UNIVERSIDADE ESTADUAL DE CAMPINAS
FACULDADE DE ENGENHARIA DE CAMPINAS
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

Este exemplar corresponde a redação final da tese defendida por José Mario De Martino e aprovada pela Comissão julgadora em 04/julho/1986.

L. P. Magalhães

UM AMBIENTE GMB* PARA O DESENVOLVIMENTO DE SISTEMAS DISTRIBUÍDOS DE CONTROLE DIGITAL

A MÁQUINA GMB*

JOSÉ MARIO DE MARTINO
ORIENTADOR : LÉO PINI MAGALHÃES

TESE APRESENTADA À FACULDADE DE ENGENHARIA DE CAMPINAS / UNICAMP COMO PARTE DOS REQUISITOS EXIGIDOS PARA A OBTENÇÃO DO TÍTULO DE MESTRE EM CIÊNCIAS.

JULHO/1986

ESTE TRABALHO CONTOU COM O APOIO DA FUNDAÇÃO DE AMPARO À PESQUISA DO ESTADO DE SÃO PAULO - FAPESP.

AGRADECIMENTOS:

Ao José Raimundo e Lotufo pelo apoio e paciência no, às vezes desgastante, cotidiano do laboratório.

Ao Armando e Márcio pela dedicação e contribuições na concretização e validação de algumas das idéias aqui propostas.

Ao Maurício pelas valiosas sugestões e pela sempre presente disposição para o debate construtivo.

Ao Léo agradeço, mais do que a paciente orientação e o inesgotável entusiasmo por este trabalho, a amizade.

À LI, SUCA e RAFA.
Eles sabem o porquê.

RESUMO

Este trabalho, no contexto da implementação de uma ferramenta experimental de ensino e pesquisa na área de controle de processos por computador, apresenta e discute um formalismo para o modelamento e programação de sistemas distribuídos de controle digital. Tal formalismo baseia-se no GMB (Graph Model of Behavior) desenvolvido na UCLA, ao qual foram incorporados os seguintes mecanismos para o processamento em tempo-real: escalonamento da ativação dos nós de controle em função de prioridades associadas às marcas do grafo do controle; marcação de arco de controle após intervalo de tempo especificado; marcação frequencial de arco de controle e marcação de arco de controle por eventos externos durante a execução do modelo (interrupção).

Em função deste novo formalismo, agora denominado GMB*, propõe-se a implementação de uma máquina com característica distribuída capaz de processá-lo.

Em adição ao modelo GMB* e sua máquina, são estabelecidas as linhas gerais de um conjunto de ferramentas de apoio à programação GMB* que, quando integradas, constituirão um ambiente para o desenvolvimento de sistemas distribuídos de controle digital.

ÍNDICE

ÍNDICE DAS FIGURAS

CAPÍTULO 1

INTRODUÇÃO

INTRODUÇÃO

A Revolução Industrial é um dos marcos referenciais para as profundas mudanças sócio-econômicas experimentadas pela humanidade a partir do século XIX, quando da substituição do sistema feudal pelo capitalista. Durante a Revolução Industrial, acompanhando a gradativa substituição do trabalho do homem pelo das máquinas, assistiu-se a uma crescente concentração dos meios de produção (os objetos de trabalho - matéria prima e produto - e os instrumentos de produção - máquinas e ferramentas) em instalações físicas apropriadas: as chamadas fábricas /Huberman 76/. Esta proximidade física dos meios de produção foi o incentivo necessário para o desenvolvimento de técnicas mais efetivas de controle da produção.

No início, os processos eram controlados e supervisionados manualmente. Gradualmente algumas destas funções, acompanhando o desenvolvimento dos instrumentos de medida e atuação, foram sendo automatizadas. Em seus primeiros momentos, os sistemas de controle se mostravam extremamente rudimentares, com as funções de medida, controle e atuação quase sempre integradas em um mesmo elemento. Pode-se citar como exemplo de dispositivos de controle da época: os reguladores centrífugos e os termostatos para controle de temperatura. O controle liga-desliga era largamente utilizado neste período.

Na década de 20, aproximadamente, surgiu a instrumentação pneumática e com ela um avanço na tecnologia dos sistemas de controle. Os primeiros instrumentos pneumáticos, entretanto, eram pesados e pouco confiáveis, resumindo-se a válvulas de controle de vazão, medidores de vazão e nível /Gomide 84/. As soluções adotadas nesta fase eram empíricas, apresentando grandes embaraços no projeto, instalação, ajuste e manutenção dos sistemas de controle.

As inovações tecnológicas surgidas durante a segunda guerra mundial tiveram impacto significativo nas técnicas de controle de processos. Em especial, com o surgimeno da instrumentação eletrônica/analógica, com a crescente modularização dos equipamentos de controle (sensores, atuadores, transdutores e reguladores) e com os primeiros esforços de padronização da transmissão de sinais, tornou-se possível a combinação dos equipamentos produzidos pelos diversos fabricantes. Esta facilidade simplificou o projeto, instalação e manutenção dos sistemas de controle do pós-guerra. Os sistemas de controle desta época eram, em geral, formados por reguladores PID, cada um controlando uma malha do processo, centralizados em uma única sala de controle. Os sensores e atuadores encontravam-se distribuídos ao longo do processo.

Com o advento dos computadores digitais (década de 60), criou-se uma grande expectativa de utilização destas máquinas no controle de processos. Tal expectativa era alimentada tanto pelas indústrias, que visavam uma maior produtividade, quanto pelos fabricantes de computadores, que vislumbravam um novo e promissor mercado para o seu produto. Algumas experiências pioneiras foram realizadas com sucesso /Astrom 85/, inúmeros estudos de viabilidade foram efetuados e logo um vigoroso desenvolvimento teve início. Para facilitar a discussão dos rápidos avanços tecnológicos que se seguiram após as primeiras aplicações do computador digital no controle de processos,

costuma-se distinguir três fases neste desenvolvimento: o período pioneiro, o período do controle digital direto (DDC - Direct Digital Control) e o período dos micro-computadores.

O período pioneiro refere-se às primeiras aplicações do computador digital no controle de processos. Os computadores disponíveis na época eram pouco integrados, lentos, caros e apresentavam baixa confiabilidade. Para justificar a instalação de uma destas máquinas, fazia-se necessário encarregá-las de um significativo número de tarefas. A baixa confiabilidade, por outro lado, limitava a ação do computador ao nível de supervisão, deixando o controle direto aos tradicionais controladores analógicos. Neste período pioneiro distinguem-se ainda dois modos de utilização do computador no nível de supervisão: o modo de auxílio ao operador, onde a máquina era utilizada como ferramenta auxiliar para o cálculo das referências (set-points) a serem seguidas pelos reguladores analógicos, e o modo automático de ajuste de referências, onde , como o próprio nome sugere, a máquina ajustava e fornecia automaticamente as referências para os reguladores. Complementando estas tarefas, o computador era ainda utilizado para calcular condições ótimas de operação, definir planos e cronogramas de produção, gerar relatórios de produção, relatórios de consumo de energia e de matéria prima, folha de pagamento e tantos outros documentos de interesse administrativo.

O período do controle digital direto é marcado pela substituição da tecnologia analógica pela digital. Nesta fase, além do controle supervisório, o computador passa a ser utilizado para o controle digital direto. A solução adotada consistia na substituição de toda a instrumentação analógica por um único computador. Esta nova solução trazia a vantagem, devido a sua característica centralizada, de facilitar estratégias de controle que considerassem a interação entre as diversas malhas do processo. Criava-se, então, a oportunidade para o ajuste automático dos parâmetros de cada malha em função de condições globais de operação. Tal característica possibilitou a implementação de algoritmos de controle mais elaborados.

A tecnologia do controle digital direto sofreu, entretanto, algumas restrições ao seu pleno desenvolvimento devido à baixa confiabilidade das máquinas da época. A estratégia de utilização de reguladores analógicos para a redundância das malhas vitais, ou mesmo a implementação destas malhas vitais diretamente com instrumentação analógica, levavam a soluções pouco vantajosas. A utilização de dois computadores trabalhando em um esquema de redundância era, na época, por razões econômicas, inviável.

O problema da confiabilidade do controle digital direto conteve por algum tempo o desenvolvimento desta tecnologia. Entretanto, os avanços experimentados na micro-eletrônica, com o aparecimento de máquinas cada vez mais integradas, mais baratas e confiáveis, viabilizou-a novamente. O surgimento dos mini-computadores (final da década de 60), pequenos e relativamente baratos, somado à experiência acumulada desde o período pioneiro, permitiu um rápido incremento nas aplicações destas máquinas em controle de processos. Os computadores não estavam mais restritos a plantas grandes e complexas. Poder-se-ia utilizá-los, também, para a solução de problemas mais modestos, expandindo, e em muito, o espectro das

aplicações viáveis. Tal expansão, em termos quantitativos, ocasionou um crescimento número dos sistemas de controle baseados em computador de 5.000 em 1970 para 50.000 em 1975 /Astrom 85/.

O contínuo avanço da micro-eletrônica na busca de computadores mais integrados e mais baratos não parou nos mini-computadores. Em 1972 surgiram os micro-computadores. O período desta tecnologia chegou, e com ele o presente. Com o advento desta nova tecnologia, os computadores se tornaram tão integrados e baratos que a sua utilização no controle de processos foi viabilizada para praticamente qualquer aplicação. A característica custo/desempenho desta máquina permite que a mesma seja utilizada inclusive como elemento controlador de uma única malha do sistema. Tal máquina foi o incentivo que faltava para o florescimento dos chamados sistemas distribuídos de controle digital - SDCD.

O chamados sistemas distribuídos de controle digital caracterizam-se por apresentarem a funcão global de controle particionada em sub-funções, com cada uma delas implementada em uma máquina distinta. Tipicamente uma sub-função estará atento ao controle de um ítem do processo. Para alcançar um baixo tempo de resposta e, por vezes, para reduzir os custos associados à transmissão dos sinais, as máquinas que suportam tais sub-funções são fisicamente alocadas nas proximidades do processo. A distribuição física dos computadores que compõem um sistema de controle distribuído, não raras vezes, acompanha o traçado geográfico do processo a ser controlado. É no contexto desta classe de sistemas que o presente trabalho pretende oferecer sua contribuição.

É interessante observar, ainda, que apesar da diversidade tecnológica apresentada, os sistemas de controle possuem a mesma estrutura funcional (figura 1.1). Sensores são utilizados para a medida das variáveis importantes do processo. Em se fazendo necessário, transdutores são utilizados para garantir a compatibilidade dos sinais enviados dos sensores ao processador de informação; ou ainda deste para os atuadores. O processador de informação é responsável, tendo por base os valores medidos do processo, pela geração dos sinais de controle a serem entregues, após a devida conversão, aos atuadores. Opcionalmente, os sistemas de controle podem ainda contemplar interações com elementos humanos a nível de operação e a nível gerencial.

A nível de operação é interessante a existência de um operador que, em situações críticas não previstas, possa optar pelo controle manual do processo. Quando em operação automática é função do sistema manter o operador atualizado sobre o andamento do processo, sinalizando as eventuais situações de alarme.

A nível gerencial faz-se útil que o sistema catalogue, compile e imprima relatórios estatísticos sobre a produção e o consumo.

Neste trabalho entende-se como sistemas distribuídos de controle aqueles sistemas, voltados ao controle de processos, que possuem a função de processamento de informação distribuída entre elementos físicos distintos. Como exemplo de sistemas distribuídos de controle pode-se citar: 1) os primeiros sistemas de controle baseados em controladores do tipo liga-desliga, onde os

dispositivos, apesar de concentrarem individualmente as funções de atuação, controle e atuação, encontravam-se distribuídos pelo processo; 2) os sistemas de controle baseados em reguladores PID analógicos do pós-guerra, onde cada regulador cuidava de uma malha; 3) os modernos sistemas distribuídos de controle digital, onde a função de controle é executada por uma rede local de micros e/ou mini-computadores. Como exemplo de sistema centralizado (não distribuído) têm-se a solução adotada no período do controle digital direto, quando um único computador cuidava de todas as malhas do processo. Convém observar que a forma de distribuição física dos elementos processadores pode ser um outro parâmetro para a classificação dos sistemas de controle. Neste trabalho, este parâmetro será utilizado apenas para uma segunda classificação de sistemas distribuídos de controle. Neste sentido, os primeiros sistemas de controle que se constituíam em um conjunto de controladores liga-desliga espalhados pela planta é um exemplo de um sistema funcional e geograficamente distribuído. Um sistema funcionalmente distribuído, mas geograficamente centralizado, é dado pelas salas de controle que reuniram em um mesmo ambiente, com o intuito de facilitar a monitoração, todos os reguladores PID que individualmente controlavam cada uma das malhas do processo.

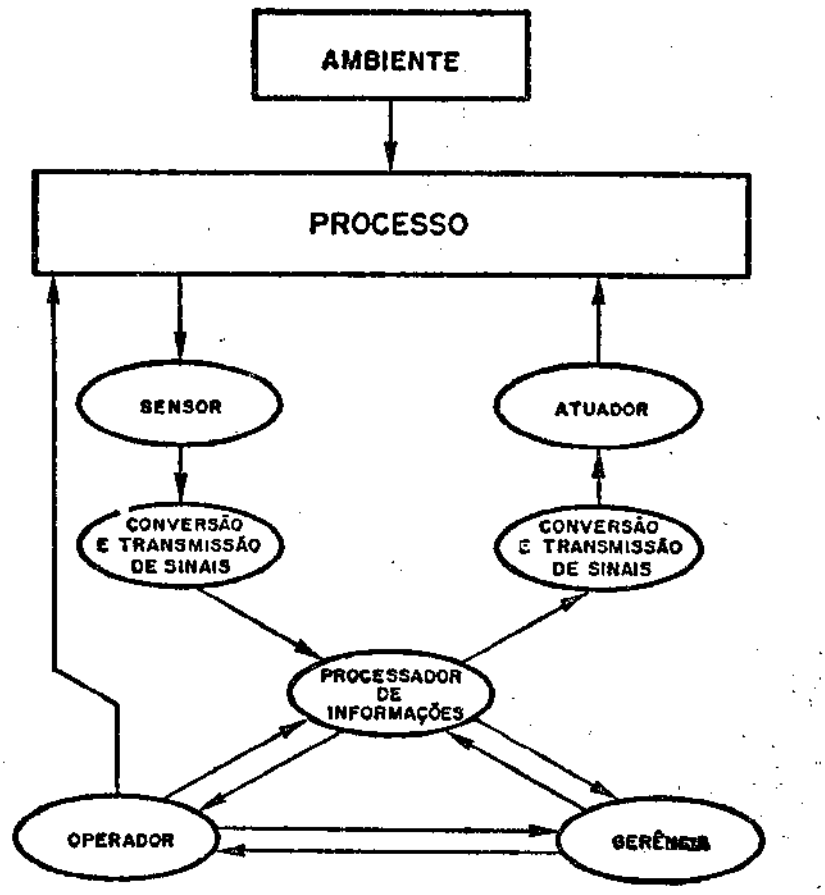


FIG.1.1 - ESTRUTURA FUNCIONAL DE UM SISTEMA DE CONTROLE

Destaque-se, também, que as diferentes filosofias de implementação (distribuída ou centralizada) apresentam propriedades diferentes. Um sistema funcionalmente distribuído pode, a princípio, ter alta confiabilidade, uma vez que a falha de um elemento

compromete uma única malha do sistema. Em um sistema centralizado a falha do processador central compromete a operação do sistema como um todo. Se neste aspecto a centralização é desvantajosa, por outro lado, é mais fácil tratar as interações entre as diversas malhas de controle do sistema nesta estrutura do que na distribuída. Como uma solução intermediária para o compromisso imposto por estas duas situações extremas, surgem os sistemas distribuídos organizados hierarquicamente. Em tais sistemas, a função do controle encontra-se particionada e distribuída em um conjunto de elementos processadores que se agrupam funcionalmente e guardam algum tipo de relação hierárquica entre si. Um exemplo de tal estrutura é o modelo de três níveis (figura 1.2):

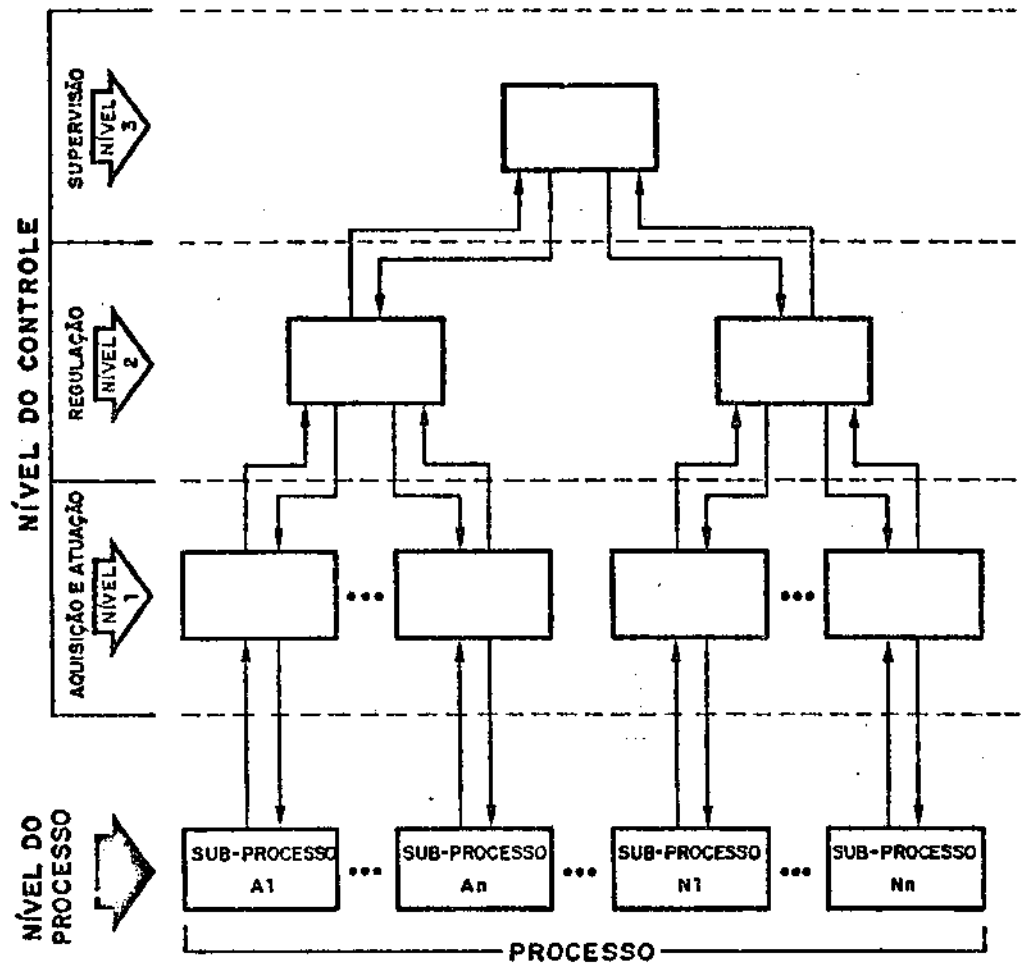


FIG.1.2 - ORGANIZAÇÃO HIERÁRQUICA DE UM SISTEMA DISTRIBUÍDO: MODELO DE TRÊS CAMADAS

Nível 1 - é a interface com o processo. Faz a aquisição de dados e atua sobre o mesmo. Os valores lidos do processo após compactação são enviados ao nível superior para tratamento. Eventualmente encontram-se neste nível mecanismos de segurança para garantir a integridade física da planta. Os elementos do nível 1, via de regra, localizam-se nas proximidades físicas do processo.

Nível 2 - é onde se localizam as funções de controle em tempo-real. Em geral, as ações de controle são tomadas a partir da manipulação dos dados oriundos do processo, seguindo a referência de operação imposta pelo nível superior.

Nível 3 - Cuida da supervisão global da operação da planta e do gerenciamento da produção. Tanto neste nível, como eventualmente no nível 2, podem existir interfaces com operadores.

No modelo de três camadas apresentado, ao se subir na hierarquia, as funções executadas vão se tornando menos críticas no tempo, porém mais complexas. Também à medida que se sobe na hierarquia, o ambiente físico dos elementos processadores torna-se menos agressivo. Pode-se ir de um ambiente de produção, com problemas de variação de temperatura, ruído, pó, etc., ao ambiente de uma sala com isolação térmica e filtragem de ar.

As soluções hierárquicas não seguem necessariamente o modelo de três camadas. A supressão ou criação de novos níveis, assim como a definição da função de cada camada, é estabelecida pelo projetista do sistema em função do problema a ser tratado.

Como características incentivadoras à adoção de sistemas distribuídos de controle digital (SDCD), pode-se listar /Farber 78/ /Netto 79/ /Shimisu 81/ /LeLann 83/ /Lages 86/:

1) Redução da complexidade funcional - Ao se dividir uma tarefa complexa em módulos pequenos e bem definidos, a complexidade do sistema pode ser reduzida, facilitando o planejamento, projeto, instalação e manutenção do sistema.

2) Extensibilidade - Um SDCD permite que a instalação de um sistema possa ser feita de maneira incremental. O crescimento incremental, por permitir a expansão a uma taxa constante de custo/desempenho, reduz os riscos de uma implementação pioneira. A extensibilidade assegura, ainda, que o sistema possa acompanhar mudanças dos requisitos funcionais e de desempenho estabelecidos para o sistema.

3) Desempenho elevado - É importante ressaltar que a adoção de um parâmetro para medir o desempenho de um sistema está intimamente associada à aplicação. Nas aplicações em tempo-real, dado que em tais sistemas a violação de restrições temporais normalmente implica em uma falha (o sistema não satisfaz as suas especificações), é usual a medida do desempenho em termos do tempo de resposta. Devido à capacidade de processamento paralelo apresentado pelos SDCD's, tais sistemas apresentam, pelo menos potencialmente, capacidade de respostas mais rápidas do que os sistemas centralizados. A modularidade dos SDCD's permite a especialização dos elementos processadores, o que por outro lado pode aumentar a eficiência do sistema.

4) Confiabilidade - Em comparação com os sistemas centralizados os SDCD's podem apresentar uma melhoria na disponibilidade e confiabilidade. Os SDCD's suportam operação degradada, onde máquinas vizinhas absorvem as funções das máquinas em situação

de falha, sem contudo parar totalmente o sistema. A estratégia de redundância dos elementos de um SDCD também pode ser considerada vantajosa, se pensada em termos do custo dos micro-computadores.

Entretanto, se por um lado a evolução do hardware associado aos sistemas de controle foi meteórica neste últimos anos, por outro, as técnicas associadas ao desenvolvimento do software não seguiu a mesma trajetória. Tal discrepância tem ponderado no custo total de um sistema, de forma cada vez mais significativa, os gastos associados ao desenvolvimento do software. Em se tratando de sistemas distribuídos de controle digital, a situação se agrava ainda mais, uma vez que o desenvolvimeno de ferramentas para o projeto, implementação e testes de software têm-se concentrado em sistemas com memória compartilhada /Magalhães 86/. Tais ferramentas são inadequadas ao desenvolvimento de programas constituídos por tarefas remotas que se comunicam através da troca de mensagens, suleitas a erros e atrasos devido ao canal físico de comunicação. Atento a esta problemática, o presente trabalho especifica um ambiente de programação para sistemas distribuídos de controle digital baseado no modelo GMB*. Este modelo é uma adaptação, para melhor atender as necessidades do processamento em tempo-real, do modelo GMB desenvolvido na UCLA /Ruggiero 78/.

O modelo GMB* utiliza-se de três domínios distintos, porém inter-relacionados. para descrever sistemas que envolvam concorrência e/ou paralelismo: domínio do controle, domínio dos dados e domínio da interpretação.

O domínio do controle preocupa-se em descrever a estrutura do controle do sistema, identificando os elementos funcionais que o compõem e as relações de precedência para a execução destas funções. Este domínio é definido por um grafo direcionado denominado grafo do controle.

O domínio dos dados descreve como os dados fluem pelo sistema, passando por pontos de transformação e pontos de armazenamento. Os pontos de transformação estão relacionados aos elementos funcionais do grafo de controle. Os armazenadores estabelecem as interfaces entre estes elementos.

O domínio da interpretação constitui a especificação procedural do sistema. Neste domínio são explicitados os procedimentos associados a cada elemento funcional do sistema e as características dos dados que os armazenadores suportam.

Associado ao modelo GMB* estabeleceu-se uma metodologia de projeto de software constituída pelas seguintes etapas:

1) DECOMPOSIÇÃO DO SISTEMA EM MÓDULOS:

1.1) DECOMPOSIÇÃO FUNCIONAL DO SISTEMA (definição do domínio do controle): A decomposição funcional do sistema em módulos é norteada pela independência dos módulos e pelo grau de paralelismo desejado para o sistema. Tais parâmetros podem ser avaliados qualitativamente segundo critérios de coesão e acoplamento /Magalhães 86/. Coesão é a medida da unidade funcional de um módulo, isto é, um

módulo altamente coeso deve desempenhar idealmente apenas uma função. Acoplamento é a medida de independência entre os módulos. O grau de acoplamento pode ser avaliado pelo volume e diversidade de tipos de dados trocados entre os módulos para que eles sejam capazes de executarem as suas funções. Módulos fracamente acoplados trocam poucos dados de um mesmo tipo.

A tarefa de decomposição funcional sofre atualmente do grave inconveniente de depender excessivamente da experiência do projetista. O resultado desta decomposição, entretanto, influi significativamente no desempenho final do sistema. Considere, por exemplo, duas situações extremas de decomposição: 1) nenhuma decomposição e 2) pulverização funcional devido a decomposição excessiva. A primeira situação sofre dos inconvenientes dos sistemas centralizados. A segunda, por outro lado, pode pagar excessiva tributação ao serviço de comunicação. A solução é, portanto, de compromisso e sensivelmente dependente da aplicação.

Pretende-se que a ferramenta proposta neste trabalho, por oferecer um ambiente de experimentação que suporta diferentes graus de decomposição abstraídos para um mesmo problema, contribua para a sistematização do procedimento de decomposição. O formalismo de representação da decomposição funcional a ser utilizado é o grafo de controle do GMB*.

1.2) DEFINIÇÃO DAS INTERFACES ENTRE OS MÓDULOS (definição do domínio dos dados): As interfaces entre os módulos são estabelecidas no formalismo GMB* através do elemento armazenador. A tal elemento pode ser associado um tipo de dado, dando subsídios a testes de consistência na troca de informação entre módulos. Uma vez efetuada a decomposição funcional e a definição das interfaces entre os módulos torna-se possível codificá-los de forma independente. O resultado desta fase é um grafo direcionado, denominado grafo de dados, que descreve como os dados fluem pelo sistema estabelecendo os pontos de armazenamento e os pontos de transformação.

1.3) ESPECIFICAÇÃO DA INTERPRETAÇÃO (definição do domínio da interpretação): Acompanhando a evolução das sub-etapas anteriores faz-se necessário um domínio que descreva e documente as funções e as interfaces dos módulos. Tal descrição pode ser alcançada refinando-se sucessivamente uma especificação funcional inicial , relativamente abstrata, até uma especificação procedural codificada uma linguagem de programação.

2) DEFINIÇÃO DAS PRIORIDADES ASSOCIADAS AOS MÓDULOS: Nesta etapa é efetuada a análise dos requisitos temporais da aplicação. Em função desta análise e da decomposição adotada na fase anterior são identificadas as prioridades dos módulos que compõem o sistema. Tais prioridades são absorvidas pelo modelo GMB* e norteiam a política de escalonamento da execução dos módulos.

3) CONFIGURAÇÃO FÍSICA/LÓGICA DO SISTEMA: Finalmente, a última

módulo altamente coeso deve desempenhar idealmente apenas uma função. Acoplamento é a medida de independência entre os módulos. O grau de acoplamento pode ser avaliado pelo volume e diversidade de tipos de dados trocados entre os módulos para que eles sejam capazes de executarem as suas funções. Módulos fracamente acoplados trocam poucos dados de um mesmo tipo.

A tarefa de decomposição funcional sofre atualmente do grave inconveniente de depender excessivamente da experiência do projetista. O resultado desta decomposição, entretanto, influi significativamente no desempenho final do sistema. Considere, por exemplo, duas situações extremas de decomposição: 1) nenhuma decomposição e 2) pulverização funcional devido a decomposição excessiva. A primeira situação sofre dos inconvenientes dos sistemas centralizados. A segunda, por outro lado, pode pagar excessiva tributação ao serviço de comunicação. A solução é, portanto, de compromisso e sensivelmente dependente da aplicação.

Pretende-se que a ferramenta proposta neste trabalho, por oferecer um ambiente de experimentação que suporta diferentes graus de decomposição abstraídos para um mesmo problema, contribua para a sistematização do procedimento de decomposição. O formalismo de representação da decomposição funcional a ser utilizado é o grafo de controle do GMB*.

1.2) DEFINIÇÃO DAS INTERFACES ENTRE OS MÓDULOS (definição do domínio dos dados): As interfaces entre os módulos são estabelecidas no formalismo GMB* através do elemento armazenador. A tal elemento pode ser associado um tipo de dado, dando subsídios a testes de consistência na troca de informação entre módulos. Uma vez efetuada a decomposição funcional e a definição das interfaces entre os módulos torna-se possível codificá-los de forma independente. O resultado desta fase é um grafo direcionado, denominado grafo de dados, que descreve como os dados fluem pelo sistema estabelecendo os pontos de armazenamento e os pontos de transformação.

1.3) ESPECIFICAÇÃO DA INTERPRETAÇÃO (definição do domínio da interpretação): Acompanhando a evolução das sub-etapas anteriores faz-se necessário um domínio que descreva e documente as funções e as interfaces dos módulos. Tal descrição pode ser alcançada refinando-se sucessivamente uma especificação funcional inicial , relativamente abstrata, até uma especificação procedural codificada uma linguagem de programação.

2) DEFINIÇÃO DAS PRIORIDADES ASSOCIADAS AOS MÓDULOS: Nesta etapa é efetuada a análise dos requisitos temporais da aplicação. Em função desta análise e da decomposição adotada na fase anterior são identificadas as prioridades dos módulos que compõem o sistema. Tais prioridades são absorvidas pelo modelo GMB* e norteiam a política de escalonamento da execução dos módulos.

3) CONFIGURAÇÃO FÍSICA/LÓGICA DO SISTEMA: Finalmente, a última

etapa da metodologia consiste na configuração física e lógica do sistema. A configuração física refere-se à definição da topologia da rede (quantidade de máquinas, posicionamento geográfico e interligação). A configuração lógica define a estratégia de distribuição do aplicativo pela estrutura física. Em linhas gerais, a configuração do sistema é guiada pela distribuição geográfica dos pontos de demanda de processamento. Uma vez estabelecida a configuração física/topológica podem-se fazer necessárias, a critério do grau de confiabilidade/desempenho exigido pela aplicação, transformações localizadas no modelo GMB* (domínios do controle, dados e interpretação) definido pelas fases anteriores. Tais transformações teriam o objetivo de tratar os problemas advindos da existência de canais físicos de interligação entre as máquinas da rede, capazes de destruir, distorcer e atrasar mensagens.

A metodologia proposta identifica e compartimenta alguns problemas associados à confecção do software aplicativo dos sistemas distribuídos de controle digital. Some-se a estas questões o teste e a validação da solução final. Cada um destes problemas estabelece, em linhas gerais, um campo de trabalho a ser explorado com vistas à sistematização do projeto e implementação de sistemas distribuídos de controle digital. Como incentivo a este trabalho de exploração, implementa-se, paralelamente ao presente trabalho, uma rede local de micro-computadores, capaz de executar o modelo GMB*, que terá por função controlar e supervisionar um ferromodelo especialmente construído para representar um sistema metroviário. Tal sistema tem o objetivo de formar uma ferramenta de ensino e pesquisa na área de controle de processos por computador.

Destaque-se que a ferramenta e a metododologia em proposição não constituem uma solução fechada e completa para os problemas associados aos sistemas distribuídos de controle digital. Antes disso, é um ambiente de experimentação que, associado a uma metodologia, pretende contribuir na identificação, e tanto quanto possível avançar nas soluções, dos problemas associados a esta classe de sistemas.

Este trabalho, além deste capítulo introdutório, é constituído por quatro outros capítulos:

O capítulo 2 apresenta e discute o modelo GMB*.

O capítulo 3 especifica o ambiente de programação GMB*, denominado máquina GMB* distribuída, que serve de suporte à metodologia discutida.

Um exemplo ilustrativo que, ao utilizar a metodologia proposta, tenta ressaltar a potencialidade deste ambiente de programação é apresentado no capítulo 4.

E finalmente, no capítulo 5 é apresentado um resumo das principais características e deficiências do ambiente e da metodologia propostos. Neste capítulo são apresentadas as sugestões para a continuação deste trabalho, dando seguimento a busca de métodos, técnicas e ferramentas para a especificação, implementação e teste de sistemas distribuídos de controle digital.

CAPÍTULO 2

O MODELO GMB E O MODELO GMB*

2.1. INTRODUÇÃO

Objetivando fornecer ferramentas de auxílio à síntese e à análise de sistemas que envolvam concorrência e/ou paralelismo, modelos computacionais apropriados foram e estão sendo desenvolvidos /Peterson 74/. A nível de síntese, tais modelos procuram propor metodologias e linguagens para a descrição da concepção estrutural do sistema a partir dos requisitos básicos definidos pela aplicação. A nível de análise, pretendem que a manipulação desta descrição permita realimentar o processo de síntese até a validação final da concepção.

Neste trabalho optou-se, em vez de um rigoroso e prolongado estudo comparativo entre os modelos já existentes, por uma aproximação prática, na tentativa de ganhar experiência e visão crítica sobre a temática do projeto e modelamento de sistemas distribuídos com restrições em tempo-real. Dentre os modelos consultados adotou-se o GMB - Graph Model of Behavior - desenvolvido inicialmente na Universidade da Califórnia /Martin 67/ /Baer 70/ /Ruggiero 78/.

O presente capítulo preocupa-se em expor o GMB, discutindo o significado de cada um de seus elementos primitivos, e apresentar o interpretador, denominado máquina GMB, responsável por sua execução. Discutem-se, ainda, algumas adaptações a serem efetuadas no modelo GMB básico, e em sua máquina, de modo a melhor adequá-los ao processamento em tempo-real, em particular, ao controle de processos. Ao formalismo que surge da adaptação do modelo básico foi dada a denominação de modelo GMB*.

2.2. O MODELO GMB BÁSICO

O modelo GMB apresenta-se como uma ferramenta apropriada para o modelamento de sistemas que envolvam concorrência /Ruggiero 78/. Tal modelo descreve o comportamento de um sistema em três domínios distintos , porém inter-relacionados: controle, dados e interpretação.

O domínio do controle preocupa-se exclusivamente com a descrição do fluxo do controle no sistema. Tal fluxo é descrito por um grafo direcionado (figura 2.2.2), denominado grafo do controle, onde os nós representam passos de computação e os arcos relações de precedência na execução destas computações. Um terceiro elemento, denominado marca, ao fluir pelo grafo estabelece uma representação dinâmica do comportamento do sistema.

O domínio dos dados descreve o fluxo dos dados pelo sistema. Um grafo bipartido direcionado (figura 2.2.4) é utilizado para representar como o conjunto de dados de entrada flui pelo sistema, transitando entre pontos de armazenamento e pontos de transformação, até a produção final do conjunto de dados de saída.

O domínio da interpretação cuida da descrição detalhada dos elementos do domínio dos dados. Para cada ponto de transformação é explicitado o processamento a ser realizado e para cada ponto de

armazenamento é descrita a característica dos elementos que ali podem ser guardados.

As inter-relações entre esses três domínios são definidas por regras de associação.

2.2.1. DOMÍNIO DO CONTROLE

Segundo a ótica do GMB, a estrutura de controle de um sistema (conjunto ordenado de ações tendentes a um resultado) pode ser modelada a partir de duas concepções básicas: eventos e condições.

Os eventos são as ações que podem ser executadas pelo sistema.

As condições são entidades lógicas responsáveis pelo sequenciamento da ocorrência dos eventos. O caráter lógico de uma condição advém do fato de que esta, em um determinado instante, pode estar satisfeita (verdadeira) ou não satisfeita (falsa).

Segundo o GMB, a ocorrência de um determinado evento está subordinada a um conjunto de condições, as chamadas pré-condições do evento, sendo que tais pré-condições podem estar associadas por duas relações básicas:

Relação E - o evento está habilitado a executar a sua ação quando todo um conjunto de pré-condições estiverem simultaneamente satisfeitas;

Relação OU - o evento está habilitado a executar a sua ação quando ao menos um elemento de um conjunto de pré-condições estiver satisfeito.

Considerando o caráter lógico das condições e as relações básicas entre pré-condições, é possível escrever uma expressão booleana para explicitar a habilitação de um evento. A formulação genérica de tal expressão é dada por uma soma (função lógica OU) de produtos (função lógica E) de variáveis booleanas associadas à veracidade das codições. Uma vez verdadeira, tal expressão indicará que o evento está habilitado e pode iniciar a sua ação.

O modelo GMB considera, ainda, que no instante de disparo de um evento, o conjunto de condições que provocaram sua ativação torna-se obsoleto, implicando na necessidade de nova satisfação das pré-condições para uma futura ocorrência do evento. Entretanto, se em um primeiro instante um evento pode atender condições, este, ao finalizar, pode criar condições para a ocorrência de novos eventos. Assim, de forma análoga à expressão lógica que define a habilitação do evento, o modelo GMB também explicita através de uma expressão booleana a relação das condições que podem ser satisfeitas após a finalização do evento.

Apoiado nestas concepções, o domínio do controle GMB propõe, para a representação da estrutura de controle de um sistema, um grafo direcionado, denominado grafo do controle, definido pela quádrupla GC=(N,A,ELE,ELS), onde:

N={n1,n2,...,nz} z⩾0, é um conjunto finito de nós ;

A={a1,a2,...,ax} x⩾0, é um conjunto finito de arcos ;

ELE={ele(n1),ele(n2),...,ele(nz)}, é um conjunto finito de expressões lógicas de entrada ;

ELS={els(n1),els(n2),...,els(nz)}, é um conjunto finito de expressões lógicas de saída .

A figura 2.2.1 apresenta a representação gráfica de cada elemento primitivo do grafo do controle. Na figura 2.2.2 tem-se um exemplo ilustrativo de um grafo do controle.

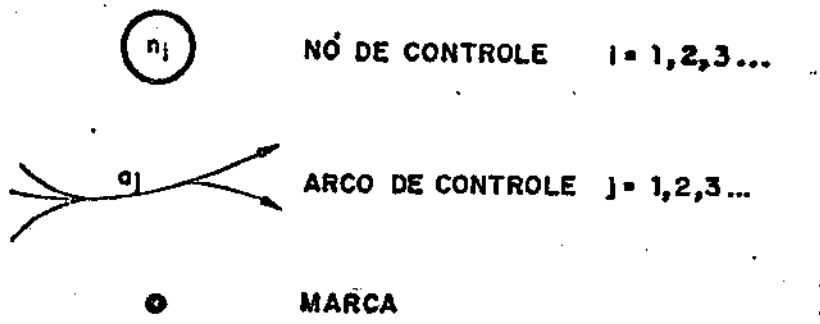


FIG.2.2.1 - REPRESENTAÇÃO DOS ELEMENTOS PRIMITIVOS DO GRAFO DO CONTROLE

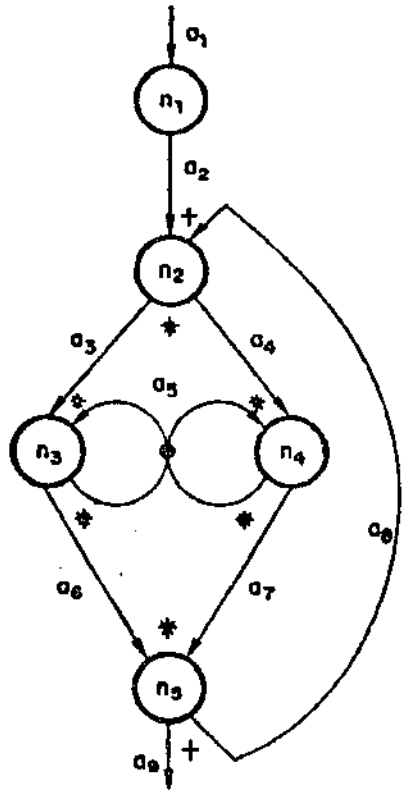
No grafo do controle, os nós representam passos de processamento (eventos); os arcos representam as condições; as expressões lógicas de entrada definem, para cada nó, as suas expressões lógicas de habilitação e as expressões lógicas de saída estabelecem as condições que podem ser satisfeitas após o término da execução dos nós.

Convém salientar que no contexto GMB, a primitiva arco de controle admite mais de um nó origem e/ou nó destino. Ao arco que entra e/ou sai de mais de um nó de controle é dada a denominação de arco de controle complexo (tal tipo de arco tem aplicação no modelamento da exclusão mútua de eventos). Este trabalho, por não estar preocupado especificamente com o desenvolvimento de técnicas de análise para o modelo , deixa para uma fase futura a discussão teórica deste aspecto, uma vez que tal tipo de arco extrapola a definição tradicional de grafo /Read 72/ /Berge 73/ /Chen 76/.

O grafo do controle, como discutido acima, é uma representação estática da estrutura de controle do sistema. Para a representação dinâmica, o domínio do controle utiliza-se de um terceiro elemento primitivo, denominado marca, que ao fluir pelo grafo estabelece uma descrição do comportamento dinâmico do modelo.

O elemento marca, como o próprio nome sugere, é um marcador de estado. Tal elemento, durante a execução do GMB, transita entre os nós e os arcos do grafo do controle. Uma marca ao residir em um arco indica que a condição que o mesmo representa está satisfeita. Já em

um nó, denota que o mesmo está ativo executando o processamento a ele associado. Na formulação analítica das expressões lógicas de entrada e saída de um nó (figura 2.2.2) as variáveis lógicas representadas pelos identificadores dos arcos são verdadeiras quando os referidos arcos estiverem marcados. Os símbolos + e * referem-se às funções lógicas OU e E, respectivamente.



(a) REPRESENTAÇÃO GRÁFICA

$N = \{ n_1, n_2, n_3, n_4, n_5 \}$

$A = \{ a_1, a_2, a_3, a_4, a_5, a_6, a_7, a_8, a_9 \}$

$ELE \{ ele(n_1), ele(n_2), ele(n_3), ele(n_4), ele(n_5) \}$

$ELS \{ els(n_1), els(n_2), els(n_3), els(n_4), els(n_5) \}$

$ele(n_1) = a_1$

$ele(n_2) = a_2 + a_8$

$ele(n_3) = a_3 * a_5$

$ele(n_4) = a_4 * a_5$

$ele(n_5) = a_6 * a_7$

$els(n_1) = a_2$

$els(n_2) = a_3 * a_4$

$els(n_3) = a_5 * a_6$

$els(n_4) = a_5 * a_7$

$els(n_5) = a_8 + a_9$

(b) REPRESENTAÇÃO ANALÍTICA

FIG.2.2.2 - EXEMPLO DE GRAFO DO CONTROLE GMB

Durante a execução do GMB (levantamento do comportamento dinâmico do sistema), a ativação de um nó é disparada pela veracidade de sua expressão lógica de entrada. No instante do disparo do nó, são retiradas as marcas dos arcos que participaram de sua ativação, sendo imediatamente depositada uma marca no seu interior para indicar o seu estado ativo. No GMB não é permitida a ativação de um processador já ativo, ou seja, o modelo não admite re-entrância. Na eventualidade de uma dada distribuição de marcas forçar esta situação, o nó será ativado sequencialmente de maneira não preemptiva. Um nó ao ser desativado (terminar o seu processamento) gera, consoante a sua expressão lógica de saída, um novo conjunto de marcas que será depositado em seus arcos de saída. Por sua vez, esta nova distribuição de marcas pode tornar verdadeira a expressão lógica de entrada de outros nós, ocasionando as suas ativações e, portanto, fazendo com que as marcas fluam pelo grafo. O histórico da movimentação das marcas pelo grafo define o

comportamento dinâmico do controle do sistema.

Por definição, o estado de um grafo do controle GMB, em um determinado instante de tempo, é dado pela distribuição de marcas em seus arcos e pelo conjunto de nós que se encontram ativos naquele dado instante de tempo.

O GMB adota ainda a seguinte nomenclatura para o grafo de controle:

- O conjunto de arcos de controle que chegam(saem) a(de) um determinado nó é denominado de conjunto de arcos de entrada(saída) do referido nó.

- O nó de controle que possue um determinado arco em seu conjunto de entrada(saída) é dito pertencer ao conjunto destino(origem) do arco em questão.

- Ao arco que possua ao menos um elemento do seu conjunto origem(destino) não pertencente ao grafo é dada a denominação de arco de controle de entrada(saída) do grafo. Todos os outros arcos que não se enquadram nesta categoria são denominados de arcos internos do grafo.

- Os arcos de controle que não se enquadram na definição de arcos internos do grafo, independentemente de serem arcos de entrada ou de saída, são denominados genericamente de arcos externos do grafo do controle.

- O nó de controle que possua ao menos um arco de entrada(saída) do grafo em seu conjunto de entrada(saída) é denominado de nó de entrada(saída) do grafo. Todos os outros nós são denominados nós internos do grafo.

- Como discutido acima, a expressão lógica de entrada de um nó é estabelecida, no caso genérico, por uma soma (OU) de produtos (E) de identificadores de arcos de controle. A cada um dos grupos de arcos relacionados pela função E que formam os termos da expressão lógica de entrada é dada a denominação de conjunto habilitador.

2.2.2. DOMÍNIO DOS DADOS

A estrutura do domínio dos dados alicerça-se nas seguintes concepções básicas: evento, dados e armazenador.

Evento, como discutido no domínio do controle, está relacionado ao conceito de ação.

Os dados são os objetos manipulados pelos eventos.

O conceito de armazenador surge da necessidade de se garantir a existência dos dados antes e depois da ocorrência de eventos. Um armazenador representa uma estrutura genérica de dados, capaz de manter informações ali depositadas.

O domínio dos dados preocupa-se em modelar a forma como os dados de entrada fluem pelo sistema, transitando entre pontos de armazenamento e pontos de processamento, até a geração dos dados de saída.

A estrutura deste domínio é dada por um grafo direcionado bipartido, denominado grafo dos dados, definido pela dupla GD=(VD,AD), onde:

VD=P U AR P ∩ AR={∅}, é um conjunto finito de vértices, formado pela união do conjunto dos processadores P com o conjunto dos armazenadores AR={ar1,ar2,...,arv} v≥0;

AD={ad1,ad2,...,adu} u≥0, é um conjunto finito de arcos.

No grafo dos dados os processadores representam os eventos: os armazenadores os lugares onde, e/ou de onde, os processadores podem colocar, e/ou retirar, dados e os arcos, denominados de arcos de dados, representam abstrações de caminhos de interligação dos processadores aos armazenadores. Os processadores tem acesso de leitura ou escrita a estes armazenadores consoante a direção dos arcos de dados.

Os processadores, por representarem eventos, podem ser associados aos nós do grafo do controle. Esta possibilidade de associação divide o conjunto destes elementos em dois sub-conjuntos complementares: os chamados processadores controlados, PC={pc(n1), pc(n2),...,pc(nz)} z≥0; e os não-controlados, PN={pn1,pn2,...,pns} s≥0, onde: P=PC U PN e PC ∩ PN={∅}.

Um processador controlado sempre está associado a um nó do grafo do controle. O processador controlado é um transformador de dados que executa o processamento definido em sua interpretação (domínio da interpretação), transformando dados de entrada em dados de saída sempre que o nó de controle a ele associado estiver ativo. Um processador controlado e seu respectivo nó representam o mesmo evento, cujo detalhamento da ação a ser efetuada é estabelecida no domínio da interpretação (seção 2.2.3).

O processador não-controlado é um transformador de dados que executa o seu processamento independentemente do estado de qualquer um dos nó do grafo do controle. Tal tipo de processador é ativado em decorrência de alterações em seus arcos de entrada. Devido a esta característica, tais arcos, por sua vez, podem ser classificados em: arcos de entrada não-sensitiva e arcos de entrada sensitiva. Os arcos de entrada não-sensitiva não participam da ativação do processador, estabelecendo meramente conexões de dados. Já os arcos de entrada sensitiva participam da ativação do processador consoante o estabelecido na expressão lógica das entradas sensitivas do referido processador - les(pi). Nestas expressões lógicas, de formulação análoga às expressões lógicas de entrada e saída dos nós de controle, uma variável booleana representada pelo identificador de um arco de dados sensitivo é verdadeira sempre que o armazenador origem do arco contiver dados. Eventuais condições nas entradas sensitivas que resultem na tentativa de ativação de um processador não-controlado já ativo são tratadas através da ativação sequencial não preemptiva do processador.

A figura 2.2.3 apresenta a representação gráfica de cada elemento primitivo do grafo dos dados. Na figura 2.2.4 tem-se um exemplo ilustrativo de um grafo dos dados.

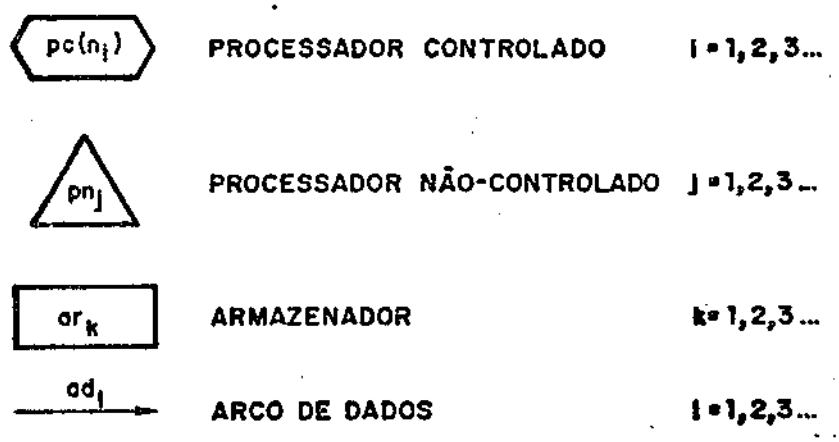


FIG.2.2.3 - REPRESENTAÇÃO DOS ELEMENTOS PRIMITIVOS DO GRAFO DOS DADOS

O estado de um grafo dos dados, em um determinado instante de tempo, é definido pelo conjunto dos processadores ativos e pelo conteúdo de todos os armazenadores naquele dado instante de tempo.

O domínio dos dados admite ainda a seguinte nomenclatura:

- O conjunto dos arcos de dados que entram(saem) de determinado vértice é denominado conjunto dos arcos de entrada(saída) do vértice.

- O vértice que possui um determinado arco em seu conjunto de arcos de entrada(saída) é dito pertencer ao conjunto destino(origem) do arco em questão.

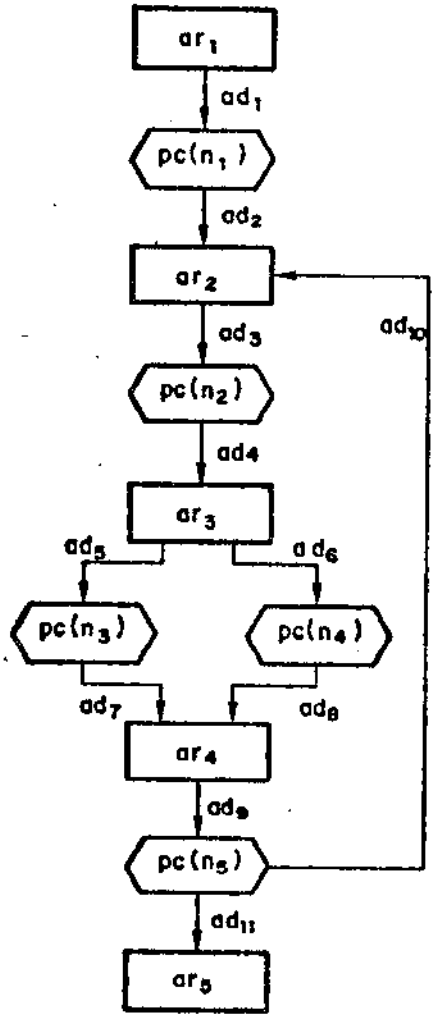
- Ao arco que possua ao menos um elemento do seu conjunto origem(destino) não pertencente ao grafo é dada a denominação de arco de controle de entrada(saída) do grafo. Todos os outros arcos que não se enquadram nesta categoria são denominados de arcos internos do grafo.

- Os arcos de dados que não são arcos internos do grafo, independentemente de serem arcos de entrada ou de saída, são denominados genericamente de arcos externos do grafo dos dados.

- Como discutido acima, a expressão lógica das entradas sensitivas de um processador não-controlado é formada, no caso genérico, pela soma (OU) de produtos (E). Os grupos de arcos de entrada sensitivos relacionados pelas função E são denominados conjuntos sensitivos.

- Ao conjunto sensitivo que em determinado instante acarreta a ativação de um processador não-controlado é dada a denominação de conjunto ativador.

- Para um dado processador, o conjunto dos armazenadores dos quais o processador pode retirar(colocar) dados é denominado domínio(contra-domínio) do processador.



(a) REPRESENTAÇÃO GRÁFICA

VD = PUAR

$P=\{pc(n_1), pc(n_2), pc(n_3), pc(n_4), pc(n_5)\}$

$AR=\{ar_1, ar_2, ar_3, ar_4, ar_5\}$

$AD=\{ad_1, ad_2, ad_3, ad_4, ad_5, ad_6, ad_7, ad_8, ad_9, ad_{10}, ad_{11}\}$

$ad_1 : (ar_1, pc(n_1))$

$ad_2 : (pc(n_1), ar_2)$

$ad_3 : (ar_2, pc(n_2))$

$ad_4 : (pc(n_2), ar_3)$

$ad_5 : (ar_3, pc(n_3))$

$ad_6 : (ar_3, pc(n_4))$

$ad_7 : (pc(n_3), ar_4)$

$ad_8 : (pc(n_4), ar_4)$

$ad_9 : (ar_4, pc(n_5))$

$ad_{10} : (pc(n_5), ar_2)$

$ad_{11} : (pc(n_5), ar_5)$

(b) REPRESENTAÇÃO ANALÍTICA

FIG.2.2.4 - EXEMPLO DE GRAFO DOS DADOS GMB

2.2.3. DOMÍNIO DA INTERPRETAÇÃO

O domínio da interpretação preocupa-se em detalhar cada elemento do domínio dos dados, especificando o processamento a ser realizado pelos processadores e o formato de dados que transitarão entre estes e os armazenadores.

A interpretação de um processador (controlado ou não-controlado) especifica a transformação de dados a ser realizada por este elemento. Em geral, esta interpretação é um procedimento que lê os dados de entrada do domínio do processador, transforma-os e

armazena os resultados no contra-domínio. A interpretação de um processador é executada sempre que o nó a ele associado estiver ativo (regras de associação - seção 2.2.4).

A interpretação de um armazenador define o formato de dados a ele associado. Toda transferência de dados de, e/ou para, o armazenador deve ser compatível com este formato de dados.

Um arco de dados estabelece uma relação estática entre processadores e armazenadores, implicando em compatibilidade de dados entre estes elementos. A interpretação de um arco de dados estabelece o formato da informação que este elemento pode conduzir. Tal interpretação pode ser deduzida implicitamente dos elementos dos conjuntos origem e destino do arco, ou ser explicitamente definida. Neste último caso, a interpretação poderá ser utilizada para teste de consistência do formato dos dados trocados entre processadores e armazenadores.

O modelo GMB não especifica, e portanto não restringe, a linguagem a ser utilizada no domínio da interpretação. Em particular, se a interpretação for explicitada em uma linguagem de programação convencional, passível de ser manipulada por um sistema real, ter-se-á o mapeamento do modelo para um sistema em operação de forma mais simples e direta. Neste trabalho, adotou-se, como formalismo para a especificação da interpretação associada aos processadores GMB, uma mescla da língua portuguesa com a linguagem de programação Pascal. Pretende-se com este formalismo apenas estabelecer a especificação funcional dos processadores, evitando o minucioso detalhamento voltado diretamente à implementação. Da linguagem Pascal utilizam-se a estrutura de atribuição, as estruturas de controle de sequenciamento (BEGIN...END, WHILE...DO, etc.) e as regras de formação de expressões. Já a língua portuguesa é utilizada para auxiliar a explicitação das ações a serem efetuadas dentro das estruturas de controle do Pascal. Convencionou-se, também, que os elementos em português serão expressos na forma de comentários da linguagem Pascal, ou seja, as frases em português sempre serão iniciadas por uma barra seguida de asterisco (/*) e finalizadas por um asterisco seguido de uma barra (*/). Do formalismo do Pascal substitui-se, ainda, a palavra reservada "PROGRAM" pela palavra "INTERPRETAÇÃO". A palavra "INTERPRETAÇÃO" sempre aparecerá seguida do identificador de um processador, fazendo associação deste com a sua interpretação. Para a interpretação dos armazenadores construiu-se o seguinte formalismo:

1. Os diversos tipos de dados que um armazenador pode comportar são identificados individualmente por campos de dado.

2. Um campo de dado é representado por um nome (uma ou mais palavras em português) delimitado pelos símbolos < >, ou pelos símbolos < > delimitando mais de um campo de dado.
Exemplos:
<ENDEREÇO INICIAL>
<<TAMANHO ARMAZENADOR><ENDEREÇO INICIAL>>
<<<TAMANHO ARMAZENADOR><ENDEREÇO INICIAL>><FATOR OCUPAÇÃO>>

3. Utiliza-se a sequência de símbolos ::= (lê-se é formado por) para associar o identificador de um armazenador, ou mesmo de um campo de dado, ao(s) campo(s) de dado que o compõe(m).

Exemplos:
ar1::=<descrição área armazenamento>
(lê-se o armazenador ar1 é formado pelo campo de dado descrição área armazenamento)
<descrição área armazenamento>::=<<localização>
<FATOR OCUPAÇÃO>>
(lê-se o campo de dado descrição área armazenamento é formado pelos campos localização e FATOR OCUPAÇÃO.

4. O nome de um campo de dado pode ser grafado com letras maiúsculas ou letras minúsculas.

O nome do campo de dado grafado em letras maiúsculas especifica que o mesmo é um campo de dado terminal. Um campo de dado terminal representa um tipo primitivo de dado, cujas características principais podem ser extraídas da análise das transformações (interpretação dos processadores) das quais será objeto. O mapeamento de um tipo de dado para um tipo convencional de dado (inteiro, real, array, etc.) deve ser efetuado na fase de implementação, atendendo as conveniências daquele momento.

O nome do campo de dado grafado em letras minúsculas indica que o campo em questão não é um campo terminal, devendo haver, portanto, uma continuação da especificação de sua interpretação até que ele seja totalmente definido em termos de elementos terminais. Esta representação intermediária em letras minúsculas tem o propósito de auxiliar a compreensão do significado dos campos de dados que compõem o armazenador.
Exemplos:
ar1::=<descrição área armazenamento>
<descrição área armazenamento>::=<<localização>
<FATOR OCUPAÇÃO>>
<localização>::=<TAMANHO><ENDEREÇO INICIAL>
ar2::=<<TAMANHO><ENDEREÇO INICIAL><FATOR OCUPAÇÃO>>
Observar que as interpretações dos armazenadores ar1 e ar2 são equivalentes.

5. Utiliza-se o abrir e fechar de parênteses para indicar genericamente que o(s) campo(s) por eles delimitados ocorre(m) um número arbitrário de vezes. A especificação e consequente limitação do número exato de campos que o armazenador comportará é deixada para a fase de implementação.
Exemplos:
ar1::=(<VELOCIDADE>)
(lê-se o armazenador ar1 é formado por um número arbitrário de campos VELOCIDADE)
ar2::=(<(<VELOCIDADE>)>
(lê-se o armazenador é formado por um campo de dado constituído por um número arbitrário de campos VELOCIDADE)

6. O símbolo / é utilizado para representar a conectiva "OU". Ao ser encontrado entre dois campos de dado implica que os dois campos podem ocorrer, entretanto, em um determinado instante apenas um deles.
Exemplo:
ar1::=<estado área armazenamento>
<estado área armazenamento>::=<VAZIA> / <CHEIA>

Por último, é conveniente ressaltar que o acesso a um

armazenador composto por um único campo de dado, embora tal campo possa ser constituído por outros campos de dado, é substancialmente diferente de um armazenador composto por vários campos de dado. No primeiro caso, o acesso ao armazenador manipula como um todo o bloco de informação contido em seu único campo de dado. Já no segundo caso, cada acesso manipula a informação contida em apenas um dos campos. Uma vez que no contexto deste trabalho os armazenadores são considerados como estruturas FIFO (first in - first out), a ordem do acesso define, em um determinado instante, qual o campo efetivamente acessado. Como exemplo suponha as seguintes interpretações para dois armazenadores distintos:

ar1::=<<descrição área armazenamento>
<descrição área armazenamento>>
<descrição área armazenamento>::=<<TAMANHO><ENDEREÇO INICIAL>
<FATOR OCUPAÇÃO>>
ar2::=<descrição área armazenamento>
<descrição área armazenamento>
<descrição área armazenamento>::=<<TAMANHO><ENDEREÇO INICIAL>
<FATOR OCUPAÇÃO>>

Neste exemplo, os armazenadores são capazes de comportar duas descrições de área de armazenamento. Um acesso ao armazenador ar1 manipula de uma só vez a informação associada às duas áreas de armazenamento. Em contraposição, um acesso ao armazenador ar2 manipula a informação referente a uma única área de armazenamento.

Segue abaixo um exemplo ilustrativo do domínio da interpretação GMD. Este exemplo, conjuntamente com o domínio do controle e domínio dos dados estabelecidos, respectivamente, nas figuras 2.2.2 e 2.2.4, formam um modelo GMD completo.

DOMÍNIO DA INTERPRETAÇÃO

INTERPRETAÇÃO DOS ARMAZENADORES

ar1::=<DADOS DE ENTRADA>

ar2::=<RESULTADO TRANSFORMAÇÃO T1 APLICADA AO CONTEÚDO DE AR1> /
<RESULTADO TRANSFORMAÇÃO T5 APLICADO AO CONTEÚDO DE AR4>

ar3::=<RESULTADO TRANSFORMAÇÃO T2 APLICADA AO CONTEÚDO DE AR2>

ar4::=<RESULTADO TRANSFORMAÇÃO T3 APLICADA AO CONTEÚDO DE AR3> /
<RESULTADO TRANSFORMAÇÃO T4 APLICADO AO CONTEÚDO DE AR3>

ar5::=<RESULTADO TRANSFORMAÇÃO T6 APLICADA AO CONTEÚDO DE AR4>

INTERPRETAÇÃO DOS PROCESSADORES

```
INTERPRETAÇÃO pc(n1) ;
BEGIN
 /* leia o conteúdo de ar1 */ ;
 /* aplique a transformação T1 aos dados lidos */ ;
 /* escreva os dados resultantes desta transformação em ar2 */ ;
 /* marque a2 */
END.
```

```
INTERPRETAÇÃO pc(n2) ;
BEGIN
 /* leia o conteúdo de ar2 */ ;
 /* aplique a transformação T2 aos dados lidos */ ;
 /* escreva os dados resultantes desta transformação em ar3 */ ;
 /* marque a3 e a4 */
END.

INTERPRETAÇÃO pc(n3) ;
BEGIN
 /* leia o conteúdo de ar3 */ ;
 /* aplique a transformação T3 aos dados lidos */ ;
 /* escreva os dados resultantes desta transformação em ar4 */ ;
 /* marque a5 e a6 */
END.

INTERPRETAÇÃO pc(n4) ;
BEGIN
 /* leia o conteúdo de ar3 */ ;
 /* aplique a transformação T4 aos dados lidos */ ;
 /* escreva os dados resultantes desta transformação em ar4 */ ;
 /* marque a5 e a7 */
END.

INTERPRETAÇÃO pc(n5) ;
BEGIN
 /* leia o conteúdo de ar4 */ ;
 IF /* os dados lidos satisfazem a condição de finalização */ THEN
 BEGIN
  /* aplique a transformação T6 aos dados lidos */ ;
  /* escreva so dados resultantes desta transformação em ar5 */ ;
  /* marque a9 */
 END
 ELSE
 BEGIN
  /* aplique a transformação T5 aos dados lidos */ ;
  /* escreva os dados resultantes desta transformação em ar2 */ ;
  /* marque a8 */
 END
END.
```

2.2.4. REGRAS DE ASSOCIAÇÃO

Os três domínios do GMB (controle, dados e interpretação) estão relacionados pelas seguintes regras de associação:

1. Para todo processador controlado do grafo dos dados existe um único nó do grafo do controle a ele associado.

2. Para todo nó do grafo do controle existe um único processador controlado a ele associado.

3. Para todo processador (controlado ou não-controlado) existe uma especificação, definida no domínio da interpretação, da transformação de dados que será por ele executada.

4. Para todo armazenador do grafo dos dados existe uma especificação, definida no domínio da interpretação, do formato de dados que ele suporta.

5. Sempre que um nó de controle estiver ativo o seu respectivo processador controlado também estará ativo, com a transformação de dados definida em sua interpretação sendo executada. Nenhum processador controlado pode estar ativo sem que seu respectivo nó de controle também esteja.

2.2.5. EXECUÇÃO DO MODELO GMB: A MÁQUINA GMB

O grafo do controle, o grafo dos dados, a interpretação e as regras de associação estabelecem uma descrição estática do comportamento do sistema. O comportamento dinâmico pode ser visualizado ao se executar o modelo por uma máquina especial denominada máquina GMB. Tal máquina é um interpretador do modelo GMB que, a partir de um estado inicial, ativa e desativa nós e processadores, movendo o modelo por uma sequência de estados. O registro desta sequência, conjuntamente com o estado inicial, determinam a descrição dos aspectos dinâmicos do comportamento do sistema. A execução de um modelo pela máquina GMB é ditada pelas seguintes definições:

1. O estado do grafo do controle em um determinado instante de tempo é definido pela distribuição de marcas nos arcos de controle e pelo conjunto de nós de controle que se encontram ativos naquele dado instante de tempo.

2. O estado do grafo dos dados em um determinado instante de tempo é definido pelo conjunto dos processadores ativos e pelo conteúdo de todos os armazenadores naquele dado instante de tempo.

3. O estado de um GMB em um determinado instante de tempo é definido pelos estados do grafo do controle e grafo dos dados naquele dado instante de tempo.

4. O estado inicial de um GMB é definido como o seu estado no instante inicial de execução.

5. Um nó de controle está habilitado quando a sua expressão lógica de entrada estiver verdadeira.

6. Para a computação da veracidade de uma expressão lógica de entrada, considera-se que as variáveis lógicas representadas pelos arcos estão verdadeiras quando o referido arco possuir ao menos uma marca.

7. Um nó de controle pode ser ativado quando estiver habilitado e seu respectivo processador estiver desativado.

8. Dois nós de controle nunca são ativados exatamente no mesmo instante. Um nó de controle é ativado pela máquina GMB em um determinado instante de tempo se, naquele instante, for o único nó que pode ser ativado: em não sendo o único, a máquina GMB

elege arbitrariamente um dos nós, entre os possíveis, para ser ativado naquele instante.

9. No instante de ativação de um nó, a máquina GMB retira as marcas de seus arcos de entrada consoante a sua expressão lógica de entrada:

a) Caso a expressão lógica de entrada seja dada por um único conjunto habilitador - retira uma marca de cada arco pertencente ao conjunto.

b) Caso a expressão lógica de entrada seja dada por uma soma de conjuntos habilitadores - escolhe arbitrariamente um conjunto habilitador cujo termo na expressão lógica de entrada esteja verdadeiro e deste retira uma marca de cada um dos seus arcos.

10. Um processador controlado é ativado sempre que seu respectivo nó de controle for ativado.

11. Um processador (controlado ou não-controlado) ao ser ativado inicia a execução de sua interpretação.

12. A máquina GMB desativa um processador (controlado ou não-controlado) ao terminar a execução de sua interpretação.

13. Um nó de controle é desativado quando o seu respectivo processador controlado for desativado.

14. No instante da desativação de um nó, consoante a interpretação associada a seu processador e sua expressão lógica de saída, marcas são geradas e depositadas em seus arcos de saída.

15. Um processador não-controlado é ativado quando estiver desativado e sua expressão lógica das entradas sensitivas for verdadeira.

2.2.6. EXEMPLO DE EXECUÇÃO DE UM MODELO GMB

A título de exemplificar a execução de um GMB considere o modelo formado pela figuras 2.2.2 (domínio do controle), 2.2.4 (domínio dos dados) e cujo domínio da interpretação está explicitado na seção 2.2.3. Considere também para este modelo o seguinte estado inicial:

ESTADO INICIAL DO GRAFO DO CONTROLE:

1. Conjunto dos nós ativos: {Ø}
2. Conjunto dos arcos marcados com uma marca: {a1,a5}
   OBS: os demais arcos não estão marcados.

ESTADO INICIAL DO GRAFO DOS DADOS:

1. Conjunto dos processadores ativos: {Ø}
2. Conteúdo dos armazenadores:
   ar1 - contém os dados de entrada do sistema:
   DADOSDEENTRADA

OBS: os demais armazenadores estão vazios.

A discussão que se segue reportar-se-á às definições acima observadas (seção 2.2.5).

No instante inicial de execução apenas o nó n1 está em condições de ser ativado (def. 5, 6 e 7). A máquina GMB ao iniciar a execução ativa o nó n1 (def. 8). A ativação deste nó implica na remoção da marca do arco a1 e no início da execução da interpretação associada ao processador pc(n1) (def. 9, 10, e 11). Neste exemplo, a interpretação de pc(n1) especifica o armazenamento em ar2 do resultado da transformação T1 aplicada ao conteúdo de ar1. Na desativação de n1 uma marca é depositada no arco a2 (def. 12, 13 e 14). Após a desativação de n1, o nó n2, e consequentemente o processador pc(n2), é ativado (def. 5, 6, 7, 8 e 10). Nesta ativação a marca residente em a2 é retirada (def. 9). A interpretação de pc(n2) avalia T2(ar2) e armazena o resultado em ar3 (def. 11). Na desativação de n2 uma marca é colocada em a3 e outra em a4 (def. 12, 13 e 14). Esta nova distribuição de marcas habilita dois outros nós: n3 e n4 (def. 5 e 6). Dado a existência de dois nós que podem ser ativados, a máquina GMB seleciona arbitrariamente um deles, por exemplo n3, para ser primeiramente ativado (def. 7 e 8). Entretando, a ativação de n3 remove as marcas de a3 e a5, impossibilitando a ativação de n4 (def. 9 , 5, 6 e 7). Neste exemplo, o arco a5 assegura a mútua exclusão de n3 e n4, que competem no acesso aos armazenadores ar3 e ar4. Em se ativando n3, tem-se a ativação de pc(n3) e a consequente execução de sua interpretação (def. 10 e 11). Na finalização de n3, a5 e a6 são marcados, habilitando n4 (def. 12, 13, 14, 5 e 6). Ao ser ativado, n4 retira as marcas de a4 e a5 (def. 7, 8 e 9). No término da execução de sua interpretação este nó marca a5 e a7 (def. 10, 11, 12, 13 e 14). Neste instante o único nó habilitado é n5 (def. 5 e 6). Sua ativação remove as marcas de a6 e a7 (def. 7, 8 e 9). A interpretação associada a este nó decide, em função do conteúdo de ar4, qual será a transformação a ser aplicada e qual o arco que será marcado (def. 14). Caso o conteúdo de ar4 resulte na marcação de a8, ter-se-á a re-habilitação de n2 e a consequente re-execução de trecho do programa.

2.3. O MODELO GMB* E A MÁQUINA GMB*

O enfoque principal deste trabalho é a definição de uma arquitetura baseada no modelo GMB que permita a sistematização do projeto e implementação de sistemas distribuídos de controle digital. Com tal preocupação, propõe-se, nesta seção, a modificação de alguns aspectos do modelo GMB com o intuito de melhor adequá-lo ao processamento em tempo-real, com ênfase às aplicações voltadas ao controle de processos. Tais modificações estão restritas a três aspectos do modelo GMB básico: abolir o elemento processador não-controlado do domínio de dados; permitir a marcação dos arcos de entrada do grafo do controle, assim como, a troca de dados através dos arcos de dados externos, durante a execução do modelo e associar prioridades às marcas do grafo do controle.

A extinção do processador não-controlado, embora este elemento se mostre apropriado ao modelamento de sistemas cuja dinâmica é ditada pelo fluxo dos dados (uma ação é disparada pela

disponibilidade de todos os seus operandos), visa fortalecer o formalismo do modelo, em particular no que se refere à análise, limitando a descrição do fluxo do controle ao, como o próprio nome sugere, domínio do controle.

No escopo deste trabalho, um sistema de processamento em tempo-real é aquele que monitora e/ou controla atividades cuja dinâmica não se adapta a nenhuma base de tempo que possa ser definida a priori pelo sistema. Neste sentido, um sistema em tempo-real deve estar sempre apto a receber e a responder a estímulos externos, sendo desejável que as resposta sejam fornecidas em tempo mínimo, ou ao menos com atraso mínimo em relação a um instante ideal de resposta. Assim, admitir a marcação dos arcos de entrada do grafo do controle e a troca de dados durante a execução do modelo, permite ao GMB contemplar a necessidade que um sistema em tempo-real possui de interagir com o ambiente exterior.

É interessante observar que permitir a marcação dos arcos de entrada do grafo do controle durante a execução do modelo pode ser reportado ao conhecido mecanismo de interrupção, de reconhecida importância na utilização da tradicional arquitetura de Von Neumann no procesamento em tempo-real. Na máquina de Von Neumann a ocorrência de uma interrupção acarreta a suspensão de um processamento (processo) e o início da execução de outro, permitindo que estímulos externos e assíncronos à máquina possam coordenar o sequenciamento da execução de seu processamento. Entretanto, a obrigatoriedade da suspensão de um processo para a execução de outro, só se faz necessária em uma máquina sequencial como a máquina de Von Neumann. Na máquina GMB, que pode ser considerada como um máquina de Von Neumann com a característica sequencial relaxada, a ocorrência de um estímulo externo à máquina está tão somente relacionada ao início da execução de um novo processamento. Desta forma, permitir que o ambiente exterior marque arcos de entrada do grafo do controle não tem outro significado na máquina GMB do que permitir que estímulos externos influam no sequenciamento da execução de seu processamento.

Consoante o estabelecido na seção anterior (seção 2.2), a correta execução de um GMB só é alcançada com a utilização de uma máquina GMB que possua tantos processadores físicos quanto for o número máximo de processadores GMB que possam estar ativos durante a execução do modelo. Esta observação implica que qualquer estrutura realizável (possua um número finito de processadores físicos) de máquina GMB impõe uma severa limitação à generalidade dos modelos que pode executar. Uma forma de abrandar tal restrição é a modificação da definição 7 da seção 2.2.5 para:

> 7. Um nó de controle pode ser ativado quando estiver habilitado, seu processador desativado e existir processador físico disponível para executar a interpretação associada a seu processador.

O custo desta modificação, entretanto, é significativo, uma vez que acentua a característica não-determinística do modelo. Agora, um nó habilitado pode ter a sua ativação postergada por tempo indeterminado devido ao procedimento de seleção para a ativação de nós estabelecido na definição 9 - seção 2.2.5. Por sua vez, este não-determinismo, por inviabilizar as respostas em tempo mínimo,

torna-se insustentável no processamento em tempo-real. Para minimizar os efeitos do não-determinismo do modelo, quando da sua execução por uma máquina com número insuficientes de processadores, propõe-se a associação de prioridades às marcas do grafo do controle. A partir de agora, associado a uma marca sempre existirá um número natural para indicar a sua prioridade. Convencionou-se também que quanto maior este número, maior é a prioridade associada à marca. Tais prioridades serão utilizadas no critério de seleção do nó que em um determinado instante de tempo será ativado.

É importante salientar que a definição das prioridades associadas às marcas é de exclusiva competência do programador (elemento que constrói o modelo), sendo definidas no domínio da interpretação e geradas nos instantes das marcações dos arcos de controle.

Ao modelo GMB que contempla as três modificações discutidas acima foi dada a denominação de modelo GMB*. A execução deste novo modelo, aos moldes do modelo GMB básico, é efetuada pela máquina GMB*, cuja dinâmica é apresentada a seguir:

1. O estado do grafo do controle em um determinado instante de tempo é definido pela distribuição de marcas nos arcos de controle e pelo conjunto de nós de controle que se encontram ativos naquele dado instante de tempo.

2. O estado do grafo dos dados em um determinado instante de tempo é definido pelo conjunto dos processadores ativos e pelo conteúdo de todos os armazenadores naquele dado instante de tempo.

3. O estado de um GMB* em um determinado instante de tempo é definido pelos estados do grafo do controle e grafo dos dados naquele dado instante de tempo.

4. O estado inicial de um GMB* é definido como o seu estado no instante inicial de execução.

5. Um nó de controle está habilitado quando a sua expressão lógica de entrada for verdadeira.

6. Para a computação da veracidade de uma expressão lógica de entrada, considera-se que as variáveis lógicas representadas pelos arcos são verdadeiras quando o referido arco possuir ao menos uma marca.

7. Um nó de controle pode ser ativado quando estiver habilitado, seu respectivo processador estiver desativado e existir processador físico disponível para executar a interpretação associada a seu processador.

8. Dois nós de controle nunca são ativados exatamente no mesmo instante. Em um dado instante, o nó de controle que efetivamente será ativado pela máquina GMB* é o nó de maior prioridade. Caso ocorra empate de prioridade, o nó a ser ativado será, dentre os mais prioritários, o que a mais tempo se encontra habilitado (critério de antiguidade cronológica). Persistindo o empate, a máquina GMB* escolhe arbitrariamente um

dos nós.

9. A prioridade de um nó habilitado é o maior valor tomado entre os valores das prioridades de seus conjuntos habilitadores que estejam verdadeiros.

10. O cálculo da prioridade de um conjunto habilitador que esteja verdadeiro é efetuado tomando-se, para cada arco do conjunto habilitador, a marca de maior valor. A prioridade do conjunto habilitador é dada pelo somatório das prioridades individuais das marcas. Deve-se notar que caso um arco participe mais de uma vez na formação da expressão do conjunto habilitador serão necessárias, no mínimo, tantas marcas quanto for a multiplicidade do identificador do arco na expressão do conjunto habilitador. No cálculo da prioridade estas marcas são consideradas em ordem decrescente de prioridades.

11. No instante de ativação de um nó, a máquina GMB* retira as marcas que participaram da computação da prioridade do nó. Caso exista mais de um conjunto habilitador com prioridade igual ao do nó, a máquina GMB* escolhe arbitrariamente um destes conjuntos e retira as marcas que participaram da computação de sua prioridade.

12. Um processador é ativado sempre que seu respectivo nó de controle for ativado.

13. Um processador ao ser ativado tem a sua interpretação executada pela máquina GMB*.

14. A máquina GMB* desativa um processador ao terminar a execução de sua interpretação.

15. Um nó de controle é desativado quando o seu respectivo processador controlado for desativado.

16. No instante da desativação de um nó, consoante a interpretação associada a seu processador e sua expressão lógica de saída, marcas são geradas e depositadas em seus arcos de saída.

2.4. CONSIDERAÇÕES FINAIS

São inúmeras as maneiras de se utilizar um modelo como ferramenta de auxílio ao projeto de sistemas /Peterson 81/. Entretanto, a eficácia de um modelo em uma dada aplicação está condicionada ao grau de desenvolvimento da seguintes técnicas de suporte: técnicas de especificação, técnicas de modelamento, técnicas de análise e técnicas de implementação.

As duas primeiras técnicas estão intimamente relacionadas à capacidade do modelo de descrever a classe de problemas a que se propõe modelar. Em particular, as técnicas de especificação preocupam-se em sistematizar a conversão da concepção inicial de um sistema, esboçada no plano das idéias, em uma descrição formal e estruturada definida pelo modelo. Já as técnicas de modelamento

abordam a problemática de como utilizar o modelo para representar sistemas já existentes, ou mesmo especificados por outra técnica de projeto.

As técnicas de análise visam definir metodologias para a determinação das propriedades e características do sistema em projeto, ou em estudo, via a manipulação do modelo.

As técnicas de implementação têm como escopo a definição de uma estratégia de mapeamento do modelo abstrato para um sistema físico em operação.

O presente trabalho aborda os aspectos da especificação, modelamento e implementação de sistemas com o auxílio do modelo GMB*.

Em particular, o próximo capítulo apresenta uma estratégia de implementação baseada no conceito da máquina GMB*. Esta máquina, constituída por uma rede local de micro-computadores, ao executar o um GMB* adquire o comportamento do sistema modelado, mapeando a abstração dada pelo modelo no sistema físico em operação. Ao final do próximo capítulo, discutem-se os aspectos relacionados à especificação e ao modelamento GMB*.

CAPÍTULO 3

IMPLEMENTAÇÃO DA MÁQUINA GMB*

3.1. INTRODUÇÃO

No presente capítulo discute-se uma proposta de implementação da máquina GMB*. Tal proposta objetiva a construção de uma ferramenta que, associada ao formalismo do modelo GMB*, permita a sistematização do projeto e implementação de sistemas distribuídos de controle digital. Na concepção desta máquina enfatizou-se a adoção de uma estrutura modular, cujo elemento básico de construção fosse, a nível físico, suficientemente convencional para aproveitar as estruturas hardware disponíveis no mercado (processador, memória e dispositivos de entrada e saída). Assumiu-se também que, enquanto ferramenta de auxílio ao projeto, tal máquina deveria suportar e permitir a avaliação de diferentes graus de centralização/descentralização de um mesmo modelo GMB*. Tais enfoques levaram à concepção de uma arquitetura distribuída, denominada máquina GMB* distribuída, formada pela interconexão arbitrária de máquinas GMB* uni-processadoras, implementadas sobre a estrutura convencional de um micro-computador. Em resumo, a arquitetura proposta é uma rede de micro-computadores, cuja linguagem básica de programação de cada elemento, assim como da rede, é o GMB*.

3.2. A MÁQUINA GMB* UNI-PROCESSADORA

O módulo básico de construção da máquina GMB* distribuída é denominado máquina GMB* uni-processadora. A organização de tal módulo é apresentada na estrutura em camadas da figura 3.2.1.

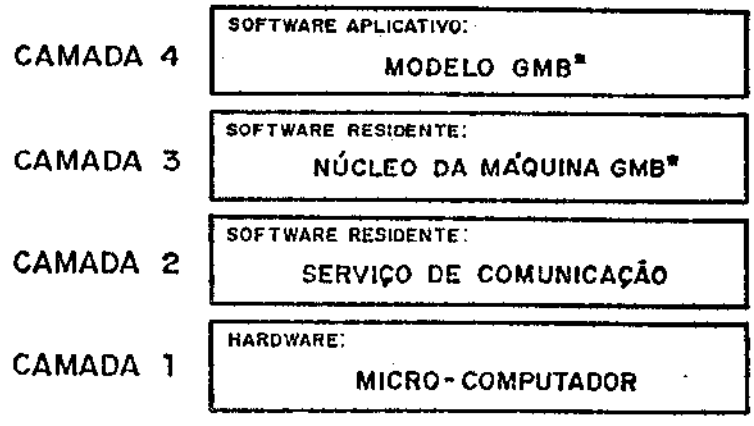


FIG.3.2.1 - ORGANIZAÇÃO DA MÁQUINA GMB* UNI-PROCESSADORA

CAMADA 1 - HARDWARE DA MÁQUINA - Fornece os recursos físicos necessários à operação da máquina GMB* uni-processadora. Neste trabalho, esta camada é constituída pelo hardware convencional de um micro-computador (figura 3.2.2). Adotou-se este tipo de hardware dada a sua disponibilidade no mercado a custos cada vez mais reduzidos. Avalia-se, também, que a adoção de uma estrutura tão amplamente difundida como a de um micro-computador é um forte incentivo à implementação da máquina GMB*.

CAMADA 2 - SERVIÇO DE COMUNICAÇÃO - Para a implementação da máquina GMB* distribuída faz-se necessária a troca de mensagens

entre as diversas máquinas GMB* uni-processadoras que a compõem. Esta camada é responsável pelos mecanismos básicos de comunicação entre as máquinas uni-processadoras. Este serviço de comunicação também é organizado em camadas, aqui denominadas de níveis, cada qual com uma função específica (figura 3.2.17) :

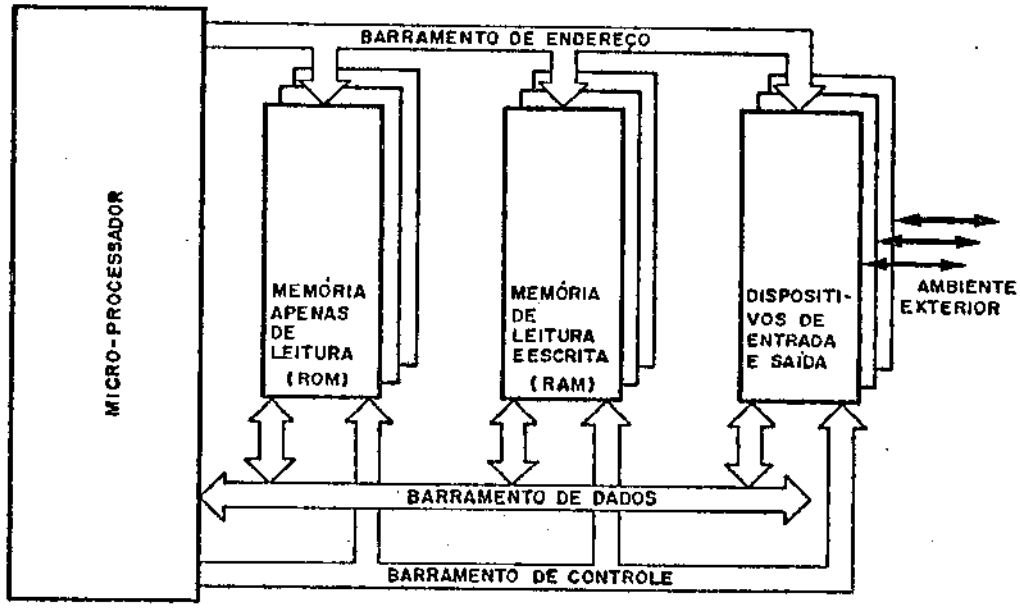


FIG.3.2.2 - ARQUITETURA TÍPICA DE UM MICRO-COMPUTADOR: CONFIGURAÇÃO MÍNIMA

NÍVEL 1 ou NÍVEL FÍSICO - Este nível engloba o hardware do canal de comunicação e os procedimentos de gerenciamento e acesso a este hardware. A função do nível 1 é a transmissão/recepção de bits, empacotados em bytes, através do canal de comunicação.

NÍVEL 2 ou NÍVEL DE QUADRO - Este nível tem por função detetar erros de transmissão. Para tanto, será calculado e anexado um código detetor de erro a cada mensagem transmitida. Este código é utilizado na recepção como elemento de teste de validade da mensagem recebida.

NÍVEL 3 ou NÍVEL DE ENLACE - A função deste nível é transformar o canal de comunicação formado pelos níveis inferiores (níveis 1 e 2) em um canal livre de erros. Este nível deve garantir que toda mensagem transmitida seja corretamente recebida. O protocolo utilizado baseia-se em um esquema de transmissão de mensagem com espera de reconhecimento denominado na literatura de "one bit sliding window protocol" /Tanenbaum 1981/.

NÍVEL 4 ou NÍVEL DE REDE - Este nível cuida do roteamento das mensagens ao longo da rede. A estratégia de roteamento a ser empregada, em uma primeira versão, será do tipo estático, onde tabelas fixas indicam, para cada destino, a rota a ser seguida.

CAMADA 3 - NÚCLEO DA MÁQUINA - Esta camada é responsável pela execução dos programas aplicativos (camada 4) segundo a dinâmica estabelecida na seção 2.3 - capítulo 2. Em uma configuração

distribuída, esta camada utiliza-se do serviço de comunicação (camada 2) para transmitir e/ou receber marcas e dados de outras máquinas. O núcleo da máquina é formado pelos seguintes elementos funcionais (figura 3.2.3) :

ESCALONADOR - O elemento escalonador efetua a seleção e ativação dos nós de controle. Este elemento, baseado na descrição do domínio do controle aplicativo (camada 4), avalia a habilitação dos nós do aplicativo e, dentre os elementos habilitados (expressão lógica de entrada verdadeira), ativa o de maior prioridade. É função do escalonador a retirada das marcas responsáveis pela ativação do nó aplicativo selecionado, assim como, pela transferência do controle da UCP para a interpretação do processador a ele associado.

PRIMITIVAS - Um conjunto de funções voltadas ao processamento GMB* e ao processamento em tempo-real serão oferecidas ao usuário na forma de primitivas. As primitivas voltadas ao processamento GMB* atendem as necessidades básicas impostas pelo modelo: escrita em armazenador, leitura de armazenador e marcação de arco de controle. Já as primitivas voltadas ao processamento em tempo-real formam uma biblioteca de funções apropriadas para este tipo de processamento que, a título de agilizar o desempenho da máquina e facilitar a confecção dos aplicativos, estará à disposição do usuário no interior da máquina GMB* uni-processadora. Em uma primeira versão, serão oferecidas as seguintes facilidades para o processamento em tempo-real:

LEITURA DE RELÓGIO EM TEMPO-REAL - Interno ao sistema existirá um relógio de tempo-real. Será facultada ao usuário a leitura deste relógio (primitiva LE$HORA).

MARCAÇÃO FREQUENCIAL DE ARCOS DE ENTRADA DO GRAFO DO CONTROLE - Será permitido ao usuário a definição de arcos de controle frequenciais. Tais arcos, uma vez ativada a marcação frequencial, são marcados automaticamente pelo núcleo na frequência definida pelo usuário (primitivas ATIVA$FREQUENCIAL e DESATIVA$FREQUENCIAL).

MECANISMOS PARA A DEFINIÇÃO DE ESTRUTURAS DE TEMPORIZAÇÃO (TIME-OUT) - Em adição aos arcos frequenciais, será ainda permitido ao usuário a definição de arcos de temporização. Tais arcos admitem que o usuário programe individualmente os seus instantes de marcação (primitivas ATIVA$ESPERA e DESATIVA$ESPERA).

MARCAÇÃO DE ARCOS DE ENTRADA DO GRAFO DO CONTROLE VIA INTERRUPÇÃO : O usuário também poderá, por programação, associar os canais de interrupção disponíveis na máquina GMB* uni-processadora a arcos de entrada do grafo do controle aplicativo. Uma vez estabelecida a associação, o atendimento de um pedido de interrupção de um determinado canal de interrupção acarreta a marcação do arco de controle a ele associado (primitiva HABILITA$INTERRUPÇÃO e DESABILITA$INTERRUPÇÃO).

RELÓGIO EM TEMPO-REAL - Somado às necessidades básicas do

processamento GMB*, embutido no núcleo da máquina, ter-se-á um relógio em tempo-real. Este relógio, além de assumir o encargo de agente temporizador (time-out) do serviço de comunicação, fornecerá o suporte necessário para a implementação das primitivas voltadas ao processamento em tempo-real.

TRATADOR DOS CANAIS DE INTERRUPÇÃO DO USUÁRIO - Este elemento, dado o hardware convencional de um micro-computador, tem por função converter os pedidos dos canais de interrupção disponíveis para uso do usuário em marcas do grafo do controle aplicativo. O usuário pode, através da utilização das primitivas adequadas, associar os canais de interrupção disponíveis a arcos de entrada do grafo do controle aplicativo. No instante desta associação, o usuário define, também, a prioridade da marca a ser gerada. Uma vez ocorrendo o pedido de interrupção de um canal, o tratador dos canais de interupções se encarrega de marcar o arco de controle a ele associado.

CAMADA 4 - PROGRAMA GMB* APLICATIVO - Esta camada é o modelo GMB* desenvolvido pelo usuário para atender às suas necessidades. Tal camada é interpretada e executada pela máquina GMB* formada pelas camadas inferiores.

Nas seções 3.2.1. e 3.2.2. são apresentadas as especificações funcionais das camadas 2 e 3. Nestas seções adotou-se como formalismo de apresentação o próprio modelo GMB*. Já as seções 3.2.3. e 3.2.4. se preocupam com a caracterização das camadas 1 e 4.

3.2.1. NÚCLEO DA MÁQUINA GMB* UNI-PROCESSADORA

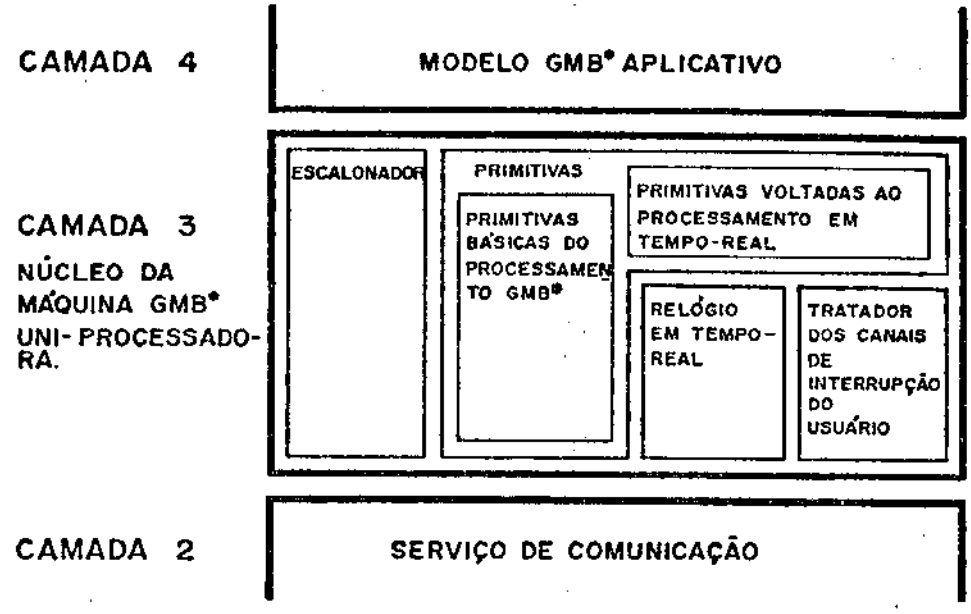


FIG.3.2.3 - ELEMENTOS FUNCIONAIS DO NÚCLEO DA MÁQUINA GMB* UNI-PROCESSADORA

O núcleo da máquina GMB* uni-processadora é o elemento responsável pela interpretação e execução dos programas GMB* aplicativos consoante a dinâmica estabelecida na seção 2.3 -

capítulo 2. A figura 3.2.3 apresenta os blocos funcionais deste núcleo.

3.2.1.1. ESCALONADOR

O escalonamento dos nós aplicativos é realizado, através da interpretação do domínio do controle aplicativo (camada 4), por um escalonador de tarefas. Ao ser executado, este escalonador avalia a habilitação dos nós aplicativos e, dentre estes, seleciona para ativação o mais prioritário, transferindo o controle da máquina para a interpretação do processador a ele associado. No instante de ativação do nó, o escalonador retira da descrição do domínio do controle aplicativo (camada 4) as marcas que participaram da ativação do nó. Tanto o critério de seleção do nó a ser ativado, como o da escolha das marcas a serem retiradas, segue o estabelecido na seção 2.3 do capítulo 2. Terminada a sua execução, a interpretação do processador aplicativo deve retornar o controle da máquina ao escalonador para que este possa selecionar e ativar um próximo elemento. Caso não exista nó que satisfaça as condições de ativação, o escalonador permanece em um ciclo de espera, testando a habilitação dos nós, até o surgimento de um candidato à ativação.

Na figura 3.2.4 são apresentados os domínios do controle e dos dados do modelo GMD* do escalonador. Segue abaixo o domínio da interpretação do referido modelo.

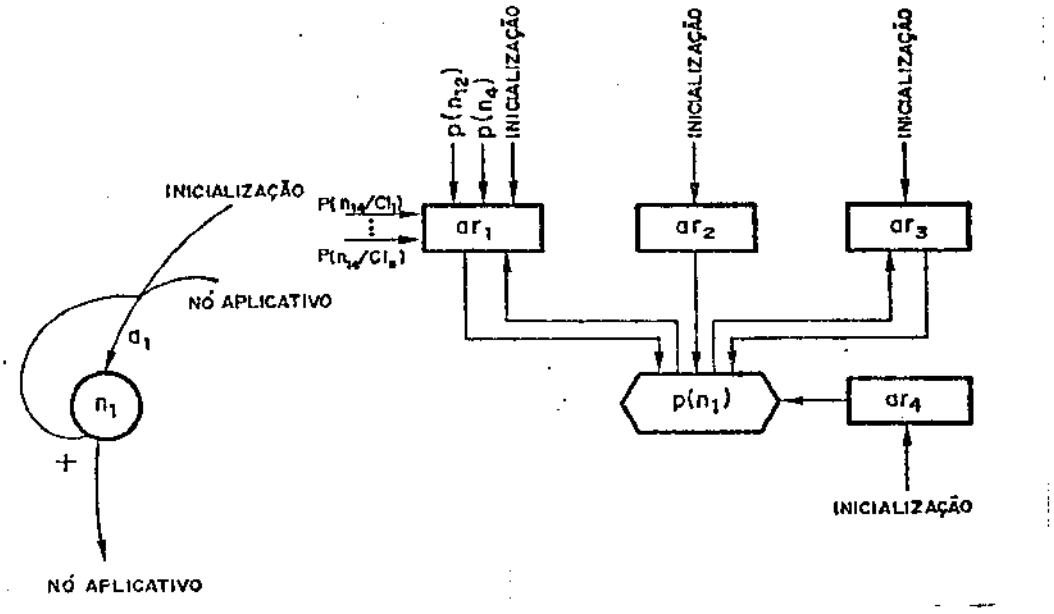


FIG.3.2.4 - MODELO GMD* DO ESCALONADOR: DOMÍNIO DO CONTROLE E DOMÍNIO DOS DADOS

DOMÍNIO DA INTERPRETAÇÃO

INTERPRETAÇÃO DOS ARMAZENADORES

ar1 - Estado dos arcos de controle do aplicativo
ar1::=(<IDENTIFICADOR ARCO DE CONTROLE>
<prioridade marcas residentes>)

```
<prioridade marcas residentes>::=(<PRIORIDADE MARCA>)

ar2 - Descrição do grafo do controle do aplicativo
ar2::=(<IDENTIFICADOR NÓ><EXPRESSÃO LÓGICA DE ENTRADA>)
ar3 - Estado dos nós de controle do aplicativo
ar3::=(<IDENTIFICADOR NÓ><TEMPO DE ESPERA>)

ar4 - Tabela de localização das interpretações dos processadores
associados aos nós de controle do aplicativo
ar4::=(<IDENTIFICADOR NÓ><localização>)
<localização>::=<IDENTIFICADOR ARCO DE CONTROLE>
```

INTERPRETAÇÃO DOS PROCESSADORES

```
INTERPRETAÇÃO p(n1) ;
BEGIN
 /* leia ar1 e ar2 */ ;
 /* verifique se existe nó habilitado */ ;
 IF /* não existe nó habilitado */ THEN
 BEGIN
  /* marque a1 com prioridade 1 */
 END
 ELSE
 BEGIN
  IF /* existe apenas um nó habilitado */ THEN
  BEGIN
   /* retire de ar1 as marcas que participaram da habilitação
   deste nó (seção 2.3 - capítulo 2) */ ;
   /* zere em ar3 o tempo de espera associado a este nó */ ;
   /* leia de ar4 a localização deste nó */ ;
   /* marque o arco de controle associado a esta localização com
   prioridade 1 */
  END
  ELSE
  BEGIN
   IF /* existe mais de um nó habilitado */ THEN
   BEGIN
    /* incremente de uma unidade o tempo de espera de todos os
    nós habilitados */ ;
    /* calcule a prioridade de cada nó habilitado (seção 2.3 -
    capítulo 2) */ ;
    /* avalie o valor da maior prioridade associada ao nós
    habilitados */ ;
    IF /* existe apenas um nó habilitado com esta prioridade */
    THEN
    BEGIN
     /* retire de ar1 as marcas que participaram da habilitação
     deste nó (seção 2.3 - capítulo 2) */ ;
     /* zere em ar3 o tempo de espera associado a este nó */ ;
     /* leia de ar4 a localização deste nó */ ;
     /* marque o arco de controle associado a esta localização com
     prioridade 1 */
    END
    ELSE
    BEGIN
     IF /* existe mais de um nó com esta prioridade */ THEN
     BEGIN
      /* leia de ar3 os tempos de espera destes nós */ ;
```

```
        /* avalie o valor do maior tempo de espera associado a
        estes nós */ ;
        IF /* existe apenas um nó com este tempo de espera */ THEN
        BEGIN
         /* retire de ar1 as marcas que participaram da habilitação
         deste nó (seção 2.3 - capítulo 2) */ ;
         /* zere em ar3 o tempo de espera associado a este nó */ ;
         /* leia de ar4 a localização deste nó */ ;
         /* marque o arco de controle associado a esta
         localização com prioridade 1 */
        END
        ELSE
        BEGIN
         IF /* existe mais de um nó com este tempo de espera */
         THEN
         BEGIN
          /* escolha um destes nós para ser ativado */ ;
          /* retire de ar1 as marcas que participaram da habilitação
          do nó escolhido (seção 2.3 - capítulo 2) */ ;
          /* zere em ar3 o tempo de espera associado ao nó
          escolhido */ ;
          /* leia de ar4 a localização do nó escolhido */ ;
          /* marque arco de controle associado a esta
          localização com prioridade 1 */
         END
        END
       END
      END
     END
    END
   END
END.
```

3.2.1.2. **PRIMITIVAS**

Durante a sua execução, um processador aplicativo pode, através de chamadas a primitivas, utilizar os serviços de um conjunto de funções residentes no interior do núcleo da máquina GMB* uni-processadora. Tais primitivas procuram atender a duas classes distintas de necessidades: as necessidades ditadas pelo processamento GMB* e as voltadas à aplicação.

O atendimento das necessidades básicas imposta pelo GMB* conduziram à criação das chamadas primitivas básicas do processamento GMD*: escrita em armazenador, leitura de armazenador e marcação de arco de controle.

Já as primitivas associadas à aplicação voltam-se, neste trabalho, exclusivamente ao processamento em tempo-real. Destaque-se que o conjunto de primitivas aqui proposto não tem a pretensão de cobrir todo o espectro de necessidades de uma aplicação voltada ao processamento em tempo-real, mas sim, ser um primeiro passo na busca de um conjunto de primitivas que atendam com eficiência as necessidades básicas deste tipo de processamento.

Nas seções que se seguem, seções 3.2.1.2.1. e 3.2.1.2.2., são

apresentadas, respectivamente, as primitivas associadas ao processamento GMB* e à aplicação.

3.2.1.2.1. PRIMITIVAS BÁSICAS DO PROCESSAMENTO GMB*

Como já mencionado, são três as primitivas básicas associadas ao GMB*: escrita em armazenador, leitura de armazenador e marcação de arco de controle. Tais primitivas oferecem os recursos mínimos para a execução de um modelo GMB*.

No que se refere aos armazenadores, em uma primeira versão, serão dois os atributos que o usuário deverá definir no domínio da interpretação : tamanho máximo e o tipo de dados que comporta.

O tamanho máximo, em bytes, define a área de memória a ser reservada para o armazenador.

O tipo de dados refere-se às características dos dados que o armazenador comporta. Nesta primeira versão, distinguem-se os seguintes tipos de dados : dados de tamanho fixo e dados de tamanho variável. Ao explicitar no domínio da interpretação que determinado armazenador comporta dados de tamanho fixo, o usuário deverá explicitar, também, o comprimento, em bytes, do mesmo. Qualquer acesso a um armazenador de tamanho fixo manipula sempre a mesma quantidade de bytes, não sendo necessário, portanto, para cada bloco de dados armazenado, guardar também a informação sobre seus respectivos tamanhos. Tal informação faz parte da descrição do armazenador e está definida no domínio da interpretação do aplicativo (camada 4). Já o mesmo não acontece com armazenadores de dados de tamanho variável. Para este tipo de armazenador faz-se necessário, para cada bloco de dados armazenado, o armazenamento conjunto da informação sobre o seu tamanho.

Como estrutura única de implementação dos armazenadores adotou-se a fila circular. Uma fila circular é formada, a nível físico, por um conjunto finito de posições adjacentes de memória, compondo uma área que possui um endereço de início e um endereço de fim. A nível lógico entende-se que o endereço subsequente ao endereço de fim é o endereço de início. Tal visualização empresta à estrutura o caráter circular. A característica de fila está associada ao mecanismo de entrada e retirada de dados. No caso, o mecanimo é FIFO, onde o primeiro elemento colocado é o primeiro a ser retirado.

Nos armazenadores de dados de tamanho variável, a informação sobre o comprimento de um bloco de dados será sempre armazenada na primeira posição disponível para a entrada dos dados, sendo seguida imediatamente dos bytes componentes do bloco de dados. Este mecanismo de passagem do comprimento de um bloco de dado de tamanho variável também está presente nas primitivas de acesso a armazenadores. No acesso a armazenadores de tamanho variável as primitivas de escrita e leitura sempre interpretarão o primeiro byte como indicador do tamanho do bloco a ser transferido.

Isto posto, segue abaixo (seções 3.2.1.2.1.1., 3.2.1.2.1.2. e 3.2.1.2.1.3.) a descrição de cada uma das três primitivas básicas a serem implementadas no núcleo da máquina GMB* uni-processadora.

3.2.1.2.1.1. **ESCRITA EM ARMAZENADOR**

ESCREVE (armazenador , acesso , dados)

onde:

armazenador - parâmetro de entrada que especifica o identificador do armazenador a ser acessado.
acesso - parâmetro de entrada que especifica o tipo de acesso a ser realizado. Os tipos possíveis de acesso são: acesso não-destrutivo e acesso destrutivo. No acesso não-destrutivo o pedido de escrita será desprezado caso o armazenador não comporte, por falta de espaço, o bloco de dados a ser transferido. No acesso destrutivo, caso haja falta de espaço, são descartados do armazenador tantos blocos de dados quantos forem necessários para a acomodação do novo bloco de dados. O mecanismo de descarte segue o critério de antiguidade (FIFO - first in / first out).
dados - parâmetro de entrada que especifica o endereço inicial do bloco de dados objeto da transferência.

Na figura 3.2.5 são apresentados os domínios do controle e dos dados do modelo GMB* desta primitiva. Apresenta-se a seguir o domínio da interpretação.

DOMÍNIO DA INTERPRETAÇÃO

INTERPRETAÇÃO DOS ARMAZENADORES

ar5 - Pedido de escrita em armazenador
ar5::=<IDENTIFICADOR ARMAZENADOR><tipo de acesso>
<ENDEREÇO DOS DADOS>
<tipo de acesso>::=<DESTRUTIVO>/<NÃO-DESTRUTIVO>

ar6 - Descrição dos armazenadores do aplicativo
ar6::=(<IDENTIFICADOR ARMAZENADOR><tipo dos dados><localização>)
<tipo dos dados>::=<tamanho fixo>/<TAMANHO VARIÁVEL>
<tamanho fixo>::=<TAMANHO>
<localização>::=<interna>/<EXTERNA>
<interna>::=<ENDEREÇO DE INÍCIO><TAMANHO MÁXIMO>
<APONTADOR DE ENTRADA><APONTADOR DE SAÍDA>
<CONTADOR OCUPAÇÃO>

ar7 - Pedido de transmissão de dados
ar7::=<IDENTIFICADOR ARMAZENADOR><tipo acesso><dados>
<tipo acesso>::=<DESTRUTIVO>/<NÃO-DESTRUTIVO>
<dados>::=<TAMANHO DO BLOCO DE DADOS><BLOCO DE DADOS>/
<BLOCO DE DADOS>

ar8 - Sequenciador de execução
ar8::=<IDENTIFICADOR ARCO DE CONTROLE>

ar59 - Armazenador da recepção
ar59::=<TAMANHO DO BLOCO DE DADOS><BLOCO DE DADOS>/
<BLOCO DE DADOS>

ar61 - Sequenciador de execução
ar61::=<IDENTIFICADOR ARCO DE CONTROLE>

ar63 - Pedido de escrita em armazenador
ar63::=<IDENTIFICADOR ARMAZENADOR><tipo de acesso>
<ENDEREÇO DOS DADOS>
<tipo de acesso>::=<DESTRUTIVO>/<NÃO-DESTRUTIVO>

ar64 - Sequenciador de execução
ar64::=<IDENTIFICADOR ARCO DE CONTROLE>

ar65 - Pedido de escrita em armazenador
ar65::=<IDENTIFICADOR ARMAZENADOR><tipo de acesso>
<ENDEREÇO DOS DADOS>
<tipo de acesso>::=<DESTRUTIVO>/<NÃO-DESTRUTIVO>

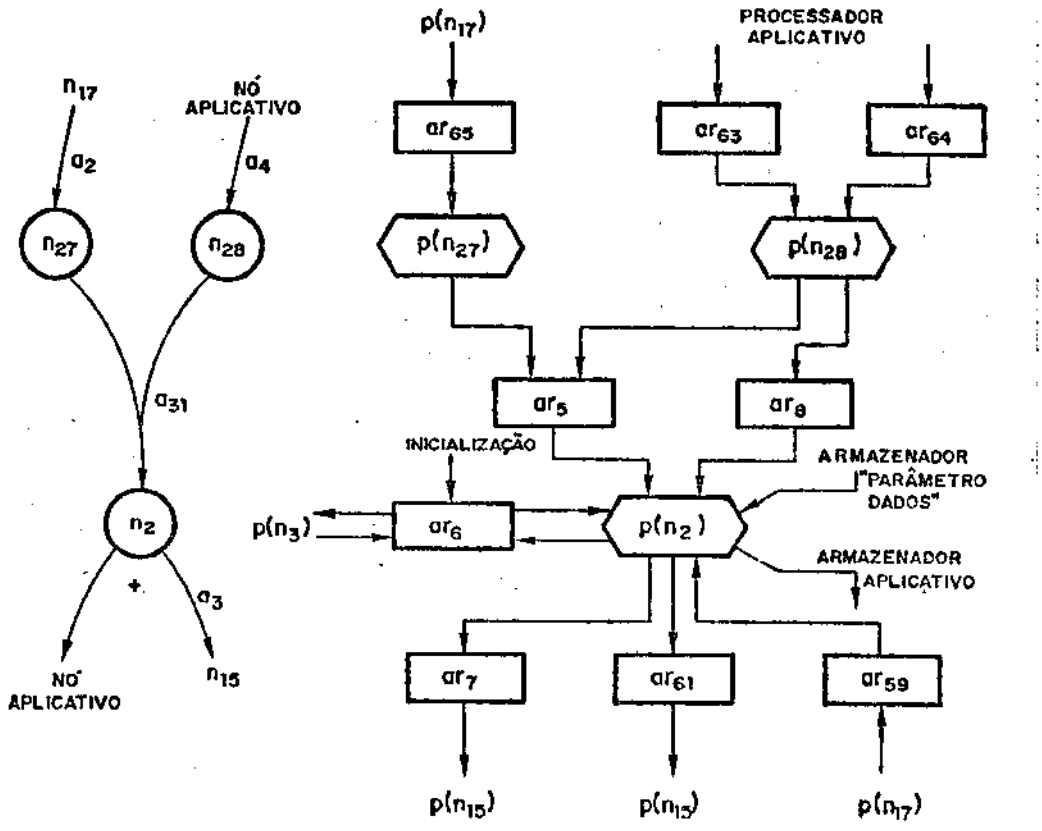


FIG.3.2.5 - MODELO GMD* DA PRIMITIVA DE ESCRITA EM ARMAZENADOR: DOMÍNIO DO CONTROLE E DOMÍNIO DOS DADOS

INTERPRETAÇÃO DOS PROCESSADORES

```
INTERPRETAÇÃO p(n2) ;
BEGIN
 /* leia de ar5 o pedido de escrita */ ;
 /* leia de ar6 a descrição do armazenador aplicativo a ser
 acessado */ ;
 IF /* localização do armazenador aplicativo é externa */ THEN
 BEGIN
  /* escreva em ar7 o identificador do armazenador a ser acessado
  e o tipo de acesso a ser realizado */ ;
  IF /* tipo de dados que o armazenador comporta é de tamanho fixo
  */ THEN
```

```
 BEGIN
  /*  leia  a  partir  do endereço dos dados  tantos  bytes  quanto
  especificado no tamanho fixo */ ;
  /*   escreva   este   bloco   de   dados   em   ar7    no    campo
  correspondente */
 END
 ELSE
 BEGIN
  IF  /*  tipo  de dados que o armazenador comporta  é  de  tamanho
  variável */ THEN
  BEGIN
   /*  leia  o byte residente no endereço dos dados (este  primeiro
   byte indica o tamanho de bloco de dados a ser transferido)*/;
   /*  escreva  em  ar7,  no campo correspondente,  o  tamanho  do
   bloco de dados */ ;
   /*  leia a partir do endereço de dados + 1 tantos  bytes  quanto
   especificado pelo tamanho do bloco de dados */ ;
   /*  escreva  em  ar7,  no campo correspondente,  este  bloco  de
   dados */
  END
 END ;
 /* leia o sequenciador de execução definido em ar8 */ ;
 IF /* sequenciador de execução especifica arco de controle */ THEN
 BEGIN
  /* escreva o identificador deste arco de controle em ar61 */
 END ;
 /* marque a3 com prioridade 2 (comunicação externa) */
END
ELSE
BEGIN
 IF /* localização do armazenador aplicativo é interna */ THEN
 BEGIN
  IF  /*  tipo  de dados que o armazenador comporta  é  de  tamanho
  fixo */ THEN
  BEGIN
   /*  verifique  se  armazenador possui  espaço  livre  suficiente
   para   acomodar   o  bloco  de  dados  objeto  da   transferência
   (tamanho máximo - contador ocupação >= tamanho fixo) */ ;
   IF */ armazenador possui espaço livre suficiente */ THEN.
   BEGIN
    /*  leia  a partir do endereço dos dados  tantos  bytes  quanto
    indicado pelo tamanho fixo */ ;
    /*  escreva  no armazenador aplicativo,  a partir  do  endereço
    indicado pelo apontador de entrada,  este bloco de dados (levar
    em  conta  a  característica  circular do armazenador aplicati-
    vo) */ ;
    /*  some  ao valor do apontador de entrada o valor  do  tamanho
    fixo  (levar em conta a característica circular do  armazenador
    aplicativo) */ ;
    /*  escreva  o  resultado desta soma em ar6 como o  novo  valor
    do apontador de entrada */ ;
    /*  some  ao valor do contador de ocupação o valor  do  tamanho
    fixo */ ;
    /*  escreva  o resultado desta soma em ar6 como  o  novo  valor
    do contador de ocupação */
   END
   ELSE
   BEGIN
```

```
  IF /* armazenador não possui espaço livre suficiente */ THEN
  BEGIN
   IF /* tipo de acesso é destrutivo */ THEN
   BEGIN
    /* some ao valor do apontador de saída o valor do tamanho
    fixo (levar em conta a característica circular do armazenador
    aplicativo) */ ;
    /* escreva o resultado desta soma em ar6 como o novo valor
    do apontador de saída */ ;
    /* leia, a partir do endereço dos dados, tantos bytes
    quanto indicado pelo tamanho fixo */ ;
    /* escreva este bloco de dados no armazenador aplicativo a
    partir do endereço indicado pelo apontador de entrada (levar
    em conta a característica circular do armazenador aplicati-
    vo) */ ;
    /* some ao valor do apontador de entrada o valor do
    tamanho fixo (levar em conta a característica circular do
    armazenador aplicativo) */ ;
    /* escreva o resultado desta soma em ar6 como o novo valor
    do apontador de entrada */
   END
  END
 END
END
ELSE
BEGIN
 IF /* tipo de dados que o armazenador comporta é de tamanho
 variável */ THEN
 BEGIN
  /* leia o byte residente no endereço dos dados (este primeiro
  byte indica o tamanho do bloco de dados a ser transferido) */ ;
  /* verifique se o armazenador possui espaço livre suficiente
  para acomodar o bloco de dados objeto da transferência
  (tamanho máximo - contador ocupação >= tamanho do bloco de
  dados +1) */ ;
  IF /* armazenador possui espaço livre suficiente */ THEN
  BEGIN
   /* coloque no endereço indicado pelo apontador de entrada o
   byte referente ao tamanho do bloco de dados a ser
   transferido */ ;
   /* leia a partir do endereço subsequente ao endereço dos
   dados tantos bytes quanto indicado pelo tamanho do bloco de
   dados */ ;
   /* escreva este bloco de dados no armazenador aplicativo a
   partir do endereço subsequente ao indicado pelo apontador de
   entrada (levar em conta a característica circular do
   armazenador aplicativo) */ ;
   /* some ao valor do apontador de entrada o valor do tamanho
   do bloco de dados transferido + 1 (levar em conta a
   característica circular do armazenador aplicativo) */ ;
   /* escreva o resultado desta soma em ar6 como o novo valor
   do apontador de entrada */ ;
   /* some ao valor do contador de ocupação o valor do tamanho
   do bloco de dados transferido + 1 */ ;
   /* escreva o resultado desta soma em ar6 como o novo valor
   do contador de ocupação */
  END
  ELSE
```

```
      BEGIN
       IF /* armazenador não possui espaço livre suficiente */ THEN
       BEGIN
        IF /* tipo de acesso é destrutivo */ THEN
        BEGIN
         WHILE  /*  não houver espaço livre suficiente (tamanho
         máximo - contador ocupação < tamanho do bloco de
         dados +1) */ DO
         BEGIN
          /* leia byte apontado pelo apontador de saída (este
          primeiro byte indica o tamanho do bloco de dados que o
          sucede no armazenador) */ ;
          /* some ao valor do apontador de saída o valor deste
          byte + 1 (levar em conta a característica circular do
          armazenador aplicativo) */ ;
          /* escreva o resultado desta soma em ar6 como o novo
          valor do apontador de saída */ ;
          /* subtraia do valor do contador de ocupação o valor do
          byte + 1 */ ;
          /* escreva o resultado desta soma em ar6 como o novo
          valor do contador de ocupação */
         END ;
         /* coloque no endereço indicado pelo apontador de entrada
         o byte referente ao tamanho do bloco de dados a
         ser transferido */ ;
         /* leia a partir do endereço subsequente ao endereço dos
         dados tantos bytes quanto indicado pelo tamanho do bloco
         de dados */ ;
         /* escreva este bloco de dados no armazenador aplicativo a
         partir do endereço subsequente ao indicado pelo apontador de
         entrada (levar em conta a característica circular do
         armazenador aplicativo) */ ;
         /* some ao valor do apontador de entrada o valor do
         tamanho do bloco de dados transferido + 1 (levar em conta a
         característica circular do armazenador aplicativo) */ ;
         /* escreva o resultado desta soma em ar6 como o novo valor
         do apontador de entrada */ ;
         /* some ao valor do contador de ocupação o valor do
         tamanho do bloco de dados transferido + 1 */ ;
         /* escreva o resultado desta soma em ar6 como o novo valor
         do contador de ocupação */
        END
       END
      END
     END
    END ;
    /* leia o sequenciador de execução definido em arB */ ;
    IF  /* sequenciador especifica identificador de arco de
    controle */ THEN
    BEGIN
     /* marque este arco com prioridade 1 */
    END
   END
END.

INTERPRETAÇÃO p(n27) ;
BEGIN
```

```
 /* leia ar65 */ ;
 /* escreva o conteúdo de ar65 em ar5 */ ;
 /* marque a31 com prioridade 3 */
END.

INTERPRETAÇÃO p(n28) ;
BEGIN
 /* leia ar63 */ ;
 /* escreva o conteúdo de ar63 em ar5 */ ;
 /* leia ar64 */ ;
 /* escreva o conteúdo de ar64 em ar8 */ ;
 /* marque a31 com prioridade 3 */
END.
```

3.2.1.2.1.2. **LEITURA DE ARMAZENADOR**

LE (armazenador , acesso , destino)

onde:

armazenador - parâmetro de entrada que especifica o identificador do armazenador a ser acessado.
acesso - parâmetro de entrada que especifica o tipo de acesso a ser realizado. São possíveis os seguintes tipos de acesso: acesso não-consumptivo e acesso consumptivo. No acesso não-consumptivo a leitura de dados faz-se através do mecanismo de cópia, mantendo-se no armazenador os dados lidos. No acesso consumptivo os dados lidos são retirados do armazenador, liberando espaço para futuros armazenamentos.
destino - parâmetro de entrada que especifica o endereço inicial a partir do qual deverão ser depositados os dados lidos. No endereço destino é colocado o tamanho do bloco de dados lido seguido dos bytes que compoem o bloco de dados propriamente dito. Caso o armazenador se encontre vazio no instante do acesso de leitura, no endereço destino é colocado o valor zero como tamanho de bloco de dados lido.

Na figura 3.2.6 são apresentados os domínios do controle e dos dados do modelo GMD* desta primitiva. Apresenta-se a seguir o domínio da interpretação.

DOMÍNIO DA INTERPRETAÇÃO

INTERPRETAÇÃO DOS ARMAZENADORES

```
ar6  Descrição dos armazenadores do aplicativo
ar6::=(<IDENTIFICADOR ARMAZENADOR><tipo dos dados><localização>)
<tipo dos dados>::=<tamanho fixo><TAMANHO VARIÁVEL>
<tamanho fixo>::=<TAMANHO>
<localização>::=<interna>/<EXTERNA>
<interna>::=<ENDEREÇO DE INÍCIO><TAMANHO MÁXIMO>
            <APONTADOR DE ENTRADA><APONTADOR DE SAÍDA>
            <CONTADOR OCUPAÇÃO>

ar9 - Pedido de leitura de armazenador
ar9::=<IDENTIFICADOR ARMAZENADOR><tipo acesso><ENDEREÇO DESTINO>
```

<tipo acesso>::=<CONSUMPTIVO>/<NÃO-CONSUMPTIVO>

ar10 - Sequenciador de execução
ar10::=<IDENTIFICADOR ARMAZENADOR>

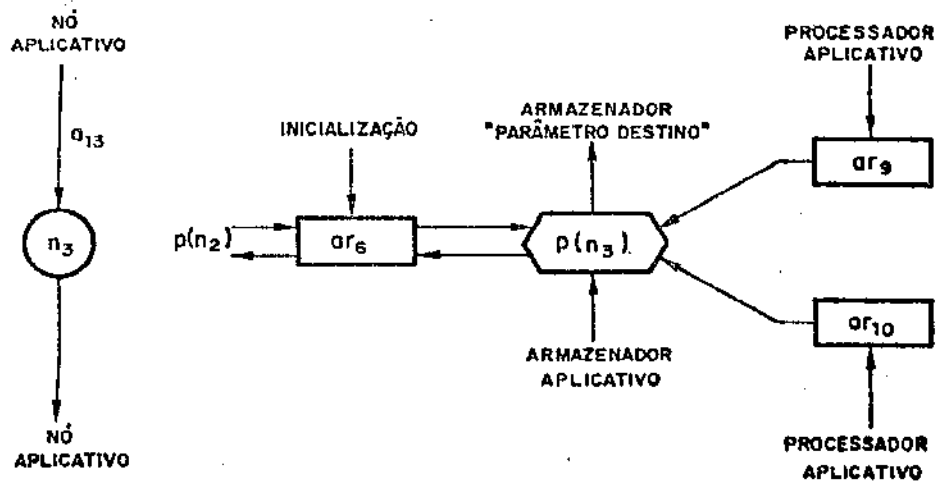


FIG.3.2.6 - MODELO GMB* DA PRIMITIVA DE LEITURA DE ARMAZENADOR: DOMÍNIO DO CONTROLE E DOMÍNIO DOS DADOS

INTERPRETAÇÃO DOS PROCESSADORES

```
INTERPRETAÇÃO p(n3) :
BEGIN
 /* leia de ar9 o pedido de leitura */ ;
 /* leia de ar6 a descrição do armazenador aplicativo a ser
 acessado */ ;
 /* verifique se o armazenador contém dados (contador
 ocupação > 0) */ ;
 IF /* armazenador aplicativo não contém dados */ THEN
 BEGIN
  /* coloque zero na primeira posição do endereço destino */
 END
 ELSE
 BEGIN
  IF /* armazenador aplicativo contém dados */ THEN
  BEGIN
   IF  /* tipo de dados que o armazenador comporta é de tamanho fixo
   */ THEN
   BEGIN
    /* coloque na primeira posição do endereço destino o valor do
    tamanho fixo */ ;
    /* leia do armazenador aplicativo, a partir do apontador de
    saída, tantos bytes quanto indicado por este tamanho (levar em
    conta a característica circular do armazenador aplicati-
    vo) */ ;
    /* coloque estes dados a partir do endereço destino + 1 */ ;
    IF /* tipo de acesso é consumptivo */ THEN
    BEGIN
     /* subtraia do valor do contador de ocupação o valor do
     tamanho fixo */ ;
     /* escreva o resultado desta subtração em ar6 como o novo
     valor do contador de ocupação */ ;
     /* some ao valor do apontador de saída o valor do
```

```
      tamanho fixo (levar em conta a característica circular do
      armazenador aplicativo) */ ;
      /* escreva o resultado desta adição em ar6 como o novo valor
      do apontador de saída */
     END
    END
    ELSE
    BEGIN
     IF /* tipo de dados que o armazenador comporta é de tamanho
     variável */ THEN
     BEGIN
      /* leia do armazenador aplicativo o byte apontado pelo
      apontador de saída (este byte indica o tamanho do bloco de
      dados a ser transferido) */ ;
      /* coloque este byte na primeira posição do endereço
      destino */ ;
      /* leia do armazenador aplicativo, a partir do endereço
      subsequente ao indicado pelo apontador saída, tantos bytes
      quanto indicado pelo tamanho do bloco de dados a ser
      transferido (levar em conta a característica circular do
      armazenador aplicativo) */ ;
      /* coloque este bloco de dados a partir do endereço
      destino + 1 */ ;
      IF /* tipo de acesso é consumptivo */ THEN
      BEGIN
       /* subtraia do valor do contador de ocupação o valor do
       tamanho do bloco de dados transferido + 1 */ ;
       /* escreva o resultado desta subtração em ar6 como o novo
       valor do contador de posição */ ;
       /* some ao valor do apontador de saída o valor do tamanho
       do bloco de dados transferido + 1 (levar em conta a
       característica circular do armazenador aplicativo) */ ;
       /* escreva o resultado desta soma em ar6 como o novo valor
       do apontador de saída */
      END
     END
    END
   END
  END ;
  /* leia de ar10 o identificador do arco que deve ser marcado
  para a continuação do aplicativo */ ;
  /* marque este arco com prioridade 1 */
END.
```

3.2.1.2.1.3. **MARCAÇÃO DE ARCO DE CONTROLE**

MARCA (arco , prioridade)

onde:

arco - parâmetro de entrada que especifica o identificador do arco de controle a ser marcado.
prioridade - parâmetro de entrada que especifica a prioridade da marca a ser depositada no arco de controle especificado.

Na figura 3.2.7 são apresentados os domínios do controle e dos

dados do modelo GMB* desta primitiva. Apresenta-se a seguir o domínio da interpretação.

DOMÍNIO DA INTERPRETAÇÃO

INTERPRETAÇÃO DOS ARMAZENADORES

ar1 - Estado dos arcos de controle
ar1::=(<IDENTIFICADOR ARCO DE CONTROLE>
<prioridade marcas residentes>)
<prioridade marcas residentes>::=(<PRIORIDADE MARCA>)

ar11 - Pedido de marcação de arco de controle
ar11::=<IDENTIFICADOR ARCO DE CONTROLE><PRIORIDADE MARCA>

ar12 - Tabela de localização dos nós destino dos arcos de controle
ar12::=(<IDENTIFICADOR ARCO DE CONTROLE><localização>)
<localização>::=<INTERNA>/<EXTERNA>

ar13 - Pedido de transmissão de marca
ar13::=<IDENTIFICADOR ARCO DE CONTROLE><PRIORIDADE MARCA>

ar14 - Sequenciador de execução
ar14::=<IDENTIFICADOR ARCO DE CONTROLE>

ar82 - Sequenciador de execução
ar82::=<IDENTIFICADOR ARCO DE CONTROLE>

ar86 - Pedido de marcação de arco de controle
ar86::=<IDENTIFICADOR ARCO DE CONTROLE><PRIORIDADE MARCA>

ar87 - Sequenciador de execução
ar87::=<IDENTIFICADOR ARCO DE CONTROLE>

ar88 - Pedido de marcação de arco de controle
ar88::=<IDENTIFICADOR ARCO DE CONTROLE><PRIORIDADE MARCA>

INTERPRETAÇÃO DOS PROCESSADORES

```
INTERPRETAÇÃO p(n4) ;
BEGIN
 /* leia de ar11 o pedido de marcação */ ;
 /* leia de ar12 a localização dos nós destino do arco */ ;
 IF /* os nós destino do arco são internos */ THEN
 BEGIN
  /* escreva em ar1 a marca solicitada */ ;
  /* leia o sequenciador de execução definido em ar14 */ ;
  IF   /* sequenciador especifica identificador de arco de
  controle */   THEN
  BEGIN
   /* marque este arco com prioridade 1 */
  END
 END
 ELSE
 BEGIN
  IF /* os nós destino do arco são externos */ THEN
  BEGIN
   /* coloque em ar13 pedido de transmissão de marca */ ;
```

```
    /* leia o sequenciador de execução definido em ar14 */ ;
    IF /* sequenciador  especifica  identificador  arco  de  controle
    */ THEN
    BEGIN
     /* escreva o identificador deste arco em ar62 */
    END ;
    /* marque a6 com prioridade 2 */
   END
  END
END.

INTERPRETAÇÃO p(n29) ;
BEGIN
 /* leia ar68 */ ;
 /* escreva o conteúdo de ar68 em a11 */ ;
 /* marque a32 com prioridade 3 */
END.

INTERPRETAÇÃO p(n30) ;
BEGIN
 /* leia ar66 */ ;
 /* escreva o conteúdo de ar66 em ar11 */ ;
 /* leia ar67 */ ;
 /* escreva o conteúdo de ar67 em ar14 */ ;
 /* marque a32 com prioridade 3 */
 END.
```

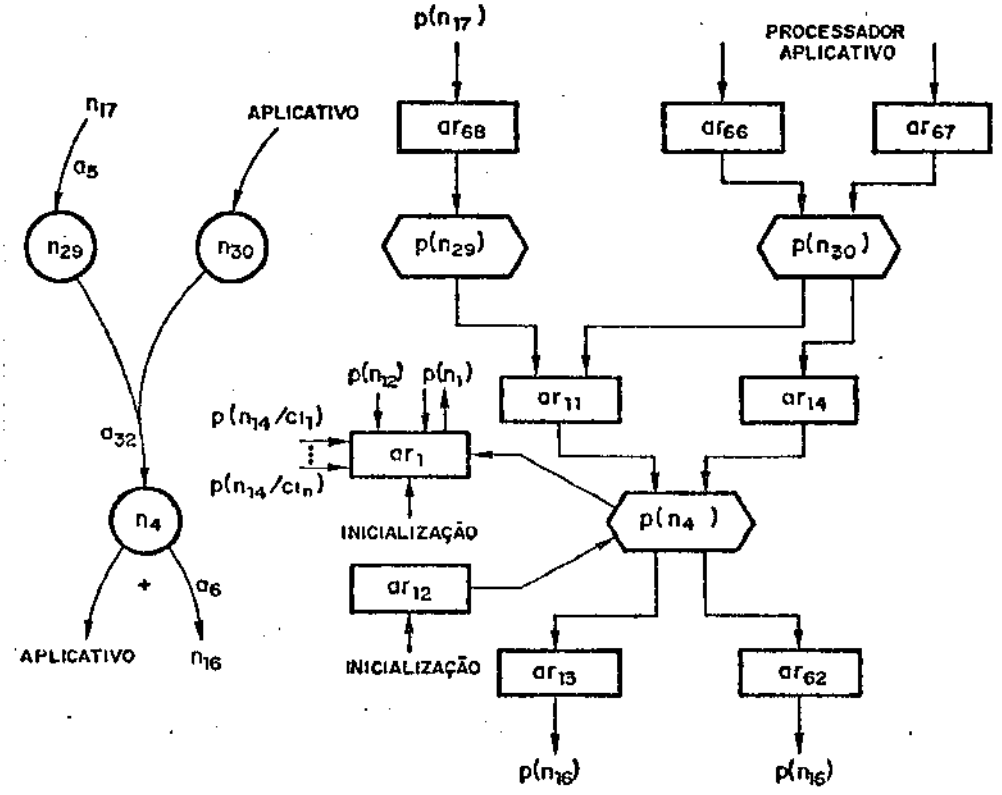


FIG.3.2.7 - MODELO GMB* DA PRIMITIVA DE MARCAÇÃO DE ARCO DE CONTROLE: DOMÍNIO DO CONTROLE E DOMÍNIO DOS DADOS

3.2.1.2.2. **PRIMITIVAS VOLTADAS AO PROCESSAMENTO EM TEMPO-REAL**

A título de agilizar o desempenho da máquina GMB* uni-processadora e facilitar a confecção dos aplicativos, será oferecida ao usuário uma biblioteca de primitivas voltadas ao processamento em tempo-real, composta, nesta primeira versão, pelo seguinte conjunto de funções:

- Leitura do relógio em tempo-real;
- Ativação de temporização (time-out);
- Desativação de temporização;
- Ativação da marcação frequencial de arco de controle;
- Desativação da marcação frequencial de arco de controle;
- Habilitação da marcação de arco de controle via interrupção;
- Desabilitação da marcação de arco de controle via interrupção.

A primitiva de leitura do relógio em tempo-real permite o acesso ao relógio em tempo real interno à máquina (seção 3.2.1.2.2.1.).

As primitivas de ativação e desativação de temporização (time-out) proporcionam mecanismos para o controle do instante de marcação de arcos de entrada do grafo do controle aplicativo (seções 3.2.1.2.2.2. e 3.2.1.2.2.3.).

As primitivas de marcação frequencial, como o próprio nome sugere, oferecem ao usuário a possibilidade de definir a marcação frequencial de arcos de entrada do grafo do controle aplicativo (seções 3.2.1.2.2.4. e 3.2.1.2.2.5.).

As primitivas de tratamento de interrupção permitem ao usuário controlar a influência de eventos externos na dinâmica de execução do aplicativo (seções 3.2.1.2.2.6. e 3.2.1.2.2.7.).

3.2.1.2.2.1. LEITURA DO RELÓGIO EM TEMPO-REAL

LE$HORA (hora)

onde:

hora - parâmetro de saída que recebe o valor da hora atual.

Na figura 3.2.8 são apresentados os domínios do controle e dos dados do modelo GMB* desta primitiva. Apresenta-se a seguir o domínio da interpretação.

DOMÍNIO DA INTERPRETAÇÃO

INTERPRETAÇÃO DOS ARMAZENADORES

ar15 - Relógio em tempo real
ar15::=<HORA>

ar16 - Resultado da leitura do relógio em tempo-real
ar16::=<HORA>

ar17 - Sequenciador de execução
ar17::=<IDENTIFICADOR ARCO DE CONTROLE>

INTERPRETAÇÃO DOS PROCESSADORES

```
INTERPRETAÇÃO p(n5) ;
BEGIN
 /* leia de ar15 a hora marcada pelo relógio de tempo-real */ ;
 /* escreva em ar16 este horário */ ;
 /*  leia  de  ar17 o identificador do arco que deve ser  marcado  a
 seguir (continuação do aplicativo) */ ;
 /* marque este arco com prioridade 1 */
END.
```

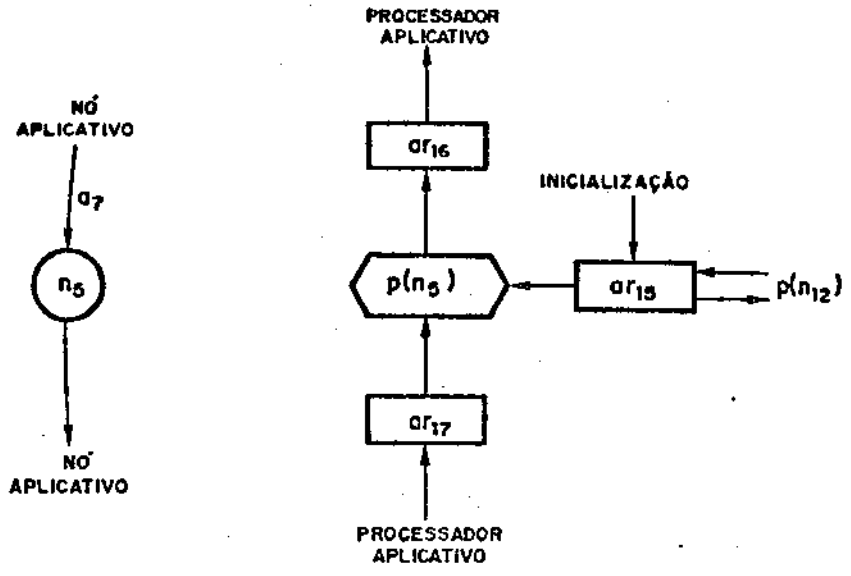


FIG.3.2.8 - MODELO GMB* DA PRIMITIVA DE LEITURA DO RELÓGIO EM TEMPO-REAL: DOMÍNIO DO CONTROLE E DOMÍNIO DOS DADOS

3.2.1.2.2.2. ATIVAÇÃO DE TEMPORIZAÇÃO (TIME-OUT)

ATIVA$ESPERA (arco , prioridade , tempo)
onde:

arco - parâmetro de entrada que especifica o identificador do arco de controle que será marcado ao final do intervalo de tempo especificado pelo usuário através do parâmetro tempo.
prioridade - parâmetro de entrada que especifica a prioridade da marca a ser depositada no arco.
tempo - parâmetro de entrada que especifica o intervalo de tempo de espera para a marcação do arco.

Na figura 3.2.9 são apresentados os domínios do controle e dados do modelo GMB* desta primitiva. Apresenta-se a seguir o domínio da interpretação.

DOMÍNIO DA INTERPRETAÇÃO

INTERPRETAÇÃO DOS ARMAZENADORES
ar18 - Fila de espera para marcação - arcos de controle frequenciais e de temporização
ar18::=(<IDENTIFICADOR ARCO DE CONTROLE><característica>
<PRIORIDADE MARCA><TEMPO DE ESPERA>)
<característica>::=<frequencial>/<DE TEMPORIZAÇÃO>
<frequencial>::=<PERÍODO>
obs: Os elementos deste armazenador estão dispostos em ordem crescente de espera.

ar19 - Pedido de ativação de temporização
ar18::=<IDENTIFICADOR ARCO DE CONTROLE><PRIORIDADE MARCA>
<TEMPO DE ESPERA>

ar20 - Sequenciador de execução
ar20::=<IDENTIFICADOR ARCO DE CONTROLE>

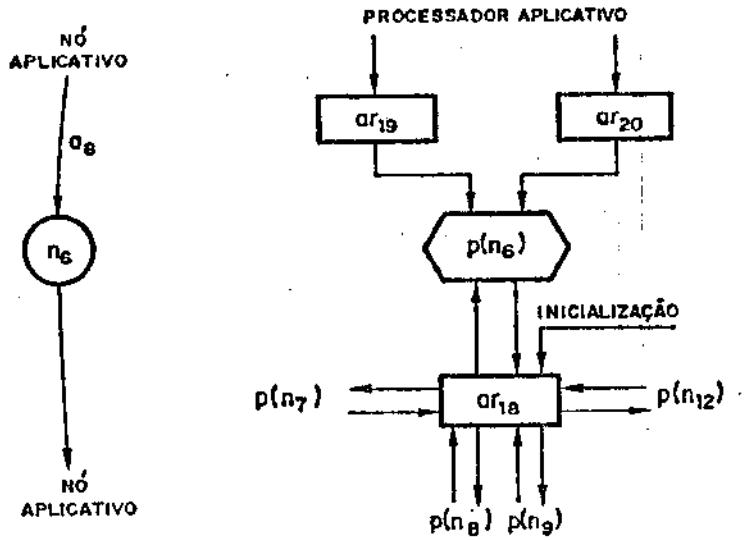


FIG.3.2.9 - MODELO GMB* DA PRIMITIVA DE ATIVAÇÃO DE TEMPORIZAÇÃO: DOMÍNIO DO CONTROLE E DOMÍNIO DOS DADOS

INTERPRETAÇÃO DOS PROCESSADORES

```
INTERPRETAÇÃO p(nB) :
BEGIN
 /* leia de ar19 o pedido de ativação de arco de temporização */;
 /* verifique se fila de espera definida por ar18 está vazia */ ;
 IF /* ar18 está vazio */ THEN
 BEGIN
  /* insira em ar18 a descrição do arco de temporização especificada
  no pedido de ativação */
 END
 ELSE
 BEGIN
  IF /* ar18 não está vazio */ THEN
  BEGIN
   /* leia o tempo de espera do primeiro elemento de ar18 */ :
   IF  /* o tempo de espera do primeiro elemento de ar18 é maior ou
   igual ao tempo de espera especificado no pedido de ativação
   */ THEN
```

```
BEGIN
 /* subtraia o tempo de espera especificado no pedido de ativação
 do tempo de espera do primeiro elemento de ar18  */ ;
 /*  escreva  em  ar18 o resultado desta subtração  como  o  novo
 valor do tempo de espera de seu primeiro elemento */ ;
 /*  insira  como novo primeiro elemento de ar18 a  descrição  do
 arco de temporização especificada no pedido de ativação */
END
ELSE
BEGIN
 IF  /* o tempo de espera do primeiro elemento de ar18 for  menor
 do  que o tempo de espera especificado no pedido de ativação  */
 THEN
 BEGIN
  REPEAT
  BEGIN
   /* verifique a existência de um próximo elemento em ar18 */ ;
   IF /* não existe próximo elemento */ THEN
   BEGIN
    /*  calcule  o  somatório dos tempos de espera  de  todos  os
    elementos de ar18 */ ;
    /*  subtraia  o  valor deste somatório do valor do  tempo  de
    espera definido no pedido de ativação */ ;
    /*  insira  no  final  de  ar18  a  descrição  do  arco  de
    temporização  objeto  da  ativação.  O  valor  do  tempo  de
    espera  a  ser  associado a este novo elemento de  ar18  é  o
    resultado da operação do passo anterior */
   END
   ELSE
   BEGIN
    IF /* existe um próximo elemento */ THEN
    BEGIN
     /* leia o tempo de espera deste elemento */ ;
     /*  some  este tempo de espera ao somatório  dos  tempos  de
     espera dos elementos anteriores */ ;
     IF  /*  o  resultado  desta  soma  for  maior  ou  igual  ao
     tempo  de  espera  especificado  no pedido  de  ativação  */
     THEN
     BEGIN
      /*  subtraia  desta  soma  o valor do tempo  de  espera  do
      último elemento pesquisado */ ;
      /*  subtraia  o valor resultante desta última  operação  do
      tempo de espera especificado no pedido de ativação */ ;
      /*  subtraia  o resultado da operação do passo anterior  do
      tempo de espera do último elemento pesquisado de ar18 */ ;
      /* faça do resultado desta subtração o novo valor do  tempo
      de espera do último elemento pesquisado de ar18 */ ;
      /*  calcule o somatório dos tempos de espera dos  elementos
      antecedentes ao último elemento pesquisado de ar18 */ ;
      /*  subtraia  o  valor deste somatório do tempo  de  espera
      especificado no pedido de ativação */ ;
      /*  insira  entre  o  último elemento pesquisado  e  o  seu
      antecessor  a descrição do arco de temporização  objeto  da
      ativação.  O  tempo  de espera a ser associado a este  novo
      elemento  de  ar18  é  o resultado da  operação  do  passo
      anterior */
     END
    END
```

```
      END
     END
     UNTIL  /*  conseguir  inserir em ar18 a descrição  do  arco  de
     temporização objeto da ativação */
    END
   END
  END
 END ;
 /*  lela  de ar20 o identificador do arco de controle que deve  ser
 marcado a seguir ( continuação do aplicativo) */ ;
 /* marque este arco com prioridade 1 */
END.
```

3.2.1.2.2.3. DESATIVAÇÃO DE TEMPORIZAÇÃO (TIME-OUT)

DESATIVA$ESPERA (arco)

onde:

arco - parâmetro de entrada que especifica o arco de controle cuja temporização de espera deve ser desativada.

Na figura 3.2.10 são apresentados os domínios do controle e dos dados do modelo GMB* desta primitiva. Apresenta-se a seguir o domínio da interpretação.

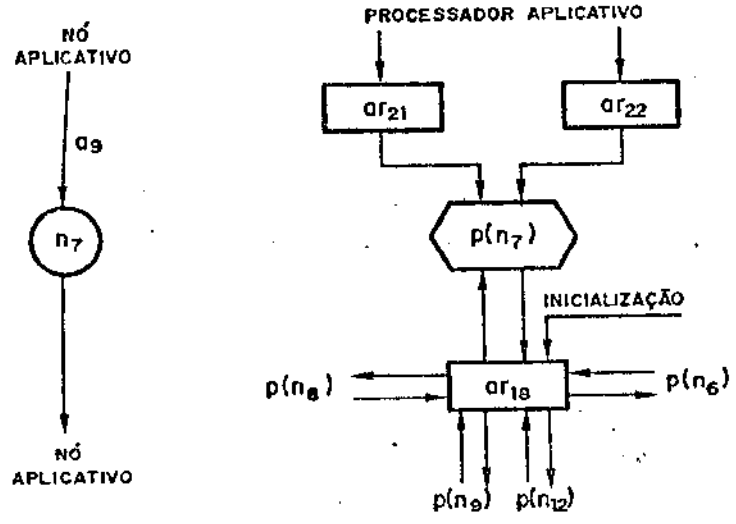


FIG.3.2.10 - MODELO GMB* DA PRIMITIVA DE DESATIVAÇÃO DE TEMPORIZAÇÃO: DOMÍNIO DO CONTROLE E DOMÍNIO DOS DADOS

DOMÍNIO DA INTERPRETAÇÃO

INTERPRETAÇÃO DOS ARMAZENADORES

ar18 - Fila de espera para marcação - arcos de controle frequenciais e de temporização
ar18::=(<IDENTIFICADOR ARCO DE CONTROLE><característica>
<PRIORIDADE MARCA><TEMPO DE ESPERA>)
<característica>::=<frequencial>/<DE TEMPORIZAÇÃO>

<FREQUENCIAL>::=<PERÍODO>
obs: Os elementos deste armazenador estão dispostos em ordem crescente de espera.

ar21 - Pedido de desativação de temporização
ar21::=<IDENTIFICADOR ARCO DE CONTROLE>

ar22 - Sequenciador da execução
ar22::=<IDENTIFICADOR ARCO DE CONTROLE>

INTERPRETAÇÃO DOS PROCESSADORES

```
INTERPRETAÇÃO p(n7) ;
BEGIN
 /* leia de ar21 o identificador do arco de temporização a ser
 desativado */ ;
 /* localize em ar18 este arco */ ;
 /* verifique a existência de um próximo elemento em ar18 */ ;
 IF /* existe um próximo elemento em ar18 */ THEN
 BEGIN
  /* some o tempo de espera do arco de temporização objeto da desa-
  tivação ao tempo de espera do elemento que o sucede em ar8 */ ;
  /* faça do resultado da operação anterior o novo tempo de espera
  do elemento subsequente ao arco de temporização objeto de desati-
  vação */
 END ;
 /* retire de ar18 a descrição do arco de temporização objeto da
 desativação */ ;
 /* leia de ar22 o identificador do arco de controle que deve ser
 marcado a seguir (continuação do aplicativo) */ ;
 /* marque este arco com prioridade 1 */
END.
```

3.2.1.2.2.4. ATIVAÇÃO DA MARCAÇÃO FREQUENCIAL DE ARCO DE CONTROLE

ATIVA$FREQUENCIAL (arco , prioridade , período)

onde:

arco - parâmetro de entrada que especifica o arco de controle a ser marcado na frequência definida pelo usuário.
prioridade - parâmetro de entrada que especifica a prioridade da marca a ser depositada no arco toda vez que o mesmo for marcado.
período - parâmetro de entrada que especifica o intervalo de tempo entre duas marcações consecutivas.

Na figura 3.2.11 são apresentados os domínios do controle e dos dados do modelo GMB* desta primitiva. Apresenta-se a seguir o domínio da interpretação.

DOMÍNIO DA INTERPRETAÇÃO

INTERPRETAÇÃO DOS ARMAZENADORES
ar18 - Fila de espera para marcação - arcos de controle frequenciais e de temporização

ar18::=(<IDENTIFICADOR ARCO DE CONTROLE><característica>
<PRIORIDADE MARCA><TEMPO DE ESPERA>)
<característica>::=<frequencial>/<DE TEMPORIZAÇÃO>
<frequencial>::=<PERÍODO>
obs: Os elementos deste armazenador estão dispostos em ordem crescente de espera.

ar23 - Pedido de ativação de marcação frequencial
ar23::=<IDENTIFICADOR ARCO DE CONTROLE><PRIORIDADE MARCA>
<PERÍODO>

ar24 - Sequenciador da execução
ar24::=<IDENTIFICADOR ARCO DE CONTROLE>

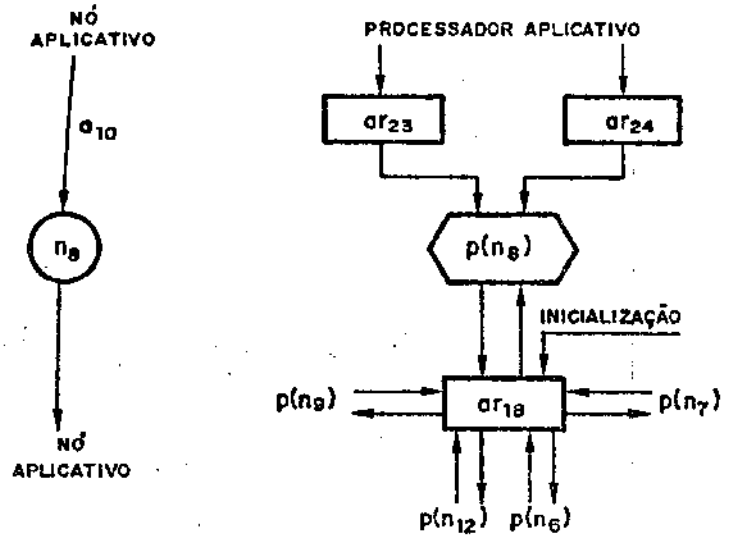


FIG.3.2.11 - MODELO GMB* DA PRIMITIVA DE ATIVAÇÃO DA MARCAÇÃO FREQUENCIAL DE ARCO DE CONTROLE: DOMÍNIO DO CONTROLE E DOMÍNIO DOS DADOS

INTERPRETAÇÃO DOS PROCESSADORES

```
INTERPRETAÇÃO p(n8) :
BEGIN
 /* leia de ar23 o pedido de ativação de marcação frequencial */ ;
 /* verifique se fila de espera definida por ar18 está vazia */ :
 IF /* ar18 está vazio */ THEN
 BEGIN
  /*  insira em ar18 a descrição do arco frequencial objeto da ati-
  vação.  O valor do tempo de espera a ser associado a este elemento
  é o valor do período especificado no pedido de ativação */
 END
 ELSE
 BEGIN
  IF /* ar18 não está vazio */ THEN
  BEGIN
   /* leia o tempo de espera do primeiro elemento de ar18 */ :
   IF /* o tempo de espera do primeiro elemento de ar18 for maior ou
   igual ao período especificado no pedido de ativação */ THEN
   BEGIN
    /*  subtraia o período  especificado no  pedido  de ativação  do
    tempo de espera do primeiro elemento de  ar18   */ ;
```

```
 /* escreva o resultado desta subtração em a18 como o novo
 valor do tempo de espera de seu primeiro elemento */ ;
 /* insira como novo primeiro elemento ar18 a descrição do arco
 frequencial objeto da ativação. O valor do tempo de espera a ser
 associado a este elemento é o valor do período especificado no
 pedido de ativação */
END
ELSE
BEGIN
 IF /* o tempo de espera do primeiro elemento de ar18 for menor
 do que o período especificado no pedido de ativação */ THEN
 BEGIN
  REPEAT
  BEGIN
   /* verifique a existência de um próximo elemento na fila de
   espera definida por ar18 */ ;
   IF /* não existe um próximo elemento */ THEN
   BEGIN
    /* calcule o somatório dos tempos de espera de todos os
    elementos de ar18 */ ;
    /* subtraia o valor deste somatório do valor do período
    especificado no pedido de ativação */ ;
    /* insira no final de ar18 a descrição do arco frequencial
    objeto da ativação. O tempo de espera deste novo elemento de
    ar18 é o resultado da operação do passo anterior */
   END
   ELSE
   BEGIN
    IF /* existe um próximo elemento */ THEN
    BEGIN
     /* leia o tempo de espera deste próximo elemento */ ;
     /* some este tempo ao somatório dos tempos de espera dos
     elementos anteriores */ ;
     IF /* o resultado deste somatório for maior ou igual ao
     valor do período definido no pedido de ativação */ THEN
     BEGIN
      /* subtraia deste somatório o valor do tempo de espera do
      último elemento pesquisado de ar18 */ ;
      /* subtraia o valor resultante desta última operação do
      valor do período definido no pedido de ativação */ ;
      /* subtraia o resultado da operação do passo anterior do
      valor do tempo de espera do último elemento pesquisado de
      ar18 */ ;
      /* faça o resultado desta subtração o novo valor do tempo
      de espera do último elemento pesquisado de ar18 */ ;
      /* calcule o somatório dos tempos de espera dos elementos
      antecedentes ao último elemento pesquisado de ar18 */ ;
      /* subtraia o valor deste somatório do valor do período
      definido no pedido de ativação */ ;
      /* insira em ar18, entre o último elemento pesquisado e o
      seu antecessor, a descrição do arco frequencial objeto da
      ativação. O valor do tempo de espera a ser associado a este
      novo elemento de ar18 é o resultado da operação do passo
      anterior */
     END
    END
   END
  END
```

```
    UNTIL /* conseguir inserir em ar18 a descrição do arco frequen-
    cial objeto da ativação */
   END
  END
 END
END ;
/* leia de ar24 o identificador do arco de controle que deve
ser marcado a seguir ( continuação do aplicativo) */ ;
/* marque este arco com prioridade 1 */
END.
```

3.2.1.2.2.5. DESATIVAÇÃO DA MARCAÇÃO FREQUENCIAL DE ARCO DE CONTROLE

DESATIVA$FREQUENCIAL (arco)

onde:

arco - parâmetro de entrada que especifica o identificador do arco de controle cuja marcação frequencial deve ser imediatamente desativada.

Na figura 3.2.12 são apresentados os domínios do controle e dos dados do modelo GMB* desta primitiva. Apresenta-se a seguir o domínio da interpretação.

DOMÍNIO DA INTERPRETAÇÃO

INTERPRETAÇÃO DOS ARMAZENADORES

ar18 - Fila de espera para marcação - arcos de controle frequenciais e de temporização
ar18::=(<IDENTIFICADOR ARCO DE CONTROLE><característica>
<PRIORIDADE MARCA><TEMPO DE ESPERA>)
<característica>::=<frequencial>/<DE TEMPORIZAÇÃO>
<frequencial>::=<PERÍODO>
obs: Os elementos deste armazenador estão dispostos em ordem crescente de tempo de espera.

ar25 - Pedido de desativação de marcação frequencial
ar25::=<IDENTIFICADOR ARCO>

ar26 - Sequenciador de execução
ar26::=<IDENTIFICADOR ARCO DE CONTROLE>

INTERPRETAÇÃO DOS PROCESSADORES

```
INTERPRETAÇÃO p(n9) :
BEGIN
 /* leia de ar25 o pedido de desativação de arco frequencial */ ;
 /* localize em ar18 o arco frequencial objeto da desativação */ ;
 /* verifique a existência de um próximo elemento em ar18 */ ;
 IF /* existe um próximo elemento */ THEN
 BEGIN
  /* some o tempo de espera do arco frequencial objeto da desati-
  vação ao tempo de espera do elemento que o sucede em ar18 */ ;
  /* faça do resultado desta soma o novo valor de tempo de espera do
```

```
  elemento subsequente ao arco frequencial objeto da desativação */
 END ;
 /* retire de ar18 a descrição do arco frequencial objeto da desati-
 vação */ ;
 /*  leia de ar26 o identificador do arco de controle que  deve  ser
 marcado a seguir (continuação do aplicativo) */ ;
 /* marque este arco com prioridade 1 */
END.
```

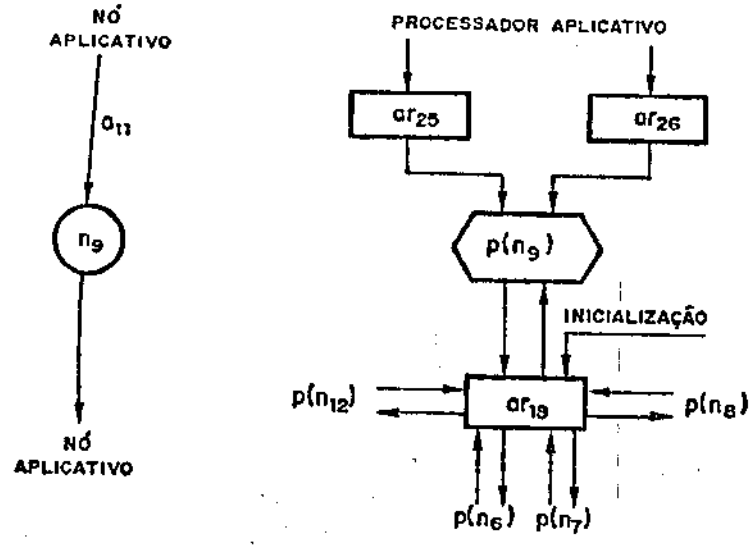


FIG.3.2.12 - MODELO GMB* DA PRIMITIVA DE DESATIVAÇÃO DA MARCAÇÃO FREQUENCIAL DE ARCO DE CONTROLE: DOMÍNIO DO CONTROLE E DOMÍNIO DOS DADOS

3.2.1.2.2.6. HABILITAÇÃO DA MARCAÇÃO DE ARCO DE CONTROLE VIA INTERRUPÇÃO

HABILITA$INTERRUPÇÃO (interrupção , arco , prioridade)

onde :

interrupção - parâmetro de entrada que identifica o canal de interrupção que deve ser habilitado.
arco - parâmetro de entrada que especifica o identificador do arco de controle que será marcado ao ser atendido um pedido de interrupção no canal especificado pelo parâmetro interrupção.
prioridade - parâmetro de entrada que especifica a prioridade da marca a ser depositada no arco quando este for marcado.

Na figura 3.2.13 são apresentados os domínios do controle e dos dados do modelo GMB* desta primitiva. Apresenta-se a seguir o domínio da interpretação.

DOMÍNIO DA INTERPRETAÇÃO

INTERPRETAÇÃO DOS ARMAZENADORES

ar27 - Máscara de interrupção

ar27::=<MÁSCARA DE INTERRUPÇÃO>

ar28 - Pedido de habilitação de interrupção
ar28::=<IDENTIFICADOR DO CANAL DE INTERRUPÇÃO>
<IDENTIFICADOR ARCO DE CONTROLE><PRIORIDADE MARCA>

ar29 - Sequenciador de execução
ar29::=<IDENTIFICADOR ARCO DE CONTROLE>

ar30/CIi (i=1,2...n) - i-ésimo componente do vetor de interrupção / canal de interrupção i
ar30/CIi::=<IDENTIFICADOR ARCO DE CONTROLE><PRIORIDADE MARCA>

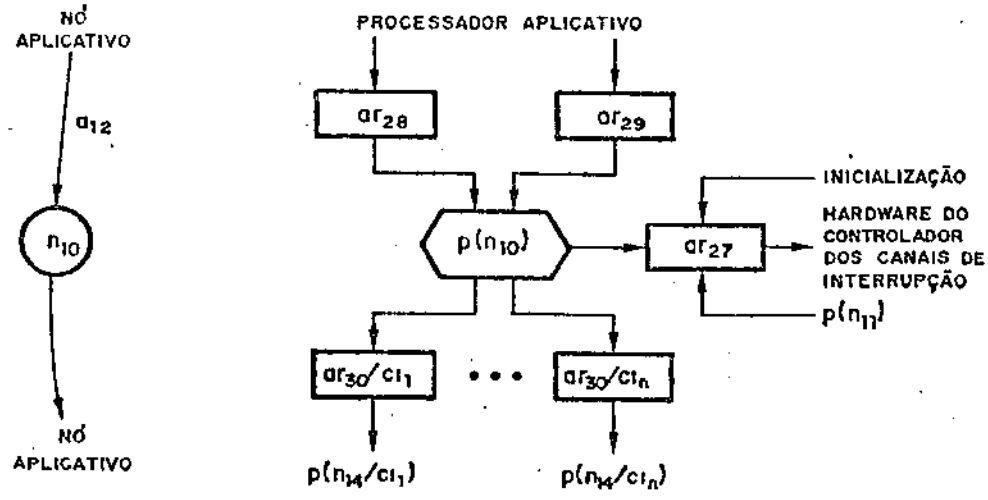


FIG.3.2.13 - MODELO GMB* DA PRIMITIVA DE HABILITAÇÃO DA MARCAÇÃO DE ARCO DE CONTROLE VIA INTERRUPÇÃO: DOMÍNIO DO CONTROLE E DOMÍNIO DOS DADOS

INTERPRETAÇÃO DOS PROCESSADORES

INTERPRETAÇÃO **p(n10)** ;
BEGIN
/* leia de ar28 o pedido de habilitação de interrupção */ ;
/* escreva no vetor de interrupção, no armazenador associado ao canal cuja habilitação foi solicitada (ar30/CIi), o identificador do arco e a prioridade da marca definidos no pedido de habilitação */ ;
/* escreva em ar27 a máscara que habilita o canal de interrupção especificado no pedido */ ;
/* leia de ar29 o identificador do arco que deve ser marcado a seguir (continuação do aplicativo) */ ;
/* marque este arco com prioridade 1 */
END.

3.2.1.2.2.7. DESABILITAÇÃO DA MARCAÇÃO DE ARCO DE CONTROLE VIA INTERRUPÇÃO

DESABILITA$INTERRUPÇÃO (interrupção)

onde :

Interrupção - parâmetro de entrada que identifica o canal de interrupção que deve ser desabilitado.

Na figura 3.2.14 são apresentados os domínios do controle e dos dados do modelo GMB* desta primitiva. Apresenta-se a seguir o domínio da interpretação.

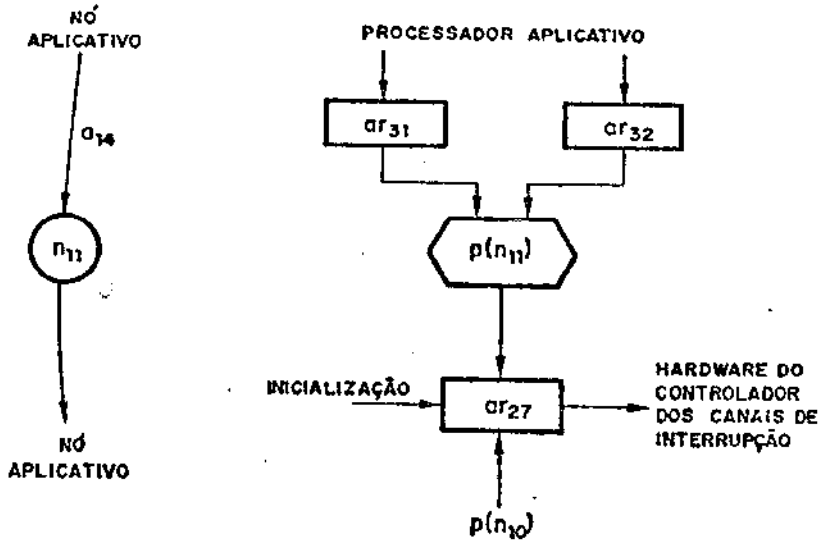


FIG.3.2.14 - MODELO GMB* DA PRIMITIVA DE DESABILITAÇÃO DA MARCAÇÃO DE ARCO DE CONTROLE VIA INTERRUPÇÃO: DOMÍNIO DO CONTROLE E DOMÍNIO DOS DADOS

DOMÍNIO DA INTERPRETAÇÃO

INTERPRETAÇÃO DOS ARMAZENADORES

ar27 - Máscara de Interrupção
ar27::=<MÁSCARA DE INTERRUPÇÃO>

ar31 - Pedido de desabilitação de interrupção
ar31::=<IDENTIFICADOR DO CANAL DE INTERRUPÇÃO>

ar32 - Sequenciador de execução
ar32::=<IDENTIFICADOR ARCO DE CONTROLE>

INTERPRETAÇÃO DOS PROCESSADORES

```
INTERPRETAÇÃO p(n11) ;
BEGIN
 /* leia de ar31 o pedido de desabilitação de interrupção */ ;
 /*  escreva em ar27 a máscara que desabilita o canal de interrupção
 constante do pedido */ ;
 /*  leia  de ar32 o identificador do arco que deve  ser  marcado  a
 seguir (continuação do aplicativo) */ ;
 /* marque este arco com prioridade 1 */
END.
```

3.2.1.3. RELÓGIO EM TEMPO-REAL

A principal função deste elemento é garantir a existência de um relógio em tempo-real interno à máquina. Para tanto, este elemento deve tratar, sem exceção, todas as solicitações de atualização de horário geradas, via interrupção, por uma base de tempo externa à máquina. Este relógio, por sua vez, dita a referência para o atendimento dos pedidos de temporização (time-out) do serviço de comunicação, assim como, serve de suporte para a implementação de um sub-conjunto da biblioteca de primitivas voltadas ao processamento em tempo-real, quais sejam: primitiva de leitura do relógio, primitivas de temporização e primitivas de marcação frequencial.

Na figura 3.2.15 são apresentados os domínio do controle e dos dados do modelo GMB* deste elemento. Apresenta-se a seguir o domínio da interpretação.

DOMÍNIO DA INTERPRETAÇÃO

INTERPRETAÇÃO DOS ARMAZENADORES

ar1 - Estado dos arcos de controle
ar1::=(<IDENTIFICADOR ARCO DE CONTROLE>
<prioridade marcas residentes>)
<prioridade marcas residentes>::=(<PRIORIDADE MARCA>)

ar15 - Relógio em tempo-real
ar15::=<HORA>

ar18 - Fila de espera para marcação - arcos de controle frequenciais e de temporização
ar18::=(<IDENTIFICADOR ARCO DE CONTROLE><característica>
<PRIORIDADE MARCA><TEMPO DE ESPERA>)
<característica>::=<frequencial>/<DE TEMPORIZAÇÃO>
<frequencial>::=<PERÍODO>
obs: Os elementos deste armazenador estão dispostos em ordem crescente de espera.

ar33 - Fila de espera dos pedidos de temporização do serviço de comunicação (time-out)
ar33::=(<CANAL DE COMUNICAÇÃO><IDENTIFICADOR ARCO DE CONTROLE>
<TEMPO DE ESPERA>)
obs: Os elementos deste armazenador estão dispostos em ordem crescente de espera.

ar34 - Acumulador temporário do relógio
ar34::=<HORA>

ar35 - Indicador do estado do tratador dos pedidos de temporização do serviço de comunicação
ar35::=<DORMENTE>/<ACORDADO>

ar36/CCI - Sequenciador de execução / canal de comunicação I
ar36/CCI::=<IDENTIFICADOR ARCO DE CONTROLE>/<VAZIO>

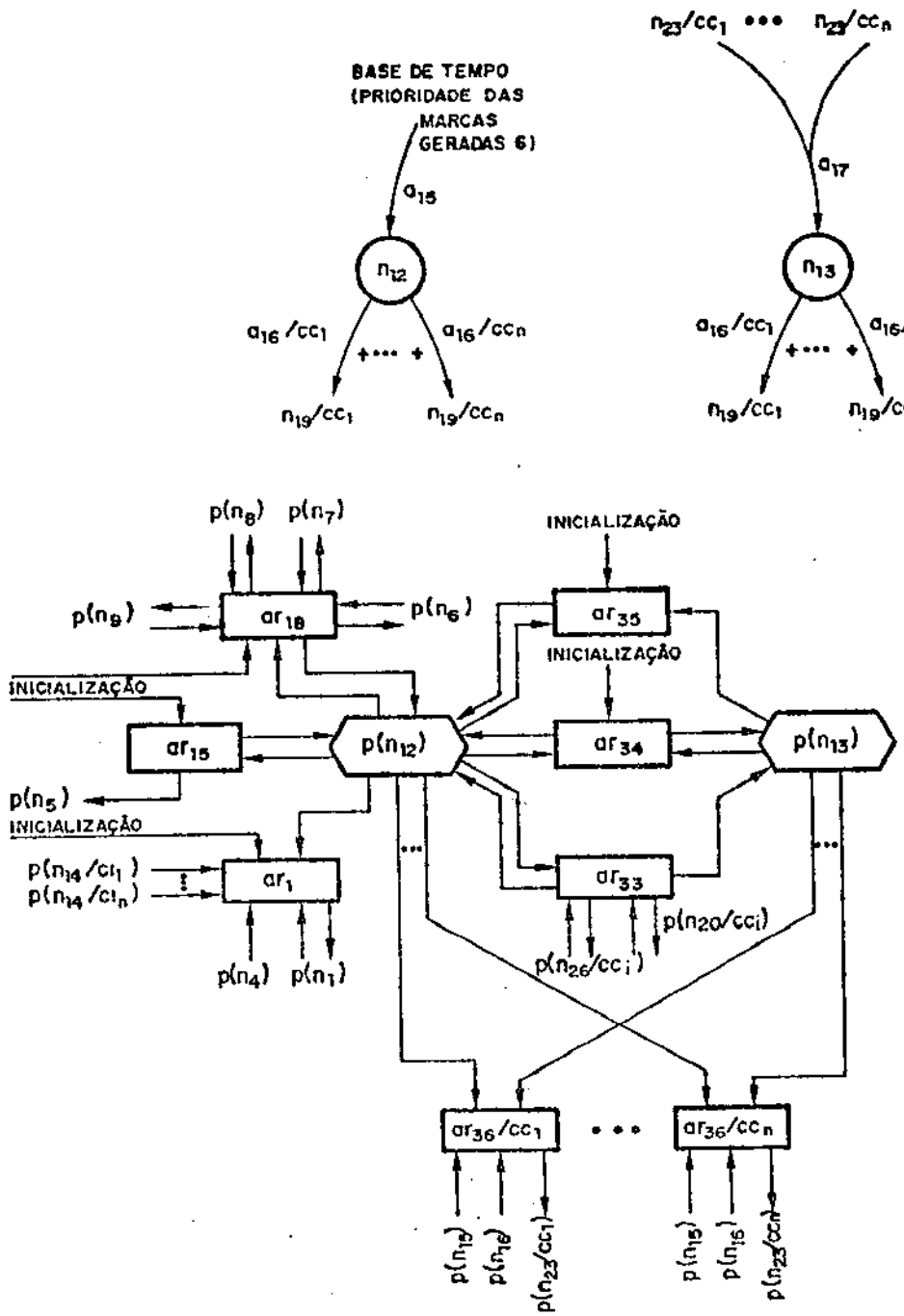


FIG.3.2.15 - MODELO GMB* DO RELÓGIO EM TEMPO-REAL: DOMÍNIO DO CONTROLE E DOMÍNIO DOS DADOS

INTERPRETAÇÃO DOS PROCESSADORES

```
INTERPRETAÇÃO p(n12) ;
BEGIN
 /* Incremente de uma unidade de tempo o relógio de tempo-real
 definido em ar15 */ ;
 /* verifique se a fila de espera definida em ar18 está vazia */ ;
 IF /* ar18 não está vazio */ THEN
 BEGIN
  /* leia o tempo de espera associado ao primeiro elemento de
```

```
ar18 */ ;
/* decremente de um unidade este tempo de espera */ ;
/* escreva o resultado da operação do passo anterior em ar18  como
o novo valor do tempo de espera de seu primeiro elemento */ :
IF /* o resultado da última operação é igual a zero */ THEN
BEGIN
 REPEAT
 BEGIN
  /*  leia  e  retire o primeiro elemento de  ar18  (descrição  de
  arco a espera de marcação) */ ;
  /* escreva em ar1 a marca solicitada por este elemento */ :
  IF /* este elemento é arco frequencial */ THEN
  BEGIN
   /*  verifique  se a fila de espera  definida  em   ar18   ficou
   vazia */ ;
   IF /* ar18 está vazio */ THEN
   BEGIN
    /*  insira  novamente  em  ar18,  como seu  primeiro  e  único
    elemento, a descrição do arco frequencial. O tempo de espera a
    ser associado a este elemento é o período especificado na  sua
    descrição */
   END
   ELSE
   BEGIN
    IF /* ar18 não está vazio */ THEN
    BEGIN
     /* leia o tempo de espera do primeiro elemento de ar18 */ ;
     IF  /*  o  tempo de espera do primeiro elemento de  ar18  for
     maior  ou igual ao período especificado na descrição do  arco
     frequencial */  THEN
     BEGIN
      /*  subtraia o tempo de espera do primeiro elemento de  ar18
      do período especificado na descrição do arco frequencial */;
      /*  escreva o resultado desta subtração em ar18 como o  novo
      valor do tempo de espera de seu primeiro elemento */ ;
      /* insira como novo primeiro elemento de ar18 a descrição do
      arco frequencial.  O tempo de espera a ser associado a  este
      elemento  é  o valor  do  período  especificado na sua  des-
      crição */
     END
     ELSE
     BEGIN
      IF  /*  o tempo de espera do primeiro elemento de  ar18  for
      menor  que  o  período  especificado na  descrição  do  arco
      frequencial */ THEN
      BEGIN
       REPEAT
       BEGIN
        /*  verifique a existência de um próximo elemento na  fila
        de espera definida em ar18 */ ;
        IF /* não existe próximo elemento */ THEN
        BEGIN
         /*  calcule o somatório do tempos de espera de  todos  os
         elementos de ar18 */ ;
         /* subtraia o valor deste somatório do período  especifi-
         cado na descrição do arco frequencial */ ;
         /*   insira  no  final  de  ar18  a  descrição  do   arco
         frequencial. O valor do tempo de espera a ser associado a
```

```
            este   elemento   é   o   resultado   da   operação do   passo
            anterior */
           END
           ELSE
           BEGIN
            IF /* existe um próximo elemento */ THEN
            BEGIN
             /* leia o tempo de espera deste próximo elemento */ ;
             /* some este tempo de espera ao somatório dos tempos  de
             espera dos elementos anteriores */ ;
             IF  /* o resultado deste somatório for maior ou igual ao
             período especificado na descrição do arco frequencial */
             THEN
             BEGIN
              /*  subtraia  deste  somatório  o  tempo  de  espera  do
              último elemento pesquisado */ ;
              /* subtraia o valor resultante desta última operação do
              período  especificado  na  descrição  do  arco   frequen-
              cial */ ;
              /*  subtraia o resultado da operação do passo  anterior
              do   tempo de espera do último elemento  pesquisado  de
              ar18 */ ;
              /*  faça  o resultado desta subtração o novo  valor  do
              tempo  de  espera  do  último  elemento  pesquisado  de
              ar18 */ ;
              /*  calcule  o  somatório  dos  tempos  de  espera  dos
              elementos antecedentes ao último elemento pesquisado de
              ar18 */ ;
              /*   subtraia  o  valor  deste  somatório  do   período
              especificado na descrição do arco frequencial */ ;
              /*  insira em ar18 entre o último elemento pesquisado e
              seu antecessor a descrição do arco frequencial. O valor
              do tempo de espera a ser associado a este novo elemento
              de ar18 é o resultado da operação do passo anterior */
             END
            END
           END
          END
          UNTIL  /* conseguir inserir  a  descrição  do  arco  frequen-
          cial */
         END
        END
       END
      END
     END
    END
   UNTIL  /*  ar18 ficar vazio */ OU /* o valor do tempo  de  espera
   do primeiro elemento de ar18 ser diferente de zero */
  END
END ;
/*   leia  o  indicador  de  estado  do  tratador  dos  pedidos  de
temporização do serviço de comunicação definido em ar35 */ ;
IF /* o indicador sinaliza estado dormente */ THEN
BEGIN
 /* verifique se a fila de espera definida em ar33 está vazia */ ;
 IF /* ar33 não está vazio */ THEN
 BEGIN
  /*  leia o tempo  de  espera  associado  ao primeiro elemento  de
```

```
    ar33 */ ;
    /* decremente de uma unidade este tempo de espera */ ;
    /* escreva o resultado da operação do passo anterior em ar33 como
    o novo valor do tempo de espera de seu primeiro
    elemento */ ;
    IF /* o resultado da operação anterior é igual a zero */ THEN
    BEGIN
     /* leia e retire o primeiro elemento de ar33 (pedido de
     temporização do serviço de comunicação) */ ;
     /* coloque o identificador do arco de controle a17
     no armazenador ar36 associado ao canal de comunicação
     especificado no pedido de temporização (ar36/CCI) */ ;
     /* mude o indicador do estado do tratador de pedidos de
     temporização do serviço de comunicação (ar35) para estado
     acordado */ ;
     /* marque o arco de controle especificado no pedido de
     temporização (a16/CCI) com prioridade 3 */
    END
   END
  END
  ELSE
  BEGIN
   IF /* o indicador sinaliza estado acordado */ THEN
   BEGIN
    /* incremente de uma unidade de tempo o acumulador temporário do
    relógio definido em ar34 */
   END
  END
END.

INTERPRETAÇÃO p(n13) ;
BEGIN
 /* verifique se a fila de espera definida em ar33 está vazia */ ;
 IF /* ar33 está vazio */ THEN
 BEGIN
  /* zere o conteúdo de ar34 */ ;
  /* mude o indicador do estado do tratador dos pedidos de tempori-
  zação do serviço de comunicação (ar35) para estado dormente */
 END
 ELSE
 BEGIN
  IF /* ar33 não está vazio */ THEN
  BEGIN
   /* leia o tempo de espera do primeiro elemento de ar33 */ ;
   IF  /* o tempo de espera do primeiro elemento de ar33 é igual a
   zero */ THEN
   BEGIN
    /* leia e retire o primeiro elemento de ar33 (pedido de tempori-
    zação do serviço de comunicação) */ ;
    /* coloque o identificador do arco de controle a17 no
    armazenador ar36 associado ao canal de comunicação especificado
    no pedido de temporização (ar36/CCI) */ ;
    /* marque o arco de controle especificado no pedido de
    temporização (a16/CCI) com prioridade 3 */
   END
   ELSE
   BEGIN
    IF /* o tempo de espera do primeiro elemento de ar33 é diferente
```

```
    de zero */ THEN
    BEGIN
     /* leia o conteúdo de ar34 */ ;
     /* subtraia este conteúdo do tempo de espera do primeiro
     elemento de ar33 */ ;
     IF /* o resultado da operação anterior é menor ou igual a zero
     */ THEN
     BEGIN
      /* subtraia o tempo de espera do primeiro elemento de ar33 do
      conteúdo de ar34 */ ;
      /* escreva o resultado desta subtração em ar34 como o seu novo
      conteúdo */ ;
      /* leia e retire o primeiro elemento de ar33 (pedido de
      temporização do serviço de comunicação) */ ;
      /* coloque o identificador do arco de controle a17 no
      armazenador ar36 associado ao canal de comunicação definido no
      pedido de temporização (ar36/CCI) */ ;
      /* marque o arco de controle especificado no pedido de
      temporização (a16/CCI) com prioridade 3 */
     END
     ELSE
     BEGIN
      IF  /*  o resultado da operação anterior é maior que  zero  */
      THEN
      BEGIN
       /* subtraia o conteúdo de ar34 do tempo de espera do primeiro
       elemento de ar33 */ ;
       /*  escreva o resultado desta subtração em ar33 como  o  novo
       valor do tempo de espera de seu primeiro elemento */ ;
       /* zere o conteúdo de ar34 */ ;
       /*  mude  o  indicador do estado do tratador  de  pedidos  de
       temporização  do  serviço de comunicação (ar35)  para  estado
       dormente */
      END
     END
    END
   END
  END
 END
END.
```

3.2.1.4. TRATADOR DOS CANAIS DE INTERRUPÇÃO DO USUÁRIO

Este elemento tem por função converter os pedidos dos canais de interrupção disponíveis ao usuário em marcas do grafo do controle aplicativo. A definição do arco a ser marcado e da prioridade da marca a ser gerada é feita, a critério do usuário, com o auxílio da primitiva de habilitação da marcação de arco de controle via interrupção (seções 3.2.1.2.2.6. e 3.2.1.2.2.7.).

Na figura 3.2.16 são apresentados os domínio do controle e dos dados do modelo GMB* deste elemento. Apresenta-se a seguir o domínio da interpretação.

DOMÍNIO DA INTERPRETAÇÃO

INTERPRETAÇÃO DOS ARMAZENADORES

ar1 - Estado dos arcos de controle do aplicativo
ar1::=(<IDENTIFICADOR ARCO DE CONTROLE>
<prioridade marcas residentes>)
<prioridade marcas residentes>::=(<PRIORIDADE MARCA>)

ar30/CIi (i=1,2...n) - i-ésimo componente do vetor de interrupção / canal de interrupção i
ar30/CIi::=<IDENTIFICADOR ARCO DE CONTROLE><PRIORIDADE MARCA>

INTERPRETAÇÃO DOS PROCESSADORES

INTERPRETAÇÃO **p(n14/CIi)** ;
BEGIN
/* leia do armazenador associado ao canal de interrupcão i (ar30/CIi) o arco a ser marcado e a prioridade da marca a ser gerada */ ;
/* escreva em ar1 a marca solicitada */
END.

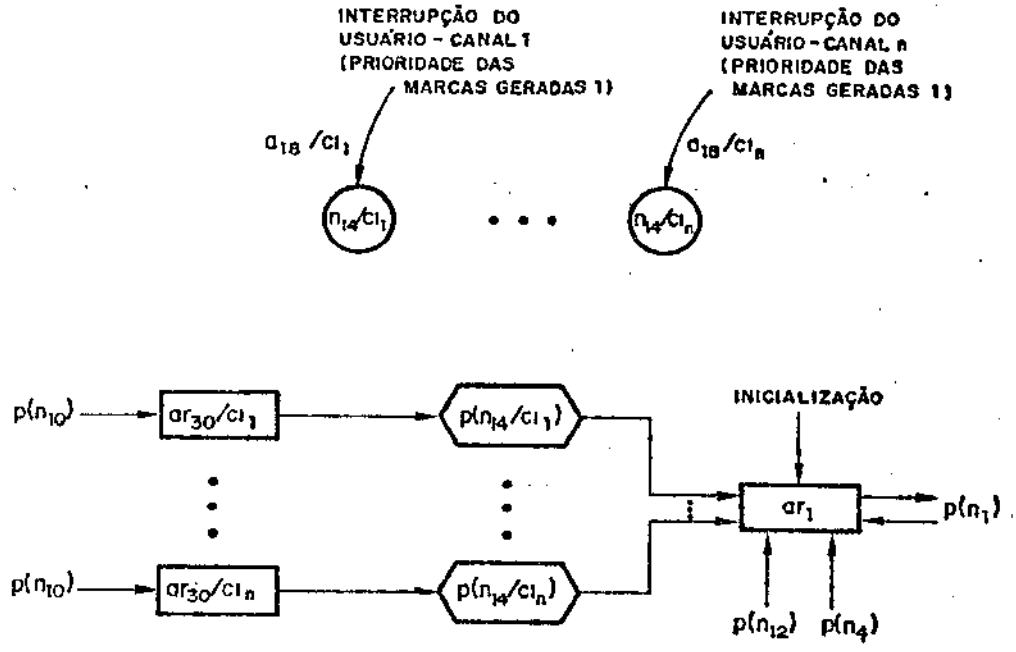


FIG.3.2.16 - MODELO GMB* DA ESTRUTURA DE TRATAMENTO DOS CANAIS DE INTERRUPÇÃO DO USUÁRIO: DOMÍNIO DO CONTROLE E DOMÍNIO DOS DADOS

3.2.2. SERVIÇO DE COMUNICAÇÃO DA MÁQUINA GMB* UNI-PROCESSADORA

Para emprestar à máquina GMB* uni-processadora a característica de módulo básico de construção da máquina GMB* distribuída, embutiu-se, ainda, em sua estrutura, um serviço de comunicação capaz de suportar a troca de marcas e dados entre os diversos módulos de uma eventual configuração distribuída. Destaque-se que a retirada deste serviço de comunicação da estrutura proposta não prejudica a

capacidade para o processamento GMB* da máquina uni-processadora (figura 3.2.1). Sem este serviço de comunicação, entretanto, tem-se o rígido comprometimento do sistema às limitações impostas por uma estrutura centralizada.

O serviço de comunicação a ser oferecido é composto por quatro níveis sobrepostos, como apresentado na figura 3.2.17.

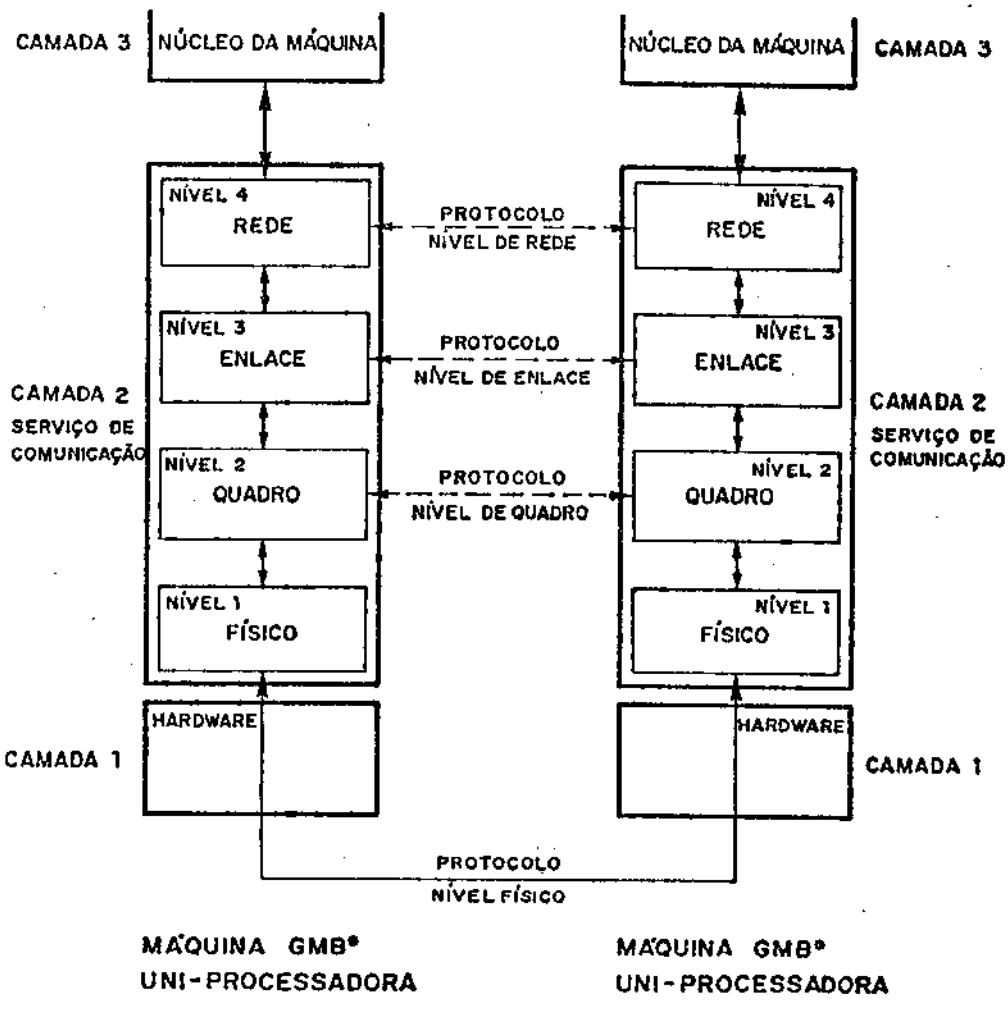


FIG.3.2.17 - ORGANIZAÇÃO DO SERVIÇO DE COMUNICAÇÃO

3.2.2.1. NÍVEL FÍSICO

O nível físico ou nível 1 do serviço de comunicação é responsável pela transmissão física dos bits componentes das mensagens através do canal de comunicação. Tal comunicação será série, assíncrona, byte-a-byte por interrupção, full-duplex, em níveis elétricos compatíveis com o padrão RS 232-C, a uma taxa de 19.200 bps. Em particular, adotou-se a pastilha USART 8251A como interface física de transmissão/recepção. O esquema a ser utilizado na interconexão física entre duas máquinas GMB* uni-processadoras é apresentado na figura 3.2.18. Para maior detalhamento consultar /Intel 81/, /De Martino 1 84/ /De Martino 2 84/.

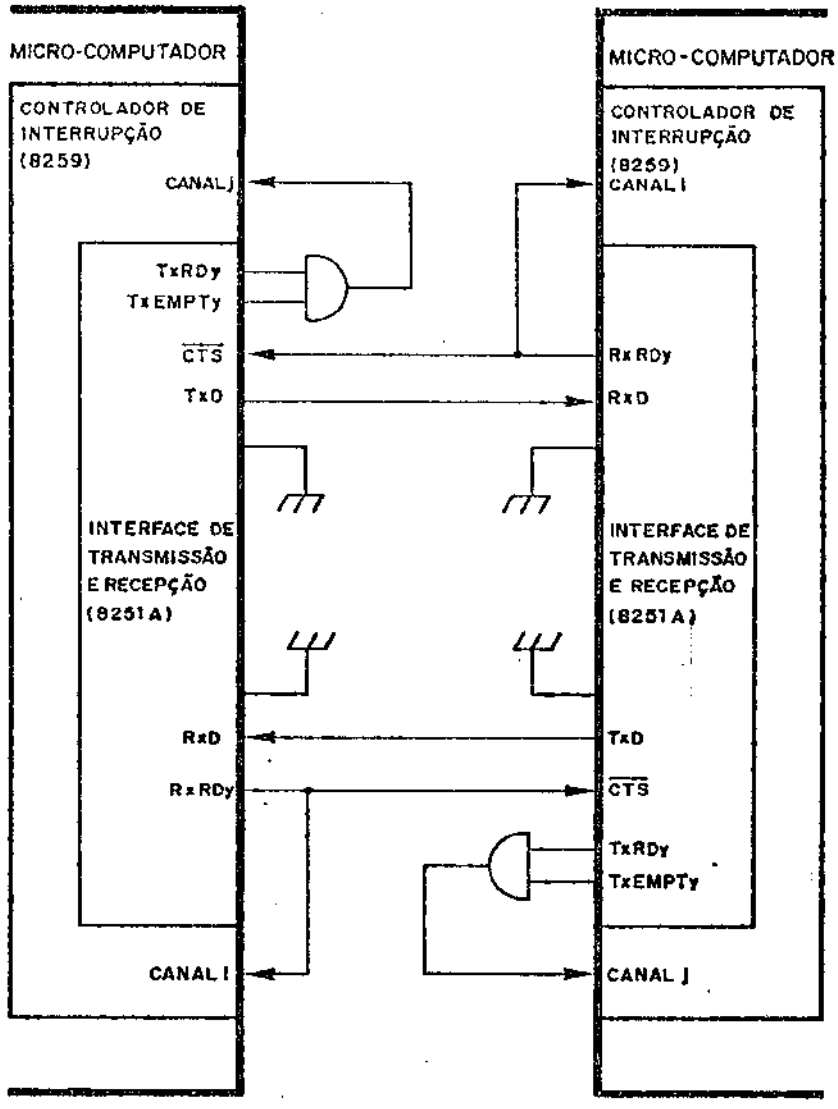


FIG.3.2.18 - INTERCONEXÃO FÍSICA DAS MÁQUINAS GMB* UNI-PROCESSADORAS

Na figura 3.2.19 são apresentados os domínios do controle e dos dados do modelo GMB* do nível físico. Apresenta-se a seguir o domínio da interpretação. Convém salientar que tal modelo descreve apenas o software de acesso e gerenciamento do hardware do canal de comunicação.

DOMÍNIO DA INTERPRETAÇÃO

INTERPRETAÇÃO DOS ARMAZENADORES

ar36/CCI - Sequenciador de execução / canal de comunicação I
ar36/CCI::=<IDENTIFICADOR ARCO DE CONTROLE>

ar46/CCI - Armazenador de pacote a ser transmitido / canal de comunicação I
ar46/CCI::=<TAMANHO DO PACOTE><pacote>

<pacote>::=<cabeçalho><CORPO DO PACOTE><CHECK-SUM><cauda>
<cabeçalho>::=<DLE><STX>
<cauda>::=<DLE><ETX>
ar47/CCI - Armazenador de entrada da recepção / canal de comunicação I
ar47/CCI::=<BYTE>

ar49/CCI - Armazenador de espera para a transmissão física / canal de comunicação I
ar49/CCI::=(<BYTE>)

ar50/CCI - Armazenador da interface de recepção / canal de comunicação I
ar50/CCI::=<BYTE>

ar52/CCI - Armazenador da interface de transmissão / canal de comunicação I
ar52/CCI::=<BYTE>

ar57/CCI - Contador dos bytes que faltam para serem transmitidos / canal de comunicação I
ar57/CCI::=<TAMANHO>

INTERPRETAÇÃO DOS PROCESSADORES

```
INTERPRETAÇÃO p(n23/CCI) ;
BEGIN
 /*  leia o primeiro byte de ar46/CCI(tamanho do pacote a ser trans-
 mitido) */ ;
 /*  leia  de ar46/CCI,  a partir do segundo,  tantos  bytes  quanto
 especificado  pelo tamanho do pacote a ser transmitido (este  bloco
 de dados é o pacote a ser transmitido) */ ;
 /*  escreva  em  ar52/CCI o primeiro byte  do  pacote  (transmissão
 primeiro byte) */ ;
 /* escreva em ar49/CCI os bytes restantes do pacote */ ;
 /* escreva em ar57/CCI o tamanho do pacote - 1 (número de bytes que
 faltam para serem transmitidos */ ;
 /* leia o sequenciador de execução definido em ar36/CCI */ ;
 IF /* sequenciador especifica arco de controle */ THEN
 BEGIN
  IF /* o arco especificado é o a17 */ THEN
  BEGIN
   /* marque a17 com prioridade 5 */
  END
  ELSE
  BEGIN
   IF /* o arco especificado não é o a17 */ THEN
   BEGIN
    /* marque o arco especificado com prioridade 1 */
   END
  END
 END ;
 /* marque a27/CCI com prioridade 1 (transmissão física do byte) */
END.
```

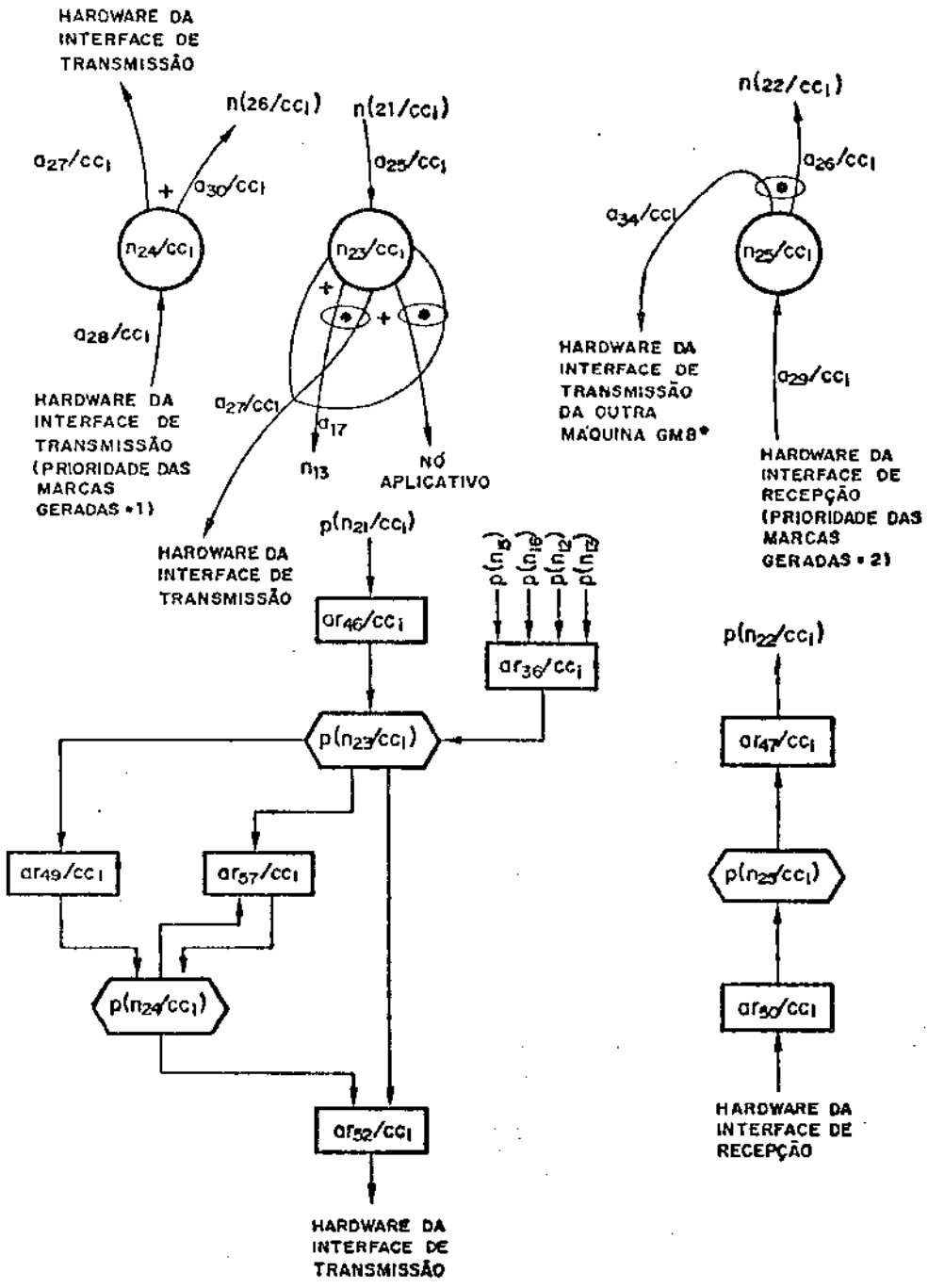


FIG.3.2.19 - MODELO GM8* DO NÍVEL FÍSICO DO SERVIÇO DE COMUNICAÇÃO: DOMÍNIO DO CONTROLE E DOMÍNIO DOS DADOS

```
INTERPRETAÇÃO p(n24/CC1) ;
BEGIN
 /* leia de ar57/CC1 quantos bytes faltam para serem transmiti-
 dos */ ;
 IF /* todo o pacote já foi transmitido (ar57/CC1 = 0) */ THEN
 BEGIN
  /* marque a30/CC1 com prioridade 2 */
 END
 ELSE
 BEGIN
  IF /* o pacote não foi inteiramente transmitido (ar57/CC1 <> 0) */
```

```
  THEN
  BEGIN
   /* decremente o conteúdo de ar57/CCI */ ;
   /* leia o próximo byte a ser transmitido de ar49/CCI */ ;
   /* escreva este byte em ar52/CCI */ ;
   /*  marque   a27/CCI  com  prioridade 1  (transmissão  física  do
   byte) */
  END
 END
END.

INTERPRETAÇÃO p(n25/CCI) ;
BEGIN
 /* leia byte de ar50/CCI (recepção física do byte) */ ;
 /* escreva este byte em ar47/CCI */ ;
 /*  marque  a26/CCI  com prioridade 3 e marque a34/CCI com  priori-
 dade 2 */
END.
```

3.2.2.2. NÍVEL DE QUADRO

As mensagens que fluirão pelo serviço de comunicação serão de tamanho variável, com limite máximo estipulado em 128 bytes. Tais mensagens serão empacotadas com a inserção de sequências especiais de caracteres delimitadoras de início e fim. Utilizar-se-á a sequência DLE (data link escape - 10 hexa) STX (start of text - 02 hexa) para sinalizar o início de um pacote e a sequência DLE ETX (end of text - 03 hexa) para a indicação de seu final.

Para garantir a transparência dos dados transportados, será inserido, na fase de empacotamento, um caráter DLE após cada DLE encontrado no corpo da mensagem. Tal estratégia assegura que o caráter DLE sempre ocorrerá aos pares no corpo da mensagem. Na recepção, caso seja detetada uma sequência ímpar de DLE's, o caráter após o último DLE deverá ser um STX (se o receptor estiver à espera de início de pacote) ou um ETX (se o receptor estiver à espera de fim de pacote). Qualquer ocorrência de DLE's em sequências diferentes das discutidas acima, sinalizará erro de recepção e recolocará o receptor no estado de espera de início de pacote.

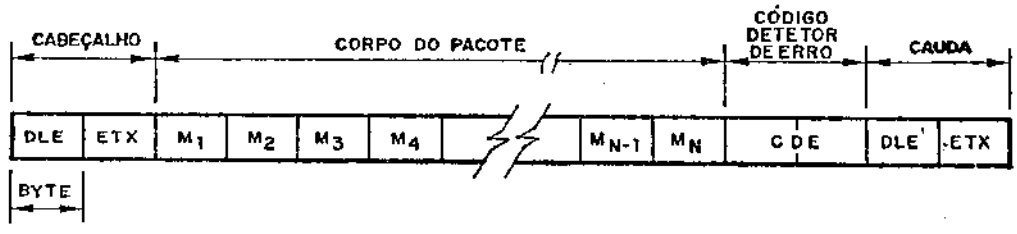


FIG.3.2.20 - FORMATO DO PACOTE TRATADO NO NÍVEL DE QUADRO

É tarefa do nível de quadro efetuar o empacotamento discutido acima, assim como, detetar os erros de transmissão que porventura ocorram no canal de comunicação. Como mecanismo de deteção de tais erros, será anexado à mensagem, antes do empacotamento, um código detetor de erros. Em uma primeira versão, ditada pela simplicidade,

o código a ser utilizado consistirá no complemento de dois do módulo 65536 do somatório dos bytes componentes da mensagem. O formato do pacote tratado por este nível é apresentado na figura 3.2.20.

Na figura 3.2.21 são apresentados os domínios do controle e dos dados do modelo GMB* do nível de quadro. Apresenta-se a seguir o domínio da interpretação.

DOMÍNIO DA INTERPRETAÇÃO

INTERPRETAÇÃO DOS AMAZENADORES

ar44/CCI - Armazenador de mensagem a ser envelopada / canal de comunicação I
ar44/CCI::=<TAMANHO DA MENSAGEM><MENSAGEM>
<MENSAGEM>::=<tipo da mensagem><CORPO DA MENSAGEM>
<tipo da mensagem>::=<tipo mensagem>/<tipo reconhecimento>
<tipo mensagem>::=<IDENTIFICADOR MENSAGEM>
<IDENTIFICADOR RECONHECIMENTO>
<tipo reconhecimento>::=<IDENTIFICADOR RECONHECIMENTO>

ar45/CCI - Armazenador de mensagem recebida / canal de comunicação I
ar45/CCI::=<TAMANHO DA MENSAGEM><MENSAGEM>
<MENSAGEM>::=<tipo da mensagem><CORPO DA MENSAGEM>
<tipo da mensagem>::=<tipo mensagem>/<tipo reconhecimento>
<tipo mensagem>::=<IDENTIFICADOR MENSAGEM>
<IDENTIFICADOR RECONHECIMENTO>
<tipo reconhecimento>::=<IDENTIFICADOR RECONHECIMENTO>

ar46/CCI - Armazenador de pacote a ser transmitido / canal de comunicação I
ar46/CCI::=<TAMANHO DO PACOTE><pacote>
<pacote>::=<cabeçalho><CORPO DO PACOTE><CHECK-SUM><cauda>
<cabeçalho>::=<DLE><STX>
<cauda>::=<DLE><ETX>

ar47/CCI - Armazenador de entrada da recepção / canal de comunicação I
ar47/CCI::=<BYTE>

ar48/CCI - Armazenador de montagem de pacote / canal de comunicação I
ar48/CCI::=(<BYTE>)

ar51/CCI - Contador de tamanho de pacote / canal de comunicação I
ar51/CCI::=<TAMANHO DO PACOTE>

ar55/CCI - Estado do receptor / canal de comunicação I
ar55/CCI::=<FORA SINCRONISMO>/<SINCRONIZANDO>/<RECEBENDO>/
<SAINDO DE SINCRONISMO>

INTERPRETAÇÃO DOS PROCESSADORES

INTERPRETAÇÃO **p(n21/CCI)** ;
BEGIN
/* leia de ar44/CCI, consoante o seu tamanho, a mensagem a ser transmitida */ ;
/* calcule o check-sum da mensagem: (65536 - (bi mod 65536)) mod

```
65536, onde bi é o somatório de todos os bytes que compoem a
mensagem) */ ;
/* anexe ao final da mensagem os 16 bits resultantes da operação
do passo anterior */ ;
/* pesquise, a partir do primeiro até o último byte (byte menos
significativo do check-sum), o bloco de dados resultante do passo
anterior, inserindo um outro DLE (10H) atrás de cada DLE que for
encontrado no corpo da mensagem */ ;
/* anexe ao bloco de dados resultante do passo anterior o cabeçalho
do pacote: DLE STX (10H 02H) */ ;
/* anexe ao bloco de dados resultante do passo anterior a cauda do
pacote: DLE ETX (10H 03H) */ ;
/* escreva o bloco de dados resultante do passo anterior em
ar46/CCI como o pacote a ser transmitido */ ;
/* escreva em ar46/CCI, no campo tamanho do pacote, o tamanho do
pacote a ser transmitido (observar que este tamanho depende do
número de DLE's inseridos no corpo da mensagem)*/ ;
/* marque a25/CCI com prioridade 2 */
END.
```

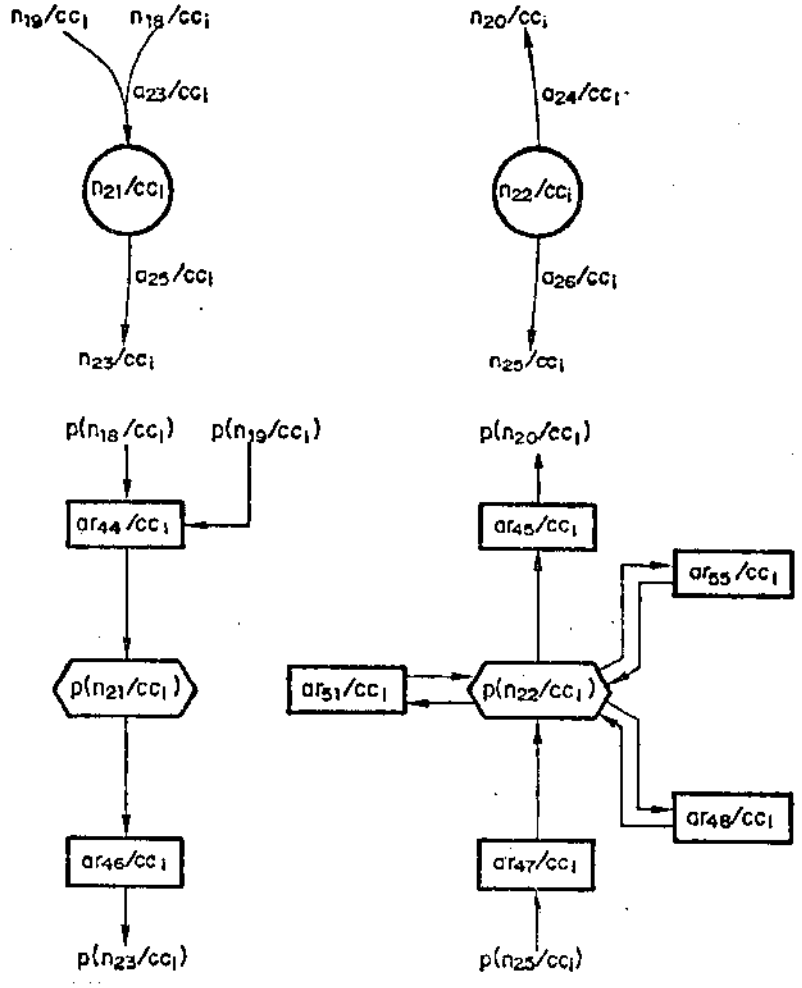


FIG.3.2.21 - MODELO GMB* DO NÍVEL DE QUADRO DO SERVIÇO DE COMUNICAÇÃO: DOMÍNIO DO CONTROLE E DOMÍNIO DOS DADOS

```
INTERPRETAÇÃO p(n22/CCI) :
BEGIN
 /* leia de ar55/CCI o estado do receptor */ ;
 IF /* o receptor está no estado fora de sincronismo */ THEN
```

```
BEGIN
 /* leia de ar47/CCI o byte recebido */ ;
 IF /* byte recebido é DLE */ THEN
 BEGIN
  /*  escreva  <sincronizando>  em ar55/CCI como o novo  estado  do
  receptor */
 END
END
ELSE
BEGIN
 IF /* o receptor está no estado sincronizando */ THEN
 BEGIN
  /* leia de ar47/CCI o byte recebido */ ;
  IF /* byte recebido é STX  */ THEN
  BEGIN
   /* zere o conteúdo de ar51/CCI */ ;
   /*  escreva  <recebendo> em ar55/CCI como o novo estado  do  re-
   ceptor */
  END
  ELSE
  BEGIN
   IF /* byte recebido não é STX */ THEN
   BEGIN
    /* escreva <fora de sincronismo> em ar55/CCI como o novo estado
    do receptor */
   END
  END
 END
 ELSE
 BEGIN
  IF /* receptor está no estado recebendo */ THEN
  BEGIN
   /* leia de ar47/CCI o byte recebido */ ;
   IF /* byte recebido não é DLE */ THEN
   BEGIN
    /* coloque o byte recebido em ar48/CCI */ ;
    /*  incremente   de   um   o   contador   de   pacote   definido   em
    ar51/CCI */
   END
   ELSE
   BEGIN
    IF /* byte recebido é DLE */ THEN
    BEGIN
     /*  escreva  <saindo de sincronismo> em ar55/CCI como  o  novo
     estado do receptor */ ;
    END
   END
  END
  ELSE
  BEGIN
   IF /* receptor está no estado saindo de sincronismo */ THEN
   BEGIN
    /* leia de ar47/CCI o byte recebido */ ;
    IF /* byte recebido é DLE */ THEN
    BEGIN
     /* escreva o byte recebido em ar48/CCI */ ;
     /*  incremente  de  uma  unidade  o  contador  definido  em
     ar51/CCI */ ;
```

```
      /* escreva <recebendo> em ar55/CCI como o novo estado do
      receptor */
     END
     ELSE
     BEGIN
      IF /* byte recebido não é DLE */ THEN
      BEGIN
       /* escreva <fora de sincronismo> em ar55/CCI como o novo
       estado do receptor */ ;
       IF /* byte recebido é ETX */ THEN
       BEGIN
        /* leia o conteúdo de ar51/CCI (tamanho do pacote lido) */ ;
        /* leia de ar48/CCI tantos bytes quanto indicado por tamanho
        do pacote - 2 */ ;
        /* some em 16 bits os bytes componentes deste bloco de
        dados */ ;
        /* some o resultado da operação do passo anterior aos dois
        último bytes residentes em ar48/CCI (check-sum) */ ;
        IF /* o resultado desta última operação é igual a zero */
        THEN
        BEGIN
         /* escreva em ar45/CCI, no campo tamanho da mensagem, o
         tamanho do pacote recebido - 2 */ ;
         /* escreva em ar45/CCI, no campo corpo da mensagem, a
         mensagem recebida (conteúdo de ar48/CCI retirado o check-
         sum) */ ;
         /* marque a24/CCI com prioridade 3 */
        END
       END
      END
     END
    END
   END
  END
 END
END.
```

3.2.2.3. NÍVEL DE ENLACE

O nível de enlace, ou nível 3 do serviço de comunicação, procura transformar o canal de comunicação oferecido pelos níveis inferiores (físico e quadro) em um canal confiável, capaz de transportar, sem perdas ou duplicações, todas as mensagens a ele confiadas. O protocolo a ser implementado neste nível é do tipo "one bit sliding window" /Tanenbaum 81/, algumas vezes denominado "alternanting bit protocol" /Davies 79/. Neste protocolo o elemento transmissor, após a transmissão de uma mensagem, coloca-se à espera de um reconhecimento, a ser enviado pelo elemento receptor, sinalizando o correto recebimento da mesma. Caso um reconhecimento demore a chegar, um mecanismo de temporização (time-out) dispara a retransmissão da mensagem evitando o dead-lock do sistema. Para evitar equívocos de sinalização de recebimento das mensagens, tanto esta, como os reconhecimento, são identificados com um bit. Este bit alternando entre "0" e "1", donde o nome do protocolo, distingue duas mensagens ou dois reconhecimentos consecutivos. Uma mensagem com identificador "0" deve ser confirmada com um reconhecimento "0",

uma mensagem "1" com reconhecimento "1".

Uma vez que o canal físico de comunicação deve comportar o fluxo bi-direcional de informação (mensagens em um sentido - reconhecimentos no outro) é conveniente, visando a racionalização da utilização do canal, que se permita, sempre que possível, a carona dos reconhecimentos nas mensagens que trafegam no sentido oposto. Tal técnica é conhecida como "piggy-backing" /Tanenbaum 81/. Entretanto, para evitar que uma eventual falta de mensagens atrase indefinidamente a transmissão de um reconhecimento, admitir-se-ão no sistema em proposição dois tipos distintos de pacotes: um que, além do reconhecimento, transporta também informações geradas pelo nível aplicativo e outro que transporta apenas informação de reconhecimento de mensagem.

Na figura 3.2.22 é apresentado o formato dos pacotes tratados pelo nível de enlace.

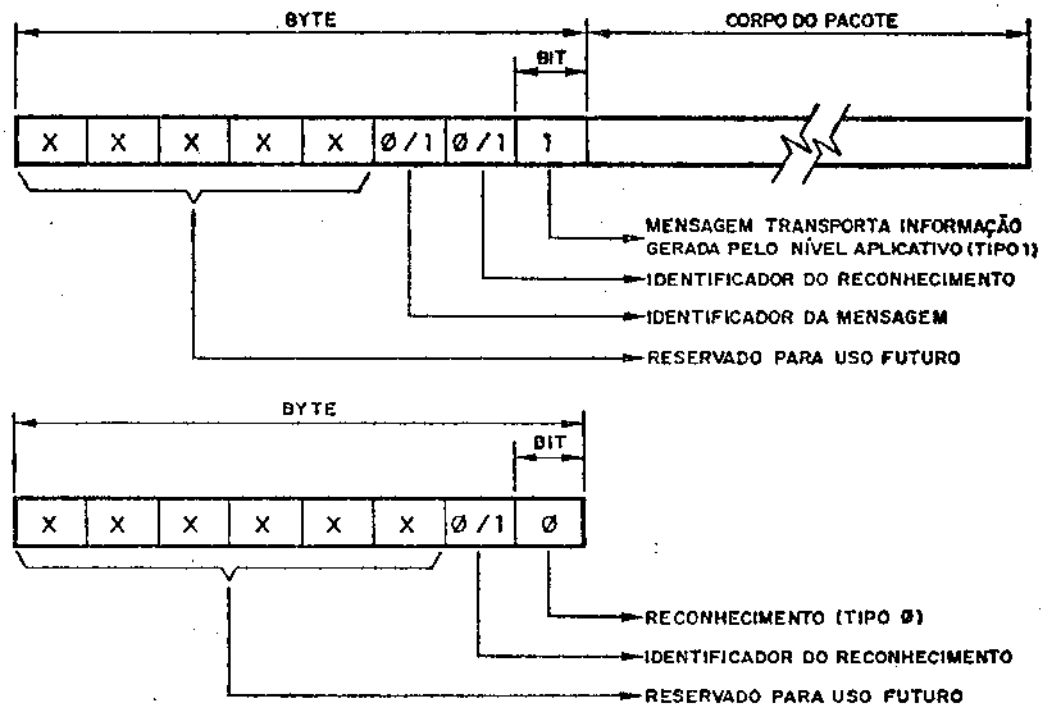


FIG.3.2.22 - FORMATO DOS PACOTES TRATADOS NO NÍVEL DE ENLACE

Na figura 3.2.23 são apresentados os domínios do controle e dos dados do modelo GMB* do nível de enlace. Apresenta-se a seguir o domínio da interpretação.

DOMÍNIO DA INTERPRETAÇÃO

INTERPRETAÇÃO DOS ARMAZENADORES

ar33 - Fila de espera dos pedidos de temporização do serviço de comunicação (time-out)
ar33::=(<CANAL DE COMUNICAÇÃO><IDENTIFICADOR ARCO DE CONTROLE>
<TEMPO DE ESPERA>)
obs: Os elementos deste armazenador estão dispostos em ordem crescente de espera.

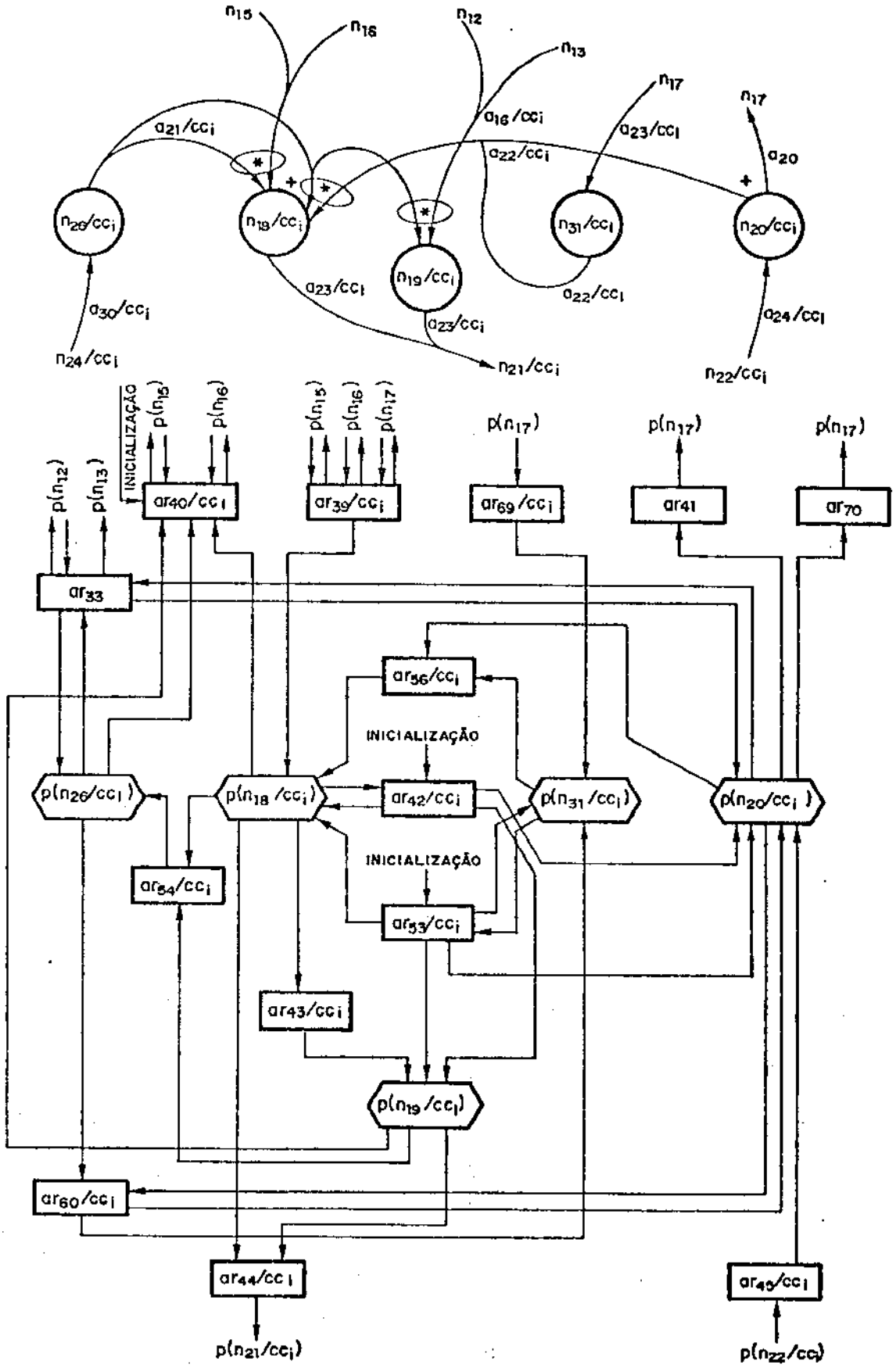


FIG.3.2.23 - MODELO GMB* DO NÍVEL DE ENLACE DO SERVIÇO DE COMUNICAÇÃO: DOMÍNIO DO CONTROLE E DOMÍNIO DOS DADOS

ar39/CCI - Fila de espera para a transmissão / canal de comunicação I
ar39/CCI::=(<TAMANHO DA INFORMAÇÃO><Informação>)
<Informação>::=<ESTAÇÃO DESTINO><tipo da informação>
<CORPO DA INFORMAÇÃO>
<tipo da informação>::=<TIPO DADOS>/<TIPO MARCA>

ar40/CCI - Indicador do estado do transmissor / canal de comunicação I
ar40/CCI::=<ACORDADO>/<DORMENTE>

ar41 - Armazenador de informação recebida
ar41::=<TAMANHO DA INFORMAÇÃO><Informação>
<Informação>::=<ESTAÇÃO DESTINO><tipo da informação>
<CORPO DA INFORMAÇÃO>
<tipo da informação>::=<TIPO DADOS>/<TIPO MARCA>

ar42/CCI - Identificador da última mensagem transmitida / canal de comunicação I
ar42/CCI::=<IDENTIFICADOR MENSAGEM>

ar43/CCI - Armazenador de espera para a re-transmissão / canal de comunicação I
ar43/CCI::=<TAMANHO DA INFORMAÇÃO><Informação>
<Informação>::=<ESTAÇÃO DESTINO><tipo da informação>
<CORPO DA INFORMAÇÃO>
<tipo da informação>::=<TIPO DADOS>/<TIPO MARCA>

ar44/CCI - Armazenador de mensagem a ser envelopada / canal de comunicação I
ar44/CCI::=<TAMANHO DA MENSAGEM><MENSAGEM>
<MENSAGEM>::=<tipo da mensagem><CORPO DA MENSAGEM>
<tipo da mensagem>::=<tipo mensagem>/<tipo reconhecimento>
<tipo mensagem>::=<IDENTIFICADOR MENSAGEM>
<IDENTIFICADOR RECONHECIMENTO>
<tipo reconhecimento>::=<IDENTIFICADOR RECONHECIMENTO>

ar45/CCI - Armazenador de mensagem recebida / canal de comunicação I
ar45/CCI::=<TAMANHO DA MENSAGEM><MENSAGEM>
<MENSAGEM>::=<tipo da mensagem><CORPO DA MENSAGEM>
<tipo da mensagem>::=<tipo mensagem>/<tipo reconhecimento>
<tipo mensagem>::=<IDENTIFICADOR MENSAGEM>
<IDENTIFICADOR RECONHECIMENTO>
<tipo reconhecimento>::=<IDENTIFICADOR RECONHECIMENTO>

ar53/CCI - Identificador da última mensagem recebida / canal de comunicação I
ar53/CCI::=<IDENTIFICADOR MENSAGEM>

ar54/CCI - Tipo da última mensagem transmitida / canal de comunicação I
ar54/CCI::= <TIPO MENSAGEM>/<TIPO RECONHECIMENTO>

ar56/CCI - Tipo da última mensagem recebida / canal de comunicação I
ar56/CCI::= <TIPO MENSAGEM>/<TIPO RECONHECIMENTO>

ar60/CCI - Indicador do estado da última mensagem transmitida / canal de comunicação I

ar60/CCI::=<ESPERA RECONHECIMENTO>/<NÃO ESPERA RECONHECIMENTO>

ar69/CCJ - Indicador do estado da mensagem encaminhada para o nível 4 pelo nível 3 do canal de comunicação J
ar69/CCJ::=<ACEITA>/<DESCARTADA>
ar70 - Identificador do canal de comunicação que recebeu a última mensagem
ar70::=<CANAL DE COMUNICAÇÃO>

INTERPRETAÇÃO DOS PROCESSADORES

```
INTERPRETAÇÃO p(n1B/CCI) ;
BEGIN
 /* mude o indicador do estado do transmissor associado a este canal
 de comunicação (ar40/CCI) para estado acordado */ ;
 /* verifique se ar39/CCI contém informação para ser transmi-
 tida */ ;
 IF /* ar39/CCI está vazio */ THEN
 BEGIN
  /* leia de ar56/CCI o tipo da última mensagem recebida */ ;
  IF /* última mensagem recebida é tipo mensagem */ THEN
  BEGIN
   /* escreva em ar54/CCI <tipo reconhecimento> (tipo da mensagem a
   ser transmitida) */ ;
   /* escreva em ar44/CCI, no campo tamanho da mensagem, o
   valor 1 */ ;
   /* leia de ar53/CCI o identificador da última mensagem recebi-
   da */ ;
   /* escreva este valor em ar44/CCI como o identificador do reco-
   nhecimento a ser transmitido */ ;
   /* marque a23/CCI com prioridade 2 */
  END
  ELSE
  BEGIN
   IF /* a última mensagem recebida é tipo reconhecimento */ THEN
   BEGIN
    /* mude em ar40/CCI o estado do transmissor para dormente */
   END
  END
 END
 ELSE
 BEGIN
  IF /* ar39/CCI não está vazio */ THEN
  BEGIN
   /* escreva em ar54/CCI <tipo mensagem> (tipo da mensagem a ser
   transmitida) */ ;
   /* leia de ar39/CCI, consoante o seu tamanho, o bloco de infor-
   mações a ser transmitido */ ;
   /* escreva este bloco de informações e a informação de seu res-
   pectivo tamanho em ar43/CCI */ ;
   /* escreva em ar44/CCI, no campo corpo da mensagem, o bloco de
   informações a ser transmitido */ ;
   /* leia de ar42/CCI o identificador da última mensagem transmiti-
   da */ ;
   /* calcule o módulo 2 deste valor (resto da divisão por dois) */;
   /* escreva o resultado desta operação em ar42/CCI como o novo
   valor do identificador da última mensagem transmitida */ ;
   /* escreva este mesmo valor em ar44/CCI como o identificador da
```

```
   mensagem a ser transmitida */ ;
   /* leia de  ar53/CCI o  identificador da  última mensagem recebi-
   da */ ;
   /*  escreva este valor em ar44/CCI como o identificador do  reco-
   nhecimento (piggy-backing) */ ;
   /* escreva em ar44/CCI,  no campo tamanho da mensagem,  o tamanho
   do bloco de informações lido de ar39/CCI + 1 */ ;
   /* marque a23/CCI com prioridade 2 */
  END
 END
END.

INTERPRETAÇÃO p(n19/CCI) ;
BEGIN
 /*  mude  o indicador do  estado do transmissor associado a este  a
 este canal de comunicação (ar40/CCI) para estado acordado */ ;
 /* escreva em ar54/CCI <tipo mensagem> (tipo da mensagem transmiti-
 da) */ ;
 /*  leia de ar43/CCI,  consoante o seu tamanho,  o bloco de  infor-
 mações  a  ser re-transmitido (mantenha uma cópia  deste  bloco  em
 ar43/CCI) */ ;
 /*  escreva  em ar44/CCI,  no campo corpo da mensagem,  o bloco  de
 dados a ser transferido */ ;
 /* leia de ar42/CCI o  identificador da  última mensagem  transmi-
 tida */ ;
 /* escreva este valor em ar44/CCI como o identificador da  mensagem
 a ser re-transmitida */ ;
 /* leia de ar53/CCI o identificador da última mensagem recebida */;
 /* escreva este valor em ar44/CCI como o identificador do  reconhe-
 cimento (piggy-backing) */ ;
 /* escreva em ar44/CCI, no campo tamanho da mensagem, o  tamanho do
 bloco de informações lido de ar43/CCI + 1 */ ;
 /* marque a23/CCI com prioridade 2 */
END.

INTERPRETAÇÃO p(n20/CCI) ;
BEGIN
 /* leia de ar45/CCI a mensagem recebida */ ;
 IF /* a mensagem recebida é do tipo reconhecimento */ THEN
 BEGIN
  /*  escreva  em ar56/CCI <tipo reconhecimento> (tipo  da  mensagem
  recebida) */ ;
  /*  leia  de  ar60/CCI se a última mensagem  transmitida  está  à
  espera de reconhecimento */ ;
  IF  /* a última  mensagem  transmitida  está  à  espera  de  reco-
  nhecimento */ THEN
  BEGIN
   /* leia de ar42/CCI o identificador da última mensagem transmiti-
   da */ ;
   IF  /*  identificador  da última mensagem trasmitida é  igual  ao
   identificador do reconhecimento recebido */ THEN
   BEGIN
    /* retire de ar33 o pedido de temporização deste canal de  comu-
    nicação (time-out) */ ;
    /*  indique em ar60/CCI que a última mensagem transmitida não se
    encontra mais à espera de reconhecimento */ ;
    /* marque a22/CCI com prioridade 2 */
   END
```

```
 END
END
ELSE
BEGIN
 IF /* a mensagem recebida é do tipo mensagem */ THEN
 BEGIN
  /*  leia  de ar60/CCI se a última mensagem  transmitida  está  à
  espera de reconhecimento */ ;
  IF  /* a última  mensagem  transmitida  está  à  espera  de reco-
  nhecimento */ THEN
  BEGIN
   /* leia de ar42/CCI o identificador da última mensagem transmi-
   tida */ ;
   IF  /* o identificador da última mensagem transmitida é igual ao
   reconhecimento recebido */ THEN
   BEGIN
    /*  retire de ar33/CCI o pedido de temporização deste canal  de
    comunicação (time-out) */ ;
    /* indique em ar60/CCI que a última mensagem transmitida não se
    encontra mais à espera de reconhecimento */ ;
    /* leia de  ar53/CCI o  identificador da última mensagem rece-
    bida */ ;
    /*  calcule   o  módulo 2  deste  valor (resto da  divisão  por
    dois) */ ;
    IF  /*  o resultado da operação do passo anterior é  igual  ao
    identificador da mensagem recebida */ THEN
    BEGIN
     /* escreva em ar41 o corpo da mensagem recebida */ ;
     /* escreva em ar41,  no campo tamanho da informação, o tamanho
     da mensagem lida de ar45/CCI - 1 */ ;
     /*  escreva em ar70 o identificador deste canal de comunicação
     (canal de comunicação J) */ ;
     /* marque a20 com prioridade 3 */
    END
    ELSE
    BEGIN
     IF /* o resultado da operação do passo anterior é diferente do
     identificador da mensagem recebida */ THEN
     BEGIN
      /*  escreva  em ar56/CCI <tipo mensagem>  (tipo  da  mensagem
      recebida */ ;
      /* marque a22/CCI com prioridade 2 */
     END
    END
   END
   ELSE
   BEGIN
    IF /* o identificador da última mensagem transmitida é diferen-
    te do reconhecimento recebido */ THEN
    BEGIN
     /*  leia de ar53/CCI  o identificador da última mensagem rece-
     bida */ ;
     /*  calcule  o módulo 2  deste  valor (resto  da  divisão  por
     dois) */ ;
     IF  /*  o resultado da operação do passo anterior é  igual  ao
     identificador da mensagem recebida */ THEN
     BEGIN
      /* escreva em ar41 o corpo da mensagem recebida */ ;
```

```
        /* escreva em ar41, no campo tamanho da informação, o
        tamanho da mensagem lida de ar45/CCI - 1 */ ;
        /* escreva em ar70 o identificador deste canal de comunicação
        (canal de comunicação i) */ ;
        /* marque a20 com prioridade 3 */
       END
      END
     END
    END
    ELSE
    BEGIN
     IF  /* a última mensagem transmitida não está à espera de reco-
     nhecimento */ THEN
     BEGIN
      /* leia de ar53/CCI o identificador da última mensagem rece-
      bida */ ;
      /* calcule o módulo 2 deste valor (resto da divisão por
      dois) */ ;
      IF /* o resultado da operação do passo anterior é igual ao
      identificador da mensagem recebida */ THEN
      BEGIN
       /* escreva em ar41 o corpo da mensagem recebida */ ;
       /* escreva em ar41, no campo tamanho da informação, o tamanho
       da mensagem lida de ar45/CCI - 1 */ ;
       /* escreva em ar70 o identificador deste canal de comunicação
       (canal de comunicação i) */ ;
       /* marque a20 com prioridade 3 */
      END
      ELSE
      BEGIN
       IF /* o resultado da operação do passo anterior é igual ao
       identificador da mensagem recebida */ THEN
       BEGIN
        /* escreva em ar56/CCI <tipo mensagem> (tipo da mensagem
        recebida) */ ;
        /* marque a22/CCI com prioridade 2 */
       END
      END
     END
    END
   END
  END
END.

INTERPRETAÇÃO p(n26/CCI) ;
BEGIN
 /* leia de ar54/CCI o tipo da última mensagem transmitida */ ;
 IF /* a mensagem transmitida é do tipo da mensagem */ THEN
 BEGIN
  /* escreva em ar33 o pedido de temporização (time-out), onde: o
  canal de comunicação é i, o identificador do arco de controle é
  a16/CCI e o tempo de espera é t (a ser definido na fase de
  implementação) */
  /* escreva em ar60/CCI que a última mensagem transmitida se encon-
  tra à espera de reconhecimento */
 END
 ELSE
 BEGIN
```

```
  IF /* a última mensagem transmitida foi do tipo reconhecimento  */
  THEN
  BEGIN
   /*  mude  o indicador do estado do transmissor associado  a  este
   canal de comunicação (ar40/CCI) para estado dormente */
  END
 END ;
 /* marque a21/CCI com prioridade 1 */
END.

INTERPRETAÇÃO p(n31/CCI) ;
BEGIN
 /*  leia de ar69/CCI o estado da mensagem encaminhada para o  nível
 de rede */ ;
 IF /* a mensagem foi aceita */ THEN
 BEGIN
  /*  escreva  em ar56/CCI <tipo mensagem> (tipo da última  mensagem
  recebida */ ;
  /* leia de ar53/CCI o  identificador da  última  mensagem  recebi-
  da */ ;
  /*  calcule   o  módulo  2  deste  valor (resto  da  divisão  por
  dois) */ ;
  /*  escreva o resultado da operação do passo anterior em  ar53/CCI
  como  o  novo  identificador  da  última  mensagem recebida */ ;
  /* leia de ar60/CCI se a última mensagem transmitida está à espera
  de reconhecimento */ ;
  IF /* a última mensagem transmitida está à espera de reconhecimen-
  to */ THEN
  BEGIN
   /* marque a22/CCI com prioridade 2 */
  END
 END
 ELSE
 BEGIN
  IF /* a mensagem foi descartada */ THEN
  BEGIN
   /*  leia  de  ar60/CCI se a última mensagem  transmitida  está  à
   espera de reconhecimento */ ;
   IF /* a última mensagem transmitida não está à espera de reconhe-
   cimento */ THEN
   BEGIN
    /*  escreva  em ar56/CCI <tipo reconhecimento> (tipo  da  última
    mensagem recebida) */ ;
    /* marque a22/CCI com prioridade 2 */
   END
  END
 END
END.
```

3.2.2.4. NÍVEL DE REDE

Dado um conjunto de máquinas GMB* uni-processadoras interconectadas, é tarefa do nível de rede cuidar do roteamento das mensagens no interior da estrutura, fazendo com que as mensagens cheguem corretamente a seus destinos. A estratégia de roteamento a ser utilizada, nesta primeira versão, será do tipo estático, onde

tabelas fixas indicam para cada destino a rota a ser seguida.

Na figura 3.2.24 é apresentado o formato dos pacotes tratados no nível de rede.

Em particular, na solução adotada utilizam-se dois níveis de tabelas de roteamento: tabelas descritivas da distribuição do aplicativo e tabelas descritivas da topologia da rede. As tabelas descritivas da distribuição do aplicativo fazem o mapeamento de cada elemento receptor de mensagens do modelo GMB* aplicativo (arcos de controle e armazenadores) às respectivas máquinas hospedeiras. Por sua vez, a tabela descritiva da topologia da rede estabelece, em cada máquina, as rotas a serem seguidas para se atingir cada uma das outras máquinas da rede. Esta divisão de tabelas de roteamento visa facilitar a implementação de estratégias adaptativas de roteamento ou mesmo de mecânismos para a configuração dinâmica do sistema.

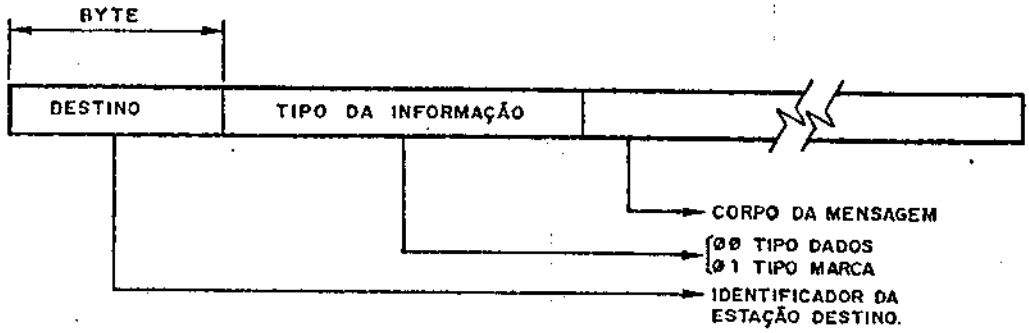


FIG.3.2.24 - FORMATOS DO PACOTE TRATADO NO NÍVEL DE REDE

Como já mencionado, uma rede de máquinas uni-processadoras constitui o que se denominou de máquina GMB* distribuída. Tal estrutura possui a característica particular de admitir nos canais de comunicação a existência de dois tipos distintos de informação: marcas e dados. É, pois, também função do nível de rede discernir tais tipos de informação, dispensando os cuidados necessários para os seus corretos roteamentos.

Vale frisar que na proposta de implementação do serviço de comunicação, assumiu-se que as interligações entre as máquinas GMB* uni-processadoras serão do tipo ponto-a-ponto. Foi considerado, também, que cada conexão entre duas máquina contará com a disponibilidade de um canal físico de comunicação exclusivo. Neste sentido, far-se-á necessaria a existência de interfaces de transmissão/recepção distintas para cada conexão, assim como, a reprodução dos protocolos dos níveis 1, 2 e 3 do serviço de comunicação para cada uma destas interfaces. A figura 3.2.25 salienta esta característica dentro do serviço de comunicação. Nesta figura as linhas tracejadas representam possíveis trajetórias de mensagens pelo serviço de comunicação.

Finalmente, convém observar que na proposta da máquina GMB* distribuída, aqui apresentada, foram impostas duas restrições básicas quanto à distribuição do aplicativo: a primeira delas determina que o acesso de leitura a um armazenador só será permitido a processadores que estejam implementados, juntamente com o armazenador acessado, na mesma máquina uni-processadora; a segunda

exige a residência na mesma máquina uni-processadora de todos os nós destinos de um mesmo arco de controle complexo.

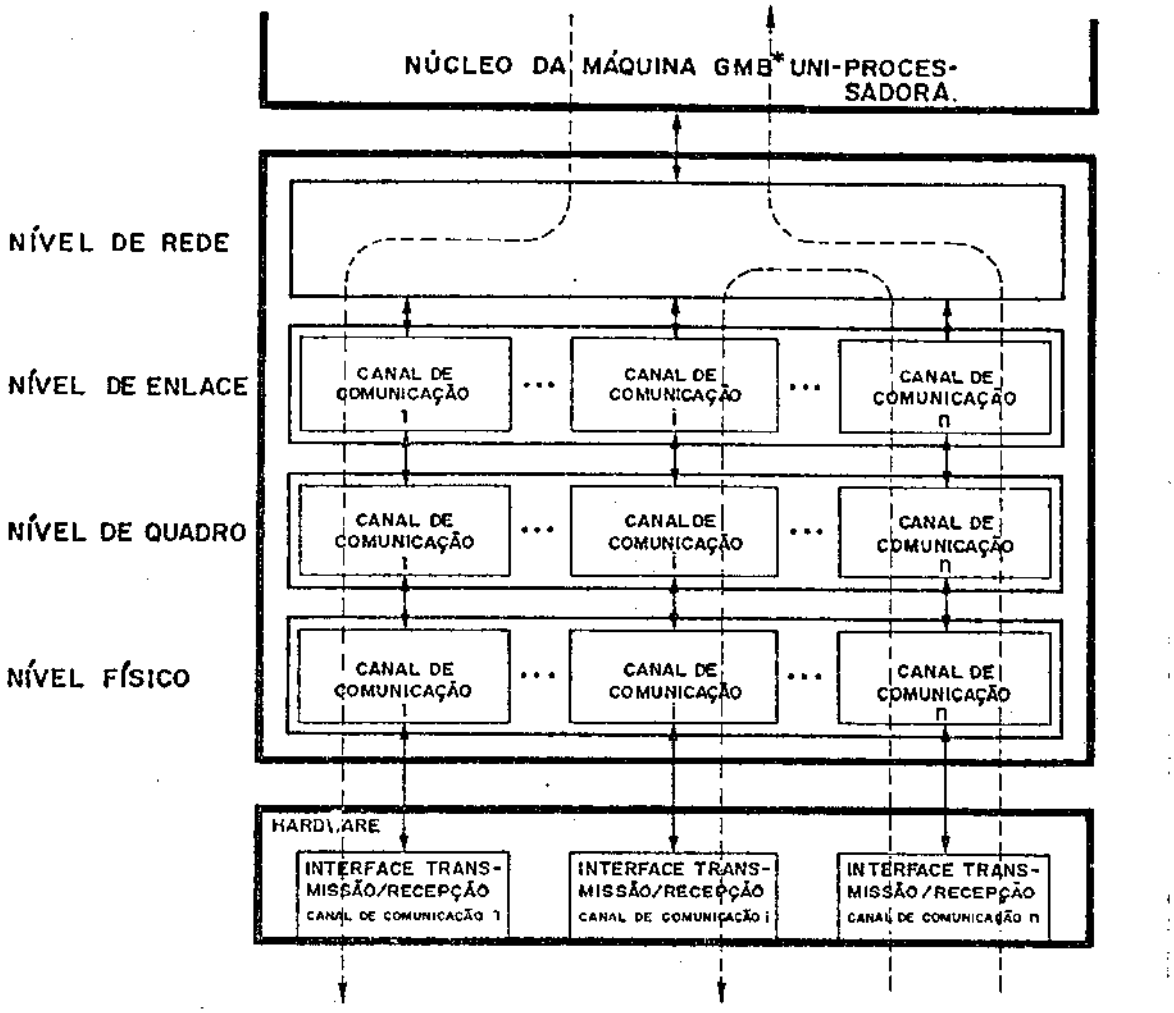


FIG.3.2.25 - SERVIÇO DE COMUNICAÇÃO: DETALHAMENTO DA MODULARIDADE DOS NÍVEIS INFERIORES

Na figura 3.2.26 são apresentados os domínios do controle e dos dados do modelo GMB* do nível de rede. Apresenta-se a seguir o domínio da interpretação.

DOMÍNIO DA INTERPRETAÇÃO

INTERPRETAÇÃO DOS ARMAZENADORES

ar7 - Pedido de transmissão de dados
ar7::=<IDENTIFICADOR ARMAZENADOR><tipo de acesso><dados>
<tipo de acesso>::=<DESTRUTIVO>/<NÃO-DESTRUTIVO>
<dados>::=<TAMANHO BLOCO DE DADOS><BLOCO DE DADOS>/
<BLOCO DE DADOS>

ar13 - Pedido de transmissão de marca
ar13::=<IDENTIFICADOR ARCO DE CONTROLE><PRIORIDADE MARCA>

ar36/CCi - Sequenciador de execução / canal de comunicação i
ar36/CCi::=<IDENTIFICADOR ARCO DE CONTROLE>

ar37 - Tabela de roteamento armazenador-estação
ar37::=(<IDENTIFICADOR ARMAZENADOR><ESTAÇÃO DESTINO>)

ar38 - Tabela de roteamento arco de controle-estação
ar38::=(<IDENTIFICADOR ARCO DE CONTROLE><ESTAÇÃO DESTINO>)

ar39/CCI - Fila de espera para a transmissão / canal de comunicação I
ar39/CCI::=(<TAMANHO DA INFORMAÇÃO><Informação>)
<Informação>::=<ESTAÇÃO DESTINO><tipo da informação>
<CORPO DA INFORMAÇÃO>
<tipo da informação>::=<TIPO DADOS>/<TIPO MARCA>

ar40/CCI - Estado do transmissor / canal de comunicação I
ar40/CCI::=<ACORDADO>/<DORMENTE>

ar41 - Armazenador de informação recebida
ar41::=<TAMANHO DA INFORMAÇÃO><Informação>
<Informação>::=<ESTAÇÃO DESTINO><tipo da informação>
<CORPO DA INFORMAÇÃO>
<tipo da informação>::=<TIPO DADOS>/<TIPO MARCA>

ar58 - Tabela de roteamento estação-porta física
ar58::=(<ESTAÇÃO DESTINO><localização>)
<localização>::=<INTERNA>/<externa>
<externa>::=<CANAL DE COMUNICAÇÃO>
ar59 - Armazenador de mensagem destinada a armazenador da estação
ar59::=<TAMANHO DO BLOCO DE DADOS><BLOCO DE DADOS>/
<BLOCO DE DADOS>

ar61 - Sequenciador de execução
ar61::=<IDENTIFICADOR ARCO DE CONTROLE>

ar62 - Sequenciador de execução
ar62::=<IDENTIFICADOR ARCO DE CONTROLE>

ar65 - Pedido de escrita em armazenador
ar65::=<IDENTIFICADOR ARMAZENADOR><tipo de acesso>
<ENDEREÇO DOS DADOS>
<tipo de acesso>::=<DESTRUTIVO>/<NÃO-DESTRUTIVO>

ar68 - Pedido de marcação de arco de controle
ar68::=<IDENTIFICADOR ARCO DE CONTROLE><PRIORIDADE MARCA>

ar69/CCJ - Indicador do estado da mensagem encaminhada para o nível 4 pelo nível 3 do canal de comunicação J
ar69/CCJ::=<ACEITA>/<DESCARTADA>

ar70 - Identificador do canal de comunicação que recebeu a última mensagem
ar70::=<CANAL DE COMUNICAÇÃO>

INTERPRETAÇÃO DOS PROCESSADORES

```
INTERPRETAÇÃO p(n15) ;
BEGIN
 /* leia de ar7 o pedido de transmissão de dados */ ;
 /* de posse do identificador do armazenador destino dos dados, leia
```

```
 de ar37 a estação destino dos dados */ ;
 /* de posse do identificador da estação destino, leia de ar58 o
 canal de comunicação que permite o acesso a esta estação */ ;
 /* calcule o tamanho do bloco de dados a ser colocado na fila de
 espera da transmissão (tamanho do pedido de transmissão + 3) */;
 /* verifique se a fila de espera associada ao canal de comunicação
 (ar39/CCI) possui espaço suficiente para a inserção do bloco de
 informação a ser transmitido */ ;
 IF /* ar39/CCI está cheio */ THEN
 BEGIN
  /* marque a3 com prioridade 1 */
 END
 ELSE
 BEGIN
  IF /* ar39/CCI não está cheio */ THEN
  BEGIN
   /* escreva em ar39/CCI, no campo tamanho da informação, o tama-
   nho do bloco de informação a ser transmitido (tamanho do pedido
   de transmissão + 2) */ ;
   /* escreva em ar39/CCI, no campo correspondente, o identificador
   da estação destino dos dados */ ;
   /* escreva em ar39/CCI, no campo correspondente, o tipo de infor-
   mação que ali foi depositada, no caso: <tipo dados> */ ;
   /* escreva o conteúdo de ar7 em ar39/CCI no campo corpo da infor-
   mação */ ;
   /* verifique em ar40/CCI o estado do transmissor associado ao
   canal I */ ;
   IF /* o transmissor está dormente */ THEN
   BEGIN
    /* leia o sequenciador de execução definido em ar61 */ ;
    IF /* o sequenciador especifica arco de controle */ THEN
    BEGIN
     /* escreva o identificador deste arco de controle em
     ar36/CCI */
    END ;
    /* escreva em ar40/CCI estado acordado */ ;
    /* marque o arco de controle associado ao canal de comunicação
    que permite o acesso à estação destino dos dados (a19/CCI) com
    prioridade 2 */
   END
   ELSE
   BEGIN
    IF /* o transmissor está acordado */ THEN
    BEGIN
     /* leia sequenciador de execução definido em ar61 */ ;
     IF /* sequenciador especifica identificador de arco de con-
     trole */ THEN
     BEGIN
      /* marque este arco com prioridade 1 */
     END
    END
   END
  END
 END
END.

INTERPRETAÇÃO p(n16) ;
BEGIN
```

```
/* leia de ar13 o pedido de marcação de arco de controle */ ;
/* de posse do identificador do arco de controle destino da marca,
leia de ar38 a estação destino desta marca (estação onde reside o
arco) */ ;
/* de posse do identificador da estação destino, leia de ar58 o
identificador do canal de comunicação que permite o acesso a
esta estação */ ;
/* calcule o tamanho do bloco de dados a ser colocado na fila de
espera de transmissão (tamanho do pedido de marcação + 3 ) */ ;
/* verifique se a fila de espera associada ao canal de comunicação
(ar39/CCI) possui espaço suficiente para a inserção do bloco de
informações a ser transmitido */ ;
IF /* ar39/CCI está cheio */ THEN
BEGIN
 /* marque a6 com prioridade 1 */
END
ELSE
BEGIN
 IF /* ar39/CCI não está cheio */ THEN
 BEGIN
  /* escreva em ar39/CCI,no campo tamanho da informação, o tamanho
  do bloco de informação a ser transmitido (tamanho do pedido de
  marcação + 2) */ ;
  /* escreva em ar39/CCI, no campo correspondente, o identificador
  da estação destino do pedido de marcação */ ;
  /* escreva em ar39/CCI o tipo de informação que ali foi deposita-
  da, no caso: <tipo marca> */ ;
  /* escreva o conteúdo de ar13 em ar39/CCI no campo corpo da
  informação */ ;
  /* verifique em ar40/CCI o estado do transmissor associado ao
  canal I */ ;
  IF /* o transmissor está dormente */ THEN
  BEGIN
   /* leia o sequenciador de execução definido em ar62 */ ;
   IF /* sequenciador de execução especifica identificador de arco
   de controle */ THEN
   BEGIN
    /* escreva o identificador deste arco de controle em
    ar36/CCI */
   END ;
   /* escreva em ar40/CCI estado acordado */ ;
   /* marque o arco de controle associado ao canal de comunicação
   que permite o acesso à estação destino do pedido de marcação
   (a19/CCI) com prioridade 2 */
  END
  ELSE
  BEGIN
   IF /* o transmissor está acordado */ THEN
   BEGIN
    /* leia o sequenciador de execução definido em ar62 */ ;
    IF /* o sequenciador especifica identificador de arco de con-
    trole */ THEN
    BEGIN
     /* marque este arco com prioridade 1 */
    END
   END
  END
 END
```

```
 END
END.
```

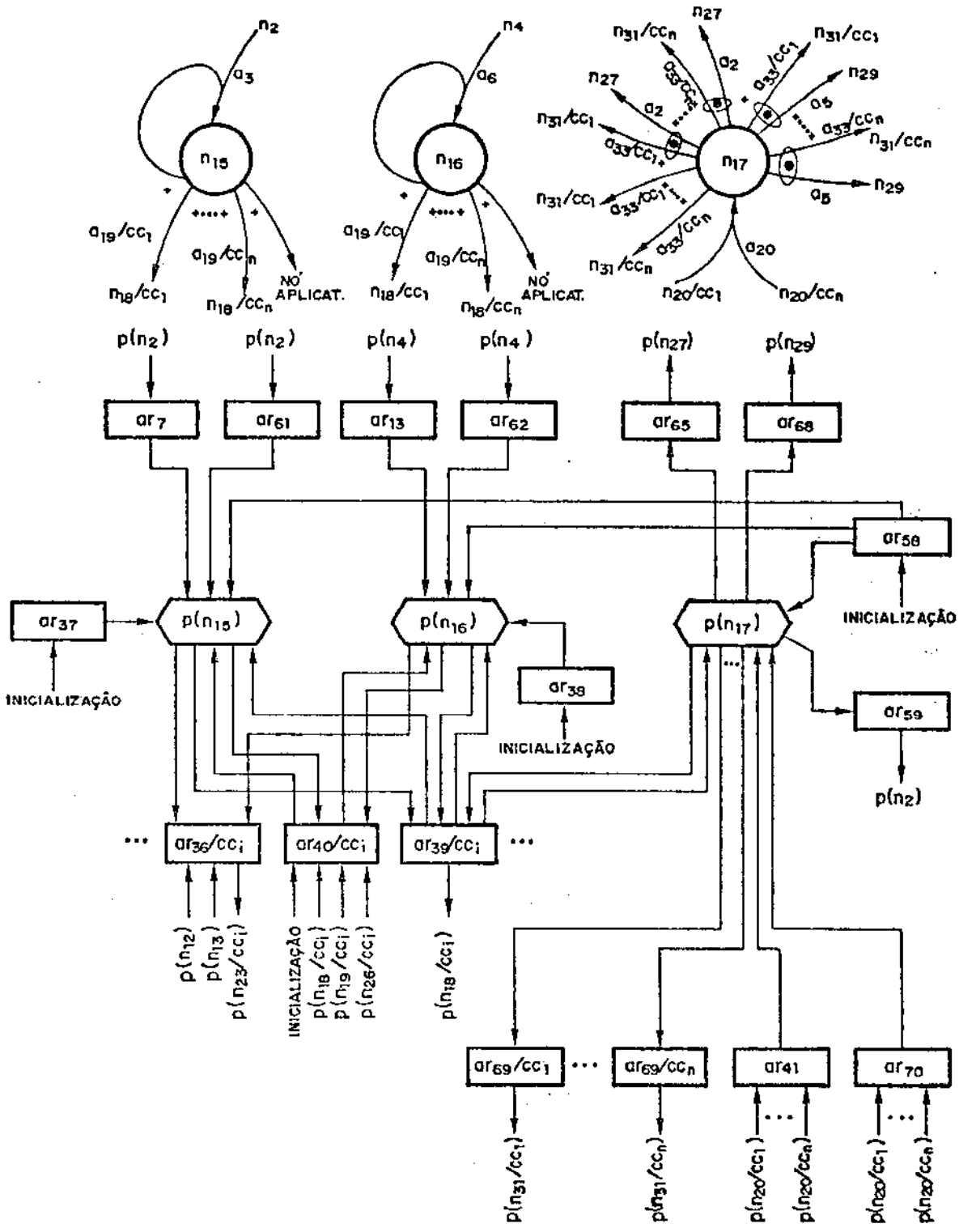


FIG.3.2.26 - MODELO GMB* DO NÍVEL DE REDE DO SERVIÇO DE COMUNICAÇÃO: DOMÍNIO DO CONTROLE E DOMÍNIO DOS DADOS

```
INTERPRETAÇÃO p(n17) :
BEGIN
```

```
/* leia o primeiro byte de ar41 (tamanho da informação) */ ;
/* leia o segundo byte de ar41 (estação destino) */ ;
/* leia de ar58 a localização da estação destino da mensa-
gem */ ;
IF /* localização é interna */ THEN
BEGIN
 /* leia o terceiro byte de ar41 (tipo informação) */ ;
 IF /* informação é tipo dados */ THEN
 BEGIN
  /* leia o quarto byte de ar41 (identificador armazenador) */ ;
  /* escreva este byte em ar65 no campo identificador arma-
  zenador */ ;
  /* leia o quinto byte de ar41 (tipo de acesso) */ ;
  /* escreva este byte em ar65 no campo tipo de acesso */ ;
  /* leia a partir do sexto byte de ar41 tantos bytes quanto espe-
  cificado por tamanho da informação (primeiro byte lido de
  ar41) - 4 */ ;
  /* escreva este bloco de dados em ar59 */ ;
  /* escreva em ar65, no campo endereço dos dados, o endereço do
  armazenador ar59 */ ;
  /* leia de ar70 o identificador do canal de comunicação que
  recebeu a mensagem (canal de comunicação j) */ ;
  /* indique no armazenador 69 associado a este canal de comuni-
  cação (ar69/CCj) que a referida mensagem foi aceita */ ;
  /* marque a2 e a33/CCj, ambos com prioridade 3 */
 END
 ELSE
 BEGIN
  IF /* informação é tipo marca */ THEN
  BEGIN
   /* leia o quarto byte de ar41 (identificador arco de con-
   trole) */;
   /* escreva este byte em ar68 no campo identificador arco de
   controle */ ;
   /* leia o quinto byte de ar41 (prioridade marca) */ ;
   /* escreva este byte em ar68 no campo prioridade marca */ ;
   /* leia de ar70 o identificador do canal de comunicação que
   recebeu a mensagem (canal de comunicação j) */ ;
   /* indique no armazenador 69 associado a este canal de comuni-
   cação (ar69/CCj) que a referida mensagem foi aceita */ ;
   /* marque a5 e a33/CCj, ambos com prioridade 3 */
  END
 END
END
ELSE
BEGIN
 IF /* localização é externa */ THEN
 BEGIN
  /* verifique se a fila de espera associada a este canal de comu-
  nicação (ar39/CCi) possui espaço suficiente para a inserção do
  bloco de informação que se deseja transmitir (o tamanho da
  informação é dado pelo primeiro byte lido de ar41) */ ;
  IF /* ar39/CCi está cheio */ THEN
  BEGIN
   /* leia de ar70 o identificador do canal de comunicação que
   recebeu a mensagem (canal de comunicação j) */ ;
   /* indique no armazenador 69 associado a este canal de comuni-
   cação (ar69/CCj) que a referida mensagem foi descartada */ ;
```

```
     /* marque a33/CCJ com prioridade 3  */
    END
    ELSE
    BEGIN
     IF /* ar39/CCI não está cheio */ THEN
     BEGIN
      /*  escreva em ar39/CCI o tamanho do bloco de informação a  ser
      transmitido (primeiro byte lido de ar41) */ ;
      /* escreva em ar39/CCI a estação destino do bloco de informação
      (segundo byte lido de ar41) */ ;
      /* leia o terceiro byte de ar41 ( tipo da informação) */ ;
      /* escreva  este  byte  em  ar39/CCI  no  campo  tipo da infor-
      mação */ ;
      /* leia de ar41,  a partir do quarto byte,  tantos bytes quanto
      especificados por tamanho da informação - 2 */ ;
      /*  escreva  este bloco de dados em ar39/CCI no campo corpo  da
      informação */ ;
      /*  leia  de ar70 o identificador do canal de comunicação  que
      recebeu a mensagem (canal de comunicação J) */ ;
      /* indique no armazenador 69 associado a este canal de comuni-
      cação (ar69/CCJ)  que  a  referida  mensagem  foi aceita */ ;
      /*  marque o arco de controle a33 associado ao canal de comuni-
      cação que recebeu a mensagem (a33/CCJ) com prioridade 3 */
     END
    END
   END
  END
END.
```

3.2.3. HARDWARE DA MÁQUINA GMB* UNI-PROCESSADORA

Na camada hardware da máquina GMB* uni-processadora (figura 3.2.1) localizam-se os recursos físicos necessários à sua operação. Tal camada sotoposta ao serviço de comunicação (seção 3.2.2.) e ao núcleo (seção 3.2.1.) estabelece um ambiente de programação GMB*. Em particular, propõe-se, como elemento de implementação desta camada, a já tradicional estrutura hardware de um micro-computador. Esta proposta foi norteada, somada à disponibilidade de mercado, pelo baixo custo desta estrutura.

3.2.4. SOFTWARE APLICATIVO

A camada software aplicativo é o modelo GMB* desenvolvido pelo usuário para atender as necessidades impostas por sua aplicação.

3.3. MÁQUINA GMB* DISTRIBUÍDA

À estrutura formada pela interconexão arbitrária de máquinas GMB* uni-processadoras é dada a denominação de máquina GMB* distribuída (figura 3.3.1).

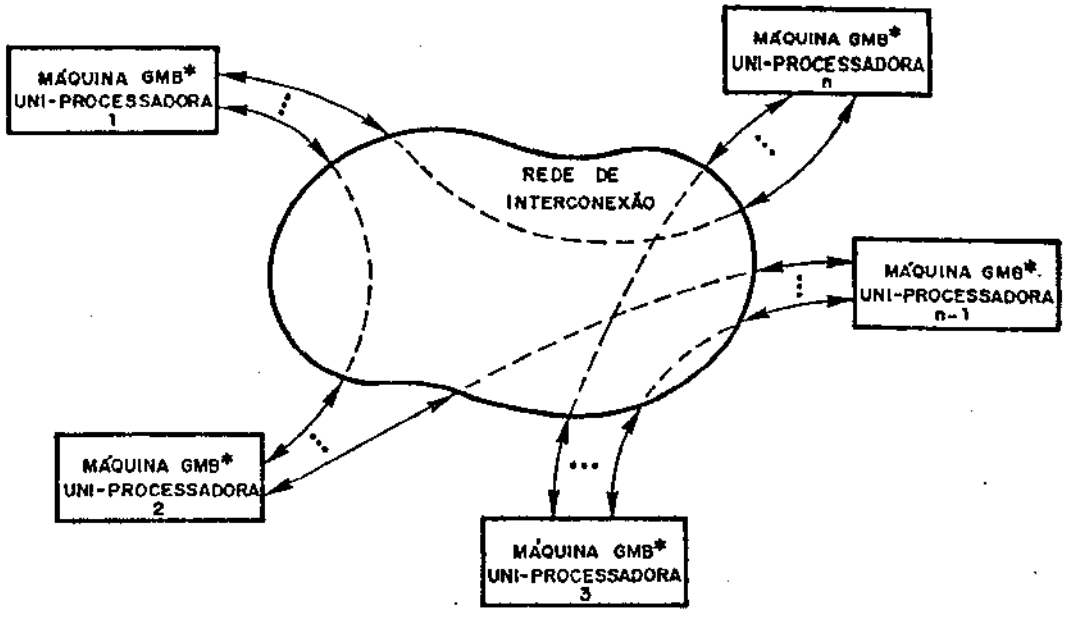


FIG.3.3.1 - ARQUITETURA DA MÁQUINA GMB* DISTRIBUÍDA

3.4. CONSIDERAÇÕES FINAIS

Associada ao ambiente de programação estabelecido pela máquina GMB* distribuída, define-se, também, uma metodologia para o desenvolvimento da camada aplicativa. Esta metodologia apoia-se no formalismo do modelo GMB* e tem por objetivo estabelecer uma sistemática para o desenvolvimento e implementação de programas GMB*. A metodologia em proposição constitui-se das seguintes etapas:

1) DECOMPOSIÇÃO DO SISTEMA EM MÓDULOS (definição do modelo GMB*):

1.1) DECOMPOSIÇÃO FUNCIONAL DO SISTEMA (definição do domínio do controle): A decomposição funcional do sistema em módulos é norteada pela interdependência dos módulos e pelo grau de paralelismo desejado para o sistema. Tais parâmetros podem ser avaliados qualitativamente segundo critérios de coesão e acoplamento /Magalhães 88/. Coesão é a medida da unidade funcional de um módulo, isto é, um módulo altamente coeso deve desempenhar idealmente apenas uma função. Acoplamento é a medida de interdependência entre os módulos. O grau de acoplamento pode ser avaliado pelo volume e diversidade dos tipos de dados trocados entre os módulos para que o mesmos realizem as suas funções. Módulos fracamente acoplados trocam poucos dados de um mesmo tipo.

1.2) DEFINIÇÃO DAS INTERFACES ENTRE MÓDULOS (definição do domínio dos dados): As interfaces entre os módulos são estabelecidas no formalismo GMB* pelo elemento armazenador. A um armazenador pode-se associar tipos de dados, estabelecendo-se subsídios para testes de

consistência das informações trocadas entre módulos.

1.3) ESPECIFICAÇÃO DA INTERPRETAÇÃO (definição do domínio da interpretação): Complementando as sub-etapas anteriores tem-se a especificação das interpretações associadas aos módulos e às suas interfaces. A especificação da interpretação pode, na verdade, ser constituída por uma hierarquia de domínios da interpretação GMB* que retratam os sucessivos refinamentos efetuados sobre a especificação funcional inicial, relativamente abstrata, até a sua conversão em uma especificação procedural codificada em uma linguagem de programação.

2) DEFINIÇÃO DAS PRIORIDADES ASSOCIADAS AOS MÓDULOS: Nesta etapa é efetuada uma análise dos requisitos temporais da aplicação. Em função desta análise e da decomposição realizada na etapa anterior são identificadas as prioridades dos módulos que compoem o sistema. Tais prioridades são absorvidas pelo modelo GMB* e definem a política de escalonamento da execução dos módulos.

3) CONFIGURAÇÃO FÍSICA/LÓGICA DO SISTEMA: Finalmente, a última etapa da metododologia consiste na configuração física e lógica do sistema. A configuração física refere-se à definição da topologia da máquina GMB* distribuída (quantidade de máquinas uni-processadoras, posicionamento geográfico e interligação). A configuração lógica define a estratégia de distribuição do aplicativo pela estrutura física. Em linhas gerais, a configuração do sistema é guiada pela distribuição geográfica dos pontos de demanda de processamento. Uma vez estabelecida a configuração física/lógica do sistema, podem-se fazer necessárias, a critério do grau de confiabilidade/desempenho exigido pela aplicação, transformações localizadas no modelo GMB* definido nas fases anteriores. Tais transformações teriam o objetivo de tratar os eventuais problemas advindos da existência de canais de físicos de comunicação, entre os diversos módulos (máquinas GMB* uni-processadoras) que compõem a máquina GMB* distribuída, capazes de destruir, distorcer e atrasar mensagens.

As tarefas de decomposição do sistema em módulos e de configuração da máquina GMB* distribuída sofrem atualmente do grave inconveniente de dependerem excessivamente da experiência do projetista. O resultado destas tarefas, entretanto, influe significativamente no desempenho final do sistema. Considere, por exemplo, duas situações extremas: 1) nenhuma decomposição e 2) pulverização funcional devido à decomposição e distribuição excessivas. A primeira situação sofre dos inconvenientes dos sistemas centralizados. A segunda, por outro lado, além da possível sub-utilização da capacidade de processamento dos módulos físicos da estrutura, paga excessiva tributação ao serviço de comunicação. A solução é, portanto, de compromisso e sensivelmente dependente da aplicação.

Pelo exposto acima, pode-se concluir que a definição de uma metodologia algorítmica para a programação GMB* coloca-se como um objetivo a ser atingido. Como um primeiro passo em direção a tal objetivo, propõe-se a exploração sistemática do modelo e metodologia

apresentados, através da utilização, tanto quanto possível exaustiva, deste instrumental na especificação e implementação de sistemas com restrições em tempo-real. Avalia-se que a exploração baseada na experimentação dará subsídios, sobretudo ao identificar discrepâncias com a realidade, à formação de um ferramental teórico e prático que contribuirá, em um ciclo contínuo, para o refinamento das técnicas de projeto e implementação de sistemas distribuídos de controle digital.

É interessante observar que a máquina GMB* distribuída, por permitir a avaliação de diferentes estratégias de distribuição de um mesmo modelo GMB* aplicativo, revela-se como uma ferramenta útil ao aprimoramento da metodologia proposta. Dado um modelo GMB* aplicativo, o programador, além de poder configurar fisicamente a máquina GMB*, com o intuito de explorar determinada topologia, pode, também, explorar e avaliar diferentes estratégias de distribuição do programa aplicativo ao modesto custo de recarregamento da estrutura.

Paralelamente ao trabalho de definição da máquina GMB* distribuída, construiu-se, em ambiente de laboratório, um processo-piloto para a sua aplicação. Este processo-piloto constitui-se de um ferromodelo organizado de forma a representar um sistema metroviário. O ferromodelo, conjuntamente com a máquina GMB*, estabelece um ambiente experimental de ensino e pesquisa na área de controle de processos por computador e constitui o assunto do próximo capítulo.

CAPÍTULO 4

APLICAÇÃO DA MÁQUINA GMB*:
CONTROLE DE UM PROCESSO METROVIÁRIO EM ESCALA REDUZIDA

4.1. INTRODUÇÃO

Neste capítulo é desenvolvido um exemplo ilustrativo da aplicação da máquina GMB* distribuída ao controle de processos. O processo escolhido para o exemplo, aproveitando experiência acumulada em diversos trabalhos conjuntos com a Companhia do Metropolitano de São Paulo, é um ferromodelo especialmente construído para representar um sistema metroviário. Este ferromodelo, conjuntamente com a máquina GMB* distribuída, pretende estabelecer um ambiente experimental de ensino e pequisa na área de controle de processos por computador.

A sistemática adotada na construção do exemplo ilustrativo consistiu no estudo e adaptação para um ambiente distribuído da solução adotada pelo Metropolitano de São Paulo. Neste sentido, o presente capítulo está dividido em duas partes: uma primeira que descreve a filosofia de operação do metropolitano de São Paulo e outra que apresenta o ferromodelo e define uma estrutura hierárquica, baseada no conceito de máquina GMB* distribuída, a ser utilizada no controle de sua operação.

4.2. O METROPOLITANO DE SÃO PAULO

O Metropolitano de São Paulo é constituído por duas linhas metroferroviárias independentes que cruzam a cidade de São Paulo em direções perpendiculares: a linha Norte/Sul, que interliga Santana (norte) ao Jabaquara (sul), e a linha Leste/Oeste, em fase de implantação, que, quando finalizada, interligará a Penha (leste) à Lapa (oeste). Por sua vez, cada uma das linhas do Metrô é composta por duas vias férreas paralelas, pelas quais, em operação normal, os trens trafegam em sentidos opostos. Nos extremos e ao longo destas vias distribuem-se estações de embarque e desembarque de passageiros.

Na linha Norte/Sul (figura 4.2.1) existem desvios de interligação que unem as vias. Tais desvios, ao permitirem a passagem de trens de uma via para outra, garantem a operação do sistema mesmo em face à obstrução parcial das vias. Ainda nesta linha, à altura das estações Ana-Rosa e Tiradentes, existem locais, denominados terminais, especialmente reservados ao estacionamento de trens. Estes terminais, além de permitirem o despacho e recolhimento das composições, permitem, também, manobras de retorno. Esta última facilidade define, em adição ao anel de tráfego externo definido pelas estações extremas, possibilidades alternativas de rotas de circulação de trens.

Existe na linha Norte/Sul, nas proximidades da estação Jabaquara, um pátio de manobras e oficinas, de onde os trens podem ser despachados e para onde podem ser recolhidos para a manutenção, limpeza, estacionamento, testes e treinamento.

Independentemente da linha considerada, as vias do Metrô de São Paulo são segmentadas em trechos independentes, cada um dotado de sensor de presença de trem, individualmente acoplados a um circuito de via. A função do circuito de via é converter o nível de

desempenho (perfil de velocidade) gerado pelo Centro de Controle Operacional (CCO) em velocidade real de deslocamento das composições.

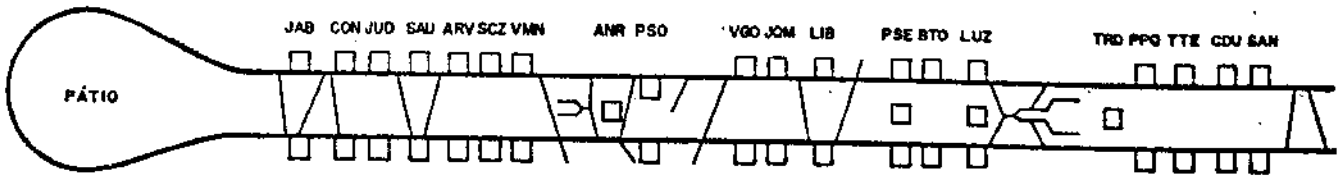


FIG.4.2.1 - ESQUEMA DA LINHA NORTE/SUL DO METROPOLITANO DE SÃO PAULO

A operação do Metropolitano de São Paulo é norteada pela solução de compromisso entre os fatores: demanda de passageiros, conforto e custo. Uma vez que a demanda de passageiros apresenta flutuações significativas ao longo do tempo, o cálculo desta solução de compromisso é efetuado subdividindo-se o horizonte de operação em períodos caracterizados pela constância de demanda de passageiros. Para cada um destes períodos é determinada a solução otimizada em função da demanda, conforto e custo. Esta solução otimizada é denominada trajetória de referência, ou trajetória nominal, e dita, como o próprio nome sugere, a referência a ser seguida pelo sistema para a obtenção da solução de compromisso durante o período considerado. A trajetória de referência estabelece, para cada trem em operação, o horário de chegada e o tempo de de parada em cada estação da linha.

A operação da linha Norte/Sul é automatizada, admitindo-se a intervenção manual de operadores humanos em situações extremas não previstas. Em operação normal, o controle do fluxo de trens é realizado por dois computadores, em operação redundante, localizados no Centro de Controle Operacional (CCO) próximo à estação Paraíso. Consoante /Magalhães 81/ e /Shin-Ting 84/ é possível enquadrar a organização do controle do Metropolitano de São Paulo em uma estrutura hierárquica composta por quatro camadas (figura 4.2.2) : OTIMIZAÇÃO, ADAPTAÇÃO, REGULAÇÃO e IMPLEMENTAÇÃO.

4.2.1. CAMADA DE OTIMIZAÇÃO

A camada de otimização tem por função gerar o programa horário (schedule) ótimo que atenda, dentro das restrições técnicas e operacionais do sistema, a solução de compromisso entre o custo da oferta de transporte e a qualidade do serviço oferecido aos usuários /Cury 79/ /Bergamaschi 82/. Este programa horário define a trajetória de referência que deve ser seguida pelo sistema, estabelecendo para cada trem injetado na linha: a rota a ser seguida, o horário de despacho, o nível de desempenho a ser desenvolvido no trajeto entre estações (nível de desempenho

nominal), o horário teórico de chegada à cada estação do percurso e o tempo de parada em cada uma destas estações (tempo de parada nominal).

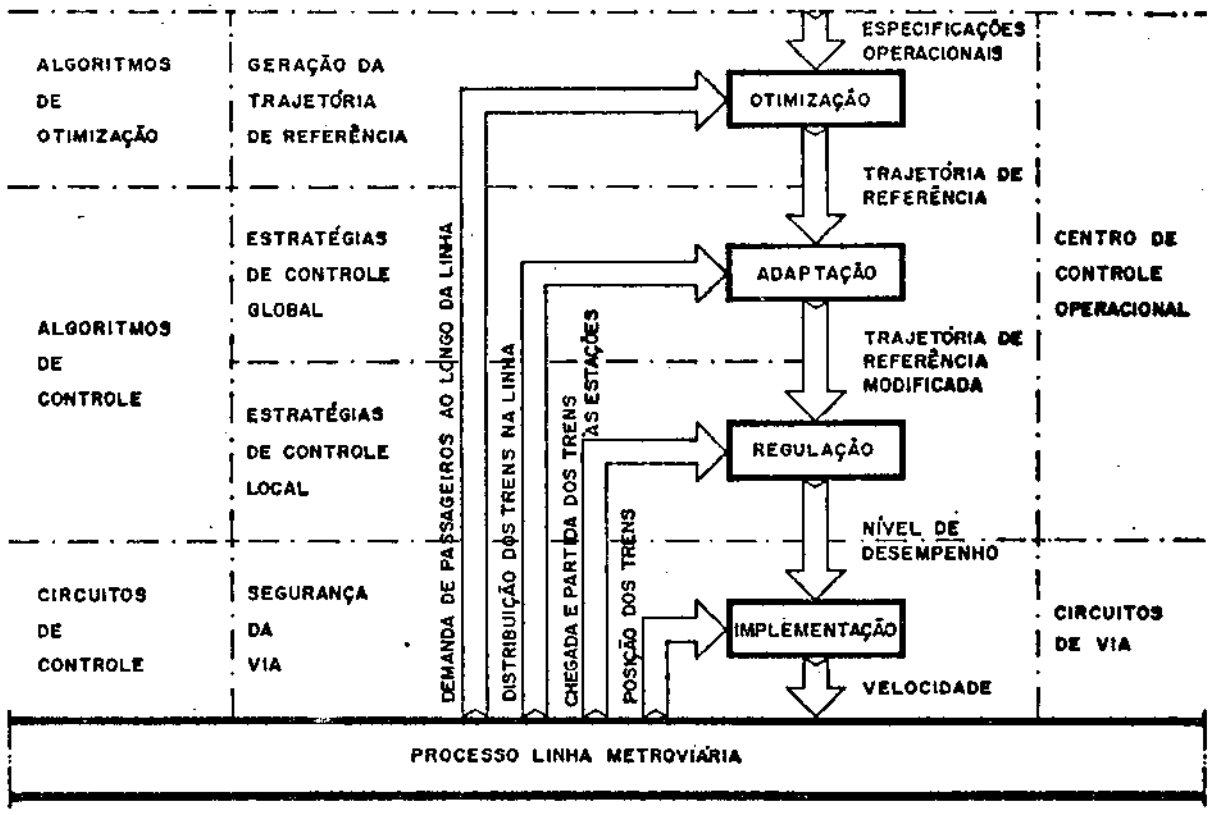


FIG.4.2.2 - ESTRUTURA DO CONTROLE DO METROPOLITANO DE SÃO PAULO

4.2.2. CAMADA DE ADAPTAÇÃO

A camada de adaptação atua no sentido de promover ajustes na trajetória de referência de forma a melhor adequá-la à situações perturbadas, que porventura tenham sido imperfeitamente incorporadas no modelo utilizado na fase de otimização. Os ajustes da trajetória de referência tentam levar em conta a distribuição dos trens ao longo da linha e os limites de atuação da camada de regulação. Distingue-se na camada de adaptação do Metrô de São Paulo três estratégias de adaptação: ESTRATÉGIA DE REVISÃO DO SISTEMA, ESTRATÉGIA DE MODIFICAÇÃO DO HORÁRIO e ESTRATÉGIA DE CONFLUÊNCIA EM INTERTRAVAMENTO /Metrô 75/ /Metrô 76/. /Metrô 80/ /De Martino 83/.

A ESTRATÉGIA DE REVISÃO DO SISTEMA atua periodicamente em uma frequência fixa pré-estabelecida, sendo constituída por duas outras estratégias de controle: ESTRATÉGIA DE DEFASAGEM NO HORÁRIO DO SISTEMA e ESTRATÉGIA DE DISTRIBUIÇÃO DE INTERVALOS.

A ESTRATÉGIA DE DEFASAGEM NO HORÁRIO DO SISTEMA calcula um valor de defasagem a ser adicionado aos horários teóricos dos trens (horário de despacho e horário teóricos de chegada às estações), de

forma a gerar temporariamente uma nova trajetória de referência (trajetória defasada), que em situações perturbadas ,dentro da faixa de velocidades permitida no sistema, possa ser seguida pelos trens. A determinação do valor de defasagem a ser aplicado é efetuada considerando-se o atraso de cada trem registrado na última estação por onde passou. Para um trem atrasado, verifica-se, dentro do limite de velocidade vigente, se o mesmo ainda tem capacidade para alcançar o próximo ponto crítico (saída de terminal ou final de via) dentro do horário previsto. Em se verificando a impossibilidade do cumprimento do horário, o atraso é considerado irrecuperável. O valor de defasagem a ser aplicado é computado como a média aritmética de todos os atrasos irrecuperáveis existentes na via no instante de aplicação da estratégia. Caso não exista trem em situação de atraso e já se tenha um defasagem positiva em vigor, aplica-se gradualmente uma defasagem negativa, visando o restabelecimento da trajetória de referência original definida pela otimização.

A ESTRATÉGIA DE DISTRIBUIÇÃO DE INTERVALOS procura manter os trens uniformemente distribuídos ao longo da linha. Neste sentido, uma vez detetado um trem com um atraso significativo, esta estratégia atua distribuindo tal atraso entre as composições que se encontrem à frente do trem problema. Esta distribuição é efetuada dentro do trecho compreendido entre a posição atual do trem problema e o próximo ponto crítico (saída terminal ou final de via), tendo por limite máximo dez estações à frente. O atraso a ser distribuído a um trem em cada estação do trecho em consideração é computado consoante o seguinte algoritmo: 1) tomando-se como referência o trem problema, estima-se os atrasos possíveis de serem recuperados pelo mesmo (tempo passível de ser economizado em relação aos valores nominais, se o trem andar com nível de desempenho máximo e tempo de parada mínimo) a partir de cada estação do trecho até o final do trecho; 2) para cada estação do trecho calcula-se o atraso recuperável viável: mínimo entre o atraso possível de ser recuperado a partir daquela estação e o atraso de trem problema detetado na última estação por onde passou; 3) Associa-se a cada estação um valor de atraso, a ser posteriormente distribuído aos trens, dado pela divisão do atraso recuperável viável da associado à estação pelo número de trens que passarão por ela, até, e incluindo, o trem problema; 4) o valor do atraso a ser distribuído a um trem em uma estação é então calculado mediante a multiplicação do atraso associado à estação por um fator de ponderação da ordem de passagem do trem pela mesma. Este fator de ponderação é, para o primeiro trem, igual ao número total de trens que passarão pela estação, decrescendo unitariamente até ser igual a um para o trem problema (o último a passar pela estação). Os atrasos a serem distribuídos aos trens são somados aos seus horários teóricos, gerando os chamados horários operacionais.

A ESTRATÉGIA DE MODIFICAÇÃO DO HORÁRIO tem como finalidade corrigir os horários teóricos dos trens que eventualmente saiam das zonas de confluência (saída de terminal) fora da sequência escalada. A correção dos horários dos trens é efetuada pela adição de uma compensação nos horários teóricos dos trens envolvidos (trens que ressequenciam) de forma a oficializar a nova sequência. Esta compensação é dada pela subtração dos horários previstos de chegada às estações da zona de confluência (Ana-Rosa e Tiradentes na linha Norte/Sul) dos dois trens envolvidos. Tal compensação é positiva para o trem que se atrasou e negativa para o trem que se adiantou. O

resultado líquido desta operação é a troca dos horários dos trens oficializando a nova sequência.

A ESTRATÉGIA DE CONFLUÊNCIA EM INTERTRAVAMENTOS prevê a existência de trens competindo pelo uso simultâneo de rotas conflitantes nas zonas de confluência (saída de terminal ou final de via). Ao ser detetado este tipo de competição, esta estratégia intervém, ajustando os tempos de chegada dos trens à zona de confluência, de modo a minimizar a interferência. Os trens envolvidos neste tipo de perturbação estão sujeitos às seguintes ações por parte desta estratégia: 1) atraso do trem da linha através do aumento de seus tempos de parada em estações, fazendo-o atingir o ponto de requisição de rota mais tarde; 2) retardamento do despacho do trem do terminal, atrasando a sua requisição de rota; 3) adiantamento do despacho do trem do terminal, fazendo-o requisitar a rota mais cedo.

4.2.3. CAMADA DE REGULAÇÃO

A camada de regulação, dentro de seus limites de atuação, procura manter o sistema operando em torno da trajetória de referência definida pela camada de otimização e modificada pela camada de adaptação. A camada de regulação tem ação restrita às estações e é disparada por dois eventos bem definidos: 1) Chegada de trem em estação (evento portas abertas), quando então define o tempo de parada do trem na estação e efetua um primeiro ajuste do nível de desempenho (perfil de velocidade) a ser utilizado no trajeto até a próxima estação; 2) Partida de trem de estação (evento portas fechadas), quando então, considerando o tempo real de parada do trem e, portanto, eventuais perturbações ocorridas devido ao fluxo de passageiros, define o nível de desempenho a ser efetivamente utilizado no trajeto até a próxima estação. Tanto o ajuste do tempo de parada quanto o ajuste do nível de desempenho são efetuados de forma a minimizar a diferença entre a previsão do horário de chegada do trem à próxima estação e o horário de chegada definido pela trajetória de referência.

4.2.4. CAMADA DE IMPLEMENTAÇÃO

Na camada de implementação tem-se os circuitos de via que, além de servirem como interfaces entre os computadores do CCO e a linha, constituem-se, também, em elementos controladores responsáveis pela segurança da via. No Metrô de São Paulo a segurança da via é traduzida por um valor de espaçamento mínimo admissível entre dois trens consecutivos. Os circuitos de via garantem automaticamente tal espaçamento, projetando restrições de velocidade aos trechos posteriores a um trecho ocupado por um trem. Esta característica é denominada sombra e assegura a ausência de colisões na linha, uma vez que um trem na sombra de outro não mais segue o perfil de velocidade definido pela regulação, mas sim, o que lhe for permitido por seu predecessor.

4.3. O SISTEMA METROVIÁRIO EM ESCALA REDUZIDA

Inspirado no Metropolitano de São Paulo, construiu-se um ferromodelo, denominado processo metroviário em escala reduzida, que reproduz, em um ambiente de laboratório, as pricipais característica de um sistema metroviário. Este ferromodelo procura definir o nível do processo de uma ferramenta de apoio ao ensino e pesquisa na área de controle de processos por computador. O nível de controle desta ferramenta será constituído por uma máquina GMB* distribuída.

Nesta seção, como exemplo ilustrativo da potencialidade da máquina GMB* distribuída como ferramenta de síntese de sistemas de controle por computador, é apresentada uma solução distribuída/hierárquica para o controle da operação do processo metroviário em escala reduzida. Tal solução segue a filosofia do metropolitano de São Paulo, sendo, porém, implementada no ambiente oferecido pela máquina GMB* distribuída.

Segue abaixo uma descrição do processo metroviário em escala reduzida. Após esta descrição, apresenta-se a solução adotada para o controle de sua operação.

4.3.1. O PROCESSO METROVIÁRIO EM ESCALA REDUZIDA

O processo metroviário em escala reduzida é constituído por um ferromodelo, cuja linha, aos moldes do Metropolitano de São Paulo, é segmentada em trechos eletricamente independentes, cada um acoplado a um circuito de deteção e alimentação /Delgado 84/.

Para aumentar a flexibilidade do modelo acoplou-se aos circuitos de via um micro-computador, denominado micro-computador de via, que tem por função permitir a configuração funcional da linha (quantidade e localização das estações, simulação do fluxo de passageiros nas estações e definição de trechos terminais), garantir a segurança da via e servir de interface entre o hardware do processo e o nível de controle.

4.3.1.1. A LINHA DO FERROMODELO

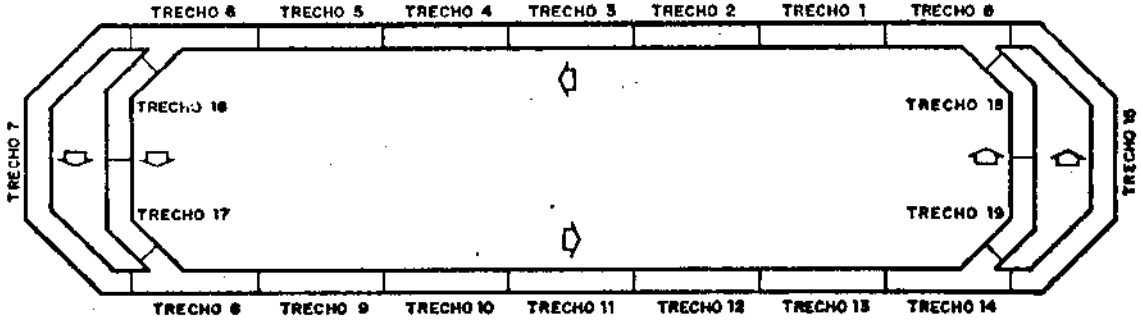


FIG.4.3.1 - A LINHA DO FERROMODELO

A linha do ferromodelo é um percurso fechado de aproximadamente 40 metros, por onde trens elétricos, escala HO, alimentados pelos trilhos, podem circular em apenas uma direção (figura 4.3.1). Para aumentar a complexidade do processo e melhor representar o sistema inspirador, instalou-se na linha do ferromodelo trechos com desvios que, a critério da configuração funcional adotada, permitem a simulação dos terminais Ana-Rosa e Tiradentes, assim como, do pátio de manobras do Jabaquara.

Atualmente, a linha do ferromodelo encontra-se segmentada em 20 trechos independentes, cada um com dois sensores para a deteção de trens ("reed-switch" sem memória acionado por imã preso à carcaça do trem). O primeiro sensor, localizado à entrada do trecho, sinaliza a entrada de trem no trecho e sua consequente saída do trem do trecho anterior; o segundo sensor, localizado a uma distância segura de frenagem do final do trecho, indica que o trem está prestes a abandonar o trecho em questão. Nos trechos com desvio, além dos sensores de deteção de trens, existe um sensor adicional, acionado pela parte móvel do desvio, para indicar o estado de alinhamento do mesmo.

O controle da alimentação de um trecho, em última instância o controle da velocidade do trem que ocupa o trecho, e a monitoração dos sensores de ocupação e alinhamento de desvio é efetuado por um micro-computador. Um circuito eletrônico, denominado circuito de deteção e alimentação, faz a interface entre cada trecho da linha e este micro-computador.

4.3.1.2. CIRCUITO DE DETEÇÃO E ALIMENTAÇÃO

O circuito de deteção e alimentação, ou simplesmente CDA, é a interface entre cada trecho da linha e o chamado micro-computador de via (figura 4.3.2). O circuito de deteção e alimentação, como o seu próprio nome sugere, processa dois tipos distintos de informação: deteção e alimentação.

No que se refere à deteção, tem-se em cada CDA dois elementos armazenadores (flip-flop), cada um associado a um sensor (reed-switch), para a memorização do estado de ocupação do trecho. Tais elementos permitem que a informação de ocupação do trecho esteja sempre disponível para leitura pelo micro-computador de via. Um trem ao entrar em um trecho gatilha o primeiro elemento armazenador, informando de sua entrada no referido trecho, e apaga os elementos armazenadores do trecho anterior, informando de sua saída do mesmo. Ao passar pelo segundo sensor, o trem gatilha o segundo elemento armazenador, informando ao micro-computador de via que se encontra prestes a abandonar o trecho. Já a deteção do estado de alinhamento dos desvios, por possuir memória própria, dispensa a necessidade de elemento armazenador no CDA. Tal deteção é realizada por uma chave mecânica de duas posições acionada pela parte móvel do desvio.

No que diz respeito à alimentação do trecho, o CDA converte o código binário de velocidade (3 bits) enviado pelo micro-computador de via em onda de tensão pulsada de largura variável. Esta modulação por largura de pulso permite ao micro-computador de via definir 8 níveis distintos de velocidade para os trens. No sistema em questão

convencionou-se que o código 0 corresponde à situação de repouso e o código 7 à velocidade máxima.

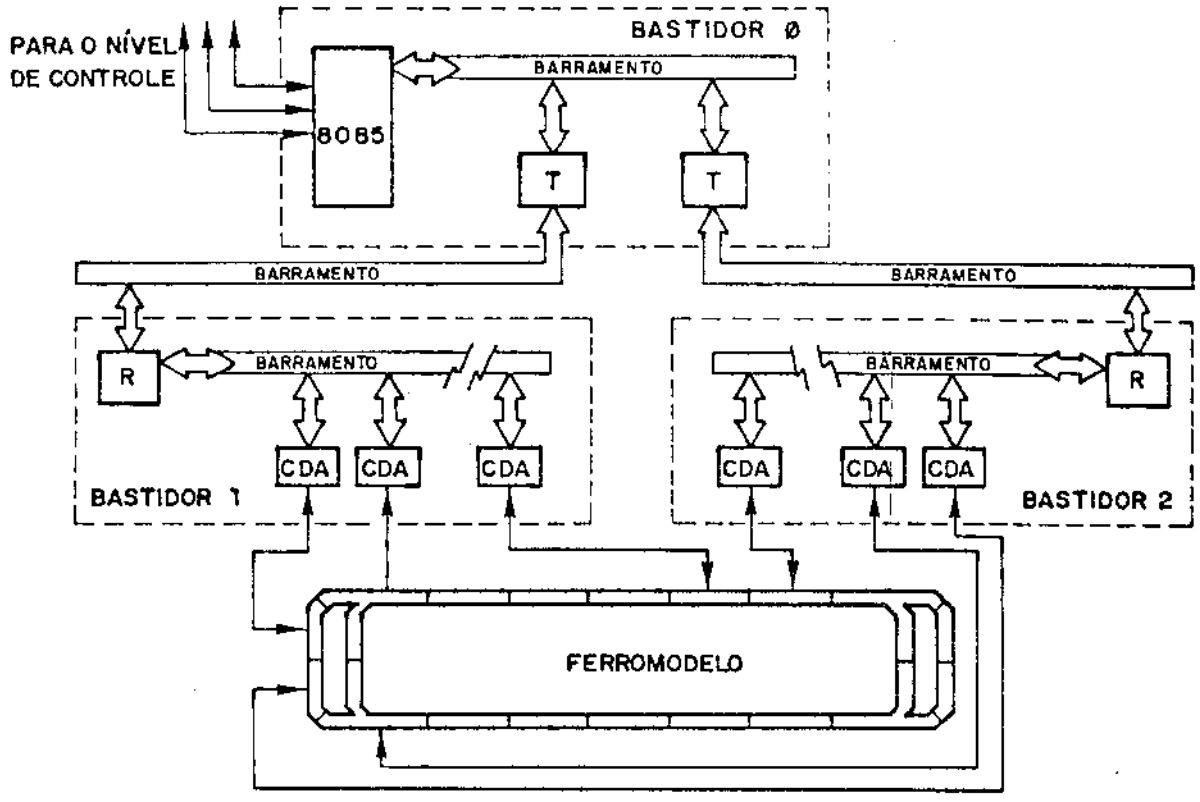


FIG.4.3.2 - INTERCONEXÃO DA LINHA DO FERROMODELO COM O MICRO-COMPUTADOR DE VIA

Um CDA associado a um trecho com desvio permite, ainda, que o micro-computador de via comande o seu alinhamento: Dois mono-estáveis, um sensível à borda de subida outro á borda de descida, chaveiam o relé de posicionamento do desvio consoante a mudança de estado de uma das linhas do barramento de dados do micro-computador de via.

Para minimizar o tempo de acesso do micro-computador de via aos CDA's, estes foram acoplados diretamente ao barramento do micro-computador. Esta conexão faz com que os CDA's sejam considerados pelo micro-computador de via como dispositivos de entrada e saída.

4.3.1.3. **MICRO-COMPUTADOR DE VIA**

Para aumentar a flexibilidade do processo e reproduzir com maior fidelidade o sistema real, acoplou-se à linha do ferromodelo um micro-computador, denominado micro-computador de via, em cuja programação distinguem-se quatro blocos funcionais (figura 4.3.3): INTERFACE COM A LINHA, SEGURANÇA DA VIA, CONFIGURAÇÃO DA LINHA, MONITOR e SERVIÇO DE COMUNICAÇÃO.

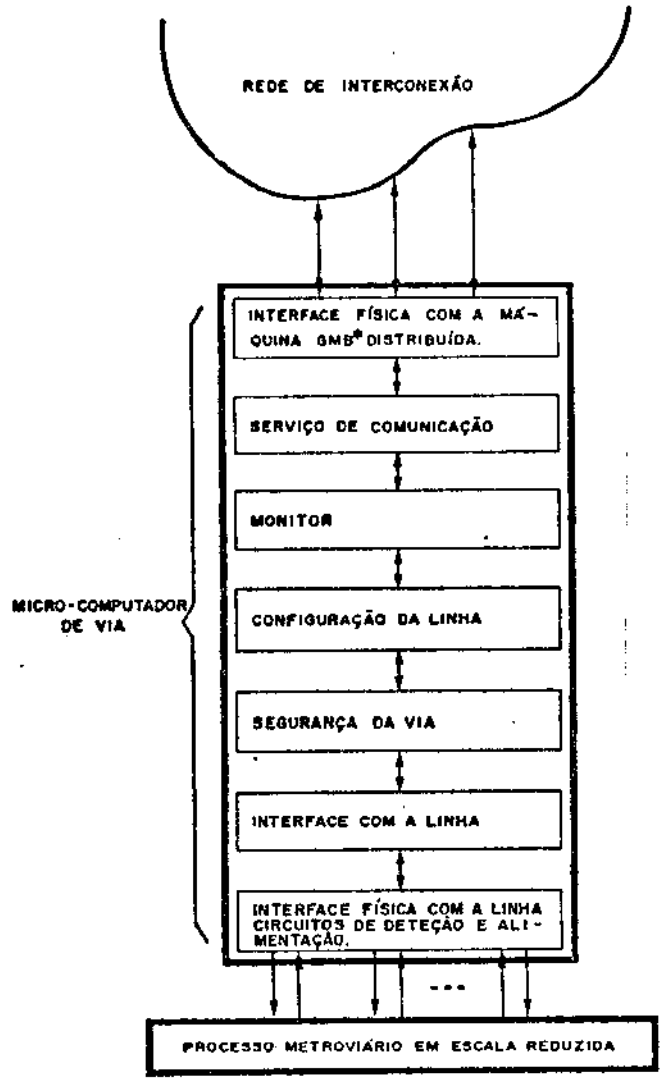


FIG.4.3.3 - ORGANIZAÇÃO FUNCIONAL DO MICRO-COMPUTADOR DE VIA

O bloco INTERFACE COM A LINHA reúne os programas de acesso e gerenciamento do hardware dos CDA's.

A SEGURANÇA DA VIA refere-se à camada de implementação da hierarquia do controle do Metrô de São Paulo (figura 4.2.2) e, como discutido anteriormente, está associado ao conceito de sombra de velocidades. Atualmente no processo metroviário em escala reduzida tem-se implementada uma versão simplificada deste conceito: um trecho ocupado por um trem projeta para o trecho anterior a velocidade zero. Tal projeção está restrita à segunda parte do trecho anterior (somente após o segundo sensor). Apesar de simplificado, este mecanismo garante a ausência de colisões na linha.

O bloco funcional CONFIGURAÇÃO DA LINHA permite ao usuário ajustar algumas características da via de forma a melhor adequá-la à representação deste ou daquele processo metroviário. Enquanto ferramenta de avaliação e testes de estratégias de controle, este

bloco funcional enriquece a potencialidade do sistema por permitir a representação de processos metroviários distintos utilizando a mesma infra-estrutura física. É possível atualmente configurar: a quantidade e a localização das estações, o fluxo de passageiros em cada estação e os diferentes anéis de circulação do sistema (definição de terminais e/ou trechos de retorno).

O MONITOR é o elemento terminal da comunicação com a máquina GMB* do nível do controle, sendo capaz de gerar e interpretar mensagens segundo o formalismo GMB*. Este elemento também é responsável pela monitoração do estado da via e pela movimentação dos trens, utilizando, para tanto, as funções dos blocos configuração da linha, segurança da via e interface com a linha.

O SERVIÇO DE COMUNICAÇÃO é a interface entre a máquina GMB*, responsável pelas camadas superiores da hierarquia de controle, e o micro-computador de via. Tal serviço de comunicação é uma reprodução do serviço de comunicação da máquina GMB* uni-processadora discutida na seção 3.2.2 - capítulo 3.

4.3.2. O MODELO GMB* DO PROCESSO METROVIÁRIO EM ESCALA REDUZIDA

Como exemplo ilustrativo da aplicação da máquina GMB* distribuída no controle de um processo, tomar-se-á o processo metroviário em escala reduzida, como discutido acima, com a configuração apresentada na figura 4.3.4.

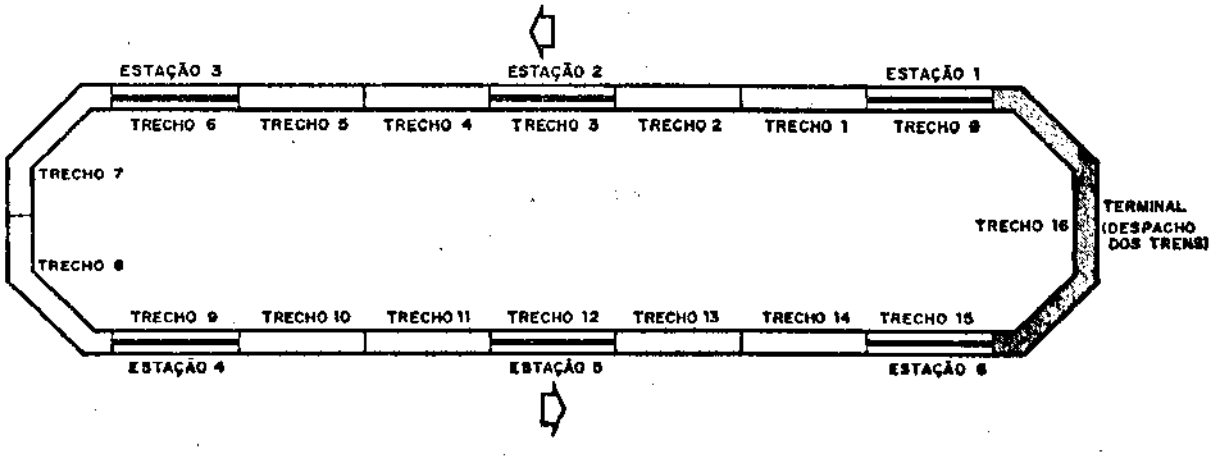


FIG.4.3.4 - CONFIGURAÇÃO DA LINHA: UM EXEMPLO

Nas figuras 4.3.5 e 4.3.6 são apresentados, respectivamente, os domínios do controle e dos dados do modelo GMB* do processo metroviário em escala reduzida com a configuração proposta. Convém ressaltar que este modelo descreve, para um dado período de operação, apenas o comportamento do sistema em regime, dispensando, por não serem relevantes para esta discussão, a descrição das etapas de início e término da operação do sistema. Considera-se, entretanto, que a trajetória de referência do sistema foi gerada a priori pela camada de otimização e para qualquer efeito já se encontra carregada no sistema. Em particular, para simplificar a

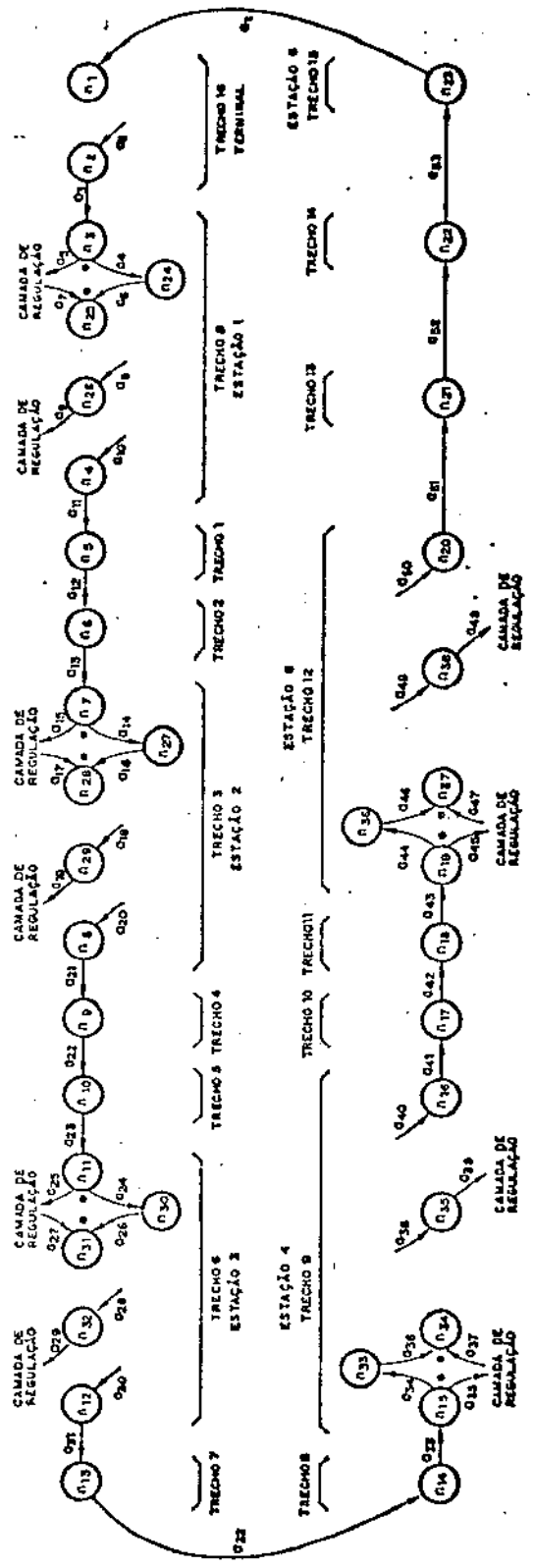


FIG.4.3.5 - MODELO GMB* DO PROCESSO METROVIÁRIO EM ESCALA REDUZIDA: DOMÍNIO DO CONTROLE

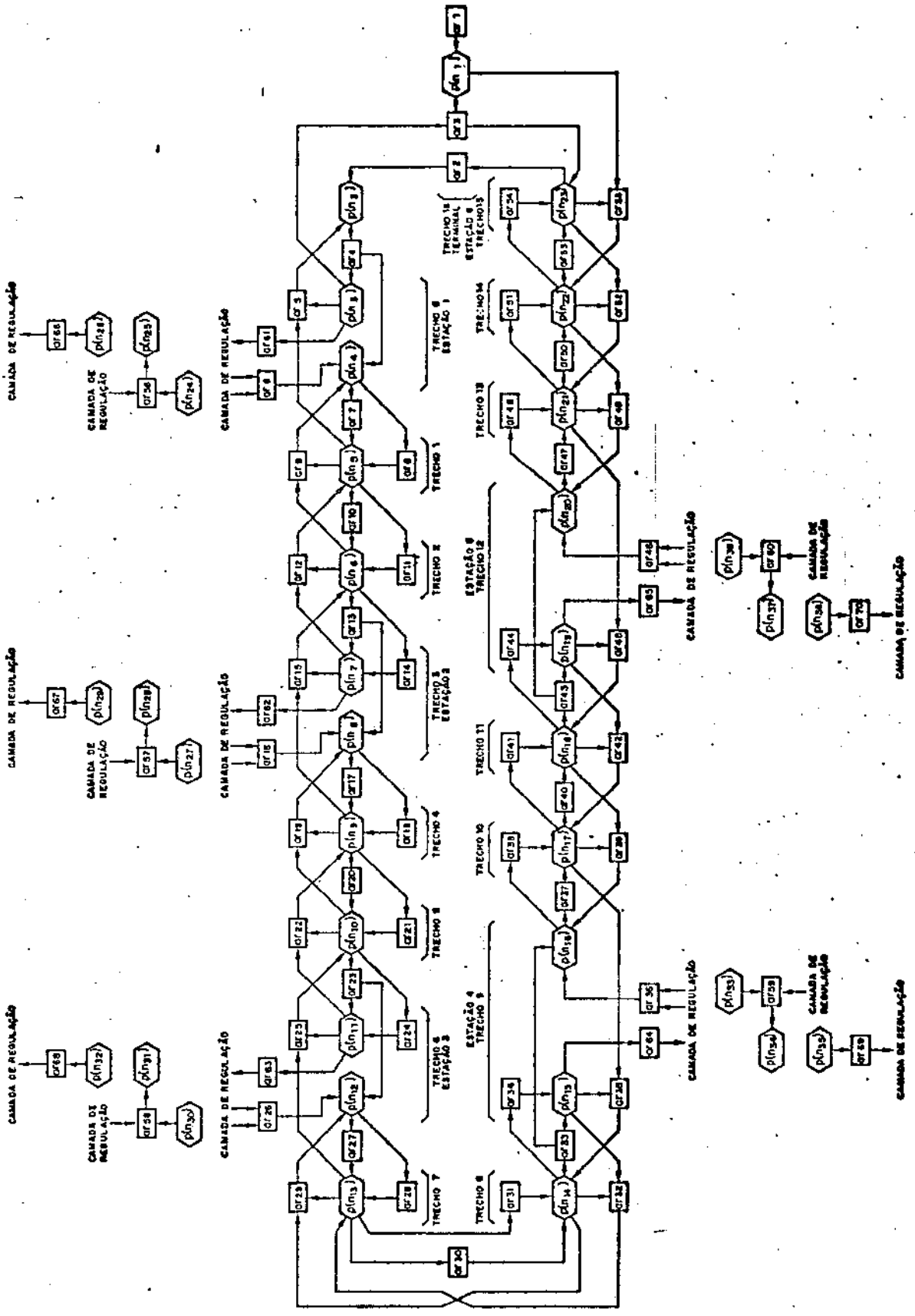


FIG.4.3.8 - MODELO GMB* DO PROCESSO METROVIÁRIO EM ESCALA REDUZIDA: DOMÍNIO DOS DADOS

apresentação, supõe-se que todos os trens em uma mesma estação possuem o mesmo valor de tempo de parada nominal e que este valor encontra-se disponível na mesma. Da mesma forma, supõe-se que na estação tem-se disponível o valor do nível de desempenho nominal a ser utilizado pelos trens no trajeto até a próxima estação.

No domínio da interpretação, dada a equivalência funcional que alguns elementos do domínio de dados apresentam entre si, evitou-se a descrição exaustiva, e por vezes enfadonha, da interpretação de todos os elementos presentes no domínio de dados. Em vez disto, agrupou-se funcionalmente os elementos deste domínio e, dos grupos assim formados, tomou-se um representante, cuja interpretação foi explicitada. As interpretações dos outros elementos podem ser deduzidas a partir da interpretação do seu equivalente funcional e pela análise dos domínios do controle e dados (fig.4.3.5 e fig.4.3.6). A equivalência funcional reflete o fato do sistema apresentar pontos de transformação distintos com processamento idêntico e sobre os mesmos tipos de dados. Para os processadores a equivalência funcional define interpretações com estruturas algorítmicas idênticas, diferindo apenas nos identificadores dos armazenadores acessados e nos identificadores dos arcos marcados. Para os armazenadores a equivalência funcional define que os mesmos possuem a mesma estrutura e comportam o mesmo tipo de dados, diferenciando-se em suas identificações (localização física). Apresenta-se a seguir a tabela de equivalência funcional, sendo então apresentadas as interpretações de um representante de cada grupo funcional.

TABELA DE EQUIVALÊNCIA FUNCIONAL

ARMAZENADORES

TABELA DE DESPACHO DOS TRENS: ar1
IDENTIFICADOR TREM: ar2, ar4, ar7,ar10, ar13, ar17, ar20, ar23, ar27,ar30, ar33, ar37, ar40, ar43, ar47, ar50 e ar53.
ESTADO DO TRECHO: ar3, ar5, ar9, ar12, ar15, ar19, ar22, ar25, ar29, ar32, ar35, ar39, ar42, ar45, ar49, ar52 e ar55.
NÍVEL DESEMPENHO - MOVIMENTO DO TREM: ar8, ar11, ar14, ar18, ar21, ar24, ar28, ar31, ar34, ar38, ar41, ar44, ar48, ar51 e ar54.
NÍVEL DESEMPENHO - CAMADA DE REGULAÇÃO: ar6, ar16, ar26, ar36 e ar46.
ACUMULADOR AUXILIAR DA GERACÃO DO TEMPO DE PARADA NA ESTAÇÃO: ar56, ar57, ar58, ar59 e ar60.
CHEGADA DE TREM NA ESTAÇÃO: ar61, ar62, ar63, ar64 e ar65.
HORÁRIO DE PARTIDA DE TREM: ar66, ar67, ar68, ar69 e ar70.

PROCESSADORES

TRECHO TERMINAL - ENTRADA (ANTES SEGUNDO SENSOR): p(n1).
TRECHO TERMINAL - SAÍDA (APÓS SEGUNDO SENSOR): p(n2).
TRECHO ESTAÇÃO À SAÍDA DO TERMINAL - ENTRADA (ANTES SEGUNDO SENSOR): p(n3).
TRECHO ESTAÇÃO À SAÍDA DO TERMINAL - SAÍDA (APÓS SEGUNDO SENSOR): p(n4).
TRECHO ESTAÇÃO À ENTRADA DO TERMINAL: p(n23).

TRECHO ESTAÇÃO COMUM - ENTRADA (ANTES SEGUNDO SENSOR): p(n7), p(n11), p(n15) e p(n19).
TRECHO ESTAÇÃO COMUM - SAÍDA (APÓS SEGUNDO SENSOR): p(n8), p(n12), p(n16) e p(n20).
TRECHO SAÍDA DE ESTAÇÃO: p(n5), p(n9), p(n13), p(n17) e p(n21).
TRECHO ENTRADA DE ESTAÇÃO: p(n6), p(n10), p(n14), p(n18) e p(n22).
SIMULADOR FLUXO DE PASSAGEIROS: p(n24), p(n27), p(n30), p(n33) e p(n36).
GERADOR TEMPO REAL DE PARADA: p(n25), p(n28), p(n31), p(n34) e p(n37).
ACIONADOR PARTIDA TREM: p(n26), p(n29), p(n32), p(n35) e p(n38).

DOMÍNIO DA INTERPRETAÇÃO

INTERPRETAÇÃO DOS ARMAZENADORES

ar1 - Tabela de despacho dos trens
ar1::=(<horário de despacho>)
<horário de despacho>::=<HORA>

ar2 - Identificador trem
ar2::=<IDENTIFICADOR TREM>

ar3 - Estado do trecho
ar3::=<estado do trecho>
<estado do trecho>::=<OCUPADO>/<DESOCUPADO>

ar6 - Nível de desempenho - camada de regulação
ar6::=<NÍVEL DE DESEMPENHO>

ar8 - Nível de desempenho - movimento do trem
ar8::=<NÍVEL DE DESEMPENHO>

ar56 - Acumulador auxiliar da geração do tempo de parada na estação
ar56::=<HORA><HORA>

ar61 - Chegada de trem na estação
ar61::=<<IDENTIFICADOR TREM><HORA>>

ar66 - Horário partida trem
ar66::=<HORA>

INTERPRETAÇÃO DOS PROCESSADORES

```
INTERPRETAÇÃO p(n1) ;
BEGIN
 WHILE /* o trem não alcançar o primeiro sensor */ DO
 BEGIN
 END ;
 /* ESCREVE(ar3,DESTRUTIVO,OCUPADO) */ ;
 /* ESCREVE(ar55,DESTRUTIVO,DESOCUPADO) */ ;
 WHILE /* o trem não alcançar o segundo sensor */ DO
 BEGIN
 END ;
 /* comande velocidade zero para o trecho */ ;
 /* LE(ar1,CONSUMPTIVO,HORÁRIO$DESPACHO) */ ;
```

```
 /* LE$HORA(HORA$ATUAL) */ ;
 IF /* HORA$ATUAL >= HORÁRIO$DESPACHO */ THEN
 BEGIN
  /* TEMPO$DESPACHO:= 0 */
 END
 ELSE
 BEGIN
  IF /* HORA$ATUAL < HORÁRIO$DESPACHO */ THEN
  BEGIN
   /* TEMPO$DESPACHO:= HORÁRIO$DESPACHO - HORA$ATUAL */
  END
 END ;
 /* ATIVA$ESPERA(a2,1,TEMPO$DESPACHO) */
END.

INTERPRETAÇÃO p(n2) ;
BEGIN
 REPEAT
 BEGIN
  /* LE(ar5,NÃO-CONSUMPTIVO,OCUPAÇÃO$PRÓXIMO$TRECHO) */
 END
 UNTIL /* próximo trecho estar desocupado */ ;
 /* comande velocidade 1 para este e para o próximo trecho */ ;
 /* LE(ar2,CONSUMPTIVO,IDTREM) */ ;
 /* ESCREVE(ar4,DESTRUTIVO,IDTREM) */ ;
 /* MARCA(a3,1) */
END.

INTERPRETAÇÃO p(n3) ;
BEGIN
 WHILE /* o trem não alcançar o primeiro sensor */ DO
 BEGIN
 END ;
 /* ESCREVE(ar5,DESTRUTIVO,OCUPADO) */ ;
 /* ESCREVE(ar3,DESTRUTIVO,DESOCUPADO) */ ;
 WHILE /* o trem não alcançar o segundo sensor */ DO
 BEGIN
 END ;
 /* comande velocidade zero para o trecho */ ;
 /* LE(ar4,NÃO-CONSUMPTIVO,IDTREM) */ ;
 /* LE$HORA(HORA$ATUAL) */ ;
 /*  concatene as variáveis IDTREM e HORA$ATUAL formando a  variável
 INFORMAÇÃO$CHEGADA$TREM */ ;
 /* ESCREVE(ar61,DESTRUTIVO,INFORMAÇÃO$CHEGADA$TREM) */ ;
 /* MARCA(a5,1) */ ;
 /* MARCA(a4,1) */
END.

INTERPRETAÇÃO p(n4) ;
BEGIN
 REPEAT
 BEGIN
  /* LE(ar9,NÃO-CONSUMPTIVO,OCUPAÇÃO$PRÓXIMO$TRECHO) */
 END
 UNTIL /* próximo trecho estar desocupado */ ;
 /* LE(ar8,CONSUMPTIVO,NÍVEL$DESEMPENHO) */ ;
 /*  calcule a velocidade a ser entregue ao trem até o próximo  tre-
 cho */ ;
```

```
 /* comande esta velocidade para este e para o próximo trecho */ ;
 /* ESCREVE(ar8,DESTRUTIVO,NÍVEL$DESEMPENHO) */ ;
 /* LE(ar4,CONSUMPTIVO,IDTREM) */ ;
 /* ESCREVE(ar7,DESTRUTIVO,IDTREM) */ ;
 /* MARCA(a11,1) */
END.

INTERPRETAÇÃO p(n5) ;
BEGIN
 WHILE /* o trem não alcançar o primeiro sensor */ DO
 BEGIN
 END ;
 /* ESCREVE(ar9,DESTRUTIVO,OCUPADO) */ ;
 /* ESCREVE(ar5,DESTRUTIVO,DESOCUPADO) */ ;
 /* LE(ar8,NÃO-CONSUMPTIVO, NÍVEL$DESEMPENHO) */ ;
 /*  calcule a velocidade  a ser entregue ao trem até o segundo sen-
 sor */ ;
 /* comande esta velocidade para o trecho */ ;
 WHILE /* o trem não alcançar o segundo sensor */ DO
 BEGIN
 END ;
 /* LE(ar12,NÃO-CONSUMPTIVO,OCUPAÇÃO$PRÓXIMO$TRECHO) */ ;
 IF /* próximo trecho está ocupado */ THEN
 BEGIN
  /* comande velocidade zero para o trecho */ ;
  REPEAT
  BEGIN
   /* LE(ar12,NÃO-CONSUMPTIVO,OCUPAÇÃO$PRÓXIMO$TRECHO) */
  END
  UNTIL /* próximo trecho estar desocupado */
 END ;
 /* LE(ar8,CONSUMPTIVO,NÍVEL$DESEMPENHO) */ ;
 /* calcule a velocidade a ser entregue ao trem até o próximo   tre-
 cho */ ;
 /* comande esta velocidade para este e para o próximo trecho */ ;
 /* ESCREVE(ar11,DESTRUTIVO,NÍVEL$DESEMPENHO) */ ;
 /* LE(ar7,CONSUMPTIVO,IDTREM) */ ;
 /* ESCREVE(ar10,DESTRUTIVO,IDTREM) */ ;
 /* MARCA(a12,1) */
END.

INTERPRETAÇÃO p(n8) ;
BEGIN
 WHILE /* o trem não alcançar o primeiro sensor */ DO
 BEGIN
 END ;
 /* ESCREVE(ar12,DESTRUTIVO,OCUPADO) */ ;
 /* ESCREVE(ar9,DESTRUTIVO,DESOCUPADO) */ ;
 /* LE(ar11,NÃO-CONSUMPTIVO,NÍVEL$DESEMPENHO) */ ;
 /* calcule a velocidade a ser entregue ao trem até o segundo   sen-
 sor */ ;
 /* comande esta velocidade para o trecho */ ;
 WHILE /* o trem não alcançar o segundo sensor */ DO
 BEGIN
 END ;
 /* LE(ar15,NÃO-CONSUMPTIVO,OCUPAÇÃO$PRÓXIMO$TRECHO) */ ;
 IF /* próximo trecho está ocupado */ THEN
 BEGIN
```

```
  /* comande velocidade zero para o trecho */ ;
  REPEAT
  BEGIN
   /* LE(ar15,NÃO-CONSUMPTIVO,OCUPAÇÃO$PRÓXIMO$TRECHO) */
  END
  UNTIL /* próximo trecho estar desocupado */
 END ;
 /* LE(ar11,CONSUMPTIVO,NÍVEL$DESEMPENHO) */ ;
 /* calcule a velocidade a ser entregue ao trem até o próximo   tre-
 cho */ ;
 /* comande esta velocidade para este e para o próximo trecho */ ;
 /* ESCREVE(ar14,DESTRUTIVO,NÍVEL$DESEMPENHO) */ ;
 /* LE(ar10,CONSUMPTIVO,IDTREM) */ ;
 /* ESCREVE(ar13,DESTRUTIVO,IDTREM) */ ;
 /* MARCA(a13,1) */
END.

INTERPRETAÇÃO p(n7) ;
BEGIN
 WHILE /* o trem não alcançar o primeiro sensor */ DO
 BEGIN
 END ;
 /* ESCREVE(ar15,DESTRUTIVO,OCUPADO) */ ;
 /* ESCREVE(ar12,DESTRUTIVO,DESOCUPADO) */ ;
 /* LE(ar14,CONSUMPTIVO,NÍVEL$DESEMPENHO) */ ;
 /* calcule a velocidade a ser entregue ao trem até o segundo   sen-
 sor */ ;
 /*  comande  esta velocidade para o trecho */ ;
 WHILE /* o trem não alcançar o segundo sensor */ DO
 BEGIN
 END ;
 /* comande velocidade zero ao trecho */ ;
 /* LE(ar13,NÃO-CONSUMPTIVO,IDTREM) */ ;
 /* LE$HORA(HORA$ATUAL) */ ;
 /*  concatene as variáveis IDTREM e HORA$ATUAL formando a  variável
 INFORMAÇÃO$CHEGADA$TREM */ ;
 /* ESCREVE(ar62,DESTRUTIVO,INFORMAÇÃO$CHEGADA$TREM) */ ;
 /* MARCA(a15,1) */ ;
 /* MARCA(a14,1) */
END.

INTERPRETAÇÃO p(n8) ;
BEGIN
 REPEAT
 BEGIN
  /* LE(ar19,NÃO-CONSUMPTIVO,OCUPAÇÃO$PRÓXIMO$TRECHO) */
 END
 UNTIL /* próximo trecho estar desocupado */ ;
 /* LE(ar16,CONSUMPTIVO,NÍVEL$DESEMPENHO) */ ;
 /* calcule a velocidade a ser entregue ao trem até o próximo   tre-
 cho */ ;
 /* comande esta velocidade para este e para o próximo trecho */ ;
 /* ESCREVE(ar18,DESTRUTIVO,NÍVEL$DESEMPENHO) */ ;
 /* LE(ar13,CONSUMPTIVO,IDTREM) */ ;
 /* ESCREVE(ar17,DESTRUTIVO,IDTREM) */ ;
 /* MARCA(a21,1) */
END.
```

```
INTERPRETAÇÃO p(n23) ;
BEGIN
 WHILE /* o trem não alcançar o primeiro sensor */ DO
 BEGIN
 END ;
 /* ESCREVE(ar55,DESTRUTIVO,OCUPADO) */ ;
 /* ESCREVE(ar52,DESTRUTIVO,DESOCUPADO) */ ;
 /* LE(ar54,CONSUMPTIVO,NÍVEL$DESEMPENHO) */ ;
 /* calcule a velocidade a ser entregue ao trem até o segundo   sen-
 sor */ ;
 /* comande esta velocidade para o trecho */ ;
 WHILE /* o trem não alcançar o segundo sensor */ DO
 BEGIN
 END ;
 /* comande velocidade zero para o trecho */ ;
 /* LE$HORA(HORA$ATUAL) */ ;
 /*  some  X1  unidades de tempo a HORA$ATUAL (o  valor  X1,  a  ser
 definido durante a implementação, simula o tempo de desembarque dos
 passageiros na estação final).  Coloque o  resultado desta operação
 na variável HORÁRIO$ENTRADA$TERMINAL*/ ;
 REPEAT
 BEGIN
  /* LE$HORA(HORA$ATUAL) */
 END
 UNTIL /* HORA$ATUAL >= HORÁRIO$ENTRADA$TERMINAL */ ;
 REPEAT
 BEGIN
  /* LE(ar3,NÃO-CONSUMPTIVO,OCUPAÇÃO$TERMINAL) */
 END
 UNTIL /* o terminal estar desocupado */ ;
 /* comande velocidade 1 para este e para o próximo trecho */ ;
 /* LE(ar53,CONSUMPTIVO,IDTREM) */ ;
 /* ESCREVE(ar2,DESTRUTIVO,IDTREM */ ;
  /* MARCA(a1,1) */
END.

INTERPRETAÇÃO p(n24) ;
BEGIN
 /*  calcule o tempo de embarque/desembarque consoante o  fluxo   de
 passageiros previsto para esta estação */ ;
 /* LE$HORA(HORA$ATUAL) */ ;
 /*  some  o tempo de embarque/desembarque à HORA$ATUAL.  Coloque  o
 resultado desta adição na variável HORÁRIO$PREVISTO$PARTIDA */ ;
 /* ESCREVE(ar56,DESTRUTIVO,HORÁRIO$PREVISTO$PARTIDA) */ ;
 /* marca(a6,1) */
END.

INTERPRETAÇÃO p(n25) ;
BEGIN
 /* LE(ar56,CONSUMPTIVO,HORA$1) */ ;
 /* LE(ar56,COMSUMPTIVO,HORA$2) */ ;
 /* LE$HORA(HORA$ATUAL) */ ;
 IF /* HORA$1 >= HORA$2 */ THEN
 BEGIN
  IF /* HORA$ATUAL >= HORA$1 */ THEN
  BEGIN
   /* HORÁRIO$PARTIDA:= 0 */
  END
```

```
  ELSE
  BEGIN
   IF /* HORA$ATUAL < HORA$1 */ THEN
   BEGIN
    /* HORÁRIO$PARTIDA:= HORA$1 - HORA$ATUAL */ ;
   END
  END
 END
 ELSE
 BEGIN
  IF /* HORA$1 < HORA$2 */ THEN
  BEGIN
   IF /* HORA$ATUAL >= HORA$2 */ THEN
   BEGIN
    /* HORÁRIO$PARTIDA:= 0  */
   END
   ELSE
   BEGIN
    IF /* HORA$ATUAL < HORA$2 */ THEN
    BEGIN
     /* HORÁRIO$PARTIDA:= HORA$2 - HORA$ATUAL */ ;
    END
   END
  END
 END ;
 /* ATIVA$ESPERA(a8,1,HORÁRIO$PARTIDA) */
END.

INTERPRETAÇÃO p(n26) :
BEGIN
 /* LE$HORA(HORA$ATUAL) */ ;
 /*  some  X2  unidades de tempo a HORA$ATUAL (o valor X2, a ser
 estabelecido durante a implementação, define o tempo de espera
 máximo para a chegada da resposta da camada de regulação). Coloque
 o resultado desta operação na variável HORÁRIO$REAL$PARTIDA */  ;
 /* ESCREVE(ar66,DESTRUTIVO,HORÁRIO$REAL$PARTIDA) */ :
 /* MARCA(a9,1) */ ;
 /* ATIVA$ESPERA(a10,1,X2) */
END.
```

4.3.2.1. ESTADO INICIAL DO MODELO GMB* DO PROCESSO METROVIÁRIO EM ESCALA REDUZIDA

O modelo GMB* discutido na seção anterior descreve os aspectos estáticos do processo metroviário em escala reduzida. Para a descrição dos aspectos dinâmicos faz-se necessária a execução do modelo a partir de um estado inicial válido.

Adotou-se como estado inicial válido a situação de linha apresentada na figura 4.3.7. Esta situação supõe a existência de seis trens, numerados de 1 a 6, em operação na linha. Estes trens são despachados do terminal em ordem crescente de identificação e, após executarem um volta completa no percurso (saída do terminal - chegada no terminal), são reinjetados na linha com a mesma identificação. No instante inicial de operação todos os trens encontram-se a caminho de uma estação (saindo de trecho anterior a

estação e entrando em trecho estação), sendo que: o trem 1 está chegando à estação 6, o trem 2 à estação 5, o trem 3 à estação 4, o trem 4 à estação 3, o trem 5 à estação 2 e o trem 6 à estação 1. Todos os trens estão trafegando com nível de desempenho nominal.

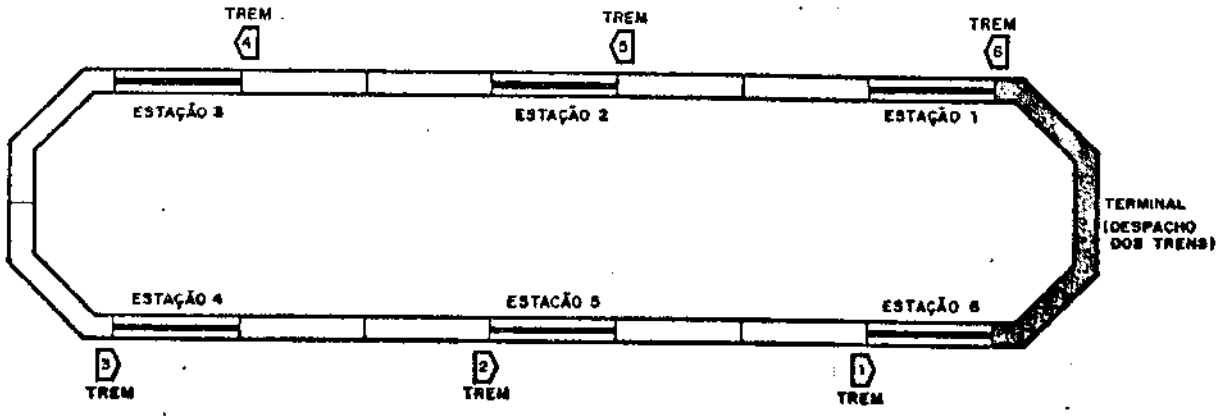


FIG.4.3.7 - DISTRIBUIÇÃO INICIAL DOS TRENS

No que se refere ao despacho dos trens do terminal considera-se, para a situação inicial da figura 4.3.7, que a tabela de despacho dos trens (ar1 do grafo de dados) contém os tempos de despachos de todos os trens para um período de operação. O primeiro elemento da tabela estabelece o horário de despacho do trem 1, o segundo o horário de despacho do trem 2, o terceiro do trem 3 e assim sucessivamente, para todos os trens, tantas vezes quanto especificado pela camada de otimização.

Finalmente, vale frisar que o modelo GMB* apresentado parte do pressuposto que será executado por uma máquina GMB* com capacidade para fazê-lo em paralelismo total (número de processadores físicos da máquina igual ao número máximo de processadores do modelo que podem estar ativo em um determinado instante de tempo).

Segue abaixo o estado inicial que reflete a situação e considerações discutidas acima.

ESTADO DO GRAFO DO CONTROLE

1. Conjunto de nós ativos: {Ø}
2. Distribuição das marcas nos arcos de controle:
   Conjunto dos arcos marcados com prioridade 1: {a3, a13, a23, a33, a43, a53}
   Os demais arcos não estão marcados.

ESTADO DO GRAFO DOS DADOS

1. Conjunto do processadores ativos: {0}
2. Conteúdo dos armazenadores:

ar1 - contém os horários de despacho dos trens para o período em análise:
HORÁRIO$DESPACHO$TREM1

HORÁRIO$DESPACHO$TREM2
HORÁRIO$DESPACHO$TREM3
HORÁRIO$DESPACHO$TREM4
HORÁRIO$DESPACHO$TREM5
HORÁRIO$DESPACHO$TREM6
HORÁRIO$DESPACHO$TREM1
HORÁRIO$DESPACHO$TREM2
.
.
.

ar3 - contém o estado de ocupação do trecho 16:
OCUPADO

ar5 - contém o estado de ocupação do trecho 0 :
DESOCUPADO

ar8 - contém o estado de ocupação do trecho 1 :
DESOCUPADO

ar12 - contém o estado de ocupação do trecho 2 :
OCUPADO

ar15 - contém o estado de ocupação do trecho 3 :
DESOCUPADO

ar19 - contém o estado de ocupação do trecho 4 :
DESOCUPADO

ar22 - contém o estado de ocupação do trecho 5 :
OCUPADO

ar25 - contém o estado de ocupação do trecho 6 :
DESOCUPADO

ar28 - contém o estado de ocupação do trecho 7 :
DESOCUPADO

ar32 - contém o estado de ocupação do trecho 8 :
OCUPADO

ar35 - contém o estado de ocupação do trecho 9 :
DESOCUPADO

ar39 - contém o estado de ocupação do trecho 10:
DESOCUPADO

ar42 - contém o estado de ocupação do trecho 11:
OCUPADO

ar45 - contém o estado de ocupação do trecho 12:
DESOCUPADO

ar49 - contém o estado de ocupação do trecho 13:
DESOCUPADO

ar52 - contém o estado de ocupação do trecho 14:
OCUPADO

ar55 - contém o estado de ocupação do trecho 15:
DESOCUPADO

ar4 - contém o identificador do trem em trânsito entre os trechos 16 e 0 :
TREM6

ar13 - contém o identificador do trem em trânsito entre os trechos 2 e 3 :
TREM5

ar23 - contém o identificador do trem em trânsito entre os trechos 5 e 6 :
TREM4

ar33 - contém o identificador do trem em trânsito entre os trechos 8 e 9 :
TREM3

ar43 - contém o identificador do trem em trânsito entre os trechos 11 e 12:
TREM2

ar53 - contém o identificador do trem em trânsito entre os trechos 14 e 15:
TREM1

ar14 - contém o nível de desempenho do trem em trânsito entre os trechos 16 e 0 :
NÍVEL$DESEMPENHO$NOMINAL

ar24 - contém o nível de desempenho do trem em trânsito entre os trechos 2 e 3 :
NÍVEL$DESEMPENHO$NOMINAL

ar34 - contém o nível de desempenho do trem em trânsito entre os trechos 5 e 6 :
NÍVEL$DESEMPENHO$NOMINAL

ar44 - contém o nível de desempenho do trem em trânsito entre os trechos 8 e 9 :
NÍVEL$DESEMPENHO$NOMINAL

ar54 - contém o nível de desempenho do trem em trânsito entre os trechos 11 e 12:
NÍVEL$DESEMPENHO$NOMINAL

ar14 - contém o nível de desempenho do trem em trânsito entre os trechos 14 e 15:
NÍVEL$DESEMPENHO$NOMINAL

Os demais armazenadores estão vazios.

4.3.3. O CONTROLE DO PROCESSO METROVIÁRIO EM ESCALA REDUZIDA

Aos moldes do Metropolitano de São Paulo, o controle do

processo metroviário em escala reduzida será efetuado por três camadas distintas: IMPLEMENTAÇÃO, REGULAÇÃO E ADAPTAÇÃO.

A camada de IMPLEMENTAÇÃO, localizada no nível do processo, garante a segurança da via, evitando que um trem avance sobre um trecho já ocupado (conceito sombra). No Metrô de São Paulo este controle é efetuado pelos circuitos de via. No processo metroviário em escala reduzida é efetuado pelo micro-computador de via.

Uma vez definida a trajetória de referência a ser seguida pelo sistema, é tarefa da camada de REGULAÇÃO manter o sistema operando em torno desta solução nominal. De caráter local, esta camada atua apenas nas estações, quando da chegada de trens, definindo o tempo de parada e o nível de desempenho até a próxima estação, de forma a manter o trem dentro do horário pré-estabelecido.

O tempo de parada definido pela regulação, entretanto, pode sofrer perturbações devido a um aumento inesperado do fluxo de passageiros. Resguardados casos mais drásticos, a variação do fluxo de passageiros é o principal fator de perturbação do sistema. Uma vez afastado de sua trajetória de referência, o sistema fica sujeito a ter esta situação agravada devido a ocorrência de interferência entre trens. Tal interferência tem lugar quando um trem retardatário começa a bloquear o avanço das composições que o sucedem. Nesta situação, a camada de regulação tem a sua capacidade de ação deteriorada, tendendo à saturação (tempo de parada mínimo e nível de desempenho máximo). Assim, em situações fortemente perturbadas, faz-se necessária a aplicação de estratégias de controle de alto nível que avaliem a situação global da linha e atuem no sentido de permitir a volta à nomalidade. Tais estratégias de alto nível constituem a chamada camada de ADAPTAÇÃO. Para o processo metroviário em escala reduzida a camada de adaptação será constituída pelas estratégias DEFASAGEM NO HORÁRIO DO SISTEMA e DISTRIBUIÇÃO DE INTERVALOS.

4.3.3.1. **CAMADA DE REGULAÇÃO**

A camada de regulação tem por objetivo manter o sistema operando em torno da trajetória de referência definida pela camada de otimização e modificada pela camada de adaptação. De caráter marcadamente local, esta camada atua apenas nas estações, quando da chegada/partida de trem, definindo o tempo de parada e o nível de desempenho a ser utilizado no trajeto até a próxima estação. A atuação da regulação está condicionada à ocorrência de dois eventos bem definidos: evento portas abertas (chegada de trem em estação) e eventos portas fechadas (partida de trem de estação).

Quando do evento portas abertas (chegada de trem), a camada de regulação (fase 1) verifica se o trem chegou à estação no horário previsto. Caso o trem esteja no horário, a regulação define o tempo de parada e o nível de desempenho como sendo os nominais. Caso contrário, assumindo valor nominal para o tempo de parada, calcula o nível de desempenho que permita ao trem, tanto quanto possível, chegar no horário à próxima estação e, em seguida, ajusta o tempo de parada de modo a minimizar eventuais discrepâncias advindas do cálculo anterior. Tais discrepâncias são devidas à reduzida gama de

valores (seis no total) que a variável nível de desempenho pode assumir. Nesta discussão, os diferentes níveis de desempenho são numerados de 1 a 6, sendo que o nível de desempenho 1 corresponde à maior média horária de deslocamento entre estações, decrescendo até o nível 6, que representa a menor média horária.

Quando do evento portas fechadas, a camada de regulação (fase 2) verifica se o tempo real de parada correspondeu ao tempo de parada definido quando da chegada do trem à estação (fase 1 da regulação). Em caso afirmativo, nada faz. Caso contrário, recalcula o nível de desempenho na tentativa de manter o trem dentro do horário previsto. Esta redefinição do nível de desempenho tenta corrigir a perturbação ocorrida na estação devido ao fluxo de passageiros.

A camada de regulação considera que o horário previsto de chegada de um trem a uma estação é dado por:

HTC(EST,TR) + DEFTR(EST,TR) + DEFSIS

onde:

HTC(EST,TR) - é o HORÁRIO TEÓRICO DE CHEGADA do trem TR à estação EST. Tal valor é definido pela camada de otimização.
DEFTR(EST,TR) - é a defasagem a ser aplicada ao trem TR na estação EST, consoante cálculo da camada de adaptação - estratégia de distribuição de intervalos.
DEFSIS - é a defasagem a ser aplicada ao sistema, segundo cálculo da camada de adaptação - estratégia de defasagem no horário do sistema.

Para a configuração do processo metroviário em escala reduzida considerado na seção 4.3.2., a atuação da camada de regulação tem lugar apenas nas estações 1, 2, 3, 4 e 5. Na estação 6 não se faz necessária a aplicação da estratégia de regulação pois, um trem, ao deixar esta estação é reconhido ao terminal sem horário previsto de chegada.

Uma vez que a camada de regulação tem atuação idêntica em cada estação da linha, apresentar-se-á aqui apenas o modelo GMB* associado a uma estação (estação 1). O modelo para as demais estações pode ser estabelecido reproduzindo-se a estrutura do modelo apresentado. Como guia para a dedução do modelo completo da camada de regulação, apresenta-se a seguir uma tabela de equivalência funcional entre os elementos deste modelo.

TABELA DE EQUIVALÊNCIA FUNCIONAL

ARMAZENADORES

NÍVEL DE DESEMPENHO - CAMADA DE REGULAÇÃO: ar6, ar16, ar26, ar36 e ar46.
ACUMULADOR AUXILIAR DA GERAÇÃO DO TEMPO DE PARADA NA ESTAÇÃO: ar56, ar57, ar58, ar59 e ar60.
CHEGADA DE TREM NA ESTAÇÃO: ar61, ar62, ar63, ar64 e ar65.
HORÁRIO DE PARTIDA DO TREM: ar66, ar67, ar68, ar69 e ar70.
ESTADO DO TREM 1: ar71
ESTADO DO TREM 2: ar72

ESTADO DO TREM 3: ar73
ESTADO DO TREM 4: ar74
ESTADO DO TREM 5: ar75
ESTADO DO TREM 6: ar76
DEFASAGEM SISTEMA: ar77, ar91, ar105, ar119 e ar133.
RESULTADO PRIMEIRA FASE REGULAÇÃO: ar78, ar92, ar106, ar120 e ar134.
HORÁRIO TEÓRICO CHEGADA TREM 1: ar79, ar93, ar107, ar121 e ar135.
HORÁRIO TEÓRICO CHEGADA TREM 2: ar80, ar94, ar108, ar122 e ar136.
HORÁRIO TEÓRICO CHEGADA TREM 3: ar81, ar95, ar109, ar123 e ar137.
HORÁRIO TEÓRICO CHEGADA TREM 4: ar82, ar96, ar110, ar124 e ar138.
HORÁRIO TEÓRICO CHEGADA TREM 5: ar83, ar97, ar111, ar125 e ar139.
HORÁRIO TEÓRICO CHEGADA TREM 6: ar84, ar98, ar112, ar126 e ar140.
DEFASAGEM TREM 1: ar85, ar99, ar113, ar127 e ar141.
DEFASAGEM TREM 2: ar86, ar100, ar114, ar128 e ar142.
DEFASAGEM TREM 3: ar87, ar101, ar115, ar129 e ar143.
DEFASAGEM TREM 4: ar88, ar102, ar116, ar130 e ar144.
DEFASAGEM TREM 5: ar89, ar103, ar117, ar131 e ar145.
DEFASAGEM TREM 6: ar90, ar104, ar118, ar132 e ar146.
NÍVEL DE DESEMPENHO NOMINAL: ar155, ar157, ar159, ar161 e ar163.
TEMPO PARADA NOMINAL: ar156, ar158, ar160, ar162 e ar164.

PROCESSADORES

PRIMEIRA FASE REGULAÇÃO: p(n39), p(n41), p(n43), p(n45) e p(n47).
SEGUNDA FASE REGULAÇÃO: p(n40), p(n42), p(n44), p(n46) e p(n48).

As figuras 4.3.8 e 4.3.9 apresentam, respectivamente, os domínios do controle e dos dados do modelo GMB* da camada de regulação - estação 1. Segue abaixo o domínio da interpretação.

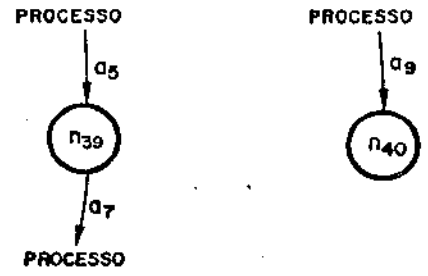


FIG.4.3.8 - MODELO GMB* DA CAMADA DE REGULAÇÃO - ESTAÇÃO 1: DOMÍNIO DO CONTROLE

DOMÍNIO DA INTERPRETAÇÃO

INTERPRETAÇÃO DOS ARMAZENADORES

ar6 - Nível de desempenho - camada de regulação
ar6::=<NÍVEL DE DESEMPENHO>
ar56 - acumulador auxiliar da geração do tempo de parada na estação
ar56::=<HORA><HORA>

ar61 - Chegada de trem na estação
ar61::=<<IDENTIFICADOR TREM><HORA>>

ar66 - Horário partida trem
ar66::=<HORA>

ar71 - Estado do trem 1
ar71::=<<atraso teórico><atraso operacional>
<IDENTIFICADOR ESTAÇÃO>>
<atraso teórico>::= <HORA>
<atraso operacional>::= <HORA>

ar72 - Estado do trem 2
ar72::=<<atraso teórico><atraso operacional>
<IDENTIFICADOR ESTAÇÃO>>
<atraso teórico>::= <HORA>
<atraso operacional>::= <HORA>

ar73 - Estado do trem 3
ar73::=<<atraso teórico><atraso operacional>
<IDENTIFICADOR ESTAÇÃO>>
<atraso teórico>::= <HORA>
<atraso operacional>::= <HORA>

ar74 - Estado do trem 4
ar74::=<<atraso teórico><atraso operacional>
<IDENTIFICADOR ESTAÇÃO>>
<atraso teórico>::= <HORA>
<atraso operacional>::= <HORA>

ar75 - Estado do trem 5
ar75::=<<atraso teórico><atraso operacional>
<IDENTIFICADOR ESTAÇÃO>>
<atraso teórico>::= <HORA>
<atraso operacional>::= <HORA>

ar76 - Estado do trem 6
ar76::=<<atraso teórico><atraso operacional>
<IDENTIFICADOR ESTAÇÃO>>
<atraso teórico>::= <HORA>
<atraso operacional>::= <HORA>

ar77 - Defasagem do sistema
ar77::=<HORA>

ar78 - Resultado primeira fase regulação
ar78::=<<horário partida><NÍVEL DE DESEMPENHO>>
<horário partida>::=<HORA>

ar79 - Horário teórico chegada trem 1
ar79::=(<horário teórico chegada>)
<horário teórico chegada>::=<HORA>

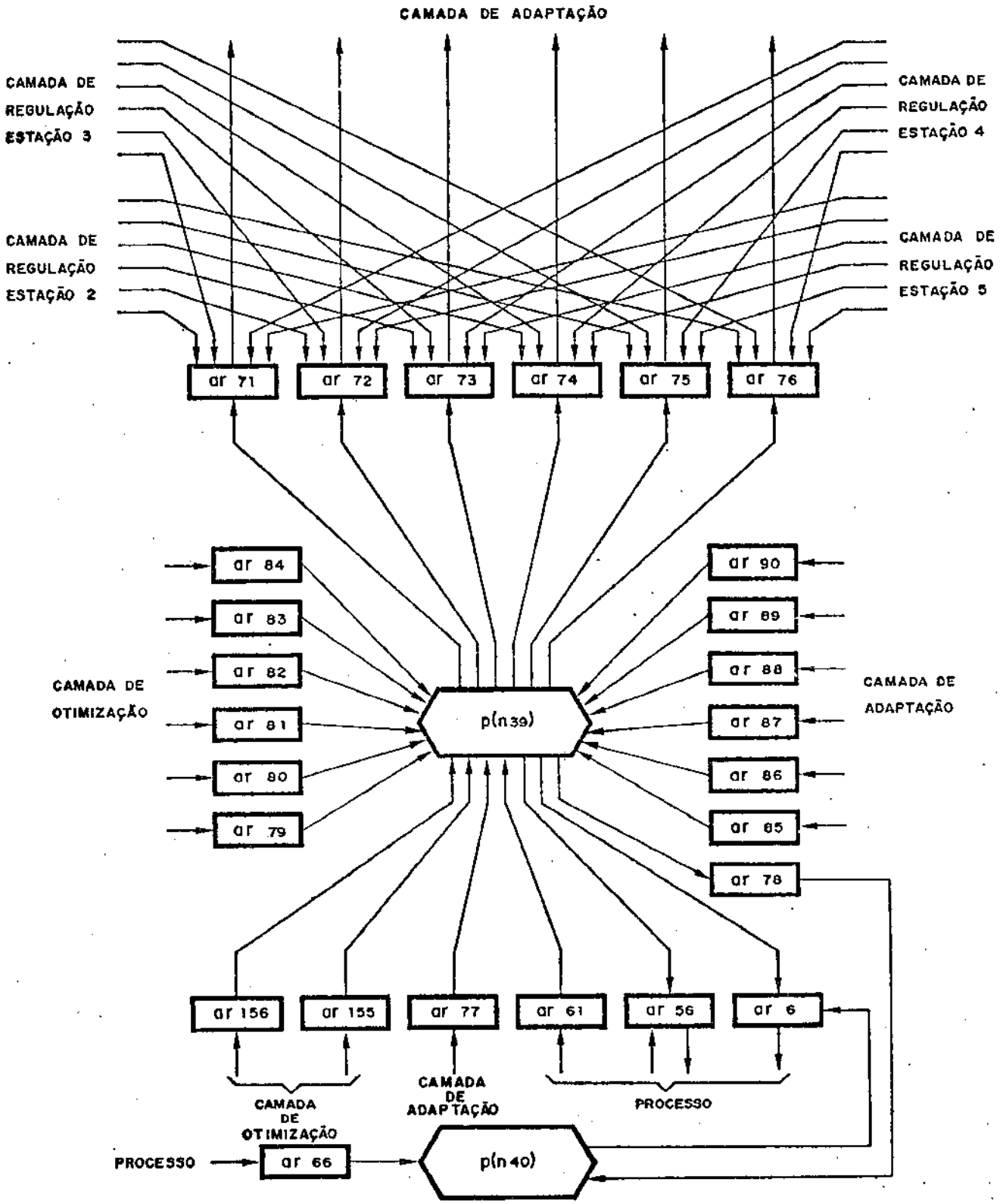


FIG.4.3.9 - MODELO GMB* DA CAMADA DE REGULAÇÃO - ESTAÇÃO 1: DOMÍNIO DOS DADOS

ar80 - Horário teórico chegada trem 2
ar80::=(<horário teórico chegada>)
<horário teórico chegada>::=<HORA>

ar81 - Horário teórico chegada trem 3
ar81::=(<horário teórico chegada>)
<horário teórico chegada>::=<HORA>

ar82 - Horário teórico chegada trem 4
ar82::=(<horário teórico chegada>)
<horário teórico chegada>::=<HORA>

ar83 - Horário teórico chegada trem 5
ar83::=(<horário teórico chegada>)
<horário teórico chegada>::=<HORA>

ar84 - Horário teórico chegada trem 6
ar84::=(<horário teórico chegada>)
<horário teórico chegada>::=<HORA>

ar85 - Defasagem trem 1 (trajetória operacional)
ar85::=<HORA>

ar86 - Defasagem trem 2 (trajetória operacional)
ar86::=<HORA>

ar87 - Defasagem trem 3 (trajetória operacional)
ar87::=<HORA>

ar88 - Defasagem trem 4 (trajetória operacional)
ar88::=<HORA>

ar89 - Defasagem trem 5 (trajetória operacional)
ar89::=<HORA>

ar90 - Defasagem trem 6 (trajetória operacional)
ar90::=<HORA>

ar155 - Nível de desempenho nominal
ar155::=<NÍVEL DE DESEMPENHO>

ar156 - Tempo parada nominal
ar156::=<HORA>

INTERPRETAÇÃO DOS PROCESSADORES

```
INTERPRETAÇÃO p(n39) ;
BEGIN
 /* LE(ar61,CONSUMPTIVO,VARIÁVEL$AUXILIAR) */ ;
 /* desmembre a variável VARIÁVEL$AUXILIAR em:  IDTREM:=<IDENTIFICA-
 DOR TREM>, HORA$CHEGADA:=<HORA> */ ;
 /* LE(ar[78+IDTREM],CONSUMPTIVO,HORÁRIO$TEÓRICO$CHEGADA) */ ;
 /* LE(ar155,NÃO-CONSUMPTIVO,NÍVEL$DESEMPENHO$NOMINAL) */ ;
 /* LE(ar156,NÃO-CONSUMPTIVO,TEMPO$PARADA$NOMINAL) */ ;
 /* LE(ar77,NÃO-CONSUMPTIVO,DEFASAGEM$SISTEMA) */ ;
 /* ATRASO$TEÓRICO:= HORA$CHEGADA - (HORÁRIO$TEÓRICO$CHEGADA + DEFA-
 SAGEM$SISTEMA) */ ;
 /* LE(ar[84+IDTREM],CONSUMPTIVO,DEFASAGEM$TREM) */ ;
```

```
IF /* ar[84+IDTREM] estava vazio */ THEN
BEGIN
  /* DEFASAGEM$TREM:= 0 */
END ;
/*  ATRASO$OPERACIONAL:= HORA$CHEGADA - (HORÁRIO$TEÓRICO$CHEGADA +
DEFASAGEM$TREM + DEFASAGEM$SISTEMA) */ ;
/* ATRASO$OPERACIONAL$ESTIMADO:= ATRASO$OPERACIONAL */ ;
/* NÍVEL$DESEMPENHO:= NÍVEL$DESEMPENHO$NOMINAL */ ;
IF /* ATRASO$OPERACIONAL$ESTIMADO > TOLERÂNCIA */ THEN
BEGIN
 REPEAT
 BEGIN
  /* NÍVEL$DESEMPENHO:= NÍVEL$DESEMPENHO - 1 */ ;
  /*  ATRASO$OPERACIONAL$ESTIMADO:=  HORA$CHEGADA  +  TEMPO$PERCUR-
  SO(ESTAÇÃO1,NÍVEL$DESEMPENHO)  - (HORÁRIO$TEÓRICO$CHEGADA + DEFA-
  SAGEM$TREM  +  DEFASAGEM$SISTEMA  +   TEMPO$PERCURSO(ESTAÇÃO1,NÍ-
  VEL$DESEMPENHO$NOMINAL)) */
 END
 UNTIL /* (ATRASO$OPERACIONAL$ESTIMADO < TOLERÂNCIA) OR  (NÍVEL$DE-
 SEMPENHO = 1) */
 END
 ELSE
BEGIN
 IF /* ATRASO$OPERACIONAL$ESTIMADO < - TOLERÂNCIA */ THEN
 BEGIN
  REPEAT
  BEGIN
   /* NÍVEL$DESEMPENHO:= NÍVEL$DESEMPENHO + 1 */ ;
   /*  ATRASO$OPERACINAL$ESTIMADO:=  HORA$CHEGADA  +  TEMPO$PERCUR-
   SO(ESTAÇÃO1,NÍVEL$DESEMPENHO) - (HORÁRIO$TEÓRICO$CHEGADA + DEFA-
   SAGEM$TREM  +  DEFASAGEM$SISTEMA  +  TEMPO$PERCURSO(ESTAÇÃO1,NÍ-
   VEL$DESEMPENHO$NOMINAL)) */
  END
  UNTIL  /*  (ATRASO$OPERACIONAL$ESTIMADO >= - TOLERÂNCIA) OR  (NÍ-
  VEL$DESEMPENHO = 8) */
 END
END ;
/*  TEMPO$PARADA:=  TEMPO$PARADA$NOMINAL   - ATRASO$OPERACIONAL$ES-
TIMADO */ ;
IF /* TEMPO$PARADA < TEMPO$PARADA$MÍNIMO */ THEN
BEGIN
 /* TEMPO$PARADA:= TEMPO$PARADA$MÍNIMO */
END
ELSE
BEGIN
 IF /* TEMPO$PARADA > TEMPO$PARADA$MÁXIMO */ THEN
 BEGIN
  /* TEMPO$PARADA:= TEMPO$PARADA$MÁXIMO */
 END
END ;
/* ESCREVE(ar8,DESTRUTIVO,NÍVEL$DESEMPENHO) */ ;
/* HORÁRIO$PARTIDA:= HORA$CHEGADA + TEMPO$PARADA */ ;
/* ESCREVE(ar58,DESTRUTIVO,HORÁRIO$PARTIDA) */ ;
/* concatene as variáveis ATRASO$TEÓRICO,  ATRASO$OPERACIONAL,  e o
identificador  da  ESTAÇÃO1,  formando a variável  INFORMAÇÃO$ADAP-
TAÇÃO */ ;
/* ESCREVE(ar[70+IDTREM],DESTRUTIVO,INFORMAÇÃO$ADAPTAÇÃO) */ ;
/*  concatene as variáveis HORÁRIO$PARTIDA e NÍVEL$DESEMPENHO  for-
```

```
 /*  concatene as variáveis HORÁRIO$PARTIDA e NÍVEL$DESEMPENHO  for-
 mando a variável INFORMAÇÃO$FASE2$REGULAÇÃO) */ ;
 /* ESCREVE(ar78,DESTRUTIVO,INFORMAÇÃO$FASE2$REGULAÇÃO) */ ;
 /* MARCA(a7,1) */
END.

INTERPRETACAO p(n40) ;
BEGIN
 /* LE(ar66,CONSUMPTIVO,HORÁRIO$REAL$PARTIDA) */ ;
 /* LE(ar78,CONSUMPTIVO,INFORMAÇÃO$FASE1$REGULAÇÃO) */ ;
 /* desmembre a variável INFORMAÇÃO$FASE1$REGULAÇÃO em: HORÁRIO$PAR-
 TIDA$CALCULADO:=<HORA>,    NÍVEL$DESEMPENHO:=<NÍVEL   DE   DESEMPE-
 NHO> */ ;
 /*   ATRASO$PARTIDA:=  HORÁRIO$REAL$PARTIDA  - HORÁRIO$PARTIDA$CAL-
 CULADO */ ;
 WHILE  /* (ATRASO$PARTIDA > TOLERÂNCIA) AND (NÍVEL$DESEMPENHO    <>
 1)  */ DO
 BEGIN
  /* NÍVEL$DESEMPENHO:= NÍVEL$DESEMPENHO - 1 */ ;
  /*  ATRASO$PARTIDA:=  HORÁRIO$REAL$PARTIDA  +   TEMPO$PERCURSO(ES-
  TAÇÃO1,NÍVEL$DESEMPENHO) - (HORÁRIO$PARTIDA$CALCULADO + TEM PO$PER-
  CURSO(ESTAÇÃO1,NÍVEL$DESEMPENHO$CALCULADO)) */
 END ;
 /* ESCREVE(ar6,DESTRUTIVO,NÍVEL$DESEMPENHO) */
END.
```

O estado inicial válido para o modelo GMB* da camada de regulação, consistente com o discutido na seção 4.3.2. é:

ESTADO DO GRAFO DO CONTROLE

1. Conjunto de nós ativos: {Ø}
2. Distribuição de marcas nos arcos de controle: Nenhum arco está marcado.

ESTADO DO GRAFO DOS DADOS

1. Conjunto dos processadores ativos: {Ø}
2. Conteúdo dos armazenadores:

ar71 - contém o estado do trem 1:
ATRASO$TEÓRICO
ATRASO$OPERACIONAL
IDENTIFICADOR$ESTAÇÃO = 5

ar72 - contém o estado do trem 2:
ATRASO$TEÓRICO
ATRASO$OPERACIONAL
IDENTIFICADOR$ESTAÇÃO = 4

ar73 - contém o estado do trem 3:
ATRASO$TEÓRICO
ATRASO$OPERACIONAL
IDENTIFICADOR$ESTAÇÃO = 3

ar74 - contém o estado do trem 4:
ATRASO$TEÓRICO

ATRASO$OPERACIONAL
IDENTIFICADOR$ESTAÇÃO = 2

ar75 - contém o estado do trem 5:
ATRASO$TEÓRICO
ATRASO$OPERACIONAL
IDENTIFICADOR$ESTAÇÃO = 1

ar76 - contém o estado do trem 6:
ATRASO$TEÓRICO
ATRASO$OPERACIONAL
IDENTIFICADOR$ESTAÇÃO = 5

ar77 - contém a defasagem atual do sistema (estação 1):
DEFASAGEM$SISTEMA

ar91 - contém a defasagem atual do sistema (estação 2):
DEFASAGEM$SISTEMA

ar105 - contém a defasagem atual do sistema (estação 3):
DEFASAGEM$SISTEMA

ar119 - contém a defasagem atual do sistema (estação 4):
DEFASAGEM$SISTEMA

ar133 - contém a defasagem atual do sistema (estação 5):
DEFASAGEM$SISTEMA

ar79 - contém os horários teóricos de chegada do TREM 1 (estação 1):
HORÁRIO$TEÓRICO$CHEGADA
HORÁRIO$TEÓRICO$CHEGADA
.
.
.

ar80 - contém os horários teóricos de chegada do TREM 2 (estação 1):
HORÁRIO$TEÓRICO$CHEGADA
HORÁRIO$TEÓRICO$CHEGADA
.
.
.

ar81 - contém os horários teóricos de chegada do TREM 3 (estação 1):
HORÁRIO$TEÓRICO$CHEGADA
HORÁRIO$TEÓRICO$CHEGADA
.
.
.

ar82 - contém os horários teóricos de chegada do TREM 4 (estação 1):
HORÁRIO$TEÓRICO$CHEGADA
HORÁRIO$TEÓRICO$CHEGADA
.
.

ar83 - contém os horários teóricos de chegada do TREM 5 (estação 1):
HORÁRIO$TEÓRICO$CHEGADA
HORÁRIO$TEÓRICO$CHEGADA
.
.
.

ar84 - contém os horários teóricos de chegada do TREM 6 (estação 1):
HORÁRIO$TEÓRICO$CHEGADA
HORÁRIO$TEÓRICO$CHEGADA
.
.
.

ar93 - contém os horários teóricos de chegada do TREM 1 (estação 2):
HORÁRIO$TEÓRICO$CHEGADA
HORÁRIO$TEÓRICO$CHEGADA
.
.
.

ar94 - contém os horários teóricos de chegada do TREM 2 (estação 2):
HORÁRIO$TEÓRICO$CHEGADA
HORÁRIO$TEÓRICO$CHEGADA
.
.
.

ar95 - contém os horários teóricos de chegada do TREM 3 (estação 2):
HORÁRIO$TEÓRICO$CHEGADA
HORÁRIO$TEÓRICO$CHEGADA
.
.
.

ar96 - contém os horários teóricos de chegada do TREM 4 (estação 2):
HORÁRIO$TEÓRICO$CHEGADA
HORÁRIO$TEÓRICO$CHEGADA
.
.
.

ar97 - contém os horários teóricos de chegada do TREM 5 (estação 2):
HORÁRIO$TEÓRICO$CHEGADA
HORÁRIO$TEÓRICO$CHEGADA
.
.
.

ar98 - contém os horários teóricos de chegada do TREM 6 (estação 2):

HORÁRIO$TEÓRICO$CHEGADA
HORÁRIO$TEÓRICO$CHEGADA
.
.
.

ar107 - contém os horários teóricos de chegada do TREM 1 (estação 3):
HORÁRIO$TEÓRICO$CHEGADA
HORÁRIO$TEÓRICO$CHEGADA
.
.
.

ar108 - contém os horários teóricos de chegada do TREM 2 (estação 3):
HORÁRIO$TEÓRICO$CHEGADA
HORÁRIO$TEÓRICO$CHEGADA
.
.
.

ar109 - contém os horários teóricos de chegada do TREM 3 (estação 3):
HORÁRIO$TEÓRICO$CHEGADA
HORÁRIO$TEÓRICO$CHEGADA
.
.
.

ar110 - contém os horários teóricos de chegada do TREM 4 (estação 3):
HORÁRIO$TEÓRICO$CHEGADA
HORÁRIO$TEÓRICO$CHEGADA
.
.
.

ar111 - contém os horários teóricos de chegada do TREM 5 (estação 3):
HORÁRIO$TEÓRICO$CHEGADA
HORÁRIO$TEÓRICO$CHEGADA
.
.
.

ar112 - contém os horários teóricos de chegada do TREM 6 (estação 3):
HORÁRIO$TEÓRICO$CHEGADA
HORÁRIO$TEÓRICO$CHEGADA
.
.
.

ar121 - contém os horários teóricos de chegada do TREM 1 (estação 4):
HORÁRIO$TEÓRICO$CHEGADA
HORÁRIO$TEÓRICO$CHEGADA

.
.
.

ar122 - contém os horários teóricos de chegada do TREM 2 (estação 4):
HORÁRIO$TEÓRICO$CHEGADA
HORÁRIO$TEÓRICO$CHEGADA
.
.
.

ar123 - contém os horários teóricos de chegada do TREM 3 (estação 4):
HORÁRIO$TEÓRICO$CHEGADA
HORÁRIO$TEÓRICO$CHEGADA
.
.
.

ar124 - contém os horários teóricos de chegada do TREM 4 (estação 4):
HORÁRIO$TEÓRICO$CHEGADA
HORÁRIO$TEÓRICO$CHEGADA
.
.
.

ar125 - contém os horários teóricos de chegada do TREM 5 (estação 4):
HORÁRIO$TEÓRICO$CHEGADA
HORÁRIO$TEÓRICO$CHEGADA
.
.
.

ar126 - contém os horários teóricos de chegada do TREM 6 (estação 4):
HORÁRIO$TEÓRICO$CHEGADA
HORÁRIO$TEÓRICO$CHEGADA
.
.
.

ar135 - contém os horários teóricos de chegada do TREM 1 (estação 5):
HORÁRIO$TEÓRICO$CHEGADA
HORÁRIO$TEÓRICO$CHEGADA
.
.
.

ar136 - contém os horários teóricos de chegada do TREM 2 (estação 5):
HORÁRIO$TEÓRICO$CHEGADA
HORÁRIO$TEÓRICO$CHEGADA
.
.

ar137 - contém os horários teóricos de chegada do TREM 3 (estação 5):
HORÁRIO$TEÓRICO$CHEGADA
HORÁRIO$TEÓRICO$CHEGADA
.
.
.

ar138 - contém os horários teóricos de chegada do TREM 4 (estação 5):
HORÁRIO$TEÓRICO$CHEGADA
HORÁRIO$TEÓRICO$CHEGADA
.
.
.

ar139 - contém os horários teóricos de chegada do TREM 5 (estação 5):
HORÁRIO$TEÓRICO$CHEGADA
HORÁRIO$TEÓRICO$CHEGADA
.
.
.

ar140 - contém os horários teóricos de chegada do TREM 6 (estação 5):
HORÁRIO$TEÓRICO$CHEGADA
HORÁRIO$TEÓRICO$CHEGADA
.
.
.

ar155 - contém o nível de desempenho nominal (estação 1):
NÍVEL$DESEMPENHO$NOMINAL

ar157 - contém o nível de desempenho nominal (estação 2):
NÍVEL$DESEMPENHO$NOMINAL

ar159 - contém o nível de desempenho nominal (estação 3):
NÍVEL$DESEMPENHO$NOMINAL

ar161 - contém o nível de desempenho nominal (estação 4):
NÍVEL$DESEMPENHO$NOMINAL

ar163 - contém o nível de desempenho nominal (estação 5):
NÍVEL$DESEMPENHO$NOMINAL

ar156 - contém o tempo de parada nominal (estação 1):
TEMPO$PARADA$NOMINAL

ar158 - contém o tempo de parada nominal (estação 2):
TEMPO$PARADA$NOMINAL

ar160 - contém o tempo de parada nominal (estação 3):
TEMPO$PARADA$NOMINAL

ar162 - contém o tempo de parada nominal (estação 4):
TEMPO$PARADA$NOMINAL

ar164 - contém o tempo de parada nominal (estação 5): TEMPO$PARADA$NOMINAL

Os demais armazenadores estão vazios.

4.3.3.2. **CAMADA DE ADAPTAÇÃO**

A camada de adaptação atua em situações fortemente perturbadas, não recuperáveis pela camada de regulação, modificando temporariamente a trajetória de referência definida pela otimização, no intuito de restabelecer a normalidade à linha. No exemplo em consideração, a camada de adaptação é acionada periodicamente em uma frequência pré-estabelecida, sendo constituída por duas estratégias distintas e independentes: ESTRATÉGIA DE DEFASAGEM NO HORÁRIO DO SISTEMA e ESTRATÉGIA DE DISTRIBUIÇÃO DE INTERVALOS.

A ESTRATÉGIA DE DEFASAGEM NO HORÁRIO DO SISTEMA efetua um constante ajuste da defasagem global a ser aplicada na trajetória de referência na tentativa de regularizar e evitar a saturação da camada de regulação. A determinação do ajuste a ser efetuado baseia-se no parâmetro ATRASO IRRECUPERÁVEL (ATIR):

$$ATIR(TR) = ATE(TR) - ATR(TR)$$

onde:

ATIR(TR) - é o ATRASO IRRECUPERÁVEL do trem TR.

ATE(TR) - é o ATRASO TEÓRICO do trem TR detetado na última estação por onde passou:
ATE(TR) = HRC(ESTU,TR) - (HTC(ESTU,TR) + DEFSIS)
HRC(ESTU,TR) - é o HORÁRIO REAL DE CHEGADA do trem TR à última estação por onde passou - ESTU. Este valor é medido pelo processo quando da chegada de trem à estação.
HTC(ESTU,TR) - é o HORÁRIO TEÓRICO DE CHEGADA do trem TR à última estação por onde passou - ESTU. Este valor é definido pela camada de otimização.
DEFSIS - é o valor atual da DEFASAGEM DO SISTEMA. Este valor é atualizado por esta estratégia.

ATR(TR) - é o ATRASO RECUPERÁVEL do trem TR até o final do percurso a partir da última estação por onde passou:
$ATR(TR) = \sum_{I=ESTU}^{ESTF} (TPE(I,NDN) - TPE(I,ND1) + TPN - TPM)$
ESTU - é o identificador da última estação por onde o trem TR passou.
ESTF - é o identificador da estação anterior à estação de final de percurso (estação 5 na configuração da seção 4.3.2)
TPE(EST,ND) - é o tempo de percursso para um trem ir da estação EST até a estação EST + 1 com nível de desempenho ND.
NDN - é o NÍVEL DE DESEMPENHO NOMINAL definido pela camada de otimização.
ND1 - é o nível de desempenho máximo admitido no sistema - NÍVEL DE DESEMPENHO 1.
TPN - é o TEMPO DE PARADA NOMINAL definido pela camada de otimização.
TPM - é o TEMPO DE PARADA MÍNIMO permitido no sistema (definido a priori).

O parâmetro ATRASO IRRECUPERÁVEL (ATIR), ao assumir valores negativos, é denominado AVANÇO POTENCIAL (AVP).

Quando da aplicação da estratégia de defasagem no horário do sistema, toma-se o ATRASO IRRECUPERÁVEL MÉDIO como fator de acréscimo ao valor vigente da defasagem do sistema. O ATRASO IRRECUPERÁVEL MÉDIO é calculado dividindo-se o somatório de todos os atrasos irrecuperáveis (somente valores positivos) existentes na linha no instante de aplicação da estratégia pelo número total de trens em operação na linha. Cuidado especial é tomado para limitar a atuação desta estratégia pois, valores excessivamente elevados de defasagem representam, ao final de um dia de operação, uma diminuição do número de viagens efetuadas e, portanto, da capacidade de transporte do sistema. A sistemática adotada para a limitação da atuação desta estratégia consiste na definição de um limite máximo para o valor do acréscimo passível de ser efetuado na defasagem a cada aplicação da estratégia. Caso o ATRASO IRRECUPERÁVEL MÉDIO, em um dado instante de aplicação da estratégia, seja zero, toma-se o AVANÇO POTENCIAL MÉDIO como parâmetro para a redução da defasagem do sistema. Ao se reduzir a defasagem do sistema deve ser observada a restrição da "não negatividade" de seu valor, uma vez que, por definição, não é permitido adiantar globalmente o sistema.

Já a ESTRATÉGIA DE DISTRIBUIÇÃO DE INTERVALOS procura, como o próprio nome sugere, manter os trens uniformemente distribuídos ao longo da linha, evitando a formação de vazios ou aglomerações de trens. Esta estratégia, ao detetar um trem suficientemente atrasado, modifica os horários teóricos de chegada de todos os trens à frente deste trem problema, incluindo este, em todas as estações, a partir da última estação por onde o trem problema passou, até o final do percurso. A idéia é manter o trem problema em sua marcha de recuperação e atrasar, ponderadamente, os que o precedem, fazendo-os recuar, cobrindo o vazio que se formou. Para tanto, a partir da primeira estação da linha, esta estratégia procura o primeiro trem com ATRASO TEÓRICO maior que certa tolerância, classificando-o como trem problema. Para todas as estações à frente deste trem, calcula o ATRASO RECUPERÁVEL VIÁVEL, mínimo entre o ATRASO TEÓRICO do trem detetado na última estação por onde passou e o ATRASO RECUPERÁVEL ESTIMADO a partir da estação até o final do percurso:

$$ARV(EST) = MIN(ATE(TRP), ARE(EST))$$

onde:

ARV(EST) - é o ATRASO RECUPERÁVEL VIÁVEL associado à estação EST.
TRP - é o identificador do trem problema.
ATE(TRP) - é o ATRASO TEÓRICO do trem TRP detetado na última estação por onde passou:
ATE(TRP) = HRC(ESTU,TRP) - (HTC(ESTU,TRP) + DEFSIS)
ESTU - é o identificador da última estação por onde o trem problema passou antes da aplicação desta estratégia.
HRC(ESTU,TR) - é o HORÁRIO REAL DE CHEGADA do trem TR à última estação por onde passou. Este valor é medido pelo processo quando da chegada do trem à estação.
HTC(EST,TR) - é o HORÁRIO TEÓRICO DE CHEGADA do trem TR à estação EST. Este valor é definido pela camada de otimização.
DEFSIS - é o valor atual da DEFASAGEM DO SISTEMA. Este

valor é atualizado pela estratégia de defasagem do sistema.

ARE(EST) - é o ATRASO RECUPERÁVEL ESTIMADO associado à estação EST.
ARE(EST) = $\sum_{I=EST}^{ESTF}$ (TPE(I,NDN) - TPE(I,ND1) + TPN - TPM)
EST - é o identificador da estação
ESTF - é o identificador da estação anterior à estação de final de percurso (estação 5 na configuração da seção 4.3.2.)
TPE(EST,ND) - é o tempo de percursso para um trem ir da estação EST até a estação EST + 1 com nível de desempenho ND.
NDN - é o NÍVEL DE DESEMPENHO NOMINAL definido pela camada de otimização.
ND1 - é o nível de desempenho máximo admitido no sistema - NÍVEL DE DESEMPENHO 1.
TPN - é o TEMPO DE PARADA NOMINAL definido pela camada de otimização.
TPM - é o TEMPO DE PARADA MÍNIMO permitido no sistema (definido a priori).

As modificações a serem efetuadas nos horários teóricos de chegada dos trens às estações envolvidas consistem em incrementá-los da quantidade:

DEFTR(EST,TR)=ARV(EST) * (NTR(EST) + 1 - ORD(EST,TR)) / NTR(EST)

onde:

DEFTR(EST,TR) - é o incremento do HORÁRIO TEÓRICO DE CHEGADA do trem TR á estação EST.
ARV(EST) - é o ATRASO RECUPERÁVEL VIÁVEL na estação EST.
NTR(EST) - é o número de trens que passarão pela estação EST, considerando-se o trem problema como último.
ORD(EST,TR) - é a ordem de passagem do trem TR na estação EST.

Nas figuras 4.3.10 e 4.3.11 são apresentados, respectivamente, o domínio do controle e o domínio dos dados da camada de adaptação. Segue abaixo o domínio da interpretação.

DOMÍNIO DA INTERPRETAÇÃO

INTERPRETAÇÃO DOS ARMAZENADORES

ar71 - Estado trem 1
ar71::=<<atraso teórico><atraso operacional>
<IDENTIFICADOR ESTAÇÃO>>
<atraso teórico>::=<HORA>
<atraso operacional>::=<HORA>

ar72 - Estado trem 2
ar72::=<<atraso teórico><atraso operacional>
<IDENTIFICADOR ESTAÇÃO>>
<atraso teórico>::=<HORA>
<atraso operacional>::=<HORA>

ar73 - Estado trem 3
ar73::=<<atraso teórico><atraso operacional>
<IDENTIFICADOR ESTAÇÃO>>

<atraso teórico>::=<HORA>
<atraso operacional>::=<HORA>

ar74 - Estado trem 4
ar74::=<<atraso teórico><atraso operacional>
<IDENTIFICADOR ESTAÇÃO>>
<atraso teórico>::=<HORA>
<atraso operacional>::=<HORA>

ar75 - Estado trem 5
ar75::=<<atraso teórico><atraso operacional>
<IDENTIFICADOR ESTAÇÃO>>
<atraso teórico>::=<HORA>
<atraso operacional>::=<HORA>

ar76 - Estado trem 6
ar76::=<<atraso teórico><atraso operacional>
<IDENTIFICADOR ESTAÇÃO>>
<atraso teórico>::=<HORA>
<atraso operacional>::=<HORA>

ar77 - Defasagem sistema (estação 1)
ar77::=<HORA>

ar91 - Defasagem sistema (estação 2)
ar91::=<HORA>

ar105 - Defasagem sistema (estação 3)
ar105::=<HORA>

ar119 - Defasagem sistema (estação 4)
ar119::=<HORA>

ar133 - Defasagem sistema (estação 5)
ar133::=<HORA>

ar85 - Defasagem trem 1 (estação 1)
ar85::=<HORA>

ar86 - Defasagem trem 2 (estação 1)
ar86::=<HORA>

ar87 - Defasagem trem 3 (estação 1)
ar87::=<HORA>

ar88 - Defasagem trem 4 (estação 1)
ar88::=<HORA>

ar89 - Defasagem trem 5 (estação 1)
ar89::=<HORA>

ar90 - Defasagem trem 6 (estação 1)
ar90::=<HORA>

ar99 - Defasagem trem 1 (estação 2)
ar99::=<HORA>

ar100 - Defasagem trem 2 (estação 2)

ar100::=<HORA>

ar101 - Defasagem trem 3 (estação 2)
ar101::=<HORA>

ar102 - Defasagem trem 4 (estação 2)
ar102::=<HORA>

ar103 - Defasagem trem 5 (estação 2)
ar103::=<HORA>

ar104 - Defasagem trem 6 (estação 2)
ar104::=<HORA>

ar113 - Defasagem trem 1 (estação 3)
ar113::=<HORA>

ar114 - Defasagem trem 2 (estação 3)
ar114::=<HORA>

ar115 - Defasagem trem 3 (estação 3)
ar115::=<HORA>

ar116 - Defasagem trem 4 (estação 3)
ar116::=<HORA>

ar117 - Defasagem trem 5 (estação 3)
ar117::=<HORA>

ar118 - Defasagem trem 6 (estação 3)
ar118::=<HORA>

ar127 - Defasagem trem 1 (estação 4)
ar127::=<HORA>

ar128 - Defasagem trem 2 (estação 4)
ar128::=<HORA>

ar129 - Defasagem trem 3 (estação 4)
ar129::=<HORA>

ar130 - Defasagem trem 4 (estação 4)
ar130::=<HORA>

ar131 - Defasagem trem 5 (estação 4)
ar131::=<HORA>

ar132 - Defasagem trem 6 (estação 4)
ar132::=<HORA>

ar141 - Defasagem trem 1 (estação 5)
ar141::=<HORA>

ar142 - Defasagem trem 2 (estação 5)
ar142::=<HORA>

ar143 - Defasagem trem 3 (estação 5)
ar143::=<HORA>

ar144 - Defasagem trem 4 (estação 5)
ar144::=<HORA>

ar145 - Defasagem trem 5 (estação 5)
ar145::=<HORA>

ar146 - Defasagem trem 6 (estação 5)
ar146::=<HORA>

ar147 - Informação para o cálculo da defasagem do sistema
ar147::=<<atraso teórico trem 1><estação trem 1>
<atraso teórico trem 2><estação trem 2>
<atraso teórico trem 3><estação trem 3>
<atraso teórico trem 4><estação trem 4>
<atraso teórico trem 5><estação trem 5>
<atraso teórico trem 6><estação trem 6>>
<atraso teórico trem 1>::=<HORA>
<atraso teórico trem 2>::=<HORA>
<atraso teórico trem 3>::=<HORA>
<atraso teórico trem 4>::=<HORA>
<atraso teórico trem 5>::=<HORA>
<atraso teórico trem 6>::=<HORA>
<estação trem 1>::=<IDENTIFICADOR ESTAÇÃO>
<estação trem 2>::=<IDENTIFICADOR ESTAÇÃO>
<estação trem 3>::=<IDENTIFICADOR ESTAÇÃO>
<estação trem 4>::=<IDENTIFICADOR ESTAÇÃO>
<estação trem 5>::=<IDENTIFICADOR ESTAÇÃO>
<estação trem 6>::=<IDENTIFICADOR ESTAÇÃO>

ar148 - Defasagem atual do sistema
ar148::=<HORA>

ar149 - Nível de desempenho nominal
ar149::=<NÍVEL DE DESEMPENHO>

ar150 - Tempo parada nominal
ar150::=<HORA>

ar151 - Informação para o cálculo da distribuição de intervalos
ar151::=<<atraso teórico trem 1><atraso operacional trem 1>
<estação trem 1><atraso teórico trem 2>
<atraso operacional trem 2><estação trem 2>
<atraso teórico trem 3><atraso operacional trem 3>
<estação trem 3><atraso teórico trem 4>
<atraso operacional trem 4><estação trem 4>
<atraso teórico trem 5><atraso operacional trem 5>
<estação trem 5><atraso teórico trem 6>
<atraso operacional trem 6><estação trem 6>
<atraso teórico trem 1>::=<HORA>
<atraso teórico trem 2>::=<HORA>
<atraso teórico trem 3>::=<HORA>
<atraso teórico trem 4>::=<HORA>
<atraso teórico trem 5>::=<HORA>
<atraso teórico trem 6>::=<HORA>
<atraso operaciona trem 1>::=<HORA>
<atraso operaciona trem 2>::=<HORA>
<atraso operaciona trem 3>::=<HORA>
<atraso operaciona trem 4>::=<HORA>

```
<atraso operaciona trem 5>::=<HORA>
<atraso operaciona trem 6>::=<HORA>
<estação trem 1>::=<IDENTIFICADOR ESTAÇÃO>
<estação trem 2>::=<IDENTIFICADOR ESTAÇÃO>
<estação trem 3>::=<IDENTIFICADOR ESTAÇÃO>
<estação trem 4>::=<IDENTIFICADOR ESTAÇÃO>
<estação trem 5>::=<IDENTIFICADOR ESTAÇÃO>
<estação trem 6>::=<IDENTIFICADOR ESTAÇÃO>

ar152 - Nível de desempenho nominal
ar152::=<NÍVEL DE DESEMPENHO>

ar153 - Tempo de parada nominal
ar153::=<HORA>
```

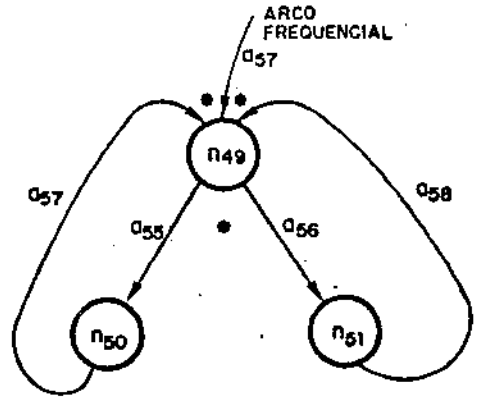


FIG.4.3.10 - MODELO GMB* DA CAMADA DE ADAPTAÇÃO: DOMÍNIO DO CONTROLE

INTERPRETAÇÃO DOS PROCESSADORES

```
INTERPRETAÇÃO p(n49)
BEGIN
 FOR /* TREM:= 1 */ TO /* 6 */ DO
 BEGIN
  /* LE(ar[70+TREM],CONSUMPTIVO,VARIÁVEL$AUXILIAR) */ ;
  /*    desmembre    a    variável    VARIÁVEL$AUXILIAR    em:    ATRA-
  SO$TEÓRICO[TREM]:= <atraso teórico>, ATRASO$OPERACIONAL[TREM]:=
  <atraso operacional> e ESTAÇÃO[TREM]:= <IDENTIFICADOR ESTAÇÃO> */
 END ;
 /*   concatene   as   variáveis   ATRASO$TEÓRICO[1],   ESTAÇÃO[1],
 ATRASO$TEÓRICO[2],   ESTAÇÃO[2],   ATRASO$TEÓRICO[3],   ESTAÇÃO[3],
 ATRASO$TEÓRICO[4],   ESTAÇÃO[4],   ATRASO$TEÓRICO[5],   ESTAÇÃO[5],
 ATRASO$TEÓRICO[6]    e    ESTAÇÃO[6],    formando    a    variável
 INFORMAÇÃO$DEFASAGEM$SISTEMA */ ;
 /* ESCREVE(ar147,DESTRUTIVO,INFORMAÇÃO$DEFASAGEM$SISTEMA) */ ;
 /* concatene as variáveis ATRASO$TEÓRICO[1], ATRASO$OPERACIONAL[1],
 ESTAÇÃO[1], ATRASO$TEÓRICO[2], ATRASO$OPERACIONAL[2], ESTAÇÃO[2],
 ATRASO$TEÓRICO[3],    ATRASO$OPERACIONAL[3],    ESTAÇÃO[3],    ATRA-
 SO$TEÓRICO[4],    ATRASO$OPERACIONAL[4],    ESTAÇÃO[4],    ATRA-
 SO$TEÓRICO[5],    ATRASO$OPERACIONAL[5],    ESTAÇÃO[5],    ATRA-
 SO$TEÓRICO[6], ATRASO$OPERACIONAL[6] e ESTAÇÃO[6], formando a va-
 riável INFORMAÇÃO$DISTRIBUIÇÃO$INTERVALOS */ ;
 /* ESCREVE(ar151,DESTRUTIVO,INFORMAÇÃO$DISTRIBUIÇÃO$INTERVALOS) */;
 /* MARCA(a55,1) */ ;
 /* MARCA(a56,1) */
 END.
```

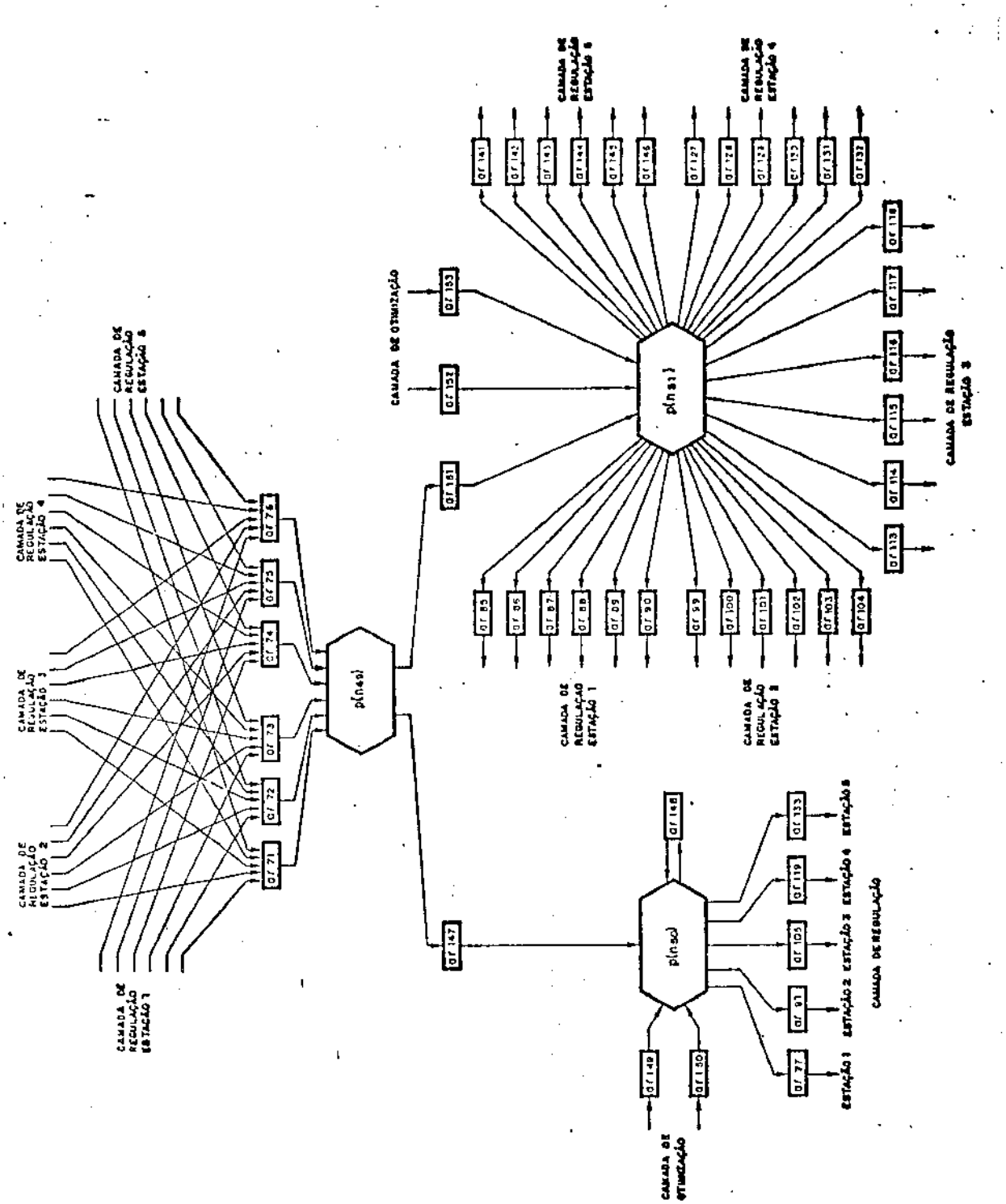


FIG.4.3.11 - MODELO GMB* DA CAMADA DE ADAPTAÇÃO: DOMÍNIO DOS DADOS

```
INTERPRETAÇÃO p(n50) ;
BEGIN
 /* LE(ar147,CONSUMPTIVO,VARIÁVEL$AUXILIAR) */ ;
 /* desmembre a variável VARIÁVEL$AUXILIAR  em:  ATRASO$TEÓRICO[1]:=
 <HORA>,  ESTAÇÃO[1]:= <IDENTIFICADOR ESTAÇÃO>,  ATRASO$TEÓRICO[2]:=
 <HORA>,  ESTAÇÃO[2]:= <IDENTIFICADOR ESTAÇÃO>,  ATRASO$TEÓRICO[3]:=
 <HORA>,  ESTAÇÃO[3]:= <IDENTIFICADOR ESTAÇÃO>,  ATRASO$TEÓRICO[4]:=
 <HORA>,  ESTAÇÃO[4]:= <IDENTIFICADOR ESTAÇÃO>,  ATRASO$TEÓRICO[5]:=
 <HORA>,  ESTAÇÃO[5]:= <IDENTIFICADOR ESTAÇÃO>,  ATRASO$TEÓRICO[6]:=
 <HORA> e ESTAÇÃO[6]:= <IDENTIFICADOR ESTAÇÃO> */ ;
 /* LE(ar149,NÃO-CONSUMPTIVO,NÍVEL$DESEMPENHO$NOMINAL) */ ;
 /* LE(ar150,NÃO-CONSUMPTIVO,TEMPO$PARADA$NOMINAL) */ ;
 FOR /* TREM:= 1 */ TO /* 6 */ DO
 BEGIN
  /* ATRASO$RECUPERÁVEL[TREM]:= 0 */ ;
  FOR /* ESTAÇÃO:= 5 */ DOWTO /* ESTAÇÃO[TREM] */ DO
  BEGIN
   /*  ATRASO$RECUPERÁVEL[TREM]:=  ATRASO$RECUPERÁVEL[TREM]  +  TEM-
   PO$PERCURSO(ESTAÇÃO,NÍVEL$DESEMPENHO$NOMINAL)     - TEMPO$PERCUR-
   SO(ESTAÇÃO,NÍVEL$DESEMPENHO$1) + TEMPO$PARADA$NOMINAL - TEMPO$PA-
   RADA$MÍNIMO */
  END
 END ;
 /* INCREMENTO:= 0 */ ;
 FOR /* TREM:= 1 */ TO /* 6 */ DO
 BEGIN
  /*  ATRASO$IRRECUPERÁVEL[TREM]:= ATRASO$TEÓRICO[TREM] - ATRASO$RE-
  CUPERÁVEL[TREM] */ ;
  IF /* ATRASO$IRRECUPERÁVEL[TREM] > 0 */ THEN
  BEGIN
   /* INCREMENTO:=  INCREMENTO + ATRASO$IRRECUPERÁVEL[TREM] */
  END
 END ;
 IF /* INCREMENTO = 0 */ THEN
 BEGIN
  FOR /* TREM = 1 */ TO /* 6 */ DO
  BEGIN
   /* INCREMENTO:= INCREMENTO + ATRASO$IRRECUPERÁVEL[TREM] */
  END
 END ;
 /* INCREMENTO:= INCREMENTO / 6 */ ;
 IF /* INCREMENTO > TOLERÂNCIA */ THEN
 BEGIN
  /* INCREMENTO:= TOLERÂNCIA */
 END ;
 /* LE(ar148,CONSUMPTIVO,DEFASAGEM$SISTEMA) */ ;
 /* DEFASAGEM$SISTEMA:= DESFASAGEM$SISTEMA + INCREMENTO */ ;
 IF /* DEFASAGEM$SISTEMA < 0 */ THEN
 BEGIN
  /* DEFASAGEM$SISTEMA:= 0 */
 END
 /* ESCREVE(ar148,DESTRUTIVO,DEFASAGEM$SISTEMA) */ ;
 /* ESCREVE(ar77,DESTRUTIVO,DEFASAGEM$SISTEMA) */ ;
 /* ESCREVE(ar91,DESTRUTIVO,DEFASAGEM$SISTEMA) */ ;
 /* ESCREVE(ar105,DESTRUTIVO,DEFASAGEM$SISTEMA) */ ;
 /* ESCREVE(ar119,DESTRUTIVO,DEFASAGEM$SISTEMA) */ ;
 /* ESCREVE(ar133,DESTRUTIVO,DEFASAGEM$SISTEMA) */ ;
 /* MARCA(a57,1) */
```

```
END.

INTERPRETAÇÃO p(n51) ;
BEGIN
 /* LE(ar151,CONSUMPTIVO,VARIÁVEL$AUXILIAR) */ ;
 /* desmembre a variável VARIÁVEL$AUXILIAR em: ATRASO$TEÓRICO[1]:=
 <HORA>, ATRASO$OPERACIONAL[1]:= <HORA>, ESTAÇÃO[1]:= <IDENTIFICA-
 DOR ESTAÇÃO>, ATRASO$TEÓRICO[2]:= <HORA>, ATRASO$OPERACIONAL[2]:=
 <HORA>, ESTAÇÃO[2]:= <IDENTIFICADOR ESTAÇÃO>, ATRASO$TEÓRICO[3]:=
 <HORA>, ATRASO$OPERACIONAL[3]:= <HORA>, ESTAÇÃO[3]:= <IDENTIFICADOR
 ESTAÇÃO>, ATRASO$TEÓRICO[4]:= <HORA>, ATRASO$OPERACIONAL[4]:= <HO-
 RA>, ESTAÇÃO[4]:= <IDENTIFICADOR ESTAÇÃO>, ATRASO$TEÓRICO[5]:=
 <HORA>, ATRASO$OPERACIONAL[5]:= <HORA>; ESTAÇÃO[5]:= <IDENTIFICADOR
 ESTAÇÃO>, ATRASO$TEÓRICO[6]:= <HORA>, ATRASO$OPERACIONAL[6]:= <HO-
 RA> e ESTAÇÃO[6]:= <IDENTIFICADOR ESTAÇÃO> */ ;
 /* ESTADO:= FALSO */ ;
 /* TREM:= 1 */ ;
 WHILE /* (ESTADO = FALSO) AND (TREM <= 6) */ DO
 BEGIN
  IF /* ATRASO$OPERACIONAL[TREM] > TOLERÂNCIA */ THEN
  BEGIN
   /* ESTADO:= VERDADEIRO */
  END ;
  /* TREM:= TREM + 1 */
 END ;
 IF /* ESTADO = VERDADEIRO */ THEN
 BEGIN
  /* PRIMEIRA$ESTAÇÃO:= 0 */ ;
  REPEAT
  BEGIN
   /* PRIMEIRA$ESTAÇÃO:= PRIMEIRA$ESTAÇÃO + 1 */ ;
   /* PRIMEIRO$TREM:= 0 */ ;
   REPEAT
   BEGIN
    /* PRIMEIRO$TREM:= PRIMEIRO$TREM + 1 */ ;
    IF /* ESTAÇÃO[PRIMEIRO$TREM] = PRIMEIRA$ESTAÇÃO */ THEN
    BEGIN
     REPEAT
     BEGIN
      /* PRIMEIRO$TREM:= PRIMEIRO$TREM + 1 */
     END
     UNTIL  /* (ESTAÇÃO[PRIMEIRO$TREM] <> PRIMEIRA$ESTAÇÃO) OR (PRI-
     MEIRO$TREM = 6) */ ;
     IF /* ESTAÇÃO[PRIMEIRO$TREM] <> PRIMEIRA$ESTAÇÃO */ THEN
     BEGIN
      /* PRIMEIRO$TREM:= PRIMEIRO$TREM - 1 */
     END
    END
   UNTIL  /*  (ESTAÇÃO[PRIMEIRO$TREM] = PRIMEIRA$ESTAÇÃO)  OR  (PRI-
   MEIRO$TREM = 6) */
  END
  UNTIL  /* (ESTAÇÃO[PRIMEIRO$TREM] = PRIMEIRA$ESTAÇÃO) OR  (PRIMEI-
  RA$ESTAÇÃO = 5) */;
  IF /* ESTAÇÃO[PRIMEIRO$TREM] = PRIMEIRA$ESTAÇÃO */ THEN
  BEGIN
   /* TREM$PROBLEMA:= PRIMEIRO$TREM */ ;
   WHILE /* ATRASO$TEÓRICO[TREM$PROBLEMA] < TOLERÂNCIA */ DO
```

```
BEGIN
 /* TREM$PROBLEMA:= TREM$PROBLEMA - 1 */ ;
 IF /* TREM$PROBLEMA = 0 */ THEN
 BEGIN
  /* TREM$PROBLEMA:= 6 */
 END
END ;
/* ESTAÇÃO$INÍCIO:= ESTAÇÃO[TREM$PROBLEMA] */ ;
/* LE(ar152,NÃO-CONSUMPTIVO,NÍVEL$DESEMPENHO$NOMINAL) */ ;
/* LE(ar153,NÃO-CONSUMPTIVO,TEMPO$PARADA$NOMINAL) */ ;
/* ATRASO$RECUPERÁVEL$ESTIMADO[6]:= 0 */ ;
FOR /* ESTAÇÃO:= 5 */ DOWTO /* ESTAÇÃO$INÍCIO */ DO
BEGIN
 /* ATRASO$RECUPERÁVEL$ESTIMADO[ESTAÇÃO]:= ATRASO$RECUPERÁVEL$ES-
 TIMADO[ESTAÇÃO+1]  + TEMPO$PERCURSO(ESTAÇÃO,NÍVEL$DESEMPENHO$NO-
 MINAL) -  TEMPO$PERCURSO(ESTAÇÃO,NÍVEL$DESEMPENHO$1) + TEMPO$PA-
 RADA$NOMINAL - TEMPO$PARADA$MÍNIMO */
END ;
FOR /* ESTAÇÃO:= ESTAÇÃO$INÍCIO */ TO /* 5 */ DO
BEGIN
 /* ATRASO$RECUPERÁVEL$VIÁVEL[ESTAÇÃO]:= ATRASO$TEÓRICO[TREM$PRO-
 BLEMA] */ ;
 IF  /*  ATRASO$RECUPERÁVEL$VIÁVEL[ESTAÇÃO]  >  ATRASO$RECUPERÁ-
 VEL$ESTIMADO[ESTAÇÃO] */ THEN
 BEGIN
  /*  ATRASO$RECUPERÁVEL$VIÁVEL[ESTAÇÃO]:= ATRASO$RECUPERÁVEL$ES-
  TIMADO[ESTAÇÃO] */
 END
END ;
FOR /* ESTAÇÃO:= 1 */ TO /* 5 */ DO
BEGIN
 /* NÚMERO$TRENS$PASSARÃO[ESTAÇÃO]:= 0 */
END ;
FOR /* TREM:= 1 */ TO /* 6 */ DO
BEGIN
 FOR /* ESTAÇÃO:= ESTAÇÃO[TREM+1] */ TO /* 5 */ DO
 BEGIN
  /*  NÚMERO$TRENS$PASSARÃO[ESTAÇÃO]:=  NÚMERO$TRENS$PASSARÃO[ES-
  TAÇÃO] + 1 */
 END ;
 /* ORDEM$LINHA[TREM]:= PRIMEIRO$TREM - TREM + 1 */ ;
 IF /* ORDEM$LINHA[TREM] <= 0 */ THEN
 BEGIN
  /* ORDEM$LINHA[TREM]:= ORDEM$LINHA[TREM] + 6 */
 END
END ;
FOR /* ESTAÇÃO:= ESTAÇÃO$INÍCIO */ TO /* 5 */ DO
BEGIN
 /*   NÚMERO$TRENS$DEFASAR[ESTAÇÃO]:=   NÚMERO$TRENS$PASSARÃO[ES-
 TAÇÃO]  - (ORDEM$LINHA[TREM$PROBLEMA] - 1) */ ;
 FOR  /* ORDEM$PASSAGEM:= 1 */  TO   /*  NÚMERO$TRENS$DEFASAR[ES-
 TAÇÃO] */ DO
 BEGIN
  /*  TREM:= TREM$PROBLEMA - NÚMERO$TRENS$DEFASAR[ESTAÇÃO] +  OR-
  DEM$PASSAGEM */ ;
  IF /* TREM <= 0 */ THEN
  BEGIN
   /* TREM:= TREM + 6 */
```

```
      END ;
      /*  FATOR$PONDERAÇÃO[ESTAÇÃO,TREM]:=  (NÚMERO$TRENS$DEFASAR[ES-
      TAÇÃO]   +  1 - ORDEM$PASSAGEM)   /   (NÚMERO$TRENS$DEFASAR[ES-
      TAÇÃO] */ ;
      /*  DEFASAGEM[ESTAÇÃO,TREM]:= FATOR$PONDERAÇÃO[ESTAÇÃO,TREM]  *
      ATRASO$RECUPERÁVEL$VIÁVEL[ESTAÇÃO] */ ;
      /*  ESCREVE(ar[85+((ESTAÇÃO - 1) * 14) + (TREM  - 1)],DESTRUTI-
      VO,DEFASAGEM[ESTAÇÃO,TREM]) */
     END
    END
   END
  END ;
  /* MARCA(a58,1) */
END.
```

O estado inicial do modelo GMB* da camada de adaptação consistente com as seções 4.3.2. e 4.3.3.1. é:

ESTADO INICIAL DO GRAFO DO CONTROLE

1. Conjunto de nós ativos: {Ø}
2. Distribuição de marcas nos arcos de controle:
   Conjunto de arcos marcados com prioridade 1: {a57,a58}
   Obs: Os demais arcos não estão marcados.

ESTADO INICIAL DO GRAFO DOS DADOS

1. Conjunto de processadores ativos: {Ø}
2. Conteúdo dos armazenadores:
   ar71 - contém o estado do trem 1:
   ATRASO$TEÓRICO
   ATRASO$OPERACIONAL
   IDENTIFICADOR$ESTAÇÃO = 5

   ar72 - contém o estado do trem 2:
   ATRASO$TEÓRICO
   ATRASO$OPERACIONAL
   IDENTIFICADOR$ESTAÇÃO = 4

   ar73 - contém o estado do trem 3:
   ATRASO$TEÓRICO
   ATRASO$OPERACIONAL
   IDENTIFICADOR$ESTAÇÃO = 3

   ar74 - contém o estado do trem 4:
   ATRASO$TEÓRICO
   ATRASO$OPERACIONAL
   IDENTIFICADOR$ESTAÇÃO = 2

   ar75 - contém o estado do trem 5:
   ATRASO$TEÓRICO
   ATRASO$OPERACIONAL
   IDENTIFICADOR$ESTAÇÃO = 1

   ar76 - contém o estado do trem 6:
   ATRASO$TEÓRICO
   ATRASO$OPERACIONAL
   IDENTIFICADOR$ESTAÇÃO = 5

ar77 - contém a defasagem atual do sistema (estação 1):
DEFASAGEM$SISTEMA

ar91 - contém a defasagem atual do sistema (estação 2):
DEFASAGEM$SISTEMA

ar105 - contém a defasagem atual do sistema (estação 3):
DEFASAGEM$SISTEMA

ar119 - contém a defasagem atual do sistema (estação 4):
DEFASAGEM$SISTEMA

ar133 - contém a defasagem atual do sistema (estação 5):
DEFASAGEM$SISTEMA

ar148 - contém a defasagem atual do sistema:
DEFASAGEM$SISTEMA

ar149 - contém o nível de desempenho nominal:
NÍVEL$DESEMPENHO$NOMINAL

ar150 - contém o tempo de parada nominal:
TEMPO$PARADA$NOMINAL

ar152 - contém o nível de desempenho nominal:
NÍVEL$DESEMPENHO$NOMINAL

ar153 - contém o tempo de parada nominal:
TEMPO$PARADA$NOMINAL

Obs: Os demais armazenadores estão vazios.

4.3.3.3. **DEFINIÇÃO DAS PRIORIDADES**

A ordem de execução dos processadores do modelo GMB* das camadas de regulação e adaptação foi estabelecida através do estudo do problema associado ao controle do processo metroviário em escala reduzida (em particular, através do estudo e adaptação da solução adotada pelo Metropolitano de São Paulo), onde procurou-se detetar e explicitar o paralelismo existente no processamento a ser efetuado.

Neste modelamento nenhum esforço especial foi envidado para se definir as prioridades associadas às marcas dos arcos de controle (um rápido exame no modelo apresentado revelará que todos os arcos são marcados com prioridade 1). Este modelo como até então proposto, a princípio, se presta tão somente para ser executado por uma máquina GMB* distribuída configurada para atender a uma situação de paralelismo total (o número de módulos da máquina deve ser igual ao número máximo de processadores GMB* que possam estar ativos durante a execução do modelo). Esta restrição, entretanto, limita as configurações possíveis para a máquina GMB* distribuída. Para contornar tal restrição e possibilitar a máxima liberdade na fase de configuração da máquina GMB* distribuída, o modelo em questão deve ser analisado sobre a ótica do uni-processamento, que, por forçar o sequenciamento, exige a definição de prioridades.

No exemplo em consideração, a ação tomada pela camada de regulação após o evento portas fechadas (segunda fase da regulação - partida de trem de estação) faz por merecer a máxima prioridade, uma vez que, por construção, o processo despachará o trem da estação após o fechamento das portas dentro de um limite fixo de tempo, quer receba ou não resposta da regulação. Impor prioridade máxima à segunda fase da regulação garante a geração da resposta esperada pelo processo no menor tempo possível, dentro das limitações da configuração da máquina GMB* adotada.

A segunda maior prioridade de sistema será atribuída à primeira fase da regulação (chegada de trem em estação).

Uma vez que as ações executadas pela camada de adaptação não possuem limitações exigentes no tempo de resposta, associar-se-á a elas prioridade mínima.

Tais considerações sobre as prioridades do processamento são absorvidas pelo modelo GMB* da camadas de regulação e adaptação através das seguintes modificações:

a) os arcos a5, a15, a25, a35 e a45 passam a ser marcados com prioridade 4.
b) os arcos a9, a19, a29, a39 e a49 passam a ser marcados com prioridade 5.

4.3.3.4. **CONFIGURAÇÃO DA MÁQUINA GMB* DISTRIBUÍDA**

Uma vez estabelecidas as prioridades através do hipótese do uni-processamento, o modelo GMB* gerado encontra-se apto a ser executado por qualquer configuração, no que diz respeito ao número de nós, de máquina GMB* distribuída. A definição de uma dada configuração de máquina, assim como dos critérios de distribuição do aplicativo em sua estrutura, são questões intimamente relacionadas com o custo e o desempenho considerados aceitáveis para o sistema. Tais questões devem ser equacionadas e resolvidas caso a caso.

No exemplo em consideração, adotou-se, a título de ilustração, a configuração física apresentada na figura 4.3.12. Para esta configuração física propõe-se a seguinte estratégia de distribuição do aplicativo:

MÁQUINA GMB* UNI-PROCESSADORA M1: CAMADA DE ADAPTAÇÃO.

ELEMENTOS ASSOCIADOS À CAMADA DE ADAPTAÇÃO:
NÓS - n49, n50 e n51.
ARMAZENADORES - ar71, ar72, ar73, ar74, ar75, ar76, ar147, ar148, ar149, ar150, ar151, ar152 e ar153.
PROCESSADORES - p(n49), p(n50) e p(n51).

MÁQUINA GMB* UNI-PROCESSADORA M2: CAMADA DE REGULAÇÃO - ESTAÇÕES 1, 2 e 3.

ELEMENTOS ASSOCIADOS À CAMADA DE REGULAÇÃO - ESTAÇÃO 1:
NÓS - n39 e n40.
ARMAZENADORES - ar61, ar77, ar78, ar79, ar80, ar81, ar82,

ar83, ar84, ar85, ar86, ar87, ar88, ar89, ar90, ar155 e ar156.
PROCESSADORES - p(n39) e p(n40).

ELEMENTOS ASSOCIADOS À CAMADA DE REGULAÇÃO - ESTAÇÃO 2:
NÓS - n41 e n42.
ARMAZENADORES - ar62, ar91, ar92, ar93, ar94, ar95, ar96, ar97, ar98, ar99, ar100, ar101, ar102, ar103, ar104, ar157 e ar158.
PROCESSADORES - p(n41) e p(n42).

ELEMENTOS ASSOCIADOS À CAMADA DE REGULAÇÃO - ESTAÇÃO 3:
NÓS - n43 e n44.
ARMAZENADORES - ar63, ar105, ar106, ar107, ar108, ar109, ar110, ar111, ar112, ar113, ar114, ar115, ar116, ar117, ar118, ar159 e ar160.
PROCESSADORES - p(n43) e p(n44)

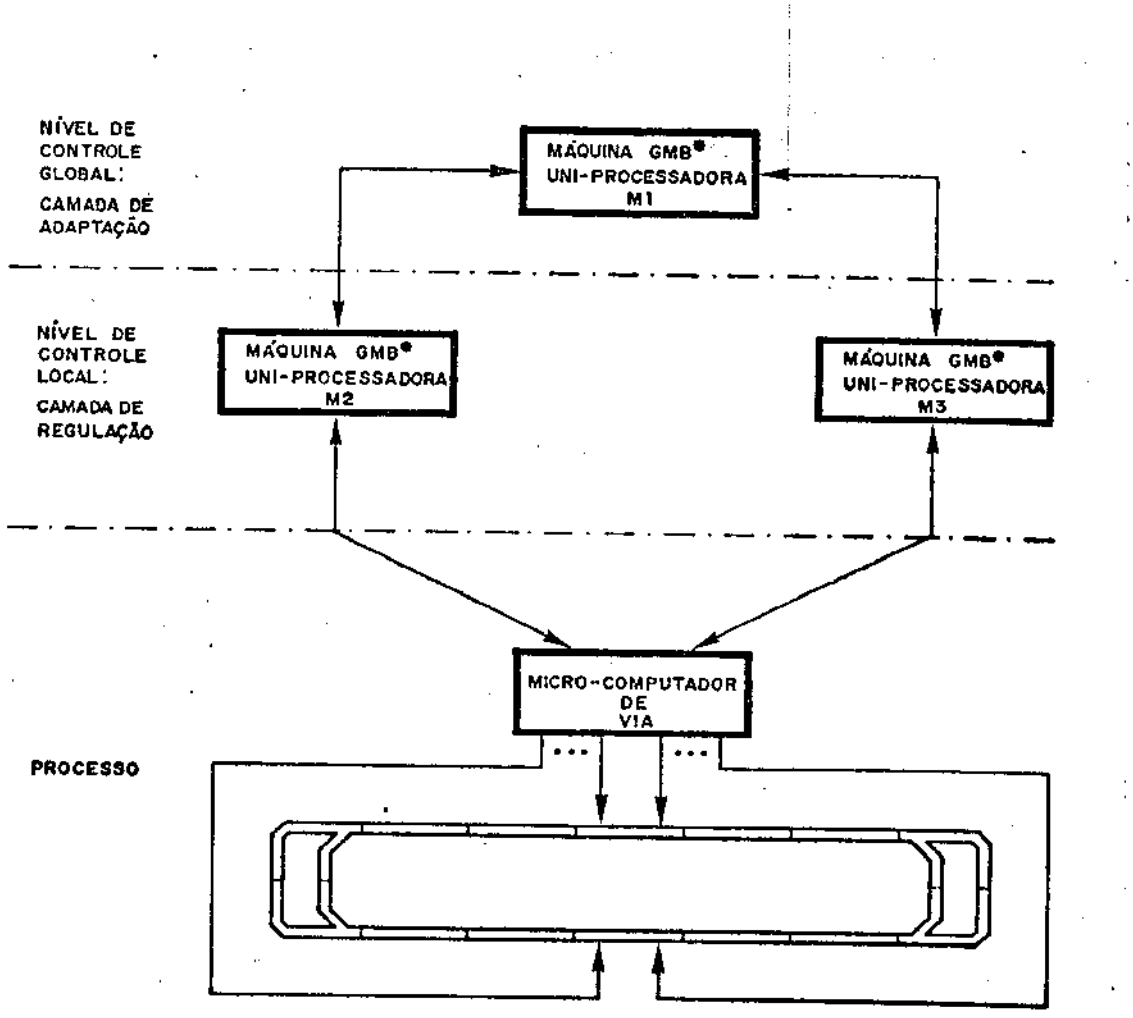


FIG.4.3.12 - CONFIGURAÇÃO DA MÁQUINA GMB* DISTRIBUÍDA PARA O CONTROLE DO SISTEMA METROVIÁRIO EM ESCALA REDUZIDA

MÁQUINA GMB* UNI-PROCESSADORA M3: CAMADA DE REGULAÇÃO - ESTAÇÕES 4 e 5.

ELEMENTOS ASSOCIADOS À CAMADA DE REGULAÇÃO - ESTAÇÃO 4:
NÓS - n45 e n46.
ARMAZENADORES - ar64, ar119, ar120, ar121, ar122, ar123, ar124, ar125, ar126, ar127, ar128, ar129, ar130, ar131, ar132, ar161 e ar162.
PROCESSADORES - p(n45) e p(n46).

ELEMENTOS ASSOCIADOS À CAMADA DE REGULAÇÃO - ESTAÇÃO 5:
NÓS - n47 e n48.
ARMAZENADORES - ar65, ar133, ar134, ar135, ar136, ar137, ar138, ar139, ar140, ar141, ar142, ar143, ar144, ar145, ar146, ar163 e ar164.
PROCESSADORES - p(n45) e p(n46).

A configuração física e lógica proposta, longe de ser uma solução ótima, consoante algum critério, pretende apenas exemplificar a aplicação da máquina GMB* distribuída no controle de um processo, na tentativa de ressaltar as potencialidades desta estrutura.

4.4. CONSIDERAÇÕES FINAIS

A metodologia de desenvolvimento de programas GMB* estabelecida na seção 3.4 define três etapas:

1. Decomposição funcional em módulos (definição do modelo GMB* sem associação de prioridades).
2. Definição das prioridades.
3. Configuração física e lógica da máquina GMB* distribuída.

No exemplo apresentado neste capítulo, a decomposição funcional em módulos baseou-se nos conceitos propostos pela técnica de refinamento passo-a-passo (stepwise refinements) /Pressman 82/.

Em um primeiro momento, a partir do estudo das estratégias de controle adotadas pela Companhia do Metropolitano de São Paulo, estabeleceu-se um modelo GMB* inicial (com alto grau de abstração) para o software de controle do processo metroviário em escala reduzida. Em linhas gerais, este modelo preliminar estabeleceu o escopo da solução a ser adotada, tendo definido a seguinte estrutura: 1) CAMADA DE REGULAÇÃO - um processador GMB* associado à cada estação da linha, tratando a chegada e partida de trens; 2) CAMADA DE ADAPTAÇÃO - um processador GMB* implementando a estratégia de revisão do sistema (estratégia de defasagem no horário do sistema e estratégia de distribuição de intervalos). A partir deste modelo inicial, através de refinamentos sucessivos do grau de abstração dos elementos por ele representado, evolui-se até o nível de detalhamento apresentado na seção 4.3.3..

A técnica de refinamento passo-a-passo estabelece que a arquitetura de um programa deve ser desenvolvida através de refinamentos sucessivos do grau de detalhamento dos procedimentos a serem executados até a completa explicitação dos mesmos em uma linguagem final de programação. Neste processo de refinamento uma hierarquia de representações dos procedimentos é desenvolvida, onde cada nível de representação contempla um detalhamento maior das

abstrações do nível superior.

Na programação GMB*, em particular, o refinamento a ser aplicado não se refere tão somente ao detalhamento procedural, comportanto, também, o detalhamento do grau de paralelismo desejado. Um refinamento de um processador GMB*, no que se refere ao paralelismo, pode ocasionar o desmembramento deste elemento em um conjunto de processadores GMB* com ativação paralela. Se por um lado o refinamento procedural tem como objetivo final a explicitação dos procedimentos em uma linguagem de programação, por outro lado o refinamento do grau de paralelismo não possui um objetivo final independente da aplicação. Antes disso, o paralelismo é um grau de liberdade a mais a ser explorado no desenvolvimento do programa.

Consoante a metodologia proposta na seção 3.4, após a definição do modelo GMB*, independentemente da estratégia de decomposição utilizada, procede-se à definição das prioridades associadas aos processadores do modelo. No exemplo em questão, avaliou-se qualitativamente os requisitos temporais da aplicação e adotou-se a prioridade discutida na seção 4.3.3.3..

Por fim, na seção 4.3.3.4., estabeleceu-se uma configuração física e lógica para a máquina GMB* distribuída. A configuração física do exemplo em questão foi condicionada aos recursos disponíveis no laboratório (LS/M DEE/FEC/UNICAMP) onde se insere este trabalho. A configuração lógica, por sua vez, acompanha funcionalmente a estrutura hierárquica estabelecida pelo Metropolitano de São Paulo. Convém salientar que a configuração lógica adotada não é única e que o custo de uma reconfiguração se resume ao custo do recarregamento da estrutura.

Finalmente, destaque-se que o sistema metroviário em escala reduzida apresentado neste capítulo, mais do que um exemplo ilustrativo, é um sistema em implantação, que, quando em operação, será explorado como um ambiente experimental de ensino e pesquisa na área de controle em tempo-real por computador.

CAPÍTULO 5

CONCLUSÕES

CONCLUSÕES

As aplicações de computadores em controle de processos caracterizam-se pela utilização destas máquinas no controle e supervisão de atividades cuja dinâmica não se adapta a nenhuma base de tempo que possa ser definida e implementada a priori no sistema computacional. Neste tipo de aplicação, faz-se necessário que o sequenciamento das tarefas a serem executadas pelo computador esteja subordinado à ocorrência de eventos externos. Tais eventos são traduzidos para o computador como requisições de processamento (interrupções), e podem ser utilizados para a definição de uma base comum de tempo entre máquina e processo e permitir que este influa diretamente na dinâmica da operação da máquina. O eficiente gerenciamento do atendimento destas requisições de processamento constitui a questão fundamental associada à programação de sistemas de controle por computador.

Em uma estrutura uni-processadora, a ocorrência simultânea de requisições de processamento ocasiona uma competição pelo uso do único recurso de processamento disponível. A existência deste tipo de competição exige a definição de uma política de atendimento das requisições em função do grau de importância (prioridade) a elas associado. Uma vez definidas as prioridades, tal política deve preocupar-se em resolver a competição que tem lugar quando da ocorrência de uma requisição com alto grau de urgência durante a execução de uma tarefa menos prioritária. Uma solução para este tipo de competição pode comportar a suspensão temporária (preempção) do processamento em curso para o atendimento da requisição pendente. Em particular, uma política de gerenciamento de requisições deste tipo não admite mais o estilo de programação sequencial, exigindo os conceitos da programação concorrente.

De uma maneira geral, os programas podem ser classificados em sequenciais e concorrentes. Um programa sequencial caracteriza-se por independer do instante e velocidade de execução, produzindo sempre o mesmo resultado para um mesmo conjunto de dados de entrada. Um programa concorrente, por outro lado, devido à superposição temporal da execução das tarefas que o compõem, pode produzir, na dependência da velocidade relativa de execução destas tarefas, resultados distintos para um mesmo conjunto de dados de entrada. Tal afirmação é particularmente verdadeira, caso as tarefas que coexistem no tempo troquem dados desordenadamente sem a existência de mecanismos apropriados de sincronização (exclusão mútua). Em verdade, a questão básica associada à programação concorrente traduz-se pela necessidade de definição de mecanismos eficientes e seguros de sincronização na troca e acesso a dados de tarefas que coexistam no tempo.

As questões associadas à sincronização e troca de dados entre tarefas concorrentes adquirem um grau de dificuldade a mais ao se considerar estruturas distribuídas de processamento. Em tais estruturas os tempos de produção e materialização de ítens de informação (sincronismo e dados) não são coincidentes para tarefas que residam em máquinas distintas. Esta visão dicotômica da produção/materialização reflete tão somente a existência de um limite físico de velocidade de transporte de mensagens pelos canais de comunicação que interligam os processadores da estrutura

distribuída. Em adição ao atraso de transporte, também deve ser ponderado como elemento complicador dos mecanismos de sincronização e troca de dados de sistemas distribuídos, a probabilidade de perda e distorção de mensagens pelos canais de comunicação.

O presente trabalho abordou a problemática do desenvolvimento do software aplicativo de sistema de controle por computador propondo um ambiente de programação concorrente, definido pela máquina GMB* distribuída, que comporta a coexistência de tarefas em sistemas com característica distribuída. A programação do ambiente em questão é efetuada a partir do modelamento GMB*, onde o programador estabelece as tarefas a serem executadas, os mecanismo de sincronização e troca de dados e a política de escalonamento a ser empregada.

Ao ser aplicado a configurações uni-processadoras, o modelo GMB* descreve, através dos domínio dos dados e da interpretação, o processamento a ser realizado e, através do domínio do controle, a política de execução deste processamento. Em particular, esta política de execução é explicitada pela topologia do grafo do controle e pela dinâmica de geração de marcas e respectivas prioridades. Na caracterização do ambiente GMB* para uma estrutura uni-processadora absorveu-se, como estabelecido no formalismo do modelo, o conceito da não preempção da execução dos processadores GMB*. Tal solução impõe uma política de escalonamento fortemente dependente do programador, onde os instantes de chaveamento de execução de tarefas são explicitados e limitados, com a ciência do programador, aos instantes de desativação dos processadores GMB*. A não preempção, se por uma lado, garante a exclusão mútua dos processadores GMB*, evitando conflito no acesso aos armazenadores, por outro, exige do programador um refinado conhecimento da dinâmica e dos tempos envolvidos durante a execução de seu programa.

Para ilustrar esta discussão considere o seguinte cenário: durante a execução de um processador GMB* uma marca é depositada em arco de controle externo, habilitando um nó de controle com prioridade superior à apresentada pelo nó ativo. Na solução de escalonamento adotada, somente após o término da interpretação associada ao processador em execução, e consequente desativação do nó a ele associado, será efetuado o escalonamento dos nós habilitados, ocasionando a ativação do nó mais prioritário. O mecanismo de escalonamento em questão pode, portanto, no pior caso, retardar o disparo de um nó pelo tempo máximo exigido para a finalização da interpretação associada a outro nó que possa estar ativo no instante de sua habilitação. Neste sentido, o programador, na avaliação dos tempos de respostas dos processamentos críticos no tempo, sempre deverá ponderar um eventual retardo de disparo. Caso a estimativa dos tempos envolvidos revele uma violação no tempo de resposta máximo exigido pela aplicação, o programador poderá aumentar a frequência do escalonamento, reduzindo, consequentemente, o atraso de disparo, desmembrando o nó (ou nós) responsável pelo retardo em conjunto de nós com fluxo de controle sequencial.

Para contornar a necessidade da estimativa do tempo de retardo associado aos processamentos críticos no tempo, visualiza-se como possível solução o relaxamento da restrição de não preempção dos nós. Esta nova política de escalonamento define uma nova versão para o núcleo da máquina GMB* uni-processadora, desta feita comportando a

multi-programação. A definição desta nova versão é deixada para um trabalho futuro que, entre outros aspectos, deverá ponderar a necessidade da criação de mecanismos para a definição de regiões críticas (trechos não preemptíveis) no interior da interpretação dos processadores GMB* e o impacto da multi-programação na análise do modelo, pois agora, o tempo de execução de um processador GMB* não dependerá somente do conjunto de dados de entrada e do estado do sistema no instante da ativação, mas dependerá também, da evolução deste estado durante a sua interpretação.

Ao ser aplicado a configurações distribuídas, o modelo GMB* descreve globalmente o sistema, explicitando o processamento a ser realizado em cada módulo da estrutura distribuída, assim como, os mecanismos de interação entre eles. Na estrutura distribuída a camada aplicativa é constituída de modelos GMB* locais que trocam dados e marcas com o objetivo de realizar a função global do sistema. Os modelos GMB* locais descrevem os processamentos a serem realizados e a política de execução destes processamentos em cada módulo uni-processador da estrutura distribuída. A interconexão destes modelos GMB* locais define os mecanismos de sincronização (interconexão dos grafos do controle) e de troca de dados (interconexão dos grafos dos dados).

A existência de um modelo que descreve globalmente o comportamento da camada aplicativa de um sistema distribuído, permite a análise global do sistema em desenvolvimento, com vistas à validação e testes do projeto em curso. Atualmente, a análise dos modelos GMB* aplicativos vem sendo realizada de forma manual sem o apoio de ferramenta adequada, mostrando-se extremamente ineficaz à medida que a complexidade dos modelos a serem analisados aumenta. Neste sentido, sugere-se como uma possível linha de continuação deste trabalho a investigação e o desenvolvimento de técnicas e ferramentas automáticas para a análise do modelo GMB*. Como exemplo de ferramenta de análise pode-se citar um simulador automático GMB*. Como ponto de partida para o desenvolvimento de técnicas formais de análise sugere-se a leitura de /Peterson 81/.

Associado à definição do ambiente de programação GMB*, estabeleceu-se também uma metodologia para o projeto da camada aplicativa da máquina GMB* distribuída. Tal metodologia subdivide-se nas seguintes etapas:

1) DECOMPOSIÇÃO DO SISTEMA EM MÓDULOS:

1.1) DECOMPOSIÇÃO FUNCIONAL DO SISTEMA (definição do domínio do controle GMB*): A decomposição funcional do sistema em módulos é norteada pela interdependência dos módulos e pelo grau do paralelismo desejado para o sistema. Tais parâmetros podem ser avaliados qualitativamente segundo critérios de coesão e acoplamento /Magalhães 86/. Coesão é a medida da unidade funcional de um módulo, ou seja, um módulo altamente coeso deve desempenhar idealmente uma única função. Acoplamento é a medida da independência dos módulos. O grau de acoplamento pode ser avaliado pelo volume e diversidade de dados trocados entre módulos. Módulos fracamente acoplados trocam poucos dados de um único tipo.

domínio dos dados GMB*): As interfaces entre módulos que compõem o sistema são estabelecidas no formalismo GMB* através do elemento armazenador. Tal elemento pode ser associado a um tipo de dado, estabelecendo critérios para testes de consistência da troca de informação entre módulos.

1.3) ESPECIFICAÇÃO DA INTERPRETAÇÃO (domínio da interpretação GMB*): Complementando e documentando as sub-etapas anteriores tem-se a especificação das interpretações associadas aos módulos e às suas interfaces.

2) DEFINIÇÃO DAS PRIORIDADES ASSOCIADAS AOS MÓDULOS: Nesta etapa é efetuada a análise dos requisitos temporais da aplicação. Em função desta análise e da decomposição funcional adotada na fase anterior são identificadas as prioridades dos módulos que compõem o sistema. Tais prioridades são absorvidas pelo modelo GMB* definindo a política de escalonamento do processamento.

3) CONFIGURAÇÃO FÍSICA E LÓGICA DO SISTEMA: Finalmente, a última etapa da metodologia consiste na configuração física e lógica do sistema. A configuração física refere-se à definição da topologia da máquina GMB* distribuída (quantidade de módulos, posicionamento geográfico e interligação). A configuração lógica define a estratégia de distribuição do aplicativo pela estrutura física. Em linhas gerais, a configuração do sistema deve ser guiada pela distribuição geográfica dos pontos de demanda de processamento. Uma vez estabelecida a configuração do sistema física e lógica podem-se fazer necessárias, a critério do grau de confiabidade/desempenho exigidos pela aplicação, transformações localizadas no modelo GMB* definido nas fase anteriores. Tais transformações teriam como objetivo a definição de mecanismos mais confiáveis de sincronismo e troca de dados de forma a garantir a interação entre os módulos físicos da máquina GMB* distribuída, contornando os problemas advindos da existência de canais físicos de comunicação entre módulos capazes de destruir, distorcer e atrasar mensagens.

A metodologia apresentada apoia-se no formalismo GMB* estabelecendo uma sistemática para a confecção da camada aplicativa de sistemas distribuídos de controle digital.

A definição de uma metodologia algorítmica para a programação GMB*, entretanto, coloca-se como um objetivo a ser perseguido. Some-se a isto, a problemática do teste e validação da solução final. Cada um destes problemas estabelece, em linhas gerais, um aspecto do modelamento GMB* a ser futuramente explorado.

A motivação inicial para o presente trabalho, deu-se no contexto da implementação de uma ferramenta experimental de ensino e pesquisa na área de controle de processos por computador. Tal ferramenta foi concebida na forma de uma rede local de micro-computadores dedicada à tarefa de controle e supervisão de um ferromodelo (capítulo 4). Neste texto, discutiu-se o ambiente de programação a ser implementado em tal ferramenta (capítulo 3). Este

ambiente define o formalismo a ser utilizado na programação da rede por um usuário interessado em controlar e supervisionar a operação do ferromodelo (capítulo 2).

Somado ao formalismo de programação definido neste trabalho, seria desejável que o usuário tivesse acesso a ferramentas de apoio ao desenvolvimento de programas. A título de exemplo pode-se citar:

1) FERRAMENTAS DE APOIO Á IMPLEMENTAÇÃO DE PROGRAMAS GMB*:

1.1) EDITOR GMB*: Elemento de apoio ao usuário para a edição dos três domínio GMB* e das regras de associação.

1.2) TRADUTOR GMB*: Elemento responsável pela tradução do programa GMB* fonte, confeccionado com o editor GMB*, em programa GMB* objeto. Um programa GMB* objeto é constituído pela interpretação dos processadores em código executável e pelas tabelas que descrevem os domínios do controle e dados no interior da máquina GMB* distribuída.

1.3) CONFIGURADOR GMB*: Elemento de apoio à fase de configuração e carga do sistema. Tal elemento deve oferecer mecanismos para a definição da topologia da rede e da estratégia de distribuição do modelo GMB* aplicativo. Em cada módulo da máquina GMB* distribuída, conjuntamente com os trechos de programas a ele associado, o configurador deve cuidar do carregamento das tabelas de descrição da topologia da rede e distribuição do aplicativo. Entre outras facilidades o configurador pode oferecer, também, mecanismos para a configuração do tamanho das filas de mensagens do serviço de comunicação.

2) FERRAMENTAS DE APOIO AO TESTE DE PROGRAMAS GMB*:

2.1) SIMULADOR GMB*: Através da simulação o usuário pode avaliar as características dinâmicas do programa em desenvolvimento, detetando eventuais erros de projeto. A simulação pode servir de suporte à avaliação das estruturas de dados (por exemplo: dimensionamento dos armazenadores e testes de consistência na troca de informações) e de controle (por exemplo: deteção de deadlock e trechos nunca executados) do programa.

2.2) DEPURADOR DINÂMICO GMB*: Elemento a ser carregado conjuntamente com o modelo GMB* aplicativo. Este elemento teria por função reportar ao usuário a dinâmica de evolução do seu programa no interior da máquina GMB* distribuída.

Tais ferramentas, entre outras, coerentemente com o esperado de um sistema voltado à experimentação, tem por objetivo oferecer ao usuário um ambiente confortável e versátil para o desenvolvimento de programas GMB*, constituindo uma evolução natural da sistema em desenvolvimento.

Finalmente, convém ressaltar que o sistema apresentado não define um solução fechada e completa para os problemas associados aos sistemas de controle de processos por computador. Antes disso, é

um ambiente de experimentação que pretende contribuir na identificação, e tanto quanto possível avançar nas soluções, dos problemas associados a esta classe de sistemas.

REFERÊNCIAS

REFERÊNCIAS

/Astrom 85/ K.J.Astrom; Process Control - Past, Present and Future; IEEE Control Systems Magazine; Volume 5, Number 3, ISSN 0272-1708; August 1985.

/Baer 70/ J.L.Baer, D.P.Bovet, G.Estrin; Legality and other Properties of Graph Models of Computations; Journal of the Association for Computing Machinery ACM, Vol.17, No.3, July 1970.

/Bergamaschi 82/ F.A.Bergamaschi, B.E.A. Milani, T.C.Hsin; Geração de Horários de Despacho de Trens em Linhas de Transporte Metroferroviário; Anais do 4o. Congresso Brasileiro de Automática; Campinas; setembro 1982.

/Berge 76/ C.Berge; Graphs and Hypergraphs; North-Holland Company, Amsterdam; 1976.

/Chen 76/ W.K.Chen; Applied Graph theory - Graphs and Eletrical Networks; North-Holland Company, Amsterdam; 1976.

/Cury 79/ J.E.R.Cury; Metodologia para a Geração Automática de "Programa Horário" Otimizado, para a Linha Metroviária de São Paulo; Dissertação de Mestrado; Campinas UNICAMP/FEC/DEE; março 1979.

/Davies 79/ D.W.Davies, D.L.A.Barber, N.L.Price, C.M.Solomonides; Computer Network and their Protocols; John Wiley and Sons; 1979.

/Delgado 84/ A.L.N.Delgado; Construção da Via de um Modelo Metro-ferroviário, 4o. Relatório de Atividades FAPESP; Janeiro 1984.

/De Martino 83/ J.M.De Martino; Relatório Técnico; Setor de Computação e Automação Industrial, DEE/FEC/UNICAMP; outubro 1983.

/De Martino 1 84/ J.M.De Martino, S.A.Spinela; Cartão de Interface Série IS18251A-V.1 Manual do Usuário; Laboratório de Sistema / Mini e Micro-computadores; Setor de Computação e Automação Industrial, DEE/FEC/UNICAMP; Junho 1984.

/De Martino 2 84/ J.M.De Martino, M.Marton, M.Marton; Cartão de Interface Série IS28251A-V.1 Manual do Usuário; Laboratório de Sistemas / Mini e Micro-computadores; Setor de Computação e Automação Industrial, DEE/FEC/UNICAMP; Junho 1984.

/Farber 78/ G.Farber; Principles and Applications of Decentralized Process Control Computer Systems; A Link Between Science and Applications of Automatic Control; Preprints of the Seventh Triennial World Congress of the International Federation of Automatic Control, Vol.1; Helsink, Finland; June 1978.

/Gomide 84/ F.A.C.Gomide; Princípios de Automática; Uma Introdução à Automação e á Informática; Notas de Aula da Disciplina IA 331 - Controle em Tempo-Real por Computador; DEE/FEC/UNICAMP; 1o. semestre 1984.

/Huberman 76/ L.Huberman; História da Riqueza do Homem; Tradução: W.Dutra; Zahar Editores; 12a. Edição; Rio de Janeiro; 1976.

/Intel 81/ Intel Component Data Catalog; Intel Corporation, Literature Department; January 1981.

/Kramer 83/ J.Kramer, J.Magee, M.Sloman, A.Lister: Conic: An Integrated Approach to Distributed Computer Control Systems; IEE Proc. Vol. 130. Pt.E No.1; January 1983.

/Lages 86/ N.A.C.Lages, J.M.S.Nogueira: Introdução aos Sistemas Distribuídos; Campinas: Editora da UNICAMP; 1986.

/LeLann 83/ G.LeLann: Distributed Systems - Architecture and Implementation, An Advanced Course: Motivations, Objectives and Characterization of Distributed Systems; Lecture Notes on Computer Science; Springer-Verlag; 1983

/Magalhães 81/ L.P.Magalhães - Graphisch-Interaktive Mensch-Maschine Kommunikation in der Prozessautomatisierung: Problemanalyse und Hardware/Softwatre Loesung; Dissertation Dr.Ing.; Alemanha Ocidental, T.H.Darmstadt; 1981.

/Magalhães 86/ M.F.Magalhães; Software para Tempo Real: Campinas: Editora Unicamp; 1988.

/Martin 67/ D.Martin, G.Estrin; Models of Computations and Systems - Evaluation of Vertex Probabilities in Graphs Models of Computations; Journal of the Association for Computing Machinery ACM, Vol.14, No.12; April 1967.

/Metrô 75/ Grupo de Controle em Tempo-Real do DEE/FEC/UNICAMP; Análise dos Programas de Controle de Trens ao Nível de Estratégias, Relatório da Tarefa 5 do Convênio Metrô/UNICAMP PS-2/160; Campinas; maio 1975.

/Metrô 76/ W.C.Amaral et al.; Estudo de Métodos Operacionais para Situações Anormais da Linha Norte-Sul, Relatório Final Convênio Metrô-Unicamp 9233/75/2/50; setembro 1976.

/Metrô 80/ ESCA Engenharia de Sistemas de Controle e Automação S/C Ltda; Especificação Funcional de Simulador para Testes de Estratégias da Linha Norte/Sul, Documento Técnico; São Paulo Janeiro 1980.

/Netto 79/ M.A.Netto, L.G.Latre, M.Jino, M.Mendes; Controle Distribuído e Descentralizado de Processos Industriais por Computador, Parte 1: O que é Controle Distribuído ?; XII Congresso Nacional de Processamento de Dados, SUCESU; São Paulo; outubro 1979.

/Peterson 74/ J.L.Peterson, Bredt; A Comparision of Models of Parallel Computation; Information Processing 74, Proceedings of the IFIP Congress, Amsterdam North-Holland; August 1974.

/Peterson 81/ J.L.Peterson; Petri Net Theory and the Modeling Of Systems; Prentice-Hall, Inc., Englewood Cliffs, N.J.07632; 1981.

/Pressman 82/ R.S.Pressman; Software Engineering: A Practitioner's Approach; MacGraw-Hill, Software Engineering and Technology Series; International Student Edition; 1982.

/Read 72/ Read: Graph Theory and Computing: Academic Press, Inc., USA: 1972.

/Rugglero 78/ W.Rugglero: Distributed Data and Control Driven Machine: Programming and Architecture, Dissertação de Ph.D.: Computer Science Departament, School of Engineering and Aplied Science: University of California,Los Angeles: November 1978.

/Shimisu 81/ E.Y.Shumisu, G.Rodrigues: Introdução ao Hardware para Controle de Processos: Anais 1o. Simpósio de Controle de Processos por Computador, SICOP: Rio de Janeiro: maio 1981.

/Shin-Ting 84/ W.Shin-Ting - MERB: Um Modelo para Aplicação em Controle de Processos: Dissertação de Mestrado: Campinas UNICAMP/FEC/DEE: setembro 1984.

/Tanenbau 81/ A.S.Tanenbaum: Computer Networks: Prentice-Hall Software Series: 1981.