UNIVERSIDADE ESTADUAL DE CAMPINAS

FACULDADE DE ENGENHARIA ELÉTRICA

Este exemplar corresponde à redação final da tese defendida por Beatriz Mascia Daltrini e aprovada pela Comissão julgadora em 12/12/86

METODOLOGIA PARA IMPLEMENTAÇÃO DE SISTEMAS PAC RECONFIGURÁVEIS

BEATRIZ MASCIA DALTRINI

Autor

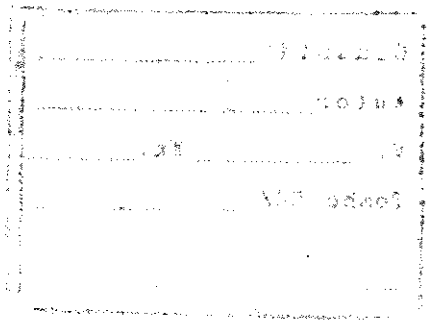
MARCIO LUIZ DE ANDRADE NETTO

Orientador

da à Faculdade de Engenharia Elétrica / UNICAMP como

isitos exigidos para a obtenção do título de Doutor em

trica.



À minha mãe.

Ao meu pai.

RESUMO

Este trabalho aborda dois itens distintos porem interrelacionados. O primeiro e uma proposta de metodologia para implementacao de Sitemas PAC reconfiguraveis. O segundo refere-se com o fato de capacitar este Sistema PAC com caracteristicas que possibilitem a execucao de comandos em paralelo, permitindo que um processamento de longa duracao seja executado como tarefa de fundo, enquanto o usuario trabalha por exemplo, em uma tarefa interativa.

Para a obtencao de um Sistema PAC reconfiguravel, primeiramente sao definidos os componentes do Sistema, a saber: O Supervisor e os modulos logicos Biblioteca de Aplicacao, Banco de Dados, Pacote Grafico, Interface com o Usuario e Utilidades. Os modulos logicos correspondem a areas de conhecimento que se encontram em constante evolucao e, portanto, passiveis de serem redefinidos. A partir dos modulos logicos e definido o Nucleo PAC, composto de grupos de funcoes basicas que definem uma interface para com os modulos logicos. Faz parte tambem do Nucleo PAC, um grupo de funcoes que se referem ao Sistema Operacional, permitindo que o Sistema PAC seja independente da instalacao. Nao faz parte do trabalho a definicao das funcoes do Nucleo PAC, porem no decorrer do trabalho, sao referenciadas algumas funcoes necessarias.

Para que o Sistema PAC permitisse comandos concorrentes, o Supervisor foi divido em quatro sub-módulos que correspondem às seguintes tarefas: Inicialização, Finalização, Geração de comando e Execução de comando. O módulo Interface com o Usuário foi dividido em Interface para comandos e Interface para dados. O fluxo de controle e dados é apresentado através do modelo GMB. O comando do usuário é composto de tarefas que são ativadas e sincronizadas. O trabalho apresenta os esquemas para ativação, sincronização e passagem de parâmetros. São apresentados também as especificações para o Gerador de comandos e o Executor de comandos.

ÍNDICE

CAPÍTULO 1 - INTRODUÇÃO

Desde o advento do computador o homem vem tentando utilizar a máquina em seu benefício, repartindo as tarefas de uma maneira racional, deixando ao homem aquelas que envolvam criatividade e intuição e à máquina a função de realizar cálculos e tomar decisões simples. Desta maneira o histórico de Sistemas de Projeto Auxiliado por Computador (PAC) se confunde com o próprio histórico da "máquina de calcular" /Alla 72/.

1.1 HISTÓRICO

Inicialmente os computadores eram máquinas muito simples, cabendo ao homem a maior parte das tarefas. Atualmente muitas tarefas que eram realizadas pelo homem podem ser realizadas pela máquina, devido ao grande progresso alcançado nas áreas de hardware, software ou mesmo com o desenvolvimento de técnicas de Inteligência Artificial, permitindo por exemplo que a máquina aprenda com o homem, através de um processo de inferência.

Assim, do mesmo modo que os computadores cresceram em capacidade e complexidade, os Sistemas PAC também o fizeram.

Grande parte do desenvolvimento ocorrido na área de hardware, notadamente em relação aos periféricos gráficos, foi decorrência da necessidade de suprir facilidades aos Sistemas PAC.

Pode-se considerar que o primeiro Sistema PAC, como concebido atualmente, foi o SKETCHPAD, desenvolvido em 1963, onde Sutherland, seu criador, introduziu os terminais gráficos interativos desenvolvidos na época no MIT /ENCA 83/.

Paralelamente ao hardware houve um grande desenvolvimento na área de software como por exemplo o aparecimento de pacotes gráficos, linguagens voltadas para gráficos, ou mesmo Sistemas Operacionais que permitem uma utilização mais eficiente de Sistemas PAC.

Inicialmente os Sistemas PAC eram sistemas voltados para desenho, onde o projeto em si não era automatizado e sim a obtenção dos desenhos. Atualmente os Sistemas PAC cobrem todo o projeto, desde a aquisição de dados, passando pelos cálculos de projeto, simulação de resultados, até os desenhos finais.

A evolução dos Sistemas PAC, da fase de desenvolvimento experimental para a fase comercial, se deu paralelamente à fixação dos fundamentos da área de estudos e projetos de Sistemas PAC.

Os primeiros sistemas comerciais demonstraram ser ferramentas fundamentais, permitindo a obtenção de projetos em menor tempo, com menor custo e maior precisão.

Esta característica foi o principal fator para a difusão de Sistemas PAC em diversas áreas de aplicação, abrangendo desde projetos navais até microeletrônica. Os Sistemas PAC então desenvolvidos eram dirigidos para a aplicação, sua implementação era baseada em técnicas e ferramentas específicas da aplicação.

No fim dos anos 70 já existiam empresas que vendiam Sistemas PAC para várias aplicações. Por esta época pode-se distinguir dois tipos de Sistemas PAC: os chamados "turn-keys" e os Sistemas abertos.

Os Sistemas abertos, geralmente desenvolvidos pela própria companhia que os utilizariam, permitiam que os projetistas do sistema fizessem modificações, introduzindo melhoramentos que advinham da constante evolução tanto na área de hardware, como na área de software.

Estas mesmas companhias passaram também a vender seus Sistemas PAC para outras companhias, sempre como um pacote fechado, que se dirigiam à uma aplicação específica. Paralelamente, fabricantes de equipamentos de computação e "software houses" entraram no mercado de Sistemas PAC, disseminando os sistemas "turn-keys". Estes sistemas "turn-keys" eram pacotes fechados, vendidos na forma de código objeto, não permitindo desenvolvimentos ou modificações. Pela sua própria característica eram comercializados juntamente com o sistema de computação e periféricos gráficos.

Se para os "turn-keys" a introdução de modificações era impossível, para os sistemas abertos, apesar de possível, poderia tornar-se inviável pelo esforço requerido. De forma a se obter sistemas abertos nos quais a introdução de modificações fosse facilitada, foi desenvolvida uma metodologia, para implementação de Sistemas PAC, que resultou no conceito de sistemas integrados.

O Sistema Integrado é um sistema modular composto de um núcelo gerenciador e vários subsistemas com tarefas específicas porém comuns a todos Sistemas PAC, como gerenciador de arquivos, pacote gráfico, etc. O primeiro sistema integrado desenvolvido foi o ICES (Integrated Civil Engineering System) no fim dos anos 60 /ENCA 83/.

Com o conceito de sistema integrado foi possível desenvolver Sistemas PAC reconfiguráveis. O IPAD /GARR 75/, por exemplo, apesar de ter sido projetado para aplicações aeroespaciais, poderia tornar-se um Sistema PAC para qualquer aplicação, pois sendo um sistema integrado, sua implementação foi baseada em conceitos gerais de Sistemas PAC, que independem da aplicação.

A partir dos anos 70 foram desenvolvidos vários sistemas integrados como o DINAS, o GENESYS e o REGENT.

O REGENT /SCHL 80/, por exemplo, é um sistema integrado composto de:

- um Núcleo,
- um subsistema de manipulação de arquivos,
- um subsistema de processamento gráfico,
- vários subsistemas orientados para aplicação.

1.2 DESCRIÇÃO DOS COMPONENTES DE UM SISTEMA PAC

A partir dos sistemas integrados pode-se definir quais os subsistemas que deverão compor o Sistema PAC /SCHL 80/ . A seguir será feita uma análise dos componentes básicos de um Sistemas PAC, com uma descrição do "estado da arte" de cada um deles.

-SUPERVISOR

O Sistema PAC deve possuir um Supervisor cujas funções serão equivalentes às funções do núcleo nos sistemas integrados. Ele deverá trabalhar sobre os recursos da máquina da instalação e terá o papel de gerenciador na interligação entre os subsistemas. O Supervisor será mais ou menos complexo na medida dos recursos requeridos para o Sistema PAC. Exemplos das funções do Supervisor podem ser encontrados nos sistemas integrados IPAD /GARR 75/ e DINAS /BEIE 78/, podendo ser citadas funções de controle de tarefas, gerenciamento de memória, comunicação entre tarefas, entre outras.

-BANCO DE DADOS

O Sistema PAC deverá possuir uma base de dados central onde estarão armazenados dados e resultados do processo de projeto. Isto não exclui a possibilidade do Supervisor ou de cada subsistema possuir sua própria base de dados. Os dados armazenados no Banco de Dados serão dados compartilhados pelo Supervisor e subsistemas.

Banco de Dados é uma área de pequisa em grande evolução. Existem modelos de dados bem definidos /DATE 76/ que entretanto não se adaptam perfeitamente à um ambiente PAC, pois foram desenvolvidos para aplicações comerciais.

Algumas discrepâncias que afetam a eficiência destes modelos são /VERN 84/ :

- em um projeto, o número de objetos é muito grande com poucas ocorrências de cada objeto; em aplicações comerciais o número de objetos é pequeno com um grande número de ocorrências

- em um projeto as associações entre objetos é muito mais complexa se comparadas com aplicações comerciais

- em um projeto tem-se tipos de dados complexos, enquanto que em aplicações comerciais tem-se tipos de dados básicos (inteiro, real, caracter...).

Com a evolução dos Sistemas PAC grupos de pesquisadores se dedicam ao estudo de modelos de estruturas de dados para PAC /VERN 84/, utilizando metodologias como Entidade-Relacionamento /CHEN 76/, e as desenvolvendo para melhor adaptá-las aos requisitos de um ambiente PAC /DELG 84/ /DELG 85/, /NEUM 80/, /NEUM 82/, como por exemplo :

- O banco de dados deve permitir diferentes representações de um mesmo objeto, sem perda de consistência. Estas diferentes representações têm a finalidade de permitir a definição de estruturas de dados mais compatíveis com as ações tomadas pelos diferentes processos que coexistem em um Sistema PAC (cálculo, saída gráfica,...).

- Em um projeto, as transações (período em que o banco de dados está inconsistente) podem ter duração de dias, havendo portanto a necessidade de um gerenciador de versões, que representam as alternativas de projeto.

- O sistema de banco de dados deverá prover primitivas de modelamento adequadas para a definição de esquemas complexos, que incorporem as especificações da aplicação, tais como regras de integridade e consistência.

A pesquisa em Banco de Dados para PAC tem se desenvolvido no sentido de excluir do programa de aplicação ou do Supervisor, qualquer procedimento que possa ser gerenciado pelo Banco de Dados /ENCA 83/, como por exemplo:

-acesso aos dados,

-manutenção da consistência dos dados,

-eliminação de redundâncias,

-permissão de acesso,

-segurança, etc.

Deste modo o subsistema Banco de Dados pode ser utilizado como um módulo fechado, cuja interação com os outros subsistemas, se restringe às funções básicas de recuperação e armazenamento em uma base de dados.

-PACOTE GRÁFICO

O Sistema PAC, como hoje é concebido, apareceu com o desenvolvimento de periféricos gráficos. Tanto quanto o Banco de Dados, o Pacote Gráfico é um dos principais componentes de um Sistema PAC /GILO 78/. Isto devido principalmente à interface amigável que um dispositivo gráfico oferece ao homem /FOLE 74/.

Na área de hardware, houve um enorme desenvolvimento nos últimos 20 anos, evoluindo do "plotter" para os modernos "raster displays". Atualmente existem diferentes dispositivos de saída gráfica para várias aplicações como plotters para obtenção de desenhos ou displays de alta resolução, utilizados na interação com o homem durante o processo de projeto.

Os dispositivos de entrada gráfica evoluiram permitindo uma interação mais fácil e precisa. Atualmente uma estação de trabalho conta com diversos dispositivos de entrada gráfica como digitalizadores, tablet, light-pen, etc.

Na área de software foram desenvolvidas várias linguagens gráficas ou extensões para as linguagens existentes.

A padronização de um pacote gráfico que desse a característica de portatilidade em relação aos periféricos utilizados foi uma preocupação desde cedo /GARR 75/. Até há bem pouco tempo, os pacotes gráficos faziam parte do Sistema de Computação da instalação, sendo geralmente implementados para acessar um periférico gráfico específico do fabricante. Deste modo a introdução de um novo periférico gráfico ficava extremamente difícil. Para resolver o problema, diferentes grupos de pesquisadores desenvolveram propostas de padrões gráficos, podendo ser citados GSPC-CORE, GKS, PHIGS /MAGA 86/ entre outros. A proposta de padrão gráfico GKS /GKS 82/ tornou-se norma internacional em 1982.

Com a finalidade de obter uma imagem cada vez mais real, os pesquisadores se dedicaram ao desenvolvimento de teorias para linhas escondidas (hidden lines), sombreamento, geração de superfícies, modelamento de sólidos, etc., fazendo com que a computação gráfica gerasse diversas sub-áreas de pesquisa, inclusive ligando-se a outras áreas como na pesquisa de Banco de Dados para Computação Gráfica /ENCA 79/.

-DIÁLOGO

Um Sistema PAC pode ser uma excelente ferramenta de projeto, porém se não possuir uma interface homem-máquina amigável, não será utilizado /BO 80/. O projeto de interfaces homem-máquina amigáveis tem sido objeto de estudos há muitos anos /MART 73/; a própria evolução da computação gráfica pode ser considerada como decorrência da necessidade de aperfeiçoar a interação com o usuário.

Não se deve porém confundir o processamento gráfico com interface com o usuário, esta deverá prover muito mais recursos além da exibição gráfica de resultados. Exemplos destes recursos são: informação dos erros ocorridos ao usuário, recuperação adequada em estados de erros, adaptação do diálogo à experiência do usuário, etc.

O muito que se tem evoluido na área de interfaces homem-máquina esbarra na dificuldade de se modelar o comportamento humano. Apesar disto, certos parâmetros, como o tempo de resposta do

usuário à um comando do computador, têm se mostrado úteis quando utilizados em diálogos adaptativos /TOZZ 79/, onde diferentes módulos de diálogo são ativados automaticamente, dependendo da experiência do usuário com o Sistema. Para resolver este problema existem propostas de pesquisadores na área de Inteligência Artificial /OHSU 79/.

Nos últimos anos, diferentes grupos de pesquisadores abordaram o tema de estruturas de diálogo, onde são formalizadas as diferentes operações básicas do diálogo e o seu sequenciamento, permitindo a definição formal do fluxo de informação que compõe o diálogo. Como exemplo, pode ser citado o modelo denominado Células de Diálogo /BORU 82/. Estes modelos deixam a cargo da implementação os recursos utilizados no diálogo, podendo este, se necessário para a aplicação, incorporar recursos gráficos /SILV 85/, /BORU 80/.

-UTILITÁRIOS

Os Sistemas PAC deverão prover pacotes de utilitários que agrupem ferramentas matemáticas, estatísticas, de simulação e outras que podem ser utilizadas pela aplicação, sendo porém não específicas da aplicação, mas sim rotinas de caráter geral. Em alguns Sistemas PAC, verifica-se que estes pacotes não são mencionados (DINAS), sendo provavelmente considerados como fazendo parte da aplicação; em outros eles são mencionados explicitamente (IPAD).

-APLICAÇÃO

é considerado como pertencente ao subsistema aplicação, qualquer procedimento ou pacote específico da aplicação para a qual o Sistema PAC foi configurado.

O subsistema aplicação pode não ser único (REGENT), a aplicação pode estar dividida em vários subsistemas, sendo cada um deles, pacotes que realizam uma determinada parte da tarefa, como por exemplo, a análise de elementos finitos /SCHL 80/.

O subsistema aplicação é mencionado na bibliografia como Biblioteca de Aplicação ou Banco de Métodos /ENCA 77/.

1.3 DESCRIÇÃO DO TRABALHO

Todos os Sistemas PAC possuem os componentes básicos descritos. A diferença entre os diversos Sistemas existentes está em como estes componentes básicos se interligam e quais destes são considerados como ferramentas básicas fornecidas pelo Supervisor, ou seja, quais subsistemas não são considerados como subsistemas e sim como parte do Supervisor.

O principal objetivo do trabalho é determinar um modelo para Sistemas PAC de tal forma que este seja facilmente reconfigurável. Para tanto, no capítulo 2 é discutido o modelo lógico para Sistemas PAC sendo apresentados os subsistemas propostos, o Supervisor e o fluxo de dados e controle no sistema.

No capítulo 3 é apresentado o conceito de Núcleo PAC que vem solucionar problemas oriundos da modularidade proposta no capítulo 2.

O capítulo 4 é dedicado a um estudo de paralelismo, uma vez que o Sistema PAC proposto deverá ter a capacidade de executar vários comandos do usuário em paralelo.

No capítulo 5 é dado um exemplo e são apresentados esquemas para a implementação dos principais módulos do Supervisor. A seguir, no capítulo 6, o fluxo de controle do sistema é modelado através de redes de Petri para o estudo de possíveis "deadlocks". O capítulo 7 apresenta as conclusões.

CAPÍTULO 2 - MODELO LÓGICO DE UM SISTEMA PAC

O modelo de um Sistema PAC reconfigurável é apresentado através da descrição de seus componentes e sua interligação. Cada componente é analisado quanto aos requisitos necessários para manter o sistema modular e reconfigurável. É apresentado um guia para a instalação do sistema proposto e a seguir é feita uma análise do impacto produzido pela substituição de um dos componentes

Conforme exposto no Cap. 1 a implementação de um Sistema PAC envolve o conhecimento de várias áreas que se encontram em constante evolução, fazendo com que o aperfeiçoamento de Sistemas PAC esteja ligado aos aperfeiçoamentos alcançados nestas áreas. Nos exemplos citados, IPAD e DINAS, pode-se observar que o EXECUTIVO (Supervisor) incorpora funções que se referem a estas áreas de pesquisa. Tanto o Executivo do IPAD como do DINAS incorporam o processamento do diálogo; o Executivo do DINAS incorpora ainda o gerenciamento da base de dados.

Muito pouco se encontra na literatura com respeito à evolução dos sistemas integrados. É sabido que grande parte dos Sistemas PAC implementados não são reconfiguráveis, e por este motivo, a bibliografia menciona principalmente os esforços desenvolvidos no sentido de se integrar pacotes já consagrados pelo uso, gerando Sistemas PAC que somem as capacidades de outros já desenvolvidos.

O objetivo deste trabalho é desenvolver uma metodologia para projeto de Sistemas PAC reconfiguráveis em uma primeira fase e, a seguir, incorporar a capacidade de gerenciar processos concorrentes, mantendo a característica de sistema reconfigurável.

O Modelo Lógico de um Sistema PAC, proposto a seguir, é baseado no conceito de sistemas integrados e tem como finalidade principal, permitir que o Sistema seja reconfigurável, de tal modo que os aperfeiçoamentos alcançados em uma área de pesquisa possam ser incorporados.

Para que um Sistema PAC seja reconfigurável é necessário que seja modular, ou seja, o Sistema PAC deverá ser composto de módulos independentes, que se comunicam através de interfaces, de tal forma que a substituição de um módulo não cause impacto no restante do Sistema. A definição dos módulos componentes do Sistema deve ser tal que um módulo seja uma unidade lógica coerente, ou seja, o módulo lógico é um agrupamento de funções correlatas. Estes módulos são pacotes de software, passíveis de serem substituidos individualmente e possivelmente encontrados à venda no mercado.

2.1 MÓDULOS INDEPENDENTES

Esta independência entre os módulos implica que em um módulo não haja referências a outro módulo. Os procedimentos constantes de qualquer módulo deverão ser autocontidos, com interfaces prédefinidas de entrada e saída, sendo toda a comunicação entre procedimentos de módulos diferentes, feita através destas interfaces /ENCA 83/.

A transferência de dados entre procedimentos será feita através de drivers, que transformam o formato de saída, definido pela interface de saída de um procedimento, no formato de entrada, definido pela interface de entrada do outro procedimento (Fig. 2.1).

Fig. 2.1 - Interligação de procedimentos

2.2 - SISTEMA PAC MODULAR

Com base no raciocínio acima e nos estudos relatados no Cap. 1, definimos o Sistema PAC como composto dos seguintes módulos lógicos, que serão chamados de subsistemas :

- um **Supervisor**, que tem como função principal, gerenciar a comunicação entre os subsistemas;
- um subsistema **Biblioteca de Aplicações** que agrupa os programas de aplicação;
- um subsistema **Gráfico** que agrupa funções gráficas;
- um subsistema **Banco de Dados** que agrupa funções para o gerenciamento de uma base de dados;

-um subsistema **Interface com o Usuário** que agrupa funções de diálogo e

-um subsistema **Utilidades** que agrupa ferramentas de uso geral.

A fig. 2.2 a seguir, mostra o modelo lógico de um Sistema PAC, baseado em subsistemas independentes, interligados via Supervisor.

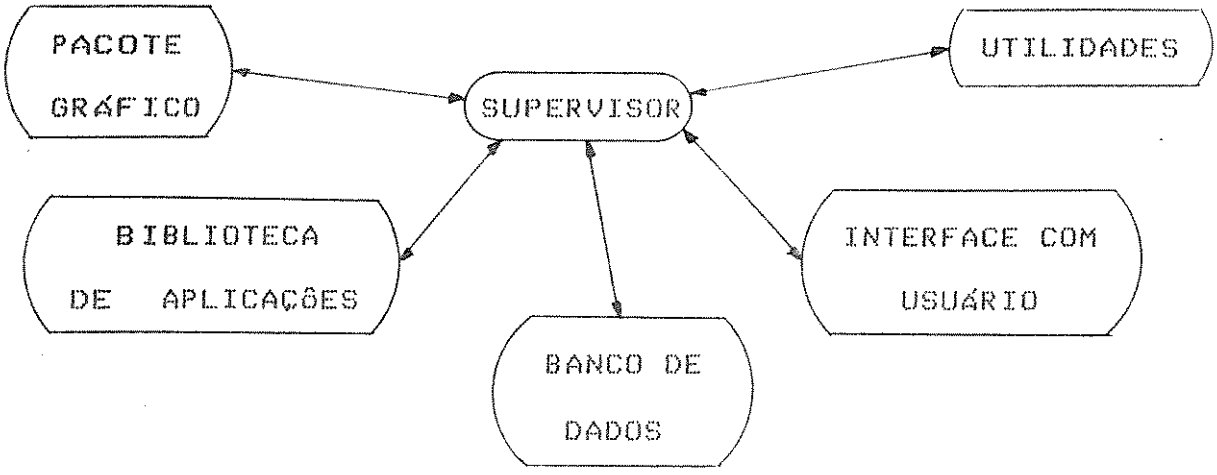


Fig. 2.2 - Modelo Lógico de um Sistema PAC

A seguir será feita uma descrição funcional do Sistema PAC, descrevendo-se o Supervisor, os diversos subsistemas e a interligação de dados e controle através do Supervisor e subsistemas.

2.3 - SUBSISTEMA BIBLIOTECA DE APLICAÇÕES

O subsistema Biblioteca de Aplicações é composto de programas de aplicação que podem ser desenvolvidos pelo usuário, ou que fazem parte de um ou mais pacotes já desenvolvidos e disponíveis no mercado. Alguns sistemas integrados possuem diversos subsistemas de aplicação. No modelo proposto este subsistema é único pois os subsistemas foram definidos como grupamentos de funções correlatas, ou seja, de acordo com seu aspecto funcional.

Este enfoque permite uma maior simplicidade para o Supervisor padronizando o tratamento dos subsistemas, porém será necessária a implementação de um gerenciador no subsistema Biblioteca de Aplicação caso haja necessidade de se interligar diferentes pacotes de software (Fig. 2.3).

A bibliografia apresenta soluções para a interligação de pacotes de software específicos /SPAN 84/, através da implementação de um gerenciador também específico. Um estudo mais aprofundado no assunto poderia determinar a possibilidade de desnvolver um gerenciador de caráter geral que permitisse a interligação de dferentes pacotes de software. Este trabalho não se aprofunda neste item porém sugere, como ponto de partida, o modelo lógico do Sistema PAC, substituindo-se o Supervisor pelo gerenciador e os módulos lógicos pelos pacotes de software.

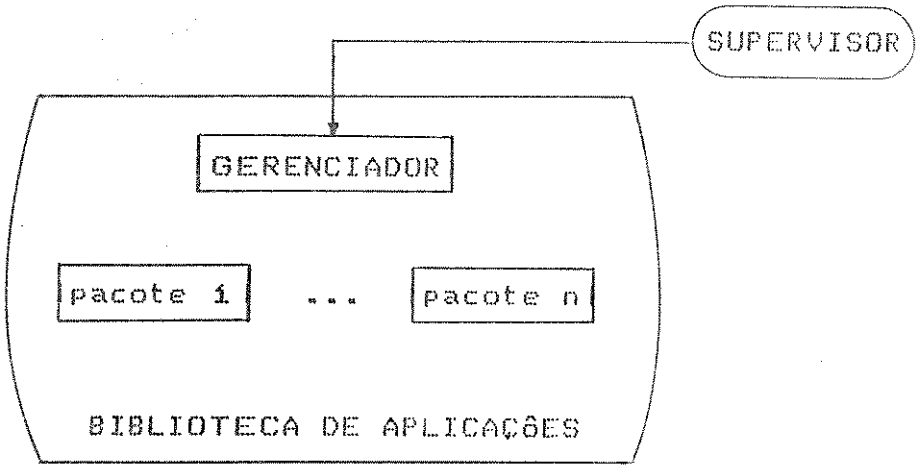


Fig. 2.3 – Subsistema Biblioteca de Aplicações com gerenciador

2.4 - SUBSISTEMA PACOTE GRÁFICO

O subsistema gráfico agrupa funções gráficas de entrada e saída, funções de gerenciamento do pacote gráfico e se necessário uma estrutura de dados interna /GKS 82/. O padrão gráfico GKS pode ser considerado como um protótipo para o subsistema gráfico pelos princípios usados para sua estrutura, tais como "Independência da Aplicação" e "Independência do dispositivo gráfico".

2.5 - SUBSISTEMA UTILIDADES

O subsistema Utilidades agrupa ferramentas matemáticas, estatísticas, de simulação e outras, como por exemplo, programa de inversão de matrizes, que podem ser desenvolvidas pelo usuário ou então fazer parte de pacotes disponíveis no mercado.

2.6 - SUBSISTEMA INTERFACE COM O USUÁRIO

O subsistema Interface com o Usuário contém os programas responsáveis pela comunicação com o usuário e está intimamente ligado ao subsistema Biblioteca de Aplicações, pois os comandos definidos para a Interface com o Usuário serão baseados nos recursos existentes na Biblioteca de Aplicações, ou seja, nos programas de aplicação.

Porém a Interface com o Usuário deverá prover também comandos de gerenciamento do Sistema, comandos para dar acesso aos recursos gráficos (subsistema Gráfico) ou à base de dados (subsistema Banco de Dados). Isto implica a existencia de uma Interface com o Usuário que agrupe os procedimentos relativos aos comandos de diálogo de caráter geral (que se aplicam à comunicação do usuário com o Sistema PAC) e procedimentos relativos aos comandos específicos da aplicação.

Interfaces com o Usuário é um tema de estudo em grande evolução, devido à exigencia atual de interfaces amigáveis, que deram origem a diálogos adaptativos e à formalização de estruturas sofisticadas de diálogo. Porém, qualquer que seja a estrutura de diálogo utilizada, ela irá acessar as tabelas que definem os recursos do sistema, ficando dependente destes apenas na sintaxe da sua linguagem de comunicação com o usuário.

Assim podemos considerar a Interface com o Usuário como um subsistema independente no aspecto da estrutura da interação, permitindo que o Sistema PAC incorpore os melhoramentos obtidos na área de estruturas de diálogo, sem que esta modificação cause impacto no restante do Sistema.

Deste modo, a Interface com o Usuário não deverá incorporar recursos gráficos, estes recursos devem ser tratados pelo subsistema Gráfico. Para a execução de um diálogo gráfico a Interface com o Usuário deverá enviar ao Supervisor a informação necessária para que este ative um procedimento do subsistema Gráfico. Este tipo de aproximação leva a uma complexidade maior para o controle do Supervisor, oferecendo no entanto a modularidade desejada.

Para finalizar, deve-se notar que qualquer que seja a estrutura de diálogo utilizada na implementação e quaisquer que sejam as facilidades implementadas (como diálogo adaptativo), a Interface com o Usuário terá duas funções básicas.

-Obter o comando do usuário e fornecer ao Supervisor os dados necessários para a execução do comando; para isto a Interface com o Usuário deverá traduzir o comando em uma linguagem intermediária padrão, reconhecida pelo Supervisor.

-Executar o diálogo interativo para obtenção dos dados e apresentação dos resultados ao usuário.

2.7 - SUPERVISOR

O Supervisor tem o papel de gerenciador do sistema PAC. Ele deverá controlar a execução de comandos do usuário e gerenciar a interligação entre os subsistemas. Para isto o Supervisor deverá se apoiar no sistema operacional da máquina da instalação.

Utilizando a metodologia Graph Model of Behavior - GMB - /RUGG 78/, temos a possibilidade de visualizar as interações dinâmicas entre os vários módulos, através de suas características de execução, fluxo de informação, etc.

Para descrever o comportamento de um sistema, o GMB utiliza três domínios distintos, porém inter-relacionados, que são :

- domínio de controle,
- domínio de dados,
- interpretação.

No domínio de controle, o GMB mostra o fluxo de controle do sistema, através de um grafo composto por nós interligados, que representam os módulos a serem ativados. Além do grafo de controle o GMB prevê uma descrição deste fluxo de controle na interpretação. Na descrição do fluxo de controle estão formalizadas as condições de ativação dos nós, os controles gerados por cada nó, o estado de controle inicial e final do sistema.

No domínio de dados o GMB mostra o fluxo de dados do sistema através de um grafo composto por processadores e armazenadores de dados interligados. A interpretação descreve formalmente o fluxo e o processamento dos dados.

Além de descrever o fluxo de controle e dados, a interpretação relaciona os dois domínios através de regras de associação, que condicionam um processamento do grafo de dados à ativação de um ou mais nós do grafo de controle.

A metodologia GMB foi escolhida pela facilidade com que representa processos concorrentes.

2.7.1 - REPRESENTAÇÃO DO FLUXO DE CONTROLE

O grafo de controle e sua descrição mostram o fluxo de controle do Sistema. O grafo de controle é composto de um conjunto de nós N e um conjunto de arcos A interligando estes nós. Os nós são ativados por marcas em seus arcos de entrada sempre que sua função lógica de entrada IL(n) for verdadeira. Uma vez ativos, estes nós ao término do processamento, geram marcas em seus arcos de saída segundo uma função lógica de saída OL(n).

A seguir, utilizando o grafo GMB, é apresentado o fluxo de controle (Fig. 2.4) do modelo proposto para o Sistema PAC (Fig. 2.2).

Ao Supervisor serão associados cinco nós de controle:

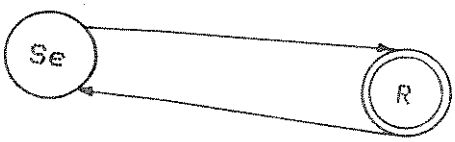
- o nó Si que representa o esquema do Supervisor responsável pela inicialização do Sistema;
- o nó Sf que representa o esquema do Supervisor responsável pelo gerenciamento na finalização dos comandos do usuário;
- o nó Sq que representa o esquema do Supervisor responsável pela geração do comando recebido da Interface com Usuário;
- o nó Se que representa o esquema do Supervisor responsável pelo gerenciamento na execução dos comandos do usuário;

A Interface com Usuário será representada por dois nós:

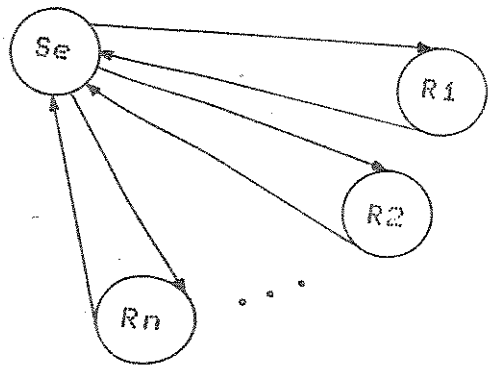
- o nó Ic que controla a aquisição de comandos e
- o nó Id que controla a aquisição de dados e apresentação de resultados.

Os nós Ut, BD, PG, e BA estão associados aos recursos dos subsistemas Utilidades, Banco de Dados, Gráfico e Biblioteca de Aplicações, respectivamente. Deve ser lembrado que um recurso, de qualquer subsistema, não pode ser partilhado, porém recursos diferentes de um mesmo subsistema podem ser ativados simultaneamente.

Por simplicidade os nós Id, Ut, BD, PG e BA serão representados por subgrafos que expandidos representarão o acesso à um unico recurso Ri de um subsistema R, como a seguir:



é equivalente à



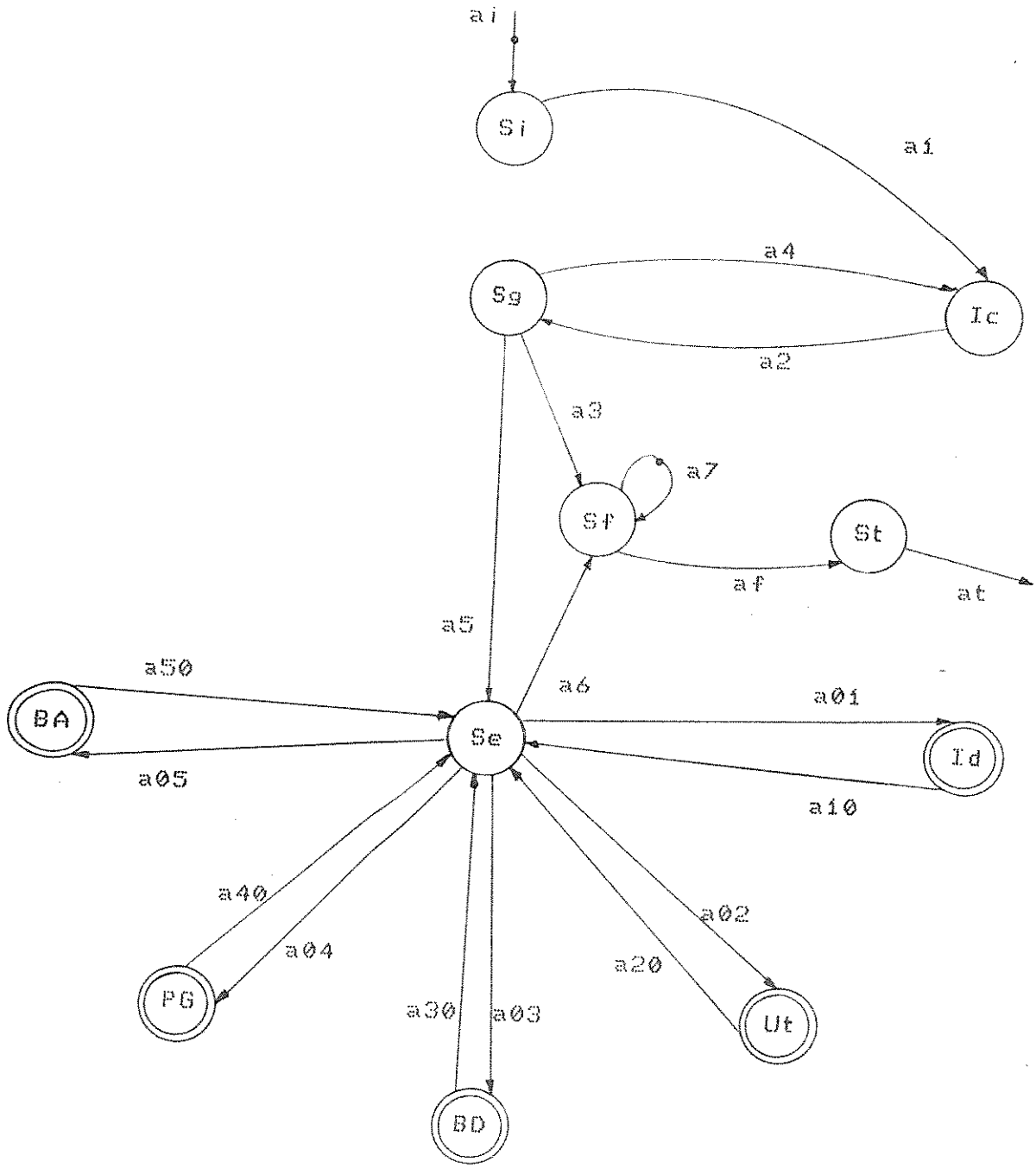


Fig. 2.4 - Grafo de controle do Sistema PAC

INTERPRETAÇÃO NO DOMÍNIO DO GRAFO DE CONTROLE

-Nós e arcos

N = (Si, Sf, St, Sg, Se, Ic, Id, Ut, BD, PG, BA)

A = (ai, af, at, a1, a2, a3, a4, a5, a6, a7, a01, a10, a02, a20, a03, a30, a04, a40, a05, a50)

-Equações lógicas de entrada e saída dos nós

IL(Si) = ai

OL(Si) = a1

IL(Ic) = a1 + a4

OL(Ic) = a2

IL(Sg) = a2

OL(Sg) = a4 * a5 + a3

IL(Sf) = a3 * a7 + a6 * a7

OL(Sf) = af * a7 + a7

IL(St) = af

OL(St) = at

IL(Se) = a5 + a10 + a20 + a30 + a40 + a50

OL(Se) = a6 + a01 + a02 + a03 + a04 + a05

IL(Id) = a01

OL(Id) = a10

IL(Ut) = a02

OL(Ut) = a20

IL(BD) = a03

OL(BD) = a30

IL(PG) = a04

OL(PG) = a40

IL(BA) = a05

OL(BA) = a50

-Multiplicidade de marcas nos arcos de saída para ativação de um nó

IQ(n,a) = 1 $\forall$ n $\in$ N , $\forall$ a $\in$ A

-Multiplicidade de marcas geradas nos arcos de saída de um nó

OQ(n,a) = 1 $\forall$ n $\in$ N , $\forall$ a $\in$ A

-Arcos de controle de entrada : (ai)

-Arcos de controle de saída : (af)

-Nós de controle de entrada : (Si)

-Nós de controle de saída : (St)

A quantidade inicial e final de marcas nos arcos é dada pela tupla

< arcos de entrada , arcos internos , arcos de saída >

especificando cada arco e a quantidade de marcas no mesmo.

-Quantidade inicial de marcas nos arcos:

Estado de controle inicial : I_TD(p) = <ai(1), a7(1), 0>

-Quantidade final de marcas nos arcos:

Estado de controle final : F_TD(p) = <0, a7(1), af(1)>

O Sistema é inicializado por uma marca no arco ai, o Supervisor irá então gerar uma marca no arco a1 para obter um comando do usuário. A execução do comando do usuário (nó Ic) gera uma marca no arco a2. Esta marca ativa o gerador de comandos do Supervisor; se o comando for FIM, uma marca é gerada no arco a3, caso contrário, são geradas as marcas no arco a5, para a execução do comando, e no arco a4, para obtenção de um novo comando do usuário.

O Supervisor, através do seu nó de controle Se, irá então gerar marcas nos arcos conforme a sequência de procedimentos dos subsistemas que deverão ser ativados para a realização da tarefa solicitada.

Terminada a execução de um comando, o Supervisor de execução (nó Se) irá colocar uma marca no arco a6, ativando o Supervisor de finalização. Para uma finalização correta (marca no arco af) deverão estar finalizados todos os comandos ativados e ter sido dado o comando FIM (marca em a3); enquanto esta condição não for satisfeita o nó Sf gerará a marca de saída no arco a7.

Para que o Supervisor gere a sequência de marcas para a realização da tarefa ele deverá possuir tabelas que o informem sobre os recursos existentes nos subsistemas. A existência destas tabelas implica que o Supervisor possua uma base de dados para seu gerenciamento interno.

A escolha da representação pelo GMB foi feita levando em conta que é possível ter mais de um comando sendo executado simultaneamente, havendo assim concorrência no processo. Um exemplo de tarefas concorrentes seria a execução de uma simulação como processamento de fundo, enquanto o usuário executa uma tarefa interativa /ASTR 81/.

2.7.2 - EXECUÇÃO DE UM COMANDO

Um comando do usuário poderá implicar em vários graus de complexidade na sua execução, podendo ser a simples ativação de um procedimento que, por exemplo, execute uma saída gráfica. Para comandos mais complexos serão ativados vários procedimentos de diferentes subsistemas, havendo então a possibilidade de otimizar a execução com a ativação simultânea de procedimentos que comportem paralelismo. Esta possibilidade será anlisada nos capítulos que se seguem, sendo por enquanto, a execução de um comando considerada como um processo sequencial.

Um exemplo do fluxo de controle para a execução de um comando é dado a seguir, utilizando o grafo de controle da figura 2.4.

A marca no arco a5 inicializa a execução do comando ativando o esquema do Supervisor que tem a função de gerar a sequência de marcas necessárias para a execução do comando.

Um exemplo de uma sequencia de marcas geradas no Sistema poderia ser:

a03, a30 - lê do Banco de Dados os parâmetros de entrada da aplicação, ativando o driver que os coloca no formato adequado, de acordo com a interface de entrada do programa de aplicação (Fig. 2.1);

a05, a50 - executa o programa de aplicação;

a03, a30 - armazena os resultados no Banco de Dados, ativando o driver que transforma o formato da interface de saída do programa de aplicação no formato interno do Banco de Dados.

Se a saída de dados do comando endereçar o periférico gráfico serão geradas as seguintes marcas:

a03, a30 - lê do Banco de Dados no formato interno e coloca os dados no formato da interface de entrada do programa que executa a saída gráfica, através da ativação do driver correspondente;

a04, a40 - executa a saída gráfica.

2.7.3 - MACROS E PRODUÇÕES

Cabe ao Supervisor a tarefa de gerenciar o sequenciamento dos procedimentos que executam um comando do usuário; o processamento realizado pela Interface com o Usuário envia para isso ao Supervisor os dados necessários.

A partir destes dados o Supervisor monta o "Macro" do comando. Este Macro é composto de "produções" sincronizadas e sequenciadas que executam o comando. Estas produções estão associadas aos dados que a Interface com Usuário envia ao Supervisor e como em um compilador /DONA 72/, constarão de uma sequencia de operações elementares necessárias para a execução de uma parte do comando.

As produções são funções da aplicação para o subsistema; são esquemas montados a partir das funções dos subsistemas que, para manter a modularidade, devem ser restritos ao subsistema. Elas não são geradas automaticamente e deverão ser implementadas quando da instalação do sistema.

Deste modo uma produção poderá ser uma desde simples ativação de um programa de aplicação, até um procedimento complexo de funções como por exemplo, teste de pré-condições, ativação ou não de um procedimento dependendo destas condições e análise dos resultados. Esta situação é facilmente encontrada em um ambiente PAC e pode ser exemplificada no armazenamento de dados composto de:

- teste de privilégio do usuário ,
- armazenamento ,
- teste de consistência dos dados .

A Fig. 2.5 a seguir mostra a hierarquia mencionada.

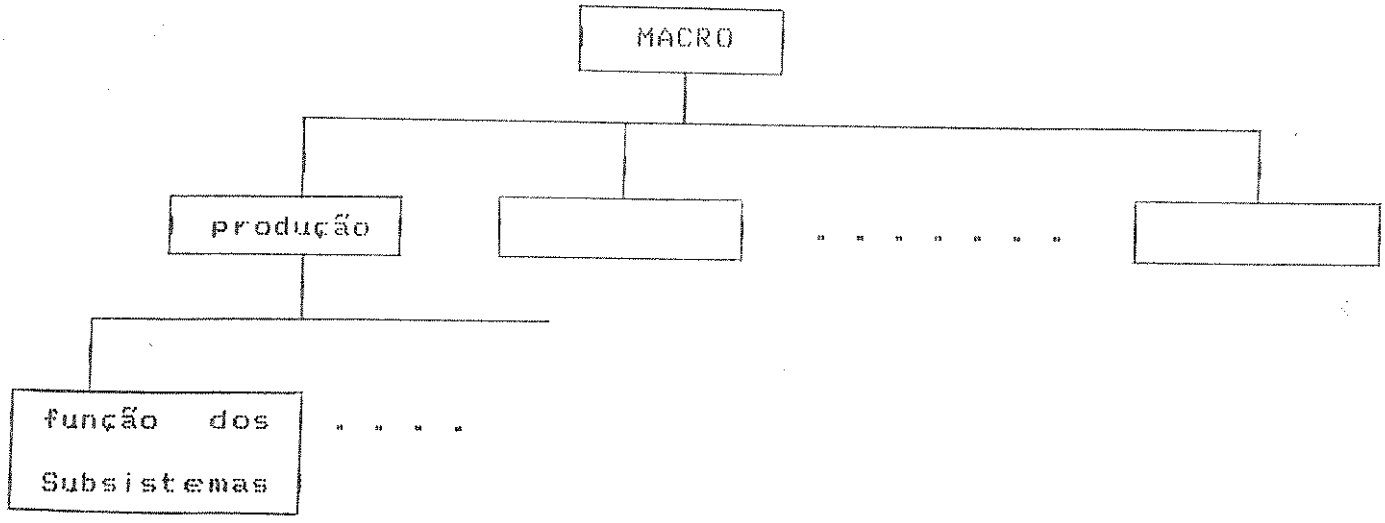


Fig. 2.5 - Estrutura hierárquica do MACRO

Se considerarmos que o Supervisor recebe da Interface com o Usuário um comando na forma :

EXE = A , ENTRADA = X/MOD , SAIDA = Y/MOD

à cada palavra chave seguida de seu parametro e à cada modificador /MOD, teremos uma produção associada.

A palavra chave EXE tem como parâmetro a produção a ser executada, por exemplo ROOT LOCUS da Biblioteca de Aplicações. Os dados de entrada para esta produção são especificados no parâmetro da palavra chave ENTRADA.

Se a entrada especificar que o usuário forneça os dados, teremos uma produção associada da Interface com o Usuário. Esta produção executará o diálogo, obtendo os dados, que serão passados para a produção ROOT LOCUS através de um driver que liga as duas

produções. Se a entrada especificar dados armazenados no Banco de Dados teremos uma produção associada para ler os dados a partir do formato padrão interno de armazenamento.

Do mesmo modo o parâmetro da palavra chave SAIDA poderá ativar produções para executar saída gráfica (subsistema Gráfico) ou para informar o usuário (subsistema Interface com o Usuário). A produção de qualquer dos subsistemas deverá ser ligada à produção ROOT LOCUS por meio de um driver.

Os modificadores para entrada e saída (/MOD) podem ser variáveis em número e cada um deles terá uma produção associada que ativará um dos subsistemas.

Por exemplo o modificador

SAIDA = PLOT/AXES : X , Y

ativará uma produção do subsistema Pacote Gráfico que gerará a saída dos eixos.

As produções fazem parte de uma tabela permanente do Supervisor que é montada na geração do Sistema PAC, pois se referenciam a procedimentos dos subsistemas. Se o Sistema for atualizado com a introdução de um programa de aplicação, a tabela de produções deverá ser atualizada também.

O Macro gerado para o comando poderá não ter existência real, isto vai depender dos recursos da máquina utilizada, pois a existência do Macro implica na existência de um "tradutor" (compilador ou

interpretador) para a sua execução . De qualquer modo o Macro representa a sequencia de procedimentos que devem ser ativados para a execução de um comando do usuário.

O comando, proveniente da Interface com o Usuário, apresentado no exemplo não terá necessariamente a forma descrita, pois sendo um comando interno ao Sistema, sem interação com o usuário, não há necessidade de ser implementado com esta sintaxe. Para um processamento rápido do comando pelo gerador de Macros a linguagem de comunicação deverá ser de baixo nivel tipo L4 /GILO 78/ onde são definidos códigos de operação e lista de dados associados. No capítulo 5 deste trabalho é apresentado um exemplo de como esta linguagem intermediária poderia ser especificada. Esta especificação é feita juntamente com a especificação do Gerador de Macros, pois a linguagem intermediária é entrada para o Gerador de Macros.

2.8 - SUBSISTEMA BANCO DE DADOS

O Banco de Dados terá importância vital na interligação dos subsistemas via Supervisor, pois todo o fluxo de dados no Sistema PAC será feito através do Banco de Dados.

Deste modo, o Banco de Dados deverá armazemar não somente os registros de dados e resultados do processo de projeto, como também registros temporários que serão utilizados na comunicação entre programas de subsistemas diferentes ou mesmo entre programas de um mesmo subsistema /ENCA 83/. Estes registros temporários são transparentes para o usuário e têm a finalidade de padronizar o formato dos dados e resultados de acordo com a interface de entrada ou saída de um procedimento (Fig. 2.1).

Para que esta comunicação entre os subsistemas seja possível, uma vez definida a aplicação, será estabelecido o modelo de dados através da especificação dos tipos de dados da aplicação (ex. polos e zeros, para aplicação em controle de processos). Cada tipo de dado da aplicação será armazenado de um modo uniforme no Banco de Dados, sendo o acesso de qualquer programa aos dados feito pelo acionamento de um driver que mapeia o formato interno de armazenamento no Banco de Dados à interface de E/S do programa.

O grafo GMB a seguir (Fig. 2.6) exemplifica este fluxo de dados, através do Banco de Dados, para o modelo apresentado para o Sistema PAC (Fig.2.2).

2.8.1 - REPRESENTAÇÃO DO FLUXO DE DADOS

O grafo de dados e sua descrição mostram o fluxo de dados do Sistema. O grafo de dados é composto de um conjunto de processadores

controlados CP, um conjunto de processadores não controlados UP, um conjunto de armazenadores DS e um conjunto de arcos de dados que ligam os processadores aos armazenadores.

Os processadores controlados estão associados aos nós do grafo de controle segundo uma regra desecrita na interpretação; o processador é ativado sempre que a regra for satisfeita para os nós ativos. Os processadores não controlados são independentes dos estados dos nós de controle, eles são ativados através de entradas sensitivas. Os armazenadores são abstrações das estruturas de dados.

A interpretação mostra como os dados são transformados pelos processadores .

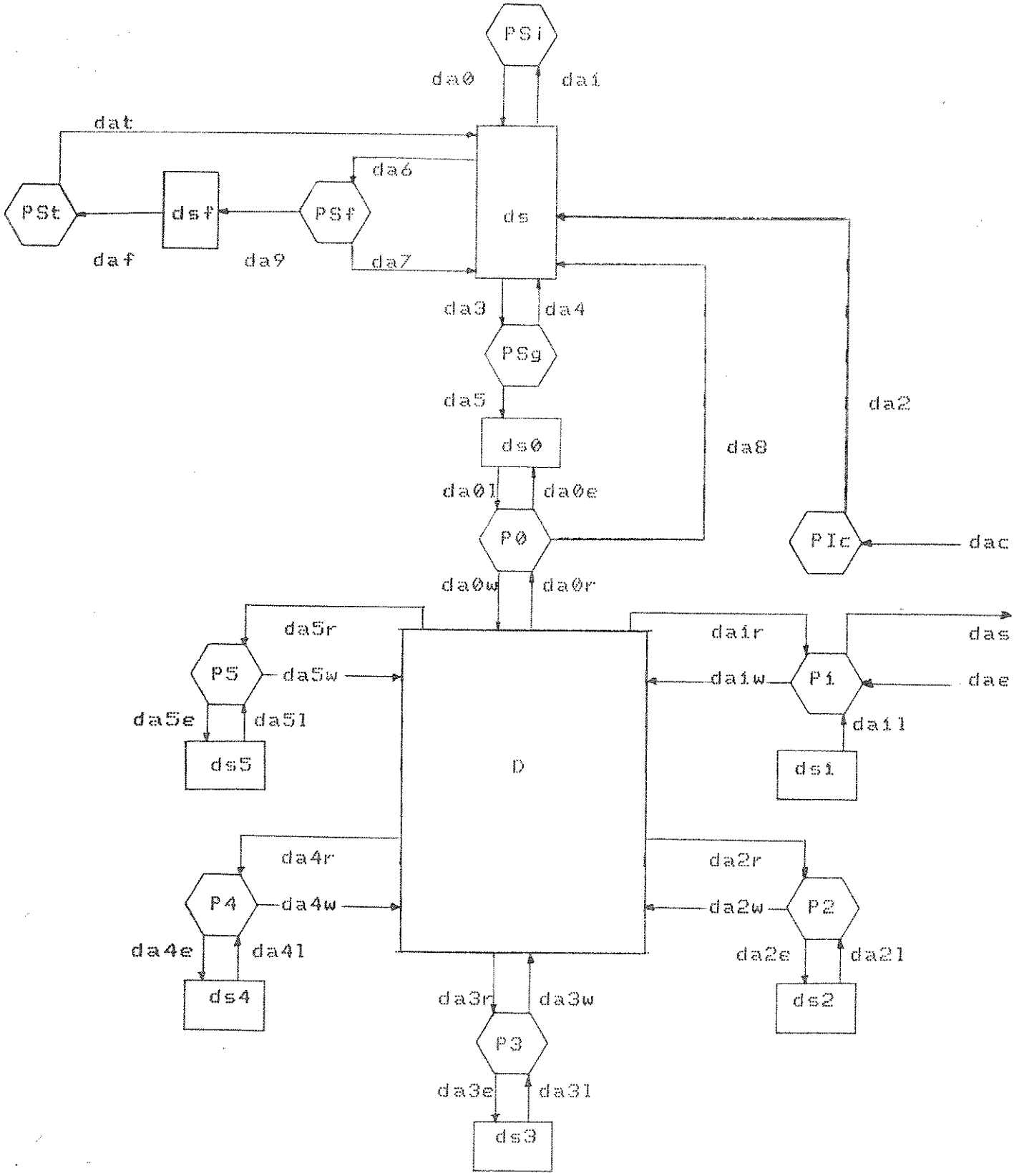


Fig. 2.6 - Grafo de Dados do sistema PAC

INTERPRETAÇÃO NO DOMÍNIO DO GRAFO DE DADOS

DV = CP U UP U DS

-Processadores controlados, processadores não controlados, armazenadores e arcos

CP = (PSi, PSf, PSt, PSg, PIc, P0, P1, P2, P3, P4, P5)

UP = ( )

DS = (D, ds, dsf, ds0, ds1, ds2, ds3, ds4, ds5)

DA = (dac, dae, das, daf, dat, da0, da1, ... , da8, da9,
da0l, da1l, ... , da5l, da0e, da1e, ... , da5e,
da0w, da1w, ... , da5w, da0r, da1r, ... , da5r)

-Descrição dos arcos

dac : ( )---->(PIc)

dae : ( )---->(P1)

das : (P1)---->( )

daf : (dsf)---->(PSt)

dat : (PSt)---->(ds)

da0 : (PSi)---->(ds)

da1 : (ds)---->(PSi)

da2 : (PIc)---->(ds)

da3 : (ds)---->(PSg)

da4 : (PSg)---->(ds)

da5 : (PSg)---->(ds0)

da6 : (ds)---->(PSf)

da7 : (PSf)---->(ds)

da8 : (P0)---->(ds)

da9 : (PSf)---->(dsf)

para i de 0 a 5

dail : (dsi)---->(Pi)

daie : (Pi)---->(dsi)

dair : (D)---->(Pi)

daiw : (Pi)---->(D)

-Arcos de dados de entrada : (dac, dae)

-Arcos de dados de saída : (das)

-Processador de entrada : (PIc, P1)

-Processador de saída : (P1)

-Regras de Associação

PSi -----> Si

PSf -----> Sf

PSt -----> St

PSg -----> Sg

PIc -----> Ic

P0 -----> Se

P1 -----> Id

P2 -----> Ut

P3 -----> BD

P4 -----> PG

P5 -----> BA

-Processamento

```
PSi : ds = psi(ds)              ; inicialização

PSf : if CONT=0 e FLAG=0        ; teste para
       then  dsf = psf(ds)      ; finalização
      end_if

PSt : ds = pst(dsf)             ; finalização

PSg : ds = pg1(ds)              ; geração do comando
      if comando=END
       then  FLAG=0             ; ( ds = pg2(ds) )
       else  FLAG=-1            ;
             CONT=CONT+1        ; ( ds = pg3(ds) )
             ds0 = pg4(ds)      ; controle para próxima tarefa
      end_if

PIc : ds = pic(dac)             ;  entrada do comando

P0  : ds0 = p01(ds0)            ;  controle da execução
      if comando terminou
       then  CONT=CONT-1        ; ( ds = p02(da8) )
       else  D = p03(D)         ; controle para próxima tarefa
      end_if
```

```
P1  : if dae
        then  D = p11(dae,ds1)   ;  entrada de dados
        else  das = p12(D,ds1)   ;  saída de dados
      end_if

P2  : D = p2(D,ds2)   ; processamento do pacote de Utilidades

P3  : D = p3(D,ds3)   ; processamento do Banco de Dados

P4  : D = p4(D,ds4)   ; processamento do Pacote Gráfico

P5  : D = p2(D,ds5)   ; processamento da Aplicação
```

O Grafo de Dados e sua Interpretação nos mostram o fluxo de dados associado ao Grafo de Controle apresentado para o Supervisor conforme a seguir.

O processador PSi inicializa o Sistema. O comando (arco dac) processado pela Interface com Usuário (processador PIc) é enviado ao Supervisor. Se este comando for FIM, o Supervisor prepara a finalização do processo (processador PSf), caso contrário é ativado o Gerador de Comandos (processador PSg). O comando gerado é processado pelo Executor de Comandos (processador P0) que controla a execução dos comandos gerando dados de controle para os processadores dos subsistemas (processadores P1 a P5) que executarão a sequência de procedimentos necessários para o comando. Quando a execução de um

comando é finalizada, o Executor de Comandos sinaliza o término através do arco da8.

A representação do modelo pelo GMB nos mostra, através do grafo de dados, que existe uma base de dados central e todo o fluxo de informação entre os subsistemas é realizado através desta.

Além da base de dados central, cada subsistema deverá possuir sua própria base de dados para gerenciamento interno. A base de dados do subsistema aplicação é especialmente necessária em se deixando em aberto a possibilidade de utilização de software existente, permitindo-se assim acoplar diferentes pacotes de aplicação dentro do subsistema Biblioteca de Aplicação, através de uma base de dados /SPAN 84/.

A imposição da existência da base de dados central se deve à restrição de modularidade do Sistema.

Se fosse permitido que subsistemas se comunicassem independentemente da base central não haveria uniformidade no tratamento da informação. Com a base central de dados será necessária a implementação de um único driver de E/S de acordo com a interface de E/S de cada produção; não haveria necessidade da implementação de diversos drivers que fizessem o mapeamento de formatos entre uma produção e diversas outras. Deste modo, o driver ativado pelo Supervisor será único para cada produção, sendo responsável pelo mapeamento do formato de E/S da produção no formato definido na estrutura de dados.

Assim, como toda a comunicação é baseada em registros da base central de dados, poderá ser gerado um acúmulo de informações intermediárias /ENCA 83/. Para que isto não ocorra o sistema gerenciador da base de dados deverá permitir a definição de registros temporários para cada tipo de dado da aplicação, registros estes que deverão ser eliminados após sua utilização.

Como estes registros são temporários, os esquemas definidos para estes registros são mais simples, não havendo necessidade de se implementar as restrições implementadas nos esquemas para os dados que serão armazenados de forma permanente. A única restrição que deverá ser implementada para os esquemas de registros temporários será o tempo de existência, para que possam ser eliminados após sua utilização.

2.9 - INSTALAÇÃO DO SISTEMA PAC

Na instalação de um Sistema PAC deverão estar disponíveis :

- a Biblioteca de Aplicações,
- o Banco de Dados,
- a Interface com o Usuário,
- o Pacote Gráfico,
- o pacote de Utilidades.

Para cada subsistema deverão ser implementados os esquemas da aplicação. Estes esquemas definem a interface do subsistema com a aplicação, através de procedimentos que utilizam as funções do subsistema. Um exemplo de esquema definido para o subsistema Gráfico são as primitivas gráficas oferecidas à aplicação. Os esquemas definidos para cada subsistema são denominados produções.

BANCO DE DADOS

Definida a aplicação, tem-se os tipos de dados da aplicação. Para cada tipo de dado da aplicação deverão ser definidas as produções para manipulação do Banco de Dados. Deverão ser definidas também, produções para manipulação de registros temporários que servirão para a comunicação entre produções de subsistemas diferentes.

BIBLIOTECA DE APLICAÇÕES

Para cada produção do subsistema Biblioteca de Aplicações deverá ser implementado o driver de E/S que mapeia os formatos de E/S da produção no Banco de Dados. Os drivers serão escritos utilizando produções do Banco de Dados, e deverão receber a informação para decidir se o armazenamento será na forma de registro temporário ou não.

PACOTE GRÁFICO

Para cada produção do subsistema Pacote Gráfico deverá ser implementado o driver de E/S para o Banco de Dados.

UTILIDADES

Para cada produção do subsistema Utilidades deverá ser implementado o driver de E/S para o Banco de Dados.

INTERFACE COM O USUÁRIO

Para a Interface com o Usuário deverá ser definida a linguagem intermediária. Como foi dito, o processador de diálogo gerará esta linguagem intermediária no processamento do comando para a comunicação com o Supervisor. Além disto deverão ser implementados os drivers de E/S das produções da Interface com Usuário.

SUPERVISOR

Para o Supervisor deverá ser implementado o Gerador de Macros, as Tabelas de Produções e Drivers de E/S, os esquemas de comunicação com os subsistemas e o esquema de execução do Macro gerado.

A Tabela de Produções será composta pelos esquemas definidos para os subsistemas em função da aplicação. Como as produções estão fechadas em um subsistema elas já podem estar "linkadas"; o fluxo de dados na execução de uma produção será interno ao subsistema, ou seja, será via estrutura de dados interna do subsistema.

A Tabela de Drivers de E/S será composta pelos esquemas definidos a partir das interfaces das produções de cada subsistema com o Banco de Dados.

Os esquemas de comunicação entre os subsistemas serão utilizados pelo Gerador de Macros para ligação das produções, utilizando os drivers de E/S implementados para cada produção.

O esquema de execução do Macro vai depender da máquina da instalação. Podem ser consideradas duas aproximações:

- utilização de um processador que traduza o Macro para código executável da máquina,
- utilização do procedimento de ativação de tarefas /PDP 1/,/PDP 2/,/PDP 3/.

2.10 - SUBSTITUIÇÃO DE UM SUBSISTEMA

Para a substituição de um subsistema será necessária a substituição :

-das produções ou funções da aplicação para o subsistema, isto porque as funções básicas dos dois subsistemas (o antigo e o novo) certamente não serão as mesmas,

-dos drivers de E/S para as produções do subsistema pois os formatos de E/S dos procedimentos não serão os mesmos.

Se o subsistema a ser substituido for o Banco de Dados, a reconfiguração será mais extensa, pois todos os drivers de E/S para as produções de todos os subsistemas deverão ser reescritos. Estes drivers são implementados utilizando produções do subsistema Banco de Dados.

CAPÍTULO 3 - NÚCLEO PAC

O Núcelo PAC é uma proposta de interface padrão para com os módulos lógicos componentes do Sistema PAC. Para as funções do Núcelo PAC, é apresentado o modelo matemático em analogia com o conceito de μ-programação. A seguir é analizada a instalação de um Sistema PAC com Núcelo PAC e o impacto decorrente da substituição de um dos módulos componentes do Sistema.

O modelo lógico de um Sistema PAC foi definido com a finalidade de se alcançar a modularidade do Sistema, restringindo-se assim o impacto causado pela substituição ou modificação de um dos subsistemas componentes do Sistema. Porém, apesar de restritas, estas modificações podem atingir um grande número de procedimentos.

Se um subsistema for substituído, certamente não haverá impacto sobre os outros subsistemas, porém as produções deste subsistema deverão ser reescritas, pois as funções dos dois módulos, o novo e o antigo, certamente não serão as mesmas. O problema é mais crítico se o subsistema substituido for o Banco de Dados; todos os drivers para as interfaces de E/S das produções de todos os subsistemas deverão ser reescritos, visto que, na implementação destes drivers, são utilizadas produções do subsistema Banco de Dados.

A solução encontrada para o problema foi a definição de funções padrões para os diferentes subsistemas. A existência destas funções padrões implica em termos as produções, escritas com estas funções, também em forma de padrões para o Sistema. O mesmo é válido para os drivers de E/S das produções; estes deverão ser implementados utilizando funções padrões para o subsistema Banco de Dados.

Este conjunto de funções padrões para os diferentes subsistemas foi chamado de Núcleo PAC.

Além dos subsistemas Interface com Usuário, Pacote Gráfico, Banco de Dados, Biblioteca de Aplicações e Utilidades já definidos no modelo de um Sistema PAC, foi acrescentado o subsistema "Sistema Operacional". A definição de funções padrões para o Sistema Operacional daria ao Sistema PAC a característica de portatilidade em relação à máquina utilizada na implementação.

As funções do Núleo PAC definem uma máquina de software abstrata para o Sistema PAC.

As funções do Núcleo PAC para um subsistema definem a interface do Sistema PAC para o subsistema. Isto implica na necessidade de serem implementadas interfaces que façam a ligação entre as funções do Núcleo PAC e as funções disponiveis do subsistema existente na instalação.

A Fig. 3.1 a seguir exemplifica o conceito de Núcleo PAC.

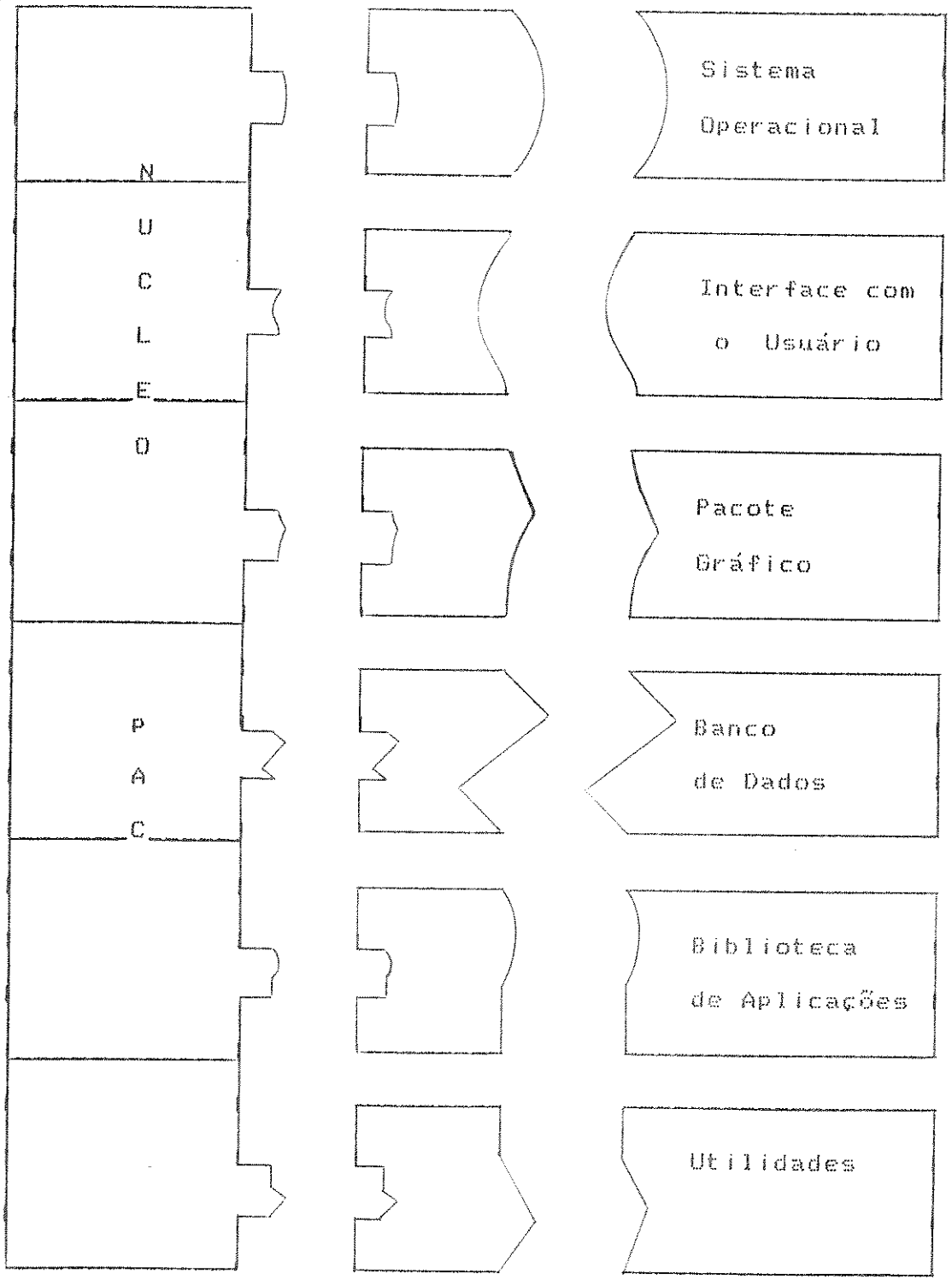


Fig. 3.1 - Núcleo PAC

Qualquer referência ao subsistema será feita através das funções do Núcleo PAC para o subsistema; as produções serão implementadas utilizando as funções do Núcleo PAC para o subsistema.

A introdução desta interface padroniza o acesso aos subsistemas; não havendo mais a restrição de procedimentos serem autocontidos, podendo um procedimento de um subsistema referenciar uma função de outro subsistema, através das funções do Núcleo PAC.

Esta aparente quebra da modularidade não irá comprometer a substituição de um subsistema pois, uma vez implementada a interface do novo subsistema, todos os esquemas montados para o Sistema PAC continuarão válidos pois utilizam funções do Núcleo PAC. Esta restrição seria particularmente grave na implementação do diálogo gráfico, o esquema necessário para implementar a ligação entre o processador de diálogo e o Pacote Gráfico via Supervisor seria bastante complexo. Com o Núcleo PAC, os esquemas desenvolvidos para o diálogo na Interface com Usuário poderão incorporar referencias às produções do subsistema Pacote Gráfico.

Com o conceito de Núcleo PAC a substituição de um subsistema fica restrita à implementação de uma nova interface para o subsistema.

3.1 - ANALOGIA COM O CONCEITO DE μ-PROGRAMAÇÃO

As funções do Núcleo PAC, em analogia com as μ-instuções, deverão ser definas no mesmo nível básico das funções de um subsistema, ou seja, elas deverão constituir uma interface para com o subsistema em um contexto de ambiente PAC, porém independente da aplicação.

Tomando como exemplo o subsistema Banco de Dados, a função do Núcleo PAC que acessa um registro poderá incorporar um teste de permissão de acesso, por ser uma característica comum aos Sistemas PAC, não devendo entretanto incorporar outros testes que se referem a esquemas específicos da aplicação.

Baseando-se em diferentes implementações existentes para um subsistema S qualquer, definimos

$$fs = (f1s, f2s, f3s, \ldots , fms)$$

como o conjunto de funções disponíveis do subsistema S.

Baseando-se nas funções necessárias para um ambiente PAC definimos

$$Fs = (F1s, F2s, F3s, \ldots , Fns)$$

como o conjunto de funções do Nucleo PAC para o subsistema S.

A relação entre as funções do subsistema e o conjunto de funções do Núcleo PAC para o subsistema, pode ser exemplificada através do conceito de μ-programação (Fig. 3.2).

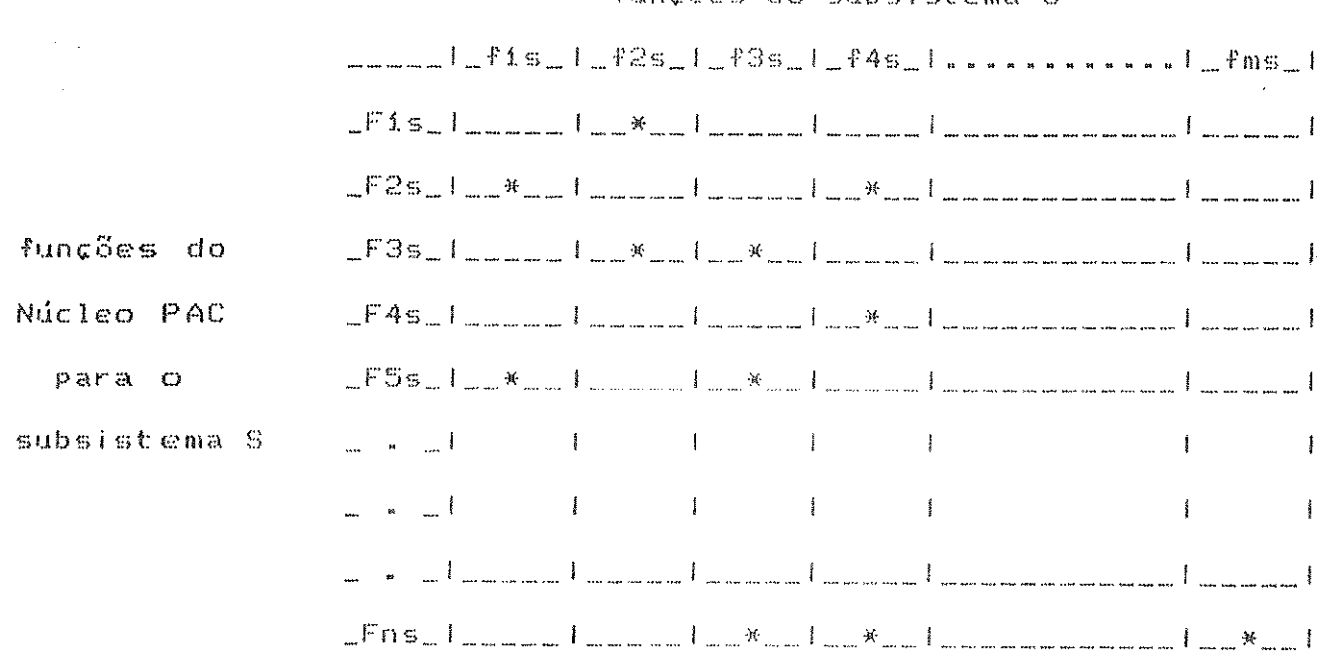


Fig. 3.2 – Relação entre funções do subsistema e funções do Núcleo PAC

As funções Fis do Núcleo PAC podem ser definidas como uma atribuição direta

$$F1s = f2s \ ,$$

ou como um sequenciamento de funções

$$F2s = p(f1s \ , \ f4s) \ ,$$

não sendo descartada a possibilidade de processamento paralelo de funções f que não interferem entre si .

A seguir, utilizando a analogia com o conceito de μ-programação, será desenvolvido um modelo matemático para as funções do Núcleo PAC, com a finalidade de obter ferramentas que permitam a automatização na geração destas funções.

3.2 - MODELAMENTO MATEMÁTICO DAS FUNÇÕES DO NÚCLEO PAC

A teoria de conjuntos será utilizada para modelar as funções do Núcleo PAC /PORT 68/.

Seja fs o conjunto de funções para o subsistema S

$$fs = (f1s, f2s, \ldots , fms).$$

Seja Ts c fs

$$Ts = (t1s, t2s, \ldots , tls) ,$$

o conjunto gerador para as funções do Núcleo PAC para o susbistema S , com as propriedades :

a) Ts é capaz de gerar qualquer função para o subsistema S,

b) Ts é o conjunto mínimo, não existe função que mapeie

$$g : Ts \longrightarrow Ts ,$$

c) a dimensão de Ts é 1.

Seja Fs o conjunto de funções do Núcleo PAC para o subsistema S

$$Fs = (F1s, F2s, \ldots , Fms).$$

Pela analogia com o conceito de u-programação, uma função do Núcleo PAC será composta de uma sequência de procedimentos, sendo cada procedimento composto de funções que comportam processamento paralelo.

Para modelar a sequência de processos é definido o conjunto Bs composto das sequências

$$(bk) = (b1k, b2k, b3k, \ldots\ldots, bpk) \qquad p>=1$$

geradas pela transformação gk,

$$gk : Ts \longrightarrow Bs .$$

Cada elemento bik da sequência (bk) está associado a um processo e portanto deverá representar o paralelismo das funções dentro do processo.

Para definir a execução sequencial ou paralela de processos, são introduzidos dois operadores que indicam :

a//b - execução paralela, a e b são executados em paralelo

a>>b - execução concatenada, a é executado antes de b.

Assim, utilizando o operador paralelo (//), a transformação gk será definida como :

$$\begin{vmatrix} b1k \\ b2k \\ . \\ . \\ bpk \end{vmatrix} = \begin{vmatrix} a11 & a12 & a13 & \ldots\ldots & a1l \\ a21 & & & & . \\ . & & & & . \\ . & & & & . \\ ap1 & . & . & \ldots\ldots & . \end{vmatrix} // \begin{vmatrix} t1s \\ t2s \\ . \\ . \\ tls \end{vmatrix}$$

onde

$$aij \in A = (0 , 1)$$

e

$$aij * tjs = \begin{cases} - \text{ NOP } \in Ts \text{ , indicando não operação se } aij = \emptyset \\ \\ - \text{ } tjs \text{ se } aij \neq \emptyset \end{cases}$$

com

$$bik = ai1 * t1s \;//\; ai2 * t2s \;//\; \ldots \;//\; ail * tls$$

Os elementos bik de cada sequência (bk) determinam quais funções podem ser executadas paralelamente .

Cada sequência (bk) determina a sequência na execução dos elementos bik.

Para obter o conjunto Fs das funções do Núcleo PAC para o subsistema S, é aplicado o operador sequencial (>>) através da transformação

$$h : Bs \longrightarrow Fs$$

definida por

$$Fis = [\, i \;\; i \;\; \ldots\ldots \;\; i \,] \;\; >> \;\; \begin{vmatrix} b1i \\ b2i \\ . \\ . \\ \\ bpi \end{vmatrix}$$

obtendo-se

$$Fis = b1i >> b2i >> \ldots\ldots >> bpi \;.$$

Não faz parte deste trabalho o estudo sobre como gerar as funções do Núcelo PAC e as produções de cada subsistema. Sendo esta tarefa realizada de modo "off line", não cabe ao usuário e sim à equipe de manutenção do Sistema. Todavia a implantação e atualização so Sistema pode ser automatizada /CAMI 85/.

As funções do Núcleo PAC podem ser geradas automaticamente, utilizando-se o modelamento matemático descrito. O Apêndice B apresenta outras opções para o modelamento de funções mais complexas.

O mesmo modelamento pode ser utilizado para gerar produções que comportem concorrência.

3.3 - IMPLEMENTAÇÃO DO SISTEMA COM NÚCLEO PAC

A implementação do Sistema PAC com Núcleo PAC segue os seguintes passos:

- implementação das interfaces com os subsistemas através da implementação das funções do Núcleo PAC, utilizando as funções básicas do subsistema,
- definição da Tabela de Produções através da implementação das funções da aplicação, utilizando as funções do Núcleo PAC,
- implementação dos drivers de E/S para as produções utilizando funções do Núcleo PAC relativas ao subsistema Banco de Dados.

3.4 - SUBSTITUIÇÃO DE UM SUBSISTEMA

Para a substituição de um subsistema em um Sistema PAC com Núcleo PAC será necessário :

- a implementação da interface com o subsistema através da implementação das funções do Núcleo PAC, utilizando as funções básicas do subsistema.

3.5 - CONSIDERAÇÕES FINAIS

A definição das funções do Núcleo PAC é uma tarefa complexa , principalmente por abranger diversas áreas de pesquisa . Esta dificuldade pode ser contornada permitindo-se que o conjunto de funções do Núcleo PAC seja aumentado na medida das necessidades. A introdução de uma nova função para o Núcleo PAC fica restrita à sua implementação na interface de seu subsistema e à geração da produção que a ativaria. Esta facilidade é particularmente importante em relação ao subsistema Biblioteca de Aplicações, permitindo sem um grande impacto no Sistema, a introdução de novos programas de aplicação.

CAPÍTULO 4 - PARALELISMO NO MODELO PROPOSTO

Os problemas relativos à sincronização e passagem de parâmetros em processos paralelos são analizados com a finalidade de prover ferramentas para a excução de comandos do usuário em paralelo, no Sistema PAC proposto.

Ao longo do texto foi citada várias vêzes a execução paralela de tarefas. Primeiramente seria interessante que tarefas de longa duração, como simulações, fossem ativadas como um processamento de fundo, enquanto o usuário trabalha em uma tarefa interativa. Em um segundo nível, devido à introdução de diversas interfaces para deixar o Sistema modular e portátil, o tempo de processamento poderá tornar-se longo, deixando lentas as interações com o usuário.

Deste modo, para que seja permitido ao usuário ativar tarefas concorrentes, o módulo Se - Macro Execução - (Fig. 2.4) deverá ser implementado com esta característica. Para as produções e funções do Núcleo PAC, como estas são geradas "off line", a característica de paralelismo poderá ser ou não incorporada, dependendo de como foram implementadas. Os mesmos esquemas que serão definidos para a execução de comandos concorrentes poderão ser utilizados na implementação de produções e funções do Núcleo PAC com paralelismo.

Existem sistemas operacionais que permitem a excução de várias tarefas simultaneamente porém o UNIX /THOM 82/, por exemplo, exige que estas tarefas já estejam previamente "linkadas". Esta exigência seria restritiva para a montagem de comandos do Sistema PAC, visto que, para a execução de um aplicativo seria interessante que houvesse flexibilidade de aquisição dos dados por diálogo, arquivo, etc. Esta flexibilidade só seria conseguida com a montagem das diferentes formas de comando, ocorrendo assim uma duplicação de código objeto.

Uma solução para o problema seria a utilização de "linkagem dinâmica", porém existem poucas máquinas com esta facilidade no sistema operacional.

A solução proposta é a utilização de um Sistema Operacional Tempo Real, sendo os comandos executados através de ativação e sincronização de tarefas.

A seguir será feito um estudo de paralelismo sobre o modelo apresentado para o Sistema PAC, utilizando o conceito de Núcleo PAC.

4.1 - EXECUÇÃO DO MACRO

Para a execução de um comando do usuário são ativadas produções. A fig 4.1 mostra como um comando vindo da Interface com o Usuário gera um Macro no Supervisor.

Supondo, como exemplo, uma aplicação em controle de processo, onde o usuário queira executar um procedimento que calcula o root-locus, a partir de dados que estão em um arquivo e cuja saída deverá ser um gráfico temos :

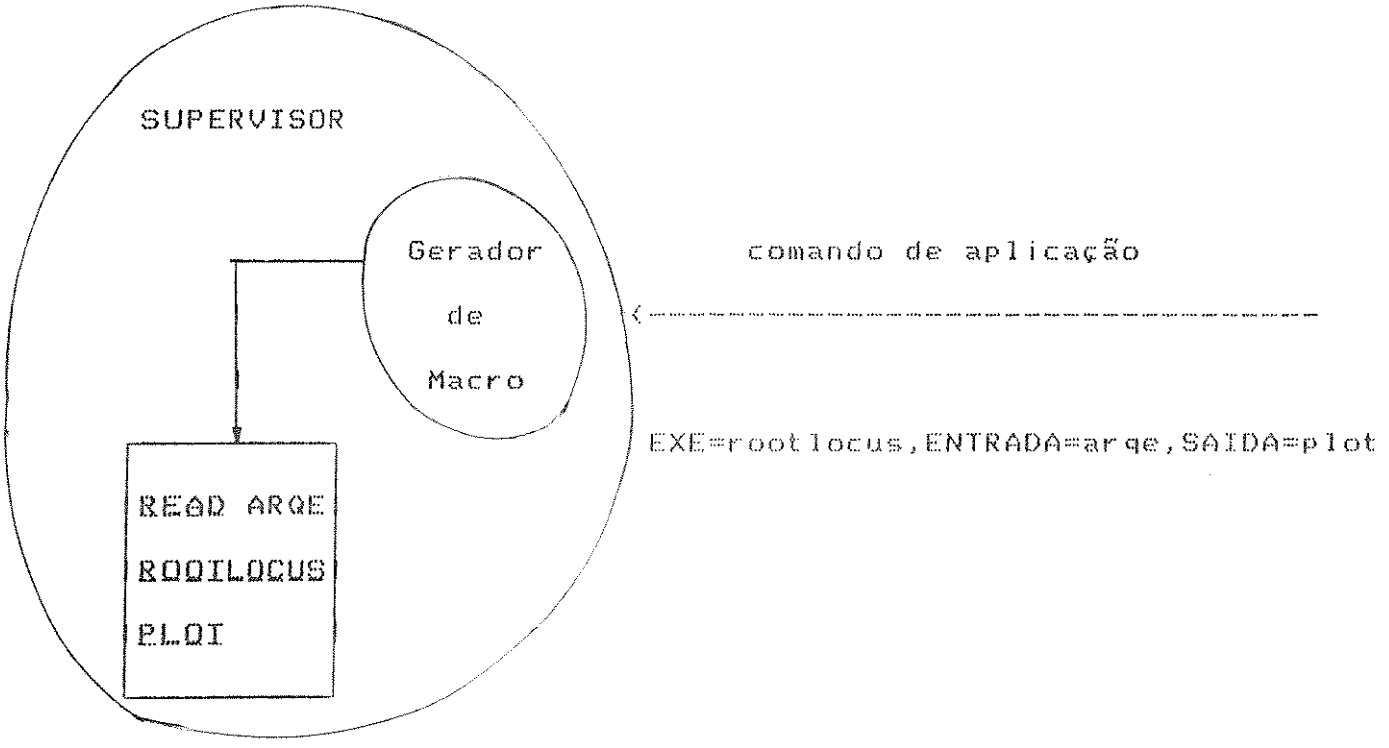


Fig. 4.1 - Geração do Macro do Supervisor

onde READ , ROOTLOCUS e PLOT são produções relativas aos subsistemas Banco de Dados, Biblioteca de Aplicações e Pacote Gráfico, respectivamente. Estas produções, por sua vez, são compostas de funções do Núcleo PAC.

A execução do Macro é realizada por funções do Sistema Operacional. Estas serão funções do Núcleo PAC relativas ao subsistema Sistema Operacional. Assim a execução do Macro seria feita através de ativação de tarefas /PDP 1/,/PDP 2/,/PDP 3/.

É importante observar que com esta aproximação existirá uma hierarquia dentro das funções do Núcleo PAC. A execução de um comando do usuário é feita através de funções do Núcleo PAC relativas ao subsistema Sistema Operacional.

No Macro do exemplo as produções seriam ativadas pela função ACTIVATE, função do Núcleo PAC relativa ao subsistema Sistema Operacional.

```
ACTIVATE (READ)
ACTIVATE (ROOTLOCUS)
ACTIVATE (PLOT)
```

O Macro seria então composto por funções do Núcleo PAC relativas ao Sistema Operacional, que ativariam produções compostas por funções do Núcleo PAC relativas aos demais subsistemas .

A figura 4.2 a seguir exemplifica esta hierarquia.

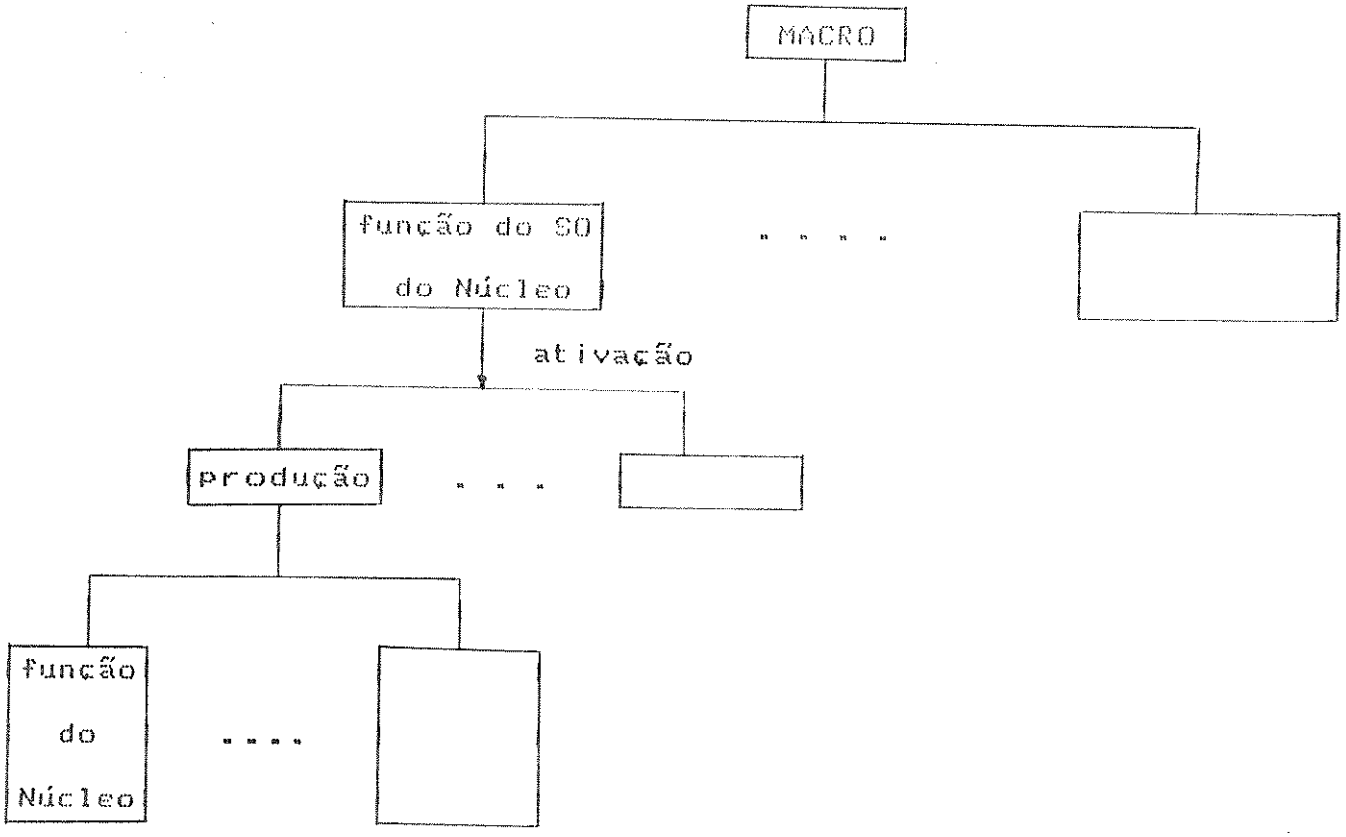


Fig. 4.2 - Composição do MACRO

4.2 - SINCRONIZAÇÃO

A execução simultânea de procedimentos implica em estudos de sincronização, sendo esta realizada através de semáforos /ANDR 85/, /ALLW 81/. Se para a aplicação, o semáforo assume somente valores 0 e 1, a sua implementação pode ser feita por meio de flags /PDP 1/, /PDP 2/, /PDP 3/.

As funções necessárias para a sincronização são :

SET FLAG (f) ,

CLEAR FLAG (f) ,

WAIT FOR FLAG (f) ,

sendo f os flags de sincronização alocados ao Sistema.

As produções e as funções do Núcleo PAC são implementadas quando da instalação do Sistema PAC e fazem parte das interfaces para os subsistemas; os flags de sincronização utilizados na implementação das funções do Núcleo PAC poderiam ser alocados na sua implementação. Porém, com a alocação pré-fixada dos flags de sincronização, poderá surgir um conflito quando existirem tarefas de fundo sendo executadas simultaneamente; procedimentos de tarefas diferentes podem estar sincronizados com um mesmo flag.

A solução para a sincronização é a atribuição dinâmica dos flags sempre que houver necessidade de sincronismo na execução de procedimentos, a um custo de um gerenciador mais complexo para o Supervisor e procedimentos mais complexos na implementação das produções e funções do Núcleo PAC.

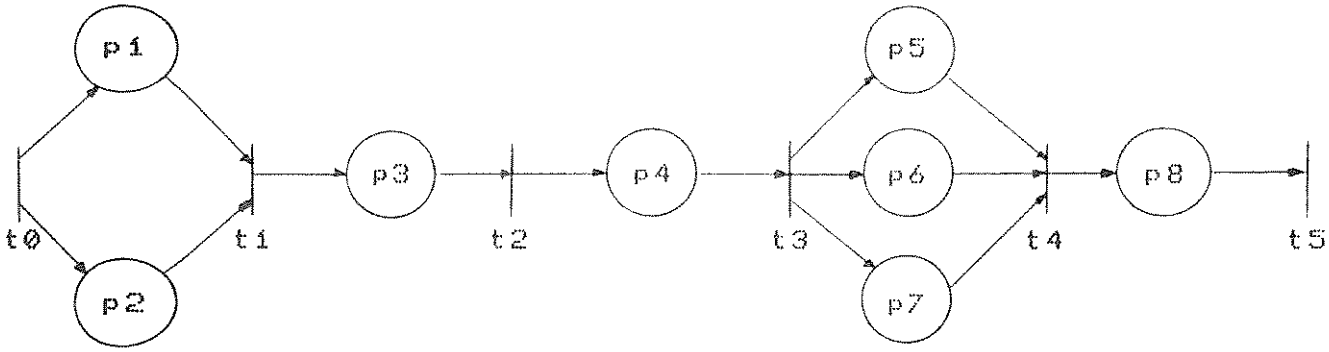
Basicamente, será necessário implementar uma estrutura de dados com os flags da implementação e duas funções de acesso:

-ALOCA FLAG (f) , retorna o nome de um flag disponível

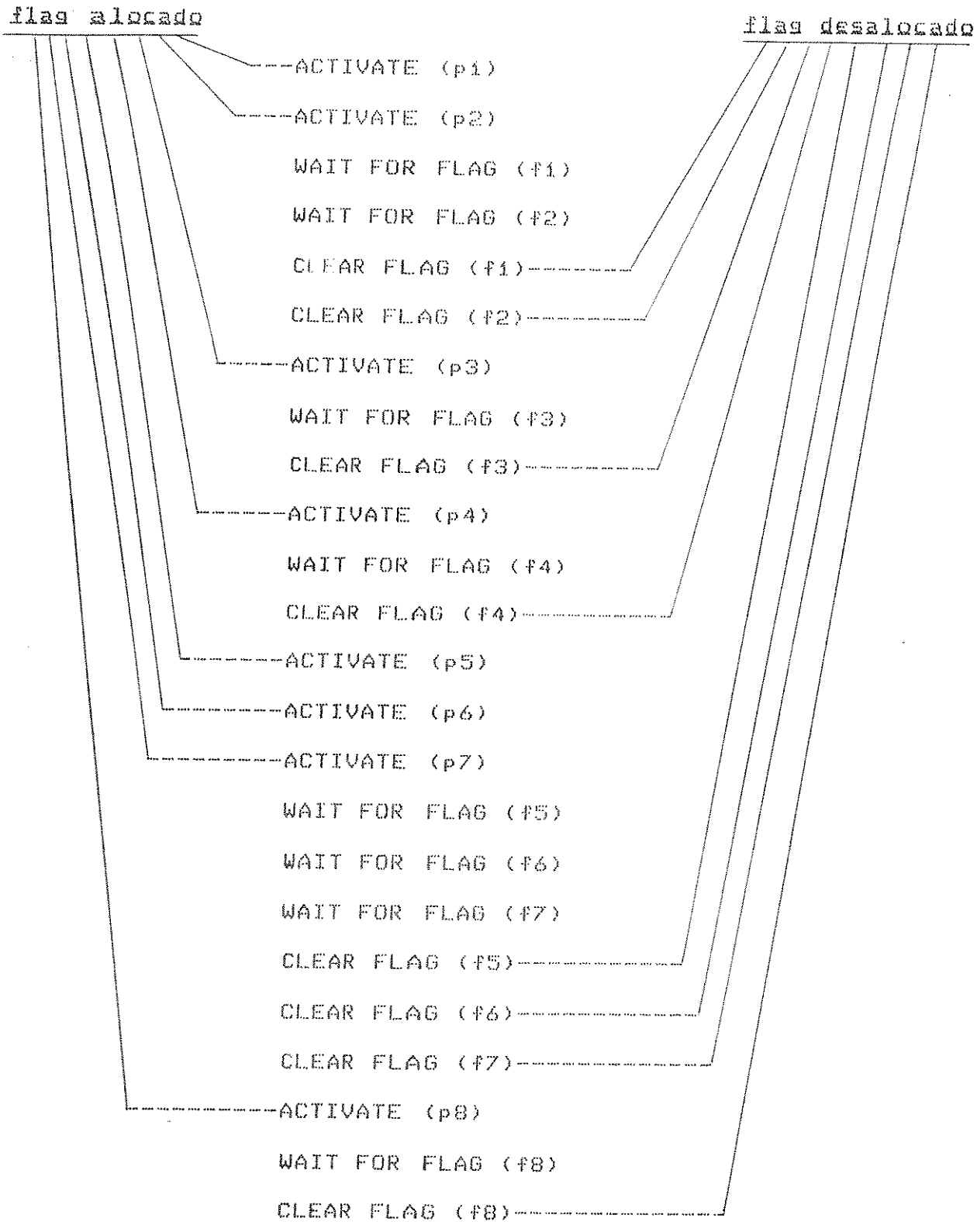
-DESALOCA FLAG (f) , sinaliza disponível o flag f.

Estas funções serão mencionadas no decorrer deste capítulo.

O grafo Petri-Net /PETE 81/ abaixo mostra um exemplo para a implementação de uma função F do Núcleo PAC com paralelismo.

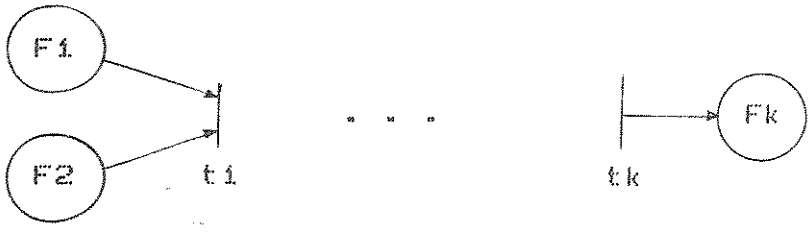


O procedimento para implementar esta função será :


flag alocado
flag desalocado
ACTIVATE (p1)
ACTIVATE (p2)
WAIT FOR FLAG (f1)
WAIT FOR FLAG (f2)
CLEAR FLAG (f1)
CLEAR FLAG (f2)
ACTIVATE (p3)
WAIT FOR FLAG (f3)
CLEAR FLAG (f3)
ACTIVATE (p4)
WAIT FOR FLAG (f4)
CLEAR FLAG (f4)
ACTIVATE (p5)
ACTIVATE (p6)
ACTIVATE (p7)
WAIT FOR FLAG (f5)
WAIT FOR FLAG (f6)
WAIT FOR FLAG (f7)
CLEAR FLAG (f5)
CLEAR FLAG (f6)
CLEAR FLAG (f7)
ACTIVATE (p8)
WAIT FOR FLAG (f8)
CLEAR FLAG (f8)

As produções são procedimentos das funções Fi do Núcleo PAC. Supondo que F1, F2, ... Fk são funções do Núcleo PAC para um subsistema e que F1 e F2 podem ser concorrentes, a implementação de uma produção será por exemplo :

```
ACTIVATE (F1)
ACTIVATE (F2)
WAIT FOR FLAG (f1)
CLEAR FLAG (f1)
WAIT FOR FLAG (f2)
CLEAR FLAG (f2)
      .
      .
      .
ACTIVATE (Fk)
WAIT FOR FLAG (fk)
CEAR FLAG (fk)
```



A execução do Macro é sempre feita de modo sequencial; para o exemplo dado teremos :

```
ACTIVATE (READ)
WAIT FOR FLAG (f)
CLEAR FLAG (f)
ACTIVATE (ROOTLOCUS)
WAIT FOR FLAG (f)
CLEAR FLAG (f)
ACTIVATE (PLOT)
WAIT FOR FLAG (f)
CLEAR FLAG (f)
```

A restrição do processamento sequencial do Macro é devido ao fato do macro ser gerado automaticamente. O Gerador de Macro seria bem mais complexo se tivesse que decidir quais das produções componentes do Macro comportam processamento paralelo. Porém, se os algorítmos de decisão forem implementados em PROLOG /CLAR 84/, o acesso à informação sobre concorrência de tarefas seria simplificado. O apêndice A apresenta um esquema para a execução do Macro com concorrência. Neste esquema é feita uma referência à um procedimento (IS) que tem por finalidade obter determinar se duas tarefas podem ser concorrentes. O procedimento IS é parte do ambiente PROLOG do Sistema, e equivale ao comando IS do PROLOG.

Apesar de demostrado no apêndice A que é possivel a execução do Macro com concorrência, por simplicidade, o Macro será tratado como um processo sequencial.

4.3 - PASSAGEM DE PARÂMETROS

Um dos problemas a resolver quando temos tarefas concorrentes é a comunicação entre estas tarefas /ALLW 81/.

Tanto as funções do Núcleo PAC como as produções estão sempre restritas à um subsistema. Deste modo o fluxo de dados é local, sendo realizado através da base de dados do subsistema. Se estas produções e/ou funções do Núcleo PAC forem implementadas com paralelismo, por ativação de tarefas, deverá ser implementado um mecanismo para a transferência de dados entre tarefas. Além dos dados, haverá também a transferência de informação de controle, como por exemplo, o flag de sincronização alocado.

No processamento do Macro, o fluxo de dados será via Base de Dados central, porém informações de controle como o flag de sincronização alocado e a identificação dos registros de dados da Base de Dados central que serão manipulados, deverão ser passados para as tarefas ativadas.

CAIXA POSTAL

A solução proposta para a passagem de parâmetros é a utilização de uma caixa postal ligando o Supervisor a qualquer procedimento ativado.

Existem sistemas operacionais que permitem a criação desta caixa postal através do mapeamento comum de seções de programas, no espaço virtual, na mesma região do espaço físico de memória /PDP 1/, /PDP 2/, /PDP 3/. Esta propriedade pode ser implementada com relativa facilidade em um sistema operacional que não a possui /MINT 83/, /COM 1/.

Resta agora o problema de permitir ou bloquear o acesso à região crítica que é a caixa postal /SHAW 74/.

Para isto foi desenvolvida a Tarefa Caixa Postal (TCP). Esta tarefa tem a propriedade de não ser reentrante, ou seja, não pode ser ativada enquanto estiver ativa, o que impede que qualquer outro procedimento, diferente daquele que ativou a TCP, tenha acesso à caixa postal. Além de controlar o acesso à caixa postal, a TCP possui ainda o gerenciador de flags que aloca ou desaloca os flags de sincronização das tarefas.

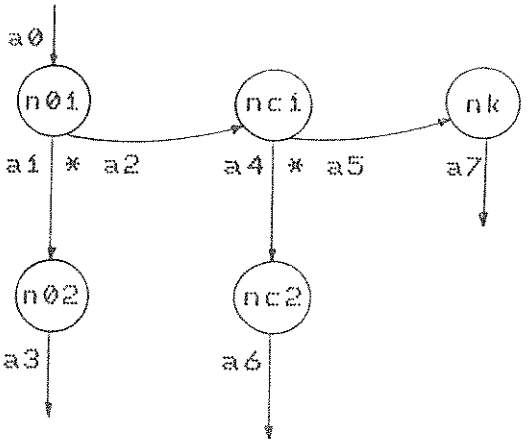
Do exposto até então pode-se deduzir que a TCP deverá ter três modos de entrada. Um modo para alocar flags de sincronização e fornecer, à tarefa a ser ativada, os parâmetros de entrada. Outro modo para desalocar flags e o terceiro para fornecer parâmetros de saída. Para isto quem ativa a tarefa solicitada é a TCP e o nome da tarefa é passado pela caixa postal. Antes de terminar o processamento, a tarefa ativada deverá também ativar a TCP para passar os parâmetros de saída e desalocar o flag de sincronização.

O "message link" do GMB exemplifica o processo de acesso à uma regiao crítica tipo caixa postal.

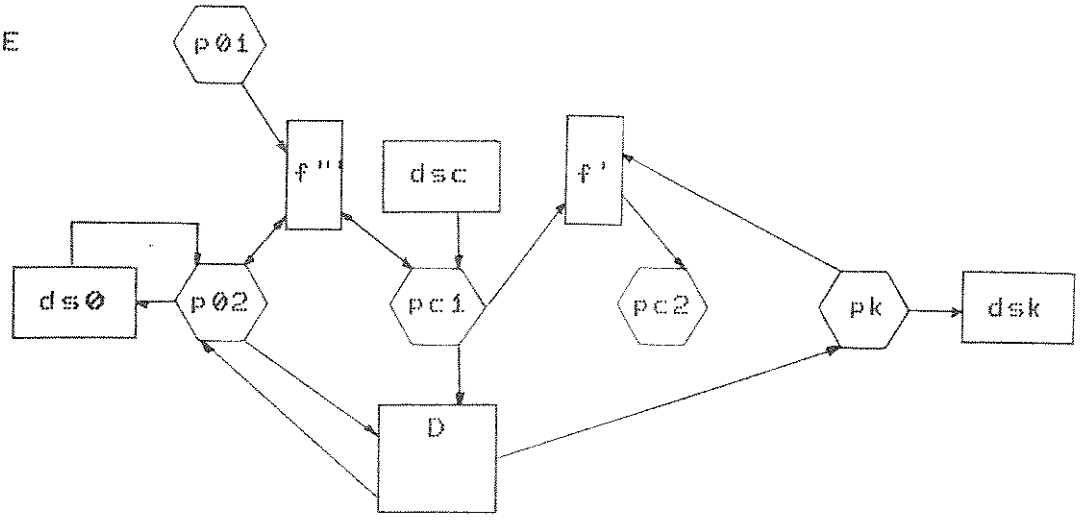
Este "message link" será modificado para modelar o processo que envolve a ativação de uma tarefa por outra via TCP. Neste caso teremos três tarefas envolvidas e não somente duas e necessitaremos de dois flags internos de sincronização, que na sincronização entre tarefas e a caixa postal serão fixos e não alocados dinamicamente.

Os Grafos de Controle e Dados abaixo exemplifica a ativação da tarefa pk pela tarefa p0 via TCP (pc).

GRAFO DE CONTROLE



GRAFO DE DADOS

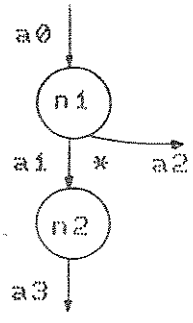


Na interpretação abaixo foi acrescentado o parâmetro RET na função ACTIVATE; este parâmetro de saída deverá indicar se a tarefa foi ativada ou não. Assim, se o procedimento foi ativado tem-se o parâmetro de retorno RET=0. Se o procedimento não pode ser ativado, o parâmetro de retorno será diferente de zero, sendo então necessário uma nova chamada da função ACTIVATE para uma nova tentativa de ativação do procedimento. Deste modo, o parâmetro de retorno RET é utilizado em um loop de espera para a ativação da tarefa; observa-se que o acesso à caixa postal é feito sempre após o loop de espera.

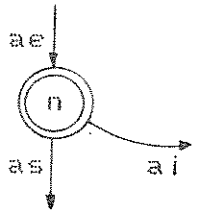
```
p01: f''=FALSE ;
     while RET.NE.0 do          ! -loop de
        ACTIVATE(pc,RET)        !  ativação
     endwhile ;                 !  da TCP
     end

p02: D=ds0 ;                    ! -passa parâmetros
     f''=TRUE ;                 !  e nome do procedimento
     while f''=TRUE do
     endwhile ;
     f0=D ;                     ! -flag alocado
     end

pk : dsk=D ;                    ! -lê parâmetros
     f'=TRUE ;
     end
```

```
pc1: while f''=FALSE do
     endwhile ;
     ALOCA FLAG (f0) ;
     D=f0 ;                          ! -envia flag alocado
     f''=FALSE ;
     f'=FALSE ;
     while RET.NE.0 do               !-ativa procedimento
        ACTIVATE(pk,RET)
     endwhile ;
     end

pc2: while f'=FALSE do               !-espera leitura
     endwhile ;                      ! da caixa postal
     end
```

Para simplificar os grafos, facilitando o entendimento, será definido o subgrafo,

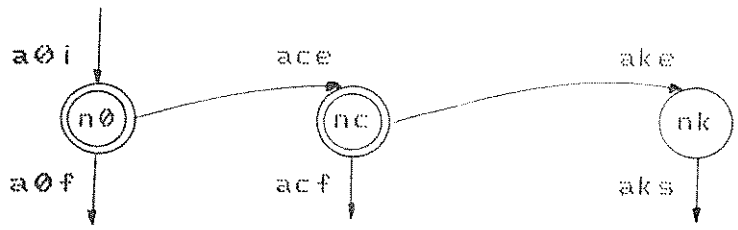


representado por

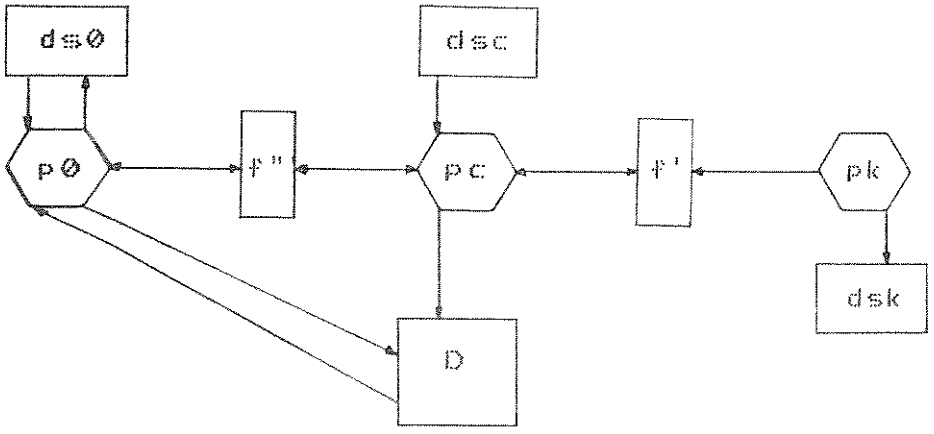


ficando as representações de dados e controle:

GRAFO DE CONTROLE



GRAFO DE DADOS



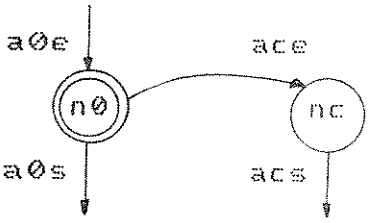
MODO_1

```
pk : dsk=D ;                          ! -1ê parâmetros
     f'=TRUE ;
     end
```

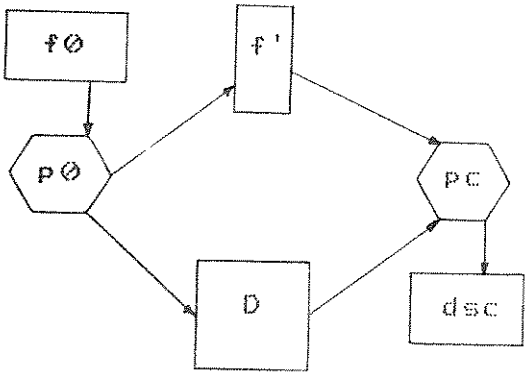
```
p0: f''=FALSE ;
    while RET.NE.0 do          ! -loop de
       ACTIVATE(pc,RET)        !  ativação
    endwhile ;                 !  da TCP
    D=ds0 ;                    ! -passa parâmetros
    f''=TRUE ;                 !  e nome do procedimento
    while f''=TRUE do
    endwhile ;
    f0=D ;                     ! -flag alocado
    end

pc: while f''=FALSE do
    endwhile ;
    ALOCA FLAG (f0) ;
    D=f0 ;                     ! -envia flag alocado
    f''=FALSE ;
    f'=FALSE ;
    while RET.NE.0 do          !-ativa procedimento
       ACTIVATE(pk,RET)
    endwhile ;
    while f'=FALSE do          !-espera leitura
    endwhile ;                 ! da caixa postal
    end
```

O procedimento para desalocar flags de sincronização será o próprio "message link" do GMB.

GRAFO DE CONTROLE



GRAFO DE DADOS



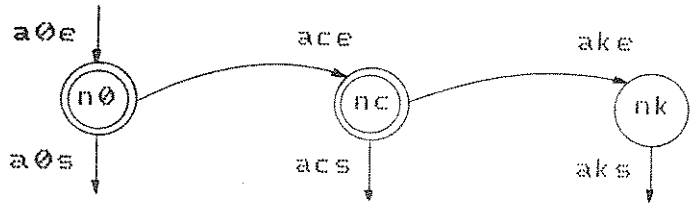
MODO_2

```
p0: CLEAR FLAG (f0) ;
    f'=FALSE ;
    while RET.NE.0 do
        ACTIVATE(pc,RET)
    endwhile ;
    D=f0 ;
    f'=TRUE;
    end
```

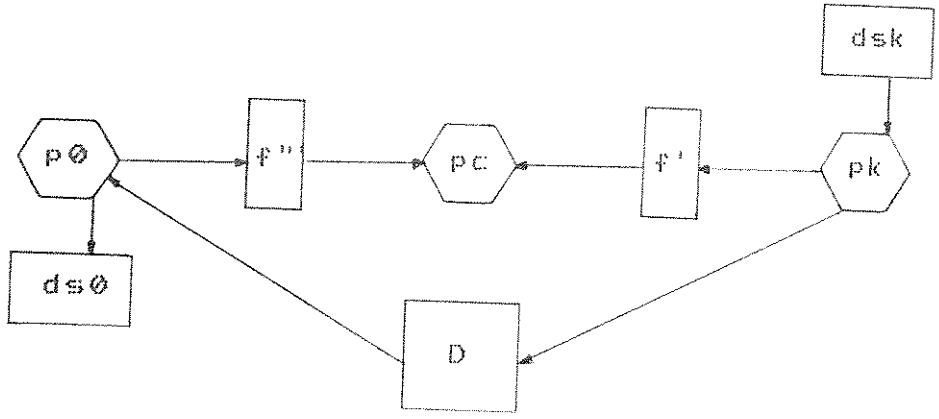
```
pc: while f'=FALSE do
    endwhile ;
    dsc=D ;
    DESALOCA FLAG (f0) ;
    end
```

Ao terminar, a tarefa pk ativa a caixa postal para passar os parâmetros de saída como pode ser visto nos Grafos abaixo.

GRAFO DE CONTROLE



GRAFO DE DADOS



MODO_3

```
p0: f''=FALSE ;
    WAIT FOR FLAG (f0) ;
    ds0=D ;
    f''=TRUE ;
    end
```

```
pk: f'=FALSE ;
    while RET.NE.0 do
      ACTIVATE(pc,RET)
    endwhile ;
    D=dsk ;
    f'=TRUE ;
    SET FLAG (f0) ;
    end
```

```
pc: while f'=FALSE do
    endwhile ;
    while f''=FALSE do
    endwhile ;
    end
```

Neste caso a TCP não trabalha nenhum dado, ela tem a função de bloquear o acesso à caixa postal.

Como conclusão , temos que no exemplo :

```
ACTIVATE (READ)
WAIT FOR FLAG (f0)
CEAR FLAG (f0)
```

- o comando ACTIVATE (READ) deve ser substituido pelo procedimento p0 do MODO_1,
- o comando WAIT FOR FLAG (f0) substituido pelo procedimento do p0 do MODO_3,
- o comando CLEAR FLAG (f0) pelo procedimento p0 do MODO_2.

Para a tarefa pk , ativada pela tarefa p0 temos :

- deve ser inicializada pelo procedimento pk do MODO_1,
- deve ser finalizada pelo procedimento pk do MODO_3.

A tarefa TCP será composta de um "case" com os tres modos :

```
case MODO do
     MODO=1  :  pc MODO_1
     MODO=2  :  pc MODO_2
     MODO=3  :  pc MODO_3
end_case
```

Com esta aproximação existe a possibilidade de "deadlock" na situação em que é pedida a ativação de um recurso já ativo. A tarefa pØ que ativa pk, o faz via TCP; esta fica em um loop de espera tentando ativar pk, que por sua vez para terminar precisa ativar a TCP.

Uma solução para o problema seria a divisão da TCP em três tarefas cada uma com uma área de passagem de parâmetros. Esta solução teria a desvantagem de bloquear a TCP de ativação, não permitindo que um outro recurso seja ativado enquanto se espera o término de uma tarefa já ativa.

Outra solução seria manter no Supervisor a informação sobre o estado dos recursos (ativo ou não ativo), e testar este estado antes de se ativar a TCP.

O procedimento p0 MODO 1 ficaria :

```
p0: while pk ativa do          ! espera até pk
    endwhile ;                 ! não ativa
    f''=FALSE ;
    while RET.NE.0 do
       ACTIVATE(pc,RET)
    endwhile ;
        .
        .
```

4.4 - CONSIDERAÇÕES FINAIS

Este estudo de paralelismo não tem a intenção de definir uma metodologia de projeto para Sistemas em Tempo Real de um modo geral, mas sómente resolver a aplicação em um Sistema PAC.

De qualquer modo as etapas seguidas para a definição da estrutura paralela se assemelham às etapas definidas em metodologias já propostas como o DARTS /GOMM 84/.

O DARTS é uma metodologia para projeto de sistemas tempo real que leva a um sistema altamente modular, com interfaces bem definidas e um reduzido acoplamento entre tarefas.

Comparando a metodologia seguida com o DARTS temos :

1) A análise do fluxo de informação no DARTS engloba dados e controle, porém no presente trabalho foi considerado mais próprio a utilização do GMB que separa o fluxo de controle e dados.

2) A decomposição das tarefas proposta seguiu os mesmos critérios do DARTS como

- dependência de E/S,
- coesão funcional,
- coesão temporal,

gerando tarefas como Supervisor módulo de inicialização, Supervisor módulo de geração de Macros, Interface com Usuário aquisição de comandos, etc.

3) A definição das interfaces entre tarefas proposta no DARTS como TCM - Tarefa de Comunicação entre Módulos e TSM - Tarefa de Sincronização entre Módulos corresponde ao estudo realizado acima. Para o problema de Sistemas PAC proposto foi escolhido um Módulo de Comunicação de Mensagens (MCM) do tipo "fortemente acoplado" conforme definido no DARTS, que implementa um "hand-shake". A implementação deste "hand-shake" implica em uma sincronização entre as tarefas. A função ACTIVATE(task), por exemplo, sincroniza e transfere parâmetros, além de englobar o que o DARTS define como Information Hiding Module (IHM) que é definido como uma estrutura de dados acessada por várias tarefas e procedimentos de acesso. O IHM equivale, na proposta apresentada, aos flags de sincronização e flags de estado dos recursos (ativo ou inativo).

4) A grande diferença entre o DARTS e a metodologia utilizada se refere ao paralelismo das tarefas. O DARTS assume que as tarefas são procedimentos sequenciais, ao passo que no Sistema PAC proposto, as tarefas são produções que podem comportar paralelismo.

CAPÍTULO 5 - EXEMPLO PARA ESPECIFICAÇÃO

Neste capítulo são apresentadas as especificações para os principais módulos do Supervisor, utilizando um exemplo de comando do usuário.

O exemplo a seguir ilustra uma alternativa para a definição dos esquemas do Supervisor, apresentando as especificações dos procedimentos com a descrição dos algorítmos, da área de comunicação e do fluxo de informação. É suposto que as funções do Núcleo PAC já estão implementadas e também as produções. Não é relevante neste estágio se as funções do Núcleo e as produções foram implementadas com paralelismo ou simplesmente ligadas através de um sequenciamento de chamadas. De qualquer modo, as produções são consideradas, no esquema de execução dos Macros, como tarefas a serem executadas sequencialmente.

Supondo definida uma linguagem intermediária (conforme 2.7.3) e que o Gerador de Macros recebe o comando :

EXE=clip, ENTRADA=tty, SAÍDA=plot/armazena(clp1)

que tem a função de executar um algorítimo de clipping, cujos dados serão fornecidos pelo usuário, cujo resultado será armazenado na Base de Dados em um registro identificado por clp1 e que gerará uma saída gráfica (plot); as produções necessárias para a execução do comando são as esquematizadas a seguir já incorporando o driver de E/S.

PRODUÇÕES

Biblioteca de Aplicações - produção CLIP

driver de entrada	lê janela de clipping lê pontos
	algoritmo de clipping
driver de saída	armazena saída clipping

Interface com Usuário - produção TTYCLP

	executa diálogo
driver de saída	armazena janela clipping armazena pontos

Pacote Gráfico - produção PLOTCLP

driver de entrada

| lê saída clipping |

| executa saída gráfica |

Banco de Dados - produções

| lê janela clipping |

| lê pontos |

| lê saída clipping |

| armazena janela clipping |

| armazena pontos |

| armazena saída clipping |

A Interface com o Usuário tem a função de gerar a sequência correta das tarefas (produções) a serem ativadas. O Supervisor deverá determinar o fluxo de dados, identificando os registros a serem manipulados e criando, se necessário, nomes internos para os registros temporários.

Utilizando-se as produções descritas, a sequência das produções para a execução do comando do exemplo será :

TTYCLP - executa o diálogo e através do driver de saída gera o registro temporário TMP1

CLIP - executa o algorítmo de clipping cujo driver de entrada lê o registro temporário TMP1 e cujo driver de saída gera o registro CLP1

PLOTCLP- executa a saída gráfica tendo como entrada o driver que lê o registro CLP1

Para exemplificar a execução de um Macro, a seguir, será mantida a notação ACTIVATE, CLEAR FLAG e WAIT FOR FLAG, devendo porém ser lembrado que estes comandos se referem aos procedimentos descritos no capítulo anterior (4.3) como p0 MODO 1, p0 MODO 2 e p0 MODO 3 respectivamente.

5.1 - SUPERVISOR

A especificação abaixo exemplifica o funcionamento global do Supervisor.

Para a execução de cada comando do usuário é utilizada uma "célula de execução" que será descrita a seguir.

```
REPEAT_UNTIL comando=END
        ACTIVATE (dialogo)
        WAIT FOR FLAG (f)          /* sincronismo */
        CLEAR FLAG (f)
        ACTIVATE (gerador_macro)
        WAIT FOR FLAG (f)          /* sincronismo */
        CLEAR FLAG (f)
        aloca celula livre
        CASE celula alocada
              celula=1: ACTIVATE (celula_1)
              celula=2: ACTIVATE (celula_2)
                 .
                 .
              celula=n: ACTIVATE (celula_n)
        END
END
```

Inicialmente o Supervisor ativa um procedimento de diálogo para obter o comando. A seguir este comando será passado para o Gerador de Macro que tem a função de preencher uma área de Macro com a sequência de tarefas a serem ativadas, bem como determinar o fluxo de dados através da identificação dos registros. Será então alocada uma célula ao comando e ativada a seguir.

Deve ser observado que não existe sincronização após a ativação da célula, o comando será executado ao mesmo tempo que o Supervisor volta a ativar o processamento de diálogo para a entrada de um novo comando.

5.2 - SUPERVISOR : EXECUÇÃO DO MACRO

A especificação abaixo mostra como a sequência de tarefas componentes do comando é executada.

```
MACRO EXECUÇÃO - CÉLULA

área de Macro - #tarefas, tarefas, data record
BEGIN
     DO_WHILE #tarefas >= 1
                ACTIVATE (tarefa_i)
                WAIT FOR FLAG (f)
                CLEAR FLAG (f)
                #tarefas = #tarefas -1
     END_DO
     desaloca célula
END
```

A área de Macro contém as informações de controle necessárias para a execução do comando. Deve ser observado que na execução do Macro a caixa postal não manipula dados da aplicação, sómente dados de controle :

- nome da tarefa
- flag de sincronização
- identificação do registro a ser lido/escrito

Cada uma das tarefas celula_1, celula_2, ... , celula_n, ativadas pelo Supervisor, no esquema anterior, é a repetição do esquema MACRO-EXECUÇÃO acima. A opção de repetir o procedimento em várias células foi para manter a característica monousuário (não reentrante) da função ACTIVATE, ou seja, uma tarefa ativada só poderá ser novamente ativada após o seu término. Esta característica é fundamental para o funcionamento correto da caixa postal e todo o estudo do fluxo de controle no sistema, foi basedo nesta característica (GMB).

A opção foi feita baseada em dois pontos:

- a quantidade de comandos que serão executados simultaneamente é pequena, sendo portanto pequena, a quantidade de código repetido das células de execução,
- o esquema da célula é simples devendo gerar pouco código de máquina.

5.3 - SUPERVISOR : GERADOR DE MACRO

A Interface com o Usuário, através do procedimento de diálogo, deverá fornecer ao Gerador de Macro a informação de quais tarefas deverão ser executadas e os dados de controle necessários para a sua execução. Sendo esta comunicação transparente ao usuário, o protocolo de comunicação deverá ser otimizado em relação ao tempo de processamento. Assim foi escolhida uma linguagem de comunicação do tipo L4 /GILO 78/ que consta basicamente de um código de operação e uma lista de dados associada.

O comando do usuário, processado pelo processador de diálogo, pode ser resumido como um procedimento a ser executado, EXE = procedimento, cujos dados de entrada serão especificados, ENTRADA = xxx, e cujos resultados serão processados conforme especificados pela saída, SAÍDA = yyy. Para a entrada e saída poderão ser especificado modificadores conforme descrito anteriormente.

5.3.1 - PROCESSAMENTO DA ENTRADA

A entrada de dados poderá ser dividida em três modos:

-entrada por diálogo,

-entrada por arquivo,

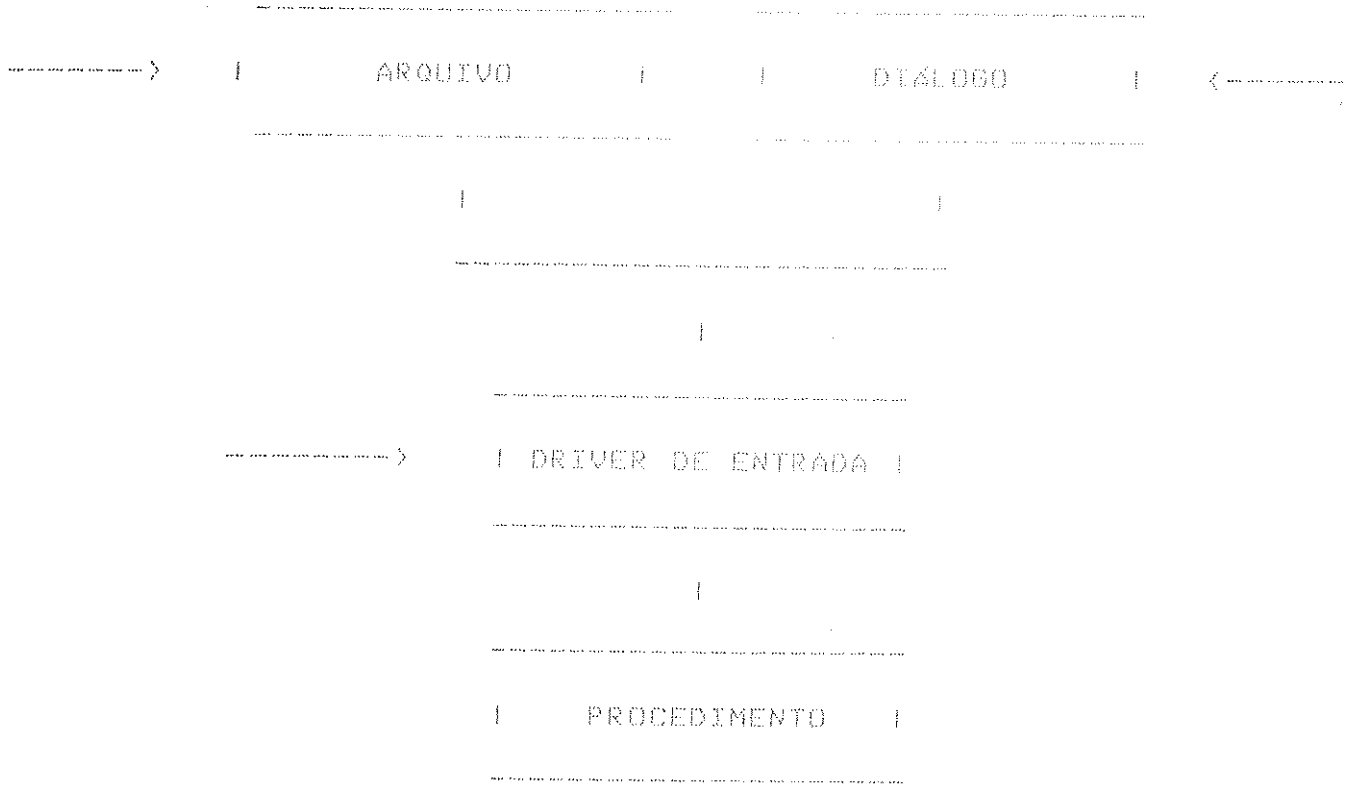
-entrada pelo Banco de Dados.

Modos mistos, como por exemplo arquivo e Banco de Dados, deverão estar englobados em um procedimento de diálogo, sendo então considerados como entrada por diálogo. A aquisição de dados "on line" também deverá ser feita através de um procedimento que pode ser considerado como pertencente ao subsistema Interface com Usuário (entrada por diálogo), ou ao subsistema Biblioteca de Aplicações.

Conforme descrito anteriormente, qualquer procedimento deverá receber os dados de entrada através de uma interface padrão com o Banco de Dados (Driver de Entrada).

Assim, se for especificada entrada por arquivo, deverá ser ativado um procedimento que leia o arquivo e o armazene em registros temporários no Banco de Dados, sendo estes registros lidos, em seguida, pelo driver de entrada do procedimento. Do mesmo modo, se for especificada entrada por diálogo, também serão gerados registros temporários que farão a ligação do diálogo com o procedimento, através do driver de entrada.

A figura a seguir mostra esta hierarquia.



Códigos para entrada:

-código = 11; entrada pelo Banco de Dados

-código = 12; entrada pelo Banco de Dados, registro indireto

-código = 13, entrada por arquivo

-código = 14, entrada por diálogo

Lista de dados associada

-código 11

-nome do driver de entrada

-nome dos registros

-#tipos de dados, tipos de dados

-código 12

-nome do driver de entrada

-nome do registro indireto

-#tipos de dados, tipos de dados

-código 13

-nome do programa que lê arquivo

-nome do arquivo

-nome do driver de entrada

-#tipos de dados, tipos de dados

-código 14

-nome do programa de diálogo

-nome do driver de entrada

-#tipos de dados, tipos de dados

Em complemento às informações fornecidas pela Interface com o Usuário, o Gerador de Macro deve suprir:

-código 12 : -nome dos registros

-código 13 : -nome dos registros temporários

-código 14 : -nome dos registros temporários

```
** ESPECIFICAÇÃO PARA O GERADOR DE MACRO - ENTRADA **


   CASE codigo

        codigo=11: i = i + 1;
                   TAREFA(i) = nome do driver de entrada;
                   DATA RECORD <--  nome dos registros;
        codigo=12: i = i + 1;
                   TAREFA(i) = nome do driver de entrada;
                   READ registro indireto;
                   DATA RECORD <---nome dos registros;
        codigo=13: i = i + 1;
                   TAREFA(i) = nome do programa que lê arquivo;
                   DATA RECORD <--- nome do arquivo;
                   aloca registros temporários;
                   DATA RECORD <--- nome registros temporários;
                   i = i + 1;
                   TAREFA(i) = nome do driver de entrada;
        codigo=14: i = i + 1;
                   TAREFA(i) = nome do procedimento de diálogo;
                   aloca registros temporários;
                   DATA RECORD <--- nome registros temporários;
                   i = i + 1;
                   TAREFA(i) = nome do driver de entrada;
   END CASE
```

5.3.2 - PROCESSAMENTO DOS MODIFICADORES DE ENTRADA

Na implementação do módulo de diálogo, poderão ser definidos vários modificadores de entrada. Porém, para o Supervisor, os diferentes modificadores são funcionalmente equivalentes; eles identificam procedimentos a serem excutados, cujos dados de entrada e saída serão acessados via Banco de Dados.

Deste modo, a Interface com o Usuário gerará um único código para identificar o modificador de entrada.

Código para modificadores de entrada:

-código = 20; modificador de entrada

Lista de dados associada:

-nome do programa

-dados de controle

** ESPECIFICAÇÃO PARA O GERADOR DE MACRO - MODIFICADORES DE ENTRADA **

```
CASE codigo
     codigo=20: i = i + 1,
                TAREFA(i) = nome do programa;
                DATA RECORD <--- dados de controle;
END CASE
```

5.3.3 - PROCESSAMENTO DO PROCEDIMENTO A SER EXECUTADO

Para o Supervisor, os diferentes procedimentos são funcionalmente equivalentes, pois os dados de entrada e saída serão acessados via Banco de Dados. Portanto, a Interface com o Usuário gerará um único código identificando a execução de um procedimento.

Código para o procedimento a ser executado:

-código = 30; procedimento a ser executado

Lista de dados associada:

-nome do programa

-dados de controle

** ESPECIFICAÇÃO PARA O GERADOR DE MACRO - EXECUÇÃO DO PROGRAMA **

```
CASE codigo
     codigo=30: i = i + 1;
                TAREFA(i) = nome do programa;
                DATA RECORD <--- dados de controle;
END CASE
```

5.3.4 - PROCESSAMENTO DA SAÍDA

De maneira similar à entrada de dados, a saída de dados pode ser dividida em três modos:

-saída por diálogo (TTY ou gráfico),

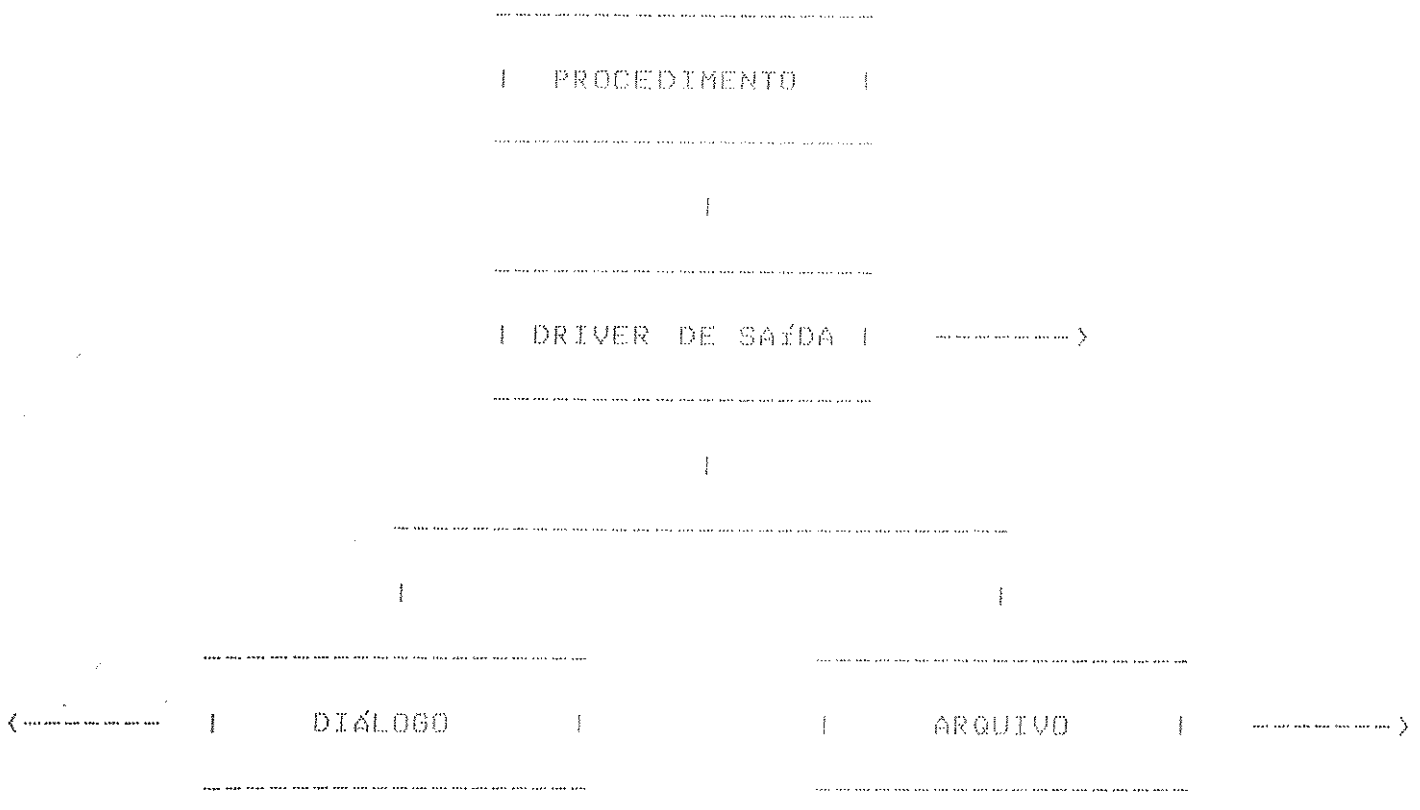
-saída por arquivo,

-saída para o Banco de Dados.

Conforme descrito anteriormente, a saída de qualquer procedimento será sempre para o Banco de Dados através do Driver de Saída.

Assim a saída para um arquivo deverá ativar um procedimento para a geração do arquivo através dos dados armazenados no Banco de Dados. Do mesmo modo, a saída por diálogo deverá acessar os dados no Banco de Dados.

A figura abaixo mostra a hierarquia.



Códigos para saída:

-código = 41: saída para o Banco de Dados

-código = 42: saída para arquivo

-código = 43: saída para diálogo

Lista de dados associada:

-código 41

-nome do driver de saída

-nome dos registros

-#tipos de dados, tipos de dados

-código 42

-nome do driver de saída

-nome do programa que gera arquivo

-nome do arquivo

-#tipos de dados, tipos de dados

-código 43

-nome do driver de saída

-nome do programa de diálogo

-#tipos de dados, tipos de dados

Em complemento às informações recebidas da Interface com o Usuário,o Gerador de Macro deve suprir:

-código 42 : -nome dos registros temporários

-código 43 : -nome dos registros temporários

** ESPECIFICAÇÃO PARA O GERADOR DE MACRO - SAÍDA **

```
CASE codigo
        codigo=41:  i = i + 1;
                    TAREFA(i) = nome do driver de saída;
                    DATA RECORD <--- nome dos registros;
        codigo=42:  i = i + 1;
                    TAREFA(i) = nome do driver de saída;
                    aloca registros temporários;
                    DATA RECORD <--- nome registros temporários;
                    i = i + 1;
                    TAREFA(i) =nome do programa que gera arquivo;
                    DATA RECORD <--- nome do arquivo
        codigo=43:  i = i + 1;
                    TAREFA(i) = nome do driver de saída;
                    aloca registros temporários;
                    DATA RECORD <--- nome registros temporários;
                    i = i + 1;
                    TAREFA(i) = nome do procedimento de diálogo;
END CASE
```

5.3.5 - PROCESSAMENTO DOS MODIFICADORES DE SAÍDA

A mesma argumentação descrita para os modificadores de entrada, é válida para justificar a utilização de um único código para identificar o modificador de saída.

Código para os modificadores de saída:

-código = 50, modificador de saída

Lista de dados associada:

-nome do programa

-dados de controle

** ESPECIFICAÇÃO PARA O GERADOR DE MACRO - MODIFICADORES DE SAÍDA **

```
CASE codigo
     codigo=50: i = i + 1;
                TAREFA(i) = nome do programa;
                DATA RECORD (---- dados de controle;
END CASE
```

5.3.6 - PROCESSAMENTO GLOBAL DO COMANDO DO USUÁRIO

É de responsabilidade da Interface com Usuário a geração e sequenciamento dos códigos bem como o fornecimento da lista de dados associada.

O Gerador de Macro irá receber estes códigos e lista de dados associada, alocar uma área de Macro e em um loop preencher a área de Macro (#TAREFAS, TAREFAS, DATA RECORD).

```
** ESPECIFICAÇÃO - GERADOR DA MACROS **
   BEGIN:
            área de códigos, #CÓDIGOS, CÓDIGOS, DATA RECORD;
            i = 0;
            k = 1;
            DO WHILE k<= #codigos
                        CASE codigo
                                    _____
                                    _____
                                    _____
                                    _____
                                    _____
                        END CASE
            k = k + 1;
            END DO
   END
```

Como pode ser observado no Grafo de Dados do Sistema (seção 2.2.6), os códigos e lista de dados associada são enviados diretamente ao Supervisor, sem passar pelo Banco de Dados, via estrutura interna de dados (área de códigos). Do mesmo modo o Gerador de Macro e o módulo de Macro Execução se comunicam via estrutura interna de dados (área de macro).

CAPÍTULO 6 - ANÁLISE POR REDES DE PETRI

O Sistema proposto será analizado a seguir com o intuito de detetar a ocorrência de "deadlocks", utilizando como ferramenta as Redes de Petri.

A rede de Petri N=(P, T, I, O) / PETE 81/ é composta por um conjunto de "lugares" P e por um conjunto de transições T ligados através de arcos. As transições são ativadas por meio de marcas alojadas nos "lugares". A regra de ativação para uma transição tj é dada pelo conjunto I(tj) = [Pi], sendo [Pi] o conjunto de lugares que estão ligados à transição tj por arcos de entrada. Quando todos os lugares do conjunto [Pi] possuirem marcas a transição pode ser ativada. Ao ser ativada a transição, as marcas de I(tj) são consumidas e são geradas marcas nos lugares do conjunto dado por O(tj) = [Po], conjunto dos lugares que estão ligados à transiçao tj por arcos de saída.

A seguir serão apresentados os conceitos utilizados na análise da rede.

- REDE PURA

Uma rede é dita pura se para todo t c T os conjuntos I(t) e O(t) são disjuntos.

- MATRIZ DE INCIDÊNCIA

Para uma rede pura pode ser definida a matriz de incidência C como:

$$\text{para } p \in P \text{ e } t \in T \qquad C(p,t) = \begin{cases} -1 & \text{se } p \in I(t) \\ 1 & \text{se } p \in O(t) \\ 0 & \text{caso contrário} \end{cases}$$

- ESTADO DO SISTEMA

O estado do sistema em um dado instante é dado pelo vetor

$$M = \begin{vmatrix} n1 \\ n2 \\ \cdot \\ \cdot \\ nk \end{vmatrix}$$

onde n1 é a quantidade de marcas no lugar li em um dado instante.

Se a quantidade de marcas nos lugares em qualquer instante for sempre 0 ou 1, o estado em um dado instante pode ser representado pelo conjunto de lugares que possuem marcas.

A partir de um estado M, o conjunto de todos os estados que o sistema pode alcançar, através de diferentes transições, é representado por [M].

O estado do sistema em um determinado instante é também denominado MARCAÇÃO.

- INVARIANTES

Invariantes (Iv) são soluções simples de inteiros não negativos da equação

$$C'M = 0 \quad ; \quad (6.1)$$

onde uma solução simples é a solução que não é composta pela soma de outras soluções.

Se para $\forall p \in P$, $Iv(p) \in (0, 1)$

então as invariantes obtidas são denominadas vetores característicos e podem ser identificadas pelo conjunto de lugares que compõem a solução da equação 6.1.

- MARCAÇÃO VIVA

Uma marcação é viva se para $\forall Mi \in [M]$, $[EMij] > 1$.

Um teorema diz que a marcação é viva se e somente se $Mi'Ivj >= 1$ para todas as inavriantes.

- DEADLOCK

Se uma marcação $Mi \in [M]$ é viva, cada marcação alcançada a partir de Mi pode ser transformada em outra marcação. Esta definição de marcação viva na rede de Petri corresponde ao conceito da não existência de deadlock no sistema.

Assim se todas as marcações pertencem à um único conjunto [M], testando-se a marcação inicial pode-seconcluir se o sietma possui ou não deadlock.

6.1 - REDE DE PETRI PARA O SISTEMA PAC

Associando os lugares :

Si - Supervisor, esquema para inicialização

S-f - Supervisor, esquema para finalização

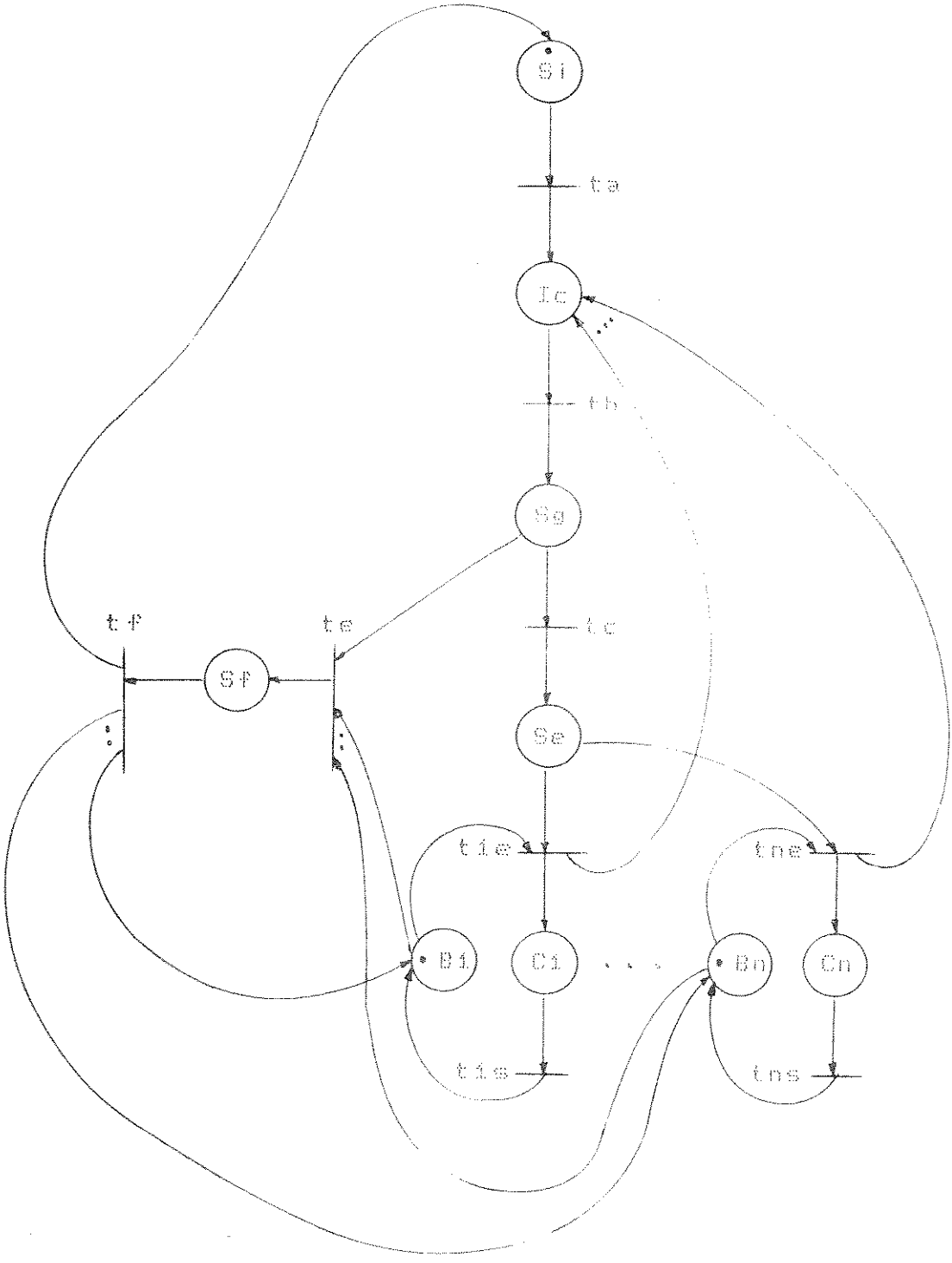
Sg - Supervisor, esquema para geração do comando

Se - Supervisor, esquema para execução do comando

Ic - Subsistema Interface com Usuário, obtenção do comando

Ci - Células de MACRO_EXECUÇÃO

temos a rede de Petri para o Sistema PAC representada a seguir.


Si
ta
Ic
tb
Sg
tc
Se
tie
tne
B1
Ci
Bn
Cn
tis
tns
tf
te
Sf

P = (Si, Ic, Sg, Se, Sf, C1, B1, ... , Cn, Bn)

T = (ta, tb, tc, te, tf, tie, tis, ... , tne, tns)

I(ta) = (Si)
I(tb) = (Ic)
I(tc) = (Sg)
I(te) = (Sg, B1, ... , Bn)
I(tf) = (Sf)

O(ta) = (Ic)
O(tb) = (Sg)
O(tc) = (Se)
O(te) = (Sf)
O(tf) = (Si, B1, ... , Bn)

para 1<= j <=n

I(tje) = (Se, Bj)
I(tjs) = (Cj)

O(tje) = (Cj, Ic)
O(tjs) = (Bj)

-ANÁLISE DA REDE

A rede apresentada é pura e assim podemos obter a matriz de incidência.

MATRIZ DE INCIDÊNCIA

	ta	tb	tc	tf	te	tie	tis	...	tne	tns
Si	-1	0	0	1	0	0	0	...	0	0
Sg	0	1	-1	0	-1	0	0	...	0	0
Sf	0	0	0	-1	1	0	0	...	0	0
C1	0	0	0	0	0	1	-1	...	0	0
B1	0	0	0	1	-1	-1	1	...	0	0
...	...	...	...	...	...	...	...	...	...	...
Cn	0	0	0	0	0	0	0	...	1	-1
Bn	0	0	0	1	-1	0	0	...	-1	1
Ic	1	-1	0	0	0	1	0	...	1	0
Se	0	0	1	0	0	-1	0	...	-1	0

O estado do Sistema em um dado instante é dado pelo vetor

VETOR DE ESTADO

$$M = \begin{vmatrix} mSi \\ mSg \\ mSf \\ mC1 \\ mB1 \\ \cdot \\ mCn \\ mBn \\ mIc \\ mSe \end{vmatrix}$$

Sendo a quantidade de marcas nos lugares em qualquer instante sempre 0 ou 1, o estado pode ser definido como o conjunto de lugares que possuem marcas. Assim o estado inicial do Sistema será:

ESTADO INICIAL

$$M0 = \{ Si, B1, \ldots, Bn \}$$

Na rede apresentada todos os estados do Sistema pertencem a um único conjunto [M].

CÁLCULO DAS INVARIANTES

Da equação 6.1 obtemos :

$$\begin{aligned} -mSi + mIc &= 0 \\ mSg - mIc &= 0 \qquad (1) \\ -mSg + mSe &= 0 \end{aligned}$$

para $1 <= j <= n$

$$\begin{aligned} mCi - mBi + mIc - mSe &= 0 \qquad (2) \\ -mCi + mBi &= 0 \end{aligned}$$

para $1 <= j <= n$

$$\begin{aligned} mSi - mSf + mB1 + \ldots + mBn &= 0 \qquad (3) \\ -mSg + mSf - mB1 - \ldots - mBn &= 0 \end{aligned}$$

De (1) temos

$$mSi = mIc = mSg = mSe = A$$

De (2) temos

$$mCj = mBj \qquad 1 <= j <= n$$

De (3) temos

$$mSf = A \quad \text{para } mBj = 0 \qquad 1 <= j <= n$$

para $1 <= j <= n$

$$mBj = mSf \quad e \quad mBk = 0 \quad 1 <= k <= n \ , \ k \neq j$$

As invariantes obtidas são "vetores característicos", podemos assim identificar as invariantes obtidas pelos conjuntos :

$I0 = (Si, Sg, Sf, Ic, Se)$

$Ij = (Sf, Bj, Cj) \qquad 1 <= j <= n$

VERIFICAÇÃO DE DEADLOCK

Como todos os estados do Sistema pertencem à um único conjunto [M], testando a marcação inicial podemos verificar se todas as marcações são vivas, e portanto, se o Sistema não tem deadlock.

Assim temos que :

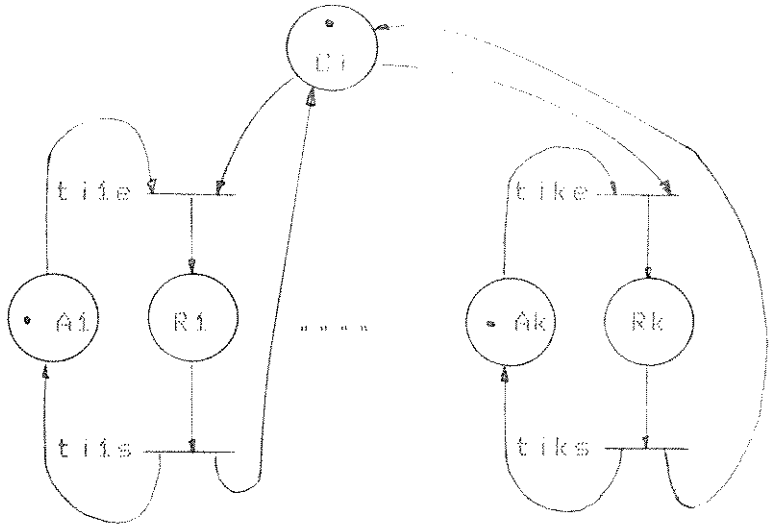
$$M0'I0 = 1$$

$$M0'Ij = 1 \qquad 1 <= j <= n$$

portanto o Sistema não tem deadlock.

6.2 - SUBREDE : ACESSO DE UMA CÉLULA AOS RECURSOS

Cada célula Ci acessa os recursos dos subsistemas por diferentes transições, conforme a rede da célula abaixo.



P = (Ci, R1, A1, ... , Rk, Ak)

T = (tije, tijs, ... , tike, tiks)

para 1<= j <=k

I(tije) = (Ci, Aj) O(tije) = (Rj)

I(tijs) = (Rj) O(tijs) = (Ci, Aj)

-ANÁLISE DA SUBREDE

MATRIZ DE INCIDÊNCIA

	tile	tils	...	tike	tiks
Ci	-1	1	...	-1	1
R1	1	-1	...	0	0
A1	-1	1	...	0	0
...	...	...	...	...	...
Rk	0	0	...	1	-1
Ak	0	0	...	-1	1

VETOR DE ESTADO

$$M = \begin{vmatrix} mCi \\ mR1 \\ mA1 \\ . \\ mRk \\ mAk \end{vmatrix}$$

ESTADO INICIAL

M0 = (Ci, A1, ... , Ak)

Todos os estados possíveis da rede pertencem a um único conjunto [M].

CÁLCULO DAS INVARIANTES

Da equação C'M = 0 temos :

$$-mCi + mRj - mAj = 0$$

$$mCi - mRj + mAj = 0 \qquad \text{para} \quad 1 <= j <= k$$

donde obtemos as invariantes

$$I0 = (Ci, R1, \ldots, Rk)$$

$$Ij = (Rj, Aj) \quad \text{para} \quad 1 <= j <= k$$

VERIFICAÇÃO DE DEADLOCK

$$M0' I0 = 1$$

$$M0' Ij = 1 \quad \text{para} \quad 1 <= j <= k$$

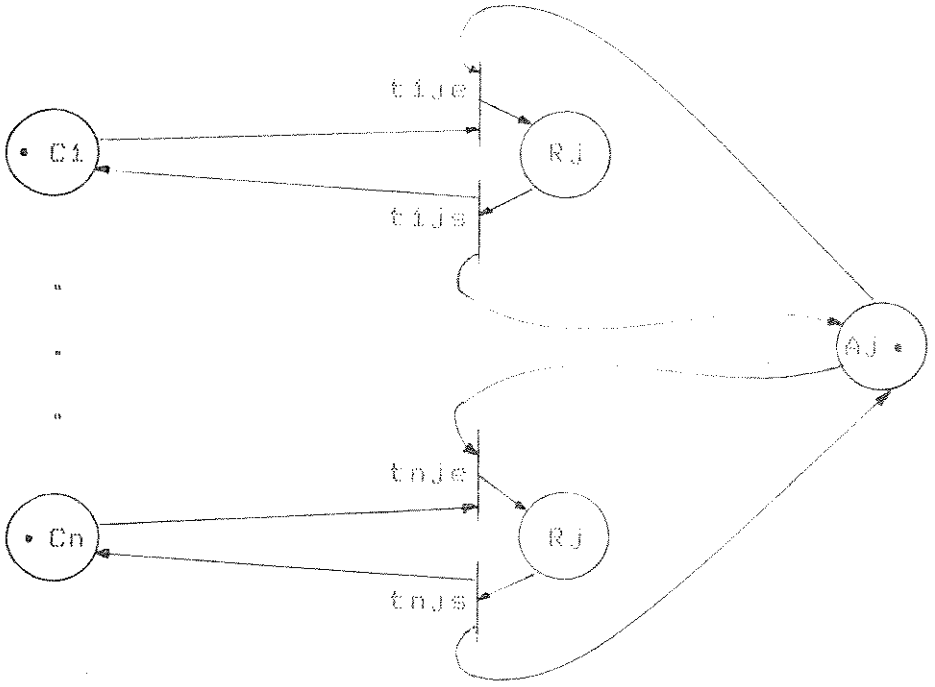
donde podemos afirmar que não existe deadlock.

6.3 - SUBREDE : ACESSO DE VÁRIAS CÉLULAS A UM RECURSO

A célula i acessa o recurso j através das transições tije e tijs, diferentes das transições tije e tijs que uma célula i acessaria o mesmo recurso.

Para que duas células não acessem o mesmo recurso simultaneamente será colocado um bloqueio (Aj) para cada recurso Rj. O bloqueio Aj está inserido na função ACTIVATE do esquema de execução da célula.

A rede abaixo representa a condição em que n células tentam acessar um único recurso.



P = { Rj, Aj, C1, ... , Cn, }

T = { tije, tijs, ... , tnje, tnjs }

para 1 <= k <=n

I(tkje) = { Ck, Aj } O(tkje) = { Rj }

I(tkjs) = { Rj } O(tkjs) = { Ck, Aj }

-ANÁLISE DA SUBREDE

MATRIZ DE INCIDÊNCIA

	t1je	t1js	...	tnje	tnjs
Rj	1	-1	...	1	-1
Aj	-1	1	...	-1	1
C1	-1	1	...	0	0
...	...	...	...	...	...
Cn	0	0	...	-1	1

VETOR DE ESTADO

$$M = \begin{vmatrix} mRj \\ mAj \\ mC1 \\ . \\ mCn \end{vmatrix}$$

ESTADO INICIAL

M0 = { Aj, C1, ... , Cn }

Todos os estados possíveis da rede pertencem a um único conjunto [M].

CÁLCULO DAS INVARIANTES

Da equação C'M = 0 temos :

$$mRj - mAj - mCk = 0$$

$$-mRj + mAj + mCk = 0 \qquad \text{para} \quad 1 <= k <= n$$

donde obtemos as invariantes

$$I0 = (Rj, Aj)$$

$$Ik = (Rj, Ck) \quad \text{para} \quad 1 <= k <= n$$

VERIFICAÇÃO DE DEADLOCK

$$M0'I0 = 1$$

$$M0'Ik = 1 \quad \text{para} \quad 1 <= k <= n$$

donde podemos afirmar que não existe deadlock.

CAPÍTULO 7 - CONCLUSÕES

Este estudo faz parte do trabalho de um grupo de pesquisadores da Faculdade de Engenharia Elétrica da UNICAMP, que tem por finalidade o projeto e implementação de um Sistema PAC para Controle de Processos.

O trabalho foi dividido em :

- implementação de um pacote gráfico segundo norma GKS,
- definição e implementação de um Banco de Dados para PAC através da expansão do modelo Entidade-Relacionamento,
- definição e implementação da Interface com Usuário,
- definição e implementação da Biblioteca de Aplicações,
- definição e implementação dos Utilitários,
- integração.

O pacote gráfico segundo a norma GKS foi implementado utilizando-se o VAX/780 do Instituto de Física da UNICAMP. A definição do Banco de Dados é tema de tese de mestrado em conclusão /DELG 84/ /DELG 85/, e a implementação é tema de tese de mestrado em fase final.

A Interface com Usuário se encontra no estágio de estudos, já existindo parte concluida /SILV 85/. A Aplicação é assunto de estudos há muitos anos, já existindo uma série de algoritmos implementados.

O tema desta tese foi decorrência do estudo da integração dos pacotes a serem gerados.

A necessidade de integração de diferentes pacotes gerou o estudo dos módulos lógicos componentes do Sistema PAC realizado no Cap. 2 deste texto. Este estudo constou da definição e descrição dos módulos lógicos, incluindo o fluxo de dados e controle entre eles /RUGG 78/.

Esta modularidade foi expandida com o conceito de Núcleo PAC, discutido no Cap. 3. A definição detalhada das funções do Núcleo PAC, como foi dito, é uma tarefa difícil neste estágio, devendo tornar-se viável quando os pacotes estiverem prontos e a integração for iniciada.

O estudo de paralelismo realizado no Cap. 4 foi motivado principalmente, pela necessidade de se implementar a característica de se ter tarefas de longa duração sendo processadas, ao mesmo tempo em que o usuário trabalha em uma tarefa interativa /ASTR 81/. Os algorítmos desenvolvidos neste capítulo foram testados utilizando o PDP 11/45, instalado na Faculdade de Engenharia Elétrica, cujo Sistema Operacional (RSX 11M) possui diretivas para sincronização e comunicação entre tarefas em tempo real.

No capítulo 5 foram especificados, através de um exemplo, os algorítmos para o Supervisor.

Algumas premissas podem ser revistas, como por exemplo :

- Seria de responsabilidade da Interface com o Usuário definir qual o sequenciamento das tarefas a serem executadas, ou deveria caber ao Supervisor ?
- Deveria o Macro ser processado sequencialmente, quando existirem produções que podem ser ativadas simultaneamente ?

Estas premissas foram colocadas para se obter uma maior simplicidade. Porém se desta simplicidade decorrer, por exemplo, uma muito maior complexidade para a Interface com Usuário, o esquema poderá ser revisto. Do mesmo modo poderá ser revisto o esquema do Supervisor - Macro Execução, para termos processamento paralelo de produções, caso existam ferramentas que simplifiquem a sua especificação /CLAR 84/.

Para finalizar, o capítulo 6 verificou teoricamente a correção do Sistema proposto, através de uma prova de que este não possui "deadlocks" /LAUT 74/.

O modelo de Sistema PAC proposto é uma evolução em relação aos sistemas integrados. Sua principal característica é a modularidade permitindo, se necessário, que novos módulos (subsistemas) sejam introduzidos. Assim, apesar de proposto no capítulo 2, que o subsistema Biblioteca de Aplicações seja único, o modelo ainda será válido se na implementação forem definidos vários subsistemas de Aplicação. O Núcleo PAC, que define uma interface para com os subsistemas, também possui a flexibilidade de permitir a introdução de novas funções.

O modelo proposto é geral, podendo cada Sistema PAC implementado incorporar características operacionais específicas. A característica operacional considerada neste estudo foi a execução em paralelo de comandos do usuário.

O estudo de paralelismo realizado contribui como um guia para a implementação e envolve diferentes níveis de especificação a saber:

- especificação do Sistema PAC segundo o GMB (capítulo 2) e análise segundo Redes de Petri (capítulo 6),
- especificação dos módulos do Supervisor (capítulo 5 e apêndice A),
- especificação dos módulos de ativação, sincronização e passagem de parâmetros em tarefas concorrentes (capítulo 4),
- especificação de ferramentas para a geração automática de funções do Núcleo PAC que comportem paralelismo (capítulo 3 e apêndice B).

Como continuação deste trabalho é proposta a especificação das funções do Núcleo PAC. Para isto, será necessário um aprofundamento maior em cada uma das áreas associadas :

- Sistemas operacionais,
- Banco de Dados,
- Pacotes Gráficos,
- Interfaces com o Usuário,
- Utilitários e
- Aplicação.

BIBLIOGRAFIA

/ANDR 85/ Andre F., Herman D., Verjus J.P., (1985). Synchronization of Parallel Programs, North Oxford Academic.

/ALLA 72/ Allan III J.J., (1972). Foundations of the many manifestation of Computer Augmented Design, Proceedings of the IFIP Working Conference on Principles of Computer-Aided Desing, North Holland.

/ALLW 81/ Allworth S.T., (1981). Introduction to Real-Time Software Design, MacMillan.

/ASTR 81/ Astrom K.J., Elmquist H., (1981). Perspective on Interactive Softtware for Computer Aided Modelling and Desing of Control Systems, Proceedings of Conference on Decision and Control, 1981.

/BEIE 78/ Beier K.-P., Jonas W., (1978). DINAS - A Transportable Executive System for Interactive Computer Aided Design, Conference on Interactive Techniques in Computer-Aided Design, Bologna, Italy.

/BO 80/ Bo K., (1980). Man-Machine Interaction, Lecture Notes in Computer Science, vol.89, cap.10, Springer Verlag.

/BORU 80/ Borufka H.G., Pfaff G., (1980). The Design of a General Purpose Command Interpreter for Graphical Man-Machine Communication, IFIP Conference on Man-Machine Communication in CAD/CAM, Japan.

/BORU 82/ Borufka H.G., ten Hagen P.J.W., Kuhlmann H.W., Weber H.R., (1982). Dialogue Cells : A Method for Defining Interactions. IEEE Computer Graphics and Applications, July/82.

/CAMI 85/ Camilo D., (1985). Tese de doutorado em fase de conclusão, FEC/UNICAMP

/CHEN 76/ Chen P.P.-S., (1976). The Entity-Relationship Model - Toward a Unified View of Data, ACM Transations on Database Systems, vol.1, n.1, MAR/76.

/CLAR 84/ Clark K.L., McCabe F.G., (1984). Micro-PROLOG : Programming in Logic, Prentice/Hall International.

/COM 1/ Centro de Pesquisas da Petrobrás - Rio de Janeiro. Comunicação pessoal.

/DATE 76/ Date C.J., (1976). An Introduction to Data Base Systems, Addison-Wesley

/DELG 84/ Delgado A.L.N., (1984). Banco de Dados no Contexto de PAC, Relatório de atividades da FAPESP, agosto/84 a janeiro/85, processo 83/1813-8.

/DELG 85/ Delgado A.L.N., (1985). Banco de Dados no Contexto de PAC, Relatório de atividades da FAPESP, fevereiro/84 a julho/85, processo 83/1813-8.

/ENCA 77/ Encarnação J., (1977). Sistemas de Informação Gráfica, Informática, Lisboa, AGO/OUT 77.

/ENCA 79/ Encarnação J., Neumann T., (1979). A Survey of DB Requirements for Graphical Applications in Engineering, International Conference on Data Bases for Pictorial Applications, Florence, Italy.

/ENCA 83/ Encarnação J., Schlechtendahl E.G., (1983). Computer Aided Design Fundamentals and Systems Architectures, Springer Verlag.

/FOLE 74/ Foley J.D., (1974). The Art of Natural Graphic Man-Machine Conversation, Proceedings IEEE, APR/74.

/GARR 75/ Garrocq C.A., Hurley M.J., (1975). The IPAD System: A Future Management/Engineering/Desing Enviroment, Computer IEEE, APR/1975.

/GILO 78/ Giloi W.K., (1970). Interactive Computer Graphics, Prentice-Hall.

/GKS/ GKS - Graphical Kernel System. Functional Description. Draft International Standard ISO/DIS 7942.

/GOMM 84/ Gomma H., (1984). A Software Design Method for Real-Time Systems. Communications of the ACM. vol. 27, number 9, September 1984.

/LAUT 74/ Lautenbach K., (1972). Use of Petri Nets for Providing Correctness of Concurrent Process Systems. Proc. of the IFIP Conference on Information Processing, North Holland.

/MAGA 86/ Magalhães L.P., (1986). Computação Gráfica, editora da UNICAMP.

/MART 73/ Martin J., (1973). Design of Man-Computer Dialogues, Prentice-Hall.

/MINT 83/ Mintz R., Paula Filho W., (1983). Como Implementar Programas Complexos de Tempo Real em Fortran Básico, anais do III Congresso da SBC, julho de 1983, Campinas SP.

/NEUM 80/ Neumann T., (1980). CAD Data Base Requirementes and Architectures, Lecture Notes in Computer Science, vol.89, cap.6, Springer Verlag.

/NEUM 82/ Neumann T., Hornung C. (1982). Consistency and Transactions in CAD Databases. Proc. 8th International Conference on Very Large Data Bases, Mexico.

/OHSU 79/ Ohsuga S., (1979). Towards Intelligent Interactive Systems, IFIP Workshop on Methodology of Interaction, Seillac, France.

/PDP 1/ Introdution to RSX11-M, manual do PDP11/45.

/PDP 2/ RSX11-M Executive Reference Manual, manual do PDP11/45.

/PDP 3/ RSX11-M Task Builder Reference Manual, manual do PDP11/45.

/PETE 81/ Peterson J.L., (1981). Petri Net Theory and the Modeling of Systems, Prentice Hall.

/PORT 68/ Porter W.A., (1968). Modern Foundations of Systems Engineering, MacMillan.

/RUGG 78/ Ruggiero W., (1978). Distributed Data and Control Driven Machine : Programming and Architecture, dissertação de PHD, Universidade da California, Los Angeles, NOV/1978.

/SCHL 80/ Schlechtendahl E.G., (1980). Lecture Notes in Computer Science ,vol.89, cap.11, Springer Verlag.

/SHAW 74/ Shaw A.C., (1974). The Logical Design of Operating Systems, Prentice Hall.

/SILV 85/ Silva M.V., (1985). Processador de Diálogo: Uma Ferramenta para Geração de Sistemas Interativos, tese de mestrado, FEC/UNICAMP, maio 1985.

/SPAN 84/ Spang III H.A., (1984). The Federated Computer-Aided Control Design System. Proceedings IEEE, vol.72, n.12, DEC/1984.

/THOM 82/ Thomas R., Yates J., (1982). A User Guide to the UNIX System, OSBORNE/McGraw-Hill.

/TOZZ 79/ Tozzi C.L., (1979). Um Terminal Gráfico Interativo Baseado na Tecnologia TV Raster, Tese de doutorado, FEC/UNICAMP.

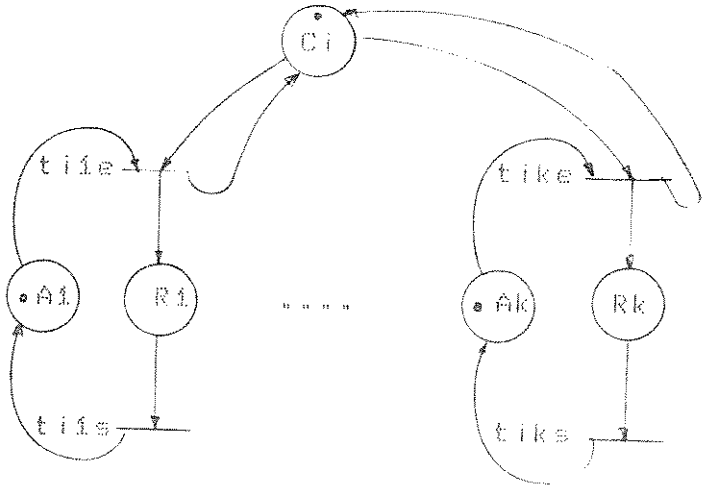
/VERN 84/ Vernadat F.B., (1984). A Commented and Index Bibliography on Data Structuring and Data Management in CAD/CAM : 1970 to mid 1983, National Research Counsil Canada, Division of Electrical Engineering.

APÊNDICE A

MACRO EXECUÇÃO COM CONCORRÊNCIA

```
área de macro = #tarefas, tarefas, data record
j = 1
i = 1
ACTIVATE (tarefa(i))
i = i + 1
DO UNTIL i <= #tarefas
        CALL IS (tarefa(i-1), tarefa(i), RET)
        IF RET = yes
                THEN
                        ACTIVATE (tarefa(i))
                        i = i + 1
                        j = j + 1
                ELSE
                        DO k=1,j
                                WAIT FOR FLAG (fn)
                        END DO
        END IF
END DO
```

REDE DE PETRI PARA A EXECUÇÃO DA CÉLULA



P = { Ci, R1, A1, ... , Rk, Ak }

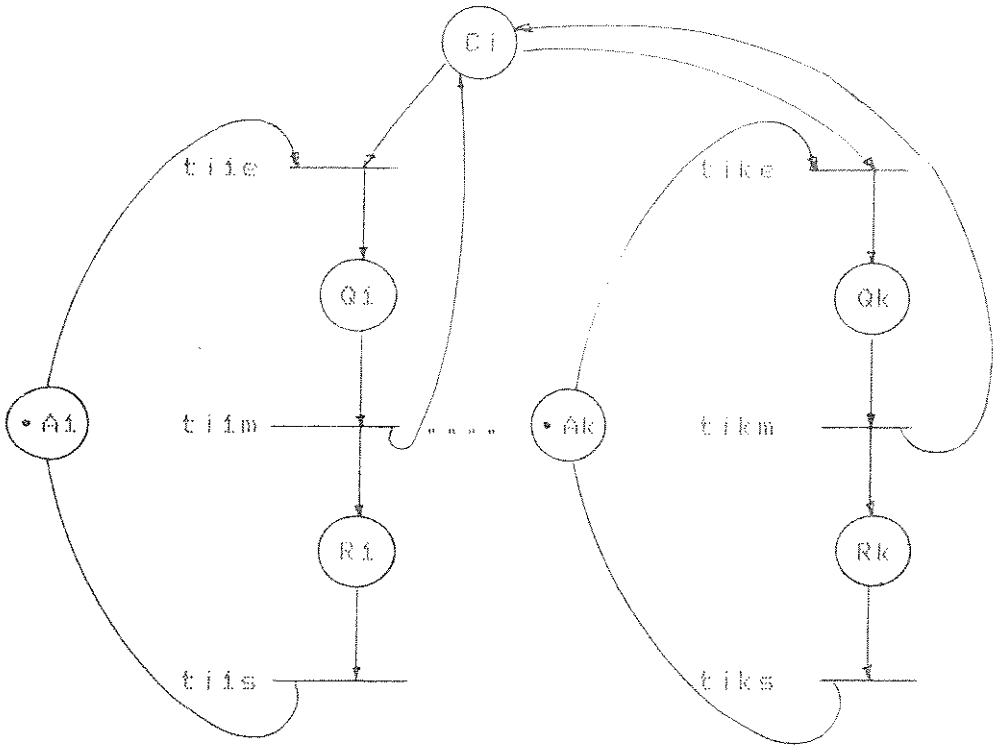
T = { tije, tijs, ... , tike, tiks }

para 1<= j <=k

I(tije) = { Ci, Aj } O(tije) = { Ci, Rj }

I(tijs) = { Rj } O(tijs) = { Aj }

Esta rede não é "pura" pois os conjuntos I(tije) e O(tije) não são disjuntos. Para que se possa analizá-la é necessário torná-la pura, acrescentando os lugares Qj conforme mostrado abaixo.



P = (Ci, Q1, R1, A1, ... , Qk, Rk, Ak)

T = (tiie, tiim, tiis, ... , tike, tikm, tiks)

para 1<= j <=k

I(tije) = (Ci, Aj) O(tije) = (Qj)

I(tijm) = (Qj) O(tijm) = (Ci, Rj)

I(tijs) = (Rj) O(tijs) = (Aj)

-ANÁLISE DA SUBREDE

Matriz de incidência

	tie	tim	tis	...	tke	tkm	tks
Ci	-1	1	0	...	-1	1	0
Q1	1	-1	0	...	0	0	0
R1	0	1	-1	...	0	0	0
A1	-1	0	1	...	0	0	0
...	...	...	...	...	...	...	...
Qk	0	0	0	...	1	-1	0
Rk	0	0	0	...	0	1	-1
Ak	0	0	0	...	-1	0	1

Vetor de estado

$$M = \begin{vmatrix} mCi \\ mQ1 \\ mR1 \\ mA1 \\ \cdot \\ mQk \\ mRk \\ mAk \end{vmatrix}$$

Estado inicial

M0 = (Ci, A1, ... , Ak)

Todos os estados possíveis da rede pertencem a um único conjunto CMJ.

CÁLCULO DAS INVARIANTES

Da equação C'M = 0 temos :

$$-mCi + mQj - mAj = 0$$

$$mCi - mQj + mRj = 0$$

$$-mRj + mAj = 0 \qquad \text{para } 1 <= j <= k$$

donde obtemos as invariantes

$$I0 = (Ci, Q1, \ldots, Qk)$$

$$Ij = (Rj, Aj) \quad \text{para } 1 <= j <= k$$

VERIFICAÇÃO DE DEADLOCK

$$M0'I0 = 1$$

$$M0'Ij = 1 \quad \text{para } 1 <= j <= k$$

donde podemos afirmar que não existe deadlock.

APÊNDICE B

A implementação das funções do Núcleo PAC, em analogia com o conceito de u-programação, foi devida à facilidade de se implementar uma estrutura de dados e algorítmos que determinem se em uma dada seqüência, duas ou mais funções podem ser executadas em paralelo, permitindo deste modo a automatização de todas as fases na geração das funções do Núcleo PAC.

Para a implementação de funções mais complexas seria necessário que o implementador fornecesse as informações sobre o paralelismo e sequenciamento dos procedimentos.

A seguir serão discutidos dois exemplos de automatização na geração das funções do Núcleo PAC que não se encaixam no modelo do capítulo 3.

EXEMPLO 1 - Geração da função por meio de uma expressão algébrica

A definição de uma função do Núcleo PAC pode ser modelada através de um grafo de precedência /SHAW 74/, conforme a figura a seguir.

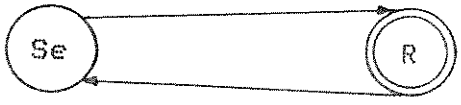
A Interface com Usuário será representada por dois nós:

—o nó Ic que controla a aquisição de comandos e

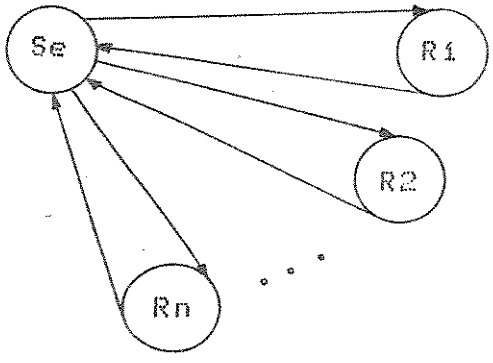
—o nó Id que controla a aquisição de dados e apresentação de resultados.

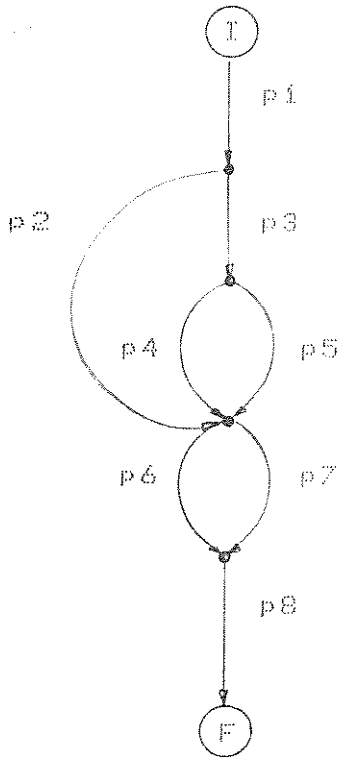
Os nós Ut, BD, PG, e BA estão associados aos recursos dos subsistemas Utilidades, Banco de Dados, Gráfico e Biblioteca de Aplicações, respectivamente. Deve ser lembrado que um recurso, de qualquer subsistema, não pode ser partilhado, porém recursos diferentes de um mesmo subsistema podem ser ativados simultaneamente.

Por simplicidade os nós Id, Ut, BD, PG e BA serão representados por subgrafos que expandidos representarão o acesso à um unico recurso Ri de um subsistema R, como a seguir:



é equivalente à





pi são procedimentos

Este tipo de grafo série/paralelo pode ser modelado em uma expressão algébrica. Para definir a execução sequencial (série) ou paralela de processos serão utilizados os dois operadores, série (>>) e paralelo (//) definidos no capítulo 3.

Por inspeção, pode ser obtida a expressão algébrica a partir do grafo:

G = p1 >> (p2 // (p3 >> (p4 // p5))) >> (p6 // p7) >> p8

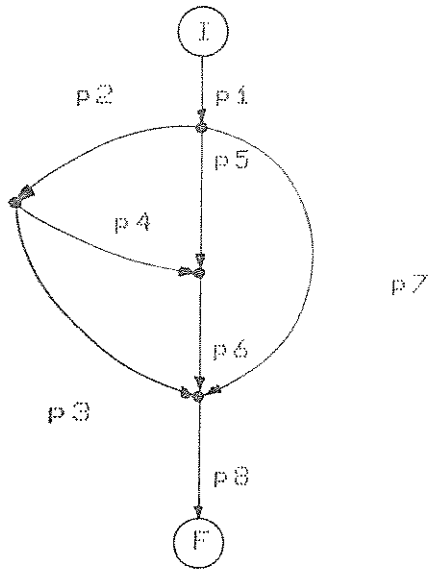
Sendo a expressão algébrica fornecida pelo implementador, ela pode ser utilizada por um analizador sintático para gerar a função, que terá a forma :

```
ACTIVATE p1
WAIT FOR p1
ACTIVATE p2
ACTIVATE p3
WAIT FOR p3
ACTIVATE p4
ACTIVATE p5
WAIT FOR p5
WAIT FOR p4
WAIT FOR p2
ACTIVATE p6
ACTIVATE p7
WAIT FOR p7
WAIT FOR p6
ACTIVATE p8
WAIT FOR p8
```

EXEMPLO 2 - Grafos de precedência geral

Podem ocorrer situações em que o grafo de execução da função não seja série/paralelo. Nestes casos não pode ser obtida a expressão algébrica.

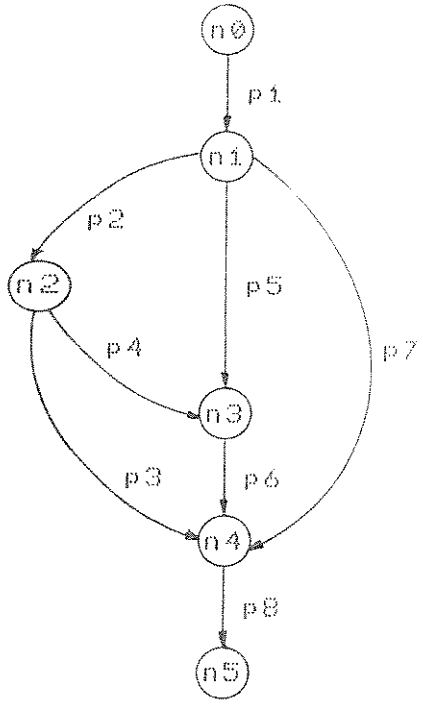
A figura a seguir exemplifica esta situação.



A função gerada por este grafo será :

```
ACTIVATE p1
WAIT FOR p1
ACTIVATE p2
ACTIVATE p5
ACTIVATE p7
WAIT FOR p2
ACTIVATE p3
ACTIVATE p4
WAIT FOR p4
WAIT FOR p5
ACTIVATE p6
WAIT FOR p6
WAIT FOR p3
WAIT FOR p7
ACTIVATE p8
WAIT FOR p8
```

Para descrever o fluxo de ativação dos processos é necessário nomear os nós :



Para automatizar a geração desta função o implementador deverá fornecer a Matriz de Entrada dos Nós (ME), a Matriz de Saída dos Nós (MS) e o Vetor de Ativação dos Nós (VAN) que serão definidos a seguir.

Matriz de Entrada dos Nós - ME(n,p)

Sendo n o nó e p o processo

$$ME(n,p) = \begin{cases} 1 \text{ se p é entrada do nó} \\ 0 \text{ caso contrário} \end{cases}$$

Matriz de Saída dos Nós - MS(n,p)

Sendo n o nó e p o processo

$$MS(n,p) = \begin{cases} 1 \text{ se p é saída do nó} \\ \\ 0 \text{ caso contrário} \end{cases}$$

Vetor de Ativação dos Nós

Os elementos do vetor de ativação dos nós são expressões lógicas dos processos que entram no nó.

$$VAN(n) = f(p)$$

No grafo dado como exemplo temos :

$$VAN(n) = \begin{bmatrix} INICIO \\ p1 \\ p2 \\ p4 \ . \ p5 \\ p3 \ . \ p6 \ . \ p7 \\ p8 \end{bmatrix}$$

O algorítmo que gera a função a partir das matrizes e vetor é descrito a seguir.

ALGORÍTMO PARA A GERAÇÃO DA FUNÇÃO

1 - O vetor de ativação dos nós é inicializado com INICIO (VAN0).

2 - Os processos ativados (ACTIVATE p) são obtidos pelo produto

[MS] [VAN0]

3 - A ativação destes processos gera um novo vetor de ativação dos nós (VNAi). Porém para que novos nós sejam ativados é necessário que alguns processos estejam terminados. Os processos que devem ser terminados (WAIT FOR p) são obtidos pelo produto

[ME] [VNAi]

4 - O novo vetor de ativação dos nós passa a ser VNAi

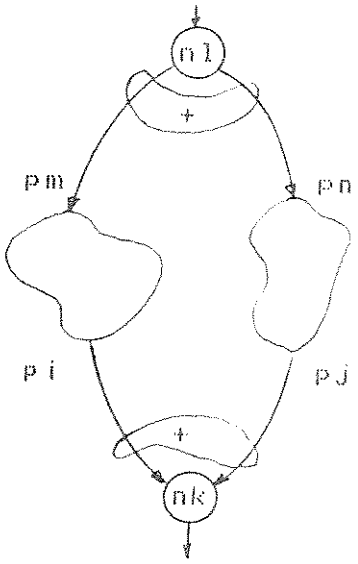
VNA0 = VNAi

5 - Os passos 2,3 e 4 são repetidos até que o vetor de ativação dos nós seja nulo.

O modelamento do grafo de precedência geral tem a característica de permitir modelar estruturas do tipo IF, CASE e DO. A seguir será apresentada cada uma destas estruturas.

ESTRUTURA IF

O grafo abaixo representa uma estrutura IF



Com esta estrutura o Vetor de Ativação dos Nós teria a expressão abaixo para o nó nk

$$VAN(nk) = pi + pj$$

A saída condicional do nó nl será modelada, na Matriz de Saída dos Nós, por uma variável que indica a condição.

...	... nl ...
.	.
.	.
pm	... c ...
pn	... $\bar{c}$ ...
.	.
.	.

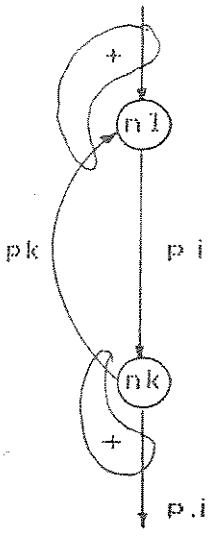
Para o processamento do algorítmo será atribuido um valor à condição.

ESTRUTURA CASE

A estrutura CASE será modelada de modo equivalente à estrutura IF, a única diferença será a variável condicional da Matriz de Saída dos Nós. Para o CASE não aparecerá a condição e sua negação e sim diferentes variáveis que representam as diversas condições c1, c2, ..., ck.

ESTRUTURA DO

O grafo abaixo exemplifica uma estrutura DO



onde pi representa o processo que é repetido e pk o processo de controle do DO.

Na saída do nó nk será modelada a condição que faz o controle do DO.