Universidade Federal de Santa Catarina

Programa de Pós-graduação em Engenharia Elétrica

"Implementação de um Sistema de Cálculo de Campos Eletromagnéticos em Estações de Trabalho e Ambiente Unix"

Dissertação submetida à Universidade Federal de Santa Catarina para a obtenção do título de Mestre em Engenharia Elétrica

Luís Alberto Pereira

Florianópolis, julho de 1992.

Implementação de um Sistema de Cálculo de Campos Eletromagnéticos em Estações de Trabalho e Ambiente Unix.

Luís Alberto Pereira

Esta dissertação foi julgada adequada para a obtenção do título de Mestre em Engenharia, especialidade Engenharia Elétrica e aprovada na sua forma final pelo curso de Pós-graduação em Engenharia Elétrica.

Prof. João Pedro Assumpção Bastos
Orientador

Prof. João Pedro Assumpção Bastos
Coordenador do Curso de Pós-graduação
em Engenharia Elétrica

Banca Examinadora :

Prof. Paulo César da Silva Jucá, Mestre

Prof. Jairo Branco Machado, Doutor

Prof. Adroaldo Raizer, Doutor

Prof. João Pedro Assumpção Bastos, Dr. D'Etat

Aos meus pais, Cármen e Luiz.

Agradecimentos

Ao CNPq, pelo apoio financeiro na realização do presente trabalho.

Ao professor João Pedro Assumpção Bastos, pela orientação e amizade.

Aos professores do GRUCAD, pelas sugestões apresentadas no decorrer do trabalho.

Ao professor Adroaldo Raizer, pelas sugestões e correções feitas na redação da dissertação.

Aos colegas do GRUCAD, pela ajuda e amizade.

A todas as pessoas que de uma forma ou de outra colaboraram na realização deste trabalho.

Agradeço sobretudo à minha família pelo constante apoio, carinho e incentivo.

Índice

1. Resumo

Neste trabalho é apresentado a implementação de um sistema de cálculo de campos eletromagnéticos em duas dimensões para estações de trabalho e sistema operacional Unix. A biblioteca gráfica utilizada é a do sistema X-Window (Xlib) e o método matemático empregado é o método dos elementos finitos. Inicialmente são abordados os principais aspectos relacionados com a organização do sistema. Na seqüência é feita a descrição do sistema implementado, incluindo a interface gráfica. A seguir são apresentados exemplos de utilização do sistema e comparações com um sistema semelhante desenvolvido para computadores pessoais.

O sistema implementado pode ser usado em áreas de pesquisa, projeto e ensino. Sendo dotado de uma interace gráfica, o sistema também é muito fácil de ser usado.

II. Abstract

In this work is shown the development of a system for electromagnetic field calculation in two dimensions for worstations and Unix operational system. The graphical library used is the X-Window Library (XLib). The method used is the finite element method. First, the main aspects related to the system organization are discussed. In the following, the system implementation is described, including a graphical interface. In the last part, examples and comparisons with a system developped for personal computers are shown.

The system which was implemented can be used in research, design and teaching areas. Having a graphical interface, the system is of very easy use.

III. Introdução

A determinação de campos eletromagnéticos é um assunto que ao longo dos anos tem despertado interesse não só de engenheiros como também de pesquisadores de áreas afins. Embora as equações que regem os fenômenos eletromagnéticos sejam há muito conhecidas, sua solução analítica é restrita a casos específicos e bem conhecidos. Desta forma, a tendência atual na solução destes fenômenos é o uso de técnicas numéricas. O desenvolvimento de métodos matemáticos adequados e o surgimento de equipamentos computacionais mais poderosos levou ao desenvolvimento de sistemas de cálculo capazes de determinar de forma satisfatória os campos associados a certos fenômenos eletromagnéticos [12].

As técnicas numéricas desenvolvidas substituem com vantagens as técnicas manuais e, quando implementadas em computadores adequados, permitem a visualização gráfica imediata dos resultados. Desta forma, a sua essência consiste em determinar as grandezas de interesse a um certo fenômeno com maior exatidão, propiciando não só um projeto mais racional dos dispositivos, mas também uma análise mais realista. Eles atualmente representam uma ferramenta poderosa e indispensável em áreas de projeto e instituições de pesquisa e ensino[1,2,12].

Pode-se afirmar que o desenvolvimento destes sistemas nos últimos anos é fruto não só do esforço dos pesquisadores no sentido de desenvolver ferramentas robustas e eficientes, mas também da popularização do uso do computador e dos avanços tecnológicos desta área [2,12,16].

O método matemático mais utilizado na resolução das equações diferenciais que regem os fenômenos eletromagnéticos é o método dos elementos finitos, devido às vantagens que ele oferece em relação a outros [1,2,12,13,14,15,16].

Sendo o desenvolvimento deste tipo de sistema muito ligado aos recursos computacionais disponíveis, a escolha de um equipamento é de fundamental importância. Um dos equipamentos que pode ser utilizado é o PC (Personal Computer), os quais na maioria das vezes usa o sistema operacional DOS (Disk Operational System). Embora de fácil manuseio e aquisição, ele impõe algumas restrições que impedem que sistemas mais potentes sejam desenvolvidos.

As principais dificuldades se referem ao limites de memória que o sistema operacional impõe (640 kbytes) e a velocidade de processamento. Muito embora já existam computadores pessoais e processadores que estão aptos a usarem mais do que este limite, uma aplicação normal no DOS não pode ultrapassar o limite de 640 kbytes, já que o mesmo é uma imposição do sistema operacional e não do hardware da máquina. Ainda que o limite de memória possa ser resolvido, por exemplo, a partir da adoção de um outro sistema operacional (OS/2, Windows, Xenix, etc..), isto não têm sido uma prática comum entre os usuários, já que implica custos. Além disso as exigências do usuário no que se refere a interface dos sistemas têm aumentado a partir da popularização do uso de interfaces gráficas[i] em substituição às interfaces baseadas em linhas de comando. Neste aspecto, deve-se ressaltar que não existe no momento um padrão definitivo de interface gráfica para computadores pessoais [23,27,29,30].

Por outro lado, o uso crescente de estações de trabalho no Brasil tem motivado o desenvolvimento de sistemas de cálculo de campos adaptados a este tipo de equipamento. Dotadas de capacidade de memória, velocidade de processamento e recursos gráficos superiores aos PC's, elas representam atualmente uma excelente opção para o desenvolvimento de sistemas de cálculo de campos. Contudo, o preço de uma estação ainda é superior ao de um PC. O sistema operacional empregado pela grande maioria de estações é o Unix [27]. Elas também possuem uma interface gráfica que facilitam o uso não só do sistema operacional como também dos programas de aplicação. O sistema gráfico adotado pela totalidade de fabricantes é o X-Window (Versão 11), desenvolvido para trabalhar em redes [5,6,17].

O presente trabalho consiste na implantação de um sistema de cálculo de campos em estações de trabalho e ambiente Unix. O trabalho foi desenvolvido de forma a utilizar ao máximo os recursos em termos de capacidade de memória e performance gráfica do equipamento. A utilização das bibliotecas gráficas do sistema X-Window exige que a programação da parte gráfica seja feita em linguagem C, já que as chamadas às bibliotecas só são possíveis a partir desta linguagem. Procurou-se também utilizar rotinas matemáticas escritas em Fortran já existentes, sempre que possível, a fim de que as mesmas não precisassem ser reescritas. Para a implantação do sistema dentro desta filosofia foi necessário um estudo de ambas as linguagens a fim de posibilitar o uso de rotinas escritas em C

[i] é uma forma gráfica de interação com o usuário implantada através de janelas onde são mostrados objetos gráficos tais como menus, botões e ícones; este padrão contrasta com as interfaces por meio de linhas de comando.

e de rotinas escritas em Fortran num mesmo programa. Na implantação da interface gráfica utilizou-se um pacote de rotinas gráficas chamado de XView Toolkit, além das rotinas existentes na biblioteca do sistema X-Window (Xlib). Para tanto, também foi necessário um estudo do funcionamento do sistema X-Window, a fim de que a programação gráfica dentro deste sistema pudesse ser feita. Além disso, foi necessário utilizar um estilo de programação diferente do estilo tradicional de programação, chamado de programação com notificador, o qual controla de forma centralizada o fluxo de operações do programa [3,4,5,6].

Alternativamente, poderia-se implantar o sistema da mesma forma que o mesmo se encontra atualmente em PC's utilizando a biblioteca gráfica GKS e somente a linguagem Fortran, dentro do estilo tradicional de programação. Embora menos trabalhosa, esta abordagem leva a uma utilização apenas parcial dos recursos disponíveis da estação, tal como a utilização de uma interface gráfica (composta de menus, ícones, botões, painéis, etc..), integração ao sistema de janelas e facilidade de interação com o usuário, entre outros. Os programas implantados dentro desta filosofia possuem um funcionamento idêntico tanto na estação como no PC, estando desta forma limitados pelos recursos de entrada e saída de dados oferecidos pela linguagem Fortran e pelos recursos gráficos oferecidos pelo GKS. Por outro lado, esta abordagem permite que os programas criados para o PC rodem na estação praticamente sem alteração. Pelos motivos expostos optou-se pela implantação de uma forma diferente que a utilizada em PC's, em favor da melhor utilização dos recursos da estação.

A implantação do sistema em estações também visa a desenvolver uma ferramenta capaz de analisar estruturas até o momento impossíveis de serem analisadas com o sistema para PC's, tais como estruturas com elevado número de elementos, bem como obter resultados mais precisos, uma vez que se dispõe de um maior espaço de memória.

1. Sistemas de Cálculo de Campos Baseados em Técnicas de CAD

1.1 Introdução

Neste capítulo serão descritos os aspectos e conceitos fundamentais dos sistemas de cálculo de campos baseados em técnicas de CAD (Computer Aided Design) voltados para a solução de problemas em engenharia elétrica. Também serão abordadas as etapas básicas de processamento e as principais configurações nas quais os mesmos são implantados. Finalmente será apresentado o sistema EFCAD desenvolvido no GRUCAD (Grupo de Concepção e Análise de Dispositivos Eletromagnéticos) para computadores pessoais.

Este tipo de sistema de cálculo tem assumido uma grande importância atualmente na área de engenharia, dada a facilidade que oferecem para a determinação de campos elétricos e magnéticos nas mais diversas configurações e estruturas. Desta forma o conhecimento de sua organização e funcionamento é importante quando se pretende implantar o sistema em outros equipamentos.

1.2 Etapas de Processamento

Os sistemas de cálculo de campos eletromagnéticos geralmente podem ser divididos em 3 etapas básicas [2,32]:

- pré-processamento;
- processamento;
- pós-processamento.

Estas três etapas estão representadas esquematicamente na figura 1.1. Além disso, eles podem incluir ainda uma base de dados para facilitar a troca de informações entre as várias etapas.

A seguir será feita uma descrição da função de cada uma destas etapas e como as mesmas estão interligadas dentro do sistema.

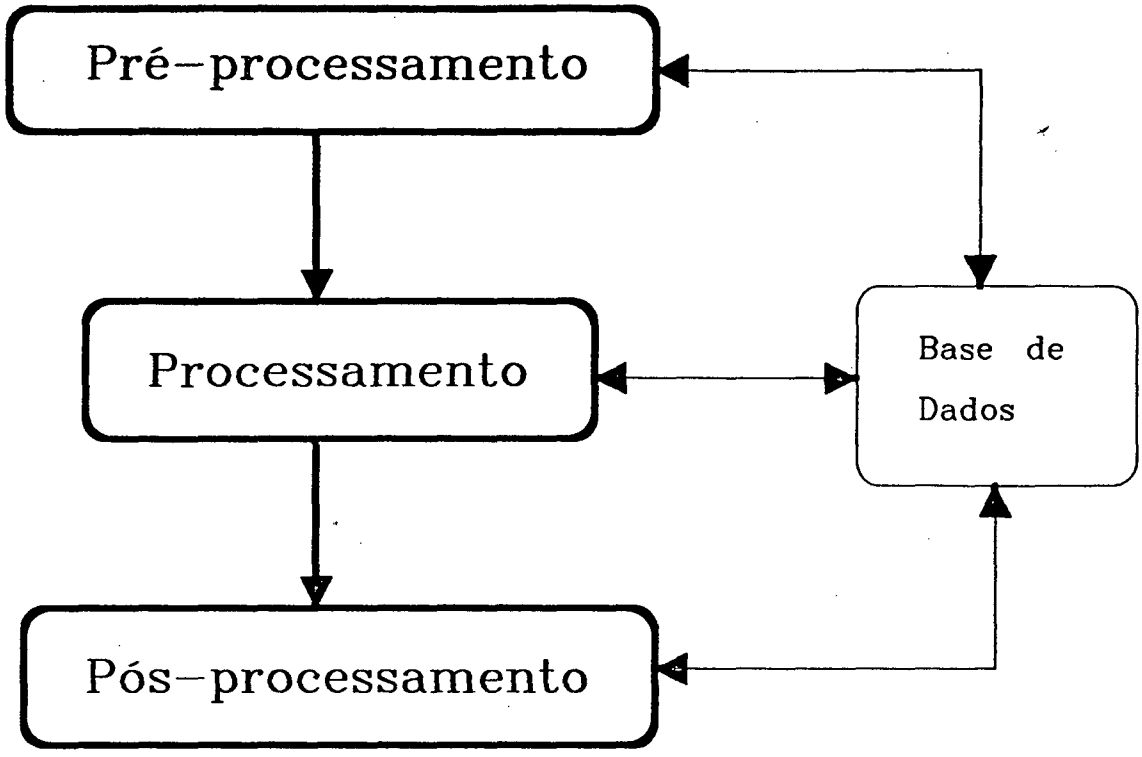

figura 1.1 - Etapas de processamento.

1.2.1 Pré-processamento

Esta etapa consiste na descrição geométrica da estrutura sob análise. Também são definidas as propriedades físicas de cada região que a compõe, caracterizados os materiais, fontes de campo e condições de contorno impostas à geometria.

O pré-processador[1] pode incluir também um módulo de discretização da estrutura, o qual criará os elementos que o processador utilizará na solução do problema. Este módulo também é conhecido como gerador de malha ou malhador. A geração da malha pode ser feita de forma automática ou manual [2]. Nos sistemas modernos a geração da malha é feita de forma totalmente automática, existindo, no entanto, a possibilidade de o usuário alterar alguns parâmetros da malha, como, por exemplo, o tamanho dos elementos, inserção de nós, supressão de nós, etc. Há também a possibilidade de inclusão de um módulo de otimização de malha, o qual gerará uma malha que possibilite uma minimização de erros nos resultados de cál-

[1] Os termos pré-processador, processador e pós-processador se referem aqui aos módulos que realizam as etapas de pré-processamento, processamento e pós-processamento, respectivamente.

culo. Visto que o erro em geral não é uniforme em todas as regiões da estrutura este módulo também permite uma uniformização do erro dentro de uma certa região [12,15,16].

Nesta etapa a interação com o usuário é importante, pois é a parte que geralmente demanda mais tempo do usuário. Uma interação com o usuário pouco eficiente fará com que a tarefa de definição de dados da estrutura seja longa e tediosa. Desta forma, devem ser feitos esforços para tornar a etapa de pré-processamento eficiente, a fim de que o sistema como um todo também seja eficiente.

A implementação de um pré-processamento eficiente também está condicionada à linguagem, ao sistema operacional e o equipamento utilizados. Equipamentos, tais como estações de trabalho, e o emprego de linguagens como o C e C++, propiciam recursos que tornam esta etapa mais fácil, já que possuem recursos que facilitam a interação com o usuário, entre outros. Geralmente nesta etapa também é utilizada uma biblioteca gráfica, tal como GKS, PHIGS, X-Window, a fim de propiciar uma melhor visualização e interação com o usuário.

1.2.2 Processamento

Nesta etapa as equações diferenciais parciais que caracterizam o fenômeno são efetivamente resolvidas. A partir da discretização feita na etapa de pré-processamento, é montado um sistema de equações lineares, ou não-lineares, que será então resolvido pelo processador. O resultado fornecido pelo processador será o valor das grandezas em estudo em determinados pontos da estrutura [2,13,14,15].

Sendo o processamento a etapa central do sistema, todo o esforço deve ser feito para que o mesmo seja robusto, veloz e eficiente.

Os resultados gerados pelo processador serão em seguida passados ao pós-processador numa forma conveniente para o mesmo.

A máquina usada na etapa de processamento deve possuir a velocidade de cálculo e memória adequados, sendo geralmente a máquina com melhor performance em cálculos do sistema quando se trabalha em um ambiente distribuído.

Nesta etapa também está implantado o método de resolução matemática das equações que regem o fenômeno. O método mais freqüentemente usado é o método dos elementos finitos [1,2,12,13,14,15,16].

Sendo este método já conhecido há muito tempo, ele só começou a ser utilizado quando se dispôs de equipamentos com velocidade e memória suficiente para a sua implantação prática. Ele é considerado hoje como um método de resolução de equações diferenciais parciais e que pode ser empregado em vários campo da matemática aplicada, tais como a mecânica dos fluidos, transmissão de calor, cálculo estrutural, além de dispositivos eletromagnéticos [14,12]. Ele passou a ser usado com maior ênfase a partir do início dos anos 70, sendo a sua evolução intimamente ligada aos avanços feitos nos equipamentos computacionais.

O princípio do método reside na discretização de uma dada região em domínios menores chamados de elementos. Dentro de cada elemento a função desconhecida é aproximada por uma função de aproximação, a qual fornece valores aproximados para as grandezas de interesse dentro de cada elemento. Os elementos podem ser linhas, triângulos, quadriláteros, tetraedros, etc...[14,15].

Com a aplicação do método obtém-se valores aproximados das grandezas que verificam as equacões diferenciais e as condições de contorno impostas ao caso.

1.2.3 Pós-processamento

A partir de dados fornecidos pelo processador, o pós-processador fará uma exploração gráfica e numérica dos resultados, permitindo ao usuário não só um conhecimento do valor das grandezas mas também uma visualização e interpretação melhor dos resultados [12,13,32,33].

O pós-processador pode gerar a seguinte saída, entre outras:
- valores locais de grandezas tais como indução;
- valores globais de grandezas como fluxo, força, energia e indutância;
- linhas eqüipotenciais;
- indução sobre uma certa região.

O pós-processador também deve ser capaz de calcular outras grandezas a partir dos valores fornecidos pelo processador.

Quando a visualização do fenômeno for importante para o usuário nesta etapa, é essencial que se utilizem máquinas que permitam uma boa performance gráfica. Nesta etapa é importante também que a interação com o usuário seja eficiente.

Com o uso cada vez mais freqüente de redes de computadores, pode-se distribuir as três etapas até aqui discutidas em diversas máquinas pela rede, fazendo um uso mais racional das potencialidades individuais de cada máquina. Assim, pode-se usar máquinas com boa velocidade de cálculo e memória para fazer o processamento e máquinas com alta resolução gráfica para as etapas de pré e pós-processamento.

1.3 Estruturas de Sistemas de Cálculo de Campos

Do ponto de vista da implantação em um sistema computacional, as três etapas básicas podem estar agrupadas de várias maneiras em um sistema de cálculo de campos [2,33]. Na figura 1.2 estão mostradas três estruturas comumente encontradas. Na figura 1.2a está mostrada uma estrutura onde as 3 etapas estão agrupadas em um único bloco. Esta estrutura também é chamada de monomodular e apresenta a vantagem do rápido acesso às informações e dados necessários a cada etapa de processamento. Por outro lado, sua implantação exige equipamentos de maior capacidade de memória. A evolução do sistema também fica dificultada, pois o sistema deve sempre ser alterado na sua totalidade.

No configuração mostrada na figura 1.2b as etapas de pré e pós-processamento foram reunidas num único bloco. A principal vantagem desta estrutura é o fato de o processamento poder ser feito em uma máquina de alta velocidade enquanto que as etapas de pré e pós processamento poderem ser feitas em máquinas de alta performance gráfica. A ligação entre as etapas geralmente é feita por meio de arquivos.

Na configuração mostrada na figura 1.2c cada uma das etapas constitui um bloco separado. Esta configuração é bastante usada para a implementação de sistemas de cálculo de campos em equipamentos de menor capacidade. Como os módulos estão separados esta configuração também facilita o desenvolvimento do sistema.

De um modo geral, cada uma das etapas até aqui descritas pode também ser subdividida em mais de um módulo. Por exemplo, a etapa de pré-processa-

mento pode ser subdividida entre um módulo de descrição geométrica e um módulo de geração de malha, conforme mostrado na figura 1.3.

A grande maioria dos sistemas de cálculo de campos atualmente emprega uma estrutura multimodular.

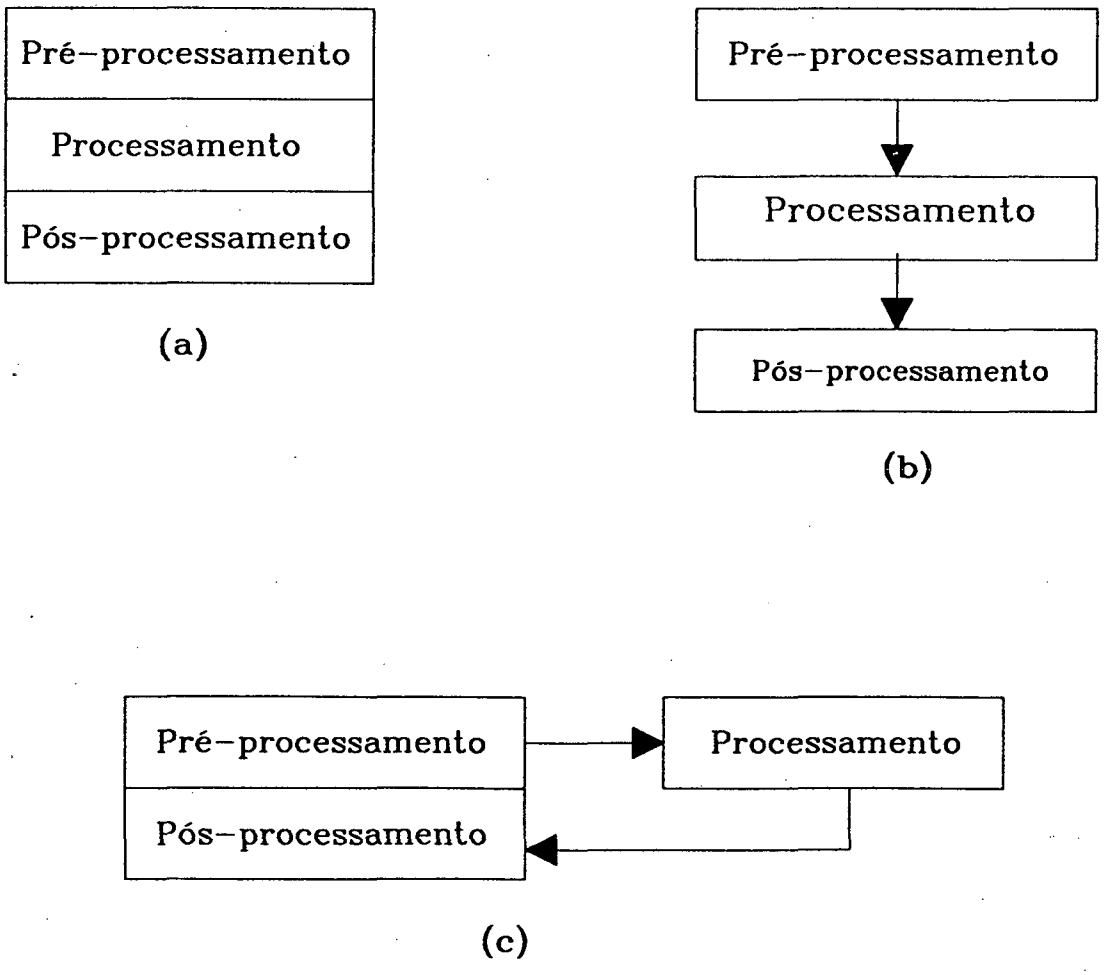

figura 1.2 - Estruturas dos sistemas de cálculo de campos

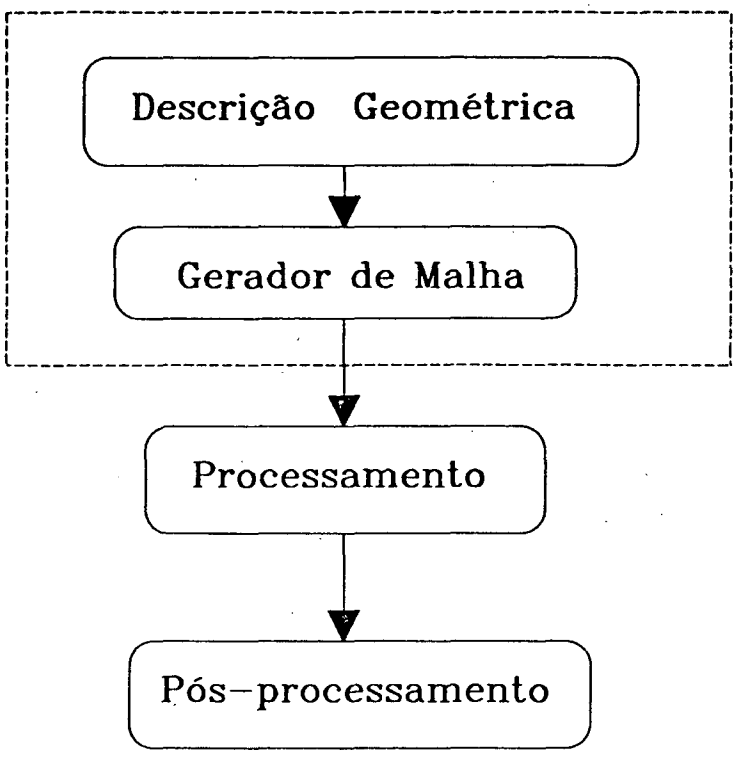

figura 1.3 - Etapa de pré-processamento subdividida

1.4 Sistema EFCAD para PC's [1,35]

Neste ítem será apresentado o sistema de cálculo de campos EFCAD, desenvolvido no GRUCAD (Grupo de Concepção de Análise de Dispositivos Eletromagnéticos).

O sistema EFCAD é um software de cálculo de estruturas eletromagnéticas em duas dimensões e de propósitos gerais que foi concebido originalmente para trabalhar em computadores pessoais. A sua primeira versão foi concebida há cerca de 6 anos e, ao longo dos anos, ele vem sendo aperfeiçoado e acrescido de novas possibilidades e extensões. Este sistema está sendo usado atualmente por vários grupos de pesquisa e empresas não só do Brasil mas também do exterior.

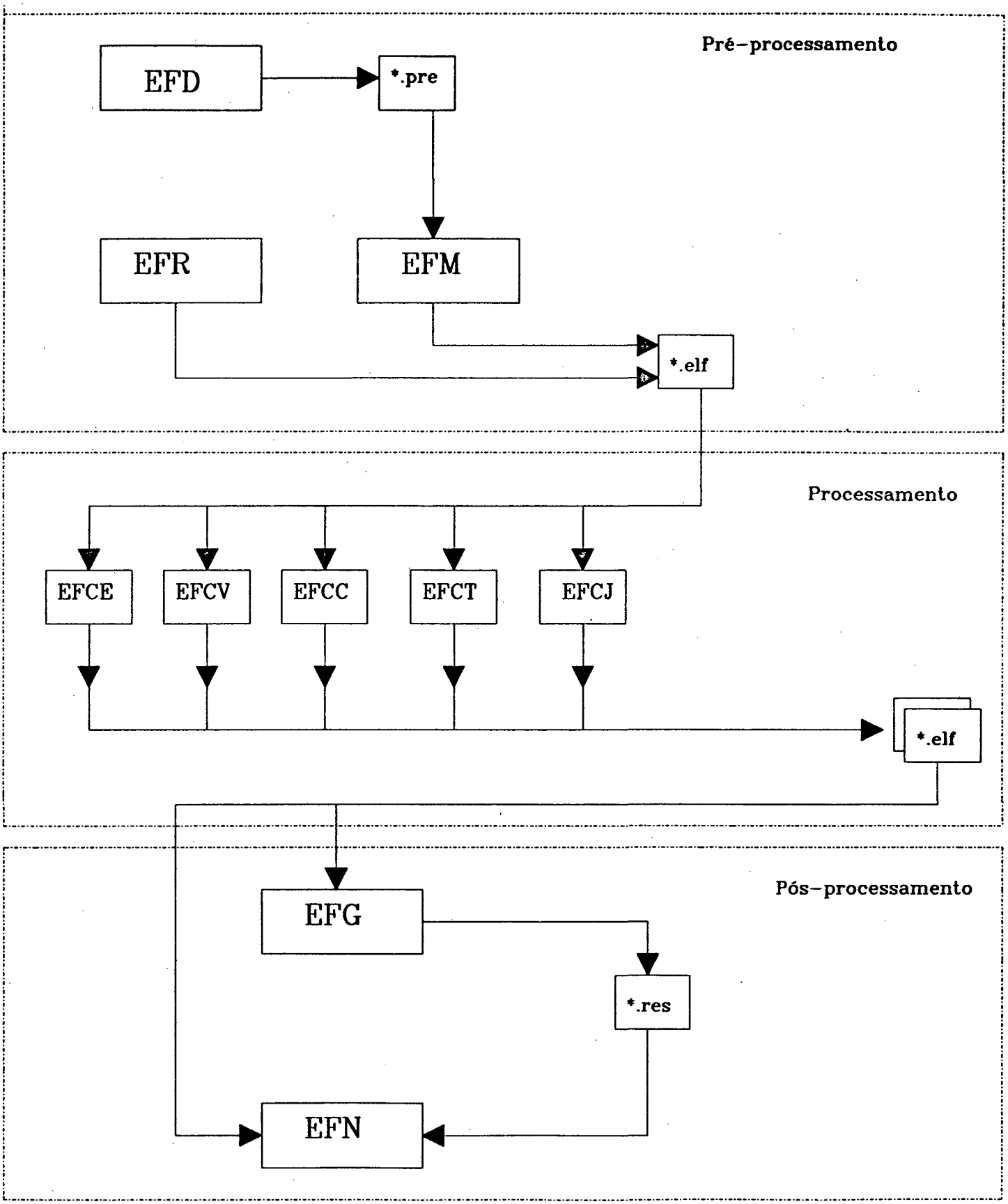

figura 1.4 - Estrutura geral do EFCAD.

Na verdade, um sistema como o EFCAD é de extrema importância em qualquer área que se proponha a analisar e projetar estruturas eletromagnéticas, tais como máquinas elétricas, contatores, transformadores, cabos, etc... O sistema permite que se obtenham resultados muito mais precisos do que os obtidos com fórmulas empíricas e aproximadas, dando ao usuário muito mais segurança no projeto e análise. Além disso, é possível que se façam projetos sem que protótipos sejam de fato construídos, reduzindo custos e acelerando o desenvolvimento e a otimização de dispositivos.

O EFCAD utiliza o método dos elementos finitos para a resolução das equações que regem os fenômenos.

Os principais tipos de problemas que podem ser tratados com o EFCAD são:

- problemas estáticos de potencial ;
- problemas de estruturas eletromagnéticas com velocidade;
- problemas cujas fontes estão acopladas a circuitos externos e alimentados em corrente;
- problemas cujas fontes estão acopladas a circuitos externos e alimentados em tensão;
- problemas de penetração de campos em partes condutoras de estruturas;
- problemas com fontes em regime permanente senoidal;
- problemas envolvendo correntes induzidas;
- problemas axi-simétricos;
- problemas não-lineares;
- problemas envolvendo resolução passo a passo no tempo.

1.4.1 Estrutura Geral

O sistema EFCAD é constituído de uma estrutura multimodular interligada por meio de arquivos. Ele foi concebido originalmente para trabalhar no sistema operacional DOS e os módulos que o compõem foram escritos em linguagem Fortran, utilizando uma biblioteca gráfica compatível com o padrão GKS. Cada módulo destina-se a uma tarefa específica dentro do sistema. A sua estrutura geral está mostrada na figura 1.4.

Esta estrutura é extremamente favorável para a sua implantação em PC's pois cada módulo pode utilizar praticamente toda a memória disponível da máquina quando for executado. Além disso, esta estrutura também facilita o desenvolvimento do sistema, já que cada módulo é independente.

A figura 1.4 também mostra em linhas pontilhadas os módulos que constituem as etapas de pré-processamento, processamento e pós-processamento.

A seguir será feita uma descrição breve de cada um destes módulos.

1.4.2 Módulo de Entrada de Dados

Na primeira versão do EFCAD este módulo trazia alguns incovenientes, uma vez que a interação com o usuário era totalmente baseada em entradas pelo teclado e o arquivo gerado era do tipo não-formatado. Assim, o usuário raramente se sentia encorajado a alterá-lo para moldá-lo às suas necessidades. Atualmente o formato do arquivo foi alterado para o tipo formatado (caracteres ASCII) dentro de um modelo bastante simples de representação de retas e arcos de circunferências. Atualmente, é mais fácil para o usuário desenvolver um programa de entrada de dados adaptado às suas necessidades. O módulo de entrada de dados (chamado de EFD) também foi alterado para que as entradas também possam ser feitas de forma gráfica utilizando um mouse. Além disso, também são oferecidas vários formas de entradas de retas e arcos de circunferência. Desta forma, o usuário pode fazer entradas tanto pelo teclado como fazer indicações sobre uma página gráfica na tela, facilitando muito a entrada de dados.

Todavia, não é possível trabalhar com entrada de caracteres e entradas gráficas simultaneamente, pois a tela é ora gráfica, ora numérica. O usuário precisa optar por qual das formas deseja trabalhar e ficar alternando entre uma e outra forma.

O arquivo utilizado pelo EFD (com extensão *.pre*, também chamado de arquivo *.pre*) também pode ser gerado por qualquer outro programa que o usuário desejar utilizar para uma aplicação particular. Ele pode inclusive ser gerado por programas comerciais que geram arquivos no formato **.DFX*, como o VersaCAD e AutoCAD. O formato *.pre* nestes casos é obtido após a passagem por um programa conversor chamado de EFCONV. Este processo está mostrado na figura 1.5.

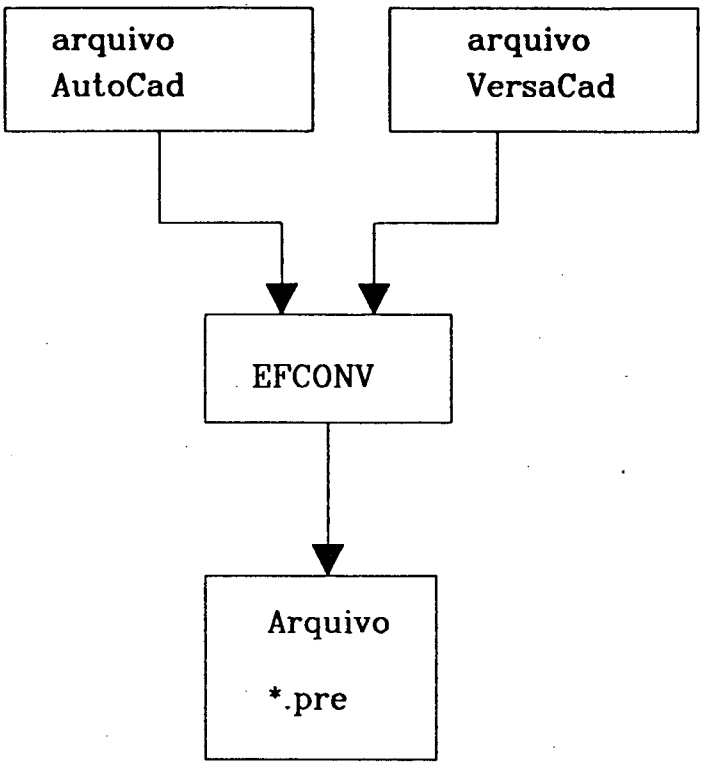

figura 1.5 - Conversão de arquivos.

O formato do arquivo *.pre* pode ser melhor explicado por meio do seguinte exemplo:

```
1000.0
1  3  0
4  2  0
3  2  1
3  2  0
1  4  0
f
x1  y1
x2  y2
x3  y3
x4  y4
f
xc1 yc1
f
```

Como se trata de um arquivo ASCII os dados podem ser separados tanto por vírgula como por espaço. O arquivo também pode ser modificado por um edi-

tor de texto qualquer. A estrutura que este arquivo representa está mostrada na figura 1.6.

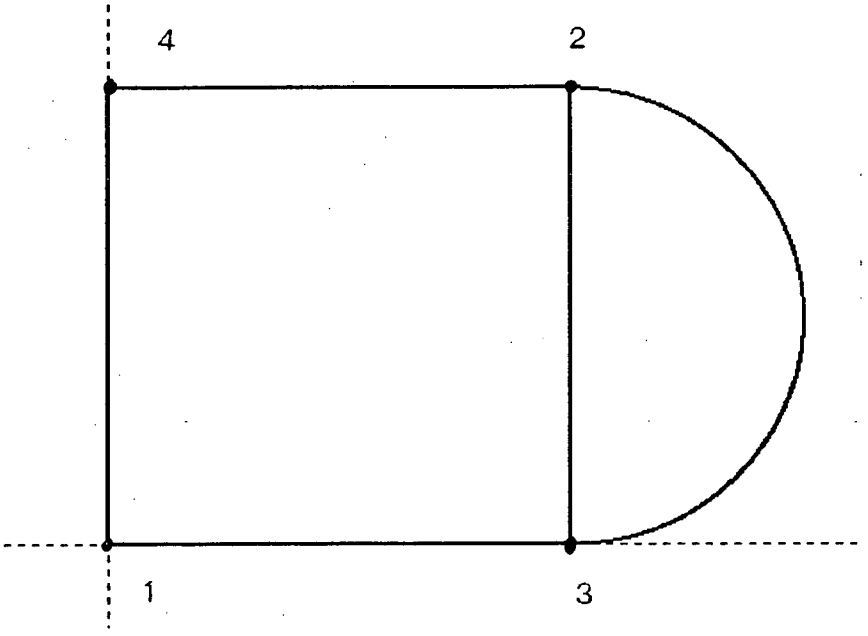

figura 1.6 - Estrutura de exemplo.

A primeira linha do arquivo define o fator de escala dos dados que se seguem, ou seja os dados serão divididos por este valor quando lidos. No caso acima, sendo o fator 1000, as coordenadas serão dadas em milímetros e, após serem divididas por 1000, estarão no sistema internacional de unidades.

As próximas cinco linhas definem os segmentos de reta e arcos de circunferência. O primeiro número indica o ponto inicial, o segundo o ponto final e o terceiro indica o tipo de segmento. Quando este terceiro número for zero indica que se trata de um segmento de reta, quando for diferente de zero indica que se trata de um arco de circunferência, sendo que o mesmo número indica o número do centro. Desta forma, na segunda linha tem-se um segmento de reta que começa no ponto 1 e vai até o ponto 3. Na terceira linha tem-se um segmento de reta que vai do ponto 4 até o 2. Na quarta linha tem-se um arco de circunferência que começa no ponto 3 e vai ate o ponto 2. O centro deste arco de circunferência é o centro número 1. Este arco de circunferência é sempre considerado no sentido anti-horário. Note-se ainda que no caso de segmento de retas a ordem dos pontos é irrelevante.

A letra *f* colocada na sexta linha indica que os próximos dados se referem às coordenadas dos pontos. Assim, as próximas quatro linhas são as coordenadas dos pontos 1, 2, 3, e 4. A próxima letra *f* indica que os dados a seguir são as coordenadas dos centros. Como no caso acima só há um arco de circunferência, só há um centro (xc1,yc1). A letra *f* no final indica o final do arquivo.

1.4.3 Módulo de Geração de Malha

O EFM é o programa responsável pela geração automática da malha. Ele trabalha com dados lidos em um arquivo formatado e escrito dentro de um formato padrão com extensão *.pre*. Os dados gerados pelo EFM são gravados em arquivo com extensão *.elf*, o qual é um arquivo não-formatado (arquivo binário) escrito numa forma particular. O arquivo *.elf* também pode ser lido e trabalhado pelo EFM.

Após a leitura do arquivo *.pre* contendo informações a respeito da geometria da estrutura, o usuário define os meios magnéticos, regiões contendo correntes, condições de contorno e periodicidade. Em seguida o programa discretiza automaticamente a estrutura e gera uma malha. As informações sobre a malha são armazenadas no arquivo com extensão *.elf*. Este arquivo será utilizado pelos módulos subsequentes de cálculo e exploração gráfica e numérica.

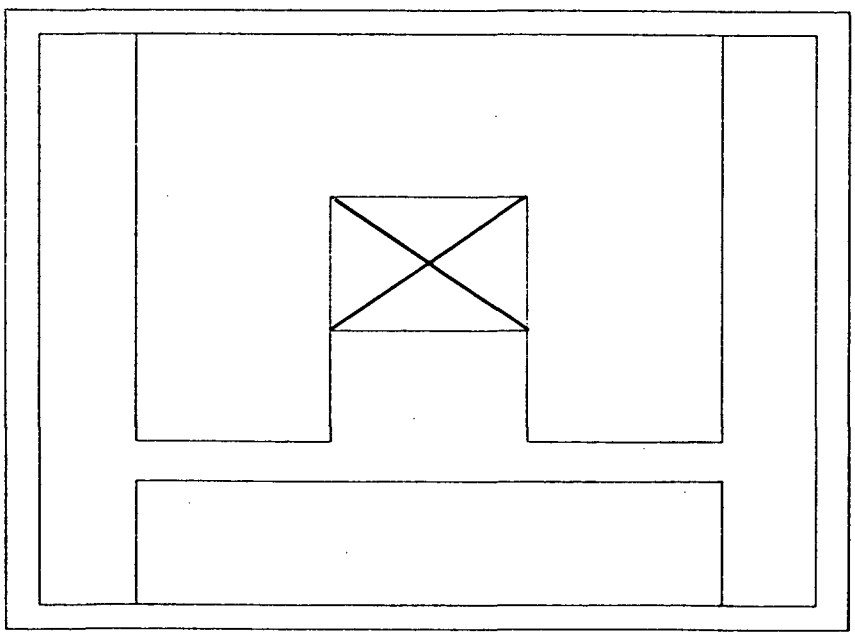

figura 1.7 - Estrutura usada para gerar a malha

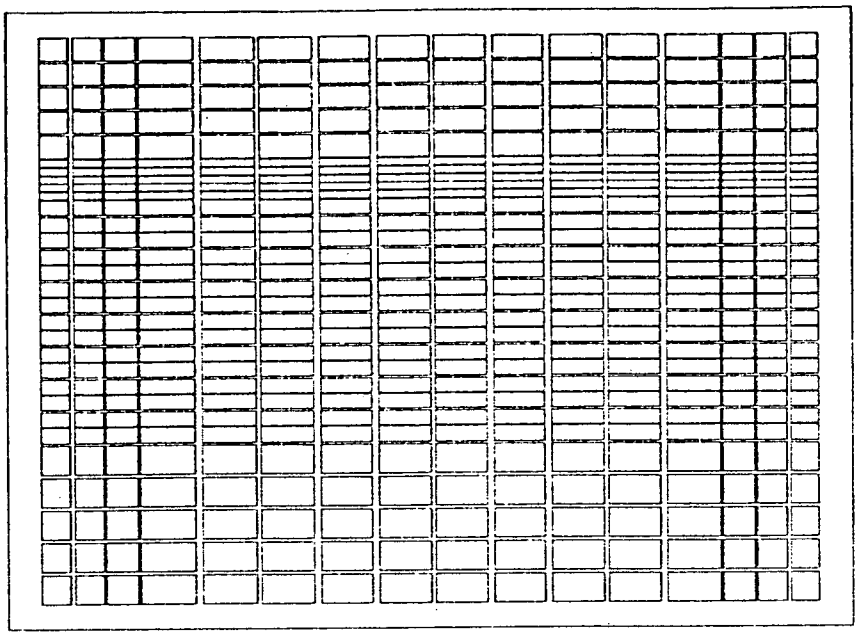

figura 1.8 - Exemplo de malha regular

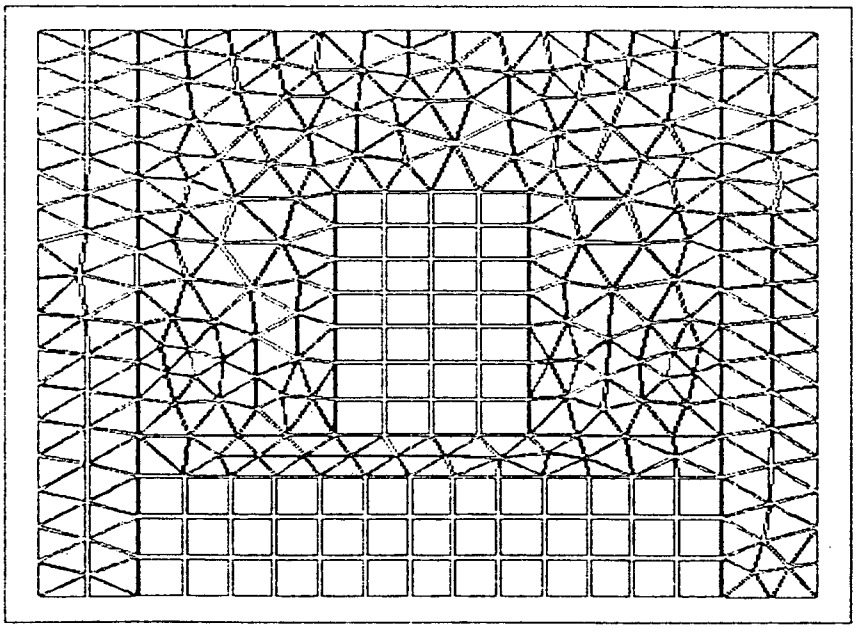

figura 1.9 - Exemplo de malha não regular

O malhador também inclui opções de edição da estrutura tais como eliminação de segmentos, união de segmentos por dois pontos, deslocamento de regiões, etc... Estas facilidades permitem ao usuário alterar a estrutura sem ter que recorrer ao programa EFD, agilizando desta forma a etapa de pré-processamento.

Existe também um módulo que faz a geração de malhas regulares e é usado quando se estuda uma estrutura cuja geometria também é regular. Este módulo é chamado de EFR. A figura 1.8 mostra o exemplo de uma malha regular enquanto que a figura 1.9 mostra uma malha não regular. Ambas foram geradas a partir da estrutura mostrada na figura 1.7.

1.4.4 Módulos de Cálculo do Sistema EFCAD para PC's

Os módulos de cálculo constituem o setor de processamento. Todos os módulos utilizam o método dos elementos finitos na resolução das equações diferenciais parciais. Eles trabalham com as informações contidas no arquivo *.elf* gerado pelos módulos de geração de malha EFM ou EFR. Após ter sido feito o cálculo é acrescentado ao arquivo os valores dos potenciais nos nós da malha, bem como uma informação indicando qual o módulo que gerou os potenciais.

Os principais módulos de cálculo são :

- EFCE, módulo que trata de problemas estáticos, ou seja de problemas invariantes no tempo;
- EFCC, módulo que trata casos onde as fontes são alimentadas por correntes em regime permanente senoidal e os meios não apresentam saturação. Para a resolução é empregada uma formulação em complexo;
- EFCJ, módulo em que a alimentação em corrente pode ter uma forma qualquer definida pelo usuário por meio de uma curva. A solução é feita passo a passo e pode-se empregar materiais com saturação;
- EFCT, módulo idêntico ao anterior mas com alimentação em tensão;
- EFCV, módulo que trata de estruturas cujas partes possuem velocidade.

Com os módulos descritos acima é possível estudar uma vasta gama de estruturas e fenômenos eletromagnéticos, sendo que nos 4 últimos a corrente induzida poderá ser levada em conta.

1.4.5 Módulos de Exploração Gráfica e Numérica

Os módulos de exploração gráfica (EFG) e exploração numérica (EFN) constituem a etapa de pós-processamento.

O módulo EFG lê as informações do arquivo *.elf* gerado pelos módulos de cálculo e faz um tratamento gráfico dos resultados, possibilitando ao usuário uma visualização extremamente útil do fenômeno. O EFG traça as curvas equipotenciais e a estrutura em estudo. É também possível a visualização da malha da estrutura bem como o traçado de uma escala de 15 cores para a a visualização das densidades de fluxo. Além disso, é possível obter-se curvas de indução em partes da estrutura.

O módulo EFG também gera um arquivo não-formatado, cuja extensão é *.res*, contendo informações que serão utilizadas pelo módulo EFN.

O módulo EFN, por sua vez, utiliza as informações contidas no arquivo *.res* para obter dados numéricos a respeito das grandezas de interesse, tais como indução, fluxo, força, etc... Este módulo pode apresentar na tela as equipotenciais e a estrutura. O usuário pode também indicar em quais os pontos da estrutura deve ser determinado os valores das grandezas.

Os resultados de ambos os módulos também podem ser impressos em uma impressora ou um plotter.

Além dos módulos de entrada de dados, geração de malha, cálculo, exploração gráfica e numérica, existe um módulo chamado de EFP que se destina a criação, alteração e eliminação de materiais que constituirão os meios. Este módulo utiliza um arquivo não-formatado chamado de *efmat.dat* que também é usado pelos demais módulos.

1.5 Conclusão

Neste capítulo foi mostrada a utilidade e importância de um sistema de cálculo de campos baseada em técnicas de CAD, bem como mostrado o sistema EF-CAD implantado em PC's. O desenvolvimento de um tal sistema está intimanente ligado aos recursos computacionais disponíveis e à forma de utilização dos mesmos. As estações de trabalho oferecem recursos computacionais e gráficos maiores que os computadores pessoais e, desta forma, o desenvolvimento de sistemas de cálculo para estações assume uma importância relevante, já que elas permitem desenvolver sistemas mais potentes. No entanto, para um aproveitamento integral das suas potencialidades é necessário o conhecimento das suas principas características.

Os principais aspectos dos equipamentos que devem ser considerados quando da implantação de sistemas de cálculo são:

- recursos do equipamento (memória, disco, etc..);
- sistema operacional;
- linguagem de programação;
- biblioteca gráfica.

No apêndice A são abordados os ítens acima, sendo dada ênfase especial aos relacionados com estações de trabalho.

No próximo capítulo será apresentado o sistema de cálculo de campos eletromagnéticos que foi desenvolvido em estações de trabalho.

2. Sistema EFCAD para Estações de Trabalho

2.1 Introdução

Neste capítulo será feita a descrição do sistema EFCAD que foi desenvolvido para estações de trabalho, também chamado de sistema EFCAD para estações de trabalho. As estações utilizadas neste trabalho foram da SUN e, sendo um equipamento de maior capacidade de memória e velocidade que o PC, permitem que certos problemas enfrentados no PC, tal como o limite de memória, sejam muito amenizados. O sistema operacional usado pelas estações é o Unix.

O sistema foi desenvolvido de forma a explorar ao máximo a potencialidade disponível na estação. Assim, foi desenvolvida uma interface gráfica a fim de permitir ao usuário executar os módulos sob um sistema de janelas. Cada um dos módulos possui uma aparência e modo de funcionamento condizente com o padrão Open Look, conferindo ao usuário uma maior facilidade de operação, além de um ótimo aspecto visual. Além disso a interface gráfica também oferece caracterísiticas não disponíveis numa interface baseada em caracteres, tais com acesso a telas de auxílio (HELP) sobre cada um dos itens ativos, fechamento do programa num ícone, acesso mais fácil aos menus, etc... Para o desenvolvimento da interface gráfica foi utilizado o XView Toolkit, descrito no apêndice A. Para a utilização deste Toolkit também é essencial que a programação seja feita em linguagem C pois todas as suas rotinas e pacotes só são acessíveis através de chamadas da linguagem C.

O sistema para PC utiliza a biblioteca gráfica GKS para a visualização dos resultados e interação com o usuário. A biblioteca gráfica utilizada no sistema para estações de trabalho foi a biblioteca do Sistema X-Window chamada de XLib, também descrita no apêndice A. Esta biblioteca foi escolhida pelo fato dela ter sido adotada pela quase totalidade dos fabricantes de estações e, desta forma, estar disponível em qualquer estação. Assim todas as funções que tratam de alguma saída gráfica, tais como traçado de linhas, preenchimento de áreas, etc..., são feitas utilizando-se chamadas às rotinas da XLib.

O modelo de programação usado foi o modelo com notificador, descrito no apêndice A. Este modelo simplifica muito o trabalho de gerenciamento de en-

tradas em cada uma das telas e menus criados, uma vez que todo este trabalho é feito de forma automática pelo notificador. Dentro deste modelo, o programador só precisa registrar as rotinas que serão chamadas toda vez que um determinado evento ocorrer em uma janela. Esta estrutura é diferente do que a empregada no sistema para PC, descrito no capítulo 1. Desta forma, a estrutura dos módulos teve que ser alterada a fim de que o notificador pudesse ser corretamente usado. Para tanto foi necessário um conhecimento profundo do fluxo de operações em cada módulo.

As rotinas matemáticas utilizadas no sistema para estações de trabalho foram as mesmas que são usadas no sistema para PC. As rotinas para PC foram todas escritas em Fortran e, como se tratam de rotinas que já foram muito testadas e em uso há muito tempo, elas foram mantidas em Fortran. Conforme mostrado no apêndice A, é possível utilizar-se do C e Fortran dentro de um mesmo módulo. Assim, as rotinas foram utilizadas tal como estão no PC, observando-se as regras descritas no apêndice A para a sua interface. O seu uso, no entanto, exigiu um estudo detalhado do sua forma de funcionamento. Além disso, também foi necessário um estudo detalhado do padrão GKS, a fim de um correto entendimento do funcionamento da rotinas gráficas.

Uma vez que se pretendeu usar outro padrão gráfico, foi preciso desenvolver várias outra rotinas para realizar as funções que o GKS realizava. Por exemplo, não existe rotinas para a transformação de coordenadas na biblioteca do Sistema X-Window. Assim, foram desenvolvidas rotinas para a transformação do sistema de coordenadas da tela para o sistema do usuário e vice-versa. Para a representação da variação de grandezas na forma de um gráfico do tipo X-Y foi preciso desenvolver rotinas específicas para formatação de escalas e traçado de gráficos na tela. O sistema para PC usa para isso rotinas desenvolvidas especificamente para o GKS e em linguagem Fortran. Como estas rotinas são dificilmente adaptáveis para estações, optou-se por desenvolvê-las novamente em C. As rotinas desenvolvidas são de caracter geral e poderão ser utilizadas em outros programas.

Assim, a implantação do sistema dentro da filosofia adotada demanda uma familiarização com conceitos e padrões de programação bastante diferentes dos tradicionalmente empregados. O entendimento e utilização destes conceitos, por sua vez, demanda um trabalho de pesquisa bastante demorado até que possam ser efetivamente empregados. Pode-se afirmar que, embora para o usuário final o

trabalho se torne mais fácil, o desenvolvimento do sistema fica muito mais difícil, pois demanda uma gama de conhecimentos maior.

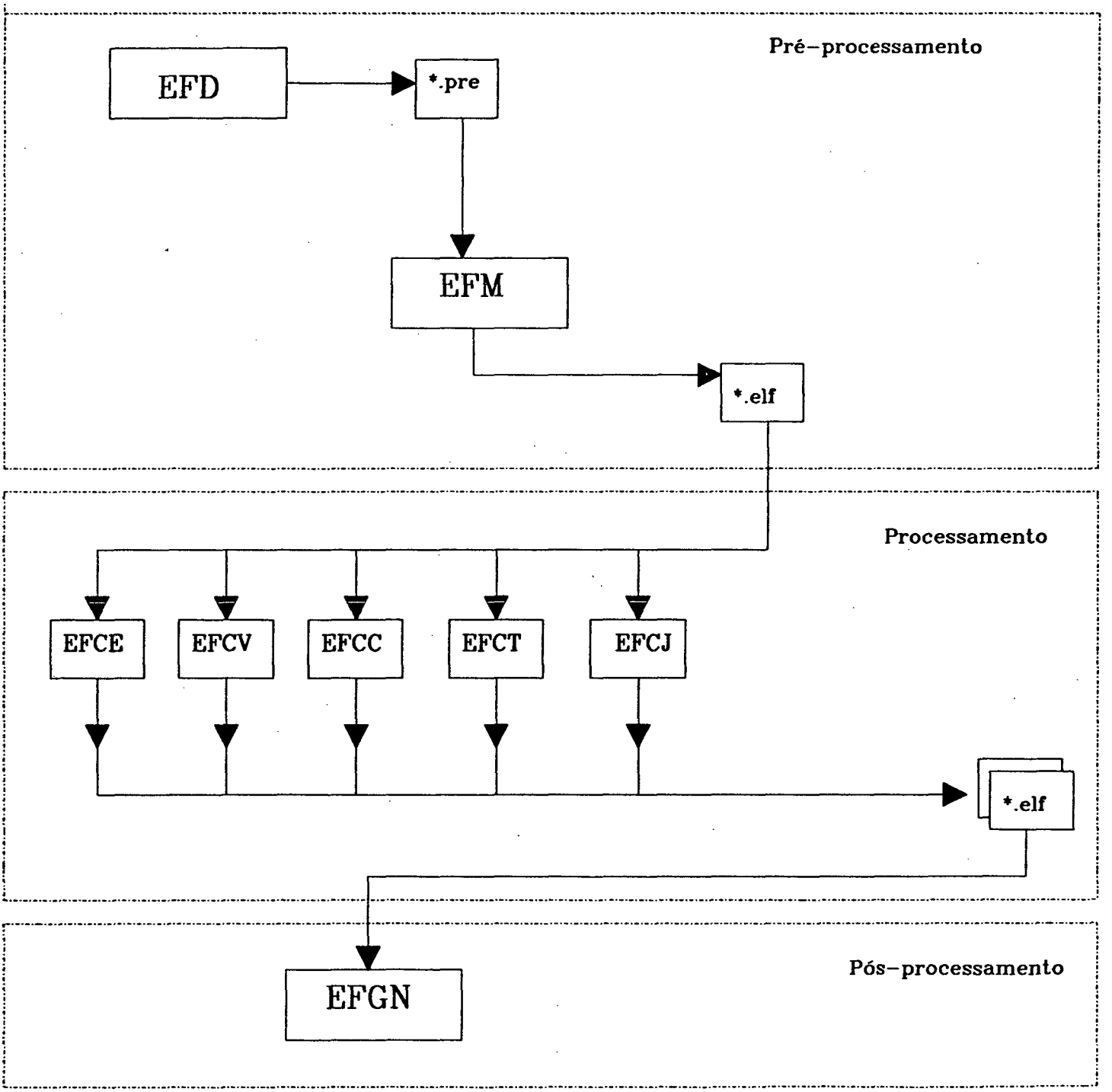

figura 2.1 - Estrutura geral do sistema EFCAD em estações de trabalho.

2.1.1 Estrutura Geral

A estrutura geral do sistema para estações de trabalho é mostrada na figura 2.1.

Pela figura 2.1 pode-se notar que o sistema possui só um módulo na etapa de pós-processamento. Graças à maior capacidade de memória da estação, foi possível a implantação do etapa de pós-processamento em um único módulo. Com isto também eliminou-se o arquivo intermediário *.res*, o qual era usado como forma de comunicação entre os módulos de exploração gráfica e numérica.

A seguir é feita uma descrição mais detalhada de cada um destes módulos.

2.1.2 Módulo de Entrada de Dados

O programa de entrada de dados, também chamado de EFD, foi todo escrito em C sendo a sua concepção totalmente original. A sua interface foi feita usando o XView Toolkit e as funções gráficas da XLib. A forma como o programa se apresenta para o usuário está mostrada na figura 2.2.

Este programa permite que se criem diversos tipos de estruturas, utilizando-se entradas tanto pelo teclado quanto pelo mouse. As curvas que o sistema EFCAD está apto a tratar são segmentos de retas e arcos de circunferências.

Nos próximos itens serão descritos as opções disponíveis neste módulo.

2.1.2.1 Manipulação de Arquivos

Ao ser acionado o botão ***arquivos...*** aparece o menu mostrado na figura 2.3. Este menu serve para a manipulação de arquivos. A primeira vez que for acionado ele mostra os arquivos com extensão *.pre* que estão no diretório corrente. Os arquivos são mostrados no lado direito do painel na forma de uma lista, a qual pode ser rolada para a frente e para trás a fim de selecionar o arquivo desejado. Pode-se criar uma nova lista de arquivos trocando-se o diretório. Pode-se também selecionar os arquivos que serão listados colocando-se uma especificação global para o nome dos arquivos no ítem ***arquivos a listar :***. Por exemplo, pode-se criar uma listagem dos arquivos que começam com *ma* colocando-se *ma** no ítem diretórios a listar. Esta forma de especificação segue as mesmas

regras para formação de nomes de arquivos usadas no sistema Unix. A listagem de arquivos é criada acionando-se o botão ***lista arquivos*** ou teclando-se ***enter*** após a especificação do nome do arquivo.

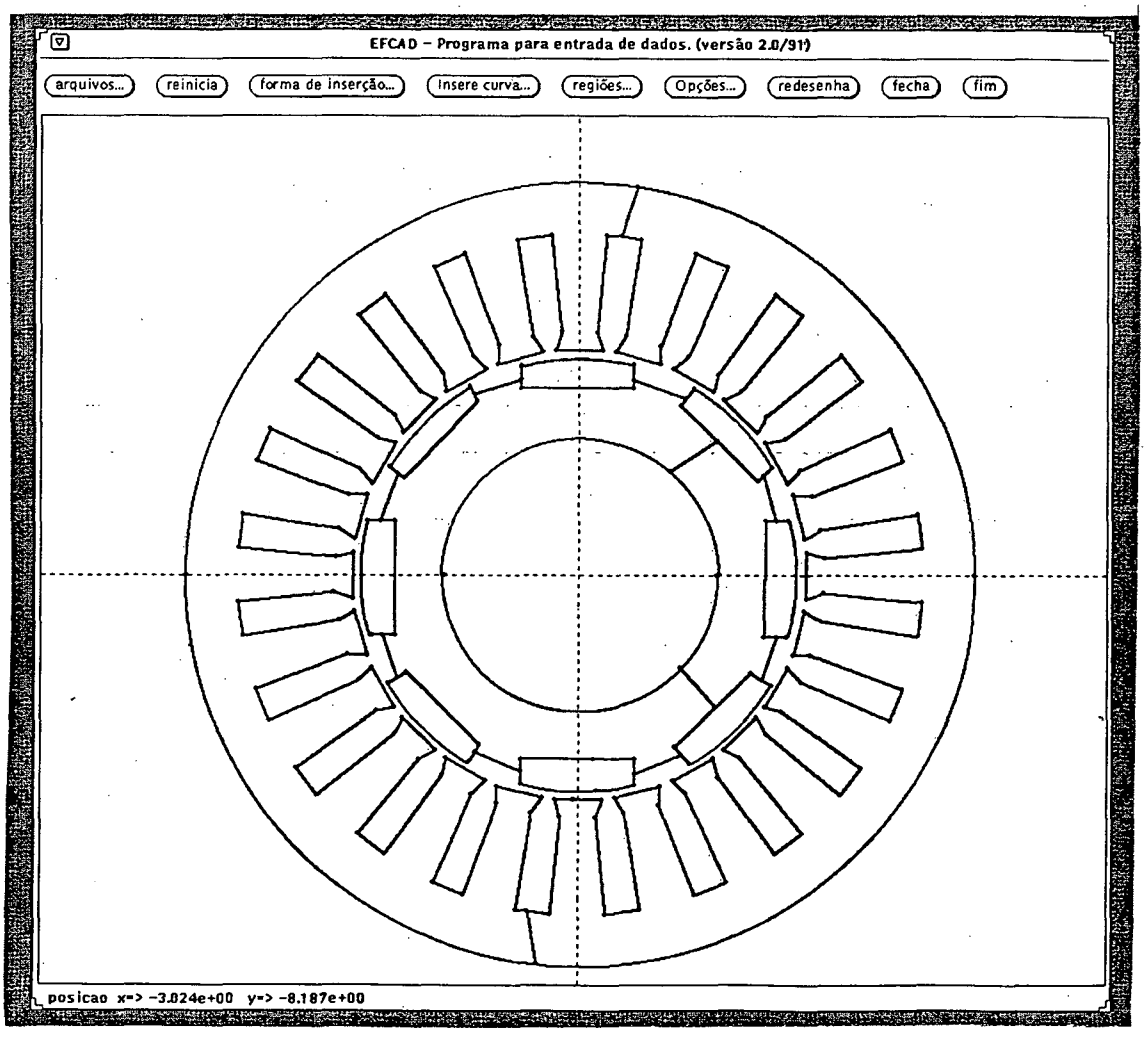

figura 2.2 - Tela principal do módulo EFD.

O arquivo escolhido é lido por meio do botão ***carrega arquivo***. Após carregado, o arquivo é desenhado na tela, podendo então ser editado.

2.1.2.2 Inserção de Curvas.

Este menu é acionado pelo botão ***insere curva*** e tem a aparência mostrada na figura 2.4. Este menu consta de uma série de ícones representando as curvas a serem inseridas e a sua ordem de inserção.

O ícone *a* representa a inserção de um segmento de reta por meio do fornecimento do seu ponto inicial (ponto 1) e o ponto final (ponto 2). Cada um destes pontos podem ser inseridos a partir do teclado ou movendo-se o mouse na tela até o ponto desejado e então teclando-se o botão esquerdo do mouse. Pode-se, igualmente, entrar com um ponto pelo mouse e o outro pelo teclado. A ordem de inserção dos pontos é irrelevante neste caso.

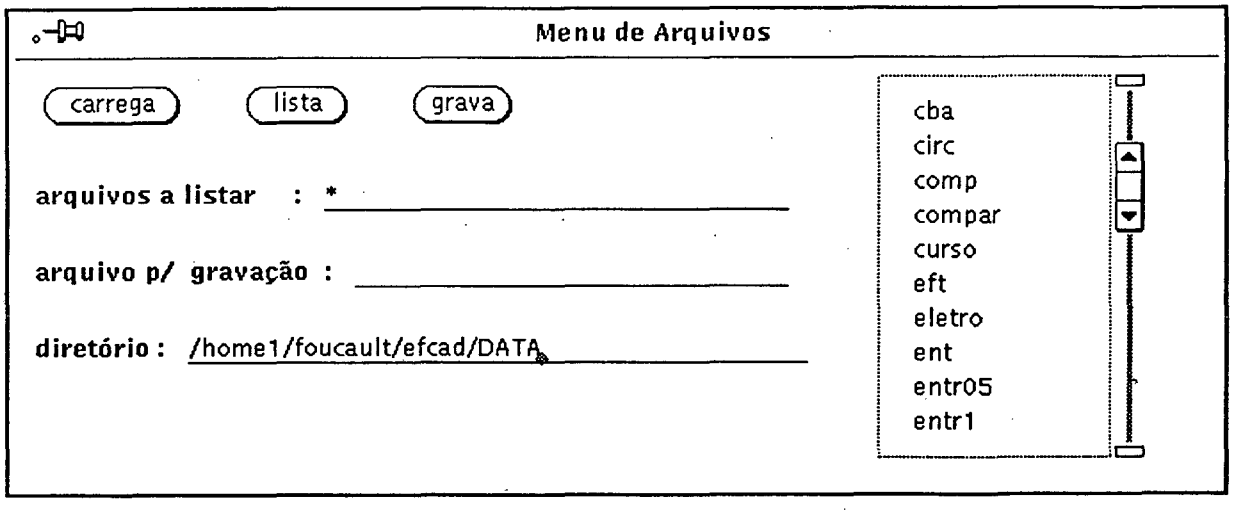

figura 2.3 - Menu de Arquivos

O ícone *b* representa a inserção de um segmento a partir do ponto inicial (ponto 1), do ângulo e módulo. O ponto inicial também pode ser inserido com mouse.

O ícone *c* representa a inserção de um arco de circunferência por meio de dois pontos e o raio. Este arco é sempre traçado do ponto 1 para o ponto 2 e no sentido anti-horáro. Com estas duas convenções o arco fica univocamente definido. Note-se que neste caso a ordem de inserção é relevante. A inversão dos pontos resultará no traçado do arco complementar. O raio também deve ser fornecido pelo teclado.

O ícone *d* representa a inserção de um arco de circunferência por meio do fornecimento de dois pontos e do centro. O arco será sempre traçado do ponto 1 para o ponto 2 no sentido anti-horário. Com esta convenção o arco também fica univocamente determinado. Também pode se inserir pontos tanto pelo teclado como pelo mouse.

O ícone *e* representa a inserção de arco de circunferência por meio do fornecimento de três pontos. Note-se que a seqüência dos pontos também é importante, uma vez que o arco é traçado do ponto 1 até o 3 passando pelo ponto 2.

O ícone *f* representa a inserção de uma circunferência a partir do fornecimento do seu centro e do seu raio. O centro pode ser definido com o mouse ao passo que o raio deve ser fornecido pelo teclado.

O ícone *g* representa a inserção de um arco de circunferência a partir do centro, raio, ângulo inicial e ângulo final. Os ângulos devem ser fornecidos em graus e o arco é traçado do ângulo inicial para o ângulo final no sentido anti-horário. O centro pode também ser inserido com o mouse.

O ícon e *h* representa a inserção de um arco de circunferência a partir do centro, raio, um ponto sobre o arco e o ângulo a partir deste ponto. Também neste caso o ângulo deve ser em graus e o centro pode ser inserido com o mouse. O arco será sempre traçado a partir do ponto no sentido anti-horário.

Em todos os casos presume-se que o arcos serão positivos. No caso de se definir valores de ângulo negativos, os arcos de círculo serão traçados no sentido horário.

Toda vez que for selecionado uma nova curva, os ítens abaixo dos ícones mudarão de acordo com os dados necessários para cada curva. Por exemplo, quando for selecionado a inserção de um arco de circunferência por meio de 3 pontos os itens mostrados serão *x1*, *y1*, *x2*, *y2*, *x3* e *y3*, quando for selecionado a inserção de uma circunferência a partir do centro e do raio são mostrados os itens *xc*, *yc* e *raio*.

Toda vez que um ponto for inserido o programa faz automaticamente uma verificação para determinar se o ponto já existe ou se está muito próximo de um existente. Se for determinado que o ponto inserido já existe não será criado um novo ponto, mas utilizado o existente. Além disso, é sempre feita uma segunda

verificação para determinar se o ponto inserido está sobre um segmento ou arco de circunferência. Quando for determinado que o ponto está sobre um segmento (ou arco) o mesmo é dividido em dois novos segmentos. A verificação para os arcos de circunferência é feita com base numa discretização dos arcos e cálculo da distância a cada um dos segmentos discretizados. Esta verificação é feita para todos os pontos inseridos e visa evitar problemas para o módulo malhador, além de criar um arquivo mais simples de trabalhar.

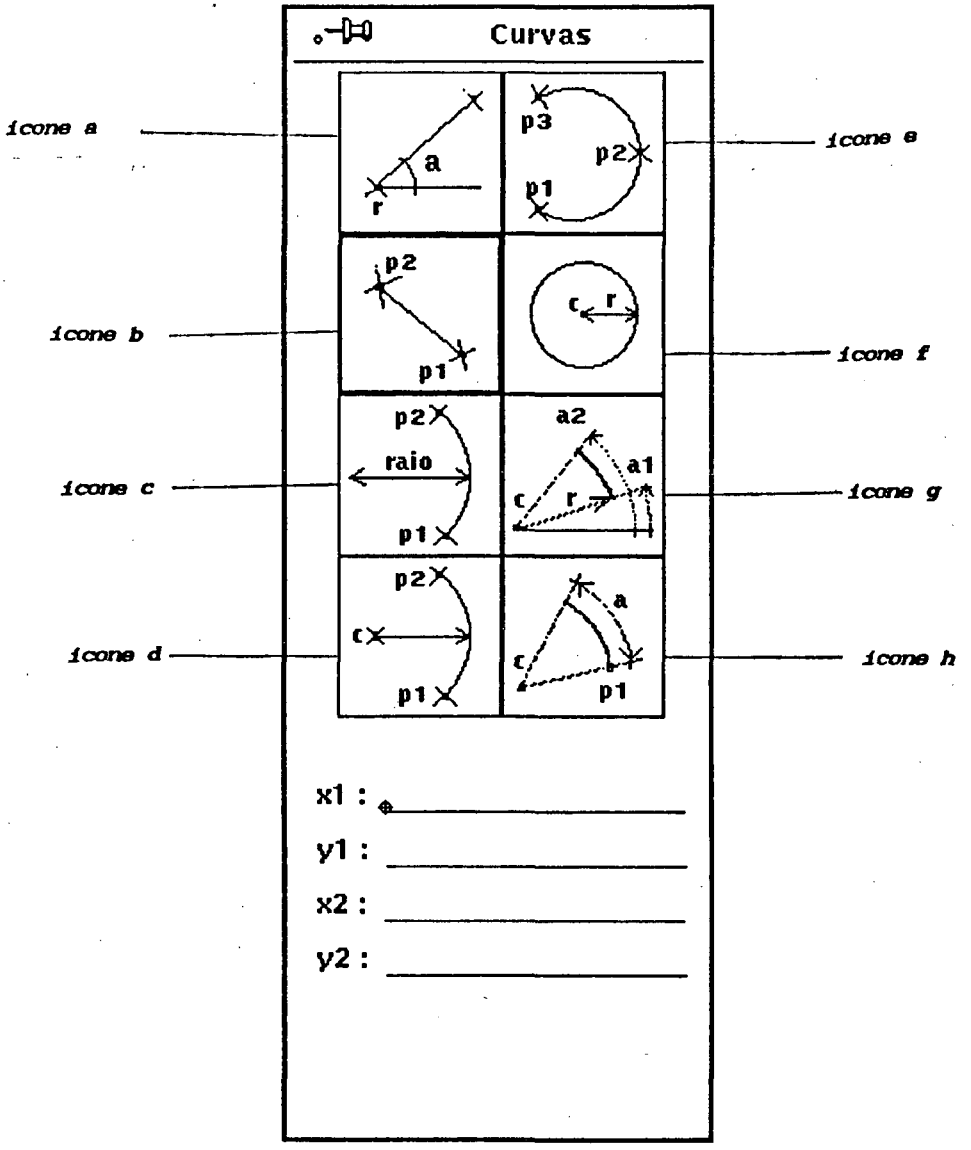

figura 2.4 - Menu de inserção de curvas.

2.1.2.3 Forma de Inserção de Curvas

Quando o botão ***forma de inserção...*** for acionado aparece menu que define como os pontos inseridos com o mouse, ou teclado, serão conectados. Este menu é mostrado na figura 2.5.

Um ponto pode ser iserido pelo mouse de 3 maneiras diferentes:
- pode ser totalmente livre, correspondendo a opção ***não conectar*** ;
- pode ser conectado a um ponto já definido, correspondendo a opção ***conectar ponto a ponto*** ;
- pode ser inserido sobre um segmento ou arco de circunferência, correspondendo a opção ***conectar a segmento/arco*** .

Quando for inserido um segmento, ou arco, com o mouse cada um dos pontos que definem o segmento podem ser inseridos de uma das 3 formas acima. Por exemplo, para inserir um arco de circunferência por três pontos pode-se inserir o primeiro ponto em um segmento existente, o segundo ponto pode-se deixar livre e o último ponto pode ser um ponto já existente. Estas opções são oferecidas de forma a dar maior flexibilidade ao usuário.

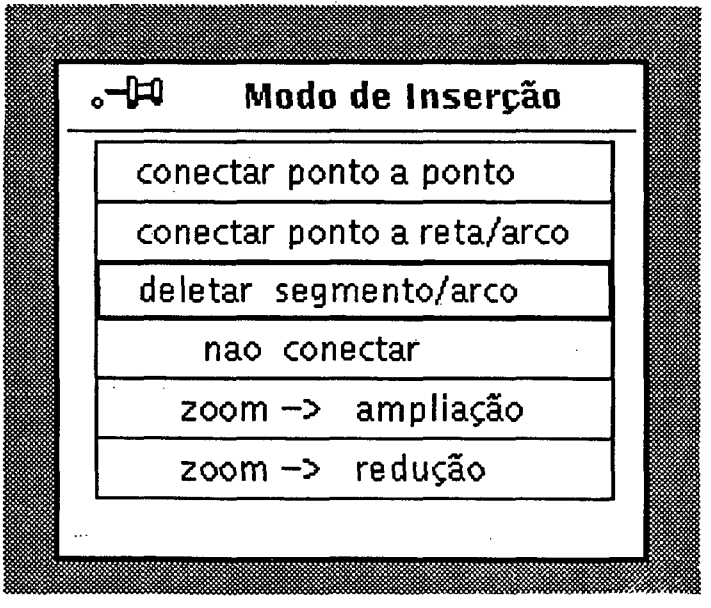

figura 2.5 - Menu de forma de inserção.

Quando se deseja conectar um ponto a um já existente, o programa automaticamente procura qual o segmento está mais próximo e qual o ponto deste segmento que está mais próximo do ponto detectado com o mouse. No caso de arcos de circunferências é feito primeiro uma discretização do arco em segmentos.

Quando se trata de conectar um ponto a um ponto já existente, o programa automaticamente procura qual ponto mais próximo do detectado com o mouse.

Este menu oferece a possibilidade de eliminação de segmentos por meio da opção ***deletar segmento/arco*** . Isto é feito indicando-se com o mouse qual o segmento que deverá ser eliminado. O programa também procura automaticamente qual o segmento mais próximo do ponto que o usuário indicou com o mouse.

Ainda neste menu pode-se fazer uma ampliação ou uma redução do desenho da estrutura por meio das opções ***zoom+*** -> ampliação e ***zoom-*** -> ***redução*** . Isto é feito a partir da indicação com o mouse do ponto em torno do qual a ampliação será feita. A estrutura é desenhada novamente com o centro no ponto indicado e ampliada (ou reduzida) por um certo fator. O fator de redução e ampliação é fixado no menu ***opções***. O zoom pode ser útil quando se deseja trabalhar com apenas parte da estrutura e para definir detalhes da mesma.

2.1.2.4 Manipulação de Regiões Geométricas

O menu que permite trabalhar com partes da estrutura como um todo é acionado por meio do botão ***regiões...*** e possui a aparência mostrada na figura 2.7.

Neste menu pode-se selecionar uma região e em seguida deslocá-la, reproduzi-la ou eliminá-la. Pode-se definir uma região da seguinte forma;

- indicação cartesiana;
- indicação polar.

No caso da indicação cartesiana a região será um retângulo o qual é indicado por dois pontos que determinam o canto superior e o inferior. Os valores ***x_min***, ***x_max***, ***y_min*** e ***y_max*** serão determinados a partir destes dois pontos indicados com o teclado ou com o mouse.

No caso da indicação polar deverá ser fornecido o raio mínimo, raio máximo, ângulo mínimo e ângulo máximo. Quando for selecionada esta opção e forem fornecidos pontos com o mouse, os valores de ângulo e raio serão calculados a partir dos pontos fornecidos. Os ângulos e raio se referem sempre à origem do sistema de coordenadas. A figura 2.7 mostra um exemplo de indicação polar de uma região. A região selecionada será mostrada em pontilhado na tela. Note-se que a ordem de fornecimento dos pontos é relevante, uma vez que, por convenção, a

região é selecionada do ponto 1 ao ponto 2. A inversão da indicação dos pontos implicará a seleção da região complementar.

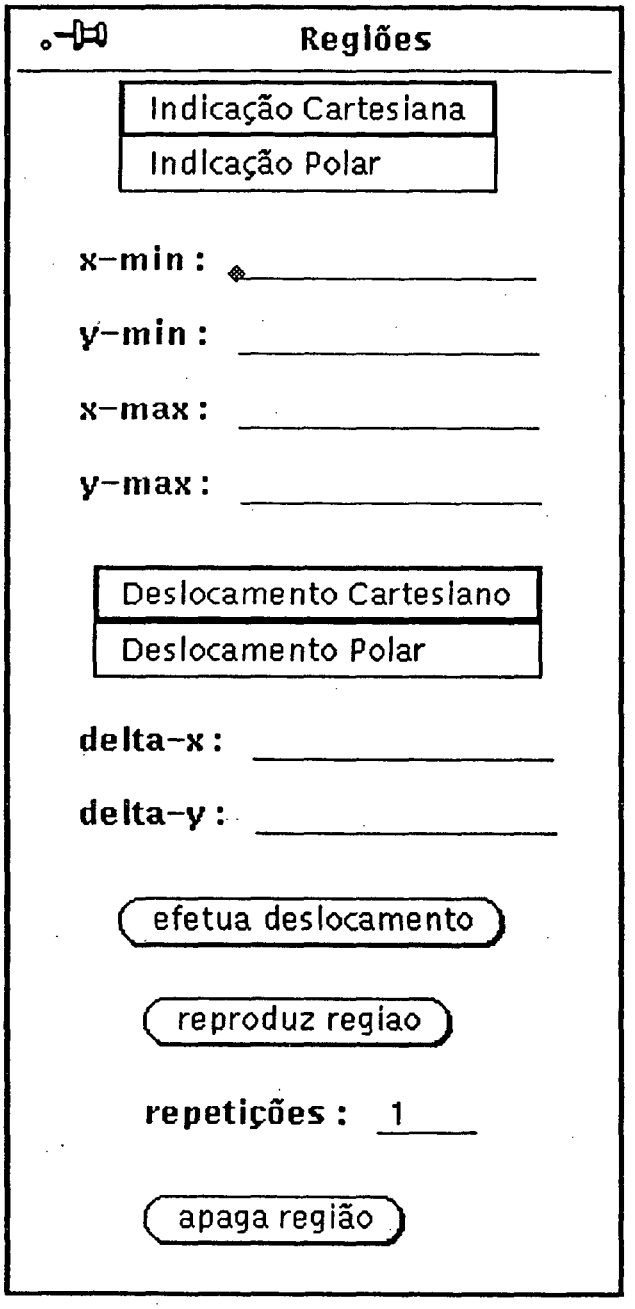

figura 2.6 - Menu de manipulação de regiões.

A indicação do deslocamento também pode ser feita de duas maneiras:

- deslocamento cartesiano;
- deslocamento polar.

No caso de um deslocamento cartesiano deverá ser indicado valores para o deslocamento na direção X e na direção Y (***delta-x*** e ***delta-y*** , respectivamente).

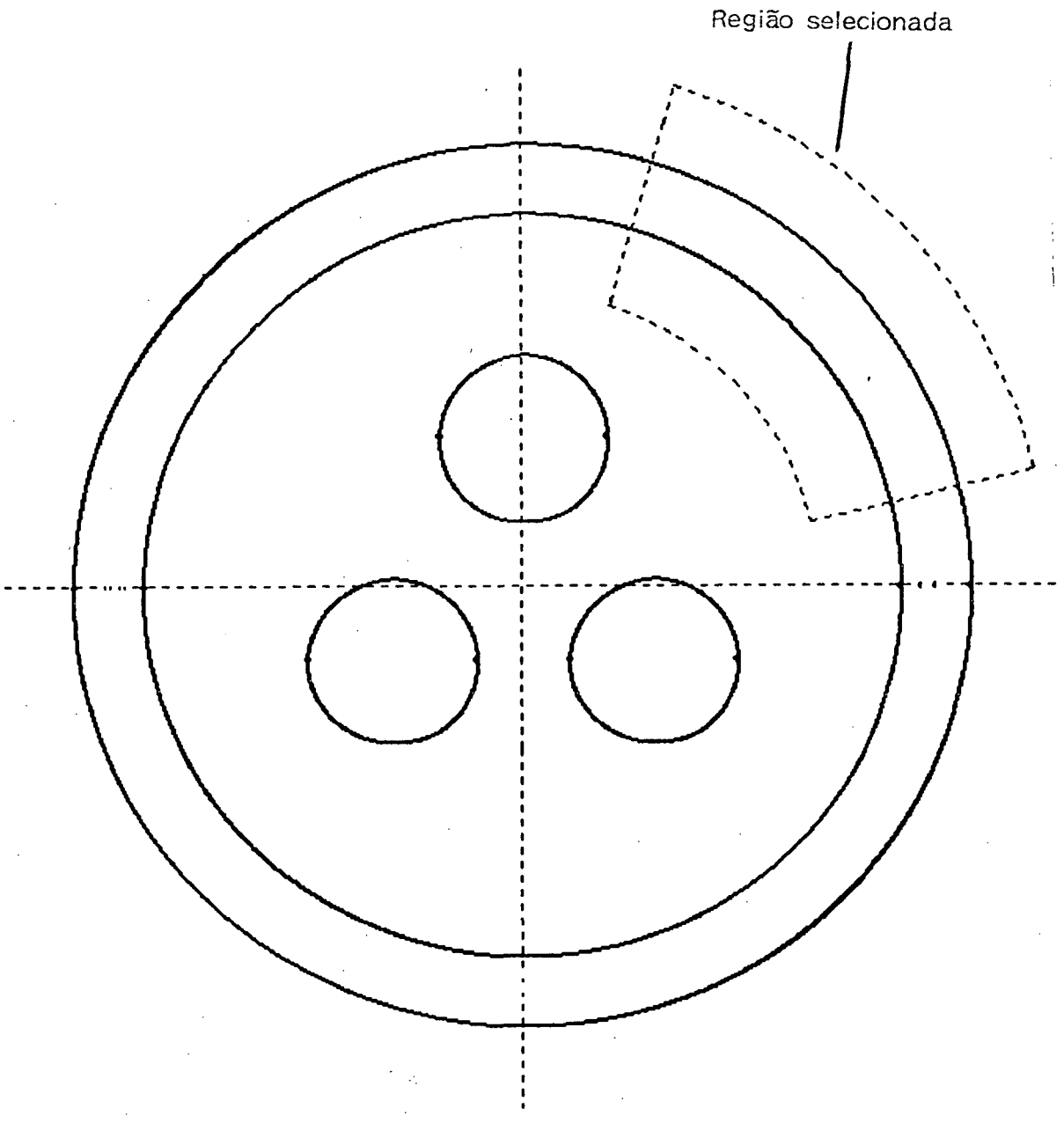

figura 2.7 - Exemplo de indicação de região.

Para um deslocamento polar deverá ser fornecido um valor de variação do raio e um valor para a variação do ângulo, (***delta-R*** e ***delta-T*** , respectivamente). Neste caso será feita uma conversão dos pontos dentro da região para coordenadas polares e, em seguida, acrescentado o deslocamento. Pode-se usar esta opção para conseguir girar partes da estrutura, definindo-se ***delta-R*** (variação do raio) como sendo zero e ***delta-T*** (variação do ângulo) como o ângulo que a re-

gião deve ser girada. Quando se trabalha com máquinas elétricas pode-se usar esta opção para alterar o entreferro da máquina. Isto é conseguido selecionando-se o rotor e em seguida definindo-se ***delta-R*** como sendo a variação do entreferro. Outras medidas da máquina tais como raio da coroa e altura das ranhuras podem ser alterados facilmente de forma semelhante.

Ao ser pressionado o botão ***efetua deslocamento*** o deslocamento é efetivamente realizado. Todos os pontos que estão dentro da região serão deslocados de acordo com o valor de deslocamento especificado. Esta opção é extremamente útil quando se deseja analisar uma estrutura móvel em várias posições, como por exemplo máquinas elétricas ou contatores.

O botão ***reproduz região*** reproduz a região selecionada. O número de vezes que a região será reproduzida é especificado no ítem repetições. O espaçamento entre uma repetição e outra é especificado nos ítens ***delta-x*** e ***delta-y*** (***delta-R*** e ***delta-T***). Esta opção também é muito útil na definição de estruturas cujas partes se repetem, como no caso das ranhuras de uma máquina elétrica. Pode-se definir uma das ranhuras e reproduzi-la ao longo da periferia do rotor ou do estator numa única operação.

Pode-se também apagar uma região selecionada da estrutura por meio do botão ***apaga região***. Esta opção também serve quando se,quer definir estruturas com partes comuns. Por exemplo, pode-se gravar num arquivo o estator de uma máquina e definir-se vários rotores para a mesma gravando-os em arquivos separados.

2.1.2.5 Opções

Além dos menus até aqui citados há um menu ativado por meio do botão ***opções*** que possui os seguintes itens:

- ***x-min*** , é o valor mínimo da coordenada X da estrutura que será mostrado na tela;
- ***x-max*** , é o valor máximo da coordenada X que será mostrada na tela;
- ***y-min*** , é valor mínimo da coordenada Y que será mostrado na tela;
- ***y-max*** , é o valor máximo da coordenada Y que será mostrado na tela;
- ***fat. zoom*** , define o fator que será usado na ampliação e redução do desenho;

- *fat. esc.* , define o fator de escala que dividirá os valores das coordenadas. É o primeiro valor que será gravado no arquivo com extensão *.pre*.

Quando for teclado *enter* após o valor de *x-max* , o valor de *y-min* e *y-max* serão ajustados de acordo com o valor de *x-min* e *x-max*. Quando for teclado *enter* após *y-max*, os valores de *x-min* e *x-max* serão ajustados. Isto deve ser feito a fim de que o desenho da estrutura não seja distorcida pois ambos os valores das escalas X e Y devem guardar uma proporção fixa entre si.

No painel principal existe ainda os seguintes botões:

- *redesenha* ;

- *fecha* ;

- *fim* .

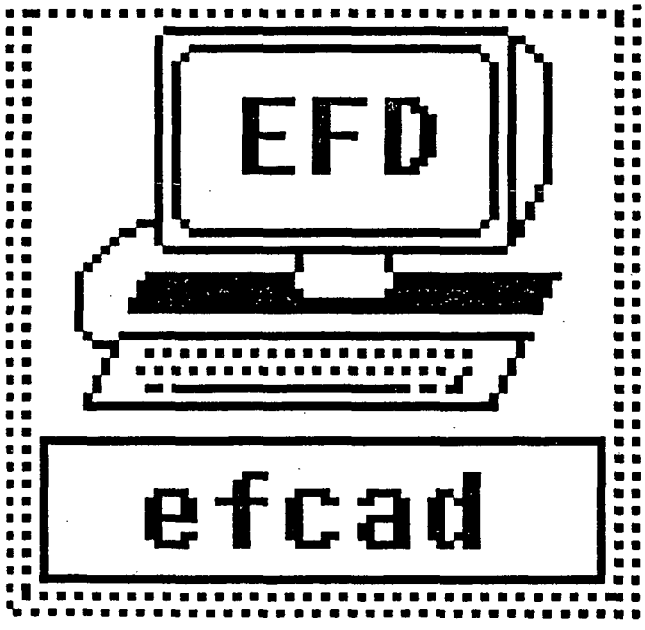

figura 2.8 - ícone do program EFD.

O botão *redesenha* , além de redesenhar a estrutura, faz com que as escalas na direção X e Y sejam redefinidas a fim de que toda a estrutura seja apresentada na tela com a melhor visualização possível. Esta opção é usada para que se possa voltar ao normal um desenho que tenha sido reduzido ou ampliado.

O botão ***fecha*** faz com que o programa seja fechado num ícone, conforme mostrado na figura 2.8. O programa pode, contudo, ser novamente aberto no ponto em que parou.

O botão ***fim*** encerra a execução do programa. Antes, porém, será enviada uma mensagem ao usuário pedindo pela confirmação.

O usuário pode obter informações sobre cada um dos botões e itens descritos no momento da execução do programa. Para isso basta posicionar o mouse sobre o botão ou ítem do qual se deseja informações e pressionar a tecla ***F1*** no teclado. Este procedimento aciona o mecanismo de auxílio que mostra uma tela contendo informações semelhante a mostrada na figura 2.9.

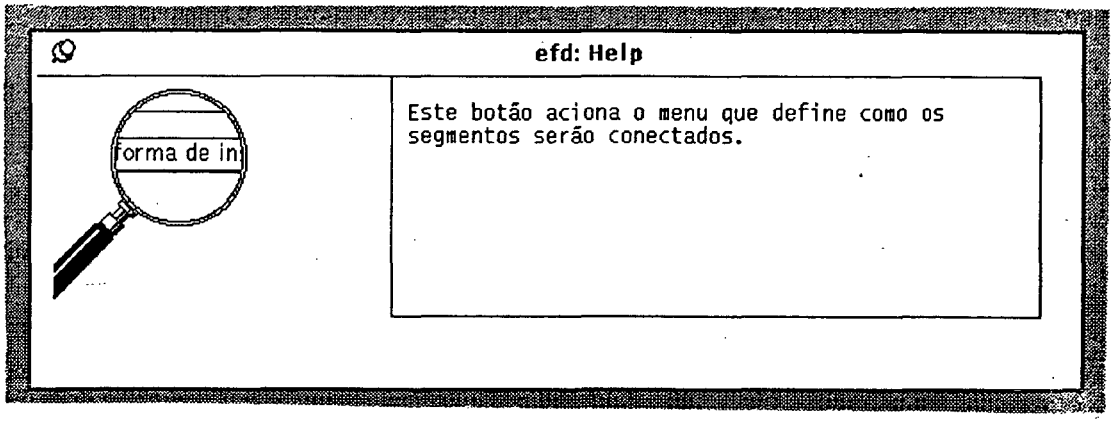

figura - 2.9 - Tela de auxílio contendo informações.

Desta forma o usuário não precisa procurar no manual para resolver dúvidas a respeito do funcionamento do programa, agilizando o processo de aprendizado e utilização.

2.1.3 Módulo de Geração de Malha

Após ter sido criado o arquivo que contém informações geométricas sobre a estrutura a ser estudada (arquivos com extensão ***.pre***) passa-se a etapa de geração de malha. O módulo que realiza esta tarefa é chamado de EFM. Após ter sido carregado, este módulo exibe a tela mostrada na figura 2.10. Neste módulo também podem ser feitas alterações na geometria da estrutura e deslocamento de regiões.

O módulo EFM pode trabalhar com uma malha que já tenha sido criada anteriormente (arquivos com extensão *.ELF*). Pode-se, igualmente, fazer alterações em uma malha existente sem ter que criá-la novamente.

A seguir é feito uma descrição do funcionamento do programa e dos seus menus.

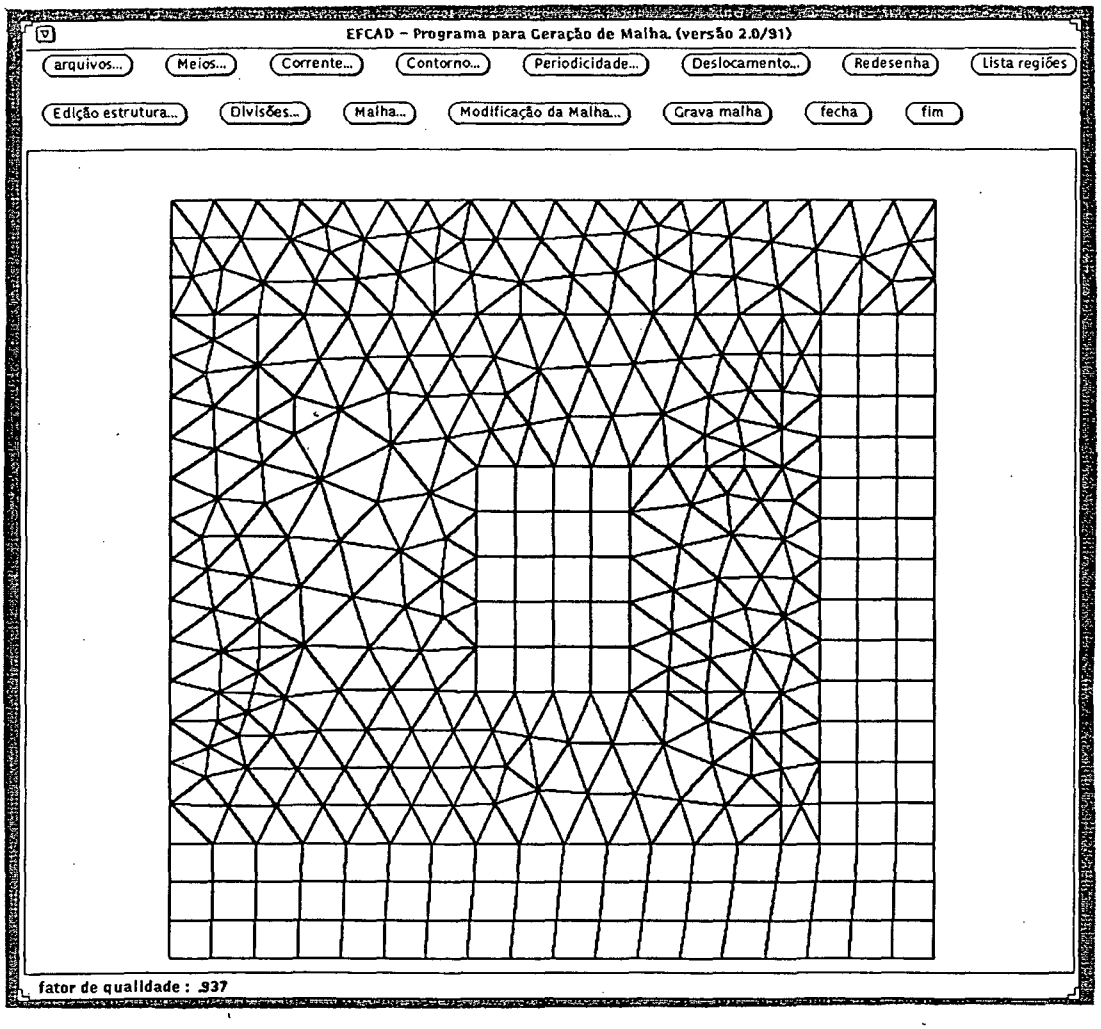

figura 2.10 - Tela principal do programa EFM

2.1.3.1 Manipulação de Arquivos

O menu de arquivos é quase idêntico ao programa EFD tendo no entanto algumas opções a mais. Ele permite listar os arquivos com extensão *.pre* e os arquivos com extensão *.ELF*. Também pode-se especificar um arquivo onde serão armazenadas as alterações feitas pelo programa EFM no arquivo lido. Toda vez que um arquivo é lido são feitas verificações quanto a segmentos duplicados, pontos muito próximos, etc... Quando necessário são feitas alterações no arquivo a fim de evitar problemas na geração da malha. Quando o usuário abandona o programa é feita uma pergunta para determinar se as modificações devem ser gravadas no mesmo arquivo que foi lido ou se deverá ser usado o arquivo especificado no ítem ***arquivo p/ alterações*** .

O menu de arquivos é mostrado na figura 2.11.

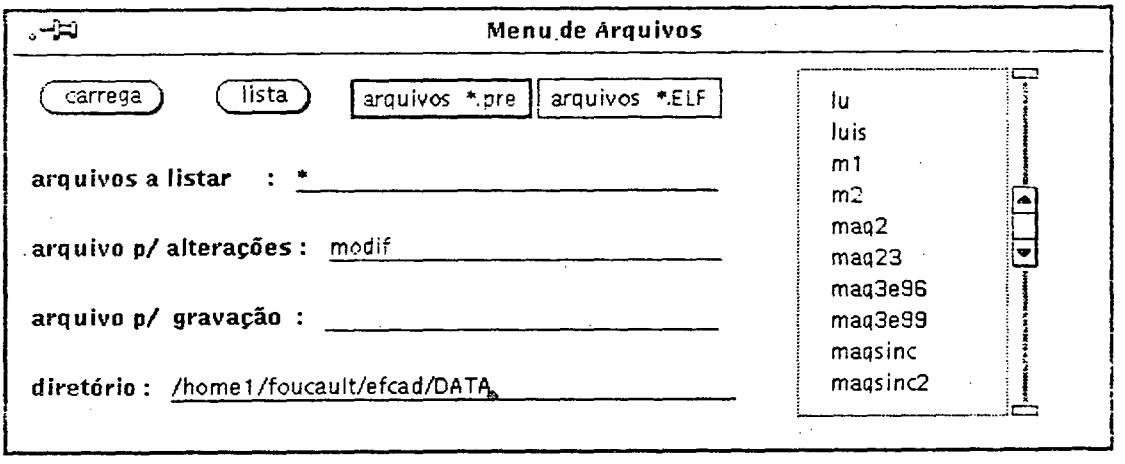

figura 2.11 - Menu de arquivos do programa EFM

2.1.3.2 Inserção de Meios

Quando for pressionado o menu ***meios...*** aparecerá o menu de inserção de meios. Este menu consiste de um único ítem contendo o número do meio a inserir. Todos os meios são inicialmente do tipo 1 (ar). Assim só é preciso entrar com os meios que são diferentes de 1. Os meios são manipulados por meio do programa gerenciador de materiais chamado EFP, descrito a seguir. O número do meio corresponde ao número do registro no arquivo de materiais ***efmat.dat*** .

Quando este menu for acionado, serão circundadas de azul as regiões que possuirem um meio diferente de 1. Isto serve como um auxílio visual a fim de identificar as regiões cujos meios já foram definidos.

Para inserir o meio o usuário define o seu número e em seguida indica com o mouse na tela a região que conterá aquele meio.

2.1.3.3 Inserção de Correntes

Este menu é acionado pelo botão ***correntes...*** e consiste de dois ítens indicando o módulo e a fase da densidade de corrente a ser inserida numa determinada região. O usuário especifica o módulo e a fase e indica na tela qual a região que possui aquele valor de densidade. Regiões cuja densidade é zero não precisarão ser inseridas.

Quando este menu for ativado as regiões onde já foram inseridas corrente serão circundadas de vermelho.

No caso de ser cometido um erro na inserção do valor, basta redefinir o valor e indicar novamente com o mouse a região.

2.1.3.4 Inserção de Condições de Contorno

Este menu, mostrado na figura 2.12, permite que as condições de contorno impostas à estrutura sejam inseridas com o auxílio do mouse.

Há duas condições de contorno que podem ser impostas à estrutura:
- condição de Dirichlet, onde um potencial é imposto;
- condição de Neumann, onde nenhuma condição é especificada. Isto significa que no caso do potencial vetor o campo é perpendicular à fronteira. No caso de potencial escalar o campo é paralelo à fronteira.

De acordo com a figura 2.12, há várias formas de inserir as condições de contorno:

- ***Dirichlet total*** , a condição de Dirichlet com potencial zero é imposta a todos os segmentos que formam a fronteira externa da estrutura;
- ***Neumann total*** , nenhuma condição é imposta à estrutura;

- ***Dirichlet parcial - indicação de pontos no contorno*** , com esta opção o usuário indica sobre a fronteira externa o ponto inicial e o ponto final. O programa estabelece a condição de Dirichlet em todos os segmentos que unem os dois pontos no sentido anti-horário;
- ***Dirichlet parcial - indicação de segmentos*** , com esta opção pode-se indicar qualquer segmento onde será imposta a condição de Dirichlet.

Em todas as opções descritas o valor do potencial imposto é especificado no ítem potencial.

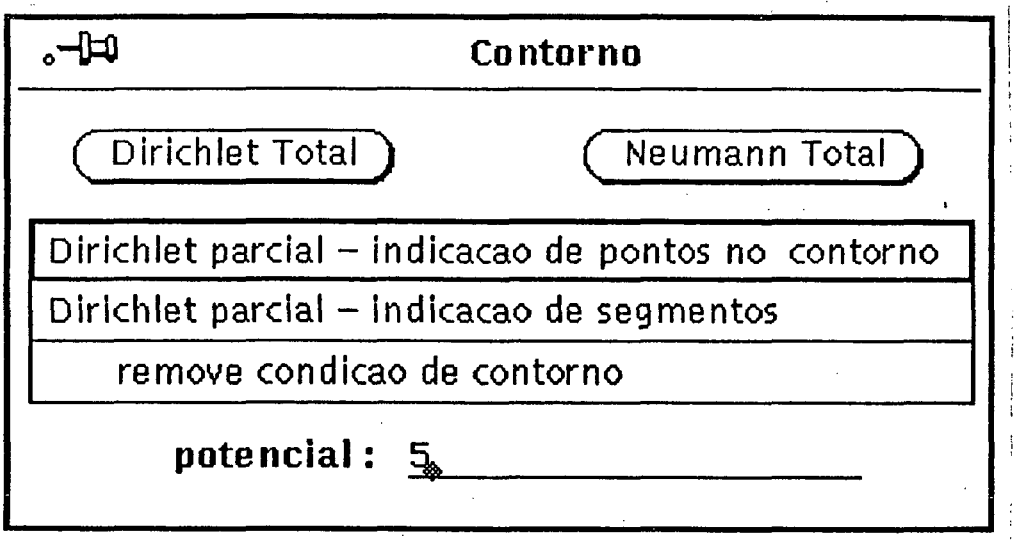

figura 2.12 - Menu de inserção de condições de contorno.

Também quando este menu for ativado os segmentos que já possuirem uma condição de contorno a eles associadas serão traçados em verde.

2.1.3.5 Inserção de Periodicidade

Neste menu são especificadas as condições de periodicidade da estrutura. A periodicidade é definida como a repetição geométrica de um domínio sem que as fontes invertam os sentidos. A anti-periodicidade é quando a geometria se repete mas o sentido das fontes se invertem. Pode haver periodicidade em X ou em Y. Existe também a periodicidade para o caso específico de máquinas elétricas. A periodicidade é indicada com o mouse sobre as regiões da estrutura dotadas de periodicidade. A indicação é feita indicando-se as linhas que delimitam a periodicidade [1,35]. A figura 2.13 mostra o menu de inserção de periodicidade.

As figuras 2.14 e 2.15 mostram exemplos de condição de periodicidade e anti-periodicidade, respectivamente.

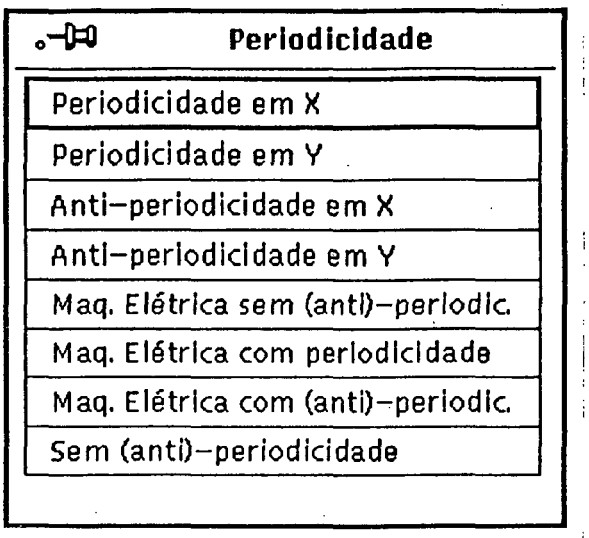

figura 2.13 - Menu de inserção de condições de periodicidade.

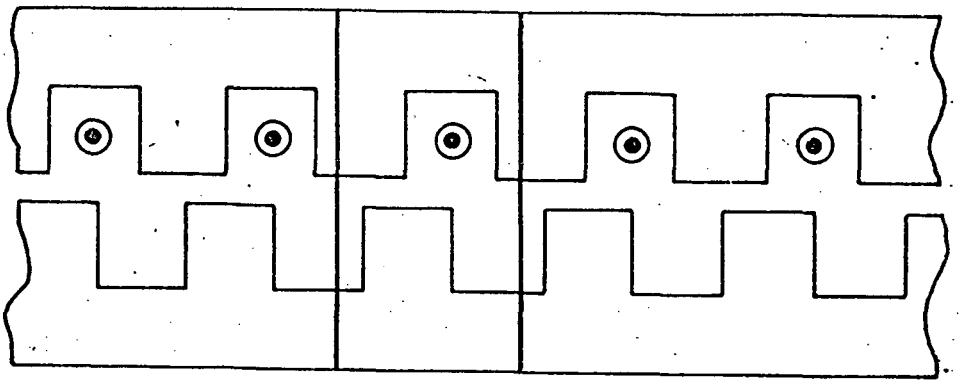

figura 2.14 - Condiçao de periodicidade

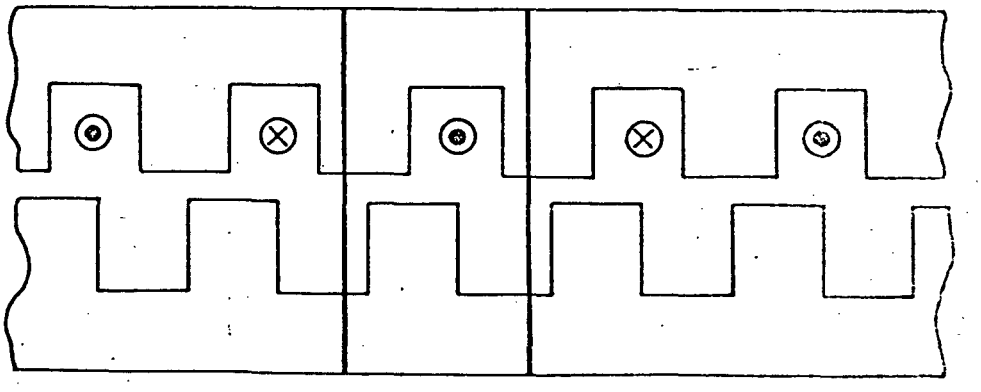

figura 2.15 - Condiçao de anti-periodicidade

2.1.3.6 Modificação da Estrutura

Dentro do programa para a geração de malha existe a possibilidade de se fazer alterações geométricas na estrutura. O menu que permite realizar estas alterações é acionado pelo botão ***edição da estrutura...***, localizado no painel principal. A figura 3.16 mostra este menu. Cada uma das opções deve ser usada com o mouse, ou seja, seleciona-se a opção e em seguida indica-se a posição com o mouse.

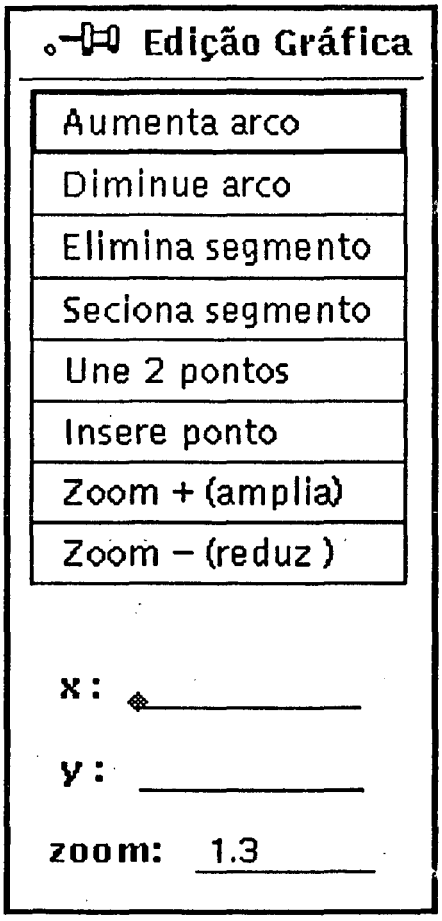

figura 2.16 - Menu de edição da estrutura.

As opções disponíveis são as seguintes:

- ***Aumenta arco*** , permite que as divisões que são feitas nos segmentos sejam aumentadas. Estas divisões são feitas pelo programa e resultam da transformação dos arcos de circunferências do arquivo original em segmentos de reta;
- ***Diminui arco*** , diminui as divisões dos segmentos;
- ***Elimina segmento*** , elimina segmentos da estrutura;

- ***Seciona segmento*** , divide um segmento em dois;
- ***Une 2 pontos*** , cria um novo segmento a partir da ligação de 2 pontos;
- ***Insere ponto*** , insere um novo ponto na estrutura;
- ***Zoom+(amplia)*** , amplia o desenho da estrutura, tal como no programa de entrada de dados;
- ***Zoom-(reduz)*** , reduz o desenho da estrutura.

Este menu foi colocado no programa para geração de malha com o objetivo de evitar que o usuário tenha que chamar o programa de entrada de dados para realizar alterações na estrutura. Todavia, como o programa de entrada de dados foi integrado ao sistema de janelas, é muito simples chamá-lo, já que é possível ter ambos os programas rodando em ***background*** , ou até mesmo simultaneamente na tela. Assim para passar de um programa para o outro basta que se feche um e abra o outro, ou que se mude a ordem das janelas na tela.

2.1.3.7 Geração da Malha

Foram desenvolvidos três menus onde se pode trabalhar com a malha. O botão ***divisões*** aciona o menu que trata da divisão de cada segmento em outros menores. Estes segmentos menores serão usados como base para a geração da malha. Este menu é mostrado na figura 2.17 e as primeiras cinco opções se referem ao tamanho das divisões que serão geradas. Um número alto de divisões conduzirá a uma malha de mais alta ordem do que um número baixo de divisões. Teclando-se sobre o botão ***gera divisões*** as divisões serão geradas e traçadas na tela. O botão ***altera divisões*** acionará um outro submenu que permite alterar o número das divisões. Este submenu também é mostrado na figura 2.17. O botão ***gera malha*** faz com que a malha seja efetivamente gerada e mostrada na tela.

O segundo menu associado com a geração da malha é acionado por meio do botão ***malha...*** . Ele serve para que a malha seja gerada e alterada. A figura 2.18 mostra as opções disponíveis.

Os primeiros ítens do menu estabelecem a densidade de malha que pode ser gerada. A densidade depende da estrutura que está sendo analisada e do espaço disponível de memória. Com o sistema implantado na estação pode-se obter malhas muito mais refinadas do que o que se conseguiria com o PC, já que se dispõe de limites de memória maiores.

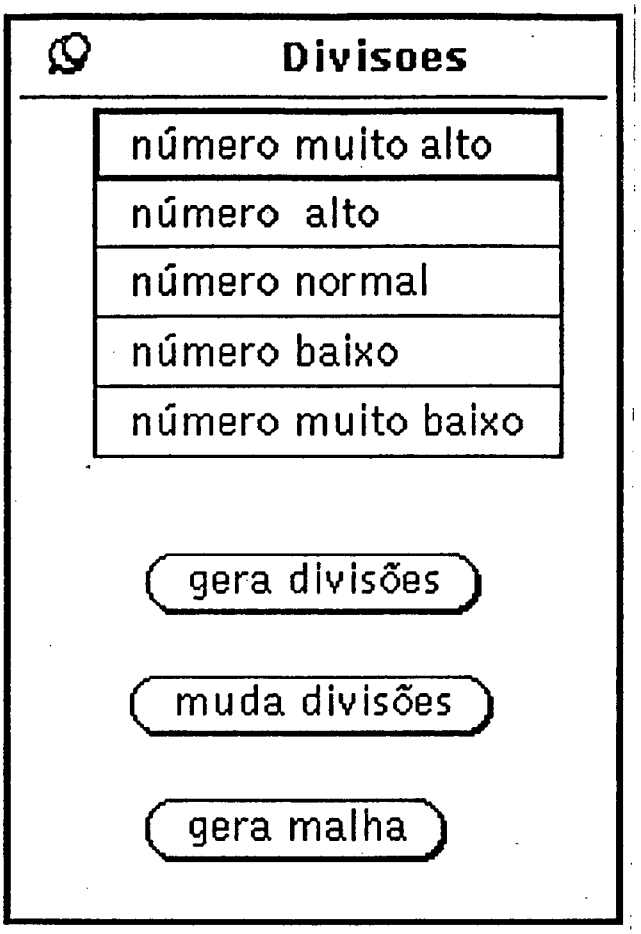

figura 2.17 - Menu para geração de divisões.

O botão ***gera malha*** gera a malha com a densidade especificada. A malha que está carregada na memória pode ser redesenhada por meio do botão ***desenha malha***. Este botão também é útil quando se deseja trabalhar com um arquivo de malha (extensão ***.ELF***) ao invés de um arquivo de desenho (extensão ***.pre***).

O terceiro menu associado com a malha é acionado por meio do botão ***modificação da malha*** localizado no painel principal. Ele está mostrado na figura 2.19 e possui as seguintes opções:

- ***Insere nó*** , usada para inserir um novo nó na malha;
- ***Suprime nó*** , usado para eliminar um nó da malha;
- ***Zoom+(amplia)*** e ***Zoom-(reduz)*** , funciona da mesma forma que nos outros menus.

Com as opções acima pode-se fazer alterações na malha de forma que ela se torne mais densa em certas regiões.

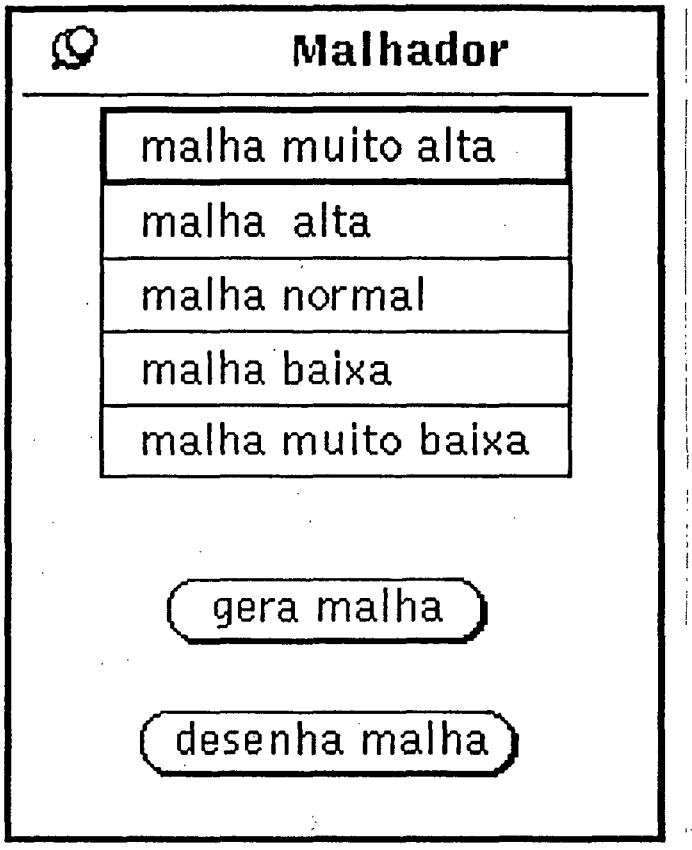

figura 2.18 - Menu de geração da malha.

Após serem feitas as modificações na malha deve se fazer uma regularização da malha por meio do botão ***regulariza***.

Existe ainda no painel principal o botão ***grava malha*** que efetua a gravação da malha no arquivo especificado pelo ítem ***arquivo p/ gravação*** no menu de arquivos.

2.1.3.8 Menus Adicionais

O painel principal contém alguns menus e botões além dos descritos acima que possuem finalidades diversas e que serão descritos aqui.

O botão ***redesenha*** desenha novamente a estrutura de forma que todo o traçado esteja contido na tela. Isto faz com que o desenho retorne ao tamanho original após ter sido ampliado ou reduzido ou após ter sido feito o traçado da malha.

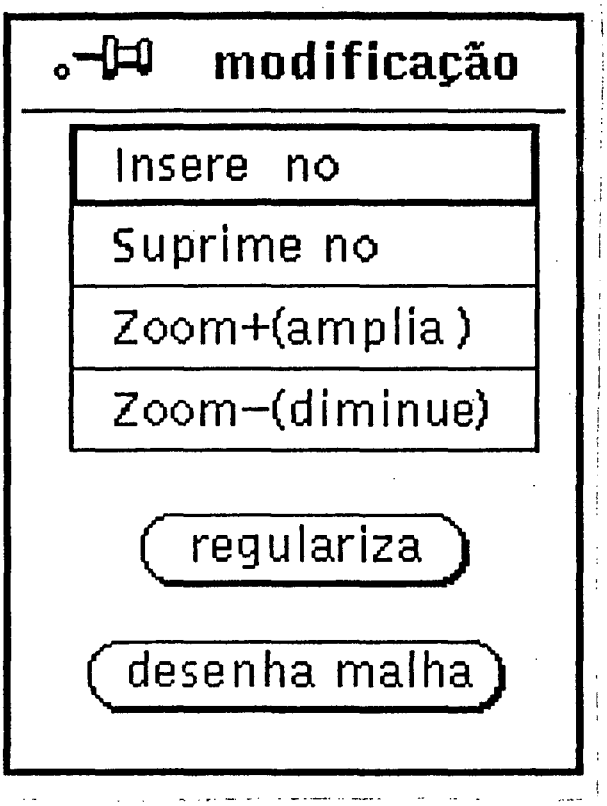

figura 2.19 - Menu de modificação da malha.

O botão ***Deslocamento...*** aciona um menu que permite fazer o deslocamento de regiões da peça. Este menu funciona de forma semelhante ao menu de manipulação de regiões do programa de entrada de dados. Todavia, neste menu não é possível reproduzir ou apagar regiões.

O botão ***lista regiões*** cria uma listagem das regiões da estrutura. nesta listagem estão contidas informações a respeito dos meios, corrente, segmentos e pontos que formam a região. Esta listagem é escrita em um arquivo chamado ***regiões***. A tela onde esta listagem aparece é uma janela de edição de texto, conforme mostra a figura 2.20 e, assim, pode-se movimentar dentro desta janela da mesma forma que quando se edita um texto sob o sistema de janelas. Este arquivo pode ser utilizado mesmo após o programa ter sido encerrado.

O funcionamento e a função dos botões ***fecha*** e ***fim*** é idêntido ao descrito no programa de entrada de dados.

Os módulos descritos até aqui compõem a etapa de pré-processamento, de acordo com a figura 2.1. Nos próximos ítens serão descritos os módulos de cálculo que compõem a etapa de processamento.

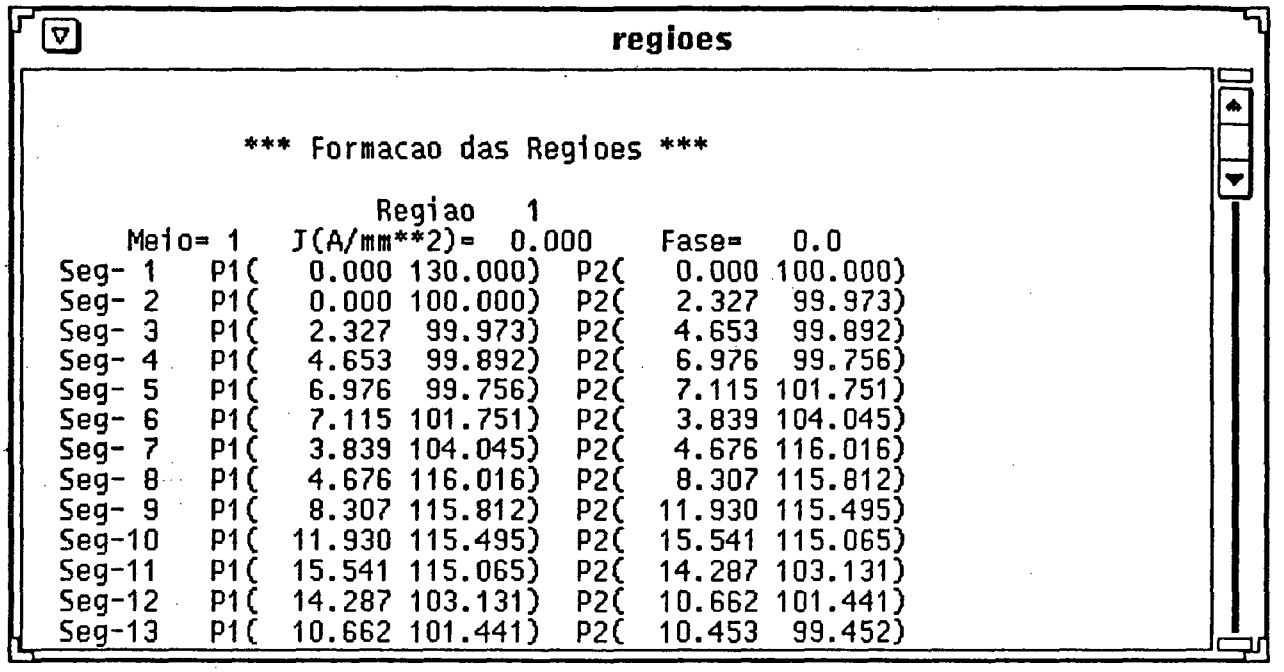

figura 2.20 - Exemplo de listagem de regiões.

2.1.4 Módulo de Cálculo Estático

Neste e nos demais módulos de cálculo são resolvidas as equações diferenciais parciais que regem os fenômenos eletromagnéticos. O método empregado para tanto é o método dos elementos finitos [1,2,12,14,15].

O programa para cálculo estático é chamado de EFCE. Quando for carregado o programa exibe a tela mostrada na figura 2.21. Este programa destina-se ao cálculo de estruturas eletromagnéticas em que as fontes de campo não variam no tempo.

O programa EFCE lê dados no arquivo (não-formatado) produzido pelo gerador de malha e que possui extensão *.ELF*.

O botão ***arquivos...*** aciona um menu que é idêntico ao mostrado no programa EFD. Quando o arquivo escolhido é carregado, a malha a ele associada é desenhada na tela.

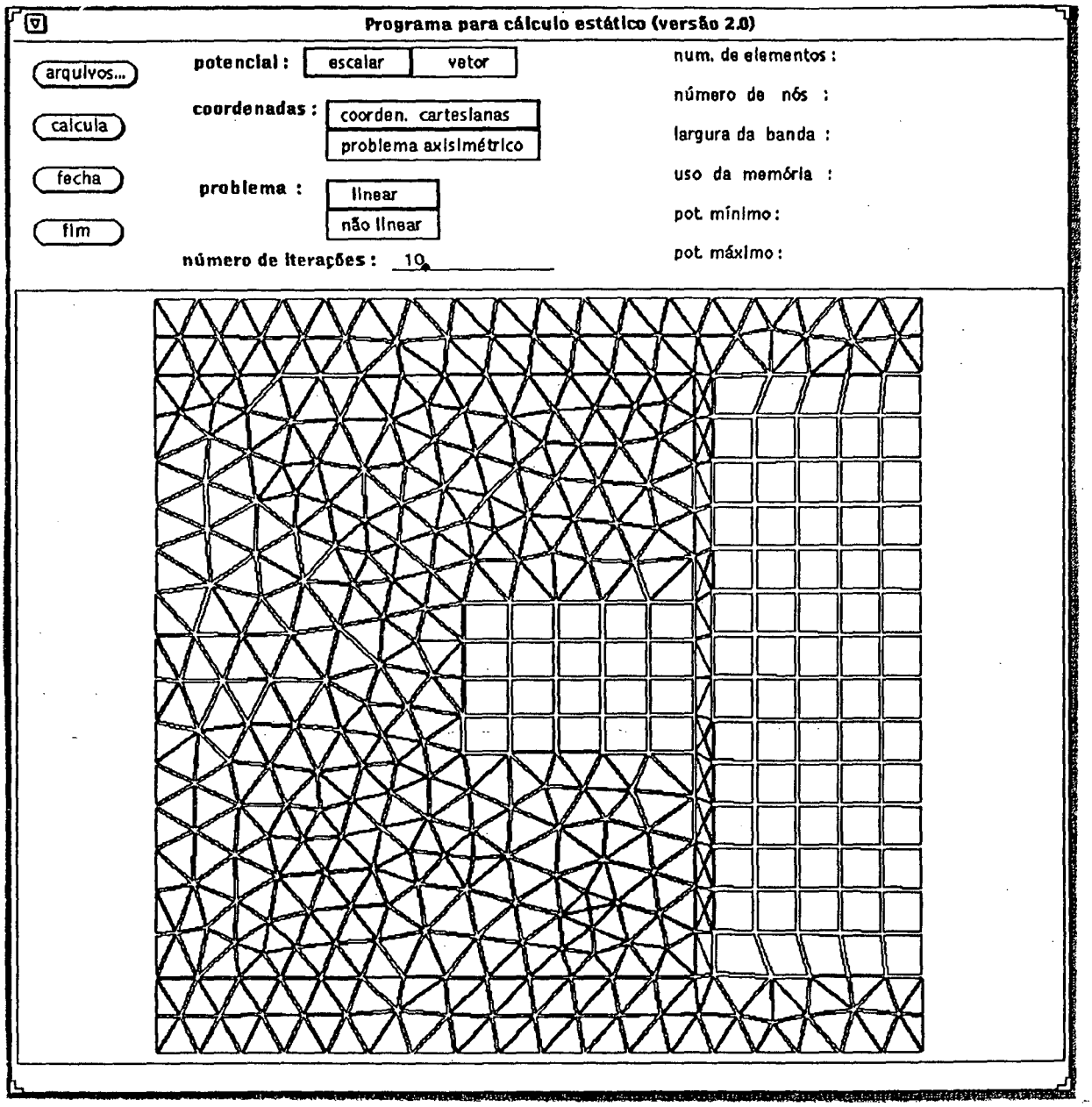

figura 2.21 - Menu principal do programa de cálculo estático.

O botão ***calcula*** efetua o cálculo com as opções estabelecidas nos itens que estão à direita deste botão.

Os botões ***fecha*** e ***fim*** funcionam da mesma forma que no programa de entrada de dados.

O menu principal apresenta as seguintes opções de cálculo:

- ***Potencial Escalar*** e ***Potencial Vetor***, indica o tipo de formulação que será utilizada;

- ***Coordenadas Cartesianas*** e ***Problema axissimétrico*** , indicam se o problema possui simetria em torno do eixo X ou não;
- ***Cálculo linear*** e ***Cálculo não-linear***. No caso linear as permeabilidades são consideradas constantes. No caso não-linear será tomada em conta a curva característica do material e sendo usado o método de Newton-Raphson [1,2,12,16,15]. A curva do material pode ser modificada usando o programa de gerenciamento de arquivo de materiais, chamado de EFP.

Após ter sido feito o cálculo o programa preenche os campos da direita com os dados relativos ao problema. Estes dados são os seguintes:

- ***num. de elementos*** , especifica o número de elementos triangulares contidos na malha;
- ***num. de nós*** , especifica o número de nós contidos na malha;
- ***largura da banda*** , número de elementos na banda da matriz utilizada para o armazenamento do sistema matricial;
- ***uso da memória*** , indica o percentual da memória alocada para o programa que foi utilizado para a resolução do sistema;
- ***Pot. mínimo*** e ***Pot. máximo*** , indicam os valores dos potenciais mínimo e o máximo que foram calculados.

Quando o programa tiver terminado o cálculo os valores são automaticamente gravados no arquivo e uma mensagem é enviada ao usuário indicando que o cálculo terminou.

2.1.5 Módulo de Cálculo com Variação no Tempo - Alimentação em Tensão

Este programa realiza cálculos de estruturas cujas fontes são variáveis no tempo é e chamado de EFCT. Ele faz uma resolução passo a passo no tempo e leva em conta as correntes induzidas para materiais cuja condutância é diferente de zero. Desta forma, define-se a curva de tensão em função do tempo para as fontes de excitação e o programa calculará as correntes que circulam nas bobinas, bem como as correntes induzidas na peça. Os dados de cada fonte podem ser gravados em arquivos, evitando que os mesmos tenham que ser redefinidos toda vez que o programa for executado.

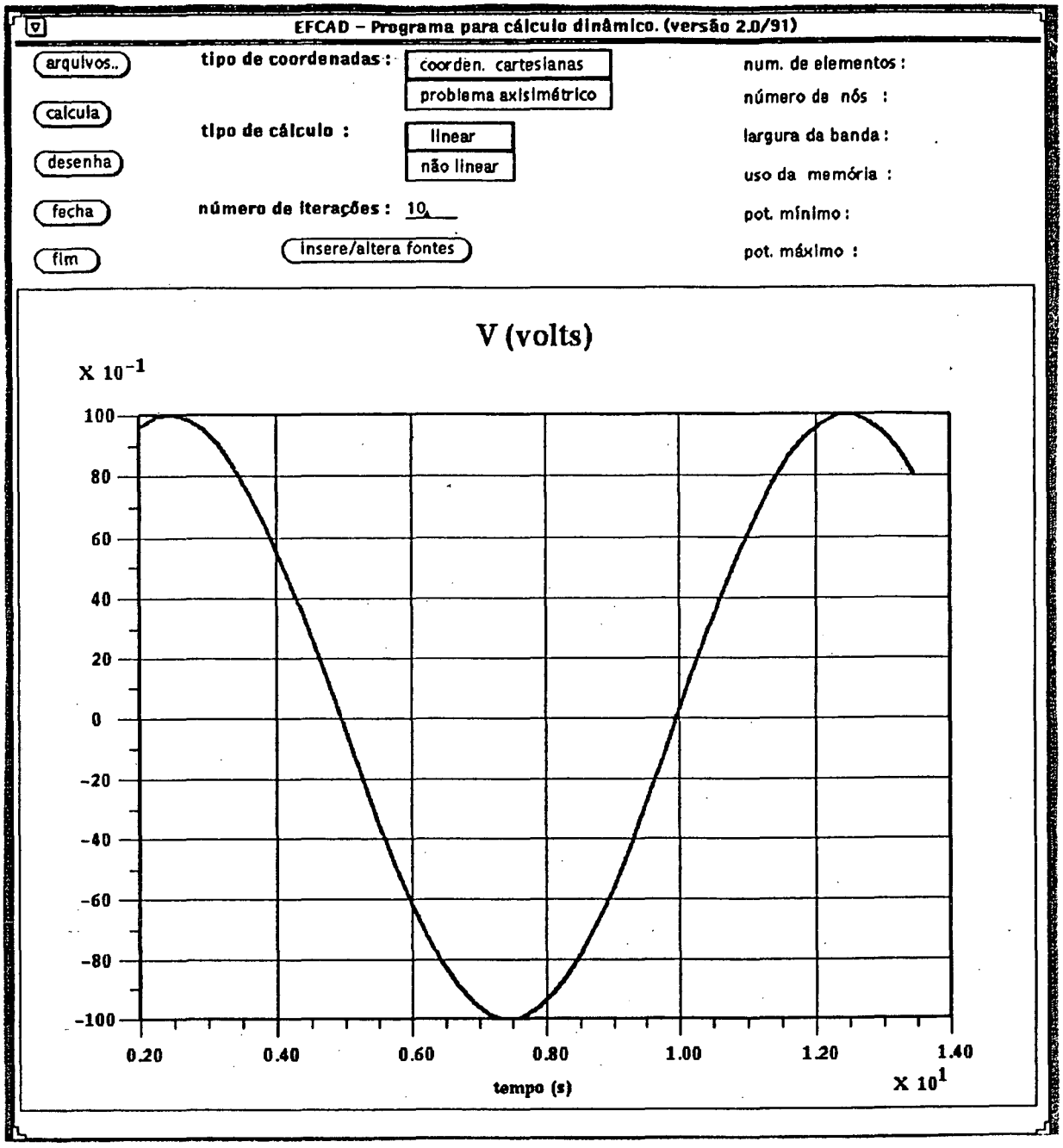

figura 2.22 - Tela principal do programa KFCT.

A tela principal deste programa é mostrada na figura 2.22. Os botões do lado esquerdo ***arquivos...***, ***calcula*** , ***fecha*** e ***fim*** funcionam de forma idêntica ao programa EFCE. Os demais itens também são idênticos ao programa EFCE.

O botão adicional ***desenha*** serve para redesenhar a malha da estrutura na tela, uma vez que com este programa pode-se também visualizar a curva de tensão de cada uma das fontes na tela.

O botão ***altera/edita fontes*** aciona o menu para entrar e alterar os dados das fontes. A figura 2.23 mostra este menu. Os números de 1 a 6 mostrados

se referem ao número da fonte que será utilizada. Este número deve ter sido associado com uma bobina no programa de geração da malha, onde, ao invés de ser fornecido o valor da densidade de corrente, deve ser fornecido o número da fonte.

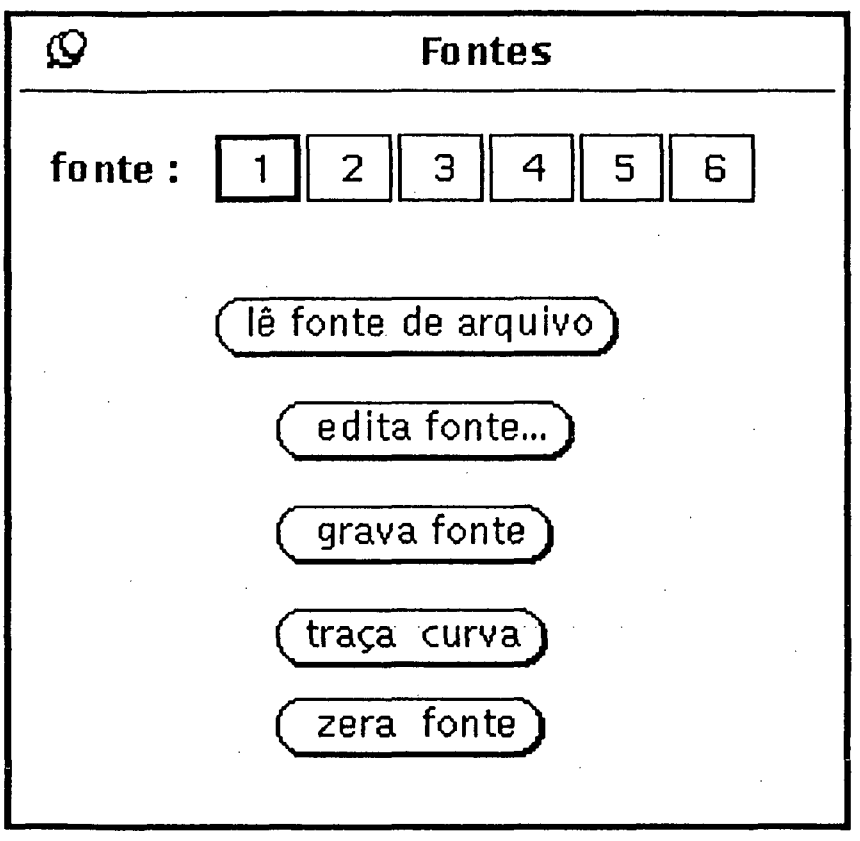

figura 2.23 - Menu para manipulação de fontes.

O botão ***lê fonte de arquivo*** faz a leitura dos dados relativos à fonte especificada e mostra o seu gráfico na tela. Para o traçado do gráfico foi desenvolvida uma rotina para a formatação de escalas e de traçado na tela. Esta mesma rotina é também utilizada por outros programas.

O botão ***edita fonte...*** aciona um submenu onde serão definidos os valores da fonte especificada no ítem fonte. A figura 2.24 mostra este submenu. Os itens no alto definem as características gerais da fonte enquanto que os da parte inferior se referem a curva de tensão versus tempo para a fonte. Toda vez que for feita uma alteração nos dados pode-se visualizá-la por meio do botão ***atualiza/visualiza fonte***. Quando se tratar de uma fonte senoidal os pontos serão automaticamente definidos, sendo desconsiderados os dados na parte inferior do menu

Dados das Fontes

tempo inicial :	2.000e+00	Amplitude (Volts) :	1.000e+01	indutância (H) :	3.000e+00
tempo final :	1.370e+01	Freqüência (Hertz):	1.000e-01	espira/mm² :	4.000e+00
passo de tempo :	5.000e-01	Fase (graus) :	2.000e+00	profundidade (m) :	5.000e+00
passo p/ grav :	7.000e-01	resistência (Ohm) :	1.000e+00	num. setores sim. :	6.000e+00

forma : [senoidal] [não senoidal] (atualiza/visualiza fonte)

t(1) :	1.000e+00	V(1) :	2.000e+01	t(26) :		V(26) :	
t(2) :	3.000e+00	V(2) :	4.500e+01	t(27) :		V(27) :	
t(3) :	7.000e+00	V(3) :	6.900e+01	t(28) :		V(28) :	
t(4) :	8.000e+00	V(4) :	-9.000e+00	t(29) :		V(29) :	
t(5) :	9.000e+00	V(5) :	-5.000e+01	t(30) :		V(30) :	
t(6) :	1.000e+01	V(6) :	3.000e+00	t(31) :		V(31) :	
t(7) :	1.200e+01	V(7) :	3.000e+02	t(32) :		V(32) :	
t(8) :	1.300e+01	V(8) :	-9.000e+01	t(33) :		V(33) :	
t(9) :		V(9) :		t(34) :		V(34) :	
t(10) :		V(10) :		t(35) :		V(35) :	
t(11) :		V(11) :		t(36) :		V(36) :	
t(12) :		V(12) :		t(37) :		V(37) :	
t(13) :		V(13) :		t(38) :		V(38) :	
t(14) :		V(14) :		t(39) :		V(39) :	
t(15) :		V(15) :		t(40) :		V(40) :	
t(16) :		V(16) :		t(41) :		V(41) :	
t(17) :		V(17) :		t(42) :		V(42) :	
t(18) :		V(18) :		t(43) :		V(43) :	
t(19) :		V(19) :		t(44) :		V(44) :	
t(20) :		V(20) :		t(45) :		V(45) :	
t(21) :		V(21) :		t(46) :		V(46) :	
t(22) :		V(22) :		t(47) :		V(47) :	
t(23) :		V(23) :		t(48) :		V(48) :	
t(24) :		V(24) :		t(49) :		V(49) :	
t(25) :		V(25) :		t(50) :		V(50) :	

figura 2.24 - Menu para definição de fontes.

O botão ***grava fonte*** grava em arquivo os dados definidos para a fonte especificada; o botão ***traça curva*** traça o gráfico da fonte e o botão ***zera fonte*** apaga todos os dados relativos a uma dada fonte.

2.1.6 Módulo de Cálculo com Variação no Tempo - Alimentação em Corrente

Este programa, chamado de EFCJ, foi concebido para realizar cálculo de estruturas cujas fontes são alimentadas em corrente e variáveis no tempo. Ele é

muito semelhante ao programa EFCT descrito no ítem 2.1.5. Ele também toma em conta as correntes induzidas em partes condutoras da estrutura utilizando o método de resolução passo a passo. A forma de onda da densidade de corrente imposta pode ser qualquer. Também neste módulo pode-se fazer cálculos não-lineares.

Todas as telas são idênticas ao programa EFCT, exceto a tela de entrada de dados das fontes, onde ao invés de se entrar com uma curva de tensão deve-se entrar com uma curva de densidade de corrente.

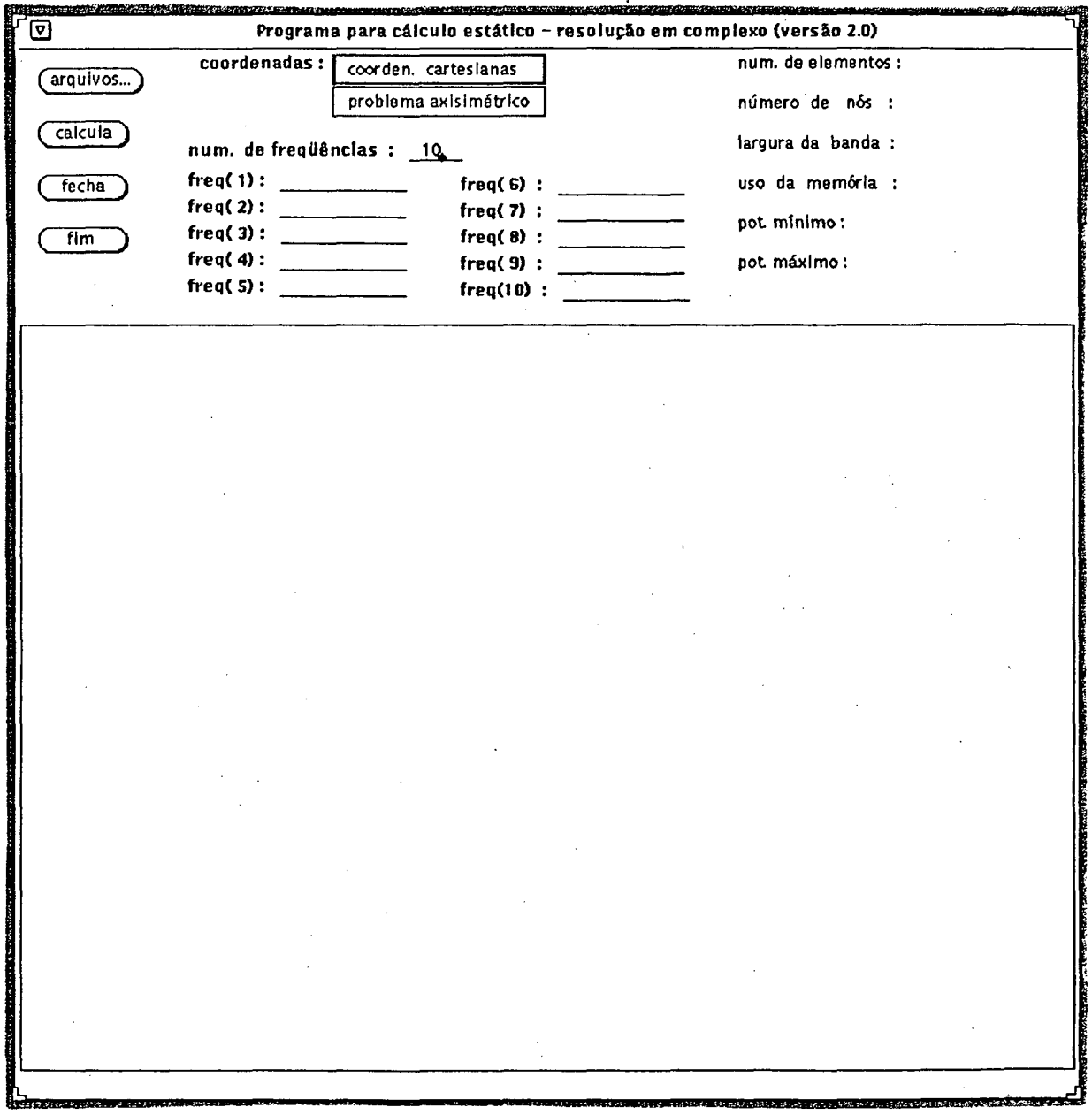

figura 2.25 - Tela principal do programa de cálculo para várias freqüências.

2.1.7 Módulo de Cálculo com Frequência Variável

Este programa é semelhante ao programa para cálculo estático. Todavia ele utiliza uma formulação em complexo para efetuar cálculos de correntes induzidas em materiais com condutividade diferente de zero. Com a formulação empregada só podem ser tratados casos lineares e em regime permanente senoidal.

A tela principal do programa está mostrada na figura 2.25. Os itens do painel principal são idênticos aos descritos anteriormente nos outros programas. Existe, no entanto, itens adicionais para a especificação das freqüências em que o cálculo será feito. O número de freqüências é estipulado no ítem ***num. de freqüências*** e nos itens subseqüentes são especificados os seus valores.

Para cada freqüência fornecida será feito um cálculo, resultando uma seqüência de potenciais que será automaticamente gravada no arquivo.

2.1.8 Módulo de Cálculo de Estruturas com Velocidade

Este módulo permite analisar estruturas que possuem partes dotadas de velocidade. O movimento relativo das partes provoca o aparecimento de correntes induzidas em partes que possuem condutividade diferente de zero. As correntes que são induzidas provocam uma deformação das linhas de campo. Como nos outros programas de cálculo, este programa lê dados gerados pelo programa gerador de malha. Ele também utiliza os dados dos materiais contidos no arquivo ***efmat.dat***.

A tela principal do programa EFCV é exatamente igual a do programa EFCE descrito anteriomente.

2.1.9 Módulo de Gerenciamento de Arquivo de Materiais

Este módulo serve para a introdução das caraterísticas dos materiais que serão utilizadas nos cálculos. Ele é chamado de EFP e a sua tela principal está mostrada na figura 2.26. Com este módulo pode-se definir novos materiais e alterar os já existentes. Também é possível a visualização das curvas características dos materiais. O arquivo que contém os materiais é chamado de ***efmat.dat***.

O botão ***criar*** apaga todos os dados que estão armazenados no arquivo ***efmat.dat***. Como esta operação pode significar a perda de dados, antes de serem

apagados os dados será pedido uma confirmação ao usuário. Caso confirmado, os dados serão realmente apagados.

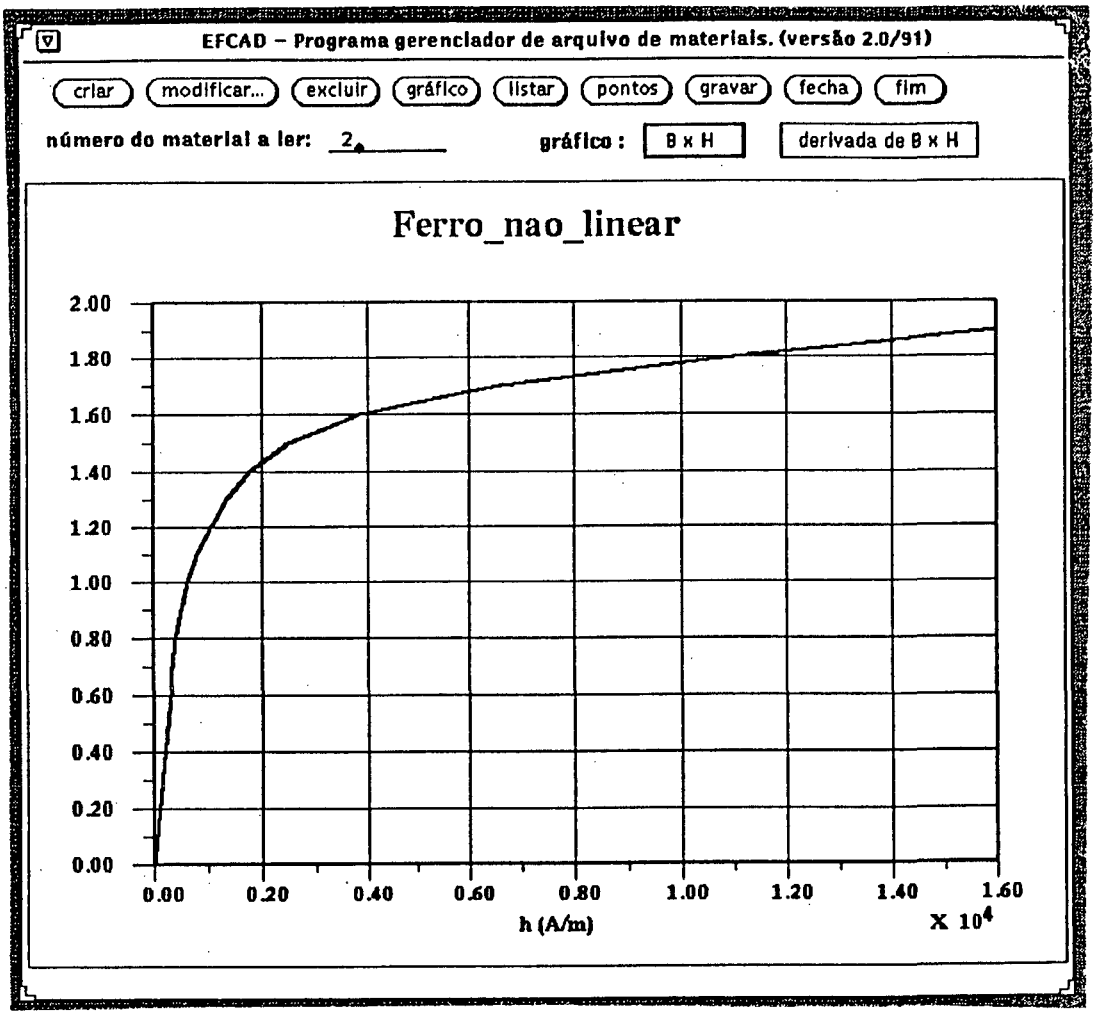

figura 2.26 – Tela principal do módulo EFP.

O botão ***modificar*** aciona o menu que altera os dados relativos ao material especificado no ítem ***material a ler***. A figura 2.27 mostra este menu. Na parte superior deve-se especificar os dados gerais do material e na parte inferior os dados das curvas características. Por exemplo, se o material for magnético deve-se fornecer os dados da curva de magnetização (curva B versus H). A cada material contido no arquivo está associado um número. Pode-se visualizar a curva característica do material acionando-se o botão ***gráfico*** . Pode-se, igualmente, visualizar a curva da derivada selecionando-se a opção ***derivada de B x H***. A figura 2.26 mostra um gráfico típico para um material ferromagnético.

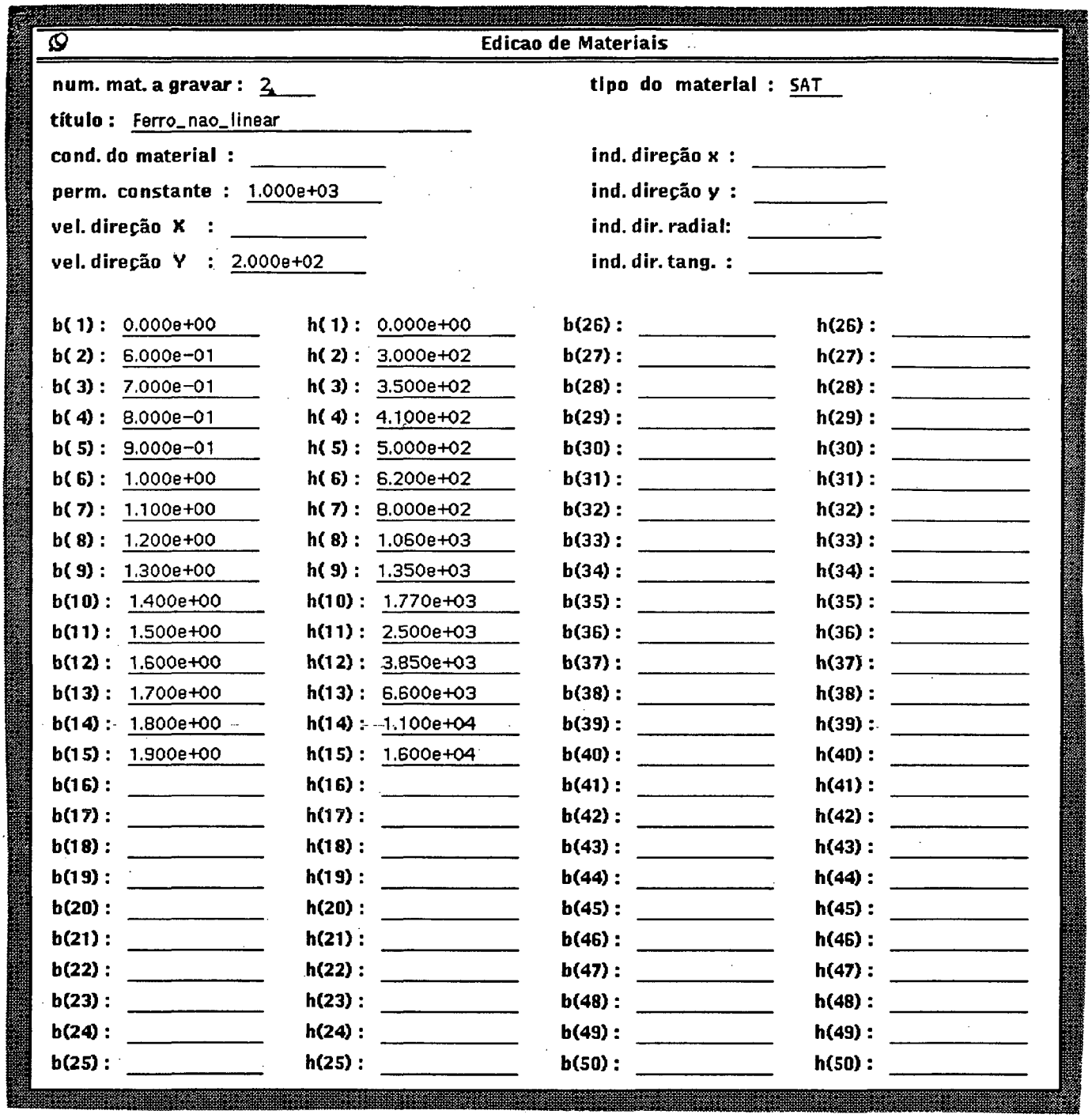

figura 2.27 - Menu de entrada de dados de materiais.

O botão ***grava*** efetua a gravação no arquivo dos dados correntemente definidos para um certo material. O número do material que será gravado é definido no ítem ***num. mat. a gravar*** do menu de edição.

O botão ***excluir*** elimina o material especificado no ítem material a ler do arquivo de materiais.

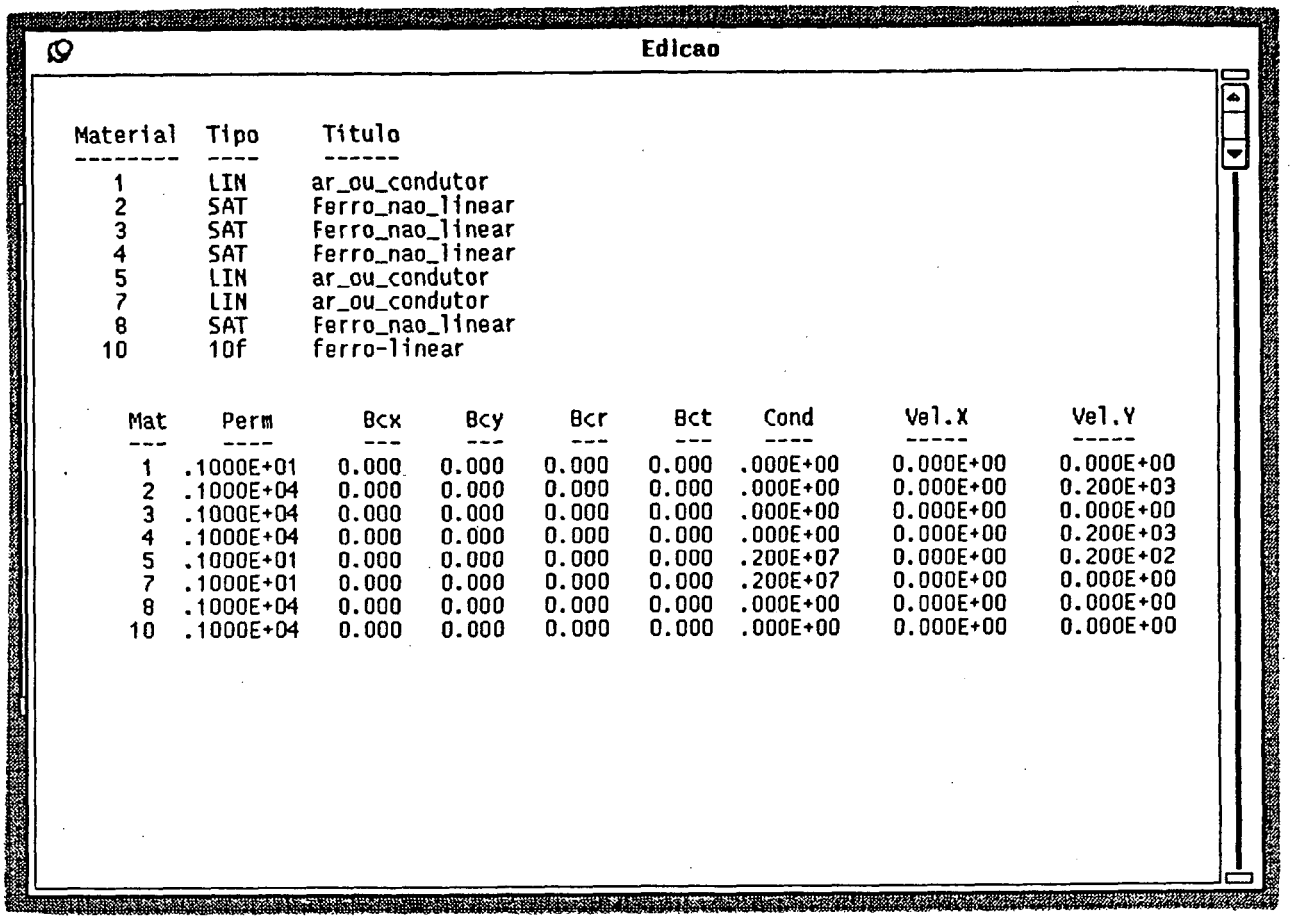

figura 2.28 - Listagem dos materiais.

O botão ***lista*** cria uma listagem de todos os materias que estão definidos no arquivo ***efmat.dat***. Esta listagem aparece em uma janela de texto conforme mostrado na figura 2.28. Nesta janela podem ser feitas operações de rolar a tela para a frente e para trás, além de outras operações permitidas neste tipo de janela.

O botão ***pontos*** cria uma listagem das características de um material na forma também de uma janela de texto semelhante a anterior.

Os botões ***fecha*** e ***fim*** funcionam da forma já descrita para os outros módulos.

2.1.10 Módulo Exploração Numérica e Gráfica

Após ter sido feito o cálculo pode-se fazer a exploração numérica e gráfica dos resultados através do módulo EFGN. Este módulo incorpora as funções de dois módulos separados no sistema para PC:

- módulo de exploração gráfica (EFG);
- módulo de exploração numérica (EFN);

Devido às facilidades oferecidas pela estação de trabalho foi possível reunir as funções dos dois módulos num único, facilitando bastante o trabalho do

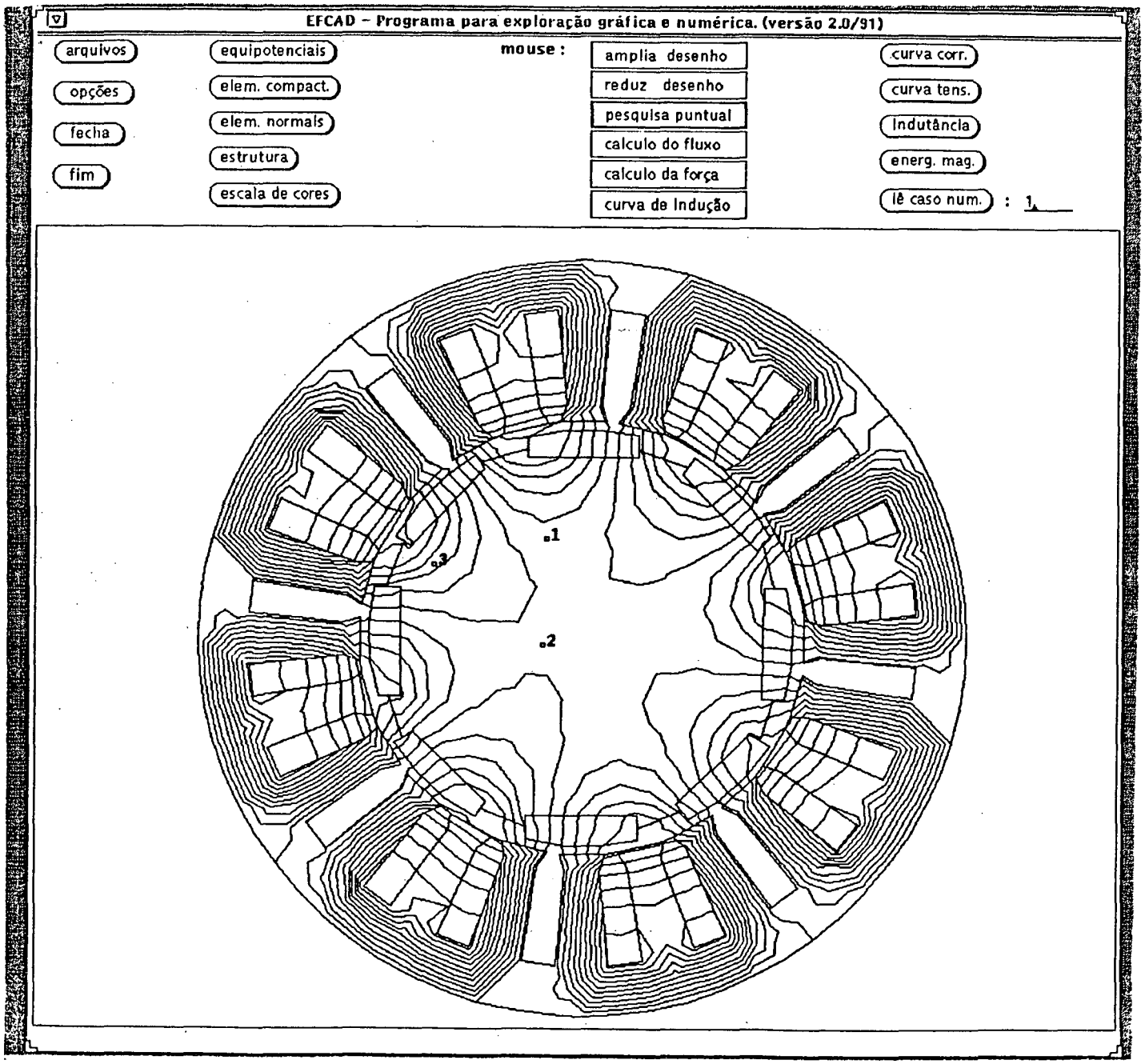

figura 2.29 - Tela principal do program EFGN.

A tela principal deste módulo é mostrada na figura 2.29.

O menu de manipulação de arquivos, acionado pelo botão ***arquivos*** , é idêntico aos anteriormente descritos. Quando um arquivo com extensão ***.ELF*** for lido o programa faz automaticamente o cálculo das eqüipotenciais e as traça na tela. No arquivo também está contida a informação sobre qual módulo de cálculo foi utilizado. Também no caso de haver mais de um conjunto de potenciais o programa informará quantos existe e o usuário poderá escolher qual o caso desejado.

O botão ***opções*** acessa o menu mostrado na figura 2.30. Neste menu são definidas as opções de traçado e cálculos que serão feitos pelo programa tais como tipo de potencial, a componente da indução que será traçada quando solicitado traçado da curva de indução sobre uma região, dados relativos às fontes, número de de equipotenciais e o fator utilizado para zoom.

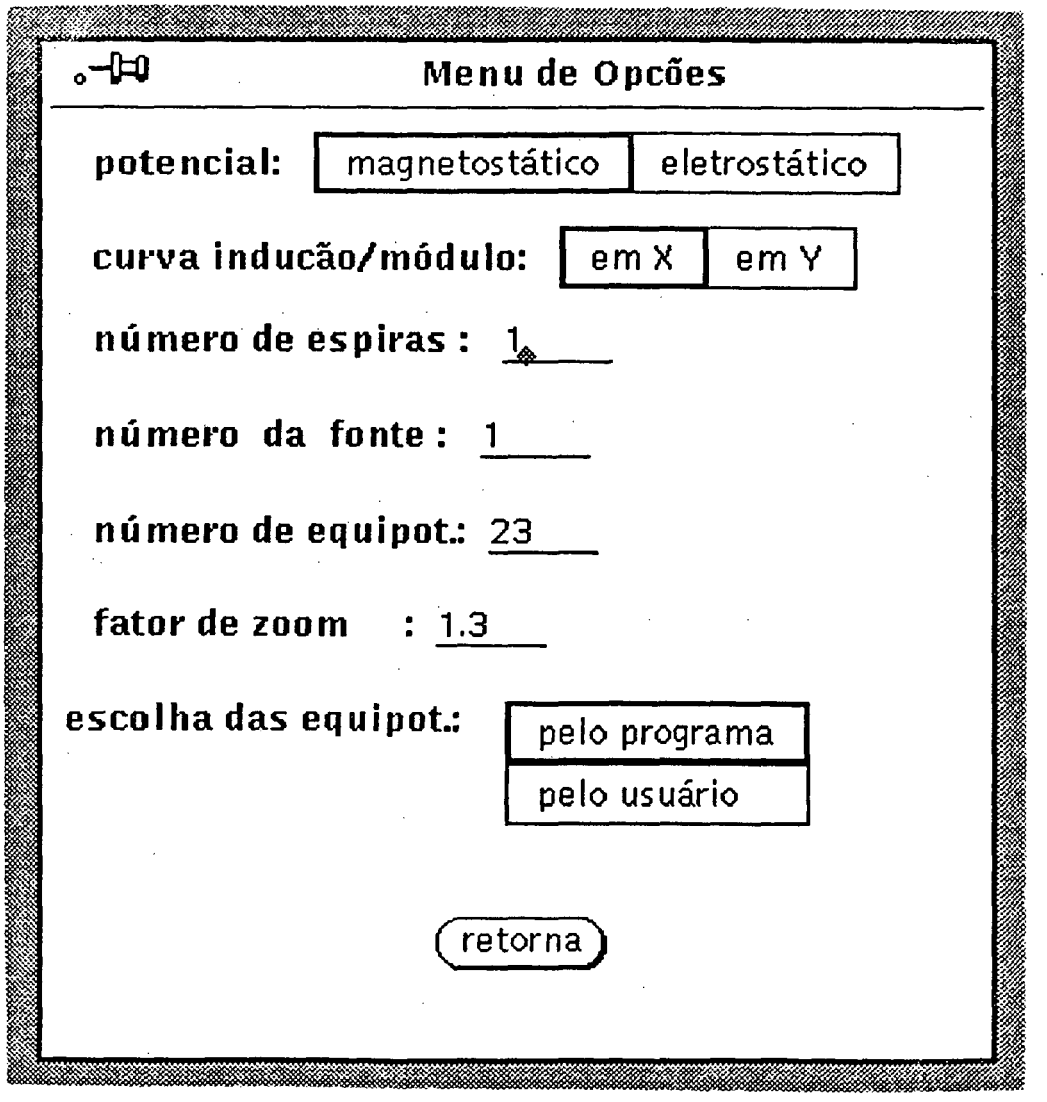

figura 2.30 - Menu de opções.

O botão ***eqüipotenciais*** faz com que as eqüipotenciais sejam calculadas e traçadas. O valor de cada eqüipotencial é apresentado numa tela que aparece à direita da área de desenho chamada de ***eqüipotenciais***. Esta janela também é do tipo texto. O número de equipotenciais que serão traçadas é estabelecido no ítem ***número de equipot.*** no painel de opções. Este número inicialmente é colocado em 23 e pode ser aumentado até 55. Os valores das eqüipotenciais são normalmente gerados automaticamente pelo programa mas podem também ser fornecidos pelo usuário. Neste último caso a janela da direita servirá também para entrar estes valores. A figura 2.31 mostra um exemplo de traçado de eqüipotenciais.

Os botões ***elem. compactados*** e ***elem. normais*** traçam a malha da estrutura. No caso de elementos compactados o traçado é feito deixando-se um pequeno espaço entre cada elemento da malha.

O botão ***estrutura*** desenha a estrutura. As regiões que contêm correntes serão traçadas em vermelho.

O botão ***escala de cores*** traça a densidade de fluxo numa escala de cores proporcional. Para tanto é usada uma escala de 45 cores variando de forma contínua do azul claro ao vermelho. Esta opção produz uma visualização que vai muito além da que o PC oferece, pois a passagem de uma cor para a outra é muito mais suave produzindo um efeito visual melhor.

O ítem ***mouse*** estabelece qual a função que o mouse executará. As escolhas possíveis são:

- ***amplia desenho*** e ***reduz desenho*** amplia o desenho de acordo com o fator estabelecido no menu de opções. Pode-se ampliar ou reduzir o desenho da estrutura, da malha e das eqüipotenciais;
- ***pesquisa pontual*** , utiliza-se o botão esquerdo do mouse para indicar os pontos onde se deseja calcular as grandezas eletromagnéticas, tais como densidade de fluxo e valores de campo. Os valores calculados são apresentados em uma janela à direita da área de desenho chamada de ***Resultados***. Esta forma de calcular as grandezas difere daquela utilizada pela versão em PC, uma vez que não é preciso indicar todos os pontos de uma vez para fazer o cálculo das grandezas e em seguida apagar a tela do desenho e mostrar os valores calculados. A figura 2.30 também ilustra este modo de operação do mouse;

- ***cálculo do fluxo,*** utiliza-se o botão esquerdo do mouse para definir-se o pontos da região sobre a qual o fluxo será calculado. Após definida a região, usa-se o botão direito para fazer o cálculo;
- ***cálculo da força*** , o funcionamento do mouse é idêntico ao anterior mas é feito o cálculo da força sobre a superfície ao invés do fluxo;
- ***curva de indução***, com o botão esquerdo do mouse define-se os dois pontos do segmento sobre o qual será traçada a curva de indução.

Na parte direita do painel principal existem ainda os seguintes botões:
- ***curva corr.*** , quando houver uma curva de corrente associada com uma bobina esta opção traçará na tela o seu gráfico. O número da bobina é especificado no ítem ***número da fonte*** do menu de opções;
- ***curva tens.*** , idem ao ítem anterior para o caso da curva de tensão;
- ***indutância*** , calcula a indutância da bobina especificada no ítem número da fonte. Também é utilizado o número de espiras especificado no painel de opções.
- ***energia magnética*** , faz o cálculo da energia armazenada na estrutura;
- ***lê caso num.:*** , lê e traça um novo conjunto de potenciais. Quando for feito o cálculo da estrutura, a cada conjunto de potenciais é associado um número (chamado de caso). Teclando-se ***enter*** após o número especificado, o cálculo e o traçado das eqüipotenciais é imediatamente feito. O mesmo efeito é conseguido acionando-se o botão ***eqüipotenciais***
.

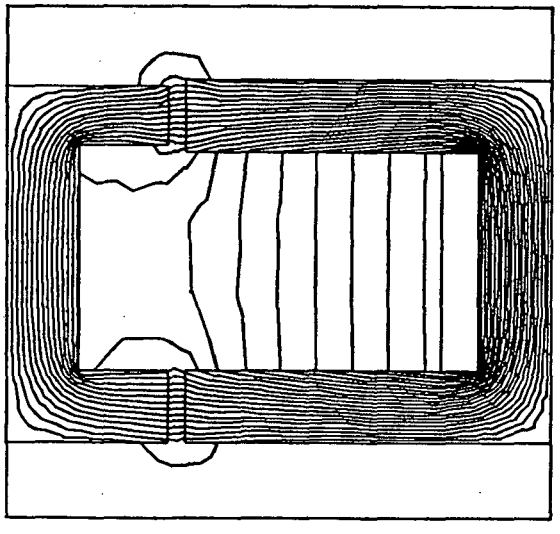

figura 2.31 - Traçado de eqüipotenciais

2.2 Conclusão

Neste capítulo foram apresentados os módulos do sistema implantado em estações de trabalho e ambiente Unix.

O módulo de estrada de dados possibilita uma entrada de dados de várias formas, além de opções de manipulação de regiões e de reprodução de partes da estrutura. Nos módulos de cálculo com variação no tempo (EFCT e EFCJ), além da interface gráfica, também foram incluídas telas para a entrada de dados das fontes e arquivos para as mesmas. Desta forma, os dados das fontes podem ser armazenados e reutilizados. A entrada dos dados das fontes, comparativamente ao PC, é mais fácil pois os erros na entrada de valores não exigem que todos os valores sejam novamente fornecidos. Nestes módulos também é possível ver o gráfico da curva que foi fornecida. Os módulos de exploração gráfica e numérica foram reunidos num único bloco, possibilitando uma exploração melhor dos resultados.

No próximo capítulo serão mostrados alguns exemplos e resultados obtidos com o sistema implantado.

3. Resultados e Exemplos

3.1 Introdução

Neste capítulo serão apresentados algumas estruturas analisadas com o sistema implantado em estações. As duas primeiras estruturas foram processados em diferentes tipos de máquinas a fim de comparar os resultados em matéria de velocidade de processamento. A segunda estrutura mostra a análise de uma estrutura com um número bastante elevado de elementos e que só pode ser analisada em estações de trabalho devido ao número de elementos que ela possui.

A última estrutura é analisada sob diversas condições tais como alimentação com tensão imposta e com freqüência variável.

3.2 Exemplo 1

O primeiro caso analisado é uma máquina de indução com corrente apenas no estator, mostrada na figura 3.1. A densidade de corrente imposta no estator é de 10.0 A/mm^2 e o material do estator e do rotor é o ferro magnético cuja curva está mostrada na figura 2.26. Conforme mostrado, foi introduzido apenas uma parte da máquina devido ao fato da máquina possuir simetria e ao limite de memória, uma vez que a estrutura será processada tanto em PC's como em estações. As condições de fronteira também são mostradas na figura 3.1.

A figura 3.2 mostra a malha utilizada e a figura 3.3 mostra as linhas de fluxo da estrutura. A malha utilizada consta de 833 elementos e 457 nós. Para esta estrutura foi feito um cálculo não linear com um máximo de 20 iterações.

A tabela 1 mostra os tempos de cálculo em segundos obtidas processando-se a mesma estrutura em estações Sun e em PC's do tipo 286, 386 e 486.

Equipamento	Tempo (s)
SPARC Station 2	6
SPARC Station 1+	11.5
Sun LCD	18
PC 486/25	57
PC 386/33	120
PC 286	780

Tabela 1 - Tempos de cálculo em segundos para a estrutura 1.

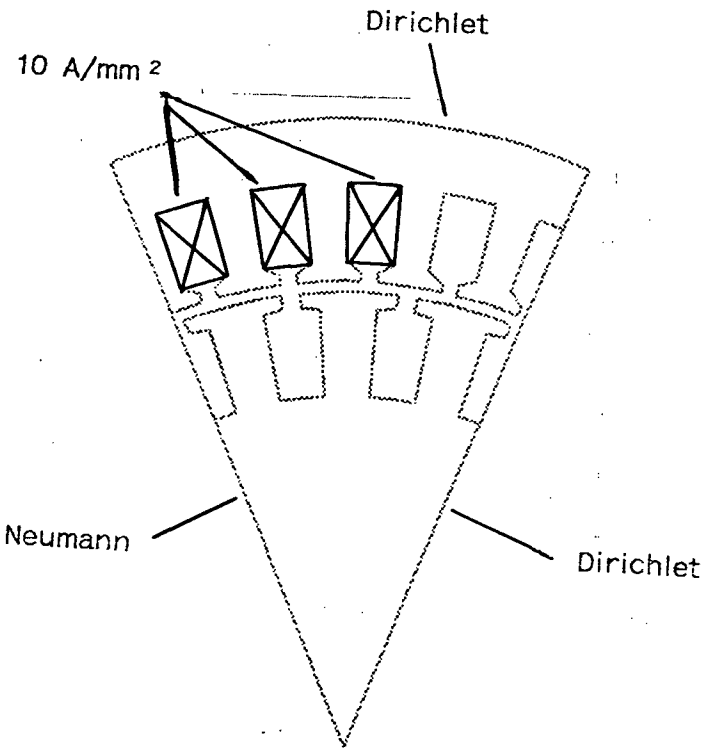

figura 3.1 - Estrutura 1.

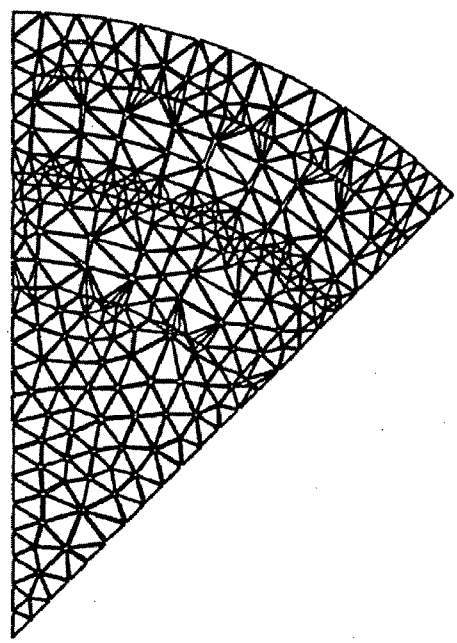

figura 3.2 Malha da estrutura 1.

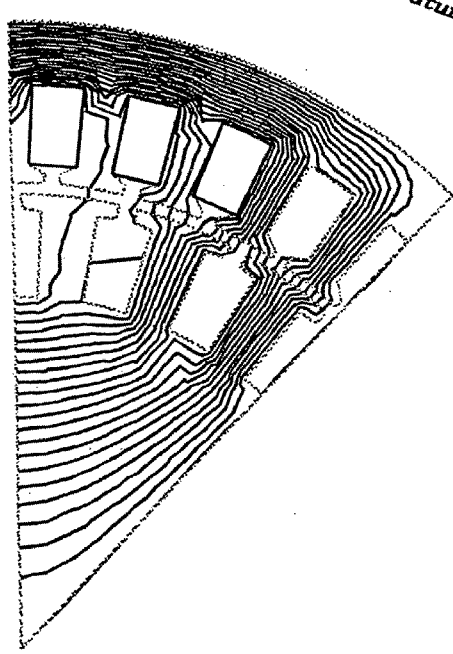

figura 3.3 Equipotenciais da estrutura 1.

3.3 Exemplo 2

A segunda estrutura analisada é uma máquina síncrona sob carga e está mostrada na figura 3.4. A densidade de corrente no estator é 10 A/mm ^{2} e no rotor é de 15 A/mm^2. O material que constitui o rotor e o estator são os mesmos que o do exemplo 1. A malha consta de 1003 elementos e 564 nós. Também nesta estrutura foi feito um cálculo não-linear com um máximo de 20 iterações. As condições de fronteira impostas ao caso também são mostradas na figura 3.4.

A tabela 2 mostra os tempos em segundos obtidos ao se processar esta mesma estrutura em várias máquinas diferentes.

Equipamento	Tempo (s)
SPARC Station 2	8
SPARC Station 1+	18
Sun LCD	29
PC 486/25	90
PC 386/33	180
PC 286	470

Tabela 2 - Tempos de cálculo em segundos para a estrutura 2.

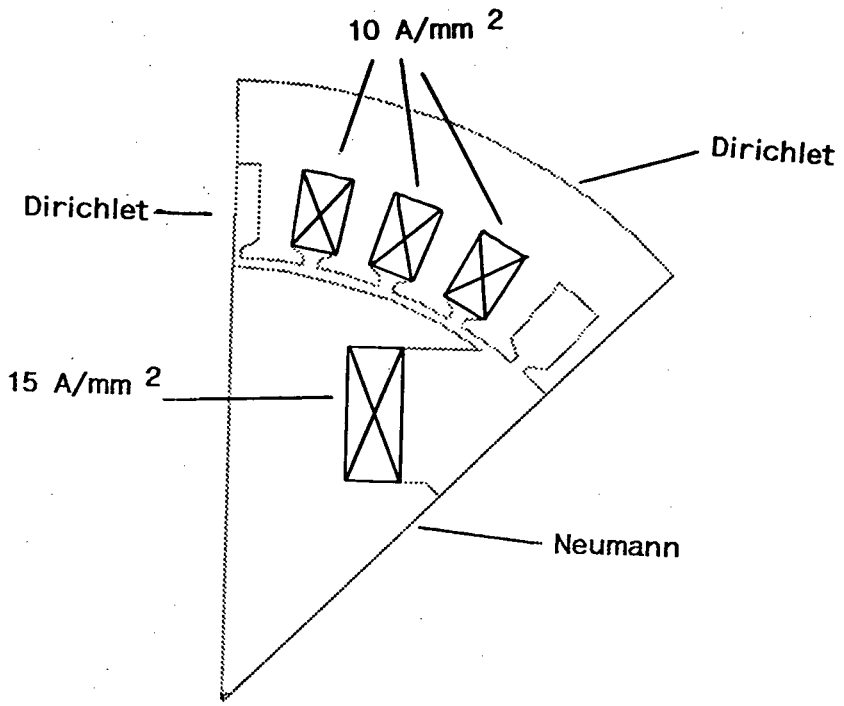

figura 3.4 - Estrutura 2.

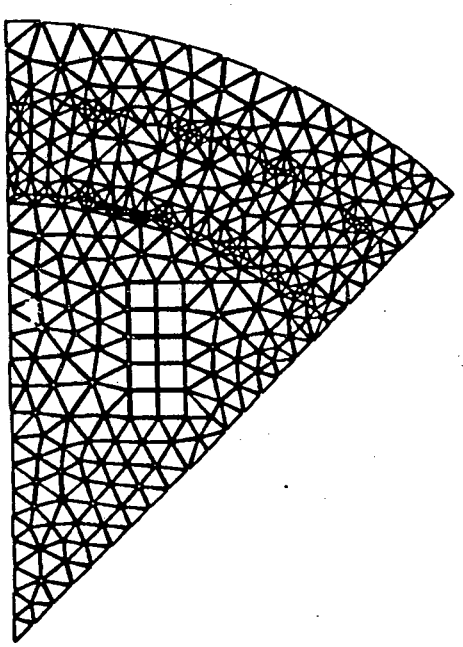

figura 3.5 Malha da estrutura 2.

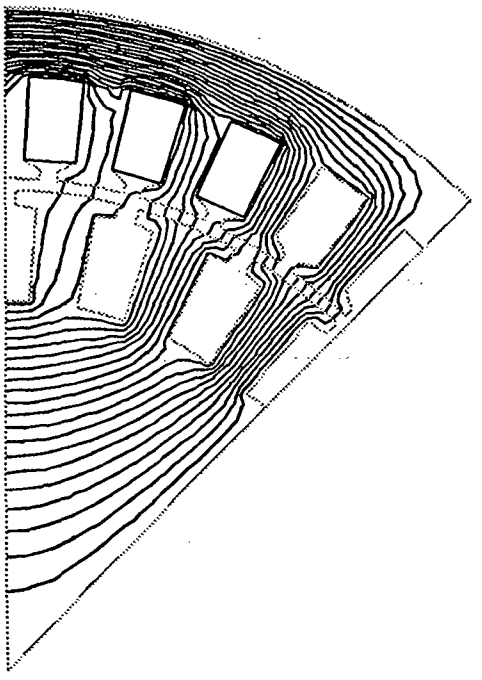

figura 3.6 Eqüipotenciais da estrutura 2.

3.4 Exemplo 3

A terceira estrutura analisada foi a máquina do exemplo 2 mas, ao invés de se analisar apenas parte da máquina, foi inserida toda a máquina. Como no caso anterior, a densidade de corrente no estator é de 10 A/mm^2 e no rotor é de 15 A/mm^2. Em muitos casos práticos pode ser útil uma análise que inclua toda a estrutura da máquina em estudo.

A estrutura está mostrada na figura 3.7. A malha para este caso está mostrada na figura 3.8 e consta de 4212 elementos e 2189 nós. A condição de Dirichlet foi imposta ao longo de toda a fronteira externa da máquina. Neste caso foi feito tanto um cálculo linear como um cálculo não-linear. Para o cálculo linear obteve-se um tempo de 11 segundos e para cálculo não linear obteve-se um tempo de 95 segundos.

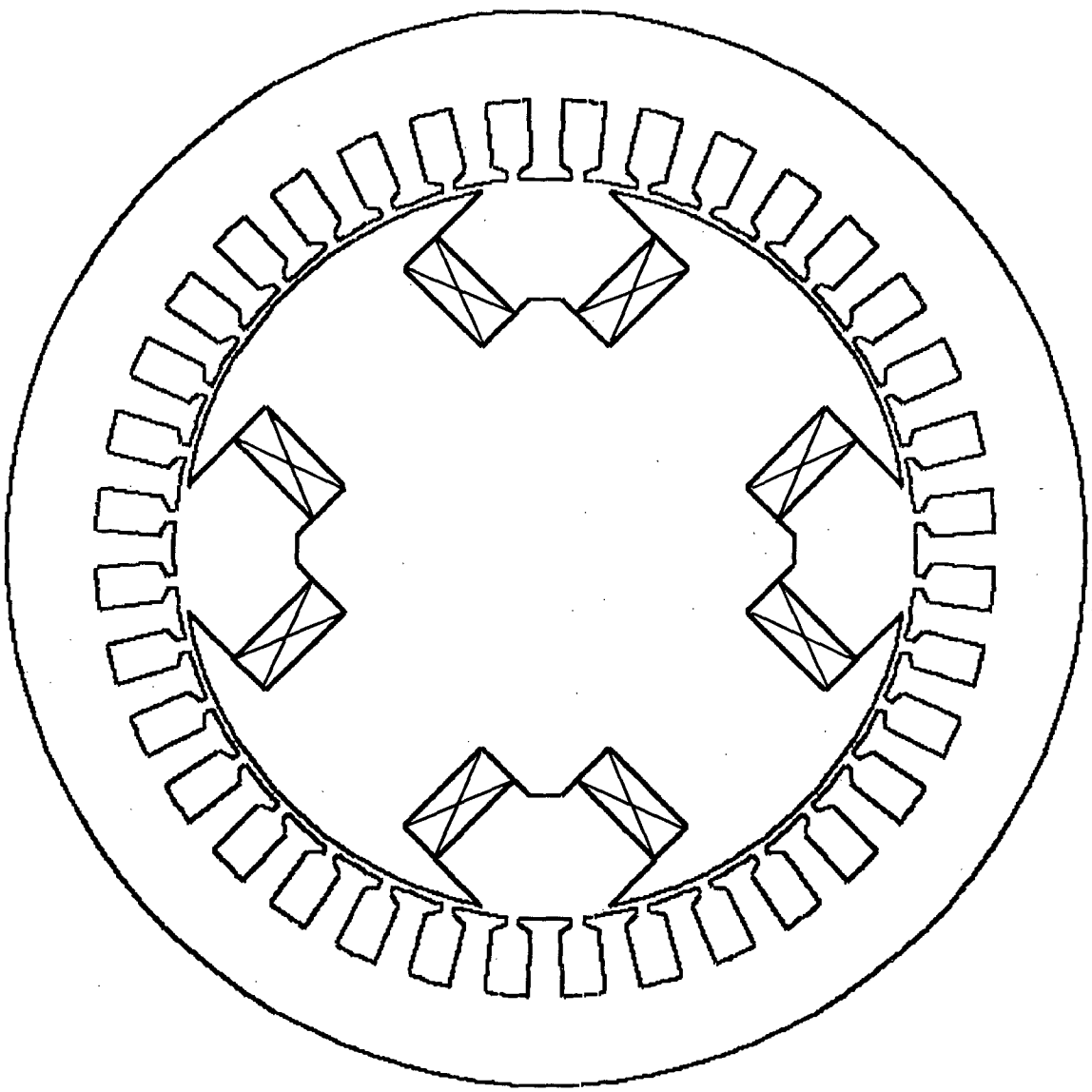

figura 3.7 Estrutura 3.

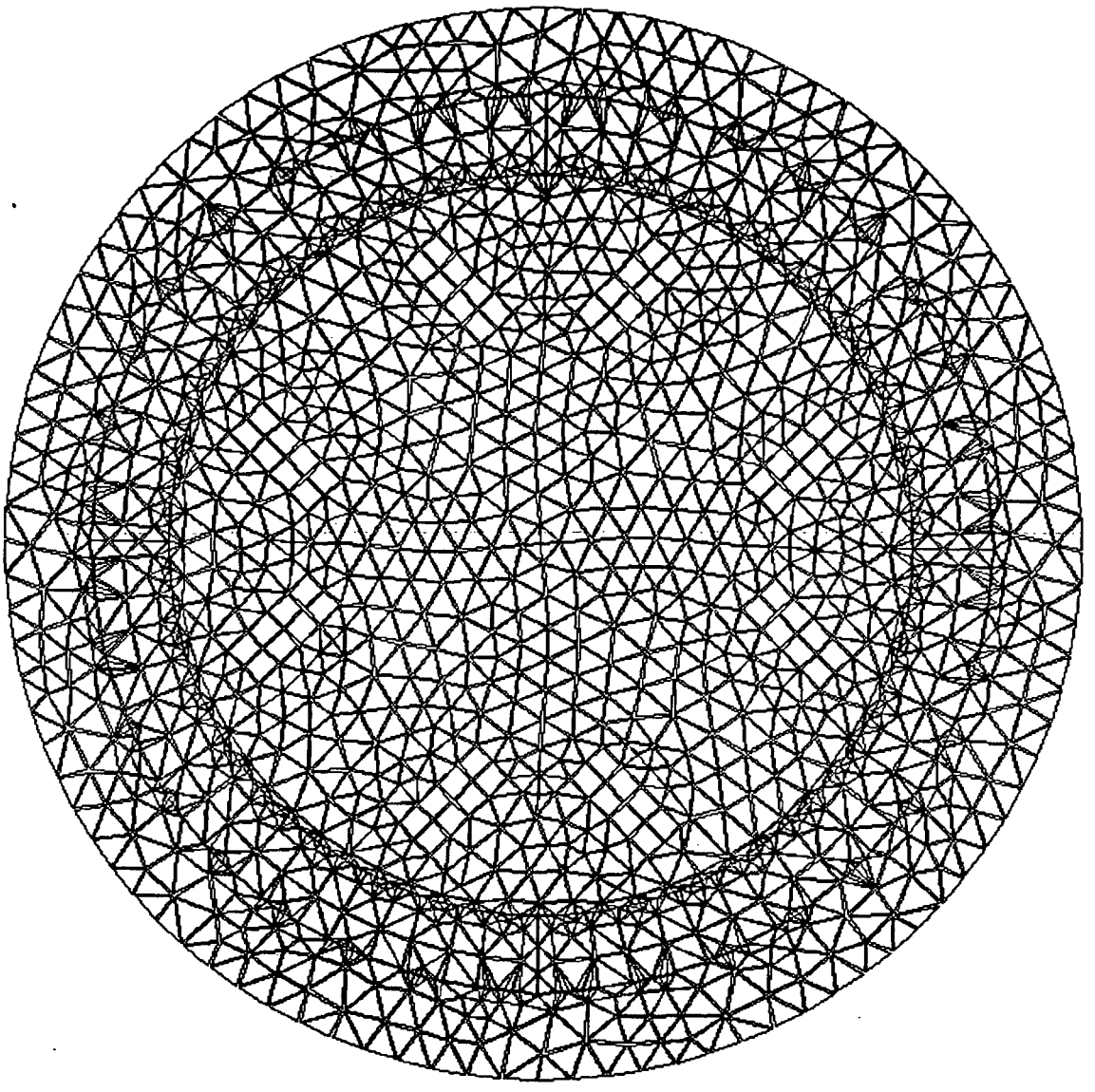

figura 3.8 Malha da estrutura 3.

3.5 Exemplo 4

A estrutura analisada neste exemplo é mostrada na figura 3.10. Trata-se de uma bobina de corrente colocada diante de uma peça condutora. A parte superior da peça é de ferro magnético. Os dados da parte condutora são os seguintes

- condutividade : 1.0 x 10^{+4} $(Ohm.m)^{-1}$
- permeabilidade relativa : 1000 .

A malha utilizada está mostrada na figura 3.11. Ela consta de 304 nós e 394 elementos. Foi imposta a condição de Dirichlet na fronteira externa da estrutura.

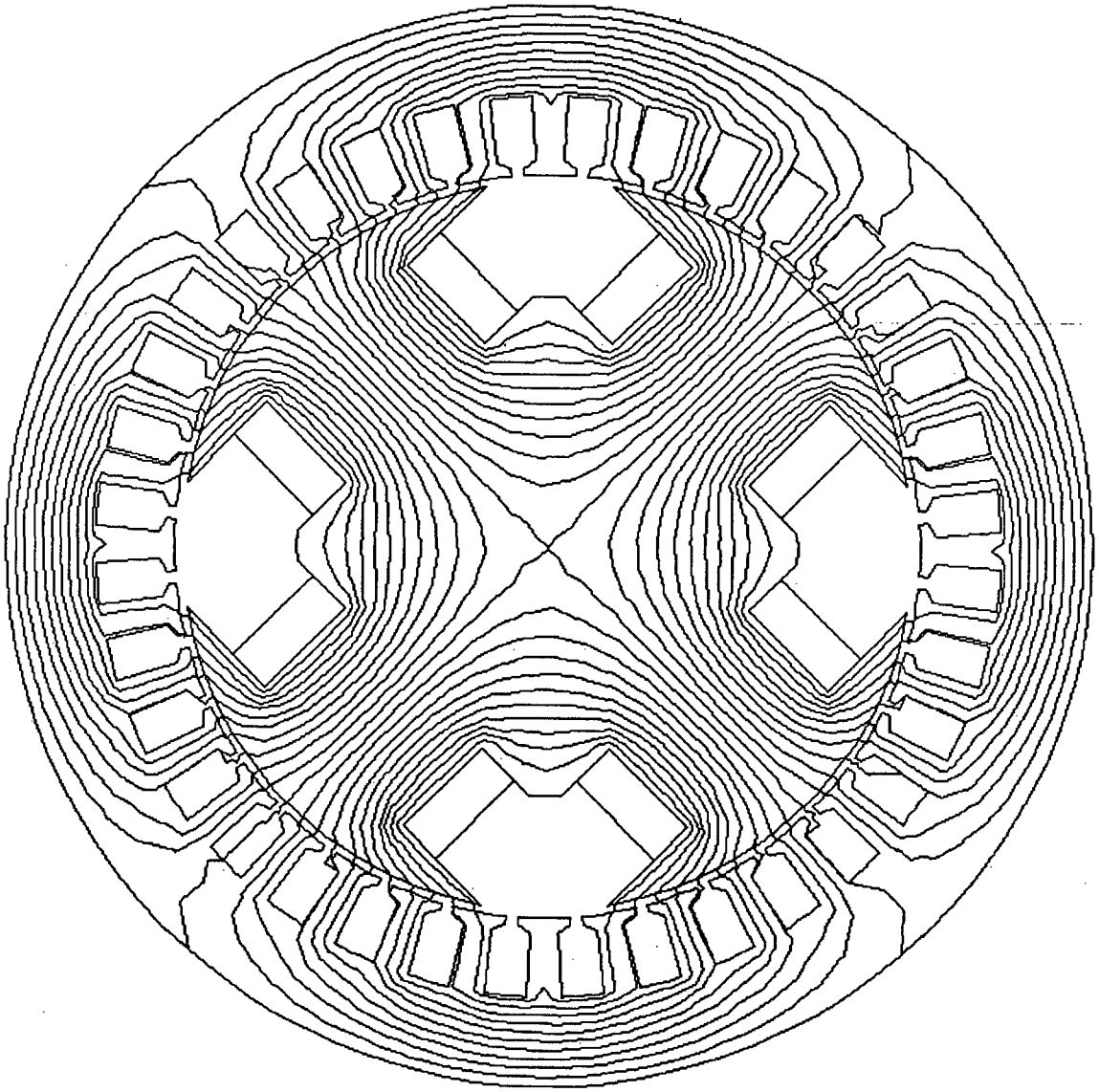

figura 3.9 Linha eqüipotenciais da estrutura 3.

Primeiramente é feita uma análise considerando a estrutura como um caso estático alimentado com uma densidade de corrente de 2 A/mm ^{2}. Verifica-se que o campo é totalmente simétrico e as linhas de campo penetram totalmente na peça, conforme mostrado na figura 3.12. Este cálculo foi feito utilizando-se o programa EFCE e considerando o caso não linear.

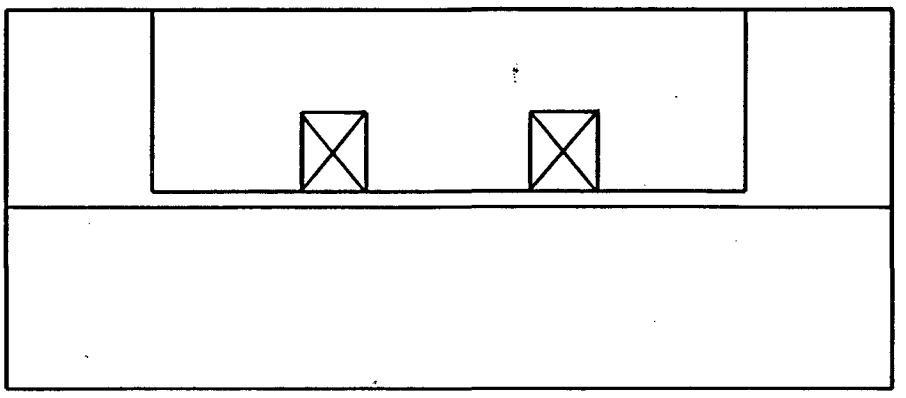

figura 3.10 - Estrutura do exemplo 4

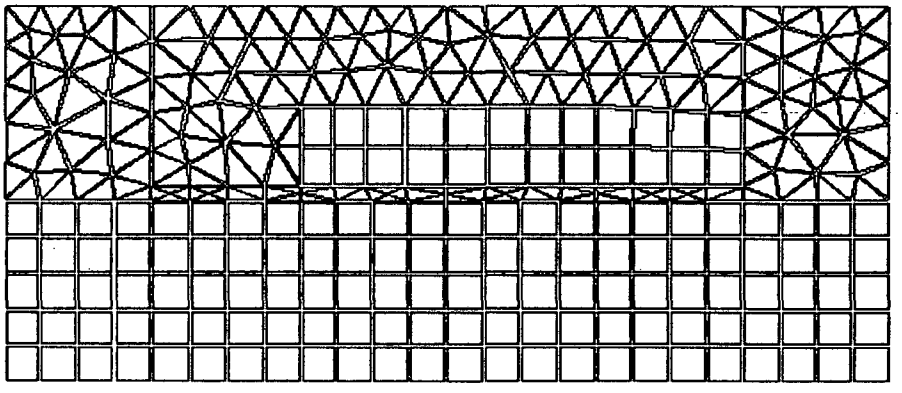

figura 3.11 - Malha do exemplo 4

A segunda análise foi feita considerando que a parte inferior da estrutura possui uma velocidade de 10 m/s, no sentido do eixo Ox positivo. Neste caso haverá correntes induzidas na peça que farão com que a configuração do campo mude em relação ao caso estático. A figura 3.13 mostra claramente esta mudança na configuração do campo. As linhas equipotenciais são deslocadas no sentido da velocidade da peça. Também neste caso foi considerado um caso não linear e com alimentação da bobina de 2 A/mm^2, como no caso anterior. O módulo utilizado foi o EFCV.

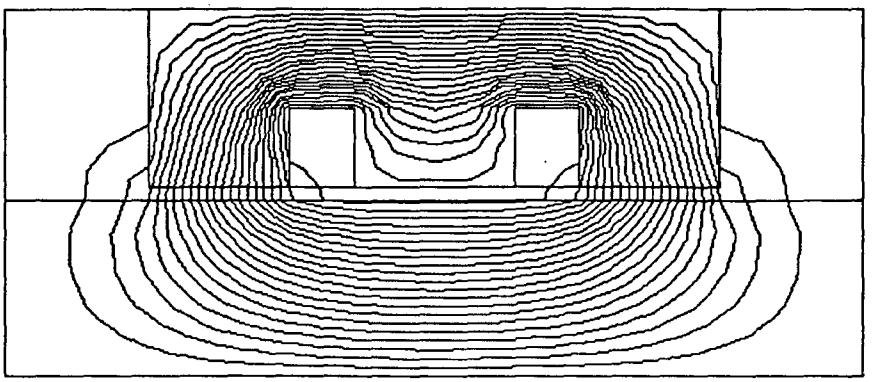

figura 3.12 - Linhas de Campo para o caso estático

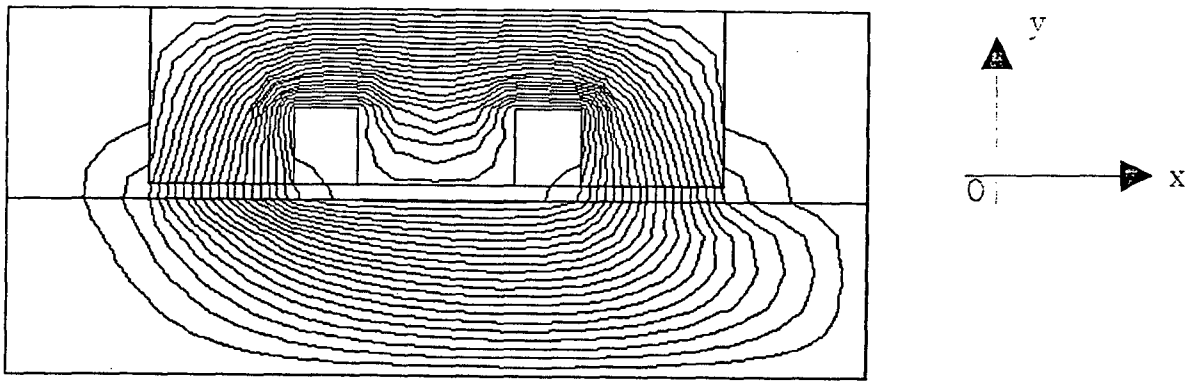

figura 3.13 - Linhas de Campo considerando a parte inferior da peça móvel

A terceira análise é feita considerando que a densidade de corrente possui uma forma senoidal com amplitude de 2 A/mm ^{2}. Foram calculadas as linhas de campo para uma freqüência de 500 e 10000 Hertz usando o módulo EFCC. Este módulo executa cálculos lineares para regime permanente senoidal. O campo obtido para 500 Hertz está mostrado na figura 3.14. Há uma pequena variação em relação ao caso estático. As linhas de campo já não penetram totalmente na peça. Este efeito é causado pelas correntes induzidas na peça que impedem que o fluxo penetre-a totalmente. A figura 3.14 está mostrada as linhas de campo para a freqüência de 10000 Hertz. Pode-se notar que a as linhas de fluxo possuem uma penetração muito menor que no primeiro caso devido ao elevado valor da freqüência.

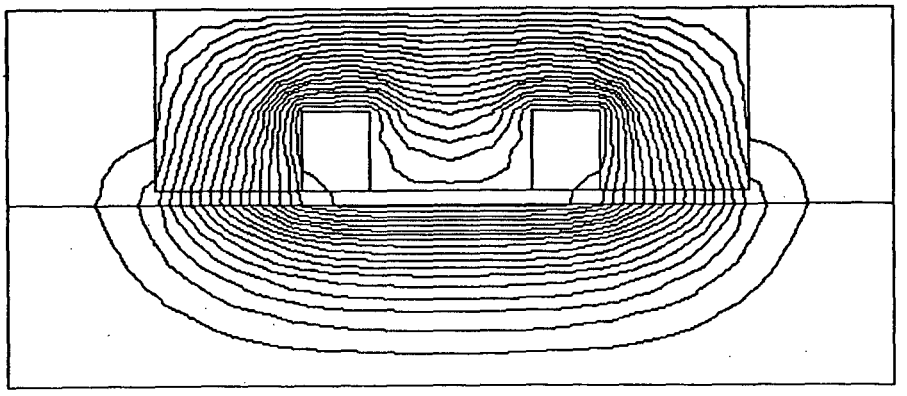

figura 3.14 - Linhas de Campo para a freqüência de 500 Hertz

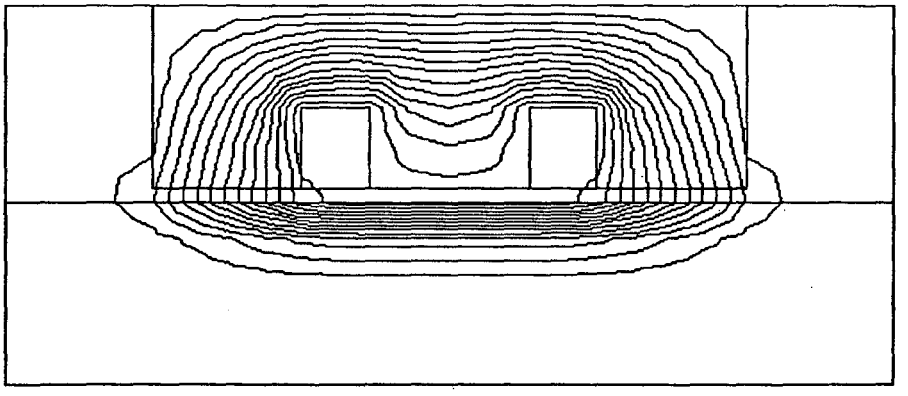

figura 3.15 - Linhas de Campo para a freqüência de 10000 Hertz

Finalmente é feita uma análise considerando que o degrau de tensão mostrado na figura 3.16 é aplicado na bobina. O objetivo é determinar a corrente induzida na peça e as linhas de fluxo em diversos instantes de tempo. Para este caso foi utilizado o programa EFCT e considerando o caso não linear. Este programa faz um cálculo passo a passo no tempo e grava os valores para os instantes escolhidos. Foram feitos 50 passos no tempo para este caso. Os dados da bobina para este caso são os seguintes:

- tempo inicial : 0
- tempo final : 1.0 x 10^{-2} s
- resistência do circuito externo : 10 Ohms
- indutância do circuito externo : 1.0 mH
- profundidade da peça : 10 cm.

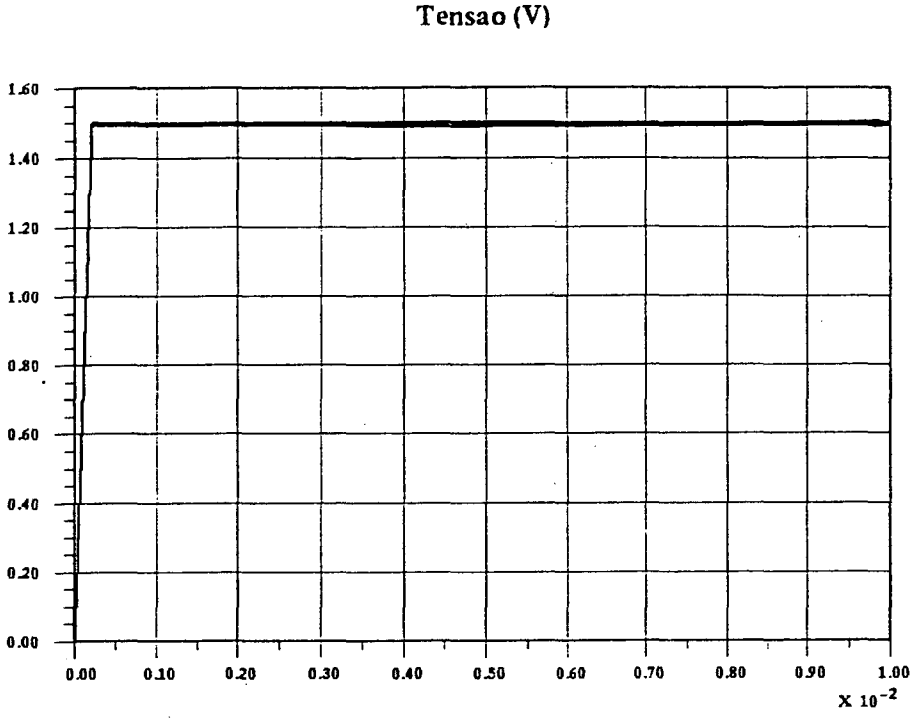

figura 3.16 - Curva de tensão imposta à bobina

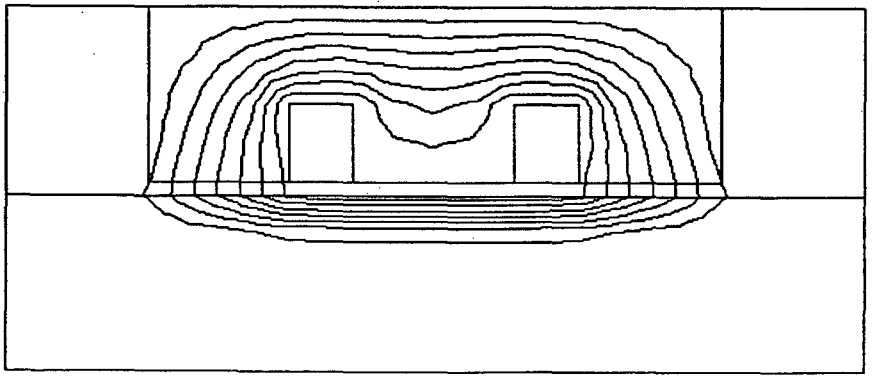

figura 3.17 - Linhas eqüipotenciais no instante 0.2 x 10^{-3}

A figura 3.17 mostra as linhas de fluxo para o instante de 0.2 x 10^{-3} s após a aplicação da tensão. Pode-se ver que o fluxo ainda não penetrou totalmente na peça. Na figura 3.18 está mostrado as linhas de campo para o instante de tempo de 0.14 x 10^{-2} segundos após a aplicação da tensão. Neste instante o fluxo já penetra quase que totalmente na peça.

A figura 3.19 mostra a curva de corrente obtida na bobina. Pode-se notar que a corrente tende para o valor de regime permanente de 0.15 A. Este valor está de acordo com as equações de circuito. Com o passar do tempo o efeito da indutância tende a desaparecer e o valor da corrente passa a ser limitado apenas pela resistência da bobina. Como tem-se uma tensão de regime de 1.5 V e uma resistência de 10 Ohms, o valor de regime da corrente será de 0.15 A.

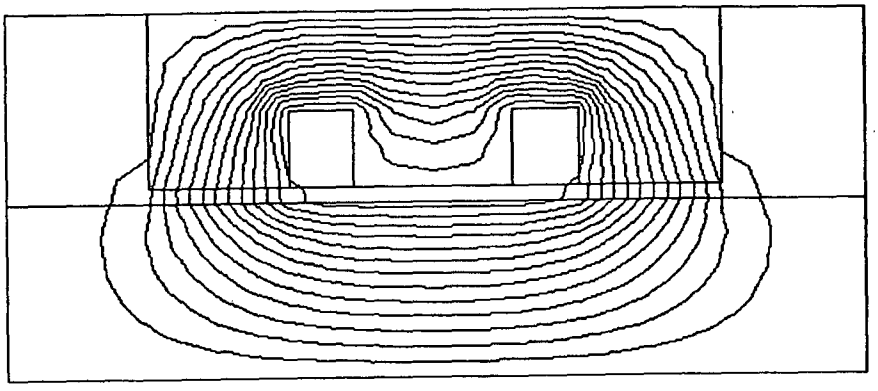

figura 3.18 - Linhas eqüipotenciais no instante 0.14 x 10^{-2} s

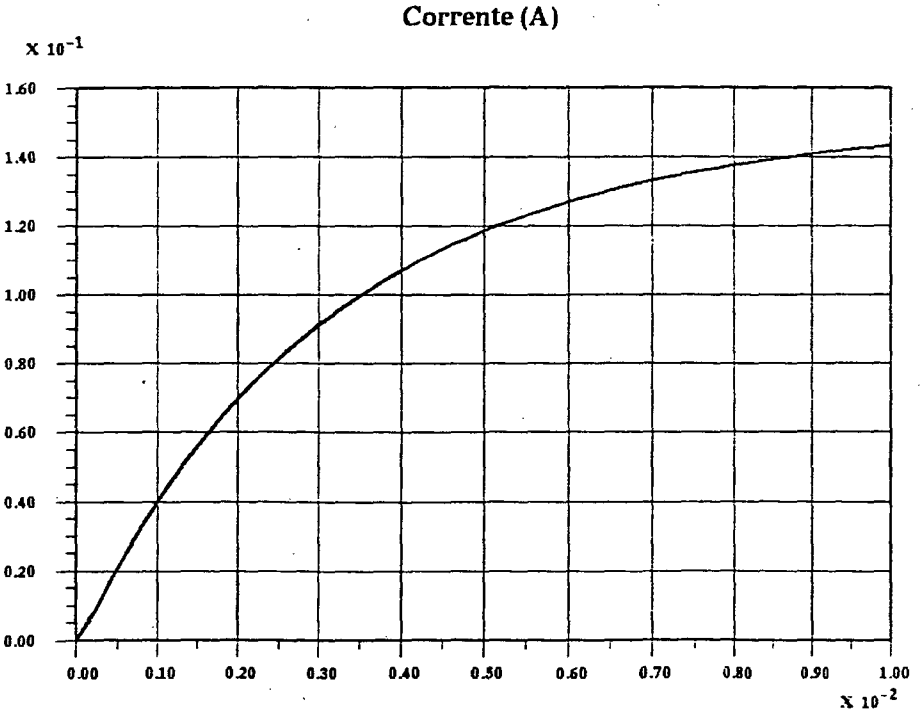

figura 3.19 - Curva de corrente calculada para a bobina

3.5.1 Conclusão

Pelos resultados obtidos com os exemplos 1 2 e 3 pode-se notar que o sistema desenvolvido possui uma performance em termos de cálculos bem acima do sistema para PC's. Note-se que entre o PC de maior performance (PC 486) e a estação de mais baixa performance (Sun LCD) ainda há uma diferença da ordem de 3 vezes.

Devido à alta capacidade de memória de uma estação, pode-se analisar estruturas que não poderiam ser analisadas em PC's. Mesmo para uma estrutura como a do exemplo 3 a performance do sistema ainda é muito alta.

O sistema desenvolvido para estações é, sobretudo, fácil de ser operado, fazendo com que as etapas de definição do problema e exploração gráfica sejam bastante rápidas. É possível manter mais de um módulo na tela ao mesmo tempo o que também facilita a redefiniçao do problema após a análise dos resultados. O exemplo 4 mostra também a flexibilidade do sistema na análise de vários tipos de estruturas e situações.

4. Conclusões

O sistema implantado demonstra ser fácil de usar e pode ser usado em áreas como projeto de dispositivos, pesquisa e ensino. Na sua forma final, o sistema está totalmente integrado ao sistema de janelas da estação e desta forma o usuário pode usá-lo da mesma forma que um software aplicativo, usufruindo de todas as vantagens que a interface gráfica do sistema oferece.

O módulo EFD, desenvolvido totalmente em C provou ser muito eficiente na definição de estruturas mais complexas, como máquinas elétricas. Além disso, o desenvolvimento da entrada de dados em linguagem C constiui um passo importante no desenvolvimento de sistemas mais modernos e utilização de recursos mais avançados de programação.

Os resultados em termos de velocidade dos módulos de cálculo e memória comprovam que o sistema é uma ferramenta muito eficiente para a análise de estruturas com elevado número de elementos, até então impossíveis de serem analisadas em computadores pessoais. Além disso, pode-se obter resultados mais precisos do que o sistema para PC's.

O módulo de pós-processamento permite uma exploração numérica e gráfica mais eficiente, pois reune ambas as funções num único módulo. Pode-se obter uma boa visualização dos resultados e também obter valores numéricos de grandezas em vários pontos da estrutura. Um dos fatores a contribuir para isso é o maior número de pontos da tela da estação. Este módulo pode ser também estendido para incluir outras possibilidades, como a simulação do movimento de partes da estrutura a partir da geração de vários arquivos. Com estes arquivos pode ser criado uma espécie de animação gráfica no computador. Este tipo de exploração gráfica é muito útil em áreas de ensino.

O desenvolvimento do sistema EFCAD para estações de trabalho da Sun e ambiente Unix, dentro da filosofia adotada, exigiu um trabalho de programação bastante grande para a sua implantação, visto que foi necessário um estudo não só da linguagem C e Fortran, mas também um estudo do interfaceamento de ambas, além do sistema Unix e vários outros aplicativos. A programação da parte gráfica exigiu um estudo do sistema gráfico X-Window e do XView Toolkit. Sendo o sistema X-Window muito mais complexo que o outros sistemas gráficos (como o

GKS) esta parte demandou um tempo bastante grande. Para o desenvolvimento de cada módulo foi necessário um estudo detalhado do funcionamento de cada um dos módulos do sistema para PC's.

O desenvolvimento da interface gráfica e utilização da biblioteca do sistema gráfico X-Window (Xlib) exigiu uma estruturação do sistema de uma forma diferente que a empregada em PC's, a fim de adaptá-lo ao sistema de janelas. Para o desenvolvimento da interface gráfica foi utilizado o XView Toolkit, o que também exigiu um estudo de como utilizar as suas rotinas e bibliotecas. Uma vez que a interface gráfica é um parte independente das rotinas matemáticas, pode-se substituí-la por outra de forma relativamente fácil, quando se desejar portar o sistema para máquinas que não usem o padrão Open Look.

Pode-se afirmar que o sistema implementado também será útil no desenvolvimento de sistemas semelhantes futuramente.

Finalmente, o sistema de interfaces gráficas desenvolvido pode ser facilmente adaptado para outros softwares de cálculo de campos, como por exemplo um sistema para cálculo em 3 dimensões.

5. Bibliografia

[1] Bastos, João Pedro Assumpção. Eletromagnetismo e Cálculo de Campos Eletromagnéticos. Editora da UFSC, 1989. 452 p.

[2] Sabonnadière, Jean- Claude e Coulomb, Jean-Louis. Elément Finis et CAO. Editora Hermes, 1986. 210 p.

[3] Schildt, Herbert. C Completo e Total. Editora McGraw-Hill, 1990. 889 p.

[4] Heller, Dan. XView Programming Manual. Editora O'Reilly & Associates, 1990. 642 p.

[5] Nye, Adrian. Xlib Programming Manual. Editora O'Reilly & Associates, 1990. 634 p.

[6] Xlib Reference Manual. Editora O'Reilly & Associates, 1990. 765 p.

[7] SunView Programmer's Guide. Sun Microsystems, 1990. 457 p.

[8] Programer's Languages Guides. Sun Microsystems, 1990. 93 p.

[9] Magalhães, Léo Pini. Computação Gráfica. Editora da Unicamp, 1985. 196 p.

[10] Rifflet, Jean-Marie. La Programmation sous Unix. Editora McGraw-Hill. 360 p.

[11] Cox, Brad J. Programação Orientada para Objeto. Editora Makron Books. 347 p.

[12] Hoole, S. Ratnajeevan H. Computer Aided Analysis and Design of Electromagnetic Devices. Editora Elsevier, 1989. 498 p.

[13] Zienkiewcz, O. C. The Finite Element Method in Engineering Science. Editora McGraw-Hill. 5 p.

[14] Norrie, Douglas H. e de Vries, Gerard. The Finite Element Method. Editora Academic Press, 322 p.

[15] Becker, Eric B., Carey, Graham F. e Oden, J. Tinsley. Finite Elements - An Introduction. Editora Prentice Hall. 300 p. v. 1.

[16] Carey, Graham F. e Oden, J. Tinsley. Finite Elements - Computational Aspects Editora Prentice Hall. 230 p, v. 3.

[17] Scheifler, R.W. e Gettys, J. The X-Window System. Software-Practice and Experience. v. 20, issue s2, p. 5-34. Outubro de 1990.

[18] Gettys, J., Karlton, P. L., McGregor, S. The X-Window System, Version. 11. Software-Practice and Experience, v. 20, issue s2, p. 35-68. Outubro de 1990.

[19] Widener, G. The X11 Inter-Client Comunications. Software-Practice and Experience, v. 20, issue s2, p. 109-118. Editora Butterworth-Heinemann. Outubro de 1990.

[20] Duce, D. A., Hopgood, F. R. A. The Graphical Kernel System. Computer Aided Design, v. 19, n. 8, p. 396-409. Editora Butterworth-Heinemann. Outubro de 1987.

[21] Shuey, D. PHIGS: a Graphical Plataform for CAD. Computer Aided Design, v. 19, n. 8, p. 410-418. Editora Butterworth-Heinemann. Outubro de 1987.

[22] Arnold, D. Computer Graphics Interface and CAD Applications. Computer Aided Design, v. 19, n. 8, p. 444-450. Editora Butterworth-Heinemann. Outubro de 1987.

[23] Vecchiet, K. S. Computer Graphics Interface: a Developper's Perspective. Computer Aided Design, v. 19, n. 8, p. 451-455. Editora Butterworth-Heinemann. Outobro de 1987.

[24] Sparks, M. R. Looking at GKS. Computer Aided Design, v. 21, n. 4, p. 254-255. Editora Butterworth-Heinemann. Maio de 1989.

[25] Howard, T. L. J. Evaluating PHIGS for CAD and General Applications. Computer Aided Design, v. 23, n. 4, p. 244-251. Editora Butterworth-Heinemann. Maio de 1991.

[26] Brodlie, K. W. , Duce, D. A. , Hopgood, F. R. A. The New Graphical Kernel System. Computer Aided Design, v. 23, n. 4, p. 312-318. Editora Butterworth-Heinemann. Maio de 1991.

[27] Perry, Tekla S., Voelcker John. The User-fdriendly Interface. IEEE - Spectrum, v. 26 n. 19. Setembro de 1989.

[28] Rosenblatt, Alfred. PC's and Workstations.IEEE-Spectrum, v. 28 n. 1. Janeiro de 1991.

[29] Andersen, David M., Sherwood, Bruce A. Portability and GUI. Revista BYTE. Novembro de 1991.

[30] Sheldon, Kennet M., Barron, Janet J. e Smith, Ben. Windows Wars. Revista BYTE. Junho de 1991.

[31] Kaplan, Gadi. A Special Guide do Engineering Workstations. IEEE-Spectrum, v. 28, n. 4. Abril de 1991.

[32] Savalle, Didier. Contribuition à la Conception et à la Realisation d'un Processeuer d'Éxploitation et de Visualisation des Résultats d'un Calcul de Champs Electromagnétiques Tridimensionnels. Tese de Doutoramento, Institut National Polytechnique de Grenoble, 1984.

[33] del Aguila, Orieta. Analyse et Structuration de Données dans les Logiciels de CAO en Electromagnetisme. Tese de Doutoramento, Institut National Polytechnique de Grenoble, 1984.

[34] MS-DOS 5.0 - User's Guide and Reference. Gateway Edition - 1991, 668 p.

[35] Bastos, João Pedro Assumpção. EFCAD - Manual de Utilização. UFSC/GRUCAD, 1992. 56 p.

[36] Ryan, Bob. The Sucession Crisis. Revista BYTE, p. 199-202. Março de 1990.

[37] Dougherty, Dale, O'Reilly, Tim. DOS Meets Unix. Revista BYTE - IBM Special Edition. p. 117-126. Setembro de 1988.

Apêndice A
Equipamentos, Sistemas Operacionais e Padrões Gráficos

i. Introdução

Neste apêndice serão apresentados e discutidos os diversos aspectos que envolvem o equipamento e sistema operacional no qual o sistema de cálculo de campos será implantado. A discussão se limitará aos computadores pessoais e estações de trabalho. Também serão abordados os principais aspectos ligados ao padrão gráfico X-Window [5,6,17].

Finalmente, serão apresentadas e discutidas as características mais importantes da linguagem C e da linguagem Fortran. Estas são as principais linguagens usadas em sistemas de cálculo de campos. Ambas são linguagens portáveis e fazem com que os sistemas possam ser implantados em várias máquinas. Uma vez que ambas possuem vantagens e desvantagens, muitas vezes pode ser útil utilizar ambas as linguagens dentro de um mesmo sistema, visando aproveitar o que de melhor existe em cada uma delas. São apresentados diversas considerações de ordem prática que são relevantes para o uso conjunto das duas linguagens. Na última parte também são apresentados alguns exemplos de como implementar uma interface entre C e Fortran.

ii. Equipamentos Utilizados em Sistemas de Cálculo de Campos

Nos últimos 10 anos o computador que mais tem se difundido entre os usuários em todo o mundo foram os PC's. Eles são hoje usados em praticamente todos os ambientes e aplicações.

No início dos anos 80 foi lançado um novo equipamento visando diminuir a grande diferença entre os PC's e os minicomputadores. Este equipamento, que possuía um sistema operacional mais poderoso que o do PC e de alta capacidade gráfica, foi então chamado de Estação de Trabalho (Workstation). Devido a dificuldades le-

gais, entre outras, este tipo de equipamento começou a ser introduzido no Brasil apenas no início dos anos 90.

A seguir será feito uma descrição das principais características dos PC's e estações de trabalho, uma vez que são os principais equipamentos utilizados para o desenvolvimento deste tipo de software.

1.1.1 Computadores Pessoais

Este tipo de computador, também chamado de microcomputador, foi lançado em 1981 pela IBM e se tornou rapidamente um padrão, sendo hoje produzido por centenas de fabricantes em todo o mundo, inclusive no Brasil. Foram eles os primeiros computadores acessíveis a engenheiros individualmente, aliviando-os de tarefa menos criativas. Os PC's foram concebidos para trabalhar com o sistema operacional DOS (Disc Operational System). Sua arquitetura original possuía um microprocessador de 16 bits do tipo 8086, uma evolução do 8085. Devido ao sucesso junto aos consumidores ele teve um avanço contínuo e, desta forma, passou-se a usar processadores mais rápidos e também ao uso de co-processadores. Posteriormente, surgiu os processadores 80286, passando esta nova geração a chamar-se PC-AT, em contraste com a anterior chamada de PC-XT. A seguir, foram lançados os PC's com processadores do tipo 80386 e 80486 com processadores de 32 bits e freqüência operação de 16 até 33 MHz, fabricados pela Intel e Motorola. Atualmente os PC's conseguem resolver problemas de engenharia, como programas de elementos finitos, em espaço de tempo relativamente curto.

O microprocessador 80486 é mais evolucionário que revolucionário. Ele combina um processador, um co-processador matemático e uma memória cache de 8 kbytes em um único chip. A memória cache é uma abreviação para um área de rápido acesso para o processador. A memória cache também pode ser subdividida entre uma área de dados e uma área contendo instruções. As futuras gerações de processadores provavelmente deverão continuar nesta tendência de usar vários processadores e memória cache em um único chip.

Os computadores pessoais são fornecidos nas mais diversas configurações possíveis, ficando difícil definir o que seja uma configuração básica para um PC. Nos sistemas de bom desempenho eles possuem:

- disco rígido em torno 80 Mbytes (ou mais) de capacidade;
- sistema operacional DOS, Windows, ou OS/2;

- vídeo do tipo VGA (Vídeo Graphics Array) de 640 x 350 ou Super VGA de 1024 x 768;
- CPU com processadores do tipo 80286, 80386 ou ainda 80486;
- memória RAM de 1 Mbytes, ou mais.

1.1.2 Estações de Trabalho

A uma primeira vista os modelos mais recentes de PC são bem comparáveis a uma estação de trabalho. No entanto, esta comparação esconde algumas características importantes tais como o sistema operacional, facilidade de conexão em rede e velocidade de realizar cálculos [15].

No ínicio dos anos 80 técnicos, engenheiros e profissionais da computação concluíram que os PC's não eram ideais para certas aplicações devido as limitações do mesmo; por outro lado, o alto custo de um minicomputador tornava o seu uso difícil. Os microcomputadores não tinham a devida capacidade de memória, velocidade de processamento e resolução gráfica exigidas para tratar certos problemas complexos em engenharia. Minicomputadores ofereciam estas características mas eram caros e não tinham a independência e portabilidade oferecidas por um computador pessoal.

Em 1981 e 1982 a Apollo e Sun Microsystems lançaram as suas estações de propósitos gerais Domain DN 100 e Sun 1, com o objetivo de preencher esta lacuna. Destinados a engenheiros, estes sistemas tinham CPU (Central Processor Unit) de alta velocidade, grande capacidade de memória RAM, vídeo de alta resolução, capacidade de conexão em rede compartilhando memória e periféricos.

Desde 1981 muitas outras companhias também passaram a fabricar estações existindo no mercado atual dezenas de marcas.

A configuração básica das estações são muito similares nas suas características, embora possam ser fornecidas pelos mais diversos fabricantes. A configuração básica atualmente possui as seguintes características:

- CPU de 32 bits;
- um processador de ponto flutuante (FPP);
- 8 a 16 Mbytes de memória RAM (Randomic Access Memory), mais uma área de swap. Em algumas estações a memória RAM pode ser estendida para até 128 Mbytes;
- memória cache em torno de 64 a 320 kbytes (incluindo dados e instruções);

- vídeo de alta resolução (geralmente 1152x900 ou 1024x1280 pontos);
- disco rígido interno de 200 a 600 MBytes de capacidade;
- sistema operacional Unix;
- sistema de janelas baseado no padrão X-Window;
- interface gráfica de acordo com o padrão Open Look[1] ou Motif[2];
- freqüência de operação de 20 a 66 MHz;
- 15 a 75 MIPS (Million of Instruction per Second);
- 4 a 9 MFLOPS (Millions of Floating Operations per Second).

Quase todas as estações mais recentes são construídas com a família de processadores da Motorola 68000. Algumas poucas usam processadores baseados no 80386. Atualmente há uma tendência para que os fabricantes utilizem chips proprietários em suas máquinas. Eles são construído dentro dos princípios da arquitetura RISC (Reduced Instruction Set Computer), ao contrário dos PC's que são concebidos dentro da arquitetura CISC (Complex Instruction Set Computer).

A maioria das estações podem trabalhar em rede e acessar um servidor de arquivos.

Enquanto que os computadores pessoais geralmente medem a performance da sua CPU em termos da freqüência de operação da máquina, as estações geralmente são medidas em MIPs (Million of Instruction per Second) que é o número de operações que a máquina pode realizar por segundo. Os primeiros sistemas ofereciam em torno de 1 MIP. Atualmente este limite está próximo de 80 MIPS.

Um fator que tem sempre diferenciado as estações dos PC's é a sua alta velocidade em operações em ponto flutuante. A maioria tem hardware especial para aumento da velocidade em operações em ponto flutuante. Este tipo de cálculo é muito usado em manipulações gráficas e em cálculos científicos especialmente na solução de equações. Algumas máquinas possuem um chip co-processador, também chamado de Processador de Ponto Flutuante (FPP - Floating Point Processor) que trabalha junto com a CPU. Outras máquinas mais caras muitas vezes, além de um Processador de Ponto Flutuante, possuem também outros chips destinados a dar maior velocidade às operações gráficas.

Uma das características mais marcantes de uma estação é o seu vídeo, geralmente maior que o de um PC. As estações mais baratas oferecem um vídeo de 15 polegadas, todavia o vídeo de 19 polegadas tornou-se um padrão. Este tamanho de

[1] Padrão de interface gráfica adotado pela Unix International.
[2] Padrão de interface gráfica adotado pela Open Software Foundation.

tela permite uso prático de várias janelas simultaneamente. Para tornar isto possível o número de pontos na tela dever ser da ordem de milhões. Por exemplo, a Sun oferece um vídeo de 1152 pontos na horizontal por 900 pontos na vertical, a HP Apollo, DEC e IBM oferecem um monitor de 1280 por 1024. Todavia, os vídeos oferecidos para PC's atualmente quase atingem estes patamares. Por exemplo, os vídeos do tipo VGA oferecem 640 por 480 pontos, enquanto que os Super VGA podem mesmo ir além destes limites. Embora as estações mais baratas tenham vídeo monocromáticos, todas as marcas também são disponíveis com vídeo colorido, possuindo normalmente 4,6, ou 8 planos[3]. As estações mais avançadas podem mesmo possuir vídeo com 24 planos para cores.

A característica mais uniforme das estações de trabalho atualmente disponíveis é o sistema operacional empregado. Praticamente todas empregam alguma versão do sistema operacional Unix. Há pouca discussão entre os fabricantes quanto ao sistema operacional a ser usado; todavia, há discordância quanto à versão a ser utilizada. Os fabricantes nos últimos anos se agruparam em torno de duas associações a OSF (Open Software Foundation) e a UI (Unix International). A OSF é formada pela IBM, HP, Digital, entre outros; a UI é formada principalmente pela Sun e pela AT&T. Ambas as associações de fabricantes adotaram uma versão do Unix e um padrão de Interface Gráfica[4]. Embora a aparência e funcionamento das interfaces gráficas não sejam idênticas todas elas são elaboradas dentro do padrão X-Window. Este padrão, elaborado no MIT (Massachussett's Institut Of Technology) e aprovado em 1989, é adotado atualmente pela grande maioria dos fabricantes de estações.

Muitas fabricantes também fornecem máquinas dedicadas a servirem como o nó central de uma rede chamadas de servidoras. Estas máquinas têm maior capacidade de memória e de disco que as normais.

iii. Sistemas Operacionais

Quando se pensa em desenvolver e implantar um sistema de Cálculo de Campos em um sistema computacional a escolha do sistema operacional é o ponto ini-

[3] O número de planos está diretamente ligado ao número de cores que podem ser usadas simultaneamente numa aplicação. O número de cores que podem ser usadas simultaneamente é dado por 2^n, onde n é o número de planos do vídeo. Este número também é o número de bits usados para endereçar cada ponto da tela.

[4] É uma forma gráfica de interação com o usuário implantada através de janelas onde são mostrados objetos gráficos tais como menus, botões e ícones; este padrão contrasta com as interfaces por meio de linhas de comando. O padrão define como estes objetos são manipulados sobre a tela pelo usuário.

cial e de vital importância. Alguns aspectos deveriam sempre ser considerados nesta etapa:

- o sistema deve ser adaptado tanto às necessidades do usuário final quanto às necessidades dos programadores;
- o sistema deve ser capaz de atender vários tipos de usuários;
- capacidade de comunicação entre os diversos processos;
- capacidade de comunicação com outras máquinas, a fim de que o usuário possa trabalhar com o sistema em máquinas diferentes;
- o sistema deve ser portável;
- deve haver número suficiente de aplicativos para o sistema;
- deve haver um conjunto de utilitários que permite um trabalho de programação eficiente;
- o sistema deve ser de fácil operação pelo usuário.

Algumas das características acima são conflitantes, motivo pelo qual dificilmente um único sistema conseguirá reunir todas elas de uma só vez.

Os programadores produzem mais quando possuem boas ferramentas de programação, isto inclui compiladores eficientes, auxílios (help) de fácil acesso, editores de texto fáceis de manusear, utilitários e um ambiente de programação capaz de integrá-los.

Por outro lado, os usuários finais têm necessidades diferentes. Eles devem ficar isolados da parafernália do computador e a eles devem ser dados ferramentas fáceis de serem usadas e aprendidas. O usuário final precisa obter resultados com o computador e não aprender sobre ele.

A grande evolução experimentada na tecnologia dos computadores, aliados aos recursos gráficos hoje disponíveis, permite aos programadores tornar o ambiente de trabalho muito mais agradável não só ao usuário final como também aos programadores.

a. Sistema Operacional DOS [34]

O sistema operacional DOS (Disc Operational System) foi concebido para trabalhar em computadores pessoais. O DOS é um sistema monousuário e monotarefa[29].

Este sistema é largamente utilizado em todo mundo e a sua grande utilização também está ligada a grande utilização dos computadores pessoais. Atualmente ele é um sistema maduro e que possui um volume muito grande de software já desenvolvido para ele, inclusive sistemas de CAD. Ao longo dos anos ele tem tido tido consideráveis melhoras. Ele não possui tantos utilitários específicos para programação, como no caso do Unix, mas mesmo assim a programação dentro deste ambiente é fácil.

Uma característica notável do DOS é que ele assegura grande compatibilidade entre várias máquinas em nível de código binário. Desta forma os programadores possuem portabilidade entre os programas executáveis. Isto se deve ao fato de que todos os sistema que empregam o DOS rodam em máquinas cuja arquitetura é semelhante, ou também chamadas máquinas homogêneas.

Embora ele não tenha sido concebido para trabalhar em rede, é possível que o mesmo opere no ambiente de uma rede.

Possivelmente a maior dificuldade com este sistema é seu limite de memória. Ele é capaz de endereçar até 640 kbytes de memória. Atualmente há possibilidade de se estender este limite por meio de programas especiais, chamados coletivamente de DOS Extenders.

A última versão do DOS (versão 5.0) permite que se utilize de forma um pouco melhor a memória instalada na maioria dos computadores. A partir de um rearranjo do mapa de memória, pode-se usar em torno de 640 kbytes para a execução de programas.

Com o DOS cada dispositivo de entrada e saída, placas gráficas, plotters, impressoras, mouse, etc..., necessita de um driver específico para cada programa e para dispositivo.

b. Unix [10]

O sistema operacional Unix foi projetado por volta de 1973 nos laboratórios da AT&T[5] para ser uma base para desenvolvimento de programas e para trabalhar em redes. Originalmente ele foi concebido para trabalhar em minicomputadores mas, devido a sua grande aceitação, passou rapidamente tanto para máquinas maiores como para menores. Dentro da idéia da sua criação, foram desenvolvidos uma sé-

[5] Abreviatura da companhia americana American Telegraph and Telephonic.

rie de programas utilitários que visavam facilitar o trabalho de programação e manutenção de programas. Todavia, para o usuário final, na sua concepção original, ele possuía algumas dificuldades e era considerado como uma base pobre até alguns anos atrás. Com o advento de interfaces gráficas esta dificuldade foi amenizada [22,23].

Sendo o sistema adotado pela quase totalidade dos fabricantes de estações de trabalho e tendo capacidade muito superior aos sistemas para PC, será feita uma descrição das principais características deste sistema nos próximos itens.

b.1 Dispositivos de Entrada e Saída

O sistema de entrada e saída do Unix é muito fácil de ser entendido. Cada dispositivo físico suportado pelo sistema aparece como um arquivo, listado no diretório /dev. Os programas e usuários rodando no Unix tratam os dispositivos como se fossem realmente arquivos. Assim, cada dispositivo pode ser acessado simplesmente acessando-se o arquivo a ele associado, inclusive a memória. Por exemplo, para imprimir, ou enviar caracteres para a impressora, ou usuário simplesmente escreve no arquivo /dev/lp, para trocar um byte na memória escreve-se no arquivo /dev/mem. Contudo, do ponto de vista do sistema operacional, os dispositivos físicos permanecem como tal e necessitam de drivers[6] para o seu manuseio. Estes drivers estão na forma de arquivos especiais.

b.2 Segurança no sistema Unix

Naturalmente que algum controle precisa ser feito sobre os dispositivos e arquivos a fim de que os dispositivos possam ser acionados adequadamente. Além disso, o usuário pode desejar que outros usuários não acessem e alterem os seus arquivos. O usuário controla este aspecto através de uma forma simples e eficiente. Cada arquivo tem um grupo de bits (também conhecidos como mode bits) que servem para configurar a proteção do arquivo, os quais são estabelecidos pelo usuário. Desta forma, o usuário estabelece de que forma os outros usuários poderão acessar um determinado arquivo. Há uma infinidade de combinações possíveis; por exemplo, pode ser estabelecido que os demais usuários só terão permissão para ler arquivos

[6] Um driver é um arquivo especial que serve para acionar um dispositivo físico. Ele é específico para cada dispositivo.

e diretórios. O administrador do sistema, no entanto, pode passar por cima destas permissões e inclusive alterá-las.

b.3 Redirecionamento da Entrada e Saída;

Normalmente as entradas e saídas dos comandos e programas são feitos pelos dispositivos padrão, que são o teclado para entrada e o vídeo para a saída. No entanto, como estes não são os únicos dispositivos que o Unix reconhece, as entradas e saídas podem ser redirecionadas para outros dispositivos válidos e reconhecidos pelo sistema. Por exemplo, o comando

```
% ls > /dev/lp
```

redireciona a saída do comando ls para a impressora[7].

Para redirecionar a entrada para algum arquivo pode-se fazer da seguinte forma

```
% spell < arqtex
```

Com este comando a entrada para o comando spell será tomada do arquivo arqtex.

b.4 Pipes

O Unix tem um mecanismo que permite encadear um comando no outro. Este mecanismo de encadeamento é conhecido como ***piping*** . Com este mecanismo pode-se criar um novo comando compondo-se vários comandos simples. O comando assim composto será chamado de pipe.

Por exemplo o comando composto;

```
% pr *.c ¦ lpr
```

fará com que sejam formatados todos os arquivos com extensão *.c* e em seguida enviados para a impressora.

---

[7] O símbolo % será usado para designar o prompt do sistema.

A princípio poderia-se pensar que apenas está sendo feito um redirecionamento da entrada, ou que é uma maneira elegante de enviar a saída para um arquivo temporário, redirecionar a entrada de lpr para aquele arquivo, imprimir e apagar o arquivo temporário. Na verdade, no Unix, todos os arquivos em uma pipeline[8] rodam simultaneamente, ou seja os comandos mais à direita vão recebendo as entradas conforme eles vão sendo processados pelos comandos mais à esquerda. Este é um recurso muito usado e que pode também ser usado para entrada e saída de programas que rodam no Unix.

Assim, cada comando pode ser encarado como um bloco básico a partir do qual muitos comandos novos podem ser construídos. Há também a possibilidade de renomear os comandos usando um mecanismo chamado *alias*.

b.5 Multitarefas

A característica de rodar mais de um programa simultaneamente é uma característica importante do Unix. Isto permite ao usuário realizar mais do que uma tarefa ao mesmo tempo e no mesmo terminal. Este é um recurso que apenas recentemente os sistemas para PC oferecem.

A capacidade de rodar várias tarefas ao mesmo tempo está associada com o mecanismo de memória virtual[9]. Ao invés de usar endereços físicos os programas operam com um espaço de memória virtual, fazendo com que vários programas operem simultaneamente. O Unix trabalha dentro de um sistema de mapeamento de memória que permite transferir blocos de programa entre o disco e a memória física. Para o usuário, no entanto, esta operação é transparente. Cada programa, também chamado de processo, tem um número que o identifica chamado de PID [37].

b.6 Multiuser

Um sistema multiusuário significa que mais de um usuário pode usar os recursos de uma máquina ao mesmo tempo.

A tendência da maioria dos sistemas operacionais atuais é tornar-se multiusuário, devido também ao fato de que o preço de uma máquina pode ser dividido

[8] Uma pipeline é linha de comandos compostos.

[9] A memória virtual inclui a área física da memória (RAM) mais um espaço do disco que também é usado como memória.

em vários usuários. Esta alternativa é na maioria dos casos mais barata do que uma do tipo monousuário.

Uma outra vantagem de um sistema multiusuário é o compartilhamento de dados entre vários usuários. Para tanto o sistema usa um mecanismo de sistema de tempo compartilhado chamado de timesharing, no qual cada usuário usa a CPU por um certo tempo.

b.7 Interpretador de Comandos (Shell)

Muitas das características descritas até aqui, enquanto implementadas fisicamente nas camadas mais baixas do sistema, são controladas por um programa chamado shell, o qual o usuário na maioria das vezes interage. O shell é rodado cada vez que o usuário entra no sistema. Sendo apenas mais um programa, ele também pode ser mudado de acordo com a conveniência do usuário. Existe várias versões do shell, sendo que as mais usadas são Bourne-shell (nome do seu criador) e o C-shell.

O shell é o responsável pela interface entre o usuário e o sistema. Ele faz com que o prompt do sistema seja mostrado, aceita os comandos e faz com que sejam executados, interpreta os processos e metacaracteres (tais como < |, >, ¦, etc...). Além disso, ele permite ao usuário escrever os seus comandos na sua própria linguagem, ou seja guardar uma série de comandos pessoais e usá-los como outro comando qualquer do sistema. O shell também permite rodar os processos em background.

O shell é também uma linguagem de programação. Como tal, ele contém mecanismos semelhantes aos encontrados em várias outras linguagens, tais como mecanismo de controle de fluxo de operações do tipo for, if-else if e case, etc...

b.8 Programas Utilitários

A linguagem C e o sistema Unix oferecem um rico conjunto de utilitários de programação que tornam a programação mais eficiente. Todos estes utilitários tem uma estrutura de comandos comum da seguinte forma

% <nome do comando> <opções> <argumentos>

Entre outros, cita-se os seguintes utilitários:

- *SCCS* (Source Code Control System), permite que todas as trocas feitas nos arquivos fontes sejam armazenadas, de tal forma que versões anteriores dos arquivos podem ser facilmente recuperadas a qualquer momento, bem como mesclar versões;
- *diff* , permite verificar as diferenças entre arquivos;
- *cb* , permite fazer mudanças na forma dos programas fontes de forma a torná-los mais fáceis de entender;
- *find* , permite localizar um arquivo dentro do sistema;
- *adb* , debugger usado no desenvolvimento de programas.

b.9 Dificuldades Impostas

O Unix foi projetado para programadores ou profissionais de computação, uma comunidade que sempre prefere comandos breves e que exigem poucos toques de teclas. Por exemplo, para se copiar arquivos o usuário digita *cp* ao invés de *copy*. Esta tendência minimalista faz com que os iniciantes e usuários tenham alguma dificuldade com o apredizado do sistema. Na realidade, esta dificuldade é inerente a qualquer sistema que usa linhas de comando como interface. Esta dificuldade pode ser diminuída com a introdução de uma interface gráfica.

Uma outra dificuldade é o preço dos programas aplicativos. Embora eles sejam atualmente quase tão numerosos quantos os aplicativos para PC's eles são geralmente mais caros que os equivalentes para PC's. Um aplicativo para Unix custa em média 2 a 3 vezes mais que um para PC, podendo atingir até 10 vezes em alguns casos.

Outra dificuldade com o Unix é a inexistência de uma versão universal, existindo várias versões espalhadas pelo mundo. Uma das razões para isto é o fato do Unix ter originalmente surgido como um sistema experimental. Os primeiros usuários sentiram-se encorajados a fazer mudanças e melhoramentos nos códigos fontes originais. Uma vez que várias companhias adquiriram os seus códigos fontes e os adaptaram às suas necessidades e o referido caráter experimental do sistema, a sua padronização é algo de difícil solução atualmente. Além disso, como o Unix pode rodar nas mais variadas arquiteturas, não há qualquer portabilidade em nível de código binário. Os programas fontes precisam ser recompilados quando se muda de arquitetura. A portabilidade está assegurada, porém, em nível de código fonte. Por exemplo, uma aplicação escrita em linguagem C usando chamadas padrão ao sistema poderão ser recompilados e rodados em outra máquina, o mesmo sendo verdade para

os utilitários. Desta forma, os fabricantes de software são obrigados a terem uma versão dos seus aplicativos para cada arquitetura em que o Unix rode. Em muitos casos, é preferível manter aplicativos na forma binária, não só para diminuir a dificuldade de instalá-lo em várias máquinas, mas também como forma de controlar as várias versões do mesmo. Constata-se, todavia, que já existe esforços no sentido de criar uma compatibilidade em nível de código binário [36].

Conforme já foi dito, os fabricantes de estações de trabalho estão polarizados em duas versões. A UI (Unix International), liderada pela Sun e *AT&T*, adota a versão SVR4 (System V Release 4). A OSF; (Open Software Foundation), liderada pela IBM, adota a versão chamada de OSF/1.

iv. Sistema Gráfico

Os sistema de cálculo de campos em geral utilizam uma biblioteca gráfica para obter um melhor visualização dos resultados de cálculo e propiciar uma interação com o usuário mais eficiente. Esta biblioteca gráfica geralmente é construída de acordo com um padrão gráfico, a fim de assegurar-se a portabilidade do sistema. Os padrões especificam a forma de funcionamento e utilização de uma certa biblioteca gráfica. Desta forma, o usuário utiliza funções padronizadas para implementar a parte gráfica do sistema.

O objetivo do estabelecimento de padrões é evitar mudanças nos programas de aplicação quando se usam máquinas diferentes.

O principal padrão usado para PC's é o GKS (Graphical Kernel System). Em estações de trabalho o padrão mais adotado é o X-Window. Existe ainda o padrão PHIGS que pode ser adotado tanto em estações de trabalho como em PC's.

v. Sistema X-Window

O sistema X-Window começou a ser desenvolvido no *MIT* (Massachussett's Institut Of Technology) em 1986 para atender as necessidades do projeto *Athena* , uma rede de computadores espalhados pelo campus do tipo e conectados em rede. Esta primeira versão foi chamada de versão 10 e alcançou grande popularidade rapidamente. Em 1988 foi formada uma comissão, chamada *X-Consortium* , que contava com a participação dos maiores fabricantes de estações para reestudar o sistema

X-Window. Como resultado surgiu a versão 11, a qual foi aprovada em 1989 e adotada pela maioria dos fabricantes [4,5].

O sistema X-Window foi concebido dentro do modelo servidor-cliente, tornando-o extremamente portável. Desta forma os programas de aplicação podem rodar em qualquer máquina onde houver um servidor do tipo X-Window (X-Server) [5,6].

Este sistema também é um sistema atualmente de domínio público, cujos códigos fontes são fornecidos pelo *MIT* [5,6].

Embora o padrão X-Window possua funções bastante poderosas, ele não possui até o momento funções que trabalhem diretamente em 3 dimensões, ficando a cargo do programa de aplicação desenvolvê-las a partir daquelas fornecidas para 2 dimensões.

O sistema X-Window é um sistema desenvolvido para monitores de vídeo gráficos do tipo bitmap. Num monitor deste tipo cada ponto da tela (pixel) é acessado por meio de um endereço. Todavia, este sistema também pode ser usado para monitores monocromáticos.

Neste sistema um display é definido como uma estação, consistindo de um teclado e um dispositivo que permite que sejam apontados objetos na tela, tal como um mouse, e uma ou mais telas.

1.1.1 Organização do Sistema X-Window

A figura A.1 mostra o esquema básico da arquitetura do sistema X-Window bem como a sua relação com as demais camadas de software. Pela figura A.1 pode-se notar que a XLib é o nível mais baixo na hierarquia de possíveis bibliotecas com as quais o programador pode escolher para implementar uma interface. A maioria dos programas no entanto usam bibliotecas de mais alto nível [4,5,6].

Os programas de aplicação se comunicam com o servidor por meio de chamadas a uma biblioteca de rotinas de alto nível em linguagem C, conhecida como XLib. A XLib tem rotinas para conexão a um particular servidor, criação de janelas, desenho de gráficos, resposta a eventos, etc... As chamadas a XLib são transcritas para pedidos na forma de um protocolo (protocol request) que são então passados para o servidor local, ou para algum outro servidor através da rede.

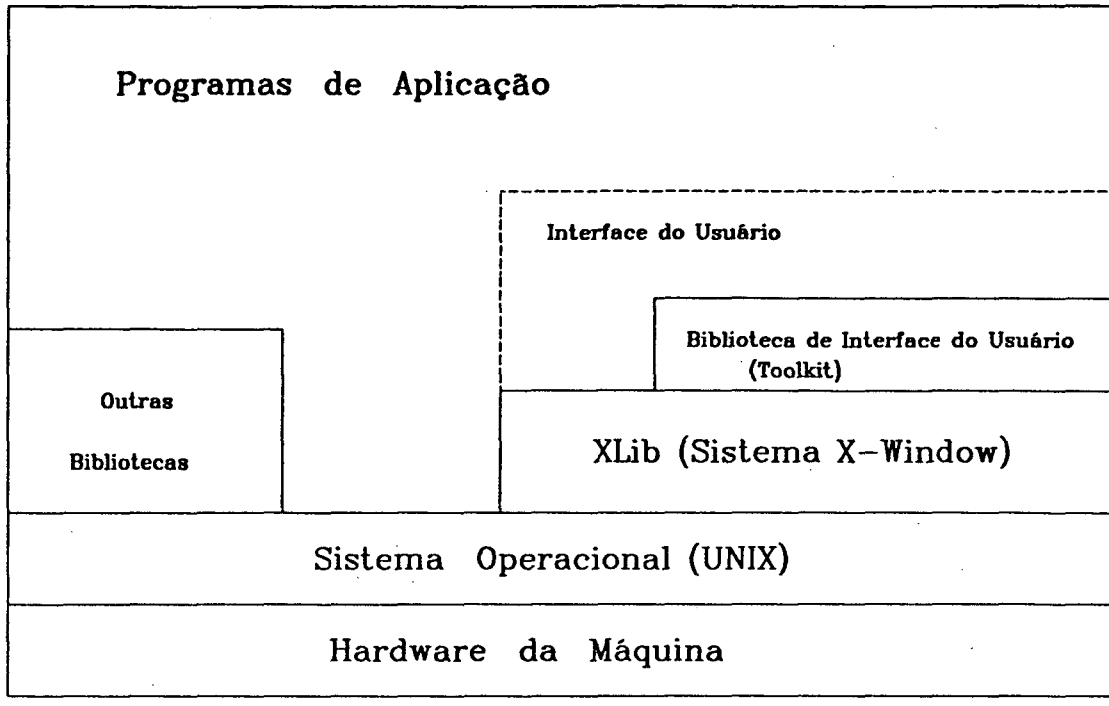

figura A.1 - Organização do Sistema X-Window

Os programas aplicativos e gerenciadores de janelas podem ser escritos somente com a XLib ou com um conjunto de rotinas de alto nível chamado de Toolkit. Os Toolkits fornecem os elementos básicos para se construir uma interface gráfica tais como buttons, menus, panels. Estes elementos podem ser manipulados pelos programadores usando técnicas de programação orientada para objetos [4,28].

Os Toolkits tornam a programação mais fácil e o acabamento final melhor. Eles têm funções internas para comunicação com o gerenciador de janelas, o que diminui bastante o trabalho de programação. Outro motivo para se usar um Toolkit é o fato de os mesmos já terem uma arquitettura padrão de interface intrínseco, o que torna a o programa de aplicação condizente com um padrão. Existem muitos destes Toolkits hoje disponíveis no mercado [29].

1.1.2 Modelo Servidor-Cliente

O sistema X-Window foi concebido para trabalhar em rede. Desta forma um programa aplicativo não precisa rodar na mesma máquina que os resultados estão sendo apresentados [4,5,6].

Neste sistema são definidos dois tipos de programas: servidores e clientes. Um programa servidor, também chamado de X-Server, é um programa que realmente manipula imagens em um display. O servidor é quem realmente controla todo o acesso ao display. Os clientes são os programas de aplicação do usuário que usam os recursos oferecidos pelo X-Window (X-Facilities) e os programas utilitários (X-Utilities). Este modelo está mostrado na figura A.2.

O servidor é um programa local e não um programa que trabalha através da rede. Desta forma, o display com que um determinado usuário trabalha está também à disposição dos demais usuários da rede. O servidor age como intermediário entre os programas clientes que estejam rodando remota ou localmente e os recursos do sistema local. O servidor executa as seguintes tarefas:

- controla o acesso ao display por múltiplos clientes;
- interpreta mensagens vindas da rede enviados pelos clientes;
- envia mensagens aos clientes via rede;
- executa o traçado de vistas bidimensionais. Os gráficos são feitos pelo servidos ao invés de diretamente pelo cliente;
- mantém estruturas de dados, incluindo janelas, cursores, fontes e contextos gráficos[10], as quais podem ser compartilhada por outros clientes.

Assim o cliente se comunica com o servidor ao invés realizar operações gráficas diretamente no terminal. O servidor é adaptado para trabalhar com um tipo particular de terminal gráfico.

O servidor é também capaz de realizar operações gráficas primitivas e mostra textos numa variedades de fontes e tamanhos diferentes. Muitas das mensagens do cliente para o servidor são curtas, contendo apenas comandos de alto nível e alguns parâmetros.

Um servidor de display fornece uma abstração em nível de programação para o dispositivo físico. Devido ao fato de que os códigos que são dependentes do hardware estarem incorporados ao servidor, um sistema de janelas apoiado neste modelo é extremamente portável [6].

Em ambiente de rede é comum para o usuário possuir programas rodando em várias máquinas. Este tipo de processamento é chamado de processamento distribuído. O processamento distribuído ajuda a solucionar o desbalanço de carga no

[10] é um conjunto de atributos agrupados e que são usados no momento de ser feito alguma operação gráfica. E estabelecido pelo programa cliente.

sistema. Quando há uma máquina sobrecarregada, os usuários podem fazer com que os programas rodem em outras máquinas.

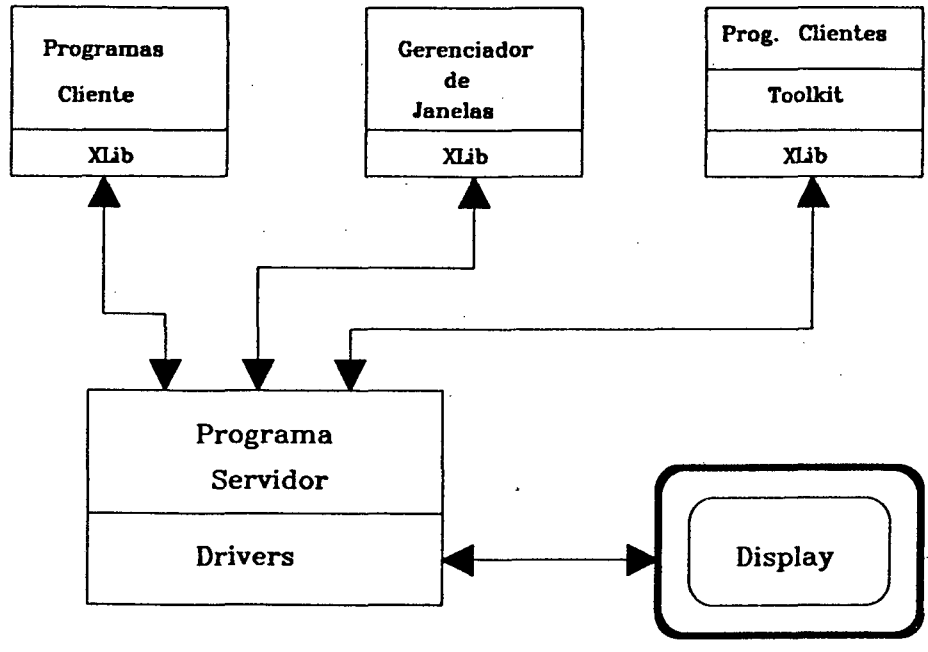

figura A.2 - Modelo Servidor-Cliente.

Atualmente já existem terminais dedicados, cuja única função é rodar um servidor do tipo X-Server, chamado de X-Terminal. O X-terminal pode ser ligado a uma rede por meio de um cabo serial ou uma placa de rede e não pode por ele mesmo rodar um programa cliente. Ele transfere esta tarefa a outro terminal. É possível também que um PC seja usado como terminal para o sistema X-Window usando um software especial, além do hardware para conectá-lo à rede.

1.1.3 Gerenciador de Janelas

Um dos objetivos originais do sistema X-Window foi não impor estilo algum de interface ao usuário. Os projetistas reconheceram que nenhuma interface seria adaptável a todos os casos e produziria um sistema de janela que não seria de utilidade a todas os programas de aplicação. Esta tarefa foi destinada ao programa gerenciador de janelas [4].

O gerenciador de janelas (Window Manager) é um programa escrito usando-se a XLib, porém que possui um status especial por convenção. Ele controla o layout das janelas que são apresentadas na tela, permite ao usuário mover, redimensionar, abrir novas janelas, iniciar novos programas de aplicação, bem como controlar a sobreposição das janelas na tela.

Ao gerenciador também cabe implantar um padrão de interface gráfica (Window Layout Policy), o qual estabelecerá a forma de funcionamento e a aparência das janelas na tela. O padrão X-Window não especifica nenhum padrão para o sistema de janelas ou para o gerenciador de janelas. Os dois padrões mais usados em estações de trabalho são os padrões Open Look e Motif. Atualmente não há uma tendência clara pela adoção de um ou de outro padrão.

vi. Interface Gráfica

A qualidade da interface com o usuário depende do nível tecnológico do hardware empregado. Os primeiros sistemas computacionais interativos se comunicavam com o usuário através de terminais do tipo TTY (abreviatura de teletypewriter), os quais só aceitavam caracteres de entrada teclados e só podiam escrever em papel, um caracter após o outro. Este foram os primeiros terminais usados para mainframes. Os TTY possuíam um outro problema: cada comando tinha que ser enviado através de uma ligação serial lenta. Uma vez que o comando chegava ao computador ele tinha que ser decodificado. Desta forma, uma CLI (Command Line Interface) tinha que minimizar a quantidade de informações a fim de facilitar o acesso ao mainframe. Esta também é uma das razões porque toda CLI tinha uma tendência de ter comandos curtos e abreviados. Esta era a única forma de acelerar o processo de envio de comandos. Assim, a primeira interface interativa com o usuário foi a CLI. A mais familiar hoje é a interface oferecida pelo sistema DOS [7,28,29].

Conforme a tecnologia evoluiu, surgiram terminais do tipo VDT (Video Display Terminal) e se tornaram amplamente disponíveis. Estes terminais podiam posicionar caracteres em qualquer lugar da tela e rapidamente se tornaram um padrão em computação. Com este tipo de terminal o usuário podia mover o cursor pela tela, bem como voltar e corrigir erros. Este tipo de terminal ainda predomina hoje em dia, principalmente em PC's.

O próximo avanço foram os vídeos de alta resolução que podiam suportar interfaces gráficas e um mouse. Este tipo de tecnologia tornou possível o surgimento

A maioria dos fabricantes de computadores atualmente fornece algum tipo de interface gráfica junto com o computador, embora ela também possa ser fornecida por terceiros. Ela é construída numa camada acima do sistema operacional e visa facilitar o uso da máquina. Pode se dizer que uma interface gráfica é um meio intuitivo de se tratar com o sistema operacional e com o hardware da máquina. Desta forma, as pessoas podem produzir mais em menos tempo e com menores custos, já que o uso do computador fica facilitado. Além disso elas têm um apelo visual muito grande influenciando desta forma as pessoas a comprarem não só os computadores mas também sistemas capazes de rodar sob uma interface gráfica.

As interfaces gráficas são projetadas para tirar vantagem do fato que figuras usualmente comunicam mais do que palavras, visto que possuem intrinsecamente mais informações do que palavras. Construções gráficas tais como ícones e buttons transmitem informações mais rápida e claramente do que um texto o faz, da mesma forma que os sinais de trânsito (reconhecidos em todo o mundo) permitem às pessoas entenderem o seu significado sem muito esforço.

Uma GUI (Graphical User Interface) procura transmitir informações por meio de símbolos, cores, e outras representações gráficas. Os elementos de uma GUI são colocados juntos dentro de uma ambiente onde cada aplicação compartilha do mesmo estilo e aparência, de forma a manter a consistência em todos os programas que rodam dentro deste ambiente. Se os programas não mantiverem todos uma consistência quanto ao estilo e a forma de interação a GUI perderá muito de sua utilidade, uma vez que o usuário terá que aprender como utilizar cada programa de aplicação em particular, além de gerar confusão entre os diversos objetos mostrados sob uma mesma tela.

O sistema X-Window, como outros sistemas, divide a tela em várias áreas de entrada e saída, cada uma das quais age como um terminal virtual independente. Usando um emulador de terminal pode-se também rodar programas cuja entrada é baseada em caracteres. Todavia pode-se rodar programas feitos para tirar vantagens de outras potencialidades gráficas oferecidas pelos sistema. As entradas são tomadas de um dispositivo para apontar objetos na tela, tal como um mouse. O mouse serve para controlar o programa por meio de imagens gráficas sobre a tela sem o uso do teclado, embora também seja possível fazê-lo com o teclado. Este método geralmente é mais fácil e mais intuitivo de aprender do que o teclado. O mouse também é usado para dirigir a entrada para uma certa janela na tela, visto que apenas uma aplicação pode receber entradas a um só tempo.

para dirigir a entrada para uma certa janela na tela, visto que apenas uma aplicação pode receber entradas a um só tempo.

Uma das primeiras interfaces gráficas a se tornarem populares foi a desenvolvida pela ***Apple*** para o seu computador pessoal chamado Macintosh e é um dos motivos do sucesso deste computador. Este sistema é rigidamente ligado ao hardware da máquina e influenciou o desenvolvimento de vários outros padrões de interface.

Enquanto o Macintosh foi concebido originalmente para uma GUI, o PC foi concebido com uma CLI. O PC usa para a entrada de comandos um arquivo chamado ***command.com*** . O ***command.com*** fornece o prompt do sistema e executa comandos internos do DOS tais como erase, copy e dir. O ***command.com*** também carrega e executa arquivos do tipo batch. Versões mais antigas do DOS requeriam que o ***command.com*** original estivesse rodando. Atualmente é possível substituí-lo por outro interpretador de comandos, incluindo uma interface gráfica como o Windows. A criação de uma interface gráfica para o PC envolve o desenvolvimento de ferramentas gráficas tais como drivers de tela, programas para manejar os objetos na tela, redimensionamento de objetos, etc... Em seguida estes programas adicionais são carregados sobre a estrutura do sistema operacional. Desta forma, as primeiras interfaces para PC eram lentas e consumiam boa parte da memória. A interface que mais se popularizou para PC foi o Windows.

1.1.1 Sistemas de Janelas

O sistema de janelas é o software do sistema que controla os displays do tipo bitmap e seus dispositivos de entrada associados. Ele fornece uma interface de alto nível para programas de aplicação , permitindo que eles desenhem e manipulem janelas sem recorrer ao sistema operacional da máquina.

A figura 2.3 mostra vários componentes típicos de um sistema de janelas. O sistema janelas é o responsável pela criação dos objetos gráficos primitivos tais como linhas ou textos, controlar a posição do cursor e o ordenamento das janelas. A ele também cabe coordenar o uso dos recursos da máquina e como os mesmos são compartilhados por outras máquinas tais como o mouse e do teclado. Geralmente é colocado um Toolkit numa camada acima do sistema de janelamento que fornece uma biblioteca de rotinas para uma manipulação mais complexa dos objetos gráficos.

O Toolkit também pode incluir rotinas para acesso ao sistema de arquivos, gerenciamento de memória e estruturação de dados que podem ser usados para a

A interface interativa com o usuário e o gerenciador de janelas permite criar, destruir, selecionar, e comunicar-se com outras janelas da tela.

1.1.2 Padrão Open Look

O Open Look é um norma, ou um padrão, contendo um conjunto de regras e definições que descrevem como a interface deve se parecer para o usuário e como ela deve operar. Este padrão não especifica como ele deve ser implantado ou detalhes em nível de hardware e software. Ele especifica, por exemplo, de que forma uma janela pode ser redimensionada, mas não especifica como o software e o hardware devem fazê-lo. Um exemplo de sistema de janelas usando o padrão Open Look é o ***OpenWindows*** , fornecido pela Sun.

O padrão Open Look lançado em 1988 pela AT&T e foi especificado como a interface para a versão SVR4 (System V Release 4.0) do Unix [4,5,6].

No Open Look a tela é chamada de Workspace. O Workspace contém janelas e ícones representando programas de aplicação. O exemplo mostrado na figura A.3 é uma tela típica do Open Look.

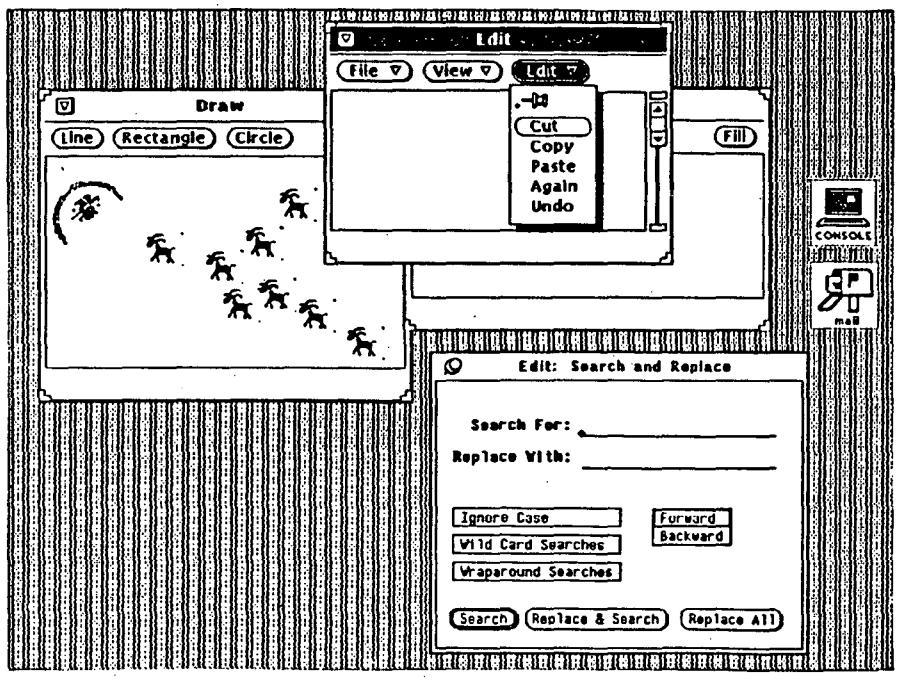

figura A.3 - Exemplo de interface gráfica

No topo de cada janela de aplicação está o cabeçalho, o qual contém o nome da aplicação a ser usada.

Abaixo do cabeçalho está a área de controle que fornece acesso para as funções do programa de aplicação, tais como abrir e fechar arquivos. A área de controle tipicamente consiste de uma linha simples de botões. O usuário tecla os botões movendo ponteiro do mouse sobre ele e tecla em seguida o botão de seleção do mouse. Os botões cujos nomes terminam em "..." significam que há um submenu associado a eles. Quando se tecla um destes botões um menu de opções aparece na tela.

Abaixo da área de controle vem a área de dados, onde o programa de aplicação mostra seus dados, cuja forma pode ser das mais diversas, como um texto, um traçado gráfico ou uma tabela.

A direita da área de dados está uma barra de rolagem que permite ao usuário se mover dentro da janela.

Existe ainda um tipo de janela especial chamado de janela de alerta. Esta janela aparece na tela toda vez que uma operação errada for efetuada, bem como antes que uma operação que resulte em perda de dados seja efetuada.

O Open Look também fornece ajuda por meio de janelas do tipo ***Help***. Estas janelas aparecem quando se aponta para o objeto para o qual se necessita de ajuda e se tecla em seguida a tecla ***HELP*** do teclado (algumas máquinas a tecla F1 tem o mesmo efeito). Esta janela dá uma descrição a respeito da função do objeto e da sua forma de funcionamento.

Os objetivos principais do padrão Open Look são fornecer as seguintes características:

- bom projeto visual das janelas;
- balanço entre simplicidade, consistência e eficiência;
- independência de dispositivos.

1.1.3 Toolkits

Conforme já dito o sistema X-Window é um sistema que não estabelece nenhum tipo de regra a respeito de como uma interface gráfica deve se parecer ou como ele deve interagir com o usuário. Ela apenas fornece as ferramentas necessá-

rias para que se construam programas de aplicação que estão de acordo com qualquer padrão de interface que se escolha.

Geralmente se usa um Toolkit de alto nível a fim de se facilitar a construção de uma interface gráfica. Por outro, lado o programa de aplicação também pode fazer chamadas para a XLib a fim de mostrar representações gráficas que não são fornecidas pelo Toolkit. Desta forma, existem alguns Toolkits que implementam o padrão Motif e outros o padrão Open Look [4,5,6]. Existem outros ainda que podem implementar ambos, de acordo com a máquina em que estiver rodando.

Geralmente os Toolkits definem classe de objetos que têm características únicas compartilhadas por cada um dos membros da classe, mas distinta para cada classe. A hierarquia de classes é baseada numa camada básica chamada de ***generic object*** da qual outros objetos são subclassificados. Esta camada define características comuns a todos os objetos tais como tamanho, posição e cores de fundo. Conforme as subclasses se tornam mais especializadas, suas definições se tornam mais específicas.

Os Toolkits também fornecem maneiras de mostrar e interagir com os objetos que os compõem, desta forma liberando o programa de aplicação desta tarefa que geralmente é complexa. Além disso, ele pode interagir com várias entidades, entre as quais:

- sistema de janelas, para fazer pedidos e criar janelas, desenhar gráficos e ser notificado dos eventos de interesse do programa de aplicação;
- sistema operacional, para ser notificado das mudanças de status de uma operação, tratar com arquivos, captar e enviar sinais para o sistema, etc..
- programa de aplicação, a fim de passar o controle para as rotinas que realmente realizam as tarefas para o qual o programa de aplicação se destina.

O Toolkit junto com suas camadas de implementação tais como a XLib age como um mediador entre o programa de aplicação, o sistema de janelas e o sistema operacional.

A seguir é feito uma descrição do Toolkit usado na implementação da interface do sistema desenvolvido neste trabalho.

1.1.4 XView

O XView é um Toolkit fornecido pela Sun Microsystems e atualmente faz parte da versão SVR4 (System V Release 4) do Unix, bem como é distribuído junto com os códigos do sistema X-Window pelo MIT [4,5,6].

Este sistema é feito para desenvolver aplicações gráficas interativas que rodam sobre o sistema X-Window. Ele fornece um conjunto de funções escritas em linguagem C que permitem criar manipular e destruir objetos gráficos, tais como menus, buttons e panels. A aparência e funcionalidade destes objetos seguem as especificações do padrão Open Look. Ele também é totalmente baseado na XLib, biblioteca de mais baixo nível do sistema X-Window.

Para o uso eficiente deste Toolkit é necessário, além do conhecimento da linguagem C, o conhecimento dos princípios da técnica de programação orientada para objetos, uma vez que ele é organizado dentro destes princípios. Além disso o seu uso não exclui o uso de funções da XLib [4,5,6,11,28].

O XView permite ao programador construir uma interface gráfica sem ter que se envolver com muitos dos detalhes que envolvem o sistema de janelas no qual o mesmo trabalha. Este sistema também extensível, ou seja, o programador pode incluir outras classes no mesmo, bem como construir classes a partir das já existentes [11].

1.1.4.1 Manipulação de eventos

Um evento no sistema X-Window é uma estrutura de dados enviados pelo servidor descrevendo que algo aconteceu e que pode ser de interesse do programa de aplicação. Por exemplo, quando o usuário pressiona uma tecla ou um botão do mouse, ou move uma janela pela tela, é gerado um evento. É tarefa do servidor distribuir eventos para as várias janelas da tela.

No XView, entre o servidor e o cliente, há um mecanismo de despacho de eventos chamado de notificador, conforme mostrado na figura A.4. Após a fase de inicialização do programa de aplicação, onde são criados e inicializados os objetos, pode-se configurar como a distribuição das entradas será feita. O método mais simples é entregar o controle do programa de aplicação para o notificador do XView. Daí em diante o notificador fará a distribuição dos eventos de forma automática para os objetos criados.

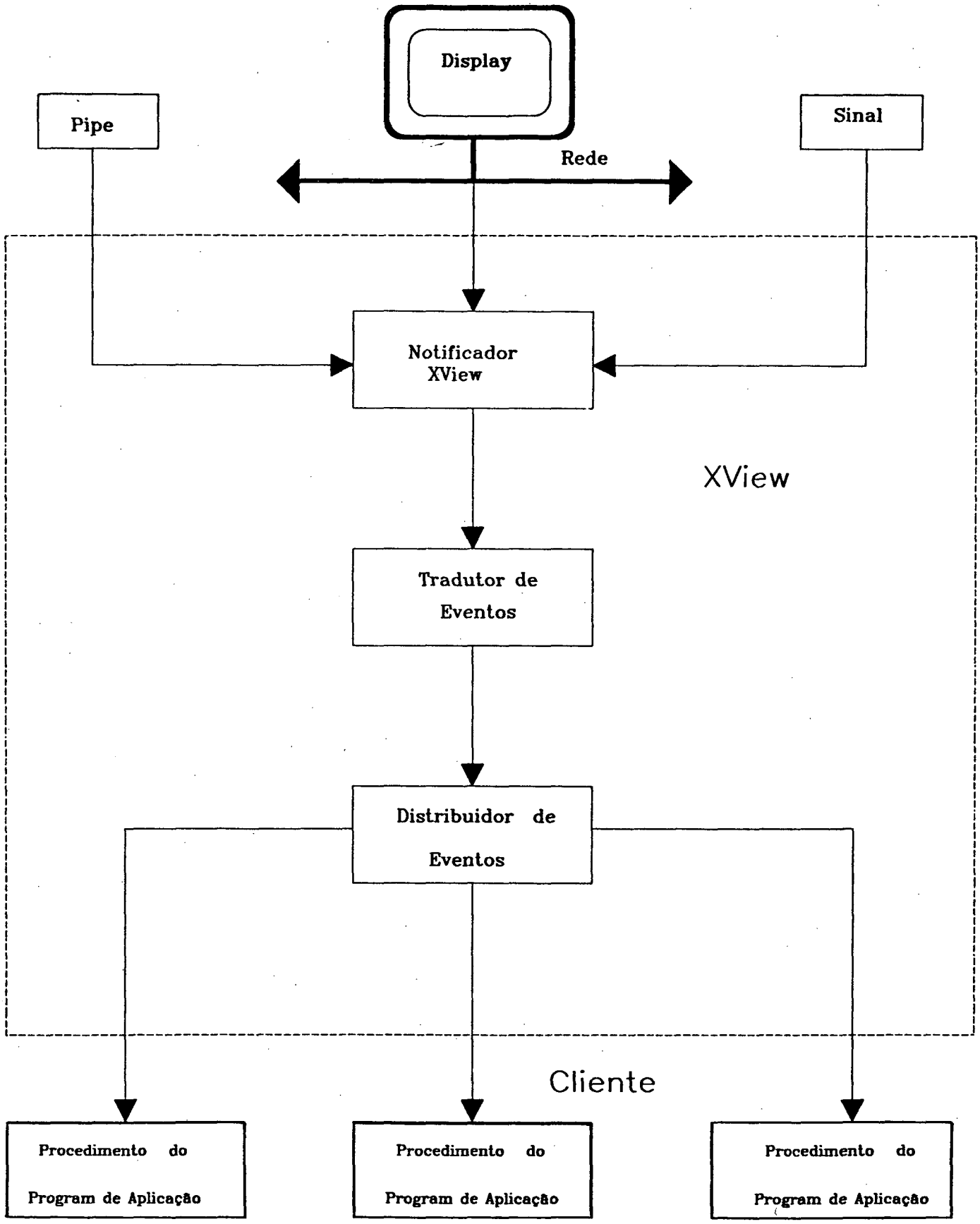

figura A.4 - Notificador do XView

Por sua vez, os objetos possuem mecanismos de comunicação com o notificador que fazem com que o programador não tenha que se envolver muito com os seus detalhes.

O ponto básico é que o programa de aplicação apenas é notificado dos eventos no qual ele especificamente demonstrou interesse. Respondendo a estes eventos o programa de aplicação pode realizar a tarefa para o qual foi desenvolvido, por exemplo, um cálculo matemático. Por exemplo, se o usuário teclar a letra "a" do teclado em alguma janela do programa de aplicação, o servidor vai passá-lo para o XView que por sua vez poderá ou não passá-lo ao programa de aplicação. Por sua vez, o programa de aplicação poderá interpretar este evento e agir de acordo. Após o programa de aplicação processar o evento, o controle retorna para o notificador de tal forma que um novo evento possa ser lido e processado.

O notificador não só faz a distribuição das entradas mas também permite a seleção de diferentes fontes de entrada. Além da manipulação dos eventos das janelas, o programa de aplicação pode manipular interrupções geradas pelo sistema operacional por meio do XView.

Os tipos de janelas que o XView fornece são as seguintes:

- ***Canvas*** , nos quais os programas podem desenhar;
- ***Text Subwindows*** , janelas de texto com possibilidades de edição de texto;
- ***Panels*** , painéis contendo itens tais como buttons, itens de escolha e réguas;
- ***TTY Subwindows*** , janelas que emulam terminais baseados em caracteres.

Estas janelas são arranjadas dentro de ***Frames*** , que são também janelas.

1.1.4.2 Modelos de Programação

Para que seja possível fazer uma programação eficiente com o XView é preciso que se entenda os conceitos básicos no qual o mesmo foi implementado.

O XView foi construído dentro dos conceitos de programação orientada para objetos. Os objetos podem ser considerados como blocos construtivos básicos, com os quais a interface com o usuário é implementada. Cada objeto tem um conjunto de propriedades que podem ser configuradas.

Os princípios no qual o XView se apóia são:

- os objetos são representados por uma hierarquia de classes;
- os objetos são dados opacos;
- os objetos têm atributos que podem ser estabelecidos via funções de passagem de mensagens;
- os objetos podem ter procedimentos associados a eles, os quais são acionados por meio de geração de eventos.

O XView define classes de objetos em uma árvore de hierarquia. Por exemplo, Frame é uma subclasse de uma classe mais geral que é a Drawable. Drawable é uma subclasse da classe Generic Object, etc... A figura 2.5 mostra a hierarquia das classes e a relação entre as classes.

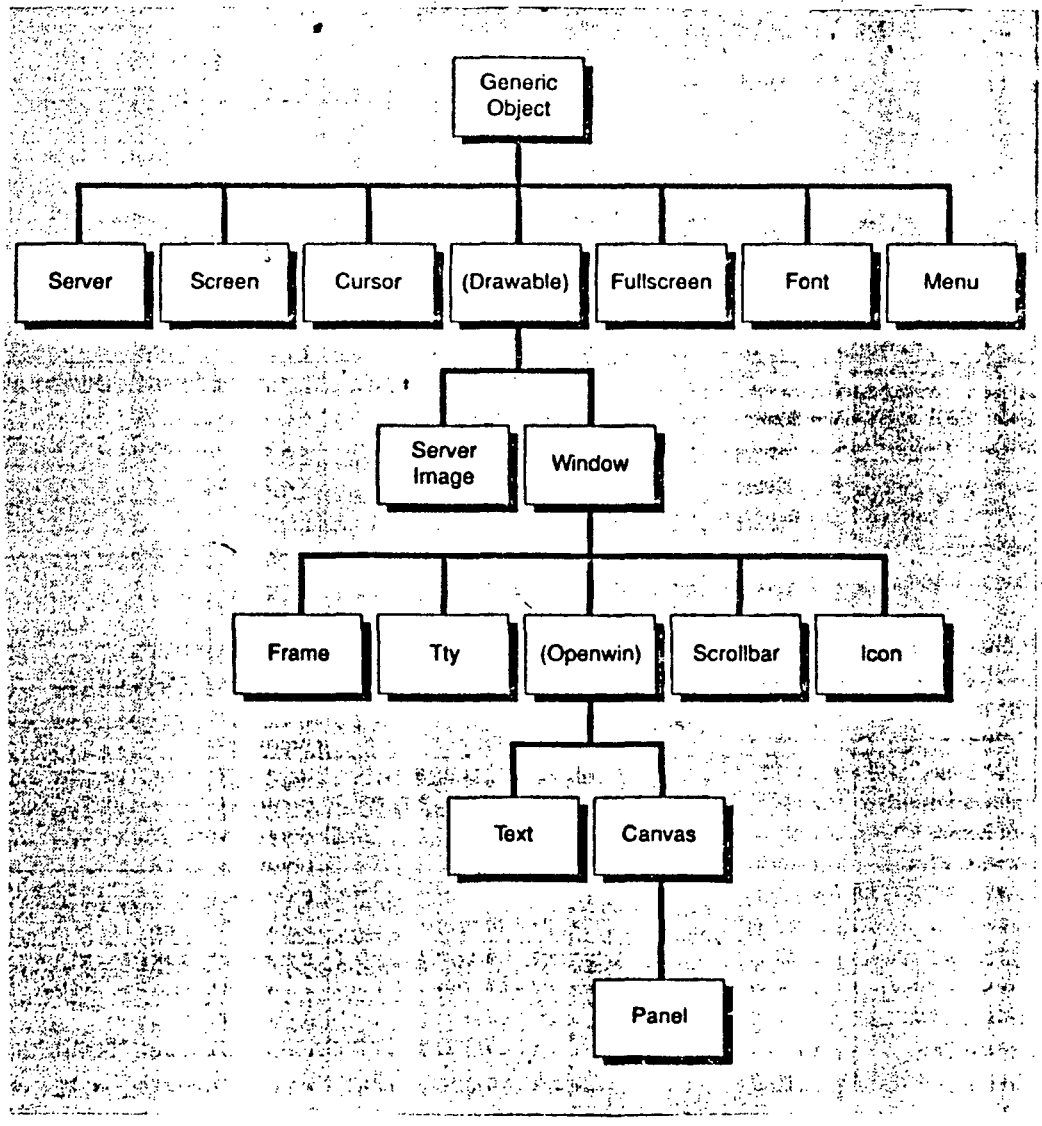

figura A.5 - Hierarquia de classes do XView

Cada classe tem características que a fazem única em relação às outras classes. Todas as classes também herdam as características de sua superclasse (classe que está acima dela) [11].

As classes contêm também propriedades que são compartilhadas por todas as instâncias dela. O estado destas propriedades pode variar entre as várias instâncias [11].

1.1.4.3 Modelo do Notificador

O XView é um sistema baseado no notificador que age como uma entidade controladora dentro de um processo do usuário. Ele lê as entrada do sistema operacional e as formata para um evento de alto nível, que é então distribuído para os objetos [4].

No estilo convencional de programação interativa o laço central de controle reside na aplicação. Um editor, por exemplo, lê um caracter, toma alguma ação baseada no caracter, em seguida lê o próximo caracter. Quando um caracter que representa o pedido do usuário para terminar é enviado, o programa encerra. A figura A.6 ilustra este estilo de programação.

Um sistema baseado em um notificador inverte esta estrutura de controle do programa. A malha principal de controle reside no notificador e não na aplicação. O notificador lê vários eventos e chama as rotinas que o usuário registrou previamente com o notificador. Estes procedimentos são também chamados de ***callback procedures***, ou ***notify procedures*** . Esta estrutura de controle é mostrada na figura A.7.

A grande vantagem desta estrutura é que o notificador assume a tarefa de gerenciar os eventos associados com os objetos. Como geralmente os programas de aplicação possuem muitos objetos, esta tarefa geralmente é de grande complexidade. Sem um notificador o programa de aplicação teria que ser responsável pela detecção e despacho de todos os eventos dos objetos. Por outro lado, com um notificador, cada componente do programa de aplicação só recebe os eventos que lhe são de interesse.

O programa de aplicação também não precisa interagir diretamente com o notificador. O XView tem um mecanismo no qual os objetos interagem diretamente

com o notificador, registrando os seus procedimentos. O programa de aplicação por sua vez registra os procedimentos com os objetos.

Quando se cria um programa de aplicação para o XView segue-se os seguintes passos:

- cria-se a várias janelas e objetos necessários para a interface;
- registra-se eventos e procedimentos;
- passa-se o controle para o notificador.

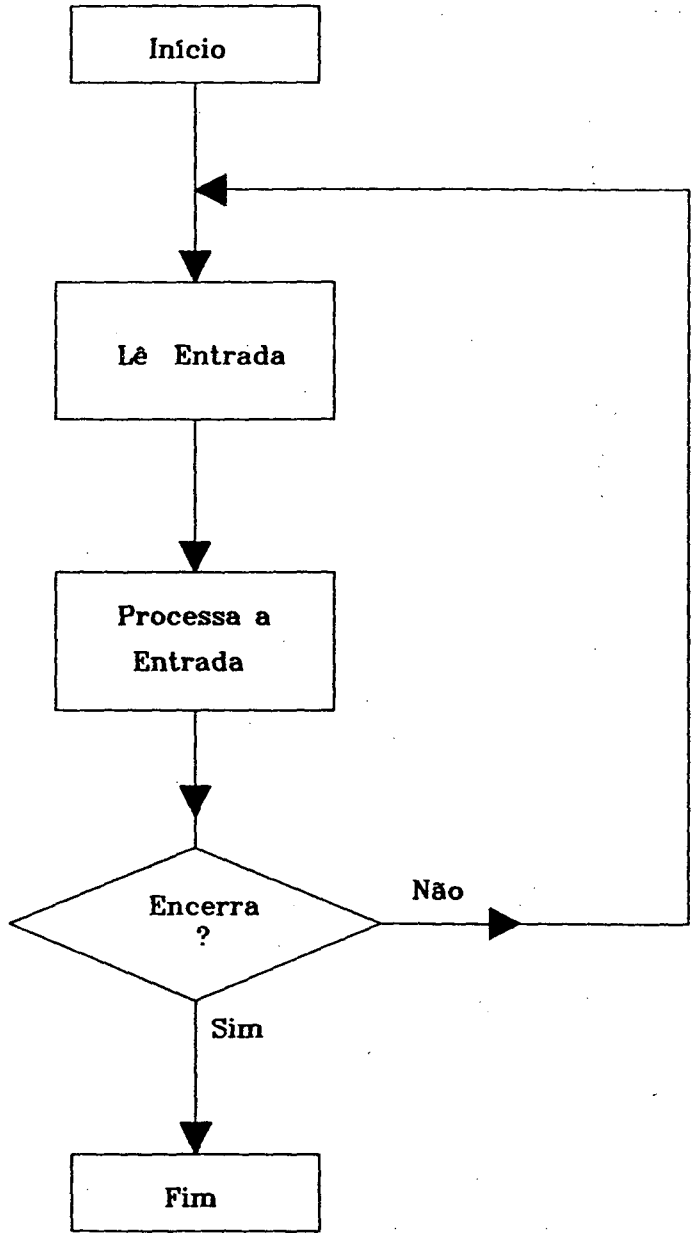

figura A.6 - Estilo tradicional de programação

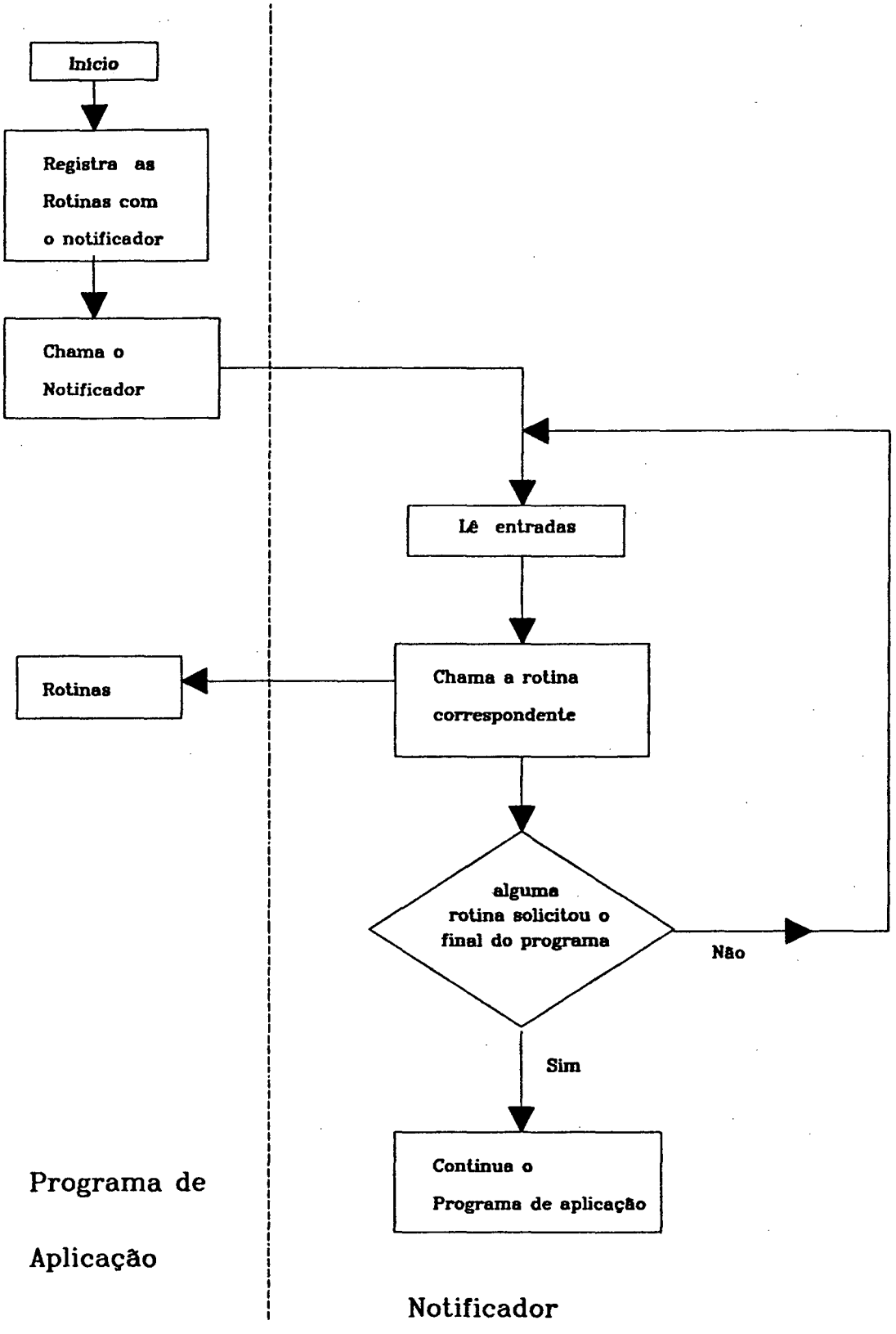

figura A.7 - Estilo de programação baseado no notificador

Desta forma o programa executa até que o usuário o encerra por meio do teclado ou de uma rotina específica para isso associado a um objeto.

vii. Linguagem FORTRAN

O Fortran (Formula Translation) surgiu em 1953 e foi a primeira linguagem de alto nível até aquele momento. Ele surgiu para facilitar a comunicação com a máquina e propiciava uma maneira mais simples de escrever programas. Antes do Fortran os programadores eram obrigados a escrever os programas diretamente em linguagem de máquina (Assembler).

A introdução do Fortran visava resolver problemas matemáticos envolvendo um grande número de repetições, especialmente envolvendo matrizes. Devido a essa facilidade oferecida pelo Fortran, existem hoje vários pacotes contendo rotinas escritas em Fortran.

A sobrevivência do Fortran por tantos anos está ligada ao fato de ele ter se tornado uma linguagem padronizada. Ou seja, um programa escrito dentro dos comandos padrões pode rodar em vários tipos de máquinas sem alterações e produzindo os mesmos resultados. Por outro lado o volume de programas escritos em Fortran é tão grande que parece ser realmente impossível que o mesmo venha a ser abandonado. O seu uso atualmente é muito difundido, principalmente no meio universitário.

O primeiro documento publicado padronizando o Fortran foi publicado em 1966 (ANSI X3.9 - 1966). Este documento descrevia o Fortran IV e eliminou o Fortran II. Ele constituia-se em 26 páginas e levou 4 anos para ser elaborado.

O próximo padrão foi o Fortran 77, que levou 7 anos para ser elaborado e continha cerca de 150 páginas. Ele substituiu o anterior e representou o trabalho de centenas de técnicos e sugestões de todo o mundo.

Atualmente está sendo discutido um novo padrão para o Fortran, que se chamará Fortran 90. Este padrão está sendo elaborado por comitês internacionais dos Estados Unidos e orgãos internacionais de padronização. O novo padrão deverá incluir:

- identificadores de tamanho maior;
- comentários do tipo in-line e linhas com vários comandos;

- operadores relacionais simbólicos, ou seja deverá ser usado > ao invés de .GT.;
- uso do comando include;
- inclusão de um procedimento dentro de outro, tal como atualmente no PASCAL;
- chamada recursiva a procedimentos, ou seja uma subrotina pode chamar ela própria, tal como no C;
- módulos contendo ambos os tipos de dados e procedimentos, com níveis de acesso público e privado; estes recursos visam viabilizar a técnica de programação orientada por objetos usando o Fortran;
- operação com bits, como atualmente na linguagem C;
- inclusão de várias operações envolvendo arranjos;
- utilitários como o DATE e TIME;
- melhoramento das estruturas de controle, incluindo estruturas como o DO WHILE e CASE;
- melhora dos padrões de entrada e saída, especialmente para registros em banco de dados;
- novos tipos de dados;
- inclusão do uso de ponteiros.

O objetivo destas alterações todas é dotar o Fortran de mecanismos atualmente de uso comum em outras linguagens como o C e Pascal.

viii. Linguagem C

O C é uma linguagem de propósitos gerais que foi projetada e implementada em 1972 no Bell Laboratories. Seu crescimento inicial estava largamente associado ao sistema Unix, o qual foi escrito dentro desta linguagem. Nos últimos anos o C tem se tornado popular também em outros ambientes e não está mais ligado a um ambiente particular [3,8].

O C foi inicialmente projetado para a programação de sistemas, isto é para escrever programas como compiladores, sistemas operacionais e editores de texto. Todavia, ele se mostrou satisfatório também em outras aplicações, incluindo banco de dados, sistemas de chaveamento usados em sistemas telefônicos, análise numérica e programas de engenharia.

A idéia inicial na criação do C foi a simplicidade com o objetivo de assegurar a eficiência, uma vez que ele se destinava a criação de sistemas. Por exemplo ele suporta como tipos básicos somente aqueles que são suportados por hardwares típicos:

- caracteres;
- inteiros, de vários tamanhos;
- números em ponto flutuante de vários tamanhos.

É possível, contudo, criar tipos mais complexos tais como estruturas e arranjos. As funções que manipulam estes tipos são escritas em bibliotecas à parte.

Algo incomum no C é que ele não fornece operações de entrada e saída como parte da linguagem. As funções de entrada e saída são definidas em bibliotecas separadas e não construídas como parte da linguagem. Esta característica contrasta com o Fortran que possui estas funções incorporadas à linguagem. As funções de alocação de memória também são fornecidas por bibliotecas separadas.

Atualmente o C é uma das linguagens mais usadas do mundo e existem compiladores para C em praticamente todos os tipos de computadores e sistemas operacionais.

Algumas das características do C são :

- flexibilidade, é possível aplicar o C para quase todas as áreas de programação e usar praticamente qualquer técnica de programação;
- eficiência, é fácil para o programador e o compilador usarem o hardware da máquina de forma eficiente.
- disponibilidade, os compiladores para C são disponíveis em praticamente todos os computadores e sistemas operacionais.
- portabilidade, o nível de portabilidade é tal que mesmo as rotinas dependentes do hardware são facilmente adaptadas.

Os comandos de controle de fluxo do C são muito mais poderosos do que os existentes no Fortran. Eles incluem estruturas do tipo IF-ELSE-ELSEIF, WHILE,DO-WHILE, FOR, SWITCH, CASE, etc..

As funções em C são recursivas, ou seja uma função pode se autoreferenciar. Este mecanismo também simplifica muito a programação em determinadas situações. Os argumentos e valores de retorno de uma função podem ser de qualquer um dos tipos básicos descritos anteriormente. A passagem de argumentos para as funções pode ser feita de duas formas básicas :

- passagem por valor;
- passagem por referência.

Quando se utiliza a passagem por valores a função recebe apenas uma cópia da valor da variável. Ou seja o valor original da variável não é alterado. Na passagem por referência é passado para a função o endereço da variável e, desta forma, o valor da variável pode ser alterado dentro da função chamada. Este mecanismo precisa ficar muito claro para o programador, principalmente quando se pretende interfacear o C com outra linguagem. No Fortran, também por definição, os valores das variáveis são passados para as rotinas e funções por referência, embora em alguns compiladores há a possibilidade de alterar esta caracterísitica [3,8].

Um fato importante no C que o distingue da maioria das linguagens de programação é a sua habilidade de alocar a memória dinamicamente. Ou seja, o programa pode decidir quanto usar de memória no momento de execução do mesmo. Desta forma a memória disponível da máquina pode ser usada de forma mais eficiente.

Apesar de sua grande popularidade, o C não possuía um padrão definido até 1983. Em 1983 a ANSI estabeleceu um comitê para estabelecer um definição moderna e compreensível do C. Como resultado deste comitê surgiu o padrão ANSI-C. Atualmente a maioria dos compiladores segue este padrão.

Uma das maiores contribuições do padrão ANSI-C foi a definição de uma biblioteca para acompanhar o C. Ela especifica funções para acessar o sistema operacional, entrada e saída formatada (scanf e printf), alocação dinâmica de memória (malloc), manipulação de strings (por exemplo, strcmp), funções matemáticas, etc...

Ela também definiu um conjunto de arquivos de cabeçalhos que podem ser incluídos em programas escritos pelo usuário, fornecendo acesso às declarações das funções e dos tipos de dados contidos nas bibliotecas.

ix. Programação Usando Mais de Uma Linguagem

Há situações em que se deseja explorar as características especiais de várias linguagens. Por exemplo, poderia ser desejável escrever as fases de entrada e saída de um programa em uma linguagem em que a manipulação de strings fosse fácil, enquanto que a computação fosse escrita em uma linguagem em que os cálculos fossem rápidos e eficientes. Além disso o uso de mais de uma linguagem poderia fa-

cilitar o uso de bibliotecas já desenvolvidas. Todavia, na prática o uso de mais de uma linguagem nem sempre é tão fácil de ser implementado. Há basicamente dois tipos de problemas a ser enfrentados [8,10,11]:

- problemas relacionados com as definições incompatíveis nas linguagens;
- problemas relacionados com implementações incompatíveis.

O primeiro ítem inclui diferenças tais como os tipos definidos numa linguagem nem sempre são os mesmos que os definidos em outra; além disso, pode haver diferenças nos mecanismos e semântica de passagem de argumentos entre funções. O segundo ítem inclui diferenças tais como a representação de dados, manuseio de memória e suporte para as funções de entrada e saída.

Devido aos problemas citados o uso de mais de uma linguagem em um programa tem sido limitado a situações e linguagens muito específicas.

x. Interface C/Fortran

Nesta seção será descrito como a interface entre a linguagem C e Fortran pode ser feita. A interface pode ser definida como um conjunto de regras que permitem a utilização de ambas as linguagens em um programa. Este tipo de interface foi implantado nos programas que foram descritos no capítulo 2. Os equipamentos utilizados são as estações de trabalho da Sun. As regras, no entanto, tem um caracter genérico suficiente para que sua implementação noutras máquinas seja possível. A maioria das regras são baseadas em convenções próprias dos padrões da linguagem e não em implementações específicas.

Respeitadas algumas convenções é possível chamar funções escritas em C do Fortran e vice-versa. Para a chamada de funções é necessário observar-se os seguintes aspectos de ambas as linguagens:

- definições de chamadas de subrotinas;
- caracteres acrescentados pelo compilador aos nomes de funções e rotinas;
- forma de passagem dos argumentos entre rotinas (por parâmetros ou valores);
- compatibilidade entre os tipos de dados;
- definir quais as bibliotecas que serão usadas;
- opções passadas ao compilador.

O termo função significa coisas diferentes no Fortran e no C. No C todos os subprogramas e rotinas são funções, embora algumas retornem um valor nulo. No Fortran todas as funções retornam valores, ao passo que as subrotinas não retornam valor. Desta forma ao ser chamada uma função escrita em C a partir do Fortran deve-se observar o seguinte:

- se a função em C retornar um valor, deve-se chamá-la da mesma forma que se chama uma função do Fortran;
- se a função escrita em C não retornar um valor, deve-se chamá-la como uma subrotina em Fortran.

Por outro, lado ao se chamar uma função escrita em Fortran a partir do C deve-se observar o seguinte:

- se for uma função em Fortran, ela deve ser chamada com um valor de retorno armazenado numa variável cujo tipo seja equivalente no C.
- se for uma subrotina em Fortran, ela deve ser chamada do C como se fosse um função que retorna um valor inteiro.

Em C existe diferença entre as letras maiúsculas e minúsculas na formação de nomes de variáveis, nomes de rotinas e funções. Como no Fortran esta diferença não existe deve se ter atenção para este detalhe. O compilador do Fortran também acrescenta um um caracter de sublinhado (_) ao nome das subrotinas. Isto é feito para evitar o conflito com outras funções. A forma mais fácil de tratar este problema consiste em acrescentar este caracter ao nome das rotinas chamadas do Fortran.

Os arranjos de variáveis sempre começam em zero no C, enquanto que em Fortran eles começam em um. Há duas maneiras de se tratar com esta caracterísitica:

- usa-se o default do Fortran; desta forma um elemento do Fortran, por exemplo, a(2), corresponderá ao elemento b(1) do C.
- especifica-se que os arranjos do Fortran comecem em zero. Por exemplo um arranjo de 3 posições poderiam ser declarado como ***integer b(0:2)*** e, desta forma, começaria em zero ao invés de um. Assim haverá uma correspondência exata entre os elementos do C e do Fortran.

Outra caracterísitica é que os elementos de arranjos do Fortran são armazenados por ordem de coluna, enquanto que no C eles são armazenados por linha. Para arranjos de duas dimensões isto não é problema, uma vez que bastará inverter os indíces. Para arranjos de mais de duas dimensões este problema exige outras considerações.

O seguinte exemplo mostra como pode ser chamada uma função escrita em C a partir de um programa principal escrito em Fortran. Este são os códigos do programa em Fortran:

```
Program refere
character c
integer   i
real      r
 .
 .
 .
call ref( c , i , r )
 .
 .
 .
end
```

Os códigos abaixo são da função escrita em C.

```
ref_( c , i , r )
char  *c;
int   *i  ;
float  *f;
{
 *c='x' ;
 *f=12.0;
 *i=10  ;
}
```

Observe-se que os valores passados para a função em C são os endereços das variáveis e portanto devem ser armazenados em ponteiros (variáveis que iniciam com um asterisco). Neste exemplo foi usado o mecanismo default do Fortran que é a passagem por referência. A chamada utilizada pelo Fortran é exatamente igual a qualquer outra chamada feita a rotinas escritas em Fortran. Também foi acrescentado uma barra ao nome da função *ref* chamada do Fortran.

O próximo exemplo mostra como passar valores para uma função escrita em C. Todas as funções em C podem retornar um valor. Os códigos do programa principal são:

```
Program   princ
integer   r , s , retint
 .
```

```
 .
 .
r=2
s=retint( r )
 .
 .
 .
stop
end
```

Os códigos da função em C são os seguintes:

```
int retint_( r )
int *r ;
{
 int s ;
 s =*r;
 s++   ;
 return ( s ) ;
}
```

O exemplo seguinte mostra como chamar uma subrotina escrita em Fortran a partir de um programa escrito em C. Os código do programa escrito em C são os seguintes:

```
main()
{
 char  t ;
 int   i   ;
 doubled ;
 float  sr;
  .
  .
  .
 exemp_( &t , &i , &d , &sr ) ;
  .
  .
  .
}
```

Os códigos da rotina em Fortran são os seguintes:

```
Subroutine  exemp( t , i , d , sr )
character t
```

```
integer   i
double precision  d
real  sr
t="v"
i=20
d =12.5e4
sr=34.0
return
end
```

Note-se que o programa principal, escrito en C, passa também os valores por referência, ou seja passa-se o endereço das variáveis usando-se o operador **&** .

O exemplo seguinte mostra como passar valores para um função escrita em Fortran a partir de um programa principal escrito em C. Os códigos do programa principal são :

```
main()
{
 int k, i ;
 .
 .
 .
 k=8 ;
 i =reti_( &k ) ;
 .

 .
}
```

Os códigos da função em fortran são os seguintes:

```
integer  function  reti( k )
integer  k
reti = k + 1
return
end
```

Deve ser notado que novamente os valores para a função em Fortran são passados por referência. Embora seja possível passar variáveis tanto por referência tanto quanto por valores a rotinas escritas em C, é impossível passar variáveis por valor para o Fortran. A solução em tal caso é passar por referência.

O exemplo a seguir mostra como pode ser tratado um arranjo quando se trabalha em conjunto com o C e o Fortran. No exemplo a seguir é passado um arranjo de nove posições para uma função escrita em C. Os códigos do programa principal:

```
Program vetor
integer i , sm
integer a(9) / 1,2,3,4,5,6,7,8,9 /
call  soma ( a , sm )
write (*, '(9i2, " - > " i3)')  (a(i), i=1,9) , sm
stop
end
```

Os códigos da função em C são :

```
soma_ ( v , s )
  int *s   ;
  int v[9] ;
  {
     int i;
     *s=0 ;
     for ( i = 0 ; i <= 8 ; i++ ) {
        *s += v[i];
     }
  }
```

Observe-se que a função em C referencia os seus elementos começando em zero ao passo que o programa em Fortran o faz a partir de um. O nome da variável definida como um arranjo em C já é por si só um ponteiro e assim a variável *v* não precisa vir precedida de um asterisco, como a variável *s* [3].