UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

88/495

PROJETO DE SISTEMAS DISTRIBUÍDOS

por

SALVADOR ANTONIO DOS SANTOS

Dissertação submetida como requisito parcial para

a obtenção do grau de Mestre em

Ciência da Computação

Prof. J. Palazzo M. de Oliveira

orientador

Porto Alegre, agosto de 1987

CATALOGAÇÃO NA FONTE

Santos, Salvador Antonio dos

Projeto de Sistemas Distribuídos. Porto Alegre, PGCC da UFRGS, 1987.

222 p.

Diss. (mestr. ci. comp.) UFRGS-PGCC, Porto Alegre, BR-RS, 1987.

Dissertação: Sistemas distribuídos; Redes de computadores; Análise de sistemas; Métodos e tempos.

A meus pais e

à minha irmã Nadir

Processamento distribuido
Sistemas distribuidos

UFRGS
CPD - PGCC
BIBLIOTECA

Nº CHAMADA: 681.3) "71"(043)
S237p

Nº REG.: 3998
DATA: 25/11/88

ORIGEM: D
DATA: 26/7/88
PREÇO: Cz$ 800,00

FUNDO: PGCC
FUNC.: PGCC

AGRADECIMENTOS

Agradeço à CAPES - Coordenação de Aperfeiçoamento de Pessoal de Nível Superior, à UDESC - Universidade para o Desenvolvimento de Santa Catarina e ao Grupo Empresarial TUPY pelo suporte financeiro; ao CPD/UFRGS - Centro de Processamento de Dados da Universidade Federal do Rio Grande do Sul, à FEJ - Faculdade de Engenharia de Joinville, à IFT - Industria de Fundição Tupy Ltda. e à ETT - Escola Técnica Tupy por me franquearem suas instalações.

Agradeço aos colegas de trabalho que comigo cooperaram, ao Sr. K. Mokross pelo constante incentivo e ao Sr. Mário Karsten, a quem devo a oportunidade de realizar esta obra.

Ao CPGCC - Curso de Pós-Graduação em Ciência da Computação da UFRGS: agradeço aos funcionários e colegas pelo apoio e companheirismo e aos docentes pelos preciosos ensinamentos, em especial aos professores Maurízio Tazza e J. Palazzo M. de Oliveira pela orientação e encorajamento.

Agradeço aos meus familiares pelo apoio, de modo particular à minha querida esposa Sandra e aos nossos filhos Cristian e Alexandre, afetuosos companheiros de todas as horas, sem o estímulo dos quais seria impossível a elaboração deste trabalho.

Finalmente agradeço a Deus por ter me confortado e protegido durante os momentos que a Ele recorri.

SUMÁRIO

LISTA DE FIGURAS 13

RESUMO 17

ABSTRACT 19

1 INTRODUÇÃO 21

1.1 Objetivos 21

1.2 Organização do trabalho 24

2 SISTEMA DE COMPUTAÇÃO 27

2.1 Introdução 27

2.2 Configuração final 27

2.3 Configuração básica 29

2.4 Topologia 33

2.4.1 Tipos de topologia 34

2.5 Resumo 36

3 COMO PROJETAR O SISTEMA DE COMPUTAÇÃO 37

3.1 Introdução 37

3.2 Exemplo de aplicação 37

3.3 Projeto da topologia 39

3.3.1 Considerações sobre os elos locais 39

3.3.2 Dificuldades para projetar a topologia 42

3.3.3 Modelos de programação linear 44

3.3.3.1 Descrição do modelo 44

3.3.3.2 Cálculo da capacidade das linhas .47

3.3.3.3 Dimensões do problema 49

3.3.3.4 Vantagens e limitações 52

3.3.4 Métodos heurísticos 55

3.3.4.1 Exemplo de um método heurístico ..56
3.3.4.2 Limitações do método apresentado .58
3.3.5 Soluções pragmáticas 60
3.3.5.1 Topologia estrela 61
3.3.5.2 Topologia hierárquica 62
3.3.6 Combinação de métodos 65
3.4 Projeto da configuração básica 67
3.4.1 Simulação 71
3.4.1.1 Modelo de computador de grande porte 71
3.4.1.2 Como traçar o gráfico 72
3.4.1.3 Limitações da simulação 73
3.4.2 Modelo analítico 76
3.4.2.1 Tempo total de CPU 77
3.4.2.2 Fator de utilização da CPU 78
3.4.3 "Benchmark" 81
3.5 Projeto da configuração final 84
3.5.1 Comprovação da capacidade dos computadores 84
3.5.2 Comprovação da capacidade das linhas 85
3.5.2.1 Vazão líquida dos dados 87
3.5.3 Escolha dos meios de comunicação 90
3.6 Resumo 90

4 DESENVOLVIMENTO DA METODOLOGIA DE PROJETO ..... .93
4.1 Introdução 93
4.2 Ciclo de vida do projeto 96
4.2.1 Ciclo de vida estruturado 98
4.2.1.1 Terminadores 99
4.2.1.2 Atividades 101
4.3 Visão geral da metodologia 106

4.3.1 Contexto 109
4.3.1.1 Usuários 109
4.3.1.2 Gerência 110
4.3.1.3 Operações 111
4.3.1.4 Fabricantes de computadores 113
4.3.1.5 Fornecedores de meios de comunicação 114
4.3.2 Etapas 114
4.3.3 Processo de estruturação da metodologia ..117
4.3.3.1 Metodologia preliminar 118
4.3.3.2 Esboço da metodologia definitiva 121
4.3.3.2.1 Etapa 1 - desenvolvimento da aplicação 122
4.3.3.2.2 Etapa 2 - determinação da área geográfica 123
4.3.3.2.3 Etapa 3 - determinação da carga 125
4.3.3.2.4 Etapa 4 - dimensionamento dos recursos 126
4.3.3.2.5 Etapa 5 - instalação do sistema 128
4.4 Resumo 131
5 DESCRIÇÃO DAS ETAPAS DA METODOLOGIA 133
5.1 Introdução 133
5.2 Desenvolvimento da aplicação 133
5.2.1 Análise estruturada 134
5.2.2 Análise segundo Yourdon 138
5.2.2.1 Documentação do sistema físico corrente 139

5.2.2.2 Transformação no equivalente lógico 140
5.2.2.3 Definição do novo sistema lógico 140
5.2.2.4 Estabelecimento da fronteira homem/ máquina 141
5.2.2.5 Análise de custo/benefício 142
5.2.2.6 Seleção de opções 142
5.2.2.7 Estabelecimento de limitantes ...142
5.2.2.8 Especificação do sistema 143
5.2.3 Alterações na sub-atividade de análise ...143
5.2.3.1 Modelo homem/máquina 144
5.2.3.1.1 Diagrama de descrição do método 146
5.2.3.1.2 Descrição dos processos automatizados 148
5.2.3.2 A necessidade de um novo modelo .150
5.3 Determinação da área geográfica 156
5.4 Determinação da carga 159
5.4.1 Determinação da demanda dos insumos 160
5.4.2 Determinação dos tempos de execução dos processos manuais 165
5.4.3 Determinação da demanda dos programas 167
5.4.4 Determinação da demanda dos arquivos 169
5.4.5 Determinação do tamanho das mensagens 170
5.5 Dimensionamento dos recursos 173
5.5.1 Projeto do arranjo físico 174
5.5.1.1 Teoria das filas 174
5.5.1.2 Dimensionamento dos postos de trabalho 177
5.5.1.3 Dimensionamento dos terminais ...179

5.5.1.4 Arranjo físico 181

5.5.2 Obtenção dos dados para o projeto da topologia 183

5.5.3 Obtenção dos dados para o projeto da configuração básica 185

5.5.3.1 Dados da aplicação 187

5.6 Instalação do sistema 193

5.7 Resumo.....196

6 CONCLUSÃO 199

6.1 Retrospectiva 199

6.1.1 Como os objetivos foram alcançados 199

6.1.2 Resultados obtidos 201

6.2 O futuro 204

6.2.1 Sugestões para futuras investigações 204

6.2.2 O que ainda precisa ser feito 207

6.2.3 Onde usar a metodologia 209

BIBLIOGRAFIA 213

LISTA DAS FIGURAS

Fig. 2.1 Configuração final 29

Fig. 2.2 Configuração básica 30

Fig. 2.3 Topologia 33

Fig. 2.4 Topologias com ligações ponto-a-ponto 35

Fig. 2.5 Topologias com ligações multiponto 35

Fig. 3.1 Croqui da rede de supermercados 38

Fig. 3.2 Topologia do sistema de computação do supermercado 40

Fig. 3.3 Sistema de computação formado por redes locais 41

Fig. 3.4 Topologia formada por elos de longa distância .42

Fig. 3.5 Dimensões de problemas de alocação de arquivos 54

Fig. 3.6 Dados do controle de estoque do supermercado ..58

Fig. 3.7 Passos do método "ADD" para o projeto da topologia 59

Fig. 3.8 Tipos clássicos de topologias 60

Fig. 3.9 Evolução das empresas 63

Fig. 3.10 Evolução das topologias com a empresa 64

Fig. 3.11 Interligação de computadores através da RENPAC 65

Fig. 3.12 Alternativas de configuração básica 68

Fig. 3.13 Comportamento dos computadores sob a mesma carga 69

Fig. 3.14 Gráfico de seleção dos computadores 70

Fig. 3.15 Modelo de um computador de grande porte 72

Fig. 4.1 Ciclo de vida dos sistemas 97

Fig. 4.2 Ciclo de vida clássico 98

Fig. 4.3 Ciclo de vida semi-estruturado 99

Fig. 4.4 Ciclo de vida estruturado 100

Fig. 4.5 Detalhes da atividade de pesquisa 102

Fig. 4.6 Detalhes da atividade de análise 103

Fig. 4.7 Detalhes da atividade de projeto 104

Fig. 4.8 Detalhes da atividade de implementação 104

Fig. 4.9 Geração de testes de aceitação 105

Fig. 4.10 Níveis das funções de uma empresa 106

Fig. 4.11 Símbolos do DFD 108

Fig. 4.12 Contexto da metodologia de projeto de sistemas distribuídos 110

Fig. 4.13 Etapas da metodologia de projeto de sistemas distribuídos 115

Fig. 4.14 Primeiro esboço da metodologia preliminar 119

Fig. 4.15 Segundo esboço da metodologia preliminar 119

Fig. 4.16 Terceiro esboço da metodologia preliminar 120

Fig. 4.17 Metodologia preliminar 121

Fig. 4.18 Ciclo de vida estruturado 124

Fig. 4.19 Primeira evolução da metodologia preliminar ..124

Fig. 4.20 Detalhes da etapa de determinação da carga ...126

Fig. 4.21 Detalhes da etapa de projeto da configuração .127

Fig. 4.22 Segunda evolução da metodologia preliminar ...128

Fig. 4.23 Detalhes da etapa de instalação do sistema ...130

Fig. 4.24 Terceira evolução da metodologia preliminar ..130

Fig. 5.1 Contexto da etapa de desenvolvimento da aplicação 133

Fig. 5.2 Detalhes da etapa de desenvolvimento da aplicação 134

Fig. 5.3 Contexto da aplicação comercializar materiais 135

Fig. 5.4 Detalhes da aplicação comercializar materiais 136

Fig. 5.5 Detalhes do processo receber materiais 136

Fig. 5.6 Dicionário de dados 137

Fig. 5.7 Diagrama de estrutura de dados 137

Fig. 5.8 Português estruturado 138

Fig. 5.9 Detalhes da atividade de análise 139

Fig. 5.10 Modelo físico corrente 140

Fig. 5.11 DFD com a fronteira homem/máquina 142

Fig. 5.12 Contexto da aplicação comercializar materiais 145

Fig. 5.13 Detalhes da aplicação comercializar materiais 146

Fig. 5.14 Detalhes do processo receber materiais 146

Fig. 5.15 Símbolos usados na descrição do método 148

Fig. 5.16 Diagrama de descrição do método (DDM) 149

Fig. 5.17 Contexto do processo automatizado 1.1.D.1 150

Fig. 5.18 Detalhes do processo automatizado 1.1.D.1 151

Fig. 5.19 Transformação do modelo lógico em físico 152

Fig. 5.20 DFD's da aplicação operar PBX 153

Fig. 5.21 Solução "on-line" da aplicação operar PBX 154

Fig. 5.22 Solução "batch" da aplicação operar PBX 155

Fig. 5.23 Contexto da etapa determinação da área geográfica 157

Fig. 5.24 Mapas dos vários níveis da área geográfica ...158

Fig. 5.25 Contexto da etapa de determinação da carga ...160

Fig. 5.26 Detalhes da etapa de determinação da carga ...161

Fig. 5.27 Tipos de sítios identificados na área geográfica 162

Fig. 5.28 Taxas dos insumos da aplicação comercializar materiais 163

Fig. 5.29 Detalhes das taxas dos insumos da aplicação comercialiar materiais 163

Fig. 5.30 Taxas dos insumos do processo receber materiais 164
Fig. 5.31 Perfil da demanda dos insumos de um processo .164
Fig. 5.32 Dados do tempo de execução dos processos manuais 166
Fig. 5.33 Cálculo das taxas de execução dos processos automatizados 168
Fig. 5.34 Demanda de execução de programas 169
Fig. 5.35 Cálculo da taxa de acesso aos arquivos 170
Fig. 5.36 Demanda de acesso aos arquivos 171
Fig. 5.37 Cálculo do tamanho das mensagens 172
Fig. 5.38 Contexto da etapa de dimensionamento dos recursos 173
Fig. 5.39 Detalhes da etapa de dimensionamento dos recursos 174
Fig. 5.40 Fila única com múltiplos servidores 176
Fig. 5.41 Representação clássica de uma fila 176
Fig. 5.42 Algorítmo e fórmulas da teoria das filas 177
Fig. 5.43 Cálculo da quantidade de postos de trabalho ..178
Fig. 5.44 Cálculo da quantidade de terminais 179
Fig. 5.45 Cálculo do tempo de uso do terminal 180
Fig. 5.46 Arranjo físico de uma sala 182
Fig. 5.47 Organização dos dados para a classificação A-B-C 189
Fig. 5.48 Curva de classificação A-B-C 190
Fig. 5.49 Contexto da etapa de instalação do sistema ...193
Fig. 5.50 Detalhes da etapa de instalação do sistema ...195

RESUMO

É proposta uma metodologia de desenvolvimento de sistemas, que reune técnicas de PED - Processamento Eletrônico de Dados, O&M - Organização & Métodos e PO - Pesquisa Operacional, possibilitando a ação integrada dos especialistas, na elaboração de um projeto de sistemas distribuídos. Os produtos da metodologia são a configuração do "hardware" e o "software" aplicativo. O "hardware" é configurado de forma estruturada em três níveis: topologia, configuração básica e configuração final. Cada nível é projetado através de programação linear, métodos heurísticos e simulação. Para viabilizar o uso destas técnicas são apresentados procedimentos de obtenção e organização de dados quantitativos. O "software" é considerado uma extensão do trabalho das pessoas, sendo projetado através da integração das técnicas estruturadas de desenvolvimento de sistemas, tempos e movimentos, bem como análise de arranjo físico. São discutidos os modelos físico e lógico dos sistemas de informação, sendo proposto um terceiro: modelo homem/maquina. A metodologia possibilita projetar o sistema em módulos, permitindo que o mesmo cresça progressivamente com a empresa.

ABSTRACT

A system development methodology is presented. It encloses techniques from EDP - Electronic Data Processing, O&M - Organization & Method, OR - Operations Research and stimulates the experts from these areas to work together, to achieve the results of a distributed system design. The main products are the hardware configuration and the aplication software. The hardware configuration is structured at three levels: topology, basic configuration and final configuration. Each level is designed through linear programming, heuristic and simulation. To suport these techniques, procedures for gathering and organizing quantitative data are presented. The software is treated as an extension of the user tasks. Techniques such as structured system development, time & motion study and layout analysis are used in the software design process. The physical and logical models of information systems are presented and a new one is introduced: the man/machine model. This methodology enables the design of a system in modules, which provides evolutionary growth with the business expansion.

U F R G S
BIBLIOTECA
CPD/PGCC

1 INTRODUÇÃO

1.1 Objetivos

Neste trabalho é proposta uma metodologia de projeto de sistemas para aplicações dispersas geograficamente.

Muitas empresas tem suas funções dispersas. Alguns exemplos são: bancos, supermercados, lojas de departamento, empresas de transporte e indústrias, entre outras. Em tais empresas, até o final dos anos 70, era adotado o processamento centralizado. Os usuários acessavam os dados, através de terminais ligados ao computador via linha telefônica. O projeto do "software" e o dimensionamento do "hardware" ocorriam em momentos diferentes. O computador era dimensionado para suportar a carga das aplicações existentes e das aplicações futuras [EIN77]. As novas aplicações consumiam a capacidade ociosa, provocando a substituição periódica de um computador por outro de maior porte. Com o barateamento dos processadores e o aumento relativo dos custos de comunicação, o processamento distribuído tornou-se mais econômico [DOW76], [CHA80].

Um sistema distribuído caracteriza-se pela existência de processadores e dados dispersos geograficamente. Há também uma rede que permite a comunicação entre os vários processadores. A bibliografia apresenta diferentes definições para sistemas distribuídos. Várias dessas definições estão reunidas em [OHA79], [STR80] e [CHA80]. Apesar da falta de uma definição precisa, praticamente todas se referem

às características inicialmente citadas: existência de mais de um processador, à dispersão geográfica dos processadores e dados, bem como à existência de um meio de comunicação entre eles.

O desenvolvimento do "software" evoluiu muito nos últimos dez anos, notadamente com a chamada "revolução dos métodos estruturados" [MAR84]. Existe a análise estruturada [DEM78], [WEI80], [GAN83]; o projeto estruturado [YOU78], [STE85]; a programação estruturada [GEL77], [MAR77], [HOL77] e outras. Apesar das dificuldades existentes para a prática efetiva das técnicas estruturadas [SUM86], sua eficácia já está sendo comprovada [SUM86]. O "ciclo de vida estruturado" proposto por Yourdon [YOU82], é uma forma de consolidação que facilita ainda mais o uso destas técnicas. Para projetar a configuração do "hardware" no entanto, restam ainda muitas questões a resolver.

Para projetar um sistema distribuído, pelo menos três questões devem ser respondidas: (1) onde alocar os computadores (2) como interligá-los e (3) qual a capacidade de cada computador. As duas primeiras questões podem ser respondidas através de trabalhos como o de Ivaldo Ohana [OHA79]. A resposta para a terceira pergunta pode ser encontrada na dissertação de Jair Strack [STR80]. Em [OHA79] são propostos modelos de programação linear para a alocação ótima de arquivos. Esta alocação ótima consiste em determinar qual o local onde os arquivos devem ficar, de maneira que o custo de comunicação seja mínimo. Esses modelos também permitem verificar se deve haver ou não ligação entre os locais

onde os arquivos estarão armazenados. Considerando que os arquivos devem estar junto aos processadores, o local onde estarão os arquivos é também o local onde estarão os processadores. Portanto, com tais modelos é possível responder as duas primeiras questões propostas no início deste parágrafo.

Em [STR80] são propostos modelos de filas, para verificar o comportamento de um computador, quando submetido a uma determinada carga. Com estes modelos é possível responder à terceira questão proposta anteriormente.

O uso de modelos pressupõe a existência de dados quantitativos devidamente coletados e organizados. Para usar os modelos de [OHA79] é preciso conhecer dados como: local de instalação dos terminais, arquivos, número de acesso aos arquivos para leitura e atualização e outros. O modelo de [STR80] pressupõe a existência de dados como número de terminais, frequência de transações, velocidade de processamento, número de acesso aos discos, quantidade de instruções processadas por transações e outros. Entretando, nos dois trabalhos não há explicação de como obter tais dados. Esta é uma das questões a serem tratadas na presente dissertação.

Como foi visto, para o projeto e desenvolvimento de "software" não só existem as ferramentas, como também há integração entre elas [YOU82]. Entretanto, para o projeto da configuração do "hardware" a situação é outra. Existem algumas ferramentas [OHA79], [STR80], que entretanto não estão integradas. Para obter os dados necessários aos modelos, organiza-los e processa-los, é preciso usar técnicas de Orga-

nização e Métodos (O&M) e Pesquisa Operacional (PO). A não ser a "praxeologia normativa" proposta por Mélese [MEL73], pouco tem sido feito para integrar estas áreas ao Processamento Eletrônico de Dados (PED). Nesta dissertação é proposta uma metodologia de projeto de sistemas distribuídos que (1) integra os trabalhos de [OHA79] e [STR80], (2) possibilita a obtenção e organização dos dados quantitativos referentes aos modelos propostos nestes trabalhos e (3) integra as técnicas de PED, O&M e PO, necessárias a obtenção de tais dados.

1.2 Organização do trabalho

O presente trabalho está organizado em três partes. A primeira é dedicada ao "hardware" sendo composta pelos capítulos 2 e 3. No capítulo 2 são apresentados os três níveis de um sistema de computação: topologia, configuração básica e configuração final. No capítulo 3 é descrito como projetar o sistema, sendo discutidas diversas formas de concepção desses três níveis.

A segunda parte tem o objetivo de descrever como coletar e organizar os dados necessários ao projeto do sitema de computação. É composta pelos capítulos 4 e 5. O capítulo 4 é dedicado à escolha de um ciclo de vida de projeto para aplicações distribuídas. São analisados os ciclos clássico e estruturado. A metodologia de projeto é proposta com base no ciclo estruturado. No capítulo 5 tal metodologia é detalhada.

A terceira parte é a conclusão. É composta pelo capítulo 6. Neste capítulo é feita uma retrospectiva de todo o trabalho. São sugeridos novos assuntos para pesquisa na área de sistemas distribuídos. A metodologia também é analisada, sendo discutidas as possibilidades de uso em outras áreas.

2 SISTEMA DE COMPUTAÇÃO

2.1 Introdução

O objetivo deste capítulo é propor um modelo de descrição do "hardware" de sistemas distribuídos.

Um sistema distribuído é composto por "software" e "hardware". O "software" pode ser dividido em dois tipos. O primeiro é o "software" básico, que é constituído pelo sistema operacional, gerenciador do banco de dados e monitor de comunicação. Estes programas costumam ser adquiridos junto com o "hardware" devido a grande dependência entre ambos. O segundo tipo é o "software" aplicativo, sendo constituído pelos programas de aplicação e pelo banco de dados.

O "hardware" é composto por processadores, periféricos e meios de comunicação. Neste trabalho o "hardware" é também chamado de sistema_de_computação. Para facilitar a atividade de projeto, está sendo proposto um modelo de descrição do sistema de computação. Tal modelo tem três níveis: configuração final, configuração básica e topologia. Tais níveis são descritos nas próximas seções.

2.2 Configuração final

A configuração final de um sistema de computação é composta por equipamentos e meios de comunicação que podem ser adquiridos no mercado. Os equipamentos a destacar são computadores e periféricos. Os computadores são comumente

classificados em micro, mini e grande porte ("mainframe"). Não é uma classificação precisa, mas tem sido universalmente adotada. Um microcomputador é uma síntese do terminal de vídeo, processador, memória (e mais recentemente discos magnéticos) num único equipamento. Um minicomputador é um conjunto formado por processador e memória, ao qual é possível ligar diretamente os periféricos. Um computador de grande porte é um equipamento no qual estão reunidos o processador central (CPU), memória e processadores de entrada e saída (também chamados canais). Para aliviar a carga da CPU, normalmente há um processador especializado em comunicação chamado processador de comunicação ("front end processor"). Os periféricos são unidades de disco, impressoras, unidades de fita e terminais de vídeo entre outros. Eles são ligados aos processadores através dos canais e unidades controladoras.

Os meios de comunicação são compostos por canais e equipamentos de comunicação. Os canais de comunicação variam conforme a distância entre os equipamentos a serem ligados. De 1 a 15 metros são usados cabos com múltiplos fios, como por exemplo o interface RS232 [DOL78]. Para distâncias de até 1500 metros são usados par trançado e cabo coaxial. Para distâncias maiores é usado o sistema de telecomunicações do país, composto de par trançado, tronco de microondas e satélite entre outros [TAR84], [TAN81], [DOL78]. Os equipamentos de comunicação são modems e multiplexadores. Na fig. 2.1, está a configuração final de um sistema de computação que suporta uma aplicação cujos usuários estão dispersos em seis cidades.

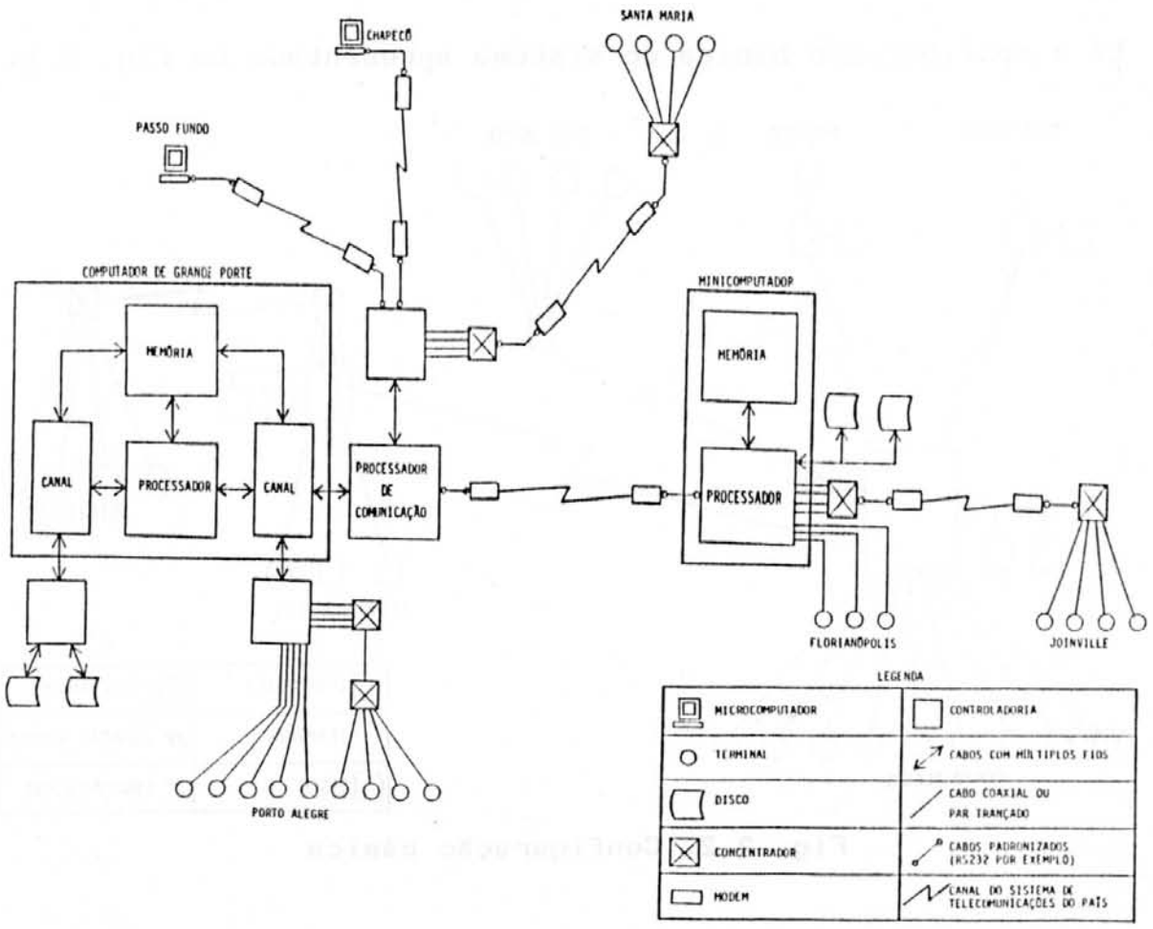


Fig. 2.1 Configuração final

2.3 Configuração básica

A configuração básica [AGA85] é um modelo simplificado do sistema de computação. O computador de grande porte e os canais de comunicação são os componentes que na configuração básica aparecem de forma simplificada.

O computador de grande porte é considerado como um equipamento no qual se ligam diretamente os periféricos. Não aparecem no modelo as unidades de controle, processadores de entrada e saída e o processador de comunicação.

O meio de comunicação é considerado como uma linha que tem determinada velocidade. Os modens e alguns cabos co-

mo o interface RS232 não aparecem no modelo. Na fig. 2.2 está a configuração básica do sistema apresentado na fig. 2.1.

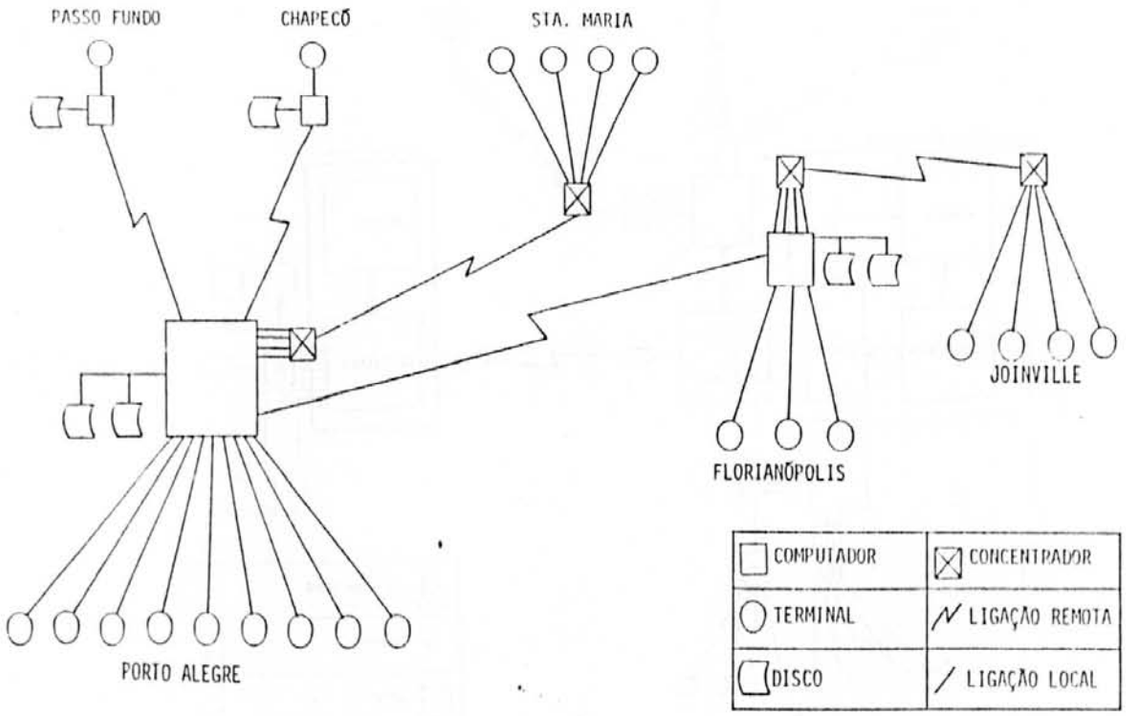


Fig. 2.2 Configuração básica

Os principais motivos que levaram às simplificações são: (1) redução da complexidade, (2) dificuldade de modelar a realidade, (3) grau de agregação dos produtos, (4) variáveis de desempenho disponíveis e (5) dependência hierárquica de comportamento.

O primeiro motivo é a redução da complexidade. Quando um produto é projetado, não é possível levar em consideração todos os detalhes desde o início. O projeto de qualquer produto de engenharia é feito em níveis crescentes de complexidade. Isso facilita o entendimento entre as pessoas e possibilita o trabalho harmônico entre os vários especialistas. Na engenharia de "software" exemplos disso são o desenvolvimento estruturado [YOU82], a análise estruturada [DEM78], [WEI80], [GAN83], projeto estruturado [YOU78], [STE85] e programação estruturada [GEL77], [MAR77], [HOL77].

O segundo motivo é a dificuldade de modelar a realidade. O resultado do projeto de um produto é impreciso por natureza. Há imperfeições no modelo pois sempre haverá diferenças entre o modelo e a realidade. Também há imprecisões nos dados. Em termos de modelo, a imprecisão pode ser diminuida se o grau de detalhamento for aumentado. Entretanto, quanto mais detalhado o modelo, maior a necessidade de dados. Sem a realidade presente, muitos dados não podem ser obtidos. Por exemplo, nos casos onde a precisão é fundamental, como aviões, carros e armas entre outros, um protótipo costuma ser desenvolvido para que os dados sejam obtidos. A partir de tais dados o projeto vai sendo ajustado. No caso de aplicações comerciais, a adoção de uma estratégia destas nem sempre é viável devido aos altos custos envolvidos. Como o produto resultante muitas vezes é único, há poucas chances de diluir tais custos.

O terceiro motivo é o grau de agregação dos equipamentos. Um computador é composto por diversas unidades. Tais unidades por sua vez podem ser compostas por outras unidades e assim sucessivamente. Se forem comparados os computadores de diversos fabricantes, é possível notar neles uma estrutura semelhante. Por exemplo, todos os computadores são formados de pelo menos um processador e uma memória. Entretanto, a partir de um grau de detalhamento maior as estruturas começam a se diversificar. Alguns computadores tem processador de entrada e saída e outros não. Devido a isso, num primeiro estágio do projeto da configuração é aconselhável desenvolver um modelo genérico, que descreva com razoável precisão os equipamentos, independentemente dos fabri-

cantes. Desta forma é possível ter uma base única para estabelecer comparações de custo e desempenho, na seleção de um equipamento que tenha vários fornecedores alternativos.

O quarto motivo são as variáveis de desempenho. Algumas variáveis já são consagradas: velocidade da CPU e tempo de acesso a disco [WIE77]. A velocidade da CPU é dada em milhões de instruções por segundo (MIP). Nesta variável está incluido o tempo de acesso à memória. Qualquer tentativa de modelar processador e memória separadamente, pode levar à inutilidade deste dado (MIP) e a consequente dificuldade de obtenção da velocidade do processador e do tempo de acesso à memória. O tempo de acesso ao disco costuma ser definido como a soma dos tempos médios do movimento do braço ("seek") e a latência. Não há razão aparente para modelar o movimento de braço e a rotação do disco separadamente. Detalhes como estes exigem dados adicionais como número do cilindro em que o arquivo está alocado no disco e o tamanho do registro físico (bloco). Isso é incompatível com o projeto do sistema a nível de configuração. Uma precisão como esta só se justifica quando do ajuste fino do sistema.

O quinto motivo é a dependência hierárquica de comportamento entre os equipamentos. As linhas de comunicação e os computadores são recursos independentes em termos de fornecedores. Entretanto o comportamento de uma linha é ditado pelo protocolo, que normalmente é uma imposição do fabricante do computador. Como normalmente existem vários fabricantes alternativos, se torna muito complexo dimensionar as linhas de comunicação levando em consideração o pro-

tocolo desde o início do projeto. A nível de configuração, a velocidade em bits/segundo é suficiente.

2.4 Topologia

A configuração básica pode ser representada como um grafo de elos e nós. Os nós representam os locais onde estão os equipamentos e os elos representam os canais de comunicação entre estes equipamentos. A disposição geográfica dos nós e as ligações entre eles é chamada topologia [TAR84]. Na fig. 2.3 está a topologia da configuração básica descrita na fig. 2.2.

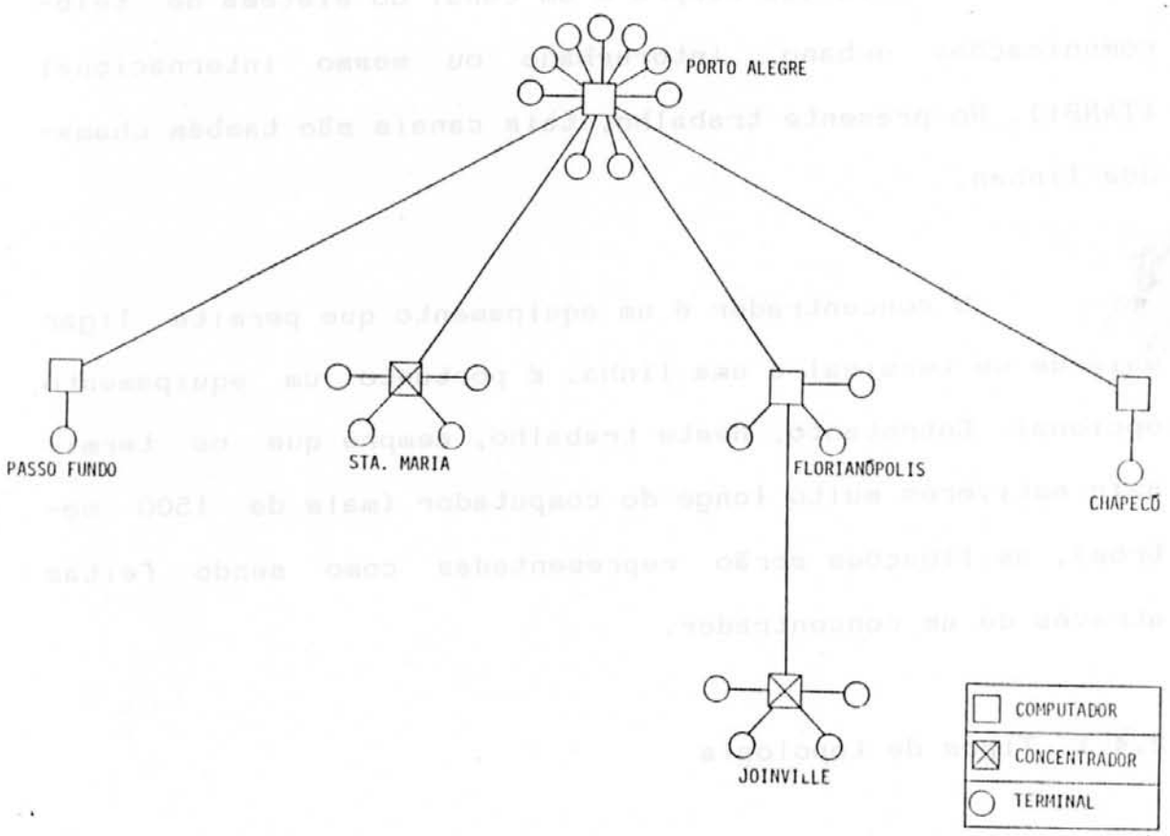


Fig. 2.3 Topologia

Cada nó pode ser implementado como um dos seguintes recursos: terminal, concentrador ou computador. Os terminais ficam nos locais de trabalho dos usuários (escritó-

rio, gabinete, sala de aula e outros). Computadores e concentradores ficam próximo aos terminais (distâncias de até 1500 metros). As distâncias computador-computador e computador-concentrador são maiores que 1500 metros.

Os elos podem ser classificados em dois tipos: local e longa distância. Um elo local representa ligação de um terminal a um concentrador ou a um computador. Neste caso o meio de comunicação pode ser um cabo coaxial ou um par trançado. Um elo de longa distância representa ligação entre dois computadores ou entre um computador e um concentrador. O meio de comunicação sempre é um canal do sistema de telecomunicações urbano, interurbano ou mesmo internacional [TAN81]. No presente trabalho, tais canais são também chamados linhas.

O concentrador é um equipamento que permite ligar mais de um terminal a uma linha. É portanto um equipamento opcional. Entretanto, neste trabalho, sempre que os terminais estiverem muito longe do computador (mais de 1500 metros), as ligações serão representadas como sendo feitas através de um concentrador.

2.4.1 Tipos de topologia

Elos e nós formam grafos que determinam os tipos de topologia. Os elos representam ligações. As ligações podem ser de dois tipos [TAN81]: ponto a ponto e multiponto. Numa ligação ponto a ponto, o canal correspondente é privativo da comunicação de apenas dois nós. Algumas das topolo-

gias resultantes destes tipos de ligações são apresentadas na fig. 2.4. Numa ligação multiponto, os canais podem ser compartilhados por mais de dois nós. A transmissão é do tipo "broadcasting" [TAN81]. As topologias resultantes deste tipo de ligação são apresentadas na fig. 2.5.

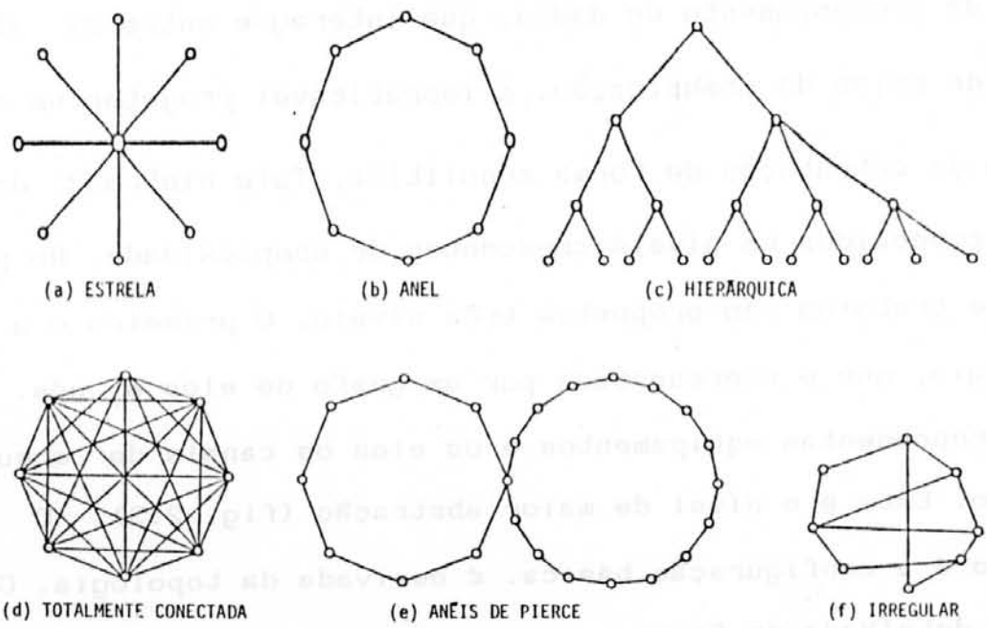


Fig. 2.4 Topologias com ligações ponto-a-ponto

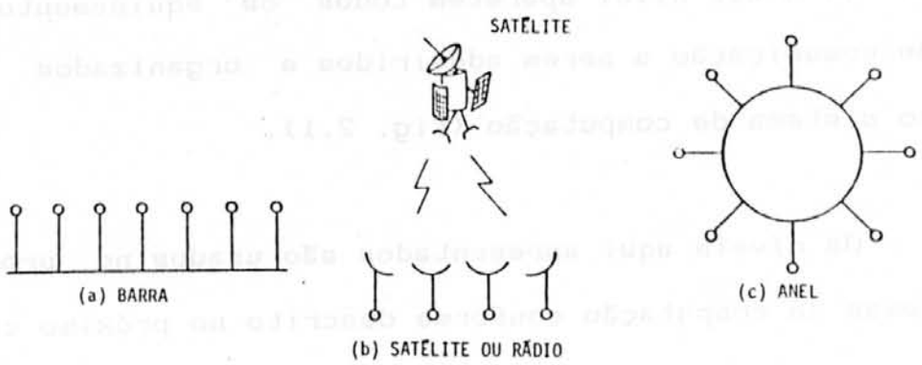


Fig. 2.5 Topologias com ligações multiponto

Em [AND75] estão descritos os tipos de interconexão que podem ser feitas entre computadores. Os autores de tal trabalho tentaram criar uma taxionomia para as topologias. Entretanto, de acordo com [TAN81], ainda não há con-

senso entre todos os autores.

2.5 Resumo

Sistema de computação é um conjunto de equipamentos de processamento de dados, que interagem entre si através de meios de comunicação. É impraticável projetar um sistema de computação de forma monolítica. Tais sistemas devem ser concebidos em níveis crescentes de complexidade. No presente trabalho são propostos três níveis. O primeiro é a topologia, que é representada por um grafo de elos e nós. Os nós representam equipamentos e os elos os canais de comunicação. Este é o nível de maior abstração (fig. 2.3). O segundo é a configuração básica. É derivada da topologia. Cada nó é detalhado de forma a representar um dos seguintes equipamentos: computador (micro, mini ou grande porte), concentrador ou terminal (fig. 2.2). O terceiro nível é a configuração final. Neste nível aparecem todos os equipamentos e meios de comunicação a serem adquiridos e organizados para formar o sistema de computação (fig. 2.1).

Os níveis aqui apresentados são usados no projeto de sistemas de computação conforme descrito no próximo capítulo.

3 COMO PROJETAR O SISTEMA DE COMPUTAÇÃO

3.1 Introdução

Este capítulo tem dois objetivos. O primeiro é estabelecer as etapas do projeto do sistema de computação. O segundo é descrever as alternativas para executar cada etapa, os modelos usados e as variáveis envolvidas.

Conforme discutido no capítulo 2, um sistema de computação pode ser descrito em três níveis: (1) topologia, (2) configuração básica e (3) configuração final. O projeto do sistema de computação deve ser executado segundo esses níveis. As razões são as que seguem.

Normalmente, quando o projeto é desenvolvido em níveis, de cima para baixo ("top-down"), fica mais simples de ser executado, é mais fácil de ser gerenciado e possibilita a elaboração de uma documentação mais efetiva entre outras vantagens. Esta estratégia de desenvolvimento, tem sido o principal fator de sucesso das metodologias da área de "software", tais como a análise [DEM78], [WEI80], [GAN83], projeto [YOU78], [STE85] e programação [GEL77], [MAR77], [HOL77]. A seguir é proposta uma aplicação fictícia, para facilitar a compreensão do restante do texto.

3.2 Exemplo de aplicação

Seja uma rede de supermercados com um depósito central e três lojas. A aplicação estudada é o controle de

estoque. No depósito central, a entrada e saída de mercadorias é registrada por meio de cinco terminais. Nas lojas tal controle também é feito por terminais. Há dois na primeira, três na segunda e um na terceira. Há também um terminal em cada caixa. A primeira loja tem seis caixas, a segunda oito e a terceira cinco. Um croqui com as lojas, depósito, terminais e distâncias é apresentado na fig. 3.1. As transações são efetuadas sobre dois arquivos (produtos e pedidos), conforme descrito a seguir.

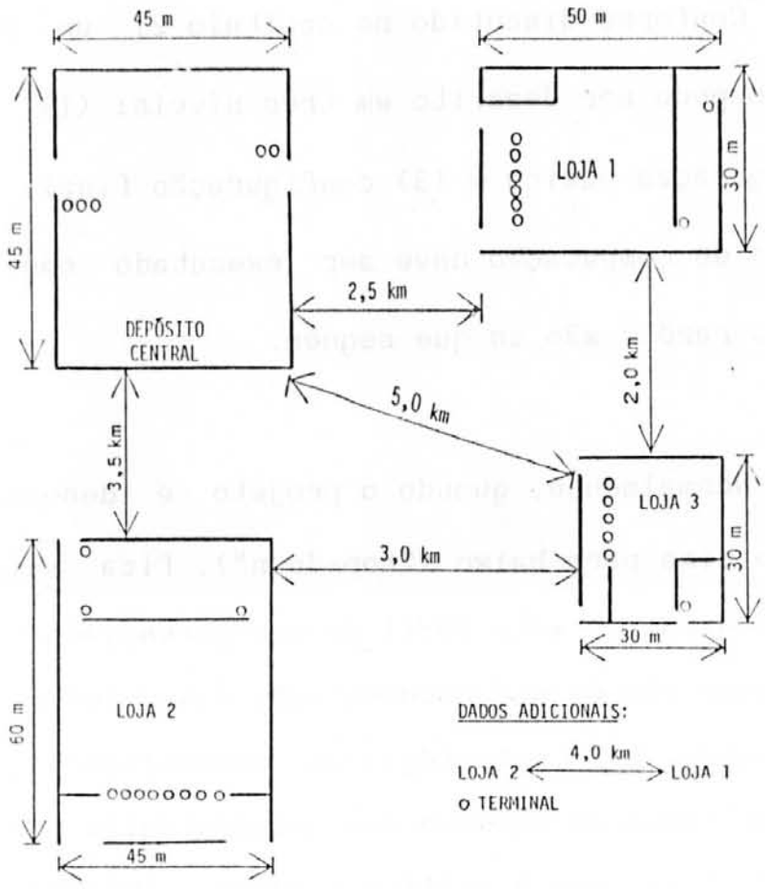


Fig. 3.1 Croqui da rede de supermercados

Existem três tipos de transações. O primeiro é a entrada de mercadorias no depósito central. A quantidade recebida é subtraída do pedido e adicionada ao estoque de produtos do depósito central. O segundo tipo é a saída de mercadorias dos depósitos. A quantidade é abatida do estoque de produtos. O terceiro é a entrada de mercadorias nos depósi-

tos das lojas. A quantidade é somada ao estoque de produtos. O exemplo aqui descrito serve de base para a descrição do projeto da topologia.

3.3 Projeto da topologia

Conforme visto no capítulo 2, a topologia pode ser representada como um grafo de elos e nós. Os nós representam terminais, concentradores ou computadores. Os elos podem ser locais ou de longa distância. A quantidade, dimensões, tipo e localização dos elos e nós dependem da aplicação.

Segundo [TAN81], projetar uma topologia é obter o desempenho ("performance") especificado ao mínimo custo. Para uma mesma aplicação pode haver uma infinidade de topologias que satisfazem à especificação. Analisar todas as alternativas para achar a de menor custo seria um trabalho impraticável. Nesta seção são apresentados métodos que permitem racionalizar esse processo de escolha, simplificando a execução do projeto. Essa questão será tratada mais adiante. A seguir será discutida a porção da topclogia formada por elos locais.

3.3.1 Considerações sobre os elos locais

Na aplicação descrita em 3.2, os terminais estão localizados nas lojas e no depósito, formando pequenos centros. Estes centros não se restringem aos supermercados, sendo comum em todos os ramos de negócio. Cada terminal é representado por um nó. Os locais onde os terminais estão

reunidos também são representados por nós. Em tais nós podem existir concentradores ou computadores. Os elos locais ligam terminais a computadores ou a concentradores. Os elos de longa distância ligam computador a computador ou a concentrador. Com um computador no depósito e concentradores nas lojas, a topologia toma a forma da fig. 3.2. Essa entretanto é apenas uma alternativa.

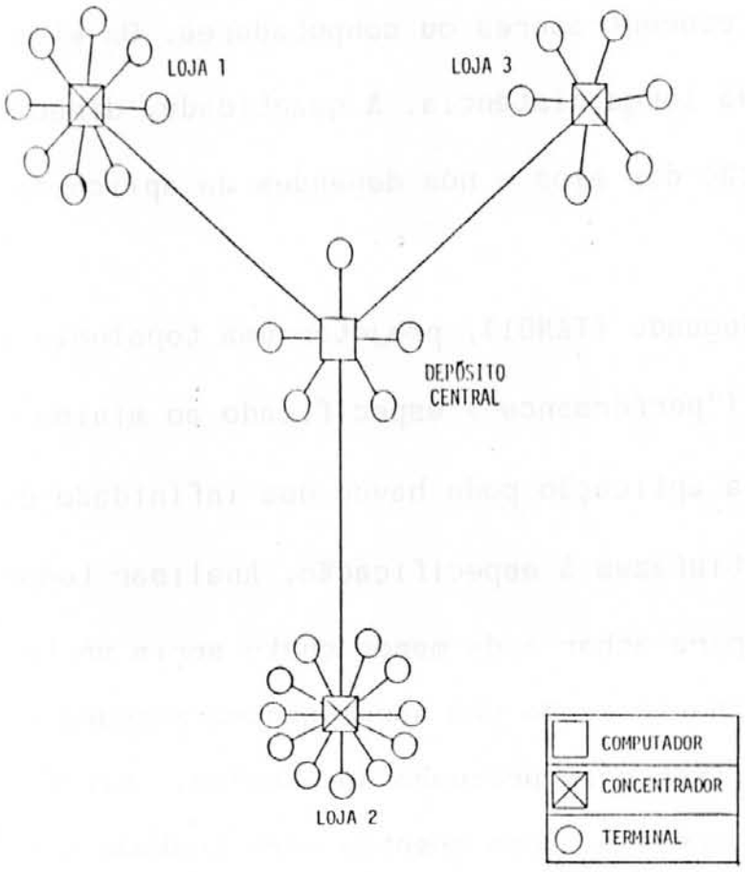


Fig. 3.2 Topologia do sistema de computação do supermercado

Se no lugar de terminais forem usados microcomputadores, é possível interligá-los em forma de rede local [TAN81], [MEN84]. Esta é a alternativa mais recente. Há entretanto uma solução de compromisso entre usar redes locais ou computadores com terminais. Caso seja usada rede local na aplicação proposta em 3.2, a topologia pode ter a forma da fig. 3.3. Neste trabalho não serão analisadas as possibilidades de uso de redes locais.

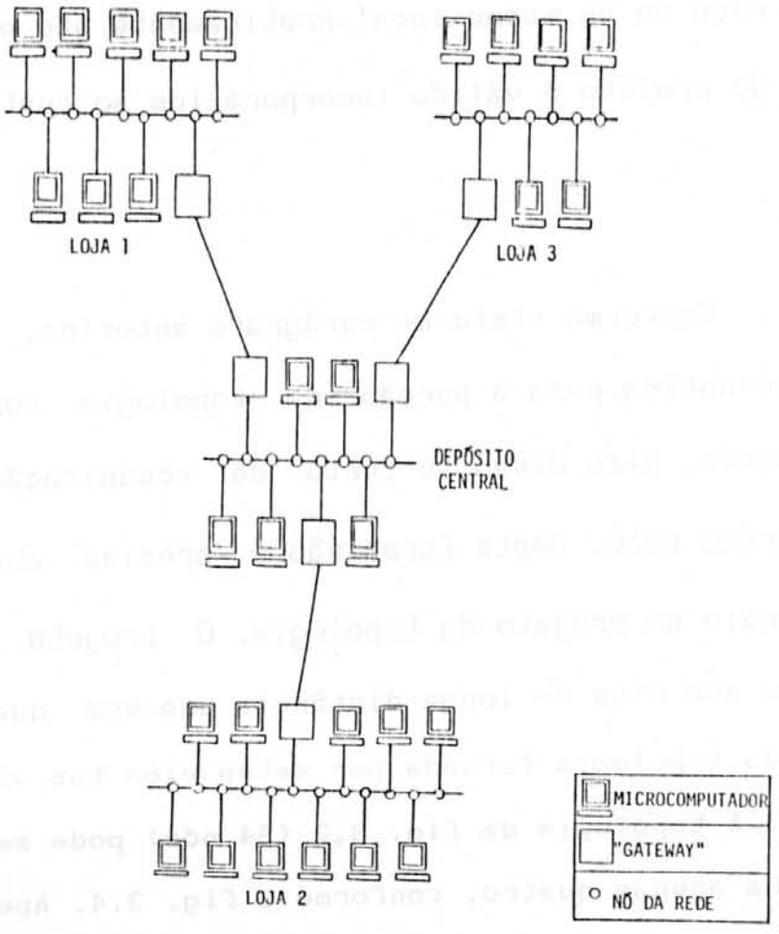


Fig. 3.3 Sistema de computação formado por redes locais

Como foi visto, projetar uma topologia consiste em escolher a alternativa que proporcione o menor custo. A complexidade do projeto aumenta ou diminui de acordo com o número de alternativas. Há duas propriedades dos elos locais que permitem diminuir as alternativas, simplificando o projeto. Essas propriedades são discutidas a seguir.

A primeira propriedade diz respeito à topologia. Os elos locais ligam terminais a computadores ou a concentradores. Desta forma, a porção da topologia formada pelos elos locais é do tipo estrela. O nó central é um concentrador ou um computador (fig. 3.2). A segunda propriedade se refere ao custo. Os elos locais são compostos por cabos e modens. Os custos destes equipamentos são relativamente fixos e bem determinados, de forma que a variação de custo en-

tre os elos de um mesmo local praticamente não existe. Para efeito de projeto é válido incorporá-los ao custo do terminal.

Conforme visto no parágrafo anterior, há somente uma alternativa para a porção da topologia formada pelos elos locais. Além disso, o custo de comunicação pode ser considerado nulo. Desta forma não é preciso considerar os elos locais no projeto da topologia. O projeto pode ficar restrito aos elos de longa distância uma vez que apenas a porção da topologia formada por estes elos tem várias alternativas. A topologia da fig. 3.2 (34 nós) pode ser simplificada para apenas quatro, conforme a fig. 3.4. Apesar destas simplificações, há outros fatores que dificultam o projeto.

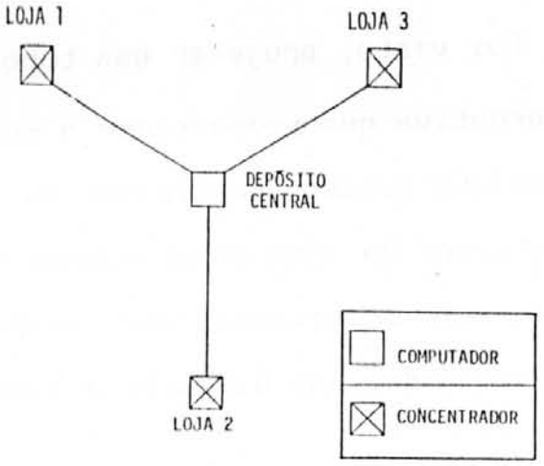


Fig. 3.4 Topologia formada por elos de longa distância

3.3.2 Dificuldades para projetar a topologia

Segundo [TAN81] o projeto de uma topologia pode ser um problema completamente intratável. Contribuem para isso três fatores. O primeiro é o número de nós. Para (n) nós, há possibilidade de existir n*(n-1)/2 elos. Desde que

cada elo pode ou não existir, a quantidade de alternativas pode ser expressa por 2**(n*(n-1)/2) . Desta forma, a topologia da fig. 3.4 é uma entre 64 possíveis. Se ao invés de 4 fossem 10 nós, as possibilidades saltariam para cerca de 30 trilhões (2**45).

O segundo fator se refere à velocidade das linhas. Se a velocidade ótima num elo fosse por exemplo 6.000 bits/segundo, não haveria como adotar uma linha desta. Os canais de comunicação disponíveis tem velocidade discretas como 50, 110, 300, 600, 1200, 2400, 4800 e 9600 bits/segundo. Fazendo com que (C#) represente o número de velocidades disponíveis, a quantidade de alternativas para a topologia passa a ser (C#+1)**(n*(n-1)/2). Caso as opções de velocidade se restrinjam a 1200 ou 2400, a topologia da fig. 3.4 é uma entre 729 possibilidades.

O terceiro fator é a quantidade de arquivos. A escolha do nó onde armazenar os arquivos é que vai determinar a quantidade de elos de longa distância, a velocidade das linhas e consequentemente o custo de comunicação. Cada arquivo pode ou não estar alocado num nó. Fazendo com que (i) represente a quantidade de arquivos, haverá n**i possibilidades de alocação dos arquivos no conjunto de nós. Com isso a quantidade de alternativas a serem analisadas no projeto da topologia passa a ser (n**i)*((C#+1)**(n*(n-1)). Caso existam 2 arquivos e 2 opções de velocidade, a topologia da fig. 3.4 é uma entre 11.664 possibilidades.

Há uma topologia ótima para cada aplicação. Tal

topologia pode ser obtida através de modelos de programação linear. Entretanto, devido aos fatores discutidos acima, isso nem sempre é viável, sendo necessário adotar formas adicionais de projeto. Tais formas são: métodos heurísticos, soluções pragmáticas e mescla das anteriores. Todas as opções de projeto são discutidas nas próximas seções.

3.3.3 Modelos de programação linear

Um dos objetivos do projeto da topologia é minimizar o custo de comunicação. Este custo depende do fluxo de dados entre os nós. Se os nós forem auto-suficientes em informações, não há fluxo de dados, os elos não existem e o custo de comunicação é nulo. Entretanto, na maioria das aplicações o mesmo dado é usado em mais de um local. Para viabilizar tal compartilhamento é preciso distribuir os dados entre os nós e estabelecer as ligações. O resultado é a topologia. A distribuição dos dados e o estabelecimento de ligações devem ser feitos visando minimizar o custo de comunicação. Tal tarefa pode ser executada através de um modelo de programação linear inteira [WAG75] como o proposto a seguir.

3.3.3.1 Descrição do modelo

O modelo aqui descrito está baseado num dos modelos propostos em [OHA79]. Sejam (m) arquivos e (n) nós. A questão é distribuir os arquivos e determinar os elos com as respectivas capacidades, de forma a que o custo de comunicação seja mínimo. São dados do problema:

n = número de nós;

m = número de arquivos;

L = n*(n-1)/2 = número de ligações;

C# = número de opções de capacidade das linhas.

C_c = capacidades das linhas de comunicação. Por exemplo $C_1 = 1$, $C_2 = 2$, ..., $C_{C\#} = 16$ mps (mensagens por segundo). O método de cálculo de (C_c) é apresentados logo após a descrição do modelo, na seção 3.3.3.2.

$C_{\ell c}$ = custo mensal da ligação (ℓ) com capacidade (c);

A_{ij} = taxa de atualização do arquivo (i) geradas pelo nó (j). Tem a mesma unidade de medida de (C_c).

P_{ij} = taxa de leitura do arquivo (i) geradas pelo nó (j). Tem a mesma unidade de medida de (C_c).

No modelo aqui proposto, os resultados do problema são expressos através de variáveis de decisão. Tais variáveis assumem os valores (0) ou (1) para representar respectivamente falso ou verdadeiro. Os resultados são:

$$Y_{\ell c} = \begin{cases} 1 & \text{se a ligação } (\ell) \text{ tem a capacidade (c)} \\ 0 & \text{caso contrário} \end{cases}$$

$$X_{ik} = \begin{cases} 1 & \text{se o arquivo (i) está armazenado no nó (k)} \\ 0 & \text{caso contrário} \end{cases}$$

Também é necessária uma variável auxiliar:

$$Z_{ijk} = \begin{cases} 1 & \text{se o nó (j) solicita pedidos ao arquivo (i) localizado no nó (k)} \\ 0 & \text{caso contrário} \end{cases}$$

Esta variável só é definida para $j \neq k$.

Devem ser satisfeitas cinco restrições:

(1) Cada ligação pode ter no máximo uma capacidade. Tal restrição é representada por (L) inequações do tipo:

$$\sum_{c=1}^{c\#} Y_{\ell c} \leq 1 \qquad \text{sendo } \ell = 1 \ldots L$$

(2) Deve existir pelo menos uma cópia do arquivo (i). Há (m) inequações:

$$\sum_{k=1}^{n} X_{ik} \geq 1 \qquad \text{sendo } i = 1 \ldots m$$

(3) A capacidade da ligação deve ser maior ou igual ao volume de transações que usam a linha. Há (L) inequações:

$$\sum_{i=1}^{m} (A_{ij} X_{ik} + A_{ik} X_{ij} + P_{ij} Z_{ijk} + P_{ik} Z_{ikj}) - \sum_{c=1}^{c\#} C_c Y_{\ell c} \leq 0$$

sendo $\ell = 1 \ldots L$ e j,k os nós ligados por (ℓ)

(4) Se o arquivo (i) está em (j), este nó não deve efetuar pedidos a outros nós. Há n*m inequações:

$$X_{ij} + \sum_{k=1}^{n} Z_{ijk} = 1 \qquad \text{sendo } i=1 \ldots m, \; j=1 \ldots n$$

(5) Para que um nó efetue pedidos a outro, é necessário que o arquivo exista neste outro nó. Há m*n*(n-1) inequações:

$$Z_{ijk} - X_{ik} \leq 0 \qquad \text{sendo } i=1 \ldots m, \; j=1 \ldots n, \; k=1 \ldots n$$

A função objetivo a minimizar é:

$$\sum_{\ell=1}^{L} \sum_{c=1}^{c\#} C_{\ell c} Y_{\ell c}$$

A seguir é descrito o método de cálculo da capacidade das linhas (C_c), usado na restrição (3).

3.3.3.2 Cálculo da capacidade das linhas

No capítulo 5 será visto como obter os dados do modelo acima descrito. Uma parte dos dados (n, m, L, A_{ij} e P_{ij}), é obtida a partir do projeto da aplicação e outra parte (C#, C_c e $C_{\ell c}$), seria fornecida pela empresa de telecomunicações (EMBRATEL). Entretanto, no modelo, a unidade de medida de (C_c) deve ser em mensagens por unidade de tempo. Desta forma, não é possível usar diretamente a unidade de

medida fornecida pela EMBRATATEL que é bps (bits por segundo). Para calcular a unidade de medida do modelo, são necessários dados adicionais do projeto da aplicação conforme descrito a seguir.

As linhas de comunicação são comercializadas com velocidades discretas. Os canais da EMBRATEL tem as seguintes velocidades de transmissão [TAR84]: 50, 75, 100, 200, 300, 600, 1200, 2400, 4800 e 9600 bps. Cada uma destas velocidades deve ser transformada em mensagens por segundo (mps). Os cálculos são efetuados em duas etapas:

(1) Cálculo_da_velocidade_mínima (V)

$$V = (((Ms + Mr) * Nc) / Tr) * K$$

onde Ms = tamanho médio da mensagem de solicitação (em caracteres);

Mr = tamanho médio da mensagem de resposta (em caracteres);

Nc = unidades de informação por caracter (em bits);

Tr = tempo de resposta;

K = fator de correção devido a erros de transmissão, protocolo de comunicação, tempo de propagação e tempo de processamento.

(2) Cálculo_da_capacidade_das_linhas (C_c)

$$C_c = (\text{velocidade do canal}) / V$$

Para efetuar o cálculo acima, ambas as velocidades devem ter a mesma unidade de medida. Caso C_c<1, o canal em questão deve ser descartado pois não atinge a capacidade mínima. Para ilustrar segue um exemplo:

Seja: Ms=120, Mr=80, Nc=8, Tr=3 e K=1,125

Então V=600 bps

Portanto C_1 = 600/600 = 1 mps

C_2 = 1200/600 = 2 ''

:

C_{c*} = 9600/600 = 16 ''

Todos os canais com velocidade menor que 600 bps não satisfazem à velocidade mínima e foram descartados.

Nesta seção foi visto como calcular a capacidade das linhas de comunicação (C_c). Na próxima será visto como determinar as dimensões do problema.

3.3.3.3 Dimensões do problema

As dimensões de um problema de programação linear podem ser expressas através da quantidade de variáveis e equações. Como será visto mais adiante nesta seção, tais medidas permitem calcular o volume aproximado de recursos de computação necessários à solução do problema, contribuindo para a decisão de usar ou não o modelo, em função dos equipamentos de computação disponíveis.

O problema de programação linear proposto na seção anterior tem dois tipos de variáveis: (1) variáveis resultado e (2) variáveis auxiliares. Tal problema pode ser resolvido através do algorítmo "branch-and-bound" [WAG75]. Para usar tal algorítmo é necessário acrescentar ao modelo mais dois tipos de variáveis. Desta forma o problema passa a ter quatro tipos. A quantidade de cada tipo é descrita a seguir.

O primeiro tipo é constituído pelas variáveis resultado: há L*C# variáveis $Y_{\ell c}$ e m*n X_{ik}. O segundo tipo são as variáveis auxiliares: há m*n*(n-1) variáveis Z_{ijk}. O terceiro tipo são as variáveis artificiais (W). Tais variáveis devem ser acrescentadas para transformar todas as restrições em (<) ou ($\leq$) [WAG75]. Desta forma, devido às restrições (2) e (4) devem ser introduzidas (m) variáveis W_{2i} e m*n variáveis W_{4ij}. O quarto tipo são as variáveis de folga (F) que também são uma exigência do algorítmo "branch-and-bound". A introdução delas elimina as inequações [WAG75]. Há uma variável para cada restrição: (L) $F_{1\ell}$, (m) F_{2i}, (L) $F_{3\ell}$, m*n F_{4ij} e m*n*(n-1) F_{5ijk}.

Como foi visto, as variáveis de folga servem para eliminar as inequações. Desta forma o modelo é transformado num sistema de equações: há (L) equações oriundas da restrição (1), (m) da restrição (2), (L) da restrição (3), n*m da restrição (4) e m*n*(n-1) da restrição (5). Há ainda uma equação adicional devido à função objetivo [WAG75]. O sistema de equações assim obtido terá (E) equações e (V) variáveis conforme segue:

$$E = 1 + L + m + L + (n*m) + m*n*(n-1)$$

$$V = L*C\# + m*n + m*n*(n-1) + m + m*n + L + m + L + m*n + m*n*(n-1)$$

Como $L = n*(n-1)/2$ então:

$$E = 1 + n*(n-1)*(m+1) + m*(n+1)$$

$$V = n*(n-1)*(C\#+4*m+2)/2 + m*(3*n+2)$$

O exemplo de aplicação proposto no início deste capítulo (seção 3.2) pode ser usado para ilustrar as dimensões do problema. Há quatro nós (n=4) e dois arquivos (m=2). Se o conjunto de capacidades das linhas for C(1, 2, 4 mps), então C#=3. Desta forma, o problema a ser resolvido terá 47 equações e 106 variáveis. é impraticável resolver manualmente um problema com tais dimensões, sendo necessário usar um computador.

A matriz com os índices do sistema de equações referido no parágrafo anterior tem 47 colunas e 106 linhas. Considerando que são necessários dois bytes para armazenar cada índice, a quantidade de memória necessária será de 9.964 bytes.

Para resolver problemas de programação linear existem vários programas. Tais programas ocupam uma quantidade fixa de memória. A quantidade de bytes calculada no parágrafo anterior, adicionada à quantidade fixa de memória,

permitem decidir se o computador disponível pode ou não ser usado na solução do problema. O algorítmo "branch-and-bound" está disponível nos computadores Burroughs através do programa TEMPO [BUR75].

Nem sempre é recomendável usar o modelo proposto na seção 3.3.3.1. Há vantagens e limitações que devem ser observadas, conforme descrito na próxima seção.

3.3.3.4 Vantagens e limitações

O uso de programação inteira para projetar a topologia apresenta duas vantagens. A primeira consiste em executar o projeto de forma quantitativa e não apenas intuitivamente. A segunda vantagem é que a solução encontrada sempre será ótima, resultando numa topologia de menor custo de comunicação. Entretanto o modelo apresenta três limitações, discutidas a seguir.

A primeira limitação é o fato de o modelo se restringir a uma situação estática. Isso contrasta com a realidade dinâmica das empresas. Por exemplo, o problema do supermercado descrito no início deste capítulo (3.2) é apenas um instantâneo ("snapshot") da empresa. Na realidade, com o passar do tempo podem surgir novas lojas aumentando o número de nós, o que obriga a reprojetar a topologia. Para determinados segmentos da economia como os bancos onde a implantação de novas agências é frequente, o uso do modelo aqui descrito torna-se pouco prático.

Caso a empresa em questão tenha um plano de expansão confiável, os efeitos da primeira limitação podem ser amenizados. Para cada expansão é possível projetar previamente a respectiva topologia. Desta forma, à medida que o plano for sendo concretizado a topologia pode ser ajustada.

A segunda limitação se refere às dimensões do problema. O exemplo do supermercado descrito no parágrafo anterior resultou num sistema de 47 equações e 106 variáveis. Se o número de arquivos fosse três (m=3), o número de equações passaria para 64 e as variáveis para 144. Se a quantidade de nós fosse cinco (n=5), as equações seriam 73 e as variáveis 164. Se arquivos e nós forem aumentados de um, o problema resulta num sistema de 99 equações e 221 variáveis. Portanto o modelo aqui proposto só pode ser usado em problemas com poucos arquivos e nós, cujo equacionamento gere um sistema de equações compatível com o computador disponível para resolvê-lo. Na fig. 3.5 são apresentados exemplos de dimensões de alguns problemas.

A terceira limitação é o fato de o modelo não cobrir totalmente a realidade. Há pelo menos três imperfeições. A primeira se refere ao fato de no modelo não estarem previstas questões de segurança e disponibilidade das linhas. Há casos onde a disponibilidade do sistema de computação deve ficar em torno de 100%. Em tais casos, a topologia deve ter elos redundantes, para que haja caminhos alternativos quando há queda das linhas, evitanto que nós fiquem isolados. A segunda imperfeição é o fato de não ter sido levada em consideração a capacidade máxima de armazenamento de cada

SE			ENTÃO		
NÚMERO DE NÓS	NÚMERO DE ARQUIVOS	Nº DE OPÇÕES DE CAPACIDADE DE LINHAS	NÚMERO DE VARIÁVEIS	NÚMERO DE EQUAÇÕES	QUANTIDADE DE BYTES PARA ARMAZENAR A MATRIZ *
n	m	C#	V	E	
2	1	1	15	8	240
2	1	3	17	8	272
2	2	1	27	13	702
2	2	3	29	13	754
3	3	1	78	37	5772
3	3	3	84	37	6216
3	4	1	101	47	9494
3	4	3	107	47	10058
4	5	1	208	98	40768
4	5	3	220	98	43120
4	6	1	246	115	56580
4	6	3	258	115	59340
5	7	1	429	203	174174
5	7	3	449	203	182294

* CADA INDICE OCUPA 2 BYTES

Fig. 3.5 Dimensões de problemas de alocação de arquivos

nó. Sem esta restrição, pode acontecer que para usar a solução ótima, seja necessário instalar em determinados nós, computadores maiores que os disponíveis no mercado. A terceira imperfeição é a questão do particionamento dos arquivos. Particionamentos horizontais [CER82] e verticais [NAV84] levam a um menor custo de comunicação. Entretanto os efeitos destas três imperfeições podem ser amenizados.

A questão de disponibilidade (primeira imperfeição), pode ser resolvida se o projetista fixar previamente os elos redundantes. Isso pode ser feito atribuindo o valor (1) às variáveis de decisão em questão (no caso $Y_{\ell c}$). Para resolver a questão de capacidade de armazenamento (segunda

imperfeição), é preciso incluir no modelo um novo conjunto de restrições, bem como um tipo adicional de variável de entrada (tamanho dos arquivos). A questão do particionamento (terceira imperfeição), poderia ser resolvida se as frações dos arquivos fossem considerados como arquivos independentes. Entretanto, tais soluções geram novas restrições e variáveis, aumentando as dimensões do problema.

Modelos mais completos são discutidos em [OHA79] e [CHE80].

Já foi visto que o modelo tende a crescer geometricamente com o número de arquivos e nós (fig. 3.5). A introdução de variáveis como capacidade dos nós, particionamento dos arquivos e outras, tornam os modelos mais complexos ainda, sendo impraticável tentar resolvê-los através dos métodos convencionais da pesquisa operacional. Quando as dimensões do problema ultrapassam a capacidade dos recursos de processamento disponíveis, uma das alternativas é usar métodos heurísticos. Tais métodos são discutidos na próxima seção.

3.3.4 Métodos heurísticos

Os métodos heurísticos são concebidos com base na experiência. Quem os formula conhece o comportamento do fenômeno, o que pode levar à solução ótima ou aceitável. Tais conhecimentos substituem o rigor matemático. Isso dimuiu a complexidade, adequando o problema aos recursos de computação disponíveis.

A alocação de (m) arquivos em (n) nós é similar ao problema clássico de localização de centros de distribuição de mercadorias ("warehouse location problem"). Em tal problema o objetivo é determinar o local onde o armazém deve ser instalado, de forma a minimizar os custos de transporte e de operação dos centros [WAG75]. Para tanto costuma ser formulado um modelo de progrmação linear inteira, sendo usado o algorítmo "branch-and-bound" para a determinação da solução ótima [WAG75]. Devido às dimensões que tais problemas costumam ter, métodos alternativos tem sido exaustivamente estudados por pessoas da área de pesquisa operacional [TAN81]. Dentre estes métodos destaca-se o "ADD", concebido po Kuehn e Hamburger em 1963 [TAN81].

O método heurístico "ADD" foi usado em [BOO79] e [TAN81] para resolver o problema de localização de concentradores em redes abertas de transmissão de dados. No trabalho aqui apresentado, está sendo sugerido que o método "ADD" seja usado para alocar (m) arquivos em (n) nós, de forma a determinar a topologia de menor custo de comunicação. Na próxima seção é apresentado um exemplo.

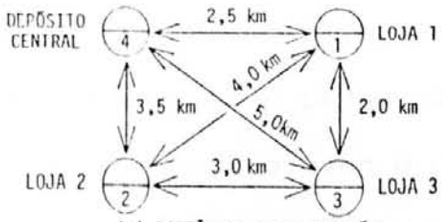
3.3.4.1 Exemplo de um método heurístico

O objetivo do método aqui descrito é obter a topologia de menor custo de comunicação. Tal método é composto de vários passos. Primeiro é feita uma lista com o volume de acessos (atualizações mais pedidos) aos arquivos. Tal lista deve ser classificada em ordem decrescente de acesso. Os arquivos são alocados nos nós segundo esta ordem. O primeiro

arquivo é alocado no nó 1. As ligações são estabelecidas e o custo calculado. Depois o arquivo é retirado do nó 1 e todas as ligações são eliminadas. A alocação então é feita no nó 2. Novas ligações são estabelecidas e o novo custo calculado. O arquivo é retirado do nó 2, as ligações são eliminadas e assim por diante até o último nó. O arquivo fica definitivamente alocado no nó que proporcionar o menor custo de comunicação. O mesmo procedimento é repetido para os demais arquivos. Há entretanto uma modificação. A partir do segundo arquivo, as ligações oriundas de alocações anteriores não podem ser eliminadas. Será apresentado a seguir, o uso deste método na aplicação de controle de estoques de uma rede de supermercados. Tal aplicação já foi descrita parcialmente na seção 3.2.

Há as lojas um, dois e três. Cada loja é representada por um nó. O nó tem o mesmo número da loja. O nó número quatro representa o depósito central. Os arquivos também são representados por números. O arquivo de produtos é o número um e o de pedidos o dois. As distâncias entre os nós, a matriz com o volume de acesso aos arquivos e a tabela com capacidade e custo das linhas de comunicação estão na fig. 3.6. Esses dados são usados para exemplificar o uso do método apresentado no parágrafo anterior.

O primeiro passo é fazer a lista com o volume de acesso aos arquivos e classificá-la em ordem decrescente. As informações desta lista estão na fig. 3.6(b). O próximo passo consiste em alocar os arquivos. Um exame na fig. 3.6(b) revela que o primeiro arquivo a ser alocado é o de número 1.



(a) DISTÂNCIA ENTRE OS NÓS

NÓ / ARQUIVO	1	2	3	4	TOTAL
PRODUTO (1)	334	450	254	84	1122
PEDIDOS (2)	24	50	8	126	208

(b) ACESSO AOS ARQUIVOS EM MENSAGENS POR HORA (MPH)

ELO / CAPACIDADE		300BPS OU 100MPS	600BPS OU 200MPS	1200BPS OU 400MPS	2400BPS OU 800MPS	4800BPS OU 1600MPS	9600BPS OU 3200MPS
NO	NÓS						
1	1-2	480	720	1120	1760	2560	3760
2	1-3	240	360	560	880	1280	1880
3	1-4	300	450	700	1100	1600	2350
4	2-3	360	540	840	1320	1920	2820
5	2-4	420	630	980	1540	2240	3290
6	3-4	600	900	1400	2200	3200	4700

(c) MATRIZ DE CUSTO DAS LINHAS DE COMUNICAÇÃO

Fig. 3.6 Dados do controle de estoque do supermercado

Todas as alternativas de alocação deste arquivo são apresentadas nas figuras 3.7(1a) até 3.7(1d). A alternativa de menor custo é a alocação do arquivo 1 no nó 3 conforme demonstrado na fig. 3.7(1c). Um novo exame na fig. 3.6(b) revela que o próximo arquivo a ser alocado é o de número 2. As opções de alocação estão nas figuras 3.7(2a) até 3.7(2d). A melhor opção é alocar o arquivo número 2 no nó 4 conforme demonstrado na fig. 3.7(2d). Com a alocação deste arquivo o procedimento é encerrado. O resultado é a topologia da fig. 3.7(2d). Na próxima seção é feita uma análise do método aqui apresentado.

3.3.4.2 Limitações do método apresentado

Seja (m) arquivos e (n) nós. No caso descrito no parágrafo anterior, m=2 e n=4. Foram analisadas 8 topologias (m*n). Entretanto, para garantir que o resultado fosse o de menor custo, seria preciso analisar 16 topologias (n**m). O método heurístico aqui proposto tem ainda uma desvantagem.

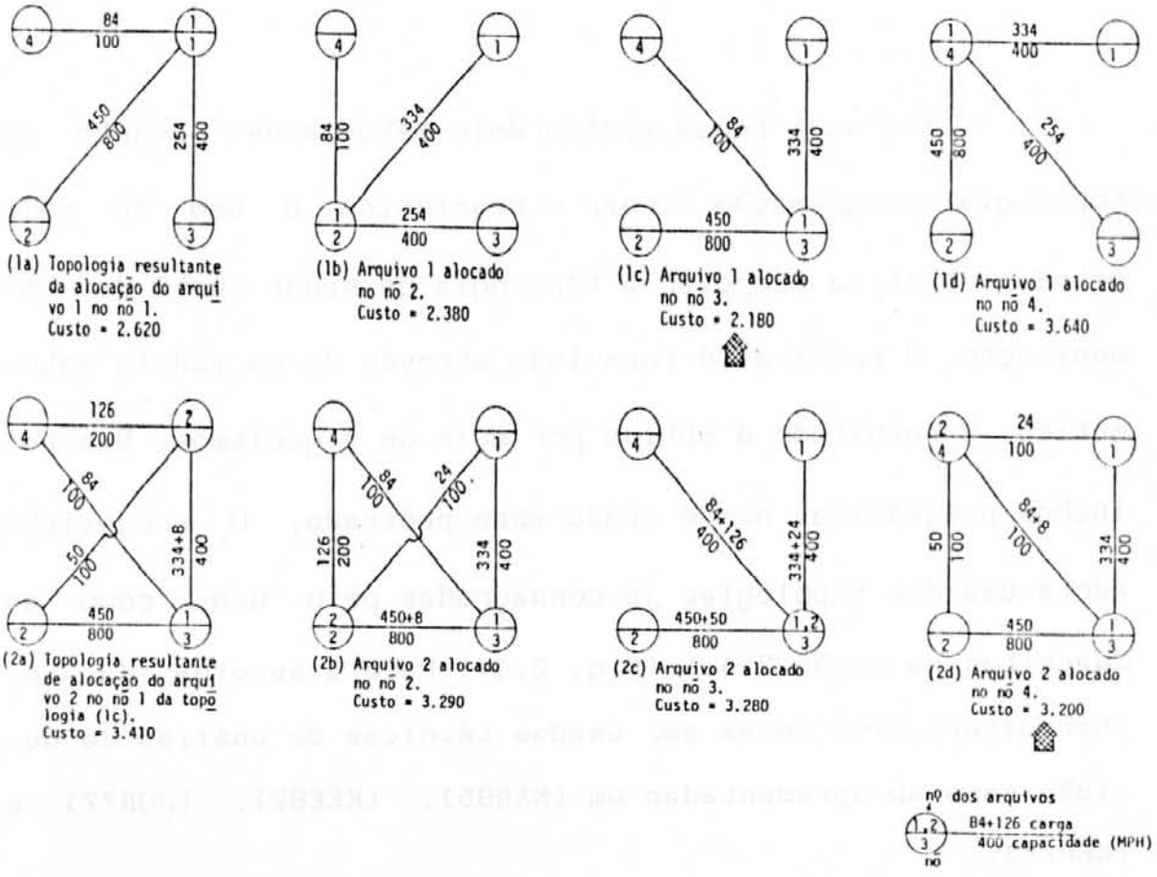


Fig. 3.7 Passos do método "ADD" para o projeto da topologia

Não leva em conta a possibilidade de múltiplas cópias dos arquivos. Se isso fosse considerado, a topologia de menor custo seria uma entre 64 alternativas ou seja (n**m)*(2**(m*(m-1))). Como é possível notar o uso do método nem sempre garante o resultado mais econômico. Apesar disso é possível chegar a resultados plenamente aceitáveis conforme ficou demonstrado na fig. 3.7.

Apesar da limitação vista anteriormente, os métodos heurísticos são um boa opção para o projeto de topologias com elevado número de nós e arquivos. Na bibliografia são apresentados vários desses métodos [BOO79], [TAN81], [PAL79], [GAV82] e [GAV85]. Uma outra maneira de projetar topologias é usar soluções pragmáticos.

3.3.5 Soluções pragmáticas

Até aqui foram vistos dois métodos de projeto da topologia: programação linear e heurística. O uso de tais métodos objetiva projetar a topologia de menor custo de comunicação. O problema é formulado através de um modelo matemático. O resultado é obtido por meio de algorítmos. Nas soluções pragmáticas não é usado esse processo. O projetista adota uma das topologias já consagradas pelo uso, como as descritas na seção 2.4.1 (fig. 2.4). Para a escolha da melhor alternativa podem ser usadas técnicas de análise de decisão como as apresentadas em [MAR85], [KEE82], [JON77] e [KEP77].

As topologias mais conhecidas são: (1) estrela e (2) hierárquica. Elas são apresentadas na fig. 3.8. Suas características, vantagens e limitações são descritas nas próximas seções.

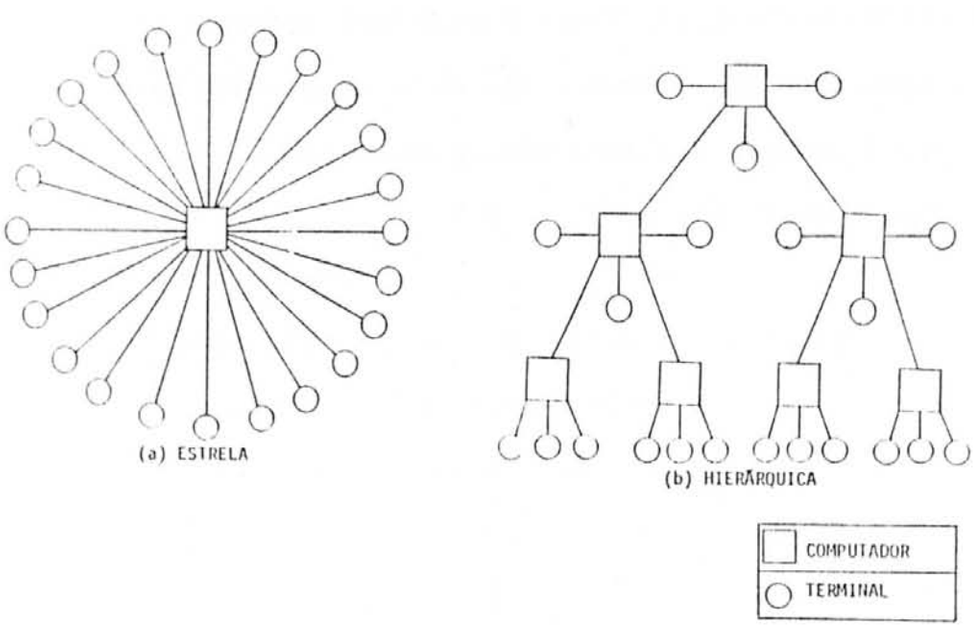


Fig. 3.8 Tipos clássicos de topologias

3.3.5.1 Topologia estrela

Na topologia em forma de estrela, todo o poder de processamento é centralizado. Apenas o acesso é distribuído através de terminais (fig. 3.8(a)). Tal topologia apresenta vantagens e limitações conforme segue.

Se destacam quatro vantagens. A primeira é a facilidade de implementação. Essa vantagem se deve a dois fatores: (1) de todas as topologia, a estrela é a mais usada [TAR86], sendo largamente conhecida e (2) o "software" de comunicação é relativamente simples, não apresentando problemas de compatibilidade de protocolo, uma vez que só há um computador envolvido. A segunda vantagem é a facilidade de controle. Como o processamento é centralizado o gerenciamento se torna mais simples. Além disso, na topologia estrela, a variedade de recursos ("hardware" e "software") é a menor possível. Com isso o número de fornecedores de tais recursos também é reduzido, o que facilita a solução de problemas operacionais. A terceira vantagem é a simplificação da administração de dados. Os bancos de dados são centralizados facilitando o controle da redundância, segurança e privacidade. A quarta vantagem é a capacidade de crescimento progressivo. Quando novos usuários necessitam de serviços de processamento, basta ligar novos terminais. Se o tempo de resposta aumentar, o computador é substituído por outro de maior capacidade.

A principal limitação é o alto risco de prejuízos devido a paralizações. Como só há um computador, todas as

atividades que dependem de processamento ficam prejudicadas quando o computador se torna inoperante. Tal limitação pode ser minimizada com o uso da topologia em forma de hieraquia, conforme será visto na próxima seção.

3.3.5.2 Topologia hierárquica

A topologia hierárquica tem a forma de uma árvore. O poder de processamento é distribuído pelos nós, exceto as folhas. Nelas ficam os terminais (fig. 3.8(b)). A topologia hierárquica geralmente é derivada da topologia estrela [BOOB2], [THI82]. Algumas vantagens são exploradas e as limitações diminuídas ou eliminadas. A principal vantagem é a capacidade de acompanhar o crescimento da empresa.

O crescimento das empresas é evolutivo e costuma passar por três estágios. O primeiro é aquele que segue sua fundação. Todas as funções são centralizadas num mesmo local. Tais funções são organizadas de maneira a ir absorvendo o aumento do volume de negócios. O consequente crescimento da empresa torna a estrutura organizacional obsoleta. É feita então uma reorganização. Com esta reorganização tem início o segundo estágio de crescimento. Os conceitos de matriz e filial são criados. Funções como contabilidade, finanças e compras permanecem centralizadas. A função vendas é distribuída. Os negócios continuam sendo expandidos através da proliferação de filiais. O grande número de filiais volta a inviabilizar a estrutura. Ocorre nova reorganização, o que origina o terceiro estágio de crescimento. O conceito de centro regional é criado. Parte das antigas atribuições da

matriz são distribuídas por tais centros. Permanecem centralizados o planejamento e controle dos negócios da empresa (fig. 3.9). A topologia hierárquica tem a vantagem de não ser substancialmente afetada por este tipo de crescimento. Basta que as práticas abaixo sejam observadas.

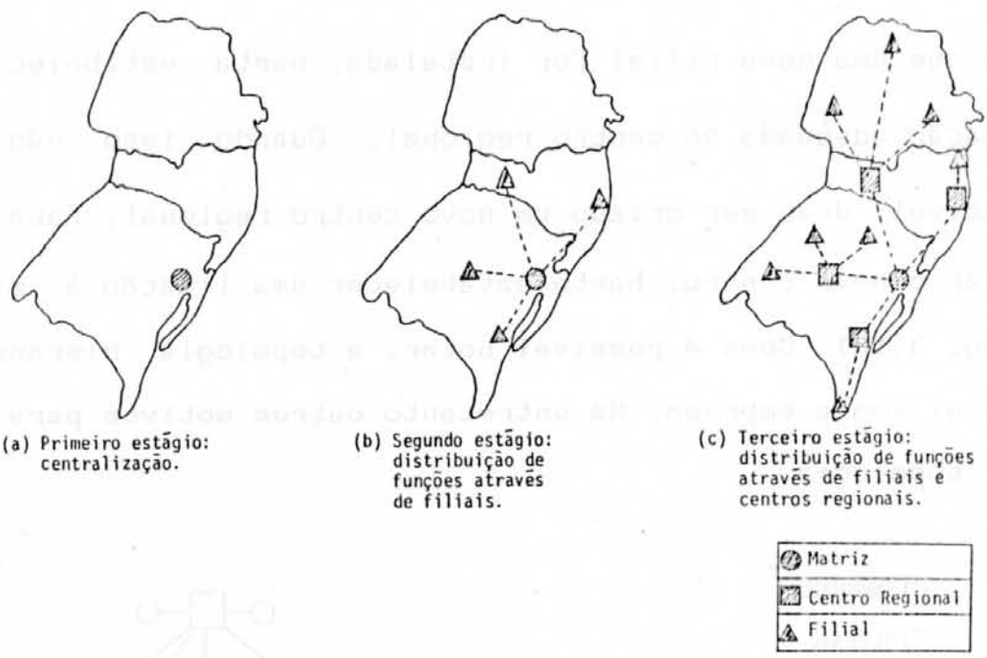


Fig. 3.9 Evolução das empresas

Durante o primeiro estágio de crescimento das empresas, o processamento costuma ser centralizado [GIB74], [NOL79]. A topologia é estrela. Na passagem para o segundo estágio, surge a oportunidade de evoluir para a topologia hierárquica. Devem ser criadas três classes de nós: matriz, filial e terminal. A matriz é a raiz. As filiais só podem ser ligadas à matriz. Os terminais são as folhas. A capacidade de processamento da matriz deve ser dimensionada para atender um número máximo de filiais. Toda vez que uma nova filial for instalada, basta estabelecer a ligação adequada à matriz. Quando isso não for mais possível, é provável que a empresa esteja entrando no terceiro estágio de crescimento.

No início do terceiro estágio, é necessário reprojetar a topologia novamente. Uma nova classe de nós chamada centros regionais deve ser criada. A topologia passa a ser uma árvore com até quatro níveis. A raiz é a matriz. Os centros regionais são ligados à matriz. As filiais são ligadas aos centros regionais. Os terminais são as folhas. Toda vez que uma nova filial for instalada, basta estabelecer a ligação adequada ao centro regional. Quando isso não for possível, deve ser criado um novo centro regional. Para instalar o novo centro, basta estabelecer uma ligação à matriz (fig. 3.10). Como é possível notar, a topologia hierárquica evolui com a empresa. Há entretanto outros motivos para usar tal topologia.

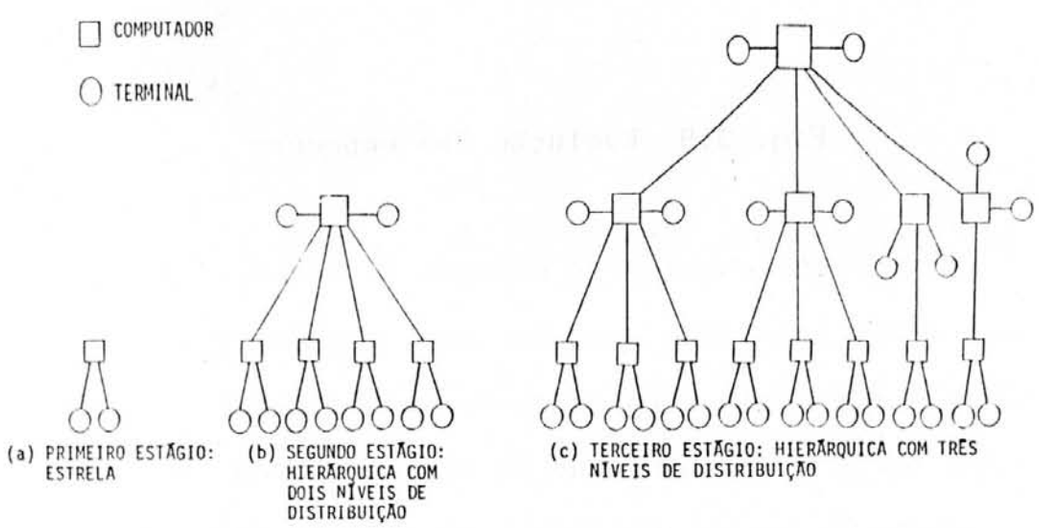


Fig. 3.10 Evolução das topologias com a empresa

No início desta seção foi visto que a topologia hierárquica costuma ser derivada da topologia estrela. Também foi afirmado que tal processo ocorre de forma a explorar as vantagens e diminuir ou eliminar as desvantagens. A principal limitação da topologia estrela é o risco de prejuízos devido à paralizações. Como existe apenas um computador, uma

eventual quebra prejudica todas as atividades que dependem de processamento. O uso da topologia hierárquica atenua os efeitos de uma eventual paralização. Como a capacidade de processamento é distribuída, um computador inoperante paraliza apenas parte do sistema. A maioria das atividades pode continuar sendo executada com os computadores restantes.

Até aqui as soluções pragmáticas se limitaram à topologias clássicas como estrela e hierárquica. Existe entretanto a alternativa de usar uma rede pública de comunicação de dados. No Brasil há a RENPAC - Rede Nacional de Pacotes [TAR86]. Um exemplo é apresentado na fig. 3.11. Outra solução para o projeto da topologia é combinar os vários métodos até aqui apresentados, conforme descrito a seguir.

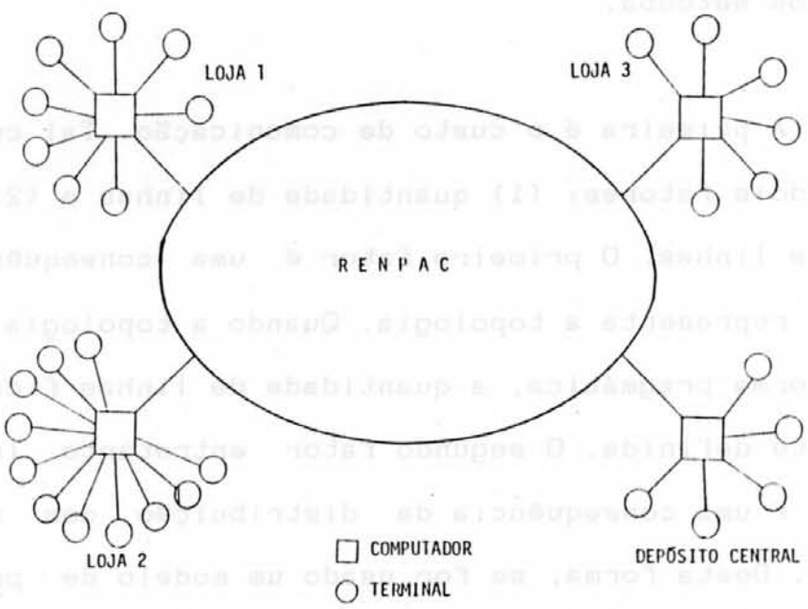


Fig. 3.11 Interligação de computadores através da RENPAC

3.3.6 Combinação de métodos

O modelo de programação linear descrito no seção 3.3.3, propicia três resultados: (1) o grafo que representa

a topologia, (2) a velocidade das linhas e (3) a distribuição dos arquivos pelos nós. Estes também são os resultados apresentados pelo método heurístico descrito na seção 3.3.4. Já o método pragmático da seção 3.3.5 só proporciona o primeiro resultado. Entretanto há meios de minimizar tal deficiência. Basta combinar os métodos conforme segue.

O projeto deve ser feito em duas partes. Primeiro a topologia é definida de forma pragmática. O resultado é um grafo. Em segundo lugar é aplicado um modelo de programação linear, para que os arquivos sejam distribuidos e as velocidades das linhas definidas. O grafo é um dado de entrada. Um modelo com tal abordagem pode ser encontrado em [OHA79]. Conforme descrito a seguir, há pelo menos três razões para combinar os métodos.

A primeira é o custo de comunicação. Tal custo depende de dois fatores: (1) quantidade de linhas e (2) velocidade das linhas. O primeiro fator é uma consequência do grafo que representa a topologia. Quando a topologia é adotada de forma pragmática, a quantidade de linhas fica automaticamente definida. O segundo fator entretanto independe do grafo. É uma consequência da distribuição dos arquivos pelos nós. Desta forma, se for usado um modelo de programação linear para distribuir os arquivos, é possível otimizar a velocidade das linhas e consequentemente o custo de comunicação.

A segunda razão é a complexidade. Conforme foi visto na seção 3.3.3.3, um modelo que ao mesmo tempo possi-

bilite projetar a topologia e distribuir os arquivos, pode se tornar um problema de grandes dimensões, necessitando de muitos recursos de computação para resolvê-lo. Se o problema for dividido em duas partes conforme anteriormente proposto, o modelo terá menos variáveis e restrições, ficando portanto mais simples chegar à solução.

A terceira razão é a necessidade de usar uma topologia previamente definida. Isso acontece quando a nova aplicação vai compartilhar o "hardware" de uma aplicação já existente. Como a topologia já está definida, só é possível racionalizar uma parte do custo de comunicação, ou seja a velocidade das linhas. Tal racionalização consiste em redistribuir os antigos arquivos juntamente com os novos, através de um modelo de programação linear.

Até aqui foi descrito como projetar a topologia. Na proxima seção será descrito como projetar a configuração básica.

3.4 Projeto da configuração básica

Conforme visto no capítulo 2, a configuração básica é um modelo do sistema de computação. Tal modelo tem o objetivo de mostrar quais os computadores que compõe o sistema, suas capacidades, onde estão situados e como são interligados (fig. 2.2).

O projeto da configuração básica é feito a partir da topologia e tem duas etapas: (1) geração de alternativas

e (2) escolha da melhor alternativa. A primeira etapa inicia com a seleção dos computadores a serem instalados em cada nó. Existem no mercado muitos computadores com capacidades similares. Para ocupar o mesmo nó, pode haver mais de uma opção. Consequentemente, a partir de uma mesma topologia é possível gerar várias configurações básicas. Como ilustração é dado um conjunto de configurações fictícias onde são reunidos computadores da LABO (mini 8043), SID (mini 5902-H), EDISA (supermicro ED690) e DIGIREDE (supermicro 8000). Tais configurações estão na fig. 3.12. Uma vez geradas as alternativas, tem início a segunda etapa.

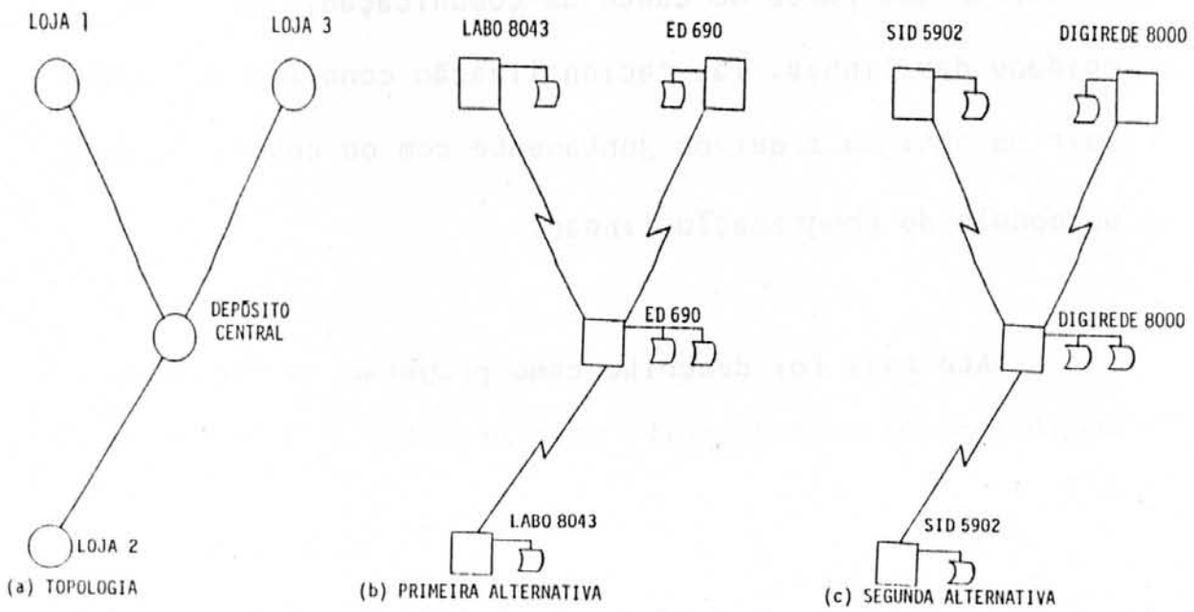


Fig. 3.12 Alternativas de configuração básica

A segunda etapa consiste em escolher a melhor alternativa. Há vários métodos que podem ser usados [MAR85], [KEE82], [JON77] e [KEP77]. No presente trabalho tais métodos não serão tratados. No restante desta seção será detalhado o processo de seleção dos computadores a serem alocados em cada nó.

Computadores diferentes tem reações diferentes sob a carga de uma mesma aplicação. Tal diferença pode ser visualizada num gráfico tempo de resposta versus número de terminais. Cada computador tem uma curva característica (fig. 3.13). Como será visto a seguir, tal curva pode ser usada para a seleção dos computadores.

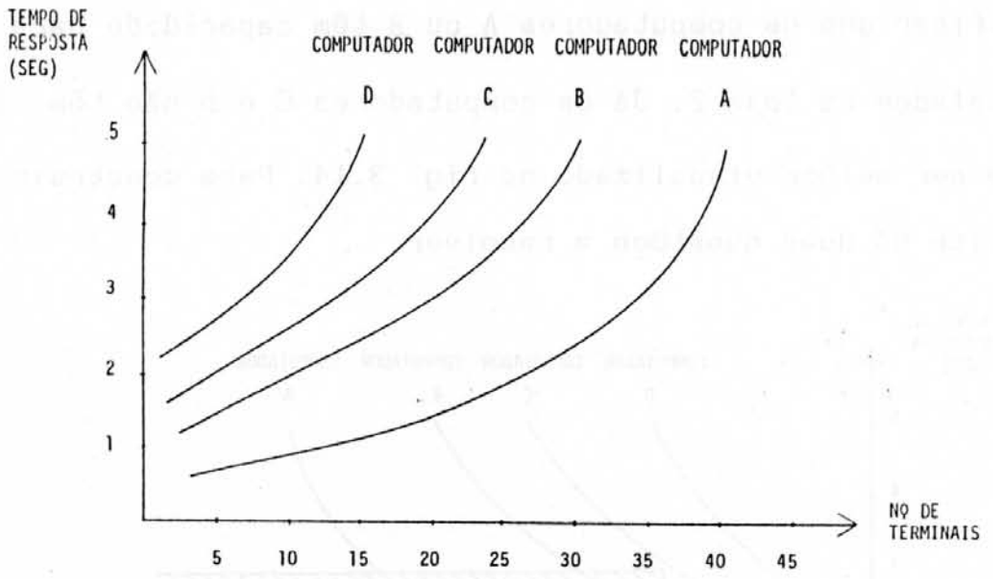


Fig. 3.13 Comportamento dos computadores sob a mesma carga

Um computador deve satisfazer à três condições para poder ser alocado num nó: atender os terminais do nó, atender os pedidos dos nós adjacentes e ter um tempo de resposta nunca superior ao especificado (normalmente 3 segundos). Há portanto três variáveis envolvidas: número de terminais, número de nós adjacentes e tempo de resposta. Se os computadores dos nós adjacentes forem considerados como terminais, é possível estabelecer as necessidades de processamento com apenas duas variáveis. Desta forma o processo de seleção pode ser feito com base no gráfico descrito no parágrafo anterior, conforme exemplo abaixo.

Seja a topologia da fig. 3.12(a): o computador da

loja 2 deve atender 12 terminais e os computadores da loja 3 e depósito central, sendo que o tempo de resposta não pode exceder a 3 segundos. Se os computadores dos nós adjacentes forem considerados como terminais, então o computador da loja 2 deve atender 14 terminais em 3 segundos no máximo. Ao colocar tais informações no gráfico da fig. 3.13, é possível verificar que os computadores A ou B têm capacidade para ser instalados na loja 2. Já os computadores C e D não têm. Isso pode ser melhor visualizado na fig. 3.14. Para construir tal gráfico há duas questões a resolver.

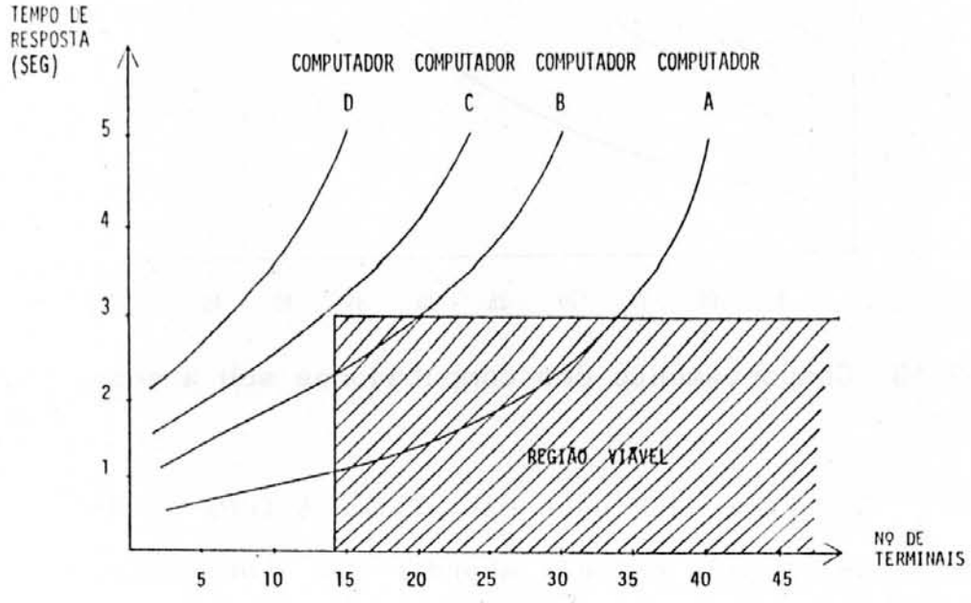


Fig. 3.14 Gráfico de seleção dos computadores

A primeira questão é como considerar os computadores dos nós adjacentes como sendo terminais. Tal questão pode ser resolvida, tornando os computadores equivalentes à um conjunto de terminais, que quando em operação geram taxas de pedidos idênticas à dos computadores. Por exemplo, se cada terminal faz 4 pedidos por minuto e os computadores dos nós adjacentes fazem 12 pedidos por minuto, então tais computadores podem ser considerados como sendo 3 terminais.

A segunda questão é como traçar as curvas características de cada computador. Há três maneiras de proceder: (1) simulação, (2) modelo analítico ou (3) "benchmark". Tais procedimentos são descritos nas próximas seções.

3.4.1 Simulação

Para fazer a simulação é preciso ter um modelo. Tais modelos podem ser encontrados em [BUC79], [STR80], [GRI82] e [STR84]. A seguir é apresentado um exemplo com base na descrição de [STR80].

3.4.1.1 Modelo de computador de grande porte

Foi considerado que o computador tem as seguintes características: suporta até (N) terminais; é multiprogramado, podendo processar até (M) transações simultâneamente; pode possuir até (D) canais de disco. O modelo está esboçado na fig. 3.15. Tal modelo funciona conforme segue.

Os terminais ativos enviam suas mensagens ao sistema e estas são colocadas na fila (F1) de acesso à memória. Cada mensagem corresponde a uma transação. Se o nível de programação (M) não estiver no limite máximo, a transação tem acesso à memória, sendo retirada da fila (F1) e colocada na fila (F2) à espera pela CPU. Assim que a CPU fique livre, ela é retirada da fila (F2) e começa a ser processada. Decorrido o tempo (C_0) o processamento é interrompido para que os dados sejam lidos. A CPU é liberada e a transação é colocada na fila (F3) à espera dos discos. Assim que um disco

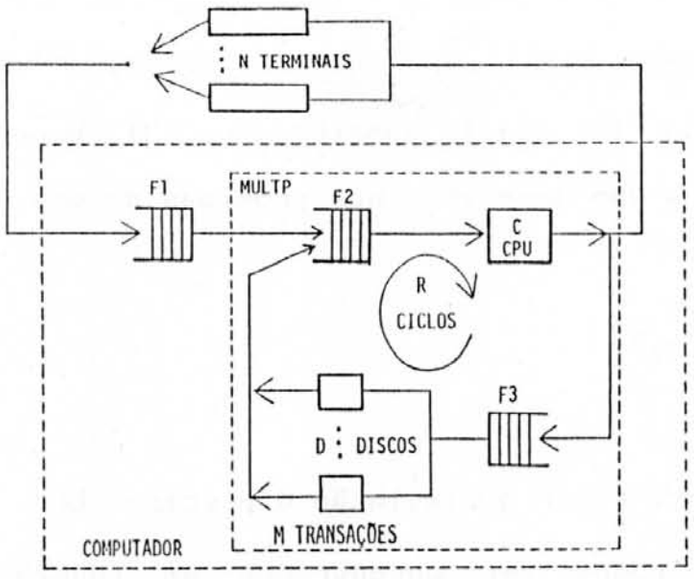


Fig. 3.15 Modelo de um computador de grande porte

(D) fique livre, ela é retirada da fila (F3) e os dados são lidos. Essa operação consome o tempo (i). A transação retorna então à fila (F2) e o ciclo se repete (R) vezes. Após o último ciclo os resultados são enviados ao terminal que disparou a mensagem. O usuário gasta o tempo (Z) para analisar os resultados, envia nova mensagem e todo o processo se repete.

Simulando o funcionamento do computador de acordo com a descrição acima, é possível traçar o gráfico tempo de resposta versus número de terminais.

3.4.1.2 Como traçar o gráfico

Para traçar o gráfico é preciso repetir a simulação várias vezes. No início deve ser considerado que só há um terminal ativo (N=1). A simulação é feita durante 10 minutos [STR80]. O objetivo é conseguir o tempo de resposta. Tal tempo é a diferença entre o instante do envio da mensa-

gem e o recebimento do resultado. Durante a simulação podem ser processadas muitas transações. Desta forma o valor obtido é um tempo de resposta médio, sendo este o número a ser registrado na coordenada do gráfico. Uma vez marcado o primeiro ponto é necessário repetir a simulação. O número de terminais deve ser aumentado de um (N=2) e a simulação executada novamente. Um novo tempo de resposta é obtido e o segundo ponto pode ser marcado no gráfico. A seguir o número de terminais deve ser aumentado (N=3), a simulação repetida e assim por diante. O processo deve parar quando a curva se tornar assintótica.

Em [STR80] e [BUC79] são apresentados resultados da simulação do modelo descrito anteriormente. Em [STR84] há orientação de como fazer a simulação, inclusive com ampla bibliografia sobre os "softwares" disponíveis para tal.

A simulação é uma maneira viável de selecionar computadores com capacidade adequada às necessidades dos nós [STR80]. Há entretanto algumas limitações conforme será visto a seguir.

3.4.1.3 Limitações da simulação

A simulação apresenta duas limitações. A primeira é a falta de modelos padronizados. Os diversos ramos da engenharia como a mecânica e a eletrônica, dispõe de modelos com os quais é possível prever o comportamento de um sistema ainda a nível de projeto. As facilidades não se limitam aos modelos. Os componentes mecânicos e eletrônicos disponíveis

no mercado, são especificados de acordo com tais modelos, facilitanto o projeto e a fabricação de produtos. A informática se ressente de tais práticas.

Ideal seria que os fabricantes fornecessem com seu produto um modelo e as especificações correspondentes. Tal modelo não é o esquema elétrico dos equipamentos. É o comportamento do computador sob a ação do sistema operacional, gerenciador do banco de dados e monitor de comunicações, conforme o exemplo descrito na seção 3.4.1.1. Juntando os dados da aplicação a tais especificações seria possível projetar adequadamente a configuração do "hardware". Isso entretanto não tem acontecido.

Os fabricantes não têm fornecido o modelo. Fornecem apenas alguns dados isolados. Para os discos há a latência e o "seek". Para a CPU há a velocidade em MIPS - milhões de instruções por segundo. Os dados do disco já são consagrados [WIE77], mas a controvérsia sobre a velocidade da CPU está aumentando [SER86], [ANO85]. Como será visto na próxima seção, há outros dados que não são fornecidos.

A falta de modelos padronizados e de dados adequados, tem três consequências. A primeira é o custo do projeto da configuração. Para modelar o comportamento dos computadores é preciso fazer pesquisa. Isso além de ser demorado exige mão de obra especializada. São poucas as empresas que têm recursos para suportar tais custos. A segunda consequência é o desbalanceamento dos sistemas em operação. Como a maioria dos computadores é instalada sem que a configuração seja

projetada, não é raro encontrar sistemas com gargalos (muito disco e pouca CPU e vice-versa), ou mesmo com excesso de capacidade para a carga aplicada. A terceira consequência é a impossibilidade de gerênciar os recursos de "hardware". Como os limites do sistema não são conhecidos, é impraticável se antecipar a aumentos de demanda, ou explicar aos usuários as ocasionais quedas de performance.

As questões discutidas anteriormente são consequência da falta de modelos padronizados. A outra limitação da simulação é o excesso de tempo para obter os resultados. Como foi visto na seção 3.4.1, a seleção dos computadores é feita com base no gráfico tempo de resposta versus número de terminais (fig. 3.14). Foi visto também que para traçar as as curvas características de cada computador é preciso ir repetindo a simulação à medida que for variando o número de terminais. Dependendo da quantidade de curvas a serem traçadas, esse processo pode demandar um tempo considerável.

Uma forma de dispender menos tempo é diminuir a precisão. Ao invés de recorrer à simulação para determinar todos os pontos da curva, é mais econômico usá-la na obtenção de alguns pontos significativos, completando o restante do traçado através de interpolação.

3.4.2 Modelo analítico

O modelo descrito na fig. 3.15 foi proposto em [BUC79] e emplamente analisado em [STR80]. Trata-se de um modelo de rede fechada, a partir do qual é possível obter várias equações. A equação de maior interesse ao presente trabalho é a que segue:

$$T = N * (C/U) - Z \qquad (I)$$

A partir de tal equação é possível construir o gráfico tempo de resposta versus terminais (fig. 3.14). Como foi visto em 3.4.1 tal gráfico é a base para a seleção dos computadores com a capacidade adequada aos nós. A seguir as variáveis da equação (I) são descritas.

(T) Tempo de resposta

É o lapso de tempo entre o envio da mensagem no terminal (o usuário aperta a tecla "enter") e a chegada do primeiro caracter da resposta.

(N) Número de terminais

É a quantidade de terminais suportados pelo computador, sob a carga de uma determinada aplicação.

(Z) Tempo de raciocínio

É o lapso de tempo entre o recebimento de uma mensagem e o envio da próxima. É um tempo médio e depende da aplicação.

As demais variáveis são (C) tempo total de CPU e (U) fator de utilização do computador. Tais variáveis são obtidas por meio de fórmulas mais complexas, sendo descritas nas próximas seções.

3.4.2.1 Tempo total de CPU

O tempo total de CPU é calculado pela fórmula abaixo.

$$C = C_0 * R \qquad (II)$$

onde:

(R) Ciclos de entrada e saída

É o número de vezes que a transação circula entre a CPU e os discos até completar o processamento.

(C_0) Tempo parcial de CPU

É o tempo dispendido pela CPU entre duas instruções de entrada e saída. É calculado pela seguinte fórmula:

$$C_0 = (1/(\mu_0 * S)) + t_0 + ((v_1 * M)/S) \qquad (III)$$

A fórmula (III) tem três fatores. O primeiro é o tempo de execução das instruções e depende da aplicação. Os dois últimos são devidos aos trabalhos necessários para o processador passar de uma tarefa à outra. Costuma ser chamado de tempo de chaveamento da tarefa ("time-per-task-switch"). As variáveis que compõe tais fatores podem ser dividi-

das em dois grupos: (1) dependentes da aplicação e (2) dependente da tecnologia dos processadores.

No primeiro grupo só há uma variável:

($1/\mu_0$) número de instruções executadas pela CPU entre duas operações de entrada e saída.

No segundo grupo há três variáveis:

(S) velocidade do processador em milhões de instruções por segundo (MIPS),

(t_0) é uma constante que depende da tecnologia da máquina e é fornecida em segundos,

(ν_1) é a quantidade de instruções que o processador executa para passar de uma tarefa à outra e

(M) é o nível de multiprogramação.

A seguir é descrito o fator de utilização da CPU.

3.4.2.2 Fator de utilização da CPU

O fator de utilização da CPU é calculado com fórmulas oriundas da teoria das filas. É um número que mede o quanto o computador está ocupado sendo definido no intervalo [0,1]. O limite inferior indica que o computador está sem carga e o superior que o computador está completamente lota-

do. Uma descrição mais detalhada sobre a origem das fórmulas que seguem pode ser encontrada em [BUC79] e [STR80].

$$U = U_{cpu} * U_1 \qquad (IV)$$

onde:

$$U_{cpu} = 1 - (1/((M!)*\sum_{k=0}^{M}((1/(M-k)!)*((C/i)**k)))) \qquad V$$

$$U_1 = 1 - (1/((N!)*\sum_{k=0}^{N}((1/(N-k)!)*((C_1/Z)**k)))) \qquad VI$$

As variáveis (M), (C), (N) e (Z) já foram descritas anteriormente. Falta definir (i) e (C_1).

(i) é o tempo dispendido na obtenção das informações do disco. Esse tempo tem duas parcelas: (1) tempo de acesso ("random-access-time") e (2) tempo de transferência. A primeira parcela é obtida pela soma dos tempos de "seek" e latência. A segunda é uma função da taxa de transferência (" transfer-rate"). Tais parcelas podem ser calculadas conforme [WIE77].

(C_1) é o tempo dispendido no processamento da transação, após ela ter deixado a fila F1 (fig. 3.15). É dado pela fórmula:

$$C_1 = (C / U_{cpu}) \qquad (VII)$$

A apresentação da fórmula (VII) completa a descrição do modelo analítico. A seguir é dado um resumo das va-

riáveis de entrada do modelo.

As variáveis são divididas em dois grupos. O primeiro é compostos por variáveis que devem ser fornecidas pelo fabricante do computador:

. modelo de filas (fig. 3.15),
. velocidade da CPU (S),
. níveis de multiprogramação (M),
. número de instruções de "overhead" devido à multiprogramação (v_1),
. valor da constante (t_0) e
. taxa de transferência, "seek" e latência para o cálculo de (i).

As variáveis do segundo grupo são aquelas que dependem da aplicação:

. número de instruções executadas pela transação em cada ciclo ($1/\mu_0$),
. número de ciclos de entrada e saída (R) e
. tempo de raciocínio (Z).

Uma vez fornecidas as variáveis descritas acima, basta usar as fórmulas (I) a (VII) para traçar o gráfico tempo de resposta versus terminais. Apesar deste processo ser mais rápido que a simulação, também apresenta limitações.

O modelo analítico tem as mesmas limitações da si-

mulação, exceto o tempo para obtenção dos resultados que pode ser menor. Há entretanto duas limitações adicionais. A primeira é a complexidade das equações. Quanto maior a quantidade de filas associadas ao modelo, tanto mais complexas se tornam as equações. Para modelos com muitas filas pode ser mais prático recorrer à simulação. A segunda limitação é a dificuldade de usar modelos analíticos para prever o resultado de situações anormais. Por exemplo, num banco, quando a fila cresce, o caixa passa a demorar mais tempo para atender os correntistas. Isso altera o tempo de raciocínio (Z). Em situações como esta a simulação é mais adequada.

Em [MEN85] são descritos outros modelos analíticos relacionados com a capacidade dos computadores.

Até aqui foram descritas duas formas de selecionar os computadores para os nós: simulação e modelos analíticos. A terceira forma é o "benchmark". Tal técnica é descrita na próxima seção.

3.4.3 "Benchmark"

A técnica de "benchmark" consiste em cronometrar a execução de um programa. Esta técnica é comumente usada para comparar o desempenho de dois ou mais computadores.

Conforme já foi visto, a seleção de computadores com capacidade adequada aos nós pode ser feita com base nos gráficos tempo de resposta versus terminais (fig. 3.14). Tais gráficos podem ser construidos através de "benchmark".

Há duas formas de proceder. Estas formas são descritas a seguir.

A primeira é cronometrar a aplicação real. As transações mais significativas (as que consomem 80% do tempo) são programadas, compiladas e instaladas. As mensagens são disparadas nos terminais de acordo com o tempo de raciocínio característico da aplicação. O tempo de resposta é cronometrado e registrado. Em intervalos regulares (por exemplo 10 minutos), a média é calculada e anotada no gráfico. Tal procedimento é repetido para 2,3,...N terminais, até a curva se tornar assintótica, indicando que o gráfico está concluído.

A vantagem deste tipo de "benchmark" é a precisão dos dados levantados. Há entretanto duas desvantagens. A primeira é o excesso de tempo dispendido. A segunda é o custo. Para realizar tal "benchmark" é preciso montar um cenário, instalar equipamentos, ter o "software" aplicativo pronto, contratar cronometristas, contar com colaboração dos usuários na operação dos terminais e proceder a uma série de ensaios antes de iniciar as cronometragens. Este procedimento é caro e difícil de realizar.

Uma variação do procedimento descrito acima é o "benchmark" sintético. Os procedimentos podem ser os mesmos. A diferença é que as transações não são reais e sim emuladas.

Uma transação emulada deve consumir os mesmos re-

cursos que a sua congênere real. Por exemplo, uma transação bancária de desconto de cheque implica em: (1) obter no terminal o número da conta, número do cheque e o respectivo valor, (2) ler um registro do disco, (3) subtrair o valor do saldo, (4) regravar o registro, (5) gravar os dados do cheque no disco e (6) enviar ao terminal uma mensagem indicando ao caixa o sucesso ou insucesso da operação. A transação emulada funcionaria como segue: (1) obter 20 caracteres no terminal, (2) ler 200 bytes do disco, (3) executar 100 instruções de máquina, (4) gravar 200 caracteres no disco, (5) gravar mais 20 caracteres no disco e (6) enviar 20 bytes ao terminal. Como é possível notar, as duas transações têm comportamentos semelhantes, consomem os mesmos recursos e levam praticamente o mesmo tempo para serem executadas. Desta forma é possível obter dados com boa precisão, sem a necessidade de programar todos os detalhes das transações. Essa é uma das razões porque a emulação é a alternativa normalmente usada para fazer "benchmark" [ANO85].

A segunda forma de construir o gráfico tempo de resposta versus terminais (fig. 3.14) através de "benchmark", é usar ferramentas automatizadas. O processo pode ser semelhante à cronometragem descrita anteriormente. A diferença é que não há terminais, operadores nem cronometristas. As transações são disparadas randomicamente por um programa que também registra o tempo de resposta. No final os dados do gráfico são impressos. Tal ferramenta é mais conveniente que a cronometragem, apresenta resultados com precisão semelhante, é mais simples de executar e tem um custo menor.

Até aqui foi visto como projetar a configuração básica. Foram descritas três alternativas: simulação, modelos analíticos e "benchmark". Na próxima seção será descrito como projetar a configuração final.

3.5 Projeto da configuração final

Conforme visto no capítulo 2, a configuração final é uma planta do sistema de computação, onde estão descritos equipamentos e meios de comunicação os quais estão disponíveis no mercado. Tal configuração é resultado de um refinamento feito na configuração básica em três etapas: (1) comprovação da capacidade dos computadores, (2) comprovação da capacidade das linhas e (3) escolha dos meios comunicação. Essas etapas são descritas a seguir.

3.5.1 Comprovação da capacidade dos computadores

O objetivo desta etapa é medir a capacidade real dos computadores.

Conforme descrito na seção 3.4, a seleção dos computadores com a capacidade adequada aos nós é feita através de uma das seguintes técnicas: simulação, métodos analíticos ou "benchmark". A melhor precisão é do "benchmark". Dependendo da técnica usada, o computador pode ter sido selecionado com base numa capacidade teórica. Desta forma, antes de adquirir os equipamentos, é recomendável saber qual a sua capacidade real.

A capacidade real pode ser obtida através de "benchmark", conforme descrito em 3.4.3. As medições devem ser feitas num sistema real, incluindo unidades controladoras e processadores de comunicação, bem como sistema operacional, gerenciador de banco de dados e monitor de comunicação. Com isso o resultado obtido será o mais preciso possível.

Conhecer o comportamento real de um sistema antes de usá-lo, é prática comum nos vários ramos da engenharia. Prédios, pontes, automóveis e aviões são alguns dos produtos que passam por inúmeros testes antes de serem liberados ao uso. Dada a crescente influência do processamento eletrônico de dados na vida das pessoas, os sistemas de computação também devem ter sua eficácia assegurada entes de entrar em funcionamento. Esta é a razão porque um computador precisa ter sua capacidade comprovada, para poder ser integrado à configuração final.

3.5.2 Comprovação da capacidade das linhas

O objetivo desta etapa é verificar se as linhas foram dimensionadas com as capacidades adequadas.

Conforme visto em 3.3, a capacidade das linhas de comunicação são definidas quando a topologia é projetada. Tal capacidade tem um limite inferior, chamado velocidade mínima. Essa velocidade reflete as necessidades da aplicação, sendo calculada pela seguinte fórmula:

$$V = (((Ms + Mr) * Nc) / Tr) * K$$

onde

Ms = tamanho médio da mensagem de solicitação (em caracteres),

Mr = tamanho médio da mensagem de resposta (em caracteres),

Nc = unidades de informação por caracter (em bits),

Tr = tempo de resposta e

K = fator de correção.

Como é possível notar, a velocidade mínima é corrigida pelo fator (K). Quando a topologia é projetada, tal fator tem um valor aproximado, pois não são conhecidos os detalhes de processamento e comunicação. Uma vez definida a configuração básica, o valor de (K) pode ser calculado com maior exatidão. Comparando posteriormente ambos os valores, é possível verificar se as linhas foram dimensionadas com as velocidades adequadas. O valor de (K) pode ser obtido pela fórmula abaixo.

K = (R / VLD)

onde:

R = velocidade da linha (bps) e

VLD = vazão líquida dos dados.

Os procedimentos de obtenção da vazão líquida dos dados (VLD) são descritos na próxima seção.

3.5.2.1 Vazão líquida dos dados

Por recomendação da ANSI a vazão líquida dos dados (VLD) é definida como:

$$VLD = \frac{\text{número de bits aceitos no destino}}{\text{tempo necessário para torná-los aceitos}}$$

De acordo com [DOL78],

```
VLD=(K1*(M-C)*((1-K2*K3*E)**M))/((M/R)+DT)
```

onde:

K1 = unidades de informação por caracter (bits/caracter),
M = tamanho da mensagem (em caracter),
C = quantidade de caracteres de controle,
K2 = unidades de informação por caracter, incluindo bits "start", "stop" e paridade (bits/caracter),
K3 = distribuição dos erros,
E = taxa de erro da linha,
R = velocidade da linha (bps) e
DT = intervalo entre blocos.

As variáveis (C), (K3), (E) e (DT) são detalhadas a seguir.

(C) Quantidade de caracteres de controle

Os caracteres de controle são devidos ao protocolo. Por exemplo, para transmitir uma mensagem em BSC1 [IBM70], são necessários 11 carácteres conforme segue.

```
S S S                E C C P          S S A P
Y Y T < T E X T O > T R R A ------> Y Y C A
N N X                B C C D          N N K D
```

(K3) Distribuição dos erros

É definido como

$$K3 = \frac{\text{total de rajadas de erro}}{\text{total de bits afetados}}$$

Uma regra pragmática: R=9600 ===> K3=0,2,
R=4800 ===> K3=0,3,
R=2400 ===> K3=0,4 e
R=1200 ===> K3=0,5.

(E) Taxa de erro da linha

É definida como

$$E = \frac{\text{quantidade de bits errados}}{\text{total de bits transmitidos}}$$

De acordo com [TAR84], a taxa de erro aceitável é 10**(-5).

(DT) Intervalo entre blocos

É o tempo dispendido entre o envio do último bit de uma mensagem e o primeiro bit da próxima. Este tempo depende do protocolo e do regime de correção dos erros de transmissão [DOL78]. Por exemplo, sejam dois computadores se comunicando via satélite, num canal "half-duplex", com protocolo BSC1:

$$DT = Tpre + Tme + Tpe + Tprr + Tmr + Tpr$$

onde

Tpre = tempo de processamento para enviar a mensagem,

Tme = tempo para reverter a direção do modem ("turnaround time"),

Tpe = tempo de propagação da mensagem enviada,

Tprr = tempo de processamento para receber a mensagem,

Tmr = tempo para reverter novamente a direção do modem e

Tpr = tempo para acusar o recebimento da mensagem (ACK).

Uma vez determinada a VLD, o novo valor do fator (K) pode ser obtido. Este fator deve ser calculado para cada linha. Se o novo valor for maior que o usado no projeto da topologia, as linhas estão com as capacidade subdimensionadas.

3.5.3 Escolha dos meios de comunicação

Nesta etapa são escolhidos os canais de comunicação, modems e multiplexadores. O que deve nortear tal escolha são os fatores que afetam o intervalo entre blocos (DT), descritos na seção anterior. Quanto maior o valor de DT, menor a vazão líquida dos dados (VLD). Desta forma os canais "full-duplex" são preferidos aos "half-duplex", os melhores modems são aqueles com baixo "turnaround" e os melhores multiplexadores são aqueles capazes de atender todas as linhas com um mínimo de retardo. Detalhes sobre os meios de comunicação podem ser encontrados em [TAR84], [DOL78] e [TAN81].

3.6 Resumo

O projeto do sistema de computação é executado de cima para baixo ("top-down"), em níveis crescentes de complexidade. Há três níveis: (1) topologia, (2) configuração básica e (3) configuração final.

O primeiro nível a ser projetado é a topologia. O objetivo é minimizar o custo de comunicação. Tal custo depende da distribuição dos arquivos pelos nós. Há três opções de projeto. A primeira consiste em usar modelos de programação linear. Esta opção proporciona os melhores resultados. Entretanto, a quantidade de variáveis e restrições que compõe os modelos, cresce geometricamente com o número de nós e arquivos, havendo um limite acima do qual o custo do projeto é muito alto. A segunda opção se refere ao uso de métodos heurísticos. A precisão nem sempre é a mesma da opção ante-

anterior, mas é possível conceber topologias com maior número de nós e aquivos. A terceira opção consiste em adotar soluções pragmáticas. A topologia é concebida a partir de tipos clássicos como os modelos estrela ou hierárquico. Nesta opção não há garantia do menor custo de comunicação. Apenas são exploradas as vantagens que tais topologias oferecem, como por exemplo a possibilidade de crescimento progressivo.

O segundo nível a ser projetado é a configuração básica. O objetivo é selecionar o computador mais adequado às necessidades de cada nó. Há três formas de proceder. A primeira consiste em fazer simulação. Para usar esta técnica é preciso conceber modelos de filas, que representem o funcionamento dos computadores. A precisão dos resultados depende de quão fiel a realidade é representada nestes modelos. A segunda opção se refere ao uso de modelos analíticos. Os resultados podem não ser tão precisos como a simulação, mas são obtidos mais rapidamente. A terceira opção consiste em fazer "benchmark". Os resultados são precisos, mas a técnica tem um custo elevado, dispende muito tempo e é de difícil realização.

O terceiro nível a ser projetado é a configuração final. O objetivo é dar forma definitiva ao sistema de computação. Isso é feito em três etapas. A primeira é a comprovação da capacidade dos computadores a serem instalados em cada nó. Tal comprovação pode ser feita através de "benchmark". O computador deve estar completo: cpu, unidades de controle, processadores de comunicação, sistema operacional, gerenciador de banco de dados e monitor de comunicação. A

segunda etapa se refere à comprovação da velocidade das linhas. Essa comprovação é feita com base na vazão líquida de dados (VLD). A VLD é calculada em função dos canais de comunicação a serem usadas ("full-duplex" ou "half-duplex"), protocolo, tempos de "turnaround" dos modems, regime de controle de erros de comunicação e outros. A terceira etapa consiste em selecionar os modems e os multiplexadores. O que deve nortear tal processo são os retardos que tais equipamentos introduzem na transmissão, diminuindo a VLD. Devem ser preferidos os equipamentos de menor retardo.

Neste capítulo foram descritas as etapas do projeto do sistema de computação. Como foi visto, o projeto está baseado em dados da aplicação e dos fornecedores de equipamentos e meios de comunicação. No próximo capítulo será descrito como obter, organizar e integrar estes dados.

4 DESENVOLVIMENTO DA METODOLOGIA DE PROJETO

4.1 Introdução

Os objetivos deste capítulo são: propor uma metodologia de desenvolvimento de sistemas distribuídos e integrar o projeto do sistema de computação descrito no capítulo anterior à esta metodologia.

Como exemplo é apresentado o seguinte procedimento: uma operadora de PBX atende o telefone, pergunta o nome de quem está sendo chamado, procura o tal nome numa lista, acha o número do respectivo ramal e completa a ligação. Há pelo menos duas formas de racionalizar este procedimento. A primeira é totalmente manual. Sobre a mesa da telefonista é colocado um fichário com os nomes e ramais. O fichário é ordenado alfabeticamente para facilitar a consulta. A segunda é automatizada. A lista está num computador, que é capaz de achar o ramal do nome teclado pela telefonista, após excecutar um milhão de instruções. Se a opção escolhida for a segunda, é preciso instalar um sistema de computação.

Supondo que para armazenar a lista são precisos 10 megabytes e que uma telefonista pode atender no máximo 100 chamadas por hora: se a taxa horária de telefonemas for 50 por hora, é preciso apenas uma telefonista e o sistema de computação pode se resumir a um microcomputador. Para que o tempo de resposta fique em torno de três segundos, a velocidade de processamento deve ser de aproximadamente 0,33 MIPS. Entretanto, se a taxa de telefonemas for 400 por hora, um

microcomputador pode não ser a solução. O número de telefonistas sobe para quatro, pelo menos. Com isso são necessários quatro teclados e quatro vídeos. A CPU deve ser mais rápida para poder suportar a carga maior. Caso se trate de uma empresa dispersa geograficamente as opções se multiplicam.

Supondo que a empresa esteja dispersa por 3 localidades: para atender a mesma taxa de telefonemas (400 por hora), poderiam ser necessárias seis telefonistas. Basta que a taxa esteja assim dividida: 320, 40 e 40 para a primeira, segunda e terceira localidades respectivamente. Isso exige seis teclados e seis vídeos. Quanto à CPU, passa a existir uma solução de compromisso entre o sistema centralizado (uma única CPU) e o distribuído (duas ou três CPUs interligadas). Como é possível notar, apesar da aplicação ser a mesma (consulta à lista telefônica), há uma série de outras variáveis que determinam a magnitude do sistema de computação. Essas variáveis são o "software" da aplicação, as tarefas executadas pelas pessoas, a distribuição geográfica da empresa e a intensidade das relações da empresa com o meio ambiente (por exemplo taxa de chamadas). Por esta razão, neste trabalho está sendo proposta uma metodologia de projeto de sistemas distribuídos, onde todas essas variáveis estão integradas. Na bibliografia não foi encontrada uma metodologia com tal característica.

Foi constatada a existencia de três grupos de metodologias. No primeiro grupo estão aquelas voltadas exclusivamente ao desenvolvimento de "software" [HIC78], [IFI82].

Em tais metodologias está pressuposto que o computador já está instalado. Portanto não incorporam o projeto da configuração. Mesmo o ciclo de vida estruturado [YOU82] é bem mais voltado para o "software", passando rapidamente pela questão do "hardware".

O segundo grupo é composto por metodologias de análise de desempenho do "hardware". Este grupo pode ser dividido em dois tipos. No primeiro tipo estão as metodologias voltadas à produtividade [SAA80], [MEN85]. Não são próprias para o projeto da configuração. São mais indicadas para examinar o comportamento de sistemas já instalados. Normalmente são eficazes quando é preciso fazer sintonia fina ("tuning") ou aumentar a capacidade de sistemas existentes. Estão baseadas em sistemas centralizados. O segundo tipo é composto por metodologias voltadas ao projeto da configuração [BUC79], [STR80], [AGA85]. Conforme foi visto no capítulo anterior, não são metodologias completas pois não especificam como obter e organizar os dados para a execução do projeto.

O terceiro tipo são as metodologias de racionalização e organização do trabalho. Estas metodologias estão centradas nas pessoas e nos intrumentos de execução das tarefas. Permitem fazer organogramas, dimensionar mão de obra, elaborar o arranjo físico ("layout"), projetar formulários, projetar métodos de execução das tarefas, estipular tempos para a execução dos trabalhos, confecionar manuais de políticas e procedimentos e outros. Tais metodologias são oriundas das área de Organização e Métodos (O&M) [ARA85] e Enge-

nharia Industrial (EI) [MAY70a], [BAR77].

Além desses três grupos de metodologias, também foram encontradas dispersas na bibliografia algumas técnicas tais como análise estruturada [DEM78], [WEI80], [GAN83], modelagem de banco de dados relacional [COD70], [COD79], organização dos dados em entidades e relacinamentos [CHE77], normalização [KEN83], projeto estruturado [YOU78], [STE85], programação estruturada [GEL77], [MAR77], [HOL77] e outros. Estas técnicas, integradas a algumas partes dos três grupos de metodologias descritas anteriormente, formam a metodologia de desenvolvimento de sistemas distribuídos que é descrita mais adiante neste capítulo.

A metodologia a ser proposta está baseada no ciclo de vida estruturado proposto por Yourdon [YOU82]. Por esta razão esse ciclo é apresentado a seguir. Os leitores já familiarizados com o assunto podem ignorar a próxima seção, sem prejudicar o entendimento do restante do capítulo.

4.2 Ciclo de vida do projeto

Em muitas empresas o sistema é analisado, projetado e implementado de uma maneira informal. Em outras, todavia, há regras a serem seguidas. As comunicações entre usuários, gerentes e a equipe de projeto deve ser documentada. Todos os envolvidos precisam estar conscientes das diversas etapas a serem cumpridas para que o projeto seja levado a cabo. Essas etapas, também chamadas fases e atividades, são executadas numa determinada sequência, estabelecendo o ciclo

do vida do projeto [YOU82].

Antes do advento das técnicas estruturadas, a maioria dos projetos era executado segundo o ciclo de vida clássico. Esses ciclos de vida tem duas características [YOU82]: (1) o sistema é implementado de baixo para cima ("bottom-up") e (2) as fases tendem a ser executadas em sequência, de forma que o término da anterior é pré-requisito para o início da próxima. O ciclo de vida que serviu de base para a metodologia apresentada em [HIC78], é uma das muitas versões do ciclo clássico (fig. 4.1).

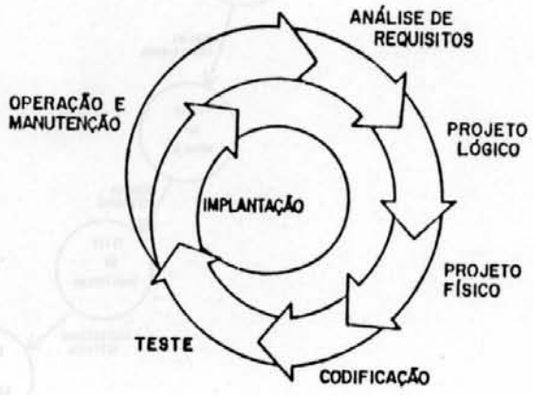


Fig. 4.1 Ciclo de vida dos sistemas

Segundo [YOU82], a maioria das metodologias de desenvolvimento de sistemas seguem variações do ciclo clássico conforme apresentado na fig. 4.2. Entretanto, nos últimos anos, algumas empresas passaram a adotar técnicas como projeto estruturado, programação estruturada e desenvolvimento de cima para baixo ("top-down"). Isso originou um novo ciclo de vida que foi chamado por Yourdon [YOU82] de semi-estruturado. Este ciclo é apresentado na fig. 4.3.

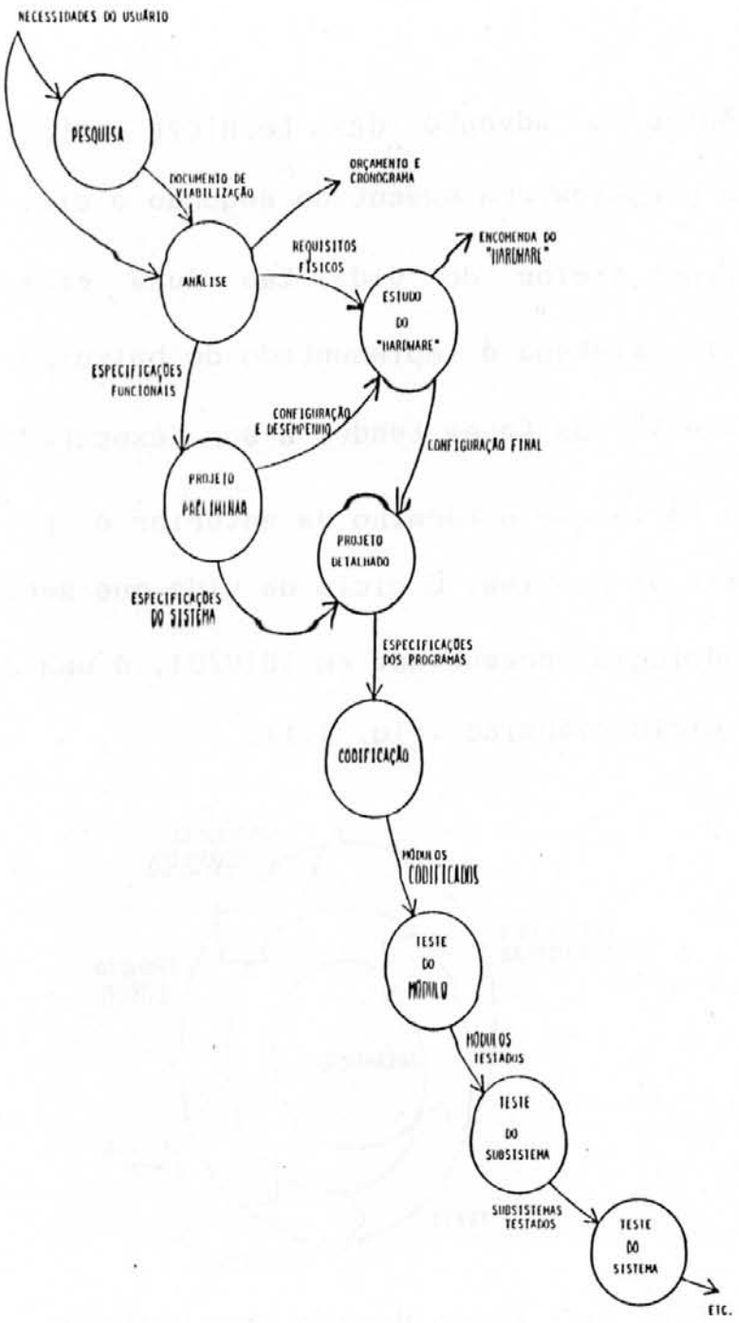


Fig. 4.2 Ciclo de vida clássico

Com o objetivo de integrar as técnicas estruturadas existentes e permitir que as várias fases do projeto possam ser executadas em paralelo, Yourdon [YOU82] propôs o ciclo de vida estruturado.

4.2.1 Ciclo de vida estruturado

O ciclo de vida estruturado tem nove atividades e todas podem ser executadas em paralelo. A palavra fase não

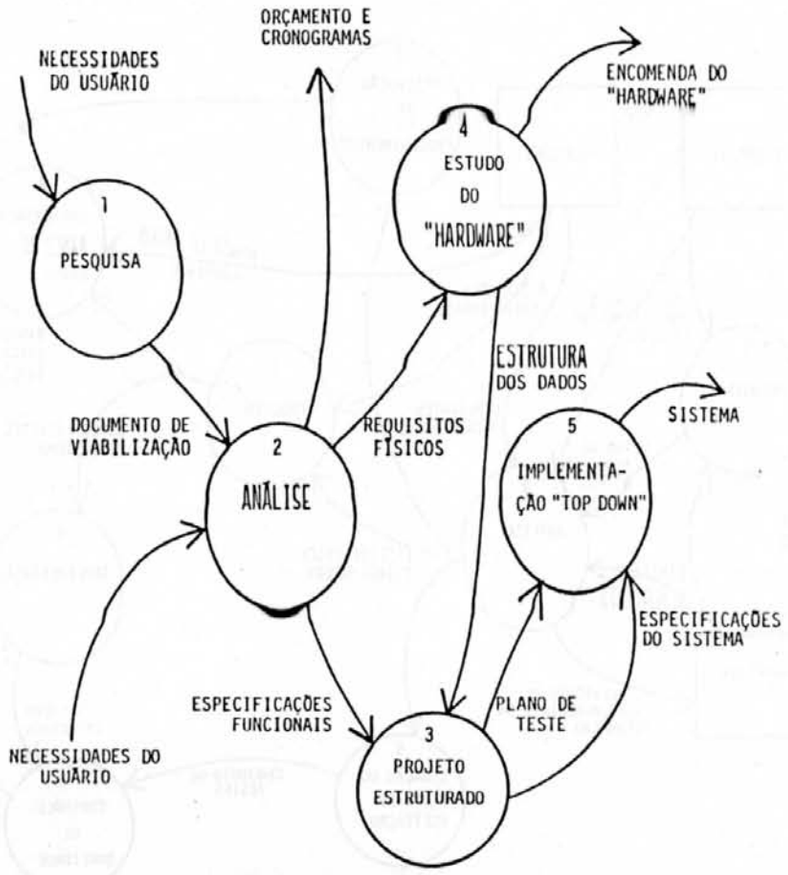


Fig. 4.3 Ciclo de vida semi-estruturado

foi usada porque induz à idéia de sequência [YOU82]. Por isso foi substituída por atividade.

Uma metodologia é um sistema para fazer sistemas [YOU82]. Por isso, o ciclo estruturado pode ser apresentado segundo a notação da análise estruturada. Yourdon o apresentou usando a simbologia proposta por Tom DeMarco [DEM78], conforme consta na fig. 4.4. Os terminadores e as atividades são descritas a seguir.

4.2.1.1 Terminadores

Terminadores ("source or sink") são entidades externas ao sistema, podendo ser pessoas, organizações ou outros sistemas de onde se originam as entradas e para onde se

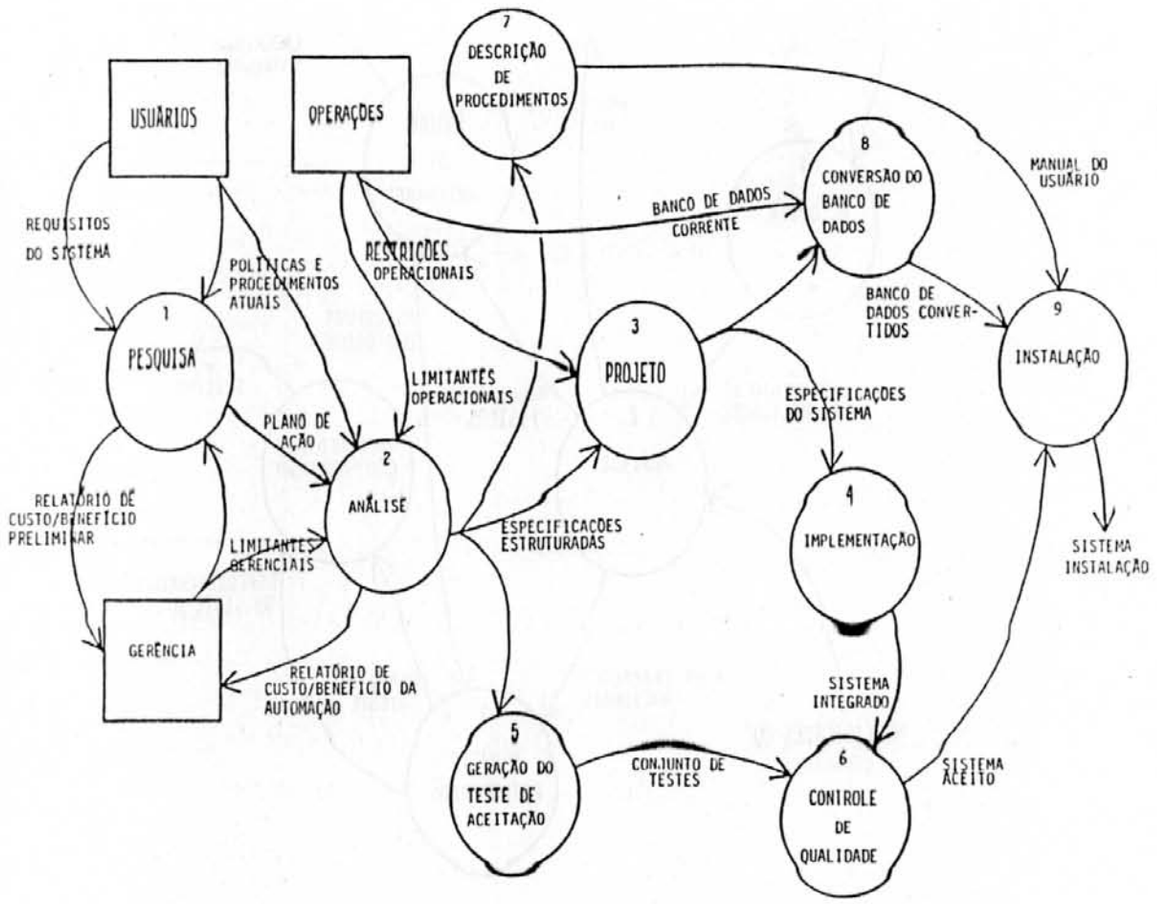


Fig. 4.4 Ciclo de vida estruturado

destinam as saídas. O ciclo de vida estruturado tem três terminadores (fig. 4.4): (1) usuários, (2) gerentes e (3) operações.

O primeiro representa os usuários, os quais formam três classes. A primeira classe é composta pelos usuários do nível estratégico da empresa. São pessoas preocupadas com a lucratividade a longo prazo ou com o retorno sobre o investimento a ser gerado pelo sistema proposto. Normalmente esta classe de usuários está envolvida com a atividade 1 (pesquisa) e com o início da atividade 2 (análise). A segunda classe é composta pelos usários do nível tático. Estas pessoas supervisionam as operações do dia-a-dia, tem boa visão da parte do negócio sob sua responsabilidade e tem rígidos orçamentos a cumprir. Por esta razão tentam obstruir o contado

dos analistas com seus subordinados para evitar "perda de tempo". A terceira classe é composta pelos usuários do nível operacional. Estas pessoas executam as operações do dia-a-dia. Quando o sistema está sendo desenvolvido, a maior parte dos contatos é feita com esta classe de usuários.

O segundo terminador representa a gerência. A gerência é uma pessoa ou grupo que aprova (e controla) o orçamento do projeto e estabelece restrições organizacionais. Por exemplo, um tipo de restrição seria a obrigatoriedade de todos os sistemas serem desenvolvidos para computadores do fabricante escolhido como padrão.

O terceiro terminador representa o setor de operações. Este setor formula padrões como tamanho máximo de programas, tipo de "software" básico a ser usado e outros, sendo mais comum nos sistemas centralizados.

4.2.1.2 Atividades

Conforme já foi visto, o ciclo de vida estruturado tem nove atividades (fig. 4.4). Tais atividades são resumidas abaixo.

Atividade 1 - Pesquisa: Esta atividade é também conhecida como estudo de viabilidade ou estudo inicial. Normalmente tal atividade inicia, quando o usuário solicita a automação de uma ou mais partes do negócio sob sua responsabilidade. São então identificadas as deficiências no ambiente de trabalho atual e estabelecidos novos objetivos. Após a

determinação da viabilidade ou não de automatizar a parcela do negócio, é formulado um plano de ação ("project chart") para levar o projeto a cabo (fig. 4.5).

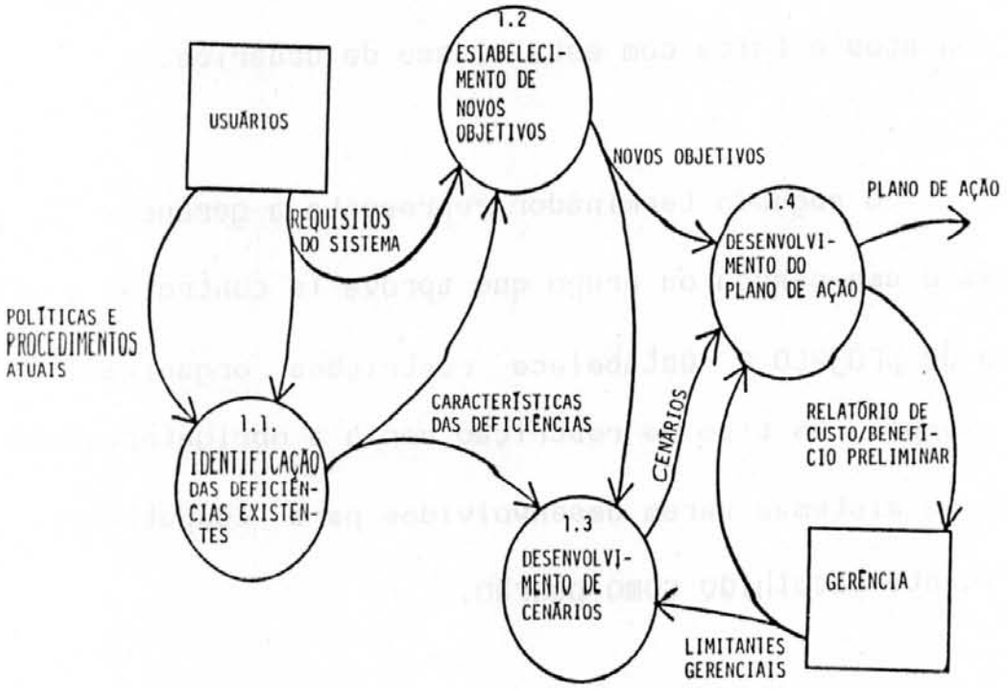


Fig. 4.5 Detalhes da atividade de pesquisa

Atividade 2 - Análise: O principal objetivo desta atividade, é transformar as necessidades dos usuários em especificações estruturadas. Isso envolve a modelagem do ambiente atual através da análise estruturada. O modelo atual é então usado como base para a modelagem do novo embiente. Uma visão geral das sub-atividades concernentes a análise é apresentada na fig. 4.6.

Atividade 3 - Projeto: O objetivo desta atividade é transformar as especificações estruturadas em módulos, hierarquias e interfaces. A técnica mais usada é o projeto estruturado. Uma visão geral desta atividade é apresentada na fig. 4.7.

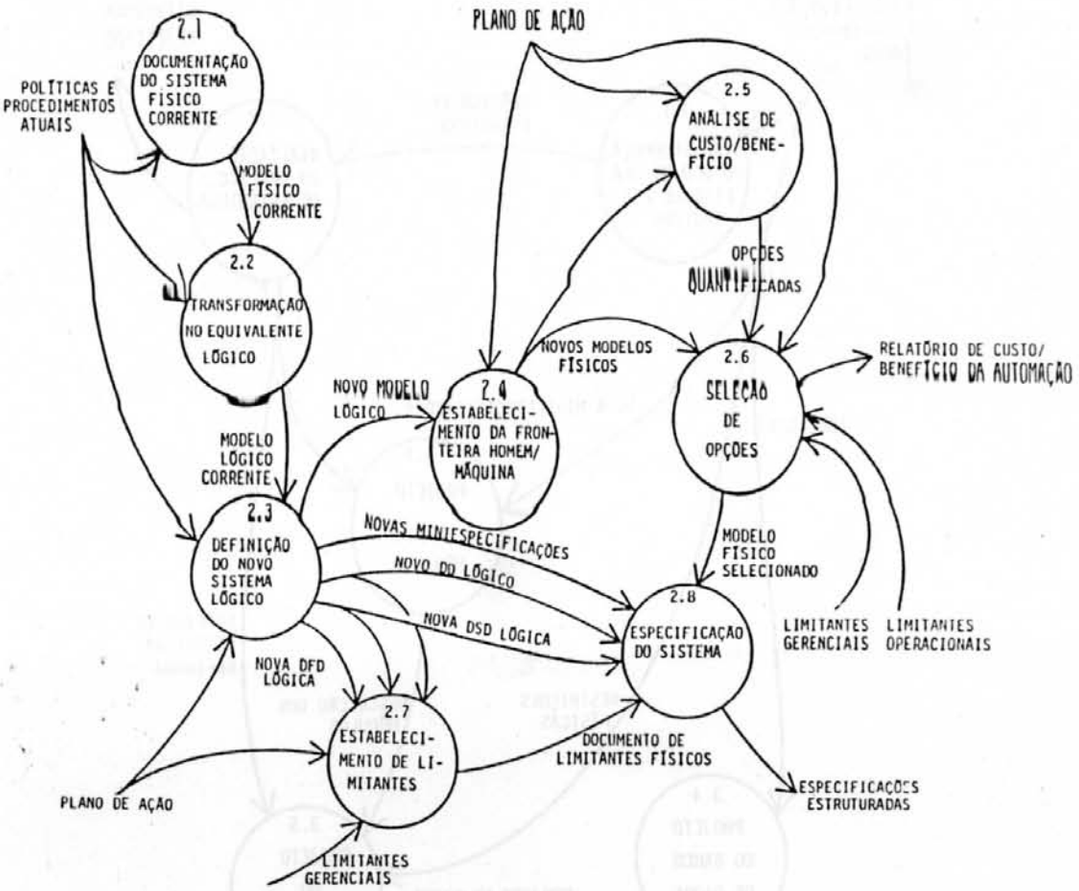


Fig. 4.6 Detalhes da atividade de análise

Atividade 4 - Implementação: Nesta atividade os módulos vão sendo codificados e progressivamente agregados até a conclusão final do "software". Para tanto são usadas duas técnicas: (1) programação estruturada e (2) implementação de cima para baixo ("top-down"). Na fig. 4.8 é apresentada uma visão geral desta atividade.

Atividade 5 - Geração de Testes de Aceitação: Esta atividade consiste basicamente em gerar alguns casos de teste. Tais testes são concebidos com base nas especificações estruturadas oriundas da atividade 3 (fig. 4.9).

Atividade 6 - Controle de Qualidade ("Quality Assurance"): Esta atividade é também conhecida como teste fi-

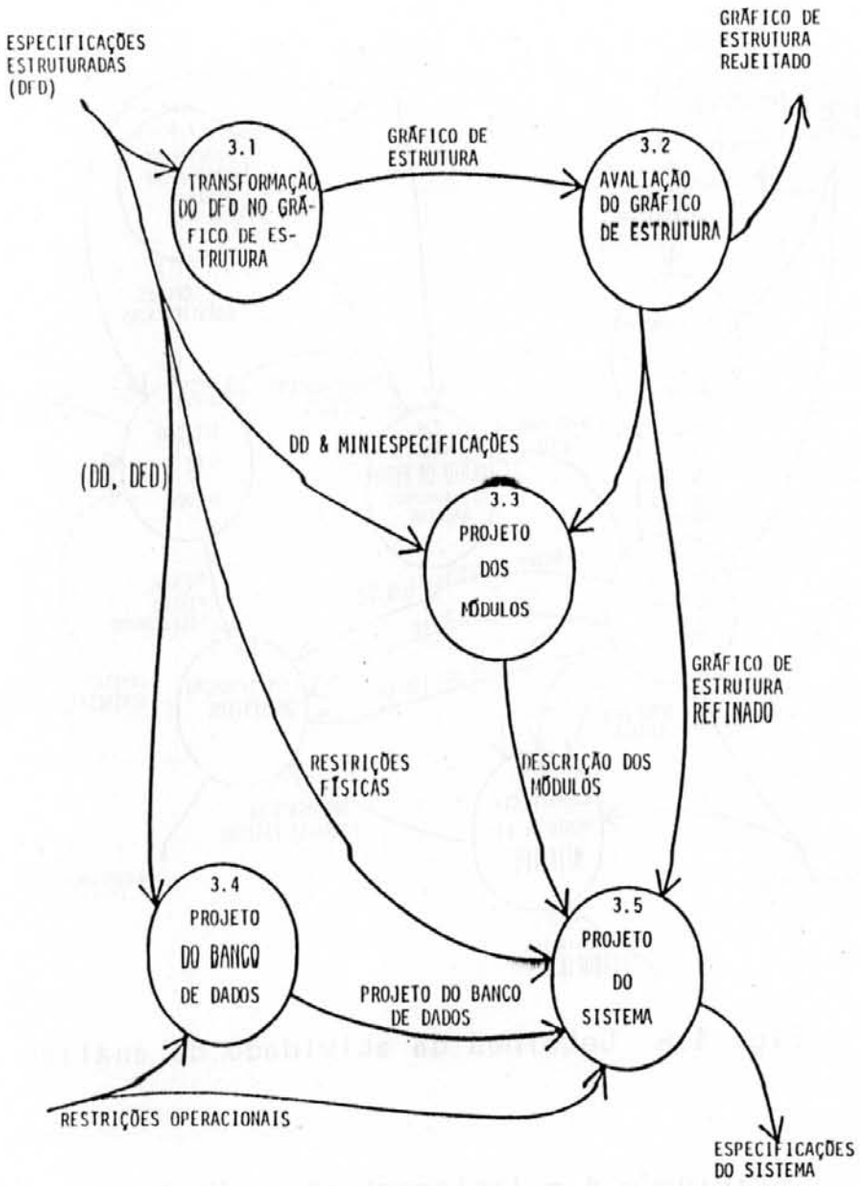


Fig. 4.7 Detalhes da atividade de projeto

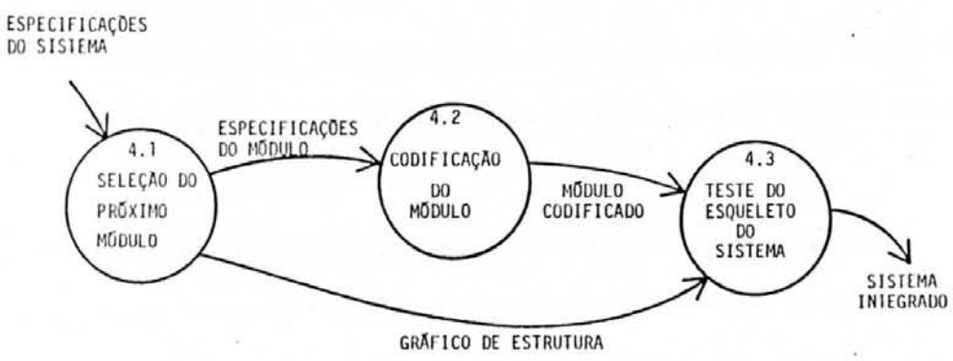


Fig. 4.8 Detalhes da atividade de implementação

nal ou teste de aceitação. Em essência o Controle de Qualidade é a execução dos casos de teste gerados na atividade 5.

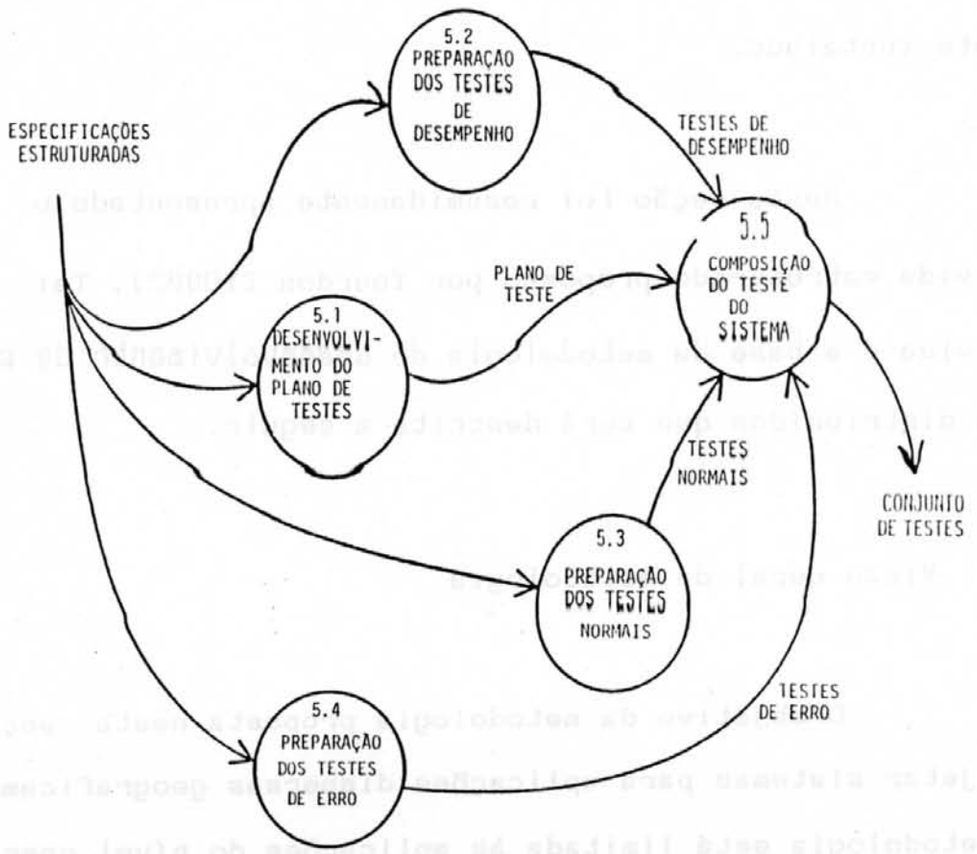


Fig. 4.9 Geração de testes de aceitação

Atividade 7 - Descrição de Procedimentos: O objetivo desta atividade é gerar uma descrição de como as pessoas devem agir no sitema. São providenciados procedimentos para a parte manual e para a parte automatizada. O produto desta atividade é o manual do usuário.

Atividade 8 - Conversão do banco de dados: Esta atividade consiste em reorganizar os dados existentes segundo a nova estrutura do banco de dados.

Atividade 9 - Instalação: Esta atividade é a transição do antigo para o novo sistema. Em alguns casos ela costuma acontecer na passagem de um dia para o próximo. Algumas vezes é feita na virada do mês. Em outras a instalação é um processo gradual onde os usuários recebem manuais,

"hardware", são treinados e o sistema vai sendo progressivamente instalado.

Nesta seção foi resumidamente apresentado o ciclo de vida estruturado proposto por Yourdon [YOU82]. Tal ciclo de vida é a base da metodologia de desenvolvimento de sistemas distribuídos que será descrita a seguir.

4.3 Visão geral da metodologia

O objetivo da metodologia proposta nesta seção é projetar sistemas para aplicações dispersas geograficamente. A metodologia está limitada às aplicações do nível operacional [CHI83] (fig. 4.10), sendo mais apropriada às empresas do setor comercial como lojas, farmácias, supermercados e outras; setor de serviços como bancos, locadoras de automóveis, hotéis e outras; bem como algumas indústrias onde as funções são dispersas.

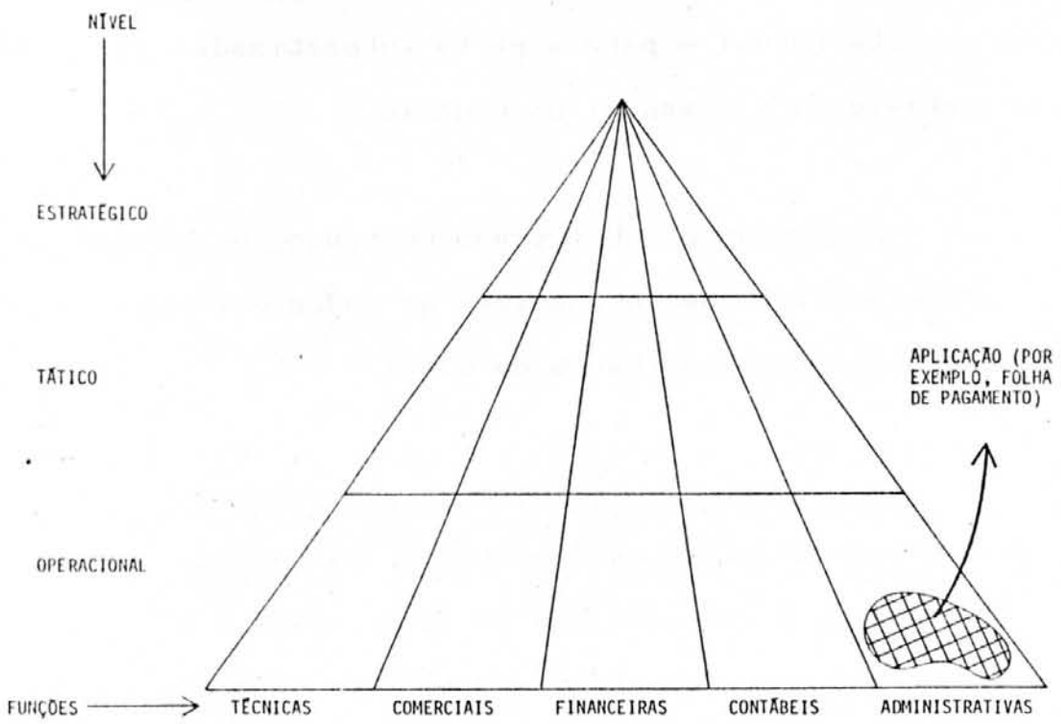


Fig. 4.10 Níveis das funções de uma empresa

A metodologia é composta de cinco etapas: (1) desenvolvimento da aplicação, (2) determinação da área geográfica, (3) determinação da carga, (4) dimensionamento dos recursos e (5) instalação do sistema. Na primeira etapa são produzidos o "software" aplicativo e redigidos os procedimentos da parte manual do sistema. Na segunda etapa são levantados os mapas que delimitam a região geográfica onde a empresa está dispersa. Na terceira etapa são registradas as demandas oriundas de cada local, como por exemplo taxa de telefonemas (ver seção 4.1). Na quarta etapa são projetados a configuração do "hardware" e o "layout" das salas. Na quinta etapa é feita a instalação de "hardware" e "software", as salas são preparadas de acordo com o "layout", as pessoas são treinadas e o sistema é posto em funcionamento.

Existem cinco entidades externas à metodologia: (1) usuários, (2) gerência, (3) operações, (4) fabricantes de computadores e (5) fornecedores de meios de comunicação. As três primeiras entidades são as mesmas do ciclo de vida estruturado descrito na seção 4.2.1.1. A quarta é formada pelos fabricantes de CPUs, unidades de disco, controladoras, terminais e outros. A quinta é composta pelos fornecedores de modems, multiplexadores e canais de comunicação. Como será visto na seção 4.3.1, estas entidades são origem de onze entradas e destino das seis saídas produzidas pela execução das cinco etapas da metodologia.

Os motivos para a estruturação da metodologia em 5 etapas, 5 entidades externas, 11 entradas e 6 saídas, serão apresentados na seção 4.3.3.

Como foi visto na seção 4.2, uma metodologia de desenvolvimento de sistemas pode ser representada por diagramas de fluxo de dados (DFD). O ciclo de vida estruturado [YOU82] foi apresentado segundo a notação de Tom DeMarco [DEM78] (fig. 4.11a). Para a metodologia proposta neste capítulo, está sendo sugerida a notação de Gane/Sarson [GAN83] (fig. 4.11b). Esta notação permite representar o fluxo de materiais. Como será visto a seguir, tal fluxo é necessário ao dimensionamento do sistema.

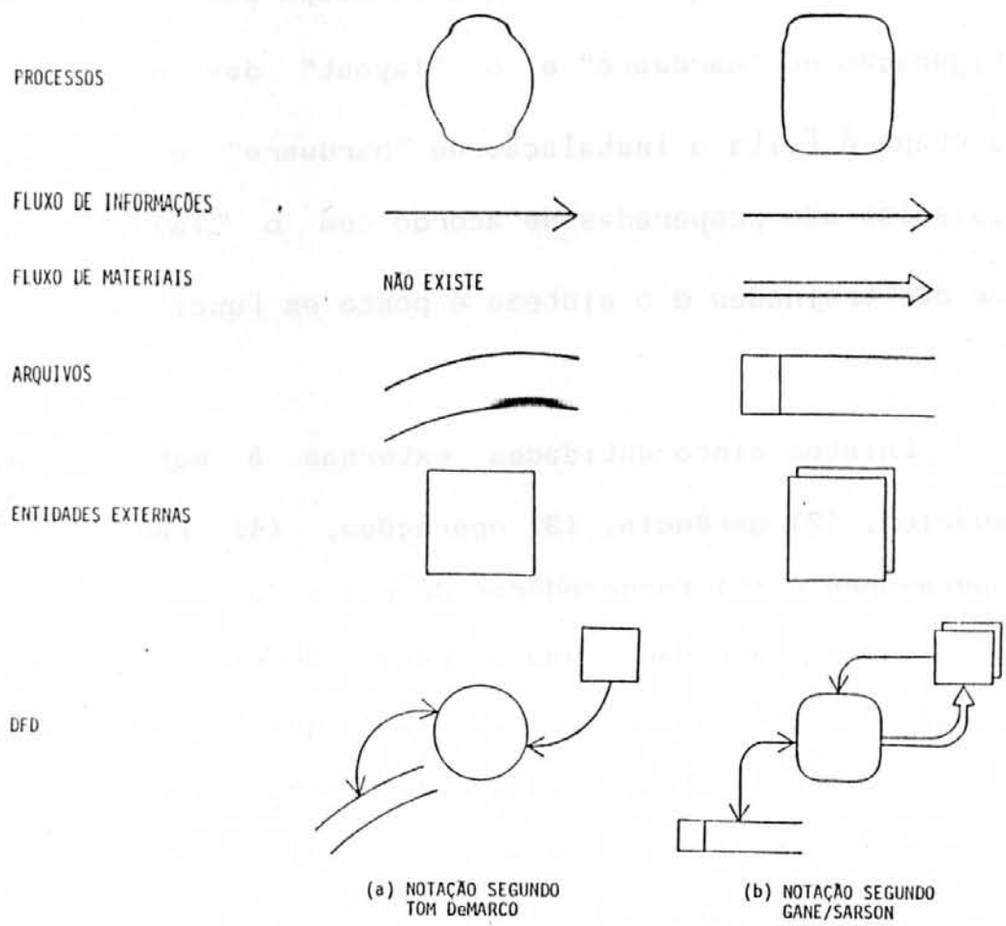


Fig. 4.11 Símbolos do DFD

A origem de alguns dados de dimensionamento do sistema são as entradas dos processos manuais. No exemplo da operadora de PBX apresentado no início da seção 4.1, a quantidade de telefonistas, número de terminais e volume de processamento são calculados em função do número de chamadas. A

chamada é a entrada de um processo manual executado pela telefonista. Como é possível notar, esta entrada é uma informação e pode ser representada tanto pela notação de Tom DeMarco como pela de Gane/Sarson. Há entretanto aplicações onde tal entrada não é informação. Por exemplo, numa aplicação de recebimento de materiais, a origem da carga a ser submetida ao sistema é a chegada de caminhões com mercadorias; numa empresa de transporte, a origem é a chegada de passageiros; num banco é a chegada de correntistas e outros. Estas entradas não são informações, sendo necessária uma maneira de representá-las. Uma forma é considerá-las como fluxo material. Tal fluxo pode ser representado pela notação de Gane/Sarson.

Na próxima seção será descrito o contexto da metodologia que está sendo proposta.

4.3.1 Contexto

Como foi visto, existem cinco entidades externas à metodologia, as quais emitem onze entradas e recebem seis saídas (fig. 4.12). Estas entradas e saídas são descritas nas próximas seções, ordenadas segundo as entidade de origem e destino.

4.3.1.1 Usuários

A entidade (1), usuários, tem as seguintes entradas e saídas (fig. 4.12):

UFRGS
BIBLIOTECA
CPD/PGCC

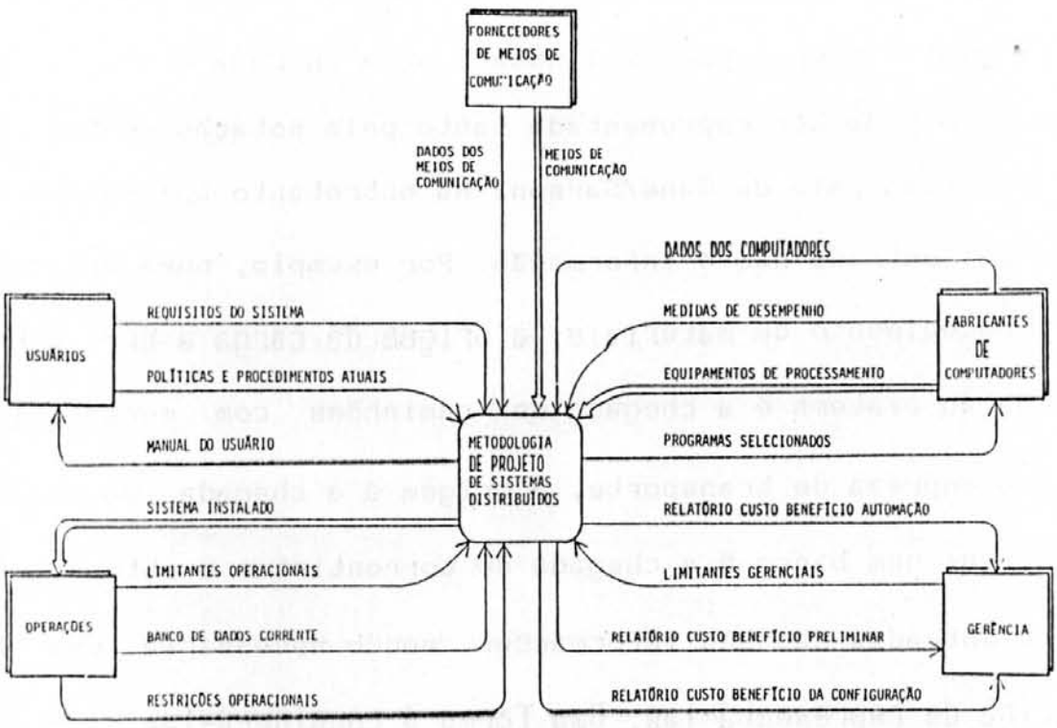


Fig. 4.12 Contexto da metodologia de projeto de sistemas distribuídos

Requisitos do sistema: é uma lista das funções requeridas e de características operacionais tais como tipo de negócio, localização da empresa, planos de expansão, tempo de resposta, tempo de espera nas filas e outros.

Políticas e procedimentos atuais: são descrições verbais ou escritas das características do sistema corrente.

Manual do usuário: são normas e procedimentos que permitem ao usuário operar corretamente o sistema. Incluem também guias de preenchimento de formulários, descrições de telas e relatórios, tabelas de codificação e outros.

4.3.1.2 Gerência

A entidade (2), gerência, tem as seguintes entra-

das e saídas (fig. 4.12):

Limitantes_gerênciais: são limitações de tempo, mão de obra e orçamento do projeto. Podem incluir também algumas restrições operacionais como a obrigatoriedade do uso de computadores de determinado fabricante, tempos de resposta de processos críticos e outros.

Relatório_de_custo_benefício: na metodologia estão sendo sugeridos três relatórios. O primeiro é o relatório de custo benefício preliminar. É produzido no início do projeto e permite à gerência decidir se é lucrativo continuar o desenvolvimento ou não. O segundo é o relatório de custo benefício da automação. É produzido durante a proposição do novo sistema, podendo ser emitido mais de uma vez. Com base nestes relatórios a gerência pode decidir qual o grau de automação. Os limites são a automação total onde praticamente todos os processos são executadas com auxílio do computador e a ausência de automação, onde tudo é executado pelas pessoas. O terceiro é o relatório de custo benefício da configuração. Este relatório é emitido pelo menos três vezes: (1) para decidir sobre o tipo de topologia a ser usado, (2) para a escolha dos computadores que irão compor a configuração básica e (3) para decidir sobre a compra dos meios de comunicação tais como modems, multiplexadores e linhas de transmissão de dados que irão compor a configuração final.

4.3.1.3 Operações

A entidade (3), operações, tem as seguintes entra-

das e saídas (fig. 4.12):

Sistema_instalado: é o conjunto de "hardware" e "software" em funcionamento. O "hardware" é composto por (1) equipamentos de processamento e (2) meios de comunicação. Os equipamentos de processamento são CPUs, memória auxiliar (disco, fita e outros) e equipamentos de interface tais como terminais e impressoras. Os meios de comunicação são linhas de comunicação, modems e multiplexadores. O "software" divide-se em (1) básico e (2) aplicativo. O "software" básico é composto pelo sistema operacional, gerenciador de banco de dados, monitor de comunicações e alguns programas especiais para o aumento de produtividade dos equipamentos. O "software" aplicativo é composto pelos programas de aplicação.

Restrições_operacionais: são restrições físicas, muitas vezes oriundas de equipamentos já instalados e que obrigatoriamente deverão ser aproveitados na nova configuração. São por exemplo tamanho de blocos de disco, "software" básico e outros.

Limitantes_operacionais: são padrões adotados pelo setor de processamento de dados da empresa, tais como tamanho máximo de memória a ser ocupado por um programa, linguagem a ser usada na codificação, nomes dos campos e outros.

Banco_de_dados_corrente: é o conjunto de arquivos usado pelo sistema (corrente) que está em operação na empresa.

4.3.1.4 Fabricantes de computadores

A entidade (4), fabricante de computadores, tem as seguintes entradas e saídas (fig. 4.12):

Programas_selecionados: é o conjunto de programas que será mais executado no sistema. Como será visto no próximo capítulo, estes programas são responsáveis por cerca de 80% da carga de processamento. São enviados a cada fabricante para levantamento de dados tais como número de instruções executadas e número de acesso a disco por ciclo de entrada e saída.

Medidas_de_desempenho: esta entrada está dividida em dois tipos. O primeiro tipo são as medidadas de desempenho tradicionais como velocidade da CPU e tempo de acesso a disco. O segundo tipo são dados obtidos a partir dos programas descritos no parágrafo anterior.

Dados_dos_computadores: são dados de "hardware" e "software" . Os dados do "hardware" são tamanho de memória, número máximo de terminais, espaço físico necessário e outros. Os dados do "software" são sistema operacional, gerenciador de banco de dados e monitor de comunicação.

Equipamentos_de_processamento: é o "hardware" propriamente dito. Inclui CPUs, memória auxiliar (disco e fita) e equipamentos de interface (terminais e impressoras).

4.3.1.5 Fornecedores de meios de comunicação

A entidade (5), fornecedores de meios de comunicação, tem as seguintes entradas e saídas (fig. 4.12):

Dados dos meios de comunicação: Há dois grupos. O primeiro é constituídos pelos dados dos canais de comunicação tais como velocidades (bps) e custos. O segundo é formado pelos dados dos equipamentos. Tais dados são os tempos de reversão dos modems ("turnaround time"), tempo de retardo dos multiplexadores e outros.

Meios de comunicação: são modems, multiplexadores e linhas de transmissão de dados.

Até aqui foi descrito o contexto da metodologia de desenvolvimento de sistemas distribuídos. Na próxima seção as cinco etapas que compõe esta metodologia serão resumidas.

4.3.2 Etapas

Conforme visto anteriormente, a metodologia proposta neste capítulo tem cinco etapas. As relações entre estas etapas são apresentadas sob a forma de DFD na fig. 4.13. A seguir é dado um resumo de tais etapas juntamente com uma descrição de seus produtos.

Etapa 1 - desenvolvimento da aplicação (fig. 4.13): o objetivo desta etapa é desenvolver o novo sistema. Envolve as atividades de análise, projeto, codificação e ou-

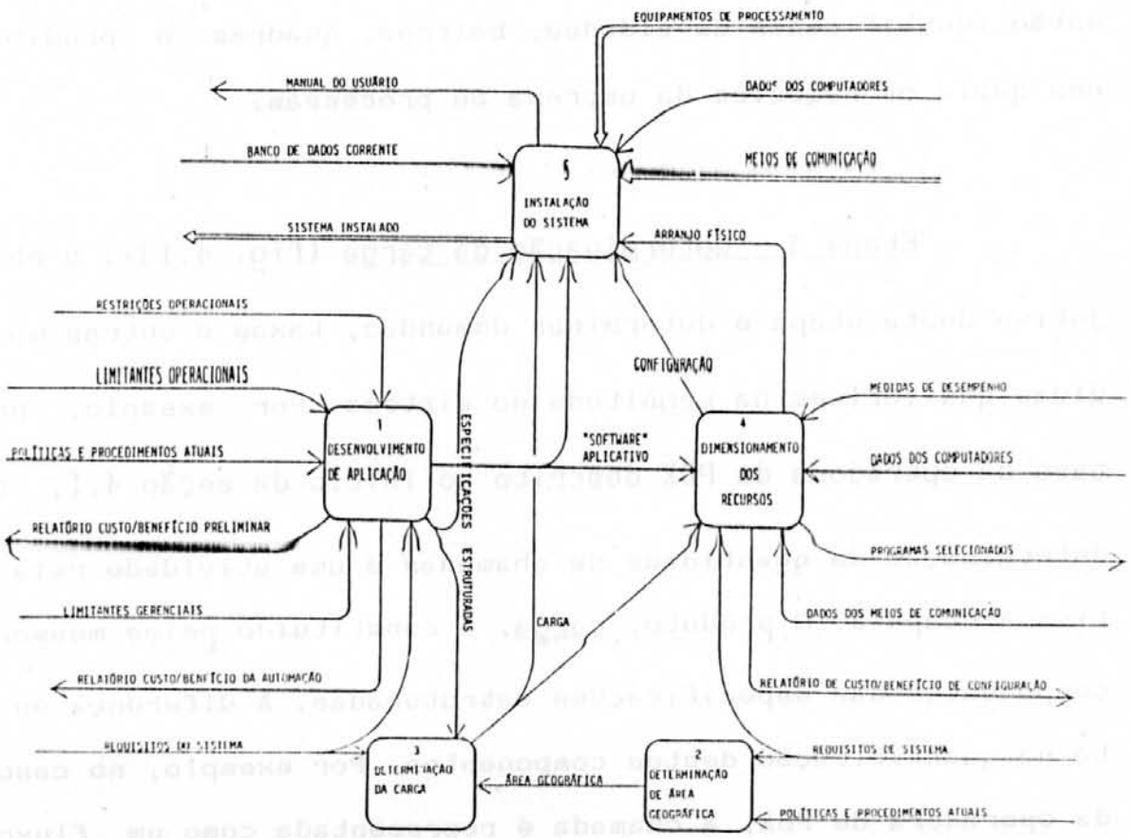


Fig. 4.13 Etapas da metodologia de projeto de sistemas distribuídos

tras. Tem quatro produtos. Os dois primeiros (1) relatório de custo benefício preliminar e (2) relatório de custo benefício da automação já foram apresentados na seção 4.3.1. O terceiro produto é o software aplicativo, que é composto pelos programas de aplicação. O quarto produto é composto pelas especificações estruturadas. Como será visto no próximo capítulo, tais especificações são descrições do fluxo de dados, arquivos, processos e uma série de restrições físicas tais como tempos de resposta, tamanho máximo de programas, desenho de telas, relatórios e outros.

Etapa 2 - determinação da área geográfica (fig. 4.13): o objetivo desta etapa é delimitar o território por onde o sistema está disperso. O produto, área geográfica, é

composto por um conjunto de mapas detalhados em níveis, onde estão identificadas as cidades, bairros, quadras e prédios nos quais os negócios da empresa se processam.

Etapa 3 - determinação da carga (fig. 4.13): o objetivo desta etapa é determinar demandas, taxas e outras medidas que influem na magnitude do sistema. Por exemplo, no caso da operadora de PBX descrito no início da seção 4.1, a determinação da quantidade de chamadas é uma atividade relativa à etapa 3. O produto, carga, é constituído pelos mesmos componentes das especificações estruturadas. A diferença está na quantificação destes componentes. Por exemplo, no caso da operadora de PBX, a chamada é representada como um fluxo de dados no DFD. A determinação da carga consiste em quantificar este fluxo, anotando sobre ele a taxa horária de telefonemas.

Etapa 4 - dimensionamento dos recursos (fig. 4.13): esta etapa tem dois objetivos. O primeiro é projetar a configuração do sistema de computação. O projeto é executado segundo as técnicas apresentadas no capítulo anterior. Para possibilitar o uso destas técnicas, foram incluídas nesta etapa atividades de coleta e organização de dados quantitativos. O segundo objetivo é projetar o arranjo físico ("layout"). Isso inclui o dimensionamento de pessoal para a execução dos processos manuais, o arranjo das mesas e cadeiras nas salas e a localização dos terminais com os quais as pessoas irão interagir com o sistema.

Esta etapa tem quatro produtos: (1) programas se-

lecionados (2) relatório de custo benefício da configuração, (3) configuração e (4) arranjo físico. Os dois primeiros foram apresentados na seção 4.3.1. O terceiro produto é composto pela topologia, configuração básica e configuração final. O quarto produto é constituído por plantas de escritórios, oficinas, laboratórios e outros.

Etapa 5 - instalação do sistema (fig. 4.13): o objetivo desta etapa é por o sistema em funcionamento. Isso inclui instalação dos móveis e equipamentos, adequação do "software" básico à configuração, adaptação dos programas de aplicação, conversão do banco de dados existente, preparação de manuais e treinamento do usuário. Os dois produtos (1) sistema instalado e (2) manual do usuário já foram apresentados na seção 4.3.1.

Até aqui foi dada uma visão geral da metodologia. A descrição detalhada das várias etapas é apresentada no capítulo 5. Na próxima seção, são discutidos os motivos que levaram a metodologia a ter a estrutura apresentada anteriormente.

4.3.3 Processo de estruturação da metodologia

O processo de estruturação da metodologia teve três partes. Na primeira foi criada uma metodologia preliminar composta de cinco etapas. Na segunda parte estas etapas foram detalhadas, o que possibilitou a definição das entidades externas. Na terceira parte foi feita uma revisão, a notação de Gane/Sarson passou a ser adotada e as últimas en-

tradas e saídas foram criadas, consolidando a metodologia. Estas partes do processo são descritas nas próximas seções.

4.3.3.1 Metodologia preliminar

A metodologia preliminar tem cinco etapas: (1) desenvolvimento de programas de aplicação, (2) determinação da área geográfica, (3) determinação da carga, (4) projeto da configuração e (5) adaptação dos programas. Estas etapas foram criadas progressivamente, conforme apresentado a seguir.

As duas primeiras etapas a serem concebidas foram a (1) desenvolvimento de programas e a (4) projeto da configuração. Como foi visto no capítulo 3, as metodologias de desenvolvimento de sistemas distribuídos existentes se limitam ao projeto da configuração do "hardware". Em tais metodologias não está especificado como obter os dados para executar o projeto. Muitos destes dados são oriundos do "software" aplicativo. Desta forma, o projeto completo de um sistema distribuído deve iniciar pelo desenvolvimento dos programas de aplicação. A etapa seguinte é o projeto da configuração. Se a metodologia preliminar fosse composta apenas destas duas etapas, poderia ser representada num diagrama de fluxo de dados conforme a fig. 4.14.

A terceira etapa a ser concebida foi a (5) adaptação dos programas. Há dois motivos para a existência desta etapa. O primeiro é a ordem em que "software" e configuração são projetados. Na metodologia proposta anteriormente (fig. 4.14), os programas são desenvolvidos em primeiro lugar. A

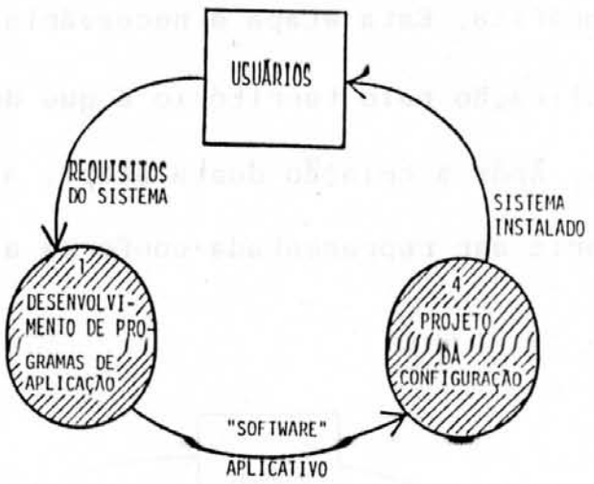


Fig. 4.14 Primeiro esboço da metodologia preliminar

configuração só é conhecida depois que o "software" está pronto. Isso exige uma etapa posterior, qual seja adaptar os programas à configuração. O segundo motivo é a variedade de processadores no sistema. Como foi visto no capítulo 3, pode ser econômico instalar computadores de capacidades diferentes numa mesma configuração. Tais computadores não precisam ser do mesmo fabricantes ou ter a mesma arquitetura. Quando a configuração tem tais características, é preciso criar versões do "software" para cada computador. Tais versões podem ser feitas na etapa de adaptação dos programas (fig. 4.15).

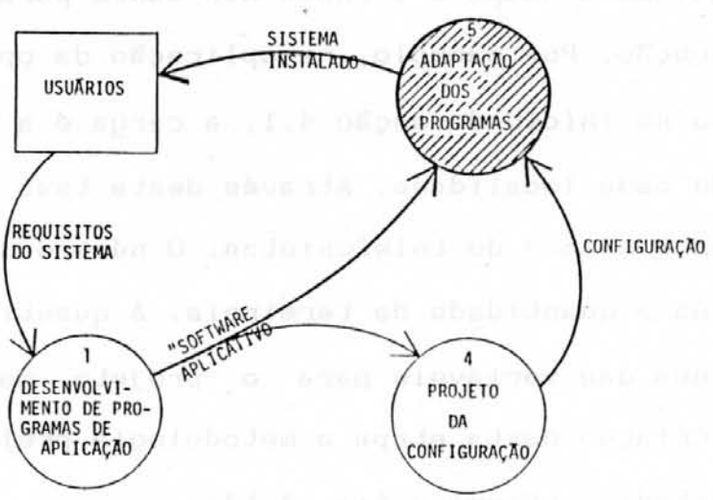


Fig. 4.15 Segundo esboço da metodologia preliminar

A quarta etapa a ser criada foi a (2) determinação da área geográfica. Esta etapa é necessária porque a distribuição da aplicação pelo território é que determina os nós da topologia. Após a criação desta etapa, a metodologia preliminar poderia ser representada conforme a fig. 4.16.

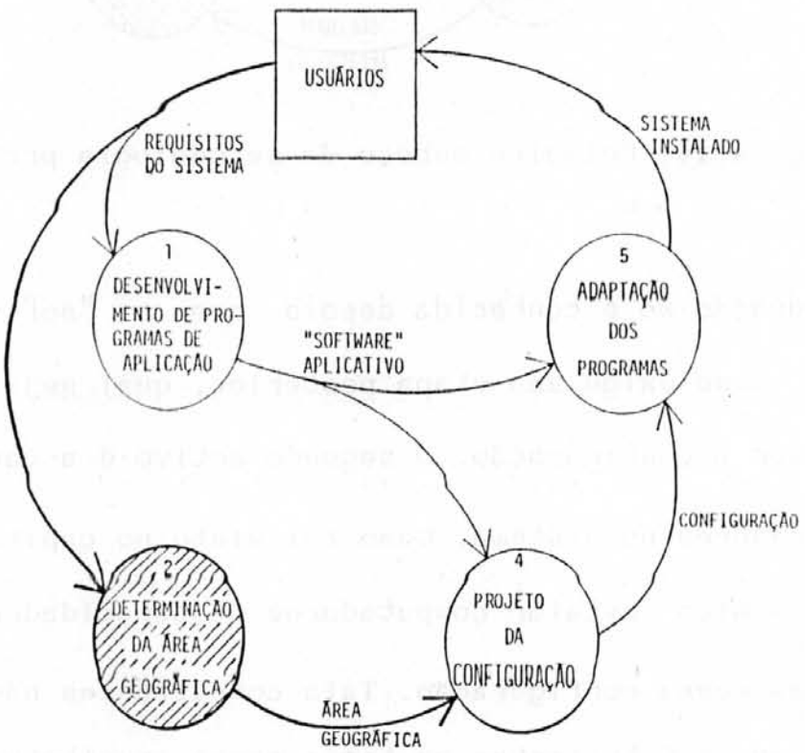


Fig. 4.16 Terceiro esboço da metodologia preliminar

A última etapa a ser criada foi a (3) determinação da carga. Esta etapa é a fonte dos dados para o projeto da configuração. Por exemplo, na aplicação da operadora de PBX descrita no início da seção 4.1, a carga é a taxa de telefonemas de cada localidade. Através desta taxa é possível determinar o número de telefonistas. O número de telefonistas determina a quantidade de terminais. A quantidade de terminais é uma das variáveis para o projeto da configuração. Após a criação desta etapa a metodologia preliminar pode ser representada conforme a fig. 4.17.

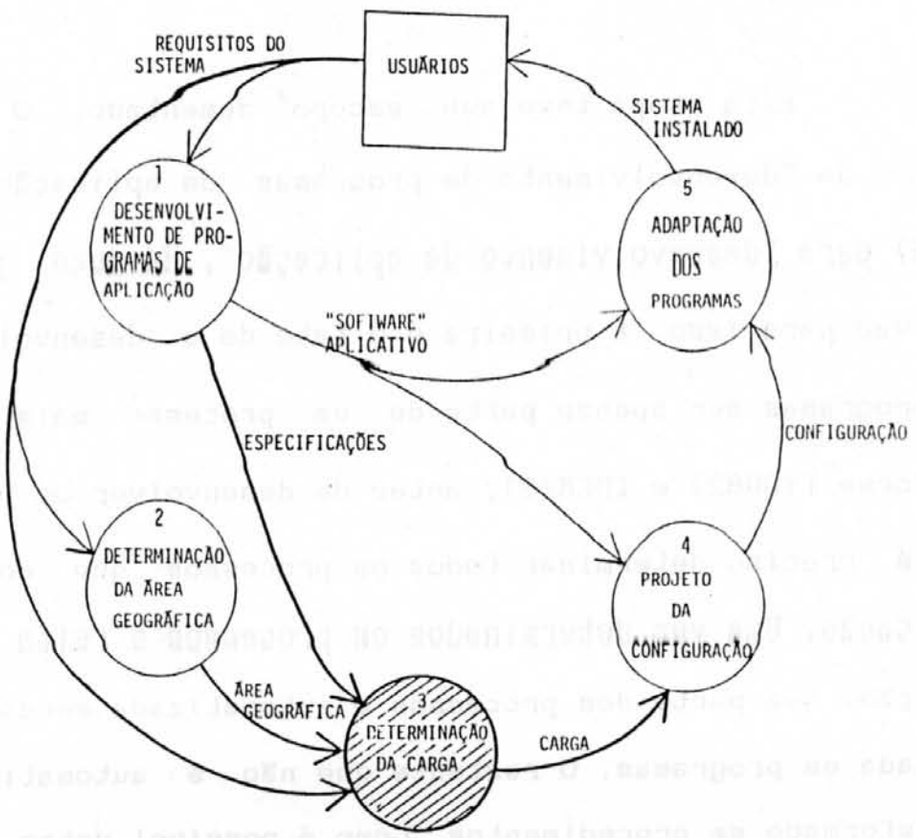


Fig. 4.17 Metodologia preliminar

Até aqui foi apresentada a primeira parte do processo de estruturação da metodologia de desenvolvimento de sistemas distribuídos, que foi descrita no início da seção 4.3. Como foi visto, nesta primeira parte foi concebida uma metodologia preliminar. Na próxima seção é apresentada a segunda parte da estruturação.

4.3.3.2 Esboço da metodologia definitiva.

Na segunda parte do processo de estruturação, as etapas propostas na metodologia preliminar foram detalhadas. Este detalhamento trouxe dois resultados: (1) determinação das atividades de cada etapa e (2) determinação das entidades externas à metodologia. Este detalhamento é descrito nas próximas seções, etapa por etapa.

4.3.3.2.1 Etapa 1 - desenvolvimento da aplicação

Esta etapa teve seu escopo aumentado. O título passou de "desenvolvimento de programas de aplicação" (fig. 4.14) para "desenvolvimento da aplicação". Há duas justificativas para isso. A primeira é o fato de o desenvolvimento de programas ser apenas parte de um processo mais amplo. Conforme [YOU82] e [DEM78], antes de desenvolver os programas é preciso determinar todos os processos que compõe a aplicação. Uma vez determinados os processos é feita uma separação. Uma parte dos processos é automatizada sendo transformada em programas. O restante que não é automatizado é transformado em procedimentos. Como é possível notar, a elaboração de programas é apenas parte de uma etapa mais abrangente qual seja "desenvolvimento da aplicação".

A segunda justificativa diz respeito à determinação da carga. O volume de processamento do sistema depende da taxa de execução de transações [AGA85]. Essas taxas normalmente tem origem nos processos manuais. Por exemplo, seja a aplicação da operadora de PBX descrita no início da seção 4.1. O evento inicial é uma chamada. O atendimento é feito manualmente pela telefonista . Para completar a ligação ela ativa a transação de consulta à lista telefônica. A taxa de execução da transação é igual à taxa de chamadas. Entretanto, caso a telefonista tenha decorado os números mais solicitados, somente para algumas chamadas a transação será ativada. Assim a taxa de execução das transações cai para uma fração da taxa de chamadas. Como é possível notar, os processos manuais tem grande influência na determinação da car-

ga do sistema. Desta forma, é preciso existir na metodologia uma etapa onde estes processos possam ser determinados. Pelas razões expostas no parágrafo anterior, o melhor lugar é a etapa 1.

Para detalhar a etapa 1 foi usado o ciclo de vida estruturado apresentado na fig. 4.4 e descrito na seção 4.2.1. As atividades que compõe o ciclo foram reunidas em dois grupos. O primeiro grupo é formado pelas atividades necessárias ao desenvolvimento da aplicação, sendo composto pelas atividades (1) pesquisa, (2) análise, (3) projeto, (4) implementação, (5) geração de testes de aceitação e (6) controle de qualidade. O segundo grupo é formado pelas atividades voltadas à implantação. Tais atividades são: (7) descrição de procedimentos, (8) conversão da base de dados e (9) instalação. As atividades do primeiro grupo originaram a etapa 1. As atividades do segundo foram incorporadas à etapa 5 (fig. 4.18).

Após o detalhamento da etapa 1, a metodologia preliminar (fig. 4.14) sofreu a primeira modificação, originando o primeiro esboço da metodologia definitiva (fig. 4.19).

4.3.3.2.2 Etapa 2 - determinação da área geográfica

Não houve necessidade de decompor esta etapa em atividades. Tal etapa permeneceu no esboço da metodologia definitiva (fig. 4.19) tal como era na metodologia preliminar.

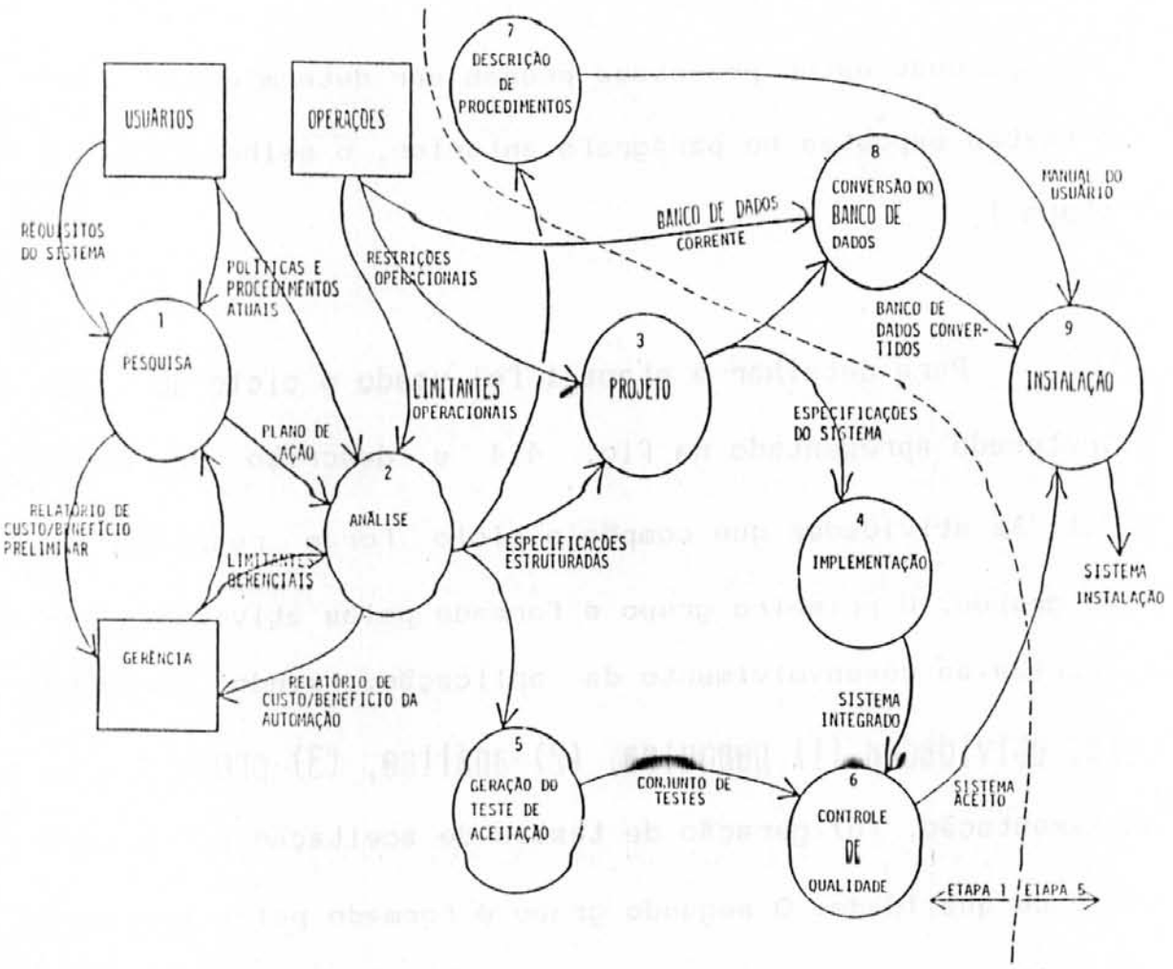


Fig. 4.18 Ciclo de vida estruturado

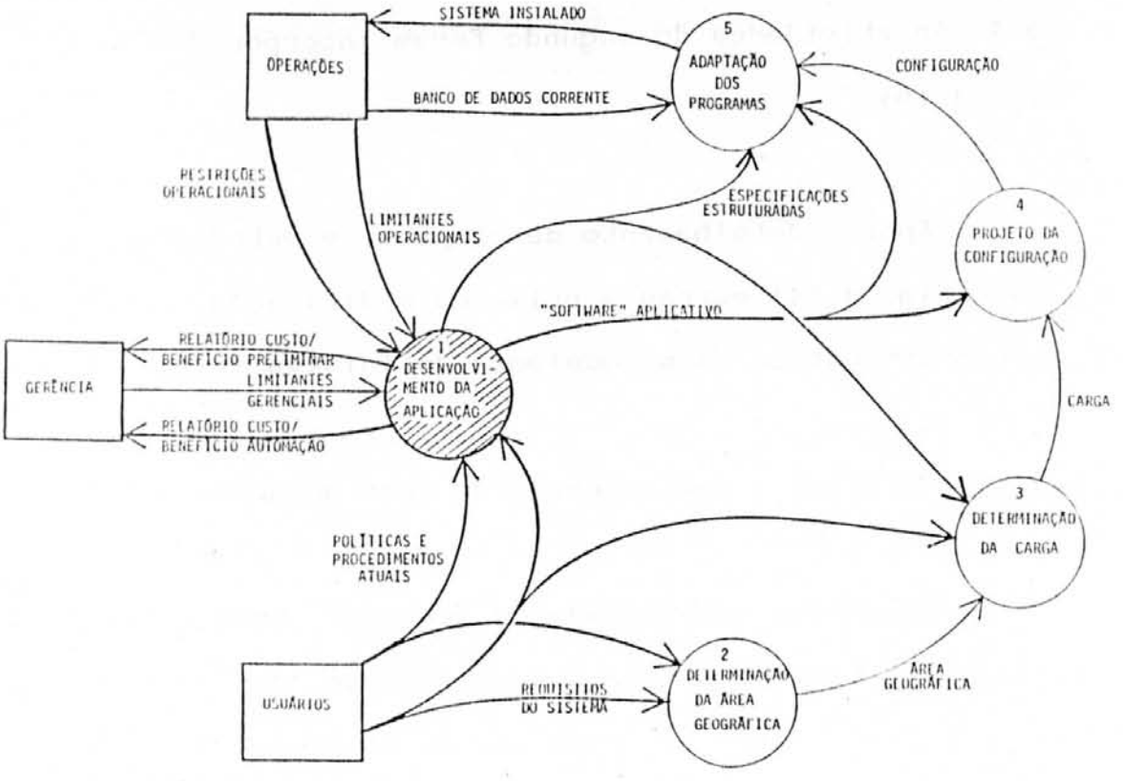


Fig.4.19 Primeira evolução da metodologia preliminar

4.3.3.2.3 Etapa 3 - determinação da carga

Como foi visto na seção 4.3.2, o objetivo desta etapa é obter dados quantitativos para o projeto da configuração final. Os dados quantitavos podem ser classificados em (1) dependentes e (2) independentes. Os primeiros são aqueles que dependem do processador. Por exemplo, o número de instruções executadas depende da arquitetura, compilador e "software" básico do computador que está processando o programa. Fazem parte do segundo tipo, os dados que independem do computador, como por exemplo a taxa de execução de um processo manual. Os dados dependentes são determinados quando do projeto da configuração na etapa 4. Na etapa 3, determinação da carga, são determinados apenas os dados independentes.

A etapa 3 foi detalhada em seis atividades: (1) determinação da demanda, (2) determinação do tempo de execução dos processos manuais, (3) determinação da demanda dos programas, (4) determinação da demanda dos arquivos, (5) determinação do tamanho das mensagens e (6) composição da carga. As relações entre estas atividades são apresentadas na fig. 4.20.

O detalhamento da etapa (3) não provocou mudanças no esboço da metodologia definitiva apresentado na fig. 4.19.

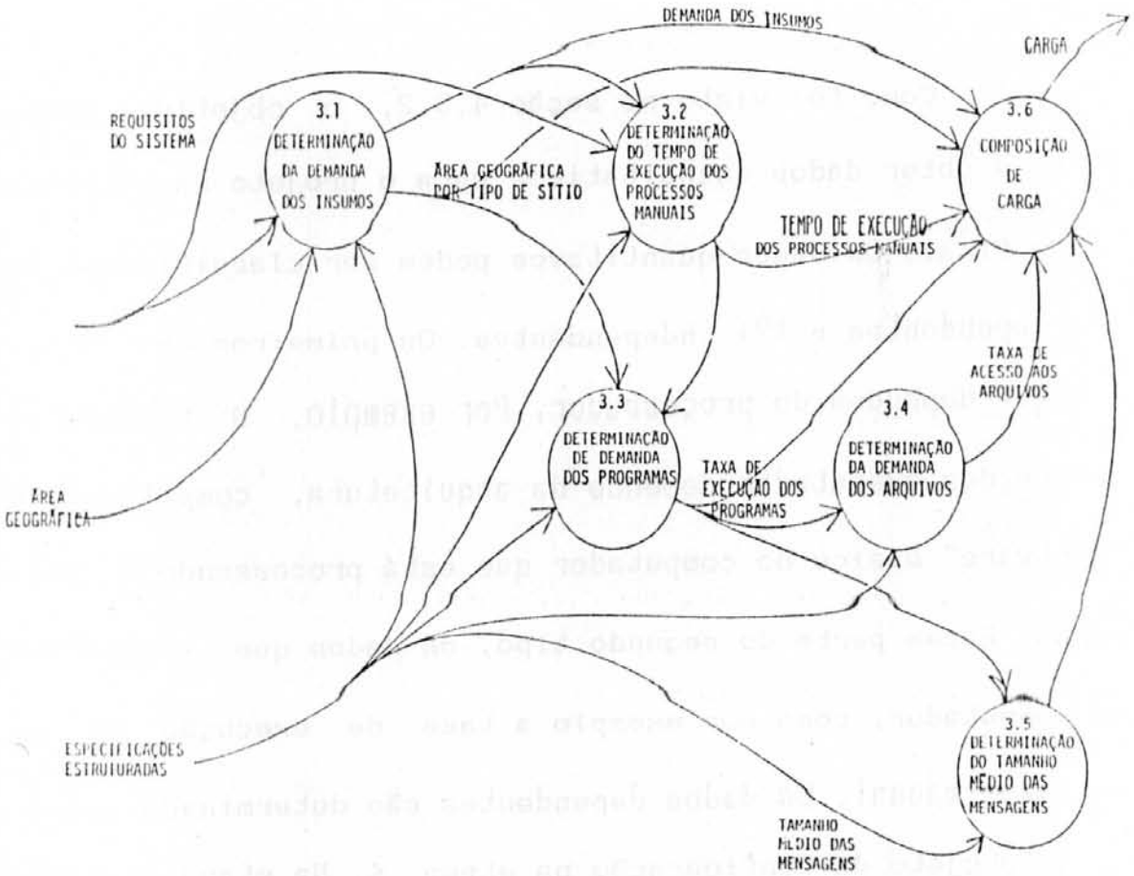


Fig. 4.20 Detalhes da etapa de determinação da carga

4.3.3.2.4 Etapa 4 - dimensionamento dos recursos

A etapa 4 teve seu escopo aumentado. O título passou de "projeto da configuração" (fig. 4.19) para "dimensionamento dos recursos". A razão desta alteração foi a inclusão de procedimentos para o projeto do arranjo físico. Como foi visto no capítulo 3, uma das variáveis do projeto da configuração é a quantidade de terminais. O valor desta variável é obtido em função da quantidade de pessoas. Isso já foi demonstrado pelo exemplo da operadora do PBX apresentado no início da seção 4.1. Entretanto não basta saber o número de pessoas, é preciso providenciar mesas e cadeiras para acomodá-las. A disposição destes recursos no espaço físico disponível, pode alterar a quantidade de terminais. Por exemplo, duas telefonistas sentadas próximas uma da outra

podem compartilhar o mesmo terminal. Entretanto isso é impraticável se elas ocuparem salas separadas. Como é possível notar, o projeto do sistema de computação é precedido do dimensionamento de vários outros recursos. O melhor lugar para a determinação destes recursos é a etapa (4).

Na etapa 4 foram reunidos os procedimentos de projeto do sistema de computação descritos no capítulo 3 e as técnicas de de concepção de "layout", dando origem a seis atividades: (1) projeto do arranjo físico, (2) obtenção dos dados para o projeto da topologia, (3) projeto da topologia, (4) obtenção dos dados para o projeto da configuração básica, (5) projeto da configuração básica e (6) projeto da configuração final. As relações entre estas atividades são apresentadas na fig. 4.21.

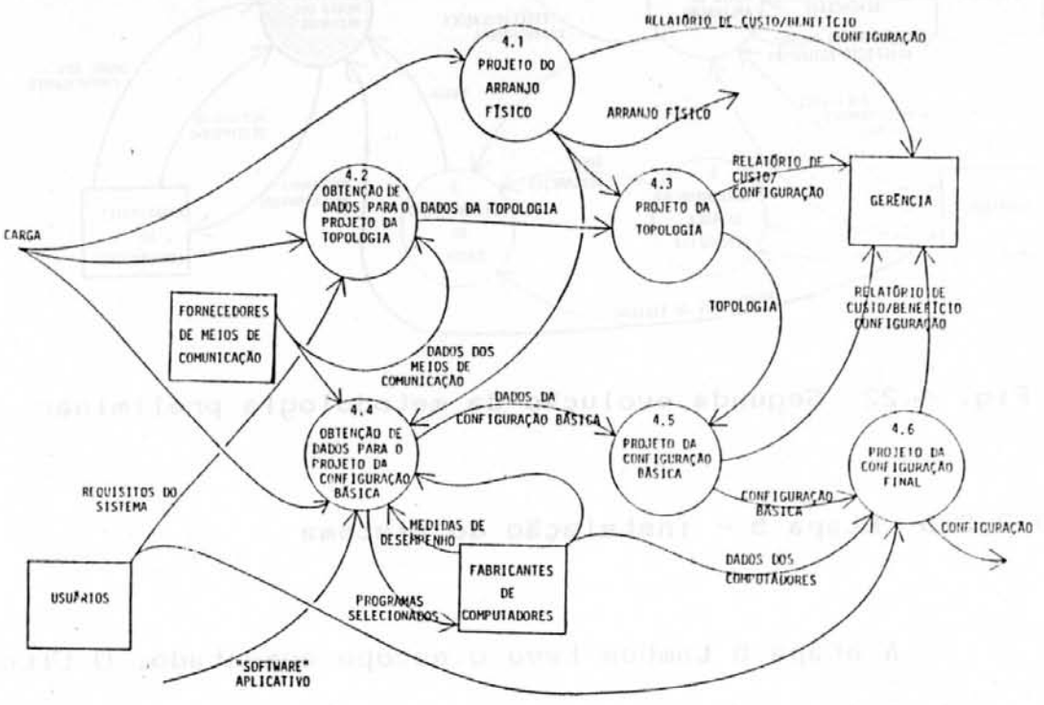


Fig. 4.21 Detalhes da etapa de projeto da configuração

Como é possível notar, para levar a cabo a etapa 4 foi preciso incluir mais duas entidades externas: (1) fabricantes de computadores e (2) fornecedores de equipamentos de comunicação. Estas entidades fornecem dados para o projeto do sistema de computação. Além disso há ainda a participação da gerência para decidir sobre custo/benefício nos vários estágios do projeto, bem como dos usuários, que fornecem alguns requisitos tais como tempo de resposta e tempo de fila entre outros. O detalhamento da etapa 4 alterou o esboço da metodologia até então descrito conforme a fig. 4.19. O novo esboço é apresentado na fig. 4.22.

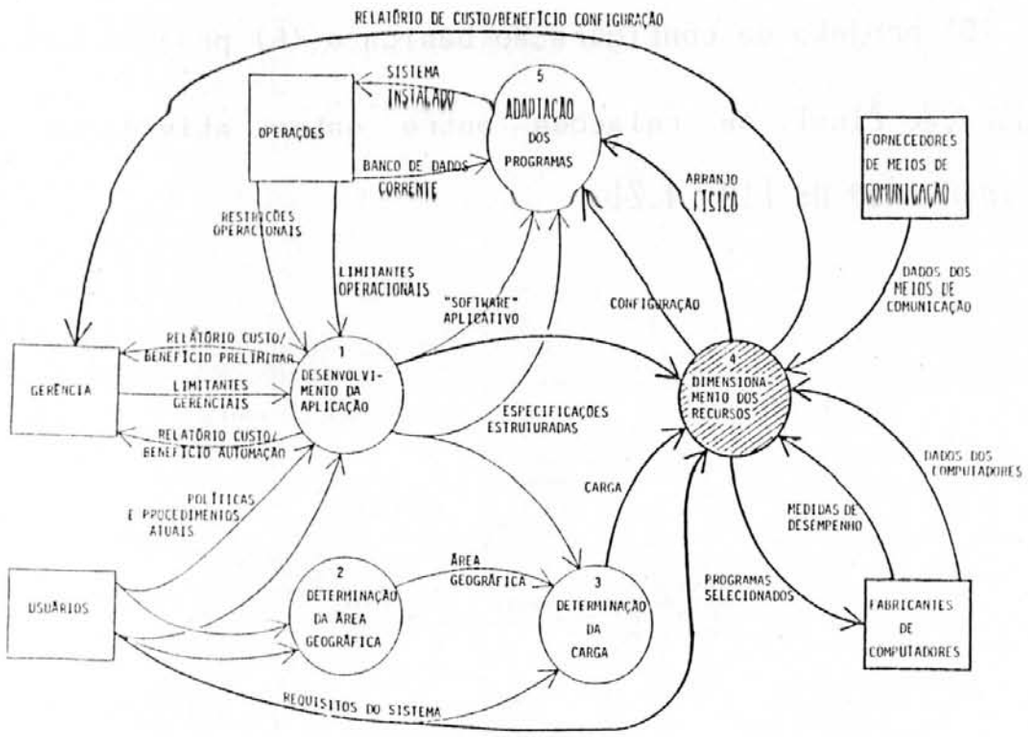


Fig. 4.22 Segunda evolução da metodologia preliminar

4.3.3.2.5 Etapa 5 - instalação do sistema

A etapa 5 também teve o escopo aumentado. O título passou de "adaptação de programas" (fig. 4.22) para "instalação do sistema". A causa dessa nova abrangência é a neces-

sidade de procedimentos adicionais para colocar o sistema em funcionamento. Para que o sistema seja posto em condições de uso não basta adaptar os programas à configuração. É preciso instalar "hardware" e "software", converter o banco de dados, preparar o manual do usuário e outros. O melhor local encontrado para incluir tais procedimentos foi a etapa 5.

A etapa 5 foi detalhada em seis atividades: (1) projeto físico do banco de dados, (2) adaptação dos programas, (3) descrição de procedimentos, (4) instalação dos recursos, (5) conversão do banco de dados e (6) instalação do "software". As atividades (3), (5) e (6) são oriundas do ciclo estruturado (fig. 4.15). A atividade (1) é necessária porque na mesma configuração pode haver mais de um computador operando com SGBDs diferentes. A atividade (2) já foi justificada na seção 4.3.3.1. A atividade (4) é necessária na prepararação das salas e instalação de móveis e equipamentos. As relações entre estas atividades são apresentadas na fig. 4.23.

Após o detalhamento da etapa 5 o esboço da metodologia preliminar passou a ter a forma apresentada na fig. 4.24.

Até aqui foi apresentada a segunda parte do processo de estruturação da metodologia de desenvolvimento de sistemas distribuídos, que foi descrita no início da seção 4.3. Como foi visto, nesta segunda parte foram detalhadas as cinco etapas da metodologia preliminar. Este detalhamento trouxe dois resultados: consolidação das etapas e determina-

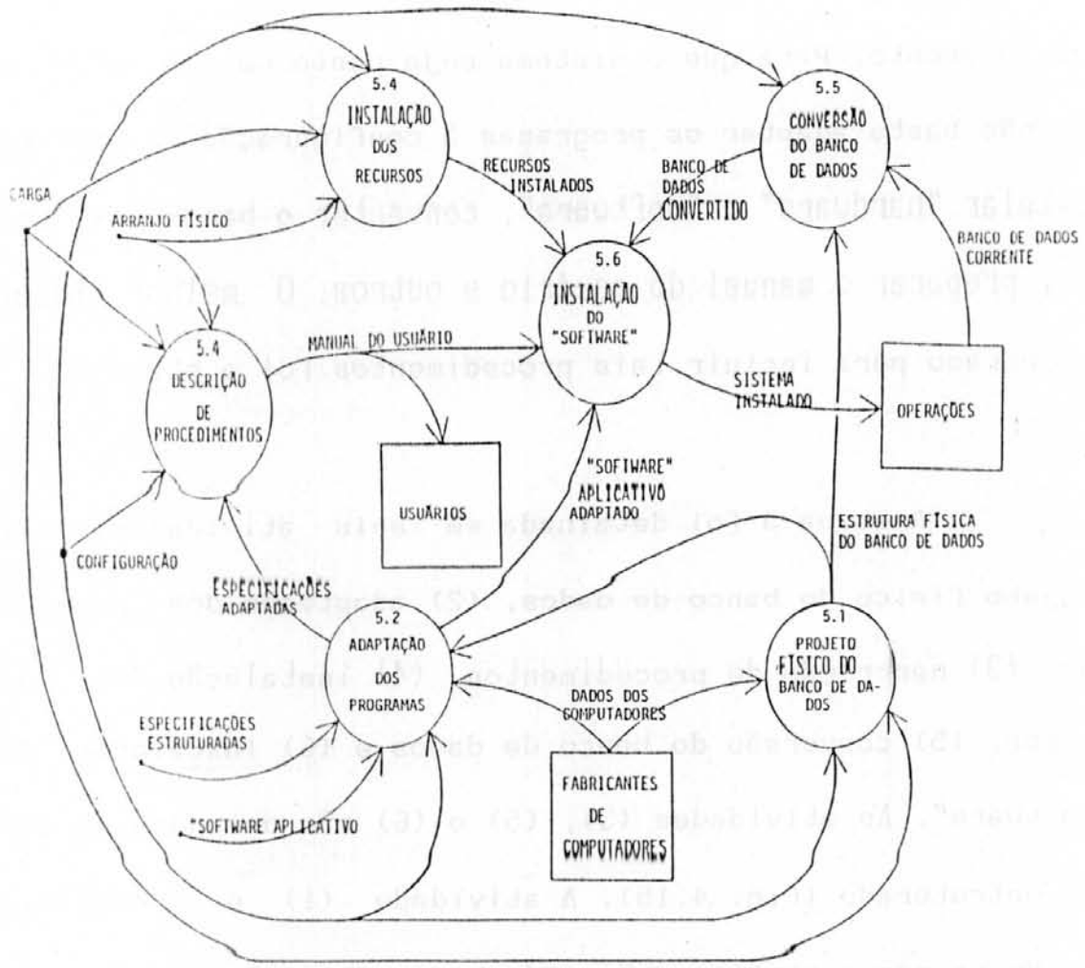


Fig. 4.23 Detalhes da etapa de instalação do sistema

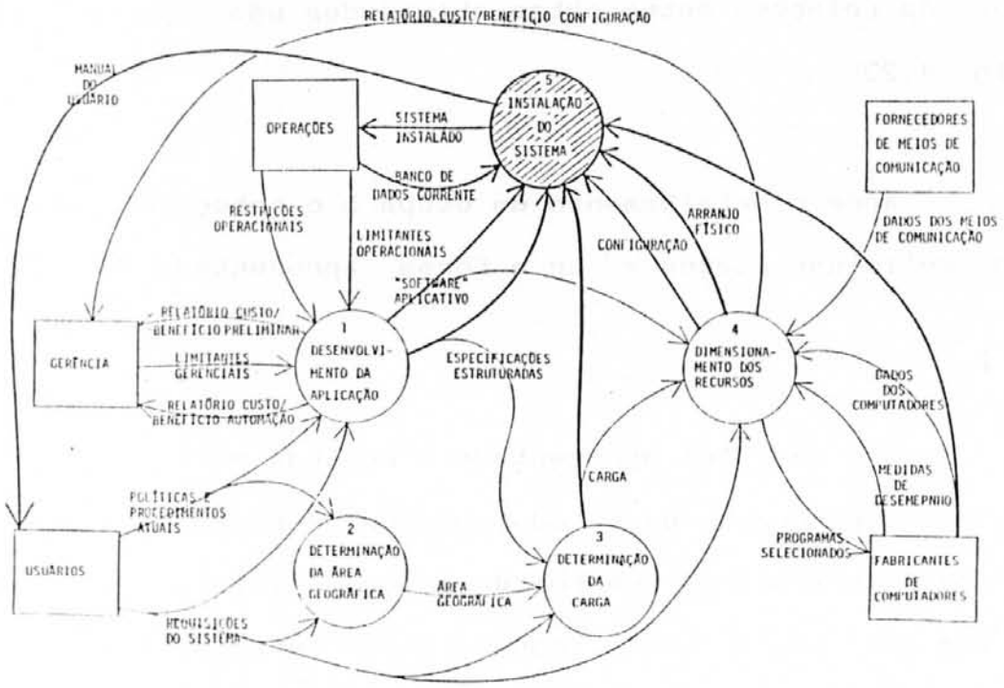


Fig. 4.24 Terceira evolução da metodologia preliminar

ção das entidades externas.

A terceira parte do processo consistiu em substituir a notação de DeMarco pela de Gane/Sarson, permitindo a descrição de fluxo material. As justificativas para esta mudança já foram apresentadas nos primeiros parágrafos da seção 4.3. Feita esta modificação, o esboço apresentado na fig. 4.24 foi modificado e metodologia pôde então ser representada conforme as fig. 4.12 e 4.13.

4.4 Resumo

Para configurar um sistema de computação não basta conhecer o "software" aplicativo a ser processado. É preciso conhecer também as tarefas executadas pelas pessoas, a distribuição geográfica e a intensidade com que a empresa troca informações e materiais com o meio ambiente. Uma metodologia de projeto de sistemas distribuídos, para ser efetiva, precisa levar em consideração todas estas variáveis. Na bibliografia não foi encontrada uma metodologia com tal característica.

Foi constatada a existência de três grupos de metodologias. No primeiro estão aquelas voltadas exclusivamente para o desenvolvimento de "software" [HIC78], [IFI82] e [YOU82]. No segundo grupo estão aquelas voltadas ao "hardware". Destacam-se as metodologias de projeto de sistemas distribuídos [BUC79], [STR80] e [AGA85]. Estas metodologias não são completas por não especificarem com obter e organizar os dados quantitativos para a execução do projeto. O terceiro

grupo é formado por metodologias de racionalização e organização do trabalho, tais como [BAR77], [MAY70a] e [ARA85]. Além destas metodologias destacam-se também o trabalho de [OHA79] sobre distribuição de arquivos e as metodologias estruturadas [YOU82]. Para tornar possível a execução do ciclo completo de projeto de sistemas distribuídos, foi proposta uma metodologia onde todos os trabalhos acima citados estão integrados.

A metodologia teve como ponto de partida os trabalhos de [STR80] e [OHA79]. São trabalhos voltadas ao projeto da configuração. Entretanto, para projetar a configuração é preciso conhecer o "software" da aplicação. Desta forma, aos trabalhos anteriores foi agregado o ciclo de vida estruturado [YOU82], uma metodologia de desenvolvimento de "software". Como a carga a ser submetida aos computadores que compõe a configuração não depende apenas do "software", mas também dos processos manuais, foram agregadas as metodologias sobre organização do trabaho das pessoas [MAY70a]. O resultado foi uma metododologia com cinco etapas.

As cinco etapas foram concebidas de modo a poderem ser executadas em paralelo como proposto em [YOU82]. Estas etapas são (1) desenvolvimento da aplicação, (2) determinação da área geográfica, (3) determinação da carga, (4) dimensionamento dos recursos e (5) instalação do sistema. No próximo capítulo elas são detalhadas.

5 DESCRIÇÃO DAS ETAPAS DA METODOLOGIA

5.1 Introdução

Os objetivos deste capítulo são: detalhar as etapas da metodologia proposta no capítulo anterior e descrever os procedimentos de obtenção e organização dos dados quantitativos necessários ao projeto de um sistema distribuído.

As etapas apresentadas no capítulo anterior (fig. 4.13), são detalhadas nas próximas seções.

5.2 Desenvolvimento da aplicação

O objetivo da primeira etapa é desenvolver o "software" da aplicação. O contexto desta etapa na metodologia é apresentado na fig. 5.1. O significado das entradas e saídas foram apresentados na seção 4.3.

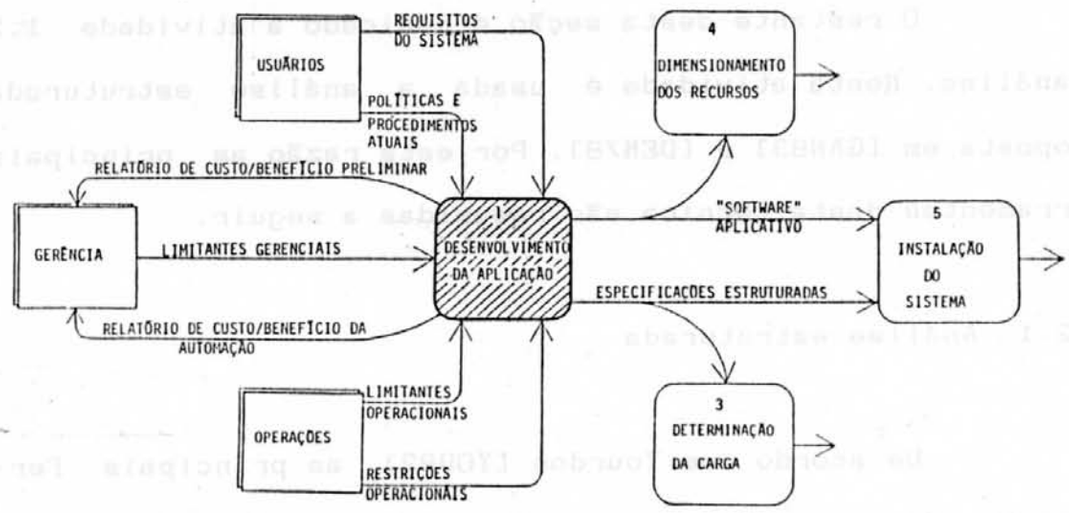


Fig. 5.1 Contexto da etapa de desenvolvimento da aplicação

Como foi visto na seção 4.3.3.2, a origem desta etapa é o ciclo de vida estruturado (fig. 4.18), sendo composta por seis atividades: (1) pesquisa, (2) análise, (3) projeto, (4) implementação, (5) geração de testes de aceitação e (6) controle de qualidade. Estas atividades foram descritas na seção 4.2.1. As relações entre elas são apresentadas na fig. 5.2.

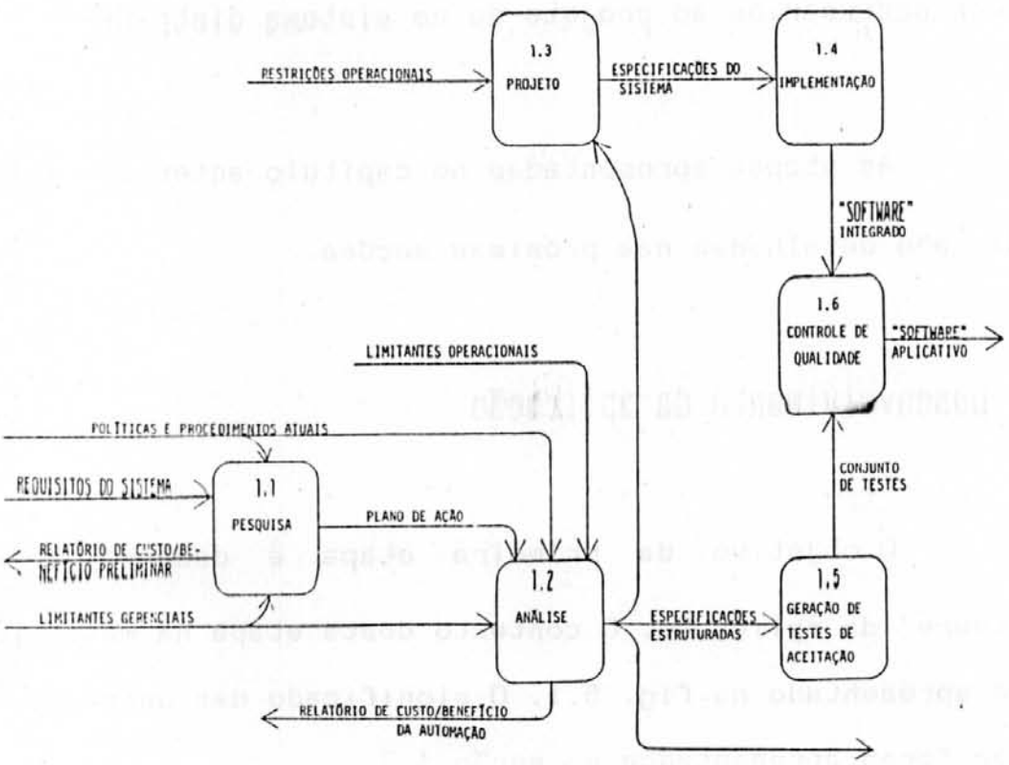


Fig. 5.2 Detalhes da etapa de desenvolvimento da aplicação

O restante desta seção é dedicado à atividade 1.2 - análise. Nesta atividade é usada a análise estruturada proposta em [GAN83] e [DEM78]. Por esta razão as principais ferramentas desta técnica são resumidas a seguir.

5.2.1 Análise estruturada

De acordo com Yourdon [YOU82], as principais ferramentas de documentação da análise estruturada são quatro: (1) diagrama de fluxo de dados, (2) dicionário de dados, (3)

diagrama de estrutura de dados e (4) português estruturado. Tais ferramentas são descritas a seguir.

A primeira ferramenta é o diagrama de fluxo de dados (DFD). Tal ferramenta vem sendo usada neste trabaho desde a seção 4.2. Existem duas notações, uma proposta por Gane/Sarson [GAN83] e outra de DeMarco [DEM78] (fig. 4.11). No restante do trabalho será usada a notação de Gane/Sarson. As razões para isso já foram apresentadas na seção 4.3. Um exemplo de uso do DFD para a modelagem de sistemas é apresentado pela sequência de figuras: fig. 5.3, fig. 5.4, e fig. 5.5.

A segunda ferramenta é o dicionário de dados (DD). Segundo [DEM78], o DD é usado para documentar o fluxo de dados, arquivos, processos e os elementos de dados. Na fig. 5.6. é dado um exemplo de descrição de arquivo.

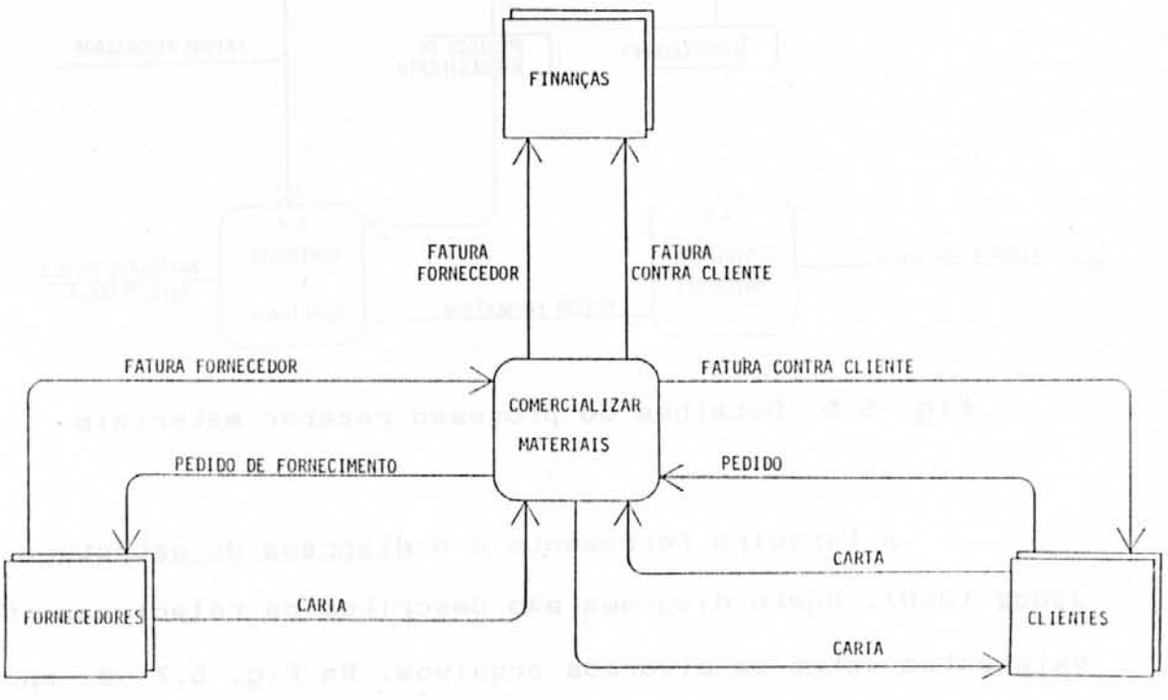


Fig. 5.3 Contexto da aplicação comercializar materiais

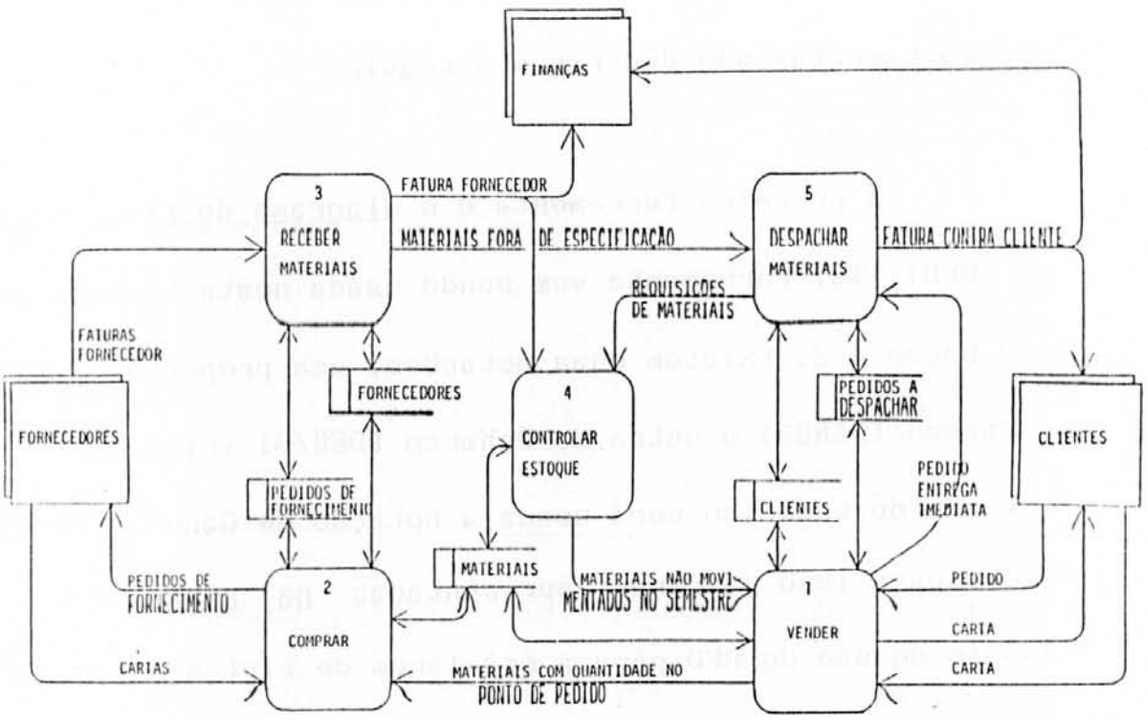


Fig. 5.4 Detalhes da aplicação comercializar materiais

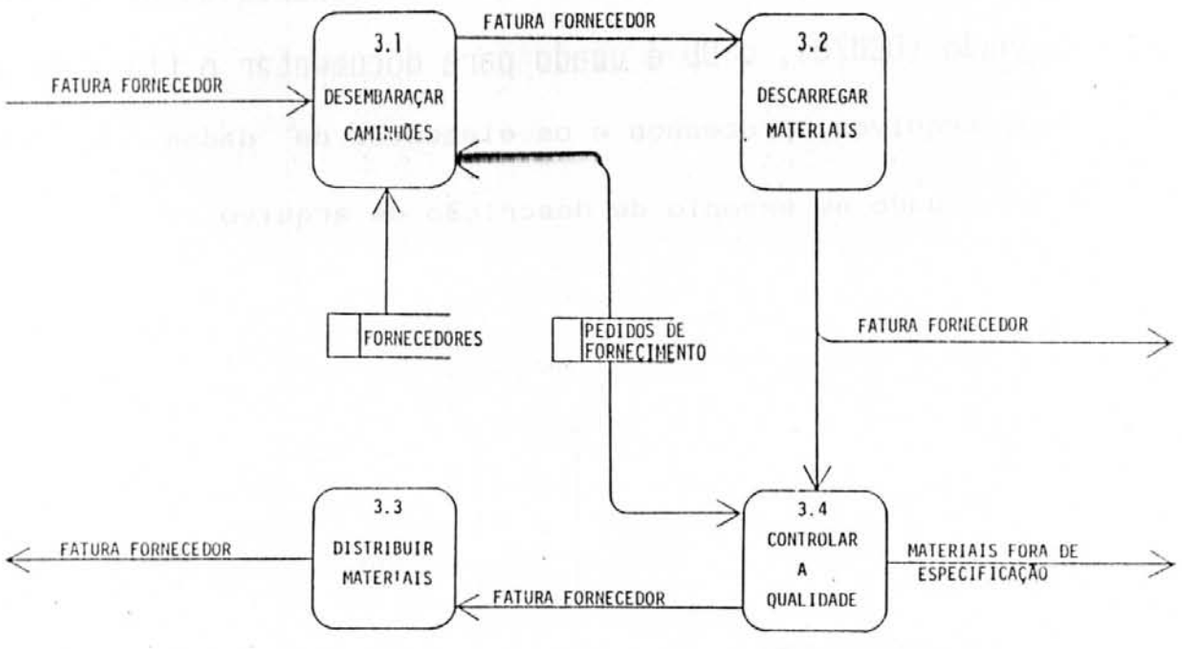


Fig. 5.5 Detalhes do processo receber materiais

A terceira ferramenta é o diagrama de estrutura de dados (DED). Neste diagrama são descritos os relacionamentos existentes entre os diversos arquivos. Na fig. 5.7 é apresentado um exemplo.

PEDIDOS-DE-FORNECIMENTO = { NÚMERO-PEDIDO-FORNECIMENTO +
NÚMERO-FORNECEDOR +
CÓDIGO-MATERIAL +
DATA-EMISSÃO +
PREÇO-UNITÁRIO-CORRENTE +
({ PREÇO-UNITÁRIO-ANTERIOR +
DATA-PREÇO-UNITÁRIO } +
CÓDIGO-COMPRADOR) +
[DÍGITO-CANCELAMENTO /
DÍGITO-PEDIDO-ABERTO] }

LEGENDA:

X = a + b ⟹ "X" é composto de "a" e "b"
X = [a / b] ⟹ "X" é composto de "a" ou "b"
X = (a) ⟹ o elemento "a" é opcional
X = {a} ⟹ "X" é composto de zero ou mais ocorrências de "a"

Fig. 5.6 Dicionário de dados

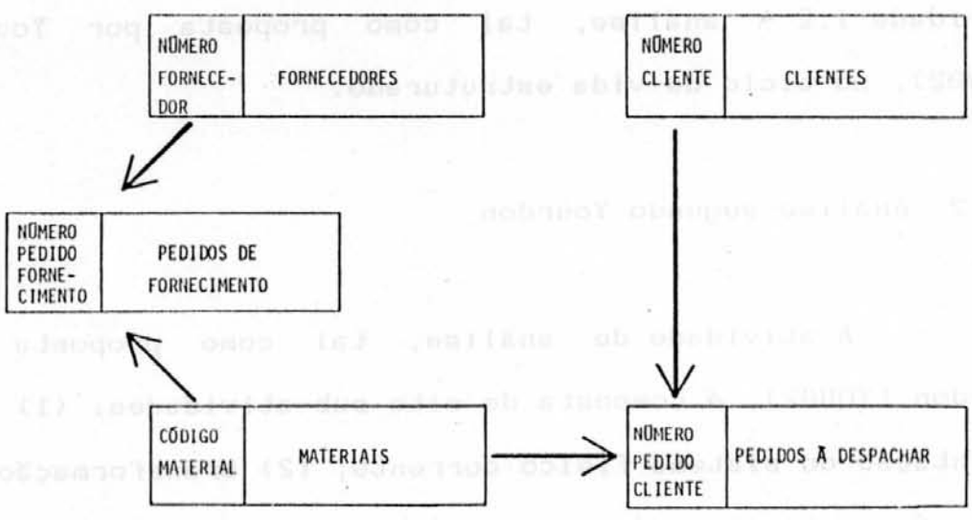


Fig. 5.7 Diagrama de estrutura de dados

A quarta ferramenta é o português estruturado. Tal ferramenta é usada para descrever os processos do último nível do DFD. O conjunto destas descrições são chamadas por Yourdon [YOU82] de mini-especificações (mini-espec). Na fig. 5.8 é dado um exemplo.

```
NÚMERO DO PROCESSO : 1.1.1
NOME DO PROCESSO   : AUTORIZAR A DESCARGA DE VEÍCULOS

                PARA CADA FATURA FAÇA:
                    VERIFIQUE SE O FORNECEDOR EXISTE;
                    SE EXISTIR
                       ENTÃO PARA CADA PEDIDO FAÇA:
                                VERIFIQUE SE O PEDIDO EXISTE;
                                SE EXISTIR
                                   ENTÃO IMPRIMA 'PEDIDO OK";
                                   SENÃO IMPRIMA "PEDIDO NÃO EXISTE";
                       SENÃO IMPRIMA "FORNECEDOR NÃO EXISTE";
                            IMPRIMA "NÃO DESCARREGAR";
                    SE NA FATURA HÁ PEDIDO NÃO EXISTENTE
                       ENTÃO IMPRIMA "DESCARGA SOB AUTORIZAÇÃO";
                       SENÃO IMPRIMA "DESCARREGAR";
                SE EXISTE FATURA A "DESCARREGAR"
                   ENTÃO INSTRUA O MOTORISTA QUANTO À DESCARGA;
                   SENÃO PEÇA AO MOTORISTA PARA RETIRAR O VEÍCULO.
```

Fig. 5.8 Português estruturado

Até aqui foram apresentadas as principais ferramentas da análise estruturada. Na próxima seção é descrita a atividade 1.2 - análise, tal como proposta por Yourdon [YOU82], no ciclo de vida estruturado.

5.2.2 Análise segundo Yourdon

A atividade de análise, tal como proposta por Yourdon [YOU82], é composta de oito sub-atividades: (1) documentação do sistema físico corrente, (2) transformação no equivalente lógico, (3) definição do novo sistema lógico, (4) estabelecimento das fronteiras homem/máquina, (5) análise de custo/benefício, (6) seleção de opções, (7) estabelecimento de limitantes e (8) especificação do sistema ("package specifications"). Uma visão geral destas sub-atividades é apresentada na fig. 5.9. A seguir tais sub-atividades são resumidas.

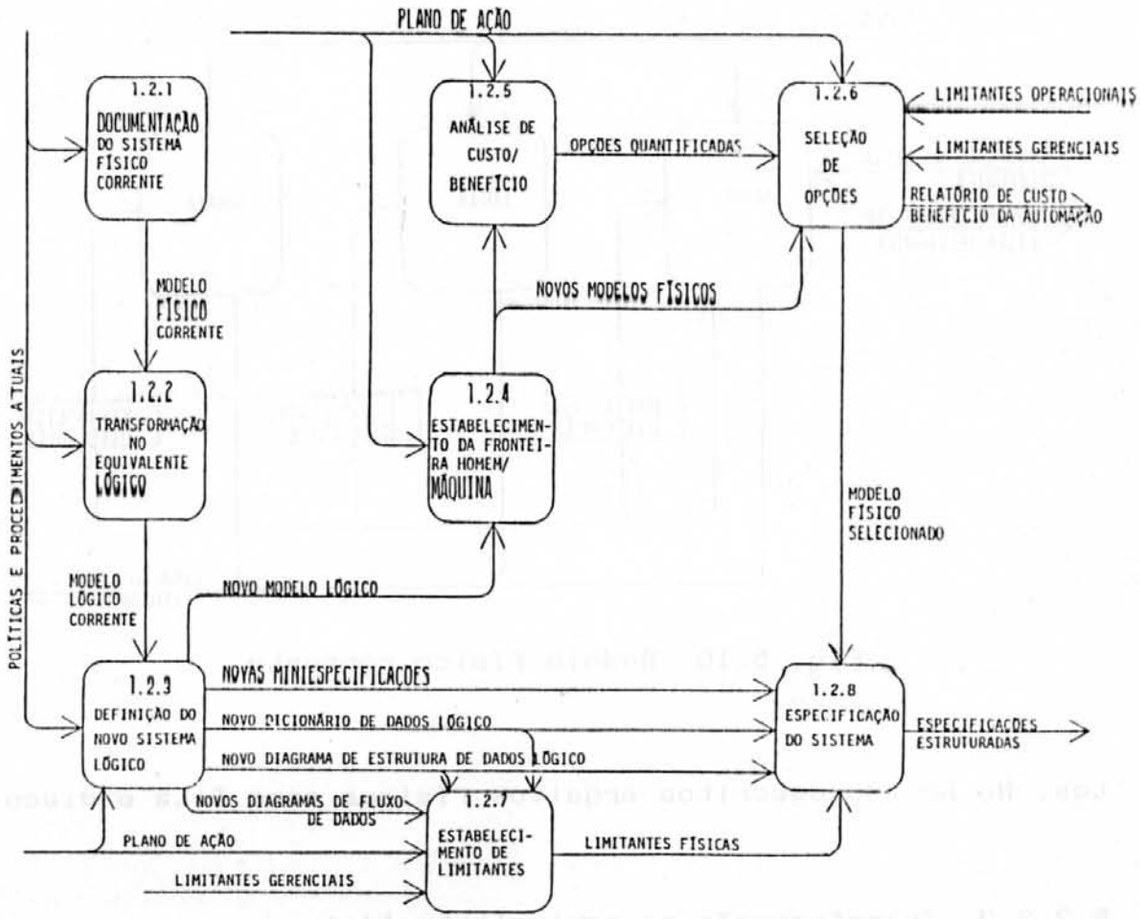


Fig. 5.9 Detalhes da atividade de análise

5.2.2.1 Documentação do sistema físico corrente

Esta é a primeira sub-atividade da análise (fig. 5.9) e tem como objetivo documentar o ambiente do usuário, tal qual existe no momento. O produto é o modelo físico corrente.

Modelo_físico_corrente: é o conjunto de DFDs e DD onde estão descritos o ambiente atual do usuário bem como o sistema computadorizado, se houver. As bolhas dos DFDs do ambiente atual tem nome de departamentos ou pessoas ao invés de funções (fig. 5.10). No DD são descritos formulários tais como faturas, requisições, cheques e outros. Nos DFDs da parte computadorizada estão descritos os programas existen-

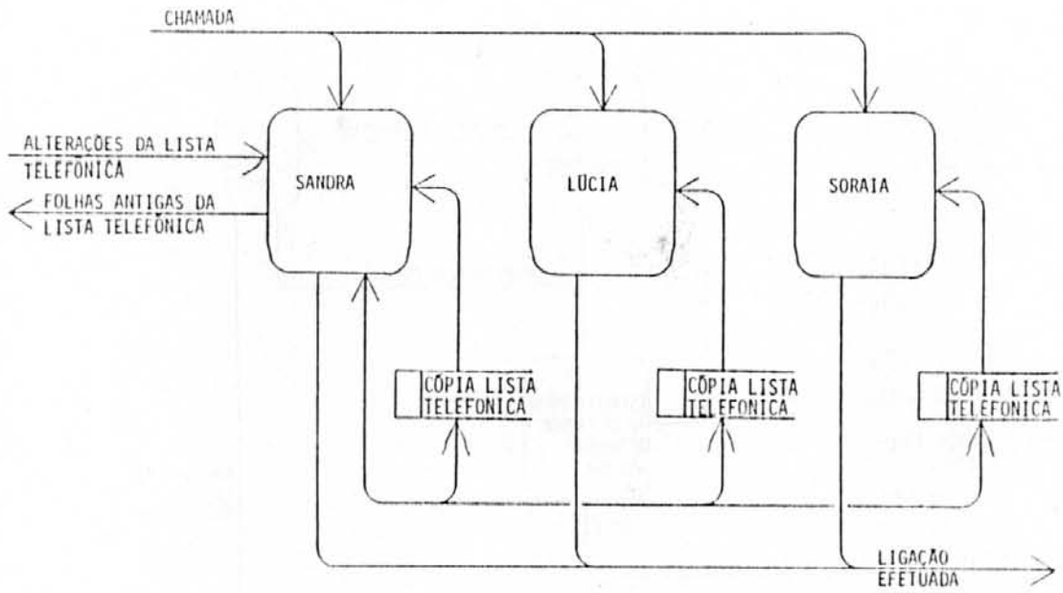


Fig. 5.10 Modelo físico corrente

tes. No DD são descritos arquivos físicos como fita e disco.

5.2.2.2 Transformação no equivalente lógico.

Esta é a segunda sub-atividade da análise (fig. 5.9) e tem como objetivo derivar o modelo físico no seu equivalente lógico. Um sistema lógico é livre de características ou restrições de qualquer implementação particular [PAGBO]. O produto é o modelo lógico corrente.

Modelo lógico corrente: é composto por um conjunto de DFDs, DD e mini-espec onde o sistema corrente é apresentado em termos lógicos, sem qualquer referência aos meios físicos através dos quais os os negócios são conduzidos.

5.2.2.3 Definição do novo sistema lógico

Esta é a terceira sub-atividade da análise (fig. 5.9) e tem como objetivo eliminar deficiências e incorporar

novas funções ao modelo lógico corrente. Caso o usuário tenha solicitado o novo sistema apenas para eliminar deficiências físicas, tais como alto tempo de resposta, dificuldade de manutenção e outros, esta sub-atividade não precisa ser executada. Isto entretanto é muito raro acontecer [YOU82]. O principal produto é o novo modelo lógico.

Novo_modelo_lógico: é composto de novos DFDs, DD, DED e mini-espec. No caso da sub-atividade 1.2.3 não ser executada, o novo modelo lógico é igual ao modelo lógico corrente.

5.2.2.4 Estabelecimento da fronteira homem/máquina

Esta é a quarta sub-atividade da análise (fig. 5.9) e tem como objetivo gerar opções de implementação. Os processos do novo modelo lógico são separados por uma linha chamada fronteira homem-máquina. Os processos que ficam dentro desta fronteira serão automatizados. O restante será executado por pessoas. Os produtos são várias opções de novos modelos físicos.

Novo_modelo_físico: é composto de DFDs e DD do novo sistema, onde estão identificados os processos e dados que serão automatizados ou executados manualmente. Na fig. 5.11 é apresentado um exemplo de DFD com tais características.

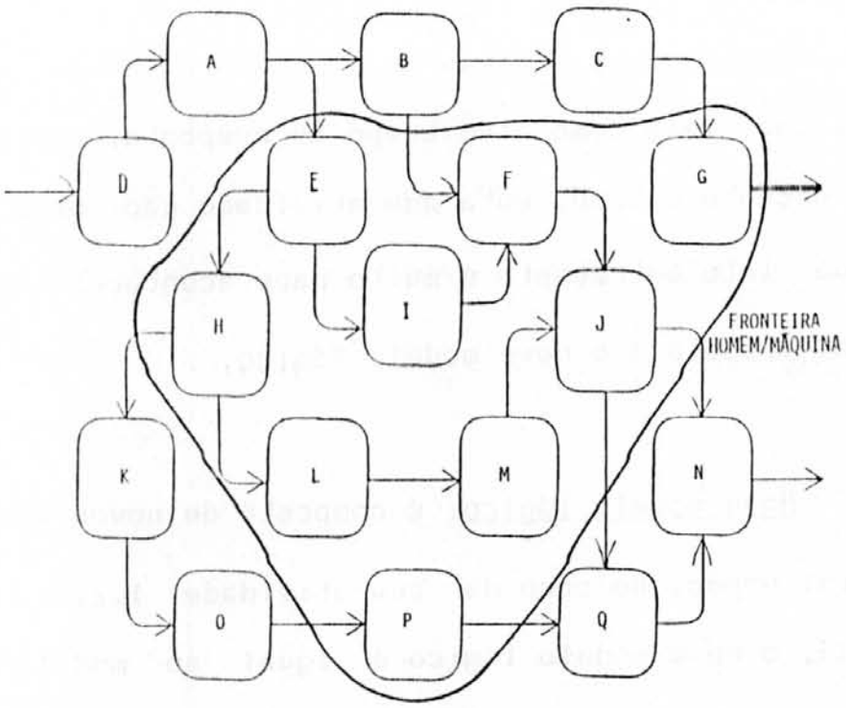


Fig. 5.11 DFD com a fronteira homem/máquina

5.2.2.5 Análise de custo benefício

Esta é a quinta sub-atividade da análise (fig. 5.9). Tem como objetivo preparar um relatório de custo/benefício para cada opção de implementação gerada pela sub-atividade anterior.

5.2.2.6 Seleção de opções

Esta é a sexta sub-atividade da análise (fig. 5.9) e tem como objetivo selecionar a melhor opção de implementação.

5.2.2.7 Estabelecimento de limitantes

Esta é a sétima sub-atividade da análise (fig. 5.9) e tem como objetivo estabelecer determinados limites. Tais limites são por exemplo tempo de resposta de alguns

processos, horário de entrega de informações críticas e outros.

5.2.2.8 Especificação do sistema

Esta é a oitava sub-atividade da análise (fig. 5.9) e tem como objetivo reunir os documentos produzidos pelas sub-atividades anteriores. O produto é constituído pelas especificações estruturadas.

Especificações estruturadas: é o conjunto de DFDs, DD, DED, miniespec e restrições físicas. As especificações estruturadas constituem-se no principal produto da atividade de análise.

Até aqui foi resumida a atividade de análise tal como proposta em [YOU82]. Na próxima seção são descritas as modificações efetuadas nesta atividade, para adequá-la à metodologia de desenvolvimento de sistemas distribuídos que está sendo proposta neste trabalho.

5.2.3 Alterações na sub-atividade de análise

Neste trabalho, a atividade de análise tal como proposta por Yourdon [YOU82] sofreu duas alterações. A primeira é a obrigatoriedade de representar o fluxo de materiais nos modelos físico e lógico. As justificativas desta alteração foram apresentadas nos primeiros parágrafos da seção 4.3. Os modelos que inicialmente tinham dois componentes (DFD e DD) passaram a ter um terceiro: DFDM - diagrama de

fluxo de dados e materiais. A segunda alteração se refere à criação de um novo produto: o modelo homem/máquina. Este modelo é projetado a partir do modelo_físico_selecionado, que é uma saída da sub-atividade seleção_da_opção (fig. 5.9).

Um exemplo de modelo físico é apresentado na fig. 5.11. Como é possível notar, existe uma linha divisória entre os processos manuais e automatizados. Para a criação do modelo homem/máquina, está sendo proposta a substituição desta linha por um novo componente: diagrama de descrição do método (DDM). Desta forma o modelo homem/máquina passa a ser formado por DFDM, DDM, DFD, DD e mini-espec. O novo modelo é descrito na próxima seção.

5.2.3.1 Modelo homem/máquina

Como foi visto, este modelo é obtido a partir de um modelo físico selecionado, produto da sub-atividade seleção_de_opções (fig. 5.9). O processo de transformação de um modelo no outro é feito na especificação_do_sistema, que é a última sub-atividade da análise (fig. 5.9). Consequentemente, as especificações estruturadas, único produto desta sub-atividade, também é alterada. As especificações, segundo Yourdon [YOU82] eram: DFDs, DD, DED, mini-espec e restrições físicas. Com a alteração que está sendo proposta o conteúdo passa a ser: modelo homem/máquina, DED e restrições físicas.

No modelo homem/máquina o sistema é descrito em níveis de abstração. O nível (0) é um DFDM onde é apresentado o contexto da aplicação (fig. 5.12). O diagrama contex-

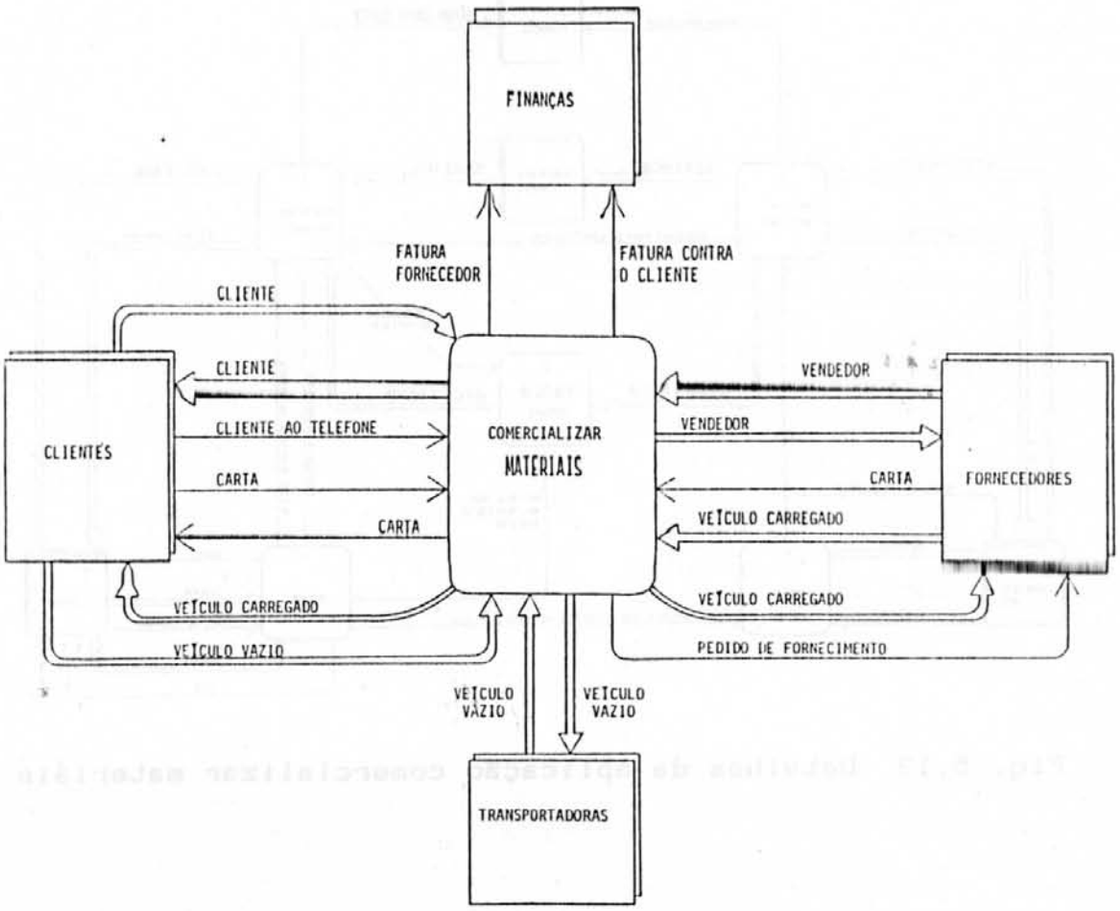


Fig. 5.12 Contexto da aplicação comercializar materiais

tual é decomposto em múltiplos processos, dando origem ao nível (1). Um exemplo é apresentado na fig. 5.13. Os processos do nível (1) são então sucessivamente detalhados em níveis inferiores (fig. 5.14). O detalhamento pára na fronteira homem/máquina.

O último nível do DFDM é composto por processos manuais. Tais processos devem ser implementados segundo as técnicas de Métodos e Tempos, que é oriunda da Engenharia Industrial [MAY70a], [BAR77]. Para tanto está sendo proposto o diagrama de descrição do método (DDM), conforme descrito na próxima seção.

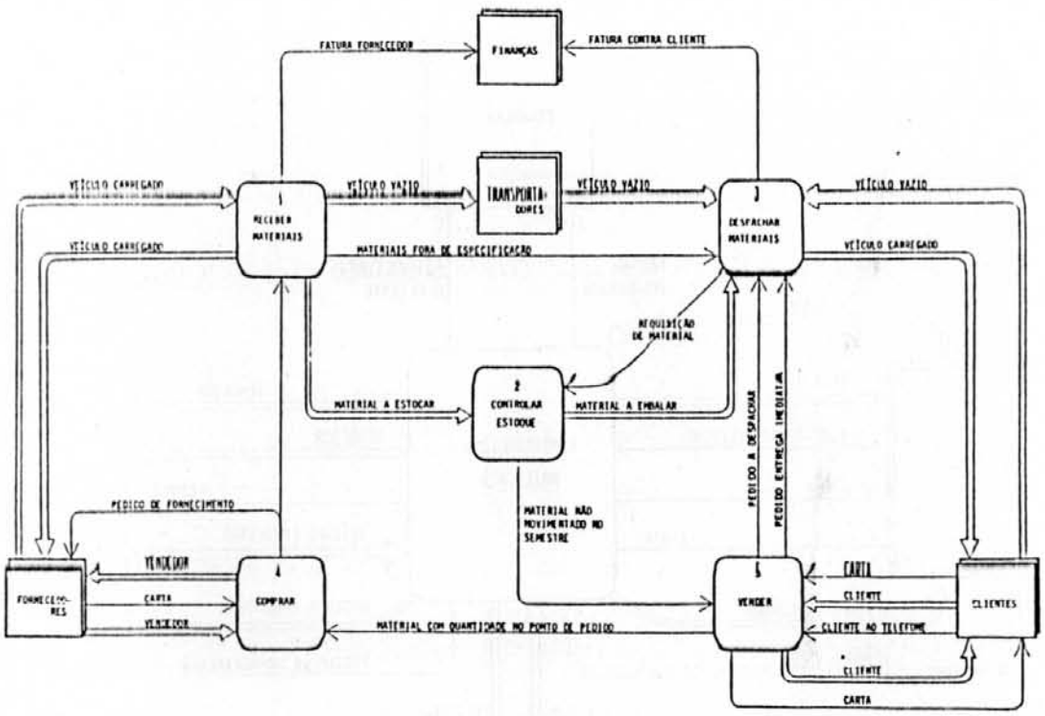


Fig. 5.13 Detalhes da aplicação comercializar materiais

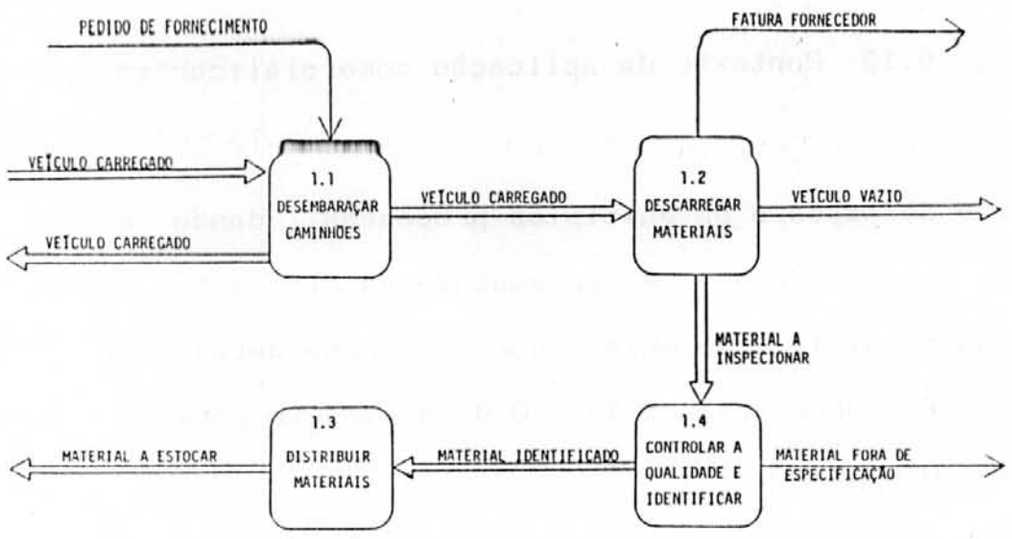


Fig. 5.14 Detalhes do processo receber materiais

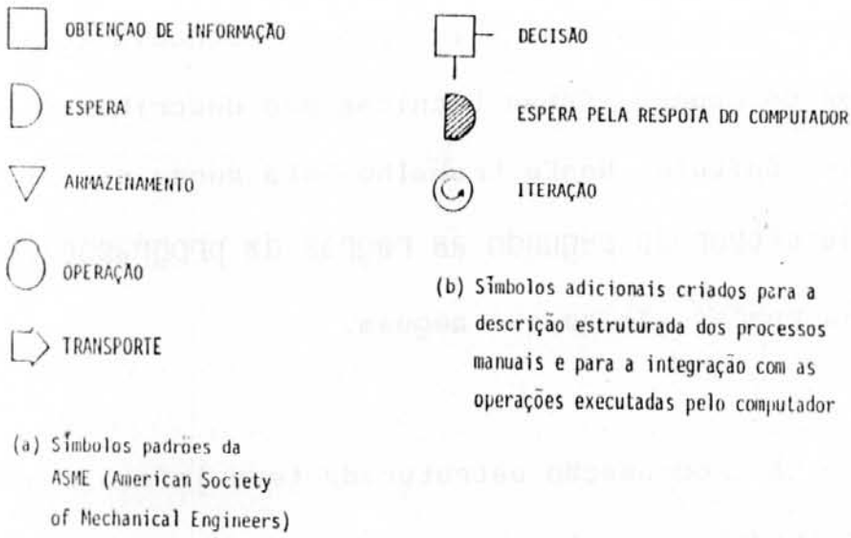
5.2.3.1.1 Diagrama de descrição do método

O DDM tem o objetivo de descrever, passo a passo, como uma pessoa deve proceder para executar um processo manual. O DDM é um diagrama consagrado tanto na Engenharia Industrial [MAY70a], [BAR77], como em Organização & Métodos

[ARA85]. O diagrama tradicional é feito com base na notação gráfica de Gilbreth e no princípio de economia de movimentos e tempos de espera. Estas técnicas são descritas mais adiante neste capítulo. Neste trabalho está sendo sugerido que o DDM seja elaborado segundo as regras da programação estruturada. As razões são as que seguem.

A programação estruturada teve início com Dijkstra em 1968 [DIJ68], revolucionou a arte de programar [HOL77], [MAR77], [GEL77] e foi definitivamente consagrada nas novas linguagens de programação [WIR74], [BAR82]. O uso desta técnica permite fazer programas com apenas três estruturas básicas: sequência, repetição e seleção. Um processo manual, como um programa, é uma sequência de operações. O uso da programação estruturada na elaboração do DDM é portanto adequada, possibilitando dar ao mesmo um formato simples e uma descrição clara e impessoal. O "inglês estruturado" de DeMarco [DEM78] é um trabalho semelhante.

Gilbreth foi contemporâneo de Taylor [TAY79] e ganhou notoriedade com a racionalização dos trabalhos da construção civil [BAR77], [GRO73]. Criou uma notação gráfica que foi aperfeiçoada e adotada como padrão pela ASME (American Society of Mechanical Engineers), cujo significado é descrito na fig. 5.15a. Esta notação facilita o uso do princípio de economia de movimentos e tempos de espera. Tal princípio parte da premissa de que se a pessoa está esperando, ou apenas movimentando-se, não está realizando trabalho útil. Uma das formas de racionalizar a execução do processo é aplicar este princípio.



(a) Símbolos padrões da ASME (American Society of Mechanical Engineers)

(b) Símbolos adicionais criados para a descrição estruturada dos processos manuais e para a integração com as operações executadas pelo computador

Fig. 5.15 Símbolos usados na descrição do método

A integração destas ferramentas (programação estruturada e a notação de Gilbreth) exigiu algumas adaptações, sendo criados três novos símbolos (fig. 5.15b). Para representar uma decisão foi adotado o símbolo , que é uma versão modificada de "obtenção de informação". Assim, logo após o mesmo, o procedimento bifurca-se em ramos diferentes (fig. 5.16). Para a iteração foi criado o símbolo . Sempre que o mesmo aparece, o procedimento forma um ciclo (fig. 5.16). Finalmente, como o processo é executada com o auxílio de um computador, foi criado o símbolo . Ele indica que enquanto a pessoa está esperando, o computador está processando as informações necessárias ao prosseguimento do trabalho (fig. 5.16). O detalhamento deste símbolo é descrito na próxima seção.

5.2.3.1.2 Descrição dos processos automatizados

Como foi visto anteriormente, o símbolo representa a execução de um processo automatizado. Estes proces-

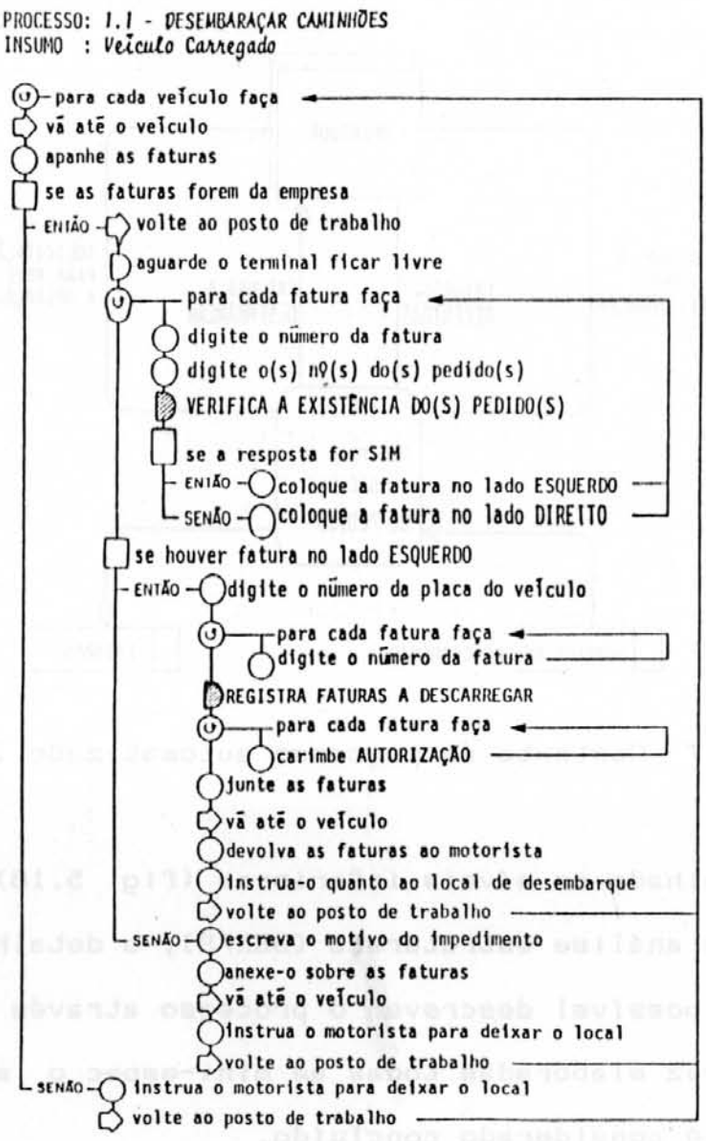


Fig. 5.16 Diagrama de descrição do método (DDM)

sos devem ser descritos através das ferramentas da análise estruturada.

Para cada símbolo é elaborado um diagrama contextual. Neste diagrama há três tipos de entidades que interagem com o processo em questão. O primeiro tipo é o operador, que é a pessoa responsável pela execução do método descrito no DDM. O segundo tipo são os arquivos. O terceiro tipo são outros processos automatizados tais como rotinas de cálculo de desvio padrão e outras. Um exemplo do DFD contextual é apresentado na fig. 5.17. Este diagrama pode ser no-

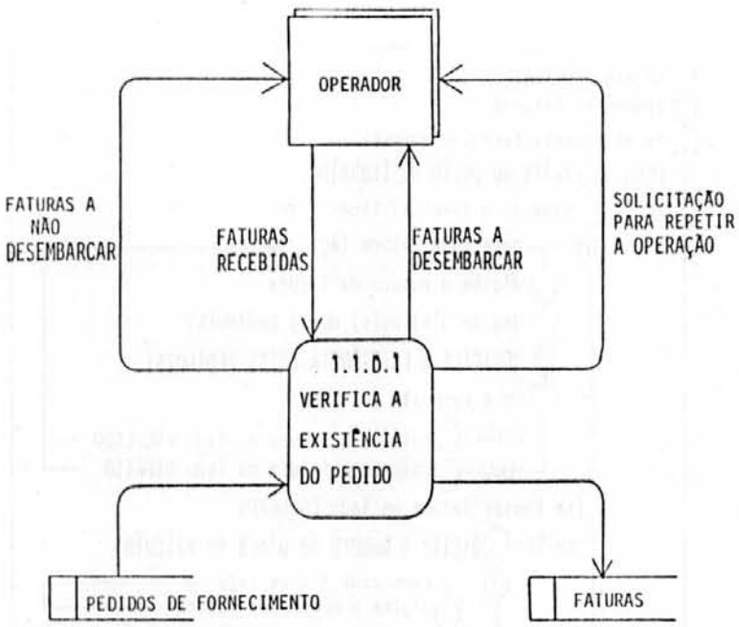


Fig. 5.17 Contexto do processo automatizado 1.1.D.1

vamente detalhado em níveis inferiores (fig. 5.18). Segundo as regras da análise estruturada [DEM78], o detalhamento pára quando é possível descrever o processo através das mini-espec. Uma vez elaboradas todas as mini-espec o modelo homem/máquina é considerado concluído.

Até aqui foi descrito o modelo homem/máquina. Na próxima seção são apresentados os motivos para a criação deste modelo.

5.2.3.2 A necessidade de um novo modelo

Como foi visto na seção 5.2.2, os principais produtos da análise são quatro modelos: (1) modelo físico corrente, (2) modelo lógico corrente, (3) novo modelo lógico e (4) novo modelo físico. Cada modelo é derivado do anterior, exceto o primeiro que é um modelo do ambiente existente. O primeiro modelo é transformado no segundo pela eliminação

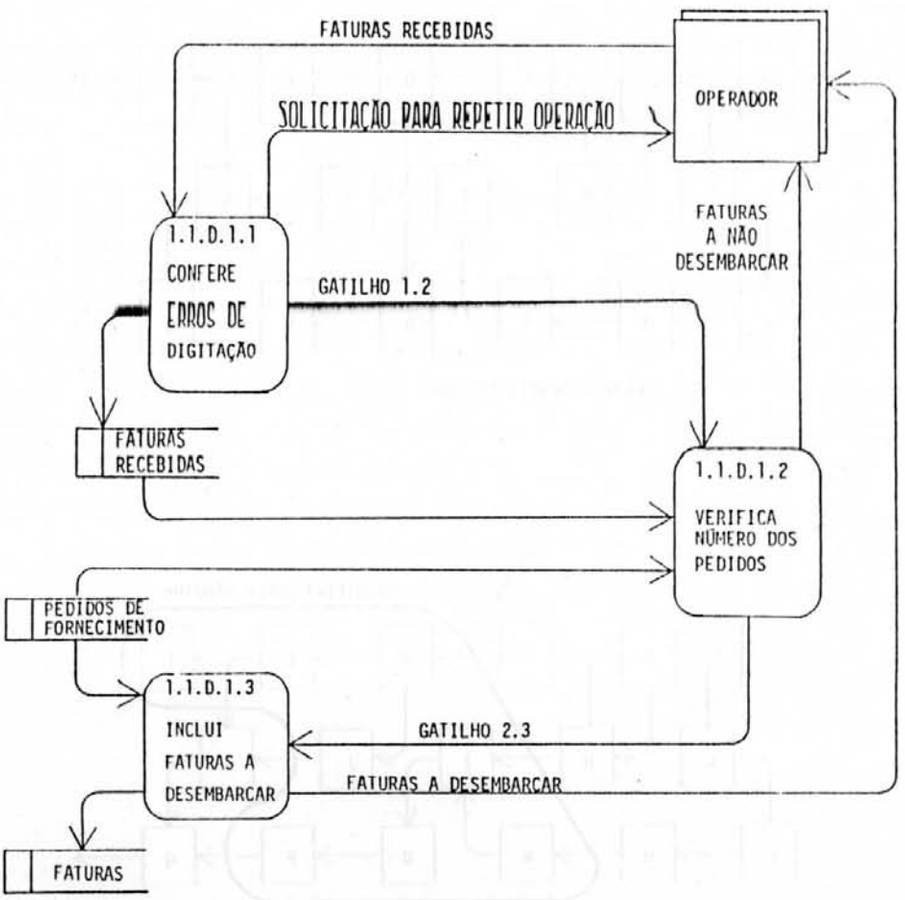
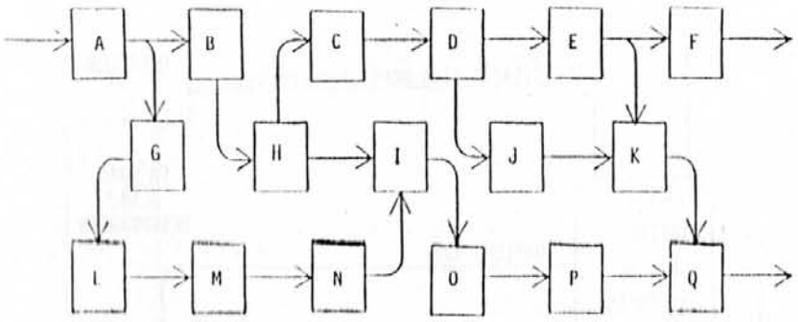


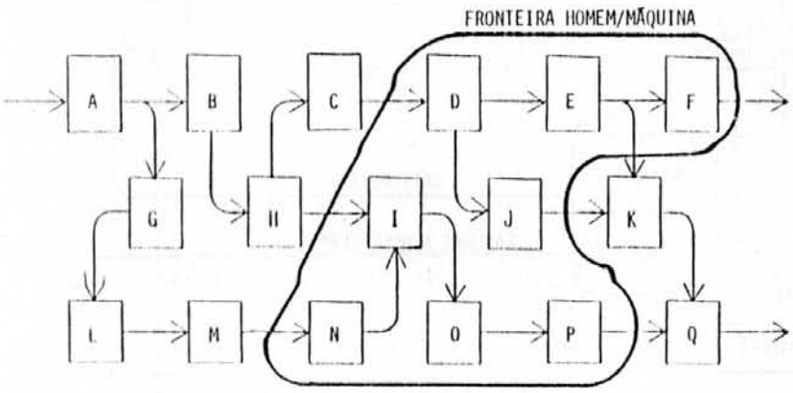
Fig. 5.18 Detalhes do processo automatizado 1.1.D.1

das características e restrições de implementação. A passagem do segundo para o terceiro é um processo de inclusão de algumas funções e eliminação de outras. O quarto modelo é uma evolução do terceiro. A maneira de obtê-lo é descrita a seguir.

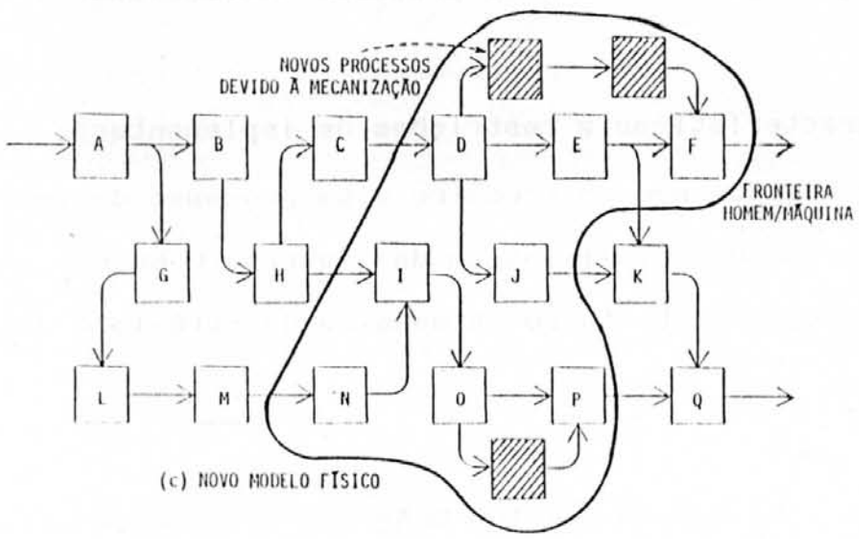
De acordo com Tom DeMarco, a concepção do novo modelo físico tem duas etapas. A primeira consiste em traçar sobre o modelo lógico uma fronteira separando a parte automatizada da manual. Na segunda etapa, a parte automatizada é ampliada pela inclusão de novos processos. Tais processos são necessários devido às características do computador no qual o sistema será implementado. Um exemplo destas transformações é apresentado na fig. 5.19. Como é possível notar,



(a) NOVO MODELO LÓGICO



(b) NOVO MODELO LÓGICO COM A FRONTEIRA HOMEM/MÁQUINA



(c) NOVO MODELO FÍSICO

Fig. 5.19 Transformação do modelo lógico em físico

a partir da segunda etapa as características do ambiente começam a ser incorporadas no sistema. Entretanto, a transformação tal como proposta por DeMarco só leva em consideração a influência do computador. Como será visto à seguir, este processo de transformação deve ser mais amplo, pois há uma

interdependência entre a implementação da parte manual e da automatizada.

Seja por exemplo o caso da operadora de PBX descrito no início da seção 4.1: os modelos lógico e físico desta aplicação são apresentados na fig. 5.20. O modelo físico não está completo porque ainda não foi executada a segunda etapa da transformação. Como é possível notar, duas

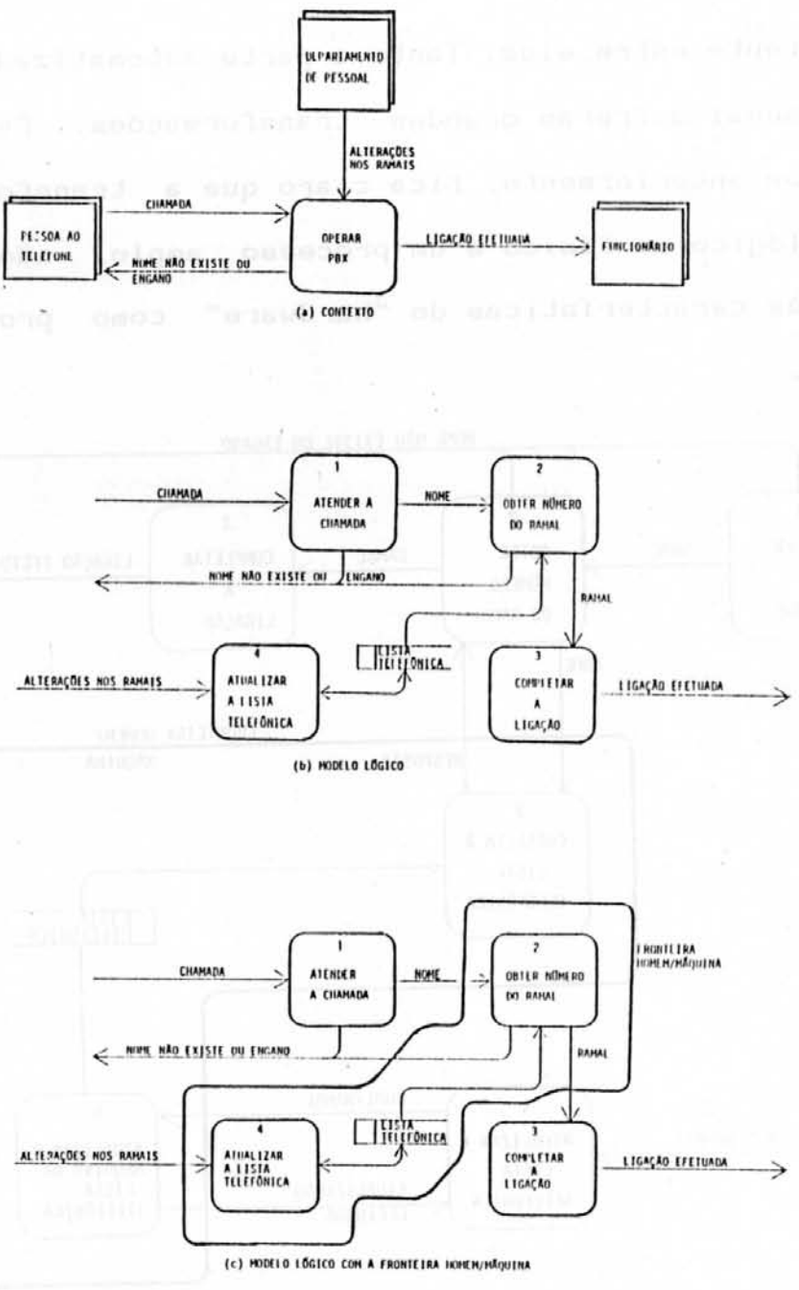


Fig. 5.20 DFD's da aplicação operar PBX

funções deverão ser automatizadas. Há pelo menos duas formas de implementação. A primeira é "on-line". O modelo físico resultante é apresentado na fig. 5.21. Comparando este modelo com o modelo lógico (fig. 5.20) é possível notar que os novos processos não foram acrescentados na parte automatizada, e sim na parte manual. A segunda forma de implementação é "batch". Na fig. 5.22 é apresentado o modelo resultante desta implementação. Uma comparação entre este modelo e o apresentado na fig. 5.20, demonstra a significativa diferença existente entre eles. Tanto a parte automatizada como a parte manual sofreram grandes transformações. Pelos fatos descritos anteriormente, fica claro que a transformação do modelo lógico em físico é um processo amplo, não limitado apenas às características do "hardware" como proposto por DeMarco.

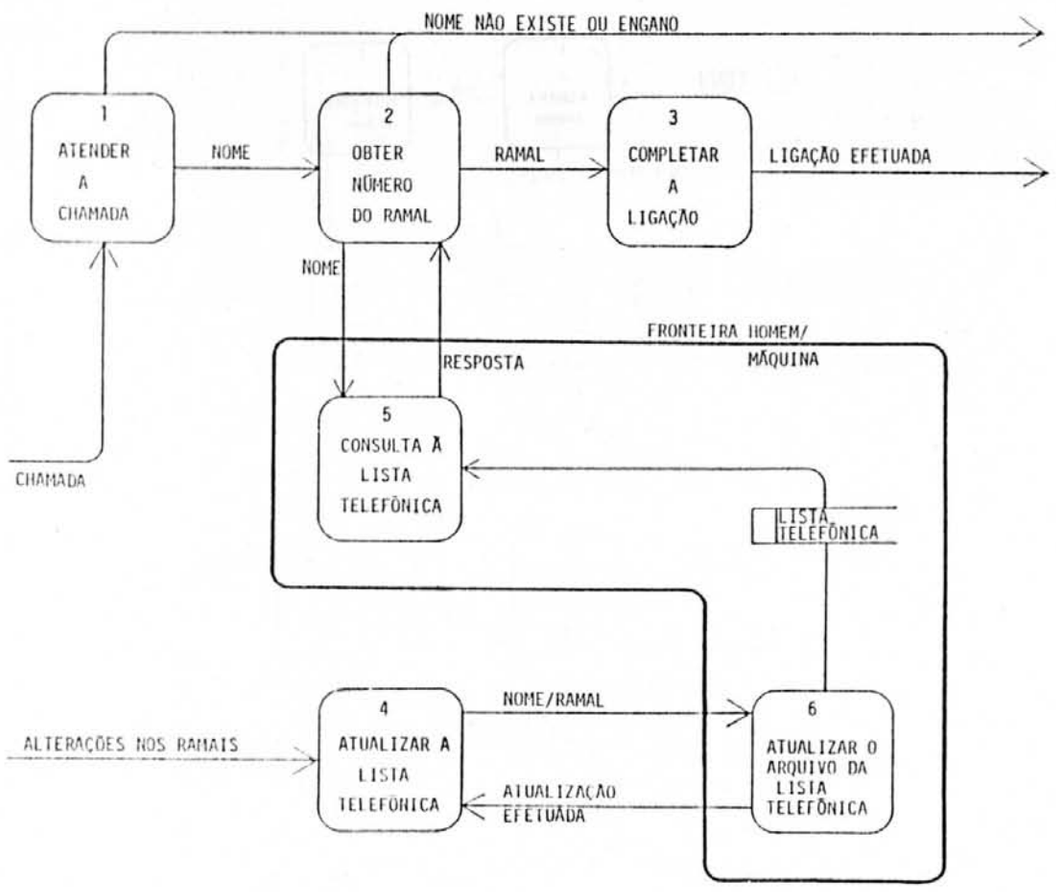


Fig. 5.21 Solução "on-line" da aplicação operar PBX

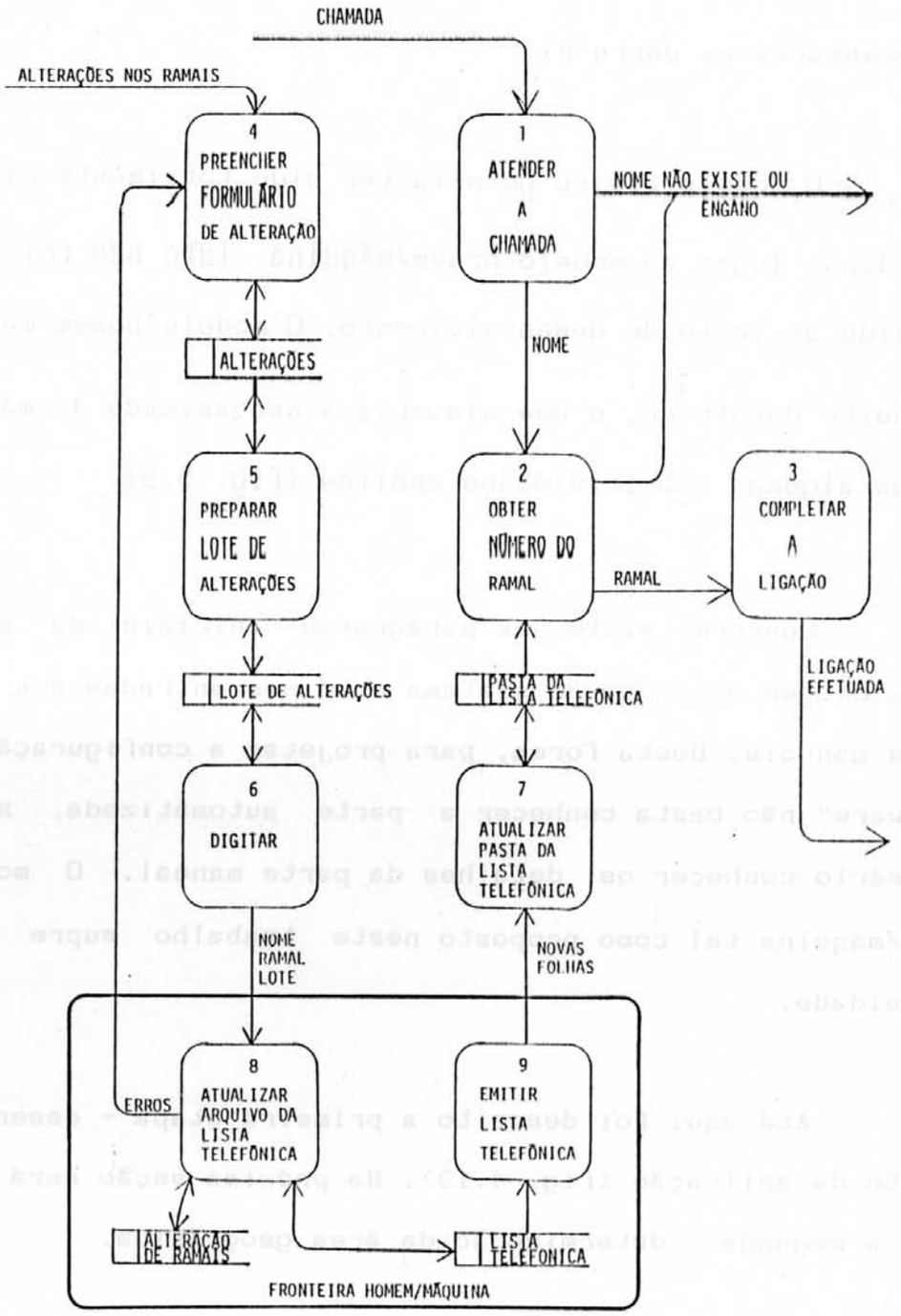


Fig. 5.22 Solução "batch" da aplicação operar PBX

Neste trabalho está sendo proposto que a transformação do modelo físico em lógico se limite apenas à primeira etapa, ou seja, ao traçado da fronteira entre os processos manuais e automatizados. A segunda etapa, inclusão de novos processos automatizados, deixa de existir, dando lugar à transformação do modelo físico no modelo homem/máquina. Como foi visto na seção 5.2.3.1.1, nesta transformação tanto a

parte manual como a automatizada são concebidas de forma a complementarem-se entre si.

O modelo físico poderia ter sido totalmente eliminado, dando lugar ao modelo homem/máquina. Isto não foi feito devido ao custo de desenvolvimento. O modelo homem/máquina é muito detalhado, o que elevaria a necessidade de mão de obra de algumas sub-atividades análise (fig. 5.9).

Conforme visto nos parágrafos iniciais da seção 4.3, a origem da carga do sistema está nas entradas dos processos manuais. Desta forma, para projetar a configuração do "hardware" não basta conhecer a parte automatizada, sendo necessário conhecer os detalhes da parte manual. O modelo homem/máquina tal como proposto neste trabalho supre esta necessidade.

Até aqui foi descrito a primeira etapa - desenvolvimento da aplicação (fig. 4.13). Na próxima seção será descrita a segunda - determinação da área geográfica.

5.3 Determinação da área geográfica

O objetivo da segunda etapa é identificar os locais por onde a empresa está dispersa. O contexto desta etapa na metodologia é apresentado na fig. 5.23. O significado das entradas e saídas foram descritas na seção 4.3.

Os produtos da segunda etapa são mapas, que descrevem em vários níveis, a área geográfica coberta pela

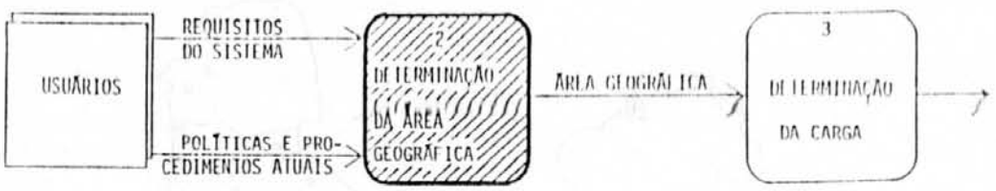


Fig. 5.23 Contexto da etapa determinação da área geográfica

aplicação. Esses mapas fornecem nomes, distâncias e a disposição física dos recursos ("layout"). Os nomes (cidades, bairros e ruas) são usados na determinação da carga (terceira etapa). A distância é uma variável fundamental no projeto da topologia [DOL78]. O "layout" tem influência no tempo de execução dos processos manuais, conforme será mais visto mais adiante neste capítulo.

A área geográfica é decomposta em níveis, de modo semelhante à decomposição funcional da análise estruturada [GAN83], [WEI80], [DEM78]. Cada nível tem um nome : território, sítio, prédio, andar, sala e posto de trabalho (fig. 5.24).

O território é uma fração do sistema geopolítico mundial. Suas fronteiras encerram as cidades onde a empresa mantém atividade, delimitando a área geográfica coberta pela aplicação. É conveniente usar as fronteiras políticas devido às diferenças legais (transmissão de dados e impostos entre outros) entre países, estados e cidades. Além disso, é comum designar as filiais das empresas (e suas matrizes) pelo nome da cidade, do bairro ou da rua onde estão situadas. O Mapa do território (fig. 5.24b) é também necessário para o projeto da topologia.

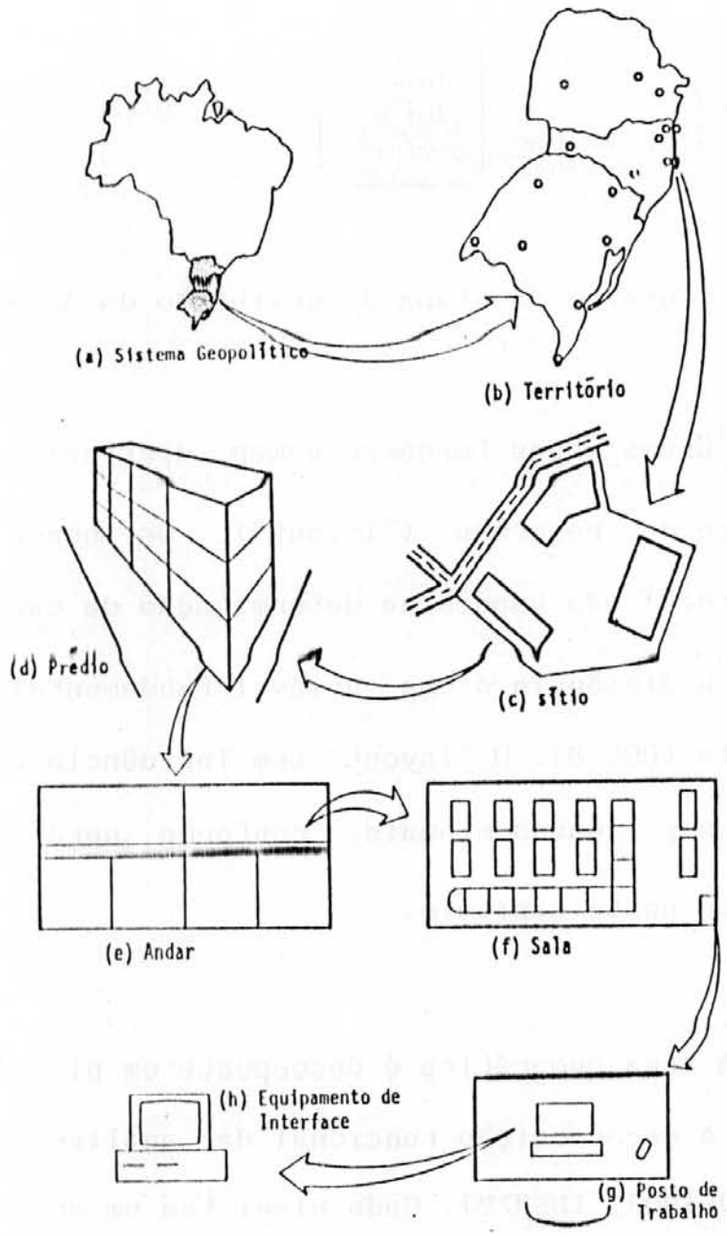


Fig. 5.24 Mapas dos vários níveis da área geográfica

Um sítio, é uma porção delimitada de uma cidade onde são desenvolvidas as tarefas de uma empresa. Por exemplo, numa rede de supermercados, cada loja é um sítio. Se o local onde a empresa está situada for cortado por áreas públicas (estradas e praças), haverá mais de um sítio. Além disso, questões históricas, de operação e outras razões técnicas (principalmente distância), podem determinar a existência de vários sítios. O Mapa do sítio (fig. 5.24c) é necessário no projeto da parte local da topologia.

Prédios são construções que abrigam pessoas, má-

quinas e materiais (fig. 5.24d). Esses prédios podem ter andares (fig. 5.24e). Cada andar está dividido em salas. Uma sala recebe a designação de escritório quando os trabalhos ali executadas se restringem ao processamento de informação; oficina quando materiais são processados; almoxarifado no caso de estocagem de produtos; laboratório se a atividade for análise química e física; e muitas outras. Nos casos onde não há prédio e os materiais são estocados (e até processados) a céu aberto, é comum a designação de pátio. Os mapas das salas (e dos pátios) são chamados "layout" (fig. 5.24f) e são necessários no cálculo do tempo de execução dos processos manuais.

O posto de trabalho é o local onde a pessoa desenvolve suas atividades. Nestes locais os móveis e equipamentos devem estar dimensionados e distribuídos ergonomicamente de modo a minimizar o esforço e maximizar a produtividade do executante da tarefa (fig. 5.24g).

Nesta seção foi descrita a segunda etapa - determinação da área geográfica (fig. 4.13). Na próxima, será descrita a terceira - determinação da carga.

5.4 Determinação da carga

O objetivo da terceira etapa, determinação da carga, é quantificar a demanda dos insumos que provocam a execução dos processos manuais e automatizados. O contexto desta etapa na metodologia é apresentado na fig. 5.25. O significado das entradas e saídas foram descritas na seção 4.3.

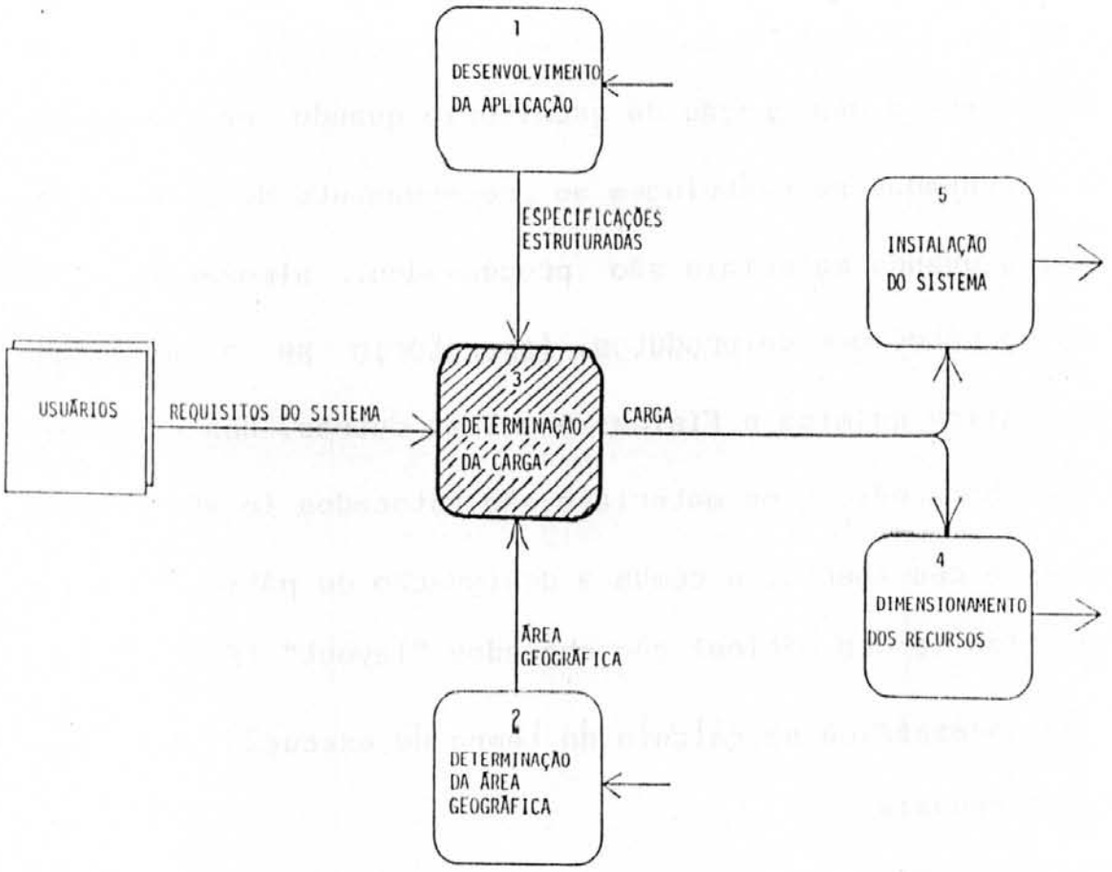


Fig. 5.25 Contexto da etapa de determinação da carga

A terceira etapa é composta de seis atividades: (1) determinação da demanda dos insumos, (2) determinação do tempo de execução dos processos manuais, (3) determinação da demanda dos programas, (4) determinação da demanda dos arquivos, (5) determinação do tamanho das mensagens e (6) composição da carga. As relações entre as atividades acima são apresentadas sob a forma de DFD na fig. 5.26.

5.4.1 Determinação da demanda dos insumos

O objetivo desta atividade é determinar a demanda dos insumos que provocam a execução dos processos manuais. Os insumos são informações ou materiais. Os produtos resultantes são (1) área geográfica por tipo de sítio e (2) diagrama fluxo de dados e materiais quantificado (DFDMQ). Tais

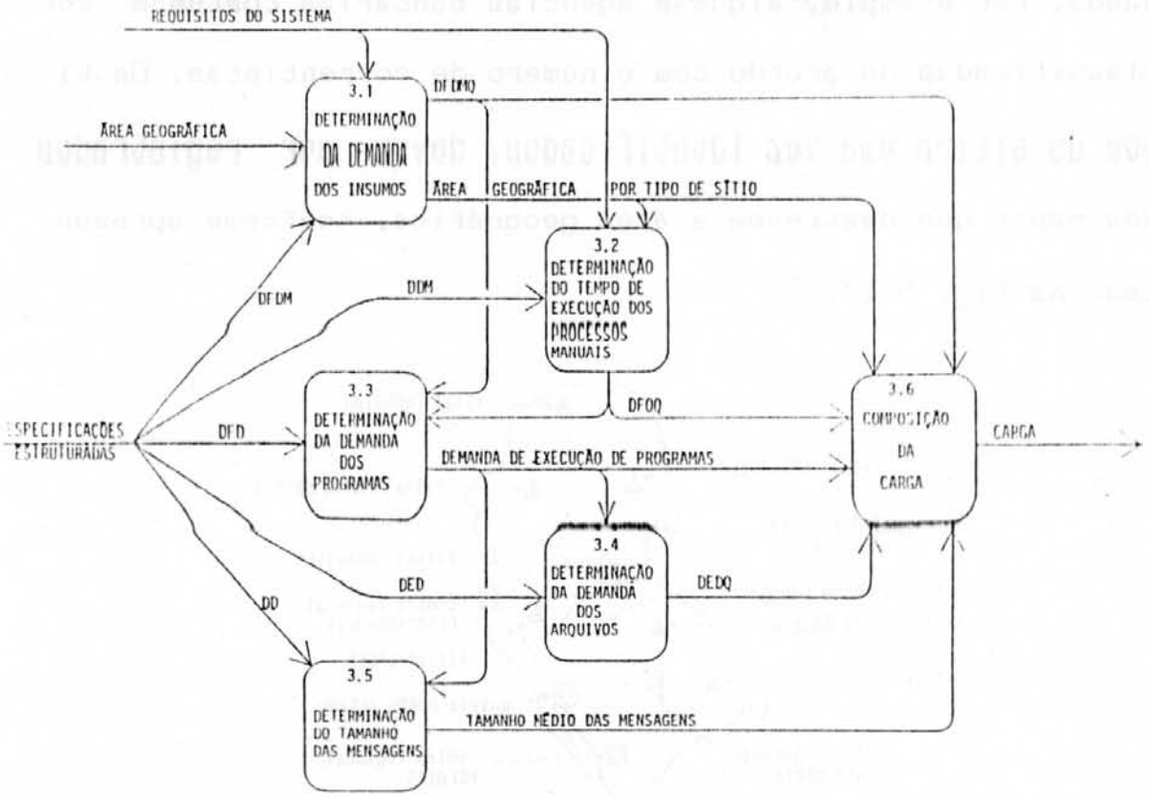


Fig. 5.26 Detalhes da etapa de determinação da carga

produtos são obtidos conforme descrito a seguir.

O primeiro produto é obtido em conjunto com os usuários. A formação de tipos é uma prática comum em empresas cujas funções são dispersas geograficamente. Por exemplo, as agências bancárias costumam ser classificadas por tipo. Existem as agências centrais, urbanas, agrícolas, de poupança e outras. O mesmo acontece com cadeias de lojas, redes de supermercados, farmácias, hotéis e algumas indústrias. Há pelo menos dois critérios para a formação de tipos. O primeiro é administrativo. Determinadas funções são dispersas e outras centralizadas. É o caso, por exemplo, das empresas de comércio. Em cada cidade geralmente existem dois tipos de sítios. Um é o depósito central de mercadorias. O outro são lojas que ficam espalhadas em diversos pontos da cidade. O segundo critério para a formação de tipos é a de-

manda. Por exemplo, algumas agências bancárias costumam ser classificadas de acordo com o número de correntistas. Os tipos de sítios uma vez identificados, devem ser registrados nos mapas que descrevem a área geográfica, conforme apresentado na fig. 5.27.

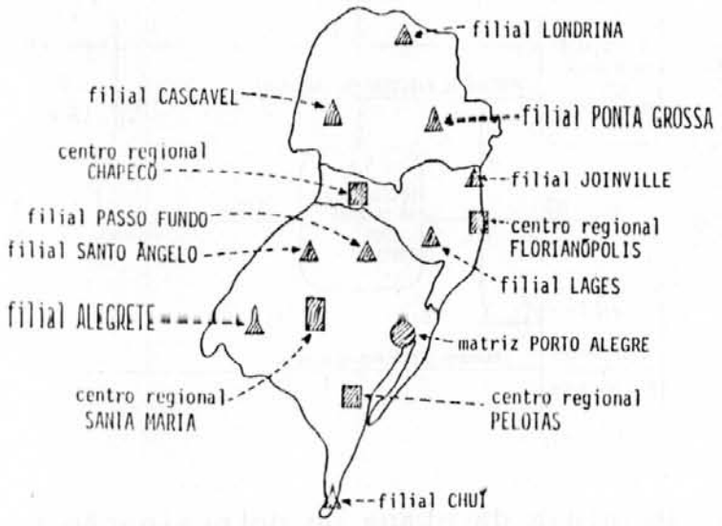


Fig. 5.27 Tipos de sítios identificados na área geográfica

A grande vantagem do conceito de tipo é a redução da complexidade.

O segundo produto também é obtido em conjunto com os usuários. A elaboração do diagrama de fluxo de dados e materiais quantificado (DFDMQ) inicia pela replicação dos DFDMs (figs. 5.12 a 5.14), de forma a haver uma cópia para cada tipo de sítio. Em cada cópia, sobre a ponta das setas que representam o fluxo, são anotadas as taxas. O diagrama assim obtido é o DFDMQ (figs. 5.28 a 5.30). As taxas são obtidas por meio de interações com os usuários. Entretanto, para os insumos críticos tais como pessoas e materiais deterioráveis, deve ser feita uma análise mais acurada, de forma a ser conhecido o perfil da demanda no período (fig. 5.31). A taxa de pico é também anotada no DFDM, juntamente com um

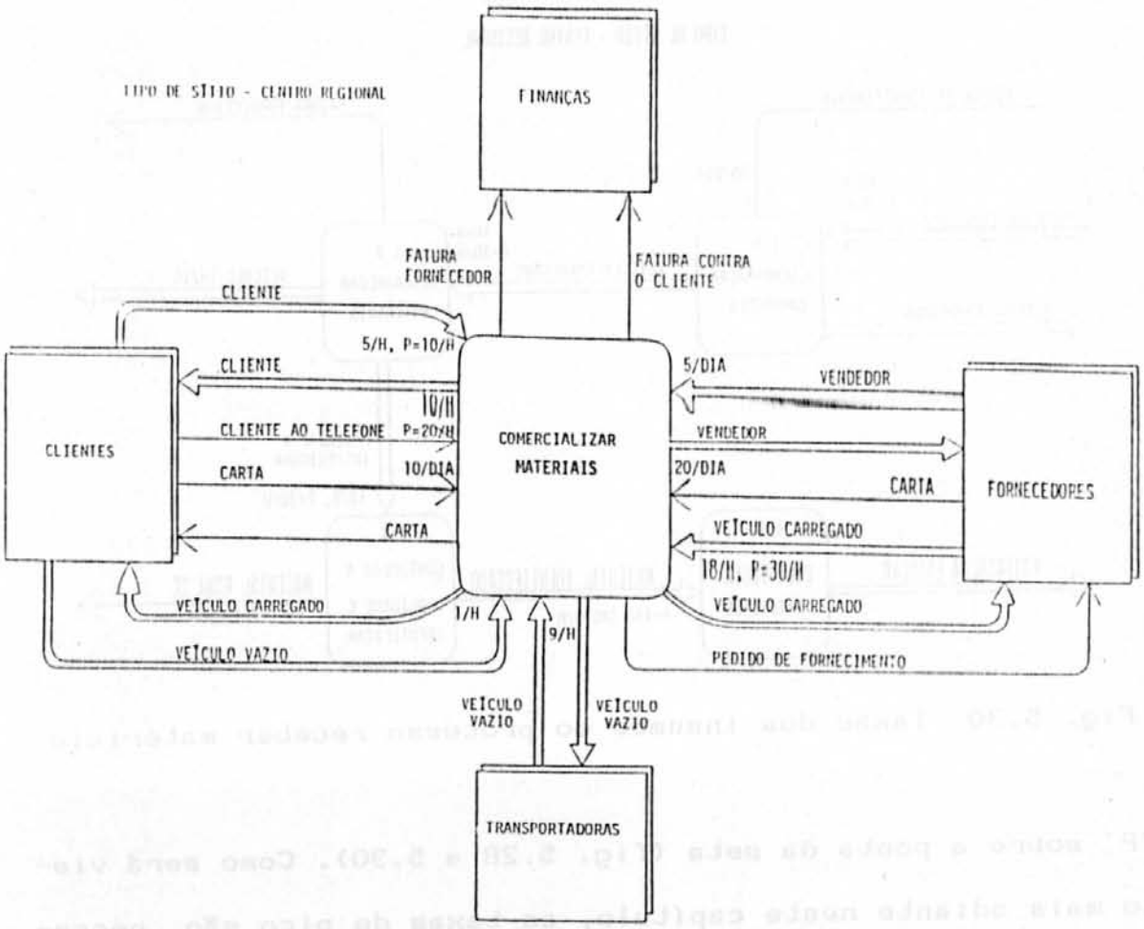


Fig. 5.28 Taxas dos insumos da aplicação comercializar materiais

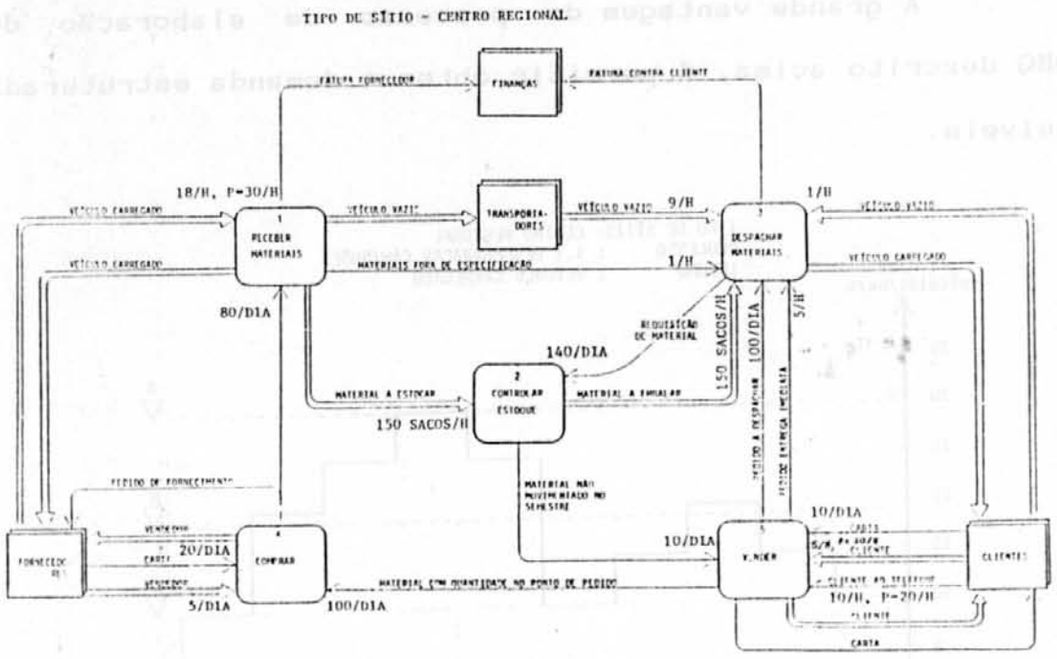


Fig. 5.29 Detalhes das taxas dos insumos da aplicação comercializar materiais

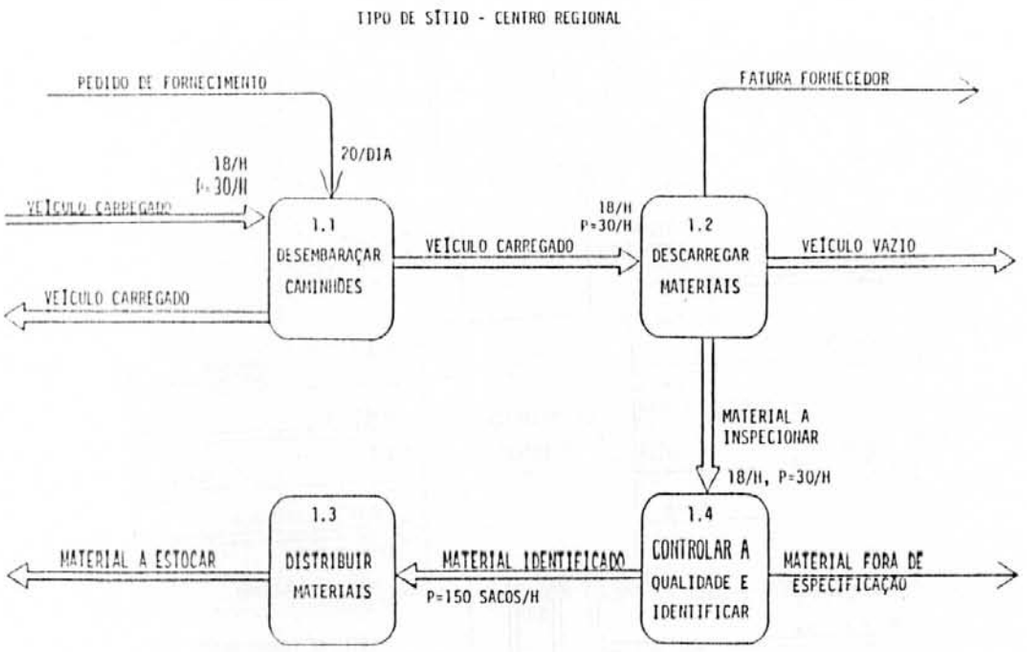


Fig. 5.30 Taxas dos insumos do processo receber materiais

'P' sobre a ponta da seta (fig. 5.28 a 5.30). Como será visto mais adiante neste capítulo, as taxas de pico são necessárias, devido aos limitados tempos de espera associados aos insumos críticos.

A grande vantagem do processo de elaboração do DFDMQ descrito acima, é permitir obter a demanda estruturada em níveis.

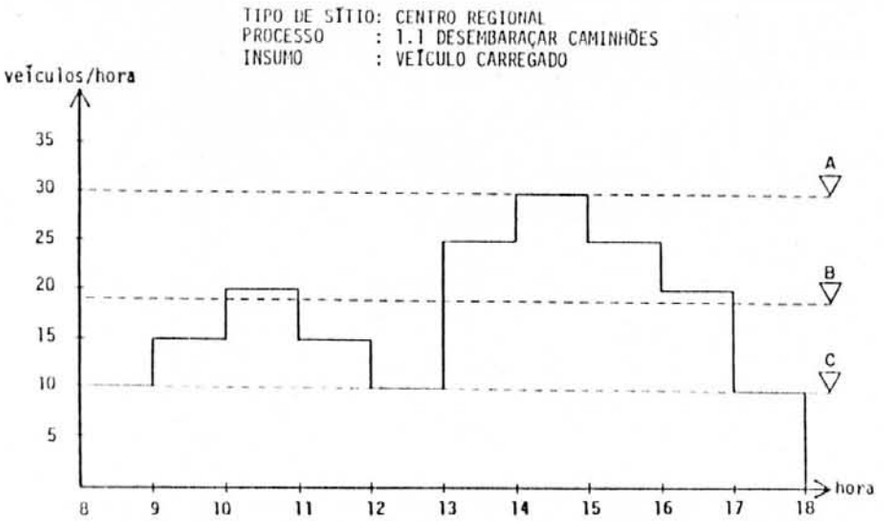


Fig. 5.31 Perfil da demanda dos insumos de um processo

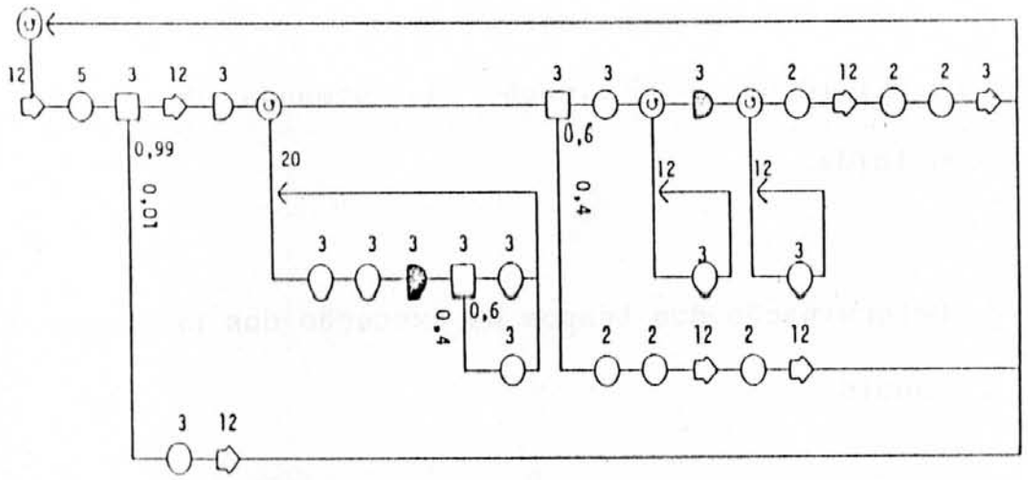
Uma vez emitido o DFDMQ (figs. 5.28 a 5.30), a primeira atividade, determinação da demanda dos insumos, está concluída.

5.4.2 Determinação dos tempos de execução dos processos manuais

O objetivo desta atividade é determinar o tempo dispendido pelas pessoas na execução dos processos manuais. O produto obtido é o diagrama de fluxo de operações manuais quantificado (DFOQ), conforme apresentado na fig. 5.26.

O processo inicia pela construção do diagrama de fluxo de Operações (DFO). Este diagrama é obtido a partir do diagrama de descrição do método (fig. 5.16), de onde as descrições são suprimidas, ficando visíveis apenas os símbolos e as setas. Cada DFO é então reproduzido, de modo a haver uma cópia para cada tipo de sítio. Sobre cada símbolo é anotado o tempo (geralmente em segundos), exceto nas iterações (fig. 5.32). Após isso são associadas probabilidades a cada ramo das decisões (fig. 5.32a). A cada iteração é associado um número de vezes para a execução do laço (fig. 5.32a). Uma vez registrados esses números no diagrama (DFOQ), é montada a expressão algébrica do tempo de execução da tarefa (fig. 5.32b).

O tempo assim obtido ainda não é definitivo. A ele devem ser adicionadas perdas inevitáveis chamadas suplementos. Tais perdas são devidas à fadiga, necessidades pessoais, interrupções administrativas e outras. Normalmente os



(a) Diagrama de Fluxo de Operações Quantificado

$$E(ts) = 12 + 5 + 3 + 0{,}99\left\{12 + 3 + 20\left[3 + 3 + 3 + 3 + 0{,}6(\ 3\ ) + 0{,}4(\ 3\ )\right] + 3 + 0{,}6\left[3 + 12(\ 3\ ) + 3 + 12(\ 3\ ) + 2 + 12 + 2 + 2 + 3\right] + 0{,}4(\ 2 + 2 + 12 + 2 + 12)\right\} + 0{,}01\ (\ 3 + 12) = 405{,}6$$

(b) Expressão do Tempo de Execução do Processo

Fig. 5.32 Dados do tempo de execução dos processos manuais

suplementos variam entre 10 e 25% do tempo de execução da tarefa.

Os tempos das operações ◯ podem ser obtidos através de interações com os usuários. Entretanto há diversas técnicas que permitem o cálculo destes tempos [MAY70b], [ZAN80], [BAR77], [GRO73]. Tempos de espera D e de transporte ⇨ são variáveis dependentes do "layout". O tempo de espera pelo computador ▨ (tempo de resposta) é geralmente aceito como até três segundos [MAR72]. Entretanto é recomendável adotar um valor especificado pelo usuário.

Em cada tipo de sítio o mesmo processo manual pode ter tempos de execução diferentes. Isso não é desejável. O ideal seria que os tempos fossem padronizados. Por exemplo, as empresas montadoras de veículos mantém padronizados os tempos de execução dos processos de manutenção dos carros de suas marcas. A cada processo é associado um tempo, não im-

portando a oficina em que o trabalho é realizado. Isso exige que cada concessionário, além de usar as mesmas descrições dos métodos e executar as operações com as mesmas ferramentas, tenha um "layout" padronizado para os postos de trabalho. Para obter o mesmo resultado num sistema de informação, o procedimento deve ser semelhante.

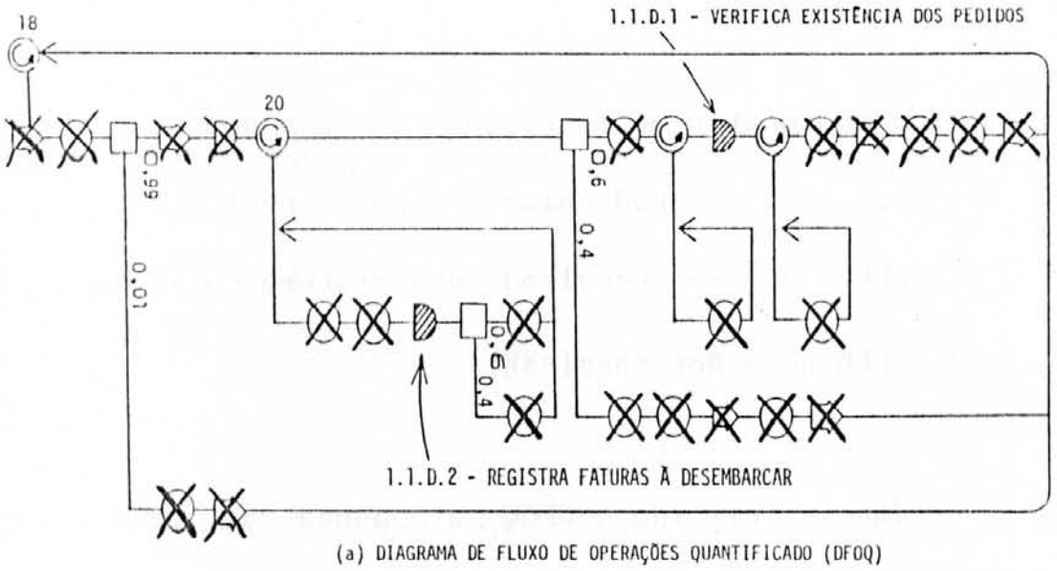
Uma vez emitido o DFOQ, a segunda atividade, determinação do tempo de execução dos processos manuais, está concluída.

5.4.3 Determinação da demanda dos programas

O objetivo desta atividade é determinar a taxa de execução dos programas por tipo de sítio. O produto obtido é a lista de execução dos programas, conforme apresentado na fig. 5.26.

O processo inicia com a duplicação dos diagramas de fluxo de operações quantificados (DFOQs). Todas as operações, esperas e transporte são eliminados. Permanecem apenas as iterações, decisões e espera pelo computador (fig.5.33a). (Para uma melhor compreensão, devem ser comparadas as figs. 5.16 e 5.33a). Como foi visto na seção 5.2.3.1.1, o símbolo ▨ representa um processo automatizado. Tal processo é um programa de computador.

Depois de construído o diagrama (fig. 5.33a), restam ainda três passos. Primeiro, sobre a iteração inicial do DFOQ deve ser anotada a taxa do insumo que ativa o processo



(a) DIAGRAMA DE FLUXO DE OPERAÇÕES QUANTIFICADO (DFOQ)

18 x 0.99(20) = 356,4

(b) DEMANDA DE EXECUÇÃO DO PROGRAMA 1.1.D.2

18 x 0,99(0,66) = 10,7

(c) DEMANDA DE EXECUÇÃO DO PROGRAMA 1.1.D.1

Fig. 5.33 Cálculo das taxas de execução dos processos automatizados

manual em questão. No exemplo da fig. 5.33a a taxa é igual a 18, sendo oriunda do diagrama de fluxo de dados e materiais quantificado (DFDMQ), apresentado na fig. 5.30. O segundo passo é a montagem das expressões matemáticas e os respectivos cálculos das taxas de execução dos programas conforme apresentado na figs. 5.33b e 5.33c. O terceiro passo é a tabulação destas taxas e a montagem da lista de execução dos programas conforme apresentada na fig. 5.34. Esta tabulação é necessária porque há a possibilidade do mesmo programa ser usado em mais de um processo manual.

Uma vez emitida a lista (fig. 5.34), a terceira atividade, determinação da demanda dos programas, está concluída.

DEMANDA DE EXECUÇÃO DE PROGRAMAS

TIPO DE SÍTIO - CENTRO REGIONAL

NÚMERO DO PROGRAMA	NOME DO PROGRAMA	TAXA HORÁRIA DE EXECUÇÃO DOS PROGRAMAS
1	2	3
1.1.D.2	REGISTRA FATURAS À DESEMBARCAR	356,4
1.1.D.1	VERIFICA EXISTÊNCIA DOS PEDIDOS	10,7
1.2.2.D.1	EXIBE FATURAS DA CARGA DO VEÍCULO	18,0
1.2.2.D.2	EXIBE PEDIDOS DE UMA FATURA	18,0
	TOTAL	403,1

Fig. 5.34 Demanda de execução de programas

5.4.4 Determinação da demanda dos arquivos

O objetivo desta atividade é determinar as taxas de acesso aos arquivos. O produto obtido é o diagrama de estrutura de dados quantificado (DEDQ), conforme apresentado na fig. 5.26.

O processo é realizado em duas etapas. Na primeira, para cada arquivo é organizada uma planilha conforme apresentada na fig. 5.35. As duas primeiras colunas são preenchidas com dados da lista de execução de programas (fig. 5.34). A taxa unitária de acessos (tanto de leitura como de atualização), tem origem nos diagramas de fluxo de dados (fig. 5.17). O total de acessos é o produto da taxa unitária pela taxa de execução dos programas. Na segunda etapa, é

DEMANDA DE ACESSO AOS ARQUIVOS

TIPO DE SÍTIO - CENTRO REGIONAL
ARQUIVO - PEDIDOS DE FORNECIMENTO

PROGRAMA	TAXA HORÁRIA DE EXECUÇÃO DOS PROGRAMAS	LEITURA		ATUALIZAÇÃO	
		TAXA UNITÁRIA	TOTAL DE ACESSOS	TAXA UNITÁRIA	TOTAL DE ACESSOS
1	2	3	4	5	6
1.1.D.2	356,4	-	-	-	-
1.1.D.1	10,7	5	53,5	1	10,7
1.2.2.D.1	18,0	-	-	-	-
1.2.2.D.2	18,0	5	90,0	-	-
TOTAL			143,5		10,7

Fig. 5.35 Cálculo da taxa de acesso aos arquivos

feita uma replicação dos diagramas de estrutura de dados (fig. 5.7), de forma a existir uma cópia para cada sítio. Sobre estes diagramas são anotados os totais de acesso aos arquivos, o que origina o DEDQ (fig. 5.36). A fonte destes dados são as planilhas (fig. 5.35).

Uma vez emitido o diagrama de estrutura de dados quantificado (fig. 5.36), a quarta atividade, determinação da demanda dos arquivos, está concluída.

5.4.5 Determinação do tamanho das mensagens

O objetivo desta atividade é determinar o tamanho médio das mensagens de solicitação e de resposta. O produto obtido é constituído destes dois valores, conforme apresen-

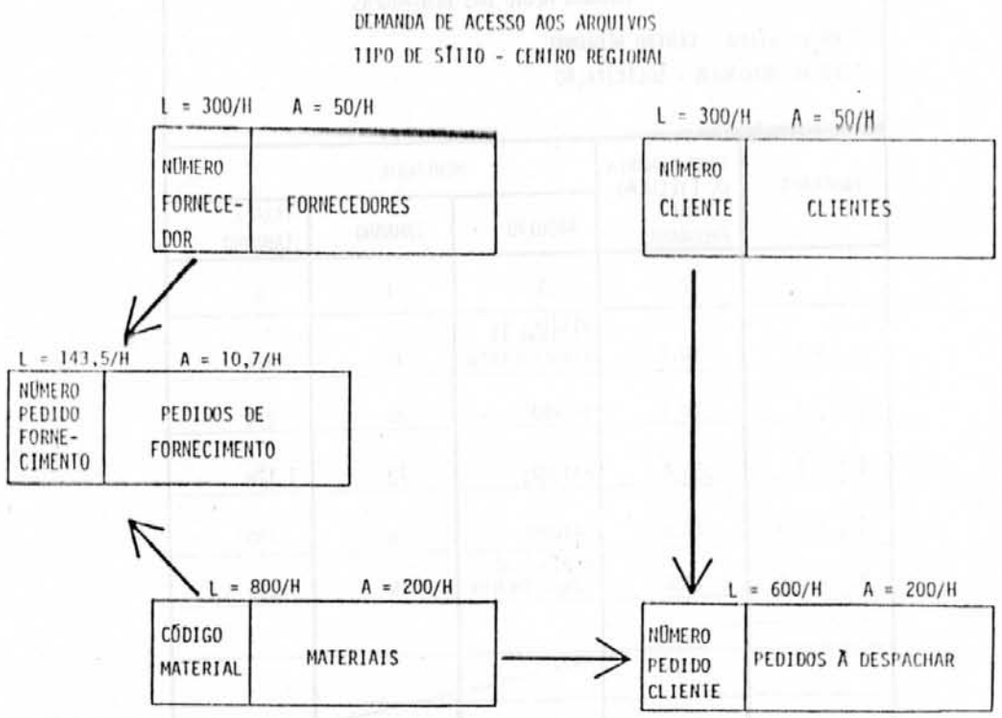


Fig. 5.36 Demanda de acesso aos arquivos

tado na fig. 5.26.

Tanto para a mensagem de solicitação, como para a mensagem de resposta, são construídas planilhas conforme a apresentada na fig. 5.37. As duas primeira colunas são obtidas da lista de execução dos programas (fig. 5.34). O nome dos arquivos tem origem nos diagramas de fluxo de dados (fig. 5.17). O tamanho unitário da mensagem é obtido do dicionário de dados, que pelas regras da análise estruturada sempre deverá estar associado ao DFD. Deve ser anotada a quantidade de bytes referentes aos campos que efetivamente são usados no programa e não o tamanho total do registro. A última coluna é o produto entre a taxa de execução dos programas e o tamanho dos arquivos (fig. 5.37). Uma vez efetuados os produtos, os valores são tabulados e é calculada a média ponderada, conforme a última linha da planilha da fig. 5.37. Este valor é o tamanho médio da mensagem.

TAMANHO MÉDIO DAS MENSANGENS

TIPO DE SÍTIO - CENTRO REGIONAL

TIPO DE MENSAGEM - SOLICITAÇÃO

PROGRAMA	TAXA HORÁRIA DE EXECUÇÃO DOS PROGRAMAS	MENSAGEM		
		ARQUIVO	TAMANHO	TAXA x TAMANHO
1	2	3	4	5
1.1.D.1	10,7	PEDIDOS DE FORNECIMENTO	30	321
1.1.D.1	10,7	FATURAS	20	214
1.1.D.2	356,4	FATURAS	20	7.128
1.2.2.D.1	18,0	FATURAS	10	180
1.2.2.D.2	18,0	PEDIDOS DE FORNECIMENTO	15	270
TOTAL	413,8			8.113
TAMANHO MÉDIO			20	

Fig. 5.37 Cálculo do tamanho das mensagens

Uma vez calculado o tamanho médio da mensagem de solicitação e da mensagem de resposta, a quinta atividade, determinação do tamanho das mensagens, está concluída.

A última atividade, composição da carga (fig. 5.26), consiste em reunir numa pasta os produtos das demais atividades da terceira etapa. Uma vez organizada esta pasta, a etapa de determinação da carga está concluída.

Até aqui foi descrita a terceira etapa - determinação da carga (fig. 4.13). Na próxima seção será descrita a quarta - dimensionamento dos recursos.

5.5 Dimensionamento dos recursos

O objetivo da quarta etapa, dimensionamento dos recursos, é quantificar equipamentos de computação, comunicação, móveis, espaço físico e pessoal necessários à operação do sistema distribuído. O contexto desta etapa na metodologia é apresentado na fig. 5.38. O significado das entradas e saídas foram discutidos na seção 4.3.

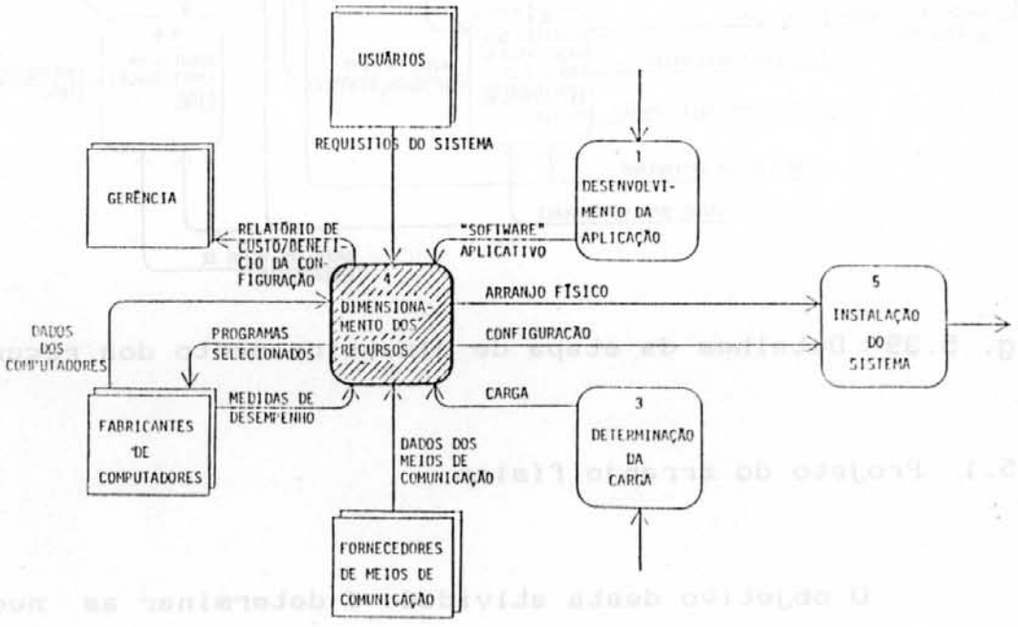


Fig. 4.38 Contexto da etapa de dimensionamento dos recursos

A quarta etapa é composta de seis atividades: (1) projeto do arranjo físico, (2) obtenção dos dados para o projeto da topologia, (3) projeto da topologia, (4) obtenção dos dados para o projeto da configuração básica, (5) projeto da configuração básica e (6) projeto da configuração final. As relações entre as atividades acima são apresentadas sob a forma de DFD na fig. 5.39.

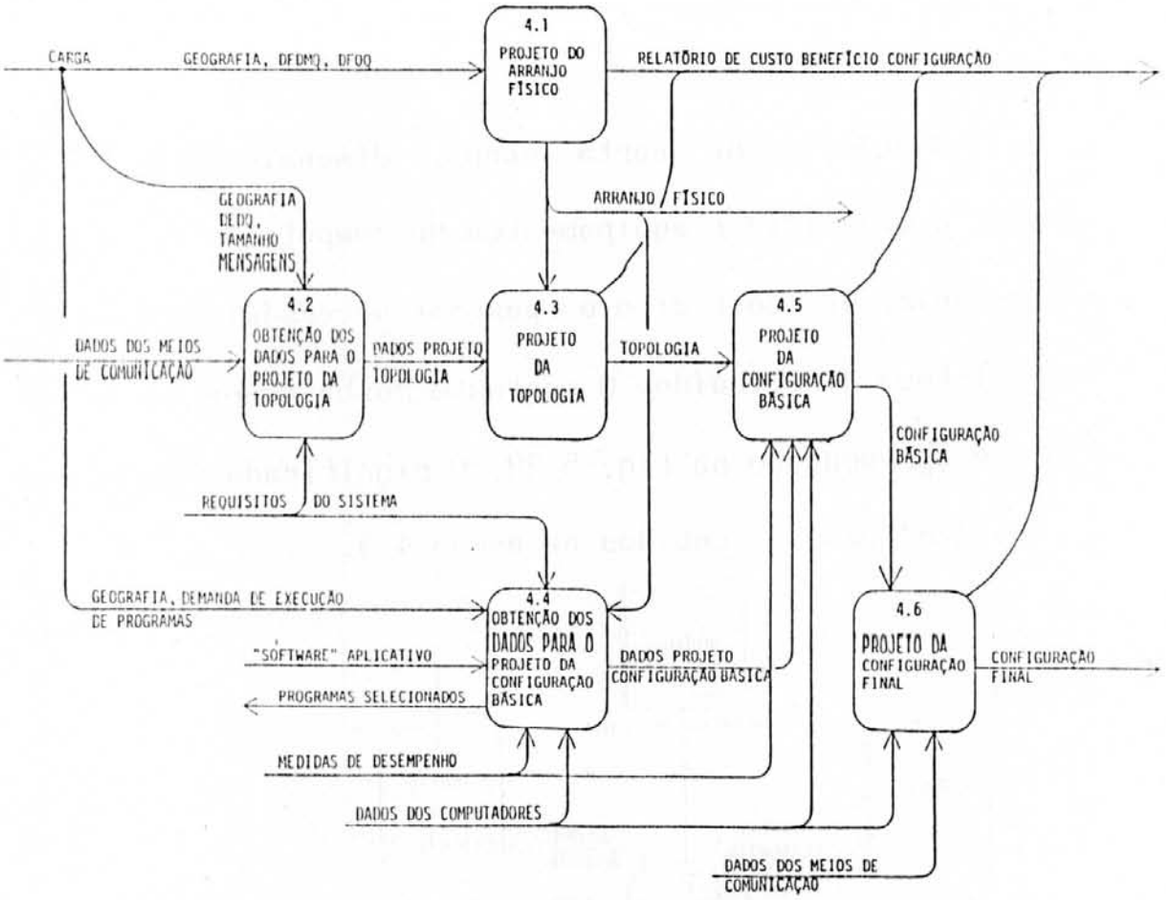


Fig. 5.39 Detalhes da etapa de dimensionamento dos recursos

5.5.1 Projeto do arranjo físico

O objetivo desta atividade é determinar as necessidades de pessoal, espaço físico, móveis e terminais necessários à execução dos processos manuais do sistema distribuído. A determinação do principal produto, o arranjo físico, é descrito mais adiante nesta seção. No dimensionamento dos recursos é usado o conceito de teoria das filas. Por esta razão, na próxima seção é dada uma breve introdução deste conceito.

5.5.1.1 Teoria das Filas

Na fig. 5.31 está ilustrado o perfil da demanda de veículos_carregados, que é o insumo da tarefa desembaracar

caminhões. Se forem alocados recursos (pessoas) para atender ao nível A da demanda, todo veículo que chegar será atendido imediatamente. Entretanto, os recursos estarão ociosos a maior parte do tempo. Se os recursos forem alocados para atender ao nível B, a ociosidade diminui, mas os picos não serão atendidos, havendo postergação e a consequente formação de fila. Se forem alocados recursos para atender ao nível C, não haverá ociosidade, mas a fila crescerá indefinidamente. Deduz-se portanto, que há uma solução de compromisso entre a quantidade de recursos e a formação de filas.

Na Teoria das Filas, o recurso é chamado servidor e é representado pela letra (M). Podem haver um ou mais servidores. O outro elemento é o usuário do recurso. Existem filas de uma única classe de usuários ou de múltiplas classes (fig. 5.40). O tempo que o servidor utiliza para atender um usuário é designado por E(ts). O tempo que o usuário permanece esperando pelo atendimento é E(tw). A taxa de chegada (demanda) de usuários é E(N). O modelo clássico de filas para representar estes valores é apresentado na fig. 5.41. O fator de utilização dos servidores é designado por ρ . A probabilidade de que todos os servidores estejam ocupados no momento em que um usuário chega, é designada por B. Há um $E(N_i)$, $E(ts_i)$, $E(tw_i)$ e ρ_i para cada classe i de usuários.

No presente trabalho, o uso da Teoria das Filas se restringe ao cálculo do número de servidores (M). Para tanto devem ser conhecidos os valores da demanda E(N), tempo de atendimento E(ts) e tempo de espera E(tw). O fator de utilização ρ , a probabilidade B e o valor de M são obtidos

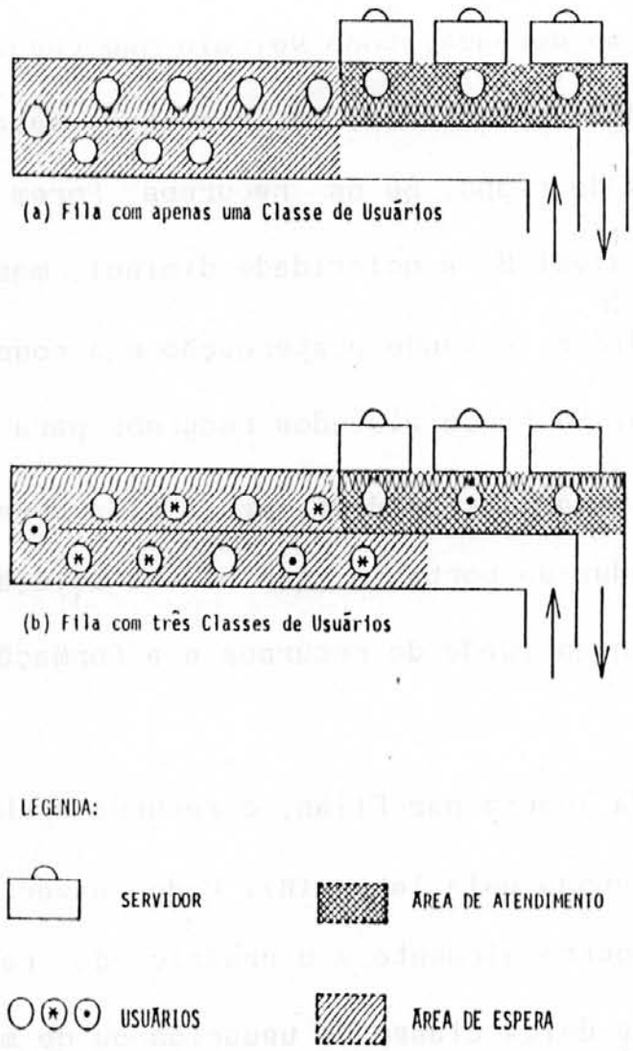


Fig. 5.40 Fila única com múltiplos servidores

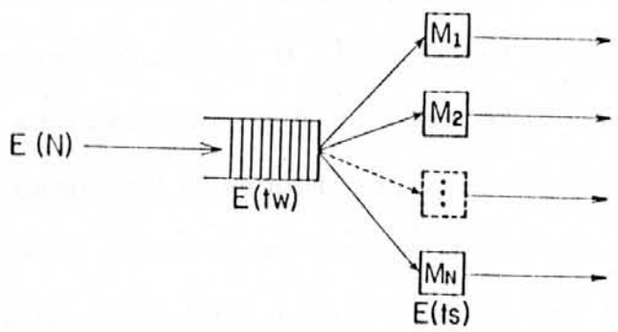
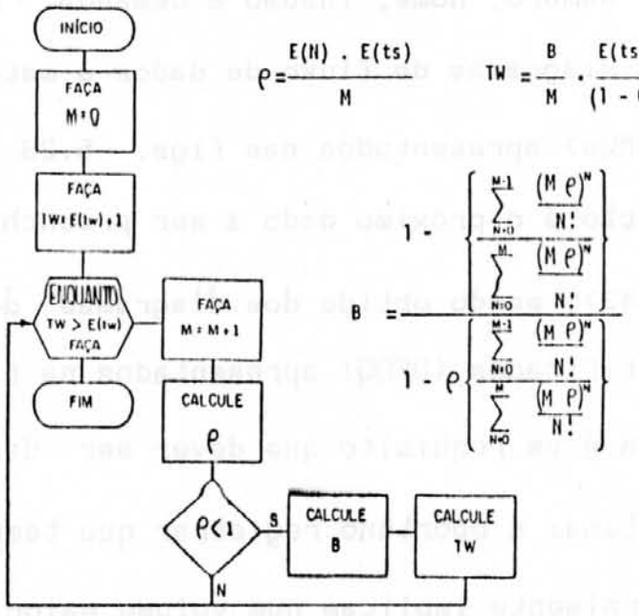


Fig. 5.41 Representação clássica de uma fila

através das fórmulas e do algorítmo descritos na fig. 5.42a.

Para o caso de haver "k" classes de usuários, o mesmo algorítmo (fig. 5.42a) pode ser usado. Entretanto os valores de E(N), E(ts) e E(tw) devem ser sintetizados atra-



$$\rho = \frac{E(N) \cdot E(ts)}{M} \qquad TW = \frac{B}{M} \cdot \frac{E(ts)}{(1-\rho)}$$

$$B = \frac{1 - \left\{ \sum_{N=0}^{M-1} \frac{(M\rho)^N}{N!} \right\}}{1 - \rho \left\{ \sum_{N=0}^{M} \frac{(M\rho)^N}{N!} \right\}}$$

(a) Algoritmo para a determinação do número de servidores (M) que satisfaça os requisitos de demanda E(N), tempo de execução E(ts) e tempo de espera E(tw).

$$E(N) = E(N_1) + E(N_2) + \ldots + E(N_k)$$

$$E(ts) = \frac{E(N_1)}{E(N)} E(ts_1) + \frac{E(N_2)}{E(N)} E(ts_2) + \ldots + \frac{E(N_k)}{E(N)} E(ts_k)$$

$$E(tw) = \frac{E(N_1)}{E(N)} E(tw_1) + \frac{E(N_2)}{E(N)} E(tw_2) + \ldots + \frac{E(N_k)}{E(N)} E(tw_k)$$

(b) Fórmulas para sintetização dos dados, no caso de haver mais de uma classe de usuários.

Fig. 5.42 Algorítmo e fórmulas da teoria das filas

vés das fórmulas descritas na fig. 5.42b.

5.5.1.2 Dimensionamento dos postos de trabalho

O objetivo desta seção é descrever como determinar a quantidade de pessoas necessárias à execução dos processos manuais. Cada pessoa ocupa um posto de trabalho. Os postos estão alocados numa sala (figs. 5.24f e 5.24g). O processo de dimensionamento é descrito a seguir.

Para efetuar os cálculos são usadas planilhas conforme a fig. 5.43. Primeiramente são preenchidos os dados

dos processos: número, nome, insumo e demanda. Esses dados têm origem nos diagramas de fluxo de dados e materiais quantificados (DFDMQs) apresentados nas figs. 5.28 a 5.30. O tempo de execução é o próximo dado a ser preenchido na planilha (fig. 5.43), sendo obtido dos diagramas de fluxo de operações quantificados (DFOQ) apresentados na fig. 5.32b. O tempo de espera é um requisito que dever ser definido pelo usuário do sistema. É oportuno registrar que tempos de espera menores, geralmente implicam num volume maior de recursos. Outro dado da planilha (fig. 5.43), é o tipo de posto de trabalho, que deve ser definido juntamente com o usuário. A decisão de que haverá apenas um tipo de posto, significa que qualquer processo poderá ser executada em qualquer posto da sala. Entretanto pode haver uma variedade muito grande de trabalhos o que inviabiliza a existência de um único tipo de posto. Neste caso mais de um tipo de posto pode ser criado, cada um especializado num elenco de processos manuais. Há também questões como a demanda e o "layout" que podem in-

DIMENSIONAMENTO DOS POSTOS DE TRABALHO

TIPO DE SÍTIO: Centro Regional — SALA: Escritório do Almoxarifado

PROCESSO						POSTOS DE TRABALHO			
Nº	NOME	INSUMO	TEMPO DE EXECUÇ.	TEMPO DE ESPERA	DEMANDA	TIPO	QTDE.	FATOR DE UTILIZ.	NOME
			E(ts)	E(tw)	E(N)		M	ρ	
1.1	Desembaraçar Caminhões	Veículo Carregado	6,8 min	30 min	30/h				
TOTAL TIPO 1			6,8 min	30 min	30/h	1	4	0,85	Recepcionista
123	Efetuar a Descarga	Veículo Carregado	30 min	10 min	30/h				
TOTAL TIPO 2			30 min	10 min	30/h	2	18	0,83	Descarregador
121	Planejar a Descarga	Veículo Carregado	5 min	5 min	30/h				
124	Conferir a Carga	Carga	6 min	15 min	30/h				
TOTAL TIPO 3			5,5 min	10 min	60/h	3	6	0,92	Controlador

Fig. 5.43 Cálculo da quantidade de postos de trabalho

fluir na decisão. Após esta definição, os valores de E(N), E(ts) e E(tw) devem ser totalizados para cada tipo de posto, através das fórmulas descritas na fig. 5.42b. Desta maneira a planilha (fig. 5.43) conterá todos os dados necessários ao cálculo do número de postos de trabalho. Esse cálculo é feito através das equações e do algorítmo descritos na fig. 5.42a.

5.5.1.3 Dimensionamento dos terminais

O objetivo desta seção é descrever como determinar a quantidade de terminais a serem usados pelos executantes dos processos manuais.

Primeiramente é preciso organizar os dados. Para tanto são construídas planilhas conforme a fig. 5.44. Para cada tipo de posto de trabalho, são preenchidos número, nome e insumo dos processos nele executadas. A fonte destes dados

DIMENSIONAMENTO DOS TERMINAIS

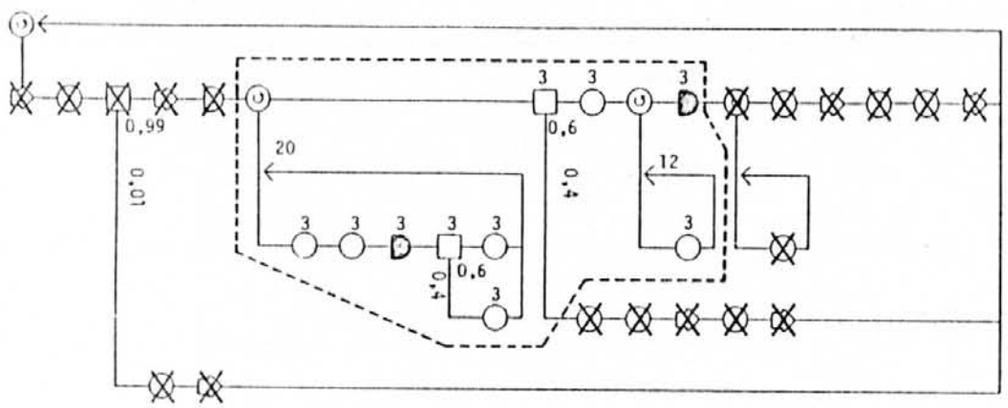
TIPO DE SÍTIO: Centro Regional — SALA: Escritório do Almoxarifado

POSTO DE TRABALHO								TERMINAL			
TIPO	NOME	Nº	NOME	INSUMO	TEMPO DE EXEC.	TEMPO DE ESPERA	DEMANDA	CATEGORIA	QTDE.	FATOR DE UTILIZ.	NOME
					E(ts)	E(tw)	E(N)		M	P	
1	Recepcionista	1.1	Desembaraçar Caminhões	Veiculo Carregado	5,4 min	30 min	30/h				
TOTAL CATEGORIA 1					5,4 min	30 min	30/h	1	3	0,90	Recepção
2	Controlador	121	Planejar a Descarga	Veiculo Carregado	2 min	5 min	30/h				
		124	Conferir a Carga	Carga	1 min	15 min	30/h				
4	Conferencista	15	Conferir Faturas	Lote de Faturas	3 min	8 min	18/h				
TOTAL CATEGORIA 2					1,8 min	9,5 min	78/h	2	3	0,78	Planejamento

Fig. 5.44 Cálculo da quantidade de terminais

é a planilha da fig. 5.43. Entretanto, só devem ser preenchidos os dados dos processos onde o computador é usado, ou seja, se houver o símbolo ▨ na descrição do método (fig. 5.16). Desta maneira a planilha (fig. 5.44) fica parcialmente preenchida. O próximo passo é a obtenção dos dados numéricos . Para cada tarefa devem ser providenciados E(ts), E(tw) e E(N). Os valores de E(N) e E(tw) são transcritos da planilha (fig. 5.43). Já os valores de E(ts) tem um processo de cálculo mais elaborado, conforme descrito a seguir.

Em primeiro lugar, deve ser providenciada uma cópia do cada diagrama do fluxo de operações quantificado (DFOQ), apresentado na fig. 5.32. As operações (e outros símbolos) cuja execução independem do equipamento de interface, devem ser suprimidas destas cópias (fig. 5.45a). (Para uma melhor compreensão devem ser comparadas as figs. 5.16 e 5.45a). Usando os dados dos símbolos remanescentes, é então montada a expressão para o cálculo de E(ts), conforme ilustrado na fig. 5.45b. Esse valor é transcrito para a planilha (fig. 5.44).



(a) DFOQ destacando a parcela da tarefa que é executada com o auxílio do terminal

$E(ts) = 0,99\left\{20\left[3 + 3 + 3 + 3 + 0,6(3) + 0,4(3)\right] + 3 + 0,6\left[3 + 12(3) + 3\right]\right\} = 324,9$ seg.

(b) Expressão do Tempo de Uso do terminal

Fig. 5.45 Cálculo do tempo de uso do terminal

Neste momento estão disponíveis na planilha (fig. 5.44) todos os dados dos processos manuais. Outro dado da planilha (fig. 5.44) é a categoria dos terminais. Através de interações com o usuário, será decidido se os ocupantes dos postos de trabalho usarão indistintamente qualquer terminal ou se determinados postos terão equipamento exclusivo. Essa decisão pode definir a existência de uma ou mais categorias. Após esta definição, os valores de E(N), E(ts) e E(tw) devem ser totalizados para cada categoria, através das equações descritas na fig. 5.42b. Desta forma a planilha (fig. 5.44) fica com todos os dados necessários ao cálculo da quantidade de terminais. Esses cálculos são feitos através das fórmulas e do algorítmo descritos na fig. 5.42a.

5.5.1.4 Arranjo físico

Uma vez obtidos os dados dos postos de trabalho e número de terminais, é possível projetar o arranjo físico. Um exemplo é apresentado na fig. 5.46. As técnicas para projeto de "layout" são apresentadas em [ARA85].

O "layout" causa grande impacto no usuário. Este é o momento adequado para uma revisão e avaliação de todos os resultados até então obtidos. É preciso estar ciente de que as expressões matemáticas da teoria das filas, estão baseadas em médias, desvios padrões e tipos de distribuição (normal, binomial, Poisson). Para efeito de projeto, sempre há a suposição de que os números usados têm as propriedades exigidas pelas fórmulas. A eventual inexistência dessas propriedades leva à imprecisão. Entretanto isso não invalida o

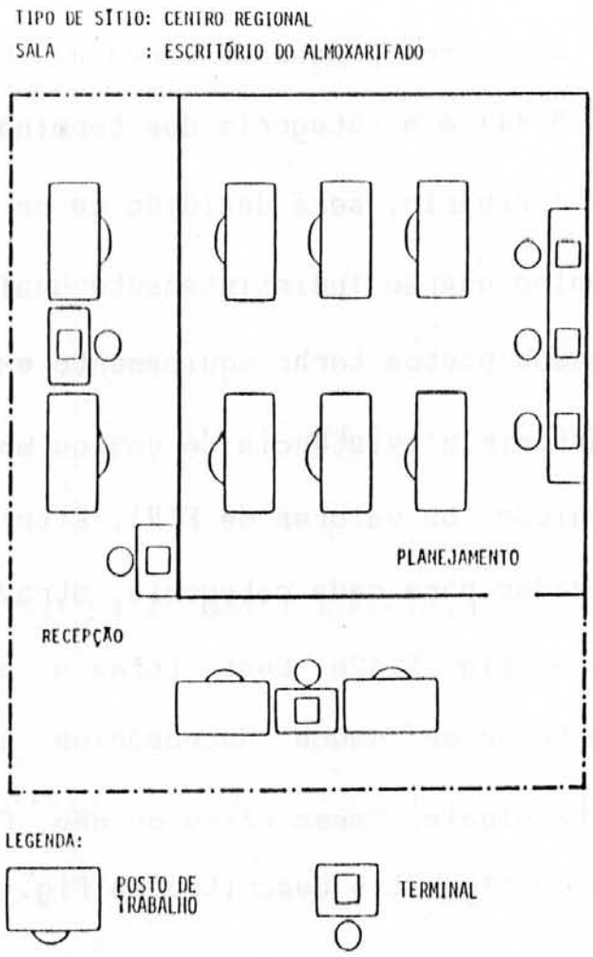


Fig. 4.46 Arranjo físico de uma sala

uso da técnica, ao contrário, possibilita estabelecer os limites do comportamento do sistema. Detalhes sobre as propriedades estatísticas das fórmulas, podem ser encontrados em [MAR72].

Durante o projeto do arranjo físico são tomadas três decisões importantes. A primeira é quando da determinação dos postos de trabalho (seção 5.5.1.2). Ali é definido se todos os processos serão executados por todos os ocupantes dos postos ou se haverá especialização. A segunda decisão ocorre no momento em que o número de terminais é calculado (seção 5.5.1.3). Tal decisão se refere ao estabelecimento de categorias. Se houver uma única categoria, todos os ocupantes dos postos de trabalho poderão usar indistintamen-

te qualquer terminal. Se houver mais de uma categoria, determinados terminais só poderão ser usados pelos ocupantes de determinados postos de trabalho. A terceira decisão é o arranjo dos móveis no espaço disponível. Poderá ser usada uma única sala, multiplas salas, biombos e outros. A combinação destas três decisões geram várias alternativas. O resultado de cada alternativa é um "layout" diferente. Por esta razão, um dos produtos da atividade de projeto do arranjo físico é um relatório de custo benefício (fig. 5.39). Tal relatório facilita o julgamento do usuário no momento da escolha.

Uma vez escolhido o melhor "layout", a atividade de projeto do arranjo físico está concluída.

5.5.2 Obtenção dos dados para o projeto da topologia

Esta é a segunda atividade da etapa 4 da metodologia de desenvolvimento de sistemas distribuídos (fig. 5.39). O objetivo desta atividade é organizar os dados necessários ao projeto da topologia.

Conforme foi visto na seção 3.3, a topologia tem duas partes. A primeira parte é formada pelos elos locais, que ligam terminais a computadores ou a multiplexadores (fig. 3.2). A segunda parte é formada pelos elos de longa distância, que ligam computador-computador ou computador-multiplexador (fig. 3.4).

Para projetar a parte local da topologia são ne-

cessários dois dados. O primeiro é o conjunto de distâncias, que pode ser obtido nos mapas que descrevem a região geográfica (fig. 5.24). O segundo é a quantidade de terminais. Tal dado pode ser obtido diretamente do arranjo físico (fig. 5.46).

A porção de longa distância da topologia pode ser projetada de quatro formas: modelos de programação linear, métodos heurísticos, soluções pragmáticas e combinações dos métodos anteriores. Tais formas foram descritas no capítulo 3. O projeto da topologia através de modelos de programação linear, é alternativa que exige a maior variedade de dados. Conforme visto na seção 3.3.3.1 estes dados são: (1) matriz de capacidade das linhas de transmissão de dados, (2) matriz com o custo destas linhas, (3) matriz de acesso para atualização dos arquivos, (4) matriz de acesso para leitura dos arquivos, (5) tamanho médio das mensagens e (6) tempo de resposta. Tais dados são obtidos e organizados como segue.

O primeiro dado é a matriz de capacidades (C_c). Tais dados são obtidos diretamente do fornecedor das linhas de comunicação (EMBRATEL). O segundo dado é a matriz de custos ($C_{\ell c}$). O número de ligações (ℓ) bem como as respectivas distâncias são obtidas do mapa que delimita a região geográfica (fig. 5.27). O número de capacidade disponíveis (c) é obtido diretamente do fornecedor. O custo das ligações pode ser calculado com base nos Regulamentos e Tarifas para Comunicação de Dados no Brasil, conforme apresentado no apêndice de [TAR84]. A fórmula básica é a seguinte:

$$C = N * m * tb * K$$

onde N = fator devido à capacidade da linha

m = fator devido à distância

tb = tarifa básica (Cz$)

K = 1 (ligação permanente) ou

0,05 por dia de utilização.

Os quatro dados restantes são obtidos como segue. A matriz de acesso para atualização dos arquivos (A_{ij}) e a matriz de acesso para leitura dos arquivos (P_{ij}), são construídas com base no diagrama de estrutura de dados quantificado (DEDQ) apresentado na fig. 5.36. O tamanho médio das mensagens é obtido na terceira etapa (fig. 5.26), conforme exemplo apresentado na fig. 5.37. O tempo de resposta normalmente é obtido junto ao usuário.

Uma vez organizados os dados, a segunda atividade é encerrada, podendo ser iniciada a terceira - projeto da topologia (fig. 5.39). O modo de proceder para levar a cabo esta atividade foi amplamente discutido na seção 3.3.

5.5.3 Obtenção dos dados para o projeto da configuração básica.

Esta é a quarta atividade da etapa 4 da metodologia de desenvolvimento de sistemas distribuídos (fig. 5.39). O objetivo desta atividade é organizar os dados necessários ao projeto da configuração básica.

Uma vez projeta a topologia, é preciso selecionar os computadores que serão instalados em cada nó. Como foi visto na seção 3.4, esta seleção é feita com base num gráfico conforme exemplo apresentado na fig. 3.14. Neste gráfico existem duas entidades. A primeira é a região viável, que estabelece os limites da aplicação. Esta região é delimitada pela abscissa número_de_terminais e pela ordenada tempo_de resposta (fig. 3.14). O número de terminais é obtido diretamente do arranjo físico (fig. 5.46). O tempo de resposta é um requisito definido pelo usuário. A segunda entidade é o conjunto de curvas características dos computadores concorrentes à ocupação do nó (fig. 3.14). Tais curvas são obtidas como segue.

Há três formas de traçar as curvas: simulação, modelos analíticos e "benchmark". Tais formas foram apresentadas na seção 3.4. O traçado das curvas através de modelos analíticos é a alternativa que exige a maior variedade de dados. Conforme visto na seção 3.4.2 estes dados podem ser divididos em dois grupos. No primeiro grupo estão os dados a serem fornecidos pelos fabricantes:

(1) velocidade da CPU (S),
(2) níveis de multiprogramação (M),
(3) número de instruções de "overhead" devido à multiprogramação (D_1),
(4) constante (t_0) devido à tecnologia da máquina e
(5) taxa de transferência, "seek" e latência para o cálculo do tempo dispendido para obter a informação no disco (i).

A obtenção dos dados acima está fora do escopo deste trabalho. Tais dados são considerados especificações dos computadores, devendo ser informados pelos fornecedores dos equipamentos. Na metodologia de desenvolvimento de sistemas que está sendo apresentada (fig. 4.12 e 4.13), este conjunto de dados é uma entrada da quarta etapa, sendo chamado de dados_dos__computadores (fig. 5.39). Em [STR84], [BUC79] e [WIE77], são apresentados alguns valores, que podem dar uma idéia da magnitude destas especificações.

Os dados do segundo grupo são aqueles que dependem da aplicação:

(1) número de instruções (de máquina) executadas pelo programa, em cada ciclo de entrada e saída ($1/\mu_0$),

(2) número de ciclos de entrada e saída (R) e

(3) tempo de raciocínio (Z).

A obtenção destes dados é descrita na próxima seção.

5.5.3.1 Dados da aplicação

O número de instruções ($1/\mu_0$) e os ciclos de entrada e saída (R) são obtidos ao mesmo tempo. O processo de obtenção tem três partes. A primeira consiste em selecionar a amostra sobre a qual são feitas as medições. Esta amostra é composta pelos programas classe A da curva de classificação ABC. Os programas classe A, apesar do pequeno número,

são responsáveis por cerca do 80% da carga do computador. A curva ABC é descrita mais adiante nesta seção.

O precursor da filosofia de classificação ABC é Paretto, um pensador italiano que viveu por volta de 1800. A popularização desta classificação se deve a Henry Ford Dixie. Dixie publicou o seu trabalho em 1951, depois de experiências bem sucedidas no controle de estoques da General Electric Corporation [RUS79].

Para fazer a classificação é preciso organizar os dados. Para tanto é construída uma planilha com seis colunas conforme a apresentada na fig. 5.47. A primeira e a quarta colunas são obtidas a partir da planilha da fig. 5.34. A segunda coluna é o número acumulado de programas. A terceira coluna é a percentagem da coluna anterior em relação ao total de programas. A quarta coluna deve estar classificada em ordem decrescente. Na quinta coluna são registradas as somas parciais da coluna anterior. A sexta coluna contém a percentagem da coluna anterior em relação ao total da coluna quatro. Com esses dados é construída a curva de classificação ABC (fig. 5.48), conforme segue.

(1) dois eixos com escalas iguais são traçados. A abscissa corresponde à percentagem acumulada de programas, que é a quarta coluna da planilha (fig. 5.47). A ordenada é a percentagem acumulada das taxas de execução, sendo formada pelos dados da sexta coluna da planilha.

(2) a curva (M) é traçada.

DADOS PARA CLASSIFICAÇÃO A-B-C DE PROGRAMAS

TIPO DE SÍTIO - CENTRO REGIONAL

PROGRAMA	NÚMERO ACUMULADO DE PROGRAMAS	% NUMERO ACUMULADO DE PROGRAMAS	TAXA DE EXECUÇÃO DOS PROGRAMAS	TAXA ACUMULADA	% TAXA ACUMULADA
1	2	3	4	5	6
1.2.D.1	1	5,9	3600	3600	45
1.2.D.2	2	11,8	1600	5200	66
1.3.D.1	3	17,6	600	5800	73
1.3.D.5	4	23,5	400	6200	78
1.4.D.4	5	29,4	350	6550	83
1.4.D.1	6	35,3	280	6830	86
1.2.2.D.5	7	41,2	264	7094	90
1.2.4.D.6	8	47,1	240	7334	93
1.1.D.3	9	52,9	104	7438	94
1.1.D.4	10	58,8	90	7538	95
1.5.1.D.1	11	64,7	80	7618	96
1.4.D.2	12	70,6	70	7688	97
1.2.D.3	13	76,5	60	7748	98
1.3.D.2	14	82,3	50	7798	99
1.4.D.5	15	88,2	40	7838	99
1.5.2.D.1	16	94,1	30	7868	99
1.2.1.D.7	17	100,0	20	7888	100

Fig. 5.47 Organização dos dados para a classificação A-B-C

(3) a reta que passa pelos pontos (D) e (E) é traçada.

(4) uma paralela à reta (DE) é traçada. Esta reta é tangente à curva (M) e determina os pontos (F) e (G).

(5) a bissetriz do ângulo (DFG) é traçada, dando origem ao ponto (H).

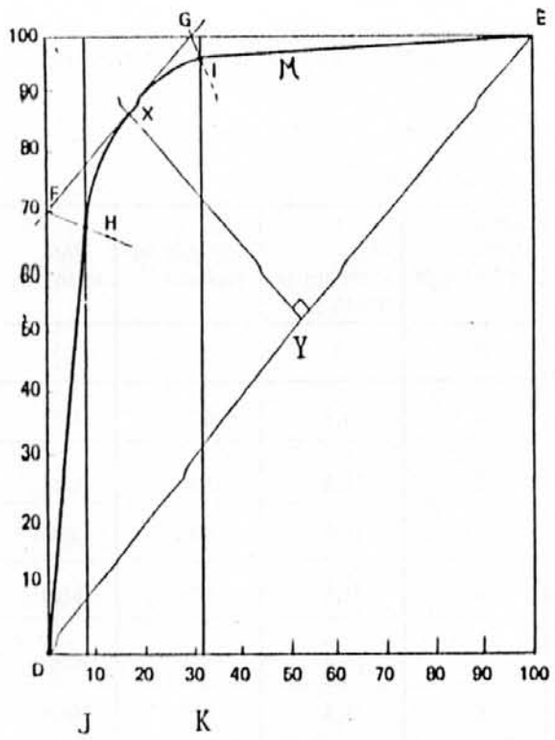


Fig. 4.48 Curva de classificação A-B-C

(6) a bissetriz do ângulo (FGE) é traçada, dando origem ao ponto (I).

(7) pelos pontos (H) e (I) são traçadas duas perpendiculares ao eixo das abscissas, dando origem respectivamente aos pontos (J) e (K).

Os programas classe A estão no intervalo (0,J], os classe B entre (J,K] e o restante é classe C.

O método de obtenção da curva ABC descrito acima encontra-se em [RUS79].

O conjunto de programas que ficou no intervalo da classe A compõe a amostra.

A segunda parte do processo de obtenção de $(1/\mu_0)$ e (R), consiste em instalar os programas da amostra nos com-

putadores candidados à ocupar o nó. Estes programas são adaptados ao "software" básico do computador em questão e compilados. Normalmente tal trabalho é feito no "show-room" do fabricante.

A terceira etapa consiste em efetuar as medições. Para tanto é feito um "benchmark". Durante um determinado intervalo de tempo o ambiente da aplicação deve ser simulado. Para efetuar esta simulação são usadas as demandas (figs. 5.28 a 5.30), as descrições do método (fig. 5.16) bem como os dados de tempo de execução dos processos manuais (fig. 5.32). As transações são disparadas num terminal. O computador deve estar provido de uma ferramenta capaz de registrar o número de instruções executadas e o número de acessos a disco ocorridos à cada transação. No final do período da simulação estes dados são listados e tabulados. O resultado é o total de transações disparadas, total de instruções executadas e total de acessos a disco efetuados. Desta forma:

$$R = \frac{\text{total de acessos a disco}}{\text{total de transações}}$$

$$1/\mu_0 = \frac{\text{total de instruções de máquina}}{\text{total de transações} \times R}$$

Como foi visto, está sendo sugerido que os dados acima sejam obtidos através de "benchmark". Esta é a maneira mais segura, uma vez que tais dados dependem de variáveis como a linguagem no qual os programas foram codificados, al-

goríTmo do programa, demanda da aplicação, método usado para executar os processos manuais, arquitetura do processador, sistema operacional, sistema gerenciador de banco de dados e outros. Um "benchmark" como o descrito anteriormente, permite que todas estas variáveis sejam levadas em consideração. O próximo dado a ser obtido é o tempo de raciocínio (Z).

O tempo decorrido entre o recebimento da resposta da transação anterior e o disparo da próxima é chamado tempo de raciocínio (Z). Normalmente este tempo varia entre 10 e 25 segundos. Alguns autores costumam usar a média (15 seg) [STR84]. Entretanto toda aplicação tem um valor característico de (Z), que pode ser calculado como segue:

$$Z = \frac{3600 \times \text{número de terminais}}{\text{total de transações por hora}} - \text{tempo de resposta}$$

O primeiro fator da fórmula acima, permite calcular o intervalo de tempo (seg) em que um usuário, operando um terminal, dispara duas transações consecutivas. Para saber o tempo de raciocínio, basta subtrair o tempo de resposta. O número de terminais pode ser obtido no "layout", conforme exemplo apresentado na fig. 5.46. O total de transações por hora está na planilha da fig. 5.34. O tempo de resposta é um requisito do sistema.

Uma vez obtidos os dados acima, a quarta atividade é encerrada, podendo ser iniciada a quinta - projeto da configuração básica (fig. 5.39). O modo de proceder para levar a cabo tal atividade foi amplamente discutido na 3.4.

A última atividade, projeto da configuração final (fig. 5.39), também foi amplamente discutida na seção 3.5.

Até aqui foi descrita a quarta etapa - dimensionamento dos recursos (fig. 4.13). Na próxima seção será descrita a quinta - instalação do sistema.

5.6 Instalação do sistema

O objetivo da quinta etapa, instalação do sistema, é reunir os recursos de "hardware", "software" e arranjo físico de acordo com o projeto, treinar pessoal e por o sistema em funcionamento. O contexto desta etapa na metodologia é apresentado na fig. 5.49. O significado das entradas e saídas foi descrito na seção 4.3.

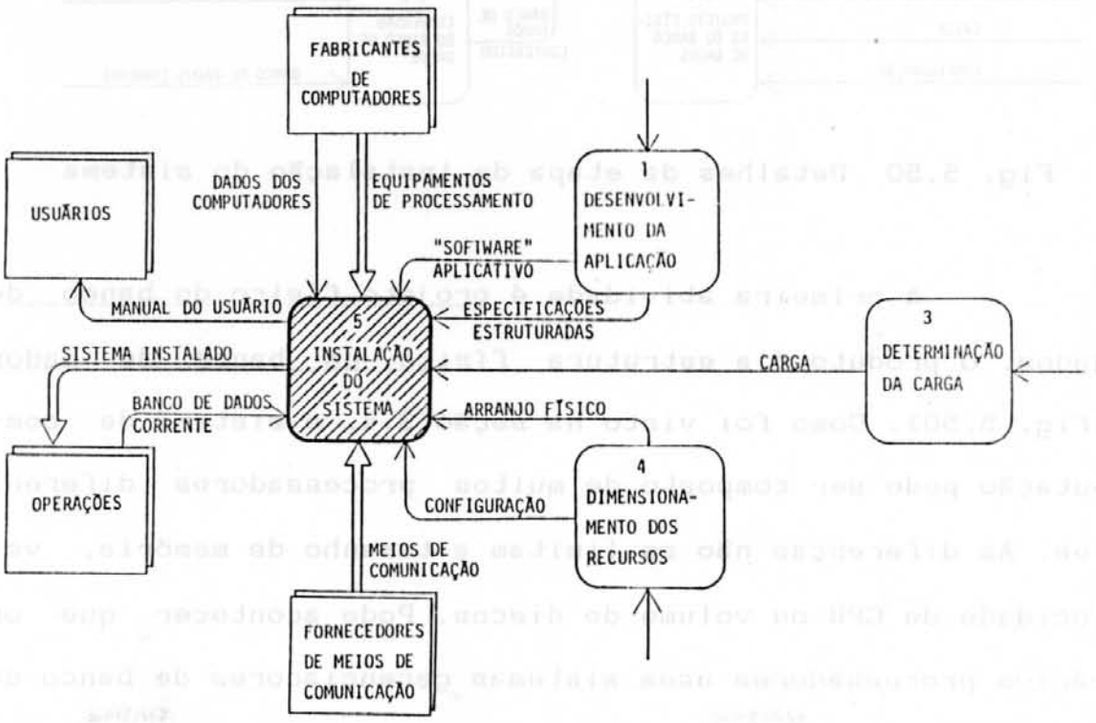


Fig. 4.49 Contexto da etapa de instalação do sistema

A quinta etapa é composta de seis atividades: (1) projeto físico do banco de dados, (2) adaptação dos programas, (3) descrição de procedimentos, (4) instalação dos recursos, (5) conversão do banco de dados e (6) instalação do "software". As relações entre as atividades acima são apresentadas sob a forma de DFD na fig. 5.50. Estas atividades são descritas a seguir.

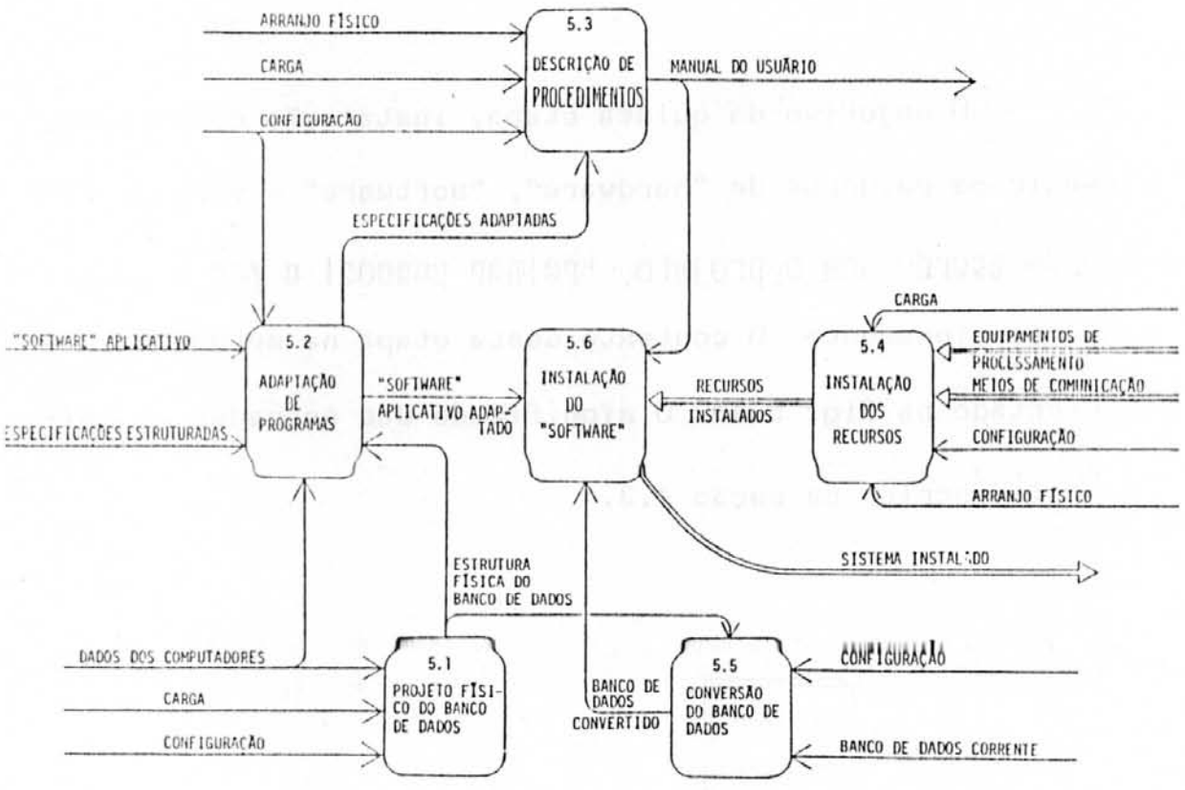


Fig. 5.50 Detalhes da etapa de instalação do sistema

A primeira atividade é projeto_físico_do_banco_de_dados. O produto é a estrutura_física_do_banco_de_dados (fig. 5.50). Como foi visto na seção 3.4, o sistema de computação pode ser composto de muitos processadores diferentes. As diferenças não se limitam a tamanho de memória, velocidade de CPU ou volume de discos. Pode acontecer que os vários processadores usem sistemas gerenciadores de banco de dados (SGBD) diferentes. Desta forma, os dados a serem armazenados precisam estar organizados segundo as característi-

cas destes SGDBs. Por esta razão são projetas tantas estruturas físicas de banco de dados quantos sejam os SGDBs.

A segunda atividade é adaptação_dos_programas. Existem dois produtos: especificações_adaptadas e software aplicativo_adaptado (fig. 5.50). Como foi visto na seção 4.3.3, as especificações estruturadas e os programas de aplicação são pré-requisito para o projeto da configuração do sistema. Como a configuração pode ser composta de vários computadores diferentes, para que os programas possam ser processados é necessário adaptá-los. Por essa razão são criadas tantas versões do "software" e das respectivas especificações, quantos forem os tipos de computadores existentes na configuração.

A terceira atividade é descrição_de_procedimentos. O produto é o manual_do_usuário (fig. 5.50). Esta atividade consiste basicamente em reunir os documentos produzidos nas várias etapas da metodologia.

A quarta atividade é instalação_dos_recursos (fig. 5.50). Esta atividade pode ser dividida em duas fases. Na primeira as salas devem ser preparadas: móveis e terminais devem ser posicionados de acordo com o "layout". A segunda fase compreende a instalação dos equipamentos de processamento de dados e dos meios de comunicação.

A quinta atividade é a conversão_do_banco_de_dados (fig. 5.50). Esta atividade consiste em reorganizar os dados existentes segundo as novas estruturas físicas.

A sexta atividade é a instalação do "software" (fig. 5.50). Esta atividade é a transição do antigo para o novo sistema. Como foi visto, ela pode acontecer na passagem de um dia para o próximo. Algumas vezes é feita na virada do mês. Em outras a instalação é um processo gradual onde os usuários recebem manuais, "hardware", são treinados e o sistema vai sendo progressivamente instalado.

Uma vez executada a atividade de instalação do "software", a quinta etapa, instalação do sitema, está concluída. Isso encerra o ciclo de projeto de sistemas distribuídos.

5.7 Resumo

Neste capítulo foram detalhadas as cinco etapas da metodologia proposta no capítulo 4, bem como descritos os procedimentos de obtenção e organização dos dados quantitativos necessários ao projeto de sistemas distribuídos.

A primeira etapa é o desenvolvimento da aplicação. Tal etapa é composta de seis atividades. É oriunda do ciclo de vida estruturado [YOU82] e tem como objetivo a produção de "software". A principal atividade é a análise. Tal atividade, de acordo com [YOU82], consiste basicamente em (1) documentar o modelo físico atual, (2) derivar o modelo lógico atual, (3) propor um novo modelo lógico, (4) estabelecer as fronteiras homem máquina e (5) propor um novo modelo físico. O modelo físico é a base para a produção do "software". No presente trabalho está sendo proposta a criação de

um novo modelo, chamado modelo homem/máquina.

O modelo físico tal como proposto em [DEM78] e [YOU82], só leva em consideração o computador. Entretanto, a implementação dos processos manuais obrigam à criação de novos processos automatizados e vice-versa. Desta forma, na metodologia aqui proposta, foram feitas modificações: uma vez obtido o novo modelo lógico, é estabelecida a fronteira homem/máquina, o que origina o novo modelo físico. O novo modelo físico só é diferente do novo modelo lógico pela existência de uma linha divisória entre os processos a serem automatizados e os manuais. Com base na fronteira homem/máquina do modelo físico é concebido o diagrama de descrição do método, que é uma implementação dos processos manuais, dando origem ao modelo homem/máquina. Este modelo é a base para a produção do "software".

As quatro etapas restantes são: (2) determinação da área geográfica, onde são levantados os mapas que descrevem o território por onde o sistema está disperso; (3) determinação da carga, onde são estabelecidas as demandas de execução dos processos manuais e automatizados; (4) dimensionamento dos recursos, onde é projetada a topologia, selecionados os equipamentos e concebido o arranjo físico e (5) instalação do sistema, onde são reunidos os recursos de "hardware", "software", móveis, espaço físico e treinadas as pessoas, o que possibilita por o sistema em funcionamento.

6 CONCLUSÃO

Os objetivos deste capítulo são: fazer uma retrospectiva do trabalho aqui apresentado e propor desenvolvimentos futuros.

6.1 Retrospectiva

Nesta seção são discutidos: como os objetivos do trabalho foram alcançados e que resultados foram obtidos.

6.1.1 Como os objetivos foram alcançados

Este trabalho foi elaborado com o propósito de atingir três objetivos. O primeiro é a integração das dissertações de Ohana [OHA79] e Strack [STR80]. Ohana propôs modelos de programação linear para distribuição de arquivos e Strack descreveu modelos de filas para a análise de desempenho de computadores. O segundo objetivo é a obtenção e organização de dados quantitativos para o uso efetivo dos modelos, uma vez que nas dissertações acima não há orientação de como obter tais dados. O terceiro objetivo é a integração das técnicas de processamento eletrônico de dados (PED), organização e métodos (O&M) e pesquisa operacional (PO). Através desta integração é possível obter e organizar os dados anteriormente referidos. Para alcançar os objetivos foi proposta uma metodologia de projeto de sistemas distribuídos.

A metodologia tem cinco etapas: (1) desenvolvimento da aplicação, (2) determinação da área geográfica, (3)

determinação da carga, (4) dimensionamento dos recursos e (5) instalação do sistema. Estas etapas não precisam ser executadas sequencialmente como nas metodologias clássicas. Elas podem ser executadas em paralelo, como no ciclo de vida estruturado proposto por Yourdon [YOU82]. Como será visto a seguir, estas etapas foram concebidas com base nos objetivos descritos no parágrafo anterior.

A quarta etapa, dimensionamento dos recursos, constitui o núcleo a partir do qual os objetivos foram alcançados. Os produtos desta etapa são (1) a configuração do sistema de computação e (2) o arranjo físico. A configuração é concebida em três níveis de abstração. O primeiro nível é a topologia, que pode ser projetada com base nos modelos descritos por Ohana [OHA79]. O segundo nível é a configuração básica. Neste nível os nós da topologia são substituídos por computadores. A seleção dos computadores com a capacidade adequada a cada nó, é feita através dos modelos propostos por Strack [STR80]. O terceiro nível é a configuração final, sendo uma abstração da realidade. É constituída de uma planta do sistema de computação, onde estão descritos equipamentos e meios de comunicação a serem adquiridos e instalados. Tal planta é projetada a partir dos dois níveis anteriormente referidos. O segundo produto, o arranjo físico, é feito com base nas técnicas de O&M, sendo uma planta da disposição dos recursos no espaço físico. As demais etapas da metodologia foram concebidas para viabilizar a quarta etapa.

A primeira etapa, desenvolvimento da aplicação, foi concebida com o propósito de produzir o "software" apli-

cativo. Os programas da aplicação são pré-requisito dos modelos propostos por Strack [STR80]. A segunda etapa, determinação da área geográfica, tem produtos como mapas e plantas, que contêm localizações e distâncias usadas nos modelos descritos por Ohana [OHA79]. A terceira etapa, determinação da carga, tem o objetivo de obter as taxas de execução de processos manuais, taxas de execução de programas, acesso aos arquivos e tamanho de mensagens. Tais dados são necessários em ambos os modelos acima referidos. A quinta etapa, instalação do sistema, tem dois propósitos. O primeiro é adaptar o "software" aplicativo à configuração. Esta adaptação é praticamente obrigatória, uma vez que os programas são produzidos antes da configuração. O segundo propósito é instalar equipamentos, treinar pessoal e por o sistema em funcionamento.

6.1.2 Resultados obtidos

Os principais resultados do trabalho aqui descrito são: (1) integração, (2) quantificação, (3) possibilidade de crescimento progressivo e (4) modelo homem/máquina.

O primeiro resultado é a integração. De um modo geral, os produtos de engenharia são projetados e fabricados por uma grande quantidade de especialistas. Tais especialistas atuam segundo um plano integrado, cada um complementando o trabalho de seus antecessores, até que o produto fique pronto e seja entregue aos usuários. Alguns exemplos são carros, prédios, estradas e outros. Os sistemas de informação também podem ser considerados produtos de engenharia.

Entretanto as várias técnicas para projeto e implementação destes sistemas são apresentadas de forma isolada na literatura. Nesta dissertação, técnicas de O&M, PO e PED foram reunidas num todo coerente e harmonioso, permitindo a ação integrada dos profissionais das várias áreas, na elaboração do ciclo completo de um projeto de sistemas distribuídos.

O segundo resultado é a quantificação. São raros os trabalhos na área de sistemas de informação que orientam quanto ao dimensionamento dos recursos. Na maior parte dos casos, os computadores são instalados sem que a carga de processamento seja previamente dimensionada. A metodologia aqui apresentada permite quantificar todos os recursos necessários ao sistema, possibilitando estabelecer os seus limites, facilitando o gerenciamento e permitindo a rápida detecção da causa de eventuais quedas de performance. Além disso, a capacidade ociosa pode ser previamente conhecida, um plano de ajuste no desempenho pode ser elaborado e a sobrecarga oriunda do crescimento vegetativo pode ser evitada.

O terceiro resultado é a possibilidade de crescimento progressivo. Bancos, lojas, farmácias, supermercados, hospitais, empresas de turismo e indústrias, atuam distribuindo geograficamente suas funções. Essa distribuição é feita através do conceito de tipo. Por exemplo, um banco pode ter vários tipos de agências: central, urbana, posto de serviço, loja de poupança, rural e outras. Cada tipo tem um elenco de funções e é dotado de recursos adequados a uma determinada faixa de demanda. Os tipos são projetados previamente. Essa prática simplifica a administração, torna a em-

presa modular e permite que as evoluções do mercado sejam absorvidas com agilidade. Como os projetos estão sempre prontos, para crescer basta implementar o tipo mais adequado à demanda do local. A metodologia aqui descrita, está baseada integralmente nesta estratégia, permitindo um crescimento progressivo do sistema de informação, perfeitamente sincronizado com a evolução real das empresas.

O quarto resultado é o modelo homem/máquina. Desde a introdução dos computadores nas empresas, os sistemas de informação passaram a ter duas partes: uma manual e outra automatizada. Para ambas foram criadas técnicas de desenvolvimento, que evoluíram paralelamente uma à outra. Os especialistas em sistemas manuais não dispõe de uma ferramenta de ligação com a parte automatizada e vice-versa, o que gera conflitos durante o projeto do sistema e ineficiência na operação. O modelo homem/máquina proposto neste trabalho estabelece uma interface entre as partes, o que resulta em três vantagens. A primeira é a uniformização da linguagem entre os especialistas. A segunda é a simplificação da operação do sistema pelo usuário, uma vez que os procedimentos são elaborados de forma clara, simples e precisa. A terceira é a eficácia. Esta vantagem resulta de: (1) o método de trabalho manual é projetado em conjunto com a parte automatizada, resultando num sistema integrado e (2) a união das técnicas estruturadas com as ferramentas clássicas de O&M e engenharia industrial, facilita a comunicação e estimula a participação mais intensa dos usuários no projeto.

Até aqui foi feita uma retrospectiva a respeito

deste trabalho. Na próxima seção são propostos desenvolvimentos futuros.

6.2 O futuro

Nesta seção são discutidos: (1) sugestões para futuras investigações, (2) o que ainda precisa ser feito para complementar este trabalho e (3) onde usar a metodologia.

6.2.1 Sugestões para futuras investigações

Durante a elaboração deste trabalho surgiram quatro questões as quais podem ser melhor investigadas no futuro: (1) projeto de sistemas distribuídos com micro-computadores, (2) uso de modelos de decisão na seleção de computadores, (3) normalização de modelos de computadores e (4) influência da dinâmica das atividade no projeto de banco de dados.

A primeira sugestão de investigação diz respeito ao projeto de sistemas distribuídos com micro-computadores. Seja por exemplo uma rede de supermercados com apenas um tipo de loja. A metodologia proposta neste trabalho permite que, com base na taxa de chegada de clientes, seja calculado o número de caixas, número de terminais e seja selecionado o computador com a capacidade adequada. Em cada loja a topologia do sistema tem a forma de estrela. O nó central é ocupado por um computador e os nós adjacentes são terminais que ficam instalados junto aos caixas. Entretanto esta é apenas uma alternativa. Uma outra possibilidade seria a instalação

de micros junto aos caixas, que ficariam ligados em rede. Esta possibilidade não foi estudada no presente trabalho, podendo ser investigada no futuro. Algumas questões a serem respondidas são: como distribuir os dados entre os micros, qual a topologia mais adequada, como varia o tempo de resposta, o uso do conceito de servidor é adequado, qual o custo da solução em micro se comparada à solução com terminais e outras.

A segunda sugestão de investigação é o uso de modelos de decisão na seleção de equipamentos. A maioria dos trabalhos nesta área concentra a análise na relação preço/performance. Há entretanto outras variáveis tão ou mais restritivas que esta, como por exemplo a garantia de continuidade de fornecimento. Não é raro encontrar nas empresas equipamento inutilizado porque o fornecedor não está mais no mercado. Poderão servir de base para a elaboração destes modelos alguns trabalhos como [MAR85], [KEE82], [JON77] e [KEP77].

A terceira sugestão diz respeito à normalização dos modelos de computadores. A maioria dos produtos comercializados têm suas especificações normalizadas. São exemplos o aço, produtos químicos, cabos de transmissão de energia, componentes eletrônicos e outros. As especificações permitem que produtos mais complexos possam ser previamente dimensionados a partir dos componentes, de forma que os limites de carga sejam conhecidos e respeitados para evitar acidentes. Prédios, pontes e aeronaves são exemplos destes produtos. Dada a crescente influência dos sistemas de infor-

mação na vida dos cidadãos, cada vez mais se torna necessário estabelecer seus limites antes de instalá-los. Isto entretanto tem sido dificultado pela falta de fornecimento de dados adequados por parte dos fabricantes.

Os dados mais comuns são a velocidade da CPU (em MIPS), os tempos de acesso a disco e as velocidades das impressoras. Tais dados são insuficientes. Há outras variáveis como o funcionamento do sistema operacional, gerenciador do banco de dados e monitor de comunicação que precisam ser levadas em conta. A sugestão é que os fabricantes forneçam vários modelos, um para cada conjunto computador/"software" básico, juntamente com os dados atualmente fornecidos. Aos usuários destes equipamentos bastaria aplicar no modelo os dados da aplicação tais como número de instruções executadas por programa, acesso aos arquivos e taxa de execução das transações, estabelecendo assim os limites de segurança de seus sistemas.

A maioria dos equipamentos eletrônicos é comercializada juntamente com seus esquemas eletricos, que na verdade são modelos dos produtos. Tais modelos podem parecer estranhos para os leigos, mas são extremamente úteis aos especialistas quando da manutenção ou uso de tais equipamentos como componentes de produtos mais complexos. O modelo de filas, tal como um esquema elétrico, cubriria esta lacuna existente na especificação dos recursos de processamento eletrônico de dados.

A quarta sugestão se refere à influência do dina-

mismo das diversas atividades da empresa no projeto do banco de dados. Existe uma defasagem entre a realidade e as informações armazenadas nos arquivos. A realidade se modifica de três formas: (1) novos objetos são criados ao mesmo tempo em que novos relacionamentos são estabelecidos entre objetos, (2) objetos e relacionamentos são eliminados e (3) as características dos objetos e relacionamentos se modificam. Os arquivos são atualizados de forma correspondente: (1) inclusão de novos registros, (2) exclusão de registros ou (3) alteração do conteúdo dos campos de registros existentes. A defasagem entre a modificação da realidade e a atualização dos arquivos deve ser a menor possível. Entretanto, para uma grande parcela de aplicações, o dinamismo provocado pela (1) criação e (2) eliminação de objetos e relacionamentos não é desejável. Há casos em que a efetividade do sistema fica comprometida se os arquivos estiverem refletindo a realidade presente. Ficam as questões: a solução é usar várias versões do mesmo arquivo? Isso não provoca redundância? O mesmo objeto poderia ter duas chaves diferentes? Os objetos deveriam ter conjuntos de atributos de acordo com o instante ("snapshot") em que os dados são atualizados ? De uma forma geral as empresas tem optado por mais de uma versão do mesmo arquivo.

6.2.2 O que ainda precisa ser feito

Para complementar este trabalho, há ainda três etapas a serem cumpridas: (1) automação de algumas partes da metodologia, (2) análise de questões gerenciais e (3) uso da metodologia numa aplicação piloto.

A primeira etapa a ser cumprida se refere à automação de algumas partes da metodologia. Analistas e programadores tem resistido ao uso da análise e projeto estruturados, devido ao grande volume de trabalho dispendido no desenho e alteração dos diagramas [YOU86]. Por esta razão, toda a parte gráfica deveria receber ferramentas automatizadas, o que racionalizaria a elaboração de DFDs, mapas e plantas. Além das ferramentas para a parte gráfica, também são necessários programas para a solução de modelos matemáticos, simulação e teoria das filas. Como é sabido, o volume de cálculos requeridos por estas técnicas, só as torna viáveis através da automação.

A segunda etapa se refere a análise de pelo menos duas questões gerenciais. A primeira se refere à organização da equipe de projeto. Na metodologia está claro que ferramentas usar em cada etapa. É preciso definir o perfil dos profissionais responsáveis pela elaboração de cada produto e qual a relação chefe-subordinado-colega entre eles. Weinberg [WEI80] apresenta sugestões à respeito desta questão. A segunda é o estabelecimento de um plano mestre para o acompanhamento do projeto, que pode ser um "check-list", uma rede PERT, um gráfico de GANTT ou outra ferramenta. O objetivo é possibilitar o estabelecimento de prazos para facilitar a atuação gerencial.

A terceira etapa a ser cumprida é o uso da metodologia num caso real. Em alguns testes efetuados ficou demonstrado que o modelo homem/máquina é uma ferramenta eficaz.

6.2.3 Onde usar a metodologia

Esta metodologia foi elaborada para produzir "software" e configurar o "hardware" de aplicões dispersas geograficamente. O resultado é um sistema de computação dedicado à uma aplicação. Portanto não é adequada ao projeto de redes abertas de transmissão de dados do tipo RENPAC. Ela deve ser usada no projeto de sistemas do nível operacional, sendo apropriada às empresas do setor comercial como lojas, farmácias, supermercados e outras; setor de serviços como bancos, locadoras de automóveis, hotéis e outras; bem como algumas indústrias onde as funções são distribuídas.

Apesar da metodologia ter sido concebida para aplicações distribuídas como as descritas no parágrafo anterior, o modelo homem/máquina a torna uma ferramenta eficaz em outras área tais como: (1) controle de processo e (2) engenharia industrial.

Existem dois ramos do controle de processo onde o modelo homem/máquina pode ser usado. O primeiro se refere aos sistemas de controle de processos industriais, como fabricação de produtos químicos, refinarias de petróleo, produção de aço, geração de energia elétrica, geração de imagens de televisão e outros. Nestes sistemas, parte do processo é comandada por computadores e parte por pessoas. Existe sempre uma sala de comando, onde há telas que tornam as informações visíveis e dispositivos que permitem às pessoas interagirem com o sistema, interferindo no andamento do processo sempre que necessário. Como é possível notar, exis-

tem processos manuais, automatizados e há interação entre eles, o que faz do modelo homem/máquina uma ferramenta ideal para a especificação do "software" para este tipo de sistema.

O segundo ramo do controle de processo que poderia ser beneficiado por este trabalho é o das máquinas automatizadas. Alguns exemplos são navios, aviões, trens, automóveis e outros. Os sistemas de controle são semelhantes aos descritos no parágrafo anterior. A diferença é que no lugar da sala de comando existe a cabine do piloto ("cockpit"). Também nestes casos há processos manuais, automatizados e interações, o que torna o modelo homem/máquina adequado à especificação do "software" de controle destas máquinas.

A segunda área onde a metodologia aqui apresentada pode ser usada é a engenharia industrial. Os processos industriais são interativos na sua maioria. Homems e máquinas vão se alternando na execução de operações, até que o trabalho seja concluído. Um exemplo é a reprodução de documentos. O operador abre a janela da máquina, coloca o original, fecha a janela, aperta um botão e espera. Um barulho característico indica que a máquina já efetuou a reprodução. O operador então repete as operações até que não haja mais originais a serem copiados. Enquanto o operador espera, a máquina faz a sua parte: obtem a imagem do original, alimenta o papel em branco, imprime, fixa a imagem e finalmete ejeta a cópia. Nos manuais das máquinas de reprodução, todo este processo é descrito através de narrativas, figuras e fotografias, com resultados nem sempre claros. Há outros exem-

plos como prensas, furadeiras, tornos, câmara de pintura, fornos de tratamento térmico e outros. O modelo homem/máquina seria uma boa alternativa para descrever tais processos, pois incorpora todas as qualidades de simplicidade e clareza oriundas das ferramentas estruturadas.

Para finalizar, é conveniente ressaltar a importância da integração entre os processos manuais e automatizados. A maioria das metodologias sempre esteve voltada para a implementação dos programas e a atenção aos processos manuais tem se resumido ao manual de procedimentos. A proposta deste trabalho é que os sistemas sejam projetados como um todo, com as interações homem/máquina sendo analisadas e implementadas, fazendo com o que o "software" não fique restrito aos processos automatizados, mas seja uma extensão do trabalho executado pelas pessoas.

BIBLIOGRAFIA

[AGA85] AGASSI, S. Performance considerations for a distributed data processing system designed for high availability. IBM Systems Journal, Armonk, 24(3/4):200-12, July 1985.

[AND75] ANDERSON, George A. & JENSEN, E. Douglas. Computer interconnection structures: taxonomy, characteristics and examples. Computing Surveys, New York, 7(4):197-213, Dec. 1975

[ANO85] ANONYMOUS et alii. A measure of transaction processing power. Datamation, New York, 31(7):112-8, Apr. 1985.

[ARA85] ARAÚJO, Luis Cesar G. Organização & Métodos. São Paulo, Atlas, 1985.

[BAR77] BARNES, Ralph M. Estudo de Movimentos e Tempos: Projeto e Medida do Trabalho. São Paulo, Edgard Blucher, 1977.

[BAR82] BARNES, J. G. P. Programming in ADA. London, Addison-Wesley, 1982.

[BOO79] BOORSTYN, Robert R. & FRANK, Howard. Large-scale network topological optimization. In: MARIANI, P. Michael & PALMER, David. Tutorial: Distributed System Design, New York, IEEE, 1979.

[BOO82] BOOTH, Grayce M. Planned migration to distributed systems. In: AUERBACH Distributed Processing Management. Phyladelphia, 1982. rep. 62-02-04.

[BUC79] BUCCI, Giacomo & STREETER, Donald N. A methodology for the design of distributed information systems. Communications of the ACM. New York, 22(4):233-45, Apr. 1979.

[BUR75] BURROUGHS CORPORATION. B7700/B6700 System tempo mathematical programming system user's manual. Aug. 1975.

[CER82] CERI, S.; NEGRI, M.; PELAGATTI, G. Horizontal data partitioning in data base design. In: SIGMOD INTERNATIONAL CONFERENCE ON MANAGEMENT OF DATA, Orlando, June 2-4, 1982. Proceedings. New York, ACM, 1982. p.128-36.

[CHA80] CHAMPINE, George A. Distributed Computer Systems, Impact on Management, Design and Analysis. Amsterdam, North-Holland, 1980.

[CHE77] CHEN, Peter Pin-Shan. The entity-relationship model - a basis for the enterprise view of data. In: NATIONAL COMPUTER CONFERENCE, Dallas, June 13-16, 1977. Proceedings. Montvale, AFIPS press, 1977. p. 77-84.

[CHE80] CHEN, Peter Pin-Shan & AKOKA, Jacob. Optimal design of distributed information system. IEEE Transactions on Computers, New York, CC-29(12):1068-80, Dec. 1980.

[CHI83] CHIAVENATO, Idalberto. Introdução à Teoria Geral da Administração. São Paulo, McGraw-Hill, 1983.

[COD70] CODD, E. F. A relational model of data for large shared data banks. Communications of the ACM, New York, 13(6):377-87, June 1970.

[COD79] CODD, E. F. Extending the database relational model to capture more meaning. ACM Transaction on Database Systems, New York, 4(4):397-434, Dec. 1979.

[DEM78] DeMARCO, Tom. Structured Analysis and System Specification. New York, Yourdon Press, 1978.

[DOL78] DOLL, Dixon R. Data Communications. New York, Wiley, 1978.

[DOW76] DOWN, P. J. & TAYLOR, F. E. Why Distributed Computing? Manchester, NCC, 1976.

[EIN77] EIN-DOR, Phillip. A dynamic approach to selecting computers. Datamation, New York, 23(6):103-8, June 1977.

[GAN83] GANE, Chris & SARSON, Trish. Análise Estruturada de Sistemas. Rio, LTC Editora, 1983.

[GAV82] GAVISH, Bezalel & PIRKUL, Hasan. Allocation of data bases and processors in a distributed computing system. In: INTERNATIONAL CONFERENCE ON MANAGEMENTE OF DISTRIBUTED DATA PROCESSING, Paris, June 23-26, 1982. Proceedings. Amsterdam, North-Holland, 1982. p.215-31.

[GAV85] GAVISH, B. Models for configuring large-scale distributed computing systems. AT&T Technical Journal, New York, 64(2.2):491-532, Feb. 1985.

[GEL77] GELDER, Allen V. Structured programming in COBOL: an approach for application programmers. Communications of the ACM, New York, 20(1):2-12, Jan. 1977.

[GIB74] GIBSON, Cyrus F. & NOLAN, Richard L. Managing the four stages of EDP growth. Harward Business Review, Boston, 52(1):76-88, Jan/Feb. 1974.

[GRI82] GRINGS, Luiz Carlos. Metodologia para avaliação de desempenho de sistemas de computação através de simulação. Porto Alegre, PGCC da UFRGS, 1982. (dissertação de mestrado)

[GRO73] GROSS, Bertram M. As Empresas e sua Administração. Petrópolis, Vozes, 1973.

[HIC78] HICE, G. F.; TURNER, W. S.; CASHWELL, L. F. System Development Methodology. Amsterdam, North-Holland, 1978.

[HOL77] HOLT, R. C. & HUME, J. N. P. Fundamentals of Structured Programming using FORTRAN with SE/K and WATIV-S. Reston, Reston Publishing, 1977.

[IBM70] IBM. General Information - Binary Synchronous Communications. New York, IBM, 1970.

[IFI82] IFIP WG 8.1 WORKING CONFERENCE ON COMPARATIVE REVIEW OF INFORMATION SYSTEMS DESIGN METHODOLOGIES, Noordwijkerhout, May 10-14, 1982. Proceedings. Amsterdam, North-Holland, 1982.

[JON77] JONES, J. Morgan. Introduction to Decision Theory. Homewood, Richard D. Irwin, 1979.

[KEE82] KEENEY, Ralph L. Decision analysis: an overview. Operations Research, Baltimore, 30(5):803-38, Sept./Oct. 1982.

[KEN83] KENT, W. A simple guide do five normal forms in relational database theory. Communications of the ACM, New York, 26(2):120-25, Feb. 1983.

[KEP77] KEPNER-TREGOE. GENCO II - Análise de Problema e Tomada de Decisão. Princenton, Kepner-Tregoe, 1977.

[MAR72] MARTIN, James. Systems Analysis for Data Transmission. Englewood Cliffs, Prentice-Hall, 1972.

[MAR77] MARCOTTY, Michael. Structured Programming with PL/I - An Introduction. Englewood Cliffs, Prentice-Hall, 1977.

[MAR84] MARTIN, James & MC'CLURE, Carma. Metodologias para Análise e Projeto de Sistemas. São Paulo, COMPUCENTER, 1984.

[MAR85] MARTIN, R.J. & TAYLOR, P.F. Micro, mini or mainframe - wat's right for you? Journal of the Operational Research Society, Birmingham, 36(3):217-24, Mar. 1985.

[MAY70a] MAYNARD, H. B. Manual de Engenharia de Produção - Métodos. São Paulo, Edgard Blucher, 1970.

[MAY70b] MAYNARD, H. B. Manual de Engenharia de Produção - Padrões de Tempos Elementares Pré-determinados. São Paulo, Edgard Blucher, 1970.

[MEL73] MÉLESE, Jacques. A Gestão pelos Sistemas. Rio, Livros Técnicos e Cientif., 1973.

[MEN84] MENASCÉ, Daniel A. & SCHWABE, Daniel. Redes de Computadores: Aspectos Técnicos e Operacionais. Rio, Campus, 1984.

[MEN85] MENASCÉ, Daniel & ALMEIDA, Virgílio A. F. Planejamento de Capacidade de Sistemas de Computação. Rio, Campus, 1985.

[NAV84] NAVATHE, S.; CERI, S.; WIEDERHOLD, G.; DOU, J. Vertical partitioning algorithms for database design. ACM Transactions on Database Systems, New York, 9(4):680-710, Dec. 1984.

[NOL79] NOLAN, Richard L. Managing the crisis in data processing. Harward Business Review, Boston, 57(2):115-36, Mar./Apr. 1979.

[OHA79] OHANA, Ivaldo. Um Estudo sobre Alocação de Arquivos em Banco de Dados Distribuídos. Porto Alegre, PGCC da UFRGS, 1979. (dissertação de mestrado).

[PAG80] PAGE_JONES, M. A Practical Guide to Structured System Design. New York, Yourdon Press, 1980.

[PAL79] PALMER, David F. Distributed computing system design at the subsystem/network level. In: INTERNATIONAL CONFERENCE ON DISTRIBUTED COMPUTING SYSTEMS, Huntsville, Oct. 1-5, 1979. Proceedings. New York, IEEE, 1979. p. 22-30.

[RUS79] RUSSOMANO, Vitor Henique. Planejamento e Acompanhamento da_Produção. São Paulo, Pioneira, 1980.

[SAA80] SAAD, Alfredo C. C. Como Prolongar a Vida Útil de uma Configuração de PD Reduzindo Custos e Melhorando a Qualidade dos Serviços. São Paulo, Edgard Blucher, 1980.

[SER86] SERLIN, Omri. Mips, drystones and other tales. Datamation, New York, 32(11):112-8, June 1986.

[STE85] STEVENS, Wayne P. Projeto Estruturado de Sistemas. Rio, Campus, 1985.

[STR80] STRACK, Jair. Metodologia para Modelagem de Sistemas Distribuídos. Porto Alegre, PGCC da UFRGS, 1980. (dissertação de mestrado).

[STR84] STRACK, Jair. GPSS Modelagem e Simulação de Sistemas. Rio, Livros Técnicos e Cientif., 1984.

[SUM86] SUMNER, Mary & SITEK, Jerry. Are structured methods for systems analysis and design being used? Journal of Systems Management, Cleveland, 37(6):18 June 1986.

[TAN81] TANENBAUM, Andrew S. Computer Networks. Englewood Cliffs, Prentice-Hall, 1981.

[TAR84] TAROUCO, Liane M. R. Redes de Comunicação de Dados. Rio, Livros Técnicos e Cientif., 1984.

[TAR86] TAROUCO, Liane M. R. Redes de Computadores Locais e de Longa Distância. São Paulo, McGraw -Hill, 1986.

[TAY79] TAYLOR, Frederick Winslow. Princípios de Administração Científica. São Paulo, Atlas, 1979.

[THI82] THIERAUF, Robert J. An analytical framework for going to DDP. In: AUERBACH Distributed Processing Management. Phyladelphia, 1982.

[WAG75] WAGNER, Harvey M. Principles of Operations Research whit Applications to Managerial Decisions. London, Prentice-Hall, 1975.

[WEI80] WEINBERG, Victor. Structured Analysis. New York, Yourdon Press, 1980.

[WIE77] WIEDERHOLD, Gio. Database Design. New York, McGraw-Hill, 1977.

[WIR74] WIRTH, Niklaus & JENSEN, Kathleen. Pascal user manual and report. Berlin, Springer-Verlag, 1974.

[YOU78] YOURDON, Edward & CONSTANTINE, Larry L. Structured Design. New York, Yourdon Press, 1978.

[YOU82] YOURDON, Edward. Managing the System Life Cycle. New York, Yourdon Press, 1982.

[ZAN80] ZANDIN, Kjell B. MOST - Work Measurement Systems.
New York, Marcel Dekker, 1980.

UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

PROJETO DE
SISTEMAS DISTRIBUÍDOS

Dissertação apresentada aos Srs.

Prof. José Palazzo M. de Oliveira

Prof. Maurizio Tazza

Prof. Norberto Hoppen

Visto e permitida a impressão

Porto Alegre, 08 / 12 / 87

pp/ Prof. Roberto Tom Price
COORDENADOR DO CURSO DE PÓS-GRADUAÇÃO
EM CIÊNCIA DA COMPUTAÇÃO