UM SISTEMA PARA DESENVOLVIMENTO DE PROGRAMAS BASEADOS NO MÉTODO DOS ELEMENTOS FINITOS

Jayme P. Gouvêa
Seção de Engenharia Mecânica.
Instituto Militar de Engenharia - IME.
Rio de Janeiro - Brasil.
Raúl A. Feijóo
Laboratório Nacional de Computação Científica
Rio de Janeiro - Brasil.

RESUMO

É apresentado o sistema SDP que auxilia o desenvolvimento de programas que utilizam o Método dos Elementos Finitos. São apresentados procedimentos para gerenciamento das memórias principal e secundária, rastreamento e depuração de rotinas, tratamento de erros, normas de documentação e uma aplicação linear desenvolvida com o auxílio deste sistema.

ABSTRACT

The SDP system that aids the development of Finite Element programs is introduced. It contains procedures to main and secondary memory management, routines trace, debuging, error manipulation and documentation rules. A linear problem application is shown.

INTRODUÇÃO

No Brasil, as instituições de ensino e pesquisa que desenvolvem "software" técnico-científico, com objetivos acadêmicos, enfrentam inúmeras dificuldades:

- reduzidos recursos humanos e financeiros,
- grande número de problemas de engenharia que podem ser resolvidos com métodos numéricos,
- existência de uma demanda crescente na utilização de "software" que está sendo suprida quase que exclusivamente por "pacotes" estrangeiros.

Grande parte deste "software" importado pode ser substituído por similar nacional, tendo em vista os resultados já alcançados em universidades e instituições de pesquisa do país. Para que este objetivo possa ser alcançado, é necessário uma cooperação mútua entre os grupos de pesquisa, para um intercâmbio de experiências e produtos, a fim de diminuir a superposição de esforços, utilizando com maior eficiência os recursos disponíveis.

Porém, a troca de informações e principalmente a troca de códigos de programação é extremamente dificultada pela variada gama de estilos de programação, algoritmos, técnicas de manipulação e armazenamento de dados, aliado a um grande número de equipamentos que, na maior parte das vezes, não são compatíveis.

De acordo com o exposto, os grupos de pesquisa interessados em trabalhar em conjunto devem compartilhar uma biblioteca comum de rotinas a adotar uma padronização para a programação, documentação e estrutura de dados.

Neste trabalho é apresentado o sistema SPD [1] que tem por objetivo proporcionar um ambiente computacional que vem ao encontro das necessidades anteriormente mencionadas.

O SISTEMA SDP

O "Sistema de Desenvolvimento de Programas Baseados no Método dos Elementos Finitos", ou simplesmente SDP, é um sistema computacional elaborado para auxiliar o desenvolvimento de códigos de programação para métodos numéricos que utilizam a técnica dos Elementos Finitos.

Basicamente, o sistema SDP é constituído por uma biblioteca de rotinas utilitárias e rotinas de cálculo e segue determinadas normas de programação, documentação e organização de dados.

Este sistema fornece uma metodologia de programação e normas de organização e documentação, criando uma linguagem comum entre grupos que se propõem a desenvolver "software" técnico-científico, permitindo o intercâmbio entre diferentes instituições de pesquisa. Pela maneira como foi desenvolvido o sistema SDP possibilita sua utilização em equipamentos de diferentes fabricantes ou até mesmo em micro-computadores.

O SDP foi desenvolvido dentro dos conceitos de programação estruturada [2] e utiliza algumas técnicas de organização de bancos de dados adaptados para os programas com aplicação em engenharia.

Com o sistema SDP procura-se alcançar outros objetivos tais como:

- diminuir ou eliminar a duplicação de esforços na elaboração de códigos,
- diminuir o tempo de preparação de programas aplicativos,
- minimizar a ocorrência de erros e facilitar a sua localização,
- facilitar a manutenção do software,
- dotar o sistema de uma estrutura flexível,
- permitir a portabilidade do sistema e
- fazer com que novos procedimentos desenvolvidos passem a ser parte do sistema.

O sistema SDP além de fornecer uma padronização para o desenvolvimento de programas, possui uma biblioteca de rotinas constituídas de procedimentos de gerenciamento, utilitários, blocos funcionais, rotinas de cálculo e programas (processadores).

Resumindo, o SDP consiste nos seguintes elementos:

a) Gerenciadores e utilitários:

- utilitários para rastreamento, depuração e tratamento de erros,
- gerenciamento da memória principal (direta),
- gerenciamento da memória secundária (arquivos em disco) e
- utilitários para fornecer tempo de execução e outros.

b) Biblioteca de procedimentos de cálculo:

- pré-processadores, processadores e pós-processadores,
- blocos funcionais e
- rotinas de cálculo e procedimentos básicos.

c) Documentação:

- normas de programação e documentação
- programas utilitários para manipulação da documentação
- manuais de documentação de gerenciadores, utilitários e da biblioteca de rotinas e
- manuais de programas utilitários e aplicativos.

Os processadores, blocos funcionais e as rotinas de cálculo seguem certas regras de programação e possuem uma hierarquia em sua organização, ver figura 1.

De acordo com esta organização pode-se conceituar cada um destes níveis.

Aplicação - é o conjunto de um ou mais processadores que perde a solução de um determinado problema.

Processador - são programas responsáveis pela inicialização dos gerenciadores e utilitários; definem a organização do banco de dados e ativam os blocos funcionais.

Blocos Funcionais - são rotinas onde se realizam as tarefas de manipulação da memória principal e secundária.

Rotinas de cálculo e procedimentos básicos - são aqueles códigos independentes da organização do banco de dados e que somente fazem uso dos procedimentos de rastreamento e tempo.

Gerenciadores e utilitários - são os procedimentos do SDP que permitem uma série de atividades tais como: gerenciamento de memória, rastreamento de rotinas, depurações, etc.

Biblioteca de elementos - são rotinas que fornecem parâmetros ou que acessam procedimentos específicos de um tipo de elemento.

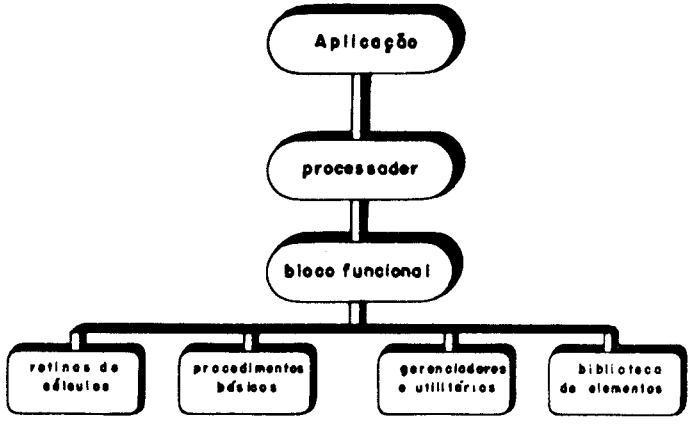

FIG. 1 Organização do sistema SDP.

RASTREAMENTO, DEPURAÇÃO E TRATAMENTO DE ERROS

Neste módulo estão disponíveis facilidades para a localização de erros durante a execução de programas e, principalmente, durante a implementação e teste de novas sub-rotinas. Estas facilidades são:

- rastreamento de sub-rotinas
- depuração e
- manipulação de mensagens de erro.

Rastreamento

O procedimento de rastreamento consiste em formar uma pilha com os nomes de todas as rotinas acessadas. Toda vez que uma sub-rotina é acessada uma função TRACE=ON é encarregada de colocar o nome desta rotina no topo da pilha. Uma outra função TRACE=OFF é responsável por retirar a rotina da pilha. Desta maneira, é possível obter em qualquer instante

da execução de um programa o fluxo das sub-rotinas acessadas. Esta sequência lógica é muito útil para a localização de erros e para o procedimento de depuração.

A partir de um código de controle, fornecido pelo usuário, pode-se ativar a impressão das mensagens "> TRACE ON nome" e "> TRACE OFF nome" toda vez que uma das funções ON e OFF for acessada.

Por exemplo:

```
SUBROUTINE SUB (X,Y,N)
DIMENSION X(N), Y(N)
TRACE=ON ('SUB    ')
       :
       :
TRACE=OFF ('SUB    ')
RETURN
```

Tratamento de Erros

Devido a modularidade do sistema SDP o tratamento dos erros deve ser feito de maneira comum a todas as sub-rotinas que compõem o sistema.

Toda vez que um erro for localizado pelo código deve-se ativar uma rotina ERROR, fornecendo-se:

- número do erro,
- mensagem de erro (até 32 caracteres) e
- um código de controle que poderá interromper a execução dependendo da gravidade do erro.

A rotina ERROR imprime além do número e da mensagem de erro, os nomes das rotinas acessadas até o momento.

É contabilizado o número de erros cometidos durante a execução de um programa. Se o programa for interrompido pela rotina EXIT será impresso o número de erros cometidos.

Por exemplo:

```
SUBROUTINE SUB (X,Y,N)
DIMENSION X(N), Y(N)
TRACE=ON ('SUB    ')
   :
   :
ICOD=2      código para parada do programa
IF(N.LE.0) CALL ERROR (ICOD,LW,1,
            'vetor c/ dimensão nula ou negat.')
   :
   :
TRACE=OFF ('SUB    ')
RETURN
END
```

Depuração

O procedimento de depuração consiste na impressão ou verificação de certos dados, o que não seria realizado durante a execução normal de um programa.

Os nomes das rotinas a serem depuradas são fornecidas para o sistema através da rotina INCDP. A sub-rotina DEBUG verifica se a rotina que está sendo executada (aquela que está no topo da pilha formada pelo procedimento de rastreamento) deve ser depurada.

Por exemplo:

```
      SUBROUTINE SUB (X,Y,N)
      DIMENSION X(N), Y(N)
      TRACE=ON ('SUB    ')
               :
               :
      CALL DEBUG (IDEPUR,LW)
      IF (IDEPUR.EQ.0) GO TO 900
      WRITE (LW,4000) (X(I), Y(I), I=1,N)
  900 TRACE=OFF ('SUB    ')
      RETURN
      END
```

Procedimentos Disponíveis

INICTR: inicializar o procedimento de rastreamento.
ON : ativar o procedimento de rastreamento de uma rotina.
OFF : desativar o procedimento de rastreamento de uma rotina.
INICDP: inicializar o procedimento de depuração. Este procedimento lê o número e os nomes das rotinas a serem depuradas (6 caracteres).
DEBUG : verificar se uma rotina deve ser depurada.
ERROR : imprimir mensagem de erro e, dependendo de um código, interromper o programa.
EXIT : terminar a execução de um programa imprimindo o número de erros detectados durante sua execução.

GERENCIAMENTO DA MEMÓRIA PRINCIPAL

A memória principal é constituída por um único vetor de trabalho - vetor A - e deve ser prevista no programa principal a seguinte especificação:

```
COMMON/VA/A (NDIM1)
```

onde MDIM1 é a dimensão do vetor de trabalho.

Vetores e matrizes podem ser armazenados como tabelas identificadas por um nome, obrigatoriamente de 4 caracteres, atribuído pelo programador. A partir deste nome, a manipulação das tabelas pode ser feita de forma automática pelo sistema SDP.

Tabelas podem ser criadas, removidas, terem seus comprimentos e nomes modificados através de uma manipulação simbólica.

Por exemplo, a criação de uma tabela é feita da forma:

```
CALL CREATE ('NOME', LONG, TIPO, IPOS)
```

onde NOME é o nome da tabela

LONG sua dimensão

TIPO tipo da tabela (real, inteira, dupla, etc.)

IPOS posição inicial da tabela no vetor de trabalho.

Para se eliminar esta tabela da memória principal deve ser executada o comando:

```
CALL REMOVE ('NOME')
```

Para o gerenciamento da memória principal é mantido um diretório com os seguintes parâmetros:

- tamanho da palavra que define o vetor de trabalho - em precisão simples -,
- última posição ocupada no vetor de trabalho,
- dimensão total deste vetor,
- número de tabelas ativas,
- número máximo de tabelas que podem ser armazenadas simultaneamente no vetor de trabalho,
- nomes das tabelas existentes no vetor de trabalho,
- posição inicial de cada tabela e
- tamanho de cada tabela medida em palavras de precisão simples.

Estes parâmetros são armazenados em uma especificação COMMON que deve ser prevista no programa principal.

Ao se utilizar o módulo de gerenciamento de memória principal se define uma hierarquia na programação. Existe um nível superior onde as tabelas necessárias são criadas ou manipuladas e um segundo nível onde estas tabelas são empregadas nos cálculos. Em todas as sub-rotinas que se enquadram no primeiro nível deve ser previsto um COMMON/VA/A (1).

No segundo nível as tabelas devem ser passadas através do vetor de trabalho da seguinte maneira:

```
COMMON/VA/A (1)
     :
     :
CALL CREATE ('TABE', N, 'REAL', ITTABE)
     :
     :
CALL SUB (A(ITTABE),...)
```

Nas sub-rotinas deste segundo nível não deve aparecer a especificação COMMON/VA/, ficando assim, estes procedimentos totalmente independentes dos gerenciadores da memória principal.

Procedimentos Disponíveis

Estão disponíveis os seguintes procedimentos para a manipulação da memória principal:

INICVA: inicializar o gerenciamento do vetor de trabalho.
CREATE: criar uma tabela dado o nome, tipo e dimensão.
AREA : criar uma tabela dado o nome, número de bytes da palavra e dimensão.
REMOVE: eliminar uma tabela do vetor de trabalho, deslocando o conteúdo das tabelas seguintes a esta.
SHIFT : deslocar para após de uma dada tabela, as tabelas compreendidas entre duas tabelas dadas, ambas inclusive.
LASTAB: eliminar todas as tabelas seguintes a uma dada tabela.
ENDTAB: eliminar todas as tabelas a partir de uma dada tabela, esta inclusive.
SHORT : atualizar o comprimento efetivo de trabalho de uma tabela.
LOCTB : localizar uma tabela no vetor de trabalho.
LAREA : imprimir uma lista dos nomes das tabelas ativas no vetor de trabalho.
CLEAR : limpar (zerar) uma tabela.
RENAME: mudar o nome de uma tabela.
NEWTAB: eliminar todas as tabelas do vetor de trabalho.

GERENCIAMENTO DA MEMÓRIA SECUNDÁRIA (EM DISCO)

As rotinas disponíveis no módulo de gerenciamento da memória secundária permitem gravar tabelas armazenadas no vetor de trabalho para um arquivo de disco e depois recuperá-las. Através destas rotinas é possível uma manipulação simbólica destas informações e o sistema SDP se encarrega da transferência das informações e do gerenciamento necessário.

O programador deve abrir um arquivo (sub-rotinas OPENI e OPENR) definindo seu nome (12 caracteres), número da unidade lógica, tipo (sequencial ou de acesso direto), número de tabelas que podem ser gravadas (somente quando se inicializa um arquivo). Se for um arquivo de acesso direto, deve ser fornecido o tamanho de cada registro e o número máximo de registros.

Para se gravar (sub-rotina SAVE) ou recuperar (sub-rotina RESTOR) tabelas, além do nome (4 caracteres) da tabela é necessário um outro identificador - IBLOCO - número da versão ou do bloco da tabela. Este procedimento de gravação não permite que uma mesma tabela seja gravada mais de uma vez em um arquivo. Assim este segundo identificador é necessário para se diferenciar tabelas de mesmo nome porém com informações divididas em grupos ou blocos.

Ao se terminar de trabalhar com um arquivo deve-se fechar este arquivo (sub-rotina CLOSE). Este procedimento grava as informações sobre o gerenciamento do arquivo e os nomes das tabelas gravadas, liberando a área em memória direta reservada para este gerenciamento. Estas informações são gravadas no início do arquivo e são fundamentais para uma utilização posterior do arquivo.

Existe um outro procedimento (sub-rotina RESERV) que atualiza as informações sobre o arquivo sem que sejam eliminadas estes dados da memória principal. Desse modo, pode-se continuar normalmente o trabalho com este arquivo, porém, em caso de interrupção do programa devido algum erro, as informações sobre o gerenciamento do arquivo não são perdidas.

Para o gerenciamento da memória secundária é mantido um diretório com as seguintes informações:

- número máximo de unidades lógicas que permite o sistema operacional da máquina,
- número máximo de arquivos (abertos simultaneamente) e
- número de arquivos que estão abertos.

Para cada arquivo, são mantidos os parâmetros:

- nome do arquivo,
- número da unidade lógica associada,
- tipo de acesso (sequencial ou acesso direto),
- tamanho do registro (acesso direto),
- próxima posição a ser gravada,
- número de tabelas gravadas e
- número máximo de tabelas que podem ser gravadas.

E é mantido em gerenciamento as informações sobre as tabelas gravadas em cada arquivo:

- nome das tabelas,
- comprimento efetivo e
- posição da tabela no arquivo.

Estas informações são armazenadas em duas especificações COMMON que devem ser previstas no programa principal.

Estrutura do arquivo

Um arquivo, sequencial ou de acesso direto, armazena as tabelas do banco de dados. Além disso, a sua estrutura permite manter gravadas as informações sobre seu gerenciamento (diretório). Assim um arquivo de dados é autocontido, facilitando-se a comunicação entre programas, ver figura 2.

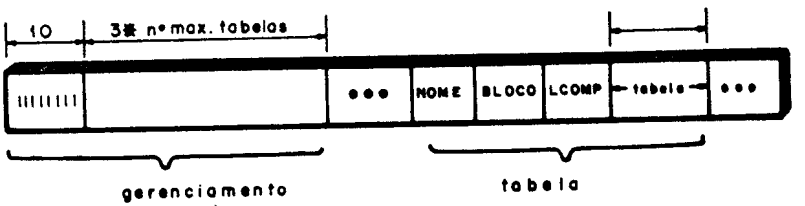

FIG. 2 Estrutura de um arquivo.

Procedimentos Disponíveis

Os procedimentos disponíveis para a manipulação da memória secundária são:

INICFL: inicializar o gerenciamento dos arquivos.
OPENI : abrir (inicializar) um arquivo sequencial ou de acesso direto. Reservar espaço para o gerenciamento.
OPENR : abrir um arquivo já existente. Ler as informações sobre o gerenciamento.
SAVE : gravar uma tabela em um arquivo.
LOAD : ler uma tabela de um arquivo.
RESTOR: recuperar uma tabela que está gravada em um arquivo.
RESAVE: regravar (na mesma posição) uma tabela em um arquivo.
RESERV: gravar as informações sobre o gerenciamento no início do arquivo.
CLOSE : fechar (encerrar) um arquivo. Gravar as informações sobre o gerenciamento no início do arquivo. Libera o espaço reservado para o gerenciamento.
LTFILE: listar as tabelas que estão gravadas em um arquivo.
LFILES: listar os arquivos que estão sendo utilizados.

GERENCIAMENTO DO TEMPO DE EXECUÇÃO

Através de certas rotinas do SDP é possível obter o tempo de execução de uma determinada parte do programa:

- tempo de cpu em uma etapa,
- tempo decorrido em uma etapa,
- tempo total de cpu até uma etapa,
- tempo total decorrido até uma etapa,
- tempo total de cpu e o
- tempo total decorrido no programa.

Estes valores podem ser armazenados e listados ao final do programa.

Procedimentos Disponíveis

ADATA : fornecer a data, hora, dia, mês e ano.
ITEMPO: inicializar contagem de tempo das etapas.
TEMPO : fornecer o tempo de execução de uma etapa.
LTEMPO: listar os tempos de execução das etapas.

DOCUMENTAÇÃO

Como já mencionado, um dos objetivos do SDP é o de contribuir para uma maior cooperação entre as instituições que elaboram "software" técnico-científico. Para conseguir tal intento, o SDP dispõe de uma estrutura para a documentação que além de apresentar informações sobre os seus procedimentos e a maneira de utilizá-los, possui facilidades para acessar estas informações.

A documentação do sistema SDP esta constituída por:

- manuais que descrevem o sistema,
- normas de programação e documentação,
- documentação de todos os procedimentos disponíveis e
- programas utilitários para manipular a documentação.

A documentação de um procedimento [3] é o conjunto de todas as informações necessárias para sua correta utilização e compreensão. Visa atender as necessidades de usuários e programadores fornecendo informa-

ções relativas a sua utilização e detalhes de programação. Esses procedimentos podem ser programas aplicativos, programas utilitários, blocos funcionais, sub-rotinas ou funções.

Esta documentação deve conter informações sobre a finalidade, entrada e saída de dados, programação, organização e estrutura de dados, de tal forma que o usuário ou programador manipule rápida e inequivocamente o procedimento, conforme seu interesse, sem necessidade de recorrer a outros manuais.

O Sistema SDP fornece dois esquemas para documentação:

- para blocos funcionais, sub-rotinas ou funções, e
- para programas aplicativos e programas utilitários.

A documentação de um bloco funcional, sub-rotina ou função segue uma padronização composta de 15 itens, a saber:

(01) Objetivo
(02) Forma de acesso
(03) Descrição
(04) Palavras-chave
(05) Referência
(06) Data e autor
(07) Argumentos
(08) Entrada e saída de dados
(09) Dimensões
(10) Especificações COMMON e DATA
(11) Chamadas
(12) Mensagens de erro
(13) Depuração
(14) Variáveis empregadas
(15) Lógica do procedimento

Os itens 1 a 13 são destinadas ao usuário, ou seja, aquele que se interessa apenas em utilizar determinado procedimento. O programador que deseja conhecer detalhes da programação deve também procurar as informações contidas nos itens 14 e 15.

A documentação de programas segue uma padronização muito semelhante a anterior, sendo composta por 12 itens:

(01) Objetivo
(02) Forma de acesso
(03) Descrição
(04) Data e autor
(05) Arquivos utilizados
(06) Estrutura do banco de dados
(07) Especificações COMMON e DATA
(08) Chamadas
(09) Mensagens de erro
(10) Depuração
(11) Variáveis empregadas
(12) Lógica do procedimento

Os programas utilitários para manipulação da documentação permitem:

- fornecer no terminal a documentação de uma rotina,

- localizar uma rotina a partir de palavras-chave,
- fornecer manual com a documentação de todos os procedimentos,
- incluir a documentação de um novo procedimento e
- modificar a documentação já existente.

APLICAÇÕES

É apresentada uma aplicação desenvolvida com o auxílio do SDP que permite a análise de problemas lineares tais como análise elasto-estática de estruturas, condução de calor segundo Lei de Fourier, etc.

De acordo com a modulação proposta pelo SDP, esta aplicação é constituída por processadores (programas), blocos funcionais, rotinas utilitárias, de cálculo e biblioteca de elementos.

Uma estrutura básica para esta aplicação pode ser constituída pelos seguintes programas:

PRE - leitura de dados (formatados): coordenadas, incidência, propriedades, carregamentos, condições de contorno e variáveis prescritas;
PRO - cálculo das matrizes e solicitações elementares;
SOL - montagem e solução do sistema de equações lineares segundo topologia correspondente a matriz global escalonada (skyline) armazenada em um único vetor de trabalho por colunas ascendentes.

Por sua vez, cada um destes processadores são constituídos por blocos funcionais. Como exemplo, o pré-processador de leitura formatada é constituído pelos seguintes blocos funcionais:

BFCOOR - construção da tabela de coordenadas (leitura e geração de dados);
BFPROP - construção das tabelas de propriedades
- incidência (leitura),
- parâmetros do grupo (leitura),
- parâmetros do elemento (biblioteca BELPAR),
- graus de liberdade dos nós do elemento (biblioteca BELGL),
- índice do material (leitura),
- propriedades (leitura) e
- propriedades adicionais (leitura através da biblioteca BELPAD);

BFECNE - construção das tabelas de carregamentos:
- parâmetros do grupo carregado,
- cargas nodais (leitura) e
- a nível de elemento (leitura através da biblioteca BELECE);

BFIDE - construção das tabelas de condição de contorno;
BFEVP - construção das tabelas de variáveis prescritas (leitura).

O processador de cálculo das rotinas de rigidez e das solicitações elementares é constituído por:

BFPXYZ - preparação de tabelas de coordenadas para o cálculo das matrizes elementares;
BFRIG - cálculo das matrizes dos grupos de elementos (biblioteca BELRIG);

BFCCE - cálculo das solicitações elementares (biblioteca BELCCE).

O processador de montagem e resolução do sistema de equações é constituído por:

BFPLM - preparação de tabelas do número de equação associada a um grau de liberdade;
BFTOP5 - cálculo da topologia da matriz global - matriz escalonada armazenada em um vetor por colunas ascendentes;
BFMON5 - montagem das matrizes elementares de acordo com a topologia anterior;
BFMCNE - montagem das solicitações nodais e elementares;
BFMVP - imposição de valores prescritos;
BFDEC3 - decomposição da matriz global (Gauss);
BFSOL3 - resolução do sistema de equações - retrosubstituição;
BFLSOL - listar a solução.

Estes três processadores permitem a leitura dos dados e a montagem e solução do sistema de equações de acordo com uma dada topologia. Além destes processadores estão disponíveis outros programas:

PRE1 - leitura de dados (não formatados);
SOL1 - montagem e solução do sistema de equações lineares segundo topologia correspondente a matriz global armazenada em 2 vetores que contem a diagonal e a parte superior por colunas descendentes;
SOL2 - montagem e solução do sistema de equações lineares segundo topologia correspondente a matriz global armazenada em um vetor que contem a diagonal sendo que a parte superior é dividida em blocos e armazenada em memória secundária;
POS - pós-processador para análise de resultados - em preparação;
PLT2 - desenho de malhas planas.

Como pode ser notado, alguns blocos funcionais acessam rotinas dependentes do tipo de elemento. Estas rotinas estão agrupadas em bibliotecas reconhecidas como BELxxx (por exemplo BELPAR, BELGL, BELRIG, etc). Estes blocos funcionais foram desenvolvidos de modo a permitir a substituição destas bibliotecas por outras de maneira totalmente transparente para o usuário do procedimento. Isto possibilita empregar toda a estrutura anterior em diferentes problemas bastando somente fornecer o nome das bibliotecas que serão ativadas pelo sistema.

CONCLUSÕES

A utilização do SDP mostrou que este sistema é uma ferramenta poderosa para o desenvolvimento de programas baseados no MEF. Ao se estabelecer uma padronização para a programação, organização de dados e documentação, cria-se uma linguagem comum entre os grupos que se propõem a elaborar "software" técnico-científico.

Por sua vez, pode-se destacar que o sistema o SDP permite diminuir o tempo de preparação de programas aplicativos, minimizar a ocorrência de erros e facilitar sua localização, e proporcionar uma estrutura modular a todo programa desenvolvido dentro do ambiente do SDP, permitindo que estes procedimentos passem a ser parte do sistema.

AGRADECIMENTOS

Os autores agradecem à FIPEC-Fundo de Incentivo à Pesquisa Científica do Banco do Brasil - pelo apoio econômico outorgado ao projeto "Desenvolvimento de Software em Engenharia Mecânica para Mini e Microcomputadores".

REFERÊNCIAS

[1] ———, "O Sistema SDP", Relatório Interno, Laboratório Nacional de Computação Científica, Rio de Janeiro, 1985.

[2] Dahl, O.J., Dijkstra, E.W. & Hoare, C.A.R., "Structured Programming", Academic Press, London, 1972.

[3] ———, "SDP: Normas para Documentação", Relatório Interno, Laboratório Nacional de Computação Científica, Rio de Janeiro, 1985.