Sistemas Operacionais Distribuídos

Arquitetura de Sistemas Computacionais Cliente/Servidor

Eduardo Nicola Ferraz Zagari

Ambiente Cliente/Servidor

1 - A Era da Computação Cliente/Servidor
2 - Características dos Sistemas Cliente/Servidor
3 - Tecnologias Cliente/Servidor
4 - Funções do Servidor
5 - Sistemas Operacionais do Servidor
6 - Escalabilidade do Servidor
7 - Sistemas Operacionais do Cliente
8 - Arquitetura Cliente/Servidor

1 - Era da Computação Cliente/Servidor

- **Bons tempos dos *mainframes***
  - **Suporte, Operação, Desenvolvimento, Segurança**
- **A revolução da computação Cliente/Servidor**
  - **Flexibilidade, Preço**
  - **Integração**
- **Internet**
  - **Revolução dentro da revolução**
- **Visão geral: prover um ambiente flexível e aberto onde *"mix-and-match"* seja a regra:**

... 1 - Era da Computação Cliente/Servidor

- **Aplicações cliente rodam predominantemente em PCs em LANs (redes locais);**
- **Servidores também estão em LANs e sabem exatamente como se comunicar com seus clientes PCs;**
- **Para *mainframes* serem servidores:**
  - Não podem enxergar PCs como terminais burros;
  - Precisam suportar protocolos par-a-par, interpretar mensagens de PCs, servir arquivos de seus PCs clientes em seus formatos nativos e prover dados e serviços aos PCs da maneira mais direta possível;

2 - Características dos Sistemas Cliente/Servidor

- **Tentativa de definição: como o próprio nome sugere, clientes e servidores são entidades lógicas separadas que trabalham juntas, possivelmente sobre uma rede, para executar uma tarefa.**
- **Características:**
  - **Serviço:** Cliente/Servidor é basicamente um relacionamento entre processos rodando em máquinas separadas. O processo servidor é um provedor de serviços. O cliente, um consumidor. Em essência, cliente/servidor provê uma clara separação de função baseada na idéia de serviço.

... 2 - Características dos Sistemas Cliente/Servidor

- **Recursos Compartilhados:** Um servidor pode servir muitos clientes ao mesmo tempo e gerenciar o acesso destes aos recursos compartilhados.
- **Protocolos Assimétricos:** Há um relacionamento de muitos para um entre clientes e servidor. Clientes sempre iniciam o diálogo, através da requisição de um serviço. Servidores permanecem passivamente aguardando requisições vindas dos clientes.

... 2 - Características dos Sistemas Cliente/Servidor

- ***Mix-and-Match*:** O software cliente/servidor ideal é independente das plataformas de hardware e de sistema operacional. Deveríamos ser capazes de combinar diferentes plataformas de clientes e de servidores.
- **Transparência de Localização:** O servidor é um processo que pode residir na mesma máquina que o cliente ou em uma máquina diferente através da rede. Os softwares cliente/servidor geralmente mascaram a localização do servidor para os clientes redirecionando as chamadas de serviço quando necessário. Um programa pode ser um cliente, um servidor ou ambos.

... 2 - Características dos Sistemas Cliente/Servidor

- **Troca de Mensagens:** Clientes e Servidores são sistemas fracamente acoplados que interagem através de um mecanismo de passagem de mensagens. A mensagem é o mecanismo de remessa das requisições de serviço e das respostas.
- **Encapsulamento de Serviços:** O servidor é um "especialista". Uma mensagem diz ao servidor qual serviço é requisitado. É da competência do servidor determinar como o trabalho deve ser feito. Os servidores podem ser atualizados sem afetar os clientes, desde que suas interfaces exportadas não sejam alteradas.

... 2 - Características dos Sistemas Cliente/Servidor

- **Escalabilidade:** Sistemas cliente/servidor podem crescer horizontalmente ou verticalmente. Crescimento horizontal significa adicionar ou remover estações de trabalho clientes com apenas um leve impacto de desempenho. Crescimento vertical significa migrar para máquinas servidoras ou multiservidoras maiores e mais rápidas.
- **Integridade:** O servidor de códigos e o servidor de dados são mantidos de forma centralizada, o que resulta em uma manutenção mais barata e na integridade de dados compartilhados. Ao mesmo tempo, os clientes permanecem personalizados e independentes.

... 2 - Características dos Sistemas Cliente/Servidor

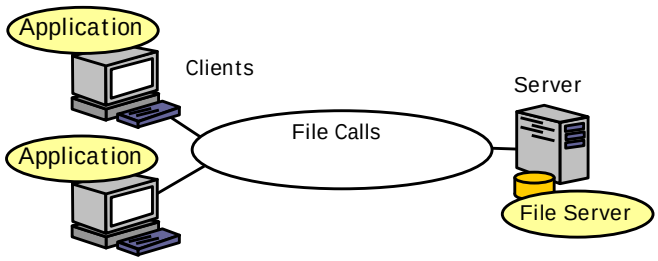
- **Tais características permitem distribuir inteligência por toda a rede, além de proverem um *framework* para o projeto de aplicações de rede fracamente acopladas.**

3.1 - Servidores de Arquivo

- Muitos sistemas com arquiteturas muito diferentes têm sido chamados cliente/servidor;
- A idéia de dividir uma aplicação em cliente/servidor tem sido usada há muitos anos para criar várias formas de soluções de software de rede;
- Cada uma destas soluções, entretanto, é diferenciada pela natureza do serviço que ela provê a seus clientes.

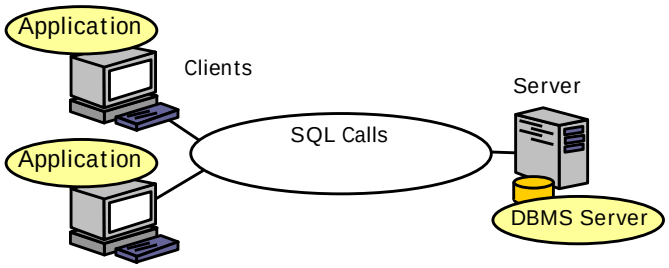
3.1 - Servidores de Arquivo

- Forma primitiva de serviço de dados
  - muitas trocas de mensagens
- Úteis no compartilhamento de arquivos na rede
- Indispensáveis na criação de repositórios de imagens, documentos e outros objetos de dados grandes



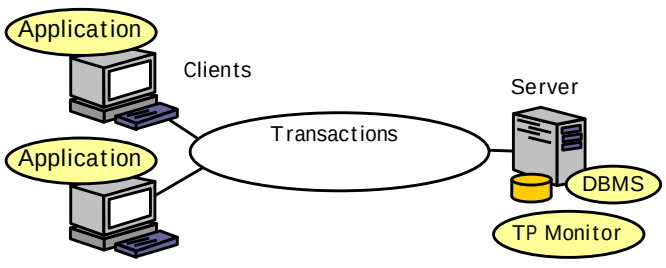
3.2 - Servidores de Base de Dados

- Maior eficiência no uso do potencial de processamento distribuído, visto que o processamento da requisição SQL se dá no lado do servidor de dados;
- O código do servidor é encapsulado pelo vendedor. Já o código da aplicação cliente precisa ser escrito.



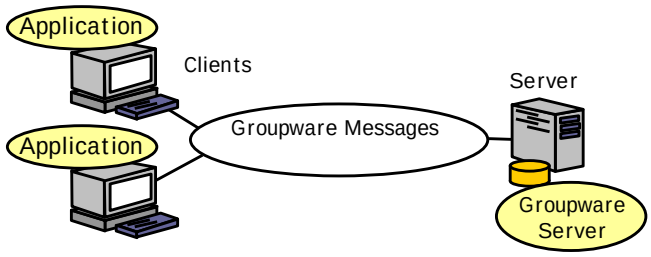
3.3 - Servidores de Transações

- Clientes invocam procedimentos remotos, que executam grupos de requisições SQL (transações);
- Deve-se escrever o código de ambos os componentes;
- Tendem a ser aplicações críticas, que requerem tempos de respostas sempre menores do que 3 segundos.



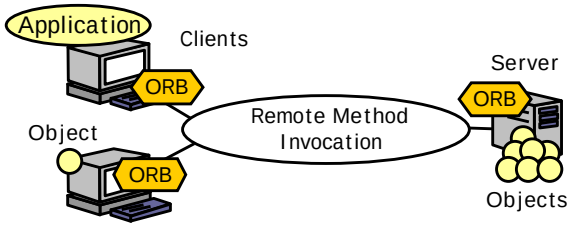
3.4 - Servidores de *Groupware*

- Groupware trata o gerenciamento de informação semi-estruturada (como texto, imagem, email etc);
- As aplicações são criadas usando-se linguagens de script providas pelos vendedores;
- O *middleware* de comunicação entre cliente e servidor é proprietário.



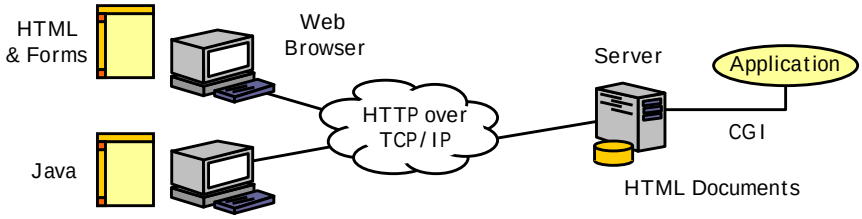
3.5 - Servidores de Objetos

- Aplicações são escritas como um conjunto de objetos comunicantes;
- Cliente invoca método em objeto remoto através de um ORB - *Object Request Broker*;
- O ORB localiza uma instância da classe daquele objeto servidor, invoca o método requerido e retorna o resultado ao objeto cliente.



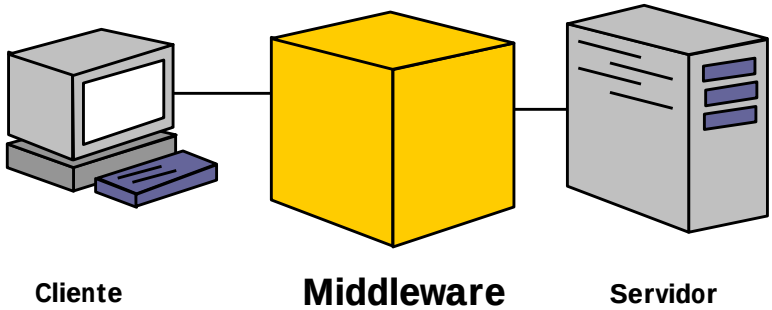
3.6 - Servidores Web

- **Este novo modelo de cliente/servidor consiste de clientes portáveis e enxutos que falam com servidores super pesados;**
- **Grosso modo, servidores Web retornam documentos quando clientes o solicitam pelo nome;**
- **A comunicação é feita usando um protocolo chamado HTTP (*RPC-like*).**



3.7 – Middleware

- **O termo *middleware* cobre todo o software distribuído necessário para se suportar as interações entre clientes e servidores:**



... 3.7 – Middleware

- O *middleware* inicia com o conjunto de APIs do lado do cliente usadas para se invocar um serviço e cobre a transmissão da requisição pela rede e sua respectiva resposta (pilha de comunicação, serviços de autenticação, RPCs, relógio de rede, etc.);
- Não inclui o software que provê o serviço de fato (servidor) nem a interface do usuário ou a lógica da aplicação (cliente);

3.8 - Divisão da Aplicação Distribuída

- Servidores pesados (servidores de groupware, transações, Web) *versus* clientes pesados (servidores de arquivos e banco de dados):
  - No modelo de clientes pesados, clientes sabem como os dados estão organizados no servidor. Provêem flexibilidade e permitem usuários criarem suas próprias aplicações;
  - Aplicações com servidores pesados são mais fáceis de gerenciar e minimizam interações na rede. Ao invés de exportar dados crus, exportam os procedimentos (ou métodos) que operam os dados;

4 - Funções do Servidor

- **Regra Básica: servir múltiplos clientes que têm interesse em um recurso compartilhado controlado pelo servidor:**
  - **esperar por requisições iniciadas pelo cliente:**
    - aguardar passivamente, na maior parte do tempo, por requisições, na forma de mensagens;
    - o servidor pode atender através de sessões dedicadas ou não;
    - deve buscar ser sempre "atencioso" com seus clientes (mesmo na hora do *rush*).

... 4 - Funções do Servidor

- **executar muitas requisições ao mesmo tempo:**
  - deve buscar atender prontamente os serviços solicitados;
  - deve ser apto a servir, de forma concorrente, múltiplos clientes, enquanto garante a integridade dos recursos compartilhados.
- **atender clientes VIP primeiro:** deve ser capaz de prover diferentes níveis de prioridade de serviço (p.ex.: *batch jobs vs on-line jobs*).

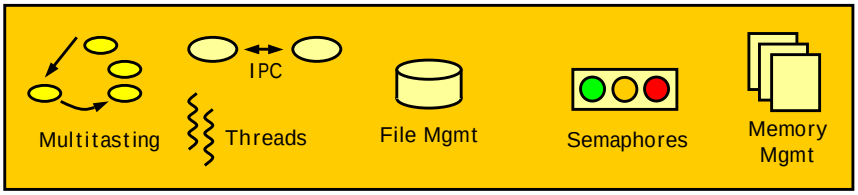
... 4 - Funções do Servidor

- **iniciar e executar atividades em *background:*** deve ser capaz de disparar tarefas em *background,* quando não são relacionadas à sua função principal (p.ex.: *downloads* fora da hora de pico).
- **manter-se executando:** o servidor geralmente é uma aplicação crítica, pois, se ele cai, causa impacto em todos os clientes que dependem de seu serviço. Por isso, assim como o ambiente em que roda, ele deve ser muito robusto.
- **outro comportamento típico:** por estar sempre requerendo mais memória e esforço computacional, o ambiente do servidor deve ser modular e escalável.

5 - Sistemas Operacionais do Servidor

- **Serviços Básicos**

Sistema Operacional Básico



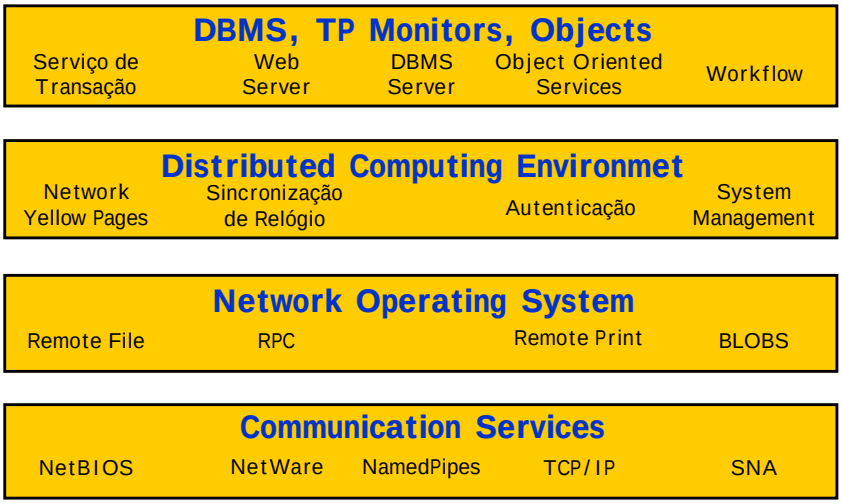
... 5 - Sistemas Operacionais do Servidor Serviços Básicos

- **Preempção de tarefas**
- **Prioridade de tarefas**
- **Semáforos -** mecanismo simples de sincronização;
- **IPC -** prover mecanismos para troca e compartilhamento de dados entre processos;
- **IPC local/remoto -** redirecionamento de chamadas de forma transparente entre processos (local e remoto);
- ***Threads* -** são unidades de concorrência dentro do próprio processo;

... 5 - Sistemas Operacionais do Servidor Serviços Básicos

- **Proteção inter-tarefas -** proteger tarefas de interferências com outros recursos (sistema de arquivos, chamadas de sistema, etc.);
- **Sistema de arquivos multiusuário de alto desempenho -** suporte a multitarefa e prover mecanismos de segurança para proteção de integridade dos dados;
- **Gerenciamento de memória eficiente;**
- **Extensões "ligáveis" em tempo de execução dinamicamente -** mecanismos que permitam o serviço crescer em tempo de execução sem necessidade de recompilação do sistema operacional.

... 5 - Sistemas Operacionais do Servidor Serviços Estendidos



... 5 - Sistemas Operacionais do Servidor Serviços Estendidos

- **Comunicação ubíqua;**

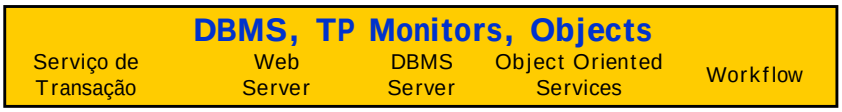
- **Extensões de Sistemas Operacionais de Rede (serviços de arquivo e de impressão);**
- **Extensões para objetos binários grandes;**

... 5 - Sistemas Operacionais do Servidor Serviços Estendidos

- Serviços de diretórios e *Yellow Pages;*
- Serviços de autenticação e de autorização;
- Extensões para gerenciamento da rede dos sistemas;
- Sincronização de relógios;

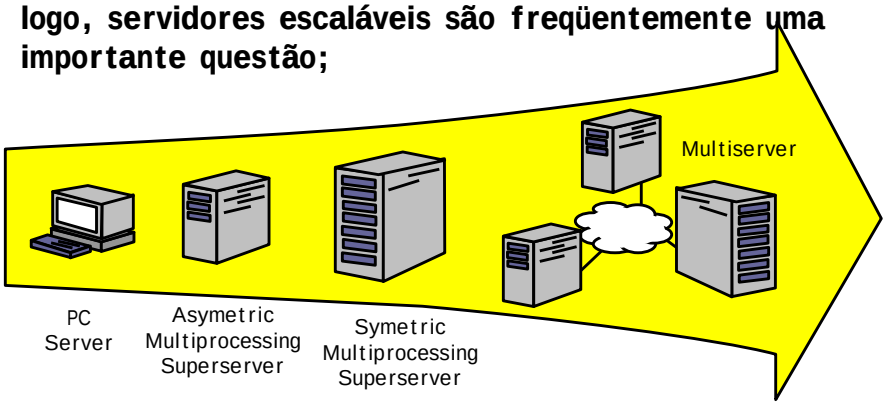
... 5 - Sistemas Operacionais do Servidor Serviços Estendidos

- Serviços de Transação e de Banco de Dados (DBMS)
- Extensões para serviços Internet (DNS, *Secure Sockets Layer, firewalls etc)*;
- Extensões para serviços orientados a objetos (*brokers,* repositórios de objetos etc);



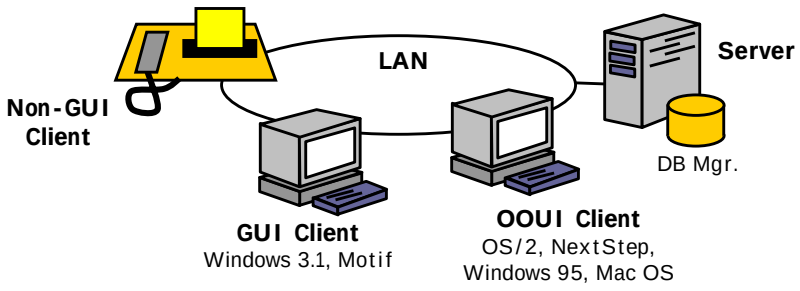
6 - Escalabilidade do Servidor

- Os limites do servidor dependem do tipo de serviço requerido pelos clientes;
- Regra: clientes sempre irão requerer mais serviços, logo, servidores escaláveis são freqüentemente uma importante questão;



7 - Sistemas Operacionais do Cliente

- Aplicações Cliente/Servidor são centradas no cliente;
- São os clientes que dão a "aparência" dos serviços prestados pelo sistema;
- Diferenças entre os clientes: o quê dispara a requisição e quais interfaces gráficas (GUI) eles provêem.

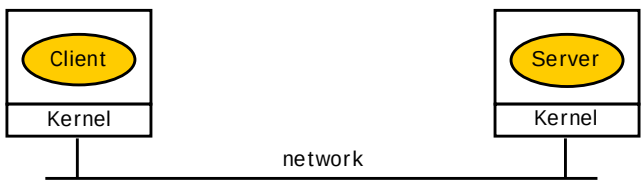


... 7 - Sistemas Operacionais do Cliente

Requisitos do Sistema Operacional	Cliente Não GUI s/multit	Não GUI c/multit	Cliente GUI	Cliente OOUI
Mecanismo Request/Reply (transparência local/remoto)	Sim	Sim	Sim	Sim
Mecanismo de transferência de arquivos (texto, imagens etc)	Sim	Sim	Sim	Sim
Multitarefa preemptivo	Não	Sim	Desejável	Sim
Prioridade de tarefas	Não	Sim	Desejável	Sim
Comunicação Inter-Processos (IPC)	Não	Sim	Desejável	Sim
Threads p/ comum/ em *background* com servidor e receb/ de resp/ deles	Não	Sim	Sim	Sim
Robustez (proteção intertarefas e chamadas de sistema reentrantes)	Não	Sim	Desejável	Sim

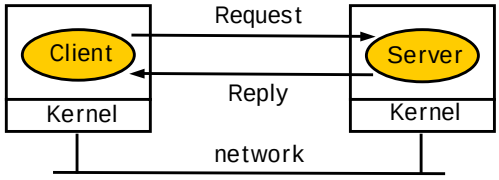
8 – Arquitetura Cliente/Servidor

- Idéia - estruturar o Sistema Operacional de forma que servidores (*servers*) oferecem serviços a seus usuários (*clients*);
- servidores e clientes executam como processos do usuário sobre o *kernel* do S.O.



... 8 – Arquitetura Cliente/Servidor

- **Para evitar *overhead* considerável, o Modelo Cliente-Servidor é baseado em protocolo sem conexão do tipo *request/reply*;**
- **Vantagens: estrutura simples, eficiência;**

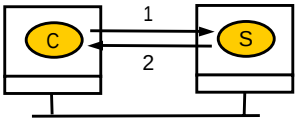


... 8 – Arquitetura Cliente/Servidor

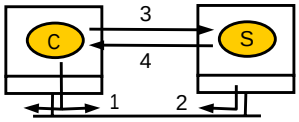
- **Serviço de comunicação com duas chamadas de sistema - enviar e receber mensagens:**
  - ***send(dest, &mptr)*** - envia a mensagem cujo endereço é dado em "*mptr*" para um processo identificado por "*dest*", mantendo-se bloqueado enquanto a mensagem está sendo enviada;
  - ***receive(addr, &mptr)*** - recebe a mensagem de um endereço "*addr*" armazenando-a no endereço dado por "*mptr*", mantendo-se bloqueado enquanto espera a mensagem;
- **Muitas variações dos procedimentos, assim como de seus parâmetros são possíveis;**

... 8 – Arquitetura Cliente/Servidor
Endereçamento

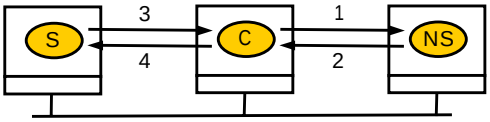
- **Para um cliente enviar uma mensagem para um servidor, ele precisa conhecer o seu endereço;**



1: Request to 243.0
2: Reply to 199.0



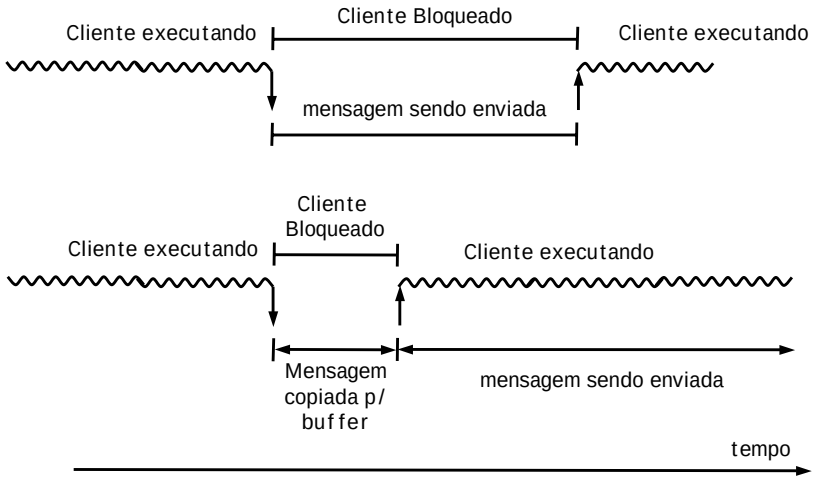
1: Braodcast 2: Here I am
3: Request ... 4: Reply ...



Name Server

1: Lookup 2: NS reply
3: Request ... 4: Reply ...

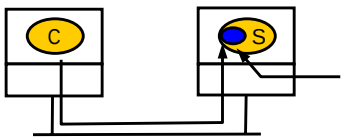
... 8 – Arquitetura Cliente/Servidor
Primitivas Bloqueantes e Não Bloqueantes



... 8 – Arquitetura Cliente/Servidor Primitivas Não Bufferizadas

- ***receive(addr, &mptr)*** **-** informa o kernel da máquina que o processo chamado está esperando por mensagens (*listenning*) no endereço "*addr"* e preparado para receber mensagens enviada para aquele endereço;
- **quando a mensagem chega, o kernel copia a mensagem p/ um buffer e desbloqueia o processo;**
- **este esquema funciona bem desde que o servidor chame *"receive"* antes do cliente chamar "*send*";**

Troca de Mensagem Não Bufferizada

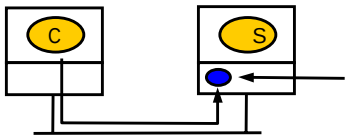


Endereço refere-se a um processo

... 8 – Arquitetura Cliente/Servidor Primitivas Bufferizadas

- **um processo interessado em receber mensagens solicita ao kernel que crie um "*mailbox*" para ele;**
- **um chamada a um *"receive"* apenas remove uma mensagem do *"mailbox"*, ou bloqueia se não há nada presente;**
- **assim, o *kernel* sabe o que fazer com mensagens que chegam e tem um local para colocá-las.**

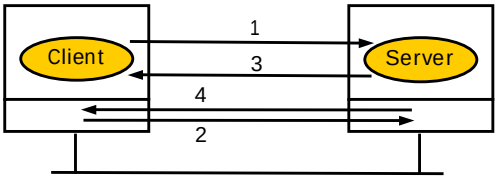
Troca de Mensagem Bufferizada



Endereço refere-se a um *mailbox*

... 8 – Arquitetura Cliente/Servidor
Primitivas Confiáveis e Não Confiáveis

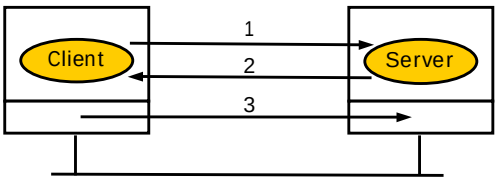
- o kernel da máquina que recebe envia um “reconhecimento” para o kernel da máquina que envia;
- o “reconhecimento” vai de um kernel para outro;
- um *request* e *reply* se tornam 4 mensagens, em decorrência dos dois “reconhecimentos”.



1: request (client p/ server)
2: ACK (kernel p/ kernel)
3: Reply (server p/ client)
4: ACK (kernel p/ kernel)

... 8 – Arquitetura Cliente/Servidor
Primitivas Confiáveis e Não Confiáveis

- o cliente é bloqueado após enviar uma mensagem;
- o kernel do servidor não envia um reconhecimento, ao invés disso o próprio *reply* faz esse papel;
- se levar muito tempo, o kernel do cliente solicita novamente, salvaguardando a perda de mensagens.



1: Request (client p/ server)
2: Reply (server p/ client)
3: ACK (kernel p/ kernel)