UNIVERSIDADE ESTADUAL DE CAMPINAS

FACULDADE DE ENGENHARIA ELÉTRICA

DEPARTAMENTO DE ENGENHARIA DA COMPUTAÇÃO E AUTOMAÇÃO INDUSTRIAL

Este ex... da tese
defe... Guido José M. Villavi-
cencio ... pela Comissão
Julgad... 03 07 92
Maurício Ferreira Magalhães
...dor

# UMA METODOLOGIA DE CRIAÇÃO DE SISTEMAS DE APOIO A PROJETOS

Por: Guido José Martins Villavicencio
Orientador: Maurício Ferreira Magalhães

Tese apresentada à Faculdade de Engenharia Elétrica da Universidade Estadual de Campinas como parte dos requisitos para a obtenção do Título de Mestre em Engenharia Elétrica

Julho 1992

UNICAMP
BIBLIOTECA CENTRAL

# AGRADECIMENTOS

Agradeço ao professor Maurício pela orientação, interesse e apoio dedicados ao longo de todo o desenvolvimento deste trabalho.

Agradeço, também, ao professor Manuel de Jesus Mendes pelo suporte e atenção dispensados.

Sou grato, ainda, ao professor Luís Gimeno pela orientação durante a fase de cumprimento dos créditos que antecedeu o início deste trabalho.

Agradeço a meus pais, irmãos e demais familiares pelo incentivo e suporte fornecidos durante toda esta longa jornada.

A minha companheira Marilena, por partilhar tanto das alegrias quanto das desilusões experimentadas durante a evolução deste trabalho, sem esquecer, também, de sua paciência e incentivo, minha eterna gratidão, extensiva, também, a seus familiares.

A todos os meus amigos da FEE, entre os quais: Manoel, Thomas, José, Wolfgang, Alexandre, Gualberto, Sinésio, Suhel e todos os que, de alguma forma, me deram apoio, o meu muito obrigado também.

Este trabalho contou o apoio financeiro da Coordenação de Aperfeiçoamento do Pessoal do Ensino Superior - CAPES.

# RESUMO

O presente trabalho discorre sobre o desenvolvimento de ambientes de apoio a projetos. A esse tipo de ambiente demos a designação de "Sistema de Apoio a Projetos" (SAP).

O objetivo principal é a proposta uma metodologia de desenvolvimento de tal tipo de sistema, o qual deve auxiliar o projetista na condução do projeto, liberando-o de tarefas repetitivas e monótonas sujeitas, portanto, a erros.

Como parte integrante do sistema, deverão ser fornecidas diversas ferramentas de apoio, inclusive um sistema de gerenciamento de banco de dados, uma interface-usuário baseada em janelas e menus, ferramentas para manipulação de listas e árvores, simuladores, etc.

Essa metodologia é composta de cinco fases sucessivas, a saber: a primeira fase trata da identificação do problema e dos recursos necessários ao SAP para solucioná-lo; na segunda fase é efetuada a configuração do SAP de forma a atender os requisitos impostos pelo projeto; a terceira fase corresponde à criação de formulários de pesquisa e de uma Base de Dados para o sistema; a quarta fase envolve a coleta de dados através dos formulários e seu armazenamento na Base de Dados; na quinta fase ocorre a manipulação das informações por parte de programas componentes do SAP, com possibilidade de criação de novos formulários e de novos arquivos da Base de Dados.

Ao final do trabalho é apresentado um estudo de caso da aplicação da metodologia proposta, na implementação de um sistema de auxílio a projetos de redes locais de computadores.

# ABSTRACT

The subject of this work is the proposal of an environment to support the development of projects. We call this type of environment: "Project Support System" (PSS).

The main objective is the proposal of a methodology of development regarding this kind of system, which must help a designer conducting the project, by means of freeing him from repetitive and monotonous tasks which are source of errors.

As part of the system, there will be a number of support tools, including a Database Management System, a windows and menus user-interface, tools to manipulating lists and trees, simulators, etc.

This methodology has five compound phases: the first phase deals with problem identification and the resources needed to solve them; in the second phase we configure the PSS in a way that it can meet the requisites imposed by the project; the third phase corresponds to the creation of query forms and a database for the system; the fourth phase deals with gathering data by means of the forms and storing this data in the system's database; in the fifth phase there will be data manipulation by the PSS' compound programs, possibly with recursive creation of additional forms and database files.

As a corollary of the work we present a case study of the application of the proposed methodology on the implementation of a local area network project support system.

# PREFÁCIO

Optou-se, neste trabalho, por preservar o original em inglês de alguns termos do jargão de informática, quando da inexistência de correspondente em português ou, em certos casos, para preservar a precisão e a acuracidade das idéias envolvidas. Na maioria dos casos, porém, as palavras estão seguidas da original em inglês entre parênteses (ou vice-versa), com vistas a uma melhor compreensão da idéia relacionada ao termo em questão.

# ÍNDICE

**Capítulo 1 - Introdução** ..... 1

1.1) Introdução ..... 2
1.1.1) Histórico ..... 2
1.1.2) O enfoque da Engenharia Concorrente (E.C.) ..... 2
1.2) Sistemas de Apoio a Projetos ..... 5
1.2.1) S.A.P. ..... 5
1.2.2) Banco de Dados ..... 7
1.2.3) Interfaces ..... 7
1.2.4) Formulários ..... 7
1.3) Objetivo do Estudo ..... 8
1.4) Linguagem de Programação utilizada ..... 9
1.5) Organização do restante da Tese ..... 10

**Capítulo 2 - Formulação do Problema** ..... 11

2.1) Introdução ..... 12
2.2) Formulação da Situação Problema ..... 12
2.2.1) Requisitos Gerais ..... 13
2.2.2) Requisitos Específicos ..... 17
2.2.2.1) Requisitos do Sistema de Gerenciamento de Banco de Dados (SGBD) ..... 17
2.2.2.2) Requisitos das Interfaces ..... 19
2.2.3) Outros Requisitos ..... 21

**Capítulo 3 - Bancos de Dados (BAPAS-DB)** ..... 23

3.1) Introdução - Bancos de Dados ..... 24
3.1.1) Item de dado ou Campo (Data Item or Field) ..... 24
3.1.2) Registro (Record) ..... 24
3.1.3) Arquivo (File) ..... 25

3.1.4) Banco de Dados ou Base de Dados
(Data Base or Data Bank) ..... 25
3.1.5) Sistema de Gerência de Banco de Dados - SGBD
(Data Base Management System - DBMS) ..... 25
3.1.5.1) Linguagem de Descrição de Dados
(Data Description Language - DDL) ..... 26
3.1.5.2) Linguagem de Consulta
(Query Language - QL) ..... 27
3.1.6) Tipos de Modelos de Dados ..... 27
3.1.6.1) Modelo de Rede ..... 27
3.1.6.2) Modelo Hierárquico ..... 28
3.1.6.3) Modelo Relacional ..... 28
3.1.6.4) Modelo Entidade ..... 28
3.2) BAPAS-DB ..... 29
3.2.1) Modelo ..... 30
3.2.2) Componentes do BAPAS-DB ..... 33
3.2.2.1) Programa de Instalação INSTAL ..... 33
3.2.2.2) Interpretador DDL ..... 33
3.2.2.3) Programa de Reconfiguração RECONF ..... 35
3.2.2.4) System Kernel ..... 36
3 2 2 5) DB-client e DB-server ..... 36
3.2.2.6) QL a linguagem de consulta ..... 37
3.2.3) Propriedades de acesso em Tempo Real ..... 37
3.2.3.1) Acessos controlados por prioridades ..... 37
3.2.3.2) Operação contínua (sem reorganizações) .... 38
3.2.3.3) Tempos de acesso garantidos ..... 38
3.2.3.4) Aplicações especiais podem ter
buffers especiais ..... 38
3.2.3.5) Arquivos na memória principal ..... 38
3.2.3.6) Aplicações em sistemas sem disco
(diskless) ..... 39
3.2.4) Características de Sistema Aberto e Flexível
do BAPAS-DB ..... 39
3.2.4.1) Várias estratégias de acesso ..... 39
3.2.4.2) Sistema padrão de buffers flexível ..... 43
3.2.4.3) Sistemas de buffers específicos para
arquivos especiais ..... 44

3.2.4.4) Arquivos na memória principal ..... 45
3.2.5) Consistência, segurança e proteção de dados no BAPAS-DB ..... 45
3.2.5.1) Estabelecimento de "checkpoints" com "before images" ..... 45
3.2.5.2) Recuperação automática ..... 46
3.2.5.3) "Log book" (after image) ..... 46
3.2.5.4) Transações (atômicas) ..... 47
3.2.5.5) Discos duplicados (mirror disks) ..... 48
3.2.5.6) Sistemas duplicados ..... 49
3.2.5.7) Segurança de dados ..... 51
3.2.6) A interface de programas do BAPAS-DB ..... 51
3.2.6.1) A interface de "baixo nível" QDBS1 ..... 52
3.2.6.2) A interface DBV (programa/BAPAS-DB) ..... 53
3.2.6.2.1) Linguagens suportadas pela interface DBV ..... 55
3.2.6.2.2) Estratégias de acesso ..... 55
3.2.6.2.3) Acesso à Base de Dados usando "C" .. 56
3.2.7) A interface SQL do BAPAS-DB ..... 57
3.2.8) Propriedades de Banco de Dados Distribuído ..... 58
3.2.9) BAPAS-DB como uma Base de Dados ativa ..... 60
3.2.10) Ferramentas para o BAPAS-DB ..... 61
3.2.11) Níveis de configuração e dados técnicos ..... 61
3.2.12) Extensões ..... 62

**Capítulo 4 - Outras Ferramentas** ..... 64

4.1) Introdução ..... 65
4.2) A Interface-Usuário ..... 65
4.2.1) Janelas ..... 66
4.2.1.1) Coordenadas de uma janela 66
4.2.1.2) Componentes de uma janela 68
4.2.1.2.1) Elementos ..... 68
4.2.1.2.2) Atributos ..... 69
4.2.1.3) Estrutura de dados da janela ..... 69
4.2.1.4) Funções relacionadas a janelas ..... 71
4.2.2) Menus ..... 72

4.2.2.1) Componentes ..... 73
4.2.2.2) Estrutura de dados do menu ..... 74
4.2.2.3) Funções relacionadas a menus ..... 75
4.2.3) Lista de cadeias de caracteres ..... 76
4.2.3.1) Estrutura de dados da lista de cadeias de caracteres ..... 76
4.2.3.2) Funções relacionadas a listas de cadeias de caracteres ..... 78
4.2.4) Descrição dos módulos ..... 78
4.3) Árvores ..... 79
4.3.1) Estrutura de dados de uma árvore ..... 80
4.3.2) Ordenamento de nós ..... 81
4.3.3) Funções relacionadas a árvores ..... 83
4.3.4) Módulos componentes ..... 83
4.3.5) Exemplos de utilização ..... 84
4.4) Formulários ..... 84
4.5) Ferramentas do Sistema Operacional ..... 85
4.5.1) Funções relacionadas às ferramentas do Sistema Operacional ..... 86
4.5.2) Módulos componentes ..... 86

**Capítulo 5 - Metodologia** ..... 87

5.1) Introdução ..... 88
5.2) A Metodologia ..... 88
5.2.1) 1ª fase - Formulação do Problema ..... 88
5.2.1.1) Caracterização do Projeto ..... 90
5.2.1.2) Requisitos Operacionais ..... 91
5.2.2) 2ª fase - Configuração do Sistema ..... 91
5.2.3) 3ª fase ..... 92
5.2.3.1) Elaboração dos formulários ..... 92
5.2.3.2) Criação da Base de Dados ..... 94
5.2.4) 4ª fase - Coleta de Informações ..... 95
5.2.4.1) Preenchimento dos formulários ..... 95
5.2.4.2) Preenchimento da Base de Dados ..... 96
5.2.5) 5ª fase - Manipulação da Informação ..... 97

**Capítulo 6 - Estudo de Caso** ..... 99

6.1) Introdução - Redes Locais ..... 100
6.1.1) Importância ..... 101
6.1.2) Histórico ..... 103
6.1.3) Topologias ..... 105
6.1.3.1) barramento ..... 106
6.1.3.2) anel ..... 106
6.1.3.3) estrela ..... 107
6.1.3.4) árvore ..... 108
6.1.4) Meios de Transmissão ..... 109
6.1.4.1) par trançado ..... 109
6.1.4.2) cabo coaxial ..... 110
6.1.4.3) fibra ótica ..... 110
6.1.5) Disciplinas de acesso ao meio ..... 111
6.1.5.1) CSMA e CSMA/CD ..... 111
6.1.5.2) Token Passing ..... 112
6.1.6) Padrões ..... 113
6.1.6.1) O padrão RM-OSI/ISO ..... 113
6.1.6.2) O padrão IEEE 802 ..... 115
6.1.6.3) O padrão MAP/TOP ..... 116
6.1.7) Perspectivas ..... 118
6.2) NETCAD - um estudo de caso ..... 119
6.2.1) Introdução ..... 119
6.2.1.1) Importância e justificativa do estudo ..... 119
6.2.1.2) Objetivos do sistema ..... 121
6.2.1.3) Amplitude e delimitação do estudo ..... 122
6.2.2) Descrição da metodologia NETCAD ..... 122
6.2.2.1) Os módulos ..... 123
6.2.3) Aplicação da metodologia à problemática de projeto de Redes Locais ..... 128

**Capítulo 7 - Conclusões** ..... 138

7.1) Sumário ..... 139
7.2) Conclusões ..... 140
7.2.1) Sobre o BAPAS-DB ..... 140

7.2.2) Sobre a Interface-Usuário ..... 141
7.2.3) Sobre a Metodologia ..... 141
7.2.4) Sobre o estudo de caso (Netcad) ..... 142
7.2.5) Conclusões Gerais ..... 142
7.3) Sugestões de Pesquisas Adicionais ..... 143
7.3.1) Ampliação da Funcionalidade do Netcad ..... 143
7.3.2) Migração para outras plataformas computacionais .. 143
7.3.3) Utilização da Metodologia em outras Aplicações ... 144

**Bibliografia** ..... 146

**Anexos** (Volume à parte)

# CAPÍTULO 1

## INTRODUÇÃO

## 1.1) Introdução

### 1.1.1) Histórico

Tem-se observado, atualmente, um crescimento considerável nas empresas de um modo geral. A competição entre as empresas tem, paralelamente, se acirrado significativamente. As empresas que conseguem colocar seus produtos no mercado com qualidade superior, a preços competitivos e no menor espaço de tempo, terão maiores chances de sobreviver e crescer.

Um dos fatores chave no sucesso de uma empresa é a maneira pela qual são desenvolvidos os seus projetos.

Projetos, analogamente aos seres vivos, têm um ciclo de vida bem definido. Parte-se da concepção, passa-se pela implementação, e chega-se até a operação e manutenção de um determinado projeto.

Com a evolução da tecnologia, os projetos tendem a tornar-se cada vez mais complexos, longos e dispendiosos.

Outro fator a ser levado em consideração é que as equipes de projeto tendem, devido a essa complexidade, a tornar-se heterogêneas e dispersas geograficamente.

A metodologia utilizada para conduzir os projetos em seus ciclos de vida desempenha, assim, um papel de importância crucial para a empresa pois, utilizando uma metodologia adequada, a empresa pode concluir seus projetos em um tempo mais curto, a um custo mais baixo e com qualidade superior aos projetos de seus possíveis concorrentes.

### 1.1.2) O enfoque da Engenharia Concorrente (E.C.)

Dado este panorama, é necessária uma evolução nas ferramentas e tecnologias de projeto, para satisfazer a demanda competitiva de produzir mais, melhor e mais rapidamente.

Existem diversas maneiras de solucionar o problema de "melhorar" a metodologia de projeto. Uma delas é através da ótica da Engenharia Concorrente (E.C.).

A E.C. baseia-se na busca da "Qualidade Total" [E-*Carter 91*], a qual envolve um balanceamento entre performance, minimização de custos, facilidade de projeto e facilidade de suporte.

Uma das premissas básicas da E.C. é a de não tentar "re-inventar a roda". Deve-se tentar aproveitar as ferramentas existentes e concentrar-se somente em projetar as que forem necessárias no caso de não existência de ferramenta pronta ou, por outro lado, na impossibilidade de integração de tais ferramentas ao sistema.

Um contato constante, ou pelo menos frequente, entre a equipe de projetistas e os potenciais fornecedores de partes pode ser de grande utilidade nesse sentido, além de adicionar ao projeto os conhecimentos do fornecedor a custo muito baixo ou, até mesmo, a custo zero. Desta forma fica, também, mais fácil estimar custos desde as fases iniciais do projeto.

As duas figuras seguintes [E-*Sprague 91*] apresentam, respectivamente, os fluxos de desenvolvimento de projetos segundo a ótica tradicional, caso da primeira figura, e de projetos conduzidos através da E.C..

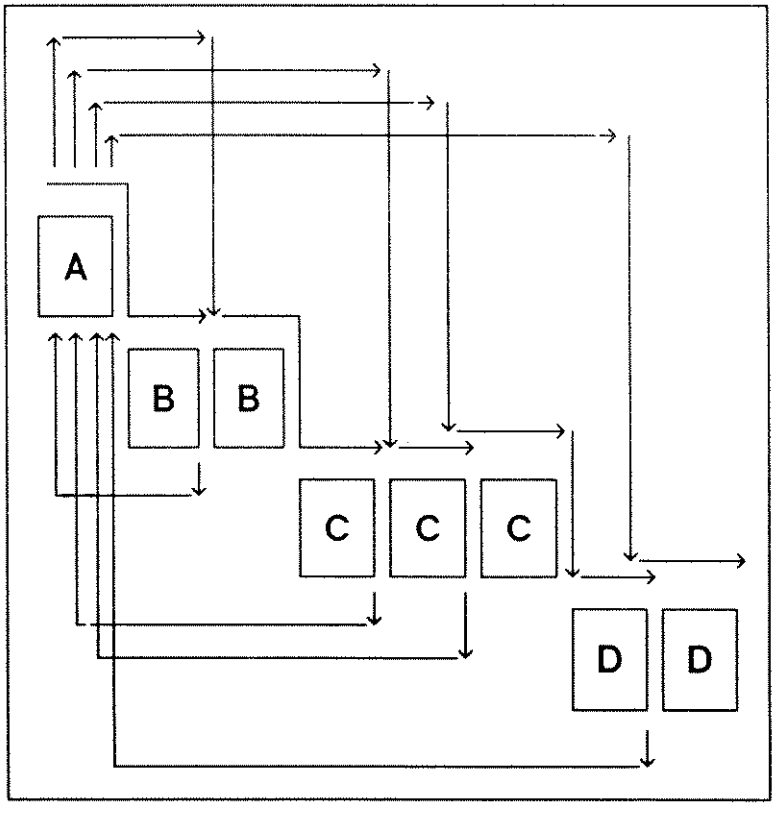



Figura 1.1 - Fluxo de desenvolvimento de projeto (tradicional)

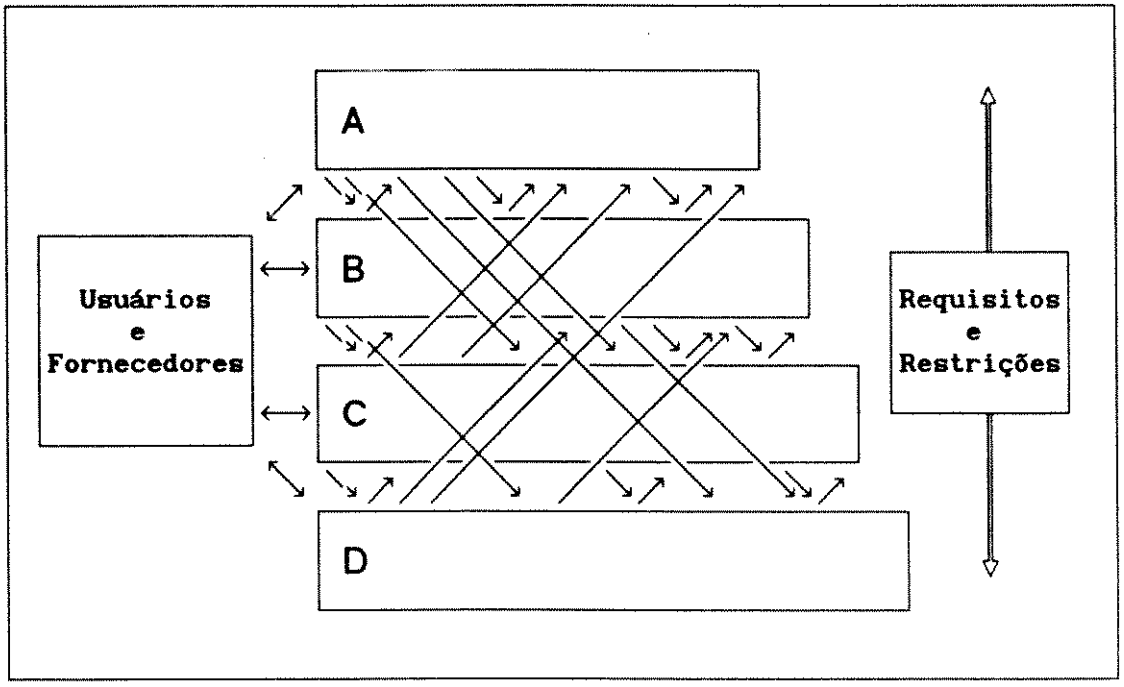



Figura 1.2 - Fluxo de desenvolvimento de projeto baseado em Engenharia Concorrente

Os requisitos básicos exigidos por um ambiente de auxílio a projetos baseado em Engenharia Concorrente são os seguintes [E-*Sprague 91*]:

- Transparência

- Interoperabilidade

- Portabilidade

- Evolução (aperfeiçoamento) contínuo

Esses requisitos levaram a que fosse definido um conjunto de serviços genéricos de gerência de informações para os sistemas que pretendessem utilizar a ótica da engenharia concorrente. Tais serviços modulares informatizados provêem as seguintes capacidades [E-*Sprague 91*]:

- Integração de ferramentas e outros serviços;

- Compartilhamento de informações;

- Comunicações via rede (multimeios) entre peritos, aplicativos, serviços e repositórios de informações;

- Coordenação da equipe;

- Memória corporativa.

## 1.2) Sistemas de Apoio a Projetos (S.A.P.)

Uma das maneiras de aplicar os conceitos da engenharia concorrente para melhorar na prática o processo de desenvolvimento de projetos, em especial os da área de engenharia, se traduz na criação de Sistemas de Apoio a Projetos que implementem partes da funcionalidade descrita para os sistemas baseados em E.C..

### 1.2.1) S.A.P.

Um projeto é um processo contínuo; cada função recebe entradas, produz saídas e vários laços de realimentação, os quais, se forem muitos e muito longos, impõem atrasos significativos no projeto. Assim, um SAP deve ajudar a reduzir o tempo do ciclo total de desenvolvimento de um projeto.

A evolução das ferramentas de projeto, de manuais para automatizadas, libera os projetistas das tarefas repetitivas e sujeitas a erros.

Em um SAP que utilize o enfoque de E.C., as pessoas que estejam trabalhando em um projeto devem poder comunicar-se "instantaneamente" entre si e acessar, compartilhar e armazenar informações atualizadas sobre o projeto de maneira transparente, não obstruída nem limitada pela separação geográfica, estrutura organizacional da empresa, complexidade do projeto, incompatibilidades de ferramentas, bases de dados e recursos computacionais.

Tomadas de decisões relativas ao projeto devem ser efetuadas com rapidez, para que não seja quebrada a continuidade do projeto.

Conforme o projeto evolui, diferentes membros da equipe terão diferentes níveis de influência sobre as decisões pertinentes ao projeto.

Um SAP deve promover a aquisição, rastreamento e satisfação dos requisitos e restrições de projeto durante todo o seu desenvolvimento.

Algumas "decisões chave" tomadas nas fases iniciais de um projeto repercutirão positiva, ou negativamente, em todo o seu ciclo de vida. Assim, um SAP deve auxiliar a identificar os problemas e gargalos de um projeto nas suas fases iniciais, de maneira que se possa eliminá-los ou contorná-los o mais cedo possível evitando, assim, a necessidade de refazer projetos várias vezes para sanar tais problemas.

Deve haver alguma forma de realimentação dos resultados alcançados nos projetos, de forma que o sistema possa "aprender", tanto com seus acertos como com seus erros. Assim, evita-se, por exemplo, cometer o mesmo erro várias vezes.

Para tanto, é de extrema importância um bom controle de projeto e uma boa documentação que inclua todas as decisões importantes e os motivos que levaram a estas decisões.

Projetos de um porte razoável são, na maioria das vezes, conduzidos por equipes "multi-disciplinares". Assim, é aconselhável que o SAP permita, ou mesmo induza, os participantes a adotarem como primeira prioridade o estabelecimento de um vocabulário (jargão) comum. Feito isto, eles podem passar a concordar sobre prioridades genéricas (globais) e somente a partir daí chegarem a consensos sobre prioridades específicas individuais [*E-Carter 91*].

Como é necessária uma forma de armazenar e recuperar as informações referentes a um determinado projeto, a base de um Sistema de Apoio a Projetos é, geralmente, um Sistema de Gerenciamento de Banco de Dados. Assim, grande parte das características do Banco de Dados refletem-se no S.A.P. como um todo.

Portanto, ao se decidir em que Sistema de Gerenciamento de Banco de Dados vai se basear um Sistema de Apoio a Projetos, deve-se proceder com especial cautela, pois as limitações inerentes ao Banco de Dados limitarão, também, o sistema em si; e trocar um Sistema de Gerenciamento de Banco de Dados por outro pode ser uma tarefa muito complexa ou trabalhosa (ou ambas).

Adicionalmente, alguns outros componentes são desejáveis em um Sistema de Apoio a Projetos, por exemplo, uma interface-usuário "amigável".

As seções a seguir, discorrem sobre esses componentes de Sistemas de Apoio a Projetos (em especial os de engenharia).

### 1.2.2) Banco de Dados

É desejável que o Sistema de Gerenciamento de Banco de Dados a ser escolhido como base de um Sistema de Apoio a Projetos tenha algumas características especiais, tais como permitir acesso compartilhado e "on-line" e, adicionalmente, confiabilidade, portabilidade, modularidade e flexibilidade elevadas.

Uma descrição mais detalhada desses requisitos é apresentada no capítulo 2.

### 1.2.3) Interfaces

Um bom programa de aplicação deve possuir uma interface poderosa. Isto é especialmente válido no caso de Sistemas de Apoio a Projetos, sendo extremamente comum julgar-se o mérito de um programa pela qualidade de sua interface.

Assim, deve-se dedicar especial atenção ao projeto da interface, pois ela é a parte do sistema mais visível, do ponto de vista do usuário. Além disso, uma interface bem projetada facilita a interação do usuário com o sistema, o que leva a uma maior produtividade.

O capítulo 2 oferece uma descrição dos requisitos da interface.

### 1.2.4) Formulários

Muitas vezes, grande parte das informações, que necessitam ser coletadas para o desenvolvimento de um projeto, encontram-se dispersas.

Como, em geral, seria inviável que cada indivíduo detentor desse tipo de informação interagisse com o sistema, é conveniente que haja a

possibilidade de obtenção desses dados de forma não interativa, ou seja, através de formulários de consulta gerados a partir do próprio sistema.

No capítulo 4 existem informações adicionais sobre formulários.

## 1.3) Objetivo do Estudo

Tendo em mente a situação-problema sobre a qual discorremos acima, a proposta principal deste trabalho é a concepção de uma metodologia de desenvolvimento de Sistemas de Apoio a Projetos, que simplifique a tarefa de implementar sistemas desse tipo.

Essa metodologia deve ter como base um Sistema de Gerenciamento de Banco de Dados com características de acesso compartilhado e "on-line", confiabilidade e flexibilidade elevadas e, possivelmente, características de tolerância a falhas.

Além disso, como parte integrante de tal metodologia, devem ser fornecidas diversas ferramentas de software que julgamos úteis em sistemas desse tipo, ferramentas estas cuja necessidade detectamos no decorrer deste estudo.

Com o objetivo de avaliar as idéias discutidas neste trabalho, é apresentado um estudo de caso de um sistema implementado de acordo com a metodologia proposta.

Cumpre notar-se que nossa proposta de Tese resumia-se, inicialmente, ao que é agora, neste trabalho, apenas um estudo de caso. Essa mudança de escopo deveu-se à constatação de que a metodologia proposta para essa aplicação particular poderia ser generalizada e, na verdade, ser utilizada no projeto de outros sistemas análogos, sem que isto exigisse modificações substanciais no método proposto.

Assim, pode-se tomar, agora, o estudo de caso apenas como um modelo para a implementação de um Sistema de Apoio a Projetos "genérico", e o objetivo maior deste trabalho passa a definição da metodologia em si.

## 1.4) Linguagem de Programação ser utilizada

A princípio, poder-se-ia considerar a utilização de qualquer linguagem de programação para implementar um S.A.P. nos moldes que descrevemos acima.

Considerações de compatibilidade e modularidade, porém, restringem a nossa escolha a somente duas opções de linguagens de programação, quais sejam: **C** e **PASCAL**.

Em nosso caso escolhemos a linguagem **C** por uma série de fatores, entre os quais podemos citar:

- modularidade;

- portabilidade; superior, no caso, à da linguagem PASCAL;

- a Base de Dados oferece uma interface a processos escritos em linguagem C;

- a interface-usuário (baseada em janelas e menus) descrita no capítulo 4 é escrita em linguagem C;

- vários outros utilitários que utilizamos estão escritos em C.

Assim, na sequência deste trabalho, consideraremos que a linguagem de programação a ser utilizada na codificação de qualquer programa ou sub-rotina necessários seja **C** (em especial, o dialeto Microsoft C [**C**-*MSC 87*]).

## 1.5) Organização do Restante da Tese

O capítulo 2 será dedicado a uma formulação mais específica do nosso problema. Serão definidos os "requisitos" de um Sistema de Apoio a Projetos e serão identificadas as ferramentas necessárias em um ambiente desse tipo.

No capítulo 3 faremos uma pequena introdução sobre Bancos de Dados em geral e apresentaremos a Base de Dados a ser utilizada em nosso estudo.

O capítulo 4 detalha outras "ferramentas" identificadas como úteis em um Sistema de Apoio a Projetos.

O capítulo 5 apresenta a nossa proposta de metodologia de implementação de um Sistema de Apoio a Projetos genérico.

O capítulo 6 é dedicado a um estudo de caso de implementação de um Sistema de Apoio a Projetos segundo a metodologia apresentada no capítulo 5. Como esse estudo de caso diz respeito a Redes Locais de Computadores, uma pequena introdução sobre esse tema é fornecida no início do capítulo.

Finalmente, no capítulo 7 são apresentadas as conclusões referentes ao trabalho.

# CAPÍTULO 2

## FORMULAÇÃO DO PROBLEMA

## 2.1) Introdução

Este capítulo destina-se a uma formulação mais detalhada do nosso problema.

Serão abordados, aqui, os requisitos de um Sistema de Apoio a Projetos (SAP), de modo a identificar as ferramentas necessárias em um ambiente desse tipo.

## 2.2) Formulação do Problema

Tipicamente, um Sistema de Apoio a Projetos é composto por um módulo de interface-usuário, um Sistema de Gerenciamento de Banco de Dados, e um conjunto de ferramentas ou utilitários dispostas ao seu redor. Veja figura.

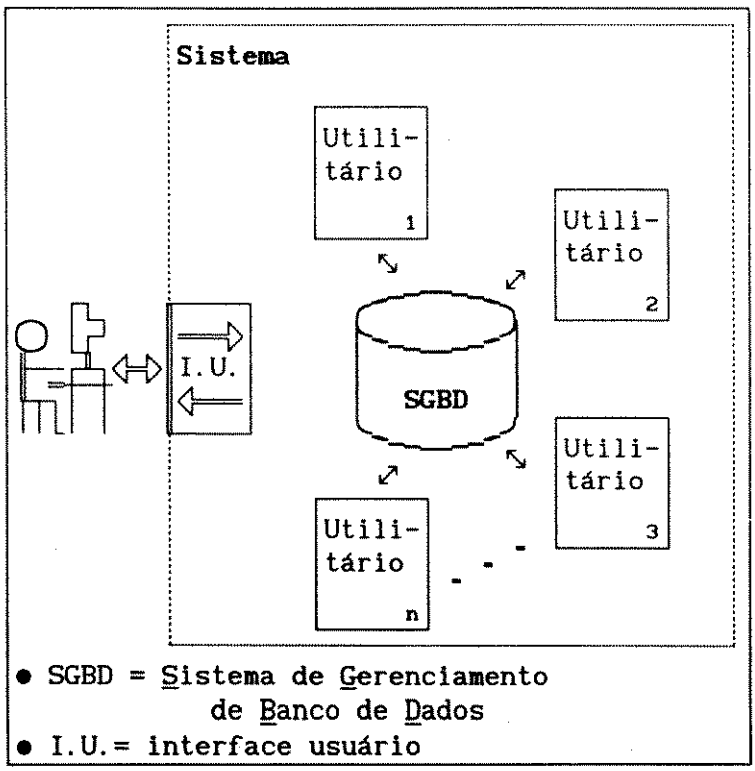



Figura 2.1 - Sistema de Apoio a Projetos

Examinando o diagrama, podemos deduzir que a base de um Sistema de Apoio a Projetos é, sem dúvida, o Sistema de Gerenciamento de Banco de Dados, de maneira que as características deste, tanto as capacidades quanto as limitações, têm reflexos no sistema como um todo.

O ponto de interação entre um usuário e o sistema é chamado de Interface-Usuário. Sendo esta a parte mais visível do sistema, é importante que ela seja bem projetada e concisa, pois isso facilita a interação do usuário com o sistema, o que pode ser um ponto a favor de qualquer sistema.

O número e a funcionalidade das ferramentas (ou utilitários) que compõem um sistema varia com sua finalidade e abrangência. É uma boa idéia manter agrupadas em módulos as ferramentas correlacionadas.

Essa estrutura de SAP, que descrevemos, é "genérica" sendo, portanto, aplicável à maioria dos sistemas desse tipo.

Estes possuem dois níveis de requisitos associados: requisitos gerais do sistema e requisitos específicos a cada componente do mesmo. As seções a seguir detalham cada um deles.

### 2.2.1) Requisitos Gerais

Os requisitos gerais do sistema dizem respeito ao sistema como um todo e são independentes da finalidade específica do sistema e do número de ferramentas que o compõem. A seguir, uma descrição desses requisitos.

*** modularidade**

Existem numerosas vantagens em se utilizar componentes modulares em um sistema. A seguir, enumeramos algumas:

- redução da redundância de código
- favorecimento da consistência dos programas

- possibilidade de reaproveitamento de código

- simplificação da tarefa de substituição e alteração de componentes

Tendo em vista o exposto acima, é importante que o sistema seja constituido de elementos modulares. Assim, um dos módulos do sistema será o Sistema de Gerenciamento de Banco de Dados e a Interface-Usuário será outro. As "ferramentas" componentes do sistema devem ser agrupadas em módulos distintos de acordo com algum critério, por exemplo, funcionalidade.

*** confiabilidade**

Confiabilidade pode ser uma característica chave em um Sistema de Apoio a Projetos. Adulterações ou perdas de dados podem conduzir a resultados imprecisos que levem a perdas de tempo e/ou dinheiro ou, em alguns casos, até mesmo a resultados catastróficos.

Assim, é interessante que o sistema tenha provimentos para detecção, isolamento e, possivelmente, correção de eventuais erros.

O ideal é que o sistema tenha características de tolerância a falhas, isto é, ocorrendo, eventualmente, uma falha em um dos componentes, o sistema possa permanecer em funcionamento, mesmo que de forma parcial e/ou degradada, atenuando os efeitos dessa falha no sistema como um todo.

*** portabilidade**

Portabilidade diz respeito à facilidade de transportar programas, desenvolvidos para um determinado tipo de sistema computacional, para outro sistema computacional qualquer, cujos hardware e software básicos tenham, possivelmente, características bastante diversas.

Dada a velocidade relativamente alta de obsolescência do hardware, a portabilidade pode favorecer em muito a longevidade de um SAP limitada, de outro modo, pelo tempo de vida útil de uma determinada plataforma

computacional.

O ideal seria que um sistema desse tipo fosse "independente" da plataforma computacional onde estivesse instalado. Como não é possível garantir este tipo de característica, mesmo porque não se pode prever o futuro, deve-se tentar manter a parte dependente de harware e softaware básico de um sistema separada da parte potencialmente independente.

* **expansibilidade**

Assim como a obsolescência da plataforma computacional pode limitar a vida útil de um sistema, a impossibilidade de melhorá-lo e expandi-lo pode ser uma limitação muito séria, a qual pode fadá-lo à obsolescência.

Desta maneira, um sistema que tenha a pretensão de sobreviver ao passar do tempo deve ser um "sistema aberto", no sentido de permitir que sejam feitas alterações e adições ao sistema original para torná-lo atualizado.

* **flexibilidade**

A flexibilidade de um sistema tem a ver com a capacidade desse sistema de permitir diferentes configurações para adequar-se às necessidades do usuário ou, por outro lado, às limitações eventualmente impostas pela plataforma computacional.

Esta não é uma característica essencial em um Sistema de Apoio a Projetos; podemos, porém, classificá-la como uma possível qualidade ou virtude de um Sistema de Apoio a Projetos.

* **documentação**

Outra característica que classificamos como "desejável" em um Sistema de Apoio a Projetos genérico, é a possibilidade de geração automática da documentação referente a um projeto em andamento.

Este tipo de facilidade é especialmente importante em sistemas distribuidos, pois vários usuários (ou equipes de usuários) interagirão com o sistema em intervalos de tempo distintos, de maneira que uma boa documentação é essencial para um bom andamento de um projeto nessas condições.

*** requisitos especiais**

Certos tipos de projetos podem impor alguns requisitos não convencionais, tais como acessos concorrentes e "on-line", segurança de dados (possivelmente através de criptografia) e operações em blocos atômicos (transações).

Assim, ao selecionarmos um Sistema de Apoio a Projetos para uma determinada finalidade, devemos verificar se o sistema satisfaz esses eventuais requisitos.

Como tais requisitos não são comumente necessários, este tipo de característica é considerada acessória, não sendo, desta forma, indispensável.

*** linguagem de programação**

Possivelmente, alguns dos componentes de um Sistema de Apoio a Projetos sejam "pacotes fechados". Outros, porém, serão programas "abertos", no sentido de que podem ser adaptados ou modificados de acordo com as necessidades.

Portanto, se houver a possibilidade de escolha, deve-se escolher, para implementar tais programas, uma linguagem de programação única que permita a interoperabilidade dos programas entre si, entre eles e a Base de Dados e, também, a Interface Usuário.

Essa linguagem de programação deve, ainda, possuir boas características de flexibilidade, modularidade, portabilidade e desempenho.

### 2.2.2) Requisitos Específicos

Os requisitos descritos acima são bem genéricos, pois são relativos ao sistema como um todo.

Além de satisfazer esses requisitos "gerais", alguns requisitos mais específicos, referentes aos elementos "chave" de um Sistema de Apoio a Projetos, devem ser, também, observados.

As seções a seguir detalham esses requisitos para cada um dos componentes.

#### 2.2.2.1) Requisitos do Sistema de Gerenciamento de Banco de Dados (SGBD)

Como já foi dito anteriormente, o alicerce de um Sistema de Apoio a Projetos é sempre um Sistema de Gerenciamento de Banco de Dados.

Apresentaremos aqui algumas das características "desejáveis" em um Sistema de Gerenciamento de Banco de Dados a ser utilizado como a base de um Sistema de Apoio a Projetos.

*** acessos concorrentes**

Vários usuários devem poder acessar o Banco de Dados simultaneamente[1]. Tais usuários podem ser tanto pessoas acessando o Banco de Dados interativamente, bem como processos (programas) que, em seu código, contenham chamadas ao Banco de Dados.

*** acesso em tempo-real**

Pode ocorrer que certos usuários, geralmente processos, tenham neces-

[1] "simultâneo" no sentido usado em sistemas de tempo compartilhado

sidade de que o sistema efetue determinadas operações em tempos garantidamente pequenos; a isso chamamos acesso em "tempo-real".

A noção de "tempo-real" é extremamente subjetiva: para um sistema pode ser um intervalo de tempo na casa dos segundos, enquanto que para outro, esse intervalo pode ser da ordem de milésimos de segundos.

O importante, no caso, é que o sistema garanta tempos máximos de resposta compatíveis com as necessidades dos usuários.

*** confiabilidade**

A confiabilidade é uma característica essencial em um Sistema de Banco de dados a ser utilizado como base em um Sistema de Apoio a Projetos, pois adulterações nos dados armazenados podem conduzir a resultados imprecisos ou, até mesmo, de consequências desastrosas.

Assim, o SGBD deve ter maneiras de assegurar que algumas operações críticas sejam efetuadas sem erros ou, na ocorrência de uma falha, prover maneiras de recuperar os dados a um estado consistente.

Deve existir, também, um esquema de contenção de disputas, pois um usuário pode tentar recuperar dados do Banco de Dados, enquanto outro usuário efetua uma atualização nesses mesmos dados. Situações como esta podem levar a inconsistências potencialmente perigosas.

É interessante, ainda, que o Sistema de Banco de Dados tenha características de tolerância a falhas, isto é, na eventualidade da ocorrência de uma falha, deve ser possível manter o sistema em funcionamento (mesmo que parcial), minimizando as consequências da falha.

*** modularidade**

É importante que o SGBD seja construido de forma modular. Isto traz diversas vantagens, entre as quais podemos citar: evitar redundâncias de código, favorecimento da consistência e do reaproveitamento de programas e simplificação de modificações e atualizações dos módulos componentes do SGBD.

* **portabilidade**

É importante que as aplicações desenvolvidas utilizando como base o Sistema de Gerenciamento de Banco de Dados em questão, possam ser portadas para outras plataformas computacionais com um mínimo de modificação de código. Para tanto, o SGBD deve ser disponível em versões para um número razoável de tipos diferentes de sistemas computacionais ou, alternativamente, deve ser disponível seu código fonte para ser compilado no sistema alvo.

* **padronização**

Deve existir um conjunto de regras bem determinadas e consistentes que possibilitem o acesso ao Banco de Dados de forma eficiente, segura e isenta de ambiguidades.

O ideal é que o sistema forneça uma linguagem estruturada de consulta (Structured Query Language - SQL).

* **flexibilidade**

O Sistema de Gerenciamento de Banco de Dados deve adequar-se à utilização em um número razoável de tipos distintos de aplicações e deve, ainda, oferecer diversas estratégias de acesso aos dados e várias possibilidades de configuração.

**2.2.2.2) Requisitos das Interfaces**

Um programa de aplicação sério deve possuir interfaces bem projetadas. Esta afirmação é especialmente verdadeira no caso de Sistemas de Auxílio a Projetos, pois não é raro que se julgue a qualidade de um sistema desse tipo

pela qualidade de suas interfaces.

Desta forma, é importante que se dedique um cuidado especial no projeto das interfaces do sistema pois, do ponto de vista do usuário, esta é sua parte mais visível. Adicionalmente, interfaces bem elaboradas facilitam a interação usuário-sistema, favorecendo a produtividade global.

Um Sistema de Apoio a Projetos possui pelo menos duas interfaces distintas: a interface Sistema-Usuário e a interface Sistema-Banco de Dados.

Veja figura a seguir:

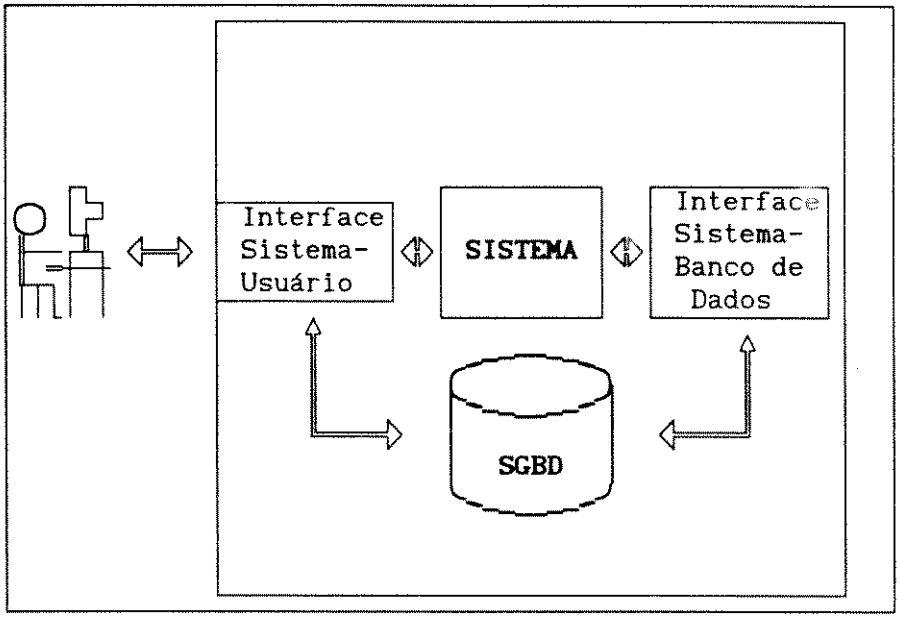



Figura 2.2 - As interfaces de um SAP

*** Interface Usuário**

É uma interface de acesso interativo que permite a um operador humano interagir com o sistema ou, alternativamente, com o SGBD diretamente.

Com a evolução do conceito de interface homem-máquina, tem-se dado, cada vez mais, importância à facilidade de interação entre o usuário e programa de aplicação. Em especial, os conceitos de interface baseada em

janelas e menus[1] dão uma nova dimensão à interação homem-máquina, facilitando a comunicação entre o sistema e o usuário, principalmente o usuário médio que, em geral, tem um conhecimento apenas superficial da arquitetura do sistema.

Sistemas de Software em geral, e Sistemas de Apoio a Projetos em particular, devem possuir uma interface com o usuário "amigável", no sentido de que ela deve ser de manejo simples e auto-explicativo, padronizada e bem organizada visualmente, possibilitar auxílio "on-line" e, principalmente, deve ocultar do usuário detalhes da implementação que não sejam essenciais para uma boa interação com o sistema.

*** Interface com o Banco de Dados**

É uma interface procedural entre programas (ou processos) e o Sistema de Gerenciamento de Banco de Dados.

O Sistema de Gerenciamento de Banco de Dados deve prover uma forma de interação com programas (ou processos), na forma de um conjunto de funções que, invocadas a partir do interior de um programa, possibilitem o acesso aos dados armazenados no Banco de Dados.

Possivelmente as funções que implementam essa interface estarão agrupadas em um módulo separado do SGBD, sendo que para cada linguagem de programação suportada haverá, provavelmente, um módulo distinto apropriado.

### 2.2.3) Outros Requisitos

*** Formulários**

É possível que uma partes consideráveis das informações que necessitem ser coletadas para o desenvolvimento de um projeto qualquer encontrem-se geograficamente dispersas.

---

[1] Veja no capítulo 4 mais sobre interfaces baseadas em janelas e menus

Como, em geral, seria inviável que cada indivíduo detentor de uma parte dessas informações interagisse diretamente com o sistema para registrá-las, é conveniente que haja a possibilidade de obtenção desses dados de forma não interativa, ou seja, através do preenchimento de formulários de consulta gerados a partir do próprio sistema.

Esses formulários devem refletir fielmente a estrutura da Base de Dados que representam, de modo a minimizar a discrepância entre os dados que se buscam e os que são obtidos.

Além disso, é interessante que cada formulário tenha um glossário contendo o seu propósito e o significado de cada campo a ser preenchido, de forma a orientar o correto preenchimento das informações requisitadas.

O capítulo seguinte discorre sobre Bancos de Dados, de uma forma geral, e apresenta o Sistema de Gerenciamento de Banco de Dados **BAPAS-DB**, que escolhemos para integrar a nossa metodologia.

# CAPÍTULO 3

## BANCOS DE DADOS (BAPAS-DB)

## 3.1) **Introdução** - Bancos de dados

Em Sistemas de Informação, o Banco de Dados[1] pode ser considerado o núcleo do sistema, pois ele é o depositário da totalidade das informações. Todas as demais ferramentas componentes do sistema são elaboradas levando em conta as características e a estrutura do Banco de Dados.

A seguir, temos algumas definições sucintas [B-*Chu 83*] de termos e conceitos fundamentais no escopo de Bancos de Dados:

### 3.1.1) **Item de dado ou Campo** (Data Item or Field)

Constitui uma unidade básica representativa de informação, identificável e possível de definir quanto ao tamanho e formato.

Um item de dado descreve um determinado atributo referente a uma entidade específica. Por exemplo:

| **Entidade** | **Atributos** |
|---|---|
| Aluno | - Nome<br>- R.A.<br>- Curso<br>- Data de Ingresso<br>- Endereço |

Figura 3.1 - Entidades e Atributos

### 3.1.2) **Registro** (Record)

Consiste de um conjunto de itens de dados (campos), reunidos de forma a caracterizar uma ocorrência de um conjunto de atributos de uma determinada

[1] consideramos que não há distinção entre **Banco de Dados** e **Base de Dados**

entidade.

A cada registro associa-se uma ou mais **chaves** identificadoras do registro, de forma a poder-se armazenar e recuperar as informações de um (ou um grupo de) registro(s).

A chave nada mais é do que um dos campos do registro. No caso do exemplo anterior, a chave poderia ser o R.A. do aluno (se considerássemos uma única chave por registro).

### 3.1.3) **Arquivo** (File)

Um arquivo é composto, normalmente, por uma coleção de registros de um mesmo tipo. Este conceito de arquivo tem origem no processamento de arquivos convencional.

Poderíamos, por exemplo, agrupar todos os registros de alunos do exemplo anterior em um único arquivo do Banco de Dados.

### 3.1.4) **Banco de Dados** ou **Base de Dados** (Data Base or Data Bank)

Uma Base de Dados é constituída de um conjunto de arquivos inter-relacionados e, possivelmente, informações sobre a estrutura dos dados, índices, dados de segurança e privacidade, etc.

Consideramos *Banco de Dados* e *Base de Dados* como sinônimos. Porém, há quem considere que um Banco de Dados é um conjunto de Bases de Dados.

### 3.1.5) **Sistema de Gerência de Banco de Dados - SGBD**
### (Data Base Management System - DBMS)

Um Sistema de Gerência de Banco de Dados corresponde a um conjunto de programas específicos para a administração de um Banco de Dados.

Desta forma, os programas de aplicação não mais acessam os dados contidos no Banco de Dados diretamente. Todas as chamadas passam a ser

feitas ao SGBD que se encarrega de mapeá-las para o Banco de Dados. Veja, na figura a seguir, uma analogia entre o processamento convencional de arquivos e o acesso de um programa a um Banco de Dados através de um SGBD.

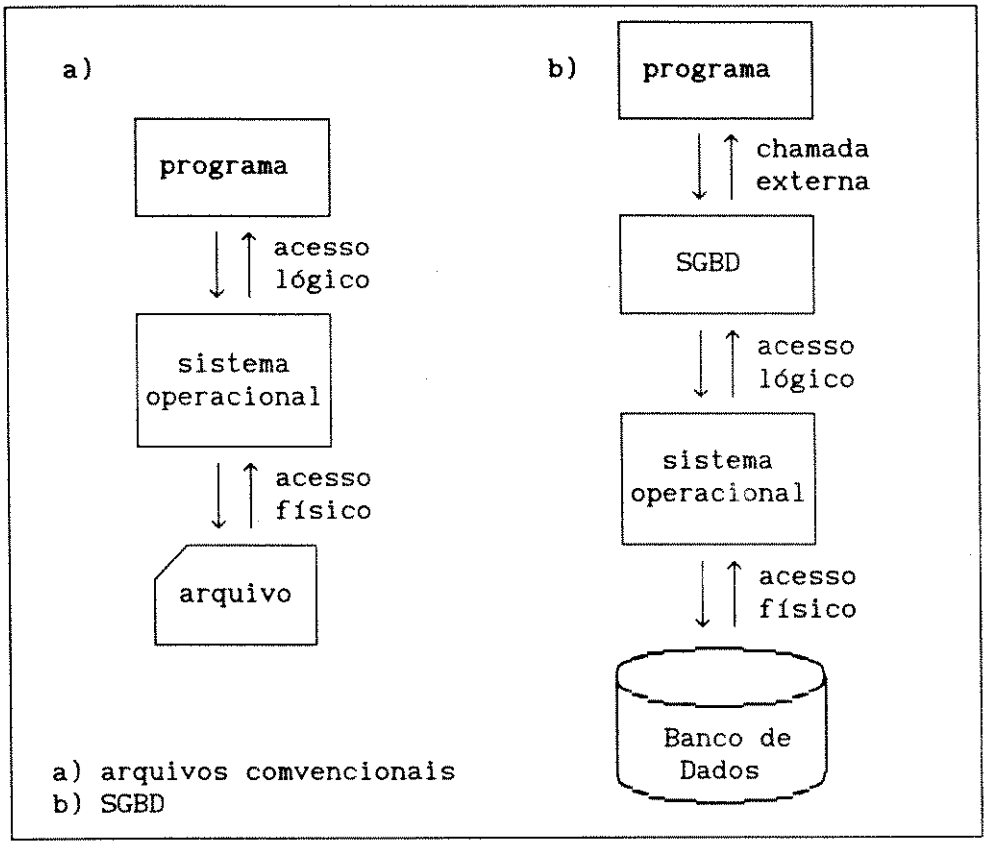



Figura 3.2 - Analogia arquivos convencionais X SGBD

Como exemplos de programas constituintes de um Sistema de Gerência de Banco de Dados, podemos citar [B-*Casanova 85*] a Linguagem de Descrição de Dados e a Linguagem de Consulta.

### 3.1.5.1) Linguagem de Descrição de Dados (Data Description Language - DDL)

Trata-se de uma linguagem específica para a criação e estruturação de

uma Base de Dados. Antes de podermos efetuar operações sobre uma Base de Dados, devemos tê-la criado, ou seja: devemos definir as estruturas dos registros que irão formar os arquivos que, por sua vez, irão compor a Base de Dados.

A linguagem que permite fazer essas operações é denominada: Linguagem de Descrição de Dados (ou, em inglês: Data Description Language).

### 3.1.5.2) Linguagem de Consulta (Query Language - QL)

Uma vez que uma Base de Dados tenha sido criada e dados nela inseridos, devemos ter maneira de recuperar as informações nela contidas. A Linguagem de Consulta é uma forma de permitir tal acesso.

Tal linguagem deve permitir acessar as informações tanto individualmente como em grupos (subsets).

### 3.1.6) Tipos de Modelos de Dados [B-*Chu 83*]

Cada tipo de modelo de dados utiliza elementos particulares básicos de representação das informações em diferentes níveis. Estes níveis encontram-se distribuídos desde a percepção do mundo real até a estrutura final dos dados de um determinado Banco de Dados.

Os modelos de dados mais conhecidos são:

### 3.1.6.1) Modelo de Rede

O Modelo de Rede utiliza o conceito de associação como unidade básica. Um modelo conceitual global pode ser elaborado a partir da definição de entidades básicas e independentes e, a partir daí, elaboram-se processos de "aninhamento" e associação entre as entidades, de tal forma que as ligações de interesse são pré-definidas, possibilitando, na fase de operação do Banco de Dados, a navegação entre os elementos do Banco de Dados.

### 3.1.6.2) Modelo Hierárquico

Este modelo pode ser considerado como um caso particular do Modelo de Rede. Utiliza como unidade básica o conceito de "aninhamento" entre as entidades básicas. Desta forma, as entidades ficam dispostas em uma árvore com diferentes níveis hierárquicos, onde a associação entre dois tipos de entidades em níveis consecutivos é única.

### 3.1.6.3) Modelo Relacional

O Modelo Relacional utiliza a relação matemática como unidade básica. Para tanto, utiliza a teoria dos conjuntos como a base formal para a descrição de modelos de dados. Operadores tradicionais[1] de conjuntos e operadores relacionais[2] podem efetuar operações sobre os dados.

A seguir, algumas relações utilizadas no Modelo Relacional:

- **Tupla**: é um conjunto adequadamente escolhido de valores de atributos
- **Relação**: é um conjunto de *tuplas* com os mesmos atributos
- **Tipo de Relação**: é a denominação de uma determinada *relação*, tomada como um conjunto específico de atributos.
- **Modelo de Banco de Dados**: é um conjunto de *tipos de relações*.
- **Banco de Dados Relacional**: é o conjunto de *relações*.

### 3.1.6.4) Modelo Entidade

Neste Modelo, utiliza-se o conceito de entidade como elemento básico. Outros conceitos como propriedades, associações e outros, são utilizados, também, na fase de percepção do mundo real.

Existem alguns "sub-modelos" derivados deste, tais como o *Modelo Entidade-Relacionamento* e o *Modelo Entidade-Conjunto*.

---

[1] união, intersecção, diferença, produto cartesiano
[2] projeção, junção, seleção

## 3.2) BAPAS-DB [*B-BAPAS 88*]

BAPAS-DB é um sistema de Banco de Dados desenvolvido pela firma alemã WERUM Gmbh com o objetivo de suportar a automação de sistemas com elevada demanda de resposta em tempo real, tolerância a falhas, flexibilidade e distribuição. Exemplos típicos de aplicações já realizadas com o BAPAS-DB são:

- Monitoramento de manufatura
- Sistemas de manutenção
- Sistemas de controle de qualidade
- Sistemas flexíveis de manufatura
- Sistemas de controle de processos
- Sistemas de configuração e controle em radiodifusão
- Sistemas militares de controle e supervisão
- Sistemas laboratoriais em hospitais
- Sistemas reprotécnicos
- Processamento de pedidos em redes de PC

Várias companhias e instituições de renome utilizam o BAPAS-DB como, por exemplo: IBM, Siemens, Daimler-Benz e Du Pont.

Atualmente, o BAPAS-DB está disponível para uma variada gama de equipamentos e sistemas operacionais, dentre os quais destacamos os seguintes:

| Computador | Sistema Operacional |
|---|---|
| família Apollo Domain DN 3000 | AEGIS (UNIX) |
| família HP 9000 | HP-UX |
| IBM PC/AT e PS/2 | DOS 3.x |
| IBM PS/2 | OS/2 |
| família PCS Cadmus 9000 | MUNIX |
| SUN workstations | SunOS (UNIX) |
| família VAX e MicroVAX | VMS |

### 3.2.1) Modelo

O modelo de Banco de Dados no qual o BAPAS-DB está baseado é semelhante ao modelo relacional, com três importantes diferenças, ditadas pelas necessidades específicas de processos técnicos, por exemplo: resposta em tempo-real. Estas diferenças são:

*** Imagem do processo = processo**

O modelo relacional é baseado no princípio do sub-conjunto (ou subset), que pode ser entendido através do seguinte exemplo: como resultado do comando "processe todos os registros (linhas) xi do arquivo (relação) X com a característica C", todos os registros xi são, em primeiro lugar, preparados e então sucessivamente processados , ou seja:

1º passo: selecionar o sub-conjunto { x1,x2,...,xn } de X

2º passo: processar x1
processar x2
...
processar xn

Durante o processamento, todos os registros xi do sub-conjunto ficam inacessíveis, ou seja, seus valores não podem ser mudados em função da ocorrência de eventos nos processos interessados. Assim:

**relacional ⇒ imagem do processo ≠ processo**

Em contraste a isto, o BAPAS-DB prepara sucessivamente os registros e imediatamente os processa, ou seja:

1º passo: selecionar x1

2º passo: processar x1

3º passo: selecionar x2

4º passo: processar x2

...

Desta maneira mudanças no processo são refletidas no processamento. Assim:

**BAPAS-DB ⇒ imagem do processo = processo**

*** Tempos de acesso garantidos**

Sistemas de Banco de Dados relacionais permitem mudanças no esquema conceitual, tais como o estabelecimento de novos atributos, visões e agregações durante a operação. As respectivas relações (arquivos) ficam inacessíveis a outros usuários, ou então os acessos são muito mais demorados durante esse tipo de operação. Certos tempos de acesso não podem ser garantidos.

O BAPAS-DB permite tais mudanças e extensões de esquema somente durante a operação suspensa. Isto não representa uma restrição séria em condições reais de operação industrial pois, em geral, mudanças e extensões correspondentes de programas são efetuadas durante tais operações, de forma que a operação deve ser suspensa temporariamente para efetuar tais alterações.

Aliado a isto, certos tempos de acesso podem ser garantidos dentro de

determinados parâmetros, descritos mais adiante.

*** Arquivos estruturados**

Suponha uma estrutura do seguinte tipo:

```
ARTIGO: { CÓDIGO          do tipo CARACTERE
        { CARACTERÍSTICAS do tipo DADOS
        { AQUISIÇÃO       do tipo DATA

DADOS:  { PREÇO           do tipo INTEIRO
        { QUANTIDADE      do tipo INTEIRO

DATA:   { DIA             do tipo INTEIRO
        { MÊS             do tipo INTEIRO
        { ANO             do tipo INTEIRO
```

O modelo relacional conhece apenas estruturas planas, o que significa que a tabela de ARTIGO tem a seguinte estrutura:

| CÓDIGO | PREÇO | QUANTIDADE | DIA | MÊS | ANO |
|---|---|---|---|---|---|
| | | | | | |
| | | | | | |

Figura 3.3 - Estrutura de "artigo" (relacional)

Em contraste, esta estrutura pode ser mapeada de imediato em linguagens modernas de alto nível, tais como: ADA, C, PASCAL e PEARL. Assim, a estrutura de ARTIGO pode ser espelhada nas linguagens acima mencionadas de uma maneira orientada ao problema (*problem-oriented*), o que simplifica o acesso da estrutura por programas. Desta maneira, a tabela de artigo tem a seguinte estrutura para o BAPAS-DB:

| CÓDIGO | CARACTERÍSTICAS | | AQUISIÇÃO | | |
|---|---|---|---|---|---|
| | PREÇO | QUANTIDADE | DIA | MÊS | ANO |
| | | | | | |
| | | | | | |

Figura 3.4 - Estrutura de "artigo" (estruturada)

### 3.2.2) Componentes do BAPAS-DB

A figura da página seguinte sumariza os componentes do BAPAS-DB.

#### 3.2.2.1) Programa de instalação INSTAL

Com este programa o usuário (administrador do sistema) define interativamente os seguintes parâmetros:

- o arquivamento da área da Base de Dados no disco;
- o tamanho máximo do dicionário de dados;
- o número máximo de arquivos da Base de Dados;
- a quantidade e tamanho dos buffers.

#### 3.2.2.2) Interpretador DDL

A linguagem de descrição de dados orientada a problema

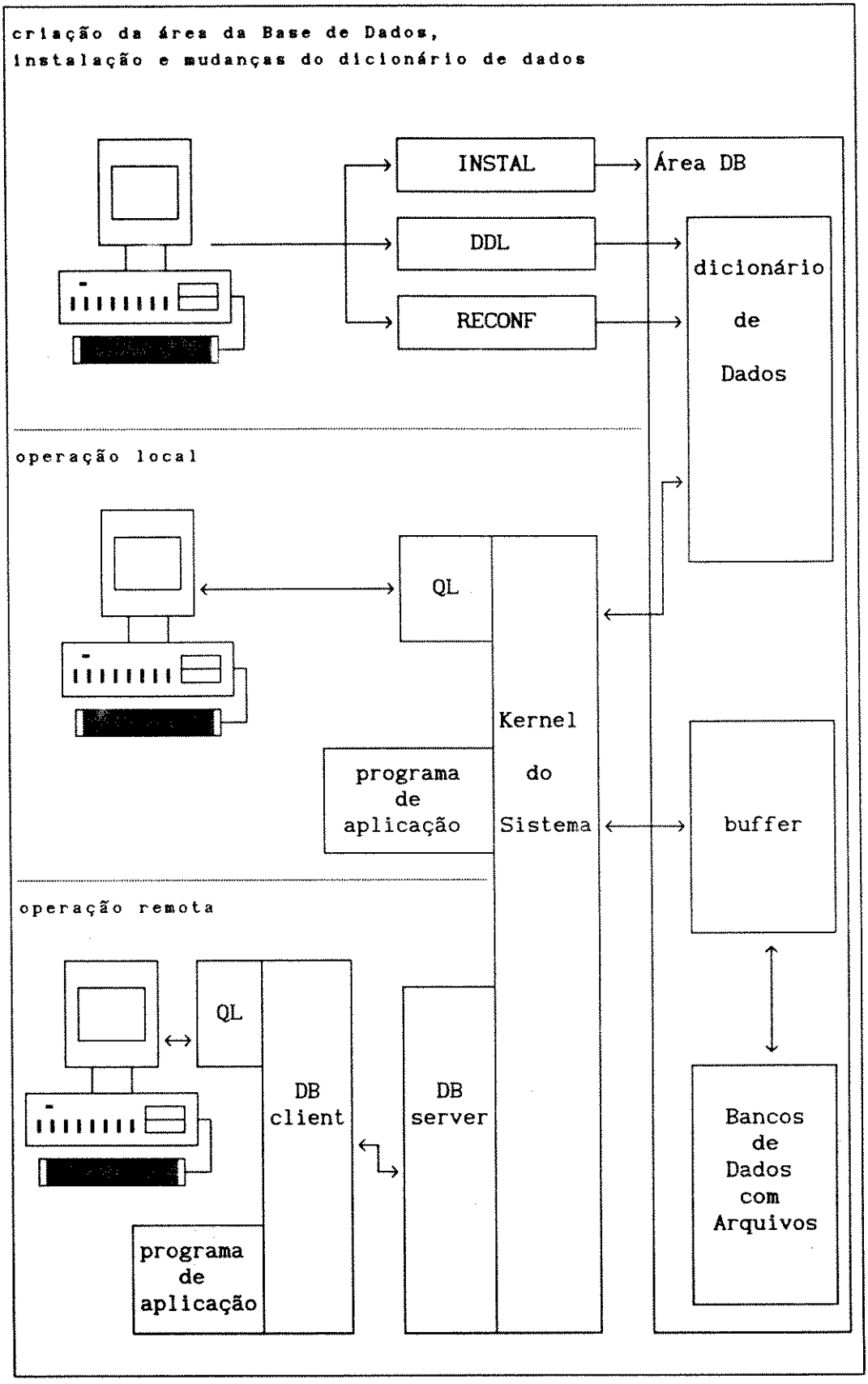



Figura 3.5 - Componentes do BAPAS-DB

(*problem-oriented*) DDL permite a definição interativa ou, alternativamente, através de um arquivo de comandos em lote, das seguintes informações (parâmetros) que vão dar origem ao dicionário de dados:

- bases de dados;
- arquivos e visões (views);
- estruturas de registro;
- privilégios de acesso.

- além disso, aloca para os arquivos:
  - estratégias de acesso
  - sistemas de buffer

### 3.2.2.3) Programa de reconfiguração RECONF

O programa de reconfiguração RECONF é usado para modificar ou estender o dicionário de dados produzido pela DDL. Ele suporta as seguintes funções, com preservação dos dados:

- modificação da estrutura de registros, por exemplo para estender um atributo;
- modificação de volume, por exemplo para aumentar um arquivo;
- modificação dos sistemas de buffer, por exemplo para aumentar a quantidade de buffers;

- troca da estratégia de acesso, por exemplo para utilizar uma estratégia de busca em árvore binária (BTREE) ao invés da estratégia FIFO;

- armazenamento externo de arquivos, por exemplo para backup.

DDL e RECONF podem ser utilizados somente durante operação suspendida a fim de garantir tempos de acesso pequenos.

### 3.2.2.4) System Kernel

O Kernel do sistema administra as bases de dados estabelecidas, executa as operações requisitadas pelos programas de aplicação ou pela linguagem de consulta (Query Language - QL). Executa, ainda, as medidas de segurança estabelecidas (p.ex.: checkpoints).

No caso mais simples (e mais eficiente), o programa de aplicação e o kernel do sistema, com capacidade de reentrância, são colocados no mesmo espaço de memória do computador; o programa de aplicação, que geralmente consiste de diversas tarefas (tasks), usa as funções do kernel através de chamadas de procedimentos orientadas ao problema (*problem-oriented*) da assim chamada interface **DBV** do BAPAS-DB.

### 3.2.2.5) DB-client e DB-server

O kernel do sistema e o programa de aplicação podem, também, ser alocados em diferentes espaços de memória; assim, o kernel do sistema é instalado como um servidor de base de dados independente, o qual o programa de aplicação acessa através do esquema de troca de mensagens, disponível no computador. Neste caso, o componente DB-server é associado ao kernel do sistema, enquanto que o programa de aplicação é associado com o componente DB-client. A comunicação entre DB-server e DB-client é feita de maneira transparente para o usuário, podendo estar o kernel do sistema e o programa de aplicação situados em diferentes computadores que se comuniquem através

de uma rede, por exemplo uma Rede Local.

#### 3.2.2.6) QL a linguagem de consulta

O BAPAS-DB oferece uma linguagem de consulta (query language), denominada QL, para consultas interativas à base de dados. A linguagem QL apresenta-se como um interpretador orientado a problema, ou então na forma de máscaras. Permite as seguintes operações sobre registros de arquivos e visões da base de dados criadas através da linguagem DDL:

- inserir;
- selecionar;
- modificar;
- apagar.

Com a ajuda dos componentes DB-server e DB-client, a linguagem QL pode também ser utilizada em sistemas distribuídos de computadores.

### 3.2.3) Propridades de acesso em Tempo Real

As características essenciais de acesso em tempo-real do BAPAS-DB são:

#### 3.2.3.1) Acessos controlados por prioridades

Programas com necessidade de resposta em tempo real, como por exemplo tarefas (tasks) C ou PEARL sob OS/2, podem acessar concorrentemente bases de dados do BAPAS-DB; tarefas com prioridade maior têm precedência sobre tarefas de menor prioridade. Registros que necessitem ser modificados são blo-

queados (locked) automaticamente. Em caso de colisão de transações, a transação de menor prioridade cede a vez, ou seja, ela retorna ao seu início; as mudanças que ela tenha efetuado sobre os dados são desfeitas. Esta característica também é aplicável a acessos de clientes a um servidor em uma rede.

#### 3.2.3.2) Operação contínua (sem reorganizações)

A maneira pela qual os métodos de acesso do BAPAS-DB foram construídos faz com que seja possível um esquema de operação contínua, isto é, sem a necessidade de um operador reorganizar os dados periodicamente.

#### 3.2.3.3) Tempos de acesso garantidos

As duas características acima, tomadas em conjunto, garantem que os tempos de acesso de tarefas (tasks) de prioridade máxima não excedam certos valores (mensuráveis e dependentes de projeto) durante a operação.

#### 3.2.3.4) Aplicações especiais podem ter buffers especiais

Adicionalmente ao sistema de buffers padrão, o BAPAS-DB pode suportar sistemas de buffers projetados especialmente para um arquivo, ou grupo de arquivos, que requeiram buffers fora do padrão para cumprir requisitos especiais de tamanho de registro e tempos de acesso.

#### 3.2.3.5) Arquivos na memória principal

Pode-se criar arquivos na memória principal para cumprir requisitos extremos de tempo de acesso. No caso padrão, o método de "checkpoint" aplica-se a esses arquivos bem como a arquivos de "background" para re-

inicialização automática a partir do disco, após uma falha do computador. Veja no item 3.6.4 mais informações sobre esse tipo de arquivo.

#### 3.2.3.6) Aplicação em sistemas sem disco (diskless)

Os arquivos em memória principal podem não ter memória de "background" em disco; assim, o usuário pode ter um sistema de banco de dados em um computador sem disco, em uma aplicação que não requeira a recuperação do estado anterior do sistema, na eventualidade de uma falha.

### 3.2.4) Características de Sistema Aberto e Flexível do BAPAS-DB

#### 3.2.4.1) Várias estratégias de acesso

Cada projeto de automação tem requisitos peculiares de acesso à base de dados. Por essa razão, o BAPAS-DB tem diversos métodos (estratégias) de acesso como padrão.

O acesso aos arquivos da Base de Dados pode ser efetuado obedecendo-se a diversos tipos de estratégias e pode-se, até mesmo, acessar cada arquivo segundo uma estratégia diferente.

Programas de aplicação podem acessar simultaneamente vários arquivos da base de dados com estratégias de acesso diferentes, sendo possível, ainda, com uma única chave acessar vários registros com estratégias diferentes.

Pode-se trocar a estratégia de acesso de um arquivo sem ter que modificar o programa de aplicação nem os diálogos QL.

A seguir temos um panorama das estratégias de acesso suportadas pelo BAPAS-DB, com uma breve descrição de cada uma.

* **HASH** - Estratégia baseada em uma tabela tipo "hash" e em chaves unívocas.

- Inserção

Quando da inserção de registros em arquivos utilizando a estratégia HASH, a "chave" é armazenada em uma tabela do tipo "hash", para possibilitar o acesso a esse registro, via a chave, eficientemente. A sequência dos registros em um arquivo, que pode ser relevante no caso de busca sequencial sem uso de chave, não obedece a qualquer ordem. No caso de atualização de registro, uma nova chave é armazenada.

- Busca

Na busca com a indicação de uma "chave", a posição do registro no arquivo é obtida da tabela "hash" via a chave; o acesso é feito, então, diretamente.

No caso de busca sem chave, o acesso é feito diretamente nos registros, de acordo com a sua sequência incidental.

- Remoção

Para a remoção de um registro, a entrada correspondente a ele na tabela "hash" é removida. O espaço correspondente ao registro no arquivo pode então ser ocupado por um novo registro.

* **FIFO & LIFO -** São estratégias sequenciais. No caso de FIFO (First In - First Out), a sequência vai do elemento inserido primeiro ao inserido por último, cronologicamente, ou seja: o primeiro a entrar é o primeiro a sair. Já, para LIFO (Last In - First Out), a direção da sequência é a inversa; assim o último a entrar é o primeiro a sair.

- Inserção

Para a inserção de registros, é obedecida a ordem cronológica correspondente. A sequência dos registros no arquivo é constante, ou seja: atualizações modificam o registro, mas não sua posição relativa.

- Busca

A busca de registros é feita de acordo com a ordem direta da entrada dos registros, no caso de FIFO, e na ordem inversa no caso de LIFO.

- Remoção

Quando um registro é removido, seu predecessor é ligado ao seu sucessor, de forma a que a ordem dos registros existentes seja preservada.

* **INDSQL** - Estratégia de acesso Sequencial-Indexada, onde o acesso a registros pode ser feito via uma ou mais "chaves" (unívocas ou não) ou, também, sequencialmente. Registros com a mesma chave podem ser organizados em um buffer circular. Possibilita a manipulação de registros de tamanho variável.

- Inserção

A inserção de registros em arquivos acessados por este tipo de estratégia obedece a duas heurísticas: a primeira é uma tabela "hash" para as "chaves", onde a disposição das chaves não obedece a nenhuma ordem; a segunda se refere a registros com a mesma chave, os quais são organizados segundo uma estratégia FIFO. Atualizações sem alteração da chave não mudam a ordem dos registros. Se houver mudança de chave, a atualização corresponde a uma remoção seguida de uma inserção.

- Busca

A busca sem chave obedece à ordem incidental das chaves da tabela hash e, para registros com a mesma chave, à estratégia FIFO. Se for indicada uma chave, ela é usada para localizar o registro na tabela hash, e os registros relacionados a essa chave são processados numa ordem FIFO.

- Remoção

Um registro pertencente a um grupo de registros com a mesma chave é removido de acordo com uma estratégia FIFO. Se não houver mais registros com essa mesma chave, a entrada correspondente da tabela hash também é removida.

* **B-TREE** - Estratégia tipo b-tree para acesso direto através de chaves (unívocas ou não), bem como acesso sequencial por ordem de valor de chave.

- Inserção

De acordo com a chave, um novo registro é inserido diretamente no arquivo via uma b-tree. Se já existir um registro com essa chave, uma mensagem de erro é gerada, caso contrário, o registro é adicionado ao(s) existente(s).

- Busca

A busca de um registro, via uma chave, acessa (via a b-tree) diretamente o registro com essa mesma chave inserido por último. Os demais podem ser acessados através da operação DBVFALL (descrita adiante). Para a busca sem chave, o acesso é feito de acordo com a ordem crescente das chaves.

- Remoção

Remoções são permitidas somente com a indicação de uma chave.

* **RNR** - Estratégia que permite acesso direto por meio dos números relativos dos registros, o que elimina a sobrecarga sobre o gerenciamento do arquivo. RNR significa Record Number ou, em português: número de registro.

* **MS-HASH** - Extensão da estratégia HASH que permite a utilização de mais de uma chave de acesso, até um máximo de 4 chaves.

* **MS-HASH-RNR** - Estratégia resultante da combinação das estratégias MS-HASH e RNR.

### 3.2.4.2) Sistema padrão de buffers flexível

O BAPAS-DB organiza seus arquivos e buffers em páginas de igual tamanho. Os comprimentos de página podem ser determinados pelo usuário (administrador do sistema) quando da instalação do sistema. O default é o comprimento da unidade de tranferência de dados do computador.

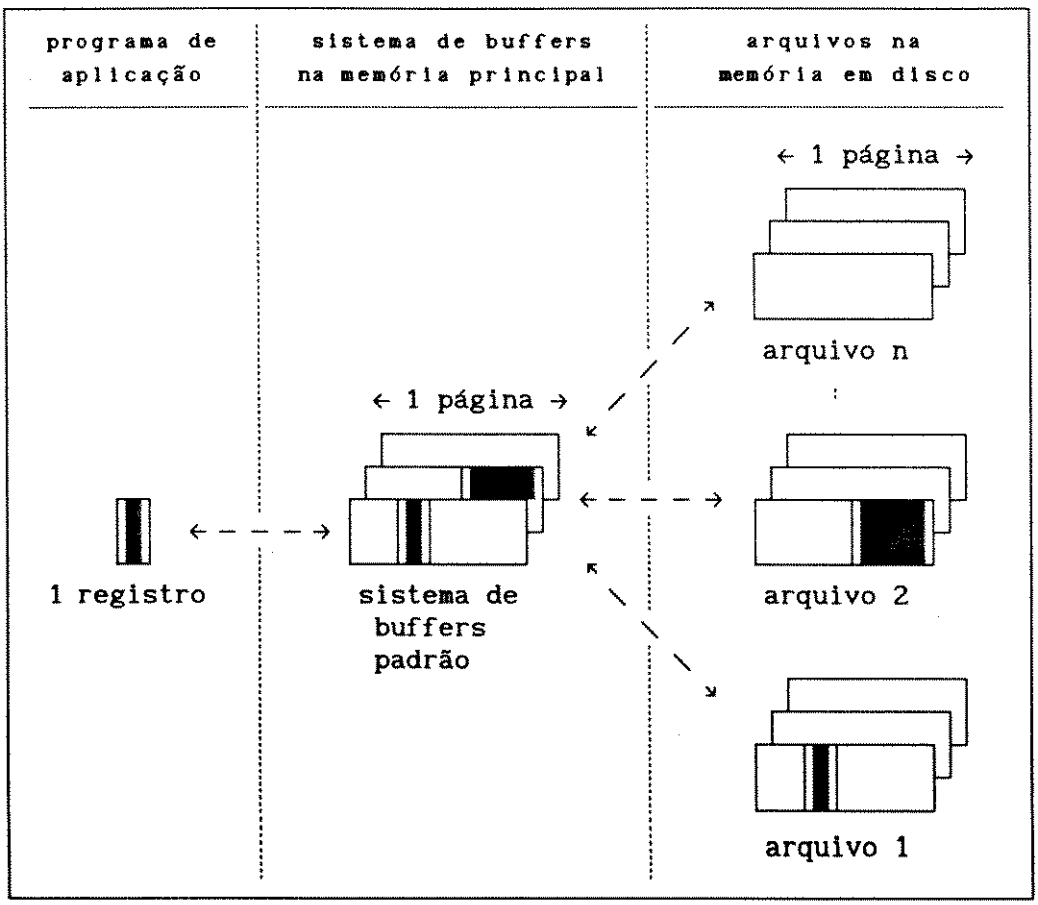



Figura 3.6 - Sistema padrão de buffers flexível

Durante a instalação, o administrador do sistema também determina quantos buffers (páginas) o sistema de buffers terá.

Em adição à estratégia de acesso escolhida, tamanho e quantidade de buffers são os fatores principais na determinação da rapidez das operações de acesso à base de dados. Ao receber o pedido de acessar um registro, o BAPAS-DB procura primeiro na página que contém esse registro no sistema de buffers; se não obtiver sucesso, passa a procurar o registro na memória de massa (ou seja, no disco), e a página correspondente é transferida para o sistema de buffers. A página processada não é imediatamente re-escrita após ser liberada mas permanece, isto sim, na memória principal até ser deslocada por uma outra página. Isto é conseguido através de um mecanismo de temporização tipo LRU (last recently used). Veja figura da página anterior.

### 3.2.4.3) Sistemas de buffers específicos para arquivos especiais

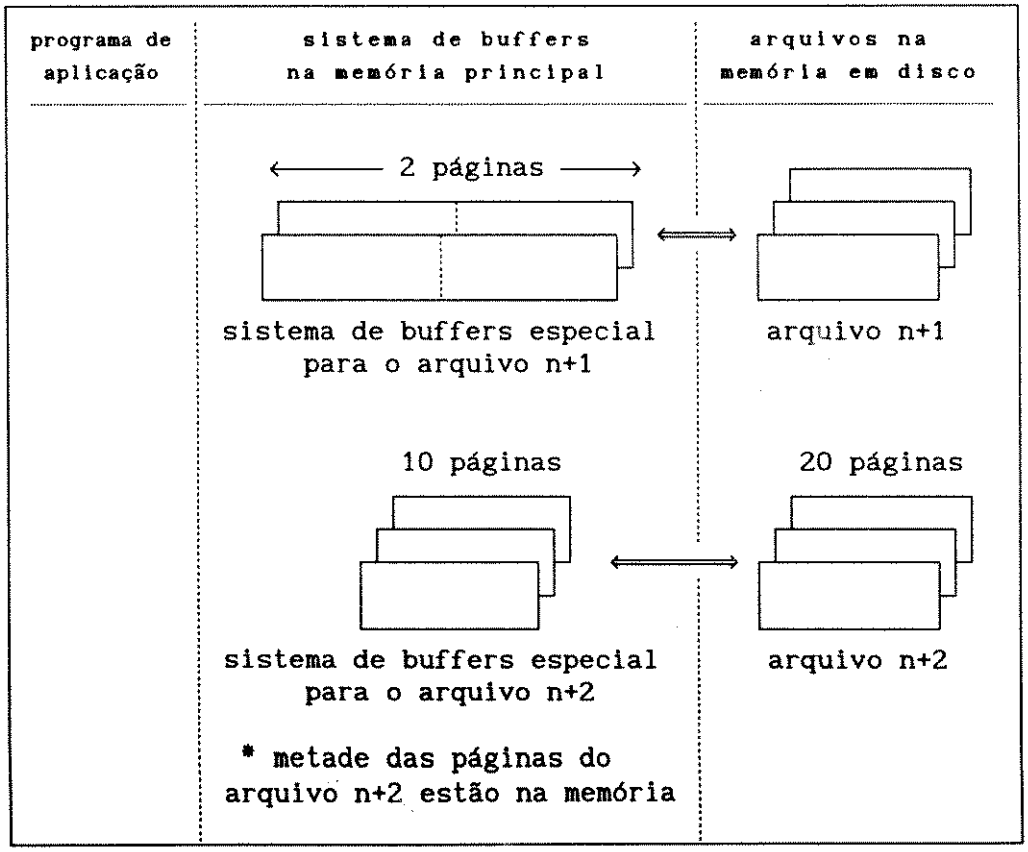



Figura 3.7 - Sistema de buffers específico para arquivos especiais

Adicionalmente ao sistema padrão de buffers, o usuário pode estabelecer outros sistemas de buffers estruturados de maneira diversa para satisfazer, de maneira otimizada, requisitos especiais de aplicações. Por exemplo: para processar um arquivo com tamanho de registro exótico assim como 64 KB, pode-se alocar um sistema de buffers individual de tamanho apropriado, ou seja 64 KB. Veja figura da página anterior.

#### 3.2.4.4) Arquivos na memória principal

Requisitos extremos de elevada velocidade de acesso podem ser alcançados através do estabelecimento dos assim chamados "arquivos de memória principal", que residem somente na memória principal e não na memória de massa (disco). O acesso a tais arquivos requer o menor tempo possível, através da estratégia de acesso especial chamada *HSP*; além disso, pode-se estabelecer "checkpoints" para tais arquivos, assim como nos arquivos padrões, residentes na memória de massa (disco).

Em casos extremos, o BAPAS-DB pode ser usado como um sistema puramente de memória principal, isto é, sem "background" na memória de massa (disco).

### 3.2.5) Consistência, segurança e proteção de dados no BAPAS-DB

#### 3.2.5.1) Estabelecimento de "checkpoints" com "before images"

O BAPAS-DB protege a consistência de seus dados com "checkpoints" e "before images"; isto significa que, após um certo número de mudanças nas páginas, todo o conteúdo dos buffers é escrito na memória de massa (background) de maneira que os dados nas duas memórias sejam consistentes. O instante de tempo ti em que é feita esta operação é chamado "checkpoint".

Se um registro é modificado, inserido ou apagado, o BAPAS-DB escreve antes o registro no assim chamado arquivo "before image" e somente após isto efetua realmente a operação sobre a base de dados. Veja figura a seguir.

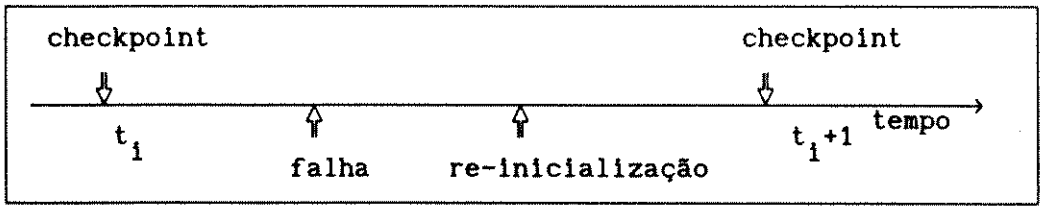



Figura 3.8 - Checkpoints

### 3.2.5.2) Recuperação automática

Se ocorrer uma falha no computador antes que ele atinja o próximo "checkpoint" (ti+1 na figura anterior), o BAPAS-DB automaticamente reinicializa o sistema utilizando os dados do último "checkpoint" (no caso, ti) para atualizar a base de dados. Desta maneira, assegura-se uma consistência dos dados. Operações de atualização, inserção e apagamento de registros, efetuadas após o último "checkpoint" anterior a uma falha, são perdidas.

### 3.2.5.3) "Log book" (after image)

A fim de evitar tais perdas de dados, o BAPAS-DB oferece a possibilidade de manter um *arquivo log* (log book) de arquivos importantes. Assim, o BAPAS-DB efetua as atualizações, inserções e apagamentos como anteriormente descrito, porém mantém um registro dessas operações em um *arquivo log* (também chamado "after image") situado na memória de massa (disco). Após uma reinicialização provocada por uma falha, é produzido, inicialmente, um estado consistente com base no último "checkpoint" e no arquivo de "before image"; após isto, BAPAS-DB efetua todas as modificações sobre os dados com o auxílio do arquivo "after image".

Desta maneira, uma falha no computador não causa perda de dados, pois obtém-se um estado totalmente consistente com o estado imediatamente anterior à falha.

### 3.2.5.4) Transações (atômicas)

O BAPAS-DB oferece a possibilidade de utilização de transações para efetuar diversas operações sobre a base de dados; ou seja: mudanças (atualizações, inserções ou apagamentos de registros) efetuadas durante a transação, não são realmente efetuadas sobre a base de dados até que a transação seja totalmente completada. Até esse momento as alterações não são visíveis a outros processos (tasks) e transações.

A cada transação está associada uma prioridade. Assim se uma transação acessa um registro reservado a outra transação (caso de conflito), a transação de maior prioridade prevalece. A transação de menor prioridade libera os registros por ela reservados e retorna ao seu ponto inicial ou, alternativamente, é morta (killed), podendo recomeçar assim que os registros necessários sejam liberados pela transação de maior prioridade.

Transações também podem ser definidas como "não interrompíveis" (uninterruptable). Uma transação não interrompível não é iniciada enquanto houver outra transação não interrompível acessando arquivos por ela necessitados. em outras palavras, em caso de conflito ela interrompe transações interrompíveis.

Um "log book" é mantido para os arquivos da base de dados utilizados em uma transação. Em caso de re-inicialização a partir do último "checkpoint", todas as transações iniciadas e completadas com sucesso entre o "checkpoint" e o ponto de falha são automaticamente repetidas pelo BAPAS-DB. Transações incompletas podem (dependendo da aplicação) ser reiniciadas no programa de aplicação. Veja a figura da página seguinte.

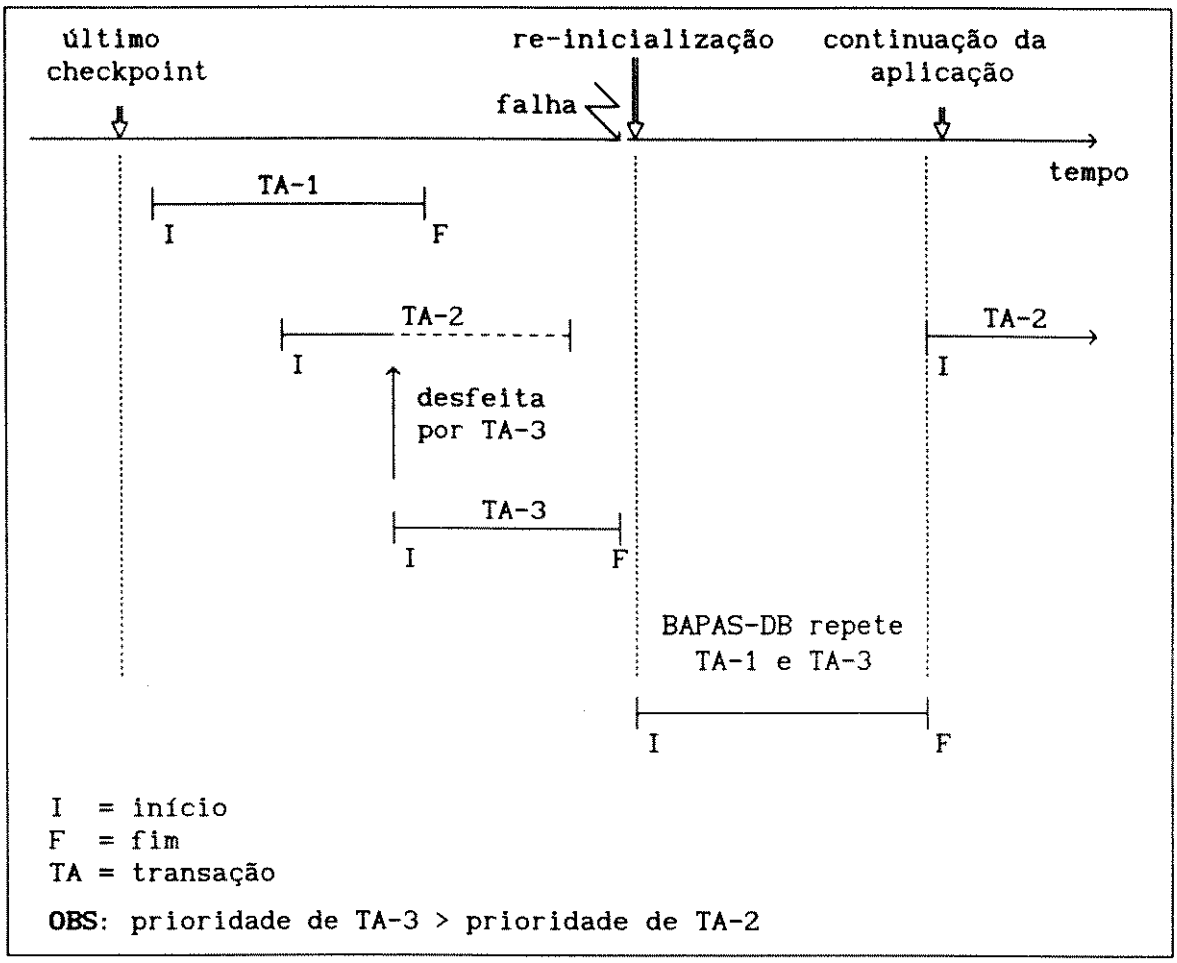



Figura 3.9 - Repetição automática de transações

### 3.2.5.5) Discos duplicados (mirror disks)

As medidas descritas até este ponto evitam uma perda de dados em caso de falha da CPU, porém, não em caso de falha da memória de massa (disco). A fim de precaver-se contra esse tipo de falha, é recomendável manter os dados em dois dispositivos fisicamente separados - um "disco mestre" e um "disco escravo".

Assim, o BAPAS-DB atualiza a base de dados duas vezes - no disco mestre (master), que é o que o BAPAS-DB usa para leitura, bem como em um segundo disco, dito "escravo" (slave). Se um dos discos falha, o outro disco torna-se mestre único. Após o disco defeituoso ter sido reparado, este é carrega-

do com uma cópia do disco mestre, durante operação suspensa. A seguir a operação com os discos mestre e escravo pode ser retomada.

Opcionalmente o recarregamento do disco escravo pode ser feito durante a operação.

Veja figura a seguir.

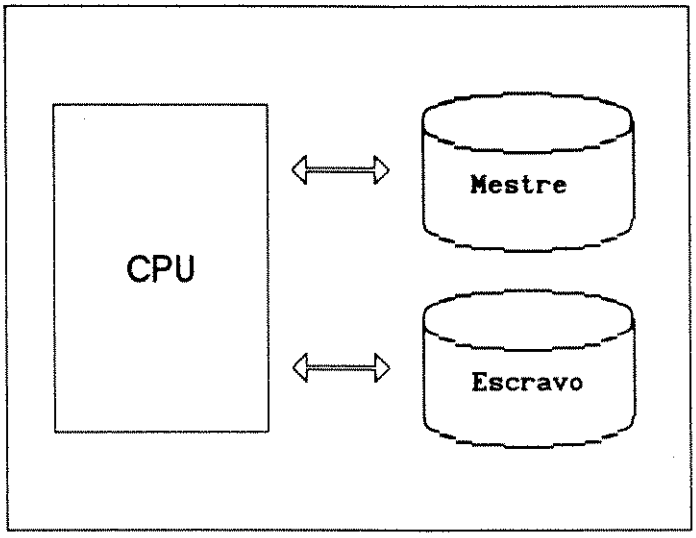



Figura 3.10 - Discos duplicados

### 3.2.5.6) Sistemas duplicados

Em adição a discos duplicados, o BAPAS-DB suporta a operação de uma configuração de dois computadores.

Dois sistemas computacionais, com um ou dois discos cada, são diretamente acoplados ou conectados por uma rede local (LAN); o BAPAS-DB é instalado em ambos. Um sistema é o sistema mestre, e o outro, o sistema escravo. Durante a operação normal o BAPAS-DB envia ordens de atualização do sistema mestre para o sistema escravo, onde são processadas pelo segundo BAPAS-DB. Se um dos sistemas falha, o outro torna-se o mestre único.

Após o reparo do sistema falho, ele pode ser carregado com uma cópia do sistema mestre durante operação suspensa. Feito isto, pode-se retomar a operação normal com mestre e escravo.

Opcionalmente, o recarregamento do sistema escravo pode ter lugar

durante a operação, ou seja, durante essa operação o BAPAS-DB do sistema mestre trabalha com tempos de resposta ligeiramente maiores.

Veja figura a seguir.

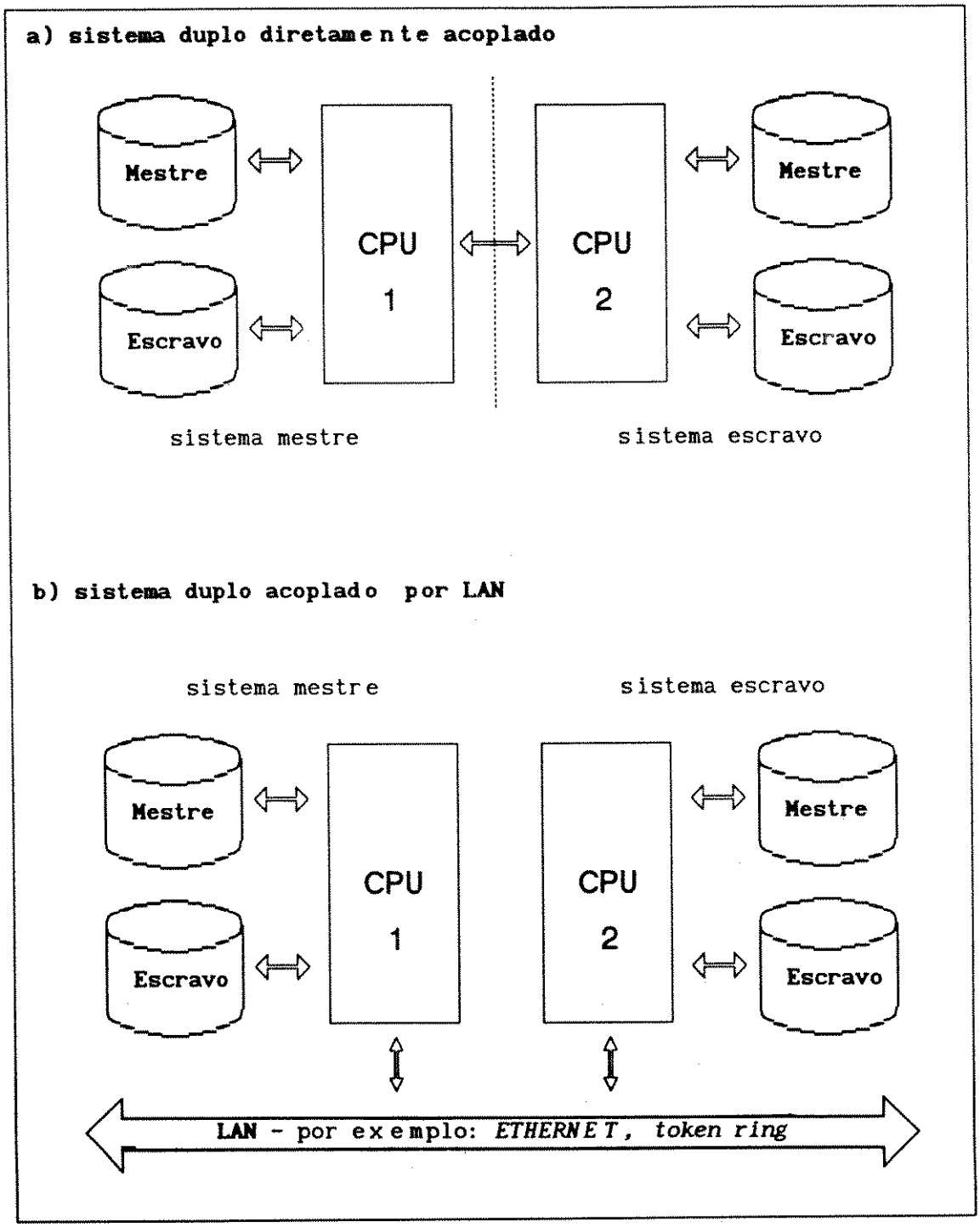



Figura 3.11 - Sistemas duplicados

### 3.2.5.7) Segurança de dados

Para proteção dos dados existem mecanismos de senha (password) a nível de base de dados e de arquivo, bem como verificação de tipo em dados (data type checking).

## 3.2.6) A interface de programas do BAPAS-DB

Uma das maneiras de acessar as informações armazenadas por uma Base de Dados BAPAS-DB é através da Linguagem de Consulta QL (Query Language). Esta é uma maneira conveniente em grande parte dos casos.

Pode ser necessário, porém, que programas acessem a Base de Dados em tempo-real. Para esse fim, o BAPAS-DB é dotado de uma interface de programas.

Uma das características essenciais de uma Base de Dados BAPAS-DB é que seus arquivos[1] podem ser acessados por processos concorrentes em tempo-real (tasks ou, em português, tarefas). Para isto, as seguintes funções estão disponíveis para esses programas:

- Especificação de arquivos da Base de Dados
- Seleção de registros[1] com certos atributos em um arquivo
- Alteração dos valores de registros
- Inserção de registros em arquivos
- Remoção de registros em arquivos

Essas funções podem ser utilizadas de duas maneiras distintas:

[1] As definições de Banco de Dados, arquivos e registros estão na parte inicial deste capítulo.

a) Através de uma interface procedural confortável e segura, denominada **interface DBV**, a qual pode ser utilizada, até mesmo, em sistemas distribuídos;

b) Através de uma interface de "baixo nível", designada (procedimento) **QDBS1**, no caso de ser necessária maior flexibilidade e/ou eficiência;

A interface **DBV** funciona como uma interface entre o programa de aplicação e o procedimento **QDBS1**, que faz parte do "Kernel" do BAPAS-DB. Em outras palavras, ela fornece os seguintes recursos:

- Administra as estruturas de dados, o Bloco de Controle de Arquivo e o cursor, pertencentes ao procedimento QDBS1, "ocultando" estes elementos do programa de aplicação;

- Oferece procedimentos confortáveis para as ações sobre a base de dados mais utilizadas, tanto para registros simples, como para subconjuntos de um arquivo;

- Gerencia acessos concorrentes, no âmbito de sistemas distribuídos, com base nos níveis padrão de comunicação.

### 3.2.6.1) A interface de "baixo nível" QDBS1

O procedimento **QDBS1** utiliza três parâmetros:

- Um File Control Block (bloco de controle de arquivo) para identificar o arquivo da base de dados selecionado;

- Um cursor que indica a posição atual no arquivo da base de dados, para um dado processo;

- Um parâmetro de ação, que indica a ação a ser efetuada.

O procedimento QDBS1 favorece a eficiência e a flexibilidade das operações de acesso aos registros de arquivos de uma Base de Dados. O preço a ser pago por tais características é uma programação mais complexa, onde tem-se que levar em consideração muitos detalhes relacionados com o acesso. Em outras palavras: o usuário tem um controle muito maior das ações sobre os arquivos e registros da Base de Dados; fica, porém, ao seu encargo, a responsabilidade de gerenciar os acessos a esses elementos.

### 3.2.6.2) A interface DBV (programa / BAPAS-DB)

Ao chamar procedimentos DBV, deve-se fornecer dois parâmetros (estruturas de dados): uma variável de arquivo (file variable), e uma condição.

A variável de arquivo possui componentes para a identificação do arquivo e do processo (tarefa ou "task"), para a análise de erros e para o tipo de acesso.

Adicionalmente, contém uma estrutura de dados variável, correspondente ao tipo de registro do arquivo.

O identificador de processo deve ser único em todo o sistema de aplicação, levando em consideração todas as máquinas participantes, o que se consegue através da numeração de todas as aplicações utilizando a interface DBV.

Subsequentemente a um acesso mal sucedido, o componente de análise de erros estará preenchido com informações detalhadas a respeito do motivo que ocasionou o erro.

O componente "tipo de acesso" determina se o registro selecionado deve ser acessado de maneira exclusiva (ou seja: bloqueado) ou não.

A variável de condição (condition variable) deve conter informações sobre se o acesso vai ser sequencial ou por "chave" e, eventualmente, a(s) chave(s). Veja figura a seguir:

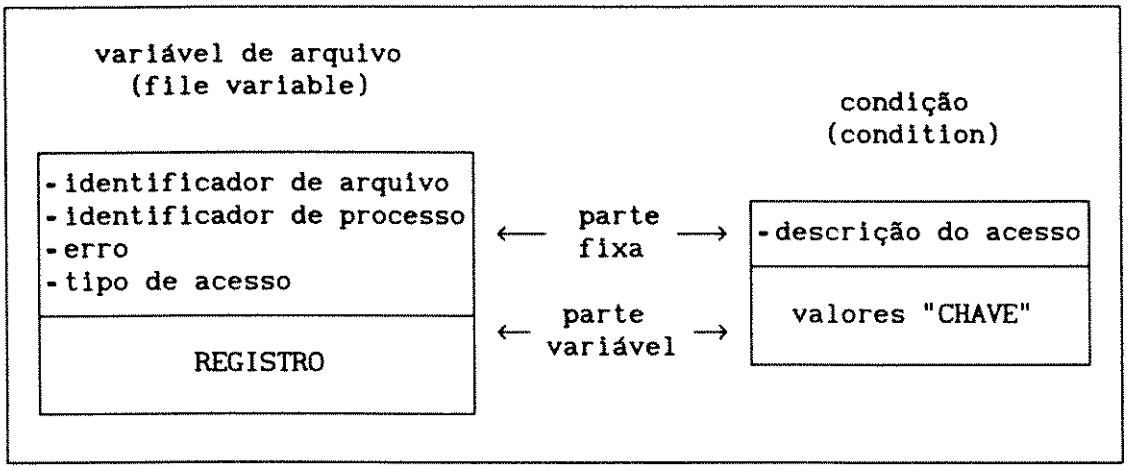



Figura 3.12 - Parâmetros para a chamada de procedimentos da interface DBV

Assim, a relação entre os programas (ou processos) de aplicação, a interface DBV e as funções do "Kernel" do BAPAS-DB pode ser explicada pela figura a seguir:

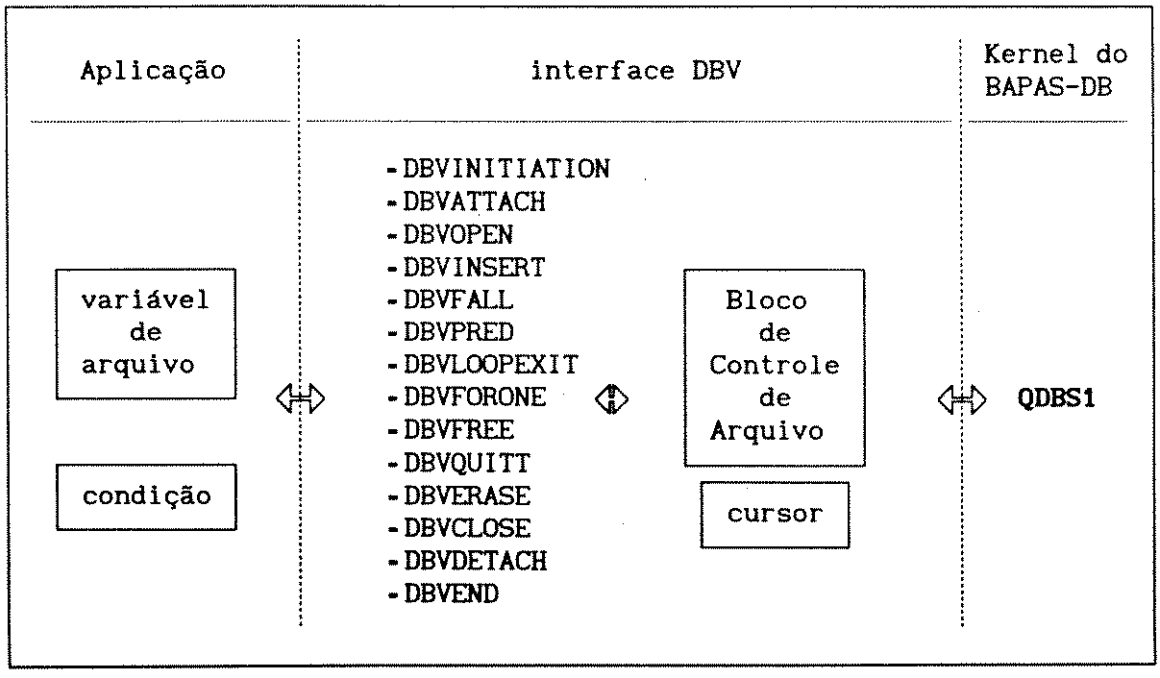



Figura 3.13 - Relação entre programa, interface DBV e o "Kernel" do BAPAS-DB

Em um sistema distribuído, a estrutura da figura anterior deve ser estendida com os módulos que gerenciam comunicações e acessos múltiplos concorrentes, respectivamente: DBV-HUELLE e TASKST. Veja figura a seguir:

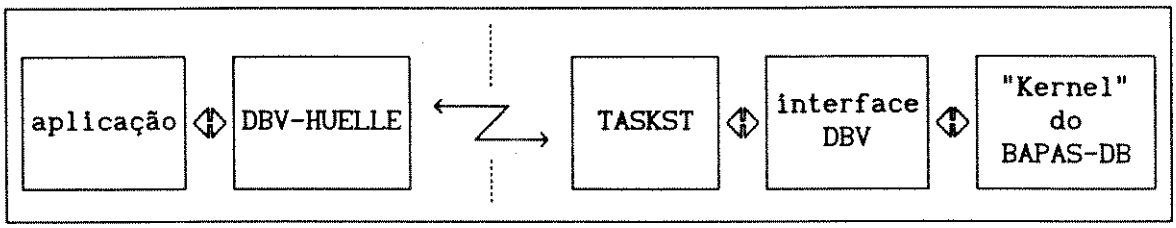



Figura 3.14 - Relação estendida para Sistemas Distribuídos

### 3.2.6.2.1) Linguagens suportadas pela interface DBV

A interface DBV de programas (processos) do BAPAS-DB permite que os programas sejam escritos e compilados em várias linguagens de programação, quais sejam: C, PASCAL, PEARL e PL/M.

Para que um programa tenha acesso às funções disponíveis na interface DBV, basta que em tempo de compilação seja adicionado o arquivo de inclusão correspondente àquela linguagem de programação.

Cada um desses arquivos de inclusão contém declarações de variáveis, tipos, constantes e tudo o mais que seja necessário para o acesso à Base de Dados pelo programa escrito em uma linguagem em particular.

Para a linguagem C, o arquivo de inclusão é denominado: **ICDBV.INC**; para a linguagem PASCAL o nome é: **IPADBV**; para PEARL é: **INTDBV** e assim por diante.

### 3.2.6.2.2) Estratégias de acesso

O acesso dos programas (processos) aos arquivos da Base de Dados é efetuado utilizando-se as estratégias de acesso descritas anteriormente.

### 3.2.6.2.3) Acesso à Base de Dados usando "C"

Para utilizar a interface DBV, cada processo deve proceder de acordo com um esquema bem definido, o qual pode ser descrito pela seguinte estrutura:

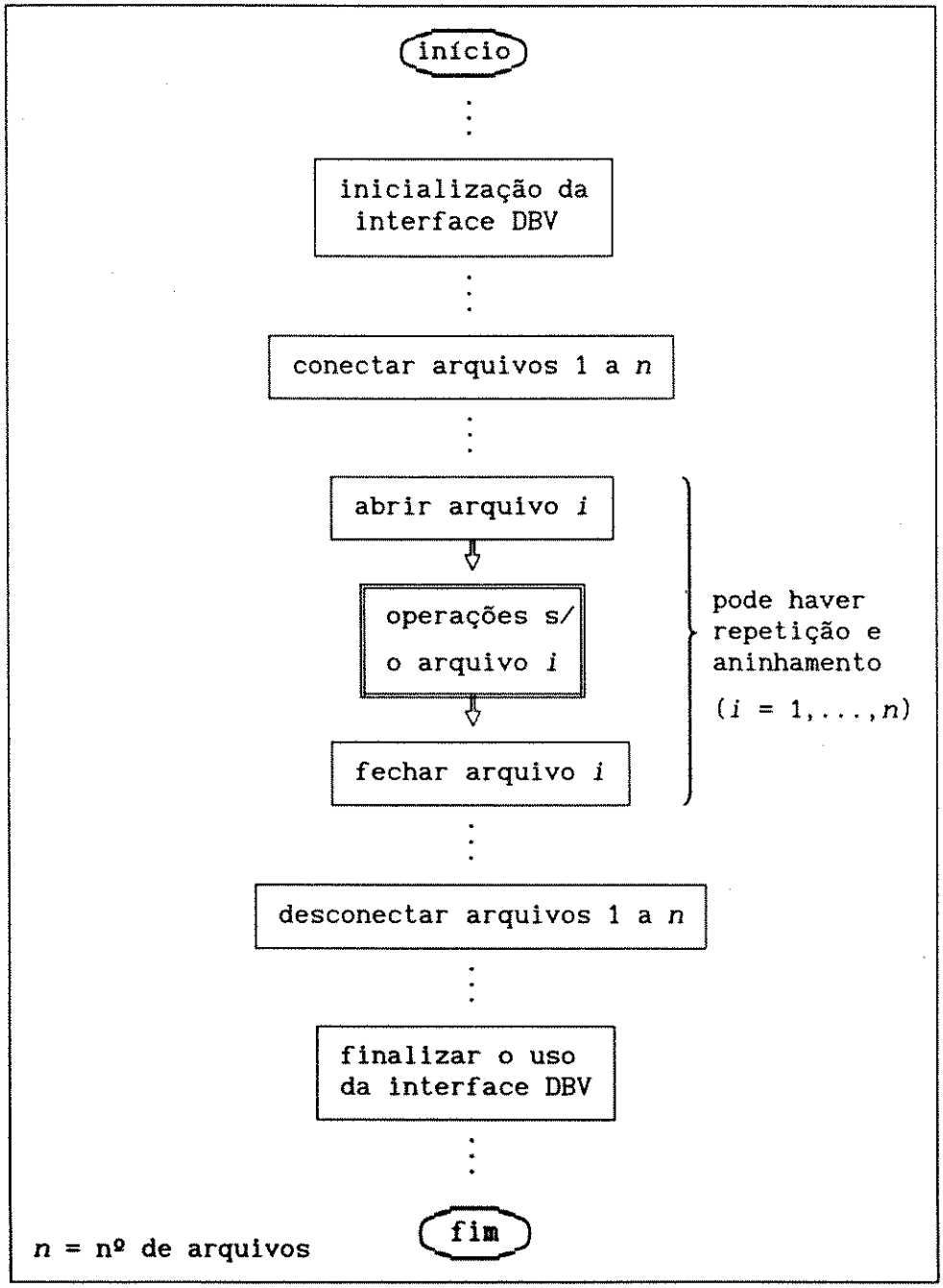



Figura 3.15 - Diagrama do processo p/ utilização da interface DBV

Os diagramas constantes do anexo 3 sumarizam as funções disponíveis na interface DBV que implementam essas e outras operações sobre uma Base de Dados. As funções estão agrupadas de acordo com suas finalidades, compreendendo operações sobre a Base de Dados e seus arquivos, operações sobre os registros, suporte a transações atômicas, suporte a tolerância a falhas e informações sobre os arquivos da Base de Dados.

O exemplo a seguir em "pseudo-código C", apresenta uma estrutura válida de um processo que utiliza chamadas à interface DBV:

```
DBVINITIATION_();
DBVATTACH_(arquivo_1);
DBVATTACH_(arquivo_2);
     ---
DBVOPEN_(arquivo_1);

trabalhando c/ o arquivo 1

DBVOPEN_(arquivo_2);

trabalhando c/ os arquivos 1 e 2

DBVCLOSE_(arquivo_2);
DBVCLOSE_(arquivo_1);
     ---
DBVDETACH_(arquivo_1);
DBVDETACH_(arquivo_2);
DBVEND_();
```

A listagem contida no anexo 2 corresponde ao arquivo "header" (cabeçalho) que deve ser incluído em programas em linguagem 'C', para permitir-lhes acesso à interface DBV do BAPAS-DB.

### 3.2.7) A Interface SQL do BAPAS-DB

Programas de aplicação, em geral, usam chamadas da interface *DBV* para inserir, alterar, ler e apagar registros da base de dados ( bem como outros tipos de chamadas ), a partir de programas escritos em C, FORTRAN, Pascal,

PEARL ou PL/M.

Existe, todavia, um pré-compilador disponível para a linguagem C que cria as chamadas da DBV apropriadas a partir de instruções SQL, com o auxílio do dicionário de dados (veja figura abaixo). Deve-se, antes, criar a base de dados utilizando a linguagem de descrição de dados DDL.

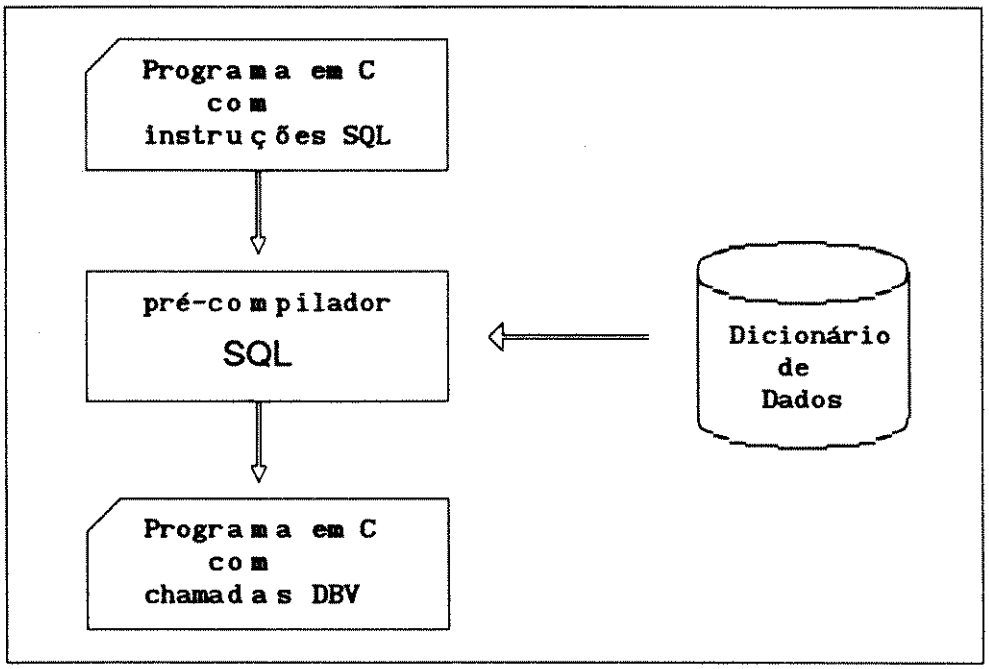



Figura 3.16 - O pré-compilador SQL

A magnitude da linguagem SQL corresponde ao padrão ANSI-SQL com limitações condicionadas ao modelo, referentes à construção de subconjuntos (subsets) e mudanças dinâmicas do esquema da base de dados, porém com extensões do conceito de transação, checkpoint e distribuição.

### 3.2.8) Propriedades de Banco de Dados Distribuído

O BAPAS-DB pode ser instalado em *m* computadores de um sistema distri-

buido de *n* computadores ( 1 ≤ m ≤ n ), com bases de dados disjuntas; o acesso a essas bases de dados pode ser feito a partir de qualquer um dos *n* computadores. Isto é feito com o auxílio dos componentes *DB-server* e *DB-client*, descritos anteriormente, e é valido também para aplicações que utilizem o pré-compilador SQL.

No caso mais simples, no qual o programa de aplicação e o BAPAS-DB estão em um mesmo computador, o programa de aplicação é "linkado" com o BAPAS-DB. Caso o BAPAS-DB esteja em outro computador, o componente DB-client é "linkado" ao programa de aplicação, enquanto que o componente DB-server é "linkado" ao BAPAS-DB.

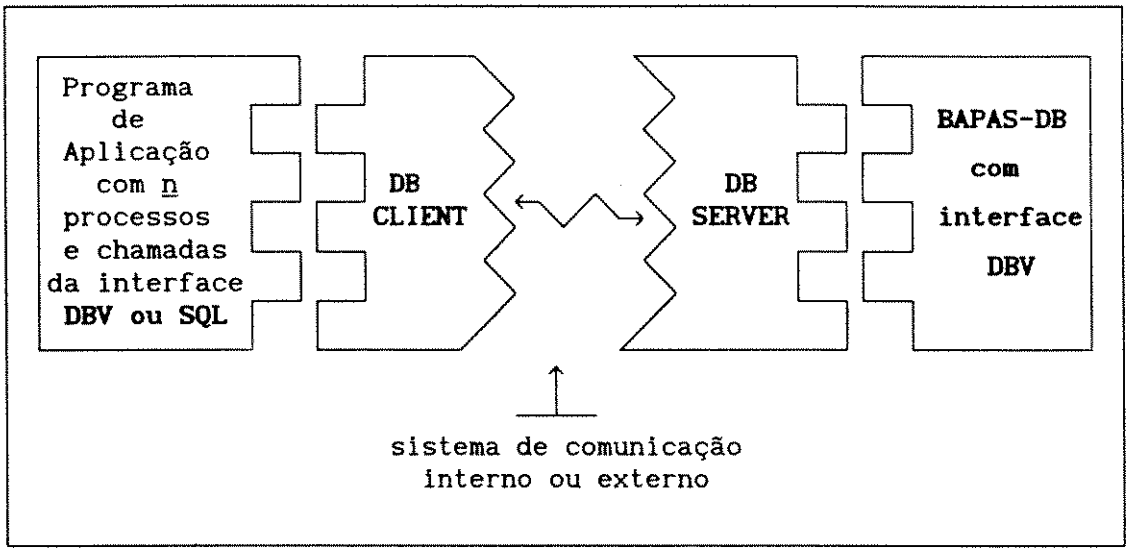



Figura 3.17 - A interface DBV

Para utilização remota, não é necessário modificar as chamadas à base de dados a nível de aplicação; as chamadas à interface DBV (ou instruções SQL) são mapeadas pelo componente DB-client para o componente DB-server correspondente na base de dados apropriada, como uma chamada remota de procedimento (*remote procedure call*). Assim, o DB-server efetua o acesso requerido à base de dados e devolve o resultado correspondente ao processo chamador.

Os componentes DB-server e DB-client foram construidos de tal maneira

que diversos servidores de base de dados podem estar instalados em uma rede; adicionalmente, DB-server e DB-client podem estar ambos instalados em um mesmo computador. Veja figura a seguir.

A configuração de uma rede é feita criando-se uma lista de configuração com os componentes *computer identification* (identificação do computador) e *network address* (endereço de rede). Esta lista deve ser conhecida por todos os participantes da rede. Uma reconfiguração ou alteração na rede é feita simplesmente modificando-se a lista de configuração, o que não afeta os programas de aplicação.

Os componentes DB-server e DB-client executam o protocolo de chamada remota de procedimento (*remote procedure call*) em camadas em uma interface de transporte virtual de acordo com o padrão ISO nível 4 (ver capítulo 6).

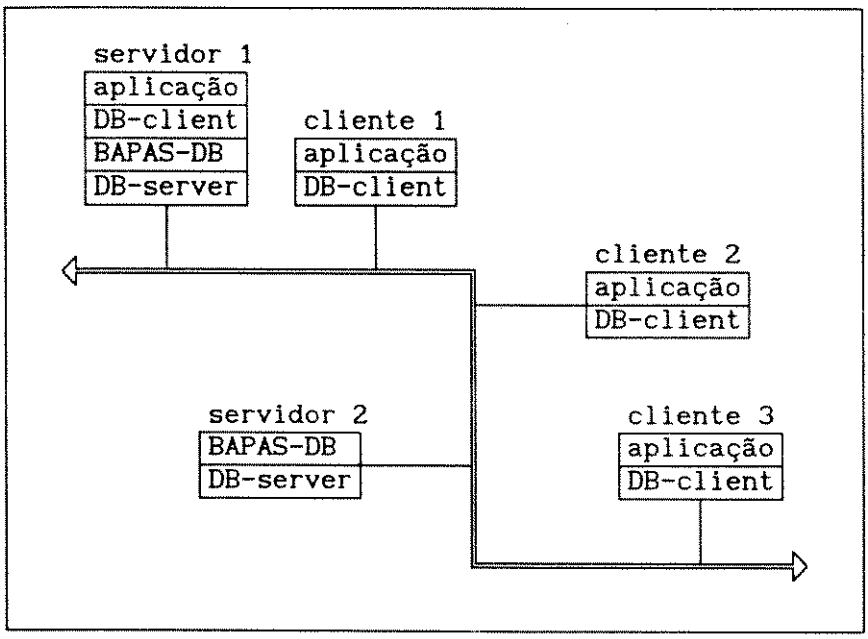



Figura 3.18 - O modelo m-clientes/n-servidores

### 3.2.9) BAPAS-DB como uma base de dados ativa

Algumas vezes em uma rede, certos clientes da base de dados necessitam

saber se outros clientes efetuaram atualizações nos dados. Com essa finalidade foi criado o componente *message routing* (roteamento de mensagem). Se uma operação sobre a base de dados é efetuada em um arquivo pré-determinado (DB-file) do DB-server, e se uma condição, também pré-determinada pelo usuário, é satisfeita, o servidor automaticamente envia mensagens a clientes (DB-clients) previamente escolhidos. Isto é feito "online" com base em uma tabela de decisão.

### 3.2.10) Ferramentas para o BAPAS-DB

A lista a seguir sumariza as ferramentas de programação disponíveis para o uso eficiente e orientado a problema do BAPAS-DB.

**BAPAS-QL** linguagem de consulta interativa ou em forma de máscaras;

**BAPAS-SQL** interface de programa com pré-compilador para o ANSI-SQL com as diferenças já descritas. Disponível para C e PEARL;

**BAPAS-MMC** sistema da máscaras para a criação interativa de diálogos e relatórios. Disponível para C e PEARL sob OS/2 e UNIX.

### 3.2.11) Níveis de configuração e dados técnicos

Veja no anexo 1 informações adicionais relativas ao BAPAS-DB. Essas informações compreendem os diversos níveis de configuração possíveis para o SGBD BAPAS-DB e, adicionalmente, os dados técnicos referentes a esse sistema.

### 3.2.12) Extensões

Apesar de todas as capacidades do SGBD BAPAS-DB descritas neste capítulo, A versão disponível para a plataforma computacional por nós utilizada (IBM PC/AT sob MS-DOS) tem algumas características que, a bem da verdade, não nos satisfizeram.

*** mensagens de erro**

A primeira dessas características é que as mensagens de aviso e erro, eventualmente emitidas por cada um dos componentes do BAPAS-DB, seguiam a seguinte forma:

$$\text{Erro} \begin{cases} \textit{classe} \\ \textit{número} \end{cases}$$

Onde *classe* e *número* correspondem a códigos numéricos.

Para descobrir o significado de cada erro que, porventura, viesse a ocorrer, fazia-se necessário uma busca no manual do BAPAS-DB, onde uma tabela de erros associava a cada erro, definido por sua classe e número, uma classificação (aviso ou erro fatal), uma descrição detalhando o seu significado e, adicionalmente, uma indicação da atitude a ser tomada em face do erro.

Como agravante, a versão do manual disponível para nosso uso não estava integralmente traduzida para o inglês e, asssim, algumas seções do referido manual permaneciam na língua original, ou seja, o alemão. A tabela de erros era uma dessas seções.

Para facilitar a identificação do significado das mensagens de erro por parte dos futuros usuários do sistema, efetuamos a tradução da referida tabela para o inglês e criamos um arquivo de texto correspondente. Desta

forma, pudemos criar funções que permitem a exibição da mensagem descrevendo o erro em lugar de apenas um código numérico.

Ao arquivo contendo as mensagens de erro foi dado o nome de **db_err.msg**, o qual consta do anexo 3.

*** funções do sistema operacional**

Outra característica da versão do BAPAS-DB a nós disponível é que, uma vez dentro de um dos módulos componentes do BAPAS-DB, por exemplo o módulo QL, não se podia efetuar funções do sistema operacional (p.ex.: mudar o diretório corrente) sem ter que abandonar o programa.

Isto nos levou a construir algumas funções com tal finalidade e agrupá-las em um módulo de programa separado. Esse módulo foi designado **tools.c**.

Veja no capítulo 4 mais informações a esse respeito e, no anexo 4, uma tabela explicativa de cada função.

O capítulo seguinte introduz algumas das outras "ferramentas" que identificamos como úteis ou desejáveis para um SAP.

# CAPÍTULO 4

## OUTRAS FERRAMENTAS

## 4.1) Introdução

Este capítulo é dedicado à descrição de alguns dos componentes de um Sistema de Apoio a Projetos identificados anteriormente. São eles, a Interface-Usuário e os Formulários.

Em adição a eles, apresentamos, também, algumas ferramentas cuja necessidade foi sendo detectada à medida em que evoluiam este projeto de Tese e, em especial, o estudo de caso descrito ao final. Nesta categoria se enquadram as estruturas de dados em árvores e as ferramentas do Sistema Operacional.

A característica em comum entre todos estes "componentes" é que não são produtos comerciais. Assim, o código fonte correspondente está disponível para análise, alterações e, possívelmente, extensões.

Tanto o "pacote" de Interface-Usuário baseada em janelas e menus, como as estruturas de dados em árvore, descritos adiante, foram obtidos a partir de livros, os quais constam da bibliografia. Foram, porém, efetuadas alterações, correções, adaptações e extensões a tais sistemas, de modo que se adequassem aos nossos requisitos.

As ferramentas foram adaptadas para suprir necessidades específicas desta Tese; podem, porém, ser utilizadas com pouca ou, até mesmo, nenhuma modificação em outros tipos de aplicações.

A seguir uma descrição de cada uma dessas "ferramentas".

## 4.2) A Interface-Usuário [C-*Weiskamp 88*]

Desejamos obter uma Interface-Usuário amigável, com características gráficas, que facilite a interação de um usuário inexperiente com um sistema de apoio a projetos. A apresentação visual deve favorecer a tomada de decisões e a entrada de dados; porém, a interface não deve se constituir em obstáculo ao usuário experiente.

A interface proposta baseia-se no uso de janelas e menus para a apresentação e entrada de dados. Veja exemplos de utilização no anexo 5.

### 4.2.1) Janelas

Janelas são objetos gráficos que nos permitem dividir a tela em um número de regiões distintas. Cada região pode ser acessada de maneira independente através de funções padronizadas, tais como escrita e leitura em uma janela.

Através do uso de janelas, pode-se organizar os dados na tela, de maneira a torná-los mais legíveis para o usuário, inclusive com o uso de cores diferentes e realçando informações importantes através de sinais sonoros, caracteres piscantes ou em vídeo reverso, etc.

Vamos definir três tipos diferentes de janelas: **psuedo, overlap** e **pointer**. O primeiro tipo de janela (psuedo) é uma janela "invisível", ou seja, não aparece na tela, porém é uma janela válida para o sistema de janelas, e é útil para, por exemplo, criar a janela base do sistema (base window).

O segundo tipo (overlap) consiste de janelas que podem ser sobrepostas parcial ou totalmente umas às outras, sem que as informações constantes delas sejam perdidas. É o tipo de janela mais comumente utilizado.

O último tipo de janela (pointer) compõe-se de uma janela e de uma seta piscante. Sua utilidade maior está na exibição de mensagens de advertência ou erro. Veja figura a seguir.

#### 4.2.1.1) Coordenadas de uma janela

Faz-se necessária a utilização de dois tipos de coordenadas para a localização de pontos na tela, quais sejam: **coordenadas absolutas** e **coordenadas relativas**. O canto superior esquerdo da tela (0,0) é uma coordenada absoluta; note que o canto superior esquerdo da janela também tem coordenadas (0,0); porém, estas coordenadas são em relação à janela e não à tela, como no caso anterior, podendo corresponder a (10,40), por exemplo, em coordenadas absolutas. Veja figura a seguir.

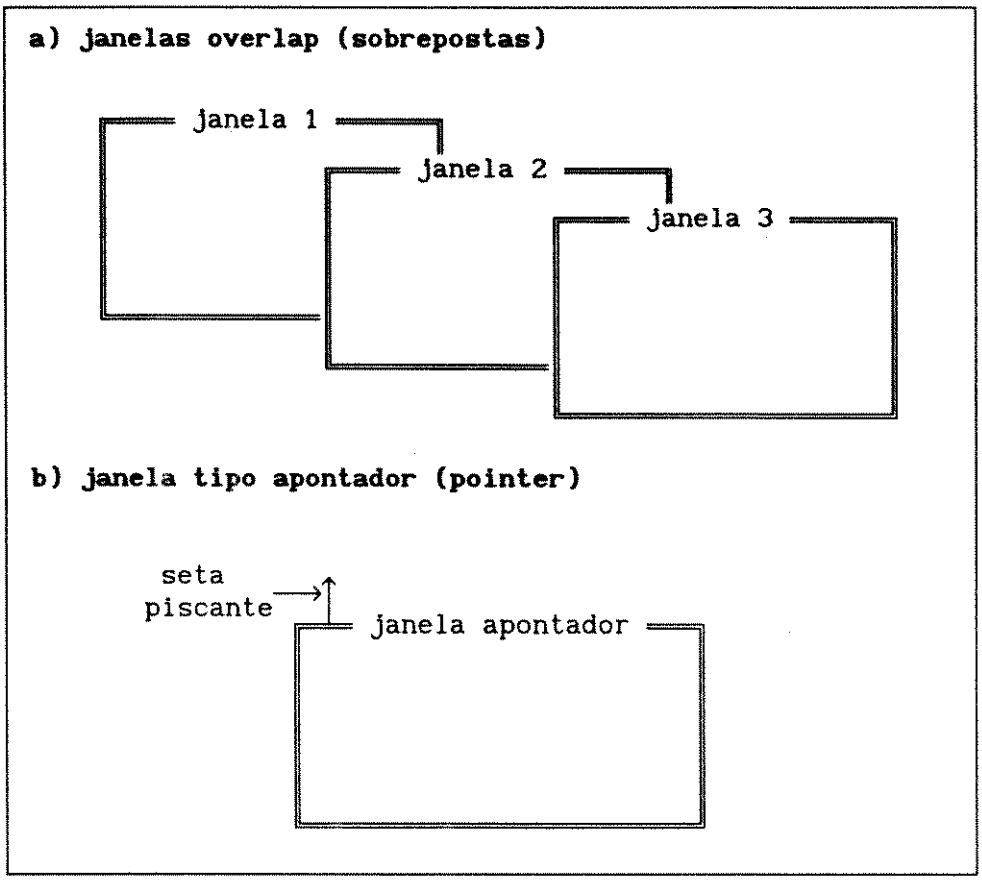



Figura 4.1 - Tipos de janelas

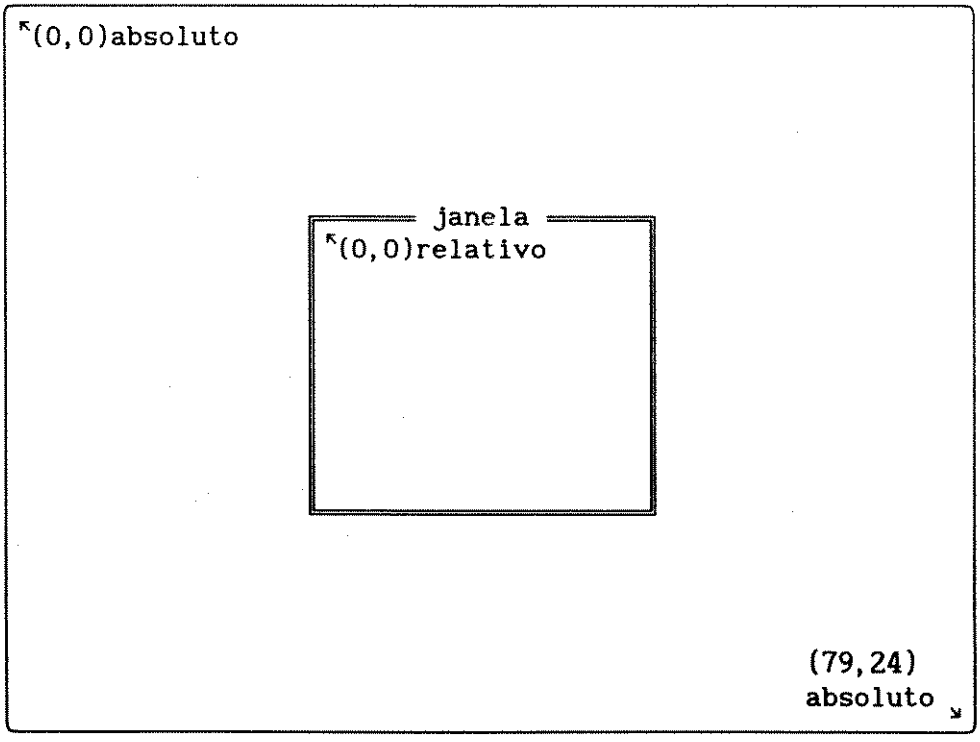



Figura 4.2 - Diagrama das coordenadas da tela e da janela

#### 4.2.1.2) Componentes de uma janela

Os componentes de uma janela podem ser divididos em duas categorias: elementos e atributos. Veja figura a seguir.

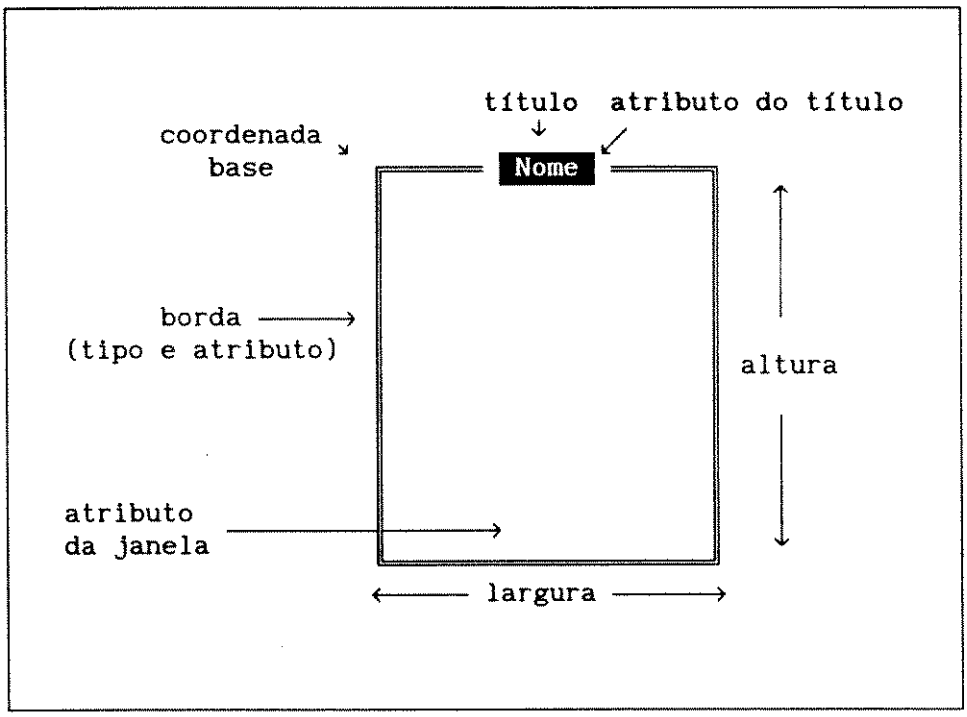



Figura 4.3 - Componentes de uma janela

##### 4.2.1.2.1) Elementos

- **coordenada base**: A coordenada base e a localização absoluta (na tela) do canto superior esquerdo da janela, dado como par (*coluna*, *linha*);

- **altura**: É o número de linhas ocupado pela janela;

- **largura**: É o número de colunas ocupado pela janela;

- **borda**: A borda (ou caixa) define as fronteiras visíveis da janela;

- **título**: A janela pode ter um título (cadeia de caracteres).

#### 4.2.1.2.2) Atributos

Cada atributo define uma cor ou característica de vídeo de um elemento da janela. O conjunto de atributos consiste do seguinte:

- **atributo da janela**

- **atributo da borda**

- **atributo do título**

### 4.2.1.3) Estrutura de dados da janela

Criou-se uma estrutura de dados especial para armazenar as informações necessárias acerca de cada janela presente em uma tela.

Como o sistema de janelas proposto permite apresentar uma janela sobreposta a outras, faz-se necessário armazenar na memória o conteúdo da região da tela na qual uma janela deve ser apresentada, de maneira que ao ser retirada a janela, seja possível devolver a tela a seu estado anterior.

Uma estrutura de dados do tipo **pilha** é a mais apropriada para esse fim, pois esse tipo de estrutura funciona segundo uma estratégia LIFO (Last In - First Out), ou seja, o elemento inserido por último é o que está no "topo" da pilha e será retirado em primeiro lugar.

Veja figura a seguir.

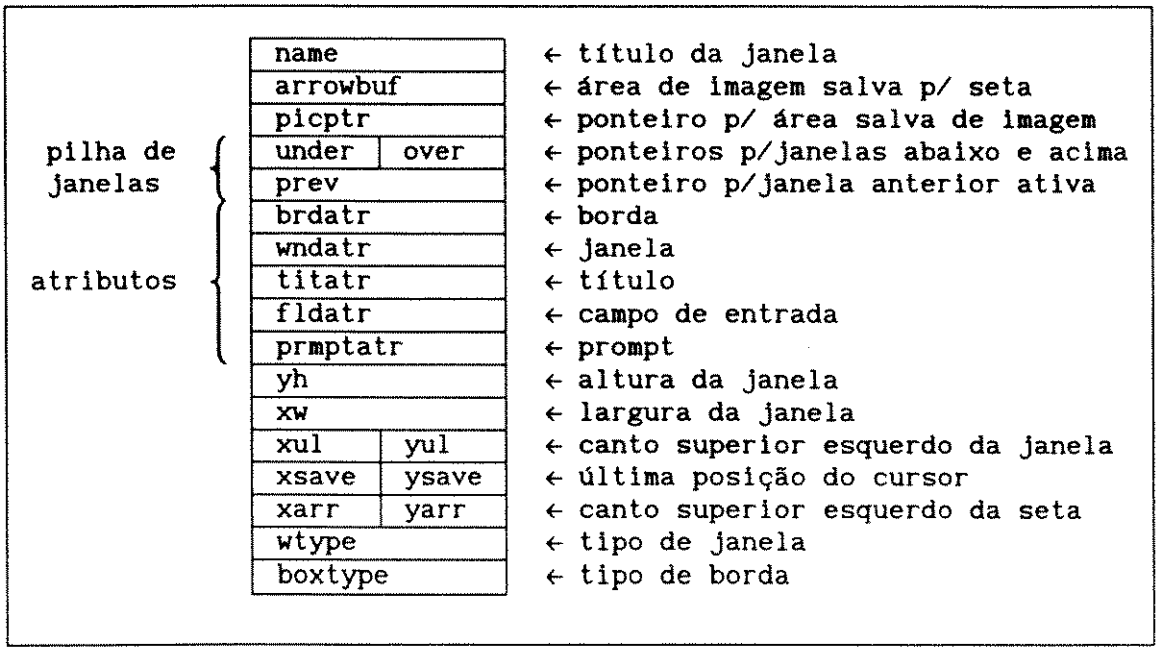



Figura 4.4 - Estrutura de dados de uma janela

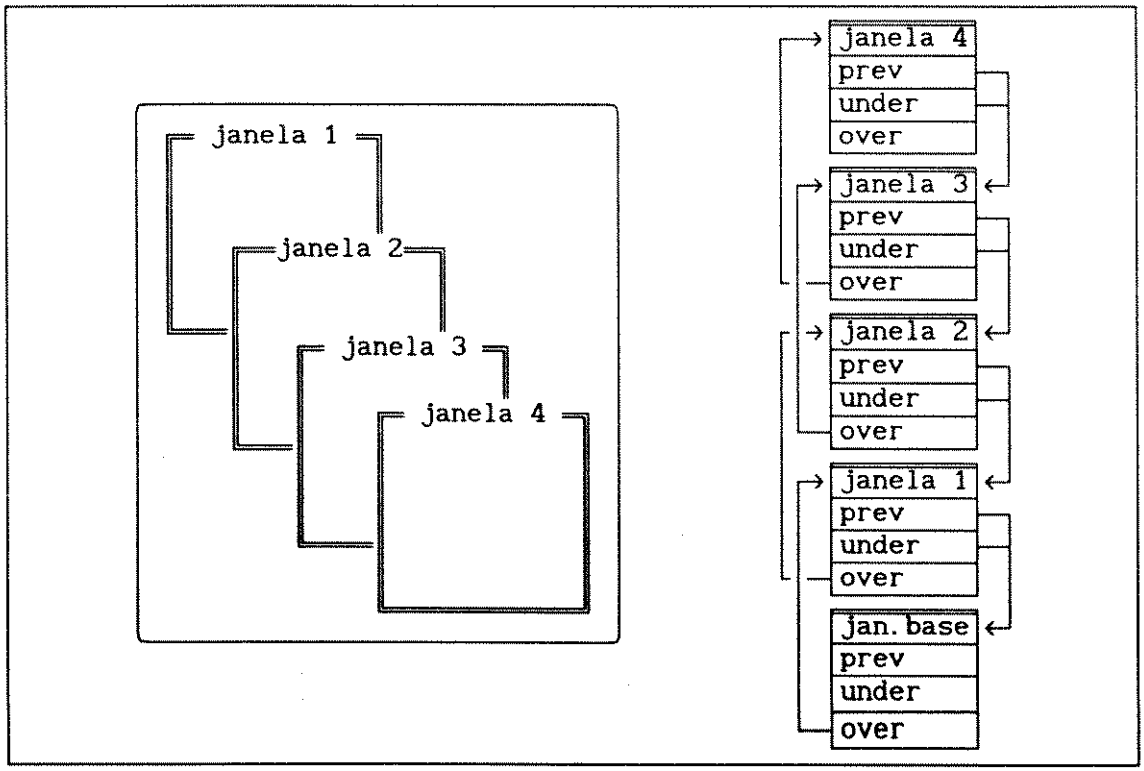



**Figura 4.5 - Pilha de janelas**

Uma pilha é um dispositivo que permite armazenar e recuperar com rapidez informações através de operações de **push** e **pop**. A pilha pode ser usada para armazenar o conteúdo da região sobre-escrita da tela e outros dados necessários para o acesso a uma janela, tais como: coordenadas base, posição do cursor e atributos.

Vamos chamar esta pilha de pilha de janelas, pois cada vez que uma janela é criada, ela deve ser armazenada na pilha; e quando ela é removida da tela, também é removida da pilha. Veja a figura anterior.

#### 4.2.1.4) Funções relacionadas a janelas

Os anexos 4 e 6 contêm uma listagem das funções que permitem implementar o sistema de janelas. Tais funções podem ser agrupadas em três categorias genéricas:

- funções de "alto nível" - Invocadas pelo usuário, permitem inicializar o sistema de janelas, criar, mover, selecionar e remover janelas;

- funções "internas" - Normalmente são invocadas apenas a partir das funções de "alto nível". Envolvem operações que devem ser transparentes para o usuário; por exemplo, manipulação da pilha de janelas;

- funções de "suporte" - funções de apoio à manipulação dos elementos de tela, por exemplo: forma e posição do cursor, rolamentos e limpeza de tela, etc.

### 4.2.2) Menus

A interface-usuário proposta possui funções que suportam a criação, acesso e movimentação dinâmica de menus com diversas cores, tamanhos e estruturas, sendo possível, inclusive, a criação de menus com número de itens maior do que o suportado pela região da tela delimitada pela janela que circunscreve o menu.

Em tal caso, a janela do menu será, na verdade, uma janela "virtual" para a lista de itens componentes do menu. A visualização dos itens não exibidos pode ser efetuada, então, por rolamento.

Veja figura a seguir:

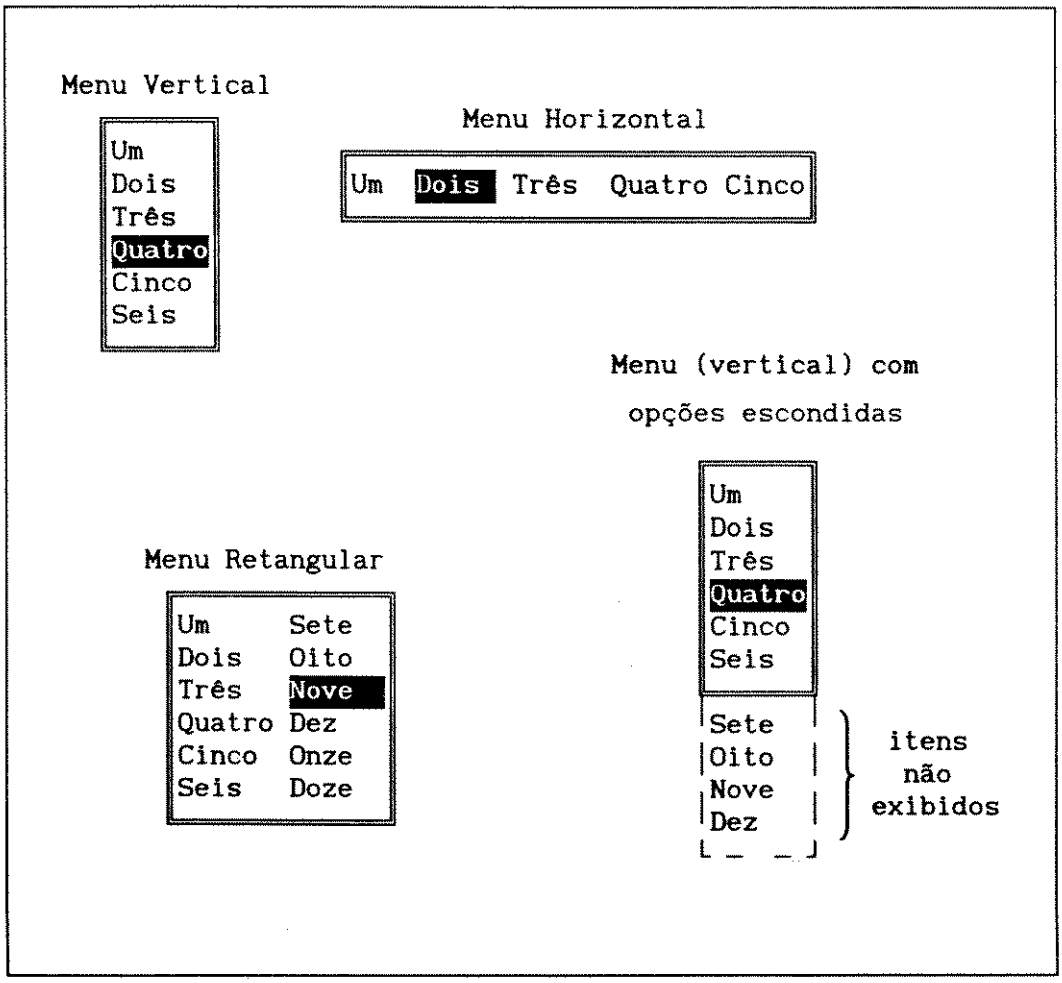



Figura 4.6 - Tipos de menus

#### 4.2.2.1) Componentes

A Figura a seguir sumariza os componentes de um menu criado em nosso sistema.

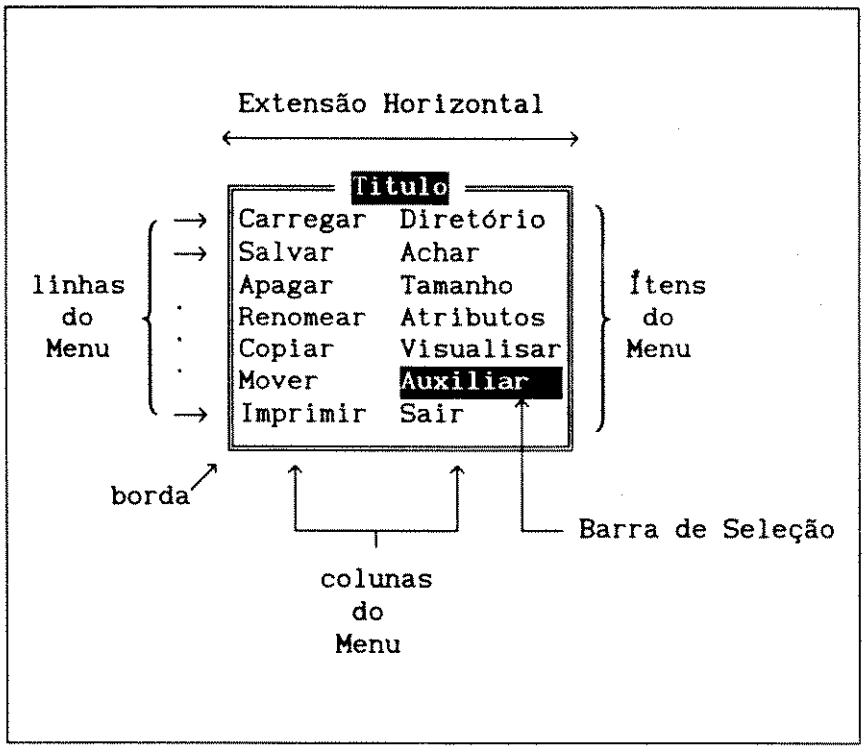



Figura 4.7 - Elementos de um menu

- **título:** é o título correspondente ao menu;

- **borda:** é o tipo de borda do menu (simples ou dupla);

- **itens do menu:** lista das opções que compõem o menu;

- **colunas e linhas:** corresponde ao número de colunas e linhas visíveis do menu;

- **extensão horizontal:** define a largura absoluta (em número de colunas) de um menu e, consequentemente, o espaçamento entre as colunas do menu;

- **barra de seleção:** define qual item do menu está selecionado. Pode-se movê-la através das teclas de cursor.

#### 4.2.2.2) Estrutura de dados do menu

Criou-se uma estrutura de dados especial para armazenar as informações necessárias acerca de cada menu presente em uma tela. Como o sistema de janelas proposto permite apresentar uma janela sobreposta a outras, e um menu tem por base uma janela, pode-se sobrepor menus uns aos outros.

As duas figuras a seguir representam, respectivamante, a estrutura de dados em que está baseado o sistema de menus, e a sequência de funções chamadas para criar um menu em determinada região da tela.

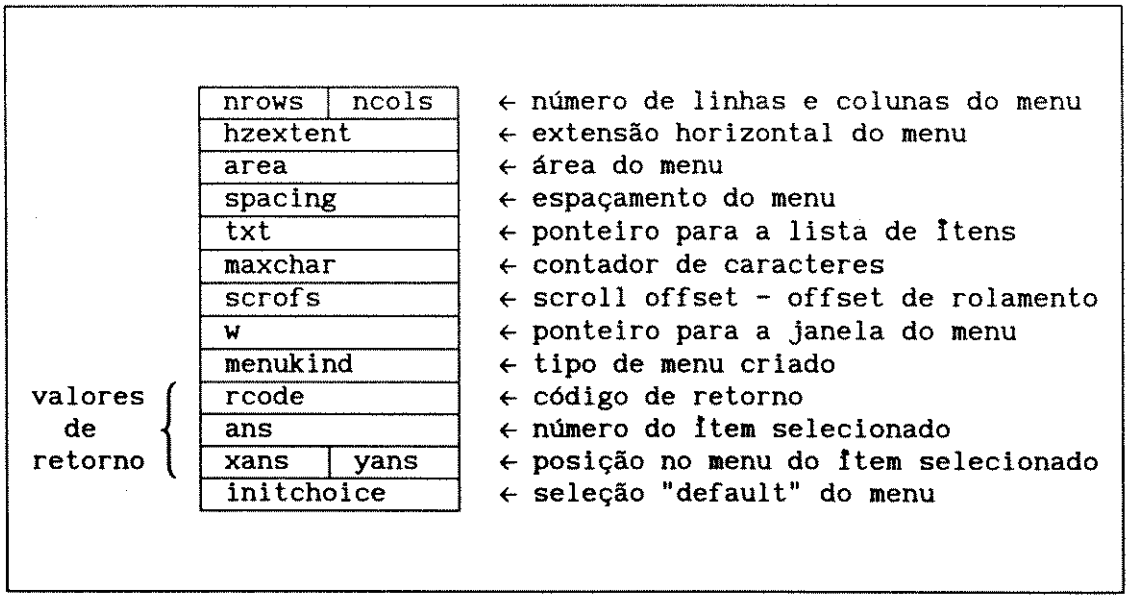



Figura 4.8 - Estrutura de dados de um menu

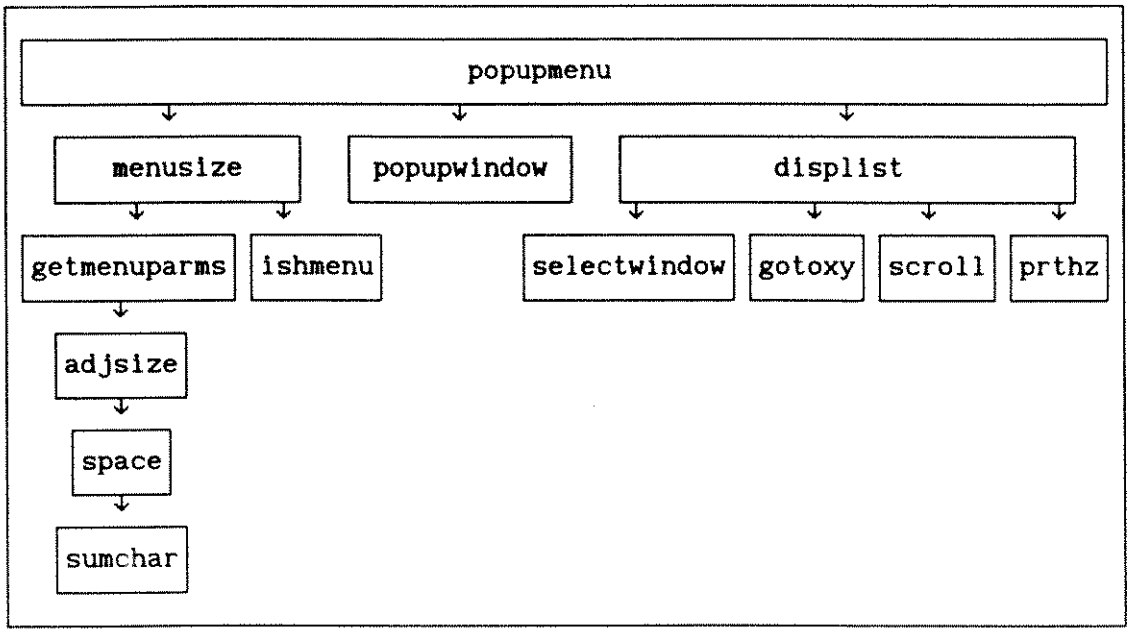



Figura 4.9 - Funções chamadas para criar um menu

Note, na primeira figura, que existem componentes da estrutura que devem ser fornecidos quando da chamada da função que cria menus, e outros componentes que devem ser calculados internamente pela função, sendo que pode ser necessário ajustar alguns parâmetros (dados ou calculados) para que o menu esteja totalmente contido na tela. Deve ser observado, ainda, que alguns componentes são valores de retorno, que são calculados quando seleciona-se um item do menu ou quando da saída do mesmo.

Por outro lado, na segunda figura é instrutivo acompanhar a sequência de chamada das funções que assessoram **popupmenu** no cálculo dos parâmetros necessários à criação de um menu.

Uma das figuras constante do anexo 4 relaciona as teclas, que movimentam a barra de seleção de um menu, e os códigos de retorno existentes.

#### 4.2.2.3) Funções relacionadas a menus

Os anexos 4 e 6 contêm uma listagem das funções que permitem implementar o sistema de menus. Tais funções podem ser agrupadas em duas categorias:

- funções de "alto nível" - Invocadas pelo usuário, permitem criar, mover e remover menus;

- funções "internas" - Normalmente são invocadas apenas a partir das funções de "alto nível". Envolvem operações que devem ser transparentes para o usuário; por exemplo, cálculo das dimensões do menu, espaçamento entre seus itens, etc.

### 4.2.3) Lista de cadeias de caracteres

Desafortunadamente a linguagem C não provê suporte para a manipulação de listas de cadeias de caracteres.

Assim, como esse tipo de lista faz-se necessário para armazenar os itens dos nossos menus, utilizou-se em nosso sistema o "pacote" descrito a seguir, o qual possibilita operações de criação, concatenação e destruição de listas de cadeias de caracteres, bem como inserção, apagamento e busca de elementos na lista, além de outras funções úteis no manuseio de listas.

O encadeamento duplo é o ideal para listas desse tipo, pois permite a movimentação nos dois sentidos da lista.

Por medida de simplicidade, porém, utilizam-se listas com encadeamento simples, cuja estrutura é descrita a seguir.

#### 4.2.3.1) Estrutura de dados da lista de cadeias de caracteres

A estrutura de dados que suporta a lista é composta de uma parte relativa à lista em si (*hn*), com apontadores para o primeiro e o último elementos da lista, e informações sobre o tamanho da lista e dos seus elementos. A outra parte é composta de uma estrutura que armazena informações sobre o elemento em si e de um apontador para o próximo elemento da lista.

Cada elemento da lista é composto de uma chave (inteiro), informações sobre espaço de memória alocado e efetivamente utilizado, e da cadeia de

caracteres propriamente dito. Veja figura a seguir.

Deve ser observado, que existem campos que armazenam informações sobre os tamanhos da lista e do elemento da lista com maior tamanho, não sendo necessário percorrê-la inteira para obter tais valores economizando-se, assim, tempo em muitas das operações efetuadas sobre as listas.

A seguir, relaciona-se o conjunto de funções que permitem operar sobre a lista e seus elementos, sendo possível a inserção (ou remoção) de elementos em qualquer posição da lista.

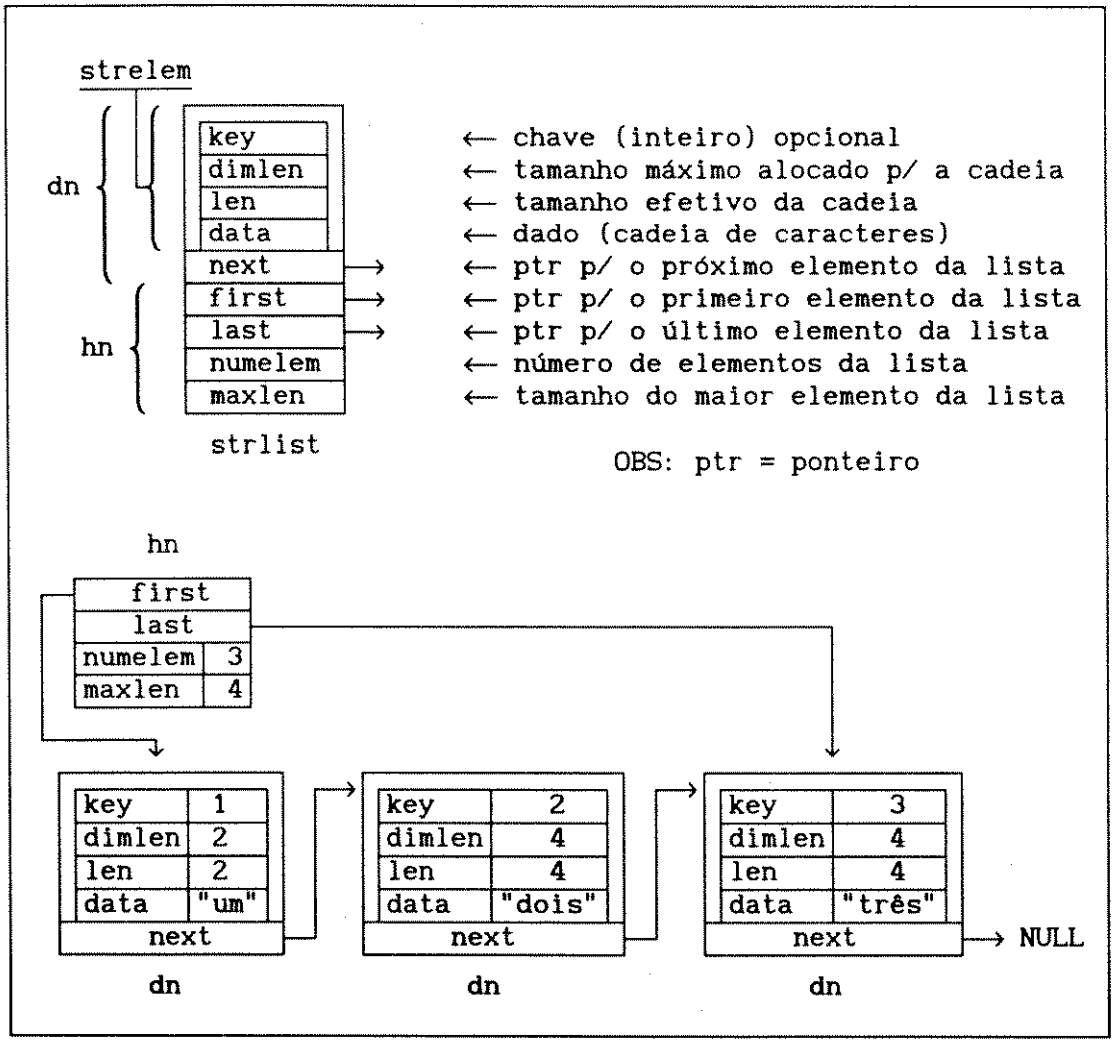



Figura 4.10 - Estrutura de dados da lista de cadeias de caracteres e exemplo

#### 4.2.3.2) Funções relacionadas a listas de cadeias de caracteres

Os anexos 4 e 6 contêm uma listagem das funções que permitem implementar as listas de cadeias de caracteres. Tais funções podem ser agrupadas em duas categorias:

- funções de "alto nível" - Invocadas pelo usuário, permitem criar, concatenar e remover listas e, também, efetuar operações de inserção, apagamento e localização de seus itens;

- funções "internas" - Normalmente são invocadas apenas a partir das funções de "alto nível". Envolvem operações que devem ser transparentes para o usuário; por exemplo, gerenciamento dos "nós" da lista.

### 4.2.4) Descrição dos módulos

* **strlst.c**: Contém as estruturas da dados e funções que implementam listas de cadeias de caracteres.

* **ioutil.c**: Contém definições para as teclas, e funções usadas para entrada e saída de dados.

* **scrnio.c**: Contém rotinas de suporte para utilização genérica da tela, e de suporte ao uso de janelas e menus.

* **popup.c**: Contém as estruturas de dados e as funções que suportam a implementação do sistema de janelas.

* **smenu.c**: Contém as estruturas de dados e as funções que suportam a implementação de menus.

* **pdmenu.c**: Contém as funções que suportam a implementação de menus do tipo "*pulldown*" (cortina).

## 4.3) Árvores [C-*Sobelman 85*]

Árvores são estruturas de dados hierárquicas muito comuns, tanto em sistemas de software como no próprio dia-a-dia. Como exemplos, podemos citar: árvores genealógicas e estruturas organizacionais de empresas.

Como a recursividade se adapta excepcionalmente bem a árvores, é comum o uso de funções recursivas em associação a este tipo de estrutura de dados.

Uma árvore (assim como uma lista ligada) é uma coleção de dados armazenados em **nós**, os quais podem ser adicionados ou retirados de uma dada árvore em tempo de execução. Entretanto, ao contrário de uma lista ligada, os nós de uma árvore mantêm certas relações de hierarquia entre si.

Veja um exemplo típico de árvore na figura a seguir.

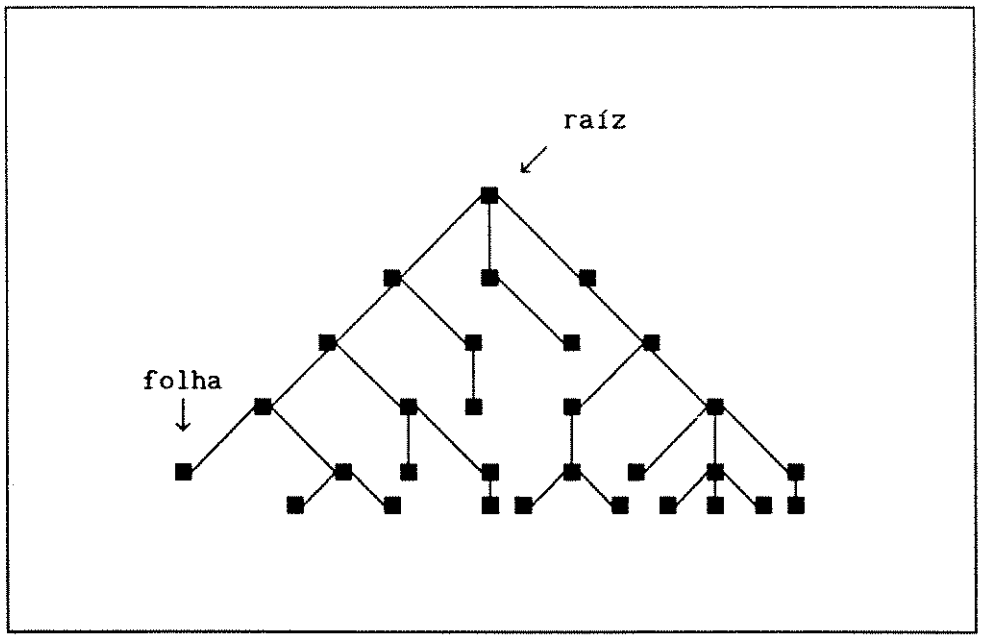



Figura 4.11 - Exemplo de uma árvore m (onde m é o nº máx.de filhos por nó)

O nó principal de uma árvore é dito ser o *nó raíz* (root) e os nós terminais (nós que não dão origem a outros nós) são chamados *nós folha* (leaf).

As relações de hierarquia entre os nós são aquelas comumente utilizadas para descrever relações de família ou parentesco, tais como: *pai, filho, prole, ancestral, descendente, etc...*

### 4.3.1) Estrutura de dados de uma árvore

A figura a seguir apresenta a disposição dos elementos de um nó genérico de uma árvore.

Note que os filhos de um nó são organizados em uma lista ligada. Utiliza-se encadeamento simples por questões de simplicidade. Poder-se-ia, porém, converter para encadeamento duplo sem grandes alterações.

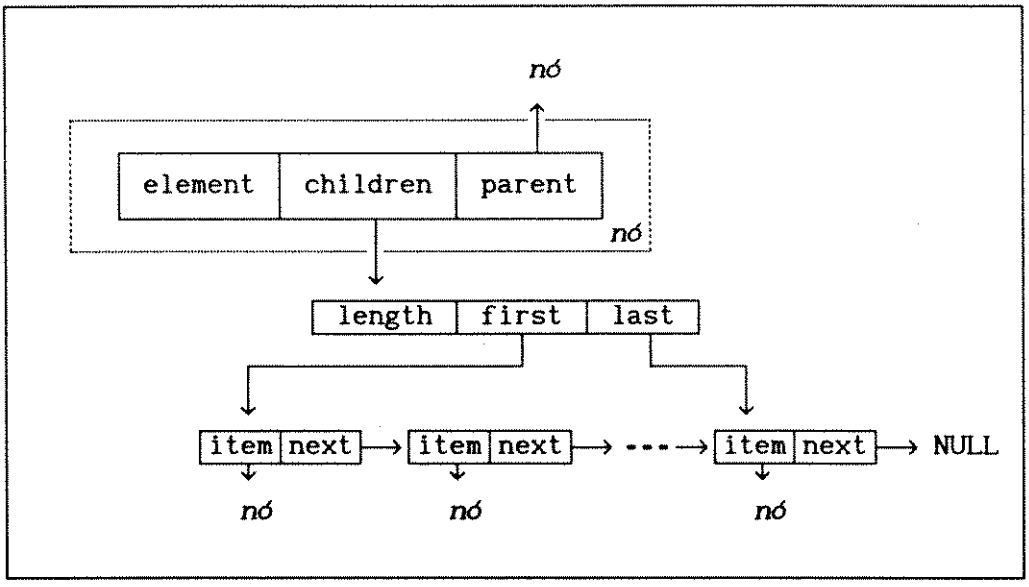
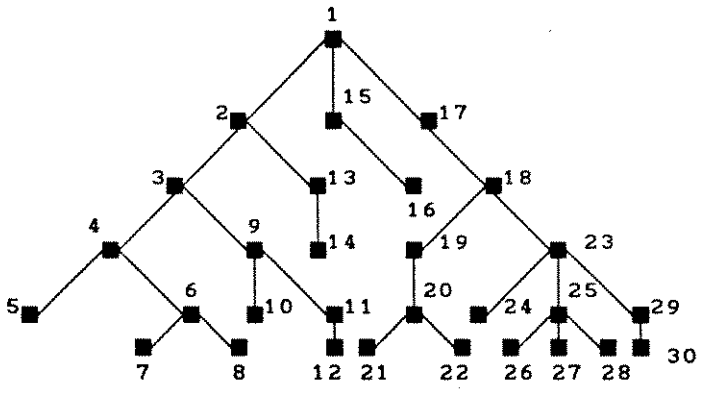



Figura 4.12 - Diagrama de um nó de uma árvore

### 4.3.2) Ordenamento de nós

O ordenamento dos nós é importante em estruturas do tipo árvore para, por exemplo, assegurar que uma busca em uma árvore cubra cada um dos nós uma única vez. O método de ordenar os nós é puramente árbitrário, porém a ordem deve ser precisa e bem definida pois o ordenamento de nós em árvores não é tão simples como em listas ligadas, dado que cada nó tem um número variável de filhos, e cada filho pode ter seus próprios filhos, e assim por diante.

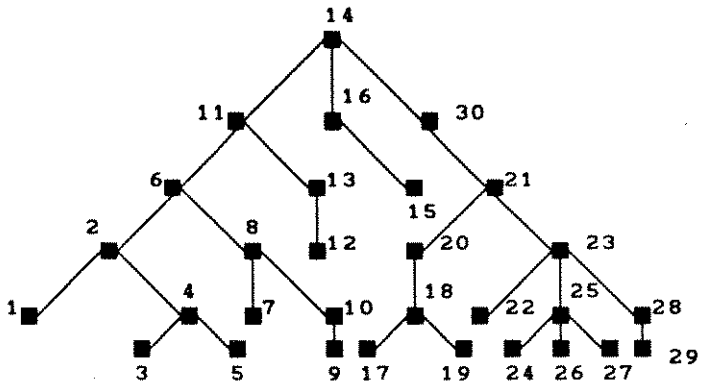
Entre os possíveis métodos de ordenamento, três são os mais populares, quais sejam: **pré-ordem** (preorder), **em-ordem** (in-order) e **pós-ordem** (postorder). Os métodos são similares, e a definição de cada um deles é feita de maneira recursiva. Veja figuras a seguir:




1) se uma árvore é composta de um único nó, a pré-ordem consiste desse nó.

2) se uma árvore consiste de mais de um nó, a pré-ordem consiste da raíz seguida da pré-ordem de cada sub-árvore, da esquerda para a direita.

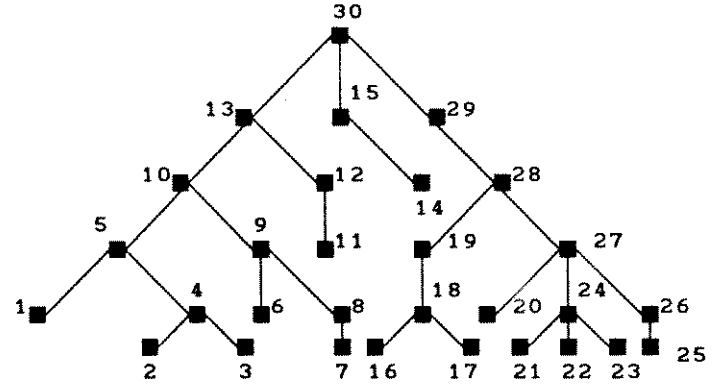
Figura 4.13 - Ordenamento de árvore em **PRÉ-ORDEM**




1) se uma árvore é composta de um único nó, a em-ordem consiste desse nó.

2) se uma árvore consiste de mais de um nó, a em-ordem consiste da em-ordem da sub-árvore do filho do canto esquerdo, seguida da sua raíz, e da em-ordem de cada sub-árvore restante, da esquerda para a direita.

Figura 4.14 - Ordenamento de árvore em **EM-ORDEM**




1) se uma árvore é composta de um único nó, a pós-ordem consiste desse nó.

2) se uma árvore consiste de mais de um nó, a pós-ordem consiste da pós-ordem de cada sub-árvore, da esquerda para a direita, seguida da raíz.

Figura 4.15 - Ordenamento de árvore em **PÓS-ORDEM**

### 4.3.3) Funções relacionadas a árvores

Os anexos 4 e 6 contêm uma listagem das funções que permitem implementar as estruturas de dados em árvore. Tais funções podem ser agrupadas em três categorias:

- funções de "alto nível" - Invocadas pelo usuário, permitem criar e remover árvores, bem como adicionar, modificar e remover nós de uma árvore. Permitem, ainda, percorrer a árvore e localizar e/ou testar seus nós.

- funções "internas" - Normalmente são invocadas apenas a partir das funções de "alto nível". Envolvem operações que devem ser transparentes para o usuário; por exemplo, o preenchimento da estrutura de dados de um nó.

- funções de "suporte" - Implementam um sistema de lista de encadeamento simples, cuja finalidade é armazenar apontadores para os "filhos" de cada nó.

### 4.3.4) Módulos componentes

* **tree.c**: Contém as estruturas de dados e as funções que implementam árvores hierárquicas onde cada nó pode ter n filhos.

* **list.c**: Contém as estruturas de dados e as funções que implementam as listas utilizadas pelos nós das árvores do módulo *tree.c* para armazenar a lista de seus filhos.

### 4.3.5) Exemplos de utilização

Veja no anexo 5, ao final da Tese, um exemplo completo de utilização das funções que implementam e permitem efetuar operações sobre a estrutura de dados em árvore descrita.

## 4.4) Formulários

Como já foi dito anteriormente, a base do sistema de apoio a projetos é um banco de dados, o qual é divido em vários arquivos. Cada arquivo vai conter informações que podem ou ser geradas pelo próprio sistema, ou requisitadas ao usuário. Neste último caso, como geralmente as informações necessárias estão dispersas através da empresa, deve-se fazer um levantamento prévio das pessoas que podem ser detentoras desse tipo de informações e, então, imprimir formulários que correspondam fielmente aos arquivos da base de dados a serem preenchidos.

Esses formulários devem, então, ser distribuidos às pessoas selecionadas para que elas os preencham, talvez sob a assistência de uma pessoa qualificada a prover orientação a respeito de eventuais dúvidas.

Preenchidos os formulários, as informações deles constantes devem ser repassadas à base de dados do sistema. É interessante utilizar-se o sistema de janelas e menus, descrito anteriormente, para facilitar o trabalho da pessoa encarregada da transcrição das informações dos formulários para a base de dados.

Os formulários devem ser concisos e visualmente "limpos", sendo desejável que sejam distribuidos em conjunto com um pequeno guia de referência que forneça informações mais detalhadas sobre cada campo dos formulários.

Para fins de referência, observe os formulários correspondentes ao estudo de caso, constantes do anexo 8.

Note que cada formulário inclui uma descrição sumária do seu objetivo, e do significado de cada um dos campos a serem preenchidos.

Para a elaboração dos formulários pode-se utilizar um editor de texto de propósito geral, ou ferramentas específicas para esse fim. No nosso caso, utilizou-se o próprio editor de textos através do qual foi redigida a Tese

(no caso, o **ChiWriter**[1]); poder-se-ia, porém, utilizar outro editor de textos qualquer, tal como o **Norton Editor**[2] ou, ainda, alguma ferramenta especialmente desenvolvida para facilitar o projeto de formulários, por exemplo: **FORMTOOL**[3].

### 4.5) Ferramentas do Sistema Operacional

Durante a execução de um programa, muitas vezes sente-se falta de ferramentas que permitam acesso a funções do sistema operacional. Pode-se, por exemplo, necessitar saber (ou mudar) a unidade de disco e o diretório em que se está trabalhando.

Outra facilidade desejável é poder-se, no decorrer do programa, invocar um editor de texto, trabalhar sobre esse texto, salvá-lo e depois retomar a execução do programa a partir do ponto em que foi suspensa.

Ao invés de um editor de texto, pode ser necessário executar um programa qualquer ou, até mesmo, invocar o interpretador de comandos do sistema operacional e depois retornar ao programa principal.

Em qualquer desses casos, é importante que se resguarde o conteúdo da tela, para que não seja perdida nenhuma informação.

Para as finalidades delineadas acima (bem como outras que possam ser detectadas "a posteriori"), criou-se um módulo especial que agrupa as ferramentas que desempenham as funções descritas. Essas funções tornam-se disponíveis (durante a execução de um programa) ao pressionar-se uma tecla especial previamente definida (p.ex.: F9).

Foram utilizadas as facilidades proporcionadas pelo sistema de gerenciamento de janelas e menus, o que facilitou sobremaneira nosso trabalho.

Ao pressionar-se a tecla que invoca o conjunto de ferramentas, um menu é apresentado na tela, podendo-se, então, executar várias funções a partir dele, tais como: verificar e/ou mudar o disco e/ou o diretório corrente; criar ou remover sub-diretórios e invocar um processador de textos ou o interpretador de comandos do sistema operacional.

Observe-se que no caso de invocar a execução de um outro programa qual-

---

[1] *ChiWriter* é marca registrada da Horstmann Software Design Corporation
[2] *Norton Editor* é marca registrada de S.Reifel & Co.
[3] *FORMTOOL* é marca registrada da BLOC Development Corporation

quer, seja este um programa genérico, um processador de textos, ou até o interpretador de comandos do sistema operacional, o programa principal deve permanecer residente na memória para sua execução poder ser retomada ao final do programa invocado.

### 4.5.1) Funções relacionadas às ferramentas do Sistema Operacional

O anexo 4 contém uma listagem das funções que permitem implementar as ferramentas de acesso ao Sistema Operacional, com as características descritas no item anterior. Como fontes de referência podemos citar [C-*TC 87*], [C-*MSC 87*] e [C-*Norton 88*].

### 4.5.2) Módulos componentes

* **tools.c**: Contém as funções que suportam chamadas ao sistema operacional (para, por exemplo, verificar o diretório corrente, listá-lo, selecionar ou criar outro, invocar um editor, etc ...).

O capítulo seguinte discorre sobre a nossa proposta de metodologia de implementação de Sistemas de Apoio a Projetos.

# CAPÍTULO 5

## METODOLOGIA

## 5.1) Introdução

Este capítulo é dedicado à proposição de uma metodologia para a criação de Sistemas de Apoio a Projetos nos moldes descritos nos capítulos anteriores e utilizando o conjunto das ferramentas já apresentadas.

Essa metodologia foi planejada para ser independente das ferramentas a ela agragadas e das (in)compatibilidades destas.

## 5.2) A Metodologia

Nossa proposta é a de uma metodologia composta de cinco "fases" sucessivas, que permitam a construção ordenada de um Sistema de Apoio a Projetos, ao estilo dos delineados no capítulo 1. Veja figura na página seguinte.

A primeira fase trata da identificação do problema e dos recursos necessários ao SAP para solucioná-lo; na segunda fase é efetuada a configuração do SAP de forma a atender os requisitos impostos pelo projeto; a terceira fase corresponde à criação de formulários de pesquisa e de uma Base de Dados para o sistema; a quarta fase envolve a coleta de dados através dos formulários e seu armazenamento da Base de Dados; na quinta fase ocorre a manipulação das informações por parte de programas componentes do SAP, com possibilidade de criação recursiva de novos formulários e de novos arquivos da Base de Dados.

Nas seções a seguir, temos uma descrição mais detalhada da funcionalidade de cada uma das fases da metodologia proposta.

### 5.2.1) 1ª fase - Formulação do Problema

Nesta "fase", tenta-se levantar informações relevantes sobre o projeto que será implementado com o auxílio do SAP e, adicionalmente, fazer um mapeamento desses requisitos contra as capacidades do SAP.

Esta "fase" está subdividida em duas sub-fases descritas a seguir.

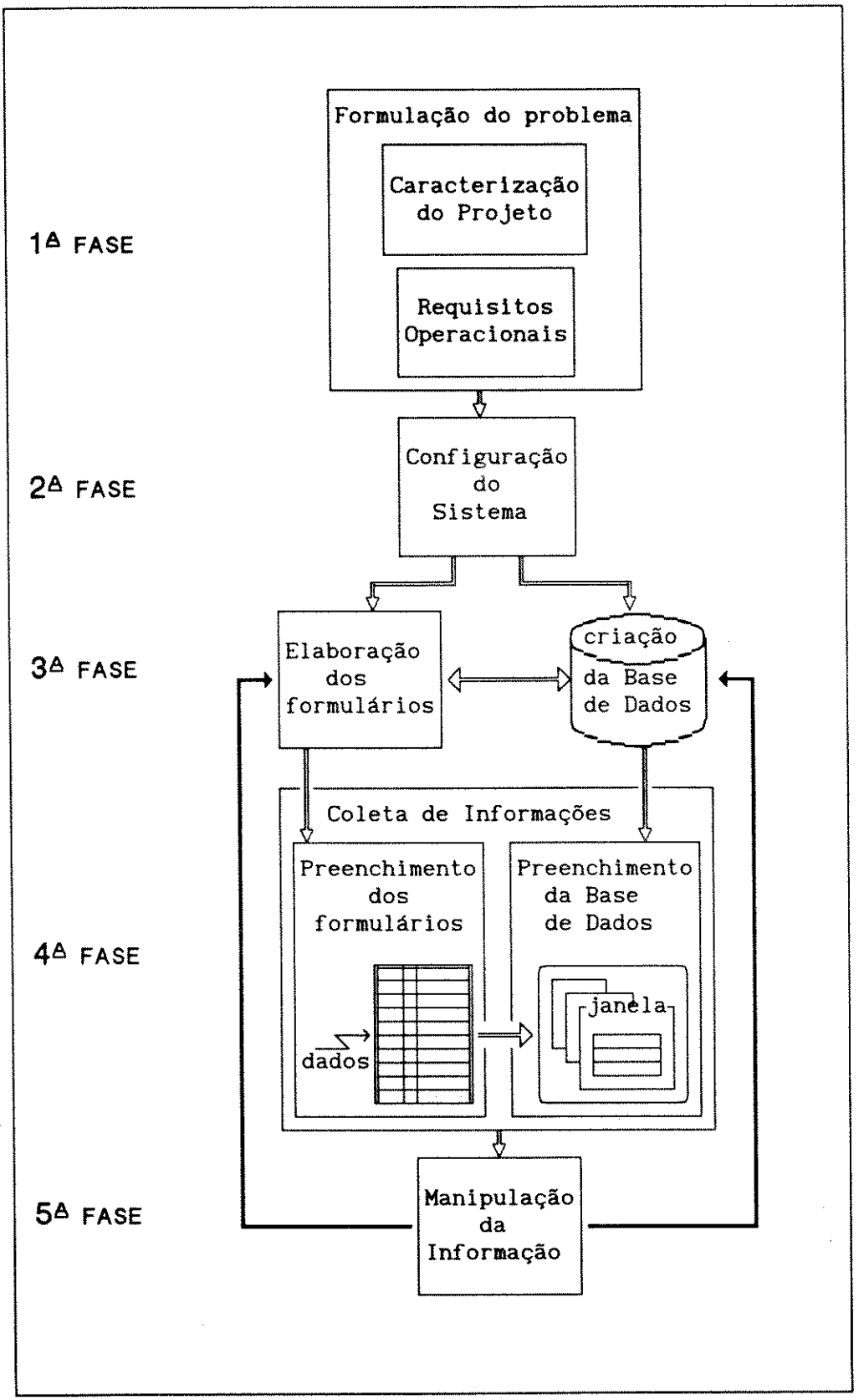



Figura 5.1 - Diagrama da metodologia proposta

### 5.2.1.1) Caracterização do Projeto

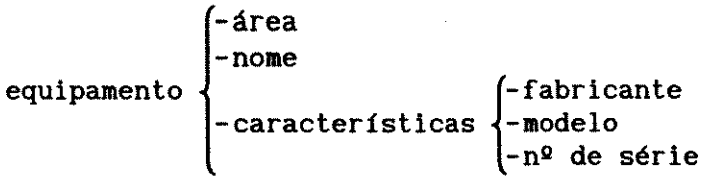
**Objetivo:** caracterizar o projeto a ser conduzido com o auxílio do SAP (tipo, características, estruturas de dados associadas, etc).

**Resultado:** definir as informações necessárias para a implementação do projeto.

O objetivo desta sub-fase é caracterizar o problema em questão, no sentido de levantar dados sobre o projeto alvo, tais como:

- identificação/classificação do tipo do projeto;
- levantamento das características do projeto;
- identificação e organização das estruturas de informação associadas ao projeto Para tanto, deve-se fazer um levantamento dos tipos e dos níveis das informações relevantes ao problema (por níveis, entenda-se a relação hierárquica entre as informações pois, em geral, é possível construir estruturas hierárquicas para descrever as relações entre os dados associados a um dado problema).

Suponha que tenhamos identificado, para um determinado projeto, uma estrutura de informações como a seguir:



Utilizaremos essa estrutura como exemplo para cada uma das fases subsequentes da metodologia que ora propomos.

#### 5.2.1.2) Requisitos Operacionais

**Objetivo:** estabelecer os requisitos de operação do sistema SAP para o projeto proposto.

**Resultado:** Definir a funcionalidade do SAP e, eventualmente, a necessidade de novas ferramentas que não se encontram presentes no sistema SAP padrão.

Nesta sub-fase, estabelecem-se os requisitos referentes ao modo como o SAP deve funcionar. Por exemplo: se o SAP vai acompanhar todo o ciclo de vida do projeto ou somente uma parte; requisitos de segurança de acesso e de integridade dos dados; requisitos de robustez e de tolerância a falhas; etc.

Eventualmente, alguns dos requisitos exigidos do SAP podem não estar implementados. Isto indica a necessidade de encontrar ferramentas já existentes que satisfaçam esses requisitos e incorporá-las ao sistema. Caso não existam ou não seja viável, por alguma razão, utilizar tais ferramentas, pode-se partir para a implementação da ferramenta.

### 5.2.2) 2ª fase - Configuração do Sistema

**Objetivo:** Selecionar o conjunto de módulos de programa que constituirá o SAP, de maneira a satisfazer os requisitos estabelecidos na fase anterior.

**Resultado:** Ao final desta fase, deveremos ter o SAP adequadamente configurado para acompanhar o projeto proposto.

### 5.2.3) 3ª fase

Nesta fase temos dois objetivos: a elaboração de formulários de pesquisa e a criação de uma Base de Dados para o sistema.

O elemento chave desta sub-fase são os formulários de pesquisa cuja finalidade é a de coletar as informações necessárias ao preenchimento dos arquivos da Base de Dados. Como tais informações, em geral, estão dispersas, é mais prático imprimir formulários de pesquisa, distribui-los às pessoas potencialmente detentoras dessas informações e, a seguir, transferir as informações colhidas para a Base de Dados.

Como existe uma ligação estreita entre os formulários e a Base de Dados, o projeto de ambos deve, de preferência, ser efetuado de forma paralela, de modo a evitar inconsistências que levem à necessidade de reprojetar os formulários e/ou a Base de Dados mais tarde.

#### 5.2.3.1) Elaboração dos formulários

**Objetivo:** Criar o "lay-out" de vários formulários de pesquisa para a obtenção das informações identificadas como "relevantes" na primeira fase.

**Resultado:** Um conjunto de formulários, os quais serão mais tarde distribuidos para preenchimento.

Esta primeira parte compreende a criação do "lay-out" dos formulários de pesquisa, os quais devem ter uma correspondência bi-unívoca com as estruturas dos arquivos da Base de Dados.

Assim, a cada arquivo da Base de Dados, cujo preenchimento dependa da coleta de informações dispersas, corresponderá um formulário de pesquisa que deverá ser preenchido pela(s) pessoa(s) apropriada(s).

Como, na maioria das vezes, o indivíduo (ou grupo de indivíduos) designado para preencher um dos formulário não intuiria o significado pretendido de cada um dos campos deste somente pelo nome e, desta maneira,

forneceria informações potencialmente incorretas, deve-se dar ao projeto do "lay-out" dos formulários uma atenção especial e, se possível, fornecer junto a cada formulário um pequeno guia para seu correto preenchimento. Tal guia deve apresentar uma descrição sucinta da finalidade do formulário e do significado de cada um dos seus campos.

É importante, ainda, que a linguagem e o jargão utilizados no formulário estejam o mais próximo possível do nível de conhecimento das pessoas que vão fornecer as informações.

O formulário a seguir é referente ao exemplo apresentado nos itens anteriores.

Observe, emoldurado no canto superior esquerdo, o nome do formulário. É aconselhável designar um nome a cada formulário para facilitar sua associação com o arquivo correspondente na Base de Dados e, assim, evitar confusões.

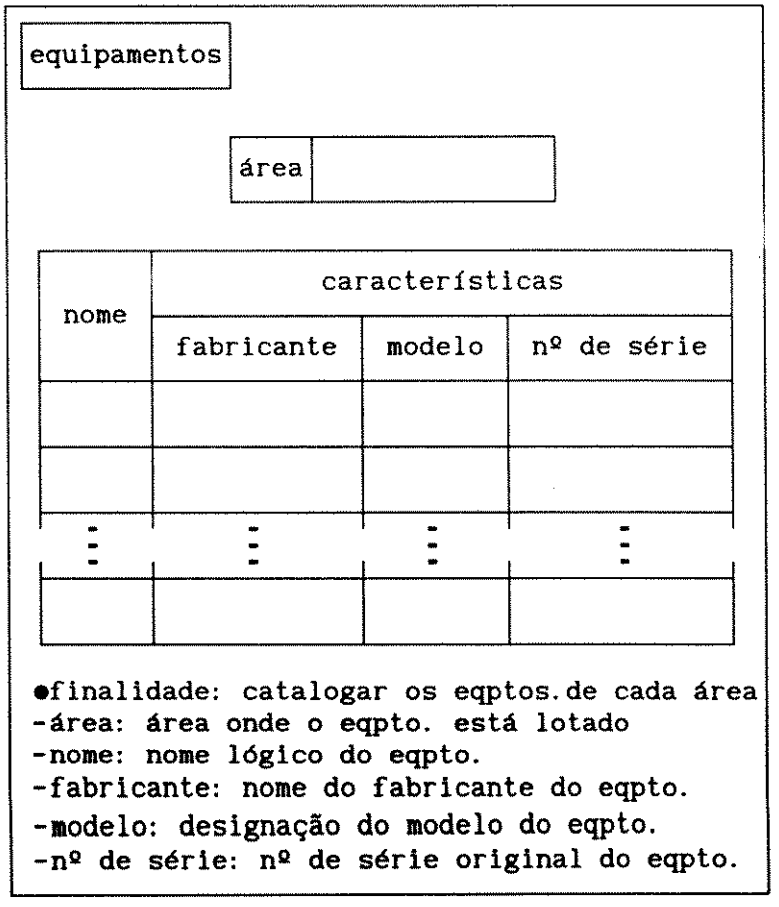



Figura 5.2 - Exemplo de formulário

Observe, ainda, que na parte inferior do formulário existe uma breve descrição de sua finalidade e do significado pretendido para cada um de seus campos.

#### 5.2.3.2) Criação da Base de Dados

**Objetivo:** Estabelecimento de uma Base de Dados para o SAP que reflita a estrutura dos formulários e, por conseguinte, as estruturas de dados do projeto, levantadas anteriormente.

**Resultado:** Uma Base de Dados central para o sistema, organizada em arquivos, que reflita a estrutura dos formulários de pesquisa.

Uma vez levantadas as estruturas de informação, e sua organização hierárquica, nas fases anteriores, deve-se, agora, criar uma Base de Dados composta por registros e arquivos correspondentes a essas estruturas de dados, para que se possa armazenar e, mais tarde, manipular as informações propriamente ditas.

Deve-se, ainda, escolher as estratégias de acesso para cada arquivo da base de dados e, se a estratégia exigir, definir um dos campos do registro associado como "campo chave" para busca e inserção. Deve-se, também, definir se será permitida a existência de mais de um registro com a mesma chave.

Para a estrutura de dados proposta no item anterior, podemos criar uma estrutura de registro de banco de dados como a seguir:

| área | nome | características | | |
|---|---|---|---|---|
| | | fabricante | modelo | nº de série |

Figura 5.3 - Exemplo de estrutura de registro

Criado o registro, o próximo passo é criar um arquivo na base de dados que utilize tal registro como base. Denominaremos tal arquivo EQUIPAMENTOS.

### 5.2.4) 4ª fase - Coleta de Informações

Esta fase também é logicamente divida em duas sub-fases distintas, uma dedicada à descrição da forma de preenchimento dos formulários de pesquisa, e a outra referente à transferência das informações coletadas através dos formulários para a Base de Dados do sistema.

#### 5.2.4.1) Preenchimento dos formulários

**Objetivo:** Obter, de pessoas chave, as informações necessárias ao preenchimento da Base de Dados do SAP, para posterior manipulação através das ferramentas agregadas ao sistema.

**Resultado:** Uma massa de formulários preenchidos, nem sempre de maneira correta, com informações que serão, a seguir, transferidas para a Base de Dados.

Antes da distribuição dos formulários, faz-se necessário um levantamento prévio das pessoas potencialmente detentoras das informações desejadas.

Após este levantamento, os formulários devem ser distribuidos às pessoas selecionadas para que elas os preencham.

Pode ser interessante que haja uma pessoa qualificada disponível para prover orientação a respeito das inevitáveis dúvidas.

A título de exemplo, observe os formulários correspondentes ao estudo de caso, os quais constam do anexo 8.

### 5.2.4.2) Preenchimento da Base de Dados

**Objetivo:** Transferência das informações, constantes dos formulários, para a Base de Dados do sistema.

**Resultado:** As informações coletadas através dos formulários estão, agora, disponíveis para as demais ferramentas do SAP.

A segunda parte desta fase da metodologia diz respeito à transferência das informações constantes dos formulários, agora já preenchidos, para a Base de Dados onde poderão, mais tarde, ser utilizadas pelos demais componentes do sistema.

É necessário mapear os formulários numa interface adequada para a transferência dos dados, deles constantes, para os arquivos correspondentes da Base de Dados.

A maneira mais prática é utilizar o sistema de janelas e menus, mencionado anteriormente, para facilitar o trabalho da pessoa encarregada da transcrição das informações dos formulários para a base de dados.

Para tanto, devem ser criadas "janelas" que permitam a entrada de dados e associar cada uma dessas janelas a um formulário e um arquivo correspondente da Base de Dados.

A figura a seguir representa a tela de um computador na qual está sendo apresentada uma janela de entrada de dados que permite a entrada de informações via teclado. Essa janela é correspondente ao exemplo utilizado em nossa metodologia.

Observe que a janela possui um título que corresponde ao formulário (e ao arquivo da Base de Dados) e o nome de cada campo precede a sua região de digitação.

Como a tarefa de digitação é de cunho meramente mecânico, dependendo da quantidade de dados a serem introduzidos, esse trabalho será entregue a um digitador que, tipicamente, não terá conhecimento da natureza do sistema, nem da estrutura da Base de Dados e nem da faixa e do tipo dos valores esperados em cada campo.

Portanto, é desejável que a janela permita edição, para efetuar correções de eventuais erros de digitação, e faça a verificação automática

dos tipos dos dados para evitar, por exemplo, que seja introduzida uma cadeia de caracteres em um campo onde se esperava um valor numérico devido, possivelmente, a uma falha do digitador.

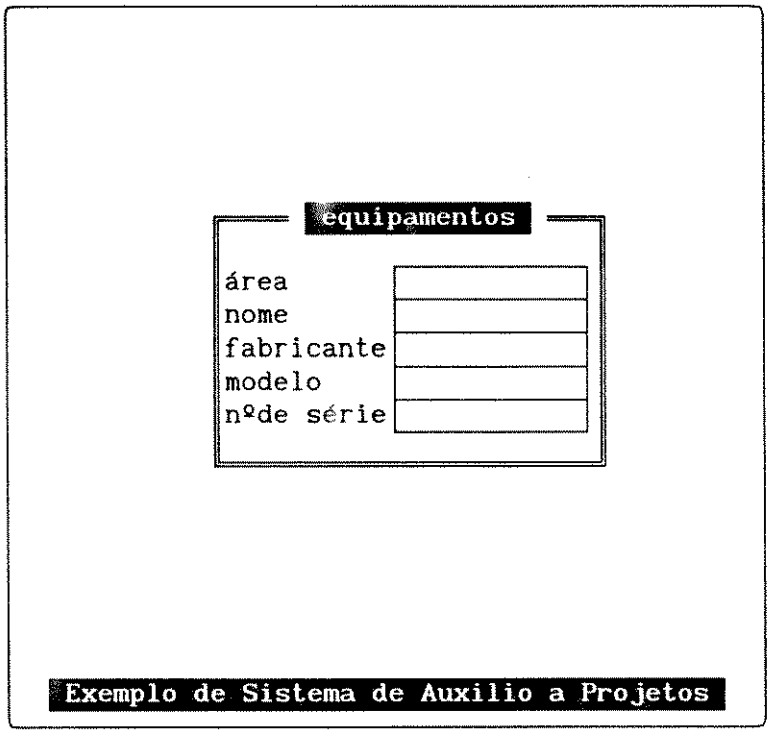



Figura 5.4 - Exemplo de janela para entrada de dados

### 5.2.5) 5ª fase - Manipulação da Informação

**Objetivo:** Utilização, por parte das "ferramentas" agregadas ao SAP, das informações armazenadas na Base de Dados do sistema, para cumprir o objetivo maior do sistema, que é o de auxiliar o desenvolvimento de um projeto.

**Resultado:** Dados referentes aos resultados da utilização das "ferramentas" do SAP, os quais deverão ser armazenados na Base de Dados do sistema.

Uma vez que já temos uma Base de Dados estruturada em arquivos e esses arquivos preenchidos com as informações obtidas por meio dos formulários de pesquisa podemos, agora, utilizar essas informações para alguma finalidade construtiva.

A manipulação das informações armazenadas na Base de Dados é feita por programas[1] os quais, por sua vez, podem também armazenar os resultados de seus processamentos na Base de Dados para posterior utilização por parte de outros programas.[1] Assim, é necessária a criação de arquivos na Base de Dados do sistema para guardar tais resultados. Veja figura a seguir.

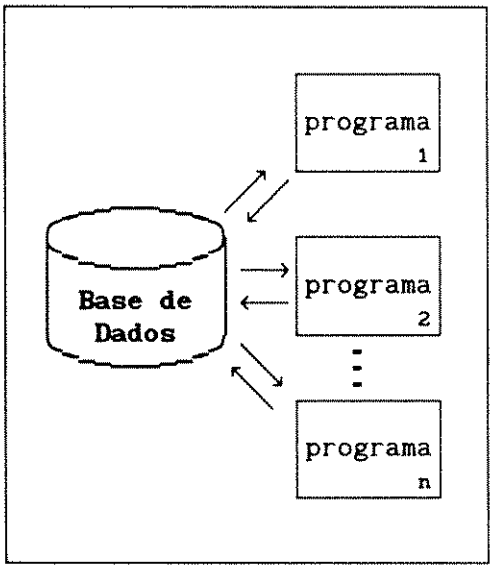



Figura 5.5 - Relação entre os programas e a base de dados

A descrição da funcionalidade de cada um desses programas de aplicação componentes do SAP é dependente do projeto e não faz parte, portanto, da metodologia em si.

O capítulo a seguir apresenta um estudo de caso da aplicação da metodologia, aqui proposta, na criação de um Sistema de Apoio a Projeto de Redes Locais de Computadores.

---

[1] ou **processos**, no caso de sistemas computacionais multi-tarefa

# CAPÍTULO 6

## Estudo de Caso

## 6.1) Introdução - *Redes Locais*

Redes Locais são uma forma de interconectar equipamentos - tais como impressoras, terminais, computadores, fac-símiles e outros - em uma rede, dentro de uma área geográfica limitada ( pequena ), de maneira que esses dispositivos se tornem acessíveis a múltiplos usuários. Isto vai permitir o compartilhamento de recursos o qual, por sua vez, leva a uma utilização mais eficiente de todos os equipamentos que compõem a rede [R-*Baker 86*]. Além disso, Redes Locais criam a possibilidade de implementação e utilização de aplicações que de outra forma não seriam viáveis, tais como: correio eletrônico, acesso concorrente a bases de dados, etc.

Entende-se, aqui, como pequena área geográfica distâncias entre 1 m e 10 Km, no máximo, separando os equipamentos que constituem a rede. Tais dimensões compreendem desde uma simples sala a um edifício inteiro ou, até mesmo, o campus de uma Universidade. Veja figura a seguir:

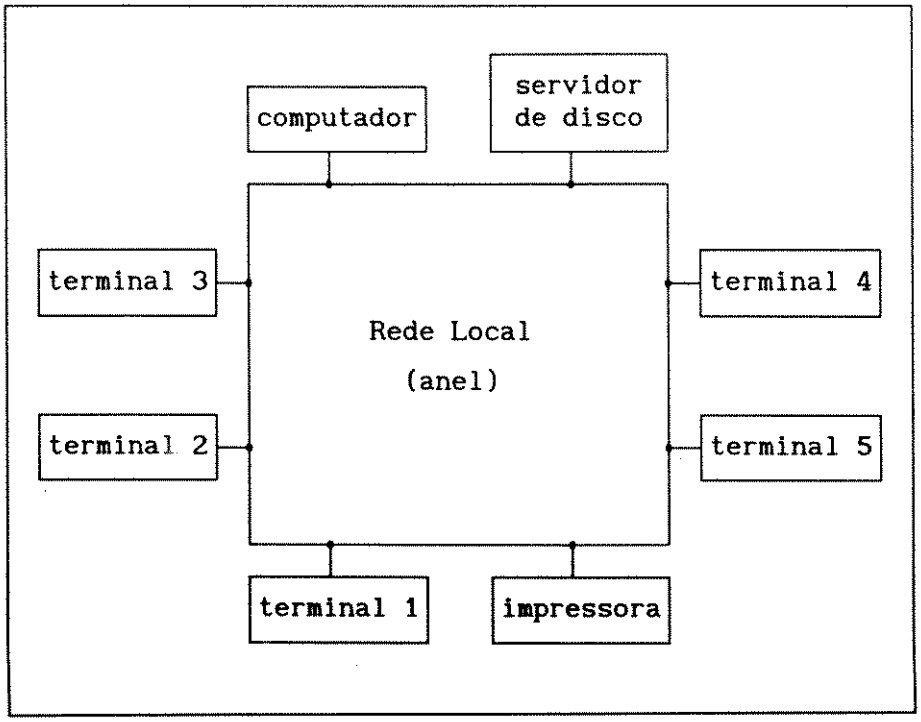



Figura 6.1 - Uma Rede Local típica

Geralmente Redes Locais são redes pequenas sob controle completo de uma única organização, ou empresa. Para possibilitar a comunicação com o mundo exterior, é comum serem providas de interfaces para redes externas, que podem ser outros tipos de redes ou, até mesmo, o sistema telefônico. Veja figura a seguir:

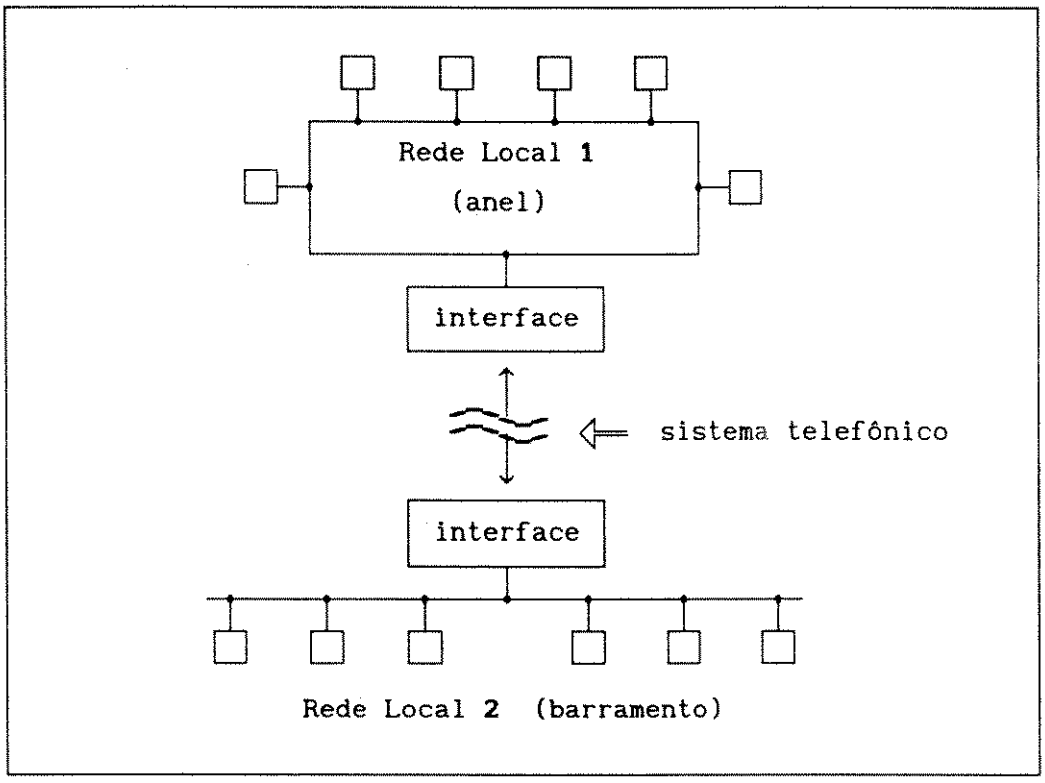



Figura 6.2 - Duas Redes Locais ligadas via sistema telefônico

### 6.1.1) Importância

Por que utilizar uma Rede Local em vez de fazer ligações ponto-a-ponto entre os equipamentos que necessitarem se comunicar? Vamos tentar responder essa pergunta através de um exemplo.

Examinemos a figura a seguir. Na parte *a)* da figura temos uma rede de conexões ponto-a-ponto (não se trata de uma Rede Local); um equipamento

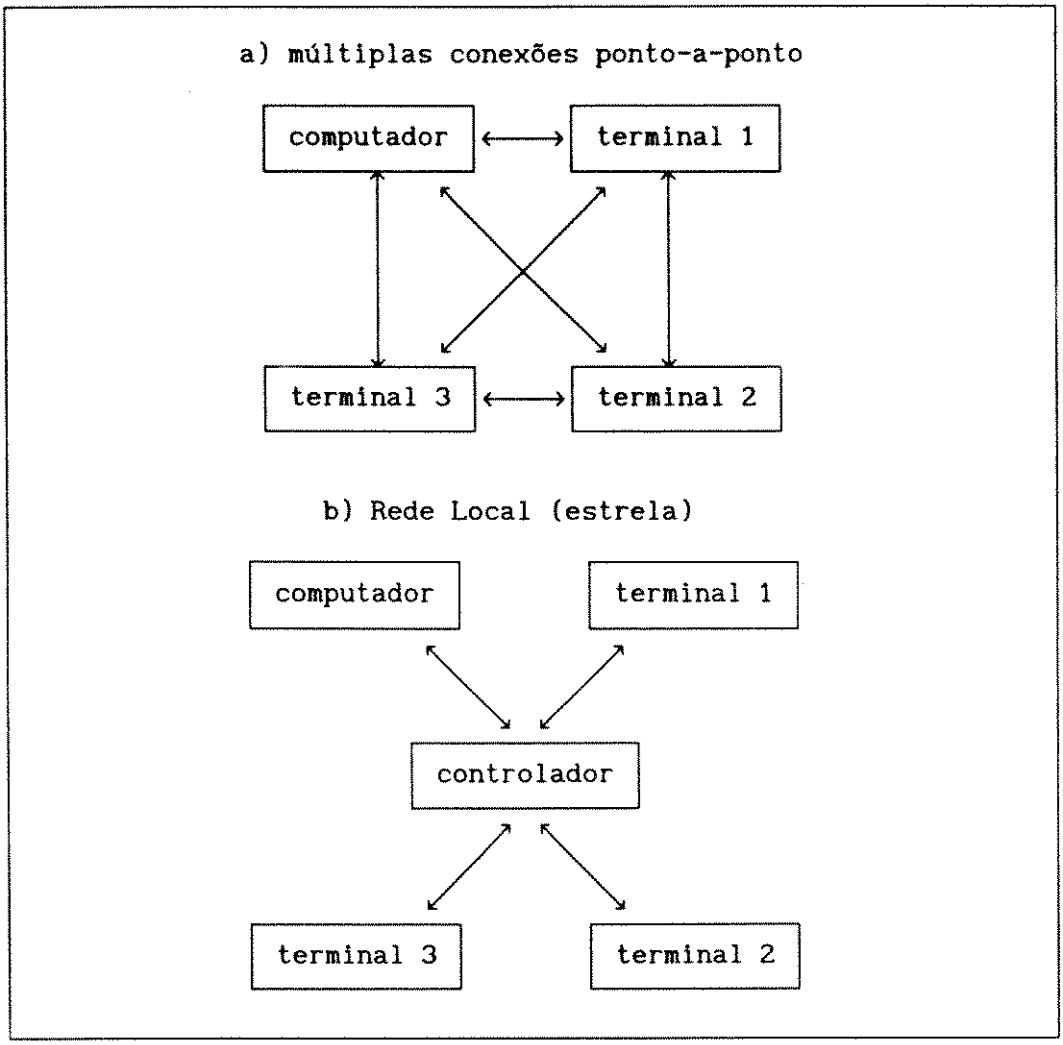



Figura 6.3 - Conexão ponto-a-ponto **X** Rede Local

requer,para se comunicar com qualquer dos outros, uma ligação dedicada. O número de conexões pode ser expresso por [R-*Baker 86*]:

$$C = C_n^k = \frac{n!}{(n-k)!\ k!} = \frac{n!}{(n-2)!\ 2!} = \frac{n(n-1)}{2}$$

onde: **C é o número de conexões necessárias**

**n é o número de nós (equipamentos)**

**k = 2 (as comunicações se dão entre pares de equipamentos)**

Note que a equação resultante é quadrática.

Já para a parte *b)* da figura temos representada uma Rede Local (tipo estrela); note que para este tipo de rede, o número de conexões é igual ao número de equipamentos, qualquer que seja este número, portanto uma equação linear.

Assim, comparando as duas equações percebemos que, para números pequenos de equipamentos, o número de conexões é parecido e, portanto, poderiamos utilizar o esquema de múltiplas conexões ponto-a-ponto. Mas considere um número razoavelmente elevado de equipamentos, por exemplo 100; no caso de usarmos uma Rede Local como a descrita, vamos necessitar, também, 100 conexões. Já no caso de conexões ponto-a-ponto seriam necessárias 4950 conexões o que na prática é inviável, tanto técnica quanto economicamente.

Porém, é necessário frisar que essa vantagem obtida com o uso de uma Rede Local não é conseguida sem sacrifícios, pois como não existem ligacões dedicadas entre cada par de equipamentos, deve-se compartilhar as existentes, o que leva a uma redução da velocidade de circulação da informação e, além disto, deve-se implementar uma disciplina de acesso ao meio para resolver conflitos, pois mais de um par de equipamentos pode querer estabelecer comunicação ao mesmo tempo.

### 6.1.2) Histórico

Redes de comunicação de longa distância, tais como a telefônica e a telegráfica existem desde há muito tempo (na verdade desde o princípio do século). A tecnologia por elas utilizada encontra-se em uma fase madura, no sentido de que as regras para projetar esse tipo de redes são bem estabelecidas e comprovadas. Existem, também, há um tempo considerável, algumas redes menores que poderiam ser classificadas como Redes Locais, tais como redes de comunicação embarcadas (no caso de navios), e redes de comunicação interna privativas em empresas. Esses tipos de reds, porém, não são nada mais que versões em escala menor da rede telefônica, não sendo apropriadas para tráfego de dados e não permitindo velocidades de transmissão elevadas.

Com o advento dos computadores, tornou-se necessário interligar as máquinas entre si e a dispositivos periféricos, tais como terminais e impressoras. A solução foi implantar redes de comunicação que englobassem

tais equipamentos. Tais redes eram, a princípio, proprietárias e dedicadas (à transmissão de dados), não permitindo integração de equipamentos de diferentes fabricantes, e nem tampouco adição de novos serviços (por exemplo: transmissão de voz).

Foi, porém, o advento da micro-informática que impulsionou de maneira decisiva a utilização das Redes Locais. Os computadores foram se tornando cada vez mais baratos e poderosos e as aplicações (programas) mais e mais complexas, grandes e caras. Além disso, alguns dispositivos periféricos que dependiam de mecanismos de precisão (a chamada *mecânica fina*), impressoras, por exemplo, não tiveram seus preços reduzidos na mesma proporção.

Com o passar do tempo, as bases de dados, pequenas a princípio, foram tomando proporções maiores e exigindo, portanto, capacidades maiores de armazenamento de massa; além do que, muitas vezes havia informações replicadas em diversas bases de dados.

Além disso, começaram a surgir novos conceitos na área de informática que exigiam novas técnicas para serem solucionados, tais como: Automação de Escritórios, Processamento Distribuido e Bases de Dados Distribuidas.

Todo esse panorama leva, intuitivamente, a imaginar uma maneira de compartilhar todos esses recursos das máquinas, programas e bases de dados.

Interligar todos esses equipamentos, em geral fisicamente próximos, em uma rede de tamanho limitado foi a solução encontrada. Nascia, assim, o conceito de Rede Local.

Isto ocorreu na década de 70 e, a partir daí, começou a aparecer uma miríade de tipos de Redes Locais, sendo que apenas algumas delas acabaram sendo implementadas comercialmente. A maioria não passou da fase de projeto piloto e muitas só foram estudadas teoricamente.

Para colocar ordem no caos de uma infinidade de tipos de Redes Locais, com características totalmente diversas, na maioria dos casos, iniciaram-se esforços de padronização das Redes Locais, de maneira a permitir, ao menos, a interconexão delas, independente do tipo e da concepção.

Atualmente os esforços nessa área se dirigem muito mais no sentido da padronização do que da criação de novos tipos de redes.

Assim, passada a euforia inicial da descoberta de um novo filão de pesquisa, a tecnologia de Redes Locais encontra-se, atualmente, em uma fase intermediária de desenvolvimento, sendo que a partir do momento em que os padrões propostos forem sendo mais aceitos e adotados, teremos passado à fase madura da tecnologia de Redes Locais.

### 6.1.3) Topologias

No contexto de Redes Locais, o termo topologia designa a maneira pela qual os equipamentos estão interconectados através da rede [R-*Zakir 88*].

Como já vimos anteriormente, em geral não é viável interligar cada equipamento a todos os outros da rede, assim deve-se adotar uma configuração que permita acesso de cada equipamento a todos os outros, porém com

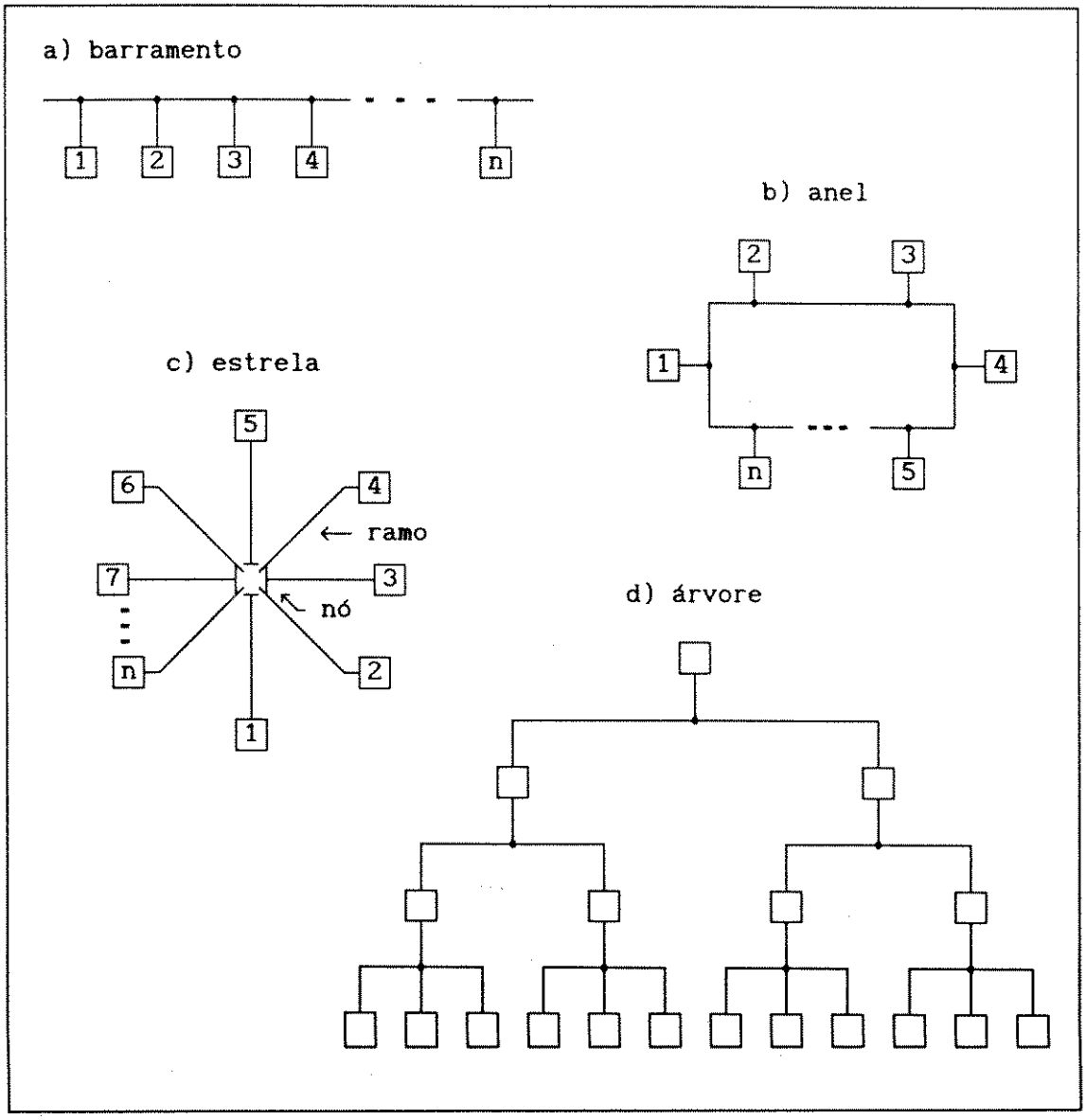



Figura 6.4 - Topologias de Redes Locais de computadores

compartilhamento de recursos.

São quatro as topologias básicas utilizadas em Redes Locais, (veja figura anterior) quais sejam: barramento, anel, estrela e árvore. Pode-se, porém, fazer combinações de vários tipos de topologias em um mesmo arranjo de rede.

Cada um dos quatro esquemas tem vantagens e desvantagens sobre os demais, conforme veremos a seguir.

#### 6.1.3.1) barramento

Neste tipo de topologia, a transmissão de dados se dá em uma "barramento", sendo que cada dispositivo da rede está a ele conectado através de um transceptor que não secciona o meio de transmissão utilizado. Veja figura anterior.

A transmissão de dados através do meio se dá de forma bidirecional. Assim deve-se ter um controle sobre essas transmissões para evitar que dois ou mais dispositivos transmitam dados ao mesmo tempo, o que ocasionaria um "embaralhamento" ou colisão dos dados tornando-os, assim, ininteligíveis.

A confiabilidade deste tipo de topologia é grande, pois uma falha em uma das estações de transmissão dificilmente afetará a rede como um todo. Porém, um problema no barramento pode ocasionar problemas bem mais sérios, pois no caso de uma quebra, por exemplo, as estações situadas em um dos lado do barramento partido não poderiam comunicar-se com as do outro lado, e vice-versa.

#### 6.1.3.2) anel

A característica principal deste tipo de topologia é que cada estação se liga a somente outras duas, sendo que uma é a sua predecessora e a outra a sua sucessora, e o conjunto forma um círculo, ou anel, fechado. Veja figura anterior.

As ligações entre estações são ponto-a-ponto e unidirecionais. Assim cada estação, ao receber uma transmissão deve ,se a mensagem não for diri-

gida a ela própria, retransmiti-la para a próxima estação, e assim por diante, até que a estação à qual a mensagem é endereçada retire-a da rede. Se nenhuma das estações retirar a mensagem, houve algum problema, e a estação que transmitiu a mensagem originalmente deve se encarregar de retirá-la da rede, pois caso contrário a mensagem circularia a rede indefinidamente.

As vantagens desta configuração são decorrentes do fato de cada estação ser repetidora podendo-se, assim, ter um número grande de estações e cobrir uma área geográfica considerável.

A grande desvantagem deste tipo de topologia é a confiabilidade, pois como cada estação é retransmissora, a ocorrência de uma falha em uma das estações paralisa a rede como um todo.

#### 6.1.3.3) estrela

Este tipo de topologia caracteriza-se pela presença de um dispositivo central de comutação de circuitos, ao qual todas as estações devem estar conectadas através de ligações ponto-a-ponto. Veja figura anterior.

Se uma estação quer comunicar-se com outra estação ou estações, ela deve efetuar um pedido de ligação para a central de comutação que se encarregará, então, de efetuar a ligação (ou ligações) desejadas e, ao fim da comunicação, de desfazer as ligações.

As vantagens deste tipo de esquema são que o controle, por parte de cada estação, é muito simples, pois o comutador se encarrega do acompanhamento da conexão durante a vida útil desta. Além disso, a tecnologia de comutador central já é bem conhecida, por ser semelhante à dos P(A)BX.

As desvantagens são a complexidade crescente do comutador central com o número de estações a serem interligadas e, também, a dificuldade para fazer a interconexão de uma estação a grupos de outras estações (*multicasting*).

Mas, a desvantagem principal diz respeito à confiabilidade, pois o sistema é altamente vulnerável a defeitos no comutador central, sem o qual a rede simplesmente não funciona.

### 6.1.3.4) árvore

Este tipo de topologia é resultado da interligação, de maneira hierarquizada, de diversas sub-redes (em geral barramentos) em uma rede maior. Veja figura anterior.

Cada sub-rede pode ter o tipo de meio físico e disciplina de acesso diferentes das demais. Assim, fazem-se necessários equipamentos para interfacear cada sub-rede com as redes imediatamente superior e inferior. Tais equipamentos podem ser "pontes" (bridges), "roteadores" (routers) ou, ainda, "comportas" (gateways), de acordo com o nível de dificuldade no interfaceamento entre as sub-redes. Veja figura a seguir.

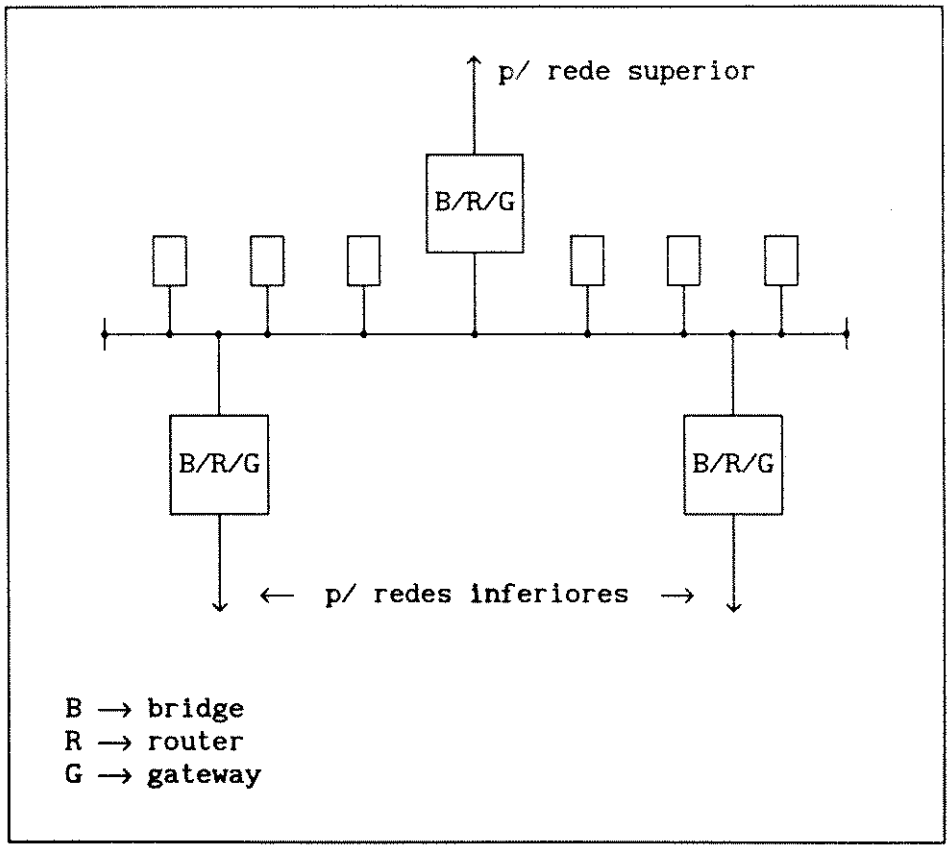



Figura 6.5 - Diagrama de uma sub-rede genérica

As vantagens deste tipo de arquitetura decorrem da divisão da rede global em um número de segmentos de rede autônomos; desta maneira, pode-se situar em uma mesma sub-rede equipamentos que se comuniquem com grande frequência, assim o tráfego por eles gerado permanecerá confinado à sub-rede, não onerando as comunicações externas, preservando-se a possibilidade de comunicações com estações situadas em outras sub-redes, ainda que esporádicas.

A confiabilidade é elevada, pois problemas em uma estação, ou mesmo em uma sub-rede, têm uma probabilidade muito pequena de afetar a rede como um todo.

Pode-se, ainda, criar sub-redes que atendam requisitos de comunicação críticos, como por exemplo: taxas de erros extremamente baixas, ou tempos de resposta garantidos ( tempo-real ). Assim, tem-se a possibilidade de "sintonizar" cada sub-rede às exigências dos equipamentos ou das aplicações comunicantes nela situados.

Este é o tipo de arquitetura geralmente utilizado em aplicações de automação de manufatura com um número razoavelmente elevado de equipamentos, por ser, entre outras coisas, extremamente modular, flexível e confiável.

### 6.1.4) Meios de Transmissão

No jargão de Redes Locais de Computadores, utiliza-se o termo meio de transmissão para denominar o caminho físico que promove a interligação das diversas estações que compõem a rede [R-Zakir *88*].

Dentre todos os meios de transmissão existentes, apenas alguns são utilizados efetivamente na implementação de Redes Locais. A seguir, uma descrição deles.

#### 6.1.4.1) par trançado

Permite a transmissão tanto de dados analógicos como digitais, sendo utilizado, geralmente, em conexões ponto-a-ponto.

A principal vantagem do par trançado é o custo por metro, que é o mais

baixo de todos os meios de transmissão.

As desvantagens são: baixas taxas de transmissão alcançáveis e as baixas imunidades a ruído e a escuta clandestina.

Em geral, não é o meio de transmissão principal, mas é utilizado, isto sim, para fazer a conexão de uma estação ao meio de transmissão principal.

#### 6.1.4.2) cabo coaxial

É o meio de transmissão mais comumente utilizado em Redes Locais, por sua grande versatilidade e custo razoável.

Em Redes Locais, são utilizados dois tipos de cabo coaxial: um de 50 Ω, também chamado de "baseband", e outro de 75 Ω conhecido como "broadband".

O primeiro é utilizado, geralmente, para transmissão de sinais digitais codificados sem modulação (ou seja: banda-base), de forma que o sinal ocupa todo o espectro de frequência disponível. A taxa máxima de transmissão está na casa de 10 Mbps e a distância máxima entre estações em torno de 1 a 3 Km.

O Segundo, por ter uma faixa de frequência útil muito grande, utiliza-se para a transmissão de sinais analógicos modulados. Desta forma pode-se ter no mesmo cabo vários canais distintos para o tráfego de dados.

A taxa de transmissão pode alcançar centenas de Mbps e o alcance de uma rede está limitado, por razões práticas, na casa de 10 a 50 Km no máximo.

O tipo de conexão mais utilizado, em ambos os casos, é o multiponto.

#### 6.1.4.3) fibra ótica

No estágio atual da tecnologia de fibras óticas, as conexões entre estações devem ser ponto-a-ponto e unidirecionais. Os sinais são digitais, e transmitidos sem modulação (em banda-base).

A grande vantagem da fibra ótica é a sua virtual imunidade a ruídos e interferências eletromagnéticas bem como à água. Aliado a isto, a velocidade de transmissão pode ser da ordem de centenas de Mbps (experimentalmente até de alguns Gbps) e a faixa útil de frequências é muito grande.

As principais desvantagens são o alto custo por metro da fibra em si e

a pequena disponibilidade e o preço elevado dos equipamentos associados à fibra, tais como: transceptores, repetidores, etc.

### 6.1.5) Disciplinas de acesso ao meio

Como vimos anteriormente, várias estações estarão conectadas a um meio físico comum. Para que a comunicação entre estações seja possível, somente uma estação pode estar transmitindo em uma certa frequência num dado instante, caso contrário haverá **colisão** entre as mensagens, o que as tornará ininteligíveis. Para contornar esse problema, deve haver uma ordem pré-definida para o controle do acesso ao meio. A essa ordem chamamos Disciplina de Acesso ao meio [R-*Giozza 86b*].

Existem inúmeras técnicas de disciplina de acesso ao meio; algumas são do tipo centralizado, ou seja, deve existir uma estação "mestre", e as demais serão suas "escravas". Este esquema é particularmente aplicável à topologia tipo estrela, que também é centralizada.

Para os outros tipos de topologias (descentralizadas) usa-se, em geral, esquemas descentralizados de controle de acesso ao meio.

As disciplinas de acesso ao meio mais utilizadas em Redes Locais são: CSMA e Token Passing, as quais passamos a descrever a seguir.

#### 6.1.5.1) CSMA e CSMA/CD

CSMA é uma sigla em inglês que significa Carrier Sense Multiple Access, ou em português, Acesso Múltiplo com Detecção de Portadora.

Para evitar colisão, uma estação que deseje transmitir deve, antes de iniciar a tranmissão, "escutar" o meio de transmissão para verificar se existe uma mensagem trafegando no meio e só iniciar a transmissão se não detectar a presença da frequência de onda portadora (que indica mensagem em trânsito). Caso o meio de transmissão esteja "ocupado", a estação deve aguardar um espaço de tempo aleatório antes de tentar retransmitir.

Pode, porém, ocorrer colisão se, por exemplo, duas estações tentarem iniciar uma transmissão ao mesmo tempo, ambas detectarem meio de transmissão

livre e iniciarem suas transmissões simultaneamente.

Em caso de colisão, as estações devem esperar um número aleatório de intervalos de tempo e então tentar retransmitir.

Neste esquema, as estações só perceberão que houve colisão quando, ao final da transmissão, não receberem a confirmação da mensagem transmitida.

A maneira de evitar este desperdício de tempo é monitorar o meio de transmissão também durante o curso da transmissão e abortar a transmissão caso seja detectada colisão de dados. Este novo esquema chama-se CSMA/CD, onde CD significa Colision Detection ou, em português: Detecção de Colisão.

CSMA/CD é um tipo de Disciplina de Acesso ao Meio largamente utilizado em Redes Locais atualmente, enquanto que CSMA puro só é descrito por motivos didáticos.

As principais desvantagens do esquema CSMA são duas; a primeira é que não existe garantia de tempo máximo para a entrega de uma mensagem, e a segunda é que o desempenho da rede piora sensivelmente com o aumento da carga.

As vantagens são que falhas em estações dificilmente afetam a rede como um todo e que o processo de remoção de estações existentes e adição de novas estações é simples e transparente às demais estações.

#### 6.1.5.2) Token Passing

Esta disciplina de acesso ao meio baseia-se na existência de um "token" (em português: ficha ou bastão) que deve circular em um "anel lógico" formado pelas estações. Uma estação só pode transmitir quando tiver a posse do token. Se não tiver dados a transmitir, ou então ao final da sua transmissão de dados, a estação passa o token à estação seguinte do anel lógico.

As vantagens desse esquema são a possibilidade de antecipar o tempo máximo necessário para a transmissão de uma mensagem (o que é potencialmente importante para aplicações em tempo-real), e a eficiência sob cargas elevadas.

As desvantagens são a complexidade, o fato de que em baixa carga perde-se muito tempo simplesmente circulando o token e, além disto, a necessidade de operações "especiais", que podem exigir a re-inicialização da

rede, para a inserção e retirada de estações do anel lógico e para a reposição do token que pode, eventualmente, perder-se.

### 6.1.6) Padrões

Como já foi mencionado, o objetivo maior de uma Rede Local é a interligação de vários equipamentos em uma rede, de modo a permitir a comunicação e o processamento cooperativo entre eles.

Se todos os equipamentos e a própria rede pertencerem ao mesmo fabricante, não deveremos ter muitos problemas em interconectá-los, pois provavelmente os equipamentos serão compatíveis.

Suponha, agora, que desejemos uma rede "genérica", no sentido de que os equipamentos provenham de diversos fabricantes e a rede, em si, seja fornecida por outra empresa qualquer. Neste caso é necessário adotar regras que cada estação da rede deve seguir para permitir a comunicação com as demais. O conjunto dessas regras compõe aquilo que chamamos padrão.

Existem vários padrões para Redes Locais que são aceitos internacionalmente. A seguir alguns deles.

#### 6.1.6.1) O padrão RM-OSI/ISO

O RM-OSI (Reference Model - Open Systems Interconection) é um padrão para redes elaborado pela ISO (International Standard Organization) que baseia-se numa hierarquia de sete níveis (módulos) de protocolo.

Este modelo foi preconizado, originalmente, para Redes de Longo Alcance, sendo, no mínimo, discutível sua utilização "as is" em Redes Locais (por causa do "overhead", por exemplo). No entanto, este padrão serve de base para os demais padrões de Redes Locais.

Os sete níveis de protocolo são divididos em dois grupos: níveis inferiores (1 a 3) e níveis superiores (4 a 7). Veja a figura 6.6.

A figura 6.7 apresenta uma descrição sucinta das funções desempenhadas por cada nível (ou camada) do padrão RM-OSI / ISO:

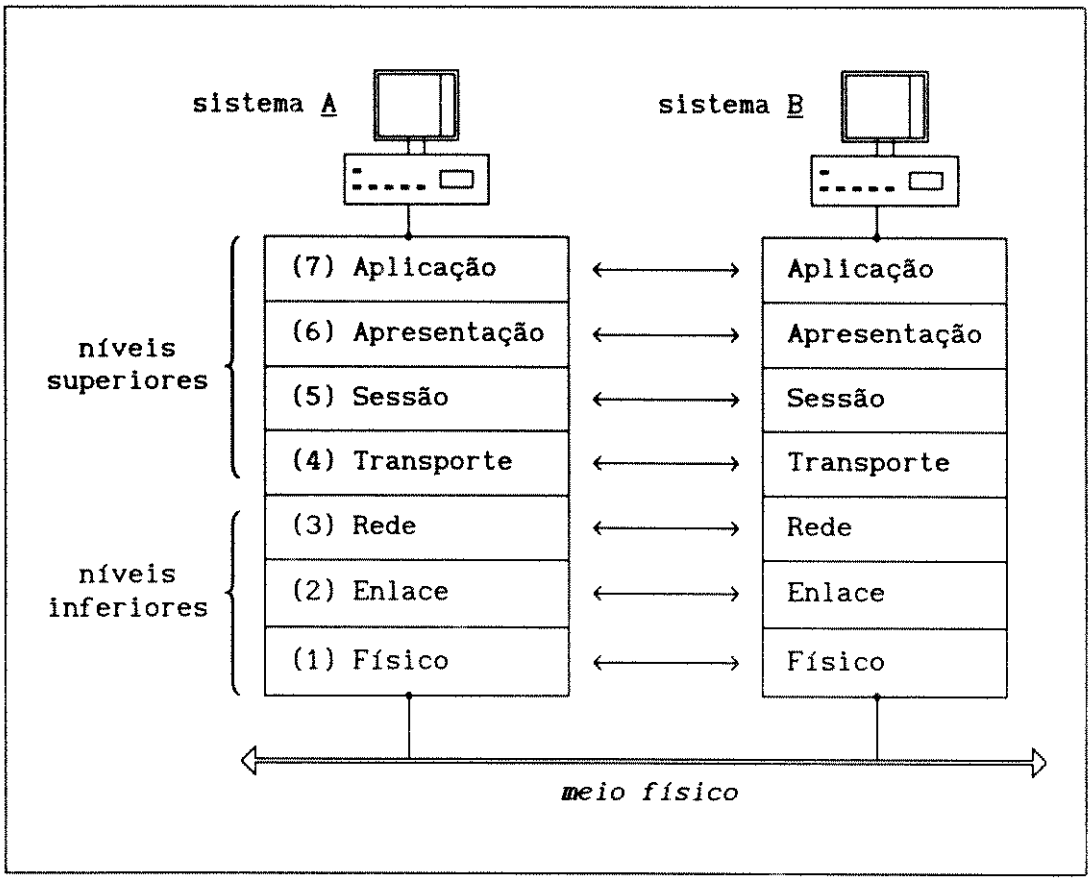



Figura 6.6 - O padrão RM-OSI/ISO

| nível | | descrição |
|---|---|---|
| nº | nome | |
| 1 | Físico | proc. mecânicos, eletromagnéticos e procedurais p/ estabelecer, manter e desativar a ligação física |
| 2 | Enlace | transmissão de quadros ao invés de bits; provê detecção de erros |
| 3 | Rede | roteamento e controle de fluxo; transmissão de pacotes (com ou sem conexão virtual) |
| 4 | Transporte | transferência confiável de dados entre processos com transparência e recuperação de erros |
| 5 | Sessão | estabelece, gerencia, mantém e termina conexões entre dois processos |
| 6 | Apresentação | transformações de dados (criptografia, compressão reformatação, etc) |
| 7 | Aplicação | serviços úteis às aplicações (transferência de arquivos, acesso a bancos de dados, etc) |

Figura 6.7 - As camadas do modelo RM-OSI/ISO

### 6.1.6.2) O padrão IEEE 802

O padrão IEEE 802 para Redes Locais constitui-se, na verdade, em uma família de padrões relacionados a alguns tipos de meios físicos e métodos de controle de acesso associados.

Este padrão segue os moldes da arquitetura em camadas do RM-OSI/ISO, porém concentra-se, de maneira especial, no equivalente às duas camadas mais baixas do modelo RM-OSI, ou seja: Física e Enlace de Dados. Veja a figura a seguir [*R-Gibson 91*].

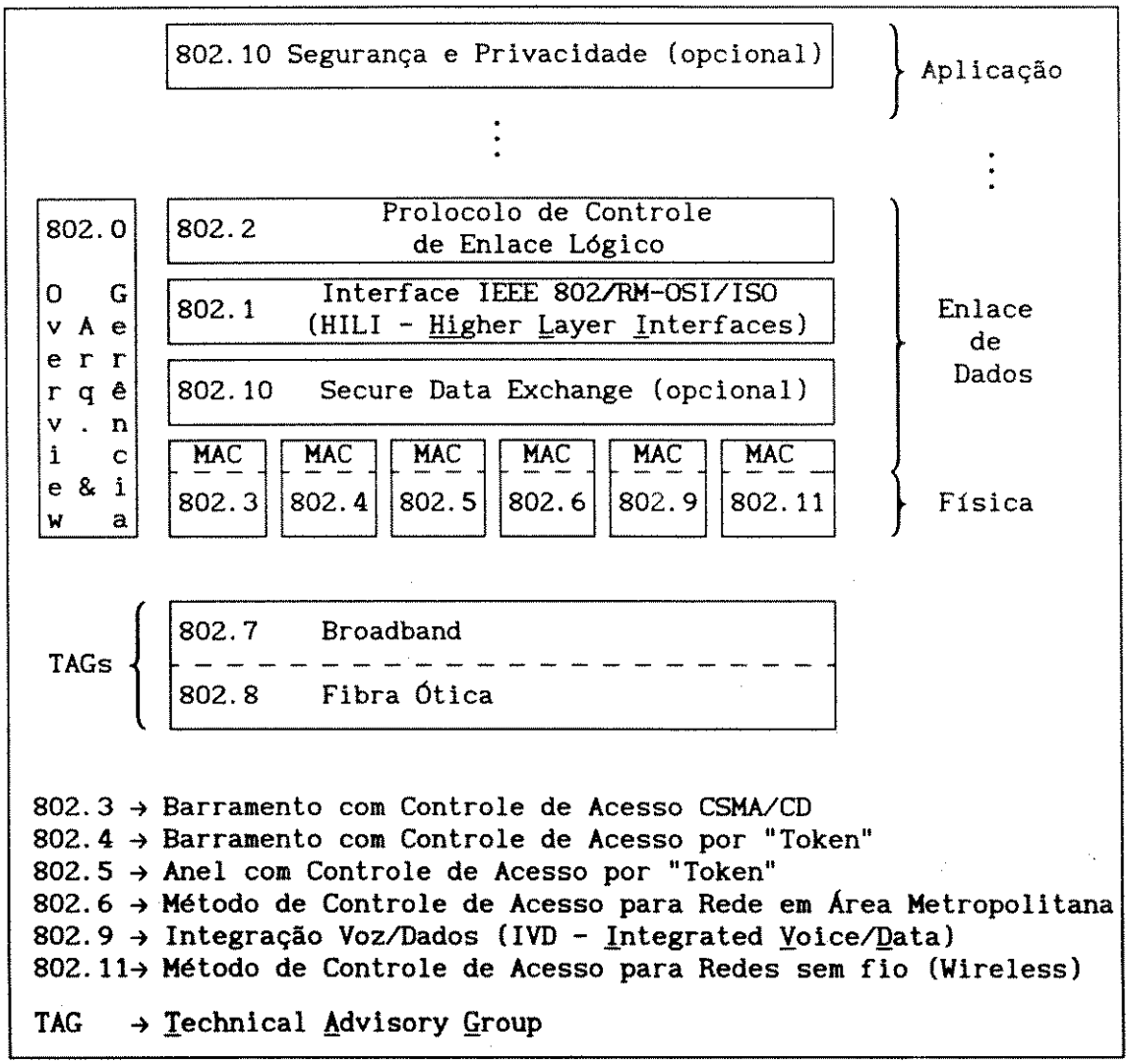



Figura 6.8 - O padrão IEEE 802 e sua relação com o RM-OSI/ISO

Utiliza-se largamente este padrão na implementação de Redes Locais, pois é melhor adaptado ao escopo de redes de tipo local e, além disto, abrange vários tipos de meios de transmissão e disciplinas de acesso, sendo, por isto, bastante abrangente.

### 6.1.6.3) O padrão MAP/TOP

MAP (Manufacturing Automation Protocol) ou, em português, Protocolo para Automação de Manufatura, é uma iniciativa da General Motors visando a integração dos dispositivos existentes em uma fábrica, ou seja, criar uma solução CIM (Computer Integrated Manufacturing ou Manufatura Integrada por Computador, em português).

<table>
<tr><th colspan="2">camada do RM-OSI</th><th>MAP (3.0)</th><th>TOP (1.0)</th></tr>
<tr><td>7</td><td>Aplicação</td><td>ISO CASE*<br>ISO FTAM*<br>MMS (RS-511)<br>MAP directory<br>MAP network management</td><td>ISO CASE*<br>ISO FTAM*<br>X.400 (CCITT)</td></tr>
<tr><td>6</td><td>Apresentação</td><td colspan="2">vazia</td></tr>
<tr><td>5</td><td>Sessão</td><td colspan="2">ISO sessão (kernel)</td></tr>
<tr><td>4</td><td>Transporte</td><td colspan="2">ISO transporte (classe 4)</td></tr>
<tr><td>3</td><td>Rede</td><td colspan="2">ISO CLNS*</td></tr>
<tr><td>2</td><td>Enlace</td><td colspan="2">IEEE 802.2 LLC<br>tipo-1, classe-1 (connectionless)</td></tr>
<tr><td>1</td><td>Física</td><td>IEEE 802.4 token bus<br>IEEE 802.4 broadband</td><td>IEEE 802.3 CSMA/CD<br>IEEE 802.3 baseband</td></tr>
</table>

- * → subset
- CASE → Common Applications Service Element
- FTAM → File Transfer Access and Management
- MMS → Manufacturing Message Service (protocolo RS-511)
- X.400→ especificação de protocolo fornecido pelo CCITT
- CCITT→ Comitê Consultif International de Téléphonie et Télégraphie
- CLNS → Connectionless Network Service

Figura 6.9 - Diagrama da correspondência dos padrões RM-OSI ⟺ MAP/TOP

TOP (Technical and Office Protocol) ou Protocolo para Escritórios e (área) Técnica, é uma iniciativa análoga da Boeing visando a automação de escritórios (áreas técnica e administrativa).

Estes padrões são, na verdade, apenas uma junção partes de outros padrões já existentes, tais como o RM-OSI/ISO e o IEEE 802.x (veja figura 6.9 [R-*Gomide 87*]), porém são extremamente versáteis para as aplicações a que se destinam.

Sistemas MAP e TOP são facilmente interconectáveis, pois baseiam-se, praticamente, no mesmo conjunto de protocolos.

Geralmente um sistema MAP consiste de um tronco principal ou "backbone" (espinha dorsal), onde se situam aplicações de gerência, planejamento, transmissão de vídeo e voz, etc; e de um número de sub-redes mais simples que podem ter características de resposta em tempo-real. Nessas sub-redes, utiliza-se, subconjuntos da especificação MAP completa (Full MAP), pois são necessárias velocidades de acesso maiores, e/ou custos menores. As arquiteturas MAP-EPA (Enhanced Performance Architecture) e Mini-MAP podem cumprir esses requisitos. Veja figura a seguir [R-*Gomide 87*].

<table>
<tr><td colspan="2">camada do RM-OSI</td><td>MAP - EPA</td></tr>
<tr><td>7</td><td>Aplicação</td><td>MMS (RS-511)</td></tr>
<tr><td>6</td><td>Apresentação</td><td rowspan="4">NULOS</td></tr>
<tr><td>5</td><td>Sessão</td></tr>
<tr><td>4</td><td>Transporte</td></tr>
<tr><td>3</td><td>Rede</td></tr>
<tr><td>2</td><td>Enlace</td><td>mensagens de 16 a 20 bytes c/ serviços de resposta rápida p/aplicações críticas</td></tr>
<tr><td>1</td><td>Física</td><td>barramento c/passagem de ficha, 5 Mbps tempo de acesso menor que 50 ms</td></tr>
</table>

Figura 6.10 - Diagrama da arquitetura do padrão MAP-EPA

### 6.1.7) Perspectivas

A adoção de padrões tornou viável a interligação, em rede, de equipamentos de fabricantes diferentes, e a tendência é a consolidação dos padrões já existentes.

No escopo de redes de longa distância, o padrão RM-OSI/ISO parece ser a tendência principal, servindo de referência, ainda, para os demais padrões de redes, especialmente as Redes Locais.

Para Redes Locais genéricas, o padrão IEEE 802 é muito bem aceito, e parece improvável que outros padrões o suplantem, pelo menos a curto prazo.

Parece inevitável a adoção de MAP e TOP como padrões para a automação de manufatura, porém estes padrões estão ainda em evolução.

Assim, padronização é a tendência maior que pode ser observada atualmente no campo de Redes Locais.

## 6.2) **NETCAD** - *um estudo de caso*

### 6.2.1) Introdução

Vamos colocar a seguinte questão: Dado que temos uma empresa com um certo número de equipamentos que permitem ou requerem interligação, e dado ainda que queremos adotar um dos padrões de Redes Locais disponíveis, por exemplo MAP, desejamos uma ferramenta de programação que nos auxilie no processo de implementação ordenada de uma arquitetura hierárquica de rede que possibilite a interligação de todos esses equipamentos, respeitando as características e requisitos de serviço de cada um. Essa rede deve ser livre de "gargalos" de comunicação, confiável, modular e expansível.

O desenvolvimento de tal Sistema de Apoio ao Projeto de Redes Locais, ao qual daremos a designação de **NETCAD**, constitui um estudo de caso da utilização da nossa proposta de metodologia de desenvolvimento de Sistemas de Apoio a Projetos, metodologia esta que é o objeto maior desta Tese.

#### 6.2.1.1) Importância e justificativa do estudo

Atualmente, do ponto de vista de recursos computacionais, uma empresa é composta de um número de equipamentos, de portes diferentes e fabricantes diversos, sendo que, geralmente, existe comunicação apenas entre alguns deles, na base de um para um (p.ex.: entre um computador e uma impressora).

Ao identificar a necessidade de implementação de facilidades de comunicação entre esses diversos equipamentos parte-se, em geral, para a implementação de soluções locais para cada área onde tal necessidade se faça presente. Por exemplo, se em determinado departamento de uma empresa existe uma impressora laser (um recurso escasso), é comum ela estar acoplada a um determinado (micro)computador ou, no máximo, existe um computador atuando como servidor de impressão associado à impressora, e conexões a um número limitado de outros computadores que necessitem desse recurso, formando uma rede "dedicada".

Tal configuração funciona bem para um número extremamente limitado de aplicações, pois se for necessária a extensão do serviço a um número maior de equipamentos ou, por outro lado, tornar-se necessário que os equipamentos suportem um outro tipo de aplicação concomitantemente, esse esquema torna-se impraticável, sendo necessária a adoção de uma arquitetura mais flexível.

Uma das possibilidades seria interligar todos os equipamentos da empresa em uma única rede, o que pode ser viável se o número de equipamentos que necessitem facilidades de comunicação não for grande, porém se esse número for razoável, advirão problemas de tráfego às mensagens geradas pelos equipamentos, tornando os tempos médios de resposta possivelmente inaceitáveis para algumas das aplicações. Além disso, geralmente um equipamento comunica-se muito com um número pequeno de outros equipamentos, sendo que comunicação para equipamentos fora desse pequeno grupo é necessária apenas esporadicamente.

Para solucionar todos esse problemas, o ideal seria formar pequenas redes locais entre os equipamentos que se comuniquem com mais frequência ou estejam fisicamente mais próximos ou, ainda, requeiram características especiais da rede (p.ex.: resposta em tempo-real). Cada uma dessas sub-redes seria, então, ligada a redes maiores que abrangeriam, por exemplo, uma determinada área da empresa, e assim por diante, até formar-se uma rede que englobe a empresa como um todo. Dessa maneira cada equipamento teria acesso

a todos os outros, mesmo que de maneira indireta; seções com tráfego intenso entre seus componentes não causariam problemas às demais seções da rede, além disso, desta forma, é possível "isolar" seções onde se façam necessárias características especiais, tais como tempos de acesso garantidos.

Como a tarefa de implementar uma rede desse porte é complexa e cara, geralmente é elaborado um plano de implementação dividido em fases, com as primeiras fases priorizando a implementação de sub-redes onde elas forem mais urgentes, e as fases finais tendo como finalidade a interconexão das diversas sub-redes formadas nas fases anteriores.

Deve-se levar em consideração muitos fatores para que o resultado final seja satisfatório, tais como: padronização, expansibilidade da rede, confiabilidade, interoperabilidade, e outros.

#### 6.2.1.2) Objetivos do sistema

Tendo em vista o exposto na seção anterior, seria altamente desejável a existência de uma metodologia de projeto para uma rede desse porte, que garantisse, ou pelo menos favorecesse, um resultado final satisfatório, ou seja uma rede hierarquizada que cobrisse toda a empresa, sem gargalos de comunicação, confiável, modular, expansível, bem documentada, de fácil manutenção e a um baixo custo.

A metodologia NETCAD é uma proposta nesse sentido. Vamos basear a proposta de rede a ser implementada em uma empresa, nas informações recolhidas junto a cada área ou setor da empresa em questão, através de formulários de pesquisa apropriados à identificação das características de comunicação dos equipamentos da empresa a serem conectados à rede que será proposta como uma possível solução para o problema de comunicação da empresa.

O sistema proposto será modular, comportando a adição de ferramentas de apoio ao projeto da rede, tais como ferramentas de análise de fluxo de dados, simulações, análise de custos, e assim por diante.

Utilizaremos uma base de dados para armazenar as informações provenientes dos formulários de pesquisa e, em adição, alguns outros parâmetros a serem calculados ou arbitrados em cada uma das fases (ou módulos) do sistema. Veja figura a seguir.

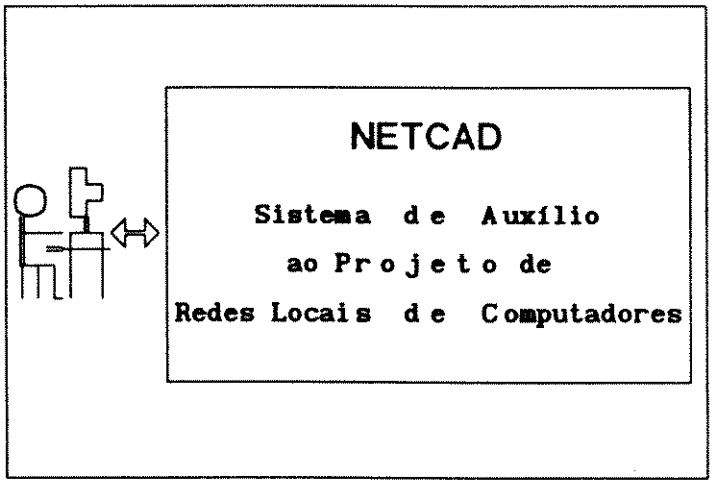



Figura 6.11 - O sistema **NETCAD**

### 6.2.1.3) Amplitude e delimitação do estudo

A metodologia NETCAD abrange somente tópicos relacionados a redes locais, em especial redes locais industriais e de automação de escritórios sendo, portanto, uma proposta de padronização de procedimentos de projeto de redes dessa natureza, não sendo aplicável, da maneira como foi definida, a redes de comunicação de longa distância, posto que para esse tipo de rede deve-se levar em consideração alguns outros parâmetros que podem ser ignorados no caso da rede ser local.

## 6.2.2) Descrição da metodologia NETCAD

Para atacar o problema de projetar uma rede local, decidiu-se dividir a tarefa em fases, de complexidade crescente, de maneira a que o sistema fosse modular e expansível. Assim, programadores podem implementar, de maneira autônoma, ferramentas de auxílio ao projeto de redes e, então, agregá-las ao sistema.

Arbitrou-se, com base em uma análise subjetiva, que o número de fases seria quatro cobrindo desde a identificação da estrutura hierárquica da

empresa até o acompanhamento da operação da rede projetada. Veja figura a seguir.

Como, porém, a tarefa de projetar um sistema desse porte é descomunal, escolheu-se investir na definição das diretrizes mestras do sistema como um todo, ficando concluido nesta Tese o primeiro módulo do sistema, parte do segundo módulo, bem como toda a interface-usuário baseada em janelas e menus, ficando a cargo de outras pessoas, e possivelmente outras Teses, completar as lacunas do sistema, e mesmo promover eventuais aperfeiçoamentos nos formulários e na base de dados.

A seguir, uma descrição da estrutura dos módulos propostos para o sistema. Observe que o primeiro módulo tem a descrição mais detalhada, enquanto que os módulos subsequentes são progressivamente menos discutidos em detalhes.

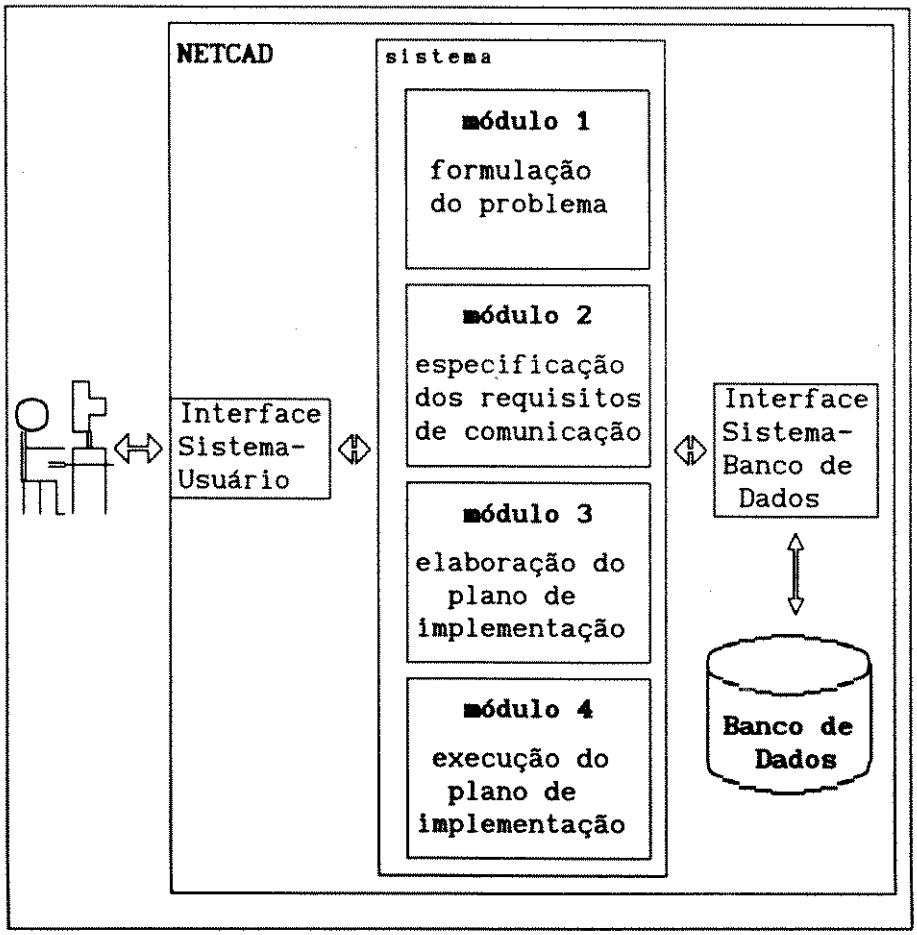



Figura 6.12 - Diagrama do sistema NETCAD (detalhado)

### 6.2.2.1) Os módulos

A cada uma das fases do projeto da rede corresponde um módulo do sistema; assim, de acordo com o exposto anteriormente, sendo quatro as fases do projeto, existem, correspondentemente, quatro módulos principais no sistema. As funções e as ferramentas disponíveis em cada módulo têm funcionalidades e complexidades crescentes, partindo-se da estaca zero no módulo 1 e chegando-se a uma proposta de rede a ser implementada ao final do módulo 4, com a a possibilidade de efetuar-se um acompanhamento da operação da rede implementada, para uma possível realimentação do sistema com vistas a torná-lo mais efetivo.

Incluiu-se em cada módulo uma opção de documentação, a qual não foi implementada mas é de suma importância para referências posteriores a cada uma das etapas do projeto de rede, e mesmo para a eventualidade de se ter que refazer partes do projeto ao serem constatados problemas de projeto ou, até mesmo, alterações nos requisitos de rede impostos pela direção da empresa, ou ainda dificuldades financeiras, por exemplo.

A seguir, uma descrição de cada um dos módulos.

* **Módulo 1** - *Formulação do Problema*

Este é o modulo de partida. Considera-se inicialmente nesta fase, que nada se sabe a respeito da empresa, dos equipamentos que ela possui e da rede a ser implementada. Assim, para começar a estabelecer o problema, inicia-se pela identificação da estrutura hierárquica da empresa em questão. Este trabalho deve ser levado a cabo por um "gerente de comunicações" ( ou congênere ) da empresa, posto que para efetuar este trabalho, o usuário do sistema deve possuir uma visão global da empresa, do ponto de vista das necessidades de comunicação da mesma.

Como resultado desta interação entre o "gerente de comunicações" e o sistema, tem-se agora, uma árvore que descreve uma hierarquia da empresa levando em conta o aspecto de comunicações; assim, tem-se um ponto de partida para o projeto da rede, pois poder-se-ia, em princípio, alocar a cada nó da árvore hierárquica uma sub-rede, sendo que a composição das várias sub-

redes corresponderia a uma rede maior que englobaria toda a empresa. Faz-se necessário, porém, lapidar esta rede, pois deve-se levar em consideração um número de outros fatores, para que a rede projetada seja viável, tanto fisicamente, como do ponto de vista de volume de tráfego, tempos de resposta a mensagens, gargalos de comunicação, e outros fatores que podem torná-la inviável na prática, inclusive custos e disponibilidade tecnológica.

Serão, ainda, gerados nesta fase, os formulários de pesquisa, os quais deverão ser preenchidos pelos responsáveis por cada área, setor, divisão (e assim por diante), da empresa. Tais formulários contêm questões muito mais específicas sobre cada um dos equipamentos (hardware e software), bem como relativas às características que cada um dos equipamentos impõe à rede, tanto do ponto de vista físico como dos serviços necessários para suportar as aplicações que requeiram comunicação. Além disto, existem formulários específicos para a coleta de informações a respeito de outros aspectos relevantes, como: facilidades de comunicação porventura já existentes na empresa, para um eventual aproveitamento no projeto da rede; expansões previstas para os equipamentos e aplicações; aplicações especiais, tais como tele-conferência, fax, e outras; requisitos especiais de qualidade de serviço, tais como taxa de erros máxima, tolerância a falhas, etc.

Deve-se considerar, ainda, a possibilidade de utilizar formulários de pesquisa distintos para tipos diferentes de empresas pois, por exemplo, questões que podem ser relevantes para fins de automação industrial, podem não ser de particular interesse para fins de automação comercial.

Como podem surgir dúvidas no preenchimento dos formulários pelas pessoas encarregadas dessa tarefa, e podem até mesmo haver distorções de interpretação, as quais se não forem corrigidas podem levar a um projeto de rede inviável, no sentido de ser super ou sub-dimensionado em determinadas áreas, o que fatalmente ocasionará pressões nos custos da rede, o ideal seria que o processo de preenchimento dos formulários fosse acompanhado por uma pessoa com um certo conhecimento acerca de redes de computadores.

Após a coleta dos formulários já preenchidos, é quase certo que nem todos os campos dos formulários estejam assinalados de maneira correta, e se assim for far-se-á necessário um novo contato com as pessoas que preencheram os formulários para sanar eventuais dúvidas, pois como esta é a fase inicial do projeto, erros ou distorções nesta etapa se propagarão às etapas subsequentes, podendo inviabilizar o projeto como um todo. Assim, vale a pena perder um pouco mais de tempo na verificação das informações e no cruzamento

dos dados para minimizar tais potenciais problemas futuros.

Preenchidos os formulários e efetuadas todas as correções necessárias, as informações constantes de cada um dos formulários devem ser inseridas na base de dados, para uso posterior por parte do sistema. Note-se que para cada tipo de formulário deve corresponder um arquivo na base de dados.

* **Módulo 2** - *Especificação dos Requisitos de Comunicação*

O objetivo deste módulo é a geração de uma proposta inicial de "Lay-out" da rede a ser implementada, tal proposta será denominada: **projeto lógico**. Para tanto, parte-se da hierarquia organizacional da empresa, e das informações a respeito de cada um dos equipamentos a serem conectados à rede, provenientes do módulo anterior e constantes, agora, da base de dados do sistema. Sobre estes dados, serão efetuadas análises como, por exemplo: *análise topológica*, *"data-flow analisys"*, e possivelmente *simulações*.

É interessante notar que neste módulo devem ser levadas em conta, por um lado, características físicas tais como a localização e distribuição geográficas dos equipamentos, visando minimizar problemas de comprimento de cabos e super-população em sub-redes. Por outro lado, paralelamente a isto, do ponto de vista de tráfego de rede, deve-se evitar que sub-redes possam apresentar gargalos de comunicação, mas também é desejável assegurar, para minimizar os custos de implantação da rede, que as sub-redes tenham um tráfego interno razoavelmente elevado que justifique sua existência, pois caso contrário, poder-se-ia agrupar diversas sub-redes em uma só, otimizando custos, pois seriam necessários menos dispositivos de rede ( amplificadores, roteadores, "gateways", etc ), além de comprimentos de cabo menores.

Um possível tema para futuros trabalhos poderia ser a automatização deste módulo, na forma de um sistema especialista (baseado em conhecimento), o qual, utilizando-se de ferramentas de teste e aplicando regras heurísticas, forneceria uma proposta de "lay-out" de rede, de forma análoga à descrita acima.

O resultado obtido ao final desta fase (ou módulo), deverá ser uma proposta de "lay-out" de rede que leva em conta aspectos físicos, topológicos e de tráfego, proposta esta que levando em consideração tais aspectos pode ser considerada viável do ponto de vista físico da implementação de

redes.

Convém notar que foge ao escopo deste trabalho a implementação das ferramentas supra-citadas, posto que nos propusemos tão somente a definir a funcionalidade deste módulo, sem a pretensão de implementar todas as ferramentas que foram vislumbradas no decorrer do processo de análise da situação-problema com a qual nos confrontamos.

* **Módulo 3** - *Elaboração do Plano de Implementação*

A finalidade deste módulo (ou fase) é elaborar um plano de implementação para a proposta de "lay-out" de rede esquematizada no módulo anterior do sistema. Deve-se levar em consideração, nesta fase, aspectos de disponibilidade de equipamentos de rede (amplificadores, "taps", transceptores, roteadores, "gateways", etc) no mercado, bem como analisar comparativamente os custos desses equipamentos para cada fabricante ou fornecedor, para obter, por exemplo, economia de escala na compra de uma grande quantidade de equipamentos de um mesmo fornecedor ou de manutenção, caso peças ou módulos de determinados equipamentos de um mesmo fabricante sejam intercambiáveis.

Deve-se gerar uma documentação detalhada e concisa desta fase, com informações minuciosas sobre o processo de coleta de informações sobre equipamentos de rede, seus fabricantes e fornecedores, bem como a respeito dos critérios comparativos que tenham, porventura, levado o projetista a optar por um determinado equipamento ou fornecedor de equipamentos de rede em determinada seção da rede ou na rede como um todo, se for o caso.

A documentação deve discorrer, ainda, detalhadamente sobre os passos a serem seguidos e os recursos a serem alocados durante o processo de execução do plano de implementação da rede proposta.

Como resultado desta fase pode-se prever, ainda, a geração de um organograma e de um cronograma referentes ao processo de implementação, às pessoas responsáveis por cada fase da implementação, e aos insumos necessários à medida que o processo de implementação avança.

* **Módulo 4** - *Execução do Plano de Implementação*

A função principal deste módulo é promover um acompanhamento do processo de implementação da rede cujo "lay-out" foi definido nos módulos anteriores. Tal acompanhamento deve ser dinâmico, no sentido de que deve reavaliar o plano de implementação, levando em conta eventuais alterações no projeto da rede em si e do cronograma de implementação da rede em função de possíveis mudanças nas características dos equipamentos ( inclusive os de rede ), inclusão de novas aplicações, imperativos de ordem física eventualmente ignorados ( por exemplo: ruído, meio ambiente agressivo, etc ), e quaisquer outras considerações que imponham alterações no projeto de rede original.

Pode-se pensar, ainda, em acompanhar a fase de operação piloto da rede e até mesmo a fase de operação definitiva, para constatar e eventualmente corrigir ( em projetos futuros ) problemas de projeto, por exemplo sub ou super-dimensionamento de sub-redes, características ambientais eventualmente desconsideradas, e quaisquer outras considerações que possam ser levadas em conta em projetos futuros, com vistas a tornar o sistema mais efetivo na tarefa de auxiliar no projeto de uma rede local de computadores.

É de suma importância a documentação da dinâmica deste processo para "sintonizar" o sistema com o projeto real ( prático ) de redes de computadores. Pode-se inclusive pensar, como proposta para trabalhos futuros, em transformar o sistema em um **sistema especialista**, ou seja, baseado em conhecimento, conhecimento este que seria adicionado à base de dados ( no caso: **base de conhecimento** ) do sistema durante todo o processo de projeto da rede, de sua implementação, e mesmo durante o acompanhamento da operação da rede, obtendo informações sobre eventuais problemas ou deficiências da rede implementada.

### 6.2.3) Aplicação da metodologia à problemática de projeto de Redes Locais

Faremos, aqui, uma breve descrição de como a metodologia proposta foi utilizada para a criação do Sistema de Apoio ao Projeto de Redes Locais de Computadores - NETCAD.

* **1ª fase** - *formulação do problema*

Nesta fase os esforços se concentram em duas tarefas principais. Numa primeira sub-fase, a meta é identificar e organizar as estruturas de informação referentes ao projeto.

Na segunda sub-fase, o objetivo é fazer um levantamento e posterior mapeamento dos requisitos, impostos ao sistema pelo projeto, contra as capacidades do SAP.

A seguir, uma descrição mais pormenorizada de cada uma dessas sub-fases.

**- caracterização do projeto**

Tenta-se, aqui, identificar, classificar e organizar as estruturas de informação (e suas relações de hierarquia) relativas ao problema; ou seja, referentes ao projeto de Redes Locais de Computadores.

Como o nosso problema é o projeto de Redes Locais, os dados têm relação com as características dos equipamentos (relevantes para comunicações), suas conexões, tamanhos das mensagens, volume de tráfego, atrasos toleráveis, requisitos de qualidade de serviço, aplicações a nível global, etc.

A seguir um painel das principais estrutras de dados identificadas para o nosso "problema". O conjunto completo das estruturas de dados propostas consta do anexo 9. O arquivo que as contém denomina-se DDLREC.

características dos equipamentos
- -Nome
- -Sub Rede
- -Fabricante e Modelo
- -tipo da CPU
- -Quantidade de memória
- -nºde portas seriais
- -Sistema operacional
- -Periféricos

Conexões entre equipamentos
- -Equipamento A
- -Equipamento B
- -Meio físico utilizado
- -Tamanho das mensagens A⇒B
- -Tamanho das mensagens B⇒A
- -Frequência das mensagens A⇒B
- -Frequência das mensagens B⇒A

Conexões entre sub-redes
- -Nome da Sub_Rede
- -Sub-Rede conectada
- -Meio físico utilizado
- -Tamanho da informação enviada
- -Frequência da informação enviada
- -Tamanho da informação recebida
- -Frequência da informação recebida

tráfego de equipamento
- -Nome do equipamento
- -Sub Rede
- -tráfego de entrada (média)
- -tráfego de entrada (pico)
- -tráfego de saída (média)
- -tráfego de saída (pico)

tráfego entre sub-redes
- -Equipamento (fonte)
- -Sub-Rede
- -Equipamento destino
- -Nome da aplicação
- -Frequência da aplicação
- -Concorrência entre aplicações
- -Atraso máximo tolerável
- -Tamanho das mensagens
- -Tráfego médio
- -Tráfego máximo

Qualidade de Serviço
- -Serviço
- -Requisitos de qualidade

Aplicações Globais
- -Nome da Aplicação
- -Nível
- -Descrição

**- requisitos operacionais**

A finalidade, aqui, é o estabelecimento dos requisitos referentes à maneira como o sistema deve operar de forma a satisfazer as necessidades do projeto.

Um dos requisitos propostos, neste caso exemplo, é que o SAP, quando completamente implementado, deve acompanhar o ciclo completo de desenvolvimento de um projeto de Rede Local de Computadores, a partir de sua concepção, chegando até a fase de operação e manutenção.

Como não se trata de uma aplicação crítica em termos de segurança e integridade dos dados, não são feitas exigências especiais a respeito de segurança de acesso e garantia de integridade dos dados do projeto nem, tampouco, de robustez e tolerância a falhas para o sistema.

Outro requisito operacional diz respeito à interface-usuário, a qual deve ser "amigável", baseando-se em janelas e menus.

O SAP deverá ser composto de quatro módulos distintos, conforme detalhado nas seções anteriores.

*** 2ª fase** - *configuração do sistema*

De posse, agora, das estruturas de dados e dos requisitos de operação do sistema, o próximo passo é estabelecer a configuração do sistema, no sentido de selecionar os módulos de programa que serão agrupados para formar o SAP com as características desejadas.

Assim, neste exemplo, foram selecionados os módulos seguintes:

- Base de Dados (Kernel) - sem os módulos de recuperação de erros, acesso em tempo-real, segurança de acesso, etc.

- Interface-Usuário - Utilização da interface baseada em janelas e menus descrita anteriormente.

- Ferramentas - Como o SAP será utilizado no projeto de Redes Locais, devem ser fornecidas ferramentas de manipulação de estruturas de dados hierárquicos (árvores), de simulação, de análise de níveis de sinal elétrico, de análise de tráfego, etc.

- Módulos - O projeto, conforme descrito anteriormente, será composto de quatro módulos distintos, entre os quais serão divididas as ferramentas.

*** 3ª fase**

**- elaboração dos formulários**

Nesta sub-fase, a temática são os formulários de pesquisa. Tratou-se, aqui, do projeto de seu "lay-out".

Para a criação dos formulários de pesquisa utilizou-se da mesma ferramenta utilizada para a edição deste trabalho de Tese, ou seja, o editor de textos ChiWriter.[1]

No desenvolvimento do projeto dos formulários de pesquisa tentou-se adotar um "lay-out" limpo para facilitar a visualização e o correto preenchimento dos campos deste.

O formulário apresentado na página seguinte corresponde ao exemplo de arquivo da Base de Dados descrito na sub-fase seguinte desta metodologia, o qual refere-se à descrição das características relevantes, do ponto de vista das comunicações, dos equipamentos de uma dada empresa.

[1] ChiWriter é marca registrada da Horstmann Software Design Corporation

AR_EQP1

Características dos Equipamentos Existentes

| Sub-Rede | |
|---|---|

| Características => / Equipamentos | Modelo (Fabricante) | CPU | Quantidade de Memória | Nº de Portas Seriais | Sistema Operacional | Periféricos Conectados |
|---|---|---|---|---|---|---|
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |

nalidade: Fornecer as características dos equipamentos que vão ser integrados à rede.

b-Rede: Nome da sub-rede (se conhecida) onde se localiza(m) o(s) equipamento(s) em questão.

delo/Fabricante: Fabricante e/ou modelo do equipamento.

U: Tipo da Unidade Central de Processamento.

antidade de Memória: Memória primária (RAM), em Kbytes, do equipamento.

mero de Portas Seriais: Número de portas seriais disponíveis no equipamento.

stema Operacional/Versão: Tipo e versão do Sistema Operacional utilizado.

riféricos conectados: Indicar todos os periféricos conectados diretamente ao equipamento.

Figura 6.15 - Exemplo de formulário

Observe, emoldurado no canto superior esquerdo, o nome do formulário, o qual é idêntico ao do arquivo correspondente da Base de Dados.

Note-se, ainda, na parte inferior do formulário, um pequeno guia para o significado de cada um de seus campos componentes e, adicionalmente, uma descrição sucinta da finalidade do formulário.

**- criação da Base de Dados**

O trabalho realizado nesta fase foi o de criar registros que espelhassem as estruturas de dados identificadas na sub-fase anterior e organizá-las em arquivos de uma base de dados.

Vamos apresentar, a título de exemplo, apenas um dos registros, e arquivo correspondente, que compõem a Base de Dados.

O conjunto completo de tais registros consta do anexo 9 na listagem do arquivo DDLREC. Os arquivos correspondentes da Base de Dados são apresentados na listagem do arquivo NETCAD.DDL do mesmo anexo.

As listagens a seguir apresentam, respectivamente, a estrutura de registro e o arquivo formado a partir deste, para a descrição das características dos equipamentos da empresa.

```
NREC = EQUIPAMENTO
       STRUCT[ NOME_EQP           CHAR(20),
               SUB_REDE           CHAR(20),
               FABRICANTE_MODELO  CHAR(20),
               CPU                CHAR(20),
               MEMORIA            FIXED(4),
               PORTAS_SERIAIS     FIXED(1),
               SO                 CHAR(20),
               PERIFERICOS_1      CHAR(20),
               PERIFERICOS_2      CHAR(20),
               PERIFERICOS_3      CHAR(20),
               PERIFERICOS_4      CHAR(20),
               PERIFERICOS_5      CHAR(20)];
```

```
NDBF = CAR_EQP1,
       FILE=D_CAR_EQP1 / VOL=100,
       REC=EQUIPAMENTO,
       STRAT=BBAUM /KEYLGT=20
                   /KEYPOS=1
                   /KEYTYP=4
                   /SYNKEY=1
                   /KEY=[ NOME_EQP ];
```

O anexo 7 contém uma descrição mais detalhada de cada uma das palavras chave apresentadas, por exemplo BBAUM (estratégia B-tree ou árvore binária).

* **4ª fase** - *coleta de informações*

**- preenchimento dos formulários**

Esta sub-fase tem como objetivo a identificação de pessoas "chave", às quais serão entregues os formulários de pesquisa para preenchimento.

Como esta tarefa só pode ser efetuada em casos práticos, somos forçados a deixá-la de lado.

**- preenchimento da Base de Dados**

Para a transferência dos dados constantes dos formulários já preenchidos para a Base de Dados, foram criadas janelas associadas a cada um dos formulários.

Devido a limitações de resolução dos monitores de computador, não é possível apresentar-se, na tela, uma janela com lay-out idêntico à versão dos formulários impressos em papel. Assim, optou-se por criar um lay-out muito mais simples, organizado verticalmente, que, porém presta-se muito bem à finalidade de permitir a entrada de dados de maneira simples e direta.

Cada janela é composta de duas partes: a parte esquerda apresenta os

nomes de cada campo do formulário e a parte direita é a área de entrada de dados, a qual permite edição do conteúdo de cada linha sendo digitada.

A figura a seguir apresenta a janela de entrada de dados correspondente ao formulário de características dos equipamentos descrito anteriormente. Observe que existe uma entrada referente a cada campo existente no formulário associado.

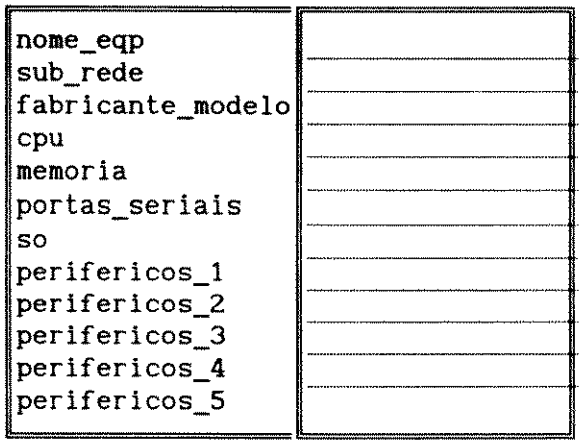



Figura 6.14 - Janela para entrada de dados

As linhas pontilhadas na parte direita da janela são apenas para ilustar a divisão em linhas da janela, não sendo apresentadas, realmente, na tela do computador.

A largura da parte direita da janela deve ser tal que acomode o comprimento máximo permitido para os campos do formulário, comprimento este ditado pelas características do Sistema de Gerenciamento do Banco de Dados.

Para cada um dos outros formulários existe, analogamente, uma janela correspondente para permitir a entrada dos dados.

* **5ª fase** - *manipulação da informação*

Qualquer programa que seja agregado ao Sistema de Apoio a Projetos,

resultante da metodologia descrita, pode acessar os dados armazenados na Base de Dados. Para tanto, tal programa deve "conhecer" a estrutura dos arquivos da Base de Dados que ele necessite acessar e, a partir dessas informações, prover funções que efetuem as operações desejadas sobre os arquivos escolhidos.

A maneira, porém, como um programa manipula as informações disponíveis na Base de Dados foge do escopo da nossa metodologia.

Os programas podem, adicionalmente, armazenar na Base de Dados os resultados de seus processamentos para fins de registro, documentação ou, possivelmente, uso posterior por parte de outros programas.

Um programa poderia, por exemplo, efetuar análises de nível de sinal elétrico, análises de tráfego, simulações e etc, utilizando os dados armazenados na Base de Dados, os quais foram obtidos através dos formulários de pesquisa. Ao final do processamento, o programa efetuaria chamadas à Base de Dados para salvar os resultados obtidos.

O capítulo seguinte apresenta as conclusões do nosso trabalho.

# Capítulo 7

# Conclusões

## 7.1) Sumário

Faremos, aqui, uma breve revisão do conteúdo de cada um dos capítulos anteriores, para que possamos oferecer uma visão global deste trabalho.

No primeiro capítulo, formulamos a situação-problema, descrevendo o panorama dos Sistemas de Apoio a Projetos, as dificuldades associadas a eles e as características desejáveis nesses sistemas. A partir dessa situação-problema, apresentamos o objetivo deste estudo, que vem a ser a criação de uma metodologia de projeto de tal tipo de sistema.

Apresentamos, ainda, as razões que nos levaram a escolher a linguagem C como a linguagem de programação principal.

O segundo capítulo apresenta uma descrição mais detalhada dos requisitos associados a um Sistema de Apoio a Projetos genérico.

O terceiro capítulo foi dedicado à descrição do que identificamos como sendo o componente central de um Sistema de Apoio a Projetos: o Banco de Dados. Assim, discorremos genericamente sobre Bancos de Dados, apresentando a terminologia e os componentes associados a eles, dedicando especial atenção ao Sistema de Banco de Dados que escolhemos como base para nossa proposta de metodologia, ou seja: o **BAPAS-DB**.

No quarto capítulo, nos concentramos na descrição de uma Interface-Usuário para os Sistemas de Apoio a Projetos a serem criados utilizando nossa proposta de metodologia. Tal interface possibilita a criação de múltiplas janelas e menus na tela do computador e pode, se utilizada com inteligência, facilitar grandemente a interação entre o usuário e o sistema. Adicionalmente durante o processo de elaboração da metodologia que estamos propondo, detectamos a necessidade de algumas "ferramentas de apoio", as quais julgamos serem úteis em um Sistema de Auxílio a Projetos. Este quarto capítulo compreende, também, a descrição de tais "ferramentas", quais sejam: **árvores** ( úteis na manipulação de dados hierárquicos ), **formulários** ( para facilitar a coleta de dados dispersos ) e **ferramentas** ( de acesso ao Sistema Operacional e suas funções ).

O quinto capítulo apresenta o que podemos considerar o ponto central deste trabalho, ou seja, a metodologia propriamente dita, a qual é composta de quatro fases sucessivas.

Como corolário deste trabalho, o sexto capítulo apresenta um estudo de caso de um sistema criado segundo a metodologia proposta. Tal Sistema de

Apoio a Projetos tem como nome NETCAD, e é o esboço de um Sistema de Apoio a Projeto de Redes Locais de Computadores.

## 7.2) Conclusões

Para a implementação da metodologia utilizamos uma coleção de vários subsistemas. Assim, apresentaremos uma conclusão específica para cada tópico e, ao final, uma conclusão geral sobre o trabalho como um todo.

### 7.2.1) Sobre o BAPAS-DB

O Sistema de Banco de Dados BAPAS-DB, escolhido para fazer parte da metodologia que propusemos, revelou-se adequado. As suas características de arquivos estruturados e facilidade de interação com programas (processos) são extremamente úteis para este tipo de aplicação. Esta última, principalmente, é fundamental na nossa metodologia, pois os acessos à Base de Dados de um sistema, no nosso caso, são comandados através de menus e janelas que existem no "programa principal" do sistema que, desta forma, oculta do usuário final detalhes de implementação e dificuldades de controle de acesso à Base de Dados. O que se consegue, assim, é um "shell"[1] que padroniza e simplifica a interação entre o usuário e a Base de Dados do sistema.

Outras características também interessantes do BAPAS-DB, como tolerância a falhas, resposta em tempo-real e distribuição não são aplicáveis na plataforma computacional que utilizamos (IBM-PC sob MS-DOS), por ser esta monotarefa e monousuário. Em sistemas multitarefa e/ou multiusuário tais facilidades são de extrema importância. Assim, seria muito interessante adaptar nosso sistema para utilizar tais recursos, por exemplo em uma rede de PS/2 sob OS/2.

Algumas peculiaridades, ainda, da versão do BAPAS-DB para PC sob DOS são a exigência de que o processador seja da família AT (pelo menos um 80286) dotado de coprocessador aritmético. Este, porém, pode ser emulado, se

[1] poderíamos traduzir este termo por "casca" ou então, "máscara"

a velocidade de cálculos em ponto flutuante não for um fator determinante para o sistema.

### 7.2.2) Sobre a Interface-Usuário

A interface-usuário utilizada é poderosa, modular, extremamente rápida (por usar acesso direto ao hardware) e versátil.

A maior limitação desta interface diz respeito à plataforma computacional para a qual foi planejada, ou seja, IBM-PC sob MS-DOS. Este tipo de plataforma tem severas limitações de quantidade de memória, além de ser mono-tarefa e mono-usuário. Como a interface utiliza controle direto do hardware de vídeo para aumentar a velocidade, torna-se uma tarefa extremamente difícil portá-la para outro tipo de sistema computacional, pois tais características são intimamente ligadas ao hardware.

Assim, para um outro tipo de plataforma computacional, por exemplo workstations SUN sob SUN/OS (UNIX), seria mais lógico utilizar outro tipo de sistema de interface-usuário, a qual fosse planejada especificamente para esse tipo de plataforma.

### 7.2.3) Sobre a Metodologia

A metodologia proposta neste trabalho foi concebida para ser bastante abrangente e genérica e, desta forma, não estar intimamente ligada a nenhuma área específica de conhecimento científico.

Assim, pode-se pensar em utilizá-la para implementar Sistemas de Apoio a Projetos voltados a aplicações em áreas totalmente diversas da apresentada em nosso estudo de caso.[1]

A metodologia mostrou-se, também, bastante útil como uma espécie de "guia conceitual" da lógica envolvida no projeto de Sistemas de Apoio a Projetos.

---

[1] Veja o ítem 7.3.3 - Utilização da Metodologia em outras aplicações

### 7.2.4) Sobre o Estudo de Caso (Netcad)

O sistema NETCAD é, na verdade, apenas o arcabouço de um Sistema de Apoio ao Projeto de Redes Locais. Foram definidas as funcionalidades de cada um de seus módulos de maneira bastante genérica, sendo implementadas apenas parte das funções previstas para o sistema. O trabalho maior focalizou a definição funcional de cada módulo e a interação entre a Base de Dados e a Interface-Usuário.

Implementamos a maior parte da funcionalidade do módulo 1 do sistema NETCAD e, apenas, um "kernel" para os demais módulos propostos.

O número de módulos e sua funcionalidade, porém, não são definições "fechadas"; isto significa que trabalhos futuros poderiam alterar tanto o número quanto o conteúdo de módulos do sistema.

Apesar de todas essas considerações, o estudo de caso cumpriu a contento a sua finalidade de provar na prática a possibilidade de integração entre os sub-sistemas de Banco de Dados, Interface-Usuário e programas de aplicação.

### 7.2.5) Conclusões Gerais

Como resultado maior deste trabalho, o que se obteve foi um conjunto de ferramentas de "software" e uma metodologia para interligá-las, de forma a permitir a criação de Sistemas de Apoio a Projetos que as utilizem.

As ferramentas foram especialmente escolhidas e, em alguns casos, desenvolvidas de modo a serem totalmente compatíveis umas com as outras.

Cada uma das ferramentas é, por si só, interessante e útil, porém quando interligadas em um sistema, tal como proposto neste trabalho, adquirem uma aplicação prática muito mais poderosa.

A metodologia nada mais é do que uma espécie de "receita de bolo" para a caracterização do problema e, a partir daí, a configuração das ferramentas de forma a compor um Sistema de Apoio a Projetos relacionado ao problema original. Esta metodologia é "genérica" e, como tal, independente da plataforma computacional e, até mesmo, do conjunto de ferramentas utilizadas, as quais podem ser modificadas ou, até mesmo, introduzidas novas ferramentas.

## 7.3) Sugestões de Pesquisas Adicionais

### 7.3.1) Ampliação da Funcionalidade do Netcad

Existe uma Tese de mestrado [*O-Zoghelb 92*], desenvolvida a partir deste trabalho, que se propõe a implementar, pelo menos parcialmente, a funcionalidade descrita para o módulo 2 do sistema NETCAD. Para tal, utiliza-se das informações armazenadas na Base de Dados como ponto de partida para os seus processamentos e salva os resultados obtidos, também, na Base de Dados do sistema.

São oferecidas, também, como propostas de futuros trabalhos, as funções descritas (mas não implementadas) nos outros módulos, bem como a ampliação daquilo que foi efetivamente implementado, possivelmente com alterações e adições à funcionalidade proposta para o sistema Netcad, como um todo, e a cada um dos seus "módulos".

### 7.3.2) Migração para outras plataformas computacionais

É importante frisar, mais uma vez, que o Sistema de Gerenciamento de Banco de Dados **BAPAS-DB** está disponível para uma ampla gama de plataformas computacionais diferentes.

Uma vez que o Sistema de Gerenciamento de Banco de Dados é a peça fundamental de um Sistema de Apoio a Projetos, pode-se pensar em "migrar" o sistema proposto para outros tipos de plataformas computacionais mais poderosas.

Os principais tipos de sistema computacional suportados por este SGBD estão descritos no capítulo referente a Bancos de Dados (capítulo 3) deste trabalho. Uma lista completa pode ser encontrada nos manuais de referência do BAPAS-DB.

A título de recapitulação vamos, novamente, citar os que são, no nosso entender, os mais largamente utilizados.

| COMPUTADOR | SISTEMA OPERACIONAL |
|---|---|
| IBM PC/AT e PS/2 | OS/2 |
| workstations SUN | SunOS (UNIX) |
| família Apollo Domain DN 3000 | AEGIS (UNIX) |
| família HP 9000 | HP-UX (UNIX) |
| família PCS Cadmus 9000 | MUNIX (UNIX) |
| família VAX e MicroVAX | VMS |

Tabela 7.1 - Versões disponíveis do BAPAS-DB

Cumpre notar, também, que fizemos algumas experiências de utilização da G.U.I. **OpenWindows** como Interface-Usuário para o sistema Netcad em ambiente SUN/UNIX.

Uma G.U.I (Graphical User Interface) é uma interface-usuário padronizada baseada em janelas.

Existem vários sistemas desse tipo disponíveis no mercado, dentre os quais podemos citar como exemplos: MS-Windows, X-Windows, SunView, OSF/Motif e o próprio OpenWindows.

Não foram feitos, porém, testes de integração entre o OpenWindows e o BAPAS-DB nesse ambiente, por não dispormos, na época, da versão do BAPAS-DB para essa plataforma. Os resultados obtidos, porém, são extremamente promissores e são indicados como uma sugestão especialmente interessante de pesquisas adicionais.

As listagens referentes a esses "testes" constam do anexo 10.

### 7.3.3) Utilização da Metodologia em outras Aplicações

Outra possível fonte de pesquisas adicionais, é a utilização da metodologia, proposta neste trabalho, para a implementação de Sistemas de Apoio a Projetos voltados para outros tipos de aplicações, que não projeto

de Redes Locais.

Poderiamos vislumbrar, por exemplo, sua utilização para a criação de sistemas desse tipo, voltados a aplicações nas áreas de medicina, controle de processos, projeto de veículos automotores, etc.

# BIBLIOGRAFIA

- [E] Engenharia Concorrente

[E-*Babish 91*] **Babish, Jim; Barton, Curtis; Wong, Joe** - "Large-scale projects: developing Patriot missiles and other big systems", IEEE Spectrum, pp.34-36 - Julho, 1991.

[E-*Burnett 91*] **Burnett, Robert W.** - "Ingredients for a booming start-up: collaboration, communications, consensus", IEEE Spectrum, pp.33-34 - Julho, 1991.

[E-*Carter 91*] **Carter, Donald E.; Baker, Barbara S.** - Concurrent Engineering, the product development environment for the 1990s (*volume one*) - Beaverton, Oregon: Mentor Graphics Corporation, 1991.

[E-*Hall 91*] **Hall, Donald** - "Concurrent Engineering: defining terms and techniques", IEEE Spectrum, pp.24-25 - Julho, 1991.

[E-*Reddy 91*] **Reddy, Ramana; Wood, Ralph T.; Cleetus, K.J.** - "The DARPA Initiative: encouraging new industrial practices", IEEE Spectrum, pp.26-30 - Julho, 1991.

[E-*Rosenblatt 91a*] **Rosenblatt, Alfred** - "Making the one-of-a-kind type of product right the first time", IEEE Spectrum, pp.36-37 - Julho, 1991.

[E-*Rosenblatt 91b*] **Rosenblatt, Alfred; Watson, George F.** - "Concurrent Engineering (*special report*)", IEEE Spectrum, p.22 - Julho, 1991.

[E-*Shina 91*] **Shina, Sammy G.** - "New Rules for World Class Companies", IEEE Spectrum, pp.23-26 - Julho, 1991.

[E-*Sprague 91*] **Sprague, Robert A.; Singh, Kamar J.; Wood, Ralph T.** - "Concurrent Engineering in Product Development", IEEE Design & Test of Computers, pp.6-13 - Março, 1991.

[E-*Turino 91*] **Turino, Jon** - "Making it work: calls for input from everyone", IEEE Spectrum, pp.30-32 - Julho, 1991.

[E-*Wheeler 91*] **Wheeler, Roy** - "Small-work projects: teamwork counts more than computer-based tools", IEEE Spectrum, pp.32-33 - Julho, 1991.

**- [R] Redes Locais de Computadores**

[R-*Archer 87*] **Archer, Rowland** - Redes de Microcomputadores IBM-PC e compatíveis - São Paulo: McGraw Hill, 1987.

[R-*autor 86*] **autor não citado** - "Rapid rise in use of LANs result of local needs", Communications Engineering and Design, pp.38-45 - Junho, 1986.

[R-*Baker 86*] **Baker, Donald G.** - Local-Area Networks with Fiber-Optic Applications - New Jersey: Prentice-Hall, 1986.

[R-*Beale 86*] **Beale, Nicolas** - "The MAP initiative", Computer-Aided Engineering Journal, pp.79-82 - Junho, 1986.

[R-*Carvalho 86*] **Carvalho, Rodney F.de** - Automação de Escritórios - Rio de Janeiro: LTC Livros Técnicos e Científicos Editora, 1986.

[R-*Cavellucci 86*] **Cavellucci, Celso et al.** - Automação de Escritórios - Campinas: Editora Cartgraf, 1986.

[R-*CONSAI 87*] **CONSAI (Consultores Associados em Automação Industrial)** - Metodologia de elaboração do plano de implantação do sistema de comunicação do SIIC - Campinas, 1987.

[R-*Crowley 79*] **Crowley, Fred; Hussain, Shahid** - "Planning data networks", Telesis, pp.22-26 - Abril, 1979.

[R-*Donnelly 84*] **Donnelly, Raymond E.; Lonneman, Marcel J.** - "Local Area Network Management and System Administration", Proceedings of the 7th International Conference on Computer Communications, pp.420-426 - Sydney, Australia: 1984.

[R-*Gibson 91*] **Gibson, R.W.** - "IEEE 802 standardization efforts", Computer Standards & Interfaces, pp.193-203 - Dezembro, 1991.

[R-*Gilhooly 85*] **Gilhooly, Denis** - "Local area networks-market still maturing", Communications Engineering International, pp.25-43 - Junho, 1985.

[R-*Giozza 86a*] **Giozza, William F.et al.** - Redes Locais de Computadores: Protocolos de Alto Nível e Avaliação de Desempenho - São Paulo: McGraw Hill, 1986.

[R-*Giozza 86b*] **Giozza, William F. et al.** - Redes Locais de Computadores: Tecnologia e Aplicações - São Paulo: McGraw Hill, 1986.

[R-*Gomide 87*] **Gomide, Fernando A.C.; Netto, Márcio L.A.** - Introdução à Automação Industrial Informatizada (EBAI) - Buenos Aires: Editora Kapelusz, 1987.

[R-*Gupta 85*] **Gupta, Ved P.** - "What is Network Planning", IEEE Communications Magazine, vol.23, nº10, pp.10-16 - Outubro, 1985.

[R-*Kaminski 86*] **Kaminski Jr., Michael A.** - "Protocols for communicating in the factory", IEEE Spectrum, pp.56-62 - Abril, 1986.

[R-*Karp 82*] **Karp, Peggy M.; Socher, Ivan D.** - "Designing local-area networks", Mini-micro Systems, pp.219-232 - Abril, 1982.

[R-*Knisley 85*] **Knisley, Joseph** - "Design and installation of local area networks (part 2)", EC&M magazine, pp.87-90 - Novembro, 1985.

[R-*Knisley 86*] **Knisley, Joseph** - "Design and installation of local area networks (part 3)", EC&M magazine, pp.77-85 - Fevereiro, 1986.

[R-*Leite 85*] **Leite, José R.E.** - "O modelo de referência para a interconexão de Sistemas Abertos", Revista telebrás, pp.11-15 - junho, 1985.

[R-*Little 81*] **Little, Donald; Morcott, Tod; Budelman, Gerald** - "Design/Selection Considerations for Local Area Data Link Systems", Digital Design, pp.58-63 - Maio, 1981.

[R-*McCarthy 85*] **McCarthy, James J** - "MAP's Impact on Process Plants", Control Engineering, pp.67-69 - Outubro, 1985.

[R-*Mokhoff 85*] **Mokhoff, Nicolas** - "Demand grows for integrated open networks", Computer Design, pp.51-61 - Setembro, 1985.

[R-*Moraes 87*] **Moraes Jr., Amaury** - "Redes Locais", Micro Sistemas, pp.24-27 - Maio, 1987.

[R-*NBS 82*] **National Bureau of Standards (NBS)** - The Selection of Local Area Computer Networks, NBS special publication nº 500-96 - Washington: U.S.Government Printing Office, 1982.

[R-*O'Connor 85*] **O'Connor, Torn; Brown, Ron; Benke, Don** - "Planning and installing the network", I&CS (The Industrial and Process Control Magazine), pp.51-59 - Agosto, 1985.

[R-*Oliveira 90*] **Oliveira, Fernando J.de; Rivera, Roberto R.M.** - Redes Locais nas Empresas - Rio de JAneiro: Editora Campus, 1990.

[R-*Prince 81*] **Prince, S.M.; Sloman M.S.** - "Communications requirements of a distributed computer control system", IEEE proceedings, vol.128 pp.21-34 - Janeiro, 1981.

[R-*Stacy 87*] **Stacy, Alan H** - The MAP book: an introduction to industrial Networking - Santa Clara, California: Industrial Networking Incorporated, 1987.

[R-*St.John 81*] **St.John, Richard** - "Planning and building a local network", Telephony, pp.24-26 - Setembro, 1981.

[R-*Zakir 88*] **Zakir Jr.,José** - Redes Locais: o estudo de seus elementos - Rio de Janeiro: LTC Livros Técnicos e Científicos Editora, 1988.

- [B] **Bancos de Dados**

[B-*Casanova 85*] **Casanova, Marco A.; Moura, Arnaldo V.** - Princípios de Sistemas de Gerência de Bancos de Dados Distribuídos - Rio de Janeiro: Editora Campus, 1985.

[B-*Chu 83*] **Chu, Shao Yong** - Bancos de Dados: organização, sistemas e administração - São Paulo: Editora Atlas, 1983.

[B-*BAPAS 88*] **Werum Datenverarbeitungssysteme** - BAPAS-DB The Open Realtime Database System for Industrial Purposes - Lueneburg, Deutshland: 1988.

- [C] **Linguagem C**

[C-*Cabral 89*] **Cabral, Fernando** - A linguagem C e o PC-BIOS - Rio de Janeiro: Editora Campus, 1989.

[C-*MSC 87*] **Microsoft Corporation** - Microsoft-C 5.0 (User's Guide, Language Reference, Run-Time Library Reference) - 1987.

[C-*Sobelman 85*] **Sobelman, Gerald E.; Krekelberg, David E.** - Advanced C: Techniques & Applications - Indianapolis: QUE Corporation, 1985.

[C-*TC 87*] **Borland International Inc.** - Turbo-C Reference Guide - Scotts Valley, California: 1987.

[C-*Weiskamp 88*] **Weiskamp, Keith** - Advanced Turbo C Programming - San Diego: Academic Press, 1988.

- [D] DOS

[D-*Norton 88*] **Norton, Peter; Wilton, Richard** - The new Peter Norton programmer's guide to the IBM-PC & PS/2 - Redmond, Washington: Microsoft Press, 1988.

- [O] **Outras referências**

[O-*Bastos 81*] **Bastos, Lília R.; Paixão, Lyra; Fernandes, Lucia M.** - Manual para a elaboração de projetos e relatórios de pesquisa, teses e dissertações - Rio de Janeiro: Zahar Editores, 1981.

[O-*Zogheib 92*] **Zogheib, Suhel G.** - "Um Sistema para Auxílio a Projeto de Redes Locais de Computadores OSI/ISO", Tese de Mestrado - Campinas: UNICAMP, 1992.

# Anexo 1

## Níveis de configuração e Dados Técnicos do BAPAS-DB

O BAPAS-DB pode ser configurado de maneira muito flexível, por ter uma construção modular. A configuração mínima, por exemplo para sistemas sem disco, compõe-se somente do kernel do sistema e da estratégia de acesso HSP, sem o uso de checkpoints e medidas de recuperação de falhas. A partir daí, pode-se obter várias outras configurações.

A configuração padrão, útil na maioria dos casos, é a seguinte:

- **configuração padrão**

- kernel do sistema pré-configurado com sistema padrão de buffers;
- estratégias de acesso FIFO, HASH e sequencial indexada;
- recuperação de falhas com checkpoints, before images e arquivos log;
- transações com controle de prioridades;
- visões (views);
- interface de programas DBV para chamadas de programas em C e PEARL;
- linguagem de descrição de dados DDL;
- linguagem de consulta QL.

Dependendo do tipo de computador, as seguintes extensões são possíveis:

- **possibilidades de extensão**

- Capacidade de configurar o tamanho do dicionário de dados, a quantidade e tamanho dos sistemas de buffers, bem como o número máximo de arquivos (máximo de 32.767);
- Estratégias

  *B-TREE* para acesso direto com ordenamento,

  acesso sequencial-indexado com buffer circular,

  acesso sequencial-indexado com tamanho de registro variável;
- Distribuição em uma rede de computadores com *n* servidores e *m* clientes através do uso de DB-server e DB-client;
- Instalação de discos duplicados através do componente DB-disk backup;
- Instalação duplicada (à prova de falhas) em dois servidores com monitoração mútua e atualização do sistema faltoso, a partir do remanescente, com ou sem operação interrompida, através do componente DB-system backup;

- Pré-compilador BAPAS-SQL;

- Sistema de máscaras BAPAS-MMC.

*** Dados técnicos**

BAPAS-DB pode ser configurado com facilidade. Os parâmetros, e seus valores máximos, são os seguintes:

| | |
|---|---|
| - Número de arquivos | 32.767 |
| - Número de registros por arquivo | 2.147.483.647 |
| - Número de páginas por arquivo | 32.767 |
| - Tamanho de uma página (em 2 bytes) | 32.767 |
| - Número de sistemas de buffers | 32.767 |
| - Número de buffers por sistema de buffer | 32.767 |
| - Tamanho de um buffer (em 2 bytes) | 32.767 |
| - Tamanho do dicionário de dados (em 2 bytes) | 32.767 |

Os seguintes limites também devem ser considerados:

| | |
|---|---|
| - Tamanho máximo de registro (fixo) | tamanho de página menos 26 bytes |
| - Tamanho máximo de registro (variável) | tamanho do arquivo |
| - Tamanho máximo de chave | 40 caracteres |
| - Número máximo de chaves por registro | 9 |
| - Número máximo de componentes de registro | tamanho do registro (em 2 bytes) |
| - Número máximo de chaves por arquivo | número de registros X máx. chaves/registro |
| - Número máximo de processos do usuário | restrito pelo S.O. |
| - Número máximo de transações abertas simultaneamente | número de processos do usuário |
| - Número máximo de arquivos abertos simultameamente | irrestrito |

| | |
|---|---|
| - Tamanho máximo de um identificador de base de dados | 20 caracteres |
| - Tamanho máximo de um identificador de arquivo | 20 caracteres |
| - Tamanho máximo de um identificador de componente de registro | 30 caracteres |
| - Tamanho da configuração mínima sem buffer | aproximadamente 120KB |
| - Tamanho da configuração máxima sem buffer | aproximadamente 500KB |

# ANEXO 2

ICDBV.INC - O ARQUIVO DE INCLUSÃO PARA A INTERFACE DE PROGRAMAS C DO BAPAS-DB

```
/************************  ICDBV  ************************************/

/*************  INTERFACE FOR DBV_UTILIZATION IN C  *******************/

/********************  GENERAL CONSTANTS  ******************************/

#define C_LENGTH_NAME                          20
#define C_LENGTH_RECORD_IN_DB_ACCESS           20    /* depends on user */
#define C_MAX_LENGTH_KEY                       20
#define C_MAX_STRATEGY_NUMBER                   9

/********** These Type Definitions must be made in the Program *********/

#define TYPE_FILEVARIABLE void
#define TYPE_DB_CONDITION void

/***************************************************************************/

/*******  SELECTION CONSTANS FOR ACCESS IN TYPE_DB_CONDITION  **********/

#define C_KEY                                  0X04000000L
#define C_KEY2                                 0X00002000L
#define C_KEY3                                 0X00004000L
#define C_KEY4                                 0X00008000L
#define MULT1                                  0X00008000L
#define MULT2                                  0X00004000L
#define MULT3                                  0X00002000L
#define MULT4                                  0X00001000L
#define MULT5                                  0X00000800L
#define MULT6                                  0X00000400L
#define MULT7                                  0X00000200L
#define MULT8                                  0X00000100L
#define C_RNR                                  0X80000000L
#define C_SEQUENTIAL                           0X00000000L
#define C_POSITION                             0X00400000L

/********  SELECTION CONSTANTS FOR EXCLUSION IN TYPE_FILE  ************/

#define C_EXCLUSIVE                            0X0020
#define C_SHARED                               0X0000

/**** SELECTION CONSTANT FOR STRUCTUREINDEX IN  TYPE_NAMEVARIABLE *****/

#define C_NV_STRUCTUREINDEX_NIL                0X0000

/***** SELECTION CONSTANTS FOR ACCESS IN TYPE_NAMEVARIABLE ************/

#define C_NV_ACCESS_DB_NAME                    0X0001
#define C_NV_ACCESS_DBF_NAME                   0X0002
#define C_NV_ACCESS_REL_NAME                   0X0003
#define C_NV_ACCESS_RELDBF_NAME                0X0004
#define C_NV_ACCESS_DBF                        0X0005
#define C_NV_ACCESS_RELATION                   0X0006
```

```
/********  SELECTION CONSTANTS FOR STATUS IN TYPE_TAVARIABLE  *********/

#define C_TA_TEMP                                   0X000C0000L
#define C_TA_PERM                                   0X000D0000L
#define C_TA_TEMPNOI                                0X010C0000L
#define C_TA_PERMNOI                                0X010D0000L

/** SELECTION CONSTANTS FOR THE STATUS MODIFIER FOR TWO PROCESSORS  ***/

#define C_MASTER_NORMAL                             0X0001
#define C_MASTER_MASTER                             0X0002
#define C_SLAVE_NORMAL                              0X0003
#define C_SLAVE_MASTER                              0X0004

/*****************  CONSTANT FOR ERROR MANIPULATION  ********************/

#define C_DBV_ERROR_TA_SETBACK                      0X0067


/* DEFINITIONS  ***********************************************************/

 typedef struct {
                  short LENGTH;
                  short KEY_ID;
                }
 TYPE_KEY_UNUSED;
 /*************/


 typedef struct {
                  short CLASS;
                  short NUMBER;
                }
 TYPE_ERROR;
 /*********/


 typedef struct {
                  short      COMPUTER_IDF;
                  short      PROCESS_IDF;
                  char       DB_NAME [ C_LENGTH_NAME ];
                  char       FILE_NAME [ C_LENGTH_NAME ];
                  TYPE_ERROR ERROR;
                  short      PROCEDUREACCESS;
                }
 TYPE_DBVERROR;
 /************/


 typedef struct {
                  short COMPUTER_IDF;
                  short PROCESS_IDF;
                  char  DB_NAME   [ C_LENGTH_NAME ];
                  char  FILE_NAME [ C_LENGTH_NAME ];
                }
 TYPE_DBV_ATTACHMENT;
 /*****************/
```

```
typedef struct {
                  short       FILE_IDF;
                  short       PROCESS_IDF;
                  TYPE_ERROR  ERROR;
                  short       EXCLUSION;
                }
TYPE_FILE;
/*******/


typedef struct {
                  char                NAME [ C_LENGTH_NAME ];
                  short               MULTIKEY_NR;
                  short               KEYPOSITION;
                  short               KEYLENGTH;
                }
TYPE_STRATEGY;
/***********/


typedef struct {
                  long                MAX_RECORDNUMBER;
                  long                ACTUAL_RECORDNUMBER;
                  short               RECORDLENGTH;
                  short               STRATEGY_NUMBER;
                  TYPE_STRATEGY       STRATEGY [ C_MAX_STRATEGY_NUMBER ];
                }
TYPE_FILE_INFO;
/************/


typedef struct {
                  short               ACCESS;
                  short               STRUCTUREINDEX;
                  char                NAME [ C_LENGTH_NAME ];
                  TYPE_ERROR          ERROR;
                }
TYPE_NAMEVARIABLE;
/***************/


typedef struct {
                  short DAY;
                  short MONTH;
                  short YEAR;
                  long  HOUR;
                }
TYPE_CP_TIME;
/**********/
```

```
typedef struct {
                  short        COMPUTER_IDF;
                  short        PROCESS_IDF;
                  TYPE_CP_TIME CP_TIME;
                  TYPE_ERROR   ERROR;
                }
TYPE_CP_VARIABLE;
/***************/


/* MODEL FOR ONE FILEVARIABLE *** Defined in the Program *************

typedef struct {
                  TYPE_FILE FILE;
                  char      RECORD [ C_LENGTH_RECORD_IN_DB_ACCESS ];
                }
TYPE_FILEVARIABLE;
*****************/


/* MODEL FOR ONE KEY ************************************************

typedef struct {
                  short LENGTH;
                  short KEY_ID;
                  char  VALUE [ C_MAX_LENGTH_KEY ];
                }
TYPE_KEY_CONDITION;
******************/


#define TYPE_CONDITION_HEAD short SEQ_ID ;\
                            long  ACCESS ;\
                            long  RNR    ;

/* MODEL FOR ONE CONDITION *** Defined in the Program *****************

typedef struct {
                  TYPE_CONDITION_HEAD
                  TYPE_KEY_CONDITION  KEY ;
                  TYPE_KEY_CONDITION  KEY2;
                  TYPE_KEY_CONDITION  KEY3;
                  TYPE_KEY_CONDITION  KEY4;
                }
TYPE_DB_CONDITION;
*****************/


/* FOR THE TRANSACTIONS ********************************************/

typedef struct {
                  short FILE_IDF;
                  short EXCLUSION;
                }
TYPE_TAFILE;
/*********/
```

```
typedef struct {
                  short          FILE_NUMBER ;
                  TYPE_TAFILE    FILE [10];
                }
TYPE_TAFILELIST;
/**************/

typedef struct {
                  short              COMPUTER_IDF;
                  short              PROCESS_IDF;
                  short              TAPRIO;
                  TYPE_TAFILELIST    FILELIST;
                }
TYPE_TAATTACHMENT;
/****************/

typedef struct {
                  short          TA_IDF;
                  short          COMPUTER_IDF;
                  short          PROCESS_IDF;
                  TYPE_ERROR     ERROR;
                  long           STATUS;
                }
TYPE_TAVARIABLE;
/**************/


/* FOR TWO PROCESSORS ************************************************/

typedef struct {
                  short       COMPUTER_IDF;
                  short       PROCESS_IDF;
                  TYPE_ERROR  ERROR;
                }
TYPE_COMPUTERDEFINITION;
/**********************/

/*****************************************************************************/

typedef struct {
                  char                IDENTIFIER [30];
                  short               DEPTH;
                  short               TYPE;
                  short               LENGTH;
                  short               DIMENSION;
                }
TYPE_REC;
/*******/


typedef struct {
                  short               NUMBER;
                  char                REC_NAME [20];
                  TYPE_REC            RECORD [50];
                }
TYPE_RECORDSTRUCTURE;
/****************/
```

```
/******************  DBV - PROCEDURES  ******************************/

extern void  pascal  DBVATTACH_     ( TYPE_DBV_ATTACHMENT*,
                                      TYPE_FILE* );
extern void  pascal  DBVCLOSE_      ( TYPE_FILE* );
extern       pascal  DBVCPTIME      ( TYPE_CP_VARIABLE* );
extern void  pascal  DBVDBFINFO_    ( TYPE_FILE*,
                                      TYPE_FILE_INFO* );
extern void  pascal  DBVDETACH_     ( TYPE_FILE* );
extern void  pascal  DBVEND_        ( TYPE_ERROR* );
extern void  pascal  DBVERASE_      ( TYPE_FILEVARIABLE* );
extern short pascal  DBVFALL_       ( TYPE_FILEVARIABLE*,
                                      TYPE_DB_CONDITION*
                                      /* ,short* */);
extern void  pascal  DBVFILECLEAR_  ( TYPE_FILE*
                                      /* ,unsigned short* */);
extern short pascal  DBVFORONE_     ( TYPE_FILEVARIABLE*,
                                      TYPE_DB_CONDITION*
                                      /* ,short* */);
extern void  pascal  DBVFREE_       ( TYPE_FILEVARIABLE* );
extern void  pascal  DBVINITIATION_ ( TYPE_ERROR* );
extern void  pascal  DBVINSERT_     ( TYPE_FILEVARIABLE* );
extern short pascal  DBVNAMES_      ( TYPE_DBV_ATTACHMENT*,
                                      TYPE_NAMEVARIABLE*
                                      /* ,unsigned short* */);
extern void  pascal  DBVOPEN_       ( TYPE_FILE* );
extern void  pascal  DBVLOOPEXIT_   ( TYPE_FILE* );
extern void  pascal  DBVQUITT_      ( TYPE_FILEVARIABLE* );
extern short pascal  DBVPRED_       ( TYPE_FILEVARIABLE*,
                                      TYPE_DB_CONDITION*
                                      /* ,short* */);
extern void  pascal  DBVTABEGIN_    ( TYPE_TAVARIABLE* );
extern void  pascal  DBVTACLOSE_    ( TYPE_TAVARIABLE* );
extern void  pascal  DBVTADETACH_   ( TYPE_TAVARIABLE* );
extern void  pascal  DBVTAOPEN_     ( TYPE_TAVARIABLE* );
extern void  pascal  DBVTAUNDO_     ( TYPE_TAVARIABLE* );
extern void  pascal  DBVTAEND_      ( TYPE_TAVARIABLE* );
extern void  pascal  DBVTAATTACH_   ( TYPE_TAATTACHMENT*,
                                      TYPE_TAVARIABLE* );
extern void  pascal  DBVTRIGGER_    ( TYPE_ERROR* );
extern void  pascal  DBVREADDICT_   ( TYPE_ERROR* );

extern       pascal  DBVZUSTAND     ( short*,
                     /*status*/       TYPE_COMPUTERDEFINITION* );

extern DBV_DORE_AKTUAL_ABSCHLIESSEN ( TYPE_COMPUTERDEFINITION* );
              /*actual_terminate*/

extern DBV_DORE_AKTUAL_VORBER       ( TYPE_COMPUTERDEFINITION* );
              /*actual_prepare*/

extern DBV_UEBERTRAGEN_DATEI        ( TYPE_FILE* );
         /*transfer_file*/

extern void  pascal  DBVGETFILEDESCR_ ( TYPE_RECORDSTRUCTURE* );
```

# Anexo 3

## Funções disponíveis na interface de programas DBV do BAPAS-DB e listagem das mensagens de Erro

| Operações sobre a Base de Dados e seus arquivos | |
|---|---|
| **DBVINITIATION_** | inicializa a interface DBV (deve ser chamada antes das demais funções da interface) |
| **DBVEND_** | finaliza o acesso à interface DBV (as demais funções, exceto DBVINITIATION, ficam indisponíveis) |
| **DBVATTACH_** | conecta um arquivo a um processo, possibilitando o acesso do processo ao arquivo conectado |
| **DBVDETACH_** | desconecta um arquivo de um processo |
| **DBVOPEN_** | abre o arquivo, possibilitando o acesso a seus registros |
| **DBVCLOSE_** | fecha o arquivo |

| Operações sobre os registros | |
|---|---|
| **DBVINSERT_** | insere um registro em um arquivo (de acordo com a estratégia de acesso a esse arquivo) |
| **DBVERASE_** | remove (apaga) um registro de um arquivo |
| **DBVFORONE_** | recupera um registro de um arquivo, de acordo a estratégia de acesso e uma condição de busca |
| **DBVFALL_** | recupera, sucessivamente, cada registro de um arquivo, usando a estratégia de acesso e uma condição |
| **DBVPRED_** | análogo a DBVFALL_, mas os registros são recuperados de acordo com o inverso da estratégia de acesso |
| **DBVLOOPEXIT_** | retorna o cursor de um arquivo para o início |
| **DBVFREE_** | atualiza e libera um registro (selecionado no modo exclusivo) de um arquivo |
| **DBVQUITT_** | libera um registro, de um arquivo, que tenha sido selecionado no modo exclusivo |
| **DBVFILECLEAR_** | remove (ou apaga) todos os registros de um arquivo |

| Suporte a Transações Atômicas | |
|---|---|
| **DBVTAATTACH_** | adiciona uma transação |
| **DBVTADETACH_** | descarta uma transação |
| **DBVTAOPEN_** | declara uma transação |
| **DBVTACLOSE_** | libera uma transação |
| **DBVTABEGIN_** | inicia o corpo de uma transação |
| **DBVTAEND_** | encerra o corpo de uma transação |
| **DBVTAUNDO_** | desfaz uma transação |

| Informações sobre os arquivos da Base de Dados | |
|---|---|
| **DBVNAMES_** | chamadas sucessivas a esta função obtêm os nomes dos arquivos que compõe a Base de Dados |
| **DBVDBFINFO_** | informações sobre cada arquivo (número utilizado e máximo de registros, estratégia de acesso, etc) |

| Suporte a Tolerância a Falhas | |
|---|---|
| **DBVTRIGGER_** | dispara explicitamente um "checkpoint" |
| **DBVCPTIME_** | obtém a data e hora do último "checkpoint" |

**Mensagens de Erro da interface DBV do BAPAS-DB**

A FILE-OPEN ERROR
A FILE-CLOSE ERROR
A FILE-READ ERROR
A FILE-WRITE ERROR
A SEEK-ERROR
A NOSPACE ERROR
A FUNCTION ERROR
A FILE-EXPAND ERROR
A DATABASE-IDENTIFIER UNKNOWN
A FILE-IDENTIFIER UNKNOWN
A NO FILE-IDENTIFIER GIVEN
A STRATEGY UNKNOWN
A WRITE-ERROR ON TRANSCTION-FILE
W FILE FILLED! MAX. NUMBER OF RECORDS REACHED
W FILE FILLED! MAX. NUMBER OF PAGES REACHED
W NEW KEY ALREADY DEFINED
W RECORD-LENGTH + MANAG.DATA > PAGE LENGTH
W KEY-LENGTH GREATER THAN 20 WORDS
W DISK-ERROR IN SYSTEM OCCURRED
W MAX. NUMBER OF FILES REACHED
W MULTIKEY ONLY WITH DIRECT-ACCESS-STRATEGIES
A BOTH DICTIONARY-FILES ARE DISTURBED
W OVERFLOW OF MULTIKEYS (MAX. IS 8)
W WRONG MULTIKEY-NUMBER
W MAXIMUM OF KEYS REACHED
W NO MORE MEMORY
W NOT ALLOWED TO DELETE FILE
W FILE ALREADY CREATED
A FILE-CREATE ERROR
W NO KEY FOR SEQUENCE IDENTIFIER DEFINED
W MORE THAN FOUR CONDITIONS AT ACCESS
A RECORD-EXPAND WITHOUT COPY-ACCESS INVALID
W OPERATION NOT IMPLEMENTED
W INVALID STRATEGY-CONFIGURATION
W NUMBER OF KEYS IN RECORD GREATER THAN 4

```
W RECORD-NUMBER ALREADY DEFINED
W RECORD-NUMBER OUT OF BOUNDS
W MEMORY-OVERFLOW
W CIRCLE-BUFFER FILLED: NO INSERT POSSIBLE
W UPDATE-ERROR: NO RECORD LOCKED
W READ-ERROR: LAST RECORD STILL LOCKED
W WRONG DATABASE-CALL-SEQUENCE
W WRONG DATABASE-OPERATION-CODE
W RECORD NOT FOUND
W SEQUENCE OF RECORDS BROKEN ON DELETION
W USER-BUFFER TOO SMALL FOR DESIRED RECORD
W INSERT VIA VIEW NOT POSSIBLE
W USER-KEYBUFFER TOO SMALL FOR DESIRED FILE
W LAST RECORD REACHED
W FILE IS EMPTY
W MIN. OF RECORD-LENGTH 4 OR 6 WORDS (STRATEGY!)
W ACCESS NEEDS RECORD-NUMBER DECLARATION
W ACCESS NEEDS KEY DECLARATION
W KEY NOT FOUND
A VIEWS NOT IMPLEMENTED
A LENGTH OF KEY IS GREATER THAN LENGTH OF RECORD
W VARIABLE RECORDLENGTH SHORTENED
W BUFFER-SYSTEM NOT GREAT ENOUGH FOR MEMORY-FILE
W WRONG FILE VARIABLE IN DBVTAOPEN
W FILE NOT DECLARED FOR TRANSACTIONS
W TCB IN DBVTAOPEN STILL ACTIVE IN ANOTHER TRANSACTION
W TCB IN DBVTABEGIN NOT FOUND IN CHAIN OF TCB'S; DBVTAOPEN FORGOTTEN ?
W TCB IN DBVTABEGIN STILL ACTIVE ?
W NO DBVTAOPEN AND DBVTABEGIN AFTER DBVTAATTACH
A NO SEQUENCE-IDENTIFIER DEFINED
W WRONG NUMBER OF DECLARED KEYS
W TCB NOT ACTIV; DBVTABEGIN FORGOTTEN ?
W CHANGE ON INDEX DATA WITHOUT FILE LOCK
W LIST-OF-FILES IS EMPTY IN DBVTAOPEN
W THE TRANSACTION WAS SET BACK TO THE ""ROLL-BACK POINT""
A MEMORY-ALLOCATION OVERFLOW
A SESSION TERMINATED
A PASS OVER BORDERS OF PAGE (FREE RECORD)
A PASS OVER BORDERS OF PAGE (ERASE RECORD)
A LAST RECORD NOT RELEASED
W FILE NOT OPENED BY DBVOPEN
W FILE ALREADY OPENED BY DBVOPEN
W FUNCTION NOT YET IMPLEMENTED
A OVERFLOW DBV-STRUCTURE (DBV-ZUGRIFFSVERWALTUNG)
A OVERFLOW DBV-STRUCTURE (DBV-TYP DB TAB)
W DATABASE/FILE-IDENTIFICATION WRONG
W RECORD WAS NOT REQUESTED BY LOCK
W KEY TOO LONG
W KEYLENGTH IS NEGATIVE
W TOO MANY FILES IN TRANSACTION DEFINED
A OVERFLOW DB-STRUCTURE (TA VERWALTUNG)
W TRANSACTION NOT OPEN. DBVTAOPEN MISSING
W TRANSACTION NOT CLOSED. DBVTACLOSE MISSING
A OVERFLOW DB-STRUCTURE (HUELLE-FILETAB)
W ATTACHMENT.COMPUTER IDF WRONG
W ATTACHMENT.PROCESS IDF WRONG
```

A OVERFLOW DB-STRUCTURE (HUELLE-PROCESS IDFS)
W NO MULTIKEY-FILES
W THERE IS A NON-DETACHED TRANSACTION AT DBVEND
A NON-DETACHED TRANSACTION AT DBVEND; SECOND FAULT DURING UNDO TRANSACTION
A OVERFLOW HSTTA VW-TABLE ON RESTART OR ROLLBACK
A OVERFLOW FELD HSTTA SPERRVW
A NO MORE BUFFER FOR DUPLICATES
A HSP-FILE IN DBV UEBERTRAGEN FILE
A OVERFLOW OF BEFORE-IMAGE-FILE
A ILLEGAL STATE MASTER/SLAVE
A Wrong subprogram statement. Internal error.
A Record too long. Adapt generation parameter.
W Too many processes. Adapt generation parameter.
A Not enough coupling tasks created.
A Not enough vice processing tasks created.
W Wrong clean-up function.
A Error on the opening a communication line
A Error on the closing a communication line
A Error on the reading a communication line
A Error on the writing a communication line

OBS: A = Abort (erro fatal)
W = Warning (aviso)

# Anexo 4

## Descrição sucinta das Funções C dos Sistemas de Listas, Janelas, Menus e Árvores,

## ✱ Funções relacionadas a janelas

| **funções relacionadas a janelas** (alto nível) | |
|---|---|
| **initwindows** | Inicializa o sistema de janelas |
| **popupwindow** | cria uma nova janela |
| **movewindow** | move uma janela para uma posição especificada |
| **cmovewindow** | move uma janela utilizando as setas do teclado |
| **rmvwindow** | remove uma janela |
| **selectwindow** | seleciona uma determinada janela como janela ativa |

| **funções relacionadas a janelas** (internas) | |
|---|---|
| **make_window_node** | Cria um "nó" de janela |
| **push_window_node** | adiciona um "nó" de janela à pilha de janelas |
| **dispose_window_node** | Remove um "nó" de janela |
| **swap_arrow** | Apaga ou exibe uma seta piscante |
| **swapvram** | Intercambia a memória de vídeo (RAM) |

| **funções relacionadas a janelas** (suporte) | |
|---|---|
| **setscrptr** | atribui à memória de vídeo o ponteiro de tela |
| **setscrmode** | seleciona o modo de tela corrente |
| **setcursor** | Define o formato do cursor |
| **wherexabs** | Determina a coordenada x absoluta |
| **whereyabs** | Determina a coordenada y absoluta |
| **gotoxyabs** | Posiciona o cursor na posição (x,y) |
| **scrollabs** | Rola a tela para cima ou para baixo um certo nº de linhas |
| **putchatrabs** | Exibe um caractere com atributo |

| funções relacionadas a janelas (suporte) - *continuação* | |
|---|---|
| **getchatrabs** | Lê um caractere com atributo |
| **putstratrabs** | Exibe uma cadeia de caracteres com atributo em uma posição especificada da tela |
| **getstratrabs** | Lê uma cadeia de caracteres com atributo em uma posição da tela especificada |
| **checkcoord** | Testa a válidade das coordenadas absolutas |
| **window** | Atribui as coordenadas da janela ativa às coordenadas internas do sistema. |
| **textcolor** | Devolve ou seleciona a cor de texto |
| **gotoxy** | Posiciona o cursor em relação às coordenadas da janela ativa |
| **wherex** | Devolve a posição x do cursor, em relação à janela ativa. |
| **wherey** | Devolve a posição y do cursor, em relação à janela ativa |
| **scroll** | Rola n linhas na janela ativa |
| **putchatr** | Exibe um caractere, com atributos |
| **getchatr** | Obtém o caractere, com atributos |
| **putstratr** | Exibe uma cadeia de caracteres, com atributos |
| **getstratr** | Obtém a cadeia de caracteres, com atributos |
| **clrscr** | Limpa a tela da janela ativa e move o cursor para a posição (0,0) relativa |
| **prthz** | Exibe uma cadeia de caracteres horizontalmente com atributos em uma dada posição da janela ativa. |
| **wrthz** | Idem prthz; porém aceita caracteres de nova linha, e faz truncamento e rolamento automatico de tela |
| **prtvt** | Exibe uma cadeia de caracteres verticalmente com atributos em uma dada posição da janela ativa |
| **prthzf** | Exibe uma cadeia de caracteres verticalmente com atributos em uma dada posição da janela ativa, com formatação |
| **scrintf** | Análoga a printf porém a saída é gerada na janela ativa. Suporta rolamento e truncamento |
| **box** | Exibe uma "caixa" de tipo especificado em coordenadas absolutas |
| **centerstr** | Exibe uma cadeia de caracteres centrada entre coordenadas absolutas, com atributo |
| **crabs** | Envia um sinal de retorno de carro para a página corrente de vídeo |
| **lfabs** | Envia n sinais de alimentação de linha para a página de vídeo corrente, com coordenadas dadas e atributo |
| **crlf** | Envia um sinal de retorno de carro e alimentação de linha para a janela ativa |

*** Funções relacionadas a menus**

| **funções relacionadas a menus** (alto nível) | |
|---|---|
| popupmenu | cria um menu |
| pulldwn | cria um sistema de menus do tipo "pulldown" |
| rmvmenu | remove um menu |
| movearound | movimenta a barra de seleção dentro de um menu |

| **funções relacionadas a menus** (internas) | |
|---|---|
| displist | Exibe a lista de ítens do menu |
| menusize | Calcula as dimensões de um menu |
| getmenuparms | Determina os parâmetros de um menu |
| adjsize | Ajusta o tamanho de um menu |
| isvmenu | Determina se um menu é do tipo "vertical" |
| ishmenu | Determina se um menu é do tipo "horizontal" |
| isrectmenu | Determina se um menu é do tipo "retangular" |
| sumchar | Determina a largura de um menu pelo cálculo do número de caracteres do maior ítem do menu |
| space | Determina o espaçamento de um menu |
| colx | Determina a coordenada x de uma coluna de um menu |
| barlocx | Determina a coordenada x da barra de seleção de um menu |
| barlocy | Determina a coordenada y da barra de seleção de um menu |
| dispbar | Exibe a barra de seleção de um menu |
| homebar | Processa a tecla <home> |
| endbar | Processa a tecla <end> |
| pgupbar | Processa a tecla <pg up> |
| pgdnbar | Processa a tecla <pg dn> |

| **funções relacionadas a menus** (internas) — *continuação* | |
|---|---|
| upbar | Processa a tecla < ↑ > |
| downbar | Processa a tecla < ↓ > |
| leftbar | Processa a tecla < ← > |
| rightbar | Processa a tecla < → > |
| firstchar | Processa o primeiro caractere de uma seleção de menu |
| adji | Determina o número de ítens do menu a serem exibidos |

A figura a seguir relaciona as teclas que movimentam a barra de seleção de um menu, e os códigos de retorno existentes.

| **Tecla** | **Tipo de Menu** | | **código de retorno** |
|---|---|---|---|
| | **Rolável** | **Não Rolável** | |
| left arrow | move p/esquerda | move p/esquerda | goleft |
| right arrow | move p/direita | move p/direita | goright |
| up arrow | move p/cima | move p/cima | goup |
| down arrow | move p/baixo | move p/baixo | godn |
| home | primeira página | sem efeito | — |
| end | última página | sem efeito | — |
| page up | página anterior | sem efeito | — |
| page down | página seguinte | sem efeito | — |
| < RETURN > | seleciona ítem | seleciona ítem | doit |
| < ESCAPE > | abandona o menu | abandona o menu | abortit |
| — | — | — | terminate |
| — | — | — | donothing |
| **Diagrama das teclas que movimentam a barra de seleção do menu** | | | |

## ♯ Funções relacionadas a listas de cadeias de caracteres

| **funções relacionadas a listas de cadeias de caracteres** (alto nível) | |
|---|---|
| **stol** | Converte uma cadeia de caracteres em uma lista |
| **prtstrlst** | Exibe uma lista |
| **appendstrlst** | Adiciona uma cadeia de caracteres ao fim de uma lista |
| **stol** | Converte uma cadeia de caracteres em uma lista |
| **prtstrlst** | Exibe uma lista |
| **appendstrlst** | Adiciona uma cadeia de caracteres ao fim de uma lista |
| **insstrlst** | Insere uma cadeia de caracteres em uma posição especificada de uma lista |
| **memberstrlst** | Verifica se uma determinada cadeia de caracteres pertence a uma lista |
| **delstrlst** | Remove uma cadeia de caracteres de uma lista |
| **concatstrlst** | Concatena duas listas |
| **convertstrlst** | Converte uma lista (de acordo com um código) |
| **updatekeystrlst** | Modifica a "chave" de um elemento de uma lista |
| **getkeystrlst** | Obtém a "chave" de um elemento de uma lista |
| **lenstrlst** | Determina o tamanho (nº de elementos) de uma lista |
| **makestrlst** | Cria uma estrutura de lista |
| **freestrlst** | Remove uma lista |
| **clrstrlst** | Limpa (esvazia) uma lista |

| **funções relacionadas a listas de cadeias de caracteres** (internas) | |
|---|---|
| **makestrnode** | Cria um nó de lista |
| **maxstrelem** | Determina o tamanho do maior elemento de uma lista |
| **prtstrelem** | Exibe um elemento de uma lista |
| **cmpstrelem** | Compara dois elementos de lista |

| **funções relacionadas a listas de cadeias de caracteres** (internas) *cont.* | |
|---|---|
| **delstrelem** | Remove um elemento de lista |
| **makestrelem** | Cria um elemento de lista |
| **getstrptr** | Obtém um ponteiro para um elemento de lista |

* **Funções relacionadas a árvores**

| **funções relacionadas a árvores** (alto nível) | |
|---|---|
| **init_tree** | Inicia uma nova árvore |
| **add_child** | Adiciona um filho à lista de filhos de um nó |
| **add_brother** | Adiciona um irmão a um nó (exceto o nó raíz) |
| **leftmost** | Acha o primeiro filho de um nó |
| **root** | Dado um nó achar a raíz da árvore a que esse nó pertence |
| **is_leaf** | Determina se um nó é, ou não, um nó "folha" |
| **pre_ord** | Percorre a árvore de acordo com o esquema *pré-ordem*, executando uma função selecionada |
| **in_ord** | Percorre a árvore de acordo com o esquema *em-ordem*, executando uma função selecionada |
| **post_ord** | Percorre a árvore de acordo com o esquema *pós-ordem*, executando uma função selecionada |
| **print_it** | Função que exibe um nó de uma árvore |
| **print_it_idented** | Função que exibe um nó de uma árvore,com identação proporcional à sua profundidade |
| **del_node** | Função que apaga um nó de uma árvore apagando também toda a sub-árvore abaixo desse nó |
| **change_node** | Função que modifica a cadeia de caracteres associada a um determinado nó de uma árvore |
| **del_tree** | Remove completamente da memória dinâmica uma árvore liberando toda a memória a ela alocada |
| **is_member** | Verifica se um nó é membro de uma árvore |
| **print_i_tree** | Função que exibe uma árvore na tela, sendo cada nó identado de acordo com sua profundidade |
| **tree2file** | Função que armazena uma árvore completa em um arquivo em disco |

| **funções relacionadas a árvores** (internas) | |
|---|---|
| **inst_info** | instancia um novo nó de uma árvore |
| **delete_it** | função que apaga um nó da árvore |
| **to_file** | função que armazena um nó em um arquivo. É usada pela função *tree2file*. |

| **funções relacionadas a árvores** (suporte) - *lista de cadeias de caracteres* | |
|---|---|
| **create** | Cria uma nova lista de cadeias de caracteres |
| **insert** | Insere um elemento no início de uma lista |
| **append** | Adiciona um elemento ao final de uma lista |
| **delete** | Remove o primeiro elemento de uma lista |
| **inst_node** | Função interna que instancia um novo nó na lista de cadeias de caracteres. Só deve ser chamada dentro de uma função |
| **assign** | Atribui uma cadeia de caracteres a um elemento de lista |

*** Funções relacionadas às ferramentas do Sistema Operacional**

| **ferramentas** | |
|---|---|
| **tools** | invoca o menu principal que torna disponíveis as funções "ferramentas do S.O." propriamente ditas |
| **dir** | apresenta uma lista rolável dos arquivos existentes no diretorio corrente |
| **pwd** | exibe o diretório corrente |
| **cd** | muda o diretório corrente |
| **md** | cria um sub-diretório |
| **rd** | remove um sub-diretório |
| **disk** | exibe e/ou muda o disco corrente |
| **lstack** | exibe a quantidade de pilha (stack) disponivel para o programa (em bytes) |
| **shell** | executa um programa, deixando o programa principal residente na memória |

# ANEXO 5

## Exemplos dos Utilitários

*** listas de cadeias de caracteres**

```
/*
                          lista.c

          Programa teste de listas de cadeias de caracteres
*/

# include <stdio.h>
# include <stdlib.h>
# include <conio.h>
# include <ctype.h>

# include "strlst.c"

void main(void)
{
 strlist lista=NULL, elemento;
 char dado[80];
 int chave, pos;

 while(1)
 {
  fflush(stdin);

  printf("\n\nPrograma teste de listas\n\n");
  printf("P -> imPrime a lista\n");
  printf("M -> procura um Membro\n");
  printf("A -> Adiciona um elemento\n");
  printf("I -> Insere um elemento\n");
  printf("C -> obtém a Chave de um elemento\n");
  printf("T -> Troca a chave de um elemento\n");
  printf("R -> Remove um elemento\n");
  printf("L -> remove a Lista\n");
  printf("S -> Saida\n");

  printf("\n\nSelecione a opção >>> ");

  switch(toupper(getche()))
  {
   case 'P': printf("\n\n");
             prtstrlst(lista);
             break;
   case 'M': printf("\nElemento >>> ");
             scanf("%s",dado);
             if(memberstrlst(dado,lista,&elemento))
             {
              printf("\n[");
              prtstrelem(elemento->dn.elem);
              printf("]");
             }
             else printf("\nElemento não encontrado !");
             break;
   case 'A': printf("\nElemento >>> ");
             scanf("%s",dado);
```

```
            printf("Chave >>> ");
            scanf("%d",&chave);
            if(appendstrlst(dado,0,chave,&lista)) printf("\nOK!");
            else printf("\nErro!");
            break;
   case 'I': printf("\nElemento >>> ");
            scanf("%s",dado);
            printf("Chave >>> ");
            scanf("%d",&chave);
            printf("Posição >>> ");
            scanf("%d",&pos);
            if(insstrlst(dado,0,chave,pos,&lista)) printf("\nOK!");
            else printf("\nErro!");
            break;
   case 'C': printf("\nElemento >>> ");
            scanf("%s",dado);
            if(getkeystrlst(dado,&chave,lista))
             printf("\nChave é %d",chave);
            else printf("\nElemento não encontrado !");
            break;
   case 'T': printf("\nElemento >>> ");
            scanf("%s",dado);
            printf("nova chave >>> ");
            scanf("%d",&chave);
            if(!updatekeystrlst(dado,chave,lista)) printf("\nOK!");
            else printf("\nElemento não encontrado !");
            break;
   case 'R': printf("\nPosição >>> ");
            scanf("%d",&pos);
            if(delstrlst(pos,lista)) printf("\nOK!");
            else printf("\nPosição inválida !");
            break;
   case 'L': printf("\nconfirme (s/n) >>> ");
            if(toupper(getche())=='S') clrstrlst(&lista);
            break;
   case 'S': exit(0);
            break;
  }
 }
}
```

*** janelas**

```
/*
                        janela.c

          Programa de teste do sistema de janelas.

     - link: janela + ioutil + scrnio + popup
*/

# include "ioutil.h"
# include "scrnio.h"
```

```
void main(void)
{
 wptr janela_1, janela_2, janela_3, janela_4, janela_ptr;

 initwindows();

 makestat("Pressione qualquer tecla",112);

 popupwindow( 3, 3,30,7,"Janela #1",overlap,&monocolors,&janela_1);
 pause();
 scrintf("Escrevendo na primeira janela do exemplo.");
 pause();

 popupwindow(13, 7,30,7,"Janela #2",overlap,&monocolors,&janela_2);
 pause();
 scrintf("Escrevendo na segunda janela do exemplo.");
 pause();

 popupwindow(23,11,30,7,"Janela #3",overlap,&monocolors,&janela_3);
 pause();
 scrintf("Escrevendo na terceira janela do exemplo.");
 pause();

 popupwindow(33,15,30,7,"Janela #4",overlap,&monocolors,&janela_4);
 pause();
 scrintf("Escrevendo na quarta janela do exemplo.");
 scrintf("\n\nMovendo a quarta janela do exemplo.");

 makestat("Mova a janela com as setas e tecle <ESC> para parar.",112);

 cmovewindow(&janela_4);

 makestat("Pressione qualquer tecla para continuar.",112);
 scrintf("\n\nVou retirar a janela #2."
         "\nAs seguintes (3 e 4) são retiradas automaticamente.");
 pause();

 rmvwindow(&janela_2);

 makestat("Pressione qualquer tecla para sair.",112);
 clrscr();
 scrintf("Fim do teste.\n\nTchau !!!");

 popupwindow(10,1,20,3,"Janela Apontador",vtpt,&monocolors,&janela_ptr);
 scrintf("Preste atenção!");
 pause()

 rmvwindow(&janela_1);
 rmvstat();
}
```

* **menus**

- simples

```
/*
                        menu.c

          Programa de teste do sistema de menus.

     - link: menu + strlst + ioutil + scrnio + popup + smenu
*/

# include "ioutil.h"
# include "scrnio.h"
# include "strlst.h"
# include "smenu.h"

void help(void);

void menu_vertical(void);
void menu_paginado(void);
void menu_retangular(void);

void f_key(int number)
{
 switch(number)
 {
  case 1 : help(); break;
  case 2 : break;
  case 3 : break;
  case 4 : break;
  case 5 : break;
  case 6 : break;
  case 7 : break;
  case 8 : break;
  case 9 : break;
  case 10: break;
 }
}

void main(void)
{
 menudesc horizontal;
 strlist lista_h;

 lista_h=stol("Vertical/Paginado/Retangular/Sair",0);

 initwindows();

 popupmenu(13,3,4,1,50,lista_h,"Menu Horizontal",&horizontal,&monocolors);
 do
 {
  makestat("Selecione com as setas e <RETURN>.                    "
           "<ESC>->Sair    F1->HELP",112);
  movearound(&horizontal);
```

```
  if(horizontal.rcode!=abortit)
   switch(horizontal.ans)
   {
    case 1: menu_vertical();
            break;
    case 2: menu_paginado();
            break;
    case 3: menu_retangular();
            break;
    case 4: horizontal.rcode=abortit;
            break;
   }
 } while(horizontal.rcode!=abortit);
 rmvmenu(&horizontal);
 rmvstat();
}

void menu_vertical(void)
{
 menudesc vertical;
 strlist lista_v;

 lista_v=stol("Opção #1/Opção #2/Opção #3/Opção #4/Opção #5/Opção #6",0);

 popupmenu(9,5,1,6,10,lista_v,"Menu Vertical",&vertical,&monocolors);
 movearound(&vertical);
 rmvmenu(&vertical);
}

void menu_paginado(void)
{
 menudesc paginado;
 strlist lista_p;

 lista_p=stol("Opção #1/Opção #2/Opção #3/Opção #4/Opção #5/Opção #6",0);

 popupmenu(25,5,1,3,10,lista_p,"Menu Paginado",&paginado,&monocolors);
 movearound(&paginado);
 rmvmenu(&paginado);
}

void menu_retangular(void)
{
 menudesc retangular;
 strlist lista_r;

 lista_r=stol("Opção #1/Opção #2/Opção #3/Opção #4/Opção #5/Opção #6",0);

 popupmenu(36,5,2,3,20,lista_r,"Menu Retangular",&retangular,&monocolors);
 movearound(&retangular);
 rmvmenu(&retangular);
}

void help(void)
{
 wptr janela;
```

```
 popupwindow(25,10,26,6,"HELP",overlap,&monocolors,&janela);
 scrintf("Janela de auxílio.\n\nPressione qualquer tecla para voltar.");
 pause();
 rmvwindow(&janela);
}
```

**- pulldown**

```
/*
                        pulldown.c

          Programa de teste do sistema de menus do tipo pulldown.

     - link: menu + strlst + ioutil + scrnio + popup + pdmenu
*/

# include <stdio.h>
# include "ioutil.h"
# include "scrnio.h"
# include "strlst.h"
# include "smenu.h"
# include "pdmenu.h"

void help(void);

void f_key(int number)
{
 switch(number)
 {
  case 1 : help(); break;
  case 2 : break;
  case 3 : break;
  case 4 : break;
  case 5 : break;
  case 6 : break;
  case 7 : break;
  case 8 : break;
  case 9 : break;
  case 10: break;
 }
}

void main(void)
{
 strlist lista, sub_lista_1, sub_lista_2, sub_lista_3;

 lista=stol("Opção 1/Opção 2/Opção 3/Sair",0);
 sub_lista_1=stol("sub_opção 1a/sub_opção 1b/sub_opção 1c/sub_opção 1d",0);
 sub_lista_2=stol("sub_opção 2a/sub_opção 2b",0);
 sub_lista_3=stol("sub_opção 3a/sub_opção 3b/sub_opção 3c",0);

 initwindows();
```

```
 makestat("Selecione com as setas e <RETURN>.                    "
          "<ESC>->Sair    F1->HELP",112);
 pulldwn(13,3,4,50,lista,&monocolors,"Menu Horizontal",
         sub_lista_1,sub_lista_2,sub_lista_3,NULL);
 rmvstat();
}

rtncode action(menudesc *mainmenu, menudesc *submenu)
{
 rtncode rc;

 rc=doit;

 switch(mainmenu->ans)
 {
  case 1: break;
  case 2: break;
  case 3: break;
  case 4: rc=terminate;
          break;
 }
 return rc;
}

void help(void)
{
 wptr janela;

 popupwindow(25,10,26,6,"HELP",overlap,&monocolors,&janela);
 scrintf("Janela de auxílio.\n\nPressione qualquer tecla para voltar.");
 pause();
 rmvwindow(&janela);
}
```

*** árvores**

```
/*
                        arvore.c

          Programa de teste das estruturas tipo árvore-n

     - link: arvore + list + tree
*/

# include <stdio.h>
# include <stdlib.h>
# include <conio.h>
# include <ctype.h>

# include "list.h"
# include "tree.h"
```

```
void main(void)
{
 info_t *raiz=NULL, *tmp;
 char dado[80];

 while(1)
 {
  fflush(stdin);

  printf("\n\n\t\tPrograma teste de árvores\n\n");

  printf("\t\tZ -> inicializa uma árvore\n");
  printf("\t\tP -> imprime a árvore identada\n");
  printf("\t\tM -> procura um elemento\n");
  printf("\t\tF -> adiciona um Filho\n");
  printf("\t\tI -> adiciona um Irmão\n");
  printf("\t\tR -> Remove um elemento\n");
  printf("\t\tE -> rEmove a árvore\n");
  printf("\t\tS -> Saída\n");

  printf("\n\n\t\tSelecione a opção >>> ");

  switch(toupper(getche()))
  {
   case 'Z': if(raiz!=NULL)
             {
              printf("\nA árvore anterior sera destruída !"
                     " Confirme (s/n) >>> ");
              if(toupper(getche())!='S') break;
             }
             printf("\nRaiz >>> ");
             scanf("%s",dado);
             raiz=init_tree(dado);
             break;
   case 'P': printf("\n\n");
             print_i_tree(raiz);
             break;
   case 'M': printf("\nElemento >>> ");
             scanf("%s",dado);
             if((tmp=is_member(dado,raiz))!=NULL)
             {
              printf("\nelemento =\t%s\nnivel =\t\t%d\npai =\t\t%s",
                     tmp->element,tmp->level,tmp->parent->element);
             }
             else printf("\nElemento não encontrado !");
             break;

   case 'F': printf("\nPai >>> ");
             scanf("%s",dado);
             if((tmp=is_member(dado,raiz))==NULL)
             {
              printf("\nNão é membro da árvore !");
              break;
             }
             printf("Elemento >>> ");
             scanf("%s",dado);
```

```
            if(add_child(dado,tmp)) printf("\nOK!");
            else printf("\nErro!");
            break;
   case 'I': printf("\nElemento >>> ");
            scanf("%s",dado);
            if((tmp=is_member(dado,raiz))==NULL)
            {
             printf("\nNão é membro da árvore !");
             break;
            }
            printf("Irmão >>> ");
            scanf("%s",dado);
            if(add_brother(dado,tmp)) printf("\nOK!");
            else printf("\nErro!");
            break;
   case 'R': printf("\nElemento >>> ");
            scanf("%s",dado);
            if(is_member(dado,raiz)==raiz) tmp=NULL;
            else tmp=raiz;
            if(del_node(dado,raiz)) printf("\nOK!");
            else printf("\nNão é membro da árvore !");
            raiz=tmp;
            break;
   case 'E': printf("\nconfirme (s/n) >>> ");
            if(toupper(getche())!='S') break;
            del_tree(raiz);
            raiz=NULL;
            break;
   case 'S': exit(0);
            break;
  }
 }
}
```

*** listas de cadeias de caracteres (mais simples) usadas pelas árvores**

```
/*
                        lista.c

          Programa de teste das listas de cadeias de caracteres usadas pelas
          árvores
*/

# include <stdio.h>
# include <stdlib.h>
# include "list.c"

void main(void)
{
 head_t *lista=NULL;
 node_t *node;
 char dado[80];
 int i;
```

```
while(1)
{
 fflush(stdin);

 printf("\n\nTeste de lista p/ árvore\n\n");
 printf("Exibir uma lista\n");
 printf("Criar uma nova lista\n");
 printf("Adicionar (ao final)\n");
 printf("Inserir (no início)\n");
 printf("Remover (o 1º elemento)\n");
 printf("Sair\n");

 printf("\nopção >>> ");
 switch(toupper(getche()))
 {
  case 'E': if(lista)
            {
             printf("xibir\n\n[");
             for(i=1,node=lista->first; node; i++,node=node->next)
             {
              printf("%d,%s]\n[",i,node->item->element);
             }
             printf("]");
            }
            break;
  case 'C': printf("riar");
            if(lista)
            {
             printf(" <- confirme (s/n) >>> ");
             if(toupper(getche())!='S') break;
            }
            lista=create();
            break;
  case 'A': if(lista)
            {
             printf("dicionar >>> ");
             scanf("%s",dado);
             append(assign(dado),lista);
            }
            break;
  case 'I': if(lista)
            {
             printf("nserir >>> ");
             scanf("%s",dado);
             insert(assign(dado),lista);
            }
            break;
  case 'R': if(lista->length)
            {
             printf("emover <- confirme (s/n) >>> ");
             if(toupper(getche())=='S') delete(lista);
            }
            break;
  case 'S': printf("air <- confirme (s/n) >>> ");
            if(toupper(getche())=='S') exit(0);
```

```
            break;
   default: printf("\a <- opção inválida !");
            break;
  }
 }
}
```

# ANEXO 6

## Descrição detalhada das Funções C dos Sistemas de Listas, Árvores, Janelas e Menus

*** funções relacionadas a árvore:**

**- funções de alto nível**

**- init_tree**

- Propósito: Iniciar uma nova árvore.

- Formato: info_t *init_tree(char *data)

- Parâmetros: data - cadeia de caracteres a ser associada ao nó raíz.

- Valor de retorno: o endereço do nó criado, ou NULL em caso de erro.

- Exemplo: node=init_tree("raíz")

**- add_child**

- Propósito: Adiciona um filho à lista de filhos de um nó.

- Formato: int add_child(char *data,info_t *t_node)

- Parâmetros: data - cadeia de caracteres a ser associada ao nó
t_node - nó onde a operação vai ocorrer

- Valor de retorno: 1 se bem sucedida, 0 caso contrário.

- Exemplo: add_child("filho",node)

**- add_brother**

- Propósito: Adiciona um irmão a um nó, exceto o nó raíz.

- Formato: int add_brother(char *data,info_t *t_node)

- Parâmetros: data - cadeia de caracteres a ser associada ao nó
t_node - nó onde a operação vai ocorrer

- Valor de retorno: 1 se bem sucedida, 0 caso contrário.

- Exemplo: add_brother("irmão",node)

**- leftmost**

- Propósito: Acha o primeiro filho de um nó.

- Formato: node_t *leftmost(info_t *t_node)

- Parâmetros: t_node - o nó a ser considerado

- Valor de retorno: o primeiro nó da lista ligada dos filhos

do nó dado, ou NULL caso contrário.

- Exemplo: filho=leftmost(node)

- **root**

- Propósito: Dado um nó achar a raíz da árvore a que esse nó pertence.

- Formato: info_t *root(info_t *t_node)

- Parâmetros: o nó a ser considerado.

- Valor de retorno: o nó raíz, ou NULL em caso de erro.

- Exemplo: raiz=root(node)

- **is_leaf**

- Propósito: determinar se um nó é, ou não, um nó "folha".

- Formato: int is_leaf(info_t *t_node)

- Parâmetros: t_node - o nó a ser considerado

- Valor de retorno: 1 se o nó for um nó "folha", 0 caso contrário.

- Exemplo: if(is_leaf(node)) return

- **pre_ord**

- Propósito: Percorre a árvore de acordo com o esquema *pré-ordem*, executando uma função selecionada.

- Formato: void pre_ord(void (*do_it)(info_t *t_node), info_t *t_node)

- Parâmetros: do_it(t_node) - função a ser executada no nó
t_node - nó de partida

- Valor de retorno: nenhum.

- Exemplo: pre_ord(print_it_idented,node)

- **in_ord**

- Propósito: Percorre a árvore de acordo com o esquema *em-ordem*, executando uma função selecionada.

- Formato: void in_ord(void (*do_it)(info_t *t_node), info_t *t_node)

- Parâmetros: do_it(t_node) - função a ser executada no nó
t_node - nó de partida

- Valor de retorno: nenhum.

- Exemplo: in_ord(print_it_idented,node)

- **post_ord**

- Propósito: Percorre a árvore de acordo com o esquema *pós-ordem*, executando uma função selecionada.

- Formato: void post_ord(void (*do_it)(info_t *t_node), info_t *t_node)

- Parâmetros: do_it(t_node) - função a ser executada no nó
  t_node - nó de partida

- Valor de retorno: nenhum.

- Exemplo: post_ord(print_it_idented,node)

- **print_it**

- Propósito: Função que exibe um nó de uma árvore.

- Formato: void print_it(info_t *t_node)

- Parâmetros: t_node - o nó a ser exibido

- Valor de retorno: nenhum.

- Exemplo: print_it(node)

- **print_it_idented**

- Propósito: Função que exibe um nó de uma árvore, com identação proporcional à sua profundidade.

- Formato: void print_it_idented(info_t *t_node)

- Parâmetros: t_node - o nó a ser exibido

- Valor de retorno: nenhum.

- Exemplo: print_it_idented(node)

- **del_node**

- Propósito: Função que apaga um nó de uma árvore apagando, também toda a sub-árvore abaixo desse nó.

- Formato: int del_node(char *data,info_t *t_node)

- Parâmetros: data - cadeia de caracteres do nó a ser apagado (para efeito de busca)
  t_node - nó de acesso à árvore (qualquer nó)

- Valor de retorno: 1 se bem sucedida, 0 caso contrário.

- Exemplo: del_node("raiz",node)

- **change_node**

- Propósito: Função que modifica a cadeia de caracteres associada a um determinado nó de uma árvore.

- Formato: info_t *change_node(char *old,char *new, info_t *t_node)

- Parâmetros: old - cadeia de caracteres original
new - nova cadeia de caracteres
t_node - nó de acesso à árvore (qualquer nó)

- Valor de retorno: o nó modificado, ou NULL em caso de erro.

- Exemplo: node=change_node("raiz","nova raiz",node)

- **del_tree**

- Propósito: Remove completamente da memória dinâmica uma árvore (a partir da raiz), liberando toda a memória alocada para tal árvore.

- Formato: void del_tree(info_t *t_node)

- Parâmetros: t_node - nó de acesso à árvore (qualquer nó)

- Valor de retorno: nenhum

- Exemplo: del_tree(node)

- **is_member**

- Propósito: Verifica se um nó é membro de uma árvore.

- Formato: info_t *is_member(char *data,info_t *t_node)

- Parâmetros: data - cadeia de caracteres correspondente ao nó a ser verificado.
t_node - nó de acesso à árvore (qualquer nó)

- Valor de retorno: retorna o nó, caso seja membro, ou NULL caso contrário.

- Exemplo: node=is_member("nova raiz",node)

- **print_i_tree**

- Propósito: Função que exibe uma árvore na tela, sendo cada nó identado de acordo com sua profundidade.

- Formato: void print_i_tree(info_t *t_node)

- Parâmetros: t_node - nó de acesso à árvore (qualquer nó)

- Valor de retorno: nenhum.

- Exemplo: print_i_tree(node)

**- tree2file**

- Propósito: Função que armazena uma árvore completa em um arquivo em disco.

- Formato: int tree2file(char *file_name,info_t *t_node)

- Parâmetros: file_name - nome do arquivo onde a árvore será armazenada
t_node - nó de acesso à árvore

- Valor de retorno: 1 se bem sucedida, e 0 caso contrário.

- Exemplo: print_i_tree("arvore.tre",node)

*** funções internas**

**- inst_info**

- Propósito: instancia um novo nó de uma árvore.

- Formato: info_t *inst_info(int level,char *data,
info_t *parent)

- Parâmetros: level - nível do novo nó
data - cadeia de caracteres do novo nó
parent - nó "pai" do novo nó

- Valor de retorno: o novo nó, ou NULL em caso de erro.

**- delete_it**

- Propósito: função que apaga um nó da árvore.

- Formato: void delete_it(info_t *t_node)

- Parâmetros: t_node - o nó a ser apagado

- Valor de retorno: nenhum.

**- to_file**

- Propósito: função que armazena um nó em um arquivo. É usada pela função *tree2file*.

- Formato: void to_file(info_t *t_node)

- Parâmetros: t_node - o nó a ser armazenado

- Valor de retorno: nenhum.

- **funções de suporte** (lista de cadeias de caracteres)

  - **create**

  - Propósito: Cria uma nova lista de cadeias de caracteres.

  - Formato: head_t *create(void)

  - Parâmetros: nenhum

  - Valor de retorno: ponteiro para a nova lista criada, ou NULL em caso de erro.

  - Exemplo: lista=create()

  **insert**

  - Propósito: Insere um elemento no início de uma lista.

  - Formato: int insert(info_t *data,head_t *list)

  - Parâmetros: data - estrutura de dados correpondente ao elemento a ser inserido (no caso: uma cadeia de caracteres)
    list - lista onde será efetuada a operação

  - Valor de retorno: 1 caso a operação seja bem sucedida, e 0 caso contrário.

  - Exemplo: insert(assign("antes"),lista)

  - **append**

  - Propósito: Adiciona um elemento ao final de uma lista.

  - Formato: int append(info_t *data,head_t *list)

  - Parâmetros: data - estrutura de dados correpondente ao elemento a ser inserido (no caso: uma cadeia de caracteres)
    list - lista onde será efetuada a operação

  - Valor de retorno: 1 caso a operação seja bem sucedida, e 0 caso contrário.

  - Exemplo: append(assign("depois"),lista)

  - **delete**

  - Propósito: Remove o primeiro elemento de uma lista.

  - Formato: info_t *delete(head_t *list)

  - Parâmetros: list - lista onde será efetuada a operação

- Valor de retorno: o elemento removido, caso a operação seja bem sucedida, ou NULL se a lista for vazia.

- Exemplo: delete(lista)

- **inst_node**

- Propósito: Função interna que instancia um novo nó na lista de cadeias de caracteres. Sua chamada só faz sentido no bojo de outra função (p.ex.: append).

- Formato: node_t *inst_node(info_t *val,node_t *ptr)

- Parâmetros: val - ponteiro para a estrutura de dados do nó
ptr - ponteiro para o nó seguinte da lista

- Valor de retorno: o novo nó, ou NULL em caso de erro.

- **assign**

- Propósito: Atribui uma cadeia de caracteres a um elemento de lista.

- Formato: info_t *assign(char *data)

- Parâmetros: data - cadeia de caracteres a ser atribuida

- Valor de retorno: o elemento atribuido, ou NULL em caso de erro.

- Exemplo: append(assign("depois"),lista)

*** funções relacionadas a janelas**

**funções de alto nível**

- **initwindows**

- Propósito: A função **initwindows** inicializa as variáveis internas necessárias para o uso do sistema de janelas. Esta função cria a janela base do sistema (*Base window*), que ocupa toda a tela.

- Formato: int initwindows(void)

- Parâmetros: nenhum

- Valor de retorno: Retorna 1 se o sistema é inicializado com sucesso, e 0 caso contrário.

- Exemplo: initwindows()

- **popupwindow**

- Propósito: Esta função cria uma nova janela em uma posição da tela designada.

- Formato: int popupwindow(int x, int y, char wd, char *n, enum windowtype wt, colorpak *cp, wptr *wnd)

- Parâmetros: x, y - coluna e linha do CSE da janela
  0 ≤ x ≤ 70
  0 ≤ y ≤ 24
  x = -1 ➡ centraliza verticalmente
  y = -1 ➡ centraliza horizontalmente
  x = -2 ➡ usa x corrente
  y = -2 ➡ usa y corrente
  wt, ht - largura e altura da janela
  n - título da janela
  wt - tipo de janela (overlap, etc)
  cp - atributos (cores) da janela (ver figura)
  wnd - endereço da janela criada

- Valor de retorno: retorna 1 se a janela é criada com sucesso, e 0 caso contrário.

- Exemplo: popupwindow(5,5,30,10,"janela",overlap, &defcolors, &wnd)

**Diagrama de cores para colorpak**

| COR | borda | atributos | | | | |
|---|---|---|---|---|---|---|
| | | borda | janela | título | campo | prompt |
| defcolors | dupla | red/black | white/gray | brown/gray | wh/lt.bl | white/gray |
| invcolors | dupla | blk/white | blk/white | blk/white | wh/lt.grn | blk/white |
| monocolors | dupla | white/blk | high inten | high inten | blk/white | white/blk |
| shades | dupla | red/black | white/gray | brown/gray | wh/lt.grn | white/gray |
| errcolors | dupla | wh/lt.red | wh/lt.red | black/red | brn/gray | black/red |
| wrncolors | dupla | brn/lt.red | brn/lt.red | black/red | brn/gray | black/red |
| msgcolors | dupla | wh/lt.grn | wh/lt.grn | brn/lt.grn | blk/white | brn/lt.grn |

- **movewindow**

- Propósito: Move a janela ativa da posição atual para uma outra posição dada.

- Formato: int movewindow(int xoffset, int yoffset, wptr *wnd)

- Parâmetros: xoffset - nova posição x

yoffset - nova posição y
wnd - janela

- Valor de retorno: Retorna 1 se bem sucedida; 0 se não.

- Exemplo: movewindow(-2,10,&wnd)

- **rmvwindow**

- Propósito: Remove uma janela criada por *popupwidow* e todas as janelas abaixo dela na pilha (criadas após ela. A janela ativa antes da sua criação torna-se a nova janela ativa.

- Formato: int rmvwindow(wptr *w)

- Parâmetros: w - janela a ser removida

- Valor de retorno: retorna 1 se a janela for removida com sucesso, e 0 se não.

- Exemplo: rmvwindow(&wnd)

- **selectwindow**

- Propósito: Salva a posição do cursor da janela ativa e seleciona como ativa uma janela dada.

- Formato: int selectwindow(wptr w)

- Parâmetros: w - ponteiro para a janela a ser ativada

- Valor de retorno: se bem sucedida retorna 1, senão 0.

- Exemplo: if(selectwindow(w)) printf("Janela selecionada !")

**funções internas**

- **make_window_node:** Cria um novo ponteiro de janela. Se o novo "nó" for criado com sucesso retorna 1, caso contrário retorna 0.

  Formato: static int make_window_node(wptr *w)

  Parâmetros: w - ponteiro para o ponteiro da janela

- **push_window_node:** Adiciona uma janela à pilha de janelas. Retorna 1 se bem sucedida, 0 caso contrário.

  Formato: static int push_window_node(wptr *w)

  Parâmetros: w - ponteiro para o ponteiro da janela

- **dispose_window_node:** Libera o espaço de memória usado por

uma janela e a destrói.

Formato: static void dispose_window_node(wptr *w)

Parâmetros: w - ponteiro para o ponteiro da janela

- **swap_arrow:** Desenha ou apaga uma seta associada a uma janela. (Usada em janelas tipo "pointer).

  Formato: static void swap_arrow(wptr w)

  Parâmetros: w - ponteiro para a janela

- **swapvram:** Efetua uma troca da imagem da janela com a imagem da área atrás da janela. As imagens são lidas e escritas diretamente na memória de video.

  Formato: void swapvram(wptr w, texelptr b)

  Parâmetros: w - ponteiro para a janela
  b - ponteiro para a imagem a ser trocada

**funções de suporte**

- **setscrptr:** Atualiza o ponteiro da tela. Se vp está situado entre 0 e 3, atribui-se à variável ponteiro de tela (scrptr) o endereço correspondente de uma página de video. Se vp estiver fora deste intervalo, a página corrente de video é obtida através de uma chamada à BIOS e esse endereço é atribuido ao ponteiro de tela.

  Formato: int setscrptr(int vp)

  Parâmetros: vp - seleção de página de video

- **setscrmode:** Seleciona o modo de tela usando uma chamada para a BIOS

  Formato: void setscrmode(int mode)

  Parâmetros: mode - o modo de tela a ser selecionado

- **setcursor:** Define o formato do cursor.

  Formato: void setcursor(int ctop, int cbot)

  Parâmetros: ctop - linha superior do cursor
  cbot - linha inferior do cursor

- **wherexabs:** Obtém a coordenada x do cursor na tela usando a interrupção 10h.

Formato: char wherexabs(void)

Parâmetros: nenhum.

- **whereyabs:** Obtém a coordenada y do cursor na tela usando a interrupção 10h.

Formato: char whereyabs(void)

Parâmetros: nenhum.

- **gotoxyabs:** Posiciona o cursor na posição (x,y) especificada. Se as coordenadas não forem válidas gera um erro fatal.

Formato: void gotoxyabs(char x, char y)

Parâmetros: x,y - posição (coluna e linha) do cursor

- **scrollabs:** Rola a tela para cima ou para baixo um dado número de linhas. Se n > 0 rola n linhas para cima; se n < 0 rola n linhas para baixo; se n = 0 limpa a tela.

Formato: void scrollabs(int n, char x, char y, char x1, char y1, unsigned char a)

Parâmetros: n - número de linhas a rolar
x,y - canto superior esquerdo da tela
x1,y1 - canto inferior direito da tela
a - atributo (para linhas vazias)

- **putchatrabs:** Exibe um caractere com atributo em uma posição da tela especificada.

Formato: void putchatrabs(int x, int y, char c, unsigned char a)

Parâmetros: x,y - posição na tela (coluna,linha)
c - caractere a ser exibido
a - atributo

- **getchatrabs:** Lê um caractere com atributo de uma posição da tela especificada.

Formato: void getchatrabs(int x, int y, char *c, unsigned char *a)

Parâmetros: x,y - posição na tela (coluna,linha)
c - ponteiro para o caractere a ser lido
a - ponteiro para o atributo

- **putstratrabs:** Exibe uma cadeia de caracteres com atributo em uma posição especificada da tela, com um tamanho máximo dado.

Formato: void putstratrabs(int x, int y, char *s,
unsigned char a, int f)

Parâmetros: x,y - posição na tela (coluna,linha)
s - ponteiro para a cadeia de caracteres
a - atributo
f - largura máxima

- **getstratrabs:** Lê uma cadeia de caracteres com atributo em uma posição da tela especificada.

Formato: void getstratrabs(int x, int y, char *s,
unsigned char *a)

Parâmetros: x,y - posição na tela (coluna,linha)
s - ponteiro para a cadeia de caracteres
a - ponteiro para o atributo

- **checkcoord:** Testa a válidade das coordenadas absolutas.

Formato: void checkcoord(int x, int y)

Parâmetros: x,y - coordenadas de tela a serem validadas

- **window:** Atribui as coordenadas da janela ativa às coordenadas internas do sistema.

Formato: void window(char x, char y, char x1, char y1)

Parâmetros: x,y - canto superior esquerdo da janela
x1,y1 - canto inferior direito da janela

- **textcolor:** Devolve a cor de texto atual se (tc==0) ou seleciona a cor de texto se (tc<>0).

Formato: unsigned char textcolor(unsigned char tc)

Parâmetros: tc - cor do texto

- **gotoxy:** Posiciona o cursor em relação às coordenadas da janela ativa.

Formato: void gotoxy(char x, char y)

Parâmetros: x,y - nova posição do cursor

- **wherex:** Devolve a posição x do cursor, em relação à janela ativa. Se as coordenadas do cursor não forem válidas, devolve 0.

Formato: char wherex(void)

Parâmetros: nenhum.

- **wherey** Devolve a posição y do cursor, em relação à janela ativa. Se as coordenadas do cursor não forem váli-

das, devolve 0.

Formato: char wherey(void)

Parâmetros: nenhum.

- **scroll:** Rola n linhas na janela ativa.

Formato: void scroll(int n)

Parâmetros: nenhum.

- **putchatr:** Exibe um caractere, com atributos, em uma posição (x,y) designada na janela ativa.

Formato: void putchatr(int x, int y, char c,
unsigned char a)

Parâmetros: veja putchatrabs.

- **getchatr:** Obtém o caractere, com atributos, de uma posição (x,y) designada na janela ativa.

Formato: void putchatr(int x, int y, char c,
unsigned char a)

Parâmetros: veja getchatrabs.

- **putstratr:** Exibe uma cadeia de caracteres, com atributos, em coordenadas designadas da janela ativa, com um tamanho máximo dado.

Formato: void putstratr(int x, int y, char *s,
unsigned char *a, int f)

Parâmetros: veja putstratrabs.

- **getstratr:** Obtém a cadeia de caracteres, com atributos, das coordenadas designadas da janela ativa.

Formato: void getstratr(int x, int y, char *s,
unsigned char *a)

Parâmetros: veja getstratrabs.

- **clrscr:** Limpa a tela da janela ativa e move o cursor para a posição (0,0) relativa.

Formato: void clrscr(void)

Parâmetros: nenhum.

- **prthz:** Exibe uma cadeia de caracteres horizontalmente com atributos em uma dada posição da janela ativa.

Formato: void prthz(int x, int y, char *s,

unsigned char a, int n)

Parâmetros: x,y - posição na janela ativa
s - ponteiro para a cadeia de caracteres
a - atributo
n - largura máxima

- **wrthz:** Idem prthz; porém caracteres de nova linha (newline) sao interpretados. Além disso, rolamento de tela e truncamento de cadeias de caracteres maiores que a janela sao levados a efeito.

Formato: void wrthz(int x, int y, char *s,
unsigned char a, int n)

Parâmetros: Idem prthz.

- **prtvt:** Exibe uma cadeia de caracteres verticalmente com atributos em uma dada posição da janela ativa.

Formato: void prtvt(int x, int y, char *s,
unsigned char a, int n)

Parâmetros: Idem prthz.

- **prthzf:** Exibe uma cadeia de caracteres verticalmente com atributos em uma dada posição da janela ativa. Análoga a prthz, porém suporta formatação.
**OBS:** a cadeia de caracteres de formato não deve exceder 255 posições.

Formato: void prthzf(int x, int y, char *fmt,
unsigned char attr, int n,...)

Parâmetros: x,y - posição na janela ativa
fmt - cadeia de caracteres de formato
attr - atributo
n - largura máxima

- **scrintf:** Análoga a printf porém a saída é gerada na janela ativa. Suporta rolamento e truncamento.
**OBS:** a cadeia de caracteres de formato não deve exceder 255 posições.

Formato: void scrintf(char *fmt,...)

Parâmetros: fmt - ponteiro para a cadeia de caracteres de formato.

- **box:** Exibe uma "caixa" de tipo especificado em coordenadas absolutas.

Formato: void box(char xul, char yul, char wd, char ht,
char btype, unsigned char a)

Parâmetros: xul, yul - CSE (canto superior esquerdo)

wd, ht - largura e altura
btype - tipo de caixa: 0 -> nenhuma
1 -> simples
2 -> dupla

a - atributo

- **centerstr:** Exibe uma cadeia de caracteres centrada entre coordenadas absolutas, com atributo.

Formato: void centerstr(char xul, char yul, char xlr, char *s, unsigned char a)

Parâmetros: xul, yul - CSE (canto superior esquerdo)
xlr - CID (canto inferior direito)
*s - ponteiro para a cadeia de caracteres
a - atributo

- **crabs:** Envia um sinal de retorno de carro para a página corrente de vídeo.

Formato: static char crabs(char xofs)

Parâmetros: xofs - offset.

- **lfabs:** Envia n sinais de alimentação de linha para a página de vídeo corrente, com coordenadas dadas e atributo.

Formato: static char lfabs(int n, char x, char y, char x1, char y1, unsigned char a)

Parâmetros: n - número de linhas a alimentar
x,y,x1,y1 - posição da janela
a - atributo

- **crlf:** Envia um sinal de retorno de carro e alimentação de linha para a janela ativa.

Formato: void crlf(void)

Parâmetros: nenhum.

*** funções relacionadas a listas de cadeias de caracteres**

**- funções de alto nível**

**- stol**

- Propósito: Converte uma sequencia de cadeias de caracteres separados por uma barra (/) em uma lista de cadeias de caracteres.

- Formato: strlist stol(char *s, int n)

- Parâmetros: s - pointer para a cadeia de caracteres

n - controla o espaço alocado para a cadeia de caracteres. Se for zero, é alocado o tamanho exato; se for menor que o tamanho da cadeia de caracteres, esta é truncada.

OBS: as chaves são geradas sequencialmente.

- Valor de retorno: a lista de cadeias de caracteres criada, ou NULL se houver erro.

- Exemplo: stol("Um/Dois/Três/Quatro/Cinco",0)

**- prtstrlst**

- Propósito: Imprime uma lista de cadeis de caracteres, onde cada elemento consiste de uma cadeia de caracteres, o espaço alocado para ela, além de sua chave. Útil para examinar uma lista.

- Formato: void prtstrlst(strlist strl)

- Parâmetros: strl - pointer p/ a lista de cadeias de caracteres a imprimir.

- Valor de retorno: nenhum.

- Exemplo: prtstrlst(sl)

**- appendstrlst**

- Propósito: Adiciona uma cadeia de caracteres ao fim de uma lista de cadeias de caracteres. Se a lista não existir é criada uma nova lista.

- Formato: int appendstrlst(char *s, int dl, int key, strlist *strl)

- Parâmetros: s - cadeia de caracteres a adicionar
dl - espaço alocado para a cadeia de caracteres; se for 0, para o tamanho exato da cadeia, senão aloca-se dl bytes e a cadeia pode ser truncada.
key - chave para a cadeia de caracteres
strl - ponteiro p/ a lista de cadeias de caracteres

- Valor de retorno: 1 se a operação for bem sucedida. 0 caso contrário.

- Exemplo: appendstrlst("teste", 0, 1, &sl)

**- insstrlst**

- Propósito: Insere uma cadeia de caracteres (e uma chave) na lista de cadeias de caracteres em uma posição especificada. Se essa posição estiver

além do fim da lista, a inserção é feita no final dela. Caso a lista não exista, é criada.

- Formato: int insstrlst(char *s, int dl, int key, int p, strlist *strl)

- Parâmetros: s - cadeia de caracteres a ser inserida
dl - espaço alocado p/ a cadeia de caracteres
key - chave para a cadeia de caracteres
p - posição de inserção da cadeia de caracteres
strl - ponteiro p/ a lista de cadeias de caracteres

- Valor de retorno: 1 se a operação for bem sucedida. 0 caso contrário.

- Exemplo: insstrlst("inserção", 0, 9, 2, &sl)

- **memberstrlst**

- Propósito: Verifica se uma cadeia de caracteres especificada pertence (é membro) de uma dada lista de cadeias de caracteres.

- Formato: int memberstrlst(char *s, strlist strl strlist *msl)

- Parâmetros: s - cadeia de caracteres a "casar"
strl - a lista de cadeias de caracteres
msl - ponteiro de retorno do nó achado

- Valor de retorno: 1 se a operação for bem sucedida. 0 caso contrário.

- Exemplo: memberstrlst("teste", sl, &node)

- **delstrlst**

- Propósito: Apaga uma cadeia de caracteres situada em uma dada posição na lista de cadeias de caracteres.

- Formato: int delstrlst(int p, strlist strl)

- Parâmetros: p - posição para apagamento
strl - a lista de cadeias de caracteres

- Valor de retorno: Se a posição dada não for válida, retorna 0. Caso a operação seja bem sucedida, retorna 1.

- Exemplo: delstrlst(1, sl)

- **concatstrlst**

- Propósito: Reúne duas listas em uma nova lista de cadeias de caracteres.

- Formato: strlist concatstrlst(strlist strl1, strlist strl2)

- Parâmetros: strl1 - a primeira lista a ser concatenada
strl2 - a segunda lista a ser concatenada

- Valor de retorno: a nova lista, que é a concatenação das duas listas dadas, ou NULL se alguma das listas não existir.

- Exemplo: sl=concatstrlst(sl1, sl2)

**- convertstrlst**

- Propósito: Converte uma lista de cadeias de caracteres, de acordo com um código, para maiúsculas, minúsculas ou sem espaços.

- Formato: int convertstrlst(strlist strl, int code)

- Parâmetros: strl - a lista de cadeias de caracteres
code - o código de conversão que pode ser:
0 - todos os caracteres p/ maiúscula
1 - todos os caracteres p/ minúscula
2 - remove todos os espaços em branco

- Valor de retorno: 1 se a operação for bem sucedida; 0 caso contrário.

- Exemplo: convertstrlst(sl, 0)

**- updatekeystrlst**

- Propósito: Atualiza a chave associada a uma cadeia de caracteres pertencente a uma lista.

- Formato: int updatekeystrlst(char *s, int key,strlist strl)

- Parâmetros: s - a cadeia de caracteres a procurar
key - a nova chave
strl - a lista de cadeias de caracteres

- Valor de retorno: Se bem sucedida retorna 1. Caso a cadeia de caracteres não for membro da lista, retorna -1.

- Exemplo: updatekeystrlst("teste", 99, sl)

**- getkeystrlst**

- Propósito: Obter a chave associada a uma determinada cadeia de caracteres pertencente a uma lista.

- Formato: getkeystrlst(char *s, int *i, strlist strl)

- Parâmetros: s - a cadeia de caracteres a procurar
  i - ponteiro de retorno para a chave achada
  strl - a lista de cadeias de caracteres

- Valor de retorno: 1 se a cadeia de caracteres for membro da lista; 0 caso contrário.

- Exemplo: getkeystrlst("teste", &i, sl)

**- lenstrlst**

- Propósito: Determinar o tamanho (número de elementos) de uma lista de cadeias de caracteres.

- Formato: int lenstrlst(strlist strl)

- Parâmetros: strl - a lista de cadeias de caracteres

- Valor de retorno: o tamanho da lista dada.

- Exemplo: len=lenstrlst(sl)

**- makestrlst**

- Propósito: Aloca espaço para a estrutura de uma lista.

- Formato: int makestrlst(strlist *strl)

- Parâmetros: strl - ponteiro (de retorno) p/ a lista de cadeias de caracteres.

- Valor de retorno: 1 se bem sucedida; 0 caso contrário.

- Exemplo: makestrlst(&sl)

**- freestrlst**

- Propósito: Apaga (remove) uma lista de cadeias de caracteres, desalocando todo o espaço de memória utilizado pela lista.

- Formato: int freestrlst(strlist *strl)

- Parâmetros: strl - a lista de cadeias de caracteres

- Valor de retorno: 1 se bem sucedida; 0 caso contrário.

- Exemplo: freestrlst(&sl)

**- clrstrlst**

- Propósito: Limpa uma lista de cadeias de caracteres. Se a lista não existir, cria uma nova lista.

- Formato: void clrstrlst(strlist *strl)

- Parâmetros: strl - a lista de cadeias de caracteres a ser limpa.

- Valor de retorno: nenhum.

- Exemplo: clrstrlst(&sl)

- **funções internas**

  - **makestrnode**

  - Propósito: Aloca espaço (de memória) e inicializa um nó de uma lista de cadeias de caracteres.

  - Formato: int makestrnode(strelem g, strlist *strl)

  - Parâmetros: g - o elemento a armazenar no nó da lista
    strlist - o nó da lista de cadeias de caracteres

  - Valor de retorno: 1 se o nó foi criado com sucesso; 0 caso contrário.

  - Exemplo: makestrnode(elem, &sl)

  - **maxstrelem**

  - Propósito: Determina o tamanho do maior elemento de uma lista de cadeias de caracteres.

  - Formato: int maxstrelem(strlist strl)

  - Parâmetros: strl - a lista de cadeias de caracteres

  - Valor de retorno: o tamanho da maior cadeia de caracteres.

  - Exemplo: len=maxstrelem(sl)

  - **prtstrelem**

  - Propósito: Imprime um elemento de uma lista de cadeias de caracteres. Tal elemento consiste de uma cadeia de caracteres, o espaço alocado para ela e a chave associada.

  - Formato: void prtstrelem(strelem e)

  - Parâmetros: e - o elemento da lista a ser impresso

  - Valor de retorno: nenhum.

  - Exemplo: prtsrtelem(sl->dn.elem)

- **cmpstrelem**

- Propósito: Compara dois elementos de lista de cadeias de caracteres.

- Formato: int cmpstrelem(strelem e, strelem g)

- Parâmetros: e, g - os elementos a serem comparados

- Valor de retorno: Se as cadeias de caracteres dos dois elementos forem iguais, retorna 1; caso contrário retorna 0.

- Exemplo: comp=cmpstrelem(sl1->dn.elem, sl2->dn.elem)

- **delstrelem**

- Propósito: Apaga um elemento de lista de cadeia de caracteres.

- Formato: void delstrelem(strelem e)

- Parâmetros: e - o elemento a ser apagado

- Valor de retorno: nenhum

- Exemplo: delstrelem(sl->dn.elem)

- **makestrelem**

- Propósito: Cria um elemento de lista de cadeia de caracteres, seguindo as regras descritas em *appendstrlst*.

- Formato: strelem makestrelem(char *s, int dl, int key)

- Parâmetros: s - cadeia de caracteres a ser armazenada
dl - espaço alocado para a cadeia
key - a chave associada à cadeia

- Valor de retorno: o elemento criado, ou NULL caso contrário.

- Exemplo: elem=makestrelem("teste", 0, 1)

- **getstrptr**

- Propósito: Retorna um ponteiro para um elemento (da lista de cadeias de caracteres) indicado.

- Formato: strlist getstrptr(int i, strlist strl)

- Parâmetros: i - posição do elemento na lista
strl - a lista de cadeias de caracteres

- Valor de retorno: O elemento indicado, ou NULL caso o in-

dice esteja fora do intervalo de valores válidos.

- Exemplo: sl1=getstrptr(1, sl)

*** funções relacionadas a menus**

**- funções de alto nível**

**- popupmenu**

- Propósito: Exibe um menu em uma dada região da tela. Os parâmetros do menu são calculados e armazenados em uma estrutura correspondente ao menu. Para controlar o movimento da barra de seleção deve-se utilizar a função **movearound.**

- Formato: void popupmenu(int x, int y, int nc, int nr, int hze, strlist sl, char *t, menudesc *sm, colorpack *cp)

- Parâmetros: x, y - canto superior esquerdo do menu
nc, nr - largura e altura do menu
hze - extensão horizontal
sl - a lista de ítens do menu
t - o título do menu
sm - ponteiro para a estrutura do menu
cp - ponteiro para a estrutura de atributos do menu

- Valor de retorno: nenhum

- Exemplo: popupmenu(1,1,1,3,6,stol("um/dois/três",0), "Menu",&mnu,&monocolors)

**- moveaound**

- Propósito: Controlar o movimento da barra de seleção dentro do menu. Esta função deve ser invocada após a crição de um menu. Reconhece as teclas das setas, PGUP e PGDN, HOME e END. Devolve um código de retorno, conforme o usuário tecle <RETURN> ou <ESC> para sair do loop.

- Formato: void movearound(menudesc *sm)

- Parâmetros: sm - ponteiro para a estrutura do menu

- Valor de retorno: nenhum

- Exemplo: movearound(&mnu)

**- rmvmenu**

- Propósito: Remove o menu especificado. Para isto, a

janela associada ao menu é removida da pilha de janelas e a lista de ítens do menu é apagada. Para manter a lista de ítens para posterior uso, pode-se remover o menu chamando *rmvwindow* diretamante.

- Formato: void rmvmenu(menudesc *sm)

- Parâmetros: sm - ponteiro para a estrutura do menu

- Valor de retorno: nenhum

- Exemplo: rmvmenu(&mnu)

**- pulldwn**

- Propósito: Cria um sistema de menus pull-down usando as funções de janelas e menus. Esse sistema consiste de um menu principal horizontal e sub-menus opcionais para cada entrada do menu principal. Deve-se definir uma ação (função *action*) para cada item que não contenha sub-menu.

- Formato: void pulldwn(int x, int y, int nc, int hze,
strlist msl, colorpak *cp,
char *title, ...)

- Parâmetros: x, y - canto superior esquerdo do menu
nc - largura do menu
hze - extensão horizontal
msl - a lista de ítens do menu
cp - ponteiro para a estrutura de atributos do menu
title - o título do menu
... - argumentos variáveis p/sub-menus

- Valor de retorno: nenhum

- Exemplo: pulldwn(1,1,3,80,stol("um/dois/tres",0),
&monocolors,"PD-Menu",stol("a/b",0),
NULL,stol("c/d",0))

**- funções internas**

**- displist**

- Propósito: Exibe a lista de opções do menu na janela correspondente. É chamada quando da criação e da "rolagem" (scrolling) de um menu.

- Formato: void displist(menudesc *sm)

- Parâmetros: sm - ponteiro p/ a estrutura do menu

- Valor de retorno: nenhum

- Exemplo: displist (&mnu)

- **menusize**

- Propósito: Determina o tamanho físico real de um menu. A largura do menu é devolvida através do parâmetro *wd.

- Formato: void menusize(int nc, int hze, strlist sl, char *t, menudesc *sm, int *wd)

- Parâmetros: nc, nr - largura e altura do menu
hze - extensão horizontal
sl - lista de ítens do menu
t - título do menu
sm - ponteiro p/ a estrutura do menu
wd - ponteiro p/ a largura calculada do menu

- Valor de retorno: nenhum

- Exemplo: menusize(2,2,10,stol("a/b/c",0),"menu",&mnu,&wd)

- **getmenuparms**

- Propósito: Calcula os parâmetros exatos do menu, inclusive tamanho e posição. Os parâmetros são armazenados na estrutura do menu.

- Formato: static void getmenuparms(int nc, int nr, int hze, strlist sl, menudesc *sm)

- Parâmetros: nc, nr - largura e altura do menu
hze - extensão horizontal
sl - lista de ítens do menu
sm - ponteiro para a estrutura do menu

- Valor de retorno: nenhum

- Exemplo: getmenuparms(2,2,10,stol("a/b/c",0),&mnu)

- **adjsize**

- Propósito: Ajusta o tamanho de um menu. Atualiza a extensão horizontal, número de colunas e linhas, a área e o espaçamento do menu.

- Formato: static void adjsize(menudesc *sm, int nc, int nr)

- Parâmetros: sm - ponteiro p/ a estrutura do menu
nc, nr - número de linhas e colunas do menu

- Valor de retorno: nenhum

- Exemplo: adjsize(&mnu,2,2)

- **isvmenu**

- Propósito: Verifica se um menu é do tipo vertical (contem apenas uma coluna e sua lista de opçöes se encaixa exatamente no menu).

- Formato: static int isvmenu(menudesc *sm)

- Parâmetros: sm - ponteiro p/ a estrutura do menu

- Valor de retorno: retorna 1 se o menu for vertical e 0 caso contrário.

- Exemplo: isvmenu(&mnu)

- **ishmenu**

- Propósito: Verifica se um menu é do tipo horizontal (contem apenas uma linha e sua lista de opçöes se encaixa exatamente no menu).

- Formato: static int ishmenu(menudesc *sm)

- Parâmetros: sm - ponteiro p/ a estrutura do menu

- Valor de retorno: retorna 1 se o menu for horizontal e 0 caso contrário.

- Exemplo: ishmenu(&mnu)

- **isrectmenu**

- Propósito: Verifica se um menu é do tipo retangular (pode ter mais de uma linha ou coluna, porém a lista de opçöes deve se encaixar exatamente no menu).

- Formato: static int isrectmenu(menudesc *sm)

- Parâmetros: sm - ponteiro p/ a estrutura do menu

- Valor de retorno: retorna 1 se o menu for retangular e 0 caso contrário.

- Exemplo: isrectmenu(&mnu)

- **sumchar**

- Propósito: Determina a largura de um menu através do cálculo do número de caracteres da lista de opçöes do menu. Este total é armazenado no componente *maxchar* da estrutura do menu.

- Formato: static void sumchar(menudesc *sm)

- Parâmetros: sm - ponteiro p/ a estrutura do menu

- Valor de retorno: nenhum

- Exemplo: sumchar(&mnu)

- **space**

- Propósito: Determina o espaçamento de um menu.

- Formato: static int space(menudesc *sm)

- Parâmetros: sm - ponteiro p/ a estrutura do menu.

- Valor de retorno: o espaçamento calculado.

- Exemplo: space(&mnu)

- **colx**

- Propósito: Determina a coordenada x de uma coluna especificada em um menu.

- Formato: static int colx(int colno, menudesc *sm)

- Parâmetros: colno - o número da coluna
sm - ponteiro p/ a estrutura do menu

- Valor de retorno: a coordenada x calculada.

- Exemplo: colx(2, &mnu)

- **barlocx**

- Propósito: Determina a coordenada x para uma posição especificada da barra de seleção.

- Formato: static int barlocx(int i, menudesc *sm)

- Parâmetros: i - o item especificado do menu
sm - ponteiro p/ a estrutura do menu

- Valor de retorno: a coordenada x calculada.

- Exemplo: barlocx(1, &mnu)

- **barlocy**

- Propósito: Determina a coordenada y para uma posição especificada da barra de seleção.

- Formato: static int barlocy(int i, menudesc *sm)

- Parâmetros: i - o item especificado do menu
sm - ponteiro p/ a estrutura do menu

- Valor de retorno: a coordenada y calculada.

- Exemplo: barlocy(1, &mnu)

- **dispbar**

- Propósito: Exibe uma barra de seleção no item especificado do menu. A barra de seleção é exibida com um atributo de cor para indicar a posição corrente do cursor no menu.

- Formato: void dispbar(int i, int attr, menudesc *sm)

- Parâmetros: i - item do menu a selecionar
attr - atributo da barra de seleção
sm - ponteiro p/ a estrutura do menu

- Valor de retorno: nenhum

- Exemplo: dispbar(1, 112, &mnu)

- **homebar**

- Propósito: Processa a tecla *home*. Quando essa tecla é pressionada, a primeira página de ítens do menu é exibida.

- Formato: static void homebar(int *i, menudesc *sm)

- Parâmetros: i - ponteiro p/ o item selecionado do menu
sm - ponteiro p/ a estrutura do menu

- Valor de retorno: nenhum

- Exemplo: homebar(&item, &mnu)

- **endbar**

- Propósito: Processa a tecla *end*. Quando essa tecla é pressionada, a última página de ítens do menu é exibida.

- Formato: static void endbar(int *i, menudesc *sm)

- Parâmetros: i - ponteiro p/ o item selecionado do menu
sm - ponteiro p/ a estrutura do menu

- Valor de retorno: nenhum

- Exemplo: endbar(&item, &mnu)

- **pgupbar**

- Propósito: Processa a tecla *pg up*. Quando essa tecla é pressionada, a página anterior de ítens do menu é exibida.

- Formato: static void pgupbar(int *i, menudesc *sm)

- Parâmetros: i - ponteiro p/ o item selecionado do menu
sm - ponteiro p/ a estrutura do menu

- Valor de retorno: nenhum

- Exemplo: pgupbar(&item, &mnu)

- **pgdnbar**

- Propósito: Processa a tecla *pg down*. Quando essa tecla é pressionada, a página seguinte de ítens do menu é exibida.

- Formato: static void pgdnbar(int *i, menudesc *sm)

- Parâmetros: i - ponteiro p/ o item selecionado do menu
sm - ponteiro p/ a estrutura do menu

- Valor de retorno: nenhum

- Exemplo: pgdnbar(&item, &mnu)

- **upbar**

- Propósito: Processa a tecla *up arrow* [↑]. Quando essa tecla é pressionada, o item anterior do menu é destacado pela barra de seleção.

- Formato: static void upbar(int *i, menudesc *sm)

- Parâmetros: i - ponteiro p/ o item selecionado do menu
sm - ponteiro p/ a estrutura do menu

- Valor de retorno: nenhum

- Exemplo: upbar(&item, &mnu)

- **downbar**

- Propósito: Processa a tecla *down arrow* [↓] Quando essa tecla é pressionada, o próximo item do menu é destacado pela barra de seleção.

- Formato: static void downbar(int *i, menudesc *sm)

- Parâmetros: i - ponteiro p/ o item selecionado do menu
sm - ponteiro p/ a estrutura do menu

- Valor de retorno: nenhum

- Exemplo: downbar(&item, &mnu)

**- leftbar**

- Propósito: Processa a tecla *left arrow* [←]. Quando essa tecla é pressionada, o item na coluna anterior do menu é destacado pela barra de seleção.

- Formato: static void leftbar(int *i, menudesc *sm)

- Parâmetros: i - ponteiro p/ o item selecionado do menu
  sm - ponteiro p/ a estrutura do menu

- Valor de retorno: nenhum

- Exemplo: leftbar(&item, &mnu)

**- rightbar**

- Propósito: Processa a tecla *right arrow* [→]. Quando essa tecla é pressionada, o item na próxima coluna do menu é destacado pela barra de seleção.

- Formato: static void rightbar(int *i, menudesc *sm)

- Parâmetros: i - ponteiro p/ o item selecionado do menu
  sm - ponteiro p/ a estrutura do menu

- Valor de retorno: nenhum

- Exemplo: rightbar(&item, &mnu)

**- firstchar**

- Propósito: Compara um caractere de entrada com o primeiro caractere de cada um dos ítens do menu. Se ocorre um casamento, a função retorna o índice do item no menu; senão, retorna 0.

- Formato: static int firstchar(char c, menudesc *sm)

- Parâmetros: c - caractere a "casar"
  sm - ponteiro p/ a estrutura do menu

- Valor de retorno: índice do item no menu.

- Exemplo: firstchar('A', &mnu)

**- adji**

- Propósito: Determina o número de items do menu que são estão visíveis.

- Formato: static void adji(menudesc *sm, int *i)

- Parâmetros: sm - ponteiro p/ a estrutura do menu
  i - ponteiro p/ o contador de ítens do menu

- Valor de retorno: nenhum

- Exemplo: adji(&mnu, &count)

# Anexo 7

## Procedimentos para a criação de uma nova Base de Dados

## 1) Introdução

Este anexo é dedicado à descrição dos procedimentos a serem adotados para a criação de um novo Sistema de Auxílio a Projetos.

## 2) Base de Dados

O primeiro passo para a criação de um Sistema de Auxílio a Projetos consiste em definir a estrutura da Base de Dados que servirá de alicerce para o sistema.

Essa estrutura compreende: a estrutura dos registros que irão compor os arquivos da Base de Dados, os arquivos em si, seu tamanho, as estratégias de busca e outros parâmetros que passamos a descrever a seguir.

### 2.1) Parâmetros de instalação da Base de Dados

Antes de poder-se efetivamente estabelecer a estrutura de uma Base de Dados, o BAPAS-DB requer que se estabeleçam certos parâmetros de instalação, utilizando-se o programa de Instalação de Bases de Dados **INSTAL**, que cria um "esqueleto" de Base de Dados que será preenchido posteriormente por registros, arquivos e estratégias.

Esse programa requisita, do usuário, alguns dados que tanto podem ser fornecidos interativamente, como em lote, através de um arquivo de "response". A seguir, apresentamos os parâmetros requisitados.

**- MANAGEMENT-DATA ON CURRENT DIRECTORY ? (Y/N) :**

"A Base de Dados será criada no diretório atual ?". A esta pergunta devemos sempre responder **Y** (sim).

**- MANAGEMENT-DATA ON GESDAT_VW ? (Y/N) :**

"O nome da Base de Dados será o padrão (GESDAT_VW) ?". Também devemos responder **Y** (sim).

**- MAXIMUM OF MANAGEMENT-FILES :**

"Número máximo de Bases de Dados ?". No nosso caso devemos indicar que será uma Base de Dados única, ou seja: responderemos **1**.

**- MAXIMUM OF MANAGEMENT-AREA :**

"Tamanho da área de gerência". Depende do tamanho da Base de Dados que iremos criar. Um múltiplo de 1024 (1 Kbyte) deve ser a resposta.

**- MAXIMUM OF DB-FILES :**

"Número máximo de arquivos da Base de Dados". Define a quantidade máxima de arquivos que a Base de Dados comportará (p.ex.: 20).

**- DO YOU WISH TO GENERATE BUFFERSYSTEMS ? (Y/N) :**

"Você quer criar sistemas de "buffers" ?". Se respondermos **Y** (sim), poderemos criar estruturas de armazenamento temporário na memória dinâmica que atuarão como "cache" do Banco de Dados, e serão feitas as perguntas a seguir:

**- NAME OF BUFFERSYSTEM :**

"Nome do sistema de buffers". Devemos fornecer um nome (6 caracteres) ao sistema de buffers sendo criado, exceto no caso do primeiro, que possui o nome padrão STANDA.

**- MAXIMUM OF BUFFERS IN BUFFERSYSTEM :**

"Quantidade de páginas do sistema de buffers". Deve-se indicar o número máximo de páginas do buffer que residirão na memória simultaneamente.

**- LENGTH OF PAGE IN BUFFERSYSTEM :**

"Comprimento de página do sistema buffers". Indica-se o tamanho de cada página do sistema de buffers.

**- DO YOU WISH TO CREATE ONE MORE BUFFERSYSTEM ? (Y/N) :**

"Você quer criar mais um sistema de buffers ?". Se respondermos **Y** (sim) as três últimas questões serão repetidas para a criação de outro sistema de buffers. Caso contrário, se respondermos **N** (não), o programa de instalação gera o arcabouço da Base de Dados baseando-se nos parâmetros fornecidos.

## 2.2) Definição da Base de Dados

Uma vez criado o "esqueleto" de Base de Dados pelo programa INSTAL descrito acima, pode-se proceder agora a definição da estrutura da Base de Dados propriamente dita, ou seja: seus registros, arquivos e estratégias.

Para efetuar a definição da estrutura da Base de Dados utiliza-se a Linguagem de Descrição de Dados (DDL) oferecida pelo BAPAS-DB, que, tal como o programa INSTAL, pode ser utilizada de maneira interativa ou em lote, através de arquivos de "response".

O ponto de partida para a definição da estrutura de uma Base de Dados é a definição dos registros e, também, das estratégias de busca que serão utilizados nos arquivos constituintes dela.

Com esses elementos definidos podemos, então, definir os arquivos que comporão a Base de Dados.

A seguir, descreveremos mais detalhadamente esses procedimentos.

### 2.2.1) Definição dos registros da Base de Dados

Os registros são as peças fundamentais dos arquivos de uma Base de Dados. No nosso caso, cada arquivo é formado por um único tipo de registro,

porém mais de um arquivo pode utilizar o mesmo tipo de registro.

Para que seja definido um novo tipo de registro, utiliza-se a palavra chave **NREC** (New Record ou, em português: Novo Registro) e definem-se os componentes do registro, como abaixo:

```
NREC =    identificador,
          STRUCT [ campo    tipo,
                   ...      ] ;
```

onde: *identificador* - é o nome do registro a ser criado,
*campo* - é o nome do campo,
*tipo* - é o tipo do campo, que pode ser:

- FIXED (precisão),
- FLOAT (precisão),
- BIT (length),

} equivalentes a **int** em linguagem **C**

- CHAR (length), → equivalente a **char**[] em C
- CLOCK,
- DURATION.

} não utilizados (normalmente)

- outro registro pré-definido

Por exemplo:

```
NREC =    valor
          STRUCT [  cruzeiros      FIXED(10),
                    centavos       FIXED(2) ];

NREC =    artigo
          STRUCT [  nome           CHAR(20),
                    numero         FIXED(5),
                    preco          valor,
                    quantidade     FIXED(4) ];
```

Para mais exemplos de definições de registros veja, nos anexos, a listagem do arquivo DDLREC (arquivo de "response" pertencente ao sistema NETCAD).

### 2.2.2) Definição das estratégias de busca da Base de Dados

A cada arquivo da Base de Dados que desejemos definir, devemos associar uma estratégia de busca, para que possam ser efetuados acessos a seus registros.

Veja no Capítulo 2 a descrição das estratégias de busca implementadas pelo BAPAS-DB e, para modelo das definições de estratégias veja, nos anexos, a listagem do arquivo DDLSTR (arquivo de "response" pertencente ao sistema NETCAD), que apresenta as definições de todas as estratégias disponíveis na versão por nós utilizada do BAPAS-DB.

Note, porém, que não é necessário que se definam todas as estratégias simultaneamente. Pode-se estabelecer somente as estratégias que serão efetivamente utilizadas pelos os arquivos da Base de Dados que estejamos implementando.

### 2.2.3) Definição dos arquivos da Base de Dados

Agora que já foram estabelecidos os registros e estratégias que serão utilizados, podemos passar à definição dos arquivos da Base de Dados, pois os registros, por si próprios, não armazenam informação alguma, pois são apenas estruturas "lógicas". As informações são armazenadas, isto sim, nos arquivos da Base de Dados que utilizam esses registros (e estratégias associadas).

A palavra-chave que inicia a definição de um arquivo da Base de Dados é **NDBF** (New Database File), ou novo arquivo da Base de Dados, em português.

Veja, a seguir, a estrutura da definição de um arquivo genérico da Base de Dados, atentando ao detalhe de que alguns parâmetros são opcionais:

```
NDBF =    identificador,
          FILE=identificador_bd / VOL=volume,
          REC=registro,
          BUFFER=buffer,
          HEADER,                    } parâmetros opcionais
          STRAT=parâmetros_da_estratégia ;
```

onde:

*identificador* - é o nome lógico do arquivo a ser criado,

*identificador_bd* - é um nome a ser usado internamente,

*volume* - indica o volume máximo de registros no arquivo,

*registro* - é o nome do tipo de registro a ser utilizado,

*buffer* - nome do sistema de buffers (default=STANDA),

HEADER - 2 bytes extra; usados sob certas cirscunstâncias,

*parâmetros_da_estratégia* - parâmetros referentes à estratégia.

Veja o exemplo a seguir, onde consideramos que já tenham sido definidos o registro artigo, como no exemplo anterior, e também a estratégia BBAUM (B-tree, ou em português: árvore binária), tal como descrito no arquivo DDLSTR constante dos anexos.

```
NDBF =    artigos,
          FILE=d_artigo / VOL=50,
          REC=artigo,
          STRAT=BBAUM    /KEYLGT=20
                         /KEYPOS=1
                         /KEYTYP=4
                         /SYNKEY=1
                         /KEY=[ nome ] ;
```

**- Definição do volume máximo de registros em cada arquivo**

Após a palavra-chave FILE, deve-se fornecer um nome de arquivo, o qual

vai ser utilizado apenas internamente e, na mesma linha, após a palavra-chave VOL, deve-se indicar o número máximo de registros armazenáveis no arquivo; no exemplo, esse número é de 50 registros.

**- Definição do registro para cada arquivo**

Embora um tipo de registro possa ser utilizado por mais de um arquivo da Base de Dados, cada arquivo deve utilizar um único tipo de registro. No exemplo anterior, a definição do tipo de registro a ser utilizado no arquivo, é fornecida após a palavra-chave REC; neste caso o registro é o artigo, definido em um exemplo anterior.

**- Definição da estratégia de busca para cada arquivo**

Cada arquivo da Base de Dados deve ter associado a si uma estratégia de busca que estabeleça uma ordem nos acessos às informações armazenadas no arquivo; no caso do exemplo anterior a estratégia é B-tree.

Deve-se, ainda, fornecer os parâmetros que sejam requeridos pela estratégia como no exemplo anterior.

Veja nos anexos, mais exemplos de definição de arquivos da Base de Dados na listagem do arquivo NETCAD.DDL, pertencente ao sistema Netcad.

## 2.3) Os arquivos de "response" para a criação da Base de Dados

A utilização dos programas INSTAL e DDL, do BAPAS-DB, de forma interativa é perfeitamente possível, porém na maioria das situações é recomendável que sejam construídos arquivos de lote ("response") que forneçam as informações necessárias a esses programas, utilizando a característica de redirecionamento de entrada do sistema operacional.

Assim, se criarmos um arquivo desse tipo para a instalação da Base de Dados e o denominarmos INST_BD.RSP, podemos invocar o programa INSTAL da seguinte maneira:

**INSTAL < INST_BD.RSP**

Para a definição de registros, estratégias e arquivos pode-se, também, criar arquivos de lote correspondentes e invocar o programa DDL assim:

**DDL** < *arquivo_lote*

## 3) Sistema propriamente dito

### 3.1) O arquivo "cabeçalho" (header)

Deve-se, para cada arquivo da Base de Dados que queiramos acessar, criar uma estrutura (no sentido da linguagem C) correspondente ao tipo de registro que o arquivo utiliza. Essa estrutura será utilizada como um "buffer" nos acessos do programa ao Banco de Dados.

Essas estruturas serão, tipicamente, definidas em um arquivo "header" que será incluído no programa em tempo de compilação.

Por exemplo, levando em conta o exemplo anterior, criaríamos uma estrutura C do seguinte tipo:

```
struct valor   {    int          cruzeiros;
                    int          centavos;      } valor;

struct artigos {    TYPE_FILE    file;
                    char         nome[20];
                    int          número;
                    struct valor preço;
                    int          quantidade;    } artigos;
```

Além dessas estruturas, esse arquivo "header" irá conter declarações de variáveis que sejam necessárias ao intercâmbio programa/BAPAS-DB, por exemplo a variável *file* do exemplo anterior.

Veja na listagem do arquivo DATABASE.H, referente ao sistema Netcad, constante dos anexos, um modelo de arquivo "header".

### 3.2) Arquivos a serem "incluídos" (includes)

Para que se possam utilizar certas facilidades que descrevemos, faz-se necessário que sejam "incluídos" em tempo de compilação os arquivos header correspondentes no Sistema de Auxílio a Projetos a ser criado.

A seguir, temos uma descrição desses arquivos:

**- referentes à Base de Dados**

*icdbv.inc* - contém definições de variáveis, tipos e protótipos[1] necessários ao funcionamento da interface DBV - entre programa e Base de Dados;

**- referentes à Interface de Janelas e Menus**

*ioutil.h* - funções de entrada e saída utilizadas na interface;

*scrnio.h* - necessário para se criar janelas simples;

*strlst.h* - para a criação de listas de cadeias de caracteres, as quais também são usadas na criação de menus;

---

[1] prototypes, conforme definido no ANSI C

*smenu.h* - necessário para se criar menus simples;

*pdmenu.h* - necessário para se criar menus tipo "pull-down"[1];

**- referentes a outros utilitários**

*tools.h* - ferramentas que podem ser invocadas de dentro do programa em tempo de execução;

*list.h* - possibilita a criação de listas de dados complexos (p.ex.: cadeias de caracteres);

*tree.h* - para poder-se criar estruturas de dados tipo "árvore" (associado a *list.h*).

### 3.3) Arquivos a serem "linkados"[2]

Além dos arquivos "header" serem incluídos na compilação do Sistema a ser criado, para a efetiva utilização das facilidades descritas, existe a necessidade de adicionar-se arquivos-objeto pré-compilados e, talvez, algumas bibliotecas ao fazer-se a "linkagem" do sistema para obter-se um programa executável.

A seguir uma lista desses arquivos:

#### 3.3.1) Objetos

**- Referentes à Base de Dados**

---

[1] possíveis traduções para este termo seriam: tipo "cortina" ou "puxados"

[2] optamos por utilizar o anglicismo "linkagem" por falta de uma palavra em português que exprima a idéia associada a essa operação de maneira satisfatória. (ligação? montagem? enlaçamento?)

*dbstrtta.obj* - estratégias do BAPAS-DB;

*user_sta.obj* - pilha disponível ao usuário no acesso à Base de Dados;

*pearlst.obj* - inicializa o sistema operacional PEARL, em cima do qual o BAPAS-DB roda;

*dbverr.obj* - mensagens de erro do BAPAS-DB;

**- Referentes à Interface Gráfica**

*ioutil.obj* - funções de entrada e saída de dados utilizadas pela interface de janelas e menus;

*scrnio.obj* - funções de suporte à criação de janelas;

*popup.obj* - funções para a criação e manipulação de janelas. Utiliza funções dos módulos anteriores;

*strlst.obj* - funções que implementam listas de cadeias de caracteres. São utilizadas pelos menus;

*smenu.obj* - funções para a criação e gerenciamento de menus simples;

*pdmenu.obj* - funções para a criação e gerenciamento de menus tipo "pull-down".

**- Referentes a outros utilitários**

*tools.obj* - funções que implementam utilidades relacionadas ao sistema operacional, as quais podem ser invocadas através de menus durante a execução do programa;

*list.obj* - funções que implementam listas de estruturas complexas (definíveis pelo usuário);

*tree.obj* - funções que implementam e gerenciam estruturas do tipo árvore (usando as listas acima).

### 3.3.2) Bibliotecas

**- Referentes à Base de Dados**

*runtime.lib* - biblioteca de funções necessárias em tempo de execução ao BAPAS-DB;

*c_dbvlib.lib* - biblioteca de funções que implementam a interface DBV entre programas e o BAPAS-DB.

**- Referentes ao compilador C**

*llibc7.lib* - biblioteca referente à linguagem C - para o modelo de memória "large" (grande) e suporte a operações em ponto flutuante.

### 3.3.3) Utilização de um arquivo de "response"

Para facilitar a tarefa de "linkar" todos os arquivos objeto e bibliotecas necessários à criação de um programa executável, é interessante que se crie um arquivo de "response" (resposta) contendo uma lista dos arquivos necessários, além de opções especiais para a formação do código executável.

Assim, uma vez criado tal arquivo (vamos, por exemplo, chamá-lo:

*response.rsp*), pode-se efetuar a "linkagem" simplesmente invocando o linker com o arquivo de "response" como argumento:

**LINK** @*response.rsp*

Veja nos anexos a listagem do arquivo "response" *l_doscdb.rsp* pertencente ao sistema Netcad para modelo.

## 4) Macetes & Dicas

Vamos, aqui, apresentar alguns "truques", regras práticas, e "bugs" descobertos após exaustivas análises dos manuais, dos exemplos de utilização, dos testes e depurações do BAPAS_DB e das demais ferramentas descritas que serão utilizadas como base para a implementação de um Sistema de Auxílio a Projetos.

### 4.1) Compilador C

O compilador C que tem compatibilidade assegurada com o sistema BAPAS-DB é o compilador Microsoft C[1]. Outros compiladores dessa linguagem podem, a princípio, ser utilizados para a criação do Sistema de Auxílio a Projetos a ser integrado com o BAPAS-DB, porém garantia de integração isenta de problemas só é oferecida a esse compilador em particular.

### 4.2) Considerações de Hardware

O sistema BAPAS-DB foi desenvolvido em PEARL, que é uma linguagem de

---

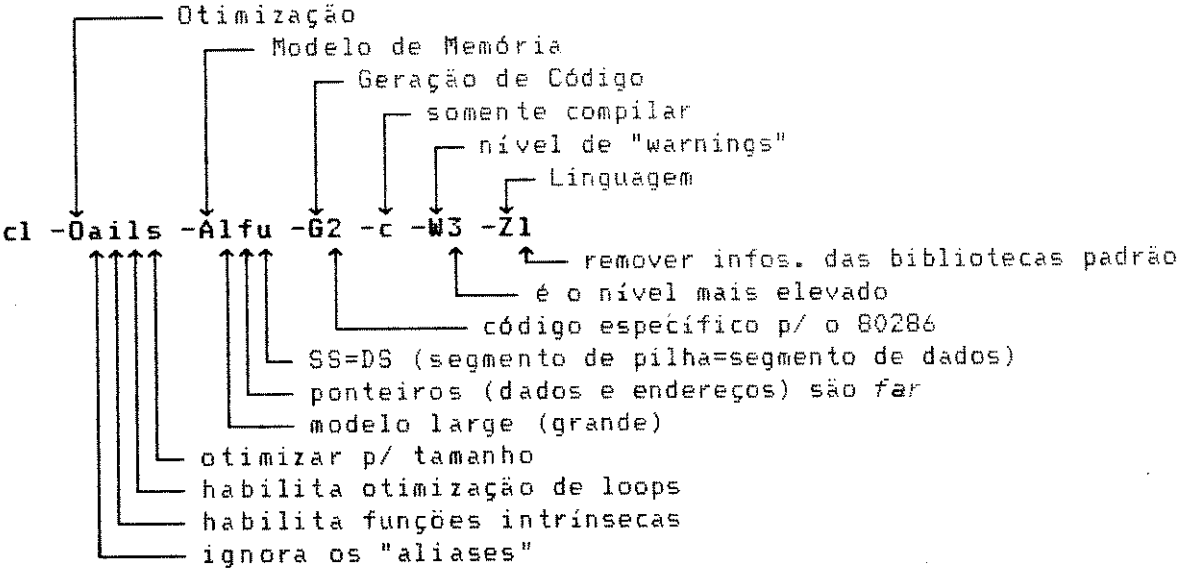
[1] Microsoft C é marca registrada da Microsoft Corporation

programação com características de tempo-real. Ocorre que tal linguagem exige, para sua execução, que seja "carregado" na memória o sistema operacional PEARL, associado a ela. Esse sistema operacional impõe algumas restrições no hardware do computador onde vai operar. No caso específico de computadores PC[1], onde implementamos o nosso estudo de caso, a exigência é de que o processador seja, no mínimo, um AT[2] ( 286 ) dotado de co-processador aritmético.

O co-processador pode, no entanto, ser "emulado" em sistemas que não o possuam. Existe, por exemplo, um programa de emulação do co-processador 8087 denominado: EM87.COM[3], testado por nós com sucesso na utilização do sistema.

## 4.3) Opções do compilador

A linha de comando sugerida para a compilação de um Sistema de Auxílio a projetos sendo criado é a seguinte:




---

[1,2] PC e AT são marcas registradas da IBM Corporation
[3] copyright de Ron Kinball

## 4.4) Opções do "linker"

Ao ser efetuada a "linkagem" do programa de um Sistema de Auxílio a Projetos que utilize chamadas ao BAPAS-DB, devem ser selecionadas algumas opções específicas do linker, quais sejam:

**lvarstck.obj** - rotinas que permitem que a área de dados do programa utilize, dinamicamente, a área não usada da pilha (stack) alocada para o programa (mas não vice-versa);

**/E(XEPACK)** - Cria uma versão "compacta" do arquivo executável do programa;

**/NOE(XTDICTIONARY)** - O linker não deve procurar em um dicionário externo palavras chave não definidas;

**/NOD(EFAULTLIBRARYSEARCH)** - evita que o linker procure nas bibliotecas padrão as referências não resolvidas;

**/ST(ACK):0x8006** - define o tamanho da pilha (stack) do programa. O número 8006 em hexadecimal (ou 32.774 em decimal), é o maior tamanho de stack alocável neste caso.

## 4.5) BAPAS-DB

### 4.5.1) Várias Bases de Dados

O nome "default" do arquivo que contém a Base de Dados usada pelo BAPAS-DB e' **GESDAT_VW.** Para o caso do sistema operacional MS-DOS[1], como os nomes de arquivos têm no máximo 8 caracteres, o nome desse arquivo será, então: **GESDAT_V.**

Para que possamos criar várias Bases de Dados independentes, o que se

pode fazer é ter para cada Base de Dados um nome particular diferente de GESDAT_V e, quando uma Base de Dados necessita ser acessada, pede-se ao sistema operacional que mude o nome do arquivo para GESDAT_V, fazem-se os acessos e, ao final da sessão, devolve-se ao arquivo seu nome original.

No nosso caso, o que fizemos foi designar que todas as Bases de Dados tivessem como extensão: **_DB**. Assim, uma Base de Dados da Unicamp teria, possivelmente, o nome de UNICAMP._DB.

Outra coisa a ser levada em consideração é que uma vez que se tenha definido a estrutura da Base de Dados de um Sistema de Auxílio a Projetos, todos os projetos desenvolvidos com esse sistema estarão associados com uma Base de Dados com estrutura idêntica.

Assim, é interessante que se crie uma Base de Dados "modelo" e, quando for necessário criar-se uma Base de Dados para um novo projeto, simplesmente faz-se uma cópia da Base de Dados modelo.

Além disso, como geralmente as Bases de Dados têm um tamanho considerável, seria melhor ainda se pudessemos "compactar"[1] esse modelo, e expandi-lo quando necessário.

### 4.5.2) DDL

A linguagem de descrição de dados DDL oferecida pelo BAPAS-DB tem algumas particularidades não documentadas, as quais fomos identificando e contornando, quando fosse o caso, através da utilização intensiva do programa.

**- chamadas de procedimentos**

A linguagem DDL permite que as definições sejam feitas, no seu interior, de maneira interativa ou através da chamada de arquivos em lote (procedimentos, na nomenclatura da linguagem DDL). No caso de procedimentos,

---

[1] Em nosso caso, utilizamos o utilitário de compactação PKZIP da Pkware Inc., que tem uma boa eficiência e é distribuído na forma de "shareware".

pode-se invocá-los da seguinte maneira:

**PROC,,***procedimento;*

onde: *procedimento* é nome do arquivo que contém as diretivas para a linguagem DDL.

Ocorre, porém, que o nome desse procedimento pode ter um tamanho máximo de 6 caracteres, e não deve possuir extensão alguma. O exemplo seguinte é de uma chamada a rocedimento válida:

**PROC**,,ddldef;

Para contornar essa limitação, pode-se invocar a linguagem DDL a partir do sistema operacional utilizando-se o redirecionamento de entrada para introduzir declarações DDL. Assim, se o nome do arquivo de procedimentos for ddldefns.ddl, e este for um nome válido para o sistema operacional, pode-se proceder da seguinte maneira:

**DDL** < ddldefns.ddl

**- nome do campo chave**

Problemas podem advir se, ao escolhermos o campo de um registro que funcionará como campo-chave para um arquivo da Base de Dados, acontecer que esse campo seja uma cadeia de caracteres e o campo imediatamente seguinte seja de um dos tipos numéricos.

Esta configuração propicia a ocorrência de um erro em tempo execução do BAPAS-DB.

Para contornar este problema devemos definir, além da posição do campo indexador no registro, também o nome desse campo, como no exemplo a seguir:

```
NREC = aluno
       STRUCT [ nome   CHAR(20),       ← campo de caracteres
                RA     FIXED(5) ] ;    ← campo numérico

NDBF = alunos,
       FILE=arq_aluno / VOL=50,
       REC=aluno,
       STRAT=HASHIN /CONF=1
                    /KEYLGT=20
                    /KEYPOS=1          ← posição do campo indexador
                    /KEY=[ nome ] ;    ← nome do campo indexador
```

**- tamanho da área de gerência**

Se durante a utilização da linguagem DDL para a definição da estrutura de uma Base de Dados, ocorrer a mensagem de erro: DICTIONARY OVERFLOW (estouro da capacidade do Dicionário de Dados) a solução é aumentar o espaço alocado para a área de gerência (MAXIMUM OF MANAGEMENT AREA). Esse tamanho deve ser um múltiplo de 1024 bytes.

### 4.5.3) problemas ( "bugs" )

Vamos, aqui, apresentar alguns problemas da versão para PC[1] sob MS-DOS[2] do BAPAS-DB, a qual utilizamos para a implementação do estudo de caso (Netcad) da utilização da metodologia proposta.

Note-se que tais problemas foram observados somente na versão utilizada, a qual apresenta sérias limitações decorrentes das características inerentes às limitações da configuração de hardware e software básico desse sistema computacional.

---

[1] IBM-PC é marca registrada da IBM Corp.
[2] MS-DOS é marca registrada da MicroSoft Corp.

**- cadeias de caracteres longas**

Um problema que só conseguimos identificar e contornar após muitas tentativas infrutíferas de solução consiste no seguinte: após compilar e linkar com sucesso (e sem erros) um sistema (no caso o Netcad), as chamadas a funções do BAPAS-DB transcorriam sem problemas, até que, ao executar alguma função que efetuasse operações sobre algum array de caracteres grande (isto é, neste caso, maior do que 20 posições), o BAPAS-DB respondia com uma mensagem de erro: "Warning, Key too Long", a todas as tentativas de acesso subsequentes. O problema só era resolvido saindo-se do sistema e reiniciando-o.

Isto nos obrigou a extensas mudanças na filosofia da codificação do nosso estudo de caso que, originalmente, previa a utilização de diversos arrays desse tipo para as mais variadas finalidades.

Assim, deve-se evitar desde o princípio, que um projeto preveja o uso de tal tipo de estrutura para evitar problemas.

**- arrays grandes**

Um problema análogo ocorre ao se utilizar arrays numéricos a partir de um certo tamanho: após efetuar-se um acesso à Base de Dados (por exemplo: uma leitura), se desejarmos armazenar os valores numéricos preenchidos na estrutura C (struct), correspondente ao tipo de registro do arquivo da Base de Dados, em um array bi-dimensional de inteiros, pode acontecer um "embaralhamento" dos dados armazenados na estrutura C.

Como forma de contornar esse problema, pode-se utilizar alocação dinâmica de memória ou, então, utilizar armazenamento temporário em disco.

Note-se que os problemas descritos foram identificados apenas para a versão do BAPAS-DB descrita, não sendo problemas inerentes ao BAPAS-DB, mas sim decorrentes de limitações do sistema computacional (hardware + software básico) utilizado.

## 4.6) Interface-Usuário

Daremos aqui algumas "dicas" para facilitar a utilização da interface-usuário baseada em janelas e menus.

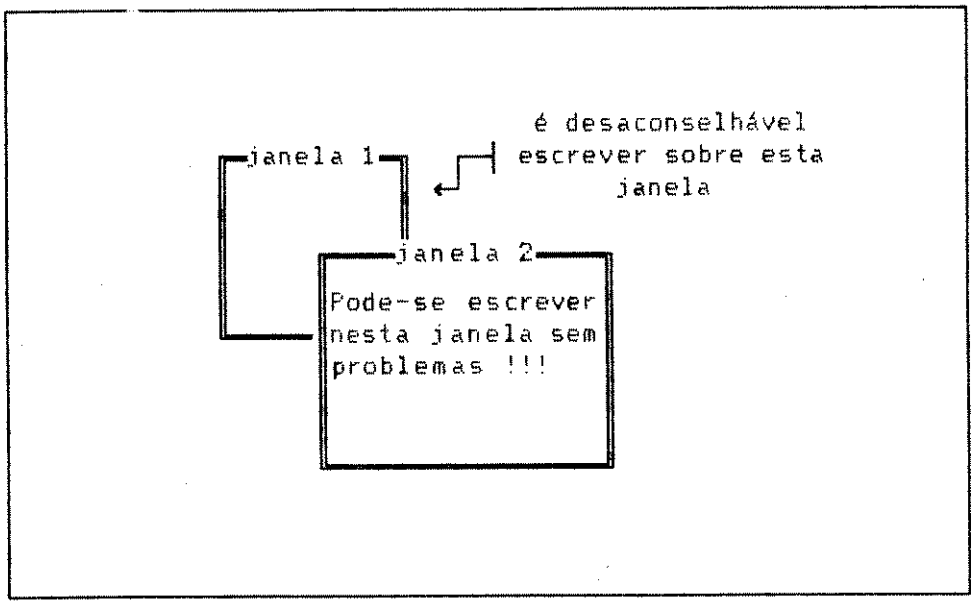
### - entrada de dados do usuário

Quando se faça necessário requisitar dados do usuário que esteja utilizando um sistema interativamente, deve-se escolher uma função que permita a edição dos caracteres em uma região bem definida da tela, cujas fronteiras não se possa ultrapassar pois, geralmente, essa entrada de dados vai tomar lugar em uma janela, cujas bordas não devem ser transpostas, sob pena de "sujar" os dados das demais janelas existentes.

A função por nós recomendada é a função **cgets**, da biblioteca **conio.h** da linguagem C, que possui todas essas características.

### - escrevendo nas janelas

Como o sistema de gerenciamento das janelas e menus (que são, na verdade, janelas também) é baseado em uma pilha, permitindo a criação de múltiplas janelas na tela simultaneamente, porém o esquema de trabalho da pilha é LIFO (Last In First Out), não possuindo a característica de sobrepor a janela selecionada às demais, deve-se escrever somente em janelas que estejam completamente visíveis, sob pena de adulterar os dados apresentados em outra janelas. Veja figura a seguir:




**4.7) Economia de memória**

**- uso de alocação dinâmica de memória**

Para otimizar o uso da memória disponível, deve-se dar preferência à alocação e liberação dinâmicas, sobre a alocação estática de memória.

Isto significa que os recursos de memória só estarão alocados a certos trechos de programa enquanto necessários, sendo liberados quando possível.

Por exemplo, para criar uma cadeia de caracteres de 20 posições (mais uma para o caractere delimitador de fim de cadeia de caracteres: \0), pode-se utilizar alocação estática, como no exemplo a seguir:

```
void main( void )
{
     char cadeia [ 21 ];

     restante do programa ...
}
```

ou, de preferência, a alocação dinâmica de memória, como no exemplo a seguir:

```
void main( void )
{
     char *cadeia;

     cadeia=(char *)malloc(21*sizeof(char));

        parte do programa que utiliza a variável cadeia

     free(cadeia);

        restante do programa ...
}
```

**- uso de arquivos em disco**

Para otimizar ainda mais a quantidade de memória disponível, pode-se criar arquivos em disco para armazenar as cadeias de caracteres que de outra forma teriam que ser residentes na memória.

Tais arquivos podem agrupar, por exemplo, as mensagens de erro do sistema, ou então mensagens de ajuda ("Help"), as quais são utilizadas pouco frequentemente e consumiriam quantidades consideráveis de memória se residentes.

O acesso a tais cadeias de caracteres em disco é muito mais lento, porém isso não compromete o desempenho global do sistema, pois tais acessos são feitos muito esporadicamente.

O arquivo DB_ERR.MSG constante dos anexos, por exemplo, é o arquivo que contém as mensagens de erro do sistema BAPAS-DB, sendo utilizado na implementação do sistema Netcad.

# Anexo 8

## Formulários de Pesquisa

CAR_EQP1

# Características dos Equipamentos Existentes

| Sub-Rede | |
|---|---|

| Características => / Equipamentos | Modelo (Fabricante) | CPU | Quantidade de Memória | Nº de Portas Seriais | Sistema Operacional | Periféricos Conectados |
|---|---|---|---|---|---|---|
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |

Finalidade: Fornecer as características dos equipamentos que vão ser integrados à rede.

Sub-Rede: Nome da sub-rede (se conhecida) onde se localiza(m) o(s) equipamento(s) em questão.

Modelo/Fabricante: Fabricante e/ou modelo do equipamento.

CPU: Tipo da Unidade Central de Processamento.

Quantidade de Memória: Memória primária (RAM), em Kbytes, do equipamento.

Número de Portas Seriais: Número de portas seriais disponíveis no equipamento.

Sistema Operacional/Versão: Tipo e versão do Sistema Operacional utilizado.

Periféricos conectados: Indicar todos os periféricos conectados diretamente ao equipamento.

CAR_EQP2

Características dos Equipamentos em Aquisição

| Sub-Rede | |
|---|---|

| Características => / Equipamento | Modelo (Fabricante) | CPU | Quantidade de Memória | Nº de Portas Seriais | Sistema Operacional | Periféricos Conectados |
|---|---|---|---|---|---|---|
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |

Finalidade: Indicar a existência de equipamentos em aquisição em um futuro próximo, fornecendo suas características.

Sub-Rede: Nome da sub-rede (se conhecida) onde se localizará(ão) o(s) equipamento(s) em questão.

Equipamento: Nome (identificador unívoco) do equipamento.

Modelo/Fabricante: Fabricante e/ou modelo do equipamento.

CPU: Tipo da Unidade Central de Processamento.

Quantidade de Memória: Memória primária (RAM), em Kbytes, do equipamento.

Número de Portas Seriais: Número de portas seriais disponíveis no equipamento.

Sistema Operacional/Versão: Tipo e versão do Sistema Operacional a ser utilizado.

Periféricos Conectados: Indicar todos os periféricos que serão conectados diretamente ao equipamento.

TRAFEGO

## Tráfego de Equipamentos

| Equipamento | | Sub-Rede | |
|---|---|---|---|
| Entrada | Bits/segundo | | |
| | Média | Pico | |
| | | | |
| Saída | Bits/segundo | | |
| | Média | Pico | |
| | | | |

| Equipamento | | Sub-Rede | |
|---|---|---|---|
| Entrada | Bits/segundo | | |
| | Média | Pico | |
| | | | |
| Saída | Bits/segundo | | |
| | Média | Pico | |
| | | | |

| Equipamento | | Sub-Rede | |
|---|---|---|---|
| Entrada | Bits/segundo | | |
| | Média | Pico | |
| | | | |
| Saída | Bits/segundo | | |
| | Média | Pico | |
| | | | |

Finalidade: Determinar as características de tráfego (requisitos de taxa de comunicação) de cada dispositivo.

### Preenchimento do formulário

1. Reunir todos os formulários (TRANSDAD) que envolvam o equipamento em questão à esquerda.

2. Calcular o valor do tráfego (frequência X tamanho das mensagens) para cada aplicação. Preencher com este valor o campo tráfego médio.

3. Somar os tráfegos médios das aplicações que forem concorrentes e comparar os resultados com os valores de tráfego individuais. O maior valor será o tráfego de saída do equipamento.

4. O cálculo do valor de pico é análogo ao anterior porém multiplica-se o resultado final por um coeficiente de ajuste de 0,75.

APL_ADIC

Aplicações a Serem Adicionadas

| Novas Aplicações | Probabilidade (alta,média,baixa) | Descrição dos Novos Serviços |
|---|---|---|
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |

Finalidade: Listar todas as aplicações que podem ser adicionadas à rede.

Novas Aplicações: Aplicações que podem ser adicionadas à rede.

Probabilidade: Probabilidade de que a nova aplicação venha realmente a ser adicionada à rede (alta,média,baixa).

Descrição dos Novos Serviços: Listagem dos serviços que podem ser antecipados.

CONEQUIP

Conexões

| Equipamentos -> | Equipamento X1 | Equipamento X2 |
|---|---|---|
| | | |

Características da Comunicação

| Descrição do Meio Físico | | |
|---|---|---|
| Tamanho das Mensagens | X1 => X2 | X1 <= X2 |
| | | |
| Frequência das Mensagens | X1 => X2 | X1 <= X2 |
| | | |

| Equipamentos -> | Equipamento X1 | Equipamento X2 |
|---|---|---|
| | | |

Características da Comunicação

| Descrição do Meio Físico | | |
|---|---|---|
| Tamanho das Mensagens | X1 => X2 | X1 <= X2 |
| | | |
| Frequência das Mensagens | X1 => X2 | X1 <= X2 |
| | | |

Finalidade: Indicar quais pares (X1,X2) de equipatos encontram-se atualmente conectados por um meio fisico de comunicação.

Equipamentos: Os equipamentos já conectados.

Descrição do Meio Físico: Indicar o tipo de meio de de transmissão utilizado.

Tamanho de Mensagem: Comprimento médio das mensagens (em bits).

Frequência das Mensagens: Indicar a frequência de ocorrência das mensagens em minutos.

ADAPLNEQ

# Adição de Aplicações c/ Novos Equipamentos

| Equipamento | | Sub-Rede | |
|---|---|---|---|

| Aplicações | Probabilidade (alta,média, ou baixa) | Tipo de Equipamento | Serviços de Comunicação | Tamanho da Mensagem | Frequência da Aplicação | Concorrência | Tráfego Estimado | Novas Instalações Físicas |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| | | | | | | | | |
| | | | | | | | | |
| | | | | | | | | |
| | | | | | | | | |
| | | | | | | | | |
| | | | | | | | | |
| | | | | | | | | |
| | | | | | | | | |
| | | | | | | | | |
| | | | | | | | | |
| | | | | | | | | |
| | | | | | | | | |
| | | | | | | | | |
| | | | | | | | | |
| | | | | | | | | |
| | | | | | | | | |
| | | | | | | | | |

Finalidade: Indicar a possibilidade de implementação de novas funções que poderão implicar na aquisição de novos recursos computacionais.

Equipamento/Sub-Rede: Nome e sub-rede (se conhecida) do equipamento a ser adquirido.

Aplicações: Nome e tipo das aplicações a serem instaladas no equipamento em aquisição.

Probabilidade: Probabilidade de que a nova aplicação venha a ser realmente instalada (alta,média,baixa).

Tipo de Equipamento: Descrição do tipo de equipamento a ser adquirido.

Serviços de Comunicação: Serviços de comunicação que serão utilizados pelas novas aplicações no novo equipamento.

Tamanho das mensagens: É o tamanho típico das mensagens geradas pela aplicação, em bits.

Frequência da Aplicação: É a frequência de execução de uma aplicação, em minutos.

Concorrência: Este campo é preenchido com os números das aplicações que são executadas simultaneamente (concorrentemente).

Volume de Tráfego Estimado: Estimativa do volume de tráfego que será gerado pela aplicação (somente aplicações "on-line").

OBS: os quatro campos acima devem ser preenchidos do mesmo modo que no formulário TRANSDAD.

Novas Instalações Físicas: Indicar se são necessárias novas instalações físicas para suportar o equipamento e/ou a aplicação.

ADIC_APL

Adição de Aplicações em Equipamentos Existentes

| Equipamento | | Sub-Rede | |
|---|---|---|---|

| Novas Aplicações | Probabilidade (alta,média, ou baixa) | Serviços de Comunicação | Tamanho da Mensagem | Frequência da Aplicação | Concorrência | Tráfego Estimado | Equipamento a ser Conectado |
|---|---|---|---|---|---|---|---|
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |

Finalidade: Indicar a probabilidade de implementação de novas aplicações para cada equipamento atualmente disponível na área.
Equipamento/Sub_rede: Nome e sub-rede (se conhecida) do equipamento onde será instalada a nova aplicação.
Novas Aplicações: Nome e tipo da nova aplicação.
Probabilidade: probabilidade de que a nova aplicação seja realmente instalada (alta,média,baixa).
Serviços de Comunicação: Serviços de comunicação que serão utilizados pela nova aplicação.
Tamanho das mensagens: é o tamanho típico das mensagens geradas pela aplicação, em bits.
Frequência da Aplicação: é a frequência de execução de uma aplicação, em minutos.
Concorrência: Este campo é preenchido com os números das aplicações que são executadas simultaneamente (concorrentemente).
Volume de Tráfego Estimado: Estimativa do volume de tráfego que será gerado pela aplicação (somente aplicações "on-line").
OBS: os quatro campos acima devem ser preenchidos do mesmo modo que no formulário TRANSDAD.
Equipamento a ser Conectado: equipamento(s) que será conectado ao equipamento em questão, através da nova aplicação.

APLNOVEQ

## Aplicações em Novos equipamentos

| Equipamento | Descrição da Aplicação |
|---|---|
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |

Finalidade: Fornecer o tipo de aplicação na qual o equipamento a ser adquirido será utilizado.

Equipamento: Nome do equipamento em aquisição no qual se deseja instalar a aplicação.

Descrição da Aplicação: Nome e tipo da aplicação a ser instalada.

COM_SUB

Comunicação já Existente entre Sub-Redes

| Sub-Rede | |
|---|---|

| Sub-Redes Conectadas => | | | | | |
|---|---|---|---|---|---|
| Meio de Comunicação Utilizado | | | | | |
| Tamanho da Informação Enviada | | | | | |
| Frequência da informação Enviada | | | | | |
| Tamanho da Informação Recebida | | | | | |
| Frequência da informação Recebida | | | | | |

Finalidade: Determinar o nível de integração atual da sub-rede em questão com outras sub-redes da empresa.

Sub-Rede: Nome (designação) da sub-rede em análise.

Sub-Redes Conectadas: Nomes das sub-redes conectadas à sub-rede em análise.

Meio de Comunicação Utilizado: Tipo de meio de transmissão utilizado.

Tamanho/Frequência da Informação Enviada: Comprimento médio em bits e frequência em minutos das mensagens enviadas.

Tamanho/Frequência da Informação Recebida: Comprimento médio em bits e frequência em minutos das mensagens recebidas.

CRES_APL

## Crescimento de Aplicações

| Nome da Aplicação | |
|---|---|

| Tipo de Crescimento | Probabilidade de Ocorrência (alta,média,baixa) |
|---|---|
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |

Finalidade: Indicar o tipo de crescimento dos requisitos de rede para cada aplicação.

Nome da Aplicação: Nome da aplicação a ser ampliada

Tipo de Crescimento: Preencher com o número do tipo conforme abaixo:

1- tráfego adicional pode ser gerado através dos dispositivos que já participam da aplicação;

2- novos dispositivos podem ser adicionados à rede no futuro para participar da aplicação;

3- outros dispositivos já existentes na rede podem vir a fazer parte da aplicação.

Probabilidade de Ocorrência: Probabilidade de que o crescimento venha a ocorrer realmente.

EXPANSUB

# Expansão Prevista Entre Sub-Redes

| Sub-Rede | |
|---|---|

| Características da Comunicação | Sub-Redes Conectadas | | | | |
|---|---|---|---|---|---|
| | | | | | |
| Tipos de Serviço | | | | | |
| Volume de Tráfego Estimado | | | | | |
| Instalação de novos equipamentos ? | | | | | |

Finalidade: Levantar as necessidades previstas de comunicação entre sub-redes, resultantes da instalação de novos equipamentos.

Sub-Rede: Nome da sub-rede em questão.

Sub-Redes Conectadas: sub-redes com comunicação com a sub-rede em questão.

Tipos de Serviço: Indicar os tipos de serviço utilizados pelas sub-redes.

Volume de Tráfego Estimado: Volume estimado de tráfego entre as sub-redes consideradas, em bits/s.

Instalação de Novos Equipamentos: Relacionar os novos equipamentos, se necessários.

QUALSERV

Requisitos de Qualidade de Serviço

| Serviço | Requisitos de Qualidade |
|---|---|
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |

Finalidade: Especificar os requisitos de qualidade associados à qualidade do serviço da rede.

Serviço: Refere-se as funções realizadas pela rede do ponto de vista de um programa de aplicação ou de um usuário.

Requisitos de qualidade: dependem do tipo de serviço.

ex: para um serviço orientado a conexão:

- taxa de terminação abrupta;
- taxa de erros detectados;
- taxa de erros residual.

## Preenchimento

1. Listar os serviços utilizados pelas aplicações;

2. Determinar os requisitos de qualidade de serviço necessários para suportar as necessidades das aplicações.

TRAF_SUB

Tráfego entre Sub-Redes na Situacao Desejável

| Sub-Rede | |
|---|---|

| Sub-Redes Conectadas -> | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Volume de Informação | | Frequência da Informação | | Volume de Informação | | Frequência da Informação | | Volume de Informação | | Frequência da Informação | |
| Serviços Desejados | Recebida | Enviada | Recebida | Enviada | Recebida | Enviada | Recebida | Enviada | Recebida | Enviada | Recebida | Enviada |
| | | | | | | | | | | | | |
| | | | | | | | | | | | | |
| | | | | | | | | | | | | |
| | | | | | | | | | | | | |
| | | | | | | | | | | | | |
| | | | | | | | | | | | | |
| | | | | | | | | | | | | |
| | | | | | | | | | | | | |
| | | | | | | | | | | | | |
| | | | | | | | | | | | | |
| | | | | | | | | | | | | |
| | | | | | | | | | | | | |
| | | | | | | | | | | | | |
| | | | | | | | | | | | | |
| | | | | | | | | | | | | |
| | | | | | | | | | | | | |
| | | | | | | | | | | | | |
| | | | | | | | | | | | | |

Finalidade: Indicar novas conexões de redes desejáveis para permitir um melhor fluxo de informação entre as sub-redes.

Sub-Rede: Nome da sub-rede em análise.

Sub-Redes Conectadas: sub-redes conectadas (a serem conectadas) à sub-rede em questão.

Serviços Desejados: Serviços atualmente utilizados e/ou novos serviços a serem utilizados.

Volume de Informação enviada/recebida: Quantidade de informação, em bits, enviada/recebida pela sub-rede.

Frequência da Informação enviada/recebida: Frequência da informação enviada/recebida pela sub-rede.

TRANSDAD

Transferência de Dados

| Equipamento | Sub-Rede | | Equipamento |
|---|---|---|---|
| | | => | |

| Aplicação | | | | Atraso Máximo (opcional) | Tamanho das mensagens (em bits) | Tráfego em bits/s | |
|---|---|---|---|---|---|---|---|
| Nº | Nome | Frequência | concorrência | | | média | pico |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |

Finalidade: Listar as aplicações que requerem a troca de dados, para cada par de equipamentos.

Equipamento/Sub-Rede: Nome e sub-rede (se conhecida) onde se localiza o equipamento comunicante a ser conectado à rede.

Nº: Número da aplicação.

Nome: Nome da aplicação que requer troca de dados entre os equipamentos.

Frequência: é a frequência de execução de uma aplicação, em minutos.

Concorrência: Este campo é preenchido com os números das aplicações que são executadas simultaneamente (concorrentemente).

Atraso máximo: é o tempo máximo de propagação de mensagens aceitável por uma dada aplicação.

Tamanho das mensagens: é o tamanho típico das mensagens geradas pela aplicação, em bits.

Tráfego (média/pico): transferência de dados média e de pico para cada aplicação, em bits/s. (somente p/aplicações tipo "online")

OBS: Outro formulário idêntico deve ser preenchido para a transferência de dados do segundo equipamento para o primeiro.

TREQSUBD

Tráfego entre Equipamentos de Sub-Rede na Situação Desejável

| Sub-Rede | |
|---|---|

| Nome do Equipamento | |
|---|---|

| Características | | | | | | |
|---|---|---|---|---|---|---|
| Aplicações | Serviços Desejados | Equipamento a ser Conectado | Volume de Informação | | Frequência da Informação | |
| | | | Enviada | Recebida | Enviada | Recebida |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |

Finalidade: Indicar como uma aplicação, implementada em cada equipamento, poderia usar os recursos de comunicação já existentes para facilitar a sua execução.

Nome/Sub-Rede: Nome e sub-rede (se conhecida) onde se localiza o equipamento em questão.

Aplicação: Listar todas as aplicações utilizadas neste equipamento.

Serviços Desejados: Serviços atualmente utilizados e/ou novos serviços a serem utilizados.

Equipamento a ser conectado: nome do equipamento (da mesma sub-rede) que será conectado ao equipamento em questão.

Volume de informação enviada/recebida: Quantidade de informação, em bits, enviada/recebida pelo equipamento.

Frequência da informação enviada/recebida: Frequência da informação, enviada/recebida pelo equipamento.

# Anexo 9

## Os arquivos NETCAD.DDL, DDLREC e DDLSTR

```
/*****************************************************************************/
/*                                                                           */
/*     N   N  EEEEE  TTTTT  CCCCC  AAAAA  DDD        DDD    DDD    L        */
/*     NN  N  E        T    C      A   A  D  D       D  D   D  D   L        */
/*     N N N  EEEE     T    C      AAAAA  D   D      D   D  D   D  L        */
/*     N  NN  E        T    C      A   A  D  D       D  D   D  D   L        */
/*     N   N  EEEEE    T    CCCCC  A   A  DDD    OO  DDD    DDD    LLLLL    */
/*                                                                           */
/*****************************************************************************/
/*                                                                           */
/*   NETCAD.DDL - Arquivo que contém a descrição da estrutura da base de     */
/*                dados usada no sistema NETCAD.                             */
/*                                                                           */
/*****************************************************************************/

BEGIN;
DB;

/*****************************************************************************/
/*                                                                           */
/* CHAMADA DO ARQUIVO COM AS DEFINIÇÕES DOS REGISTROS                        */
/*                                                                           */
/*****************************************************************************/

PROC,,DDLREC; /* nº máximo de caracteres no nome do arquivo -> 6 */

/*****************************************************************************/
/*                                                                           */
/*                                                                           */
/* CHAMADA DO ARQUIVO COM AS DEFINIÇÕES DAS ESTRATÉGIAS                      */
/*                                                                           */
/*                                                                           */
/*****************************************************************************/

PROC,,DDLSTR; /* nº máximo de caracteres no nome do arquivo -> 6 */

/*****************************************************************************/
/*                                                                           */
/*                                                                           */
/* ARQUIVO -> Hierarquia da Empresa                                          */
/*                                                                           */
/*                                                                           */
/*****************************************************************************/

NDBF = HRQ_EMPR,
       FILE=D_EMPRESA / VOL=100,
       REC=HIERARQUIA,
       STRAT=HASHIN /CONF=1
                    /KEYLGT=20
                    /KEYPOS=1
                    /KEY=[ NODE ];
```

```
/*****************************************************************************/
/*                                                                           */
/* ARQUIVO -> Características dos Equipamentos Existentes                    */
/*                                                                           */
/*****************************************************************************/

NDBF = CAR_EQP1,
       FILE=D_CAR_EQP1 / VOL=100,
       REC=EQUIPAMENTO,
       STRAT=BBAUM /KEYLGT=20
                   /KEYPOS=1
                   /KEYTYP=4
                   /SYNKEY=1
                   /KEY=[ NOME_EQP ];

/*****************************************************************************/
/*                                                                           */
/* ARQUIVO -> Características dos Equipamentos em Aquisição                  */
/*                                                                           */
/*****************************************************************************/

NDBF = CAR_EQP2,
       FILE=D_CAR_EQP2 / VOL=100,
       REC=EQUIPAMENTO,
       STRAT=BBAUM /KEYLGT=20
                   /KEYPOS=1
                   /KEYTYP=4
                   /SYNKEY=1
                   /KEY=[ NOME_EQP ];

/*****************************************************************************/
/*                                                                           */
/* ARQUIVO -> Transferência de Dados                                         */
/*                                                                           */
/*****************************************************************************/

NDBF = TRANSDAD,
       FILE=D_TRANSDAD / VOL=100,
       REC=TRANSFERENCIA_DE_DADOS,
       STRAT=BBAUM /KEYLGT=20
                   /KEYPOS=1
                   /KEYTYP=4
                   /SYNKEY=0
                   /KEY=[ EQUIPAMENTO_FONTE ];
```

```
/*****************************************************************************/
/*                                                                           */
/* ARQUIVO -> Tráfego de Equipamentos                                        */
/*                                                                           */
/*****************************************************************************/

NDBF = TRAFEGO,
       FILE=D_TRAFEGO / VOL=100,
       REC=TRAFEGO_DE_EQUIPAMENTO,
       STRAT=BBAUM /KEYLGT=20
                   /KEYPOS=1
                   /KEYTYP=4
                   /SYNKEY=1
                   /KEY=[ NOME_DO_EQP ];

/*****************************************************************************/
/*                                                                           */
/* ARQUIVO -> Requisitos de Qualidade de Serviço                             */
/*                                                                           */
/*****************************************************************************/

NDBF = QUALSERV,
       FILE=D_QUALSERV / VOL=100,
       REC=QUALIDADE_DE_SERVICO,
       STRAT=BBAUM /KEYLGT=20
                   /KEYPOS=1
                   /KEYTYP=4
                   /SYNKEY=0
                   /KEY=[ SERVICO ];

/*****************************************************************************/
/*                                                                           */
/* ARQUIVO -> Conexões                                                       */
/*                                                                           */
/*****************************************************************************/

NDBF = CONEQUIP,
       FILE=D_CONEQUIP / VOL=100,
       REC=CONEXOES,
       STRAT=BBAUM /KEYLGT=20
                   /KEYPOS=1
                   /KEYTYP=4
                   /SYNKEY=0
                   /KEY=[ EQUIPAMENTO_X1 ];
```

```
/*****************************************************************************/
/*                                                                           */
/* ARQUIVO -> Comunicação já Existente entre Sub-Redes                       */
/*                                                                           */
/*****************************************************************************/

NDBF = COM_SUB,
       FILE=D_COM_SUB / VOL=100,
       REC=COM_EXISTENTE_ENTRE_SUB_REDES,
       STRAT=BBAUM /KEYLGT=20
                   /KEYPOS=1
                   /KEYTYP=4
                   /SYNKEY=0
                   /KEY=[ NOME_SUB_REDE ];

/*****************************************************************************/
/*                                                                           */
/* ARQUIVO ->  Tráfego entre Equipamentos da Sub-Rede na Situação Desejável  */
/*                                                                           */
/*****************************************************************************/

NDBF = TREQSUBD,
       FILE=D_TREQSUBD / VOL=100,
       REC=TRF_EQPS_SUB_REDE_DESEJAVEL,
       STRAT=BBAUM /KEYLGT=20
                   /KEYPOS=1
                   /KEYTYP=4
                   /SYNKEY=0
                   /KEY=[ SUB_REDE ];

/*****************************************************************************/
/*                                                                           */
/* ARQUIVO -> Tráfego entre Sub-Redes na Situação Desejável                  */
/*                                                                           */
/*****************************************************************************/

NDBF = TRAF_SUB,
       FILE=D_TRAF_SUB / VOL=100,
       REC=TRF_SUB_REDES_DESEJAVEL,
       STRAT=BBAUM /KEYLGT=20
                   /KEYPOS=1
                   /KEYTYP=4
                   /SYNKEY=0
                   /KEY=[ SUB_REDE ];
```

```
/*****************************************************************************/
/*                                                                           */
/* ARQUIVO -> Aplicações em Novos Equipamentos                               */
/*                                                                           */
/*****************************************************************************/

NDBF = APLNOVEQ,
       FILE=D_APLNOVEQ / VOL=100,
       REC=APLICACOES_NOVOS_EQP,
       STRAT=BBAUM /KEYLGT=20
                   /KEYPOS=1
                   /KEYTYP=4
                   /SYNKEY=0
                   /KEY=[ EQUIPAMENTO ];

/*****************************************************************************/
/*                                                                           */
/* ARQUIVO -> Adição de Aplicações em Equipamentos Existentes                */
/*                                                                           */
/*****************************************************************************/

NDBF = ADIC_APL,
       FILE=D_ADIC_APL / VOL=100,
       REC=ADIC_APL_EQP,
       STRAT=BBAUM /KEYLGT=20
                   /KEYPOS=1
                   /KEYTYP=4
                   /SYNKEY=0
                   /KEY=[ EQUIPAMENTO ];

/*****************************************************************************/
/*                                                                           */
/* ARQUIVO -> Crescimento de Aplicações                                      */
/*                                                                           */
/*****************************************************************************/

NDBF = CRES_APL,
       FILE=D_CRES_APL / VOL=100,
       REC=CRESCIMENTO_APLICACAO,
       STRAT=BBAUM /KEYLGT=20
                   /KEYPOS=1
                   /KEYTYP=4
                   /SYNKEY=0
                   /KEY=[ NOME_DA_APLICACAO ];
```

```
/*****************************************************************************/
/*                                                                           */
/* ARQUIVO -> Adição de Aplicações c/ Novos Equipamentos                     */
/*                                                                           */
/*****************************************************************************/

NDBF = ADAPLNEQ,
       FILE=D_ADAPLNEQ / VOL=100,
       REC=ADIC_APL_NOVOS_EQP,
       STRAT=BBAUM /KEYLGT=20
                   /KEYPOS=1
                   /KEYTYP=4
                   /SYNKEY=0
                   /KEY=[ EQUIPAMENTO ];

/*****************************************************************************/
/*                                                                           */
/* ARQUIVO -> Expansão Prevista entre Sub-Redes                              */
/*                                                                           */
/*****************************************************************************/

NDBF = EXPANSUB,
       FILE=D_EXPANSUB / VOL=100,
       REC=EXPANSAO_SUB_REDE,
       STRAT=BBAUM /KEYLGT=20
                   /KEYPOS=1
                   /KEYTYP=4
                   /SYNKEY=0
                   /KEY=[ SUB_REDE ];

/*****************************************************************************/
/*                                                                           */
/* ARQUIVO -> Aplicações a Serem Adicionadas                                 */
/*                                                                           */
/*****************************************************************************/

NDBF = APL_ADIC,
       FILE=D_APL_ADIC / VOL=100,
       REC=APLICACOES_A_ADICIONAR,
       STRAT=BBAUM /KEYLGT=20
                   /KEYPOS=1
                   /KEYTYP=4
                   /SYNKEY=0
                   /KEY=[ NOVA_APLICACAO ];
```

```
/*****************************************************************************/
/*                                                                           */
/* ARQUIVO -> Aplicações Globais                                             */
/*                                                                           */
/*****************************************************************************/

NDBF = APL_GLOB,
       FILE=D_APL_GLOB / VOL=10,
       REC=APLICACOES_GLOBAIS,
       STRAT=BBAUM /KEYLGT=20
                   /KEYPOS=1
                   /KEYTYP=4
                   /SYNKEY=1
                   /KEY=[ NOME_APLICACAO ];

/*****************************************************************************/
/*                                                                           */
/* ARQUIVO -> Requisitos Especiais                                           */
/*                                                                           */
/*****************************************************************************/

NDBF = RQ_ESPEC,
       FILE=D_RQ_ESPEC / VOL=20,
       REC=REQUISITOS_ESPECIAIS,
       STRAT=BBAUM /KEYLGT=20
                   /KEYPOS=1
                   /KEYTYP=4
                   /SYNKEY=0
                   /KEY=[ CATEGORIA ];

STOP;
```

```
/*****************************************************************************/
/*                                                                           */
/*      DDLREC - Arquivo com as definições dos registros usados no           */
/*               sistema NETCAD.                                             */
/*                                                                           */
/*****************************************************************************/

/*****************************************************************************/
/*                                                                           */
/*       Definição da estrutura do registro usado para a descrição           */
/*       da hierarquia da empresa.                                           */
/*                                                                           */
/*****************************************************************************/
/*                                                                           */
/* REGISTRO -> Hierarquia da Empresa                                         */
/*                                                                           */
/*****************************************************************************/

NREC = HIERARQUIA
       STRUCT[ NODE   CHAR(20),
               LEVEL  FIXED(2),
               PARENT CHAR(20)];

/*****************************************************************************/
/*                                                                           */
/* REGISTRO -> Características dos Equipamentos                              */
/*                                                                           */
/*****************************************************************************/

NREC = EQUIPAMENTO
       STRUCT[ NOME_EQP           CHAR(20),
               SUB_REDE           CHAR(20),
               FABRICANTE_MODELO  CHAR(20),
               CPU                CHAR(20),
               MEMORIA            FIXED(4),
               PORTAS_SERIAIS     FIXED(1),
               SO                 CHAR(20),
               PERIFERICOS_1      CHAR(20),
               PERIFERICOS_2      CHAR(20),
               PERIFERICOS_3      CHAR(20),
               PERIFERICOS_4      CHAR(20),
               PERIFERICOS_5      CHAR(20)];
```

```
/*****************************************************************************/
/*                                                                           */
/* REGISTRO -> Transferência de Dados                                        */
/*                                                                           */
/*****************************************************************************/

NREC = TRANSFERENCIA_DE_DADOS
       STRUCT[ EQUIPAMENTO_FONTE        CHAR(20),
               SUB_REDE_FONTE           CHAR(20),
               EQUIPAMENTO_DESTINO      CHAR(20),
               NUMERO                   FIXED(2),
               NOME_DA_APLICACAO        CHAR(20),
               FREQUENCIA_DA_APLICACAO  FIXED(5),
               CONCORRENCIA             CHAR(20),
               ATRASO_MAXIMO            FIXED(5),
               TAMANHO_MENSAGEM         FIXED(6),
               TRAFEGO_MEDIO            FIXED(6),
               TRAFEGO_MAXIMO           FIXED(7)];

/*****************************************************************************/
/*                                                                           */
/*  REGISTRO -> Tráfego de Equipamentos                                      */
/*                                                                           */
/*****************************************************************************/

NREC = TRAFEGO_DE_EQUIPAMENTO
       STRUCT[ NOME_DO_EQP        CHAR(20),
               SUB_REDE           CHAR(20),
               BPS_ENTRADA_MEDIO  FIXED(5),
               BPS_ENTRADA_PICO   FIXED(6),
               BPS_SAIDA_MEDIO    FIXED(5),
               BPS_SAIDA_PICO     FIXED(6)];

/*****************************************************************************/
/*                                                                           */
/* REGISTRO -> Requisitos de Qualidade de Serviço                            */
/*                                                                           */
/*****************************************************************************/

NREC = QUALIDADE_DE_SERVICO
       STRUCT[ SERVICO   CHAR(20),
               REQUISITO CHAR(20)];
```

```
/*************************************************************************/
/*                                                                       */
/* REGISTRO -> Conexões                                                  */
/*                                                                       */
/*************************************************************************/

NREC = CONEXOES
       STRUCT[ EQUIPAMENTO_X1       CHAR(20),
               EQUIPAMENTO_X2       CHAR(20),
               MEIO_FISICO          CHAR(20),
               TAM_MENSAGEM_X1_X2   FIXED(6),
               TAM_MENSAGEM_X2_X1   FIXED(6),
               FREQ_MENSAGEM_X1_X2  FIXED(5),
               FREQ_MENSAGEM_X2_X1  FIXED(5)];

/*************************************************************************/
/*                                                                       */
/* REGISTRO -> Comunicação ja Existente entre Sub_Redes                  */
/*                                                                       */
/*************************************************************************/

NREC = COM_EXISTENTE_ENTRE_SUB_REDES
       STRUCT[ NOME_SUB_REDE              CHAR(20),
               SUB_REDE_CONECTADA         CHAR(20),
               MEIO_FISICO_UTILIZADO      CHAR(20),
               TAM_INFORMACAO_ENVIADA     FIXED(6),
               FREQ_INFORMACAO_ENVIADA    FIXED(5),
               TAM_INFORMACAO_RECEBIDA    FIXED(6),
               FREQ_INFORMACAO_RECEBIDA   FIXED(5)];

/*************************************************************************/
/*                                                                       */
/* REGISTRO -> Tráfego entre Equipamentos da Sub-Rede na Situação Desejável*/
/*                                                                       */
/*************************************************************************/

NREC = TRF_EQPS_SUB_REDE_DESEJAVEL
       STRUCT[ SUB_REDE             CHAR(20),
               EQUIPAMENTO          CHAR(20),
               APLICACAO            CHAR(20),
               SERVICO_DESEJADO_1   CHAR(20),
               SERVICO_DESEJADO_2   CHAR(20),
               SERVICO_DESEJADO_3   CHAR(20),
               SERVICO_DESEJADO_4   CHAR(20),
               SERVICO_DESEJADO_5   CHAR(20),
               EQUIPAMENTO_PAR      CHAR(20),
               VOL_INFO_ENVIADA     FIXED(6),
               VOL_INFO_RECEBIDA    FIXED(6),
               FREQ_INFO_ENVIADA    FIXED(5),
               FREQ_INFO_RECEBIDA   FIXED(5)];
```

```
/*************************************************************************/
/*                                                                       */
/* REGISTRO -> Tráfego entre Sub-Redes na Situação Desejável             */
/*                                                                       */
/*************************************************************************/

NREC = TRF_SUB_REDES_DESEJAVEL
       STRUCT[ SUB_REDE            CHAR(20),
               SUB_REDE_CONECTADA  CHAR(20),
               SERVICO_DESEJADO    CHAR(20),
               VOL_INFO_ENVIADA    FIXED(6),
               VOL_INFO_RECEBIDA   FIXED(6),
               FREQ_INFO_ENVIADA   FIXED(5),
               FREQ_INFO_RECEBIDA FIXED(5)];

/*************************************************************************/
/*                                                                       */
/* REGISTRO -> Aplicações em Novos Equipamentos                          */
/*                                                                       */
/*************************************************************************/

NREC = APLICACOES_NOVOS_EQP
       STRUCT[ EQUIPAMENTO            CHAR(20),
               DESCRICAO_APLICACAO_1 CHAR(20),
               DESCRICAO_APLICACAO_2 CHAR(20),
               DESCRICAO_APLICACAO_3 CHAR(20),
               DESCRICAO_APLICACAO_4 CHAR(20),
               DESCRICAO_APLICACAO_5 CHAR(20)];

/*************************************************************************/
/*                                                                       */
/* REGISTRO -> Adição de Aplicações em Equipamentos Existentes           */
/*                                                                       */
/*************************************************************************/

NREC = ADIC_APL_EQP
       STRUCT[ EQUIPAMENTO              CHAR(20),
               SUB_REDE                 CHAR(20),
               NOVA_APLICACAO           CHAR(20),
               PROBABILIDADE            CHAR(5),
               SERVICO_COMUNICACAO_1    CHAR(20),
               SERVICO_COMUNICACAO_2    CHAR(20),
               SERVICO_COMUNICACAO_3    CHAR(20),
               SERVICO_COMUNICACAO_4    CHAR(20),
               SERVICO_COMUNICACAO_5    CHAR(20),
               TAMANHO_MENSAGEM         FIXED(6),
               FREQUENCIA_DA_APLICACAO FIXED(5),
               CONCORRENCIA             CHAR(20),
               TRAFEGO_ESTIMADO         FIXED(6),
               EQUIPAMENTO_PAR          CHAR(20)];
```

```
/*****************************************************************************/
/*                                                                           */
/* REGISTRO -> Crescimento de Aplicações                                     */
/*                                                                           */
/*****************************************************************************/

NREC = CRESCIMENTO_APLICACAO
       STRUCT[ NOME_DA_APLICACAO    CHAR(20),
               TIPO_DE_CRESCIMENTO  FIXED(1),
               PROBAB_OCORRENCIA    CHAR(5)];

/*****************************************************************************/
/*                                                                           */
/* REGISTRO -> Adição de Aplicações c/ Novos Equipamentos                    */
/*                                                                           */
/*****************************************************************************/

NREC = ADIC_APL_NOVOS_EQP
       STRUCT[ EQUIPAMENTO              CHAR(20),
               SUB_REDE                 CHAR(20),
               APLICACAO                CHAR(20),
               PROBABILIDADE            CHAR(5),
               TIPO_EQUIPAMENTO         CHAR(20),
               SERVICO_COMUNICACAO_1    CHAR(20),
               SERVICO_COMUNICACAO_2    CHAR(20),
               SERVICO_COMUNICACAO_3    CHAR(20),
               TAMANHO_MENSAGEM         FIXED(6),
               FREQUENCIA_DA_APLICACAO  FIXED(5),
               CONCORRENCIA             CHAR(20),
               TRAFEGO_ESTIMADO         FIXED(6),
               NOVAS_INST_FISICAS_1     CHAR(20),
               NOVAS_INST_FISICAS_2     CHAR(20),
               NOVAS_INST_FISICAS_3     CHAR(20)];

/*****************************************************************************/
/*                                                                           */
/* REGISTRO -> Expansão Prevista entre Sub-Redes                             */
/*                                                                           */
/*****************************************************************************/

NREC = EXPANSAO_SUB_REDE
       STRUCT[ SUB_REDE              CHAR(20),
               SUB_REDE_PAR          CHAR(20),
               SERVICO_1             CHAR(20),
               SERVICO_2             CHAR(20),
               SERVICO_3             CHAR(20),
               SERVICO_4             CHAR(20),
               SERVICO_5             CHAR(20),
               VOL_TRAFEGO_ESTIMADO  FIXED(7),
               INST_NOVOS_EQP_1      CHAR(20),
               INST_NOVOS_EQP_2      CHAR(20),
               INST_NOVOS_EQP_3      CHAR(20)];
```

```
/*****************************************************************************/
/*                                                                           */
/* REGISTRO -> Aplicações a Serem Adicionadas                                */
/*                                                                           */
/*****************************************************************************/

NREC = APLICACOES_A_ADICIONAR
       STRUCT[ NOVA_APLICACAO       CHAR(20),
               PROBABILIDADE        CHAR(5),
               DESCR_NOVOS_SERVICOS CHAR(20)];

/*****************************************************************************/
/*                                                                           */
/* REGISTRO -> Aplicações Globais                                            */
/*                                                                           */
/*****************************************************************************/

NREC = APLICACOES_GLOBAIS
       STRUCT[ NOME_APLICACAO       CHAR(20),
               NIVEL_A              FIXED(2),
               NIVEL_B              FIXED(2),
               NIVEL_C              FIXED(2),
               NIVEL_D              FIXED(2),
               NIVEL_E              FIXED(2),
               DESCRICAO_1          CHAR(20),
               DESCRICAO_2          CHAR(20),
               DESCRICAO_3          CHAR(20)];

/*****************************************************************************/
/*                                                                           */
/* REGISTRO -> Requisitos Especiais                                          */
/*                                                                           */
/*****************************************************************************/

NREC = REQUISITOS_ESPECIAIS
       STRUCT[ CATEGORIA            CHAR(20),
               DESCRICAO_1          CHAR(20),
               DESCRICAO_2          CHAR(20),
               DESCRICAO_3          CHAR(20),
               DESCRICAO_4          CHAR(20),
               DESCRICAO_5          CHAR(20)];
```

```
/*************************************************************************/
/*                                                                       */
/*     DDLSTR - Arquivo com as definições das estratégias usadas no      */
/*              BAPAS-DB                                                 */
/*                                                                       */
/*************************************************************************/

/*************************************************************************/
/*                                                                       */
/*  Definição da estratégia de busca BBAUM ( BTREE )                     */
/*                                                                       */
/*************************************************************************/
/* CONFIGURAÇOES:                                                        */
/* -------------                                                         */
/*                                                                       */
/* KEYLGT       : 2-BYTES TAMANHO DA CHAVE                               */
/* KEYPOS       : 2-BYTES POSIÇAO DA CHAVE NO REGISTRO (início = 1)      */
/* SYNKEY  = 1  : SOMENTE UM REGISTRO POR CHAVE                          */
/*         /= 1 : POSSIBILIDADE DE MAIS DE UM REGISTRO POR CHAVE         */
/* KEYTYP  = 1  : TIPO DA CHAVE E' FIXED(15)                             */
/*          = 2 : TIPO DA CHAVE E' FIXED(31)                             */
/*          = 3 : TIPO DA CHAVE E' FLOAT                                 */
/*          = 4 : TIPO DA CHAVE E' CHAR                                  */
/*          = 5 : TIPO DA CHAVE E' CLOCK                                 */
/*          = 6 : TIPO DA CHAVE E' DURATION                              */
/*                                                                       */
/*************************************************************************/

NSTRAT = BBAUM                              /SCBLGT=9,
                KEYLGT FIXED(15)            /SCBPOS=8,
                KEYPOS FIXED(15)            /SCBPOS=6,
                KEYTYP FIXED(15)            /SCBPOS=9,
                SYNKEY FIXED(15) OPTIONAL   /SCBPOS=7;

/*************************************************************************/
/*                                                                       */
/*  Definição da estratégia de busca HASHIN (HASH c/ chaves unívocas )   */
/*                                                                       */
/*************************************************************************/
/* CONFIGURAÇOES:                                                        */
/* -------------                                                         */
/*                                                                       */
/* CONF = 1     : STANDARD                                               */
/* KEYLGT       : 2-BYTES TAMANHO DA CHAVE                               */
/* KEYPOS       : 2-BYTES POSIÇAO DA CHAVE NO REGISTRO (início = 1)      */
/*                                                                       */
/*************************************************************************/

NSTRAT = HASHIN                             /SCBLGT=21,
                CONF   FIXED(15)            /SCBPOS=1,
                KEYLGT FIXED(15)            /SCBPOS=2,
                KEYPOS FIXED(15)            /SCBPOS=3;
```

```
/*************************************************************************/
/*                                                                       */
/*  Definição da estratégia de busca FIFO e LIFO                         */
/*                                                                       */
/*************************************************************************/
/* CONFIGURAÇÕES:                                                        */
/* -------------                                                         */
/*                                                                       */
/* CONF = 1     : STANDARD                                               */
/* CONF = 2     : BUFFER-CIRCULAR                                        */
/* CONF = 3     : COMPRIMENTO DE REGISTRO VARIAVEL                       */
/*                                                                       */
/* CYCLE  = 1   : SOBRE-ESCREVER NO BUFFER CIRCULAR                      */
/* CYCLE /= 1   : NAO SOBRE-ESCREVER NO BUFFER CIRCULAR                  */
/*                                                                       */
/*************************************************************************/

NSTRAT = FIFO                              /SCBLGT=9,
              CONF  FIXED(15)              /SCBPOS=2,
              CYCLE FIXED(15) OPTIONAL     /SCBPOS=1;


NSTRAT = LIFO                              /SCBLGT=9,
              CONF  FIXED(15)              /SCBPOS=2,
              CYCLE FIXED(15) OPTIONAL     /SCBPOS=1;

/*************************************************************************/
/*                                                                       */
/*  Definição da estratégia de busca INDSQL (Sequêncial indexada) sem    */
/*  buffer circular e comprimento de registro variável.                  */
/*                                                                       */
/*************************************************************************/
/* CONFIGURAÇÕES:                                                        */
/* -------------                                                         */
/*                                                                       */
/* CONF = 1     : STANDARD                                               */
/* KEYLGT       : 2-BYTES Comprimento da chave                           */
/* KEYPOS       : 2-BYTES Posição da chave no registro (início = 1)      */
/* KEYNO        : Máximo de chaves por arquivo                           */
/*                ( DEFAULT: Máximo de registros por arquivo )           */
/*                                                                       */
/*************************************************************************/

NSTRAT = INDSQL                                /SCBLGT=21,
                CONF   FIXED(15)               /SCBPOS=1,
                KEYLGT FIXED(15)               /SCBPOS=2,
                KEYPOS FIXED(15)               /SCBPOS=3,
                KEYNO  FIXED(31) OPTIONAL      /SCBPOS=12;
```

```
/*************************************************************************/
/*                                                                       */
/*  Definição da estratégia de busca HSP ( Residente em Memória )        */
/*                                                                       */
/*************************************************************************/
/* CONFIGURAÇÕES:                                                        */
/* -------------                                                         */
/*                                                                       */
/* CONF    = 1  : Estratégia BBAUM (BTREE)                               */
/*         = 2  : FIFO com comprimento de registro variável              */
/*         = 3  : FIFO com buffer-circular ( futuro )                    */
/*         = 4  : FIFO com buffer circular                               */
/*                              e comprimento de registro variável       */
/*         = 5  : Estratégia BBAUM (BTREE) com um registro por chave     */
/*         = 6  : Estratégia BBAUM (BTREE) com um registro por chave     */
/*              :               e comprimento de registro variável       */
/*         = 7  : Estratégia BBAUM (BTREE) com mais de um registro por chave*/
/*         = 8  : Estratégia BBAUM (BTREE) com mais de um registro por chave*/
/*              :               e comprimento de registro variável       */
/* KEYLGT       : 2-BYTES Comprimento da chave                           */
/* KEYPOS       : 2-BYTES Posição da chave no registro (início = 1)      */
/* KEYTYP  = 1  : tipo da chave e' FIXED(15)                             */
/*         = 2  : tipo da chave e' FIXED(31)                             */
/*         = 3  : tipo da chave e' FLOAT                                 */
/*         = 4  : tipo da chave e' CHARACTER                             */
/*         = 5  : tipo da chave e' CLOCK                                 */
/*         = 6  : tipo da chave e' DURATION                              */
/* MEMLGT       : Comprimento de memória para comprimento de registro    */
/*              : variável no HW                                         */
/*                                                                       */
/*************************************************************************/

NSTRAT = HSP                                /SCBLGT=22,
             CONF     FIXED(15)             /SCBPOS=1,
             KEYPOS   FIXED(15) OPTIONAL    /SCBPOS=2,
             KEYLGT   FIXED(15) OPTIONAL    /SCBPOS=3,
             KEYTYP   FIXED(15) OPTIONAL    /SCBPOS=4,
             MEMLGT   FIXED(31) OPTIONAL    /SCBPOS=5;

/*************************************************************************/
/*                                                                       */
/*  Definição da estratégia de busca SATZNR (RNR - número de registro)   */
/*                                                                       */
/*************************************************************************/
/* CONFIGURAÇÕES:                                                        */
/* -------------                                                         */
/*                                                                       */
/* CONF = 2    : STANDARD                                                */
/*                                                                       */
/*************************************************************************/

NSTRAT = SATZNR                     /SCBLGT=15,
               CONF  FIXED(15)      /SCBPOS=1;
```

```
/*************************************************************************/
/*                                                                       */
/*  Definição da estratégia de busca INDSQL (Sequencial indexada) com    */
/*  buffer circular e comprimento de registro variável.                  */
/*                                                                       */
/*************************************************************************/
/* CONFIGURAÇÕES:                                                        */
/* -------------                                                         */
/*                                                                       */
/* CONF = 2     : Comprimento de registro variavel                       */
/* CONF = 3     : Buffer circular                                        */
/* CONF = 4     : CYCLE-BUFFER-AU (Especial: Buffer circular com DUPKEY) */
/* CONF = 5     : CYCLE-BUFFER-CONF5 (Especial: Buffer circular com DUPKEY */
/*                                    e ACCESS-MODE especial)            */
/* KEYLGT       : 2-BYTES Comprimento da chave                           */
/* KEYPOS       : 2-BYTES Posição da chave no registro (início = 1)      */
/* KEYNO        : Maximo de chaves por arquivo                           */
/*                quando CONF=3, número de Buffers circulares a criar    */
/*                ( DEFAULT: Número de registros )                       */
/*                                                                       */
/* NOFCYCBU     : Número de registros por buffer circular                */
/*                                                                       */
/* CYCLE  = 1   : SOBRE-ESCREVER NO BUFFER CIRCULAR                      */
/* CYCLE /= 1   : NAO SOBRE-ESCREVER NO BUFFER CIRCULAR                  */
/*                                                                       */
/*************************************************************************/

NSTRAT = INDSQC                                 /SCBLGT=85,          /*4.061PLA*/
                KEYLGT   FIXED(15)              /SCBPOS=66,
                KEYPOS   FIXED(15)              /SCBPOS=79,
                KEYNO    FIXED(15) OPTIONAL     /SCBPOS=80,
                NOFCYCBU FIXED(15) OPTIONAL     /SCBPOS=81,
                CONF     FIXED(15)              /SCBPOS=82,
                CYCLE    FIXED(15) OPTIONAL     /SCBPOS=85;


/*******************************  Fim de DDLSTR  *******************************/
```

# ANEXO 10

## ARCABOUÇO DO SISTEMA NETCAD PARA G.U.I. OPENWINDOWS

Este e´ um programa de teste da utilização do sistema XWindows para workstations SUN como interface-usuário para o sistema NETCAD.

Os arquivos componentes do sistema, são os descritos abaixo:

| | |
|---|---|
| README .DOC | Este arquivo |
| NETCAD .ICO | ícone para o programa Netcad |
| NETCAD_M.ICO | máscara para o ícone do programa Netcad |
| BAPAS_DB.ICO | ícone para o BAPAS-DB |
| DCA .ICO | ícone do nosso Departamento na UNICAMP |
| NETCAD32.ICO | ícone para ser usado no "file manager" |
| CL . | arquivo com a linha de comando para compilar o Netcad |
| NETCAD .C | O programa Netcad |
| NETCAD .INF | O "help" do programa Netcad (seu nome deve ser mudado para "netcad.info". |

```
# CL

cc $* -I$OPENWINHOME/include -L$OPENWINHOME/lib -lxview -lolgx -lX11 -o $*
.out
```

```
/* NETCAD.C */

#define HELPPATHNAME "."

#define WHITE      0
#define RED        1
#define GREEN      2
#define BLUE       3
#define NUM_COLORS 4

#include <stdio.h>
#include <xview/xview.h>
#include <xview/frame.h>
#include <xview/panel.h>
#include <xview/svrimage.h>
#include <xview/icon.h>
#include <xview/cms.h>
#include <xview/notice.h>
short open_bits[]={
#include"dca.ico"
};
short closed_bits[]={
#include"netcad.ico"
};
short closed_bits_mask[]={
#include"netcad_m.ico"
};
short bapas_bits[]={
#include"bapas_db.ico"
};

void modulo_1(),modulo_2(),modulo_3(),modulo_4(),documentacao(),db_access(),
     projeto_preliminar(),
     selecao_de_formularios(),
     geracao_de_formularios(),
     entrada_de_dados(),
     hierarquia_basica(),
     facilidades_ja_existentes(),
     aplicacoes_globais(),
     requisitos_especiais(),
     carregar(),
     get_name_iniciar(),
     iniciar(),
     listar(),
     modificar(),
     salvar();

Frame frame,bapas_frame,subframe1,subframe2,subframe3,subframe4,
      subframe1_1,
      subframe1_2,
      subframe1_3,
      subframe1_4,
```

```
    subframe1_5,
    subframe1_1_1,
    subframe1_1_2,
    subframe1_1_3,
    subframe1_1_4,
    subframe1_1_1_2;
Panel panel;
Server_image open_image,closed_image,closed_image_mask,bapas_image;

void main(argc,argv)
 int argc;
 char *argv[];
{
  extern char *getenv();
  char *helppath,buf[BUFSIZ];
  Icon icon;
  Rect rect;
  Xv_singlecolor fg,bg;
  Cms cms;
  static Xv_singlecolor colors[]={
       { 255, 255, 255},   /* white */
       { 255,   0,   0},   /* red   */
       {   0, 255,   0},   /* green */
       {   0,   0, 255},   /* blue  */
  };

  fg.red=0; fg.green=0; fg.blue=255;
  bg.red=255; bg.green=255; bg.blue=255;

  cms=(Cms)xv_create(NULL,CMS,
                     CMS_CONTROL_CMS, TRUE,
                     CMS_SIZE,        CMS_CONTROL_COLORS+4,
                     CMS_COLORS,      colors, NULL);

  xv_init(XV_INIT_ARGC_PTR_ARGV,&argc,argv,NULL);

  sprintf(buf,"HELPPATH=%s:",HELPPATHNAME);
  if(helppath=getenv("HELPPATH"))
   strcat(buf,helppath);
  putenv(buf);

  open_image=(Server_image)xv_create(NULL,SERVER_IMAGE,
                                     XV_WIDTH,64,
                                     XV_HEIGHT,64,
                                     SERVER_IMAGE_BITS,open_bits,
                                     NULL);

  closed_image=(Server_image)xv_create(NULL,SERVER_IMAGE,
                                       XV_WIDTH,64,
                                       XV_HEIGHT,64,
                                       SERVER_IMAGE_BITS,closed_bits,
                                       NULL);


  closed_image_mask=(Server_image)xv_create(NULL,SERVER_IMAGE,
                                            XV_WIDTH,64,
                                            XV_HEIGHT,64,
                                            SERVER_IMAGE_BITS,closed_bits_mask,
```

```
                                        NULL);

bapas_image=(Server_image)xv_create(NULL,SERVER_IMAGE,
                                    XV_WIDTH,64,
                                    XV_HEIGHT,64,
                                    SERVER_IMAGE_BITS,bapas_bits,
                                    NULL);

icon=(Icon)xv_create(frame,ICON,
                     ICON_TRANSPARENT,TRUE,
                     ICON_IMAGE, closed_image,
                     ICON_MASK_IMAGE, closed_image_mask,
                     NULL);

rect.r_width=(int)xv_get(icon,XV_WIDTH);
rect.r_height=(int)xv_get(icon,XV_HEIGHT);
rect.r_top=830;
rect.r_left=1080;

frame=(Frame)xv_create(NULL,FRAME,
                       FRAME_LABEL,     "NETCAD",
                       FRAME_BACKGROUND_COLOR, &bg,
                       FRAME_FOREGROUND_COLOR, &fg,
                       FRAME_INHERIT_COLORS,TRUE,
                       FRAME_NO_CONFIRM, FALSE,
                       FRAME_SHOW_FOOTER,TRUE,
                       FRAME_LEFT_FOOTER,"     Depto. de Engenharia da Computacao e Automacao Industrial",
                       FRAME_ICON,       icon,
                       FRAME_CLOSED_RECT,&rect,
                       XV_WIDTH,         500,
                       XV_HEIGHT,        100,
                       XV_X,             210,
                       XV_Y,              50,
                       NULL);

panel=(Panel)xv_create(frame,PANEL,
                       XV_HELP_DATA,"netcad:Netcad",
                       WIN_CMS, cms,
                       NULL);

(void)xv_create(panel,PANEL_MESSAGE,
                PANEL_LABEL_IMAGE,open_image,NULL);

(void)xv_create(panel,PANEL_BUTTON,
                XV_HELP_DATA,"netcad:Modulo 1",
                PANEL_LABEL_STRING, "Modulo 1",
                PANEL_NOTIFY_PROC,   modulo_1, NULL);

(void)xv_create(panel,PANEL_BUTTON,
                XV_HELP_DATA,"netcad:Modulo 2",
                PANEL_LABEL_STRING, "Modulo 2",
                PANEL_NOTIFY_PROC,   modulo_2, NULL);

(void)xv_create(panel,PANEL_BUTTON,
                XV_HELP_DATA,"netcad:Modulo 3",
                PANEL_LABEL_STRING, "Modulo 3",
                PANEL_NOTIFY_PROC,   modulo_3, NULL);
```

```
(void)xv_create(panel,PANEL_BUTTON,
                XV_HELP_DATA,"netcad:Modulo 4",
                PANEL_LABEL_STRING, "Modulo 4",
                PANEL_NOTIFY_PROC,   modulo_4, NULL);

(void)xv_create(panel,PANEL_MESSAGE,
                XV_HELP_DATA,"netcad:BAPAS-DB",
                PANEL_LABEL_IMAGE,bapas_image,
                PANEL_ITEM_COLOR,CMS_CONTROL_COLORS+GREEN,
                PANEL_NOTIFY_PROC,db_access, NULL);

bapas_frame=(Frame)xv_create(frame,FRAME_CMD,
                          FRAME_LABEL,"BAPAS - DB",
                          FRAME_CMD_PUSHPIN_IN,TRUE,
                          FRAME_SHOW_RESIZE_CORNER,TRUE,
                          FRAME_SHOW_FOOTER,TRUE,
                          FRAME_LEFT_FOOTER,"Netcad",
                          XV_WIDTH,   120,
                          XV_HEIGHT,  290,
                          XV_X,       100,
                          XV_Y,       500, NULL);

panel=(Panel)xv_get(bapas_frame,FRAME_CMD_PANEL);
xv_set(panel,XV_HELP_DATA,"netcad:BAPAS-DB",
            WIN_CMS, cms,
            NULL);
(void)xv_create(panel,PANEL_BUTTON,
                /*XV_HELP_DATA,"netcad:",*/
                PANEL_LABEL_STRING, "Find ALL",
                /*PANEL_NOTIFY_PROC,   ,*/ NULL);

 (void)xv_create(panel,PANEL_BUTTON,
                /*XV_HELP_DATA,"netcad:",*/
                PANEL_LABEL_STRING, "Find KEY",
                /*PANEL_NOTIFY_PROC,   ,*/ NULL);

 (void)xv_create(panel,PANEL_BUTTON,
                /*XV_HELP_DATA,"netcad:",*/
                PANEL_LABEL_STRING, "Insert",
                /*PANEL_NOTIFY_PROC,   ,*/ NULL);

 (void)xv_create(panel,PANEL_BUTTON,
                /*XV_HELP_DATA,"netcad:",*/
                PANEL_LABEL_STRING, "Modify",
                /*PANEL_NOTIFY_PROC,   ,*/ NULL);

 (void)xv_create(panel,PANEL_BUTTON,
                /*XV_HELP_DATA,"netcad:",*/
                PANEL_LABEL_STRING, "Erase KEY",
                /*PANEL_NOTIFY_PROC,   ,*/ NULL);

 (void)xv_create(panel,PANEL_BUTTON,
                /*XV_HELP_DATA,"netcad:",*/
                PANEL_LABEL_STRING, "Erase ALL",
                /*PANEL_NOTIFY_PROC,   ,*/ NULL);

 (void)xv_create(panel,PANEL_BUTTON,
                /*XV_HELP_DATA,"netcad:",*/
```

```
                PANEL_LABEL_STRING, "File Info",
                /*PANEL_NOTIFY_PROC,   ,*/ NULL);

(void)xv_create(panel,PANEL_MESSAGE,
                XV_HELP_DATA,"netcad:BAPAS-DB",
                PANEL_LABEL_IMAGE,bapas_image,
                PANEL_ITEM_COLOR,CMS_CONTROL_COLORS+GREEN,
                NULL);


subframe1=(Frame)xv_create(frame,FRAME_CMD,
                           FRAME_LABEL,"Modulo 1",
                           FRAME_CMD_PUSHPIN_IN,TRUE,
                           FRAME_SHOW_RESIZE_CORNER,TRUE,
                           FRAME_SHOW_FOOTER,TRUE,
                           FRAME_LEFT_FOOTER,"Netcad",
                           XV_WIDTH,   170,
                           XV_HEIGHT,  160,
                           XV_X,       100,
                           XV_Y,       100, NULL);

subframe2=(Frame)xv_create(frame,FRAME_CMD,
                           FRAME_LABEL,"Modulo 2",
                           FRAME_CMD_PUSHPIN_IN,TRUE,
                           FRAME_SHOW_RESIZE_CORNER,TRUE,
                           FRAME_SHOW_FOOTER,TRUE,
                           FRAME_LEFT_FOOTER,"Netcad",
                           XV_WIDTH,   110,
                           XV_HEIGHT,  100,
                           XV_X,       300,
                           XV_Y,       100, NULL);

subframe3=(Frame)xv_create(frame,FRAME_CMD,
                           FRAME_LABEL,"Modulo 3",
                           FRAME_CMD_PUSHPIN_IN,TRUE,
                           FRAME_SHOW_RESIZE_CORNER,TRUE,
                           FRAME_SHOW_FOOTER,TRUE,
                           FRAME_LEFT_FOOTER,"Netcad",
                           XV_WIDTH,   110,
                           XV_HEIGHT,  100,
                           XV_X,       500,
                           XV_Y,       100, NULL);

subframe4=(Frame)xv_create(frame,FRAME_CMD,
                           FRAME_LABEL,"Modulo 4",
                           FRAME_CMD_PUSHPIN_IN,TRUE,
                           FRAME_SHOW_RESIZE_CORNER,TRUE,
                           FRAME_SHOW_FOOTER,TRUE,
                           FRAME_LEFT_FOOTER,"Netcad",
                           XV_WIDTH,   110,
                           XV_HEIGHT,  100,
                           XV_X,       700,
                           XV_Y,       100, NULL);

panel=(Panel)xv_get(subframe1,FRAME_CMD_PANEL);
xv_set(panel,XV_HELP_DATA,"netcad:Modulo 1",NULL);
(void)xv_create(panel,PANEL_BUTTON,
                XV_HELP_DATA,"netcad:Projeto Preliminar",
```

```
                PANEL_LABEL_STRING,"Projeto Preliminar",
                PANEL_NOTIFY_PROC,projeto_preliminar,
                NULL);

(void)xv_create(panel,PANEL_BUTTON,
                XV_HELP_DATA,"netcad:Selecao de Formularios",
                PANEL_LABEL_STRING,"Selecao de Formularios",
                PANEL_NOTIFY_PROC,selecao_de_formularios,
                NULL);

(void)xv_create(panel,PANEL_BUTTON,
                XV_HELP_DATA,"netcad:Geracao de Formularios",
                PANEL_LABEL_STRING,"Geracao de Formularios",
                PANEL_NOTIFY_PROC,geracao_de_formularios,
                NULL);

(void)xv_create(panel,PANEL_BUTTON,
                XV_HELP_DATA,"netcad:Entrada de Dados",
                PANEL_LABEL_STRING,"Entrada de Dados",
                PANEL_NOTIFY_PROC,entrada_de_dados,
                NULL);

(void)xv_create(panel,PANEL_BUTTON,
                XV_HELP_DATA,"netcad:Documentacao",
                PANEL_LABEL_STRING,"Documentacao",
                PANEL_NOTIFY_PROC,documentacao,
                NULL);

subframe1_1=(Frame)xv_create(frame,FRAME_CMD,
                        FRAME_LABEL,"Projeto Preliminar",
                        FRAME_CMD_PUSHPIN_IN,TRUE,
                        FRAME_SHOW_RESIZE_CORNER,TRUE,
                        FRAME_SHOW_FOOTER,TRUE,
                        FRAME_LEFT_FOOTER,"Netcad",
                        FRAME_RIGHT_FOOTER,"Modulo 1",
                        XV_WIDTH,   180,
                        XV_HEIGHT,  120,
                        XV_X,       150,
                        XV_Y,       130, NULL);

subframe1_2=(Frame)xv_create(frame,FRAME_CMD,
                        FRAME_LABEL,"Selecao de Formularios",
                        FRAME_CMD_PUSHPIN_IN,TRUE,
                        FRAME_SHOW_RESIZE_CORNER,TRUE,
                        FRAME_SHOW_FOOTER,TRUE,
                        FRAME_LEFT_FOOTER,"Netcad",
                        FRAME_RIGHT_FOOTER,"Modulo 1",
                        XV_WIDTH,   200,
                        XV_HEIGHT,  100,
                        XV_X,       250,
                        XV_Y,       130, NULL);

subframe1_3=(Frame)xv_create(frame,FRAME_CMD,
                        FRAME_LABEL,"Geracao de Formularios",
                        FRAME_CMD_PUSHPIN_IN,TRUE,
                        FRAME_SHOW_RESIZE_CORNER,TRUE,
                        FRAME_SHOW_FOOTER,TRUE,
                        FRAME_LEFT_FOOTER,"Netcad",
```

```
                        FRAME_RIGHT_FOOTER,"Modulo 1",
                        XV_WIDTH,   200,
                        XV_HEIGHT,  100,
                        XV_X,       350,
                        XV_Y,       130, NULL);

subframe1_4=(Frame)xv_create(frame,FRAME_CMD,
                        FRAME_LABEL,"Entrada de Dados",
                        FRAME_CMD_PUSHPIN_IN,TRUE,
                        FRAME_SHOW_RESIZE_CORNER,TRUE,
                        FRAME_SHOW_FOOTER,TRUE,
                        FRAME_LEFT_FOOTER,"Netcad",
                        FRAME_RIGHT_FOOTER,"Modulo 1",
                        XV_WIDTH,   180,
                        XV_HEIGHT,  100,
                        XV_X,       450,
                        XV_Y,       130, NULL);

subframe1_5=(Frame)xv_create(frame,FRAME_CMD,
                        FRAME_LABEL,"Documentacao",
                        FRAME_CMD_PUSHPIN_IN,TRUE,
                        FRAME_SHOW_RESIZE_CORNER,TRUE,
                        FRAME_SHOW_FOOTER,TRUE,
                        FRAME_LEFT_FOOTER,"Netcad",
                        FRAME_RIGHT_FOOTER,"Modulo 1",
                        XV_WIDTH,   140,
                        XV_HEIGHT,  100,
                        XV_X,       550,
                        XV_Y,       130, NULL);

panel=(Panel)xv_get(subframe1_1,FRAME_CMD_PANEL);
xv_set(panel,XV_HELP_DATA,"netcad:Projeto Preliminar",NULL);
(void)xv_create(panel,PANEL_BUTTON,
                XV_HELP_DATA,"netcad:Hierarquia Basica",
                PANEL_LABEL_STRING,"Hierarquia Basica",
                PANEL_NOTIFY_PROC,hierarquia_basica,
                NULL);

(void)xv_create(panel,PANEL_BUTTON,
                XV_HELP_DATA,"netcad:Facilidades ja Existentes",
                PANEL_LABEL_STRING,"Facilidades ja Existentes",
                PANEL_NOTIFY_PROC,facilidades_ja_existentes,
                NULL);

(void)xv_create(panel,PANEL_BUTTON,
                XV_HELP_DATA,"netcad:Aplicacoes Globais",
                PANEL_LABEL_STRING,"Aplicacoes Globais",
                PANEL_NOTIFY_PROC,aplicacoes_globais,
                NULL);

(void)xv_create(panel,PANEL_BUTTON,
                XV_HELP_DATA,"netcad:Requisitos Especiais",
                PANEL_LABEL_STRING,"Requisitos Especiais",
                PANEL_NOTIFY_PROC,requisitos_especiais,
                NULL);

subframe1_1_1=(Frame)xv_create(frame,FRAME_CMD,
                        FRAME_LABEL,"Hierarquia Basica",
```

```
                        FRAME_CMD_PUSHPIN_IN,TRUE,
                        FRAME_SHOW_RESIZE_CORNER,TRUE,
                        FRAME_SHOW_FOOTER,TRUE,
                        FRAME_LEFT_FOOTER,"Netcad",
                        FRAME_RIGHT_FOOTER,"Modulo 1",
                        XV_WIDTH,   480,
                        XV_HEIGHT,  400,
                        XV_X,       200,
                        XV_Y,       180, NULL);

subframe1_1_2=(Frame)xv_create(frame,FRAME_CMD,
                        FRAME_LABEL,"Facilidades ja Existentes",
                        FRAME_CMD_PUSHPIN_IN,TRUE,
                        FRAME_SHOW_RESIZE_CORNER,TRUE,
                        FRAME_SHOW_FOOTER,TRUE,
                        FRAME_LEFT_FOOTER,"Netcad",
                        FRAME_RIGHT_FOOTER,"Modulo 1",
                        XV_WIDTH,   220,
                        XV_HEIGHT,  100,
                        XV_X,       300,
                        XV_Y,       180, NULL);

subframe1_1_3=(Frame)xv_create(frame,FRAME_CMD,
                        FRAME_LABEL,"Aplicacoes Globais",
                        FRAME_CMD_PUSHPIN_IN,TRUE,
                        FRAME_SHOW_RESIZE_CORNER,TRUE,
                        FRAME_SHOW_FOOTER,TRUE,
                        FRAME_LEFT_FOOTER,"Netcad",
                        FRAME_RIGHT_FOOTER,"Modulo 1",
                        XV_WIDTH,   200,
                        XV_HEIGHT,  100,
                        XV_X,       400,
                        XV_Y,       180, NULL);

subframe1_1_4=(Frame)xv_create(frame,FRAME_CMD,
                        FRAME_LABEL,"Requisitos Especiais",
                        FRAME_CMD_PUSHPIN_IN,TRUE,
                        FRAME_SHOW_RESIZE_CORNER,TRUE,
                        FRAME_SHOW_FOOTER,TRUE,
                        FRAME_LEFT_FOOTER,"Netcad",
                        FRAME_RIGHT_FOOTER,"Modulo 1",
                        XV_WIDTH,   200,
                        XV_HEIGHT,  100,
                        XV_X,       500,
                        XV_Y,       180, NULL);

panel=(Panel)xv_get(subframe1_1_1,FRAME_CMD_PANEL);
xv_set(panel,XV_HELP_DATA,"netcad:Hierarquia Basica",
            PANEL_ITEM_X_GAP,40, NULL);
(void)xv_create(panel,PANEL_BUTTON,
               XV_HELP_DATA,"netcad:Carregar",
               PANEL_LABEL_STRING,"Carregar",
               PANEL_NOTIFY_PROC,carregar,
               NULL);

(void)xv_create(panel,PANEL_BUTTON,
               XV_HELP_DATA,"netcad:Iniciar",
               PANEL_LABEL_STRING,"Iniciar",
```

```
                PANEL_NOTIFY_PROC,get_name_iniciar,
                NULL);

 (void)xv_create(panel,PANEL_BUTTON,
                XV_HELP_DATA,"netcad:Listar",
                PANEL_LABEL_STRING,"Listar",
                PANEL_NOTIFY_PROC,listar,
                NULL);

 (void)xv_create(panel,PANEL_BUTTON,
                XV_HELP_DATA,"netcad:Modificar",
                PANEL_LABEL_STRING,"Modificar",
                PANEL_NOTIFY_PROC,modificar,
                NULL);

 (void)xv_create(panel,PANEL_BUTTON,
                XV_HELP_DATA,"netcad:Salvar",
                PANEL_LABEL_STRING,"Salvar",
                PANEL_NOTIFY_PROC,salvar,
                NULL);

 subframe1_1_1_2=(Frame)xv_create(frame,FRAME_CMD,
                          FRAME_LABEL,"iniciar",
                          FRAME_SHOW_RESIZE_CORNER,TRUE,
                          FRAME_SHOW_FOOTER,TRUE,
                          FRAME_LEFT_FOOTER,"Netcad",
                          FRAME_RIGHT_FOOTER,"Modulo 1",
                          XV_WIDTH,   305,
                          XV_HEIGHT,  20,
                          XV_X,       450,
                          XV_Y,       230, NULL);
 panel=(Panel)xv_get(subframe1_1_1_2,FRAME_CMD_PANEL);
 xv_create(panel,PANEL_TEXT,
           XV_HELP_DATA,"netcad:Iniciar",
           PANEL_LABEL_STRING, "Nome da Empresa:",
           PANEL_VALUE_STORED_LENGTH, 20,
           PANEL_NOTIFY_PROC,iniciar, NULL);

 window_fit(panel);
 window_fit(frame);

 xv_main_loop(frame);
}

void modulo_1()
{
  xv_set(subframe1,XV_SHOW,TRUE,NULL);
}

void modulo_2()
{
  xv_set(subframe2,XV_SHOW,TRUE,NULL);
}

void modulo_3()
{
  xv_set(subframe3,XV_SHOW,TRUE,NULL);
}
```

```
void modulo_4()
{
  xv_set(subframe4,XV_SHOW,TRUE,NULL);
}

void projeto_preliminar()
{
  xv_set(subframe1_1,XV_SHOW,TRUE,NULL);
}

void selecao_de_formularios()
{
  xv_set(subframe1_2,XV_SHOW,TRUE,NULL);
}

void geracao_de_formularios()
{
  xv_set(subframe1_3,XV_SHOW,TRUE,NULL);
}

void entrada_de_dados()
{
  xv_set(subframe1_4,XV_SHOW,TRUE,NULL);
}

void documentacao()
{

  xv_set(subframe1_5,XV_SHOW,TRUE,NULL);
}

void hierarquia_basica()
{
  xv_set(subframe1_1_1,XV_SHOW,TRUE,NULL);
}

void facilidades_ja_existentes()
{
  xv_set(subframe1_1_2,XV_SHOW,TRUE,NULL);
}

void aplicacoes_globais()
{
  xv_set(subframe1_1_3,XV_SHOW,TRUE,NULL);
}

void requisitos_especiais()
{
  xv_set(subframe1_1_4,XV_SHOW,TRUE,NULL);
}

void carregar(item,event)
 Panel_item item;
 Event *event;
{
  int result;
```

```
 result=!notice_prompt(panel,NULL,
                      /* NOTICE_FOCUS_XY, event_x(event), event_y(event),*/
                       NOTICE_MESSAGE_STRINGS, "A hierarquia residente",
                                               "na memoria sera perdida !!!",
                                               "Continuar ?",
                                                NULL,
                       NOTICE_BUTTON_YES, "Nao",
                       NOTICE_BUTTON_NO, "Sim", NULL);
}

void get_name_iniciar()
{
  xv_set(subframe1_1_1_2,XV_SHOW,TRUE,NULL);
}

void iniciar(item,event)
 Panel_item item;
 Event *event;
{
  printf("Empresa: %s\n",xv_get(item,PANEL_VALUE));
  xv_set(subframe1_1_1_2,XV_SHOW,FALSE,NULL);
}

void listar()
{
}

void modificar()
{
}

void salvar(item,event)
 Panel_item item;
 Event *event;
{
  int result;

  result=!notice_prompt(panel,NULL,
                       /* NOTICE_FOCUS_XY, event_x(event), event_y(event),*/
                        NOTICE_MESSAGE_STRINGS, "A hierarquia anterior na",
                                                "na Base de Dados sera perdida!!!",
                                                "Continuar ?",
                                                 NULL,
                        NOTICE_BUTTON_NO, "Sim",
                        NOTICE_BUTTON_YES, "Nao", NULL);
}

void db_access()
{
  xv_set(bapas_frame,XV_SHOW,TRUE,NULL);
}
```

:Netcad
Este e´ um Sistema de auxilio `a implementacao de Redes Locais de computadores.
#
:BAPAS-DB
Esta e´ a Base de Dados do sistema. E´ propriedade da WERUM Gmbh.
#
:Modulo 1
Este modulo auxilia o usuario a descrever a hierarquia da empresa em analise, com o proposito de propor uma rede hierarquicamente organizada inicial.
O usuario obtem ainda os formularios de pesquisa para obtencao de mais dados para o sistema, e os insere na base de dados.
#
:Modulo 2
** PARA IMPLEMENTACAO FUTURA **

Este modulo tem o objetivo de aprimorar a rede proposta do modulo anterior, utilizando as informacoes adicionadas pelos formularios de pesquisa, gerando ao seu final uma proposta de rede dita "intermediaria".
#
:Modulo 3
** PARA IMPLEMENTACAO FUTURA **

Este modulo tem por objetivo gerar uma proposta de implementacao real da rede, baseada na rede proposta no modulo anterior, podendo ser efetuadas simulacoes, analise de custos, e outras analises que forem cabiveis.
#
:Modulo 4
** PARA IMPLEMENTACAO FUTURA **

Este modulo tem por objetivo acompanhar a execucao do projeto de rede proposto no modulo anterior.
Pensa-se, ainda, estender o sistema para a fase de operacao da rede.
#
:Projeto Preliminar
Fase inicial de coleta de informacoes do usuario com a finalidade de elaborar uma proposta inicial de rede a ser implementada.
#
:Selecao de Formularios
** PARA IMPLEMENTACAO FUTURA **
#
:Geracao de Formularios
Esta opcao apresenta um menu para a impressao dos formularios de pesquisa.
#
:Entrada de Dados
Esta opcao apresenta o menu para a entrada, na Base de Dados, dos dados coletados atraves dos formularios de pesquisa preenchidos.
#

:Documentacao
** PARA IMPLEMENTACAO FUTURA **
#
:Hierarquia Basica
Construir, com auxilio do usuario, uma arvore que descreva a hierarquia da empresa.
#
:Facilidades ja Existentes
Reportar a existencia de eventuais facilidades de comunicacao ja existentes na empresa em questao.
#
:Aplicacoes Globais
O usuario pode definir de antemao as aplicacoes que terao que ser suportadas pela rede, e os niveis (da hierarquia) em que elas deverao ser implementadas.

Ex: FAX, video-conferencia, transmissao de voz, etc.
#
:Requisitos Especiais
O usuario pode definir de antemao requisitos que imponham caracteristicas especiais `a rede a ser implementada.
Ex: controle de processos (restricoes de tempo de acesso, taxa de erros admissivel e tolerancia a falhas).
#
:Carregar
Carregar (da base de dados para a memoria) uma arvore que descreve a hierarquia de uma empresa.
#
:Iniciar
Iniciar a construcao da arvore que descreve a hierarquia de uma empresa.
#
:Listar
Apresentar a arvore que descreve a hierarquia de uma empresa.
#
:Modificar
Modificar a arvore que descreve a hierarquia de uma empresa.
#
:Salvar
Salvar (da memoria para a base de dados) a arvore que descreve a hierarquia de uma empresa.
#

```
/* Format_version=1, Width=64, Height=64, Depth=1, Valid_bits_per_item=16
*/
        0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,
        0x0000,0xF031,0xF061,0xE000,0x0000,0xD871,0x98E3,0x0000,
        0x0000,0xD8B1,0x9963,0x0000,0x0001,0xF1B3,0xE361,0x8000,
        0x0001,0x99F3,0x03E0,0xC000,0x0001,0x9B33,0x0660,0xC000,
        0x0001,0xF333,0x0667,0x8000,0x0000,0x0000,0x0000,0x0000,
        0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,
        0x0000,0x003E,0x3C00,0x0000,0x0000,0x0033,0x3600,0x0000,
        0x0000,0x0033,0x3600,0x0000,0x0000,0x0073,0x7C00,0x0000,
        0x0000,0x0066,0x6600,0x0000,0x0000,0x0066,0x6600,0x0000,
        0x0000,0x0078,0x7C00,0x0000,0x0000,0x0000,0x0000,0x0000,
        0x0000,0x0000,0x0000,0x0000,0x0000,0x007F,0xFF00,0x0000,
        0x0000,0x03FF,0xFFE0,0x0000,0x0000,0x0F80,0x00F8,0x0000,
        0x0000,0x3E00,0x003E,0x0000,0x0000,0x7000,0x0007,0x0000,
        0x0000,0xE000,0x0003,0x8000,0x0000,0xFF80,0x00FF,0x8000,
        0x0000,0xCFFF,0xFFF9,0x8000,0x0000,0xE0FF,0xFF83,0x8000,
        0x0000,0xFC00,0x001F,0x8000,0x0000,0xDF80,0x00FD,0x8000,
        0x0000,0xC7FF,0xFFF1,0x8000,0x0000,0xC07F,0xFF01,0x8000,
        0x0000,0xC000,0x0001,0x8000,0x0000,0xC000,0x0001,0x8000,
        0x0000,0xC000,0x0001,0x8000,0x0000,0xC000,0x0001,0x8000,
        0x0000,0xC000,0x0001,0x8000,0x0000,0xC000,0x0001,0x8000,
        0x0000,0xC000,0x0001,0x8000,0x0000,0xC000,0x0001,0x8000,
        0x0000,0xC000,0x0001,0x8000,0x0000,0xC000,0x0001,0x8000,
        0x0000,0xC000,0x0001,0x8000,0x0000,0xC000,0x0001,0x8000,
        0x0000,0xC000,0x0001,0x8000,0x0000,0xC000,0x0001,0x8000,
        0x0000,0xC000,0x0001,0x8000,0x0000,0xC000,0x0001,0x8000,
        0x0000,0xC000,0x0001,0x8000,0x0000,0xC000,0x0001,0x8000,
        0x0000,0xC000,0x0001,0x8000,0x0000,0xC000,0x0001,0x8000,
        0x0000,0xC000,0x0001,0x8000,0x0000,0xC000,0x0001,0x8000,
        0x0000,0xC000,0x0001,0x8000,0x0000,0xC000,0x0001,0x8000,
        0x0000,0xC000,0x0001,0x8000,0x0000,0xE000,0x0003,0x8000,
        0x0000,0x7C00,0x001F,0x0000,0x0000,0x1FC0,0x01FC,0x0000,
        0x0000,0x07FF,0xFFF0,0x0000,0x0000,0x007F,0xFF00,0x0000
```

```
/* Format_version=1, Width=64, Height=64, Depth=1, Valid_bits_per_item=16
*/
        0x0000,0x0000,0x0000,0x0000,
        0x0000,0x0000,0x0000,0x0000,
        0x0000,0x0000,0x0000,0x0000,
        0x0000,0x0000,0x0000,0x0000,
        0x0000,0x0000,0x0000,0x0000,
        0x0000,0x0000,0x0000,0x0000,
        0x0000,0x0000,0x0000,0x0000,
        0x0000,0x0000,0x0000,0x0000,
        0x07F8,0x01FF,0xF001,0xF800,
        0x0FFC,0x03FF,0xF801,0xF800,
        0x0FFE,0x03FF,0xF803,0xFC00,
        0x0E0F,0x0380,0x0003,0x9C00,
        0x0E07,0x8380,0x0003,0x9C00,
        0x0E03,0xC380,0x0003,0x9C00,
        0x0E01,0xE380,0x0007,0x0E00,
        0x0E00,0xE380,0x0007,0x0E00,
        0x0E00,0xE380,0x0007,0x0E00,
        0x0E00,0xE380,0x0007,0x0E00,
        0x0E00,0xE380,0x000E,0x0700,
        0x0E00,0xE380,0x000E,0x0700,
        0x0E00,0xE380,0x000E,0x0700,
        0x0E00,0xE380,0x000E,0x0700,
        0x0E00,0xE380,0x001C,0x0380,
        0x0E00,0xE380,0x001C,0x0380,
        0x0E00,0xE380,0x001C,0x0380,
        0x0E00,0xE380,0x001C,0x0380,
        0x0E00,0xE380,0x0038,0x01C0,
        0x0E00,0xE380,0x003F,0xFFC0,
        0x0E00,0xE380,0x003F,0xFFC0,
        0x0E00,0xE380,0x003F,0xFFC0,
        0x0E00,0xE380,0x0070,0x00E0,
        0x0E00,0xE380,0x0070,0x00E0,
        0x0E00,0xE380,0x0070,0x00E0,
        0x0E01,0xE380,0x0070,0x00E0,
        0x0E03,0xC380,0x00E0,0x0070,
        0x0E07,0x8380,0x00E0,0x0070,
        0x0E0F,0x0380,0x00E0,0x0070,
        0x0FFF,0x03FF,0xF8E0,0x0070,
        0x0FFE,0x03FF,0xF9C0,0x0038,
        0x07FC,0x01FF,0xF1C0,0x0038,
        0x0000,0x0000,0x0000,0x0000,
        0x0000,0x0000,0x0000,0x0000,
        0x0000,0x0000,0x0000,0x0000,
        0x0000,0x0000,0x0000,0x0000,
        0x0000,0x0000,0x0000,0x0000,
        0x0000,0x0000,0x0000,0x0000,
        0x0000,0x0000,0x0000,0x0000,
        0x1FFF,0xFFFF,0xFFFF,0xFFFC,
        0x1000,0x0000,0x0000,0x0004,
        0x1000,0x0000,0x0000,0x0004,
        0x1000,0x0000,0x0000,0x0004,
```

```
0x1199,0x8987,0x8618,0x6784,
0x1199,0xC98C,0x061C,0xE6C4,
0x1199,0xE998,0x0B1E,0xE6C4,
0x1199,0x7998,0x0B17,0x6784,
0x1199,0x3998,0x1F92,0x6604,
0x1199,0x198C,0x1192,0x6604,
0x10F1,0x0987,0x9190,0x6604,
0x1000,0x0000,0x0000,0x0004,
0x1000,0x0000,0x0000,0x0004,
0x1000,0x0000,0x0000,0x0004,
0x1FFF,0xFFFF,0xFFFF,0xFFFC,
0x0000,0x0000,0x0000,0x0000,
0x0000,0x0000,0x0000,0x0000
```

```
/* Format_version=1, Width=64, Height=64, Depth=1, Valid_bits_per_item=16
*/
        0x0000,0x0000,0x0000,0x0000,
        0x0000,0x0000,0x0000,0x0000,
        0x0000,0x0000,0x0000,0x0000,
        0x0000,0x0000,0x0000,0x0000,
        0x1860,0x0060,0x0000,0x0018,
        0x1860,0x0060,0x0000,0x0018,
        0x1C60,0x0060,0x0000,0x0018,
        0x1C60,0x0060,0x0000,0x0018,
        0x1A61,0xE1F8,0x1F0F,0x81D8,
        0x1A63,0x3060,0x3180,0xC338,
        0x1966,0x1860,0x6000,0xC618,
        0x1967,0xF860,0x6007,0xC618,
        0x18E6,0x0060,0x600C,0xC618,
        0x18E6,0x0060,0x6018,0xC618,
        0x1863,0x1866,0x3198,0xC338,
        0x1861,0xF03C,0x1F0F,0x61D8,
        0x0000,0x0000,0x0000,0x0000,
        0x0000,0x0000,0x0000,0x0000,
        0x0000,0x0000,0x0000,0x0000,
        0x0000,0x0000,0x0000,0x0000,
        0x0000,0x0000,0x0000,0x0000,
        0x0000,0x0000,0x0000,0x0000,
        0x0000,0x0000,0x0000,0x0000,
        0x0000,0x0000,0x0000,0x0000,
        0x0000,0x0000,0x0000,0x0000,
        0x0000,0x0000,0x0000,0x0000,
        0x001F,0x001F,0x803F,0x0000,
        0x0011,0x0010,0x8021,0x0000,
        0x0011,0x0010,0x8021,0x0000,
        0x0011,0x0010,0x8021,0x0000,
        0x381F,0x001F,0x803F,0x001C,
        0x2804,0x0004,0x0004,0x0014,
        0x2FFF,0xFFFF,0xFFFF,0xFFF4,
        0x2800,0x4020,0x0002,0x0014,
        0x3800,0x4020,0x0002,0x001C,
        0x0000,0x4020,0x0002,0x0000,
        0x0000,0x4020,0x0002,0x0000,
        0x0001,0xF0F8,0x000F,0x8000,
        0x0001,0x1088,0x0008,0x8000,
        0x0001,0x1088,0x0008,0x8000,
        0x0001,0x1088,0x0008,0x8000,
        0x0001,0xF0F8,0x000F,0x8000,
        0x0000,0x4020,0x0002,0x0000,
        0x01E0,0x4020,0xF002,0x0F00,
        0x0120,0x4020,0x9002,0x0900,
        0x01E0,0x4020,0xF002,0x0F00,
        0x0080,0x4020,0x4002,0x0200,
        0x0080,0x4020,0x4002,0x0200,
        0x0FFF,0xF0FF,0xFF0F,0xFFF0,
        0x0221,0x0041,0x1001,0x0880,
        0x0221,0x0041,0x1001,0x0880,
```

```
0x0F7B,0xC0F7,0xBC07,0xBDE0,
0x094A,0x4094,0xA404,0xA520,
0x0F7B,0xC0F7,0xBC07,0xBDE0,
0x0200,0x0000,0x0000,0x0040,
0x0200,0x0000,0x0000,0x0040,
0x3FFF,0xFFFC,0x3FFF,0xFFFC,
0x0882,0x2088,0x0882,0x2088,
0x0882,0x2088,0x0882,0x2088,
0x3DEF,0x7BDE,0x3DEF,0x7BDE,
0x2529,0x4A52,0x2529,0x4A52,
0x3DEF,0x7BDE,0x3DEF,0x7BDE,
0x0000,0x0000,0x0000,0x0000,
0x0000,0x0000,0x0000,0x0000
```

```
/* Format_version=1, Width=64, Height=64, Depth=1, Valid_bits_per_item=16
*/
        0x0000,0x0000,0x0000,0x0000,
        0x0000,0x0000,0x0000,0x0000,
        0x0000,0x0000,0x0000,0x0000,
        0x3FFF,0xFFFF,0xFFFF,0xFFFC,
        0x3FFF,0xFFFF,0xFFFF,0xFFFC,
        0x3FFF,0xFFFF,0xFFFF,0xFFFC,
        0x3FFF,0xFFFF,0xFFFF,0xFFFC,
        0x3FFF,0xFFFF,0xFFFF,0xFFFC,
        0x3FFF,0xFFFF,0xFFFF,0xFFFC,
        0x3FFF,0xFFFF,0xFFFF,0xFFFC,
        0x3FFF,0xFFFF,0xFFFF,0xFFFC,
        0x3FFF,0xFFFF,0xFFFF,0xFFFC,
        0x3FFF,0xFFFF,0xFFFF,0xFFFC,
        0x3FFF,0xFFFF,0xFFFF,0xFFFC,
        0x3FFF,0xFFFF,0xFFFF,0xFFFC,
        0x3FFF,0xFFFF,0xFFFF,0xFFFC,
        0x3FFF,0xFFFF,0xFFFF,0xFFFC,
        0x0000,0x0000,0x0000,0x0000,
        0x0000,0x0000,0x0000,0x0000,
        0x0000,0x0000,0x0000,0x0000,
        0x0000,0x0000,0x0000,0x0000,
        0x0000,0x0000,0x0000,0x0000,
        0x0000,0x0000,0x0000,0x0000,
        0x0000,0x0000,0x0000,0x0000,
        0x0000,0x0000,0x0000,0x0000,
        0x0000,0x0000,0x0000,0x0000,
        0x001F,0x001F,0x803F,0x0000,
        0x0011,0x0010,0x8021,0x0000,
        0x0011,0x0010,0x8021,0x0000,
        0x0011,0x0010,0x8021,0x0000,
        0x381F,0x001F,0x803F,0x001C,
        0x2804,0x0004,0x0004,0x0014,
        0x2FFF,0xFFFF,0xFFFF,0xFFF4,
        0x2800,0x4020,0x0002,0x0014,
        0x3800,0x4020,0x0002,0x001C,
        0x0000,0x4020,0x0002,0x0000,
        0x0000,0x4020,0x0002,0x0000,
        0x0001,0xF0F8,0x000F,0x8000,
        0x0001,0x1088,0x0008,0x8000,
        0x0001,0x1088,0x0008,0x8000,
        0x0001,0x1088,0x0008,0x8000,
        0x0001,0xF0F8,0x000F,0x8000,
        0x0000,0x4020,0x0002,0x0000,
        0x01E0,0x4020,0xF002,0x0F00,
        0x0120,0x4020,0x9002,0x0900,
        0x01E0,0x4020,0xF002,0x0F00,
        0x0080,0x4020,0x4002,0x0200,
        0x0080,0x4020,0x4002,0x0200,
        0x0FFF,0xF0FF,0xFF0F,0xFFF0,
        0x0221,0x0041,0x1001,0x0880,
        0x0221,0x0041,0x1001,0x0880,
```

```
0x0F7B,0xC0F7,0xBC07,0xBDE0,
0x094A,0x4094,0xA404,0xA520,
0x0F7B,0xC0F7,0xBC07,0xBDE0,
0x0200,0x0000,0x0000,0x0040,
0x0200,0x0000,0x0000,0x0040,
0x3FFF,0xFFFC,0x3FFF,0xFFFC,
0x0882,0x2088,0x0882,0x2088,
0x0882,0x2088,0x0882,0x2088,
0x3DEF,0x7BDE,0x3DEF,0x7BDE,
0x2529,0x4A52,0x2529,0x4A52,
0x3DEF,0x7BDE,0x3DEF,0x7BDE,
0x0000,0x0000,0x0000,0x0000,
0x0000,0x0000,0x0000,0x0000
```

```
/* Format_version=1, Width=32, Height=32, Depth=1, Valid_bits_per_item=16
*/
        0x0000,0x0000,
        0x7FFF,0xFFFC,
        0x4000,0x0006,
        0x4902,0x0016,
        0x4D77,0x7716,
        0x4D52,0x4176,
        0x4B72,0x4756,
        0x4B42,0x4556,
        0x4972,0x7776,
        0x4000,0x0006,
        0x5800,0x0036,
        0x5FFF,0xFFF6,
        0x5804,0x4036,
        0x400E,0xE006,
        0x4EEE,0xEEE6,
        0x4EE4,0x4EE6,
        0x4444,0x4446,
        0x4FFE,0xFFE6,
        0x4110,0x1106,
        0x4390,0x1386,
        0x43B8,0x3B86,
        0x5C38,0x3876,
        0x5C10,0x1076,
        0x4810,0x1026,
        0x4FFE,0xFFF6,
        0x4444,0x4446,
        0x4EEE,0xEEE6,
        0x4EEE,0xEEE6,
        0x4000,0x0006,
        0x7FFF,0xFFFE,
        0x3FFF,0xFFFE,
        0x0000,0x0000
```