MODELOS DE SISTEMAS DISTRIBUÍDOS

Edeyson Andrade Gomes

www.edeyson.com.br

Agenda

- Visão geral
- Modelos de Arquitetura de Sistemas Distribuídos – MASD
  - Camadas de softwares
  - Arquiteturas de sistema
  - Variações
  - Interfaces e objetos
  - Requisitos de projeto para arquiteturas distribuídas
- Modelos fundamentais – MFSD
  - Modelo de interação
  - Modelo de falhas
  - Modelo de segurança

Visão Geral

VISÃO GERAL

- Um modelo de arquitetura de um sistema distribuído envolve o posicionamento de suas partes e os relacionamentos entre elas.

- Os modelos fundamentais envolvem uma descrição mais formal das propriedades comuns a todos os modelos de arquitetura.
  - Ausência de um relógio global.
  - A comunicação entre processos é por troca de mensagens.
  - Atrasos e vunerabilidade devido a rede de comunicação.

VISÃO GERAL

- Modelos fundamentais:
  - Modelo de interação
    - Trata do desempenho e da dificuldade de estabelecer limites de tempo.
  - Modelo de falha
    - Especificação das falhas de processos e canais de comunicação
  - Modelo de segurança
    - Apresenta o conceito de canal seguro.

VISÃO GERAL

- Cada modelo é destinado a fornecer uma descrição abstrata e simplificada, mas consistente, de um aspecto relevante do projeto de um sistema distribuído.

VISÃO GERAL

- Dificuldades e ameaças para os sistemas distribuídos:
  - Modos de uso que variam muito
    - Variações na carga. (Ex.: Web)
    - Interconexão. (Ex.: mal contato em cabos)
    - Largura de banda e latência. (Ex.: aplicações multimídia)
  - Ampla variedade de ambientes de sistema
    - Heterogeneidade de hardware, sistemas operacionais e rede.
  - Problemas internos
    - Relógios não sincronizados, conflito de dados, etc.
  - Ameaças externas
    - Ataques a integridade e ao sigilo dos dados, negação de serviço, etc.

Modelos de Arquitetura

Modelos de Arquitetura

- A arquitetura de um sistema é sua estrutura em termos de componentes especificados separadamente.
- O objetivo é atender as demandas atuais e futuras.
- Tornar o sistema confiável, gerenciável, adaptável e rentável.

Modelos de Arquitetura

- Simplifica e abstrai as funções dos componentes individuais.
  - Posicionamento dos componentes na rede de computadores.
  - Inter-relacionamentos entre os componentes.

- Classificação dos processos:
  - Processos cliente.
  - Processos servidor.
  - Processos *peer-to-peer*.

Modelos de Arquitetura

Camadas de SW

Camadas de SW

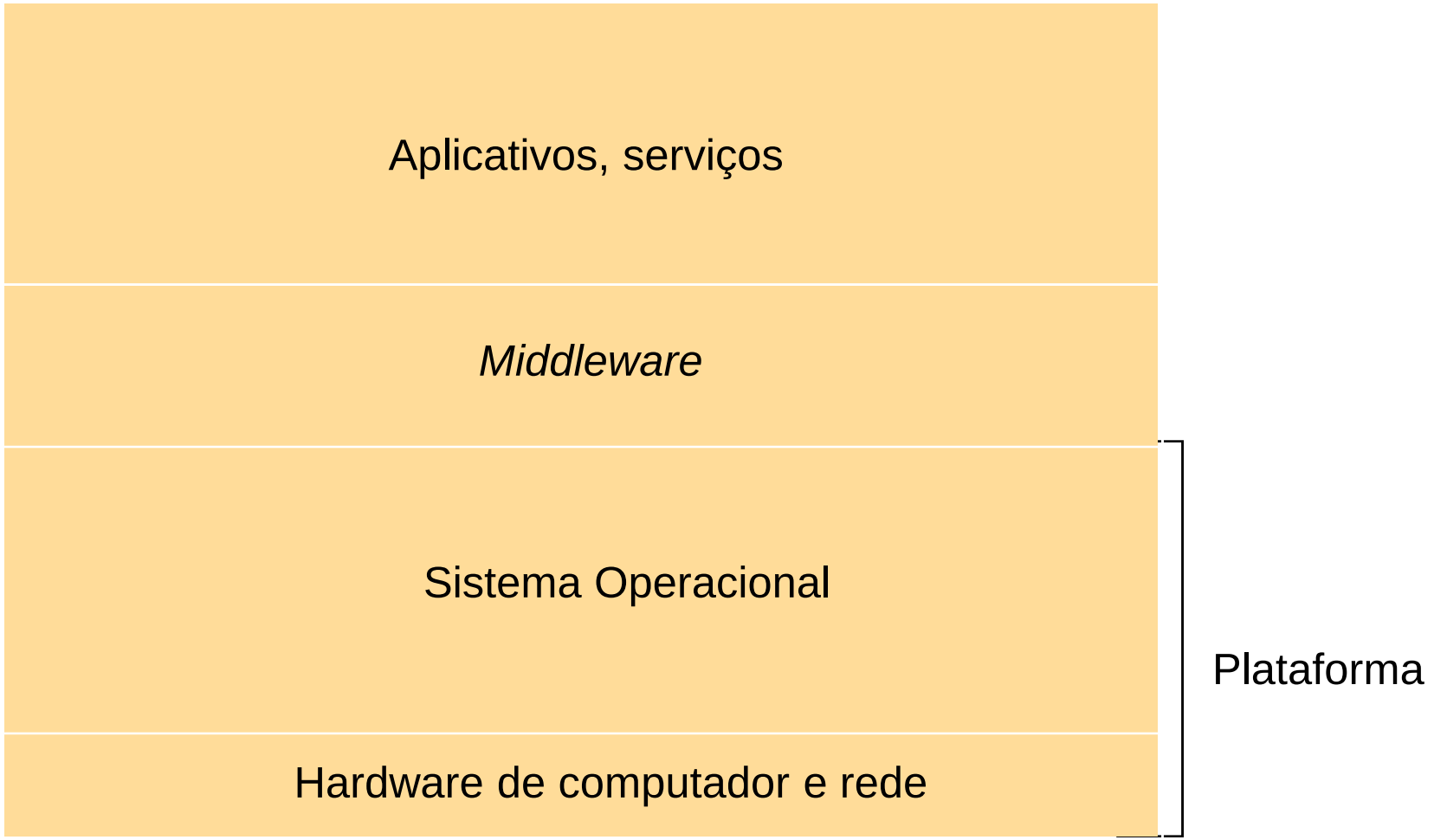
- A visão orientada para processo e serviço pode ser expressa em termos de camadas de serviço.
- Um servidor é um processo que aceita pedidos de outros processos.
- Um cliente é um processo que solicita algum serviço a um ou vários processos servidores.

Camadas de SW

- Um serviço é provido por um ou mais servidores.

- As camadas de hardware e software de mais baixo nível são freqüentemente denominadas de ***plataforma*** para sistemas e aplicativos distribuídos.
  - Ex.: Intel x86/Windows, Intel x86/Linux, PowerPC/Mac OS X, etc.

Camadas de SW



Camadas de SW - Middleware

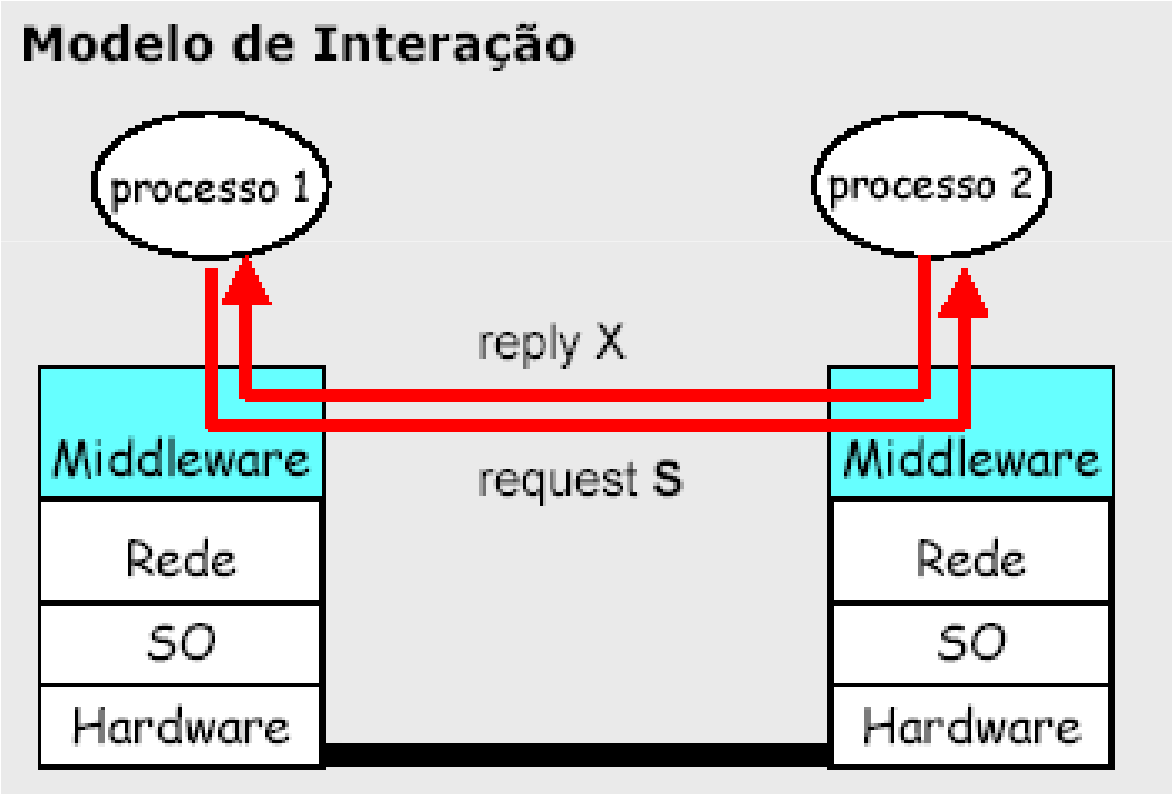
- Definições
  - Conjunto reusável e expansível de serviços e funções, comumente necessários por parte de várias aplicações distribuídas para funcionarem bem em um ambiente de rede
  - Mascara a heterogeneidade e fornece um modelo de programação homogêneo para os programadores de aplicativos.
  - Composto por um conjunto de processos ou objetos.

Camadas de SW - Middleware

- Definições
  - Simplifica as atividades de comunicação de programas aplicativos por meio do suporte de abstrações:
    - Invocação a métodos remotos.
    - Comunicação entre um grupo de processos.
    - Notificação de eventos.
    - Particionamento, posicionamento e recuperação de objetos de dados.
    - Replicação de objetos de dados compartilhados.
    - Transmissão de dados multimídia em tempo real.

Camadas de SW - Middleware

- Comunicação

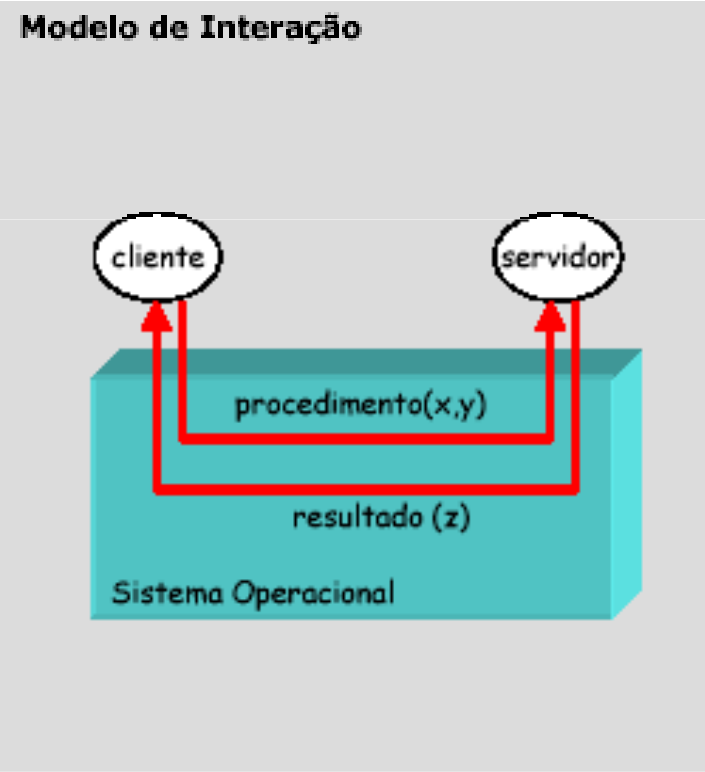


Camadas de SW - Middleware

- Produtos e padrões orientados a objetos
  - RPC – *Remote Procedure Call*
  - Java RMI – Java *Remote Method Invocation*
  - CORBA (*Object Management Group* – OMG) – *Common Object Request Broker Architecture*
    - Utiliza *Interface Definition Language* – IDL
  - DCOM (Microsoft) – *Distributed Component Object Model*
  - *Web Service* – Serviços Web
  - RM-ODP – *Reference Model for Open Distributed Processing* do *International Standards Organization* (ISO)/*International Telecommunication Union-Telecommunications* (ITU-T)

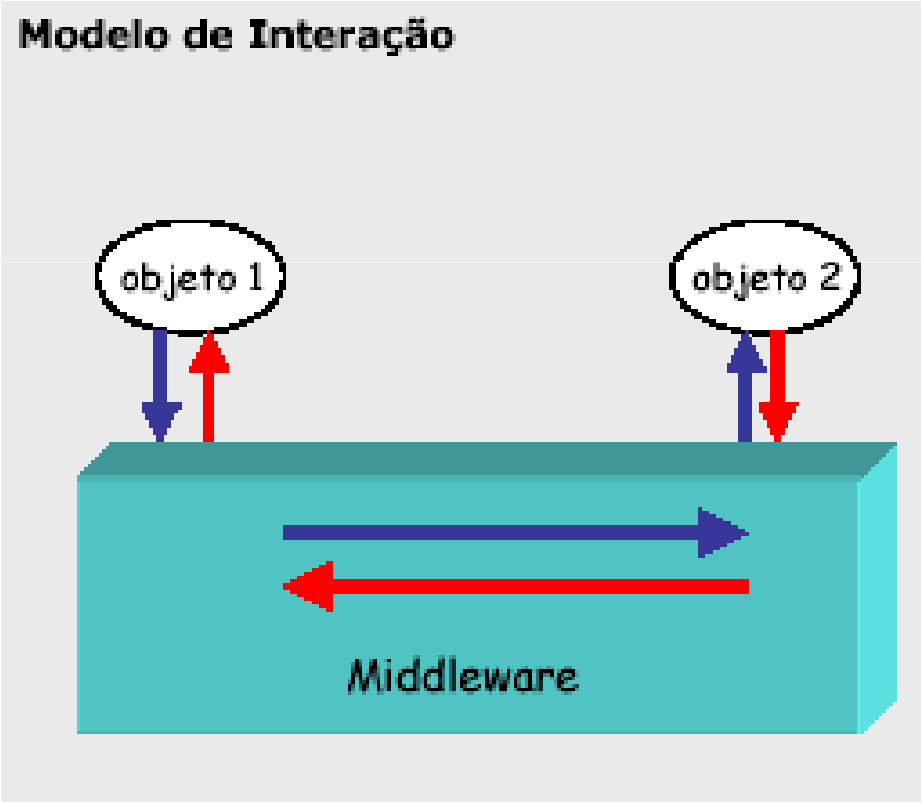
Camadas de SW - Middleware

- *Remote Procedure Call* – RPC



Camadas de SW - Middleware

- *Object Request Broker* – ORB



Camadas de SW - Middleware

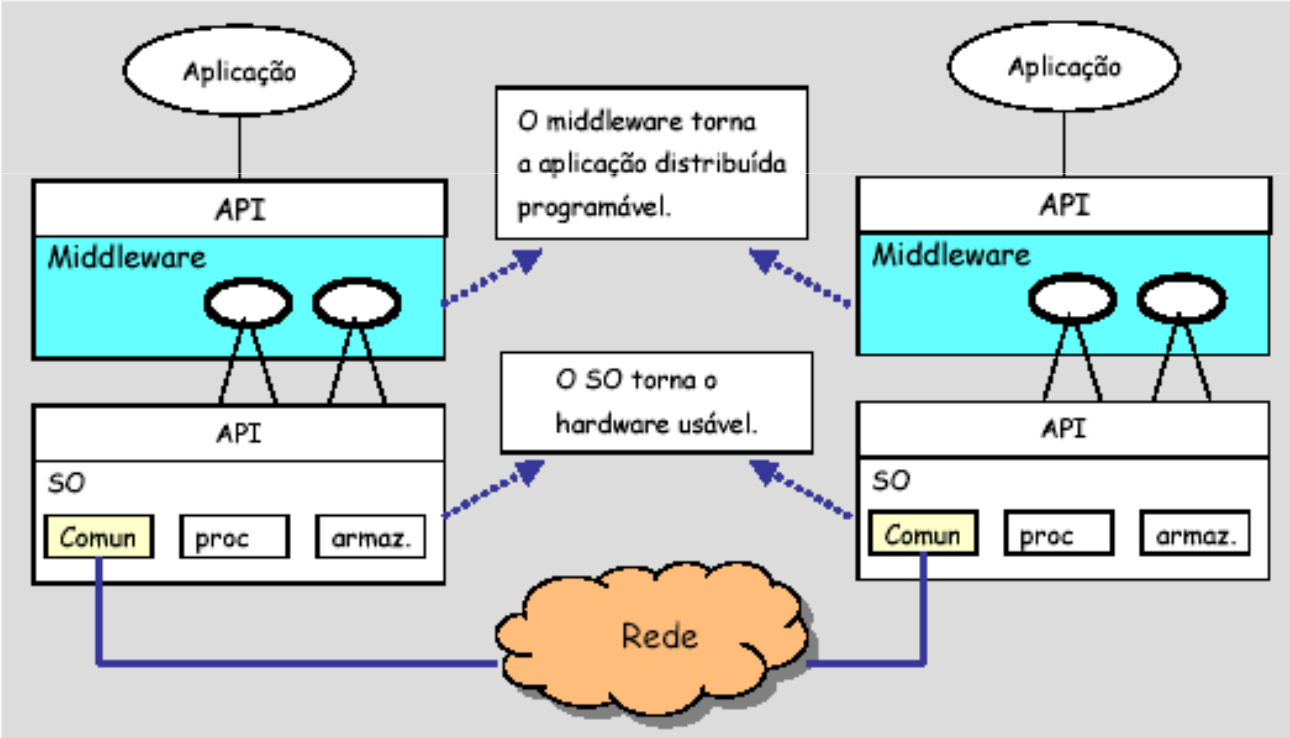
- Atribuição
  - *Object brokers*: permitem que objetos se encontrem em um sistema distribuído e interajam uns com os outros.
  - *Object services*: permitem criar, nomear, mover, copiar, armazenar, deletar, restaurar e gerenciar objetos.

Camadas de SW - Middleware

- Serviços oferecidos
  - Fornecem serviços de infra-estrutura a serem usados por programas aplicativos.
  - Os serviços são fortemente vinculados ao modelo de programação distribuída fornecido pelo *middleware*.
  - Exemplos de serviços do CORBA:
    - Atribuição de nomes (nomeação)
    - Segurança
    - Transações Distribuídas (ACID – Atomicidade, Consistência, Isolamento e Durabilidade)
    - Armazenamento persistente
    - Notificação de eventos

Camadas de SW - Middleware

- *Application Programming Interface* – API

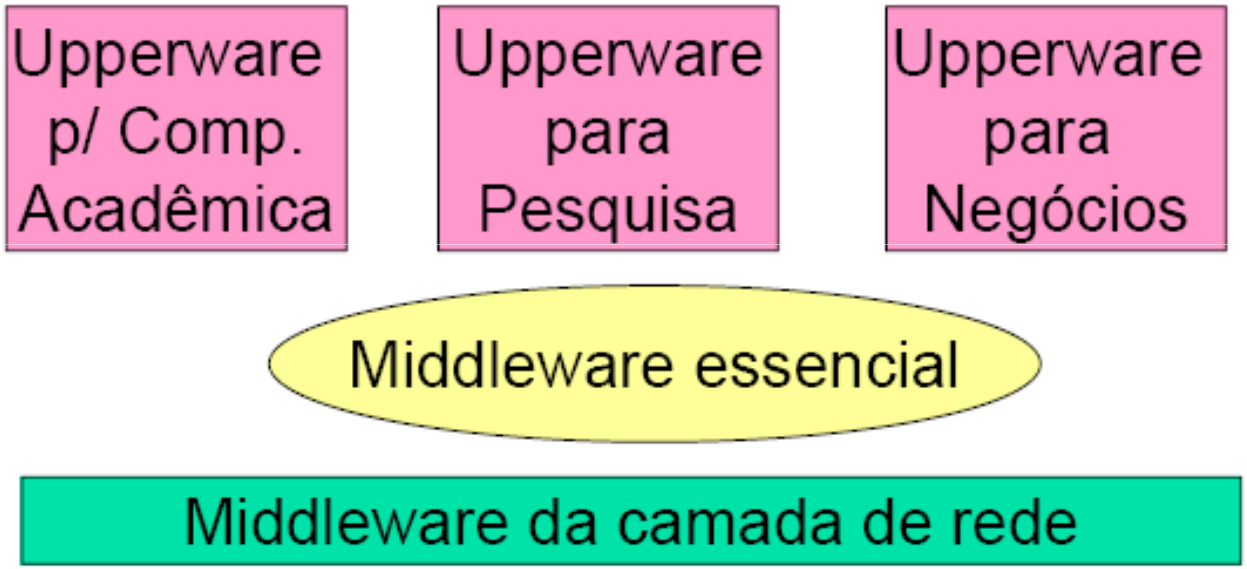


Camadas de SW - Middleware

- *Distributed Object Computing* – DOC
  - *Host Infrastructure*:
    - Ex.: *Java Virtual Machine* – JVM, .Net para *Web Services*, etc.
  - *Distribution*:
    - Ex.: CORBA, RMI, DCOM e SOAP (*Simple Object Access Protocol*)
  - *Commom Services*:
    - Ex.: Serviços CORBA, Sun's *Enterprise Java Beans* (EJB), .Net para *Web Service*, etc.
  - Obs.:
    - SOAP é um protocolo que troca mensagens em XML (e*Xtensible Markup Language*).
      - XML é uma recomendação da *Word Wide Web Consortium* (W3C).

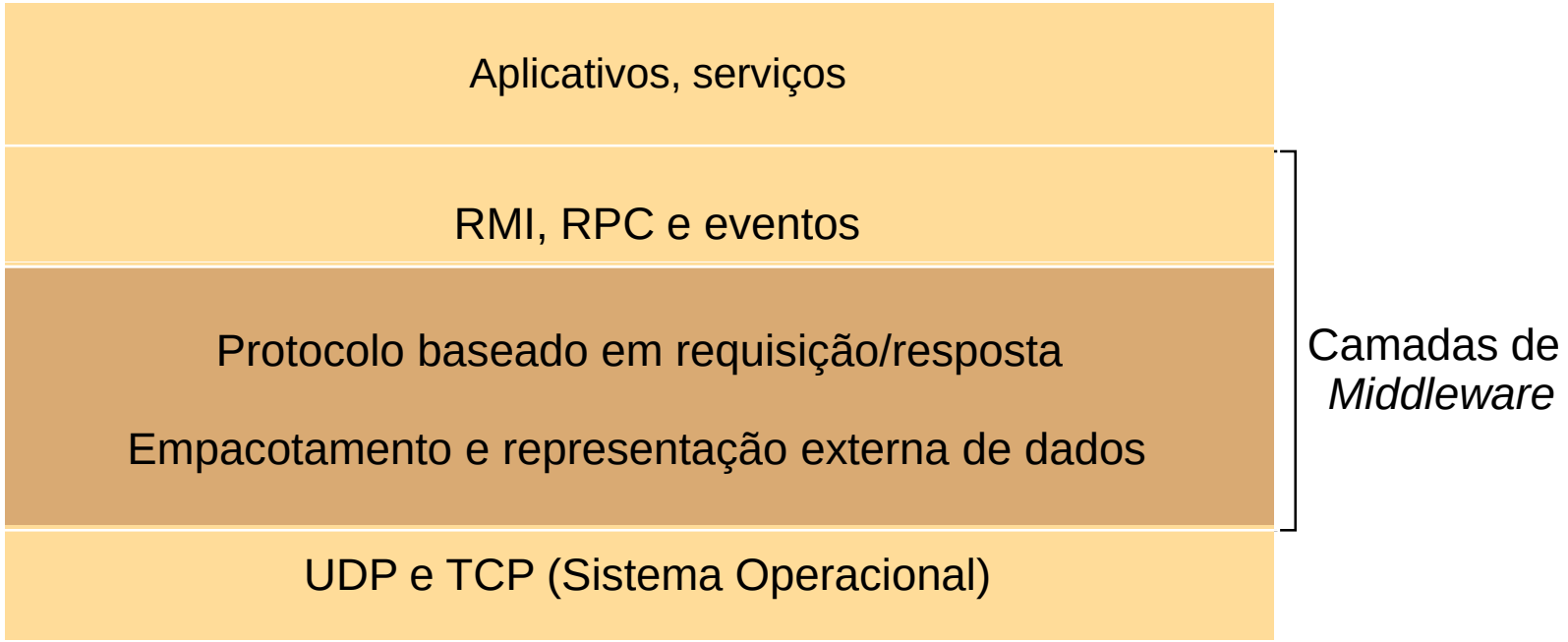
Camadas de SW - Middleware

- Camadas



Camadas de SW - Middleware

- Camadas



Camadas de SW - Middleware

- *L*imitações
  - Muitos aplicativos distribuídos se baseiam completamente nos serviços oferecidos pelo *middleware*.
  - Princípio fim-a-fim. (artigo de Saltzer, Reed e Clarke)
    - http://www.reed.com
    - Ex.: Serviço de e-mail sobre TCP/IP
  - O comportamento correto em programas distribuídos depende de verificações, mecanismos de correção de erro e medidas de segurança em muitos níveis.

Modelos de Arquitetura

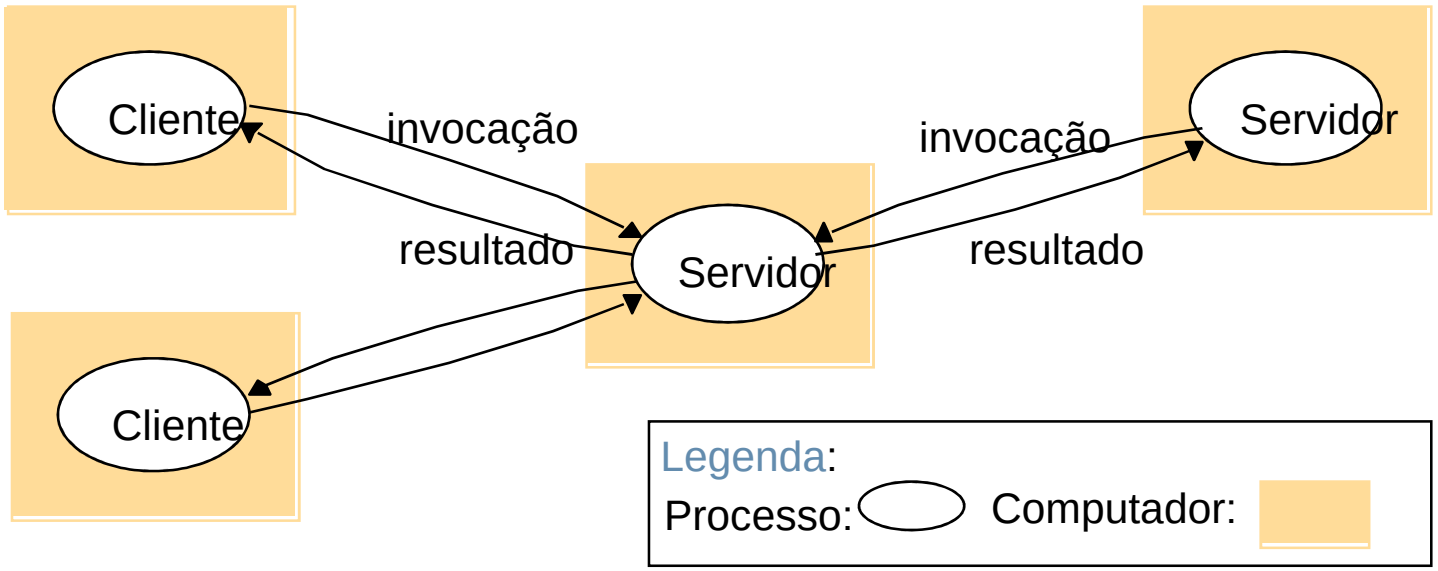
Arquiteturas de Sistemas

Arquitetura de Sistema

- Implicações importantes no desempenho, na confiabilidade e na segurança do sistema resultante.
- Os processos possuem responsabilidades bem definidas e interagem para realizar uma atividade útil.
- Dois tipos principais de modelos de arquitetura:
  - Cliente/Servidor
  - *Peer-to-Peer* (P2P)
- Variações nos modelos de arquitetura.

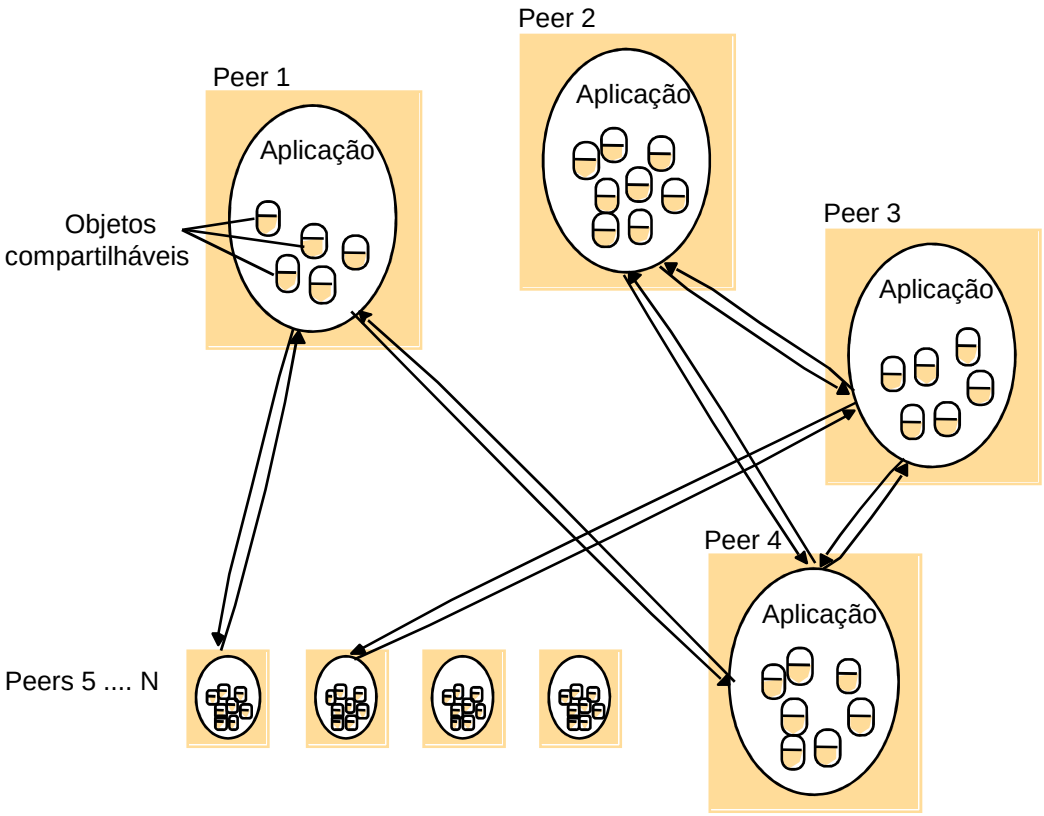
Arquitetura de Sistema

- Cliente/Servidor
  - Servidores também podem ser cliente.
    - Pouca necessidade de sincronização entre os processos cliente de uma operação e servidor de outra operação no mesmo servidor.



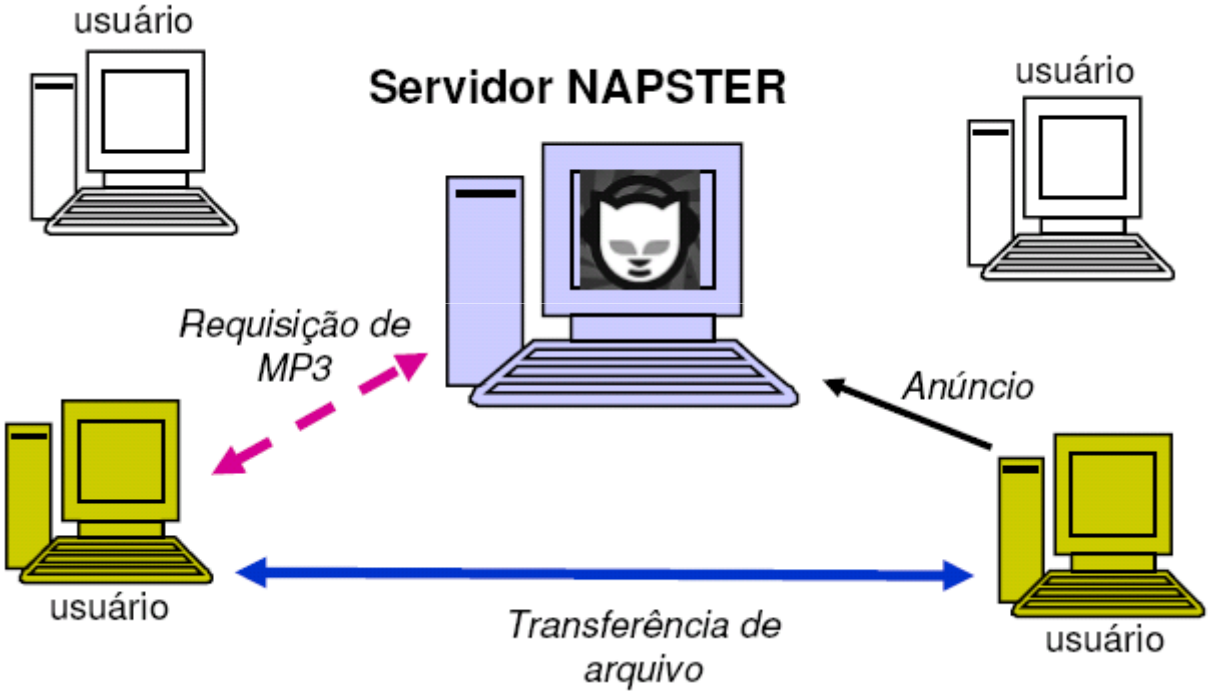
Arquitetura de Sistema

- *Peer-to-Peer*
  - Não há distinção entre processos clientes e servidores.
  - Maior escalabilidade comparando com o cliente/servidor.



Arquitetura de Sistema

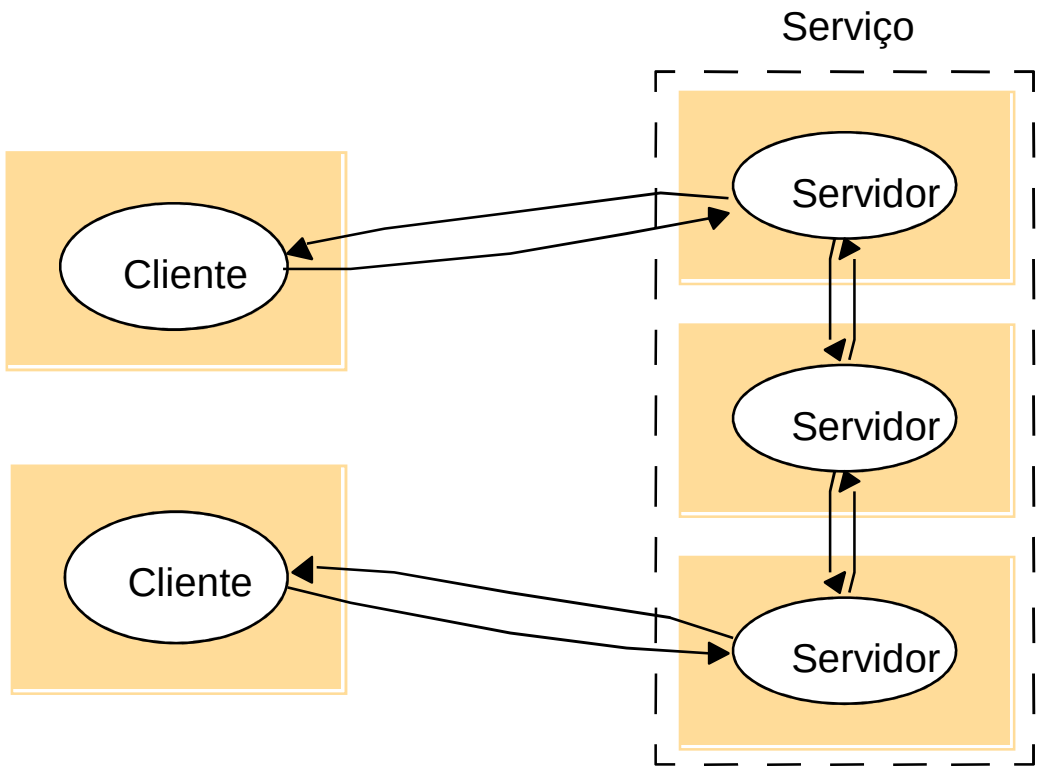
- *Peer-to-Peer* (ex.: Napster)



- Outros exemplos:
  - Emule, Bitorrent, Skype, MSN Messenger, Yahoo, etc.

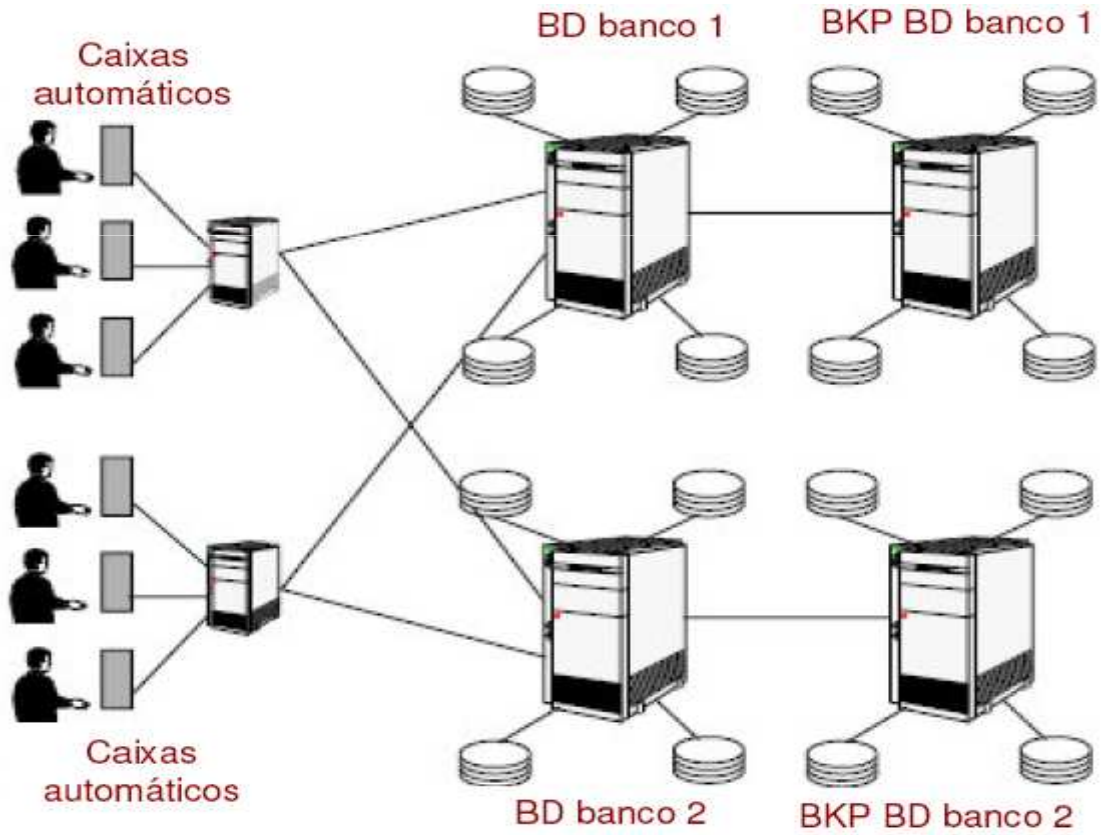
Arquitetura de Sistema

- Serviços fornecidos por vários servidores.
  - Exemplos: Web, NIS, bancos, *clusters*, etc.



Arquitetura de Sistema

- Serviços fornecidos por vários servidores. (Ex.: Bancos)



Arquitetura de Sistema

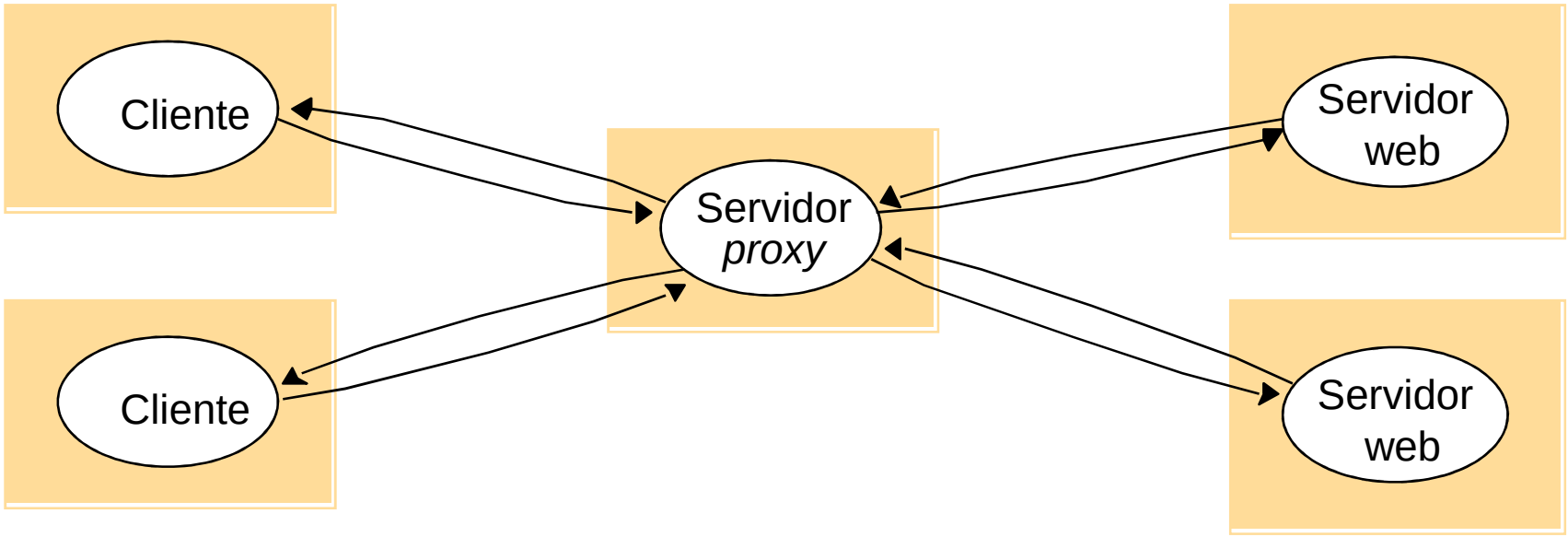
- Serviços fornecidos por vários servidores. (Ex.: *Clusters*)
  - Minimiza o custo de HW
  - Agrupamento físico
  - Pequeno espaço geográfico
  - Compartilhar recursos
    - De processamento
    - De armazenamento
    - De memória, etc.
  - SW e HW homogêneos
  - Redes de alta velocidade

Arquitetura de Sistema

- Serviços fornecidos por vários servidores. (Ex.: *Clusters*)
  - Aumentar a disponibilidade de serviços
    - Se um nodo falhar, outro assume.
  - Equilibrar carga de trabalho
    - Um ou mais computadores do cluster atuam como distribuidores da carga entre os demais
  - Alto desempenho
    - Para resolver tarefas complexas que podem ser decompostas em sub-tarefas, cada uma rodando num nodo do cluster.
    - Implementação mais comum: Linux e software livre para implementar paralelismo = *Beowulf cluster*

Arquitetura de Sistema

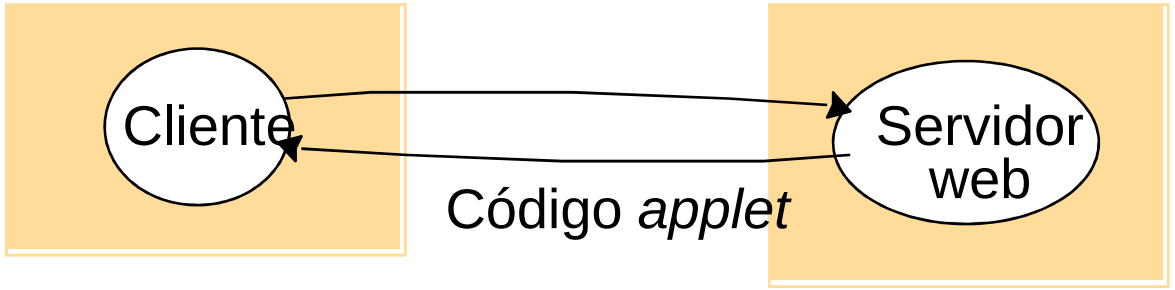
- Servidores *proxies* e Cache



Arquitetura de Sistema

- Código móvel

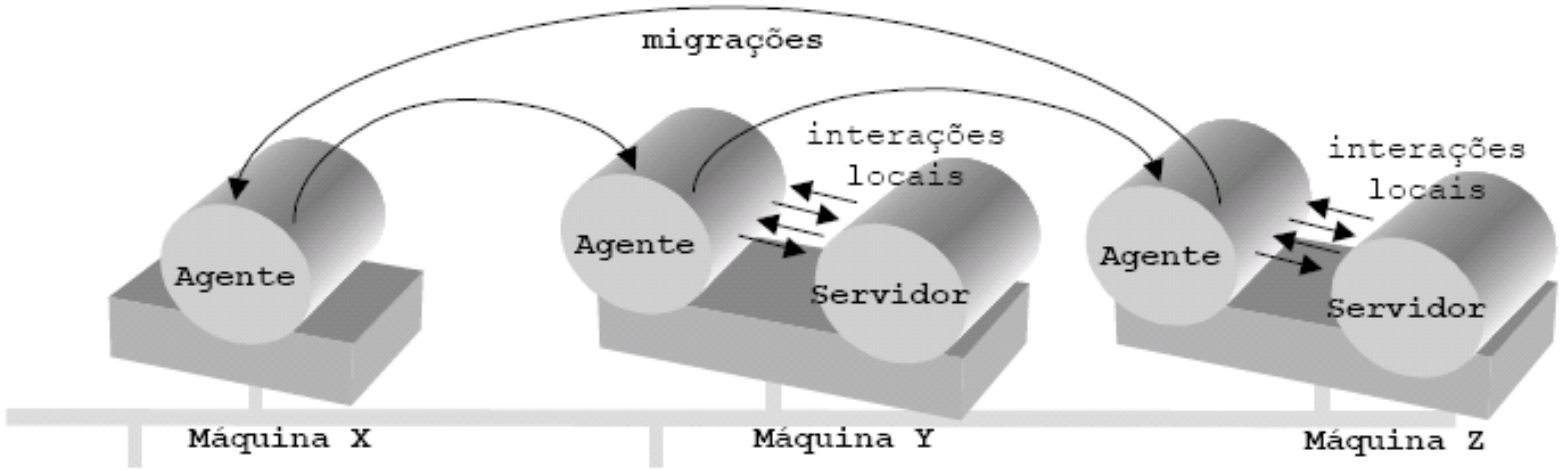
a) Requisição do cliente resulta no *download* de um *applet*



b) O cliente interage com o *applet*

Arquitetura de Sistema

- Agentes móveis
  - Programa em execução (código e dados).
  - Pode efetuar várias requisições aos recursos locais.
  - Reduz o custo e tempo da comunicação se comparado a C/S.
  - Ameaça em potencial a segurança.

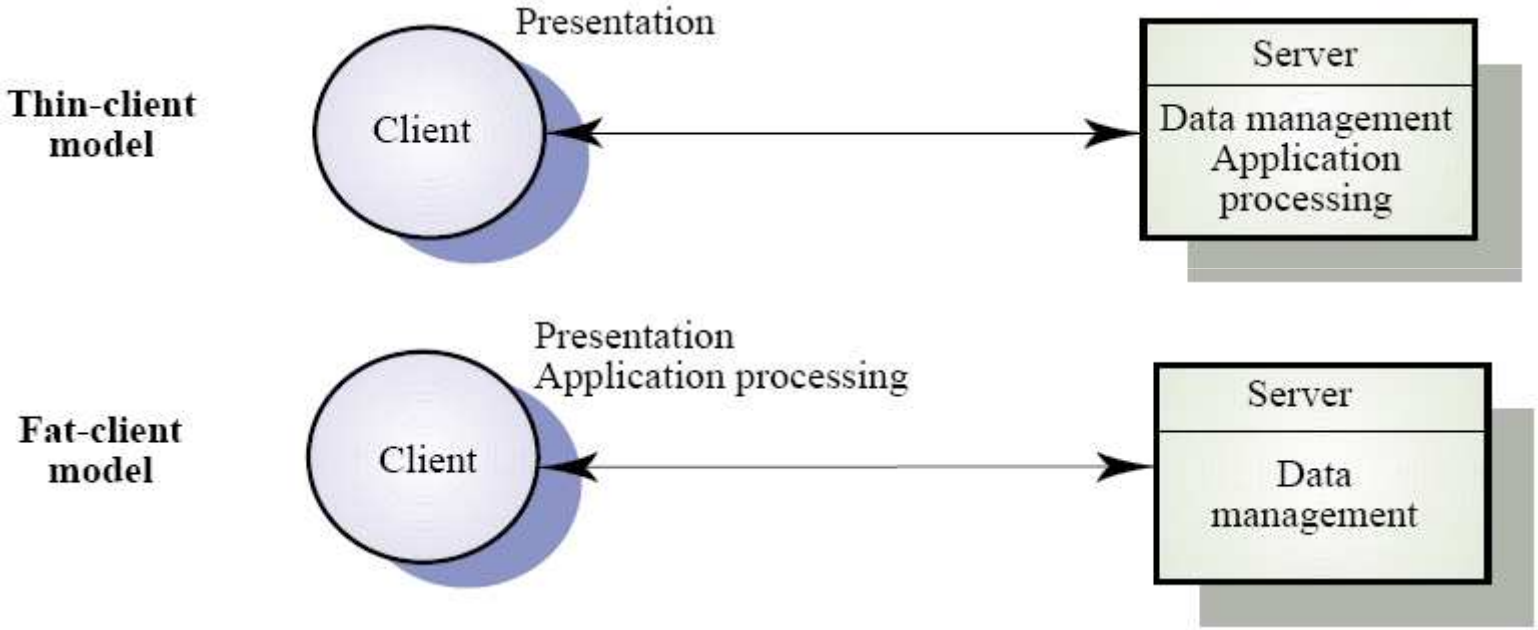


Arquitetura de Sistema

- Dispositivos móveis
  - *Laptops, Personal Digital Assistant* (PDA), celular, etc.
  - Redes cabeadas e redes sem fio (*Global System for Mobile* – GSM , 3G, Wireless, Bluetooth, etc.)
    - Conectividade variável
  - Transparência de mobilidade é um problema freqüente.
    - Interoperabilidade espontânea (descoberta de serviços)
  - *Global Positioning System* – GPS
  - *MobileIP* (IP Móvel)

Arquitetura de Sistema

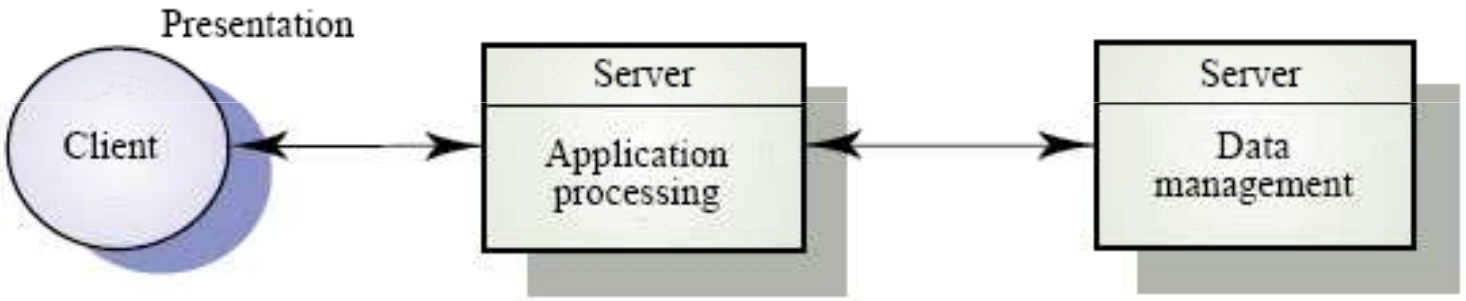
- Computadores de rede & clientes "leves"



- Obs.: Desktop é diferente de computadores em rede!

Arquitetura de Sistema

- Cliente/Servidor 3-*Tier*



Modelos de Arquitetura

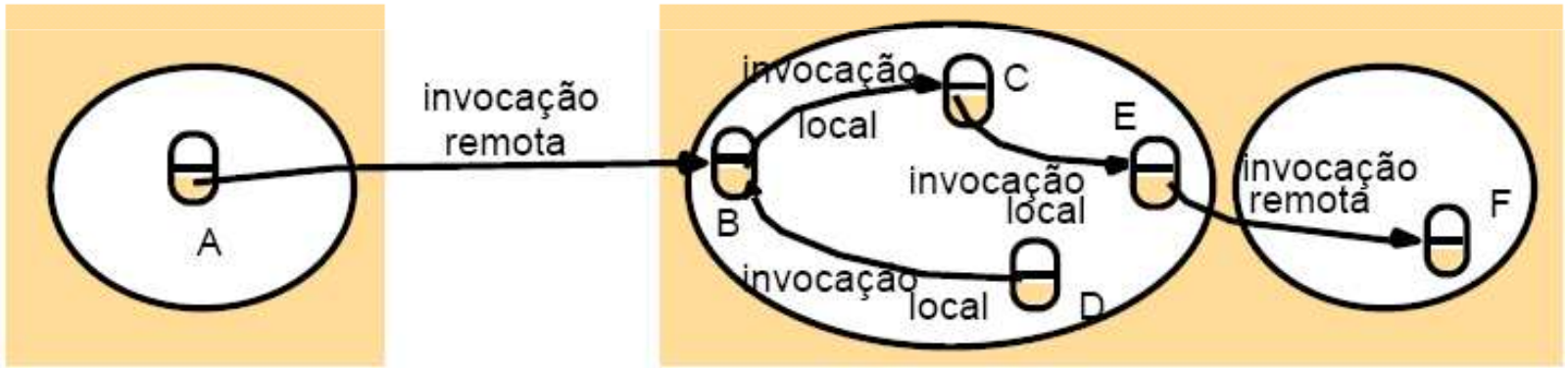
Interface e Objeto

Interface e objeto

- As definições de interfaces especificam funções que podem ser invocadas em um processo.
- Cada processo servidor possui uma interface fixa (C/S pura).
- Processos distribuídos construídos de maneira OO.
- *Muitos objetos podem ser encapsulados em processos servidores e referências a eles são passadas aos outros processos para que seus métodos sejam acessados por invocação remota*. [Coulouris]
  - Java RMI
  - CORBA

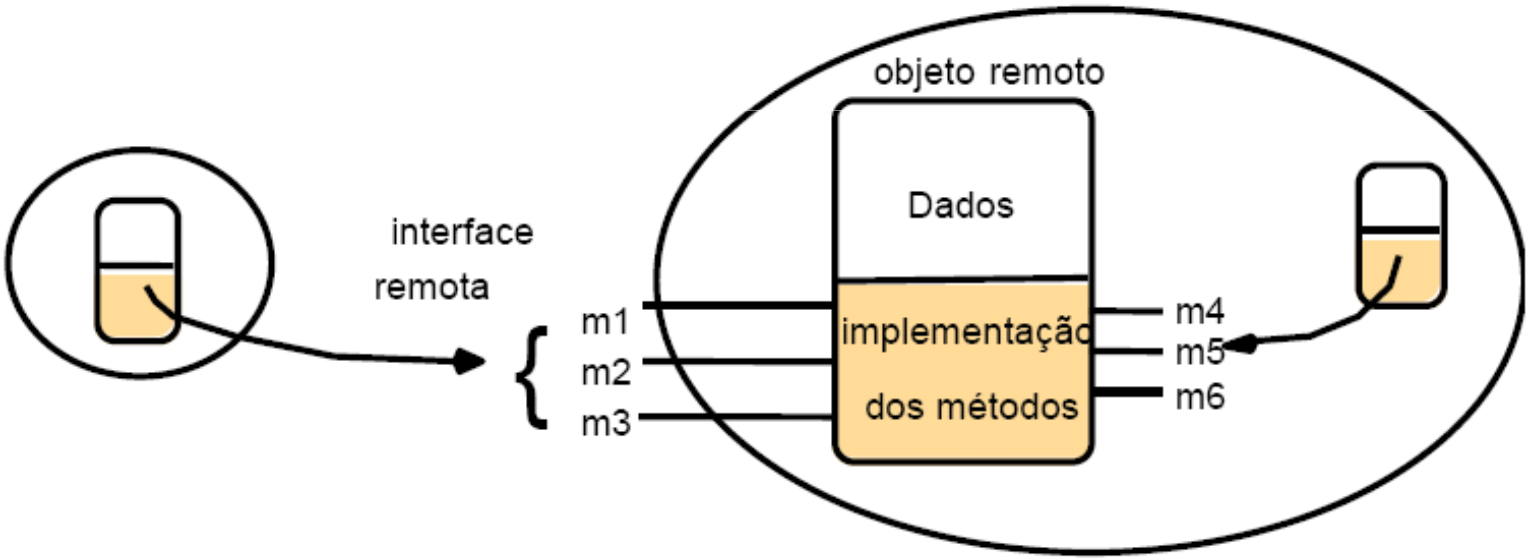
Interface e objeto

- Um processo Servidor ou Peer pode ser visto como uma coleção de objetos.
  - Objetos Remotos X Objetos Locais



Interface e objeto

- Um processo Servidor ou Peer pode ser visto como uma coleção de objetos.
  - Interface Remota



Requisitos

- Problemas de desempenho
  - Reatividade (Capacidade de resposta)
    - Atrasos no sw remoto, na comunicação, *middleware*, S.O., processos, etc.
  - *Throughput* (Taxa de rendimento)
    - Velocidade com que o trabalho computacional é feito.
  - Balanceamento de carga
    - *Applets*, vários computadores para um serviço e migração de tarefa.
- Qualidade de Serviço – QoS
  - Capacidade dos sistemas de atender prazos.
  - Depende da disponibilidade de recursos nos momentos apropriados.

Modelos de Arquitetura

Requisitos

Requisitos

- Uso de cache e replicação
  - Protocolo de consistência de cache
- Dependabilidade
  - Pode ser definido como correção, segurança e confiabilidade.
  - Técnicas
    - Tolerância a falhas
      - Falhas no hardware, software e na comunicação.
      - Redundância ajuda a confiabilidade, mas é dispendiosa e limitada.
    - Segurança

MODELOS FUNDAMENTAIS

Modelos Fundamentais

- Contém elementos essenciais para o entendimento e raciocínio a respeito do comportamento de um sistema.

- Precisa tratar:
  - Quais são as principais entidades presentes no sistema?
  - Como elas interagem?
  - Que características afetam seu comportamento individual e coletivo?
- Objetivos:
  - Tornar explícitas as suposições relevantes.
  - Generalizar o que for possível ou impossível.

Modelos Fundamentais

- Aspectos dos SD nos modelos fundamentais:
  - Interação
    - Processos interagem passando mensagens e na coordenação.
    - A comunicação ocorre com atrasos.
  - Falhas
    - Define e classifica as falhas.
    - Fornece uma base para a análise de seus efeitos e projeto do tratamento.
  - Segurança
    - Define e classifica as formas que ataques podem assumir.
    - Fornece uma base para a análise das possíveis ameaças e o projeto de métodos de resistência.

Modelo de Interação

Modelo de Interação

- Sistemas distribuídos são compostos por vários processos.
  - Algoritmos distribuídos descrevem seus comportamentos e estados.
  - As mensagens são enviadas para transferir informações entre processos e para coordenar suas atividades.
- Dois fatores que afetam significativamente a interação de processos em SD:
  - O desempenho da comunicação.
  - A impossibilidade de manter uma noção global de tempo única.

Modelo de Interação

- Desempenho da comunicação
  - Latência
    - Atraso entre o início da transmissão e o início da recepção
      - Primeiro bit no enlace.
      - Atraso no acesso à rede.
      - Tempo de processamento
  - Largura de banda
    - Capacidade de transmissão total em um certo instante.

Modelo de Interação

- Desempenho da comunicação
  - Latência
  - Largura de banda
  - *Jitter*
    - Medida de variação do atraso entre os pacotes sucessivos de dados
      - Pode ser responsável por erros e perda de sincronismo em comunicações síncronas em alta velocidade
      - Solução para minimizá-lo: utilização de *buffer*
      - Armazenar os dados a medida que eles chegam e encaminhá-los à aplicação a uma taxa constante
      - Exemplo de uso: VoIP

Modelo de Interação

- Relógios de computador e eventos de temporização
  - Cada computador possui seu relógio que é usado pelos processos locais para medir o tempo dos eventos locais.
  - Taxa de desvio do relógio (*drift*)
    - Diferença relativa para um relógio de referência perfeito.
  - *Global Positioning System* – GPS (precisão de 1 microssegundo)

Modelo de Interação

- Duas variantes do modelo de interação
  - SD Síncronos
    - Necessita estabelecer limites de tempo inferior e superior para:
      - Execução de cada etapa de um processo.
      - Transmissão da mensagem
        - cada mensagem transmitida sobre um canal é recebida dentro de um tempo limitado conhecido
      - Drift dos relógios
        - Cada processo tem um clock local cuja taxa de desvio do tempo perfeito tem um limite conhecido
    - Complexidade
    - Ex.: Multimídia
      - Necessidade de um *stream* de dados ser entregue antes do seu *deadline*

Modelo de Interação

- Duas variantes do modelo de interação
  - SD Síncronos
    - É difícil chegar em limites de tempo realísticos e prover garantia aos mesmos
      - A falta de garantia implica que qualquer projeto baseado sobre os valores estimados não é confiável
    - O modelo de um algoritmo como um sistema síncrono pode ser útil para dar uma idéia de como ele se comportará em um sistema distribuído real.
    - Em um SD síncrono o uso de timeouts permite detectar a falha de um processo
      - Limite de tempo

Modelo de Interação

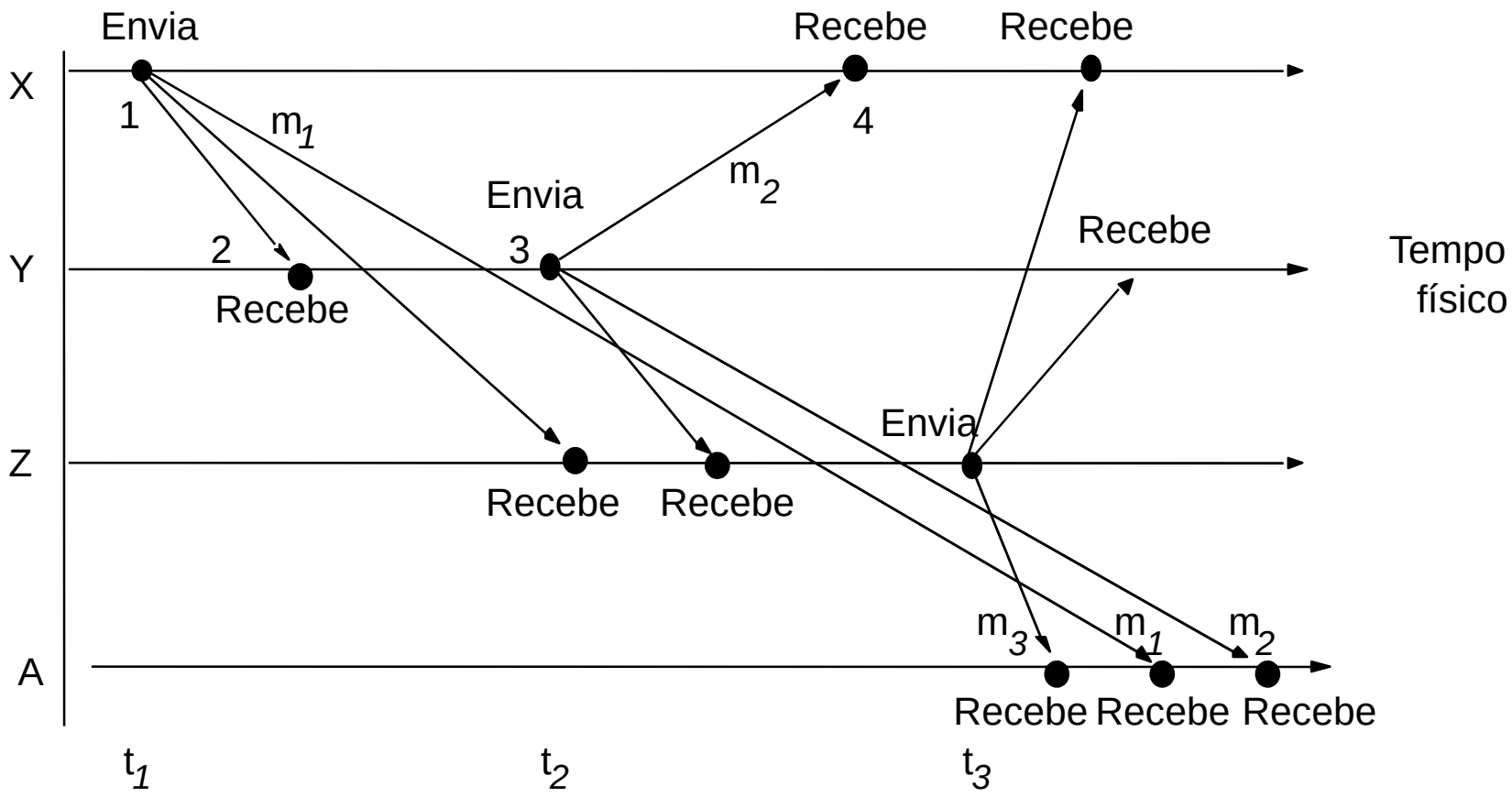
- Duas variantes do modelo de interação
  - SD Assíncronos: Não síncronos
    - Ex.: Internet.
  - Qualquer solução válida para um SD assíncrono também é válida para o síncrono.
  - SDs reais são comumente assíncronos
    - Processos compartilham processadores e canais de comunicação compartilham a rede

Modelo de Interação

- Ordenação de eventos
  - Determinar se um evento ocorreu em um processo antes, depois ou simultaneamente com outro evento em outro processo.
  - A execução de um sistema pode ser descrita em termos da ocorrência de eventos e de sua ordem, mesmo sem relógios precisos
  - Ex.:
    - X envia uma mensagem com assunto REUNIÃO para Y, Z e A.
    - Y e Z respondem com assunto Re: REUNIÃO.
    - Possibilidades para A?

Modelo de Interação

- Ordenação de eventos



Modelo de Interação

- Ordenação de eventos
  - O relógio lógico permite deduzir a ordem que as mensagens devem ser apresentadas.
  - Cada evento tem um número que corresponde a uma ordem lógica.
    - X envia m1 antes que Y receba m1
    - Y envia m2 antes que X receba m2
    - Y recebe m1 antes de enviar m2

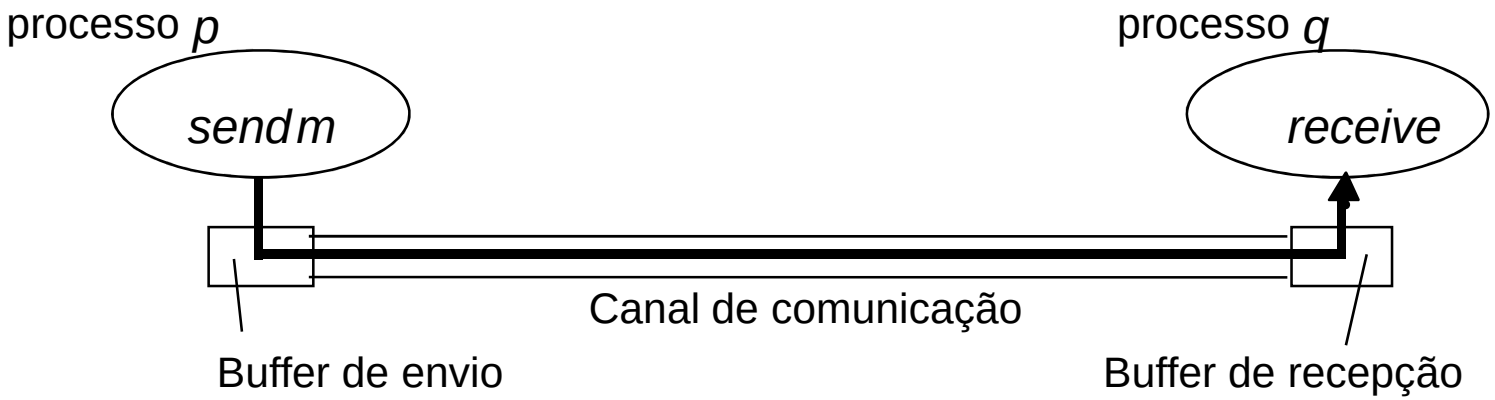
Modelo de Falhas

Modelo de Falhas

- Processos e canais de comunicação podem divergir do que é considerado um comportamento correto ou desejável.
- Proporciona um entendimento dos seus efeitos e conseqüências.
- Tipos de modelos de falhas:
  - Falha por omissão
  - Falhas arbitrárias
  - Falhas de sincronização

Modelo de Falhas

- Falhas por omissão
  - Processo ou canal de comunicação não executa as ações.
  - Falha por omissão de processo
    - O processo falha se outros processos podem determinar que isso ocorreu.
  - Falha por omissão na comunicação



Modelo de Falhas

- Falhas arbitrárias
  - Qualquer tipo de erro pode ocorrer
  - No processo
    - Omitir passos do processamento ou efetuar processamento indesejado
  - Na comunicação
    - Mensagens corrompidas
    - Mensagens inexistentes
    - Softwares de comunicação freqüentemente detectam.

Modelo de Falhas

Classe da falha	*Afeta*	*Descrição*
Parada por falha	Processo	Outros processos podem detectar
Colapso	Processo	Outros processos podem não detectar
Omissão	Canal	Sai do buffer de envio e não chega no buffer de recepção
Omissão de envio	Processo	Não chega ao buffer de envio
Omissão de recepção	Processo	Processo não "pega" no buffer de recepção
Arbitrária	Processo ou Canal	Processo/Canal exibe comportamento arbitrário

Modelo de Falhas

- Falhas de temporização
  - Aplicáveis aos SD síncronos.

Classe da falha	*Afeta*	*Descrição*
Relógio	Processo	Relógio local do processo ultrapassa o limite da taxa de desvio.
Desempenho	Processo	Processo excede o limite de tempo entre duas etapas.
Desempenho	Canal	Transmissão da mensagem supera o limite de tempo definido.

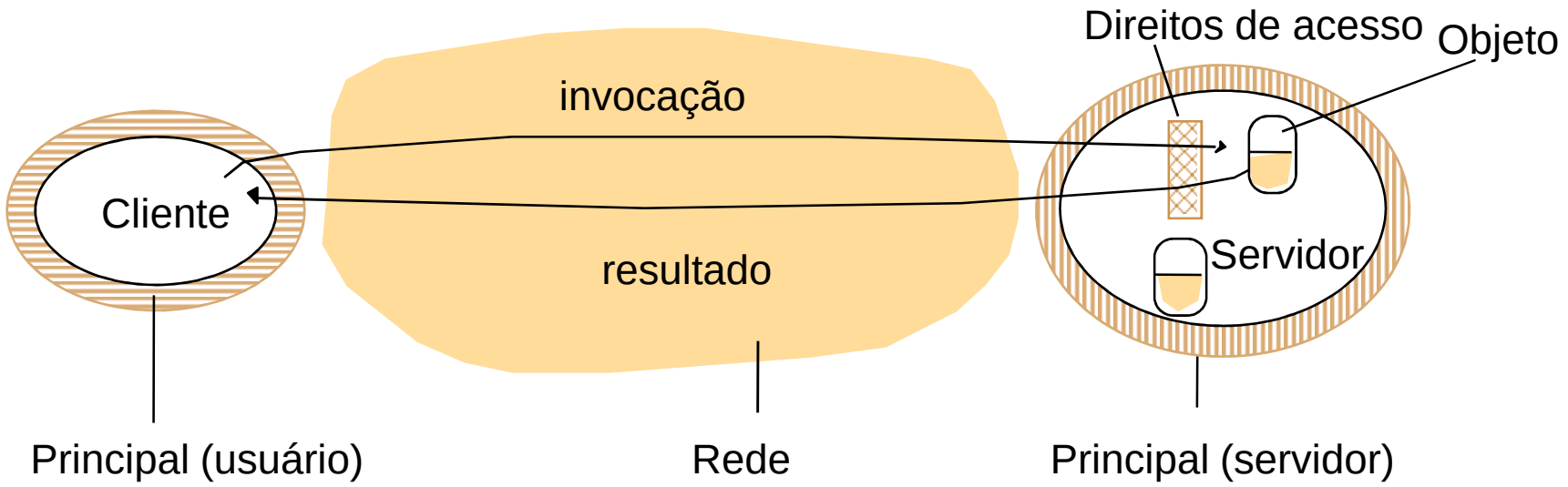
Modelo de Falhas

- Mascaramento de falhas
  - Um serviço *mascara* uma falha ocultando-a completamente ou convertendo-a em um tipo de falha mais aceitável. Ex.: Checksum.
- Confiabilidade da comunicação de um para um
  - Validade: mensagens são entregues independente do tempo.
  - Integridade: mensagens idênticas e sem duplicidade.
    - Medidas de segurança.
    - Números de seqüência

Modelo de Segurança

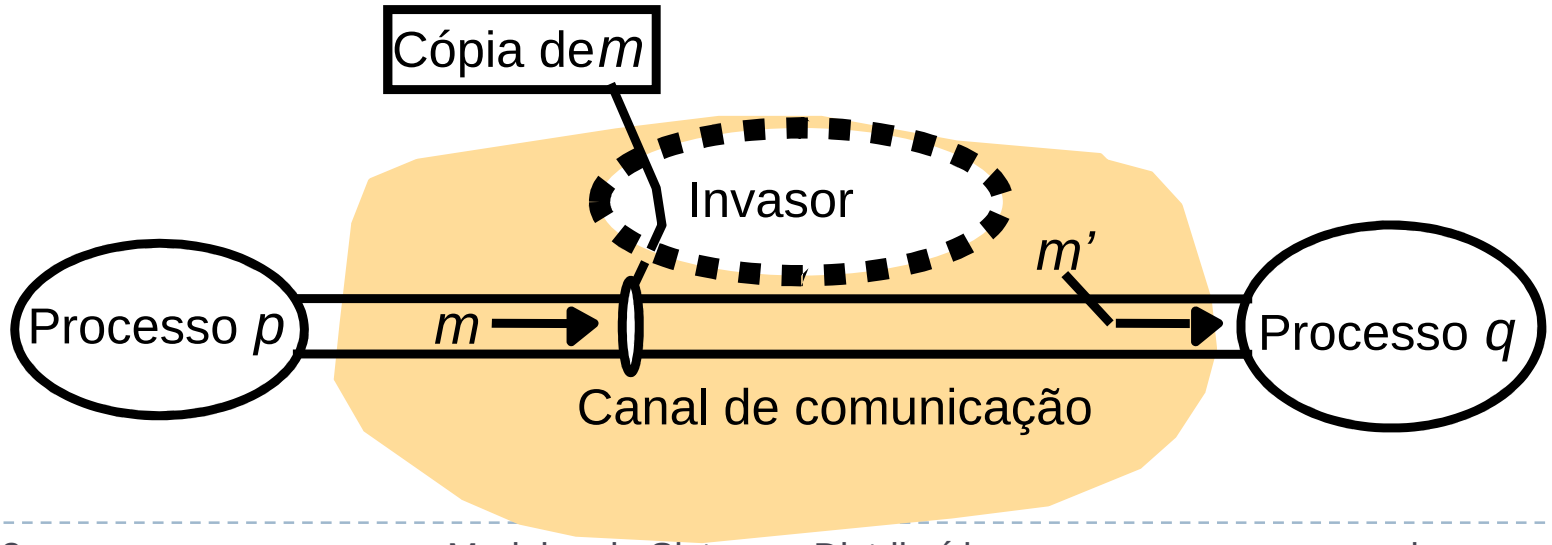
Modelo de Segurança

- Tornar seguros os processos e os canais de comunicação, e eliminar acesso não autorizados.
- Proteção de objetos



Modelo de Segurança

- Tornando processos e suas interações seguros
  - Ameaça aos processos
    - Processos podem não identificar emissor de mensagens (*spoofing*).
  - Ameaça aos canais de comunicação
    - Cópia, alteração ou injeção de mensagens no canal.



Modelo de Segurança

- Anulando ameaças à segurança
  - Criptografia e segredos compartilhados
  - Autenticação
    - Garante a identidade de um principal
  - Canais seguros (Criptografia + Autenticação). Ex.:VPN e SSL
    - Privacidade e integridade