GERENCIADOR DE INTERFACES GRÁFICAS PARA

SISTEMAS DE TEMPO REAL

Norberto Ribeiro Bellas

TESE SUBMETIDA AO CORPO DOCENTE DA COORDENAÇÃO DOS PROGRAMAS DE POS-GRADUAÇÃO DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA DE SISTEMAS E COMPUTAÇÃO.

Aprovada por:

Prof.Ronaldo Cesar Marinho Persiano,D.Sc

(Presidente)

Prof.Aquilino Senra Martinez, D.Sc

Prof.Rubens Nascimento Mello, D.Sc

RIO DE JANEIRO, RJ - BRASIL

ABRIL DE 1990

BELLAS, NORBERTO RIBEIRO

Gerenciador de Interfaces Gráficas para Sistemas de Tempo Real [Rio de Janeiro] 1990.

vii, p. 121 29,7cm (COPPE/UFRJ, M.Sc., Engenharia de Sistemas e Computação, 1990)

Tese - Universidade Federal do Rio de Janeiro, COPPE

1. Gerenciador de Interações Homem-Máquina e de funções Gráficas Aplicadas à Sistemas de Tempo Real I.COPPE/UFRJ II. Título (série).

Resumo da Tese apresentada à COPPE/UFRJ como parte dos requisitos necessários para obtenção do grau de Mestre em Ciências (M.Sc.)

GERENCIADOR DE INTERFACES GRÁFICAS PARA SISTEMAS DE TEMPO REAL

Norberto Ribeiro Bellas

Abril de 1990

Orientador: Ronaldo Cesar Marinho Persiano

Programa : Engenharia de Sistemas e Computação

Este trabalho teve por objetivo a conceituação e definição de um gerenciador de interfaces gráficas para sistemas de tempo real. Como conseqüência do trabalho foi desenvolvido um "software" genérico, aplicado a sistemas de tempo real, que permite ao projetista do sistema aplicativo o uso de ferramentas de interação homem-máquina. Estas ferramentas poderão ser usadas na definição do ambiente interativo das funções gráficas necessárias para um sistema de tempo real de monitoração e controle de processos.

Abstract of Thesis presented to COPPE/UFRJ as partial fulfillment of the requirements for the degree of Master of Science (M.Sc.).

GRAPHIC INTERFACE MANAGER APPLIED TO REAL-TIME SYSTEMS

Norberto Ribeiro Bellas

April of 1990

Thesis Supervisor: Ronaldo Cesar Marinho Persiano

Department : System and Computing Engineering

The conception and definition of a graphic interface manager applied to real-time systems, was the main purpose of the present work. Based on this work it was developed a software applied to real-time systems, which allows to the system analyst the use of man-machine interaction tools. These tools may be used for the definition of the interactive environment, and also for the definition of graphic functions that are necessary in a real-time system for process control and monitoring.

I N D I C E

CAPITULO I

INTRODUÇÃO

Este trabalho é um estudo que tem como objetivo central a conceituação e implementação de um gerenciador de interações homem-máquina aplicado a processos de tempos críticos (sistemas de tempo real).

A implementação final do sistema desenvolvido foi denominada de GIGSTR - Gerenciador de Interfaces Gráficas para Sistemas de Tempo Real, e tem aplicabilidade voltada em especial para área de monitoração de processos tecnológicos.

É um fato incontestável a grande evolução em termos de técnicas de interação homem-máquina durante a década de 80. Tal evolução deu-se principalmente pela utilização de recursos gráficos de modo a permitir uma interação mais "amigável" entre o usuário e o sistema computacional aplicativo. O advento da difusão de microcomputadores, associado ao aumento dos recursos gráficos, tanto em resolução como em velocidade, representou uma nova visão das formas de interação do homem com o computador, visão esta que pode ser constatada nos conceitos interativos, introduzidos pelo "Macintosh", "Presentation Manager", "Windows" e etc.

Consideramos natural que em função do aspecto do mercado tecnológico, estes novos conceitos de interação homem-máquina foram definidos e implementados para situações genéricas, enfocando o atendimento a segmentos de grande volume comercial.

A motivação inicial para realização deste trabalho originou-se da constatação que dentro de uma determinada classe de sistemas, denominados de tempo real, em particular nos sistemas de monitoração e controle de processos,

não existiu a mesma evolução em termos de interação homem-máquina que foi notada dentro do contexto de sistemas comerciais genéricos.

Afim de podermos identificar quais os pontos críticos existentes nos sistemas de tempo real, que dificultam a implementação de um gerenciador de interfaces homem-máquina gráfico e que represente a atual tendência evolutiva dos demais segmentos, apresentamos no Capítulo II uma descrição superficial do ambiente de um sistema de tempo real.

Em função das características dos sistemas de tempo real, podemos relacionar as seguintes dificuldades que envolvem a implementacão de um gerenciador de interfaces homem-máquina gráfico:

- Os sistemas de tempo real são em geral altamemte dependentes do "hardware", isto é, a solução sistêmica é função do "hardware" utilizado, criando deste modo uma dificuldade para a aplicação de soluções genéricas, em especial com relação a parte gráfica.

- Os sistemas de tempo real devem operar de forma otimizada com relação aos parâmetros de tempo de processamento e armazenamento de memória, tornando-se difícil a aplicabilidade, na maioria dos casos, de pacotes gráficos genéricos, tais como o "GKS" e o "CORE".

- Normalmente é necessário o estabelecimento de padrões específicos de interação homem-máquina, de acordo com o perfil do usuário e com o tipo de aplicação de tempo real. Por exemplo, sistemas de tempo real voltados para a segurança de instalações nucleares, devem considerar os aspectos de interação homem-máquina em função do perfil do usuário, no caso o operador do reator nuclear, de modo a garantir rapidez e confiabilidade na interpretação das informações.

Atualmente as soluções genéricas adotadas para os sistemas de tempo real, não englobam grande parte das

facilidades de interação homem-máquina gráficas disponíveis em outras áreas, principalmente em função do alto grau de parametrização da solução, o que em geral restringem as opções de desenvolvimento do "SOFTWARE" aplicativo.

A partir da análise dos problemas relacionados a interação homem-máquina em sistemas de tempo real, e na tentativa de propor uma solução, baseamos o desenvolvimento do GIGSTR em duas linhas principais:

- O GIGSTR deve possuir a capacidade de permitir que suas funções gráficas básicas operem em tempo real, de modo a possibilitar a construção de uma interface homem-máquina gráfica genérica, o qual, possa ser aplicada a estes tipos de processos. Com o objetivo de atender a este requisito o GIGSTR foi projetado utilizando conceitos de tarefas concorrentes e priorizáveis, e mecanismos de exclusão mútua, sincronização e deteção de "over run". Ainda dentro desta linha o GIGSTR procura encapsular as funções que são diretamente ligadas ao "hardware", por exemplo, acesso ao processador gráfico, em "drivers" de modo a tornar o pacote gráfico portátil e o mais independente do "hardware" possível.

- O GIGSTR deve possibilitar ao projetista do sistema aplicativo o uso de ferramentas de interação homem-máquina gráficas sem limitar as soluções que melhor atendam à aplicação. Isto é feito tornando disponível ao projetista não somente as funções gráficas de "alto nível" (barras, gráficos, históricos e etc.), que são usadas para compor um ambiente de interação homem-máquina independente dos programas aplicativos, como também permitindo o acesso "controlado", pelo gerenciador, dos programas aplicativos às primitivas gráficas básicas. Este acesso "controlado" às primitivas gráficas básicas vem preencher eventuais necessidades de algum aplicativo, que não são totalmente supridas pelas funções gráficas de "alto nível".

A arquitetura do GIGSTR foi projetada para operar como uma estação gráfica de um computador hospedeiro

qualquer. Tal arquitetura é particularmente interessante para aplicações de processos, que dependem de uma capacidade computacional relativamente alta para a execução de modelos aplicativos complexos e com uma taxa significativa de operações com estruturas de banco de dados. Dentro dessas necessidades, um dos conceitos fundamentais para a otimização da utilização da estação gráfica em questão, é que os valores por ela representados, seja na forma gráfica ou digital, reflitam em tempo real as possíveis variações que ocorrem dentro de um banco de dados aplicativo do computador hospedeiro.

As características de tempo real do GIGSTR encontram-se distribuidas em dois módulos distintos: no ambiente gerenciador de interfaces homem-máquina e no protocolo de comunicação entre a estação gráfica e o computador hospedeiro. O primeiro destes módulos do GIGSTR opera no ambiente do computador hospedeiro sendo responsável pela conexão, via protocolo de comunicação, entre o conjunto banco de dados/programas aplicativos e o conjunto de tarefas do GIGSTR instalado na estação gráfica. O segundo módulo do GIGSTR, é formado pelo conjunto de tarefas concorrentes e priorizáveis, residentes na estação gráfica, que são responsáveis pela operação em tempo real das funções de interação homem-máquina gráficas. É importante notar que no ambiente da estação gráfica, todo o contexto de tarefas de tempo real encontra-se sob responsabilidade do GIGSTR, sendo que o caráter interativo da aplicação é refletido apenas pelas especificações do projetista e não por programas aplicativos executáveis.

Os conceitos e métodos utilizados para o desenvolvimento do GIGSTR encontram-se descrito neste trabalho na forma de capítulos.

No Capítulo II, procuramos dar uma visão geral sobre sistemas de tempo real, abordando os principais requisitos de implementação para esta classe de sistemas.

No Capítulo III, apresentamos definições conceituais para ambiente de interface homem-máquina, enfocando conceitos gerais para especificação de telas e estruturas de diálogo.

No Capítulo IV, apresentamos conceitos básicos para desenvolvimento do GIGSTR e noções sobre sistemas de gerência de interface com o usuário (SGIU). A seguir descrevemos, no Capítulo V, a descrição funcional do GIGSTR.

No Capítulo VI, apresentamos a implementação do GIGSTR em um ambiente computacional formado por um microcomputador "PC/AT" (Estação Gráfica) comunicando-se com um "Microvax" que tem a função do computador hospedeiro.

No Capítulo VII concluímos, pela observação das funções já implementadas no sistema, os resultados alcançados.

No Anexo I, apresentamos uma descrição dos dados utilizados pelo GIGSTR.

CAPITULO II

CONCEITOS GERAIS SOBRE SISTEMAS DE TEMPO REAL

Muitos autores propuseram definições sobre o que é um sistema de tempo real, sempre fazendo uma divisão entre o "sistema controlado" (função de "hardware") e o "sistema de controle" (função de "software") ALLWORTH [1], ou entre o "software de tempo real" (função de "software") e o "sistema de tempo real" (função de "hardware") GLASS [2]. A divisão entre o "hardware" e o "software" de um sistema de tempo real permite uma análise do que seja o sistema de uma forma mais simples, permitindo identificar de uma maneira mais fácil, as diferenças entre os sistemas de tempo real e os sistemas que processam em lote.

O que caracteriza um sistema de tempo real é seu tempo de resposta à mudanças no ambiente controlado, que é, em geral, da ordem de milisegundos ou no máximo de três segundos dependendo do ambiente controlado. A necessidade de responder ao ambiente, isto é, receber a indicação de mudança, processá-la e mandar a resposta, em um intervalo de tempo o mais curto possível, e o fato de que muitas aplicações envolvem o controle de processos, faz com que os "softwares" desenvolvidos para estes sistemas tenham características específicas, tais como:

- Confiabilidade: um sistema de tempo real tem de ser confiável, isto é, o tempo médio entre falhas (MTBF "Mean Time Between Failure") deve ser o maior possível e o tempo médio de reparo (MTTR "Mean Time to Repair") deve ser o menor possível; além disso, o sistema pode falhar, mas sem provocar uma parada total, ou seja, poder oferecer uma operação degradada mas que seja utilizável.

- Reatividade: O sistema deve reagir às mudanças no ambiente o mais rápido possível, ou seja, o tempo que uma

transação permanece no computador, desde que a informação seja totalmente recebida até que se comece a transmitir a resposta, deve ser o mínimo possível.

- Abrangência: O sistema deve prover ações corretas para todas as possíveis situações, que eventualmente possam ocorrer no ambiente controlado, ou no próprio sistema. Determinar todas as possíveis ocorrências no ambiente controlado ou do próprio sistema, pode ser uma das tarefas mais difíceis do projeto do sistema.

- Economia: Um sistema de tempo real, devido a natureza de sua utilização e de seu custo, que é proporcional a sua complexidade, deve ser tratado como os produtos comerciais, ou seja, deve otimizar suas acões dentro da lógica do custo das mesmas. Nem sempre a melhor (ou mais rápida) solução é utilizada, requerendo mais tempo de análise para se chegar a melhor relação de custo e benefício.

II.1. COMPLEXIDADE DE EQUIPAMENTOS

A complexidade dos equipamentos necessários às aplicações de tempo real pode variar muito desde um equipamento que tenha apenas um periférico (terminal) de entrada, até uma configuração complexa com muitos periféricos de entrada, distribuídos por locais inclusive geograficamente distantes entre si, como, por exemplo, um sistema de reserva de passagem aérea.

Para sistemas que necessitam de uma disponibilidade de praticamente 100%, é recomendável a utillização de uma configuração redundante, tanto para o processamento (CPU) quanto para as linhas de comunicação. Uma configuração redundante envolve não só um custo maior, mas também um supervisor do sistema mais complexo, mas principalmente, em razão de alguma falha, o sistema deve ser trocado automaticamente pela sua duplicata. O custo relativo

do "hardware" necessário para a duplicação do sistema, deve ser comparado com os prejuízos decorrentes de uma eventual parada do sistema para reparos, o que pode acontecer nos momentos mais imprevisíveis.

II.2. TEMPO DE RESPOSTA NECESSÁRIO (Intervalo entre Eventos)

O tempo de resposta em sistemas computacionais pode variar muito, dependendo da aplicação (por exemplo, o balconista de uma agência de viagens deseja uma resposta do sistema de reserva de passagens aéreas da ordem de três segundos, já para controlar uma plataforma de petróleo, cinco minutos são suficiente e para mandar instruções para a loja de uma indústria, meia hora é um tempo razoável). Porém, em muitos sistemas de controle, o tempo de resposta exigido é da ordem de milisegundos, ou menos. Este tempo inclui a interrupção do que o computador estava fazendo, do tratamento da entrada, do processamento da mesma e do envio da resposta, se for o caso. Ao chegar uma nova entrada, se o computador está tratando outra, ela será colocada numa fila, acarretando um aumento do tempo de resposta. O intervalo entre a ocorrência de eventos no ambiente controlado é outro fator a ser considerado quando se projeta um sistema. Em geral, na área de aplicações de sistemas de tempo real, a distribuição destas ocorrências varia com o tempo, randomicamente, embora com uma certa sazonalidade, as ocorrências de eventos tem uma escala de tempo maior. A capacidade do sistema deve ser projetada para atender um máximo (pico) de entradas por unidade de tempo, sem comprometer a confiabilidade do mesmo, e deverá manter a informação, caso haja uma falha momentânea ("FAULT") por excesso de entradas, para que posteriormente sejam processadas.

II.3. BANCOS DE DADOS DE TEMPO REAL

Um banco de dados de tempo real está sujeito a situações diferentes das que ocorrem com um banco de dados comum, por sofrer alterações a todo instante, em registros

diferentes ou não. O gerenciador do banco de dados deve manter a integridade, garantindo que uma determinada transação, após começada, termine antes da execução da transação seguinte, e que as alterações executadas repercutam instantaneamente, isto é, valem para a transação seguinte. Para garantir o sincronismo das informações, cada transação, antes de ser executada, deve ser validada, para só então ser colocada numa fila de transações a serem efetuadas, a qual pode ser "inteligente" o suficiente para permitir que mais de uma transação de leitura seja executada ao mesmo tempo, porém apenas uma de escrita num determinado instante de tempo. Tudo isto sem comprometer o desempenho da aplicação.

II.4. SISTEMAS DE CONTROLE E SUPERVISÃO

Nos últimos anos, com a redução do custo dos computadores , muitas empresas puderam adquirir o equipamento necessário a um sistema de tempo real, e não apenas os militares como era de costume. Com isto, houve o desenvolvimento de sistemas para várias aplicações, tais como: controle de processos, automação industrial, pesquisa científica, sistemas aeroespaciais e muitas outras áreas. Um sistema é considerado como sendo de controle, quando, através de ações do usuário ou decisões lógicas do sistema, atuam sobre o ambiente controlado. É de supervisão quando o sistema tem a função de monitorar o ambiente controlado e as ações de controle sobre o ambiente controlado serão efetuadas manualmente e não através do sistema.

Um sistema de supervisão de uma usina de geração de energia elétrica, por exemplo, pode monitorar cerca de dois mil pontos, entre analógicos e digitais, num intervalo de tempo de dois segundos. Um sistema como esse distribui os pontos que são continuamente monitorados, criando o conceito de cadência. O conceito de cadência determina os intervalos de tempos que os conjuntos de aquisição dos pontos irão atuar, por exemplo, o sistema de supervisão de uma usina faz a leitura de um conjunto de pontos (pressões) a cada dois segundos, mas

existem outros conjuntos de pontos que a cadência de aquisição são de quatro ou oito segundos. Com isso são as cadências fornecem informacões de quando os pontos serão adquiridos e também de como serão tratados, validados e convertidos.

II.5. CONCEITO DE MAQUINA VIRTUAL

Para implementar um sistema de tempo real, é necessário ter um conjunto de "facilidades" pré-definidas que permitam a utilização dos recursos do computador da melhor maneira possível, facilitando a programação em tempo real. Este conjunto de "facilidades" mascara o funcionamento de um computador, que não seja multiprocessável, que executa uma instrução após a outra, através do conceito de máquina virtual que permite multiprogramação, concorrência, tratamento de interrupções, gerenciamento de recursos e relógio extremamente preciso, para o gerenciamento das interrrupções de tempo. Estas facilidades são implementadas através de rotinas, as quais, em conjunto, formam o que se chama de Rotinas do Executivo.

II.6. COMUNICAÇÃO E RECURSOS (Processos)

Para uma gerência otimizada de processos, a máquina virtual oferece rotinas do executivo para a sincronização entre processos, que podem ser implementadas a partir das primitivas "WAIT" e "SIGNAL", as quais se utilizam de Semáforos. Os semáforos fornecem um método que permite a sincronização e o controle de filas de vários tipos, como por exemplo, filas de tarefas esperando para serem executadas, filas de acesso ao banco de dados, filas de acesso a periféricos, filas de sincronização e sinalização entre processos; mas não transmitem mensagens. Para implementar comunicação entre processos, é utilizado uma área comum aos processos, chamadas de "buffer", "mailbox" ou "message pool", controlada por semáforos. O conceito de "mailbox" é bastante

comum nos sistemas mais modernos, por não envolver a troca de um bloco de informações, e sim de um ponteiro para a área comum, onde a informação está armazenada. Esta estratégia diminui o tempo gasto pelo supervisor do sistema em transmitir os dados entre as tarefas. Pode-se usar também canais de comunicação, ligando diretamente uma tarefa a outra, através do qual um grande número de informações pode fluir, sem prejudicar o desempenho do sistema.

No gerencimento da utilização de recursos, utilizam-se semáforos para garantir o acesso exclusivo a um determinado recurso que, por razões de consistência, devam ter somente um processo alocado a ele. Isto se consegue se, enquanto um processo estiver usando o recurso, deixar o semáforo associado ao mesmo "fechado", trancando o uso do recurso. Estes recursos, chamados de recursos compartilhados, devem receber tratamento especial para evitar um "deadlock", em que, por falha no gerenciamento, o recurso fica permanentemente com o semáforo "fechado", impedindo sua utilização e causando uma falha no sistema, por exemplo, se o recurso for um banco de dados e ao terminar sua transação, um processo deixou de "abrir" o semáforo, nenhum processo conseguirá atualizar o banco de dados, ficando todos os que tentarem esperando o semáforo "abrir" para continuar.

II.7. SINCRONIZAÇÃO DE PROCESSOS

A sincronização entre processos pode ser feita mediante o uso das primitivas "Wait", "Signal" e "Schedule" PRESSMAN [3]. A rotina "Wait" permite que um processo espere pela ocorrência de um determinado evento, geralmente um semáforo, para continuar sua execução; "Signal" permite que uma tarefa sinalize a ocorrência de um evento, como por exemplo, "abrir" um semáforo; "Schedule" permite que uma tarefa espere por um determinado tempo ou uma determinada hora. Com estas primitivas implementadas, e um esquema de prioridades de execução, o esforço de se sincronizar as tarefas de um

determinado sistema fica simplificado, restando o maior trabalho a cargo do supervisor do sistema. A função do supervisor do sistema é de executar a troca da tarefa que está executando neste instante por outra, que está pronta para executar e tem mais prioridade. Neste ponto uma interrupção de "hardware" pode ser tratada como um semáforo, obrigando a uma revisão, pelo supervisor, da fila de processos prontos para executar, após tratar da mesma.

CAPITULO III

DEFINIÇÃO DE CONCEITOS PARA AMBIENTE DE INTERFACE HOMEM-MAQUINA PARA SISTEMAS DE TEMPO REAL

O objetivo deste capítulo é definir conceitos gerais de interface homem-máquina, destacando aqueles utilizados em sistemas de tempo real. Estes conceitos permitiram a definição de um conjunto de ferramentas, definidas no Capítulo V, a serem utilizadas por projetistas que desenvolvem aplicações de tempo real. É importante ressaltar que não é objetivo deste capítulo abordar todas as técnicas e conceitos de gerência de interface homem-máquina, e sim alguns aspectos relevantes à criação de sistemas de tempo real.

A Interação do homem com o computador, a exemplo do comportamento humano, envolve três tipos básicos dos processos humanos GREEN [4]: Percepção, Cognição e Atividade Motora. A função de um Projetista de Sistemas é de projetar técnicas de interação que permitam comunicação do homem com a máquina compatíveis com os processos humanos citados.

A percepção é o processo pelo qual os estímulos físicos são recebidos pelos órgãos receptores, transmitidos ao cérebro e reconhecidos. O estímulo dominante utilizado em um computador é o visual, com isso, para se realçar esse estímulo, são usados recursos técnicos tais como: variações de cores, intensidade de brilho, cintilação.

A psicologia cognitiva está relacionada com o processo humano de aquisição, organização e recuperação de uma informação, isto é, da assimilação dos estímulos físicos recebidos e da tomada de decisão baseada nestes estímulos. O estudo da cognição trata por exemplo de estruturas hierárquicasde cardápio ("menu"), tais como: quantidade de itens presentes, definição do texto ou abreviaturas.

A atividade motora refere-se ao processo humano de execução de uma determinada ação física após ter o homem recebido e reconhecido um determinado estímulo e decidido sobre a resposta a ser dada a este estímulo.

Outros fatores humanos são tratados, quando se está projetando uma interface homem-máquina e serão abordados ao longo deste capítulo.

Atualmente, em função da importância de sistemas computacionais dentro dos vários segmentos da sociedade, encontram-se disponíveis um grande número de publicações sobre interfaces homem-máquina e com diferentes abrangências, sejam nas áreas comercial, industrial ou científica. Este capítulo está baseado principalmente nas publicações do "EPRI" FREY, SIDES e colaboradores [5], e, SWEZEY e DAVIS [6], e de nossa experiência obtida no desenvolvimento do sistema de Monitoração de Parâmetros da Usina Nuclear Angra-I MARTINEZ, SCHIRRU e THOME [7]. Outras publicações foram objeto de pesquisa que contribuiram para o trabalho: GREEN [8] Projeto e Implementação do Sistema de Gerenciamento de Interfaces Homem-Máquina da Universidade de Alberta, CANADA; SIBERT [9] Projeto de um Sistema de Gerenciamento de Interfaces Homem-Máquina para Prototipação e Investigação da Aplicabilidade; MAGUIRE [10] Técnicas de Projeto de Interfaces Homem-Máquina Gráficos e Revisão dos Fatores Humanos; THIMBLEBY [11] analisa Perspectivas de Problemas de Interfaces com Usuário; KENNETH [12] Definição e Utilização de Interfaces Utilizando Ícones; BRITTS [13] Estruturas de Diálogo. STRUBBE [14] Relatório sobre Regras, Modêlos, Estruturas e Construção de Sistema de Gerência de Interface com o Usuário. MONTE [15] Sistemas de Interface com o Usuário; THOMAS e HAMLIN [16] Técnicas de Interação, com relação a Entradas Gráficas; BRAESICKE [17] Interfaces com o Usuário; FOLEY, WALLACE e CHAN [18] Fatores Humanos Ligados a Técnicas de Interação para a Computação Gráfica; GOLDBERG [19] Ambiente de Programação Interativa (SMALLTALK-80).

III.1. CONCEITOS GERAIS PARA ESPECIFICAÇÃO DE TELAS

O termo tela é a tradução de "display" do inglês, significando uma forma de comunicação visual do homem com a máquina, através de monitores de video. Um dos fatores importantes ao se definir uma tela é a seleção do que se objetiva mostrar, tendo como regras: O posicionamento, em locais de maior destaque, dos elementos da tela que tem maior prioridade; organização dos elementos de cima para baixo e da esquerda para a direita na ordem de sua utilização; aproveitamento de todas as convenções que são do conhecimento do usuário; estabelecimento do ponto focal de cada tela afim de centrar a atenção do usuário.

Os principais conceitos para especificação de telas estão relacionados à clareza na forma de apresentação das informacões (inteligibilidade), àquantidade de informações apresentadas (densidade), à integração do conteúdo, o efeito visual causado, e à representação de formatos e fidelidade cognitiva.

III.1.1. Inteligibilidade

Todos os elementos de uma tela devem ser bem escolhidos afim de assegurar que a informação seja transferida corretamente, isto é, os elementos da tela devem permitir ao usuário uma interpretação correta das informações, em um tempo suficientemente crítico, que possibilite a tomada de ações requeridas pela tela. Rótulo, côr, forma e cintilação são técnicas utilizadas para realçar uma informação, aumentando a inteligibilidade das telas.

Em sistemas de tempo real, principalmente aqueles voltados para segurança, o caráter temporal da interpretação da informação assume um papel decisivo, pois a interpretação da informação deve ter um grau máximo de correção em um tempo mínimo de observação do usuário. A maioria

dos sistemas de tempo real são utilizados para deteção de situações anormais de uma determinada operação, ressaltando condições de alarme e mensagens prioritárias de ações a serem tomadas. Dentro desta classe de sistemas vários estudos foram realizados, sendo que um dos mais significativos encontra-se na Tabela III.1. Esta tabela foi produzida pelo "EPRI" a pedido da indústria Nuclear Americana. Objetiva a otimização da interpretação do usuário em função de inteligibilidade, baseada em técnicas para evidenciar um elemento da tela levando em consideração a motivação para esta evidenciação, bem como, o meio físico utilizado. A Tabela III.1 foi baseada em um perfil de usuário compatível com o dos operadores de centrais nucleares.

A seguir descrevemos algumas técnicas genéricas, que permitem evidenciar elementos de uma tela.

O rótulo é a maneira pela qual podemos destacar o significado da informação. Em geral, rótulos são textos alfanuméricos, dispostos, em posição horizontal, próximos aos elementos a serem evidenciados. As abreviações, que não possam ser evitadas, utilizam em geral as cinco primeiras consoantes.

A combinação de cores é importante pois se não forem bem escolhidas causarão uma sensação desagradável ao usuário, como por exemplo a utilização de cores brilhantes como fundo de tela, ou o uso da côr amarela sobre a verde. As cores se utilizadas inadequadamente podem prejudicar a legibilidade da tela, não devendo ser usadas mais do que dez cores em uma única tela e se cada côr tem um significado especial, os limites de utilização de cores deverá ser de no máximo de seis, sendo preferível apenas quatro. Um aspecto importante, que deve ser considerado, na utilização de cores, é que aproximadamente oito por cento do sexo masculino tem deficiência de percepção de cores SWEZEY e DAVIS [6]. Um outro aspecto sobre a utilização de cores, é a possível perda de um dos três canhões de côr para monitores RGB ("RED"", "GREEN", "BLUE"), uma solucão apresentada para uma possível perda de uma das cores básicas, seria mostrar três símbolos com as três

1 - BOM 2 - UTILIZÁVEL 3 - NÃO RECOMENDADO TÉCNICAS PARA EVIDENCIAR UM ELEMENTO	RAZÃO PARA EVIDENCIAR				MEIO FÍSICO DO DESENHO			
	ALARME	VALOR POUCO USO	MENSAGEM DE ALTA PRIORIDADE	ENTRADA DE DADOS ERRADOS	TERMINAL ALFANUMÉRICO	TERMINAL GRÁFICO	CÓPIA EM PAPEL IMPRESSORA ALFANUMÉRICA	TRAÇADOR
LUMINOSIDADE	2	1	1	1	1	1	—	—
TAMANHO	3	1	1	1	1	1	1	1
POLARIDADE DA IMAGEM	2	2	2	2	1	1	—	3
CARACTERÍSTICAS DIFERENTES	3	2	2	2	1	2	1	2
SUBLINHADO	3	2	2	2	2	2	2	2
COR	2	1	1	1	3	1	3	1
CINTILAÇÃO	1	3	2	3	1	1	—	—
ENQUADRAMENTO	3	3	1	2	2	1	2	1
SETAS	3	2	3	2	3	2	3	2

Tabela III.1 - Seleção de técnicas para evidenciar um elemento de acordo com a razão e o meio físico do desenho. (adaptada do EPRI)

cores básicas em uma posição de destaque da tela, afim de que o usuário possa identificar um possível defeito do equipamento.

Sistemas de tempo real utilizam cores para evidenciar o estado do processo. Em geral, utilizam o verde para indicar que o estado do processo está normal, o amarelo para indicar que o estado está saindo de sua condição normal e o vermelho para indicar condição anormal. Uma quarta côré utilizada para indicar que um determinado valor está fora dos limites do instrumento, e não deve ser considerado para análise.

As informações contidas em uma tela podem ser representadas por valores, textos, gráficos e códigos. Uma outra maneira de se representar uma informação é através de formas (figuras e símbolos). A percepção humana assimila formas de uma maneira mais fácil do que um texto. De regra o ser humano distingue com facilidade cerca de 15 a 20 formas diferentes. Os sistemas de tempo real utilizam as formas para representar esquemas do processo, por exemplo, em um sistema de controle ferroviário, as formas indicam que um cruzamento está ou não bloqueado, se há desvio da linha, ou do próprio trem.

A cintilação de um elemento na tela é um efeito visual muito interessante, mas deve ser utilizado com parcimonia para casos de extrema importância. Outras alternativas se apresentam para se destacar uma informação de baixa prioridade: sublinhar, fechar em parenteses ou reverso. A utilização da cintilação em sistemas de tempo real é recomendada, desde que não conflite com a prioridade das cores; por exemplo, não deveria ser mais prioritário um amarelo cintilante do que o vermelho.

III.1.2. Densidade da Tela

A densidade de uma tela define o nível de saturação das informações apresentadas ao usuário. A

apresentacão excessiva de informações em uma mesma tela, normalmente causa dificuldade para absorção das informações pelo usuário. Se uma tela com alta densidade de informação não puder ser dividida em outras telas, o projetista deverá tentar amenizar os efeitos desagradáveis, provocados pela densidade excessiva, evidenciando os aspectos prioritários.

III.1.3. Integração do Conteúdo da Tela

A integração do conteúdo da tela é determinada pelo conjunto de informações agrupadas e o efeito visual causado. É importante agrupa-las de forma ordenada e clara para que a tela não se torne um amontoado de informações, com consequente rejeicão por parte do usuário. Não existem regras estabelecidas para se determinar uma integração do conteúdo da tela, pois sempre haverá uma razão pela qual a mesma foi definida desta ou daquela maneira. Desse modo, seria interessante se ao término do desenho de uma tela o projetista verificasse alguns itens que tentam identificar pontos críticos na integração do conteúdo da tela, permitindo que o projetista proceda modificações na tela, caso alguns desses itens sejam verificados. Os itens que tentam identificar pontos críticos na integração da tela são: existência de muitas escalas diferentes; dificuldade para determinar o relacionamento entre as informações da tela; existência de muitas regras para interpretar a tela; o desenho requer conversão ou tradução entre os elementos da tela; existência de itens que deveriam ser distintos e aparentam semelhanças.

III.1.4. Orientação de Formatos para Representação em Tela

A orientação de formato é a maneira com que o elemento será organizado e terá um destaque das informações significativas. Um formato apropriado requer duas considerações, a saber: quais informações são necessárias e como elas são utilizadas. Afim de se obter um formato

apropriado, algumas considerações podem auxiliar ao projetista na decisão da escolha, tais como: o formato deve ter um relacionamento significativo entre como a informação é mostrada e o uso operacional da mesma pelo usuário; o formato deve se adequar ao nível de precisão da grandeza a qual o usuário está acostumado; o formato deve atrair a atenção do usuário.

III.1.5. Fidelidade Cognitiva

A fidelidade cognitiva de uma tela é representada pela coesão do que se está mostrando, com o modelo (conhecimento natural) do ambiente do usuário. Quando a informação se encaixa dentro de categorias ou conceitos já entendidos pelo usuário, os processo de aprendizagem e interpretação das informações são rápidos, caso contrário a aprendizagem e interpretação das informações são vagarosas.

III.2. ESTRUTURAS DE DIALOGO

As estruturas de diálogo são os meios de comunicação entre o homem e a máquina, cuja função é intermediar a ação do usuário com o computador e definir a maneira pela qual o computador responderá ao usuário. A definição apropriada de uma estrutura de diálogo, depende da natureza da tarefa, da sofisticação do usuário, da freqüência de uso dos comandos, a média de transferência das informações e da disponibilidade do equipamento. Na Tabela III.2 é apresentada a análise das estruturas de diálogo e equipamentos contra as características do sistema e do usuário. Esta tabela também faz parte do estudo de sistemas de segurança de tempo real, anteriormente citado, que o "EPRI" realizou para a indústria americana.

As estruturas de diálogo apresentadas nos itens subsequentes, referem-se a: cardápios, perguntas e respostas, preenchimento de campos, comandos, linguagem natural, descrição

	1 - BOM 2 - UTILIZÁVEL 3 - NÃO RECOMENDÁVEL	CARACTERÍSTICAS DO SISTEMA E DO USUÁRIO													
		SOFISTICAÇÃO DO USUÁRIO				NÚMERO DE COMANDOS				MÉDIA DA FREQUÊNCIA DE USO DOS COMANDOS			ÍNDICE DE TRANSMISSÃO DE DADOS DO COMPUTADOR PARA O TERMINAL		
	MEIO DE DIÁLOGO	ALTO	MÉDIO	BAIXO	VARI-ÁVEL	MUITO ALTO	ALTO	MÉDIO	BAIXO	ALTA	MÉDIA	BAIXA	ALTO	MÉDIO	BAIXO
ESTRUTURAS DE DIÁLOGO	"MENU"	2	1	1	1	2	2	1	1	2	2	1	1	2	3
	PERGUNTAS E RESPOSTAS	3	2	1	2	3	3	2	1	2	2	1	1	2	2
	FORMAS	2	2	2	2	3	3	2	1	2	2	2	1	2	3
	COMANDO	1	2	2	2	1	1	2	2	1	2	3	1	1	1
	LINGUAGEM NATURAL	3	2	2	2	2	2	2	2	2	1	2	1	1	1
EQUIPAMENTOS	MONITOR SENSÍVEL A TOQUE	1	1	1	1	2	2	1	1	2	1	1	2	1	1
	CANETA DE LUZ	1	1	1	1	3	2	1	1	2	1	1	2	1	1
	"MOUSE", "TRACKBALL", "JOY-STICK"	3	2	1	2	2	2	2	1	2	2	2	1	2	3
	TECLAS DEDICADAS	1	1	1	1	3	3	2	1	1	2	2	1	1	1
	TECLADO ALFANUMÉRICO	1	2	2	2	2	2	2	1	1	2	2	1	1	1

Tabela III.2 - Seleçao da estrutura de diálogo e meio físico de acordo como as características do sistema e do usuário. (adaptada do EPRI)

de apoio e botão em tela. As estruturas de diálogo apresentadas acima estão diretamente relacionadas à interação com sistemas aplicativos, não tratando de outras funções computacionais de que é exemplo o editor de texto.

As estruturas de diálogo estão relacionadas com os meios físicos pelas quais serão manuseadas pelo usuário. A seleção das estruturas de diálogo apropriadas não deve ser um obstáculo para que o usuário obtenha algum tipo de informação, para isso deve-se ajustar as estruturas de diálogo e os meios físicos ao tipo de usuário.

III.2.1. Estrutura de Cardápio

O cardápio é uma lista de opções disponíveis para o usuário efetuar uma seleção. A estrutura de cardápio é uma das estruturas mais utilizadas para interação do usuário com a máquina, devendo ter o seu tempo de apresentação o mais reduzido possível e clareza bastante na forma de apresentação. As estruturas de cardápio não devem conter muitas informações, possibilitando ao usuário encontrar dentro de uma visualização global o item desejado. Uma solução para estruturas de cardápio com muitos itens de seleção, seria dividi-las em outras estruturas de cardápios por atividade. Caso a divisão em novas estruturas de cardápio não proceda, efetuar paginação, mas sempre tentando evitar uma sobrecarga da estrutura de cardápio.

III.2.2. Estrutura de Perguntas e Respostas

A estrutura de perguntas e respostas está baseada em perguntas feitas pela máquina ao usuário. Uma vez que a máquina efetua perguntas ao usuário, todas as requisições são efetuadas através de perguntas e respostas, por exemplo: "Que tela deseja visualizar?" ou "Qual a faixa de temperatura deseja monitorar?". Não é recomendada a utilização deste tipo de estrutura para sistemas de tempo real, devido ao longo tempo

necessário à leitura e interpretação das perguntas.

III.2.3. Estrutura de Preenchimento de Campos

A estrutura de preenchimento de campos é bem semelhante à estrutura de perguntas e respostas, uma vez que para cada resposta existe uma pergunta, e para se obter o resultado final faz-se necessário responder a todas as perguntas. Por exemplo, para se obter o histórico de um determinado parâmetro o usuário terá que fornecer as informações sobre o intervalo de tempo e qual o parâmetro a ser mostrado. No caso da utilização de uma estrutura de perguntas e respostas o usuário seria questionado primeiro sobre o intervalo de tempo e depois sobre o parâmetro a ser mostrado. Já na estrutura de preenchimento de campos, o usuário receberia de uma só vez as duas perguntas. Havendo a possibilidade de as respostas terem valores esperados("default"), elas já estariam preenchidas, minimizando esforços do usuário.

III.2.4. Estrutura de Comandos

A estrutura de comandos permite ao usuário requisitar uma informação ao computador, sem a necessidade de que o computador efetue perguntas ou mostre cardápios ao usuário. Geralmente os comandos são curtos, e representam abreviações mneumônicas associadas ao comando específico. A estrutura de comandos requer mais treinamento do usuário, do que as demais estruturas, devido a necessidade do usuário assimilar os possíveis comandos, afim de obter uma informação. Algumas aplicações de tempo real utilizam a estrutura de comandos, como uma estrutura auxiliar, para usuário melhor treinado.

III.2.5. Estrutura de Linguagem Natural

Muitos esforços vem sendo feito para produzir

uma interface entre o homem e a máquina, como se fosse uma pessoa conversando com outra, sendo através da teclado ou através da voz. A complexidade, ambiguidade e outros casos especiais presentes na estrutura de linguagem natural, tornam esta estrutura de difícil implementação. A utilização de estruturas de linguagem natural para sistemas de tempo real ainda se encontram na fase de estudos, o que não nos permite uma avaliação.

III.2.6. Estrutura de Descrição de Apoio

A estrutura de descrição de apoio ("HELP") tem por objetivo auxiliar ao usuário sobre a forma de utilização do sistema, descrevendo em forma de textos ou símbolos a funcionalidade de um determinado campo a ser preenchido, um item de seleção ou um botão em tela. A utilização de hipertexto complementa as descrições de apoio, permitindo que ou mais palavras ou símbolos sejam selecionáveis, ou seja, a partir de uma descrição de apoio, o usuário pode efetuar seleções ou obter uma descrição de apoio de um item que consta de uma descrição de apoio.

III.2.7. Estrutura de Botão em Tela

A estrutura de botão em tela objetiva representar um ícone ou texto, que constam da tela, a fim de que o mesmo possa ser selecionável. Um botão pode ser comparado a um item de seleção de um cardápio, com funções de selecionar telas, cardápios e etc. Uma outra função, além de selecionar, é a utilização do botão para atuar como um "botão" com funções binárias, por exemplo, o usuário pode abrir ou fechar uma válvula que esteja sendo representada na tela por um ícone.

III.3. MEIOS FÍSICOS DE DIÁLOGO

Os principais meios físicos de comunicação entre

o homem e a máquina utilizados em sistemas de tempo real são a caneta de luz ("light pen"), manete ("joy-stick"), ratinho ("mouse"), "trackball", teclas dedicadas e monitores com sensibilidade a toque ("touch screen"). Estes equipamentos sãoutilizados para seleção (descrição de apoio ("helps"), cardápios, telas). A maioria dos sistemas de tempo real de nosso conhecimento, utilizam o teclado como meio físico de diálogo, não o teclado alfanumérico usual e sim um teclado especial, onde cada tecla tem uma função no sistema (teclas dedicadas). As teclas dedicadas simulam o ambiente de trabalho do usuário e representam um meio mais rápido de se gerar comando do que os demais meios físicos citados acima. Em complemento às teclas dedicadas, outros meios físicos são utilizados em sistemas de tempo real, sendo a caneta de luz e o "trackball" os primeiros a serem utilizados e atualmente os sistemas utilizam ratinho ou tela de toque. Um outro meio físico de diálogo é a voz (sintetizadores de vozes). Já existem projetos em estudo para a utilização deste meio de comunicação para sistemas em tempo real.

CAPITULO IV

CONCEITOS BASICOS PARA DESENVOLVIMENTO DO GIGSTR

Os sistemas aplicativos para ambiente de processamento de tempo real, têm como principal finalidade monitorar e controlar o funcionamento de complexos tecnológicos, denominados de processo, como por exemplo: usina, metro, submarino ou um simples ar-condicionado.

A função dos sistemas de monitoração é a de colher periódicamente as informações do processo, chamadas de variáveis, pontos ou sinais do processo, apresentando-as ao usuário seja através de formas (diagramas de barras, gráficos, fluxogramas e etc.), cores ou textos, com o objetivo de auxiliar à operação do processo, através da integração das informações mais importantes ou gerando novos tipos de informações, como por exemplo, fornecer diagnóstico da situação de operação do processo. As informações adquiridas estão diretamente relacionadas com a atividade do processo, por exemplo, para uma usina nuclear os principais tipos de informações sao: pressão, temperatura, vazão, nível e etc., já para um sistema de monitoração de um avião são: combustível gasto, pressão do óleo, altitude, rota e etc.

Podemos qualificar os sistemas de monitoração de 3 maneiras. A primeira quando as informações são apresentadas ao usuário e não necessitam ser armazedas para futuras análises, como é o caso dos automóveis de ultima geração, que utilizam sistemas de tempo real, para calcular média de consumo, velocidade média, autonomia e etc. A segunda quando as informações geradas são adquiridas a uma taxa muito alta, sendo impossível que o usuário acompanhe as possíveis variações no decorrer do processo, por exemplo, um detetor em um acelerador de partículas colhe informações em intervalos da ordem de nanosegundos e grava estas informações, em dispositivos físicos

do tipo fita ou disco magnético, para posteriores avaliações. A terceira quando, tanto as informações adquiridas no instante de suas variações, quanto as já passadas são importantes para o acompanhamento da operação ou futuras análises, como é o caso de uma usina nuclear, que necessita das informações atuais combinadas com as anteriores para análises de tendências durante a operação ou somente das informações antigas, por exemplo, quando da ocorrência de um acidente nuclear, o operador não se preocupa com a causa e sim em re-estabelecer a operação normal, e após o término do acidente é que com as informações de antes, durante e após o período crítico se analisam as causas.

A função dos sistemas de controle é a de executar ações sobre o processo controlado, ou seja, são sistemas que reagem a eventos externos sobre o processo através de alguma ação. Por exemplo, um sistema de piloto automático de um avião "sabe" que naquela etapa do vôo o avião deve permanecer a uma determinada altura e "faz" constantemente as correções necessárias nos comandos do avião para que este permaneça naquela altitude, utilizando as informações adquiridas do processo durante o vôo e informações previamente definidas sobre o controle de operação. Podemos qualificar os sistemas de controle em automatizados e manuais. Em geral, quando dizemos que um sistema é de controle de processos, nos referimos a sistemas que controlam todo o funcionamento do processo tecnológico, sem que o usuário tenha que interferir nas ações do sistema, como é o caso do piloto automático de um avião, que a partir da programação inicial da rota o sistema controla todas as operações de vôo. Uma outra forma de controle é através de atuações do usuário sobre o processo controlado, por exemplo, um sistema de controle de uma usina hidroelétrica, onde o operador tem de abrir/fechar as comportas ou ligar/desligar um gerador elétrico, com isso, o sistema efetua uma ação que foi ordenada pelo operador.

A seguir apresentamos uma das possíveis arquiteturas (simplificada) de sistemas aplicativos de tempo real, figura IV.1, com o objetivo de identificarmos onde o

GIGSTR se encaixa no contexto das aplicações de tempo real.

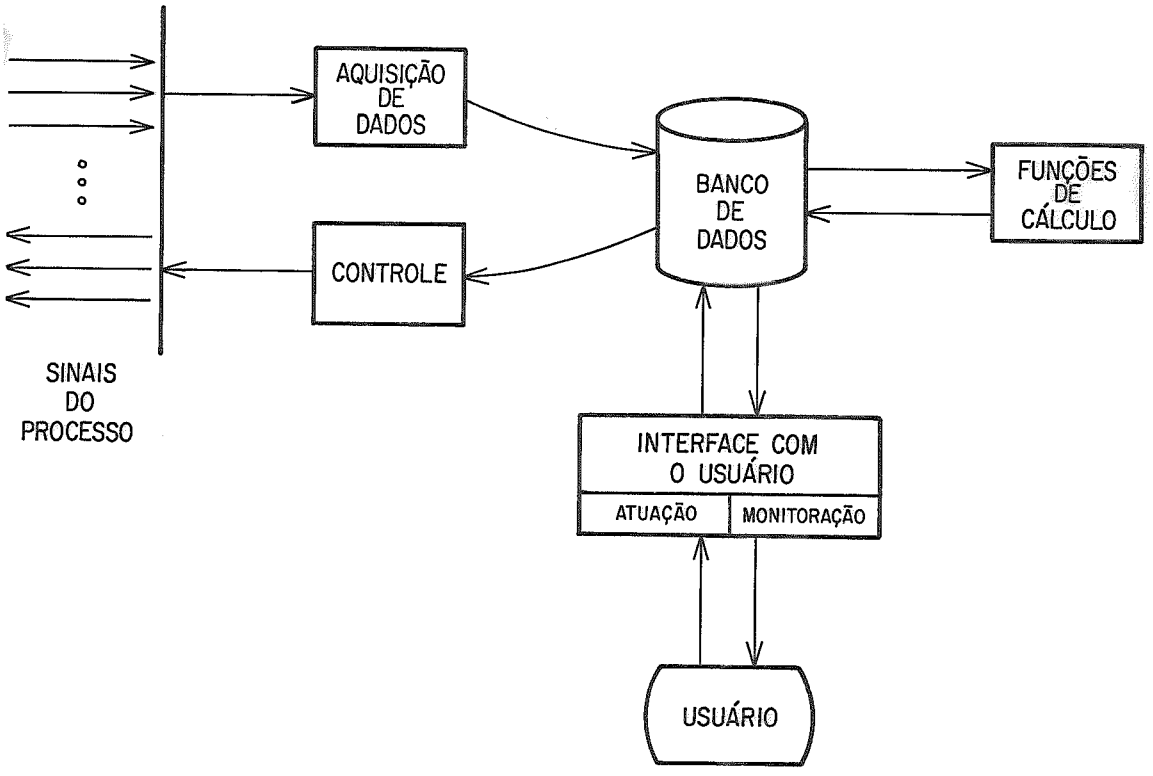

FIGURA IV.1 - Arquitetura de um sistema de tempo real.

Através da figura IV.1 podemos observar que todas as funções do sistema estão relacionadas ao banco de dados da aplicação, com o objetivo de centralizar as informações. A razão de se centralizar as informações é dada pela necessidade de independência de processamento das funções, dentro de um sicronismo temporal, com isso, podemos ter "ao mesmo tempo" aquisição de dados, funções de cálculos, controle e interação com o usuário, integrados a mesma base de dados.

A função de aquisição de dados é responsável pela aquisição de sinais elétricos (analógicos ou digitais), dentro de períodos (ciclos de aquisição) pre-determinados para cada conjunto de sinais, por exemplo, temperaturas, velocidades, níveis, pressões e etc. Esses sinais correspondem aos valores colhidos pelos sensores, que serão verificados,

tratados e convertidos em unidade de engenharia, para serem armazenados no banco de dados.

A função de controle é a de enviar sinais elétricos que atuarão sobre o processo, conforme descrito anteriormente, a partir das informações do banco de dados, sendo elas adquiridas ou previamente definidas.

As funções de cálculos englobam todas as particularidades da aplicação, que a partir das informações adquiridas do processo, novas informações são geradas. Essas informações são chamadas de variáveis calculadas.

A função de interface com o usuário é reponsavel em apresentar as informações (monitoração) ao usuário, em como o usuário efetua uma alteração no banco de dados (atuaçao sobre o processo) e em definir como o usuário interage com as funções do sistema.

Como descrito anteriormente, o banco de dados da aplicação reúne todas as informações das variáveis do processo, e um banco de dados é formado de variáveis adquiridas do processo ou variáveis calculadas. A uma variável estão associados atributos, que podem ser identificados como estáticos ou dinâmicos. Os atributos estáticos, que não sofrem alterações durante o processamento em tempo real, definem as características da variável, por exemplo, código de identificação, descrição, tipo (analógica, digital, calculada), escala superior e inferior do instrumento associado à variável, unidade de engenharia com que o sinal foi convertido e etc. Os atributos dinâmicos, ao contrário dos estáticos, são aqueles que durante o processamento em tempo real , podem, sofrer alterações, por exemplo, valor convertido em unidade de engenharia, qualidade do sinal adquirido, limites superior e inferior de operação normal ("set-point"), estado indicando a faixa de operação (normal, alerta, perigo) e etc.

Os sistemas de tempo real pesquisados, são em geral pacotes fechados que envolvem "hardware" e "software"

específicos, com o objetivo de atender a diversos tipos de operações, desde a aquisição de dados até a apresentação de informações ao usuário. É nosso entendimento que para determinadas operações principalmente no tocante a interação homem-máquina, os sistemas pesquisados, tais como: MAX-SD da ELEBRA, AUTOMIC II da PHT, A500 da W&d, SESC 2023 da EUROCONTROL e EPY100 da ECIL P&D, apresentam restrições. Estas restrições são em primeiro lugar em relação ao "hardware", onde o "software" atende as necessidades da aplicação, mas tem de ser processado em um equipamento específico, que, não raro, é super ou subdimensionado para as operações requisitadas. Em segundo lugar em relação às funções aplicativas do "software", que abrangem as funções gráficas, mas que, nem todas podem ser apresentadas ao usuário, na forma que o projetista gostaria, principalmente em função das limitações de tempo de processamento impostos pelos pacotes gráficos. Finalmente existem as restrições em relação ao diálogo homem-máquina, pois observamos que normalmente a interação é parte integrante das soluções de "hardware" e "software" e não estão sujeitas a intervenção do projetista, para que este possa decidir como será efetuado de maneira mais eficiente o diálogo homem-máquina.

Os conceitos utilizados para a definição do gerenciador de interfaces gráficas GIGSTR, aplicados a um ambiente de processamento de tempo real, estão baseados em estudos das funções necessárias a sistemas de tempo real e estudos de interação homem-máquina. Com o objetivo de dar suporte ao desenvolvimento de um "software" executável em tempo real, onde, baseado na figura IV.1, a aplicação é responsável pela aquisição de dados, controle sobre o processo, funções de cálculos e atualizações das variáveis adquiridas e calculadas em um banco de dados da aplicação. E o GIGSTR é responsável pela interface com o usuário, através da atualização sincronizada das informações do banco de dados da aplicação nos monitores de vídeo (monitoração ou representação de dados), atuação manual sobre o processo, controle do ambiente de interação homem-máquina (estruturas de diálogo) e deteção de "over run".

Os sistemas que tratam da interação homem-máquina são conhecidos como sistemas de gerência de interface com o usuário (SGIU). Com a função de mediar a interação entre o usuário e a aplicação, satisfazendo os requisitos do usuário e obtendo do mesmo os dados de entrada necessários ao processamento dos algoritmos da aplicação. O SGIU aceita como entrada uma especificação de diálogo que descreve detalhadamente a estrutura da interação. Esta especificação é normalmente independente do programa de aplicação, apesar de ambos estarem ligados logicamente. Ela livra o programador da aplicação das complicações relativas à interação, permitindo que ele se concentre nos problemas específicos da aplicação. A Figura IV.2 descreve a arquitetura básica de uma aplicação que deve ser escrita pelo programador, sabendo-se que a obtenção dos dados de entrada é da responsabilidade exclusiva do SGIU. Por outro lado, o projetista dos diálogos está livre dos detalhes da aplicação, voltando sua atenção unicamente para a melhoria dos fatores humanos envolvidos na interface com o usuário.

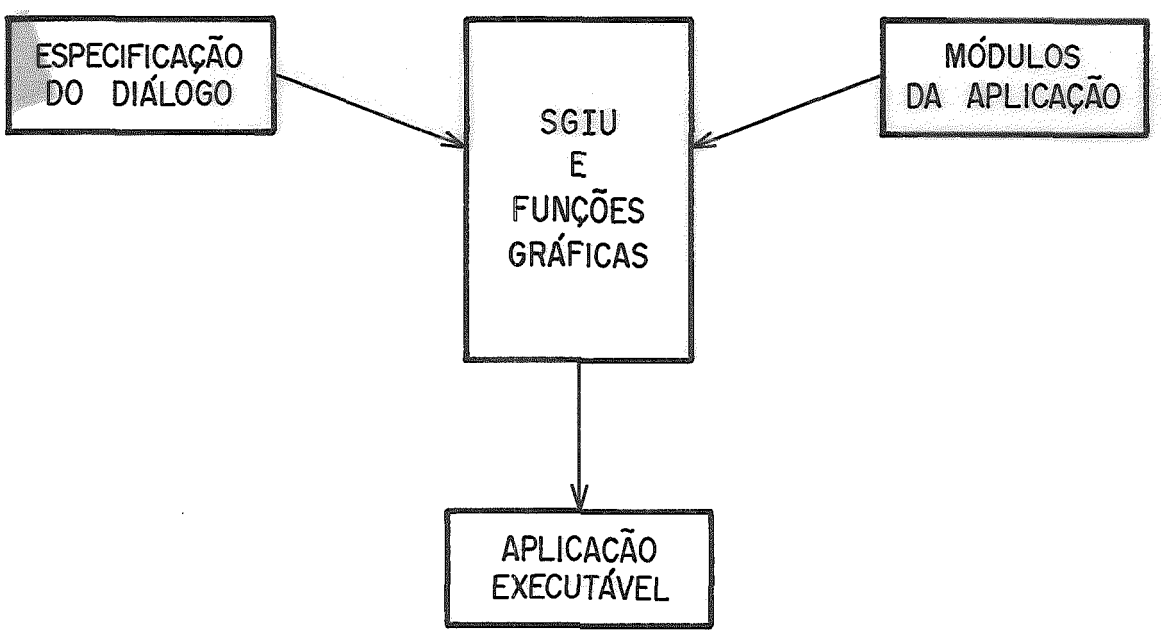

FIGURA IV.2 - Arquitetura Básica de uma Aplicação Interativa usando SGIU.

Os SGIU's obtem do projetista as especificações de diálogo, e do programador os módulos da aplicação. Estas especificações de diálogo compiladas juntamente com os módulos aplicativos e com a biblioteca gráfica, geram uma aplicação executável. A arquitetura do gerenciador de interfaces gráficas para sistemas de tempo real (GIGSTR), Figura IV.3, não pode ser considerada exatamente como um SGIU por duas razões. A primeira é que o SGIU é virtualmente genérico para qualquer tipo de aplicação, enquanto o GIGSTR está sendo definido para ser genérico dentro do universo das aplicações de tempo real. A segunda é que as funções aplicativas do sistema (representação de dados (monitoração), atuação manual sobre o processo e diálogo) não são módulos programados que serão compilados ao GIGSTR, e sim, funções predefidas, pelo projetista da aplicação, que serão armazenadas e gerenciadas pelo GIGSTR.

A função de representação de dados permite que uma ou mais variáveis do banco de dados da aplicação, sejam representadas em tela e atualizadas dinâmicamente, sem a intervenção da aplicação. As ferramentas previstas para representação de dados são: representação de valores em forma de barras; representação de gráficos bidimensionais; representação de valores históricos em formatos gráficos; representação de valores em formatos gráficos, com o objetivo de visualizar a tendência da variável; representação de simbolos utilizados em fluxogramas, para indicar, em geral, situações binárias, tais como: o estado ligado ou desligado de válvulas, bombas, ventiladores, turbinas e outros equipamentos do processo que se está monitorando; representação de primitivas gráficas, tais como: traçar linha, polígono, retângulo, círculo e outras primitivas gráficas básicas, para execução de condições excepcionais, que não foram definidas no conjunto de funções de alto nível citadas acima.

A função de atuação manual sobre o processo, permite que o usuário altere o valor corrente de uma variável no banco de dados da aplicação, para que a aplicação, se for o caso, efetue uma atuação sobre o processo através da função de controle.

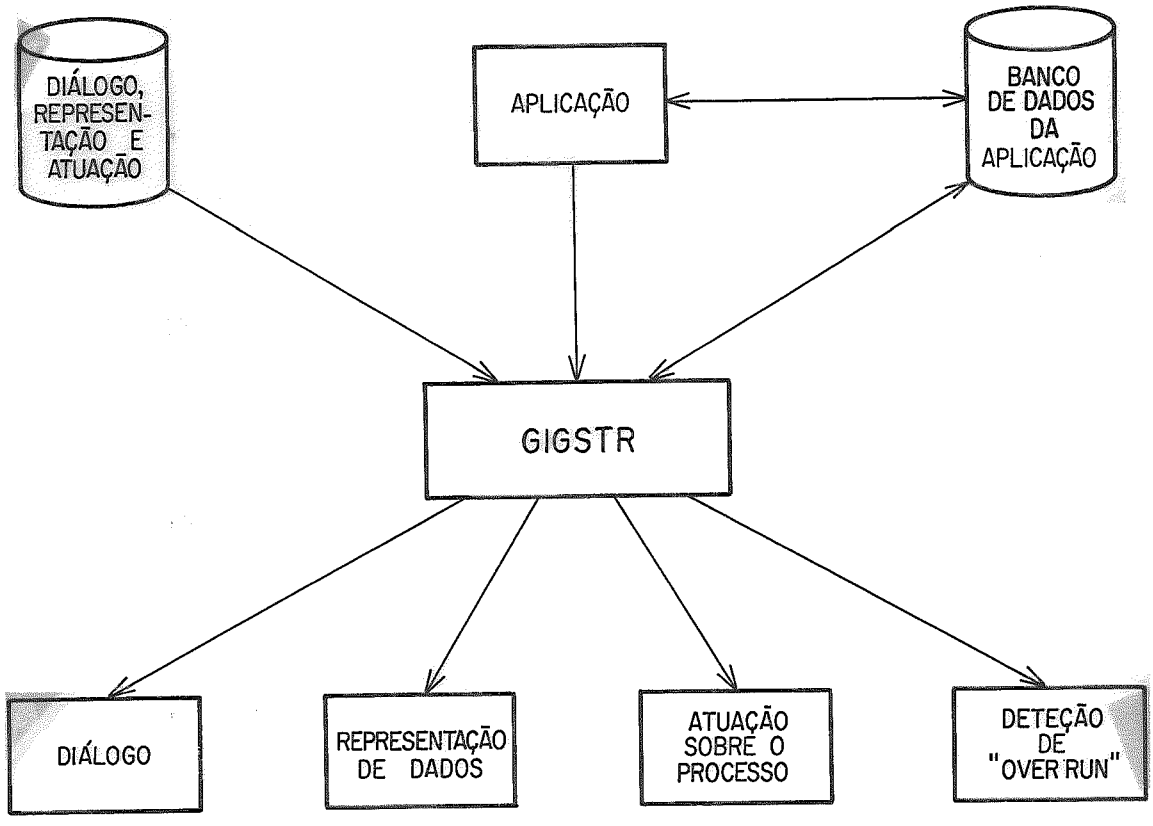

FIGURA IV.3 - Arquitetura do GIGSTR.

A função de diálogo permite que o projetista da aplicação defina a forma de interação homem-máquina do sistema, uitilzando uma ou todas as estruturas de diálogo que compõem o GIGSTR. As estruturas de diálogo disponíveis são: estrutura de cardápio, estrutura de descrição de apoio, estrutura de tecla dedicada, estrutura de comando e estrutura de botão em tela. Duas outras funções, que não são propriamente estruturas de diálogo, são descritas dentro desta função: controle de acesso (senha) e fim de processamento.

A função de deteção de "over run" é permitir que o projetista da aplicação, possa "medir" se o sistema consegue adquirir as informações do processo, efetuar os cálculos necessários e apresentar as informações ao usuário dentro do ciclo de aquisição, ou seja, se uma informação chega e o sistema não tem tempo de processá-la antes da chegada de uma

nova informação, tendo o projetista de redimensionar o tempo do ciclo de aquisição dos dados ou redimensionar as funções do sistema.

A aplicação interage com o GIGSTR através da função de representação de dados, utilizando as primitivas gráficas (retângulo, linha, preenchimento de área e etc.), para executar outras funções não previstas pelo GIGSTR. A aplicação "diz" ao GIGSTR o que e quando fazer, deixando a cargo do GIGSTR a função de execução, com isso, a aplicação não tem funções que efetuam diretamente interface com o usuário. Uma outra forma de interação entre o GIGSTR e a aplicação, é através de uma ordem da aplicação ao gerenciador para que apresente ao usuário uma determinada tela que o mesmo não tenha solicitado, permitindo que a aplicação tenha meios de alertar, se for o caso, ao operador de eventual anomalia, em função das informações dinâmicas do processo.

O banco de dados da aplicação é utilizado pelo GIGSTR para obtenção dos atributos das variáveis, que serão representadas nos monitores de vídeo (leitura) e para atualização dos valores correntes das variáveis que sofrerem atuações manuais (gravação). É importante ressaltar que o GIGSTR não é responsável pela manutenção direta do banco de dados da aplicação e que utiliza rotinas de leitura e gravação, desenvolvidas pela aplicação, que serão incorporadas ao programa executável do GIGSTR.

A base de dados do GIGSTR contém informações de diálogo, representação de dados e atuação sobre o processo, que são gerados previamente pelo projetista da aplicação. A seguir definiremos o ambiente computacional do GIGSTR e os conceitos de tela, para que possamos detalhar as funções do GIGSTR.

IV.1. Ambiente Computacional

O GIGSTR foi definido de modo a atuar como uma estação gráfica (microcomputador ou estação gráfica de

trabalho) Figura IV.4, comunicando-se com um computador, denominado hospedeiro, que tem a função de adquirir, processar e atualizar o banco de dados da aplicação.

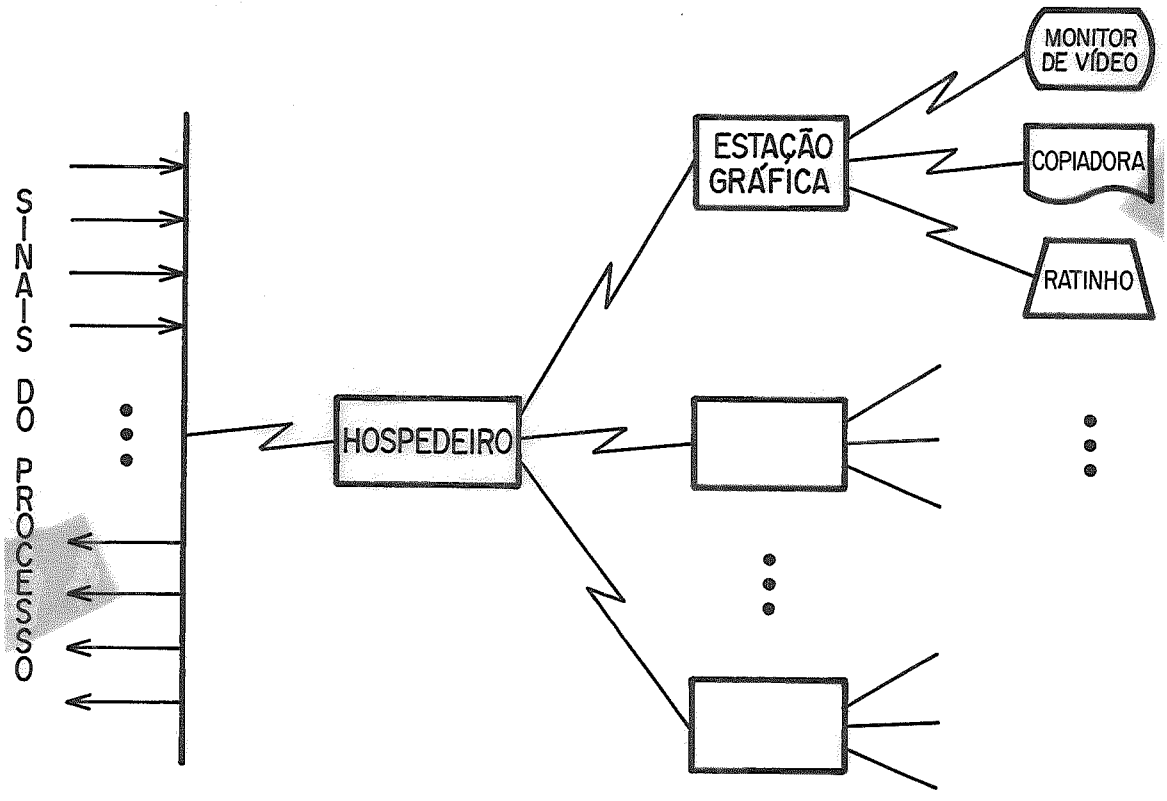

FIGURA IV.4 - Arquitetura Computacional do GIGSTR.

O processamento do GIGSTR será executado parte na estação gráfica e parte no computador hospedeiro. A estação gráfica tem as funções de interação com o usuário (diálogo), deteção de "over run", apresentação das informações, que serão adquiridas do processo pelo computador hospedeiro e enviadas à estação gráfica, quando de uma monitoração (representação de dados), e permitir que o usuário altere o valor corrente de uma variável, para que a mesma seja enviada ao comuptador hospedeiro, a fim de ser atualizada no banco de dados da aplicação.

A característica de tempo real do GIGSTR encontra-se essencialmente no compromisso da sincronização

entre os períodos da aquisição dos dados e os períodos de atualizações destes dados nos monitores de vídeo. Esta sincronização pode ser exemplificada, considerando-se um determinado processo, que para ser qualificado como de tempo real deva adquirir, converter e processar os sinais de entrada em períodos cíclicos de dois segundos. Dentro deste sistema o GIGSTR deverá possibilitar a atualização dos dados em tela também dentro de períodos cíclicos de dois segundos, sincronizados com a aquisição. É importante notar que existem dentro da estação gráfica, várias funções de interação homem-máquina concorrentes com a atualização dos dados em tela, implicando assim que a implementação do GIGSTR tenha uma arquitetura de processamento concorrente e priorizável, de modo a permitir o sincronismo periódico entre a aquisição de dados em tempo real e a sua consequente atualização nos monitores de vídeo.

IV.2. Definição dos Conceitos de tela

Todas as funções de diálogo do GIGSTR estão relacionadas à telas, e as funções de representação de dados e atuação sobre o processo, são partes integrantes da tela, com isso, o GIGSTR "sabe" o que fazer a partir da identificação da tela. Podemos dividir as telas em 3 grupos: telas de estruturas de diálogo, tela base e telas atualizáveis.

As telas de estruturas de diálogo definem o ambiente interativo do sistema, que serão definidas ainda neste capítulo, são: tela de cardápio, tela de descrição de apoio, tela de comando, tela de controle de acesso (senha) e tela de fim de processamento. A estrutura de diálogo "botão em tela" é parte integrante das telas, podendo ser definida nas telas de diálogo, base ou atualizável.

A tela base é uma parte da tela que não sofre atualizações e estará sempre presente em todas as telas do sistema, com o objetivo de mostrar ao usuário as principais funções do sistema aplicativo. A existência da tela base não é

obrigatória, ficando a cargo do projetista a sua criação ou não, mas somente uma tela base pode ser definida para o sistema aplicativo, pois as demais telas de atualização irão compor o restante da tela a ser apresentada no monitor de vídeo. As Figuras IV.5 e IV.6 apresentam dois exemplos de tela base e as áreas retangulares contíguas restantes, são destinadas para as telas atualizáveis.

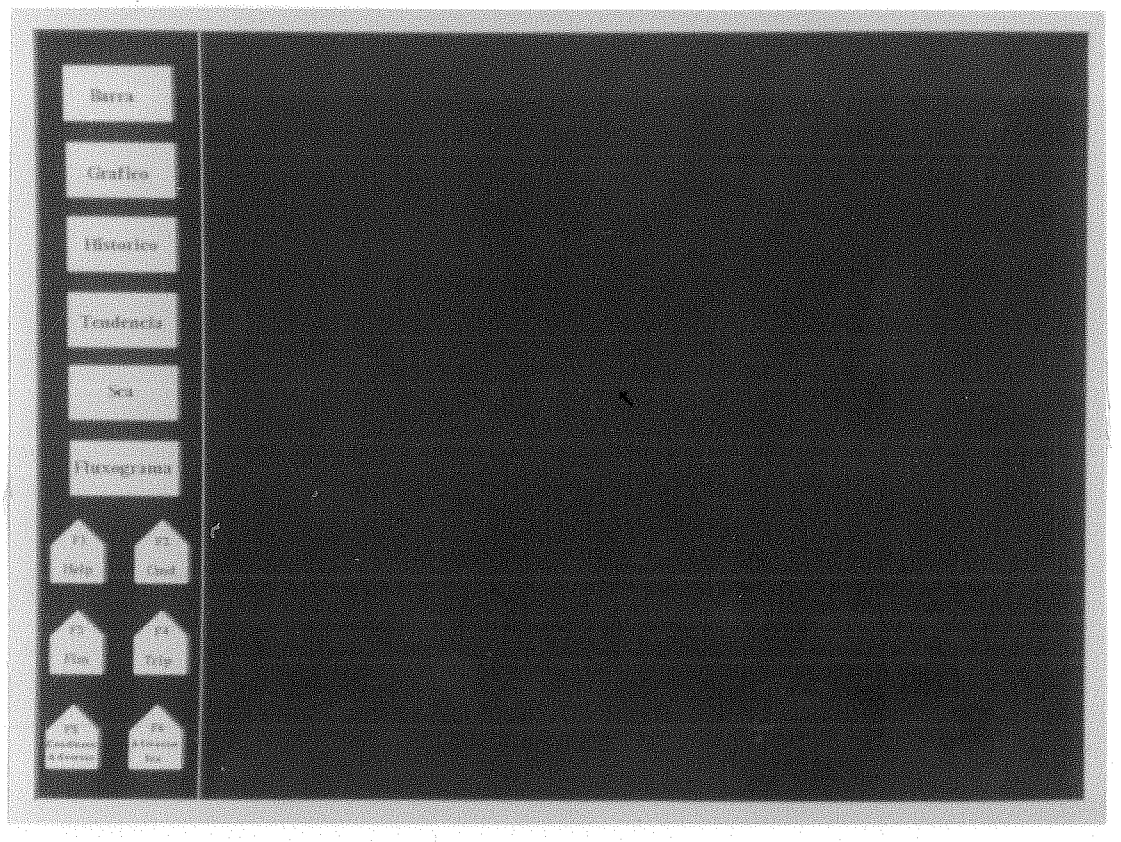

FIGURA IV.5 - Tela Base e Atualizável.

As telas atualizáveis são utilizadas para a definição das funções de representação de dados e atuação sobre o processo. Uma tela atualizável é composta de uma parte estática e outra dinâmica. Na parte estática são definidos os textos, desenhos, símbolos e etc. que não sofrem atualizações durante o processamento. E na parte dinâmica são definidas as funções de representação de dados, que sofrem atualizações dinâmicas durante o processamento, e as funções de atuação sobre o processo, que permitem alterações de valores correntes

das variáveis do banco de dados da aplicação. Uma tela atualizável ocupa uma região da tela, retangular e contigua, a qual, é demarcada pela região livre da tela base. O objetivo de se ter uma tela base e várias telas atualizáveis, deve-se ao fato de que sempre se apresenta uma única tela de cada vez, tendo o efeito de que todas as telas têm uma parte em comum (tela base), e na realidade não se define várias vezes esta parte em comum, economizando tempo de confecção, processamento e espaço de armazenamento. Por outro lado após a demarcação da região atualizável, efetuar um redimensionamento desta região, implicará em grande esforço para recuperar as demais telas prontas. A Figura IV.5 mostra o lado esquerdo como tela base e o restante como tela atualizável e a Figura IV.6 apresenta a tela base ocupando as partes superior e inferior da tela e a tela atualizável ocupando o centro.

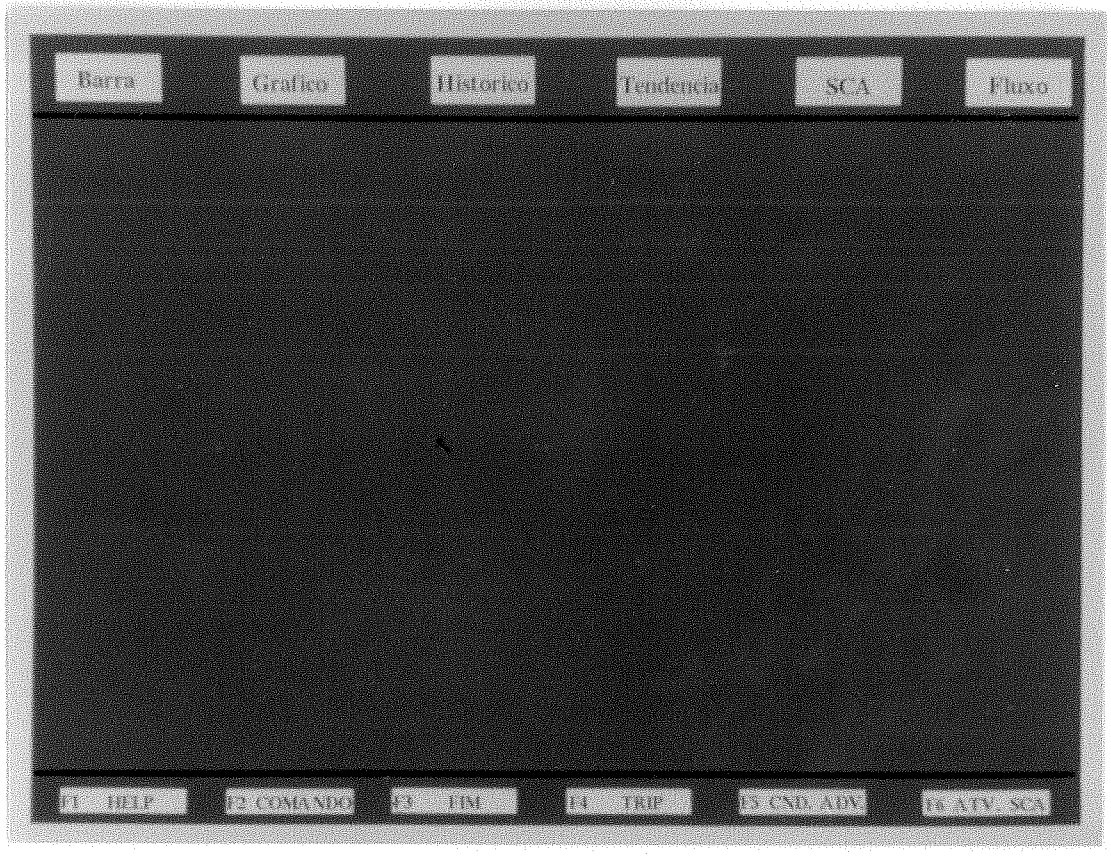

FIGURA IV.6 - Tela Base e Atualizável.

É importante ressaltar que dentro do GIGSTR uma

tela do monitor de vídeo, ou seja, a tela "física" visualizada pelo usuário, pode ser composta de várias telas "lógicas" (tela base, tela atualizável, tela de cardápio, etc.), sendo que, ao mesmo tempo podemos ter uma única tela base, uma única tela atualizável e várias telas de diálogo superpondo as telas base e atualizável. Concluindo, a tela base, se definida, é única no sistema, e é apresentada uma única vez não sendo mais retirada do monitor de vídeo; pode-se definir várias telas atualizáveis, mas somente uma de cada vez poderá ser apresentada ao mesmo tempo; pode-se definir uma ou todas as estruturas de diálogo disponíveis no GIGSTR, e as estruturas de diálogo são apresentadas superpostas às telas base e atualizável.

IV.3. Descrição das Funções de Representação de dados

Através das funções de representação de dados é criada uma parte da base de dados do GIGSTR , ou base de dados atualizáveis, que serão utilizadas pelo processamento da estação gráfica, estabelecendo um protocolo de comunicação, para solicitar dados do computador hospedeiro. As funções gráficas de representação de dados do processo previstas no GIGSTR são: barras, gráficos, textos, históricos, tendências, símbolos e primitivas gráficas.

Toda as funções de representação de dados são formadas por dados que definem as características das funções, as quais, são previamente definidos pelo projetista da aplicação e armazenados na base de dados atualizáveis. Para cada uma das funções de representação de dados, existe pelo menos uma variável do banco de dados da aplicação associada, que será representada em tela por uma das funções.

Um outro tipo de dado que compõem as funções de representação de dados, é o atributo da variável do banco de dados da aplicação, ou seja, não necessariamente a representação tem de ser do valor corrente adquirido do processo, podendo ser qualquer um dos atributos do banco de dados da aplicação, como por exemplo, descrição da variável,

unidade de engenharia, limites de escala do instrumento e etc.

IV.3.1. Descrição da Função Barra

A representação de valores por meio de barras, tem a finalidade de evidenciar uma informação, tanto pelo volume quanto pela côr. A função de barra, em geral, é utilizada para a representação de diagramas de barras e representação de volumes, tais como: nível, vazão e pressão. A seguir definiremos os atributos necessários à confecção de uma barra, utilizando a Figura IV.7, para exemplificar.

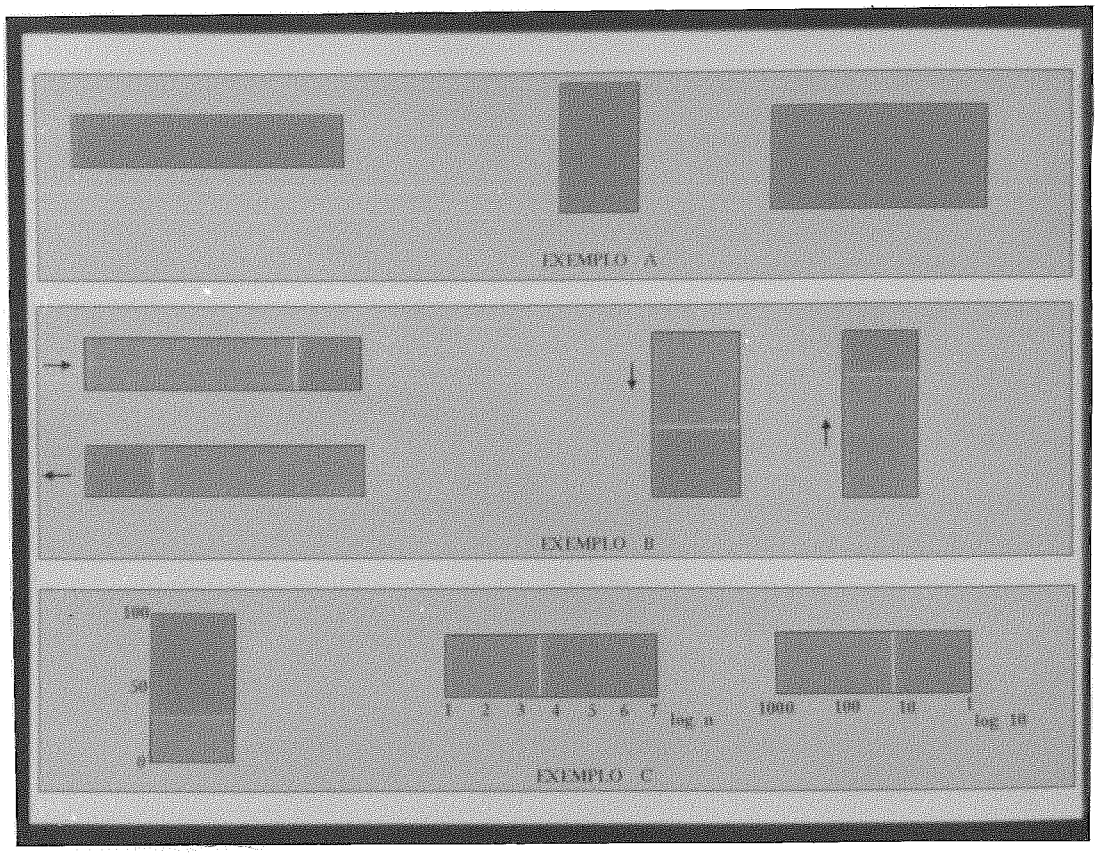

FIGURA IV.7 - Exemplos da Função Barra.

- Borda da barra (exemplo a), para identificação dos limites de atuação da barra.

- Sentido de direção (exemplo b), do qual a

barra irá iniciar o preenchimento, podendo ser: horizontal, iniciando da esquerda ou da direita e vertical iniciando de cima ou de baixo.

- Estilo de preenchimento (exemplo b) da barra e do fundo da barra.

- Côr de fundo da barra, no exemplo "b" todas as cores de fundo estão representadas pela côr azul claro. A côr da barra em si é tratada em tempo de execução.

- Código da variável a ser representada pela barra.

- Tipo de escala a ser utilizada (exemplo c), podendo ser: linear ou logaritmica.

- Valores de limite superior e limite inferior da variável e que será representada pela barra, para o cálculo do fator de escala.

IV.3.2. Descrição da Função Gráfico

O objetivo da função gráfico é a marcação com um cursor sobre uma área previamente definida como sendo o gráfico. Os gráficos são bidimensionais, devido ao fato de que processamentos de tempo real, de monitoração na sua maioria, não requerem gráficos tridimensionais. A Figura IV.8 exemplifica um gráfico onde somente a marcação do cursor é objetivo da função gráfico, os demais desenhos e valores foram definidos por outras funções (estáticas e dinâmicas). A seguir definiremos os atributos necessários à função gráficos, utilizando a Figura IV.9 para exemplificar.

- Borda do gráfico para identificação dos limites do cursor.

- Sentido de direção: horizontal , vertical

(exemplo a), ou ambos.

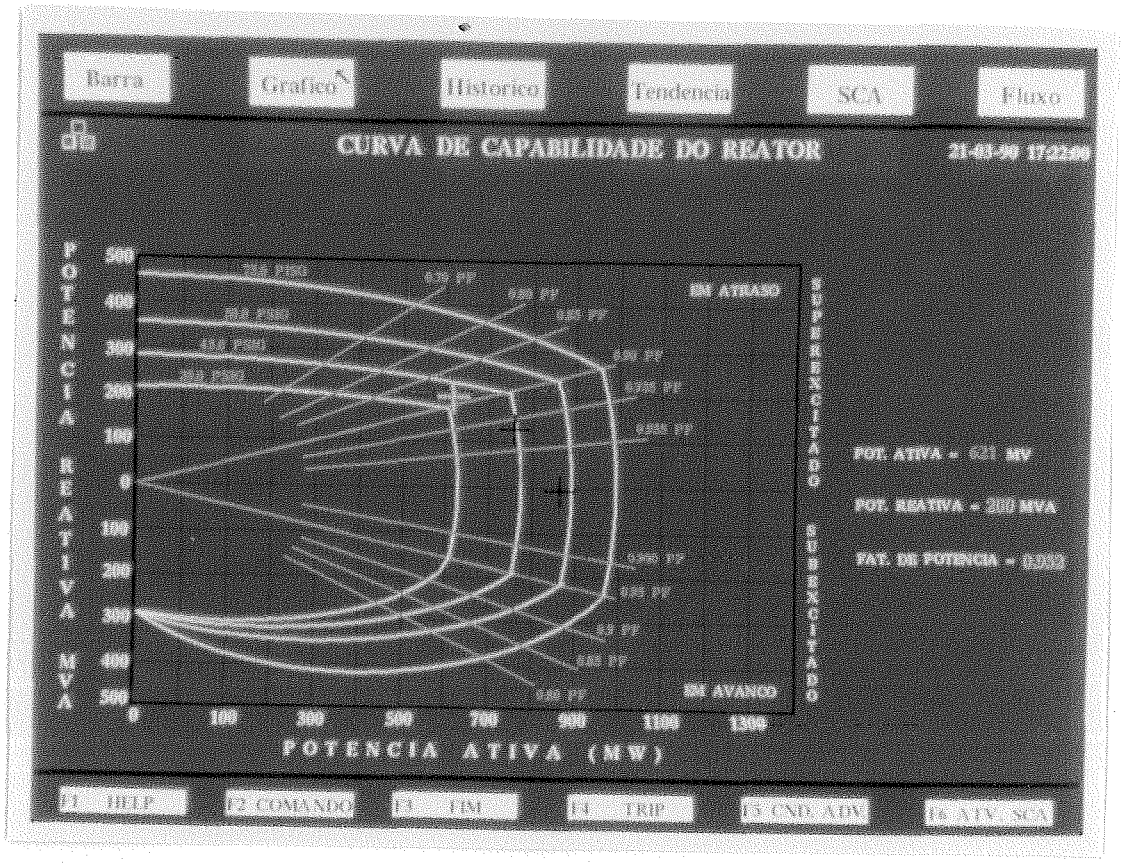

FIGURA IV.8 - Exemplo da Função Grafico.

- Tipo do cursor (exemplo a), podendo ser: o cursor padrão em formato de uma cruz ou, cursor que ocupa toda a área do gráfico ou um cursor a ser definido pelo projetista (exemplo b).

- Código das variáveis que representam os eixos X e Y dependendo do sentido de direção .

- Identificar os atributos das variáveis (X e Y). Em geral um gráfico representa o valor atual de uma variável, mas também pode ser qualquer outro atributo numérico da variável, como por exemplo: "set point" de alta ou de baixa (exemplo c).

- Tipos de escala a serem utilizadas pelos eixos

X e Y, podendo ser: linear ou logaritmica.

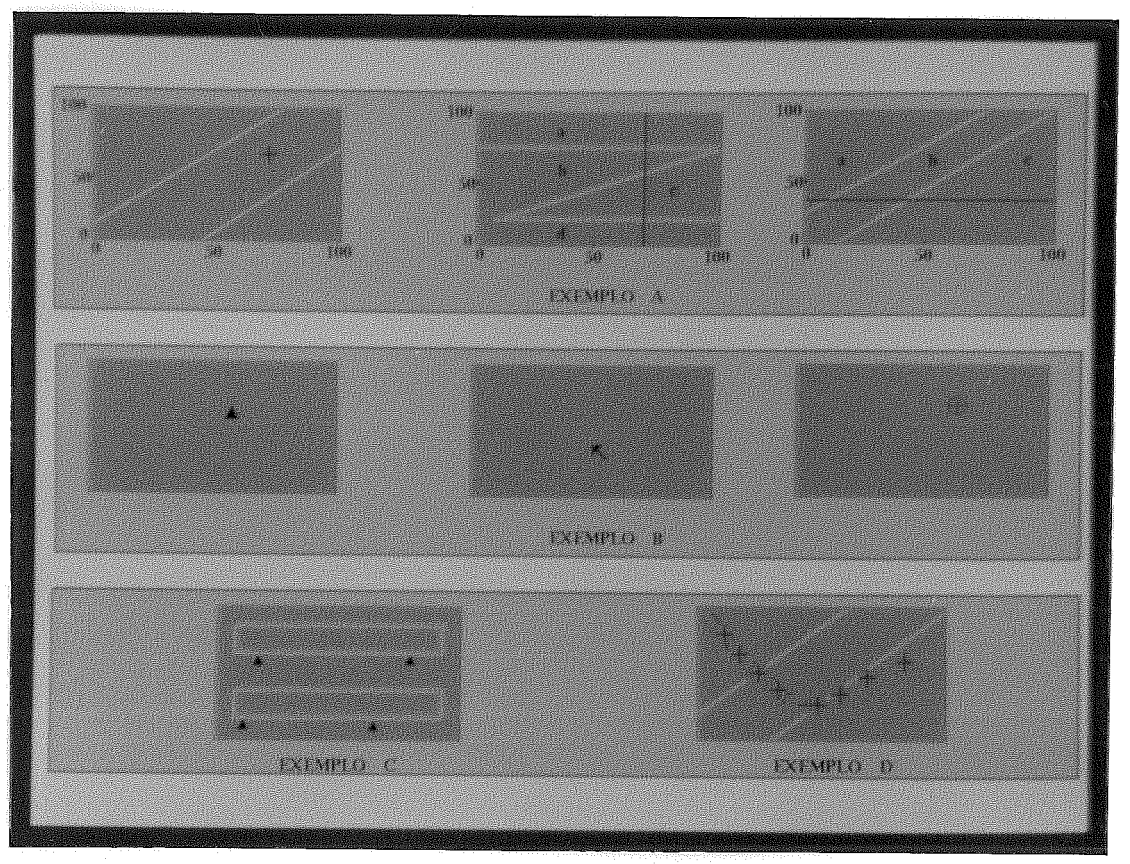

FIGURA IV.9 - Exemplos de Cursores da Função Gráfico.

◦ Valores de limites superior e inferior que representam os eixos X e Y.

◦ Identificar se o cursor deixa ou não rastro por onde ele passa (exemplo d) e qual a côr do rastro. O objetivo de se ter rastro é uma forma do usuário analisar a variação durante um determinado tempo.

IV.3.3. Descrição da Função Texto

A função de texto tem por objetivo representar atributos de uma variável, independente de seu formato (real, inteiro ou caracter). A função transforma os formatos reais e

inteiros em caracter, de acordo com os formatos fornecidos pelo projetista. A seguir definiremos os atributos necessários à função de texto. A Figura IV.8 apresenta valores que foram representados pela função de texto.

- As coordenadas (X e Y) iniciais da seqüência de caracteres.

- Código da variável e o atributo, onde atributo, pode ser: descrição, valor atual, "set-point", unidade de engenharia, e outros de acordo com a aplicação.

- O formato da variável é composto de três informações: tipo do formato (caracter, real ou inteiro), tamanho do campo e quantidade de casas decimais para os valores reais.

- Tipo de caracter (padrão, romano, itálico e etc.).

- Tamanho do caracter.

- Direção (horizontal ou vertical).

IV.3.4. Descrição da Função Histórico

O objetivo de gráficos históricos é o de mostrar a variação de uma variável do banco de dados da aplicação durante um intervalo de tempo, permitindo uma possível análise detalhada após qualquer evento ocorrido ou no decorrer do processo. A seguir definiremos os atributos que compõem a função de histórico, utilizando a Figura IV.10 para exemplificar.

- Borda do histórico para a identificação dos limites.

- Código da variável a ser monitorada.

- Hora de início e fim dos valores a serem monitorados.

FIGURA IV.10 - Exemplo da Função Histórico.

- Sentido de direção de preenchimento do histórico, podendo ser: horizontal iniciando da esquerda ou da direita e vertical iniciando de cima ou de baixo.

- O tipo de preenchimento do histórico pode ser: Acender um ponto na tela, traçar uma linha a partir da base até o valor podendo uní-los ou não, traçar uma barra a partir da base até o valor podendo enchê-la ou não.

- Côr a ser utilizada para traçar o histórico.

- Estilo de preenchimento, utilizado na opção de barra cheia.

- Tipo de escala, podendo ser: linear ou

logaritmica.

- Valores de limite superior e inferior da variável.

IV.3.5. Descrição da Função Tendência

Os atributos de gráficos de tendência são exatamente iguais aos de gráficos de histórico. A função de tendência monitora uma variável a partir da montagem da tela e a cada ciclo de aquisição do processo é acrescentado no gráfico a marcação do novo valor e os demais são deslocados. A Figura IV.11 mostra alguns exemplos de gráficos de tendência.

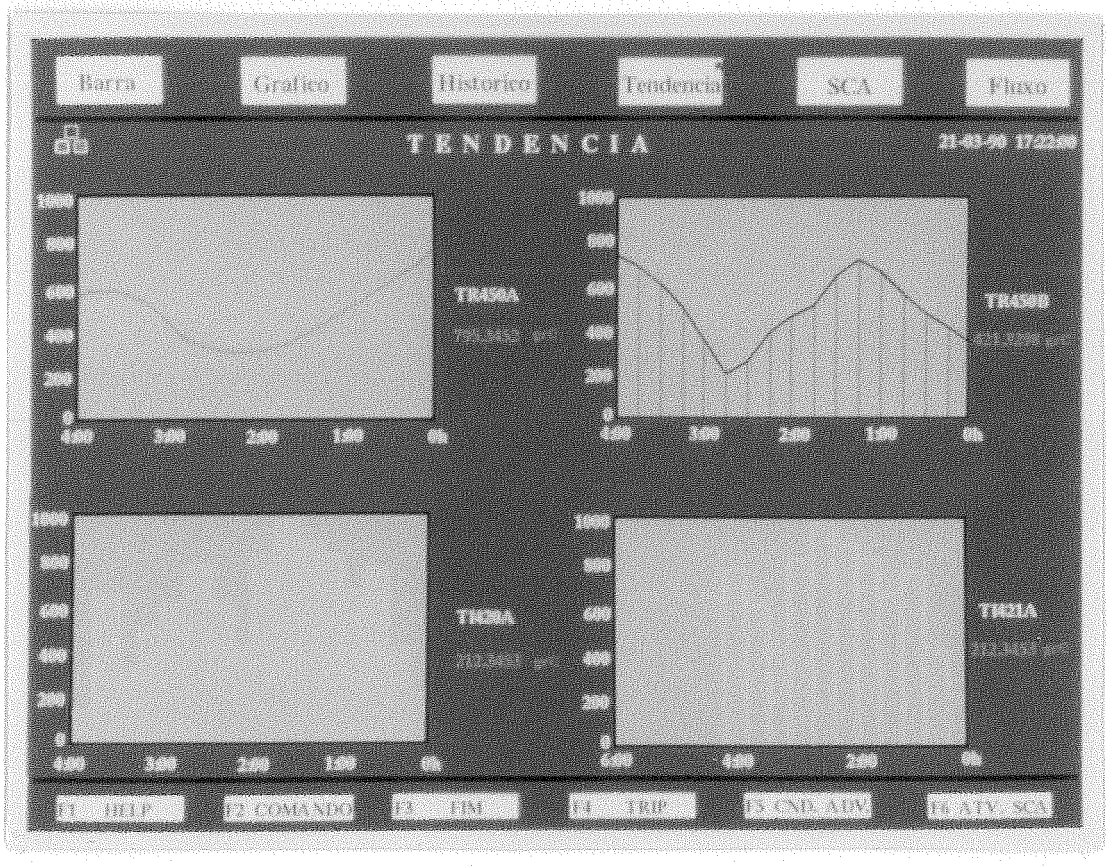

FIGURA IV.11 - Exemplos da Função Tendência.

IV.3.6. Descrição da Função Símbolo

A função de símbolos permite que o projetista

represente um equipamento ou parte de um equipamento (bomba, ventilador, turbina, válvula e etc.) através de uma figura simbólica. A Figura IV.12 mostra um fluxograma com alguns símbolos definidos. A função de símbolo deve permitir mais de uma representação para cada símbolo, ou seja, o projetista pode definir que uma bomba por exemplo, seja representada por três figuras que correspondem a desligada (0), ligada (1) e com defeito (2).

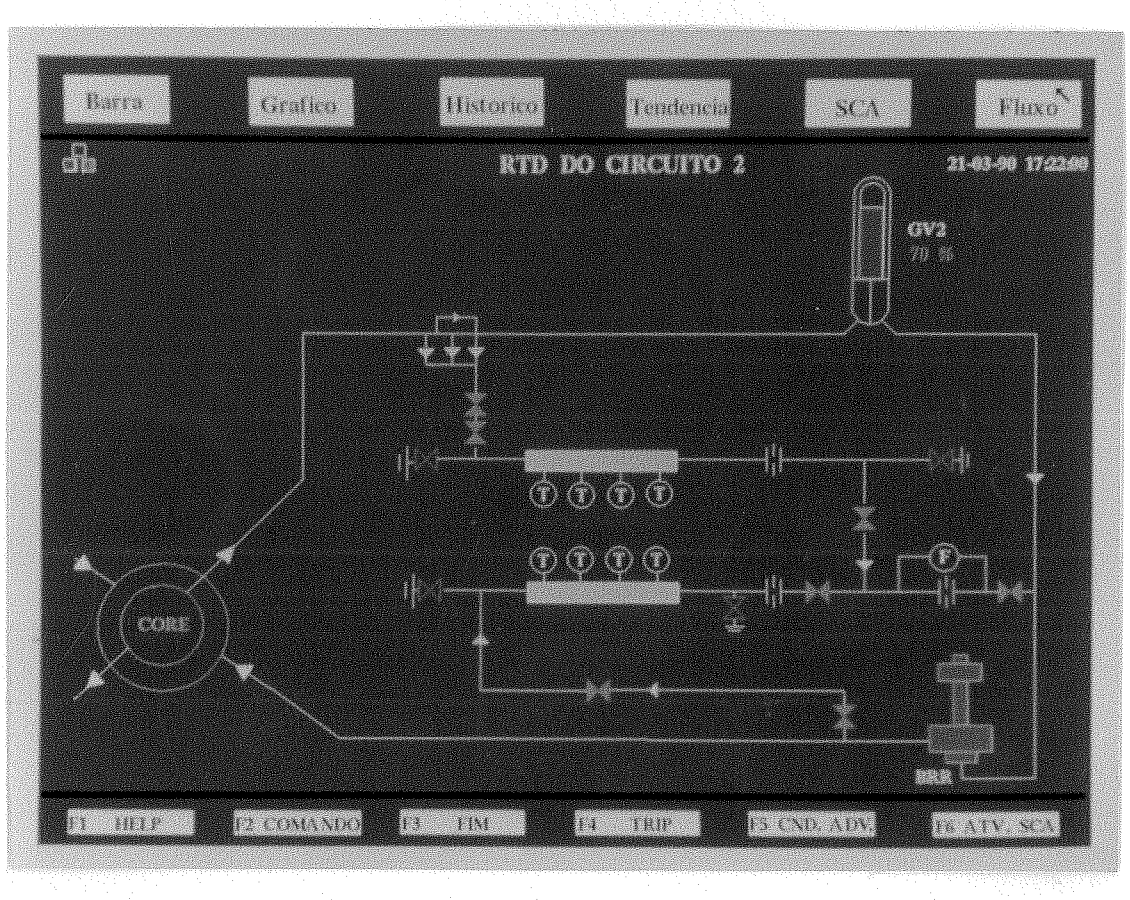

FIGURA IV.12 - Fluxograma Representado pela Função Símbolo.

IV.3.7. Descrição da Função Primitivas Gráficas

O objetivo de se ter um conjunto de funções gráficas ou primitivas gráficas, se dá pelo fato de que as demais funções de representação de dados atendem às aplicações de tempo real, mas uma ou outra necessidade da aplicação poderá não ser atendida com as funções padrões definidas (barras,

gráficos, históricos e etc.). As funções gráficas serão programadas pela aplicação e o GIGSTR será um intermediário entre o processamento efetuado pela aplicação no computador hospedeiro e a exibição dessas funções gráficas na tela. O GIGSTR recebe da aplicação (hospedeiro) a ordem de executar um conjunto de funções gráficas sobre a tela que se está monitorando. A seguir definimos as principais primitivas gráficas e seus correspondentes atributos.

- Arco de círculo (X e Y do centro, início do ângulo, fim do ângulo, raio e côr).

- Retângulo (X e Y de início, X e Y de fim, preenchimento ou não, estilo de preenchimento e côr).

- Arco de Elipse (X e Y do centro, início do ângulo, fim do ângulo, raio de X e Y, preenchimento ou não, estilo de preenchimento e côr).

- Linha (X e Y de início, X e Y de fim e côr).

- Acender um pixel (X e Y do pixel e côr).

- Polígono (X e Y de Início, quantidade de pontos, X e Y de cada ponto, preenchimento ou não, estilo de preenchimento e côr).

IV.4. Descrição da Função de Atuação Sobre o Processo

O usuário pode atuar sobre o banco de dados da aplicação, com objetivo de alterar o atributo (valor atual) de uma variável. São dois os tipos de variáveis: digitais que representam valores, em geral, binários e analógicas que representam valores reais.

São duas as formas de atuação sobre uma variável digital. Uma onde o usuário posiciona sobre um símbolo (válvula) e troca o estado atual pelo seu inverso,

obtendo em seguida a mudança do símbolo que foi reconhecido e tratado pelo processamento hospedeiro. A outra forma de atuação sobre uma variável digital, é através de teclas dedicadas, que a cada vez que são pressionadas ou selecionadas em tela, mudam o estado corrente da variável. A forma de interação será definida no item que trata sobre estruturas de diálogo.

A forma de atuação sobre as variáveis analógicas se dará através de preenchimento de campos, onde o usuário preenche o valor a ser atualizado.

IV.3. Descrição das Estruturas de Diálogo

As estruturas de diálogo para a definição da interação homem-máquina do GIGSTR são: estrutura de cardápio, estrutura de descrição de apoio, estrutura de tecla dedicada, estrutura de comando e estrutura de botão em tela. Duas outras funções, que não são propriamente estruturas de diálogo, são descritas neste item, controle de acesso (senha) e fim de processamento. As estruturas de diálogo são determidas pelo projetista da aplicação, através da base de dados do GIGSTR.

IV.3.1. Descrição da Estrutura de Cardápio

O cardápio é uma lista de opções disponíveis para o usuário efetuar seleções de telas. A seguir definiremos alguns conceitos para a elaboração de cardápios, utilizando a Figura IV.13 para exemplificar.

Um cardápio pode ser paginável, onde o usuário poderá avançar ou retornar páginas, através do teclado ou selecionando um botão na tela.

O projetista poderá definir mais de uma coluna de seleção.

Para cada item de seleção no cardápio, o projetista poderá associar uma descrição de apoio.

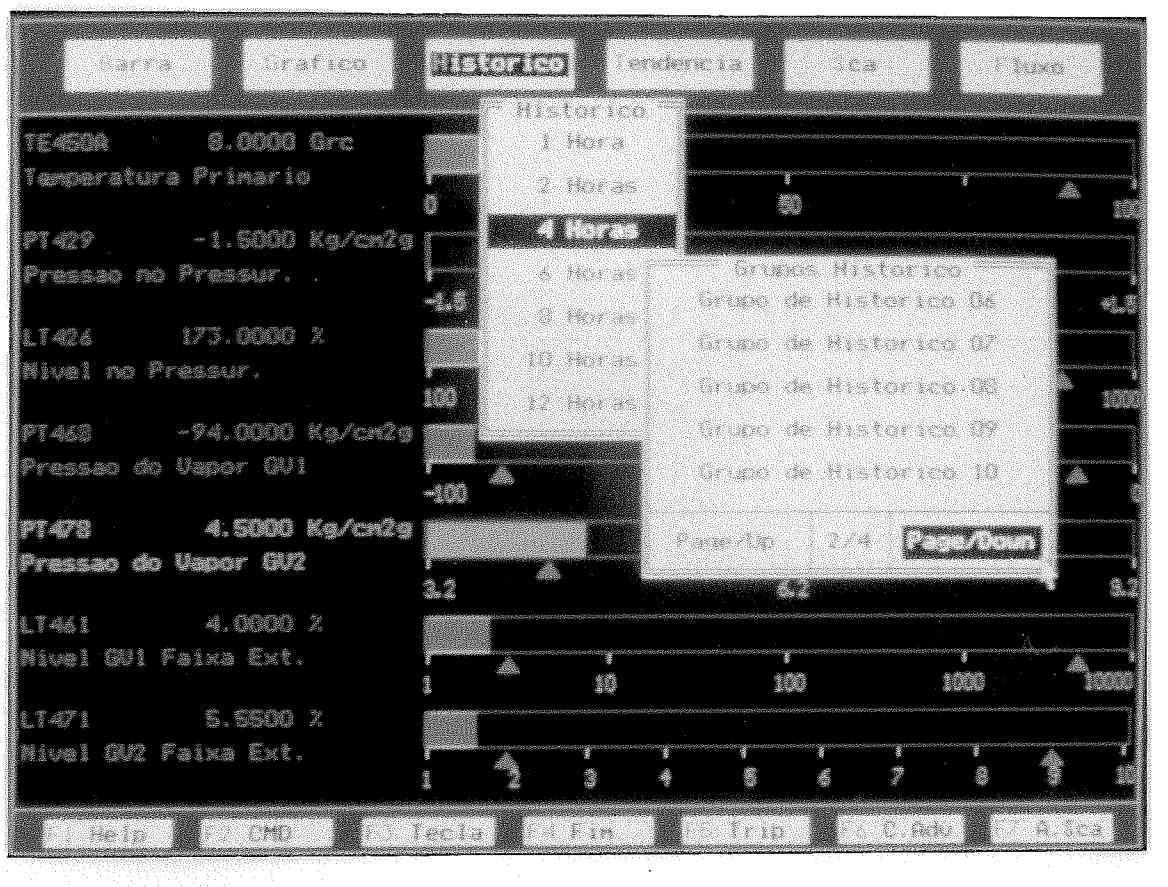

FIGURA IV.13 - Estrutura de Cardápio.

IV.5.2. Descrição da Estrutura de Descrição de Apoio

O objetivo da estrutura de descrição de apoio é criar uma independência do manual do usuário, de modo a faciliter a operação interativa do sistema aplicativo. A estrutura de descrição de apoio funciona de modo a auxiliar o usuário, durante a utilização do sistema aplicativo, como por exemplo, informar a descrição detalhada de qualquer campo da tela ou item de seleção dos cardápios ou botão na tela.

A estrutura de apoio deve prover hipertexto,

onde textos ou símbolos são selecionáveis. A Figura IV.14 apresenta um exemplo de estrutura de descrição de apoio, onde o usuário pode navegar, entre os itens selecionáveis, através do ratinho ou do teclado, podendo selecionar um item ou obter uma descrição de apoio detalhada sobre o item onde o cursor está posicionado.

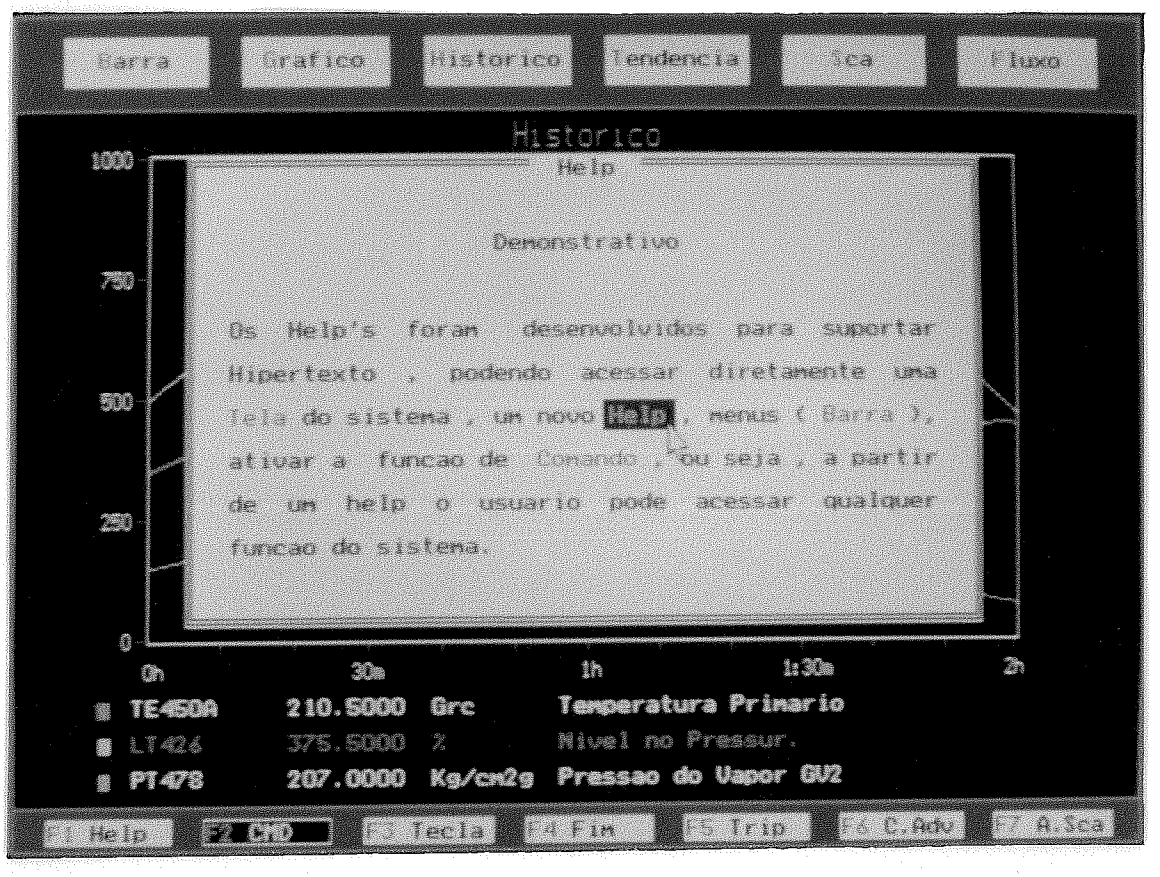

FIGURA IV.14 - Estrutura de Descrição de Apoio.

IV.5.3. Descrição da Estrutura de Tecla Dedicada

As teclas dedicadas compõem um dos meios de navegação do GIGSTR, facilitando ao usuário a obtenção de uma tela. A Figura IV.14 apresenta no rodapé da tela, uma definição de sete botões que estão associados à teclas dedicadas, os quais, serão utilizados para exibir as funções das principais teclas dedicadas do sistema aplicativo.

O primeiro botão associa a tecla F1 à estrutura

de apoio, onde o usuário poderá obter uma descrição de apoio detalhada de qualquer campo da tela ou item selecionável, bastando posicionar sobre a opção desejada e pressionar a tecla F1.

O segundo botão associa a tecla F2 à estrutura de comando, que é uma função do sistema GIGSTR.

O terceiro botão associa a tecla F3 à uma tela que não contém funções de atualização e sim um desenho do posicionamento das teclas dedicadas e as funções as quais estão associada.

O terceiro botão associa a tecla F4 ao término do processamento.

Os três últimos botões estão associados às funções de atuação sobre o processo. São variáveis digitais com a função de alterar o estado (valor corrente) da variável e enviá-las ao processamento hospedeiro, para que sejam atualizadas no banco de dados da aplicação.

IV.5.4. Descrição da Estrutura de Comando

A estrutura de comando é um meio de navegação entre as telas do sistema aplicativo, onde o usuário digita o código da tela ou do cardápio que deseja visualizar. A estrutura de comando não abrange uma linguagem natural e sim uma estrutura padronizada através de códigos.

IV.5.5. Descrição da Estrutura de Botão

A estrutura de botão define áreas da tela, que podem ser selecionadas pelo ratinho ou teclado. Os botões podem ser representados por ícones ou textos, onde o projetista elabora o conteúdo da estrutura de botão especificando: a borda da área à ser reconhecida pelo ratinho, a forma de

identificação quando o cursor está sobre o botão e a forma de navegação através do teclado. Para cada botão deve estar associado uma descrição (estrutura de descrição de apoio), uma seleção ou uma função binária (liga ou desliga), onde a seleção poderá ser para uma tela, um cardápio ou uma das funções do GIGSTR tais como: comando e fim de processamento.

IV.5.6. Descrição da Estrutura de Controle de Acesso (Senha)

O GIGSTR deve prover ao projetista uma forma de controle de acesso ao sistema aplicativo e independente por tela. Uma senha poderá ser modificada durante o processamento.

IV.5.7. Descrição da Estrutura de Fim de Processamento

A estrutura de fim de processamento é a maneira pela qual o usuário interrompe o processamento.

IV.6. Descrição da Função de "Over Run"

Uma das principais ferramentas para aplicações de processamento de tempo real é a deteção de "OVER RUN" e a mesma pode ocorrer de duas maneiras. A primeira quando o sistema não consegue adquirir e processar as informações de um ciclo de aquisição , dentro dos limites de tempo estabelecidos pelo processo (2 segundos por exemplo), o que é função da aplicação a ser executada no computador hospedeiro. A segunda maneira que possibilita a ocorrência de um "OVER RUN" é quando as informações adquiridas e processadas, pelo computador hospedeiro, são transmitidas à estação gráfica e ela não consegue atualizar dentro dos limites de tempo estabelecidos pelo processo, o GIGSTR identifica este tipo deteção de "OVER RUN". O ajuste temporal em projetos de tempo real é extremamente complexo e pode ser considerado como único para cada implementação aplicativa. Tendo esta necessidade em vista, definimos o GIGSTR com um mecanismo de deteção de "RUN

RUN" independente dos programas aplicativos, de modo a possibilitar ao projetista uma ferramenta de sincronização do projeto de tempo real e de avaliação do balanço entre as funções a serem executadas e o desempenho do equipamento.

O mecanismo de deteção de "over run" do GIGSTR independente dos programas aplicativos, provêm como consequência natural de sua arquitetura concorrente, pois todas as características de sincronização do processamento de tempo real da estação gráfica encontram-se embutidas no gerenciador e não nos aplicativos.

IV.7. Descrição da Base de Dados do GIGSTR

A arquitura da base de dados do GIGSTR, figura IV.15, está baseada nas informações das telas (base, atualizável, cardápio, descrição de apoio, comando e etc.) e associado a cada tela, pode-se ter informações de seleção e atualização.

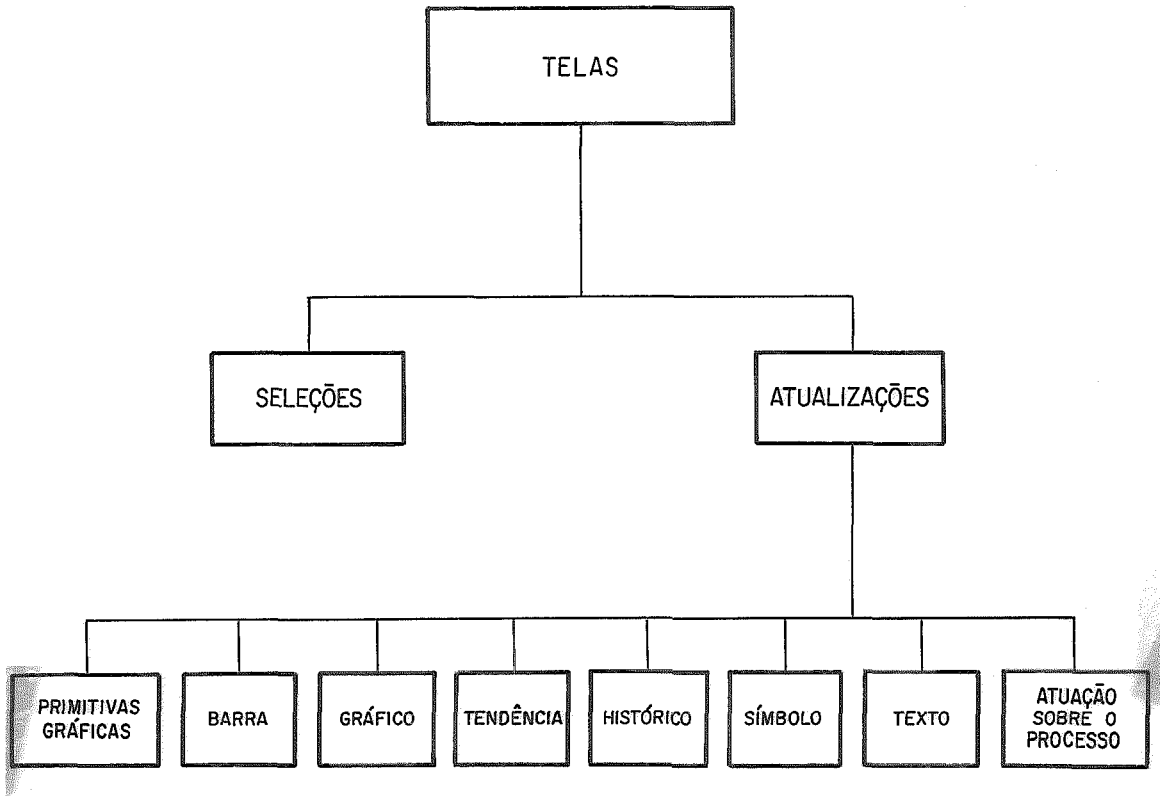

FIGURA IV.15 - Arquitetura da base de dados do GIGSTR.

As informações de seleção definem os itens de seleção dos cardápios e os botões em tela, e, as informações de atualização definem as funções de representação de dados (barra, texto, gráfico e etc.) e a função de atuação sobre o banco de dados da aplicação. Com o objetivo de não sobrecarregarmos o atual texto, definimos no anexo I cada uma das estruturas da base de dados do GIGSTR.

Para que o projetista da aplicação possa gerar a base de dados do GIGSTR, faz-se necessário a criação de um procedimento automatizado, o que não é parte integrante deste trabalho, objetivando agilizar e simplificar o trabalho do projetista. A seguir descrevemos algumas das funções do procedimento automatizado, o qual, denominamos de Editor Gráfico.

O Editor Gráfico torna disponível ao projetista do sistema aplicativo, ferramentas para criar, remover, alterar e consultar estrutura de dados, que serão utilizadas pelo processamento do GIGSTR em ambiente de tempo real.

São quatro as funções básicas do Editor Gráfico: A primeira onde são definidos os atributos do ambiente, tais como: definição de côr, estilo de linha, estilo de preenchimento e tipo de caracter. A segunda função é composta das ferramentas para desenhos estáticos, tais como: desenhar retângulos, polígonos e círculos. A terceira função do Editor Gráfico é composta das ferramentas que permitem a definição das funções de representação de dados (barra, gráfico, histórico, tendência, símbolo, primitivas gráficas e texto), e das ferramentas de atuação sobre o banco de dados da aplicação. A quarta função é composta de ferramentas que definem o diálogo, através das opções de cardápio, descrição de apoio, comando, controle de acesso (senha), botão, teclas dedicadas e fim de processamento.

Os atributos do ambiente gráfico são recursos utlizados pelos conjuntos de ferramentas de desenhos estáticos, representação de dados, atuação e diálogo que permitem

selecionar cores, estilo de preenchimento de uma área, estilo de linha, espessura de linha, estilo de polígono, tipo e tamanho de caracter e etc. A seguir descrevemos alguns dos possíveis atributos do ambiente gráfico, utilizando a figura IV.16 para exemplificar.

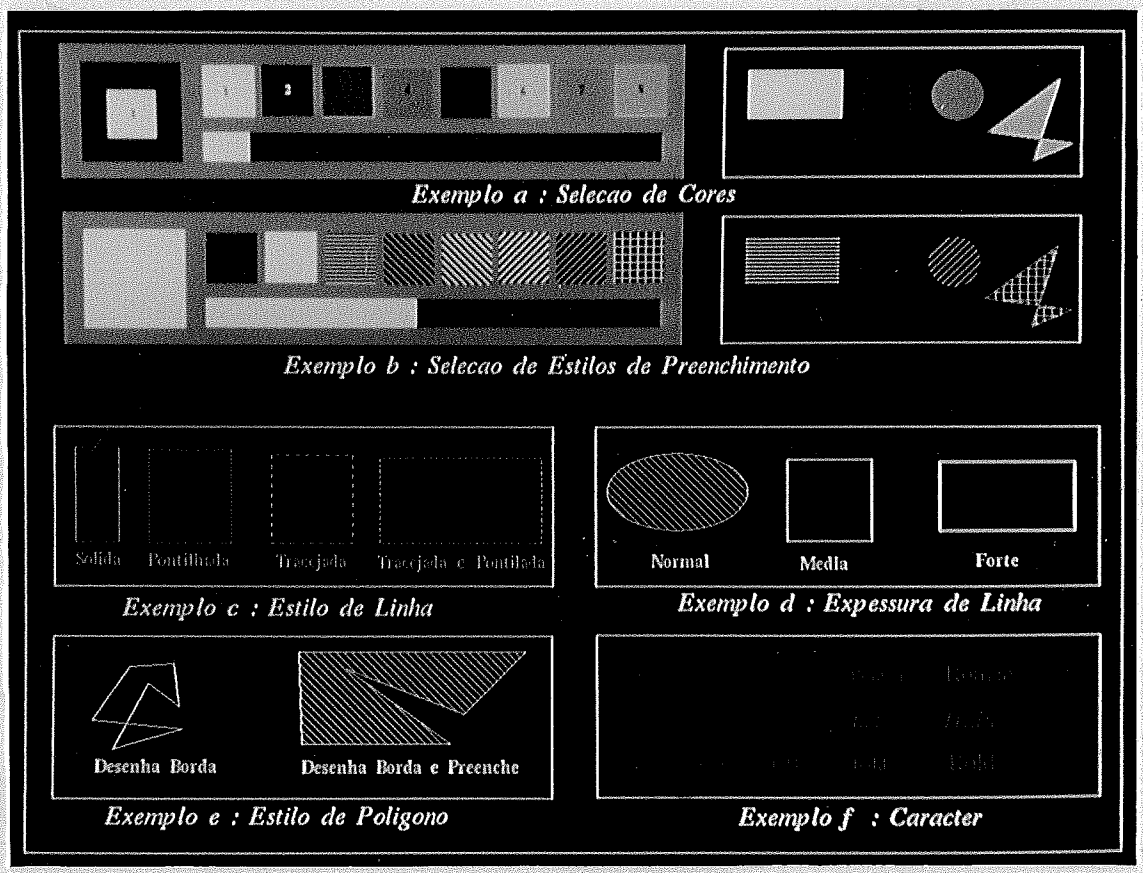

FIGURA IV.16 - Atributos do Ambiente Gráfico.

A Figura IV.16.a apresenta uma forma de seleção de cores, onde o usuário pode visualizar um conjunto de cores, bem como manipular uma barra de rolamento das cores existentes. A barra de rolamento está localizada em baixo das caixas de cores, e permite avançar ou voltar todas as cores disponíveis, bem como visualizar a quantidade que está sendo mostrada (côr branca) e a quantidade restante, representada pela côr preta. Os dois quadrados superpostos, definidos no canto esquerdo, indicam a côr atual (quadrado menor) e a côr de fundo (quadrado maior). Normalmente os editores gráficos não rotulam as cores, o que julgamos

trabalhoso para os projetistas devido à grande quantidade de cores e dificuldade em distingüi-las. Uma solução para diferenciar as cores seria enumerá-las.

- A Figura IV.16.b apresenta uma forma de seleção para estilos de preenchimento idêntica à seleção de cores. Os estilos de preenchimento são usados para preencher uma área com um determinado padrão, como por exemplo: sólido, hachurado e trançado.

- A Figura IV.16 apresenta exemplos dos demais atributos, que dispensam definições por serem auto-explicativas.

A função de desenhos estáticos é formada por um conjunto de ferramentas para efetuar desenhos inanimados, seja na tela base, na tela atualizável ou em qualquer outra tela a ser projetada, utilizando os atributos do ambiente gráfico como recursos auxiliares. A seguir descrevemos algumas das ferramentas de desenhos estáticos, apresentadas nas figuras IV.17 e IV.18.

- Desenhos de retângulos, quadrados, círculos, paralelogramos, elipses, textos, linhas e poligonos.

- Preenchimento de área.

- Edição ampliada de uma imagem.

- Criar curvas através de técnicas de Bézier ou Polinomiais de modo a atender perfeitamente as necessidades de geração de curvas das aplicações de tempo real.

- Recurso de desfazer a última ação executada.

- Limpar a tela.

- Selecionar uma área e movê-la, ou copiá-la,

para uma outra posição na tela.

- Salvar e recuperar toda a tela ou somente parte da tela.

- Posicionar sobre uma côr e obter o código ou descrição da côr.

- Mudar uma determinada côr por outra, a fim de escolher a mais adequada.

- Criar uma malha sobre uma área demarcada, com a possibilidade de variação de largura e altura da malha.

- Apagar uma área demarcada.

- Efetuar rotações e inversões com uma imagem.

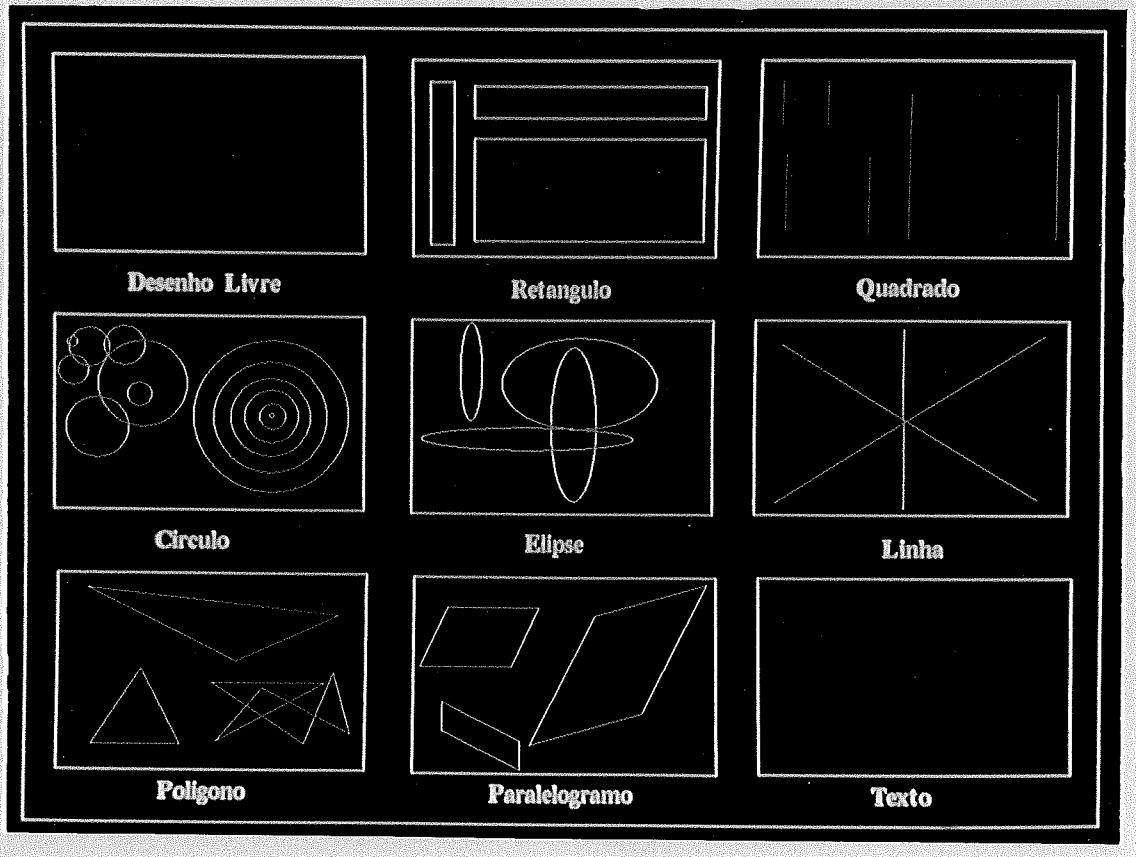

FIGURA IV.17 - Funções Gráficas Estáticas

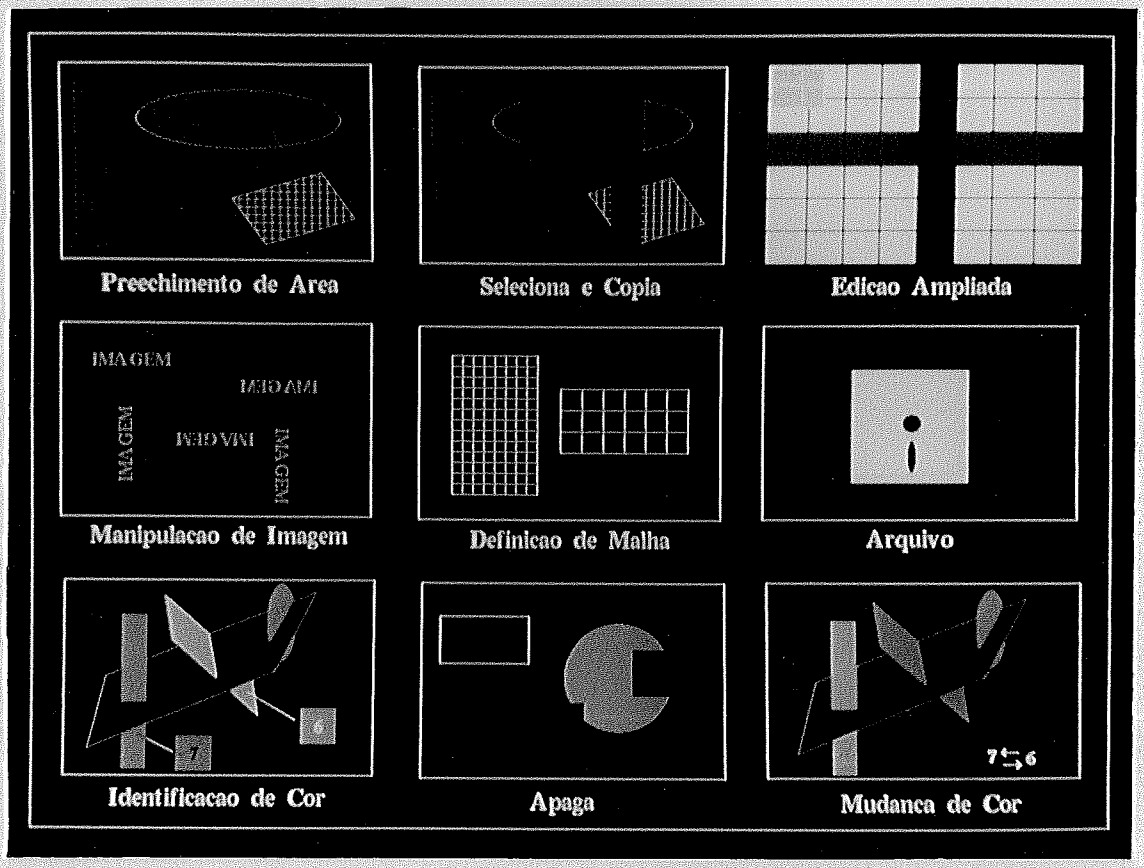

FIGURA IV.18 - Funções Gráficas Estáticas

As ferramentas de representação de dados, atuação sobre o banco de dados da aplicação e diálogo, englobam as necessidades de geração dos dados de cada uma das funções correspondentes, as quais, já foram descritas nos itens acima.

CAPITULO V

DESCRIÇÃO FUNCIONAL DO GIGSTR

O GIGSTR tem o seu processamento dividido em dois ambientes computacionais, que através de um protocolo de comunicação, recebe ou envia, informações entre o computador hospedeiro e a estação gráfica. O primeiro ambiente constitui-se no processamento efetuado na estação gráfica, que denominamos de gerência de interfaces graficas. O segundo ambiente computacional é o processamento executado no computador hospedeiro, que denominamos de processamento hospedeiro. A figura V.1 apresenta o diagrama de fluxo de dados, primeiro nível, do GIGSTR.

O processamento da Gerência de Interfaces Gráficas do GIGSTR torna disponivel as estruturas de diálogo e as funções aplicativas a serem utilizadas pelo usuário final, através das seguintes tarefas: controlar a interação homem-máquina ; manipular telas a pedido do usuário ou por decisão do processo, (através do processamento hospedeiro); comunicação com o processamento hospedeiro, enviando a tela a ser monitorada ou a variável com seu respectivo valor a ser processada pelo computador hospedeiro, recebendo os dados atualizáveis (dinâmicos) da tela solicitada, ou de uma nova tela com os respectivos dados atualizáveis, neste último caso indicando que o processo deseja que o usuário monitore uma nova tela.

O processamento no computador hospedeiro pode ser dividido em dois ambientes distintos. Um destes ambiente, denominado anteriormente de processamento hospedeiro, processa as funções requisitadas pelo GIGSTR, enviando os dados atualizáveis (dinâmicos) da tela recebida ou enviando uma nova tela que o processo deseja que seja monitorada em tempo de ocorrência de um evento assíncrono, além de receber uma tela à

ser monitorada pela estação gráfica ou uma variável a ser atualizada no banco de dados da aplicação. O outro ambiente de processamento do computador hospedeiro não é destinado às funções do GIGSTR e sim às funções particulares de cada aplicação, como por exemplo, aquisição de sinais do processo, armazenamento desses sinais em um banco de dados da aplicação, processamento de variáveis calculadas e controle sobre o processo.

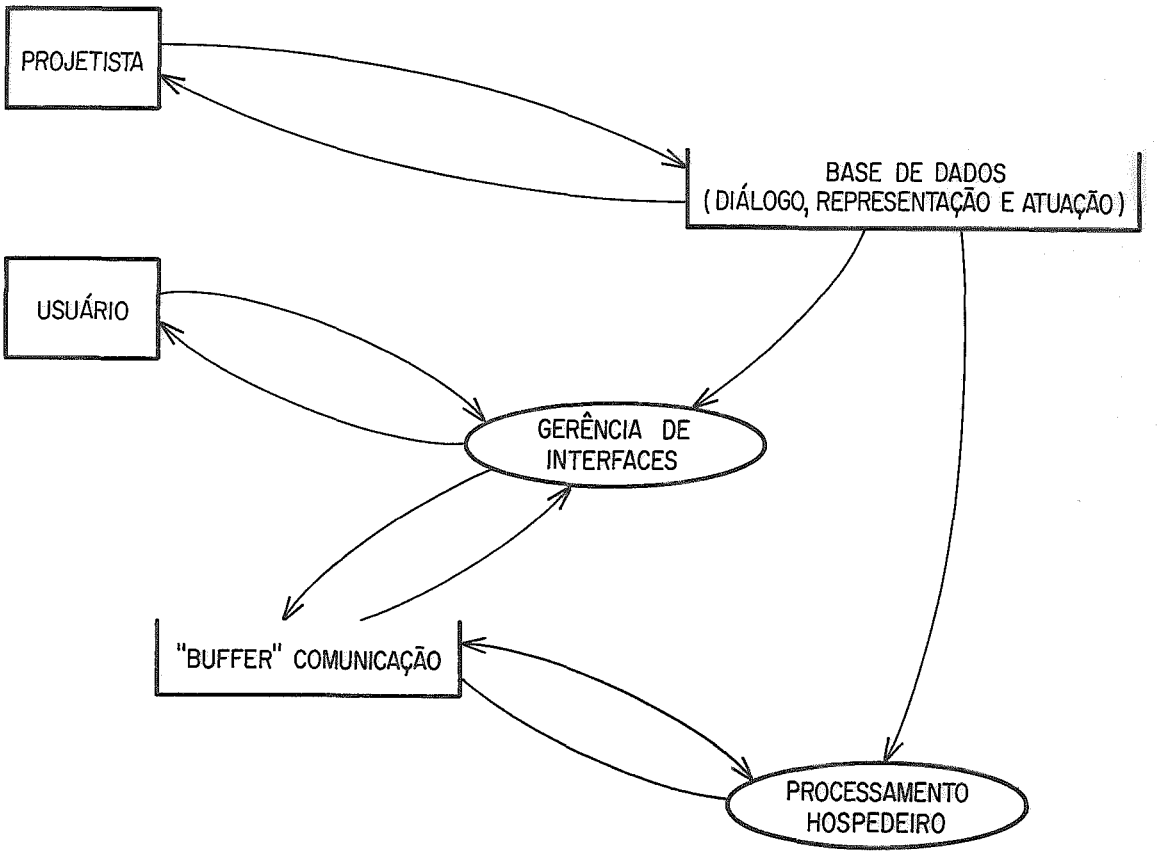

FIGURA V.1 - Diagrama de fluxo de dados (1º nível) do GIGSTR.

Utilizaremos conceitos da metodologia de análise estruturada de sistemas, proposta por GANE e SARSON [20], para efetuarmos a descrição funcional , mais especificamente o "diagrama de fluxo de dados" (DFD), que mostra as origens e os destinos dos dados, identifica e denomina as funções lógicas e os grupos de elementos de dados que ligam uma função a outra, além de identificar os depósitos de dados. O DFD utiliza de símbolos que representam os processos (elipses), depósitos de

dados (retângulos abertos), entidades externas (quadrados) e fluxo de dados (setas).

V.1. *Descrição Funcional da* Gerência de Interfaces Gráficas *do GIGSTR*

A descrição funcional da gerência de interfaces gráficas está baseada no diagrama de fluxo de dados (primeiro nível) Figura V.2. Em uma análise superficial do DFD podemos visualizar duas seqüências de processamento. Uma onde o usuário efetua interações, através de ações com o ratinho e teclado, que serão tratadas pelo controle da interface homem-máquina (IHM). O resultado desta seqüência é a apresentação de informações para o usuário e a comunicação com o processamento hospedeiro. A outra seqüência de processamento parte do processamento hospedeiro enviando dados a serem atualizados em tela pela gerência de interfaces gráficas.

A seguir faremos uma descrição funcional dos processos (elipses).

V.1.1. *Processos "Ratinho" e "Teclado"*

Os processos "Ratinho" e "Teclado" referenciados na Figura V.2 itens 1 e 2 respectivamente, objetivam adquirir ações efetuadas pelo usuário através dos meios físicos ratinho e teclado, armazenando essas ações como eventos a serem processados pelo controle de IHM. Os eventos são ações do usuário para interagir com o sistema, como por exemplo, posicionar o cursor, selecionar um item do cardápio ou selecionar um botão na tela.

V.1.2. *Processo "Controle de IHM"*

O processo de "Controle de IHM" referenciado na Figura V.2 item 3, tem a função de processar cada um dos

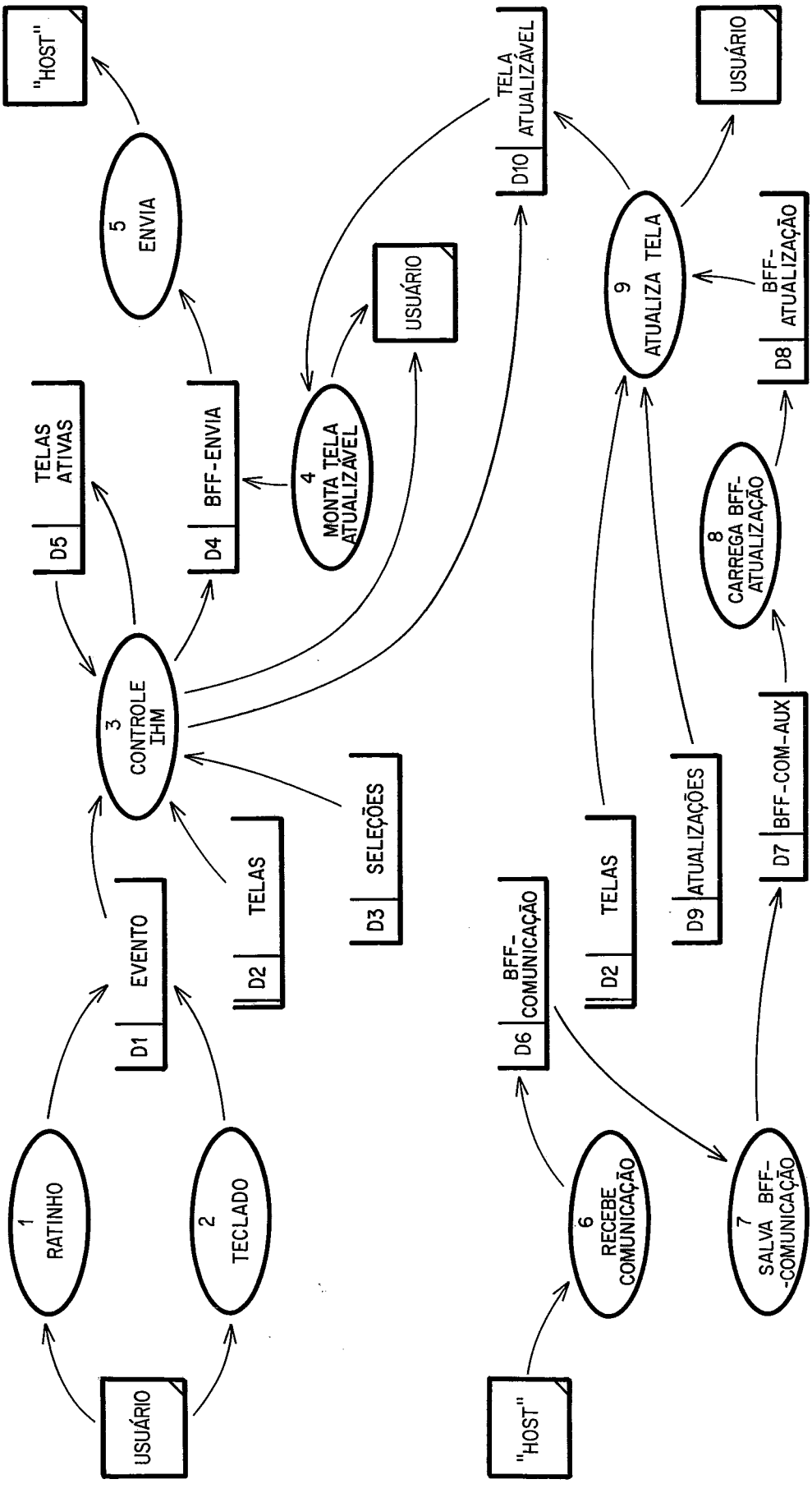

Figura V.2 - Diagrama de fluxo de dados (1º nível) da Gerência de Interfaces.

possíveis eventos gerados pelo ratinho e teclado, com o objetivo de produzir uma saída em terminal de vídeo, como por exemplo, montar uma tela, exibir um cardápio, selecionar um botão na tela e etc.

Os eventos produzidos estão relacionados às telas do sistema, onde uma tela pode ser tela base ou tela atualizável ou estruturas de interação com o usuário, tais como: tela de cardápio, tela de descrição de apoio, tela de controle de acesso (senha), tela de comando, tela de atuação e tela de fim de processamento. Existem determinados tipos de eventos que não estão diretamente relacionados a uma determinada tela e sim a todas as telas, estes eventos são produzidos pelas teclas dedicadas, que serão definidas posteriormente.

O processo de controle de IHM está estruturado para identificar uma tela de modo genérico, isto é, o controle de IHM não necessita saber a priori quais funções de interação foram usadas pelo projetista para compor uma determinada tela. Com isso o processo de controle de IHM só irá identificar uma tela, a partir do momento que o usuário a solicite, tornando necessário que a estrutura de criação da tela seja padronizada. A estrutura de criação de uma tela pode conter: uma imagem estática , campos de atualizações dinâmicos e itens de seleção.

O processo de controle de IHM foi definido baseado no conceito de que uma tela, seja ela de qualquer tipo (cardápio, base, atualizável, etc.), possue uma associação unívoca com um determinado conjunto de "itens de seleção". Deste modo o controle de IHM trata os itens de seleção independentemente do tipo da tela, por exemplo, o controle de IHM tratará os itens de seleção de uma tela de cardápio da mesma forma que os itens de seleção de uma tela base, sendo o tratamento dependente somente do tipo do item de seleção e não da função da tela ao qual este item está associado. Deste modo a navegação em uma tela é dirigida pelos itens de seleção que a tela contém, ou seja, uma tela contém itens de seleção que apontam para outras telas, sejam de atualização ou não (telas

de cardápio, telas de descrição de apoio, tela de comando ou de fim de processamento).

O controle de IHM trabalha com o método de tela ativa, ou seja, todos os eventos são direcionados para a última tela montada, a menos dos eventos de teclas dedicadas. Por exemplo, numa tela de cardápio onde um dos itens de seleção chama uma nova tela de cardápio, os itens de seleção do primeiro cardápio ficam congelados e todos os eventos produzidos pelo ratinho ou teclado serão associados para o segundo cardápio.

O controle de IHM trata os eventos de teclas dedicadas de duas formas distintas. Em uma das formas a tecla dedicada seleciona uma das telas do sistema (tela atualizável, cardápio, comando, etc.). Na outra forma a tecla dedicada está direcionada para a atuação de uma variável digital, alterando o estado (0 e 1) atual desta variável. O controle de IHM prepara uma mensagem (em "buffer") com a variável e seu novo estado, de modo a permitir que o processo "envia" (Figura V.2 item 5) transmita a alteração para o computador hospedeiro, com o objetivo de atualizar a base de dados da aplicação. A seguir faremos algumas considerações sobre as telas do sistema.

A tela base tem o objetivo de auxiliar o usuário, exibindo as principais funções disponíveis, permitindo o selecionamento das mesmas. Caso o projetista tenha definido a tela base, a mesma será montada uma única vez e será exibida durante todo o processamento, não sendo removida a menos da condição de "fim de processamento".

A tela atualizável compõe juntamente com a tela base a área física do monitor de vídeo. A tela atualizável é a única tela que não é montada pelo controle de IHM, devido ao fato de que uma tela atualizável pode ser solicitada pelo usuário ou por decisão da aplicação (processo executado no computador hospedeiro).

As telas de cardápio e de descrição de apoio tem

o seu processamento dentro do controle de IHM executado de modo similar, onde os itens de seleção do cardápio são tratados da mesma forma que os itens selecionáveis do hipertexto.

A tela de comando tem um processamento dentro do controle de IHM diferente das demais, pois o usuário irá digitar o código de uma tela e não selecionar itens.

A tela de atuação permite que o usuário efetue alterações sobre variáveis analógicas, digitais ou calculadas, utilizando uma estrutura de preenchimento de campos. O controle de IHM monta um "buffer" com o código das variáveis e seus respectivos valores (alterados), para que sejam enviados ao computador hospedeiro, com o objetivo de atualizar a base de dados da aplicação.

A tela de fim de processamento é também tratada pelo controle de IHM, podendo conter itens de seleção (sim ou não), com o objetivo de confirmar a opção de término. O término de processamento implica em avisar ao computador hospedeiro que a conexão será desfeita.

V.1.3. Processo "Monta Tela Atualizável"

O processo "Monta Tela Atualizável" (Figura V.2 item 4) tem a função de montar uma tela atualizável solicitada pelo usuário ou montar uma tela que o processamento hospedeiro da aplicação requer que seja monitorada. Uma outra função é montar o "buffer" a ser enviado para o computador hospedeiro, contendo o código da tela atualizável, solicitada pelo usuário, com o objetivo de que o processamento hospedeiro passe a enviar os dados de atualização.

V.1.4. Processo "Envia"

O processo "Envia" (Figura V.2 item 5) transmite os "buffers" de mensagens da estação gráfica para o

computador hospedeiro, são dois os conjuntos de informações que podem compor um "buffer", um contendo somente o código da tela atualizável e o outro contendo o código da tela e as variáveis, com os respectivos valores, a serem atualizados no banco de dados da aplicação.

V.1.5. Processo "Recebe Comunicação"

O processo "Recebe Comunicação" (Figura V.2 item 6) monta um "buffer", denominado de comunicação, com informações transmitidas pelo computador hospedeiro. Os dados recebidos sempre vem agrupados, caracterizando uma tela. O processo recebe comunicação é que detecta se o sistema está ou não em "over run", através da verificação do "buffer", ou seja, se um conjunto de dados está chegando e o "buffer" anterior ainda não foi tratado, podemos evidenciar que o processamento na estação gráfica não consegue executar todas as suas funções dentro do ciclo (por exemplo 2 segundos) que o processo aplicativo requer.

V.1.6. Processo "Salva BFF-Comunicação"

O processo "Salva BFF-Comunicação" (Figura V.2 item 7) copia o "buffer" de comunicação para um "buffer" de comunicação auxiliar, após o "buffer" de comunicação ter sido completado. A necessidade de salvar o "buffer" de comunicação em uma outra área, é evitar o "over run". Enquanto o "buffer" anterior está sendo tratado pelo processo de atualização de tela (Figura V.2 item 9) o "buffer" auxiliar pode conter os dados do último ciclo e liberar o "buffer" de comunicação para que receba os dados do próximo ciclo.

V.1.7. Processo "Carrega BFF-Atualização"

O processo "Carrega BFF-Atualização" (Figura V.2 item 8) copia o "buffer" de comunicação auxiliar para um

"buffer" de atualização, com o objetivo de tornar disponível para o processo de atualização de tela, os dados referentes as atualizações dinâmicas.

V.1.8. Processo "Atualiza Tela"

O processo "Atualiza Tela" (Figura V.2 item 9) tem a função de atualizar os dados dinâmicos das telas atualizáveis, tais como: barra, gráfico, texto, histórico, tendência, símbolo e primitivas gráficas. O "buffer" de atualização contém, quando da primeira vez que a tela irá ser atualizada, todas as variáveis definidas para a tela em questão. A partir da primeira vez o "buffer" de atualização só conterá as variáveis que sofreram modificações. O "buffer" de atualização é formado pelo código da tela, indicativo de mudança de tela e uma estrutura (diferente) para cada função dinâmica. O código da tela (recebido) é uma forma de confirmar que os dados são pertinentes à tela montada. A necessidade de se ter um indicativo de mudança de tela, deve-se ao fato de que o processo hospedeiro aplicativo, pode requerer que o usuário monitore uma tela diferente daquela que ele está monitorando. O processo atualiza tela, identifica cada uma das estruturas de dados, pelo primeiro campo de cada estrutura, que referenciam o código de atualização. Cada tela atualizável contém um conjunto de funções atualizáveis (barra, texto, gráfico etc.) identificadas pela sua função.

Os dados que compõem a função barra são: código de atualização, valor atual da variável o qual a barra está representando e a côr com que a mesma será preenchida. A Figura V.3 apresenta um diagrama de barra, contendo barras horizontais, sendo preenchidas da esquerda para a direita. Apresenta também algumas barras em escala linear, logaritmo neperiano e logaritmo na base decimal.

Os dados que compõem a função texto são: código de atualização, o texto propriamente dito e a côr do texto. A

Figura V.3 apresenta exemplos da função texto. Os demais atributos que compõem o texto, tais como: tipo de caracteres, posição inicial e tamanho do caracter, foram previamente definidos. O processo atualiza tela já recebe os textos em seus formatos definidos pelo projetista, ou seja, o processamento hospedeiro ao enviar os textos monta-os já no formato de exibição em tela.

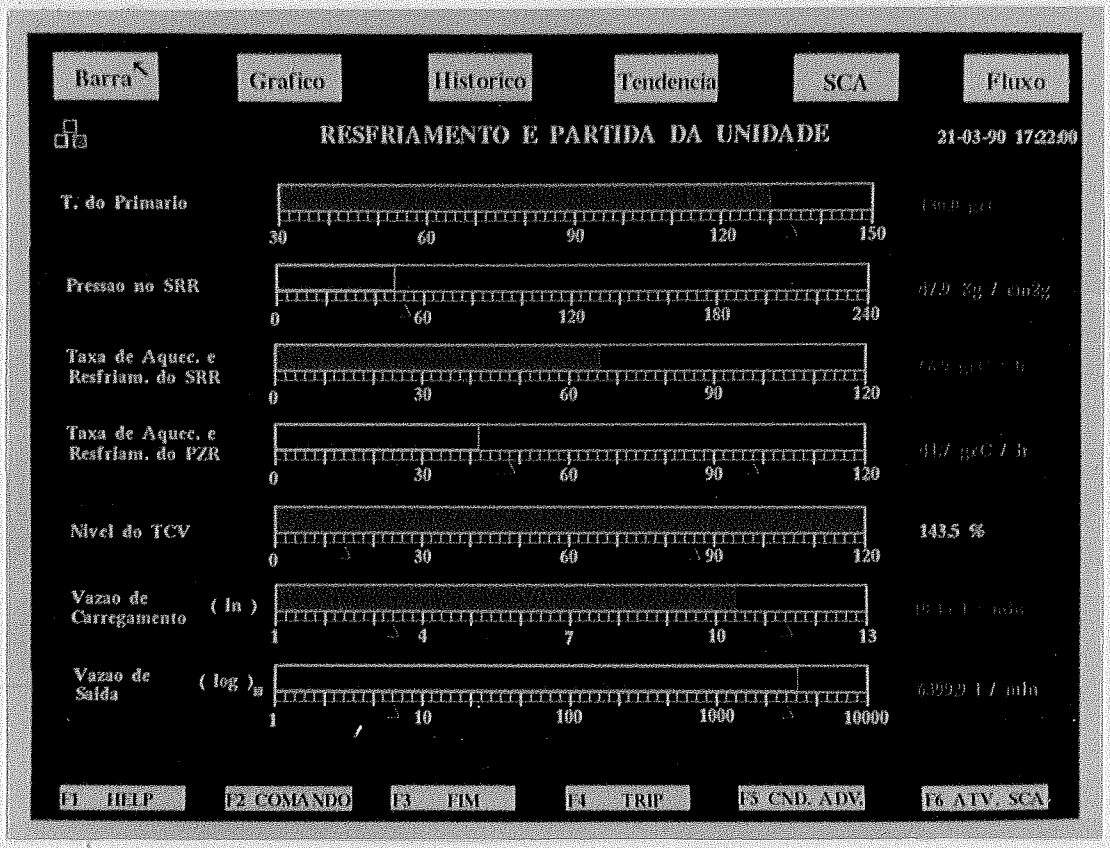

FIGURA V.3 - Diagrama de Barras.

Os dados que compõem a função **gráfico** são: valores dos atributos das variáveis especificadas. A função gráfico objetiva traçar um cursor, previamente definido, sobre um gráfico gerado através das funções de desenhos estáticos (Figura V.4). Uma característica da função gráfico é que o cursor que indica o valor de uma variável sobre um gráfico, pode ou não, deixar um rastro por onde ele passa.

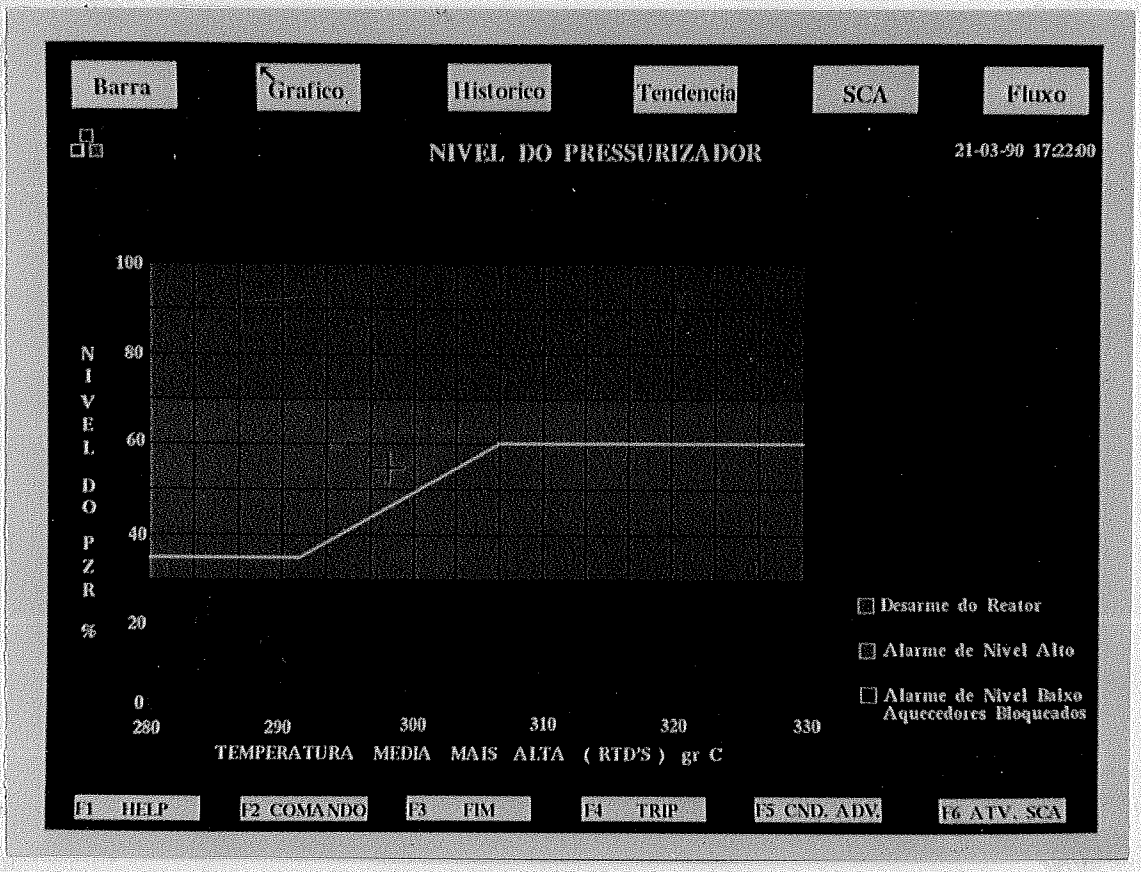

FIGURA V.4 - Exemplo de Gráfico.

Os dados que compõem a função **histórico** são: um vetor contendo os valores de uma determinada variável durante um intervalo de tempo previamente definido. A função histórico traça um gráfico bidimensional (Figura V.5), onde um dos eixos é formado pelos valores obtidos do vetor e o outro eixo é determinado pelo intervalo de tempo, o qual, os dados foram adquiridos. A função de histórico representa graficamente os valores das seguintes maneiras: acende pontos na tela podendo uní-los ou não, traça linhas a partir da base podendo uní-las ou não e traça uma barra podendo enchê-la ou não.

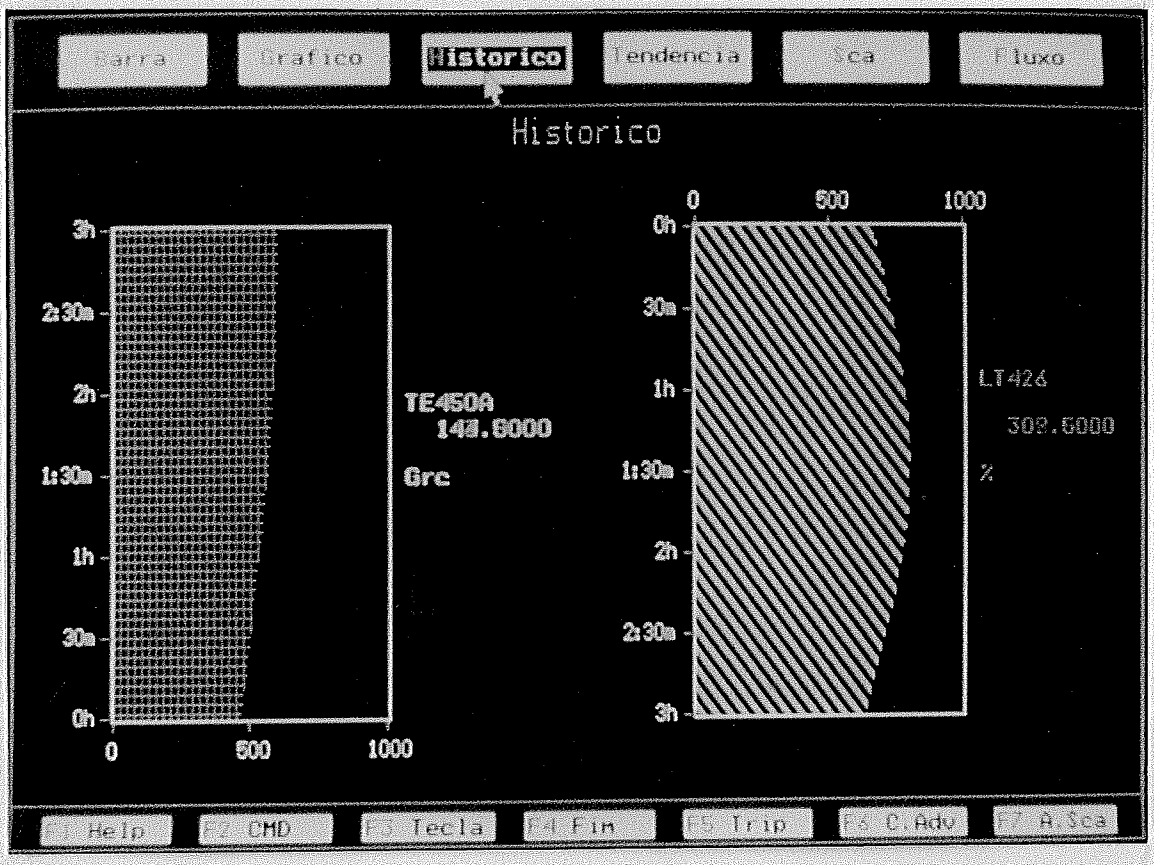

FIGURA V.5 - Exemplo de Históricos.

O dado que compõe a função **tendência** é o valor atual da variável. A função tendência sempre recebe o valor atual da variável, independente de sua variação ou não. O objetivo da função é apresentar a variação, em forma gráfica, de uma determinada variável ao longo de um intervalo de tempo pre-estabelecido. Para cada valor que chega a função tendência desloca o gráfico, que contém a representação (gráfica) dos valores anteriores, de uma unidade do espaço ocupado pelo gráfico de tendência e a escala de tempo por ele representada, inserindo o novo valor na forma gráfica pre-definida. As formas gráficas para o traçado de um gráfico de tendência são as mesmas utilizados pelos gráficos de histórico.

O dado que compõe a função **símbolo** é o "status" da variável, que em geral é representado por um valor binário. Uma outra forma de representação seria o indicativo, por exemplo, de que um gerador está inoperante.

Os dados que compõem a função **primitivas gráficas** são determinados pelos parâmetros de cada uma das funções , por exemplo, para se acender um "pixel" na tela os parâmetros relativos a esta função são: valor de X e Y do "pixel" e a côr com que o "pixel" será aceso. A função primitivas gráficas objetiva a complementar as demais funções básicas do GIGSTR, que por ventura não atendem a uma determinada função "especifica" da aplicação. O processo "atualiza tela" do GIGSTR através das primitivas gráficas interpreta os "comandos" gráficos, que foram enviados pelo computador hospedeiro, mas adquiridos e processados pela aplicação.

A Figura V.6 apresenta uma tela com desenhos estáticos e a figura V.7 representando uma pluma radioativa, sobre o desenho da figura V.6, que foi atualizada pelas funcões primitivas gráficas, sendo que os cálculos e a forma de representação na tela foi efetuado por processos da aplicação.

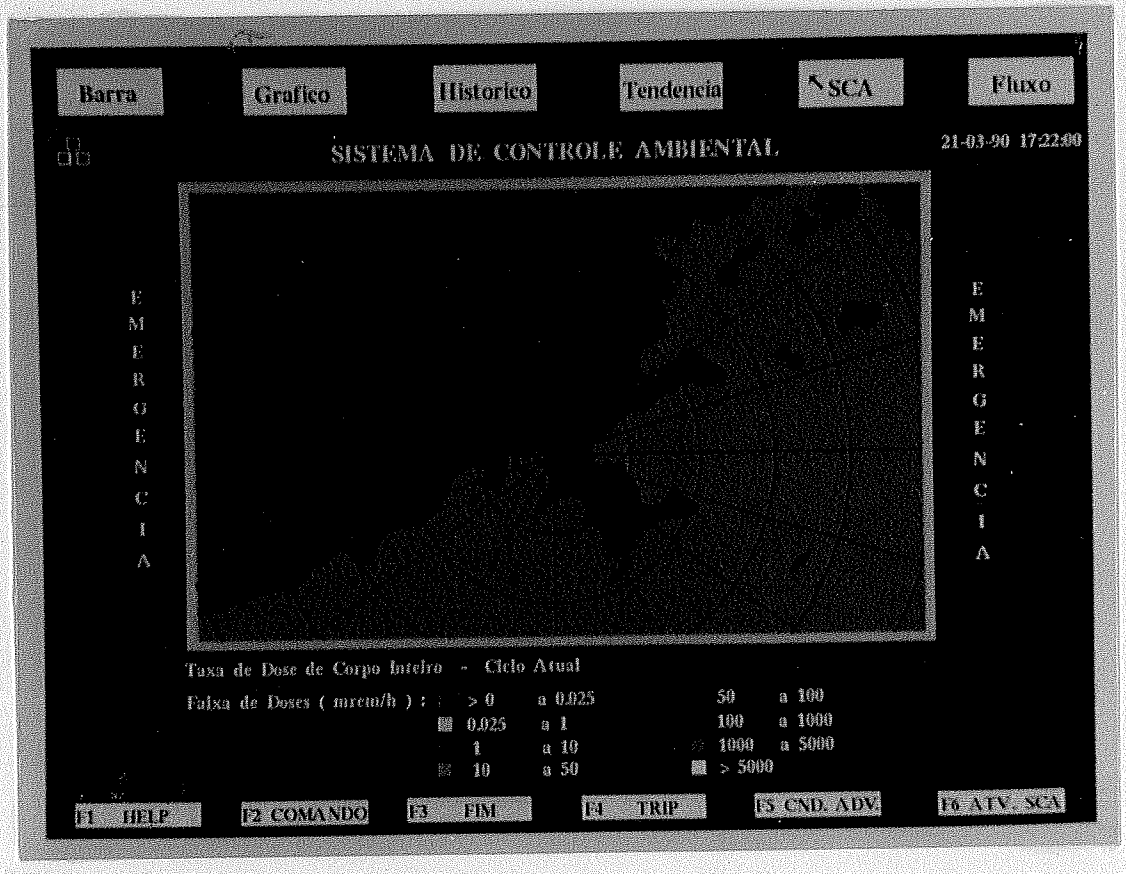

FIGURA V.6 - Localização da Usina Nuclear de Angra-I.

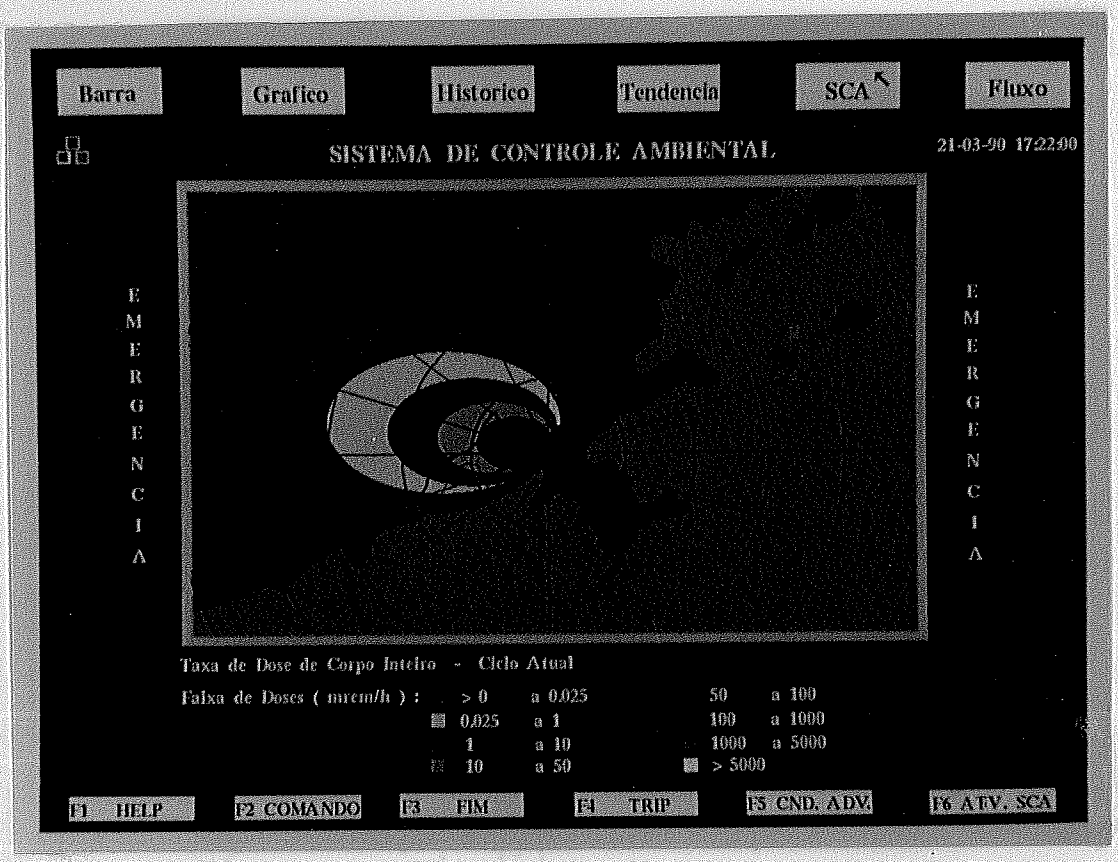

FIGURA V.7 - Representação de uma Pluma Radioativa.

V.2. *Descrição Funcional do Processamento Hospedeiro*

O processamento do computador hospedeiro pode ser dividido em dois ambientes, conforme apresenta o Diagrama de Fluxo de Dados (primeiro nível) Figura V.8. O DFD apresenta uma linha tracejada com o objetivo de identificar o processamento efetuado pelo GIGSTR (lado esquerdo) e o processamento efetuado pela aplicação (lado direito).

O processamento a ser efetuado no computador hospedeiro pelo GIGSTR tem a função de receber os dados enviados pela estação gráfica, que podem ser de uma tela a ser monitorada pela estação gráfica ou de uma variável a ser gravada no banco de dados da aplicação. Uma outra função que faz parte do GIGSTR é a preparação dos dados atualizáveis (dinâmicos) da tela recebida, para que sejam enviados à estação gráfica.

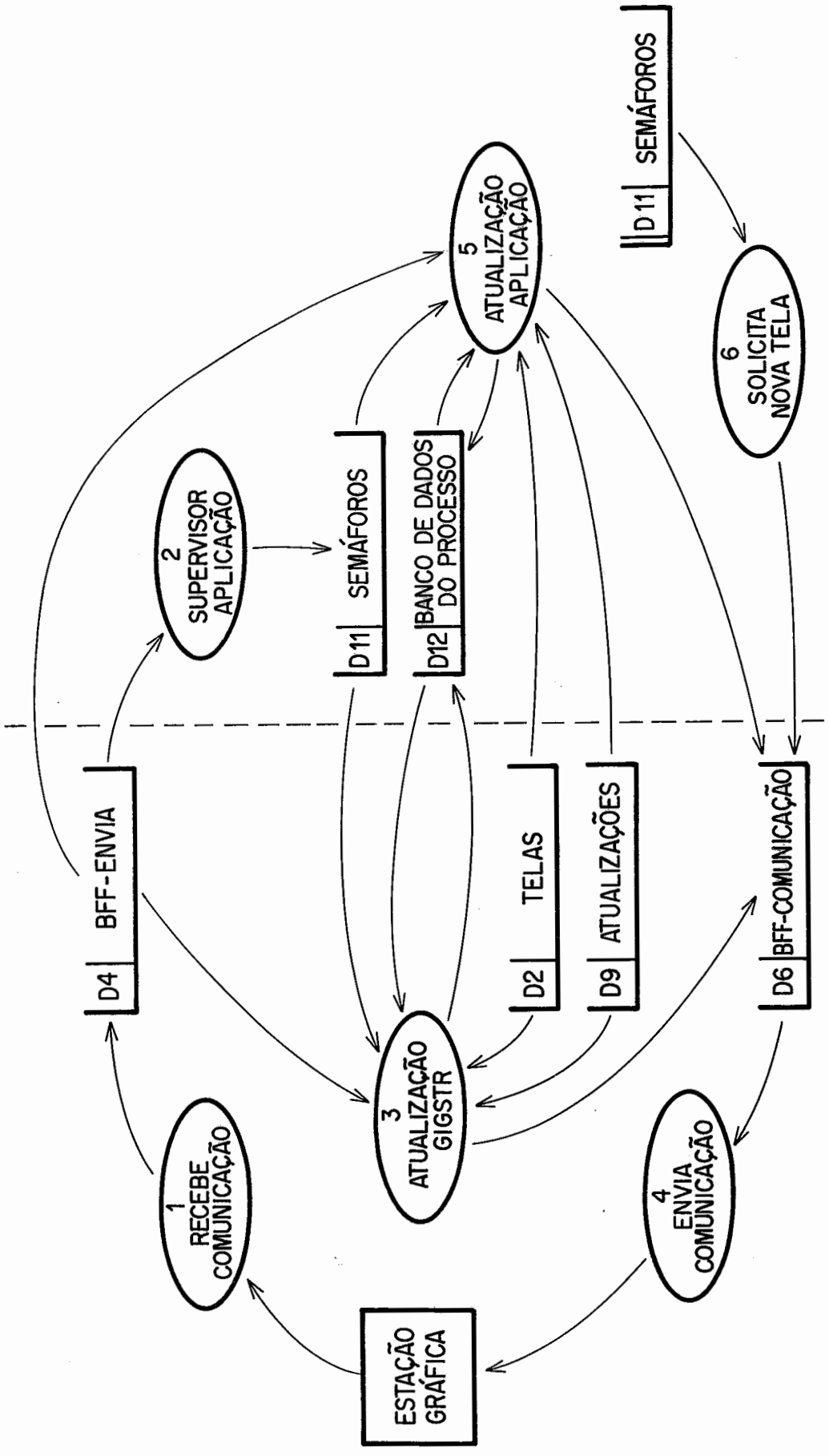

Figura V.8 - Diagrama de fluxo de dados (1º nível) do processamento Hospedeiro.

A outra parte do processamento a ser realizada no computador hospedeiro, pertence à aplicação, como por exemplo, adquirir e armazenar os sinais do processo, processar variáveis calculadas, controle sobre o processo e etc. O DFD apresenta três processos que serão utilizados para exemplificar funções da aplicação, que interagem com o GIGSTR.

A seguir apresentamos uma descrição dos processos que compõem o processamento hospedeiro.

V.2.1. Processo "Recebe Comunicação"

O processo "recebe comunicação" descrito na Figura V.8 item 1, tem a função de receber os dados transmitidos pela estação gráfica, armazenando-os para que sejam identificados pelo processo "supervisor" (Figura V.8 item 2) e consequentemente tratados pelos processos "atualização GIGSTR" e "atualização aplicação". Os dados recebidos podem ser de uma tela a ser monitorada ou de um valor de uma variável a ser atualizada no Banco de Dados da aplicação, caracterizando uma atuação do usuário sobre o processo aplicativo.

V.2.2. Processo "Supervisor da Aplicação"

O processo "supervisor da aplicação" (Figura V.8 item 2) é um processo aplicativo que gerencia todo o processamento a ser atualizado no computador hospedeiro, não é objetivo deste trabalho descrever as funções do processo "supervisor da aplicação" e sim definir como o GIGSTR interage com os processos aplicativos. Em geral os processos supervisores controlam os demais processos, utilizando a técnica de identificação de eventos, eventos esses que são determinados pelos ciclos de aquisição dos dados. Por exemplo, a cada segundo é feita a aquisição de um conjunto de variáveis analógica, com isso, o processo supervisor dispara ordenadamente os processos de aquisição, armazenamento e

cálculos desse conjunto de variáveis analógicas.

Um dos métodos utilizados em processamento de tempo real, para que o supervisor controle os processos ordenadamente ou concorrentemente é através de semáforos e manipulação da prioridade dos processos. Os processos do GIGSTR serão regidos pelo processo "supervisor da aplicação" da mesma forma que os demais processos da aplicação. Onde o supervisor identifica a chegada de dados da estação gráfica e dispara o processo "recebe comunicação" para recebê-los. Os demais processos do GIGSTR ocorrem de acordo com o evento de recebimento da informação e os eventos cíclicos de aquisição dos dados da aplicação.

V.2.3. Processo "Atualiza GIGSTR"

O processo "atualiza GIGSTR" (Figura V.8 item 3) efetua dois procedimentos de atualização. Um procedimento é a atualização de dados no banco de dados da aplicação, através de informações provenientes da estação gráfica, que caracteriza uma atuação do usuário sobre o banco de dados da aplicação, cabendo ao supervisor decidir se os dados atualizados serão ou não atuados sobre o processo (controle do processo). O outro procedimento a ser realizado pelo processo "atualiza GIGSTR" é a preparação dos dados atualizáveis (dinâmicos) para que sejam exibidos pela estação gráfica.

A preparação dos dados atualizáveis a serem enviados à estação gráfica está baseada no recebimento de um código de tela, que foi solicitada pelo usuário. A partir desse código de tela o processo identifica, através da estrutura de dados "telas" (D2), quais as funções atualizáveis que compõem a tela e utilizando a estrutura de dados "atualizações" (D9), que contém informações sobre cada função de atualização, prepara um "buffer" com os dados atualizáveis, obedecendo as seguintes regras:

- A função barra é formada pelo atributo valor

atual e atributo côr da variável. Em geral a côr é representada pelo estado da variável e é função da aplicação a correlação entre o código de côr e o código de estado, por exemplo, estado 1 corresponde a côr verde, que quer dizer que a variável está dentro dos limites de "set-point".

- A função de gráfico é formada pelos valores dos atributos da variável, onde os atributos da variável podem ser qualquer um definido como inteiro ou real, por exemplo, valor atual, "set point" de alta ou de baixa e etc. A determinação de um ou dois valores a serem enviados é dada pela definição do gráfico (direção).

- A função de texto é formada pelo atributo da variável (descrição, valor atual, "set point" e etc.) e a côr do texto. Os valores reais que requerem formatos especiais, serão transformados em caracteres com o objetivo de evitar que a estação gráfica, efetue a transformação na hora de exibição da informação em tela.

- A função de histórico é formada por um vetor contendo os valores de uma variável durante um intervalo de tempo já passado. Uma das definições da função histórico é a hora de início e hora de fim que determina o traçado do gráfico de histórico, uma outra definição é de quanto em quanto tempo o histórico é armazenado pela aplicação. Com estas informações e mais a área na tela (pixel) que o histórico ocupa, podemos calcular o tamanho do vetor a ser enviado. Por exemplo, se tivermos um intervalo de tempo de 3600 segundos, um ciclo de armazenamento das variáveis históricas de 4 segundos e uma área na tela de 450 "pixels", teremos 450 valores a serem transmitidos, onde cada valor será representado por um "pixel" e os dados foram obtidos um sim e outro não devido a quantidade de informações históricas e o espaço disponível na tela, ou seja, 3600 segundos contém 900 valores (3600/4) armazenados, resultando na obtenção de um dado sim e outro não em função do tamanho em "pixel" (450).

- A função tendência é formada pelo valor atual

da variável. Tendo uma característica diferente das demais funções, pois mesmo que o valor atual e anterior sejam iguais, assim mesmo serão transmitidos à estação gráfica.

- A função de símbolo é formada pelo "status" da variável digital, com o objetivo de representar mudanças de estados (em geral binários), por exemplo, ventilador ligado ou desligado, válvula aberta ou fechada e bomba funcionando ou não.

V.2.4. Processo "Envia Comunicação"

O processo "envia comunicação" (Figura V.8 item 4) tem a função de transmitir os dados do computador hospedeiro para a estação gráfica.

V.2.5. Processo "Atualização da Aplicação"

O processo "atualização da Aplicação" (Figura V.8 item 5) não faz parte do processamento do GIGSTR e foi representado para exemplificar que qualquer processo da aplicação pode atuar sobre a estação gráfica, através das função primitivas gráficas (traçar linha, retângulo, polígonos ou acender um pixel na tela).

V.2.6. Processo "Solicita Nova Tela"

O processo "solicita nova tela" (Figura V.8 item 6) também não faz parte do processamento do GIGSTR, e foi representado para exemplificar que qualquer processo da aplicação pode requerer que uma particular tela seja apresentada ao usuário para que seja monitorada.

V.3. DESCRIÇÃO DOS DADOS

A apresentação da descrição dos dados

encontra-se no Anexo I, contendo uma descrição dos depósitos de dados e seus respectivos elementos de dados.

V.4. SEQÜENCIA DE ATIVAÇÃO DA GERÊNCIA DE INTERFACES GRÁFICAS

A seqüência de ativação da gerência de interfaces gráficas objetiva representar o controle de execução de processamentos concorrentes, utilizando padrões gráficos para representar sinalizadores (semáforos), processos e prioridades. A seqüência de ativação não representa entrada e saída de dados e sim eventos que condicionam a ativação de um determinado processo e quais os eventos que são ativados por um determinado processo.

A representação gráfica utilizada na Figura V.9 para descrever a seqüência de ativação da gerência de interfaces gráficas do GIGSTR foi desenvolvida e utilizada no projeto do sistema de monitoração de parâmetros de segurança da usina nuclear de Angra-I, SCHIRRU, MARTINEZ, THOMÉ, BELLAS e colaboradores [21]. Este tipo de representação gráfica foi baseada na teoria de redes de "PETRI" PETERSON [22], não sendo objetivo deste trabalho o detalhamento e a consistência deste formalismo de "PETRI" modificado. Decidimos usar esta representação em função da facilidade de interpretação que a mesma permite.

O modêlo da Figura V.9 pode ser facilmente interpretado bastando, levar em conta as seguintes regras da representação gráfica:

1. "TOKEN": O conceito de "token" utilizado para a rede de "PETRI" é mantido. Este conceito é baseado no fato de que a "CPU" só pode executar instruções de forma sequencial, e portanto num determinado tempo somente uma única tarefa (ação) pode estar sendo processada pela "CPU". O "token" pode ser representado como uma marca que "flui" entre as tarefas a medida que as mesmas estão sendo processadas.

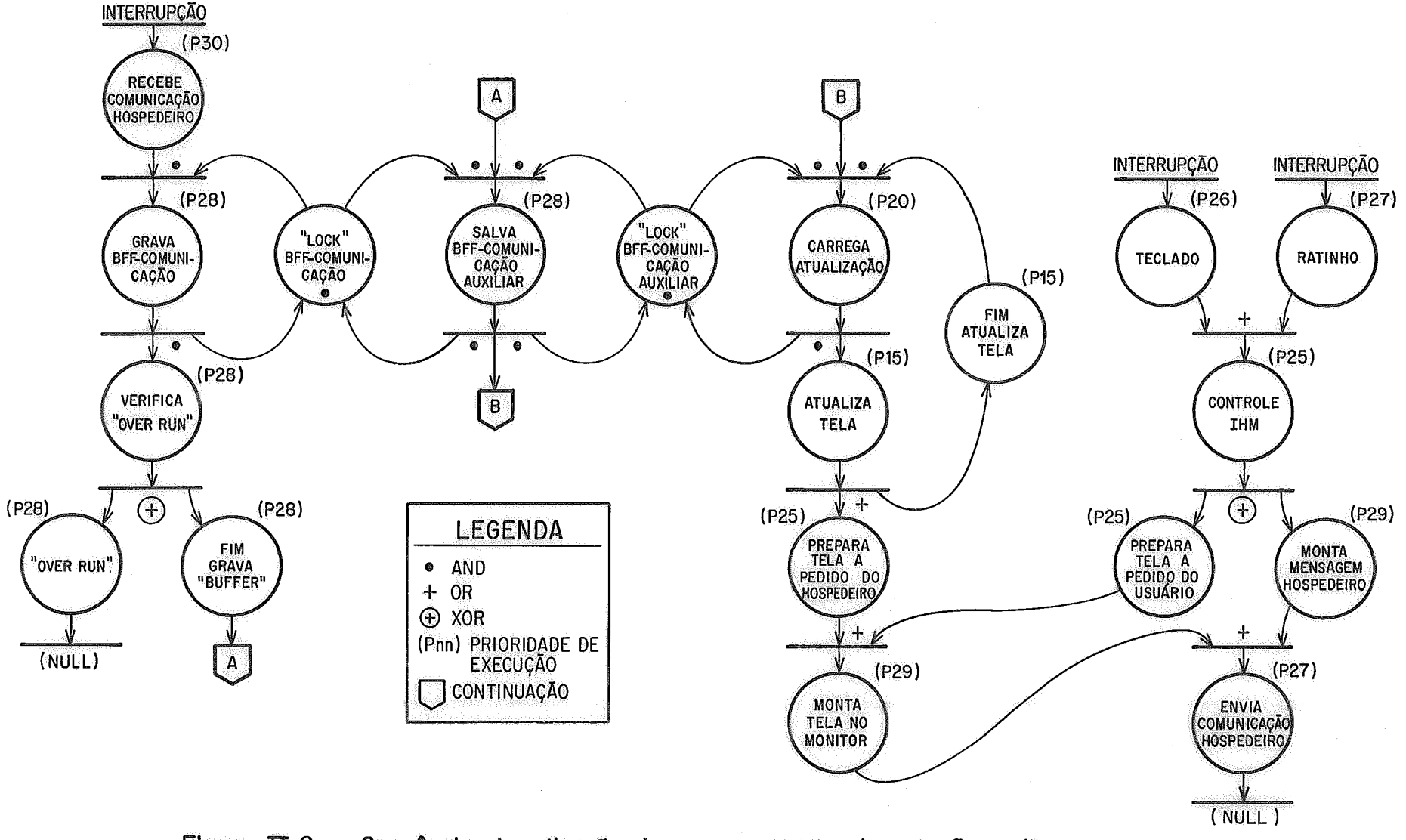

Figura V.9 - Sequência de ativação do processamento da estação gráfica.

2. CÍRCULO: Um círculo significa uma tarefa (condição da CPU), não interruptível no mesmo nível de prioridade. Para que uma determinada tarefa tenha o direito a usar a "CPU" em um determinado tempo, existem eventos que estabelecem as condições de início. Ao término de uma tarefa são criados eventos que servirão de condições iniciais para outra tarefa.

3. TRAÇOS: Os traços significam transições a serem "queimadas", isto é, cada transição é formada por uma lógica Booleana de eventos e quando o resultado da lógica for verdadeiro para os eventos ligados (existentes em um determinado tempo), a transição tem as suas condições iniciais satisfeitas (eventos de entrada ligados) possibilitando que a tarefa associada seja processada pela "CPU". Sempre deve existir uma tarefa (círculo) associada a uma transição (traço), deste modo com as condições iniciais satisfeitas (transição "queimada") a tarefa é processada (o círculo possue um "token"). A única excessão é o caso de uma transição (NULL), que caso ocorra deixará a CPU no estado de processamento "IDLE".

4. SETAS: As setas têm funções análoga aos DFDs, só que em vez de representarem um fluxo de dados, representam um fluxo de controle ("tokens"), isto é, eventos de entrada (condição para o processamento da tarefa) e eventos de saída (resultado do processamento da tarefa). As setas podem ser vistas como o "caminho" por onde "fluem" os "tokens".

5. PRIORIDADES: As prioridades são associadas às tarefas e definem, em caso de eventos assíncronos, que ocorram coincidentemente, qual das tarefas, que possui o "token", gerado por esses eventos, deverá em primeiro lugar ter o acesso à CPU.

6. LÓGICA DO DIAGRAMA DE FLUXO DE CONTROLE: Eventos síncronos ou assíncronos gerados por interrupções de "hardware" (relógio, I/O, etc.) ou pelo processamento de tarefas, geram "tokens" (um "token" para cada seta de saída),

quando a lógica Booleana de eventos ligados que representam as condições iniciais de uma transição é satisfeita (o "token" chega na transição), esta é "queimada" e consequentemente a tarefa associada, é executada pela "CPU" (tarefa que passa a conter o "token"). Ao término da tarefa novos eventos são gerados (gerando novos "tokens") provocando a "queima" de outras transições, mantendo desse modo o processo em operação.

A fim de possibilitar uma melhor visualização da lógica de controle de Figura V.9 vamos analisar uma das possíveis situações em que o sistema poderia encontrar-se em um determinado instante de tempo: o usuário aciona a tecla de seleção do ratinho, gerando deste modo uma interrupção de "hardware", e consequentemente gerando um "token" para a tarefa "ratinho" da Figura V.9. Neste ponto podemos imaginar o "token" como representado pelo símbolo de um círculo pequeno cheio e posicionado dentro do círculo que representa a tarefa "ratinho". A transição (traço) que permitirá que a tarefa "controle de "IHM" seja processada pela CPU é formada pelas condições iniciais dos eventos "foi acionado o teclado" ou "foi acionado o ratinho". Como neste caso a transição é formada por uma lógica de "OU" o "token" que se encontra dentro da tarefa "ratinho" pode "fluir" em direção a transição, que será "queimada", possibilitando que o "token" passe agora a residir dentro da tarefa "controle IHM" e consequentemente esta tarefa estará sendo processada pela CPU.

Vamos supor que neste mesmo instante de tempo ocorreu uma interrupção de "hardware" devido a chegada de dados comunicados pelo hospedeiro. Imediatamente foi gerado um "token" na tarefa "Recebe Comunicação Hospedeiro" da Figura V.9, e para este instante de tempo duas tarefas possuem "token" ("Controle de IHM" e "Recebe Comunicações Hospedeiro"). Em virtude do fato da tarefa "Recebe Comunicação Hospedeiro" ter prioridade de execução (P30) maior que a prioridade da tarefa "Controle IHM" (P25) a CPU deverá interromper o processamento da tarefa "Controle IHM" e deverá passar a executar a tarefa "Recebe Comunicação Hospedeiro". Deve-se notar que neste instante existem 2 "tokens" no sistema e portanto vários

caminhos para o "fluxo" dos mesmos. O fluxo de processamento da CPU será função da prioridade das tarefas que possuem "tokens" e das condições de eventos gerados (novos tokens) no sistema a medida em que com o passar do tempo as tarefas forem sendo processadas. Ao simularmos os possíveis caminhos que formam o fluxo de controle da Figura V.9 devemos ter em mente que os círculos que representam mecanismos de exclusão mútua tem por definição da rede de Petri um "token" associado.

A representação gráfica utilizada permite a análise do fluxo de controle do sistema a ser implementado, facilitando não só a identificação de situações que poderiam provocar um "deadlock", como possibilitando uma modelagem da simulação do projeto de tempo real.

A seguir definimos os eventos e semáforos que deveriam ser implementados a partir da Figura V.9 de modo a representar a seqüência de ativação do processamento da gerência de interfaces gráficas. Também teceremos alguns comentários sobre a forma de interpretação da Figura V.9.

- Interrupção - É um evento que indica uma interrupção do "hardware", tais como: linha de comunicação, teclado ou ratinho.

- E1 - Evento que indica "over run" no sistema, ou seja, chegaram novas informações do computador hospedeiro e a gerência de interfaces ainda não tinha processado as informações anteriores.

- E2 - Evento indicando que novas informações vindas do computador hospedeiro estão disponíveis.

- E3 - Evento indicando qual a seleção feita com o ratinho ou teclado em função da lógica do controle de IHM.

- E4 - Evento indicando que tem informações a serem atualizadas na tela.

- E5 - Evento indicando término de atualização

das informações em tela.

- E6 - Evento indicando que tem tela de atualização a ser apresentada no monitor de vídeo.

- E7 - Evento indicando que o "buffer" de comunicação está pronto para ser enviado ao computador hospedeiro.

- E8 - Evento indicando uma ação de interação do usuário, através dos dispositivos físicos ratinho ou teclado.

- S1 - Semáforo para a exclusão mútua do acesso para a área do "buffer" de comunicação.

- S2 - Semáforo para a exclusão mútua de acesso para a área do "buffer" de comunicação auxiliar.

A forma de ativação de um processo é efetuada pela combinação de eventos ativos (ligados). Por exemplo, o processo "Salva BFF-Comunicação Auxiliar" só será ativado se o "buffer" de comunicação estiver liberado e (.) a confirmação de que os novos dados vindos do computador hospedeiro foram completados (E2) e (.) o "buffer" de comunicação auxiliar esteja liberado para uso.

A priorização dos processos complementa a seqüência de ativação, com o objetivo de identificar os processos que serão "travados" temporariamente em função de outros processos mais prioritários quando da ocorrência de eventos assíncronos, como por exemplo, o processo "Recebe Comunicação" deve ser o de maior prioridade, para garantir que não haja perda de informações durante a transmissão de dados do computador hospedeiro para estação gráfica.

É importante notar que a representação gráfica utilizada, possibilita identificar facilmente os mecanismos de exclusão mútua, que devem ser implementados, como por exemplo, na Figura V.9 os processos "Carrega Atualização" e "Salva

BFF-comunicação Auxiliar" são mutuamente exclusivos, pois para serem ativados dependem da liberação da área "buffer" de comunicação auxiliar, que possui um bloqueio de semáforos para acesso de leitura e escrita, permitindo deste modo que apenas um dos processos seja "liberado" pelos semáforos de "lock".

CAPITULO VI

IMPLEMENTAÇÃO DO GIGSTR

O GIGSTR foi implementado com base nas especificações definidas nos capítulos IV e V deste trabalho. foram utilizados um computador "MICROVAX II" com sistema operacional VAX/VMS com a função de hospedeiro e um microcomputador PC/AT-286 para exercer as funções da estação gráfica. O objetivo deste capítulo é o de descrever as técnicas e especificações utilizadas para a implementação do GIGSTR dentro de um conceito de prototipação.

A seguir serão apresentados os conceitos sistêmicos da implementação do GIGSTR, tanto na estação gráfica como no computador hospedeiro, ressaltando-se que a ligação lógica e física entre os dois ambientes é definida pelo protocolo de comunicação entre os mesmos, sob o controle dos processos de comunicação do GIGSTR.

VI.1. Implementação do GIGSTR na Estação Gráfica

O ambiente computacional que compõe a estação gráfica é formado por: microcomputador PC/AT com processador INTEL 80286, coprocessador INTEL 80287, memória de 640 Kbytes, placa gráfica EGA, ratinho LOGITECH de 3 botões, 2 (duas) placas seriais, um monitor de vídeo colorido MULTISYNC e sistema operacional MS-DOS.

O sistema operacional MS-DOS não é um sistema operacional indicado para o processamento de tempo real com tarefas concorrentes, principalmente quando comparado com um sistema operacional do tipo UNIX, que permite ferramentas naturais de sincronização e priorização de tarefas.

Levando-se em conta que o principal objetivo deste trabalho é a conceituação de um gerenciador de interação homem-máquina gráfico para sistemas de tempo real, a idéia de uma implementação prototipada do GIGSTR nos pareceu bastante indicada, e dentro deste ponto de vista o ambiente MS-DOS oferece vantagens em relação ao UNIX, principalmente em função das ferramentas de desenvolvimento disponíveis (turbo C, debug, documentação e etc.). É para nós bastante claro que a implementação em forma final do GIGSTR, deverá ser feita sob um ambiente operacional do tipo UNIX, contudo para a implementação do protótipo do GIGSTR o ambiente operacional MD-DOS é satisfatório.

Afim de poder dispor das facilidades de ferramentas de desenvolvimento de "software" do ambiente MS-DOS adotamos as seguintes técnicas para um ambiente concorrente e priorizável sob o MS-DOS.

• Instalação de programas residentes ligados às interrupções do equipamento (COM1 e COM2), que operam de modo concorrente com o controle de IHM e que se comunicam através de áreas comuns de memória protegida por semáforos, de modo a garantir a integridade de acesso para leitura e escrita.

• Utilização do nível de priorização dado pelo MS-DOS às interrupções do equipamento, tais como as interrupções das portas seriais (COM1 e COM2), que são atendidas de modo mais prioritário, permitindo a interrupção do contexto aplicativo para o tratamento da comunicação com o computador hospedeiro (COM1) ou do ratinho (COM2).

De modo a permitir um relativo controle do ambiente MS-DOS, os programas do GIGSTR foram escritos na linguagem C (turbo C), enquanto os programas referentes a comunicação foram escritos em linguagem de montador.

O processamento do GIGSTR foi dividido em três processos, na tentativa de explorar os poucos recursos de concorrência e prioridade do sistema operacional MS-DOS. O

primeiro processo esta associado à porta serial COM1, com a função mais prioritária de comunicação com o computador hospedeiro. O segundo processo será associado à porta serial COM2, para que seja feito o controle do dispositivo ratinho; e o terceiro processo controlará a IHM e as atualizações em tela, não estando associado a nenhuma porta serial e com isso terá a menor prioridade de execução. A seguir faremos uma descrição superficial destes 3 processos.

VI.1.1. Processo "Comunicação na Estação Gráfica"

O processo de comunicação com o computador hospedeiro foi desenvolvido em linguagem montador. O protocolo de comunicação definido (para comunicação serial (RS232)), não utiliza os padrões de controle de comunicação, tais como "XON" e "XOFF", que garantem que toda informação foi recebida. Com o objetivo de minimizar o tempo de transmissão o protocolo de comunicação está baseado no fato do processamento hospedeiro informar a quantidade (em 2 "bytes") de dados a serem transmitidos para cada "buffer" e a deteção de perda de informações será realizada através do balanço entre a quantidade de informações enviada e a quantidade recebida.

O processo de comunicação mantém 4 áreas partilhadas com o processo de controle de IHM e atualizações em tela. A primeira área é o "buffer" contendo as informações recebidas do processamento hospedeiro. A segunda área é um semáforo que indica que chegaram dados do processamento hospedeiro e o "buffer" está cheio. A terceira área é um semáforo indicando que está havendo transmissão e o "buffer", contendo informações da transmissão anterior, ainda não foi processado, caracterizando um "over run" no sistema. A quarta área é um semáforo indicando o início de transmissão de dados.

VI.1.2. Processo "Controle do Ratinho"

O controle do ratinho é executado através do

"software" da Logitech, mas controlado pelo processo de controle de IHM e atualização de telas. O GIGSTR controla as ações do ratinho, tais como: liga e desliga, área de atuação na tela, leitura da posição e qual botão está apertado e etc., utilizando o "software" básico que atua diretamente na porta serial (COM2).

VI.1.3. Processo "Controle de IHM e Atualizações em Tela"

O processo de controle de IHM e atualizações em tela engloba logicamente todos os processos constantes na Figura VI.1, a menos dos processos de comunicação (comunicação com o computador hospedeiro e controle do ratinho).

O processo em questão apresenta três funções básicas. A primeira função é o controle de execução das rotinas. A segunda função é o controle de navegação da interface homem-máquina. A terceira e última função é a atualização de dados em telas, compostas por funções gráficas dinâmicas.

VI.1.3.1. Controle de Execução

O controle de execução das rotinas que compõem o processo de controle de IHM e atualizações em tela, tem a função de distribuir as tarefas de uma forma cíclica, de acordo com a ocorrência dos eventos, tais como: comunicação, teclado, atualização e etc. Os eventos são controlados por meio de indicadores objetivando simular um ambiente concorrente que utiliza semáforos para sincronizar o processamento. A seguir apresentamos uma definição dos indicadores e um pseudo-código do processo de controle de execução baseado na Figura V.9 para o ambiente MS-DOS são:

Os indicadores que compõem o controle de

execução são:

- EF1: indica que chegaram dados de comunicação e o "buffer" já está cheio.

- EF2: indica que o ratinho se movimentou ou um dos botões foram apertados.

- EF3: indica que o teclado foi acionado.

- EF4: indica que o "buffer" de comunicação já foi trocado pelo "buffer" de comunicação auxiliar. Permitindo a troca do "buffer" de comunicação pelo "buffer" de atualização.

- EF5: indica que houve "over run" no sistema.

- EF6: indica "lock" no "buffer" de atualização.

- EF7: indica que uma tela (cardápio, apoio, comando e etc.) está superposta sobre uma tela de atualização. Devido ao conceito de tela ativa toda ação de entrada ou atualização de tela será sobre a mesma, não permitindo uma atualização de tela, caso uma outra esteja superposta.

- EF8: indica início de transmissão de dados do computador hospedeiro para a estação gráfica. Este indicador é utilizado para desligar o ratinho (temporariamente) durante uma transmissão; com isso o processo de comunicação ganha mais prioridade de execução.

- EF9: indica fim de atualização.

Pseudo-código do Controle de Execução

Enquanto não é fim de processamento faça:

```
Se houver alteração do ratinho
então
        liga Ef2-ratinho
fim se

Se teclado ativado
então
        liga EF3-teclado
fim se

Se EF8-início-transmissão ligado
então
        desabilita ratinho
        desliga EF8-início-transmissão
fim se

Se EF5-over-run ligado
então
        envia mensagem
        desliga EF5-over-run
        despreza "buffer" de comunicação
fim se

Se EF1-comunicação ligado e ( ( EF4-troca comunicação
desligado) ou (EF7 ligado ) )
então
        habilita ratinho
        desliga EF1-comunicação
        troca "buffer" de comunicação pelo "buffer"
        de comunicação auxiliar
        Liga EF4-troca-atualização
```

```
fim se

Se EF4-troca-comunicação ligado e
EF6-lock-buffer-atualização desligado
então
            troca "buffer" auxiliar pelo "buffer" de
            atualização
            desliga EF4-troca-comunicação
            liga EF6-lock-buffer-atualização
fim se

Se Ef6-lock-buffer-atualização ligado e
EF9-fim-atualização desligado
então
            desliga    EF6-lock-buffer-atualização
            desliga EF9-fim-atualização
fim se

Se EF6-lock-buffer-atualização ligado e EF7 desligado
então
            atualiza tela com dados enviados pelo
            hospedeiro (controle-atualiza)
fim se

Se EF2-ratinho ligado
então
            atualiza tela ativa (controle-evento (ratinho))
fim se

Se EF3-teclado ligado
então
```

```
se evento é tecla dedicada
então
     atualiza tela ativa (controle-evento
     (tecla-dedicada) )
 senão
     atualiza tela ativa (controle-evento
     (teclado) )
fim se
fim se
fim enquanto.
```

Os indicadores EF9-fim-atualização e EF7 serão ligadas, de acordo com os eventos de fim de atualização e ativação de uma tela sobre a tela atualizável respectivamente.

VI.1.3.2. Controle de Navegação da Interface Homem-Máquina

O controle de navegação da interface homem-máquina tem a função de manipular todas as telas do sistema, onde cada operação de interação homem-máquina é tratada como uma tela, tais como: cardápio, mensagem de erro, descrição de apoio, comando, botão em tela, controle de acesso (senha), confirmação e preenchimento de campos para atuação sobre o banco de dados da aplicacao.

Em geral o projeto da interface homem-máquina é representado por um diagrama de transição (grafo). A figura VI.1 (parte 1 e 2) apresenta o diagrama de transição, não em forma de grafo, representado pelos possíveis eventos gerados através do ratinho ou teclado sobre uma função (tela).

O processo de controle de navegação também tem a função de enviar dados da estação gráfica para o computador hospedeiro quando da exibição de uma tela atualizável, ou quando de uma atualização no banco de dados da aplicação.

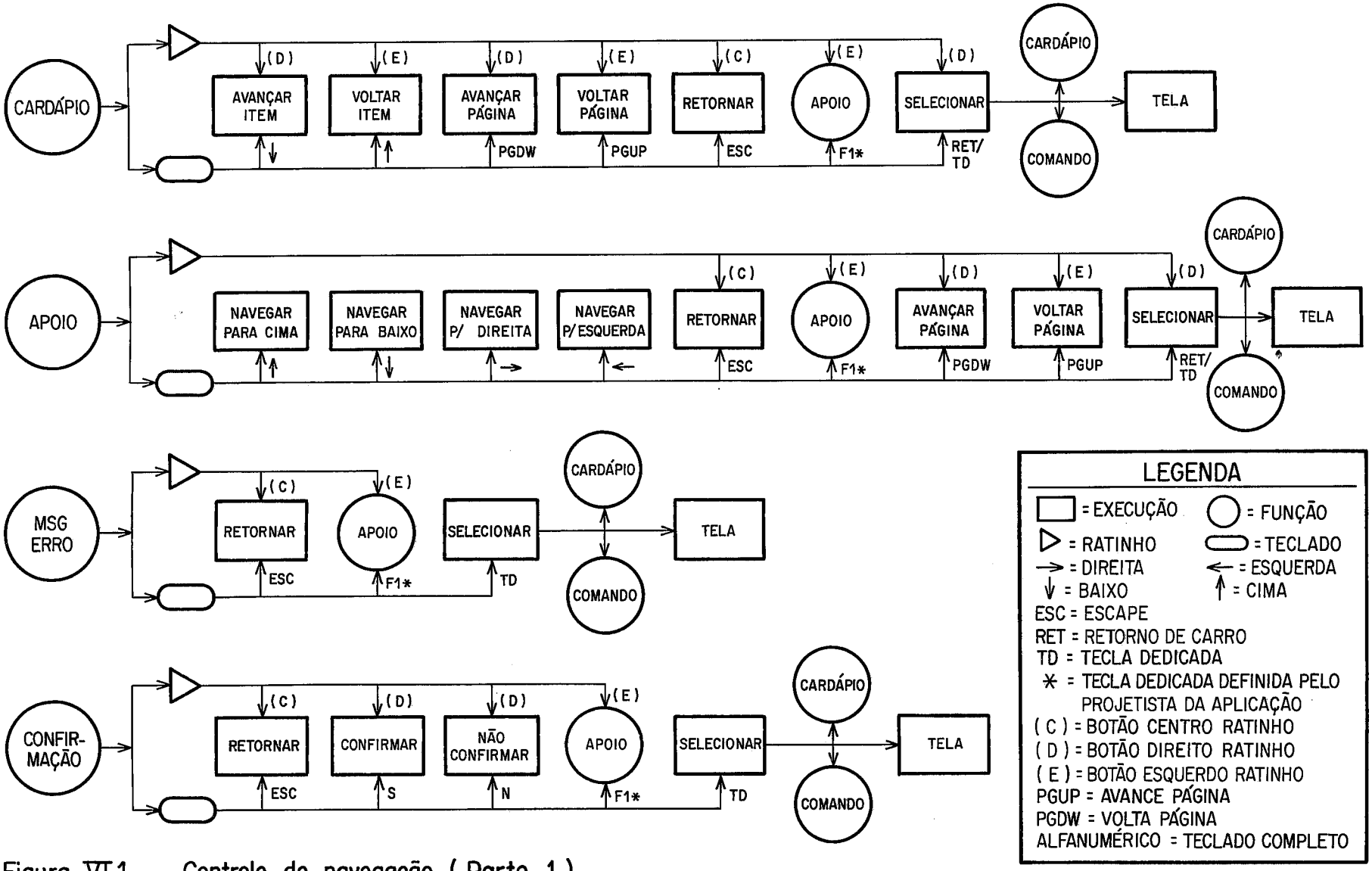

Figura VI.1 - Controle de navegação (Parte 1).

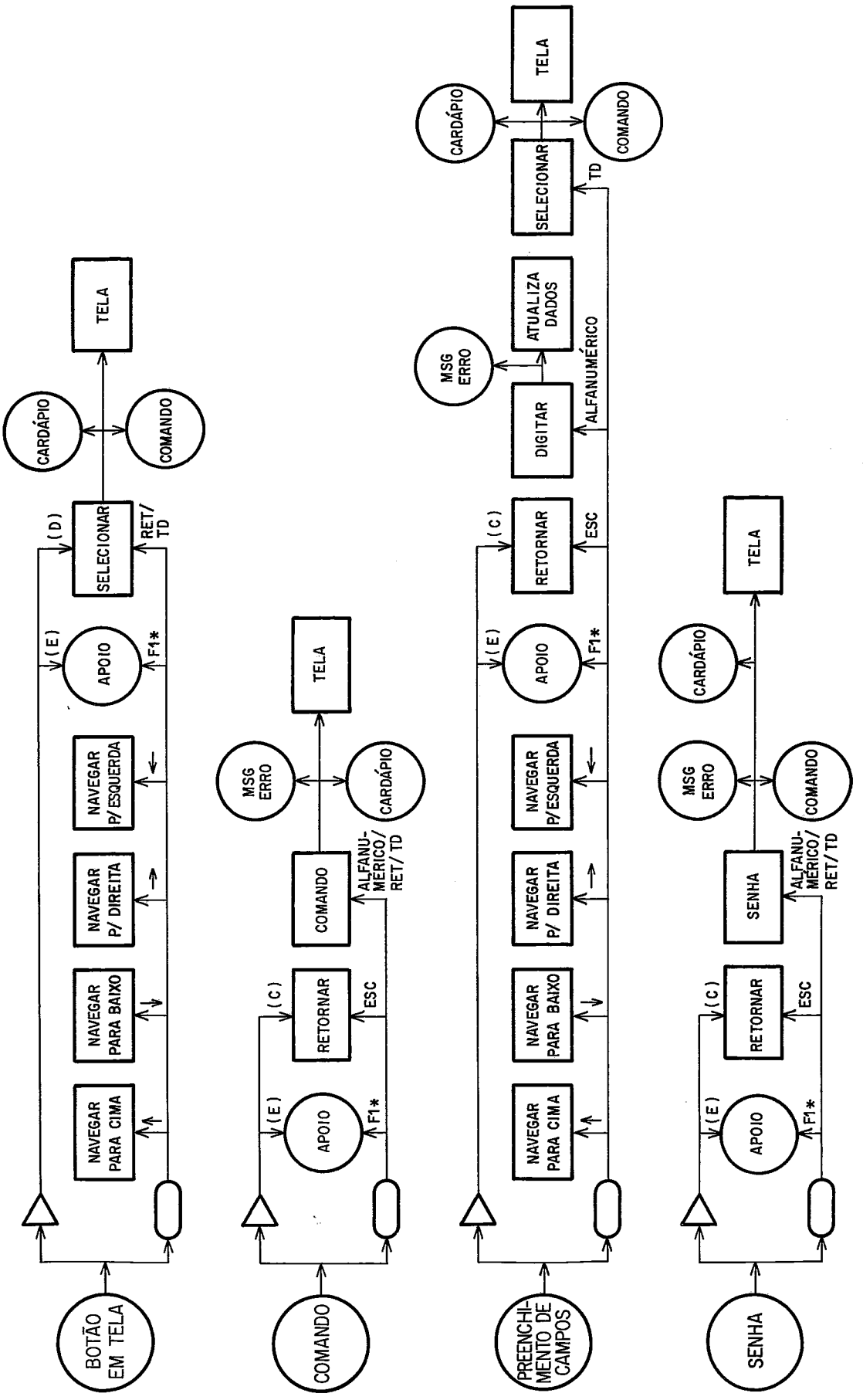

Figura VI.1 - Controle de navegação (Parte 2).

VI.1.3.3. Atualização de Dados em Tela

É importante realçar que o processo de atualização de dados em tela compreende a atualização das funções dinâmicas (barra, gráfico, histórico, etc.) das telas atualizáveis, deixando para o controle de navegação as funções de exibição das telas e controle de navegação entre as telas.

O processo de atualização de dados em tela processa periodicamente uma função dinâmica, utilizando o "buffer" de atualização, que contém dados vindos do computador hospedeiro.

O "buffer" de atualização não tem um tamanho padrão de preenchimento, devido ao fato de que somente as variáveis (do banco de dados da aplicação) que sofreram alterações, entre a última aquisição e a aquisição corrente, serão transmitidas. Deste modo objetiva-se economizar tempo de transmissão e de atualização em tela de uma mesma informação já atualizada.

O processo de atualização de dados em tela só efetua atualizações de dados em tela, quando a mesma for a tela ativa, por exemplo, se um cardápio sobrepõe-se a tela atualizável, o mesmo passa a ser a tela ativa, congelando as atualizações da tela atualizável, até a remoção do cardápio. Esta restrição é função da falta de recursos do MS-DOS para o processamento concorrente, não devendo existir quando da implementação em ambientes do tipo UNIX.

VI.1.4. Encapsulamento das Funções Gráficas

Objetivando o desenvolvimento de um Gerenciador de Interfaces Gráficas para processamento de tempo real portátil de um ambiente computacional para outro, no que se refere a funções gráficas, o GIGSTR utiliza um conjunto mínimo de rotinas gráficas, e a partir das mesmas amplia as

necessidades gráficas. O GIGSTR tem a função de identificar a placa gráfica e adequar o seu processamento a mesma, permitindo que o projetista da aplicação defina a interação homem-máquina e as funções de monitoração e atuação sobre o processo, sem se preocupar com o tipo de placa gráfica que o seu ambiente computacional contém.

As rotinas gráficas definidas no turbo C reconhecem diversas placas gráficas, tais como: EGA, CGA, VGA e etc., mas através do encapsulamento das rotinas gráficas inclusões, tais como, da placa gráfica DATANAV, são viáveis através da substituição da biblioteca gráfica do turbo C, pela biblioteca gráfica da DATANAV.

VI.2. *IMPLEMENTAÇÃO DO GIGSTR NO COMPUTADOR HOSPEDEIRO*

A implementação do GIGSTR no tocante ao modo de processamento hospedeiro foi realizada em um MICROVAX II com 5Mb de memória e 2 discos com capacidade de armazenamento de 70Mb cada. O sistema operacional do Microvax II utillizado foi o VAX/VMS, sistema este que possui todas as ferramentas necessárias para a implementação de aplicações de tempo real.

O processamento implementado no computador hospedeiro foi dividido em quatro processos, onde três destes processos pertencem ao GIGSTR, que são: processo recebe comunicação da estação gráfica; processo atualiza "buffer" de comunicação, com dados do banco de dados da aplicação e o processo que envia o "buffer" de comunicação à estação gráfica. O quarto processo implementado é o processo supervisor, pertencente a aplicação, que foi desenvolvido para simular um processamento de tempo real, afim de testar as funções do GIGSTR.

VI.2.1. Processo "Recebe Comunicação"

O processo "recebe comunicação" após a sua

instalação fica em "wait mailbox" aguardando dados da estação gráfica. Ao receber um "buffer" de comunicação o processo passa o mesmo para o processo supervisor através de "mailbox" e retorna ao estado de "wait mailbox".

VI.2.2. Processo "Atualiza Buffer de Comunicação"

O processo atualiza "buffer" de comunicação é ativado pelo supervisor da aplicação, para a preparação do "buffer" contendo os dados a serem atualizados na estação gráfica. Através dos dados de tela e atualizações gerados pelo editor gráfico do GIGSTR, e descritos no Anexo I, o processo prepara o "buffer" de comunicação, com dados do banco de dados da aplicação, e posteriormente envia através de "mailbox" ao processo supervisor.

Os dados constantes do "buffer" são formados para cada função dinâmica (barra, texto, gráfico, etc.), onde cada função é uma rotina que compõe uma parte do "buffer". Por exemplo, os dados que compõem a função de barra, são: valor atual e côr da variável.

VI.2.3. Processo "Envia Buffer à Estação Gráfica"

O processo envia "buffer" à estação gráfica, recebe do processo supervisor o "buffer" de comunicação, através de "mailbox", e o transmite para a estação gráfica.

VI.2.4. "Processo Supervisor"

O processo supervisor foi desenvolvido para simular o processamento da aplicação, que adquire os sinais do campo, em cadência de 2 segundos, e os armazena no banco de

dados da aplicação, com o objetivo de testar a comunicação entre o computador hospedeiro e a estação gráfica.

CAPITULO VII

CONCLUSÕES

Os resultados obtidos através da implementação do protótipo do GIGSTR nos levam a concluir que:

- A implementação do GIGSTR em um ambiente computacional do tipo PC/AT ou PC/386 é viável para sistemas de tempo real, que objetivam monitorar e controlar processos, as quais não requisitam baixos tempos de respostas. O protótipo do GIGSTR utilizando um microcomputador PC/AT como estação gráfica, um Microvax II como hospedeiro e uma linha serial com velocidade de 19.200 BAUDs, é capaz de atualizar uma tela, que contenha 40 funções dinâmicas (barras, textos, gráficos e etc.), na ordem de 2 segundos.

- Mesmo o protótipo sendo implementado no sistema operacional MS-DOS, utilizando os conceitos e técnicas, objeto deste trabalho, verificou-se um bom desempenho em relação ao tempo de processamento, mas que uma futura implementação em ambiente operacional concorrente do tipo do UNIX será de real utilidade para tempos críticos.

- A implementação do Editor Gráfico para a preparação dos dados a serem utilizados no processamento de tempo real do GIGSTR, é parte indispensável do produto "executável" como um todo. O Editor Gráfico não foi implementado no protótipo do GIGSTR, o que acarretou em um grande esforço na preparação dos dados de teste, devido a necessidade de cada função a ser testada no protótipo, principalmente no controle sobre os "pixels" das funções de representação de dados e na preparação da base de dados de tela, que contém itens de seleção (botões em tela).

Após uma análise, sob o ponto de vista da

interação homem-máquina, dos "softwares" mais frequentemente comercializados dentro da área de supervisão e controle de processo, tais como: MAX-SD da ELEBRA, AUTOMIC II da PHT, A500 da WEG, SESC 2023 da Eurocontrol, EPY100 da Ecil/P&D entre outros, notamos que no geral estes produtos foram desenvolvidos para atender a operação de processos industriais genéricos. A característica de definição do aplicativo em função do "preenchimento de campos" dos parâmetros da instalação dispensa praticamente a necessidade do projetista de sistemas, mas por outro lado impõe rigidez na definição dos ambientes aplicativos e interativos, uma vez que os mesmos são pré-definidos pelo produto.

As definições aplicativas padronizadas e os conceitos interativos baseados em teclas dedicadas ou comandos usados por esses produtos fazem com que os mesmos atendam a um perfil do operador representativo do mercado de processos industriais, onde o sistema é considerado um integrador das informações analógicas dos painéis de operação, não exigindo-se uma profundidade nas funções de diagnóstico, gerência e otimização da operação. Acreditamos que o aspecto relevante do trabalho apresentado é justamente a tentativa de definir uma conceituação de interação homem-máquina gráfica, representada pelo GIGSTR, que visa a atender aos sistemas de tempo real que exigem normalmente soluções específicas, e uma grande participação do projetista de sistemas em função da complexidade do processo industrial ou tecnológico. Principalmente em virtude que o objetivo destes sistemas não é apenas monitorar ou controlar o processo, mas também exercer um papel de gerenciamento sobre a planta, diagnosticando e otimizando a sua operação.

A arquitetura do GIGSTR baseada em computador hospedeiro e estação gráfica foi definida de modo a atender aos sistemas de tempo real que necessitam de grande capacidade computacional, geralmente suprido por computadores de médio e grande porte, e onde o usuário possui um perfil sofisticado de operação, necessitando portanto de ferramentas interativas com o ambiente computacional que possibilitem uma eficiente

navegação entre as telas aplicativas, de modo a suprir o nível de detalhamento adequado à complexidade do processo físico envolvido.

Podemos citar como exemplo de um gerenciador de interações homem-máquina gráfico para aplicações de supervisão e controle em tempo real, de processos complexos, o produto PROVUE comercializado pela Digital. Este produto foi desenvolvido para operar especificamente em ambientes de tempo real, em uma arquitetura com um VAX sendo hospedeiro e estações gráficas baseadas em PDP11/73, ligadas em rede ETHERNET.

O PROVUE foi utilizado no sistema de monitoração dos parâmetros de segurança da usina nuclear de Angra I desenvolvido pela COPPE/UFRJ. Tivemos a oportunidade de observar que este produto, apesar da eficiente performance para a operação em tempo real exigiu um grande esforço no desenvolvimento do aplicativo, de modo a suprir as deficiências intrínsicas ao produto no tocante a interação homem-máquina. As deficiências apresentadas pelo PROVUE residem basicamente na falta de ferramentas de "alto nível" (janelamento, hipertexto, etc.) que permitam a utilização de conceitos atuais de interação homem-máquina, além das estruturas de teclas dedicadas e comandos. Obrigando portanto que grande parte do ambiente interativo, tivesse de ser definido a nível do aplicativo afim de suprir funções que naturalmente deveriam ser do gerenciador.

O protótipo do GIGSTR desenvolvido neste trabalho foi em grande parte definido como decorrência das necessidades geradas pela experiência adquirida na implantação, e na utilização, de sistemas de tempo real de grande porte, como o da usina nuclear de Angra I, e portanto acreditamos que os conceitos e mecanismos de interação homem-máquina gráficos aqui apresentados possam ser de utilidade para futuros empreendimentos na área deste tipo de ambiente de tempo real.

REFERÊNCIAS BIBLIOGRÁFICAS

[1] ALLWORTH, S.T., Introduction to Real-Time Software Design, The MacMillan Press Ltd., London, 1982.

[2] GLASS, R.L., Real-Time Software, Prentice-Hall Inc., New Jersey, 1983

[3] PRESSMAN, R.S., Software Engineering, MacGraw Hill Book co., 1987.

[4] GREEN, M., "Report on Dialogue Specification Tools", Department of Computing Science University of Aberta, Edmonton, Alberta, Canada, 1984.

[5] FREY, P.R. e SIDES, W.H.JR., "Computer-Generated Display System Guidelines", EPRI-Electric Power Research Institute, Volume 1, September, 1984.

[6] SWEZEY, R.W. e DAVIS, E.G., "A Case Study of Human-Factors Guidelines in Computer Graphics", IEEE CG&A, November, 1983.

[7] SCHIRRU, R., MARTINEZ, A.S. e THOME,Z.D., "A new concept of safety parameter display system", memoria del seminario sobre energia nuclear en america latina, American Nuclear Society, section latino americano, Mexico, 1986.

[8] GREEN, M., "The University of Alberta user Interface Management System", SIGGRAPH, Vol. 19, Number 3, 1985.

[9] SIBERT, J.L., HURLEY, W.D. e BLESER, T.W., "An Object-Oriented user Interface Management System", SIGGRAPH, Volume 20, Number 4, 1986.

[10] MAGUIRE, M.C., "A Review of Human Factors Guidelines and Thecniques for the Design of Graphical Human-Computer Interfaces", Computer & Graphics, Volume 9, 1985.

[11] THIMBLEBY, H., "Failure in the Technical user Interface Design Process", Computer & Graphics, Volume 9, 1985.

[12] KENNETH, N.L., "Iconic Interfacing", IEEE-CG&A, March, 1983.

[13] BRITTS, S., "Dialog Management in Interactive Systems: A Comparative Survey", University of Strockholm, SYSLAB report, N° 45, October, 1986.

[14] STRUBBE, H.J., "Reporton Role, Model, Structure and Construction of a IMS", Prime Computer Inc., 500 Old Connecticut Path, Framinghan, Ma 01701, USA, 1984.

[15] MONTE, L.C.M., "Sistemas de Interface com o Usuário", Programa de Engenharia de Sistemas e Computação, COPPE/UFRJ, Novembro 1987.

[16] THOMAS, J.J. e HAMLIN, G., "Graphical Input Interaction Technique (GIIT)", Computer Graphics, January, 1983.

[17] BRAESICKE, C., "User Interface", ACM SIGSOFT, Software Engineering Notes, Volume 10, N° 2, April 1985.

[18] FOLEY, J.D., WALLACE, V.L. e CHAN, P., "The Human Factors of Computer Graphics Interaction Thecniques", IEEE CG&A, November, 1984.

[19] GOLDBERG, A., "Smalltalk-80: The Interactive Programming Environment", Addison Wesler, 1984.

[20] GANE, C. e SARSON, T., Analise Estruturada de Sistemas, Livros Técnicos e Científicos Ed. SA, Rio de Janeiro,

la.Edição, 1983.

[21] SCHIRRU, R., MARTINEZ, A.S., THOMÉ, Z.D. e BELLAS, N.R., "Sistema de Monitoração de Parâmetros de Segurança da Usina Nuclear de Angra-I", Programa de Engenharia Nuclear COPPE/UFRJ, Laboratório de Monitoração de Processos, Outubro 1989.

[22] PETERSON, J.L., PETRI Net Theory and Modeling of Systems, Prentice-Hall, Inc., Englewood Cliffs, N.J. 07632.

A N E X O I

DESCRIÇÃO DOS DADOS

Conforme referido na Seção V.3, apresentamos aqui um resumo dos depósitos de dados, relativo ao Diagrama de Fluxo de Dados (DFDs) das Figuras V.2 e V.8.

DEPÓSITOS DE DADOS

Modêlo Sintático

- < nome do depósito . ,(referência aos DFDs) >
  conteúdo: < estrutura de dados incluídas >

- Evento (REF.D1)
  conteúdo: Ratinho, Teclado.

- Telas (REF.D2)
  conteúdo: Telas.

- Seleções (REF.D3)
  conteúdo: Seleção

- BFF-Envia (REF.D4)
  conteúdo: Comunicação.

- Telas Ativas (REF.D5)
  conteúdo: Telas Ativas.

- BFF-Comunicação (REF.D6)
  conteúdo: Comunicação.

- BFF-COM-AUX (REF.D7)
  conteúdo: Comunicação.

- BFF-Atualização (REF.D8)
  conteúdo: Comunicação.

- Atualizações (REF.D9)

  conteúdo: Atualizações, Barra, Gráfico, Tendência, Histórico, Símbolo, Textos.

-Tela Atualizável (REF.D10)

  conteúdo: Tela Atualizável

- Semáforos (REF.D11)

  conteúdo: Semáforos

- Banco de Dados do Processo (REF.D12)

  conteúdo: Atributos da Variável

Estrutura de Dados

  Modelo Sintático

  - < Nome da Estrutura >

  - < Descrição Resumida >

  Componentes: < Estruturas e/ou Elementos de Dados Constituintes >

- Evento

  Ações produzidas pelo Ratinho ou Teclado.

  Componentes: Ratinho (Posição de X, Posição de Y, Botão), Teclado (Código da Tecla).

- Telas

  Contém os elementos que definem as telas.

  Componentes: Código da tela, Tipo da tela, XI-borda, YI-borda, XF-borda, YF-borda, Código da tecla dedicada, QTD-seleções, Índice de seleções, QTD-atualização, Índice de atualização, Nome do arquivo.

- Seleção

  Contém os elementos que definem um item de seleção, para cardápios, Botões em tela, descrições de apoio, Confirmação, Tela base e Tela atualizável.

  Componentes: XI-borda, YI-borda, XF-borda, YF-borda,

Índice da tela (seleção), Índice da tela (apoio), Tipo de seleção, Direita, Esquerda, Cima, Baixo.

- Comunicação

  Área temporária para receber e enviar dados entre o computador hospedeiro e a estação gráfica. Os componentes variam em função da comunicação.

- Telas Ativas

  Estrutura de Auxílio à execução, indicando a tela que está sendo exibida e está com o controle dos eventos produzidos pelo ratinho e teclado.

  Componente: Código da tela.

- Atualizações

  Estrutura que mantém a ligação entre a tela e as funções de atualização.

  Componentes: Tipo da função, Índice da função.

- Barra

  Contém os elementos que definem a função de barra.

  Componentes: Código da função, XI-borda, YI-borda, XF-borda, YF-borda, Direção, Escala, Limite superior, Limite inferior, Estilo da barra, Estilo de fundo da barra, Côr de fundo, Índice da Variável.

- Gráfico

  Contém os elementos que definem a função de gráfico.

  Componentes: XI-borda, YI-borda, XF-borda, YF-borda, Direção, Escala (X), Escala (Y), Limite superior (X), Limite inferior (X), Limite superior (Y), Limite inferior (Y), Tipo-cursor, Rastro, Côr do cursor, Côr do rastro, Índice da variável (X), Índice da variável (Y), Atributo (X), Atributo (Y).

- Texto

  Contém os elementos que definem a função de texto.

  Componentes: X, Y, Tipo de caracter, Direção, Formato, Índice da variável, Atributo, tamanho do caracter.

-Histórico

Contém os elementos que definem a função de histórico.

Componentes: XI-borda, YI-borda, XF-borda, YF-borda, Intervalo, Direção, Hora de início, Hora de fim, Função, Côr, Limite superior, Limite inferior, Escala, Estilo de preenchimento, Índice da variável.

-Tendência

Contém os elementos que definem um gráfico de tendência.

Componentes: XI-borda, YI-borda, XF-borda, YF-borda, Intervalo, Direção, Função, Côr, Limite superior, Limite inferior, Escala, Estilo de preenchimento, Índice da variável.

- Símbolo

Contém os elementos que definem a função de símbolo.

Componentes: Código do símbolo.

- Atributos da Variável

Contém os atributos das variáveis do banco de dados do processo. Cada aplicação tem um conjunto de atributos, portanto citaremos alguns a título de ilustração.

Conteúdo: Índice da variável, Valor atual, Banda morta, Descrição, Tipo da variável, Unidade de Engenharia, "Status", Limite superior, Limite inferior.

- Tela Atualizável

Área de armazenamento temporária contendo o código da tela que está sendo atualizada.

Conteúdo: Código da tela.

- Semáforo

Lista de indicadores para controle de execução.

Conteúdo: Semáforos (n).

Elementos de Dados

Modelo Sintático

- < Nome do Elemento >
  < Descrição Resumida >

- Atributo

  Código que identifica os elementos de dados pertencentes às variáveis do banco de dados da aplicação.

- Baixo

  Tecla que indica o sentido de descer o cursor.

- Banda Morta

  Variação não representativa de um valor.

- Botão

  Botão do ratinho.

- Cima

  Tecla que indica o sentido de subir o cursor.

- Código do Símbolo

  Identificador utilizado para representar figuras.

- Código da Tecla

  Identificador da Tecla acionada pelo usuário.

- Código da Tela

  Identificador das telas do sistema.

- Código de Tecla Dedicada

  Identificador utilizado para associar uma tecla a uma tela.

- Côr

  Auto explicativa.

- Descrição

  Descrição da variável.

- Direita

  Tecla que indica o sentido de mover o cursor para a direita.

- Direção

  Sentido de preenchimento de uma função, podendo ser:

horizontal da direita ou esquerda e vertical de cima ou de baixo.

- Escala

  Tipo de escala, linear ou logarítmica.

- Esquerda

  Tecla indicando o sentido de mover o cursor para a esquerda.

- Estilo

  Estilo de preenchimento de uma área na tela.

- Formato

  O formato é utilizado na representação de textos, podendo ser real (Fnn.nn), inteiro (Inn) ou caracter (Annn).

- Função

  Identifica a função da tela de interação (cardápio, apoio, comando e etc.).

- Hora

  Hora de início e fim , para determinar o intervalo de tempo, o qual o histórico será representado.

- Índice de Atualização

  Identificador da função de atualização da tela.

- Índice da Função

  Identificador de função de atualização.

- Índice de Seleção

  Identificador dos itens selecionáveis da tela.

- Índice da Variável

  Identificador da variável do banco de dados da aplicação.

- Limite

  Limites de escala da variável.

- Nome do Arquivo

  Nome do arquivo de tela ("RASTER FILE").

- QTD-Seleção

  Quantidade de itens de seleção na tela.

- QTD-Atualização

  Quantidade de atualizações (dinâmicas) na tela.

- Rastro

  Identifica se um gráfico vai ou não deixar rastro por onde o cursos irá passar.

- "Status"

  Identifica se uma variável do banco de dados da aplicação, está dentro dos limites de escala.

- Semáforos

  Sinalizador utilizado para controle de execução dos processos.

- Tamanho do caracter

  Auto explicativa.

- Tipo de caracter

  Identifica o tipo de caracter a ser utilizado na função de textos (Padrão, Romano , Itálico e etc.).

- Tipo do cursor

  Identificador do tipo de cursor a ser utilizado pela função de gráfico.

- Tipo de função

  Identificador das funções de atualização (barra, gráfico, hitórico e etc.).

- Tipo de seleção

  As seleções podem ser para uma tela ou, em caso de um cardápio, para paginação.

- Tipo da tela

  Idenfitica o tipo da tela (cardápio , descrição de apoio, controle de acesso (senha), botão e etc.).

- Tipo da Variável

  Identifica se a variável é digital, analógica ou calculada.

- Unidade de engenharia

  É um atributo da variável do banco de dados da aplicação, que identifica a unidade de engenharia da variável.

- Valor atual

  É um atributo da variável do banco de dados da aplicação, que representa o valor da última aquisição de dados.