



**MINISTÉRIO DA EDUCAÇÃO**  
Secretaria de Educação Profissional e Tecnológica  
Instituto Federal de Educação, Ciência e Tecnologia de São Paulo  
Pós-Graduação Lato Sensu em Docência para a Educação Profissional e  
Tecnológica

**LUCIVALDO MIRANDA DOS SANTOS**

**Redes de computadores**  
**Fundamentos da Arquitetura clientes/ Servidores**

Tocantins / TO

2023

**LUCIVALDO MIRANDA DOS SANTOS**

**Redes de computadores**  
**Fundamentos da Arquitetura clientes/ Servidores**

Trabalho Final de Curso apresentado Programa de Pós-graduação *Lato sensu* em Docência para a Educação Profissional e Tecnológica do Instituto Federal de Educação, Ciência e Tecnologia de São Paulo - Campus Barretos, como requisito parcial para a conclusão do curso.  
Orientador(a): Prof. Me(a). Marcio Tilles de Souza Malta.

## **RESUMO**

### **IMPLEMENTAÇÃO DE UM SISTEMA PARA AUXILIAR NA RASTREABILIDADE DE USUÁRIOS NO ÂMBITO DA UFSM**

**AUTOR:** Daniel Vicentini de Barcelos **ORIENTADOR:** Tiago Antonio Rizzetti Este trabalho apresenta o estudo que a Universidade Federal de Santa Maria está implementando, um Captive Portal para Moradia Estudantil na Casa do Estudante localizada Campus do centro da cidade de Santa Maria, com o intuito de prover autenticação de usuários e guarda das conexões de acesso à Internet Acadêmica e Comercial, considerando a Norma Complementar do DSIC (21/IN01/DSIC/GSIPR), que trata das "Diretrizes para o registro de eventos, coleta e preservação de evidências de Incidentes de Segurança em Redes" e o Marco Civil da Internet. A rede da Moradia Estudantil não se integra as demais da Instituição, possui um domínio de broadcast único. Seu contato com a rede da UFSM e Internet Acadêmica/Comercial se dá através de um IP único ao sofrer NAT. Após realização de testes, os quais se mostraram bastante satisfatório, foi realizada a implementação do sistema, que está em pleno funcionamento e em processo de expansão.

**Palavras-chave:** Servidores, Portal, firewall, RADIUS, LDAP.

## **ABSTRACT**

### **IMPLEMENTATION OF A SYSTEM TO AUXILIATE USER TRACEABILITY IN THE FIELD OF IFTO**

**AUTHOR:** Lucivaldo Miranda dos Santos

This work presents the study that the Institute de Technology de Tocantins IFTO implementing a Captive Portal for Student Housing in the Student House located in Campus of the city center of Santa Maria with the purpose of providing user authentication and guarding of Internet access connections Academic And Commercial, considering the DSIC Supplementary Standard (21 / IN01 / DSIC / GSIPR), which deals with the "Guidelines for the recording of events, collection and preservation of evidence of Security Incidents in Networks" and the Civil Internet Framework. The Student Housing network is not integrated with the rest of the Institution, it has a unique broadcast domain. Your contact with UFSM and Academic / Commercial Internet is through a unique IP when suffering from NAT. After performing tests, which proved to be quite satisfactory, the system was implemented, which is in full operation and in the process of expansion.

**Keywords:** Captive Portal, firewall, pfSense, RADIUS, LDAP

## LISTA DE SIGLAS

|      |                                                                  |
|------|------------------------------------------------------------------|
| GUI  | Graphical User Interface                                         |
| SOR  | Sistema Operacional de Rede                                      |
| SNA  | <i>Systems Network Architecture</i>                              |
| IPC  | Inter Process Communications                                     |
| SGBD | Sistema Gerenciador de Banco de Dados                            |
| OSI  | <i>Open System Interconnection</i>                               |
| Ifto | Instituto Federal do Tocantins                                   |
| IGAM | Instituto Mineiro de Gestão das Águas                            |
| IQA  | Índice da Qualidade das Águas                                    |
| IVA  | Índice da Preservação da Vida Aquática                           |
| SP   | São Paulo                                                        |
| IFSP | Instituto Federal de Educação, Ciência e Tecnologia de São Paulo |

## Sumário

|      |                                                                                                                                                                                                         |    |
|------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----|
| 1    | INTRODUÇÃO .....                                                                                                                                                                                        | 9  |
| 2    | CAPÍTULOS DE DESENVOLVIMENTO .....                                                                                                                                                                      | 9  |
| 3    | Cliente .....                                                                                                                                                                                           | 10 |
| 4    | Servidor .....                                                                                                                                                                                          | 10 |
| 4.1  | Vantagens .....                                                                                                                                                                                         | 11 |
| 4.2  | Confiabilidade .....                                                                                                                                                                                    | 11 |
| 4.3  | Matriz de Computadores agregando capacidade de processamento .....                                                                                                                                      | 11 |
| 4.4  | O Sistema cresce facilmente .....                                                                                                                                                                       | 11 |
| 4.5  | .....                                                                                                                                                                                                   | 11 |
| 4.6  | O Cliente e o Servidor possuem ambientes operacionais individuais /Sistemas Abertos .....                                                                                                               | 11 |
| 5    | Desvantagens .....                                                                                                                                                                                      | 11 |
| 5.1  | Manutenção .....                                                                                                                                                                                        | 12 |
| 5.2  | Ferramentas .....                                                                                                                                                                                       | 12 |
| 5.3  | Treinamento .....                                                                                                                                                                                       | 12 |
| 5.4  | .....                                                                                                                                                                                                   | 12 |
| 5.5  | Gerenciamento .....                                                                                                                                                                                     | 12 |
| 6    | - Modelos da Arquitetura Cliente / Servidor .....                                                                                                                                                       | 12 |
| 6.1  | - Arquitetura C/S Simples .....                                                                                                                                                                         | 12 |
| 6.2  | - Arquitetura C/S em Dois Níveis .....                                                                                                                                                                  | 12 |
| 6.3  | - Arquitetura C/S Multinível .....                                                                                                                                                                      | 13 |
| 6.4  | Arquitetura C/S Par-Par .....                                                                                                                                                                           | 14 |
| 7    | - Alguns Tipos de Processos numa Arquitetura Cliente/Servidor .....                                                                                                                                     | 14 |
| 7.1  | - Processamentos Distribuídos .....                                                                                                                                                                     | 16 |
| 8    | Processamento Distribuído .....                                                                                                                                                                         | 16 |
| 8.1  | - Camadas da Arquitetura Cliente / Servidor .....                                                                                                                                                       | 17 |
| 9    | - Redes de Computadores .....                                                                                                                                                                           | 20 |
| 9.1  | - Protocolos .....                                                                                                                                                                                      | 20 |
| 9.2  | Há duas décadas, diversos esforços vêm sendo traçados para se estabelecer um padrão único para redes de computadores. No entanto, foram definidos não apenas um, mas vários modelos de referência. .... | 20 |
| 9.3  | - O Modelo de Referência <i>OSI/ISO</i> .....                                                                                                                                                           | 20 |
| 10   | - TCP/IP .....                                                                                                                                                                                          | 21 |
| 10.2 | - Aspectos de Sincronismo e Passagem de Mensagem .....                                                                                                                                                  | 26 |
| 10.3 | - Conexão TCP/IP .....                                                                                                                                                                                  | 27 |
| 11   | - Sockets .....                                                                                                                                                                                         | 40 |
| 12   | - Bancos de Dados Orientados a Objetos .....                                                                                                                                                            | 42 |
| 12.1 | - Introdução .....                                                                                                                                                                                      | 42 |
| 12.2 | - SGBDOO GemStone .....                                                                                                                                                                                 | 43 |
| 12.3 | - SGBDOO POET .....                                                                                                                                                                                     | 45 |
| 13   | ASPECTOS CONCLUSIVOS/CONCLUSÃO/CONSIDERAÇÕES FINAIS .....                                                                                                                                               | 46 |
|      | REFERÊNCIAS .....                                                                                                                                                                                       | 47 |



Eixo Tecnológico: Curso de Pós Graduação

Curso Técnico: Lato Sensu em Docência para Educação Profissional Tecnológica

Disciplina:TFC- Trabalho Final de Curso

Tema: Fundamentos da Arquitetura clientes/ Servidores

Objetivo: Esclarecer a Conectividade entre Servidores e Clientes

Descrição das principais atividades desenvolvidas: Comunicação, conexão, usabilidade dos equipamentos eletrônicos computacionais na educação.

## 1 INTRODUÇÃO

Este trabalho tem por objetivo mostrar as características técnicas que envolvem a Arquitetura Cliente/Servidor, desde a sua concepção teórica até a implantação da comunicação entre Cliente e Servidor por meio de uma biblioteca de Sockets.

No item 2 será definido a parte teórica da Arquitetura Cliente/Servidor bem como alguns aspectos de definição desta arquitetura, além de mostrar os modelos existentes para implantação desta arquitetura. Em 3 serão mostrados alguns tipos de processamentos existentes, porém enfocando o processamento distribuído. Em 4 serão vistos os conceitos básicos sobre Rede de Comunicação, enfatizando o protocolo TCP/IP e a biblioteca de Sockets. Em 5 são apresentadas as características técnicas de alguns SGBDOOs numa arquitetura Cliente/Servidor. Finalmente, em 6, é feita uma breve conclusão com respeito ao modelo de arquitetura apresentado.

## 2 CAPÍTULOS DE DESENVOLVIMENTO

A arquitetura Cliente/Servidor vem sendo desenvolvida há vários anos, porém em pequenos passos. Primeiro, a realocação de aplicações em Mainframe para as chamadas plataformas abertas rodando, Sistema Operacional UNIX. Depois, com relação a abordagem dos dados, saindo de Sistemas de Arquivos ou Banco de Dados Hierárquicos localizados em Mainframes para Sistemas de Banco de Dados Relacional, e posteriormente, a importância da capacidade gráfica dos pacotes de “front-end” existentes, facilitando a interação com o usuário (MCKIE, 1997).

Vários aspectos sobre uma definição da arquitetura Cliente/Servidor podem ser descritos.

- ☐ O termo Cliente/Servidor refere-se ao método de distribuição de aplicações computacionais através de muitas plataformas. Tipicamente essas aplicações estão divididas entre um provedor de acesso e uma central de dados e numerosos clientes contendo uma interface gráfica para usuários para acessar e manipular dados.
- ☐ Cliente/Servidor geralmente refere-se a um modelo onde dois ou mais computadores interagem de modo que um oferece os serviços aos outros. Este modelo permite aos usuários acessarem informações e serviços de qualquer lugar.
- ☐ Cliente/Servidor é uma arquitetura computacional que envolve requisições de serviços de clientes para servidores. Uma rede Cliente/Servidor é uma extensão lógica de a programação modular.

Portanto, uma definição para a arquitetura Cliente/Servidor seria a existência de uma plataforma base para que as aplicações, onde um ou mais Clientes e um ou mais servidores, juntamente com o Sistema Operacional e o Sistema Operacional de Rede, executem um processamento distribuído.

Um sistema Cliente/Servidor poderia ser então, entendido como a interação entre Software e Hardware em diferentes níveis, implicando na composição de diferentes computadores e aplicações.

Para melhor se entender o paradigma Cliente/Servidor é necessário observar que o conceito chave está na ligação lógica e não física. O Cliente e o Servidor podem coexistir ou não na mesma máquina (RENAUD, 1994). Porém um ponto importante para uma real abordagem Cliente/Servidor é a necessidade de que a arquitetura definida represente uma computação distribuída (MCKIE, 1997).

Algumas das características do Cliente e do Servidor são descritas a seguir: (SALEMI, 1993) (HULQUIST, 1997)

### 3 Cliente

- [Cliente, também denominado de “*front-end*” e “*WorkStation*”, é um processo que interage com o usuário através de uma interface gráfica ou não, permitindo consultas ou comandos para recuperação de dados e análise e representando o meio pela qual os resultados são apresentados. Além disso, apresenta algumas características distintas:
- É o processo ativo na relação Cliente/Servidor.
- Inicia e termina as conversações com os Servidores, solicitando serviços distribuídos.
- Não se comunica com outros Clientes.
- Torna a rede transparente ao usuário.

### 4 Servidor

- Também denominado Servidor ou “*back-end*”, fornece um determinado serviço que fica disponível para todo Cliente que o necessita. A natureza e escopo do serviço são definidos pelo objetivo da aplicação Cliente/Servidor. Além disso, ele apresenta ainda algumas propriedades distintas:
- É o processo reativo na relação Cliente/Servidor.
- Possui uma execução contínua.
- Recebe e responde às solicitações dos Clientes.
- Não se comunica com outros Servidores enquanto estiver fazendo o papel de Servidor.
- Presta serviços distribuídos.
- Atende a diversos Clientes simultaneamente.

Outras características dos processos Cliente e Servidor podem ser encontradas em (KALAKOTA, 1997).

Alguns tipos de serviços que um Servidor pode proporcionar são:

- ☐ Servidor de Arquivos
- ☐ Servidor de Impressora
- ☐ Servidor de Banco de Dados

- ☐ Servidor de Redes
- ☐ Servidor de Telex
- ☐ Servidor de Fax
- ☐ Servidor X-Windows
- ☐ Servidor de Processamento e Imagens
- ☐ Servidor de Comunicação e etc.

O estilo de interação entre o usuário e o Cliente não precisa, necessariamente, ser feita por poderosas interfaces gráficas. Porém, já que o poder de processamento local do Cliente está disponível, pode-se retirar todo seu proveito, através de interfaces gráficas - GUI (Graphical User Interface), para melhor rendimento do usuário no seu trabalho.

Dentre as muitas vantagens da arquitetura Cliente/Servidor, pode-se citar: (SALEMI, 1993).

#### **4.1 Vantagens**

#### **4.2 Confiabilidade**

Se uma máquina apresenta algum problema, ainda que seja um dos Servidores, parte do Sistema continua ativo.

#### **4.3 Matriz de Computadores agregando capacidade de processamento**

A arquitetura Cliente / Servidor provê meios para que as tarefas sejam feitas sem a monopolização dos recursos. Usuários finais podem trabalhar localmente.

#### **4.4 O Sistema cresce facilmente**

Torna-se fácil modernizar o Sistema quando necessário.

#### **4.5**

#### **4.6 O Cliente e o Servidor possuem ambientes operacionais individuais / Sistemas Abertos**

Pode-se misturar várias plataformas para melhor atender às necessidades individuais de diversos setores e usuários.

Além destas vantagens, pode-se encontrar dentro de uma arquitetura Cliente/Servidor a interoperabilidade das estações Clientes e Servidoras entre as redes de computadores, a escalabilidade da arquitetura visando o crescimento e a redução dos elementos constituintes, a adaptabilidade de novas tecnologias desenvolvidas, a performance do hardware envolvido na arquitetura, a portabilidade entre as diversas estações que compõem a arquitetura e a segurança dos dados e processos (MCKIE, 1997).

Embora o avanço da arquitetura Cliente/Servidor tenha trazido uma variada gama de facilidades para o desenvolvimento de aplicações distribuídas, também possui algumas desvantagens:

## **5 Desvantagens**

### 5.1 Manutenção

As diversas partes envolvidas nem sempre funcionam bem juntas. Quando algum erro ocorre, existe uma extensa lista de itens a serem investigados.

### 5.2 Ferramentas

A escassez de ferramentas de suporte, não raras vezes obriga o desenvolvimento de ferramentas próprias. Em função do grande poderio das novas linguagens de programação, esta dificuldade está se tornando cada vez menor.

### 5.3 Treinamento

A diferença entre a filosofia de desenvolvimento de software para o microcomputador de um fabricante para o outro, não é como a de uma linguagem de programação para outra. Um treinamento mais efetivo torna-se necessário.

### 5.4

### 5.5 Gerenciamento

Aumento da complexidade do ambiente e a escassez de ferramentas de auxílio tornam difícil o gerenciamento da rede.

## 6 - Modelos da Arquitetura Cliente / Servidor

Existem cinco tipos de modelos para a implantação da arquitetura Cliente/Servidor em processamentos distribuídos: (SALEMI, 1993)

### 6.1 - Arquitetura C/S Simples

A primeira abordagem para um sistema distribuído é a arquitetura Cliente/Servidor Simples. Nesta arquitetura, o Servidor não pode iniciar nada. O Servidor somente executa as requisições do Cliente. Existe uma clara função de diferenciação: Pode-se estabelecer que o Cliente é o mestre e o Servidor é o escravo, como mostra a figura 1.

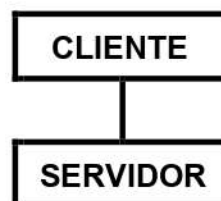


Figura 1 - Arquitetura Cliente/Servidor Simples

### 6.2 - Arquitetura C/S em Dois Níveis

A configuração usual Cliente/Servidor encontrada na maioria das empresas, é aquela em que existem vários Clientes requisitando serviços a um único Servidor. Esta arquitetura se caracteriza como sendo *centrada no Servidor* (figura 2a). Porém na visão do usuário, ele imagina que existem vários

Servidores conectados a somente um Cliente, ou seja, *centrado no Cliente* (figura 2b). Entretanto, com as várias ligações de comunicação possíveis, existe na realidade uma mistura de Clientes e Servidores (figura 2c).



Figura 2 - (a) Arquitetura C/S em Dois Níveis - Centrado no Servidor



Figura 2 - (b) Arquitetura C/S em Dois Níveis - Centrado no Cliente

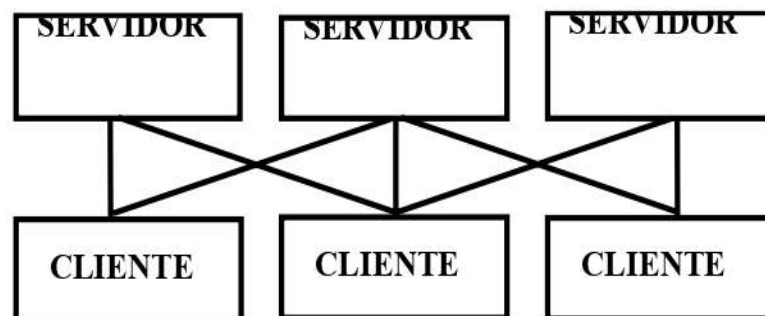


Figura 2 - (c) Arquitetura C/S em Dois Níveis - Comunicação Mista

### 6.3 - Arquitetura C/S Multinível

Nesta arquitetura (figura 3), permite-se que uma aplicação possa assumir tanto o perfil do Cliente como o do Servidor, em vários graus. Em outras palavras, uma aplicação em alguma plataforma será um Servidor para alguns Clientes e, concorrentemente, um Cliente para alguns Servidores.

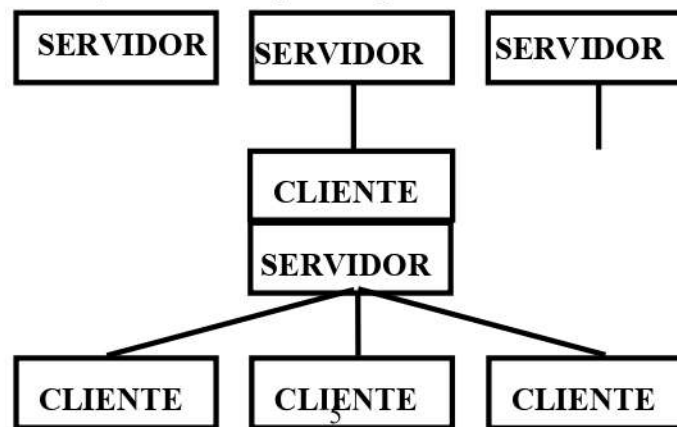


Figura 3 - Arquitetura C/S Multinível

#### 6.4 Arquitetura C/S Par-Par

Esta arquitetura pode ser vista como o caso mais geral da arquitetura Cliente/Servidor, ilustrado na figura 4. Cada um dos nodos desta arquitetura assume tanto o papel de Cliente quanto de Servidor. Na verdade, torna-se pouco funcional lidar com quem é o Cliente ou o Servidor. É o caso onde o processo interage com outros processos em uma base pareada, não existindo nenhum Mestre ou Escravo: qualquer estação de trabalho pode iniciar um processamento, caso possua uma interface de comunicação entre o usuário e o processo Cliente.

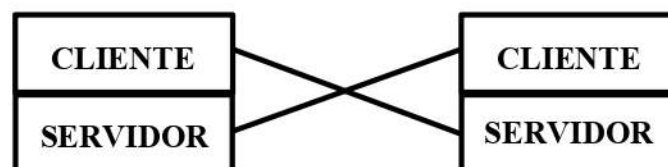


Figura 4 - Arquitetura C/S Par-Par

### 7 - Alguns Tipos de Processos numa Arquitetura Cliente/Servidor

A arquitetura Cliente/Servidor divide uma aplicação em processos, que são executados em diferentes máquinas conectadas à uma Rede de Computadores, formando um único sistema. O paradigma da tecnologia Cliente/Servidor, serve como um modelo, entre outros (ANDREWS,1991), para interação entre processos de softwares em execução concorrente.

Os processos ou tarefas, a serem executadas são divididas entre o Servidor e o Cliente, dependendo da aplicação envolvida e das restrições impostas pelo Sistema

Operacional de Rede (SOR). Quanto mais avançado for o Sistema Operacional de Rede, menor será a aplicação em si, uma vez que a implementação do código para acessar a rede já se encontra definido no SOR.

Atualmente dois tipos de processamentos estão sendo divulgados, Processamento Distribuído e Processamento Cooperativo. A característica de cada um destes é descrita a seguir (KALOKATA,1997), e neste trabalho o enfoque será destinado ao processamento distribuído.

#### *Processamento Distribuído*

A distribuição de aplicações e tarefas se faz através de múltiplas plataformas de processamento. O processamento distribuído implica que

ess  
as aplicações/tarefas irão ocorrer em mais de um processo, na ordem de uma transação a ser concluída. Em outras palavras, o processamento é distribuído através de duas ou mais máquinas e os processos, na maioria, não rodam ao mesmo tempo, por exemplo, cada processador realiza parte de uma aplicação em uma sequência. Geralmente, o dado usado em um ambiente de processamento distribuído também é distribuído através de plataformas.

#### *Processamento Cooperativo*

A cooperação requer dois ou mais processadores distintos para completar uma

simples transação. O processamento cooperativo é relatado para ambos os processos cliente-servidor. É um a forma de computação distribuída onde dois

u mais processadores distintos são requeridos para completar uma simples transação de negócio s. Normalmente esses programas interagem e executam concorrentemente como diferentes processos. Os processos cooperativos também são considerados como um estilo de Cliente/Servidor através da arquitetura de mensagens, que devem obedecer a um determinado padrão.

No contexto do presente trabalho pretende-se utilizar as características do processamento distribuído. Este tipo de processamento apresenta duas configurações para uma arquitetura Cliente/Servidor. A primeira, que é representada por três camadas, é responsável pela visualização da interação entre os aplicativos e o hardware, como pode ser visto na figura

A segunda representação, também fornecida em três camadas, mostra como é tratada a divisão da funcionalidade de uma aplicação, segundo as configurações do *Gartner Group*, como pode ser visto na figura 6.

Processamento Cliente-Servidor

**Cliente**

**Servidor**

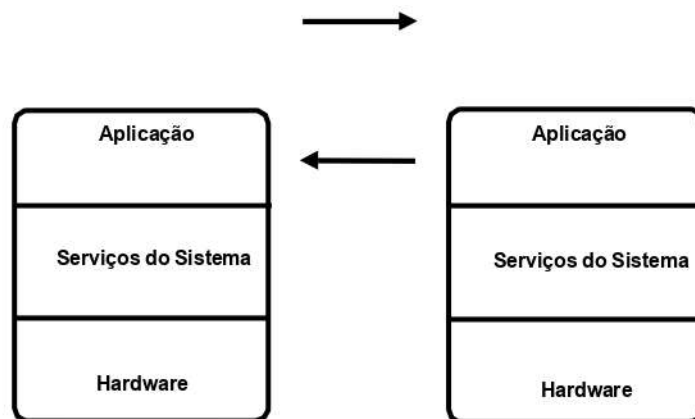


Figura 5 - Sistema Cliente/Servidor

### 7.1 - Processamentos Distribuídos

O processamento distribuído, também denominado de processamento concorrente utiliza-se do mecanismo de passagem de mensagens para a comunicação entre processos, que podem ser de quatro tipos básicos; Filtro, Peer (não hierárquico), Cliente e Servidor (RENAUD, 1994), como mostrado na figura 7 a seguir.

Modelo de Distribuição de Processos

## 8 Processamento Distribuído

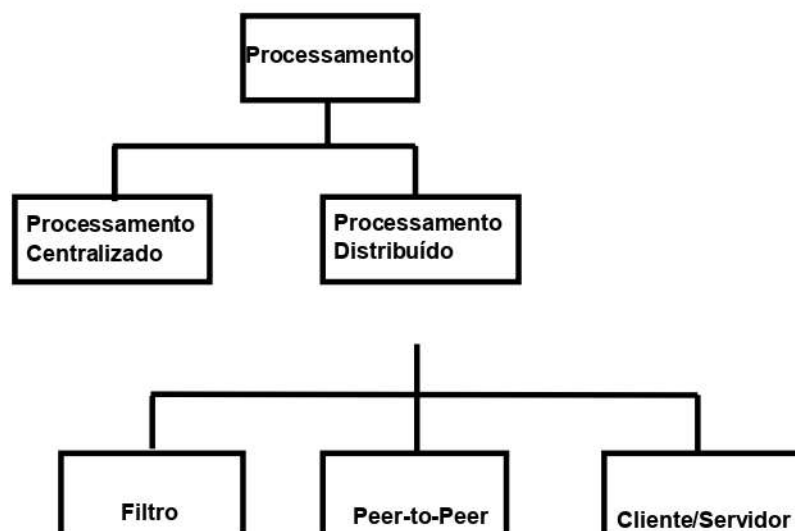


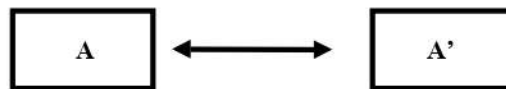
Figura 6 - Processamento Distribuído

#### Características do **Processo de Filtro**:

- Determina uma conversão na mensagem de comunicação entre o usuário e o *host*. Ex.: Ligação de um desktop com um mainframe através de um emulador

#### Características do **Processo Peer-to-Peer** (não hierárquicos):

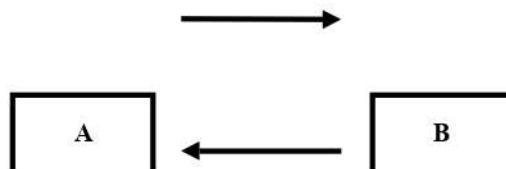
- São processos “clones” rodando em todas as máquinas e prestando serviços uns para os outros.
- Não existem processos servidores, estabelecendo um Servidor dedicado.
- Cada processo pode ser Cliente e Servidor para outros processos.



Durante a execução das consultas, os processos Cliente e Servidor são confundidos com os processos Peer-to-Peer, porque este processo foi desenvolvido com base na LU6. 2 (Logical Unit versão 6.2) do SNA (*Systems Network Architecture*) da IBM (RENAUD,1994).

#### Características do **Processo Cliente / Servidor**:

- Existem processos distintos: o processo cliente é diferente do processo servidor.
- Os processos servidores tornam a estação Servidora dedicada ao seu trabalho.
- Processos clientes são sempre clientes.
- Processos servidores podem se tornar processos clientes de outros Servidores.



O modelo que utiliza processos Cliente/Servidor depende do cliente para inicializar a comunicação. Caso o Servidor comece a comunicação, o processo instalado fica sendo o Peer-to-Peer

A característica básica da arquitetura Cliente/Servidor é a que processos Clientes enviampedidos a um processo Servidor, que retorna o resultado para o Cliente. O processo Cliente fica então liberado da ação do processamento da transação podendo realizar outros trabalhos.

### 8.1 - Camadas da Arquitetura Cliente / Servidor

A arquitetura Cliente/Servidor é dividida em três camadas básicas, como ilustra a figura 8. A camada de Aplicação consiste dos processos da aplicação, entre eles, os processos Cliente e Servidor. A camada de Serviços de Sistemas compreende o Sistema Operacional (SO) e o Sistema Operacional de Rede (SOR), destinando-se ao controle do hardware. Por último a camada de hardware, onde estão localizados os periféricos, ligados aos Clientes e Servidores.



Figura 7 - Três Camadas da Arquitetura Cliente/Servidor

A tecnologia Cliente/Servidor pode existir tanto no nível da camada de Aplicação, quanto no da camada de Serviços do Sistema. A coexistência do paradigma nestas camadas surge em função da hierarquia das atuações no sistema. Caso o “usuário” seja externo ao sistema, então os processos Cliente e Servidor compõem a camada da Aplicação, enquanto que, se o “usuário” for um programa de aplicação o Cliente é um processo redirecionador, e o Servidor será um processo respondedor da camada de Serviços dos Sistemas.

A utilização de sistemas Cliente/Servidor pela camada de aplicação utiliza Serviços do Sistema não Cliente/Servidor (figura 8), entretanto, sistemas não Cliente/Servidor, ao nível da aplicação utilizam Serviços do Sistema Cliente/Servidor (RENAUD, 1994).

Para sistemas Cliente/Servidor na camada de aplicação, a camada Serviços do Sistema oferece somente um mecanismo de IPC (InterProcess Communication) para troca de mensagens. Por outro lado, a camada Serviços do Sistema configurado como Cliente/Servidor, é responsável por gerenciar o redirecionamento das solicitações de gravação/leitura, por exemplo. É importante notar que a diferença entre os sistemas Cliente/Servidor nas camadas de Aplicação e Serviços do Sistema, é o equilíbrio entre a quantidade de processamento tanto no lado do Cliente quanto no lado Servidor.

Existem vários sistemas que podem ser baseados na estrutura Cliente/Servidor. O uso mais frequente são as aplicações de Banco de Dados usando processos SQL (*Structured QueryLanguage*) de front-end, para acessar remotamente, as bases de dados.

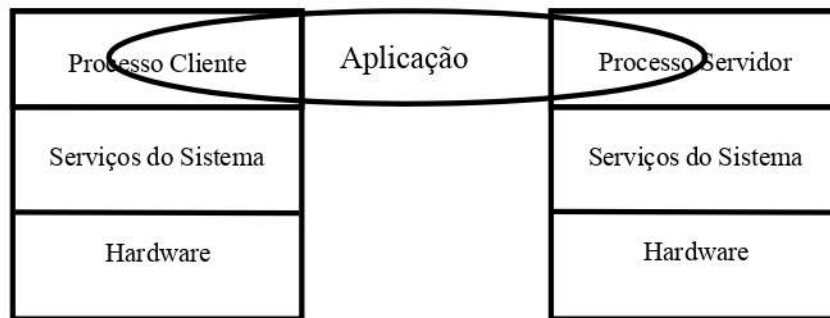
A figura 7 mostra uma estrutura baseada num Servidor de Arquivos. Esta estrutura ocasiona um maior fluxo de informações na rede, uma vez que todo o arquivo será transferido para o Cliente para então ser trabalhado.

A figura 7 demonstra outra possibilidade de se estruturar um Sistema baseado na arquitetura Cliente/Servidor, que consiste na utilização de um Servidor de Banco de Dados dedicado, podendo coexistir normalmente com o Servidor de arquivos. Neste

momento, com um Servidor de Banco de Dados exclusivo, o fluxo de informações trafegadas na rede diminui, já que, somente a resposta da consulta é retornada ao Cliente, ao invés de transferir todo o arquivo como era feito anteriormente.

O Cliente, neste tipo de estrutura é o usuário, passando a ter uma visão da aplicação como se as partes, Cliente e Servidor, fossem algo único.

A figura 11 exemplifica esta visão.



Servidor de Banco de Dados

Este tipo de estrutura favorece o aumento da performance da rede de comunicação, possibilitando assim, um maior número de ligações simultâneas de diversos Clientes com diversos Servidores (RENAUD, 1994).

Embora a arquitetura Cliente/Servidor possa parecer uma nova versão do modelo de arquivos compartilhados, através da Rede Local baseada em microcomputadores, suas vantagens são inúmeras. Fundamentalmente, ambos os modelos provêm capacidade de processamento distribuído e permitem o compartilhamento de informação. Entretanto, no Modelo de Arquivos compartilhados baseado em Servidor de Arquivos, o Cliente além de executar a aplicação, executa também o motor da Base de Dados, que por sua vez acessa essas Bases de Dados remotamente como se fossem locais. O Servidor de Arquivos envia arquivos inteiros através da rede para o Cliente processar localmente, ocasionando congestionamento na rede, enquanto que, no modelo Cliente/Servidor, o Cliente executa parte da aplicação, sendo deixado para o Servidor a tarefa da administração da Base de Dados, comumente

de dados apropriado, como resultado da consulta efetuada pela aplicação.

Desta forma, na arquitetura Cliente/Servidor, cada Servidor pode suportar um número maior de usuários, uma vez que é o Cliente que gerencia a aplicação e a interface com o usuário. Além do mais, com a crescente conectividade entre máquinas e sistemas operacionais, pode-se escolher para Cliente o ambiente de software e hardware que melhor se adequa às necessidades de cada aplicação do usuário, sem ter que se preocupar com o Servidor.

A melhor divisão de tarefas entre o Cliente e o Servidor depende de cada aplicação em si. Se o Servidor for apenas um Sistema Gerenciador de Banco de Dados, deixando para o Cliente o resto do processamento, ou se algumas tarefas como controle de acesso, análise dos dados, e validação dos comandos é executada pelo Servidor, esta é uma decisão do construtor do Sistema em função das características do negócio do usuário. Estas diferenças serão mostradas no tópico a seguir.

## 9 - Redes de Computadores

Normalmente várias perguntas são feitas para se tentar entender como clientes e servidores se comunicam, que tipo de protocolo é utilizado, e como um cliente encontra o servidor desejado dentro de uma rede de computadores. Embora estas perguntas possam parecer muito difíceis, na realidade elas são relativamente fáceis.

Existem vários conceitos e aspectos comuns às comunicações Cliente/Servidor necessários ao entendimento do funcionamento da comunicação. Nesta seção será dada uma visão geral sobre os conceitos básicos de protocolos, passagem de mensagem, os aspectos de conexão e sincronismo de processo.

### 9.1 - Protocolos

**9.2 Há duas décadas, diversos esforços vêm sendo traçados para se estabelecer um padrão único para redes de computadores. No entanto, foram definidos não apenas um, mas vários modelos de referência.**

### 9.3 - O Modelo de Referência OSI/ISO

Em março de 1977, foi constituído pela ISO um grupo de trabalho para estudar a padronização da interconexão de sistemas de computação. Foi definida, inicialmente, uma arquitetura geral, denominada *MODELO DE REFERÊNCIA OSI* (*Open System Interconnection*), para servir de base para a padronização da conexão de sistemas abertos (TAROUÇO, 1986) (TANENBAUM, 1996)

O modelo OSI possui sete camadas, como apresentado na figura 13. Cada camada especifica um grupo específico de tarefas de comunicação. A norma descreve o escopo funcional de cada camada e os requisitos para a interface com as camadas adjacentes (serviços).

|                               |                                                            |
|-------------------------------|------------------------------------------------------------|
| <b>Camada de Aplicação</b>    | - Seleção de serviços apropriados à aplicação.             |
| <b>Camada de Apresentação</b> | - Formatação/Reformatação de dados. Ex.: Criptografia.     |
| <b>Camada de Sessão</b>       | - Interface do usuário para o estabelecimento de conexões. |
| <b>Camada de Transporte</b>   | - Controle de intercâmbio de mensagens entre usuários.     |
| <b>Camada de Rede</b>         | - Controle de intercâmbio de mensagens na rede.            |
| <b>Camada de Enlace</b>       | - Transmissão livre de erros.                              |
| <b>Camada de Física</b>       | - Interface elétrica para transmissão bit a bit.           |

Figura 13 - Camadas do Modelo de Referência OSI

A figura 14 mostra que quando a mensagem passa da camada  $n + 1$  para a camada  $n$  não são acrescentados outros dados relevantes à camada  $n$ . Estes dados são retirados quando a mensagem chega na camada de mesmo nível na estação destino. Estes acréscimos podem ser informações tais como: tipo da mensagem, endereços, tamanho da mensagem, código de detecção de erro, etc.

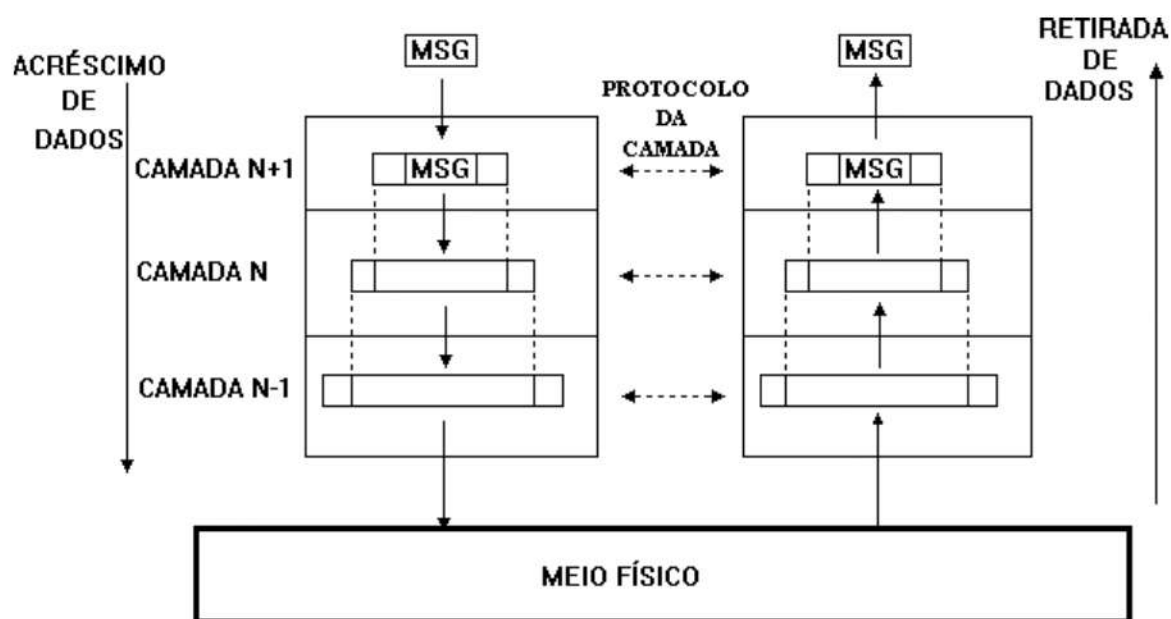


Figura 14: Arquitetura de protocolos em camada

O modelo OSI/ISO especifica todas as primitivas de comunicação para que haja troca de mensagens entre as camadas (AMARAL,1993) (COMER,1993) (DUMAS,1995) (RENAUD,1994) (TANENBAUM,1996). Cada camada adiciona um cabeçalho para que haja uma identificação na máquina destino.

A figura 15 mostra os principais protocolos usados nas comunicações Cliente / Servidor baseados na Microsoft, Internet e IBM.

Figura 15 - Comparação entre o modelo OSI / ISO e outros Protocolos

Tendo em vista os objetivos do trabalho pretendido pela presente tese, somente o protocolo de comunicação TCP/IP será descrito, em função da sua interação com *sockets*, já que este será o meio de comunicação adotado na implementação do protótipo aqui proposto. O leitor poderá referenciar-se a (COMER,1993) (DAVIS,1994) TANENBAUM96] para obter maiores detalhes sobre os demais protocolos.

## 10 - TCP/IP

O TCP/IP distingue-se dos demais protocolos pelo seu endereçamento universal. Cada máquina em uma rede possui um endereço que a identifica. A camada TCP é orientada à conexão, enquanto a camada IP trabalha sem conexão (COMER,1993) (DAVIS,1994)(DUMAS,1995). Cabe à camada IP o trabalho de distribuir os datagramas entre as máquinas de uma rede. Para fazer este serviço, ele possui um único endereço de 32 bits que contém informações suficientes para identificar univocamente uma rede e um determinado computador.

Este endereço é comumente escrito em decimal de quatro bytes separados por um ponto. Por exemplo:

11011111, 00000001, 00000010, 00000101, representaria o endereço 223.1.2.5.

Existem 5 classes onde estes endereços são divididos. A figura 16 mostra estas classificações. (QUINN,1996) (RENAUD,1994)

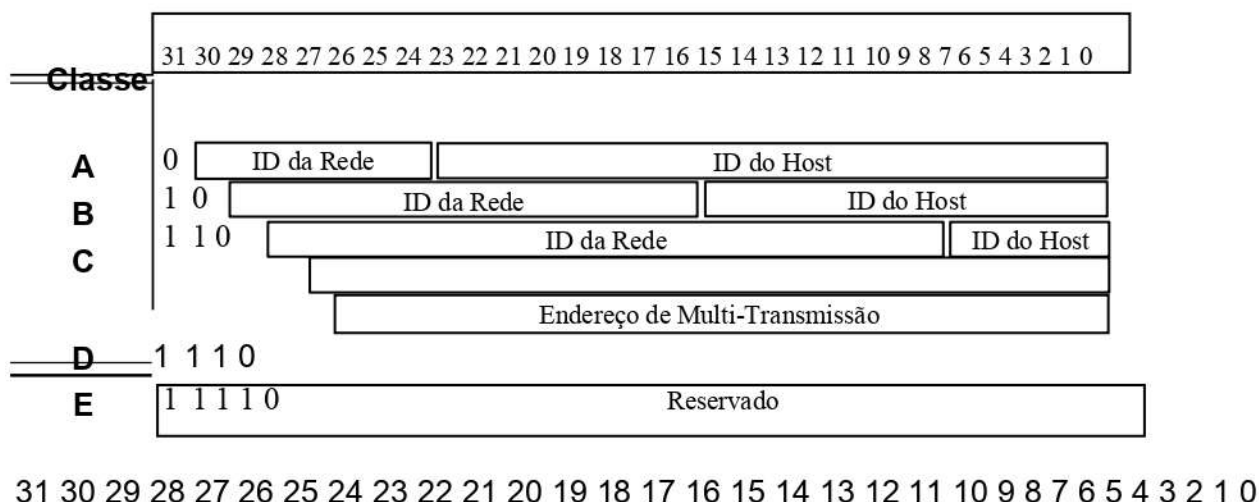


Figura 16 - Classes do protocolo TCP/IP

O número de redes e *hosts* que podem ser endereçados pelas Classes A, B, C, D e E, bem como o escopo de seus endereços IP são definidos na tabela 2.

Por exemplo, é possível endereçar até 65.136 *hosts* para cada rede da classe B. Portanto,  $65.136 \times 16.382 = 1.067.057.952$ . Isto quer dizer que mais de 1 milhão de *hosts* podem ser endereçáveis pela classe B, já que certos endereços IP não podem ser utilizados por serem destinados a trabalhos específicos, como o endereçamento de Gateways e de *Broadcasting*.

| Class es | Num. de Redes | Endereço de Rede            | Num. de Hosts | Endereço de Hosts   |
|----------|---------------|-----------------------------|---------------|---------------------|
| A        | 128           | 0.0.0.0 - 127.0.0.0         | + 16M         | 0.0.0 - 255.255.255 |
| B        | 16.284        | 128.0.0.0 - 191.255.0.0     | 65.136        | 0.0 - 255.255       |
| C        | + 2M          | 192.0.0.0 - 223.255.255.255 | 256           | 0 - 255             |
| D        |               | 224.0.0.0 - 239.255.255.255 |               |                     |
| E        |               | 240.0.0.0 - 255.255.255.255 |               |                     |

Tabela 2 - Limite do número de redes e hosts em cada endereço IP

Com isto, quando se tem um endereço de rede na classe B e deseja-se aumentar o número de redes possíveis, em detrimento da capacidade de máquinas que

podem ser instaladas, ou vice-versa, utiliza-se a técnica da máscara de rede (*NetMask*). Esta técnica permite a criação de subredes através da modificação local da linha divisória que separa os bits de identificação das redes e dos *hosts*. Cada subrede também é representada pela notação de pontodecimal. Os bits da *NetMask* definidos como “um” identificam a porção da rede, enquanto os definidos por “zero”, representam os *hosts*.

Por exemplo, caso uma empresa possua o endereço IP 166.78.4.139 da classe B, isto indica que o computador reside na rede 166.78 e que tem uma identificação de host 4.139. Porém, caso os administradores da rede decidam utilizar a *NetMask* 255.255.255.0 para aumentar o número de redes internas à empresa, é necessário que se faça uma combinação AND, do endereço IP original com a *NetMask*, alterando a identificação de rede para 166.78.4 e a identificação de host para 139. Pode-se dizer com isso, que o computador é o host 139 na subrede 166.78.4.0 (endereço de gateway).

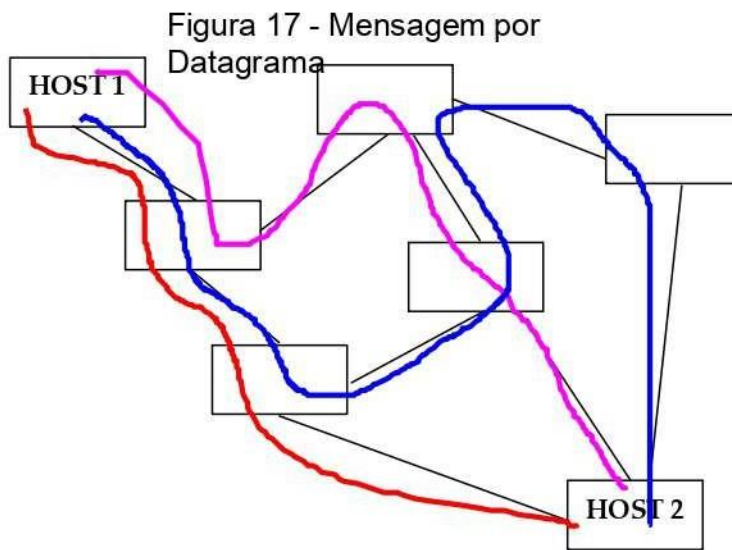
## **12- Aspectos de Conexão**

Existem dois tipos de conexões usados para a comunicação. O primeiro é chamado *sem conexão*, onde cada mensagem encontra seu próprio caminho até o seu destino (COMER,1993) (DUMAS,1995) (QUINN,1996). Cada mensagem é independente de todas as outras mensagens, podendo atingir rotas diferentes até o seu destino. A Internet utiliza este tipo de conexão.

Logo, cada mensagem deve ter toda a informação de endereçamento necessária para a sua entrega. Dependendo do serviço de entrega usado, as mensagens podem chegar fora de ordem ou serem mal encaminhadas no trânsito.

As mensagens numa comunicação *sem conexão* são chamadas de *datagramas*. Essa forma de comunicação também é conhecida como *comutação de pacotes*. Pode-se fazer uma analogia deste tipo de comunicação com o serviço de entrega dos correios.

Não existe uma rota preestabelecida quando se utiliza o envio de mensagem por meio de datagrama. A figura 17 mostra os diversos caminhos que as mensagens podem percorrer.



A mensagem, que é dividida em vários pacotes, pode seguir caminhos distintos, tanto do *host 1* para o *host 2*, quanto no sentido inverso, podendo, inclusive, chegar ao seu destino em tempos diferentes, havendo assim, a necessidade de reordenação da mensagem pela parte responsável da comunicação e pela aplicação do *host* de destino.

O outro tipo de comunicação é o *baseado em conexão*, onde um *circuito* prévio é estabelecido antes

que a troca de dados se efetue. Quando um circuito estiver conectado, cada mensagem segue uma sequência e sempre é direcionada ao longo do mesmo circuito (COMER, 1993) (DUMAS, 1995) (QUINN, 1996).

Logo, cada mensagem só precisa de um identificador de circuito para ser direcionada para o seu destino. O recebimento de cada mensagem normalmente é confirmado e, se for o preciso, o controle de fluxo é usado para regular a velocidade com que as mensagens são enviadas.

A comunicação *baseada em conexão*, conforme apresentado na figura 18, garante a transmissão das mensagens. Estas são também divididas em pacotes, sem perda para o seqüenciamento dos pacotes nos *hosts* destinos, diminuindo assim o trabalho de implementação na parte de comunicação do aplicativo.

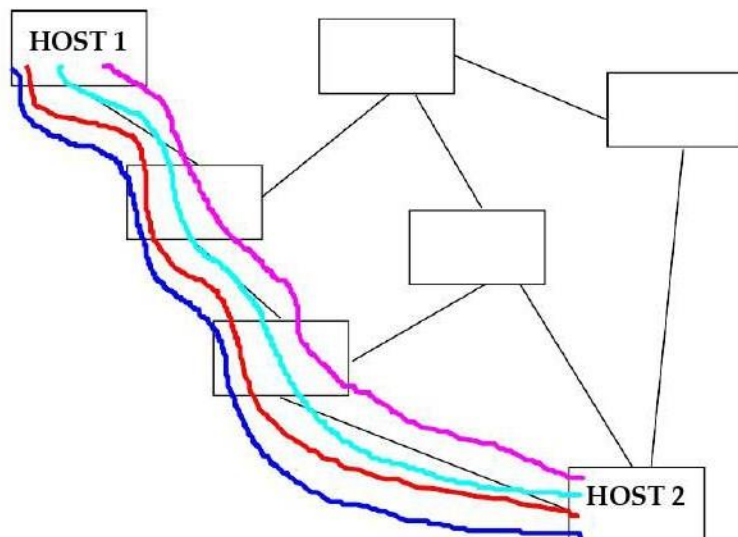


Figura 18 - Mensagem baseada em conexão

Quando a comunicação é finalizada, torna-se necessário desfazer a conexão para poder liberar os recursos de rede utilizados.

Deve-se notar que as mensagens sempre chegam ordenadas neste tipo de conexão.

Um exemplo de comunicação baseado em conexão o sistema telefônico.

A tabela 3 a seguir compara os dois tipos de comunicação.

| 10.1 Característica              | Baseado em Conexão                                                                      | Sem Conexão                                                         |
|----------------------------------|-----------------------------------------------------------------------------------------|---------------------------------------------------------------------|
| <b>Tipo de Mensagem</b>          | Fluxo de dados                                                                          | Datagrama                                                           |
| <b>Rota</b>                      | Estática                                                                                | Dinâmica                                                            |
| <b>Endereçamento de Mensagem</b> | Endereço de destino completo para estabelecer circuito; depois somente o ID do circuito | Endereço de destino completo em todas as mensagens                  |
| <b>Confiabilidade</b>            | Seqüenciado, controle de erro e fluxo, entrega garantida                                | Sem garantias: mensagens podem ser perdidas ou chegar fora de ordem |
| <b>Opções</b>                    | Podem ser negociadas durante a preparação                                               | Não disponíveis                                                     |

| Sincronismo | Explícito                          | Implícito        |
|-------------|------------------------------------|------------------|
| Overhead    | Preparação e liberação do circuito | Rota da mensagem |

Tabela 3 - Tipos de Conexão

## 10.2 - Aspectos de Sincronismo e Passagem de Mensagem

A comunicação entre Cliente e Servidor procede de forma implícita. Quando o Cliente espera a resposta da mensagem enviada para continuar o seu processamento, diz-se que o protocolo utilizado é um protocolo com *bloqueio*, onde o sincronismo entre Cliente e Servidor está implícito no mecanismo de passagem de mensagem (TANENBAUM,1995).

Caso o Cliente possa continuar suas tarefas, enquanto espera a resposta da mensagem, o protocolo de comunicação é um protocolo *sem bloqueio*. Isto ocorre quando o sistema operacional do Cliente é multitarefa ou multiprocessamento, possibilitando ao Cliente executar outras tarefas enquanto aguarda a resposta do Servidor.

A teoria de programação concorrente é baseada na noção de processos de comunicação sendo executados em paralelo a outros processos. Esses processos se comunicam

compartilhando memória ou passando mensagens por meio de um canal de comunicação compartilhado (AMARAL,1993) (VIDAL,1994). O termo IPC (Interprocess Communication) se refere às técnicas utilizadas na passagem de mensagem.

No compartilhamento de memória, os processos concorrentes compartilham uma ou mais variáveis, e utilizam a mudança de estados dessas variáveis para se comunicarem. Essa técnica inclui espera ocupada, semáforos, regiões críticas condicionais e monitores (TANENBAUM,1995). Como esta técnica exige que os processos estejam na mesma máquina, não são considerados base para a programação Cliente/Servidor.

Em técnicas baseadas na passagem de mensagem, os processos enviam e recebem mensagens explicitamente, em vez de examinar o estado de uma variável compartilhada (AMARAL,1993) (COMER,1993). O benefício principal da passagem de mensagem é que existe pouca diferença entre o envio de mensagens a processos remotos ou locais. Portanto, a passagem de mensagem é poderosa para criação de aplicações em rede. Outra vantagem é que mais informações podem ser trocadas numa mensagem do que por mudança no estado de uma variável compartilhada.

A figura 19 mostra a troca de mensagem entre cliente e servidor sem o bloqueio por parte do cliente.

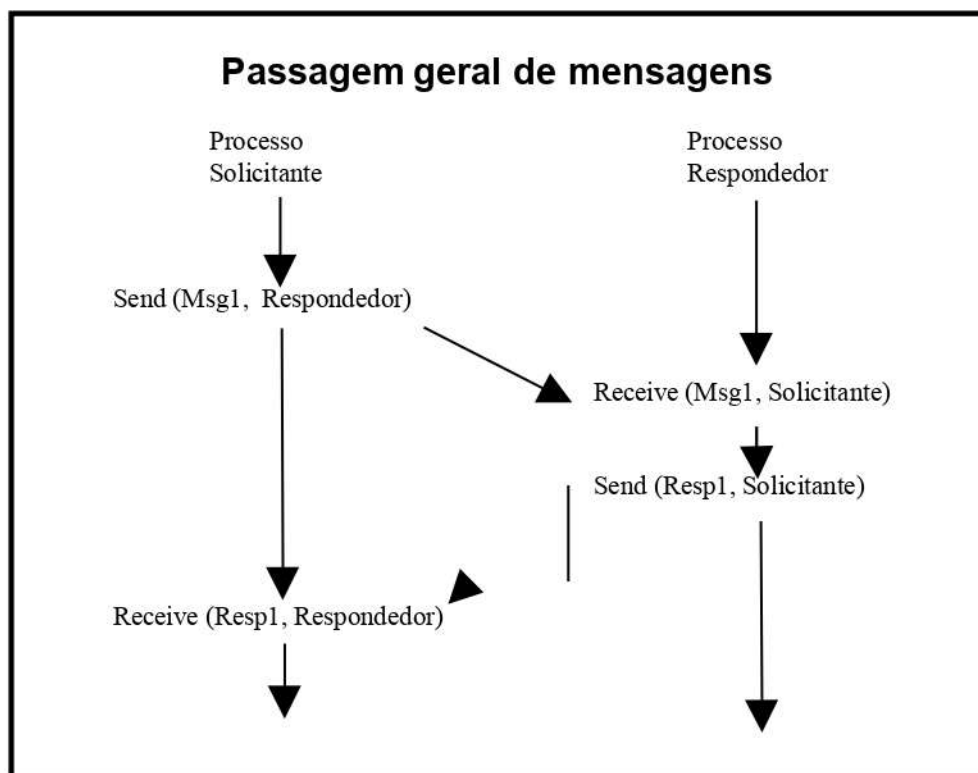


Figura 19 - Troca de mensagem sem bloqueio

### 10.3 - Conexão TCP/IP

Uma vez que a base de comunicação iria ser efetuada por meio de *sockets*, foi estabelecido também que o protocolo de comunicação utilizado seria o TCP/IP, possibilitando assim uma possível conexão futura com a plataforma UNIX.

Para este desenvolvimento, a plataforma a ser utilizada é o Windows NT 3.5 ou 3.51, que apresenta a robustez necessária para os processos multitarefas. A escolha deste ambiente também se deve ao fato dele suportar vários tipos de protocolos para comunicação, entre eles o TCP/IP (QUINN,1996).

A figura 20 mostra os protocolos de comunicação do Windows NT.

O protocolo TCP/IP é o protocolo padrão utilizado pela Internet, que possibilita a interligação de diversas plataformas. Existem vários serviços que utilizam o protocolo TCP/IP, entre eles pode-se encontrar o FTP, Telnet, E-mail, Ping e Finger, como pode ser visto na figura 21, junto com sua referência à camada do modelo OSI/ISO.

Na camada de Transporte pode-se notar a existência de dois padrões, o TCP, que é a comunicação com conexão e o UDP, que é a comunicação sem conexão, como já foi mencionado anteriormente na seção 4.2. Para o desenvolvimento deste protótipo foi adotado o padrão TCP, onde o controle de erro e o seqüenciamento dos pacotes das mensagens são automaticamente tratados, o que não ocorre no padrão UDP, necessitando que estes tratamentos sejam desenvolvidos.

O padrão TCP permite a abertura de uma conexão virtual entre a máquina fonte e a destino em todas as camadas do modelo OSI/ISO.

|   |                                                               |                                  |                                  |                                  |         |
|---|---------------------------------------------------------------|----------------------------------|----------------------------------|----------------------------------|---------|
| 7 | Aplicação<br>Windows sockets                                  | Aplicação<br>Windows sockets     | Aplicação<br>Windows sockets     | Aplicação<br>Windows sockets     |         |
| 6 |                                                               |                                  | APIs proprietárias<br>(opcional) | APIs proprietárias<br>(opcional) |         |
| 5 |                                                               | APIs proprietárias<br>(opcional) | Aplicação<br>(cont.)             | Aplicação<br>(cont.)             |         |
|   |                                                               |                                  |                                  | APIs proprietárias<br>(opcional) |         |
|   |                                                               |                                  |                                  | Aplicação<br>(cont.)             |         |
|   |                                                               | Windows Socket API               |                                  |                                  |         |
|   | <b>Windows sockets Dynamic Link Library<br/>(Winsock.DLL)</b> |                                  |                                  |                                  |         |
|   | APIs proprietárias                                            |                                  |                                  |                                  |         |
| 4 | TCP<br>/IP                                                    | DEC<br>Net                       | NetBEUI                          | Appletalk                        | SPX/IPX |
| 3 |                                                               |                                  |                                  |                                  |         |
|   | Driver de APIs padrões                                        |                                  |                                  |                                  |         |
| 2 | Driver de Protocolos Múltiplos (ODI, Packet<br>Driver, NDIS)  |                                  |                                  |                                  |         |
|   | APIs de Hardware proprietário                                 |                                  |                                  |                                  |         |
| 1 | Interface de Rede (Ethernet, Token Ring,<br>Serial, etc.)     |                                  |                                  |                                  |         |

Figura 20 Protocolos do Windows NT

|                    |  | <b>Modelo OSI/ISSO</b> | <b>Protocolo TCP/IP</b> |
|--------------------|--|------------------------|-------------------------|
| Camadas superiores |  | Aplicação              | Aplicação               |
|                    |  | Apresentação           | <b>FTP, SMTP e DNS</b>  |
| Camadas inferiores |  | Sessão                 | <b>Telnet</b>           |
|                    |  | Transporte             | <b>TCP e UDP</b>        |
|                    |  | Rede                   | <b>IP, ARP e ICMP</b>   |
|                    |  | Link de Dados          | Link de Dados           |
|                    |  | Física                 | Física                  |

Figura 21 Protocolo TCP/IP

A Internet utiliza o padrão UDP, uma vez que as comunicações baseadas no protocolo TCP/IP se utilizam de portas de comunicação que, associadas aos *sockets*, permitem a troca de mensagens.

O protocolo TCP/IP disponibiliza 999999 portas (comprovação empírica), sendo que as portas de número 0 até a porta de número 1023 são reservadas para serviços pré-determinados, como por exemplo, a porta 23, para o serviço Telnet, a porta 21 para o FTP, e assim por diante. Devido a este limite, caso a Internet utilizasse o padrão TCP, as máquinas que respondessem a um número muito grande de acessos, acabariam limitando a sua utilização.

No padrão UDP, por sua vez, quando existe uma solicitação de comunicação, o endereço IP do remetente da mensagem segue junto com a mensagem para o destinatário, de forma que o disponibilizador do serviço possa posteriormente enviar a resposta do serviço solicitado.

É importante salientar que, para que as comunicações transcorram normalmente, as APIs não necessariamente precisam ser as mesmas, muito embora os protocolos devam ser os mesmos, como pode ser visto na figura 22.

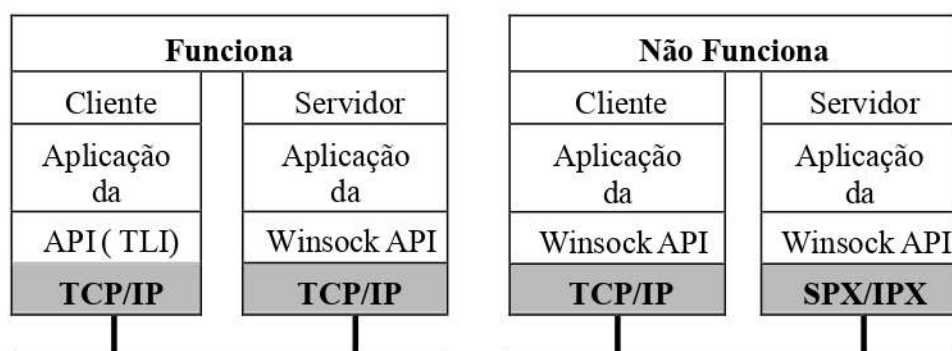


Figura 22 - Mesmo protocolo para uma comunicação correta

Os demais protocolos suportados pelo Windows NT, permitem a conexão com outros tipos de plataformas, como pode ser visto na tabela 4.

| Protocolo  | Plataforma |
|------------|------------|
| NetBEUI    | Microsoft  |
| I          | Novell     |
| SPX/IP     | Mcintos    |
| X          | h          |
| Apple Talk |            |

Tabela 4 - Demais protocolos do Windows NT

O protocolo NetBEUI é de propriedade da Microsoft e permite a conexão com máquinas que estejam com os ambientes Windows 3.x, Windows NT 3.x / 4.0 e Windows 95 (DAVIS,1994). Ele é o responsável pela camada de Transporte e Rede, acrescentando uma extensão para a camada de Link de Dados. Conecta-se diretamente com o NetBIOS na camada de sessão (RENAUD,1994), que oferece as funcionalidades de transporte e rede do NetBEUI, como por ser visto na figura 15. Possui dois tipos de frames, um para fornecer um fluxo de dados confiável e o outro

para frames de informação não numerados usados em Datagramas.

Os protocolos SPX/IPX são uma implementação dos protocolos XNS da Xerox. O protocolo SPX, igual ao SPP da Xerox, oferece um serviço de fluxo de dados confiável,

enquanto que o IPX, igual ao IDP, oferece um serviço de datagrama, permitindo a conexão entre máquinas do tipo PC que estejam rodando o Sistema Operacional de Rede da Novell (RENAUD,1994).

Por sua vez, o protocolo Apple Talk permite a conexão com máquinas da plataforma Macintosh, e é um protocolo totalmente particular. Os principais protocolos usados por máquinas Macintosh são os DDP - Datagram Delivery Protocol, ATP - AppleTalk Transaction Protocol e ADSP - AppleTalk Data Stream Protocol (RENAUD,1994).

## 11 - Sockets

O *Socket* teve origem na Universidade de Berkeley, como sendo a API (Application Programming Interface) de desenvolvimento do protocolo TCP/IP para o ambiente UNIX (COMER,1993). Ele é um dos mecanismos que o IPC possui para a troca de mensagens entre processos.

Um *Socket* é similar a um descritor de arquivo. Ele identifica o ponto final (*endpoint*) para a comunicação e é implementado como um inteiro positivo (ROBERTS,1995).

Existe uma diferença sutil, porém importante, entre descritores de arquivos e *socket*. Um descritor de arquivo é ligado a um arquivo ou dispositivo específico quando criado pelo comando *open*. Um descritor de *socket* *não* é ligado a local algum quando criado pelo comando *socket*. Uma aplicação pode decidir ligar-se a um endereço explicitamente usando o comando *bind*, ou pode fornecer endereços dinamicamente quando envia datagramas usando o comando *send*. Portanto, *sockets* *podem* ser usados como uma interface para transportes de rede baseados *em conexão* e *sem conexão* (COMER,1993) (DUMAS,1995).

O *socket*, que é um manipulador (*handle*), está associado a um largo conjunto de dados armazenados na implementação do protocolo de rede. O termo *socket* é usado no S.O. UNIX para fornecer uma interface tipo arquivo às redes. Quando uma operação para criar um *socket* é chamada, o sistema retorna um manipulador, como descritor de *socket*. Esse descritor é usado em todos os outros comandos relacionados ao *socket* e é intercambiável com o descritor de arquivo usado em funções *read* e *write*.

Os dados associados ao *socket* incluem várias informações, como o endereço IP das máquinas que estão se conectando, as portas dos dois lados da conexão TCP e o estado da conexão corrente.

A camada de *sockets* dentro do contexto da comunicação corresponde a camada de Sessão do Modelo OSI/ISO, que tem como função o gerenciamento das sessões de comunicação processo a processo entre os *hosts*. Ela é também responsável por estabelecer e encerrar as conexões entre os aplicativos cooperantes (DUMAS,1995).

Dentro do protocolo TCP/IP, a camada TCP abrange a camada de Sessão e Transporte do Modelo OSI/ISO, favorecendo diretamente a programação dos *sockets* sobre o protocolo TCP/IP.

Para o ambiente Windows foi desenvolvido o *Windows Sockets - Winsock*, visando facilitar a interconexão entre as estações Windows, através do protocolo TCP/IP, que é base para o acesso a Internet. *Windows Sockets* é uma interface aberta para programação em rede sob o Microsoft Windows (QUINN,1996).

A especificação da interface do Windows sockets define claramente a divisão entre a aplicação de rede e o protocolo da rede.

A sua versão mais recente é o Winsock 2 (revisão 2.0.8 - 19/05/95). A versão 2.0 da especificação do Windows sockets se associa a arquitetura que o Windows NT utiliza para suportar múltiplos protocolos de fornecedores variados (QUINN,1996).

O Winsock API aumenta a funcionalidade dos *sockets* de Berkeley, ao acrescentar extensões específicas do Windows para suportar a natureza orientada a mensagens do S.O. baseado no Windows.

Todas as aplicações que hoje em dia acessam a Internet diretamente da residência do usuário, usando FTP, E-mail, Finger, Telnet, SMTP, entre outros, utilizam os sockets como base de comunicação, através do protocolo TCP/IP (ROBERTS,1995).

Os Sistemas Operacionais Windows NT e Windows 95 já possuem dentro de suas bibliotecas, as rotinas para suportar a programação para o protocolo TCP/IP via sockets (DAVIS,1994).

Para o ambiente Windows, o relacionamento entre as aplicações e o ambiente de rede, pode ser demonstrado na figura 23.

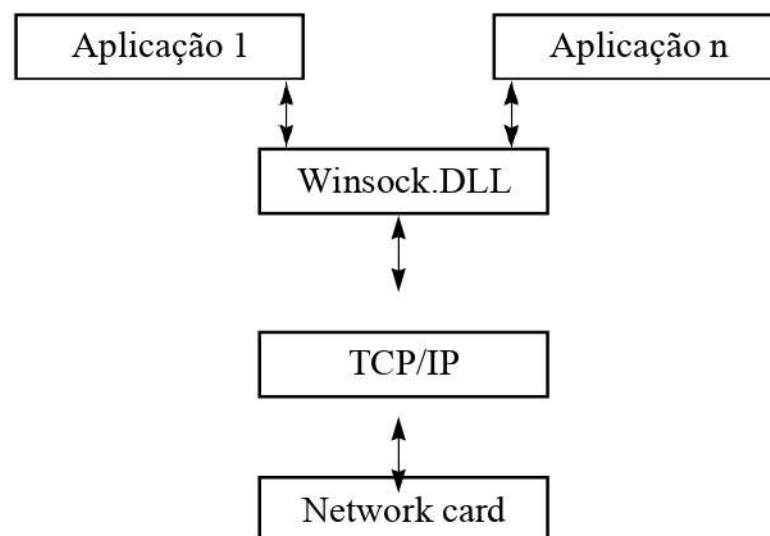


Figura 23 - Relacionamento da biblioteca dos *sockets* no ambiente Windows

Na API dos sockets é possível determinar se a comunicação que se vai estabelecer é *com conexão - STREAM*, ou *sem conexão - DATAGRAMA*.

Para se trabalhar numa comunicação *baseada em conexão*, na camada de transporte do TCP/IP, o protocolo utilizado é o TCP, determinando que somente após o estabelecimento de uma comunicação é possível efetuar troca de mensagens.

Para a abertura de uma conexão com *sockets*, é necessário que várias funções da biblioteca Winsock sejam chamadas. A figura 24 demonstra a seqüência de

operações tanto no lado Servidor quanto no lado Cliente.

No caso de se estabelecer uma conexão Stream, o Servidor é primeiramente inicializado. A função Socket define o descritor no qual a aplicação se associará quando desejar transmitir uma mensagem. Posteriormente a função Bind interliga o descritor ao endereço IP da máquina servidora e a porta TCP/IP na qual as

Uma vez estabelecida a conexão, a troca de mensagem passa a ser efetuada e a função Recv retira do buffer de leitura as informações que foram trocadas entre o Cliente e o Servidor. Caso não se consiga enviar alguma mensagem pela função Send, por causa do estouro da capacidade do buffer, os dados são armazenados e retransmitidos posteriormente.

Os dados OOB (*Out-of-Band*) possuem um nível de prioridade de transmissão acima dos outros dados. Isto possibilita uma alteração na sequência a ser transmitida, porém esta interferência na sequência fica transparente ao usuário.

Ao término da comunicação, o Cliente utiliza a função CloseSocket para fechar a conexão, liberando a porta do processo filho do Servidor para ser ligada a outro processo de comunicação. O Servidor só irá utilizar esta função quando ele for desligado.

Portanto, primeiro o Servidor é posto no "ar" (1), em seguida o Cliente escolhe o nome do Servidor e a porta no qual será feita a conexão (2a) e solicita uma conexão (2b), quando a conexão é aceita, o Servidor cria um processo filho, numa nova porta TCP/IP (3), associando o pedido do Cliente à esta porta, permitindo assim a troca de mensagens (4), deixando a porta

original da conexão do Servidor livre para efetuar novas conexões, como pode ser visto na figura 28.

existentes, e permite que a função CloseSocket seja ativada automaticamente ao se verificar a inexistência de qualquer tipo de troca de mensagem após um determinado tempo, possibilitando assim a otimização das portas do protocolo TCP/IP.

## **12 - Bancos de Dados Orientados a Objetos**

### **12.1 - Introdução**

Os Sistemas Gerenciadores de Bancos de Dados Orientados a Objetos (SGBDOOs) surgiram da necessidade de suportar a programação orientada a objeto. Era necessário um repositório de dados para guardar os objetos criados pelas linguagens Otaias como a Smalltalk e a C++, entre outras. Esses objetos eram conhecidos como objetos persistentes (MARTIN, 1994) (NAVATHE, 1994), ou seja, dados que permanecem no sistema após a conclusão de um processo.

Os SGBDOOs tornaram-se importantes para certos tipos de aplicações com dados complexos, tais como CAD (*Computer Aided Design*) e CAM (*Computer Aided Manufacturing*) e o SIG (*Sistemas de Informações Geográficas*), servindo também para manipular BLOBs (Binary Large Objects), a exemplo de imagem, som, vídeo, texto e transações de longa duração (MARTIN, 1994) (NAVATHE, 1994) (KHOSHAFIAN, 1993) (RUMBAUGH, 1994).

A para o outro. Contudo, quando uma quantidade grande de objetos é transferida, primeiramente eles são empacotados para minimizar os acessos a rede.

## 12.2 - SGBDOO GemStone

O OODBMS GemStone (GEMSTONE,1997), que utiliza a técnica de objetos distribuídos, oferece uma arquitetura Cliente/Servidor escalável e robusta com o seu sistema de controle de objetos. A comunicação do GemStone se baseia no ORB (*Object Request Broker*), que por sua vez utiliza o CORBA (*Common Object Request Broker Architecture*) para organizar os padrões dos objetos que servirão para a troca de mensagens. A OMG (*Object Management Group*) garante então, os padrões de interoperabilidade entre os sistemas de objetos gerenciados pelo CORBA, como visto nas figuras 30 e 31.

A OMG deliberou os seguintes padrões de trabalho para o Inter-ORB:

- ◇ o GIOP (*General Inter-ORB Protocol*) que especifica o formato das mensagens e a representação dos dados para toda a intercomunicação do ORB. O GIOP garante, portanto, que a ordenação dos bytes transmitidos não são problemas entre os ORBs existentes.
- ◇ o IIOP (*Internet Inter-ORB Protocol*) que permite a ligação entre o GIOP e o protocolo de comunicação TCP/IP.

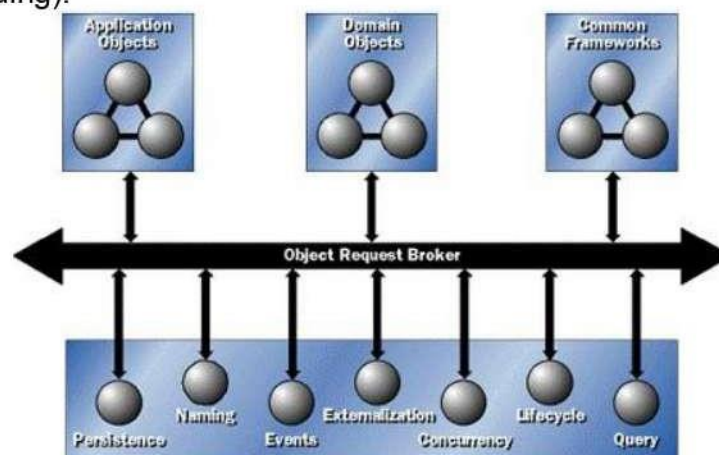
O CORBA 2.0 possui ambos os protocolos de comunicação: o GIOP e o IIOP.

A arquitetura de objetos distribuídos do CORBA estende os limites do GemStone para as linguagens de objetos adicionais, interfaces de programação de aplicações e para serviços não objetos.

A adoção da padronização CORBA garante a interoperabilidade entre os diversos sistemas de objetos nas empresas.

A grande migração da indústria de sistemas monolíticos para sistemas distribuídos, fez com que a OMG focasse o CORBA como seu ponto de ação central, reunindo seus esforços para a padronizar os objetos componentes de sistemas de informações distribuídos.

A plataforma Windows desenvolve seus aplicativos baseados na tecnologia de objetos distribuídos através do COM (Component Object Model) e o OLE (Object Linking and Embedding).



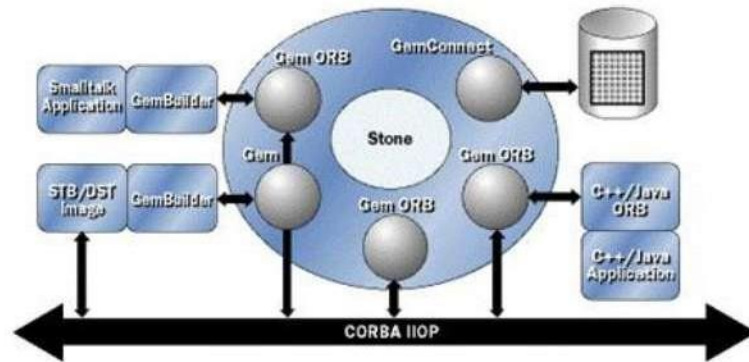


Figura 31 - Padronização CORBA para a Comunicação de Objetos Distribuídos

### 12.3 - SGBDOO POET

Outro OODBMS conhecido que também possui uma arquitetura Cliente/Servidor é o POET. O POET Engine e a aplicação cliente coexistem como um processo sem qualquer sobrecarga na comunicação (POET,1997).

A firmeza no gerenciamento dos dados junto com as facilidades da API disponíveis, permite ao usuário desenvolver aplicativos com características Cliente/Servidor, com uma total transparência da localização dos dados.

Uma aplicação pode abrir múltiplos banco de dados, que podem ser locais ou remotos. O UOS (*Universal Object Server*) do POET emprega uma arquitetura Cliente/Servidor convencional, na qual múltiplos clientes acessam um ou mais servidores através da rede. Esta estrutura de comunicação representa uma arquitetura Cliente/Servidor em dois níveis

formando uma comunicação mista podendo ser vista na figura 2.2c.

O UOS utiliza protocolos de rede padrão como o TCP/IP e o IPX/SPX, permitindo assim a operação transparente em redes homogêneas ou heterogêneas, fornecendo ao usuário o máximo de flexibilidade.

O sistema POET suporta tanto *single-threaded* quanto *multi-threaded*. Os clientes *single-threaded* incluem um *thread-safe* API que permite a serialização das chamadas para API do POET. Portanto, quando ocorre uma chamada, esta chamada fica em estado de bloqueio. Os clientes *multi-threaded* suportam sistemas operacionais *multi-threaded*, incluindo o Windows NT, OS/2 e UNIX.

*Single-threaded* é uma operação sequencial na qual são alocadas as informações pertinentes ao processo, tais como identificação do processo e área de memória utilizada em cada *thread* criado. Entretanto, *Multi-threaded* permite o compartilhamento destas informações, quando da criação de processos filhos no processo pai, otimizando assim, a utilização da memória (PHAM,1996). Os clientes incorporam a arquitetura de múltiplos leitores / único escritor, (*multiple reader/single writer*) usando consultas não bloqueadas e canais de comunicação simultâneos para um ou mais servidores POET.

### 13 ASPECTOS CONCLUSIVOS/CONCLUSÃO/CONSIDERAÇÕES FINAIS

Atualmente, a maioria das empresas está migrando para a computação distribuída, através de uma arquitetura Cliente/Servidor. Por ser uma área nova, é necessário o investimento em equipamentos, pessoal, treinamento e serviços para conseguir responder a demanda deste mercado globalizado.

A importância e a credibilidade desta tecnologia podem ser verificadas em função de grandes empresas, tais como: Oracle, Sybase, Informix, Digital e IBM, que desenvolveram soluções para atender a essa nova plataforma. Esta oferece acesso a dados localizados em diferentes Servidores, sem que o usuário perceba que eles podem estar vindo até de países diferentes, realidade facilmente conseguida através da Internet. Esta facilidade permite uma integração entre micros, *mainframes* e redes, a fim de se obter a melhor que a computação distribuída pode oferecer.

Este relatório apresentou as características básicas da arquitetura Cliente/Servidor.

A arquitetura Cliente/Servidor demonstra que veio realmente para ficar, e nada indica que um novo modelo esteja prestes a tirar o seu lugar.

Para acompanhar o avanço tecnológico nesta área, esta arquitetura foi implementada no SIGO (Sistema Gerenciador de Objetos) (MOURA, 1997), transformando-o de um sistema monousuário em um sistema multiusuário (DESCHAMPS, 1997).

## REFERÊNCIAS

1. (AMARAL,1993) Amaral, W. H. "Arquitetura Cliente/Servidor Orientada a Objeto" Tese de Mestrado, IME, 1993.
2. (ANDREWS,1991) Andrews, G. R. "Concurrent Programming: Principles and Practice" Benjamin/Cummings, Redwood City, CA, 1991.
3. (AQUINO,1995) Botelho, Tomás de Aquino Tinoco, "Análise de Desempenho da Arquitetura Cliente/Servidor Orientada a Objeto", Tese de Mestrado, IME, dezembro/1995.
4. (BANCILHON,1992) Bancilhon, F. et al "Building an Object-Oriented Database System: The Story of O<sub>2</sub>", Morgan Kaufmann, 1992.
5. (COMER,1993) Comer, Douglas E. & STEVENS, David L. "Internetworking With TCP/IP Vol.III: Client/Server Programming and Application (Socket Version)". Prentice Hall, Englewood Cliffs, New Jersey, USA, 1993.
6. (CUSTER,1993) Custer, Helen "Windows NT", Microsoft Press, Makron Books, SãoPaulo, 1993.
7. (DAVIS,1994) Davis, Ralph "Windows NT Networking Programming" Addison Wesley, 1994
8. (DESCHAMPS,1997) Deschamps, Eduardo R.P. "Sistema Gerenciador de Objetos emum Arquitetura Cliente/Servidor", Tese de Mestrado, IME, abril/1997.
9. (DUMAS,1995) Dumas, Arthur "Programando WinSock", Axcel Books, Rio de Janeiro, 1995.
10. (GEMSTONE,1997) GemStone Inc. "GemStone's Role in CORBA System"  
<http://gemstone.com/Products/WP/corbawp.htm>.
11. (HULQUIST,1997) Hultquist, Steve <ssh@vnet.ibm.com> "FAQ about Client/Server"<http://non.com/news.answers/client-server-faq.html>, 1997
12. (KALAKOTA,1997) Kalakota, Ravi "FAQ about Client/Server"  
<kalakota@uhura.cc.rochester.edu> <http://non.com/news.answers/client-server-faq.html>.
13. (KHOSHAFIAN,1993) Khoshafian, Setrag "Object Oriented Database", John Wiley & Sons, Inc., Canadá, 1993.
14. (LEFEBVRE,1994) Lefebvre, Alain "L'Architecture Client-Serveur", Armand Colin, 2<sup>e</sup>édition, 1994.

15. (MARTIN,1994) Martin, James "Princípios de Análise e Projeto baseados em Objetos"Ed. Campus, Rio de Janeiro, 1994.
16. (MCKIE,1997) McKie, Stewart "Everything you wanted to know about Client/Servercomputing but were afraid to ask" Eye on Technology,  
<http://www.duke.com/controller/Issues/DecJan/Clientse.htm>
17. (MOURA,1997) Moura, A.M.C.; Freitas, Luciani et al; "SIGO: Ambiente para Desenvolvimento de Aplicações Não-Convencionais em Banco de Dados" Relatório Técnico, RT016/DE9/ABR97, IME, Rio de Janeiro, Abri/1997.
18. (NAVATHE,1994) Navathe, Shamkant B. & Elmasri, Ramez "Fundamentals ofDatabase Systems" 2nd Ed., Benjamin Cummings, CA, 1994.