um serviço fornecido por múltiplos servidores



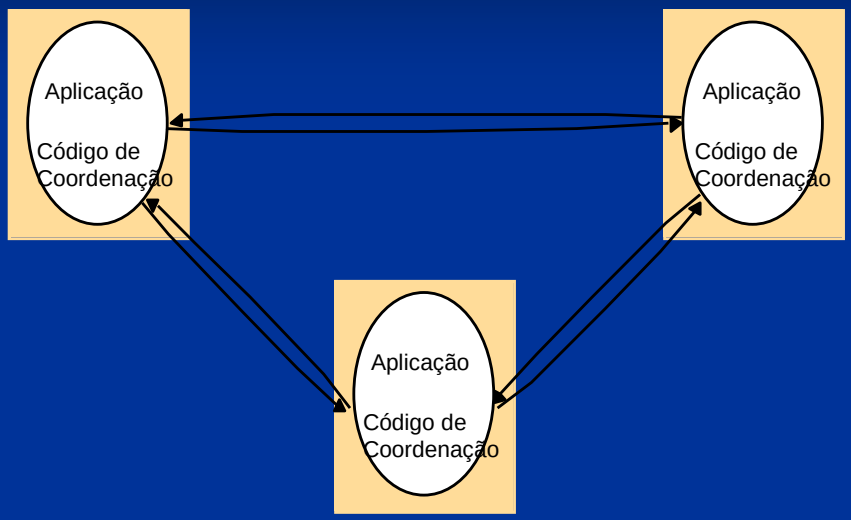
17

Figura 2.4
Servidor Web proxy



18

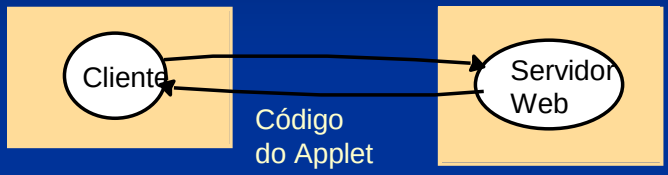
Figura 2.5
Uma aplicação distribuída baseada em pares de processo (P2P)



19

Figura 2.6
Applets Web

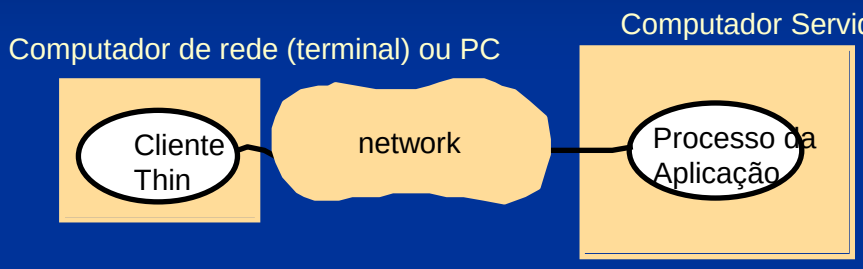
a) cliente requisita o download do código do applet



b) cliente interage com o applet

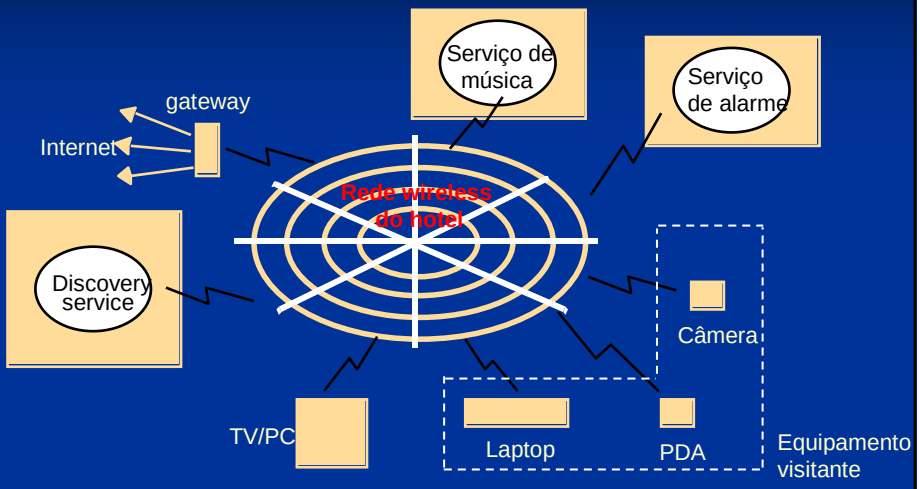
20

Figura 2.7
Cliente Thin and computador servidor



21

Figura 2.8
rede sem-fio (wireless) em um hotel



22

Rede e Comunicação entre Processos

- Parâmetros de desempenho
  - Latência (L): pode ser definido como o tempo requerido para transferir uma mensagem vazia entre dois computadores. É uma medida dos atrasos de software (atrasos de roteamento, elemento estatístico dependente da carga) envolvidos no acesso a rede no emissor e no receptor;

23

Rede e Comunicação entre Processos

- Parâmetros de desempenho
  - Taxa de transferência de dados (TTD): é a velocidade em que dados podem ser transferido entre dois computadores na rede, uma vez a transmissão ter iniciada, cotada em bits por segundo (bits/s).

24

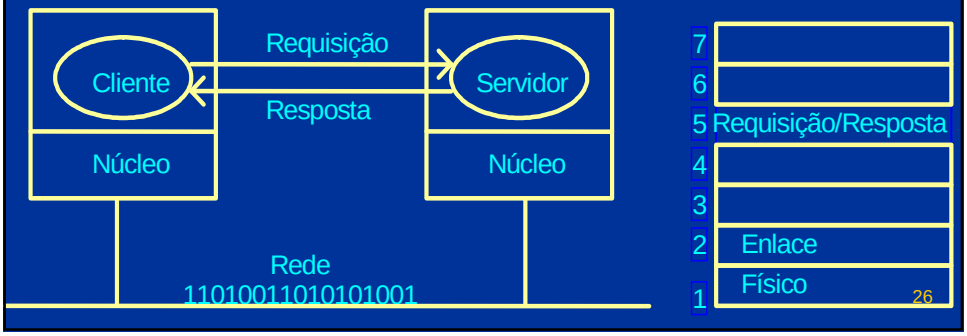
Comunicação entre Processos

- Parâmetros de desempenho
  - Tempo de transferência da mensagem (TTM): tempo requerido para transferir uma mensagem contendo um comprimento *length* (em bits) entre dois computadores:
    - TTM = (L + length) / TTD
  - Throughput (ritmo de transferência): mede a largura de banda da rede, o volume total de tráfego que pode ser transferido através da rede em um dado (intervalo) tempo.

25

Comunicação entre Processos

- Elementos básicos da comunicação
  - Transmissão
  - Endereçamento
  - Sincronismo
  - Enfileiramento (buferização)
  - Confiabilidade



26

Comunicação entre Processos

- Transmissão de dados
  - Dados em programas são *estruturados* enquanto que mensagens carregam informação *seqüencial*:
    - Linearização/Restauração de dados
  - Heterogeneidade na representação de dados em computadores:
    - Uso de um formato externo comum;
    - Inclusão de uma identificação de arquitetura na mensagem;

27

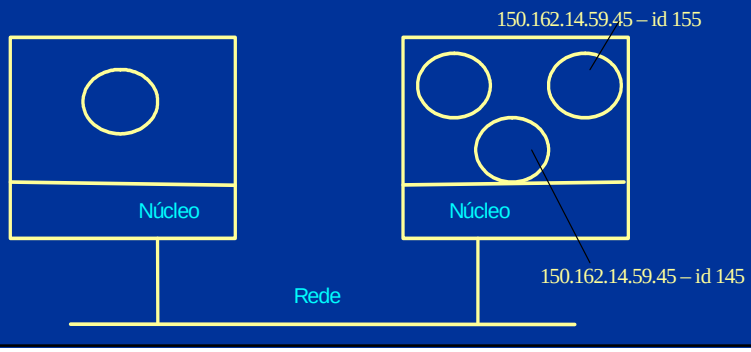
Comunicação entre Processos

- Transmissão de dados
  - Marshalling/Unmarshalling:
    - Marshalling:
      - Linearização de uma coleção de itens de dados estruturados;
      - Tradução dos dados em formato externo;
    - Unmarshalling:
      - Tradução do formato externo para o local;
      - Restauração dos itens de dados de acordo com sua estrutura;

28

Comunicação entre Processos

- Endereçamento
  - Esquemas:
    - Endereçamento máquina.processo;
    - Endereçamento máquina.id-local;
    - Endereçamento máquina.porta;



29

Comunicação entre Processos

- Endereçamento
  - Esquemas:
    - Descoberta de endereço via broadcasting (difusão);
    - Descoberta de endereço via um servidor de nomes;
    - Problemas potenciais: transparência de localização, sobrecarga, escalabilidade (componente centralizado).

30

Comunicação entre Processos

- Sincronismo
  - Comunicação síncrona:
    - Primitiva *send* é bloqueante: processo cliente aguarda enquanto o núcleo envia a mensagem para o processo servidor;
    - Primitiva *receive* é bloqueante: processo servidor aguarda até que o núcleo receba uma mensagem endereçada para aquele processo;

31

Comunicação entre Processos

- Sincronismo
  - Comunicação assíncrona:
    - Primitiva send é não-bloqueante: o processo cliente aguarda somente enquanto a mensagem é copiada para o buffer do núcleo;
    - Primitiva receive é não bloqueante: o processo servidor simplesmente comunica o núcleo que espera receber uma mensagem;

32

Comunicação entre Processos

- Enfileiramento
  - Situações:
    - Send ocorre antes de Receive;
    - Um cliente faz um Send enquanto o servidor ainda atende a outro cliente;
  - Solução trivial: clientes devem insistir ... ;
  - Solução pragmática: mailbox (uma fila de mensagens controlada pelo núcleo):
    - Mailbox criado a pedido do servidor;
    - Mensagens endereçadas ao mailbox;

33

Comunicação entre Processos

- Confiabilidade
  - Mensagens se perdem, atrasam, duplicam;
  - Abordagens:
    - Send tem semântica não confiável: as aplicações devem garantir entrega de mensagens (ex: timeout);
    - Mensagem de acknowledgement enviada pelo servidor (no nível núcleo);
    - Mensagem de acknowledgement implícita na resposta do servidor

34

Comunicação entre Processos

- Confiabilidade

Código	Tipo	De	Para	Descrição
REQ	Request	C	S	O cliente deseja um serviço
REP	Reply	S	C	Resposta do servidor para o cliente
ACK	Ackowledgment	*x*	y	O pacote anterior chegou
AYA	Are you alive?	C	S	Investiga se o servidor não parou
IAA	I am alive	S	C	O servidor não parou
TA	Try again	S	C	O servidor está lotado
AU	Address unknown	S	C	Nenhum processo está usando aquele endereço

35

Comunicação entre Processos

- Em uma certa comunicação cliente-servidor, o cliente envia as mensagens **REQ, REQ, AYA, AYA, ACK,** não necessariamente nesta ordem, enquanto que o servidor envia as mensagens **IAA, IAA, ACK, REP, TA**, também não necessariamente nesta ordem. Supondo que o meio de comunicação é confiável, isto é, nenhuma mensagem se perde, corrompe ou duplica, descubra uma ordem possível de envio dessas mensagens e faça um diagrama ilustrando a situação.

36