TÍTULO: Metodologia para Desenvolvimento de Software em Sistemas Distribuídos

AUTORES: Antonio Luiz Bogado Fernandes
Maurício Moszkowicz
Lucia Della Valle

ENTIDADE: CEPEL - Centro de Pesquisas de Energia Elétrica
Departamento de Eletrônica DPET)
Cidade Universitária - Ilha do Fundão
C. Postal 2754
21.941-Rio de Janeiro-RJ
Tel.: 270-0112 R. 143

RESUMO

O CEPEL vem desenvolvendo um sistema de supervisão para sistemas elétricos que incorpora características peculiares tal como arquitetura com multimicroprocessadores organizados segundo um barramento serial.

Tendo em vista esta restrição e a grande complexidade do software envolvido, apresentamos uma metodologia ao desenvolvimento do software que tem entre outros, os seguintes macro-objetivos:

- Redução do custo de desenvolvimento e manutenção do software;
- Tornar mais observável (aumentar a capacidade de administração) o desenvolvimento do projeto, criando um meio de comunicação fracamente conexo entre as pessoas que participam do desenvolvimento.

A metodologia divide o desenvolvimento do software em seis fase distintas:

- Análise dos requisitos: apresenta um método para a definição das propriedades gerais do sistema, verificando os contornos do projeto, analisando o contexto e as funções principais;
- Especificação: apresenta métodos de decomposição das funções em objetos, define as interfaces e descreve os efeitos de cada função;
- Projeto: define a alocação dos objetos no sistema, fornece um plano de implementação para os objetos, refina suas estruturas de controle descrevendo as suas entradas/saídas/efeitos.
- Implementação: consiste na elaboração dos algoritmos, rotinas e programas de uma forma geral;
- Teste: observa o comportamento do software implementado e examina a sua disponibilidade;
- Operação: é a entrega do produto final ao usuário, podendo envolver uma atualização/manutenção do sistema.

Como conclusão do trabalho serão apresentados os resultados obtidos pela aplicação da metodologia a partes do sistema de supervisão ora em desenvolvimento no CEPEL.

Palavra Chave: Engenharia de Software e Sistemas Distribuídos

METODOLOGIA PARA DESENVOLVIMENTO DE SOFTWARE EM SISTEMAS DISTRIBUÍDOS

1. INTRODUÇÃO

O CEPEL vem desenvolvendo um sistema de supervisão para sistemas elétricos para ser utilizado nos centros de despacho regional de concessionárias de energia elétrica. Os despachos regionais têm como encargo a coordenação das manobras, o controle de tensão e a normalização após perturbações de uma área pré-determinada do sistema elétrico de uma empresa.

Para fazer face às necessidades especificadas por FURNAS Centrais Elétricas S.A. para seus centros de supervisão regional o CEPEL propôs a esta empresa o desenvolvimento de um centro de arquitetura modular à base de microprocessadores. Esta solução permitiu um pleno domínio da tecnologia de projeto de centros de supervisão dentro da filosofia do processamento distribuído. As vantagens desta estratégia são óbvias:

- o usuário, no caso, FURNAS possui total conhecimento do sistema em todos os seus aspectos de hardware e software podendo portanto moldá-lo às suas necessidades;
- a estrutura modular, de grande flexibilidade, permite, quando necessária, a introdução de novas funções no sistema e assegura uma longa vida útil ao mesmo devido às facilidades de expansão.

O sistema protótipo está em fase final de implementação e já se tem uma versão em operação no centro regional de FURNAS de Jacarepaguá. A transferência da tecnologia para a indústria também já foi iniciada. A primeira das duas indústrias que fabricarão o sistema já iniciou o processo de absorção da tecnologia e a segunda será escolhida até o final de 1982.

A Figura 1 mostra o esquema de um centro de supervisão dentro da concepção modular proposta.

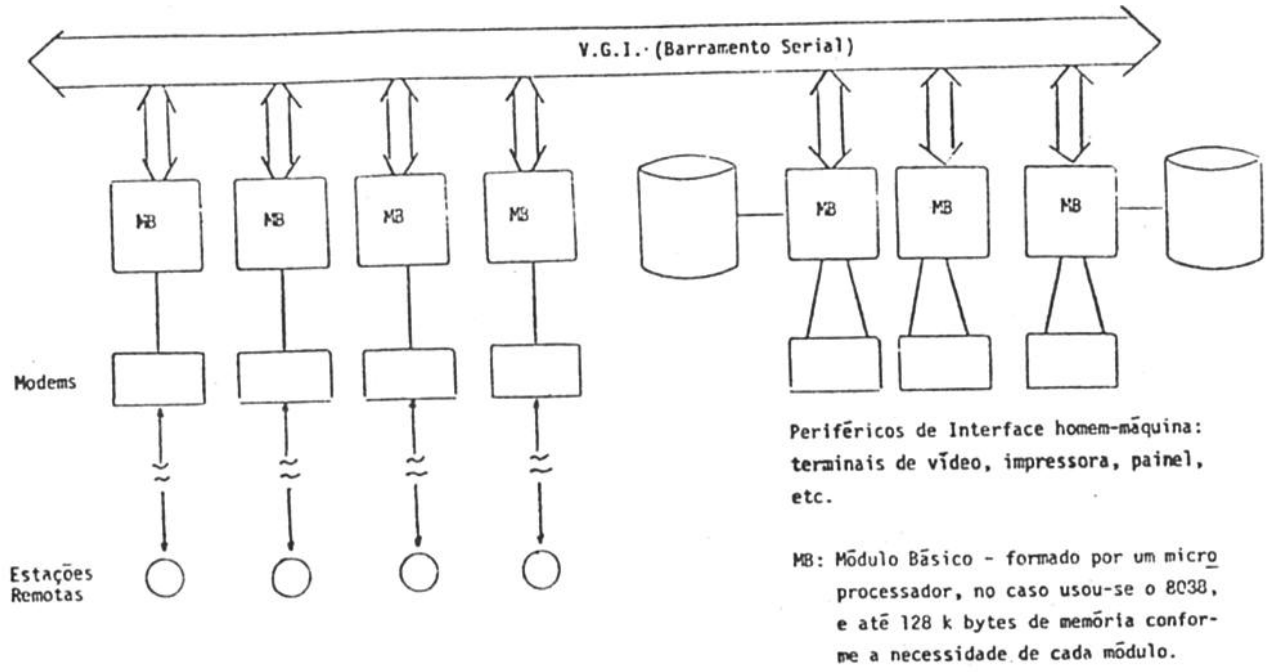


Figura 1

O projeto do sistema básico foi conduzido em paralelo com o do sistema de aplicação, o que permitiu o "overlapping" de etapas, facilitado pela arquitetura modular. Por outro lado, a constante interação entre os requisitos da aplicação e as capacitações alcançáveis pelo sistema básico permitiram definir-se um casamento ótimo que atendesse aos diversos condicionamentos econômicos e cronológicos.

A complexidade do projeto, que ao ser concluído terá durado cerca de 6 anos e envolvido recursos humanos da ordem de 72 engenheiros-mês e materiais da ordem de 1 milhão de dólares, merece toda uma sistemática de desenvolvimento. Esta sistemática que, a bem da verdade, está sendo consolidada "on the job" é hoje um dos principais sub-produtos da realização do trabalho. Nas páginas a seguir procura-se descrever os aspectos mais relevantes da metodologia utilizada.

2. METODOLOGIA GERAL

As motivações para a adoção de uma metodologia definida em projetos de engenharia de software são bem conhecidas: custos crescentes para desenvolvimento do software, idem para sua manutenção, a tradicional dificuldade de se obter boa documentação, problemas decorrentes da alta coesão normalmente encontrada entre os participantes do projeto, baixa visibilidade pela gerência do andamento das diversas frentes de projeto, etc.

Inúmeros métodos de trabalho vêm sendo propostos para fazer face à esses problemas. Os enfoques variam nos detalhes de implementação mas há uma certa unanimidade no tocante às fases básicas do desenvolvimento: estabelecimento dos requisitos, análise dos requisitos, busca de soluções, especificação da solução, projeto, implementação, testes e manutenção. Uma documentação paralela a todas estas etapas também é uma necessidade de reconhecimento geral.

O trabalho que vimos realizando está seguindo esta cronologia básica como não poderia deixar de ser. Sentiu-se entretanto, por diversas vezes, necessidade de melhor caracterizar as diversas etapas e, principalmente, ao tratarmos de um sistema que já tem uma estrutura frouxamente conexa (vide barramento serial comum na figura 1) tirar proveito da mesma para criar ferramentas de desenvolvimento que favorecessem ao máximo o gerenciamento das configurações do projeto.

As seções a seguir descrevem cada uma das etapas seguidas, dando para cada uma: a caracterização que nos pareceu a mais correta, exemplos das atividades do projeto do centro de supervisão realizadas dentro dela e as ferramentas criadas para o trabalho com uma arquitetura do tipo distribuída.

3. DEFINIÇÃO DOS REQUISITOS

O trabalho de definição dos requisitos foi conduzido basicamente por FURNAS, com a participação do CEPEL. Seu resultado consta de extenso documento designado CEP-170 e que rege os termos da concorrência aberta por FURNAS para escolha do fabricante de seus centros de supervisão regional.

Os requisitos estabelecem as diversas funções do software de aplicação, a configuração dos equipamentos periféricos, tempos de resposta, índices de desempenho, ocupação, confiabilidade, reparabilidade, etc.

Por carência de espaço não nos cabe aqui fazer esta descrição. Ao longo das seções seguintes serão feitas referências aos requisitos envolvidos nos exemplos abordados.

4. ANÁLISE DE REQUISITOS

Esta fase dos trabalhos consiste em um processo de submeter cada um dos requisitos de perfomance (oriundos do usuário) a uma espécie de crivo formado por um outro conjunto de requisitos, alguns também oriundos do usuário e outros da experiência da engenharia. A Figura 2 ilustra este procedimento.

As recomendações produzidas por este processo vão constituir o quadro de restrições a que a fase seguinte - a Busca da Solução - deverá obedecer ao procurar a solução ótima. Ou seja, para cada requisito funcional r_{fj} será obtido um conjunto de recomendações $\{r_{1j} \ldots r_{\ell j}\}$ a partir de um mapeamento dos conjuntos $\{r_{f1} \ldots r_{fp}\}$, $\{r_{o1} \ldots r_{ok}\}$ e $\{r_{ei} \ldots r_{em}\}$.

Vejamos agora como podem ser caracterizados os conjuntos acima, como o processo de crivagem se realiza e, usando a aplicação mencionada, mostrar um exemplo deste tipo de trabalho.

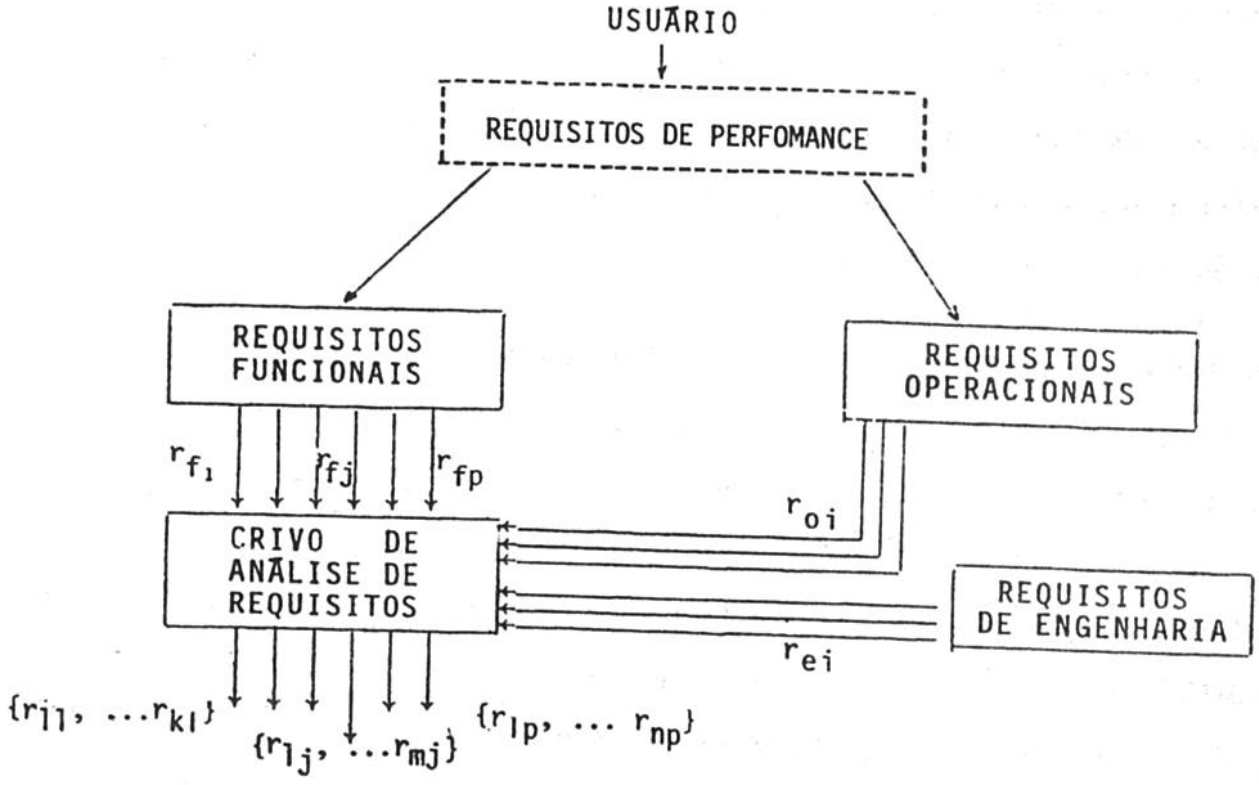


Figura 2

Em primeiro lugar é necessário separar dentro dos requisitos definidos pelo usuário quais os requisitos funcionais e quais os operacionais. Os primeiros são normalmente mais explícitos e de fácil caracterização. São considerados funcionais porque podem ser descritos pelo conceito matemático de função: espaço de partida, espaço de chegada e mapeamento do primeiro no segundo. Englobam: variáveis a tratar, tipos de tratamento, taxas de tratamento, formas de exteriorização dos resultados, parâmetros , formas de modificação dos parâmetros, prioridades para tratamento e exteriorização, etc.

Os requisitos operacionais são mais imprecisos e muitas vezes são omitidos pelo usuário já que não constam das suas preocupações básicas que são cumprir os objetivos últimos do sistema. A forma de descrição destes requisitos não é padronizada como a dos funcionais pela diversidade de noções que envolvem. Alguns exemplos destes requisitos são: índices de disponibilidade do fluxo de informações, índices de observabilidade, índices de recuperabilidade, critérios para expandibilidade quantitativa , critérios para expandibilidade funcional, linguagens para comunicação homem-máquina (ergonometria), etc.

Finalmente, os requisitos de engenharia são aqueles ditados pelos recursos tecnológicos disponíveis, funções do custo admitido para o projeto , maturidade industrial, etc, adicionados da chamada "experiência" adquirida em projetos anteriores. Salvo os dados técnicos dos elementos de hardware, os requisitos de engenharia também fazem parte de uma coleção de difícil definição pela sua heterogeneidade. Por exemplo: custos admissíveis, recursos humanos disponíveis, conceitos arquitetônicos tais como: multiprocessamento, programação estruturada, programação concorrente, sistemas operacionais, bancos de dados, objetos, monitores, etc.

O processo de crivagem será examinado a seguir tomando por base alguns elementos do sistema de supervisão. Cabe porém, a ressalva de que as recomendações produzidas pela Análise de Requisitos são por vezes conflitantes e caberá à etapa seguinte (Busca de Soluções) administrar estes possíveis conflitos.

Alguns dos principais requisitos funcionais do Centro de Supervisão são:

1 - Recolher dados de unidades terminais remotas;

2 - Tratá-los (filtrá-los, convertê-los);

3 - Verificar condições de alarmes;

4 - Enviar estes dados e condições a diversos dispositivos de exteriorização;

5 - Conservar uma imagem permanentemente atualizada dos dados;

6 - Responder a consultas sobre esta imagem;

7 - Enviar dados/comandos indicados pelo operador.

Alguns requisitos operacionais básicos são:

1 - Conservar um índice de disponibilidade do fluxo de dados superior a 99,8%;

2 - Permitir a observação de condições de erro com uma cobertura superior a 95% e com um retardo inferior a 5 segundos;

3 - Recuperar a plena capacidade de operação na presença de falhas isoladas em menos de 1 minuto e de falhas duplas em menos de 1 hora;

4 - O sistema deverá poder atender a um volume de dados variáveis numa escala de 1:3;

5 - Não está prevista expandibilidade funcional, além das apresentadas na figura 3, devendo-se contudo poder admitir um acréscimo da complexibilidade sub-funcional numa escala de 3:4;

6 - A linguagem de comunicação homem-máquina terá sua semântica definida a priori devendo ser possível aumentar o seu vocabulário numa escala de 1:2.

Quanto aos requisitos de engenharia, no exemplo considerado, os principais se referem à escolha de uma arquitetura do tipo distribuído.

A realização completa do processo de crivagem é bastante trabalhosa e, a nosso ver, maiores desenvolvimentos precisam ser feitos nesta área para aumentar a eficiência deste tipo de atividade.

Consideremos por exemplo o requisito funcional da detecção de condições de alarmes e sua notificação ao homem através de diversos dispositivos de exteriorização (CRT's, impressoras, painéis mímicos, etc.). Obtivemos as seguintes recomendações:

1 - As detecções de alarmes deverão ser feitas nos próprios módulos (tanto de hardware quanto de software) em que ocorrerem ou onde foram primeiramente observados.

2 - O alarme detectado deve ser codificado também a nível do módulo detector.

3 - Após a codificação o aviso de alarme deve ser expedido:

   i) a um coordenador central para inserção do mesmo em uma lista ordenada por prioridade. Como esta lista deve ser triplicada (por questões de segurança), surge a necessidade de um coordenador que garanta a homogeneidade das três listas.

   ii) aos diversos dispositivos de exteriorização escalados para a emissão de alarmes (CRT's, painel mímico e "logger" de alarmes).

Como se pode observar, no caso do requisito em questão, as recomendações obtidas encaminham as soluções a serem buscadas sob uma ótica descentralizante visando a máxima modularidade e confiabilidade.

Procedendo-se desta forma para todo o conjunto de requisitos funcionais obteve-se um documento de natureza conceitual, não muito extenso, mas de profundo significado para a realização da etapa seguinte.

Outro subproduto desta fase é a obtenção de um primeiro nível de particionamento funcional onde são discriminadas funções básicas e as funções de aplicação (figura 3).

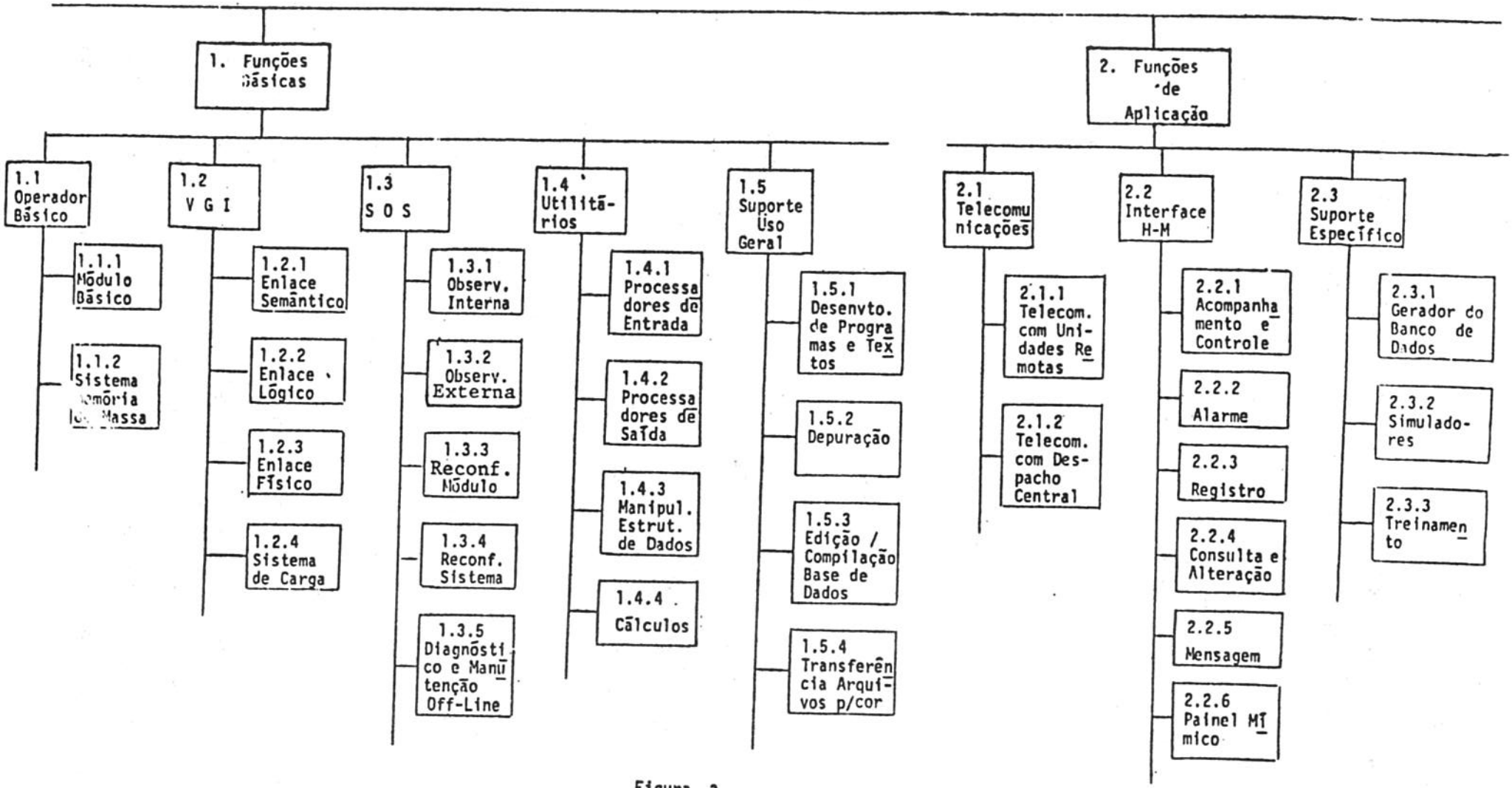


Figura 3

SOS.: Sistema de Observação e Salvamento

VGI.: Via Geral de Interconexão

5. BUSCA DE SOLUÇÕES

Esta é a etapa onde se obtém as soluções para o projeto do sistema. Os insumos são os requisitos e as recomendações para o sistema obtidos anteriormente.

O processo de obtenção das soluções é iterativo, onde um conjunto de possíveis soluções passa por uma cadeia de decisão cujos nodos são os requisitos ou as recomendações (figura 4)

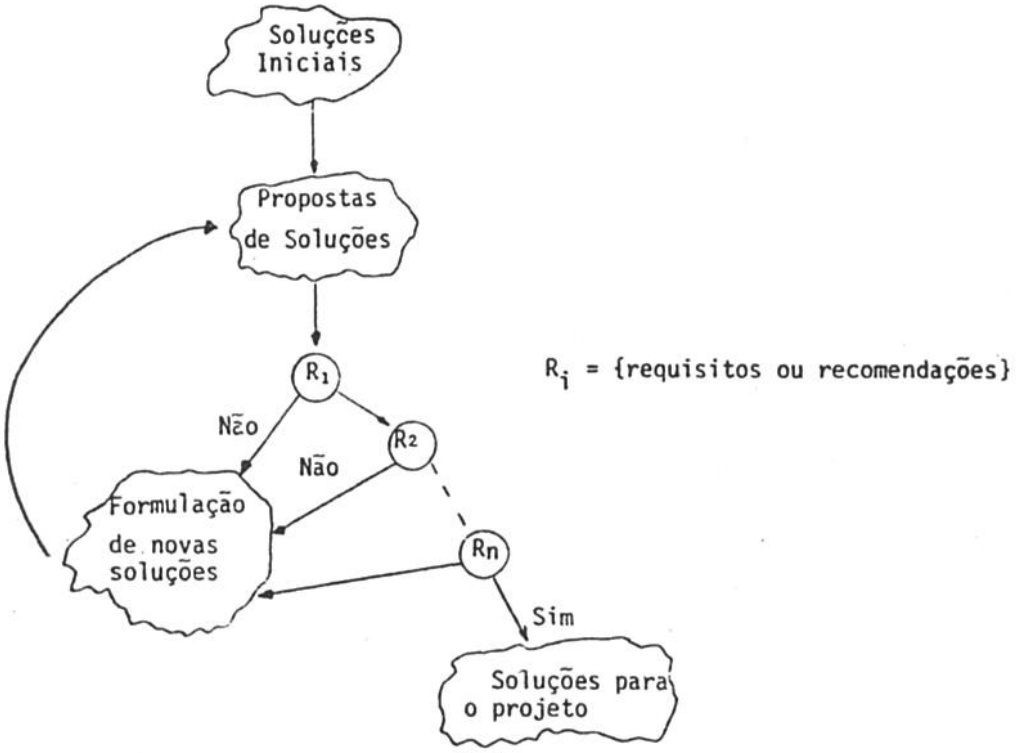


Figura 4

Para o processo de busca de soluções foi adotada uma linguagem de descrição que possui duas entidades notáveis:

i) objetos - são os blocos básicos da descrição. Um objeto representa um conjunto de dados e as operações (métodos) que podem ser executadas sobre estes dados, sendo somente acessível ao mundo externo através de mensagens. A utilização destes blocos básicos se justifica facilmente em sistemas distribuídos, dado o baixo grau de coe

são resultante. Este baixo grau é indispensável quando se deseja poder evoluir o sistema sem causar maiores abalos às partes já construídas. Uma das recomendações que nortearam o projeto foi a de se poder inserir ou extrair objetos sem a reescrita dos demais. Outra vantagem do uso de objetos é a de proporcionar um isolamento entre os diversos projetistas ou programadores, permitindo que estes conheçam o microcosmo do objeto e ~~não~~ o macrocosmo do sistema.

ii) Diagramas de sequência de ações ("Action Sequence Diagram"-ASD) . Os ASD's são compostos por objetos ligados de forma tal a representarem o desempenho de uma determinada função.

A simbologia adotada nessa primeira proposta foi a seguinte:

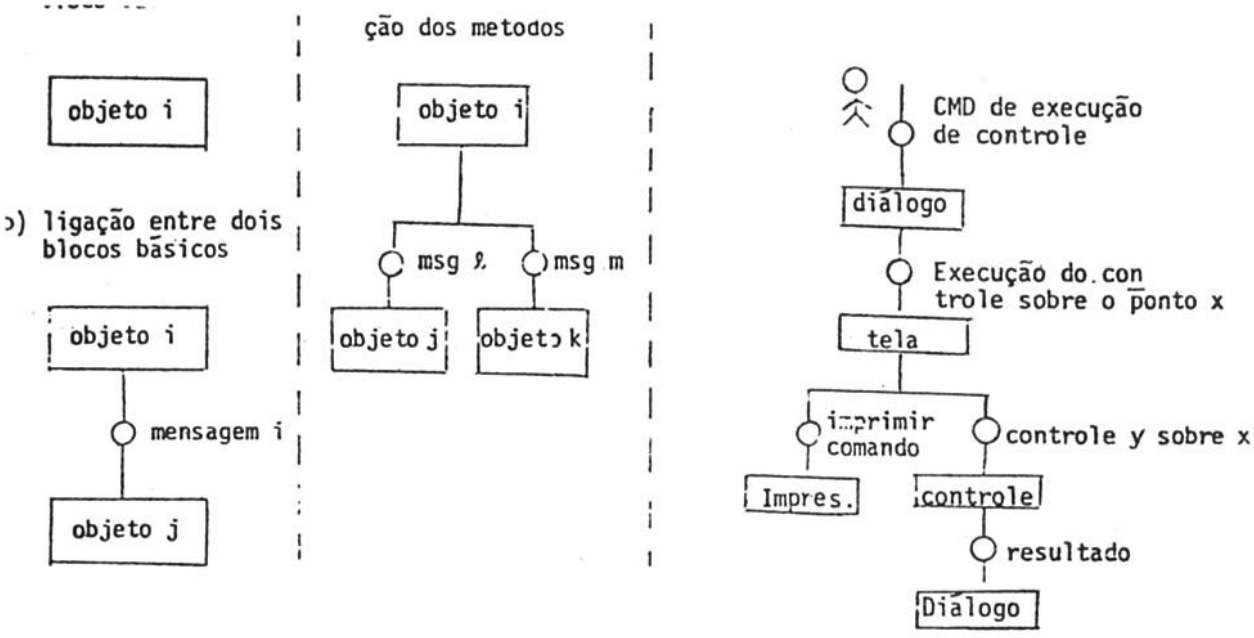


Figura 5

5.1 - Individualização dos Objetos do Sistema

Para completar a linguagem adotada para a descrição das soluções , fica apenas faltando como se determinam os objetos do sistema. O método utilizado consistiu em se fazer um mapeamento de todas as operações necessárias à realização das funções (passo 1). Associa-se a cada operação o tipo de dado que deve operar (passo 2). Grupa-se as operações segundo categorias de abstração comuns (passo 3) obtendo-se, então, os objetos (Figura 6).

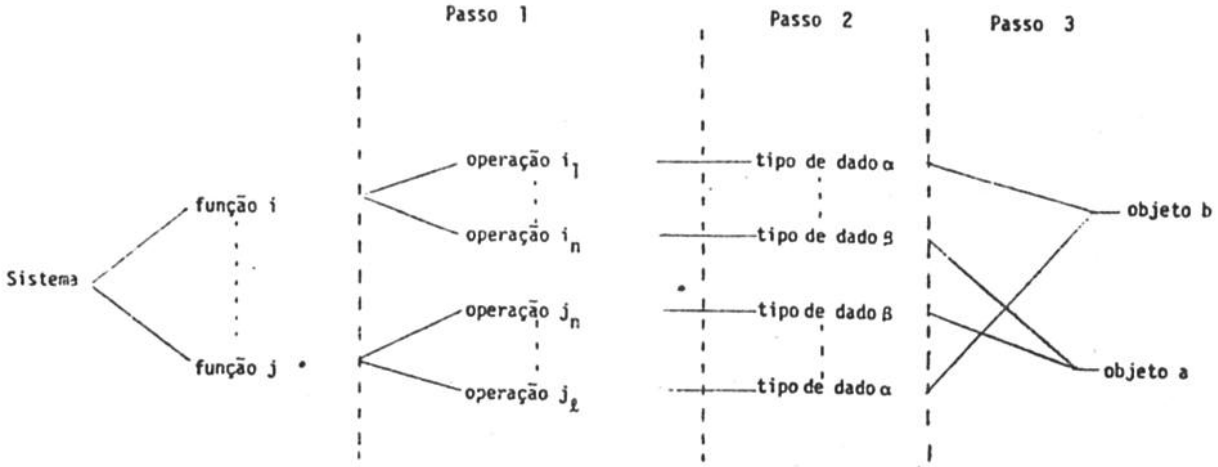


Figura 6

A partir da decomposição de funções como as da Figura 3, realizada de acordo com os requisitos, as novas funções (subfunções) foram descritas através de um processo iterativo de definição dos ASD's e objetos respectivos.

As figuras 7, 8 e 9 descrevem um exemplo de resultado da aplicação deste processo às funções de Acompanhamento e Controle.

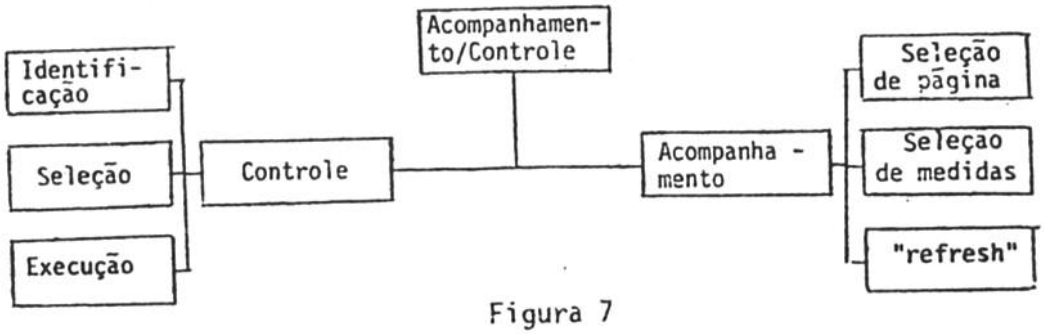


Figura 7

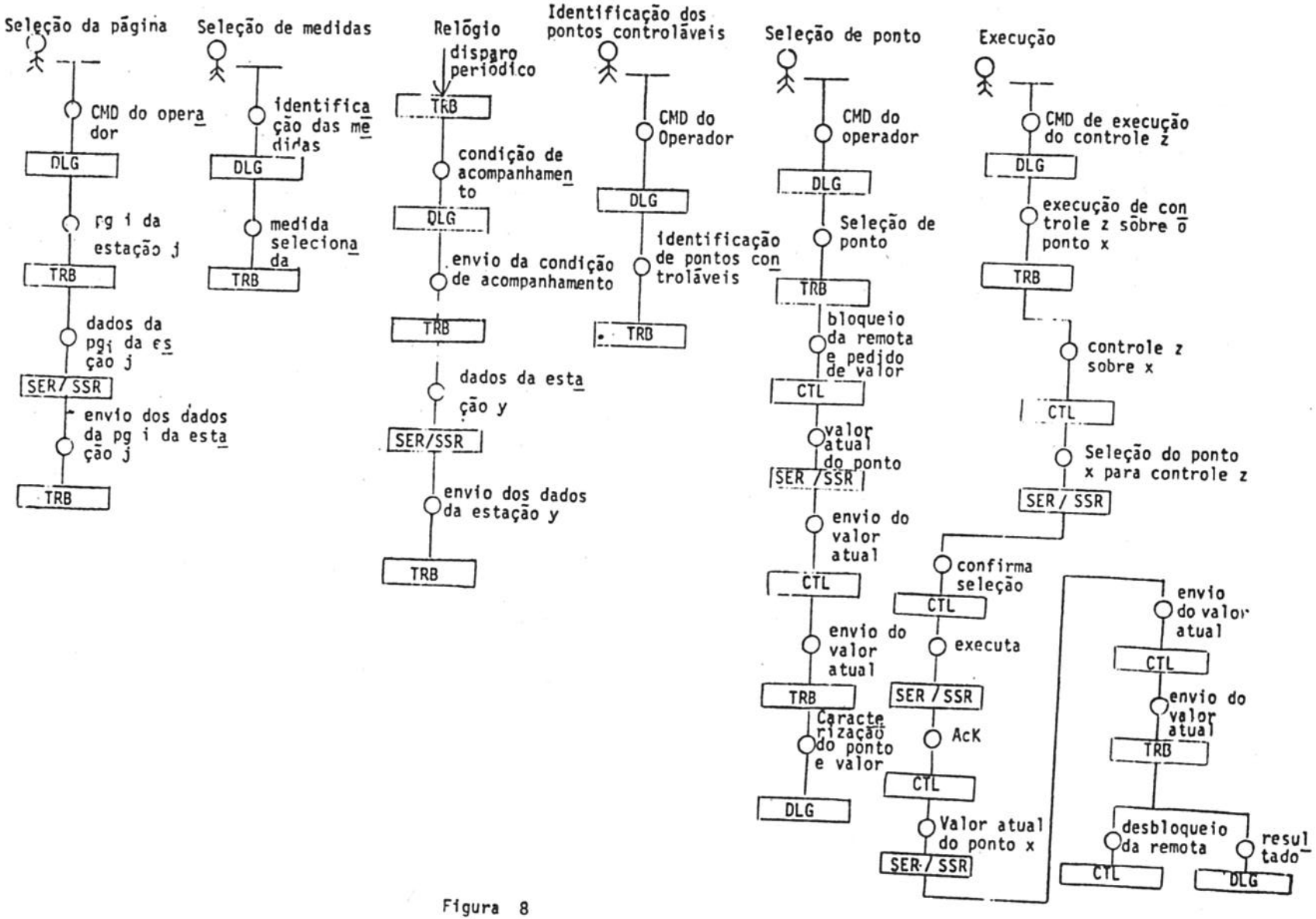

Seleção da página
CMD do operador
DLG
pg i da estação j
TRB
dados da pg i da estação j
SER/SSR
envio dos dados da pg i da estação j
TRB
Seleção de medidas
identificação das medidas
DLG
medida selecionada
TRB
Relógio
disparo periódico
TRB
condição de acompanhamento
DLG
envio da condição de acompanhamento
TRB
dados da estação y
SER/SSR
envio dos dados da estação y
TRB
Identificação dos pontos controláveis
CMD do Operador
DLG
identificação de pontos controláveis
TRB
Seleção de ponto
CMD do operador
DLG
Seleção de ponto
TRB
bloqueio da remota e pedido de valor
CTL
valor atual do ponto
SER/SSR
envio do valor atual
CTL
envio do valor atual
TRB
Caracterização do ponto e valor
DLG
confirma seleção
CTL
executa
SER/SSR
AcK
CTL
Valor atual do ponto x
SER/SSR
Execução
CMD de execução do controle z
DLG
execução de controle z sobre o ponto x
TRB
controle z sobre x
CTL
Seleção do ponto x para controle z
SER/SSR
envio do valor atual
CTL
envio do valor atual
TRB
desbloqueio da remota
CTL
resultado
DLG

Figura 8

OBJETOS	OPERAÇÕES	DADOS
Diálogo	- Análise sintática e semântica dos comandos recebidos do despachante - Consulta e alteração do estado interno de diálogo	- Definição sintática da linguagem - Vocabulário de comandos - Estado interno de diálogo
Tela de Trabalho (TRB)	- Busca e exteriorização da página no vídeo - Alteração do estado de exteriorização da página	- Diretório das páginas - Conteúdo das páginas - Estado da exteriorização da página (inclusão ou retirada de dados da página).
Controle (CTL)	- Bloqueio e desbloqueio de uma estação para controle - Execução da lógica necessária à efetuação do controle	- Estado do controle - Dicionários de sequências de controle
SER/SSR	- Aquisição de dados do sistema elétrico e de supervisão - Envio de comandos ao sistema elétrico e de supervisão	- Estado do sistema elétrico e de supervisão

Figura 9

6. ESPECIFICAÇÃO DAS SOLUÇÕES DO PROJETO

Esta etapa apresenta a descrição das soluções que passaram pelo crivo da etapa de Busca de Soluções. Esta descrição apresenta minuciosamente as operações e os dados dos objetos dos ASD's escolhidos. Cada operação identificada deverá ser descrita por seu conjunto de entrada, de saída, seu efeito e condições de exceção (opcional), conforme mostra a tabela 1.

As descrições dos dados também deve ser feita cuidadosamente, identificando formatos, volumes, formas de acesso e sobretudo analisando possíveis inconsistências.

Tabela 1

Exemplo: ESPECIFICAÇÃO DA OPERAÇÃO 'SELEÇÃO DE PÁGINA'

ENTRADAS: Conjunto de comandos possíveis:

{ $S_i | S_i$ = sigla de estação, acesso direto a uma página definida, página anterior/seguinte, colocação/retirada dos nomes dos dispositivos no diagrama }

SAÍDAS: - pedidos de página à Tela de Trabalho
- envio de críticas ao despachante sobre comandos recebidos

EFEITOS: - fixação do status da estação sob acompanhamento, onde status = < nome da estação, página em exibição, medidas selecionadas, exibição ou não dos nomes dos dispositivos >
- estados de exceção

7. PROJETO

Esta etapa é responsável pela seguinte atividades:

i) Mapeamento dos objetos no sistema distribuído;

ii) Definição dos "chips de software" que implementem os objetos;

iii) Definição do sistema de interligação dos chips para a construção do sistema.

7.1 - Mapeamento dos objetos no sistema distribuído

O mapeamento dos objetos no sistema distribuído poderá ser feito "off-line", "on-line" ou de forma mista. No modo "off-line" cabe ao projetista determinar de forma rígida a localização dos objetos no sistema. Esta localização é fixa.

No modo "on-line", os objetos podem migrar no sistema de acordo com a disponibilidade dos recursos computacionais. Na forma mista, alguns objetos são localizados de forma "off-line" e outros, podem migrar no sistema.

Os principais critérios a serem obedecidos no mapeamento dos objetos são os seguintes:

i) indivisibilidade do objeto: este critério deve ser perseguido de forma tal a manter mínima a coesão entre os diversos componentes do sistema.

ii) atendimento às recomendações da análise de requisitos e disponibilidade de recursos computacionais: no caso em questão, existem operadores que possuem funções semelhantes no sistema tais como os operadores de comunicação com as estações remotas (vide figura 1). Esta replicação de operadores dará origem ao particionamento dos dados de um objeto, mantendo-se porém, os métodos replicados e independentes entre si.

7.2 - Definição dos chips de software que implementarão os objetos

Para o projeto dos objetos utilizamos uma disciplina rígida que resultou nos chamados chips de software.

Um chip é constituído, basicamente, das partes mostradas na Figura 10.

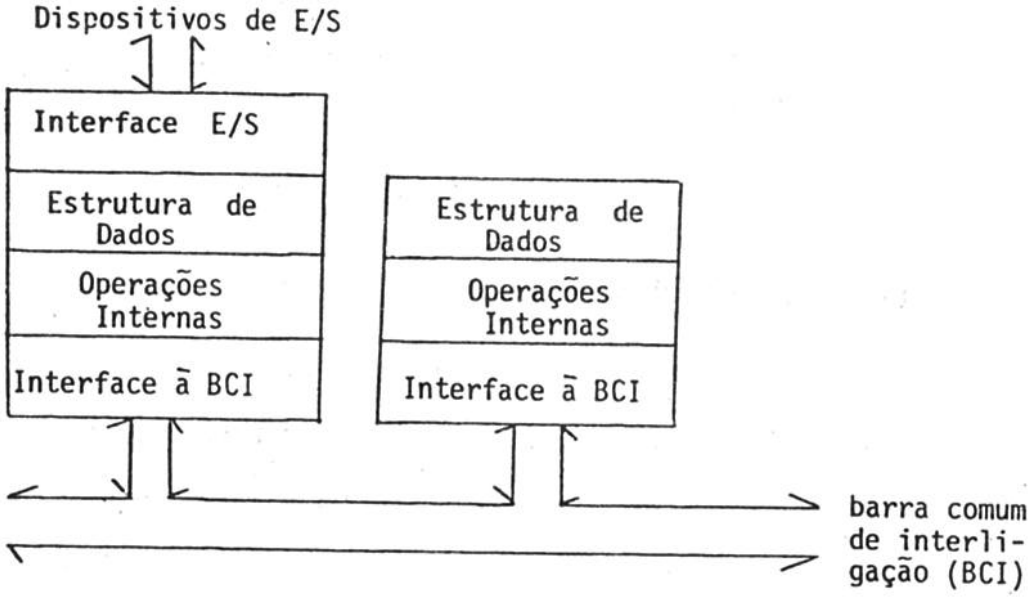


Figura 10

A filosofia subjacente à criação do conceito de chip foi a de se criar uma analogia entre o software e o hardware. Vê-se, no futuro, a existência de bibliotecas (manuais) que contenham os chips já implementados. Visualizamos que a fase de projeto deverá ser a da individualização dos chips para a realização do sistema, eventualmente criando-se um novo chip. A fase de implementação terá apenas de unir os chips de forma conveniente para a realização do sistema.

7.2.1 - Operações Internas

Representam o conjunto de procedimentos e funções necessários à execução dos métodos do chip. Internamente ao chip os métodos poderão ser executados de forma puramente combinacional (o conjunto de saída é função do de entrada), ou sequencial (o conjunto de saída é função do de entrada e do de estados internos ao chip). O projetista deverá utilizar estratégias de sincronismo interno que se fizerem necessárias à execução dos métodos. Por exemplo: um pedido de leitura pode ser inibido ou postergado se estiver em curso um processo de alteração de uma área de memória.

7.2.2 - Estrutura de Dados Interno

A organização dos dados internamente ao chip é totalmente livre, ficando esta tarefa a cargo do programador que irá implementar o chip. Estes dados não são acessíveis ao sistema, criando-se portanto, um escopo restrito ao chip.

7.2.3 - Interface à BCI

Esta é a única parte "visível" de um chip aos demais do sistema (ponto de observação do chip). Por apresentar esta propriedade, qualquer alteração nesta interface repercutirá nos demais chips.

Sua função é a de selecionar o conjunto de operações internas ao chip para a realização de um método. Por exemplo: o chip de comunicação com a estação remota ao receber um pedido de execução do método "varredura dos pontos analógicos" deverá executar as seguintes operações internas:

i - Montagem das mensagens a serem enviadas às remotas para obtenção dos seus dados analógicos;

ii - Comunicação com a remota para a obtenção de dados;

iii - Preparação do buffer de dados recebidos da remota.

7.2.4 - Interface com Dispositivos de E/S

Representa o conjunto de "handlers" que especializam o chip a um determinado tipo de dispositivo de E/S. Por exemplo: terminal de vídeo. A troca de um dispositivo por outro deve resultar apenas na alteração desta interface, mantendo-se intacto o resto do chip.

7.3 - Sistema de Interligação dos Chips para a Construção do Sistema

Idealmente, este sistema deve ser passivo. Na prática, porém, o que se observou foi que este sistema agregou todos os recursos comuns (utilitários) aos chips, tais como: sistema operacional, sistema de comunicação entre chips, etc.

OBS.: O sistema de comunicação entre os chips (enlace semântico) foi um utilitário concebido de forma tal a permitir com que o programador se abstraia da localização física de um chip. Caso os chips estejam localizados em operadores diversos, o enlace semântico funcionará como um adaptador de bus entre os operadores (Figura 11)

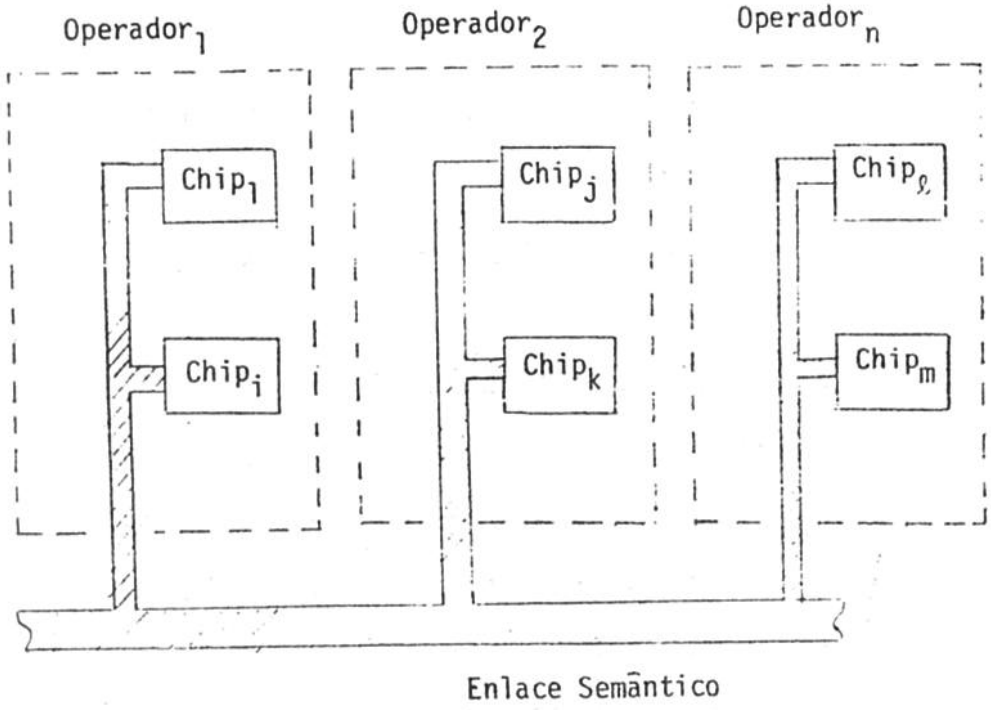


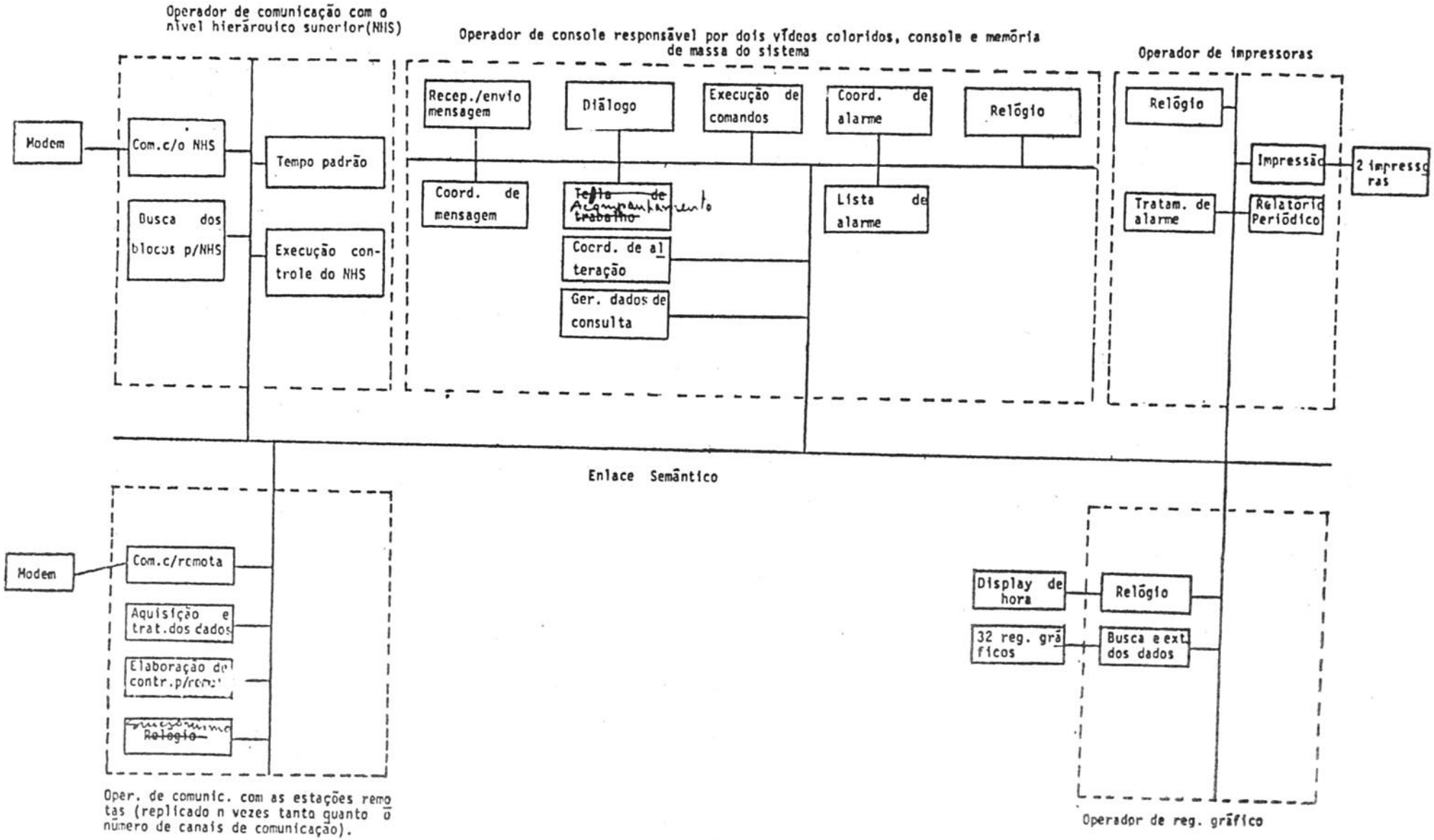
Figura 11

Na prática os utilitários são rotinas que se agregam ao chip, podendo desta forma estar presentes em todos os chips. Ao se agregarem, os utilitários passam a apresentar as características do chip, tal como a sua prioridade, stack, etc.

Fazendo-se uma analogia com o hardware, o sistema de interligação dos chips pode ser comparado ao conjunto placa de circuito impresso (elemento passivo) e fontes de alimentação (elemento ativo).

7.4 - Resultado da fase de projeto

A figura 12 apresenta um possível mapeamento dos objetos, a nível de chips, do sistema de supervisão.



EXEMPLO DE MAPEAMENTO DOS OBJETOS NO SISTEMA DE SUPERVISÃO

Figura 12

8. IMPLEMENTAÇÃO E TESTE

A implementação do sistema deve seguir um plano diretor elaborado na fase de projeto. Este plano estabelece o seguinte:

i - a nível de sistema: define camadas sucessívas de chips a serem implementadas/testadas para a obtenção das funções. Camadas sucessivas podem acrescentar mais chips ou métodos nos chips existentes.

ii - a nível de chip: estabelece a prioridade com que as diversas operações deverão ser implementadas tendo em vista os requisitos das diversas camadas do ítem i.

A técnica para a confecção deste plano diretor é a do tipo zig-zag (zig-zag approach), que combina os modos "top-down" e "bottom-up". Esta técnica se justifica em sistemas de controle de processos pela necessidade de se atender a requisitos de tempo de resposta restritos. Estes requisitos enfatizam a necessidade de consolidação das partes dos chips responsáveis pelas interfaces de entrada e saída.

Para a construção dos chips foram adotadas técnicas que visam auxiliar a administração da implementação, tais como:

i - criação de um módulo de descrição do chip ou utilitário. Este módulo deve conter as seguintes informações: título do módulo, data de entrega, descrição suscintados, arquivos de dados e programas que compõem o chip, procedures ou funções externas ao módulo referenciadas neste;

ii - descrição dos algoritmos das rotinas utilizando-se linguagem estruturada antes de se partir para a codificação. O código deverá possuir como comentário frases do algoritmo em linguagem estruturada (pretende-se garantir desta forma a estruturação do código).

9 - CONCLUSÃO

O trabalho aqui apresentado foi uma primeira iniciativa visando criar uma mentalidade de visualização do software como uma área de engenharia no âmbito do projeto de Centros de Supervisão. Deve-se notar porém, que diversas das técnicas mencionadas já vinham sendo adotadas por iniciativa pessoal. Nosso objetivo é sistematizar tais procedimentos.

Os resultados até agora obtidos já comprovam a validade desta iniciativa. Observa-se na equipe do projeto uma maior polarização em níveis de competência, resultando uma maior eficiência, melhor documentação das diversas fases e maior observabilidade.

Prevê-se como extensão deste trabalho as seguintes atividades:

- Adaptação das técnicas mencionadas a outros projetos de controle em tempo real;
- Aperfeiçoamento da metodologia proposta principalmente no que tange a: Sistemática de Análise de Requisitos, sistema convencional de definição de requisitos, linguagem para descrição dos ASD's e objetos, criação de bibliotecas de objetos.

Agradecimento:

Agradecemos a todos os participantes do projeto do Centro de Operação Regional de FURNAS pela sua ativa participação no desenvolvimento dos métodos expostos no artigo, e em especial, aos engenheiros Antonio Joaquim Setubal, Cesar A. Boavista, Homero G. Andrade, José A. Motta e Sergio P. Cotrim.

BIBLIOGRAFIA

(1) BERGLAND, G.D.; "A Guided Tour of Program Design Methodologies"; Computer, October 1981.

(2) BERSOFF, E.H.; HENDERSON, V.P.; SIEGEL, S.G.; "Software Configuration Management: An Investment in Product Integrity"; Prentice Hall, Englewood Cliffs, N.J., 1980.

(3) DALY, E.; "Management of Software Development"; IEEE Transactions on Software Engineering, May 1977.

(4) ENOS, J.C.; VAN Tilburg, R.L.; "Software Design"; Computer, February 1981.

(5) FERNANDES, A.L.B.; ANDRADE, H.G.; GRACIA, J.; MONTEIRO, M.C.B.; MOSZKOWICZ, M.; APPEL, O.; COSTA, R.S.; FREITAS, C.A.B.; MOTTA, J.A.P.; GARROFÉ, P.H.S.; COTRIM, S.P.R.; "Projeto de Centros de Supervisão de Sistemas de Energia Elétrica"; Anais 1º SICOP, Maio 1981.

(6) FURNAS - Centrais Elétricas S.A.; "Expansão do Sistema de Furnas, Sistemas de Supervisão e Controle - CEP 170", Expecificações Técnicas, junho 1981.

(7) LISKOV, B.; ZILLES, S.; "An Introduction to formal Specifications of Data Abstractions"; Current Trends in Programming Methodology, Prentice Hall Inc., 1977.

(8) ROSS, D.T.; SCHOMAN K.E.; "Structured Analisys for Requirements Definition"; IEEE, Transactions on Software Engineering, January 1977.

(9) XEROX LEARNING RESEARCH GROUP,"The Smalk Talk - 80 System"; Byte, August 1981.

(10) YOURDON, E. Constantine, L.; "Structured Design: Fundamentals of a Discipline of Computer Program and Systems Design"; Prentice Hall Inc., 1979.