



Fundação Getúlio Vargas
Escola de Administração
de Empresas de São Paulo
Biblioteca



3406/81



1198103406

APLICAÇÕES DE MEDIDA DO TRABALHO EM ATIVIDADES DE PROGRAMAÇÃO DE COMPUTADORES

Dissertação apresentada, sob a orientação do Prof. Wolfgang Schoeps, como exigência parcial para a obtenção do grau de MESTRE EM ADMINISTRAÇÃO, à Congregação da Escola de Administração de Empresas de São Paulo da Fundação Getúlio Vargas.

Feito o depósito exigido em lei.
Todos os direitos reservados.
Proibida no todo ou em parte, publicação e/ou reprodução por qualquer processo, sem prévia autorização do autor. *Protocolo 1362*

*de 08/09/82 - Biblioteca
nacional. Litrificado
nº 27.266 - folha 261 -
hno 19 - 13.09.82*

SÃO PAULO - 1979

Feito o depósito exigido em lei.
 Todos os direitos reservados.
 Proibida no todo ou em parte, pu-
 blicação e/ou reprodução por qual-
 quer processo, sem prévia autorização
 do autor.

\$

| Escola de Administração de Empresas de São Paulo | |
|---|-----------------|
| Data | 65.015 |
| 10.11 | F812a |
| N.º de Volume | registrado por: |
| 3406/81 | M |

elis.
 2,2

681.3:658.5

COMISSÃO JULGADORA

COMISSÃO JULGADORA

AGRADECIMENTOS

Ao Prof. Wolfgang Schoeps pelo incentivo e objetividade manifestados constantemente ao orientar a realização deste trabalho;

à Francis, Paulo, Odair e todo o pessoal do C.P.D. da Escola de Administração de Empresas de São Paulo da Fundação Getúlio Vargas, pelas facilidades de hardware e de software que me foram oferecidas,

meu muito obrigado.

Í N D I C E

| | pág. |
|---|------|
| <u>CAPÍTULO I - INTRODUÇÃO</u> | 8 |
| 1. Objetivos Gerais | 8 |
| 2. Situação do Setor de Processamento Eletrônico de Dados no Brasil | 9 |
| 3. Descrição do Problema | 13 |
| 4. Análise do Corpo de Conhecimentos | 16 |
| 5. Resumo | 25 |
| <u>CAPÍTULO II - CARACTERÍSTICAS DO TRABALHO ANALISADO</u> | 27 |
| 1. A Função do Processamento Eletrônico de Dados | 27 |
| 2. Desenvolvimento de Sistemas para Computador. Ci clo de Vida de Sistemas | 28 |
| 2.1. Fase I - Concepção do Sistema | 31 |
| 2.2. Fase II - Especificações Funcionais | 32 |
| 2.3. Fase III - Projeto Detalhado | 33 |
| 2.4. Fase IV - Programação | 35 |
| 2.5. Fase V - Implantação | 38 |
| 2.6. Fase VI - Operação | 38 |
| 3. Organização da Função de Processamento de Dados | 39 |

CAPÍTULO III - FATORES QUE AFETAM O TRABALHO ANALISA-

| | |
|--|----|
| <u>DO</u> | 45 |
| 1. Fatores Ambientais | 45 |
| 1.1. Linguagem Utilizada | 45 |
| 1.2. Restrições de Hardware | 46 |
| 1.3. Acessibilidade do Computador | 46 |
| 1.4. Metodologia Empregada | 47 |
| 1.5. Padrões de Documentação | 49 |
| 1.6. Suportes de Software | 49 |
| 1.7. Procedimentos de Segurança | 50 |
| 1.8. Instalações Físicas | 50 |
| 2. Variáveis Inerentes a Cada Programa | 51 |
| 2.1. Tipo de Programa | 51 |
| 2.1.1. Programas de Atualização / Manutenção de Arquivos | 53 |
| 2.1.2. Programas de Consistência / Validação de Campos | 57 |
| 2.1.3. Programas de Impressão de Relatórios | 59 |
| 2.1.4. Programas que Organizam Arquivos. | 61 |
| 2.1.5. Programas que Formatam Registros. | 63 |
| 2.2. Características das Entradas e Saídas .. | 65 |
| 3. Experiências do Programador | 70 |

| | |
|--|----|
| <u>CAPÍTULO IV - ANÁLISE DAS TÉCNICAS EXISTENTES PARA ME-</u> | |
| <u>DIR O TRABALHO ANALISADO</u> | 73 |
| 1. Conceitos Básicos de Medida do Trabalho | 73 |
| 2. Técnicas de Medida do Trabalho | 79 |
| 2.1. Aproveitamento da Experiência Anterior .. | 80 |
| 2.1.1. Técnica dos Dados Históricos | 80 |
| 2.2. Observação Direta | 81 |
| 2.2.1. Técnica da Auto-Avaliação | 81 |
| 2.2.2. Técnica da Amostragem | 83 |
| 2.2.3. Técnica Cronometragem | 85 |
| 2.3. Síntese | 86 |
| 2.3.1. Técnica dos Dados-Parão | 86 |
| 2.3.2. Técnica dos Tempos Predeterminados | 87 |
| <u>CAPÍTULO V - DESCRIÇÃO DA METODOLOGIA PARA OBTENÇÃO</u> | |
| <u>DOS TEMPOS PADRÕES DAS TAREFAS DE PROGRA-</u> | |
| <u>MAÇÃO DE COMPUTADORES. UM CASO PRÁTICO .</u> | 89 |
| 1. Levantamento de Dados | 90 |
| 1.1. Esforço de Programação | 90 |
| 1.1.1. Aplicação de Amostragem do Traba- | |
| lho | 92 |
| 1.1.2. Aplicação da Técnica da Auto-Ava- | |
| liação | 95 |

| | |
|---|-----|
| 1.1.3. Determinação do Esforço Havido. Correções | 98 |
| 1.2. Quantificação das Variáveis de Cada Pro- grama | 103 |
| 2. Análise Estatística dos Dados Levantados e Ob- tenção do Tempo Normal | 107 |
| 2.1. Análise dos Dados Brutos | 108 |
| 2.2. Análise da Correlação das Variáveis com o Esforço em Dias | 109 |
| 2.3. Análise de Regressão Propriamente Dita . | 109 |
| 3. Efetuação de Estimativas e Cálculos dos Índi- ces de Eficiência | 111 |
| 4. Um Caso Prático | 114 |
| <u>CAPÍTULO VI - CONCLUSÕES</u> | 135 |
| - Referências Bibliográficas | 142 |
| 1. Livros | 143 |
| 2. Artigos | 145 |
| - Anexo | 153 |

LISTA DE TABELAS

| | pág. |
|--|------|
| 1. Resultados obtidos com a aplicação de amostragem do trabalho | 120 |
| 2. Distribuição relativa do tempo útil dos programadores | 121 |
| 3. Características globais da amostra de 109 programas utilizados na pesquisa | 122 |
| 4. Características das variáveis de cada um dos programas atualizadores | 123 |
| 5. Características das variáveis de cada um dos programas de consistência | 124 |
| 6. Características das variáveis de cada um dos programas listadores | 125 |
| 7. Características das variáveis de cada um dos programas organizadores | 126 |
| 8. Características das variáveis de cada um dos programas formatadores | 127 |
| 9. Matriz dos coeficientes de correlação simples para o caso de programas atualizadores | 128 |
| 10. Matriz dos coeficientes de correlação simples entre as variáveis envolvidas para o caso de programas de consistência | 129 |
| 11. Matriz dos coeficientes de correlação simples entre as variáveis envolvidas para o caso de programas listadores | 130 |
| 12. Matriz dos coeficientes de correlação simples entre as variáveis envolvidas para o caso de programas organizadores | 131 |
| 13. Matriz dos coeficientes de correlação simples entre as variáveis envolvidas para o caso de programas formatadores | 132 |
| 14. Equações para obtenção dos tempos normais | 133 |
| 15. Equações para obtenção dos tempos padrão | 134 |

C A P Í T U L O I

INTRODUÇÃO

1. OBJETIVOS GERAIS

O presente trabalho tem por objetivo contribuir para o incremento dos recursos e técnicas utilizadas na administração de projetos de sistemas para computador.

A abordagem proposta para se atingir este objetivo consiste na utilização de conceitos inerentes e peculiares à Administração da Produção.

Em termos específicos, o que se visa é aplicar recursos e técnicas habituais à área de Medida do Trabalho, durante o desenvolvimento de sistemas de processamento de dados.

A hipótese é a de que é possível planejar, organizar e controlar o trabalho das pessoas envolvidas no desenvolvimento de sistemas automatizados através do uso do instrumental de Medida do Trabalho.

Raramente algo desta disciplina tem sido aplicado fora da área estritamente fabril. Mesmo as empresas industriais que exercem severo controle sobre a eficiência da sua mão de obra direta, pouco progrediram em termos de avaliar

a eficiência da mesma, quando não diretamente ligada à produção, tais como os serviços administrativos ou de escritório e quando existirem, os serviços dos profissionais de processamento de dados. No caso destes últimos não deixa de ser uma grande contradição. Já que, o processamento eletrônico de dados, que é o principal agente provocador de melhorias e incrementos de produtividade em vários setores e áreas da empresa, na maioria das vezes sofre de problemas crônicos de improdutividade. Sendo que muitas vezes nem se pode falar em produtividade ou improdutividade por absoluta falta de índices, parâmetros ou critérios para se constatar algum desses estados. Não é sem propriedade que se diz que os projetos de sistemas automatizados têm todos uma única característica comum:

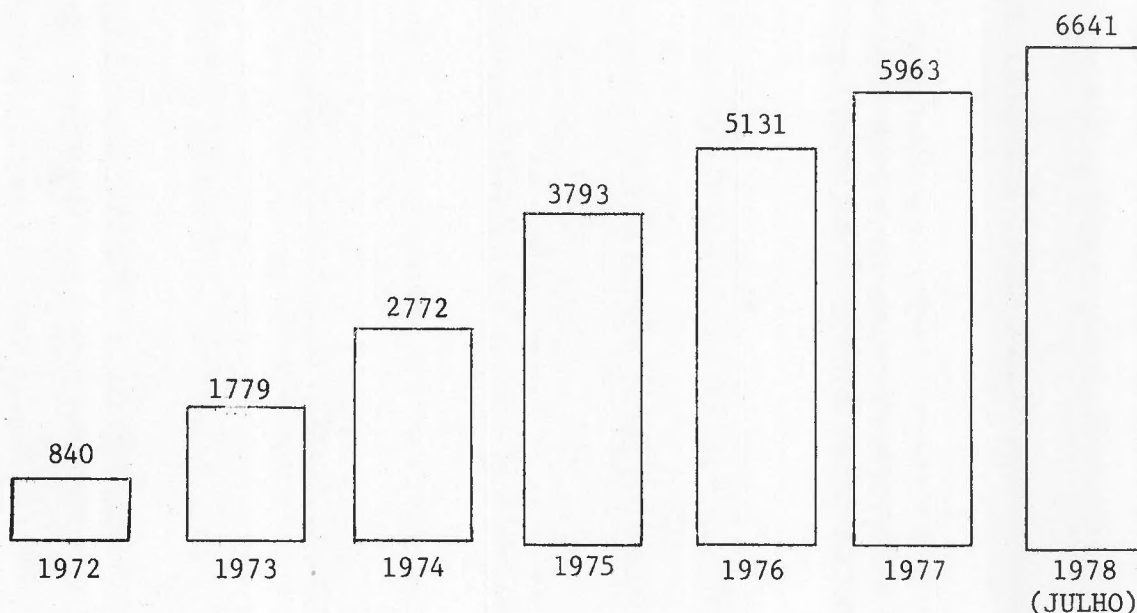
- Estão sempre atrasados.

Entretanto vários componentes da função de processamento de dados poderiam e devem ser analisados sob uma ótica de Administração da Produção. O presente trabalho concentra sua atenção em um desses componentes que é a tarefa de produzir programas para computador.

2. SITUAÇÃO DO SETOR DE PROCESSAMENTO ELETRÔNICO DE DADOS NO BRASIL

Segundo dados recentemente divulgados pela CAPRE - Comissão de Atividades de Processamento Eletrônico, até ju

lho de 1978 haviam em operação no Brasil 6.641 computadores. A evolução do parque computacional brasileiro de 1972 até esta data tem se comportado conforme mostrado no quadro abaixo.



Em 1976 e em 1977 os gastos com o setor de processamento de dados no Brasil representaram aproximadamente 1% do PIB, sendo que a estimativa do total de dispendio em 1978 atinge cerca de US\$ 2 bilhões, o que deve elevar um pouco aquela porcentagem. Em outros países a proporção entre gastos com processamento de dados e PIB apresenta as seguintes conotações:

| | | |
|-----------------|---|--------|
| ITÁLIA | : | 2% |
| INGLATERRA | : | 3% |
| ESTADOS UNIDOS: | | 4% (1) |

¹ Carlos A. A. Oliveira, "Recursos Brasileiros em Computação: Valor de Mercado", Dados e Idéias, vol. 3, Nº 3: 63-73, Janeiro 78.

A partir das estatísticas disponíveis o que se conclui é que o setor de processamento de dados está em evolu-
ção no Brasil, nada indicando haverem perspectivas de re-
troação.

Para o presente trabalho o que há de mais relevan-
te dentro de todas as estatísticas, é que do total gasto com
o setor, ano a ano tem crescido a porcentagem do componente
mão de obra, o qual se situa atualmente em torno de 60% do
total de gastos, tendo sido de 55% em 1975 e de 58,5% em
1976 (FONTE: CAPRE). Isto significa que investir em racio-
nalização e otimização de equipamentos e intalações, hoje em
dia, pode não ser mais eficiente do que investir em melho-
rias na alocação e utilização dos recursos humanos disponí-
veis.

E como se distribuem esses recursos humanos?

Ainda segundo a CAPRE, em 1977, mais de cem mil
profissionais atuavam diretamente ligados com processamento
de dados no Brasil, distribuídos da seguinte forma:

| <u>CATEGORIA</u> | <u>Nº DE PROFISSIONAIS</u> | <u>% S/TOTAL</u> |
|------------------|----------------------------|------------------|
| Preparadores | 24.339 | 24,0 |
| Digitadores | 37.522 | 37,0 |
| Operadores | 16.002 | 15,8 |
| Programadores | 12.190 | 12,0 |
| Analistas | <u>11.359</u> | <u>11,2</u> |
| TOTAL | 101.412 | 100,0 |

Baseando-se em dados informados pela SUCESU - SP

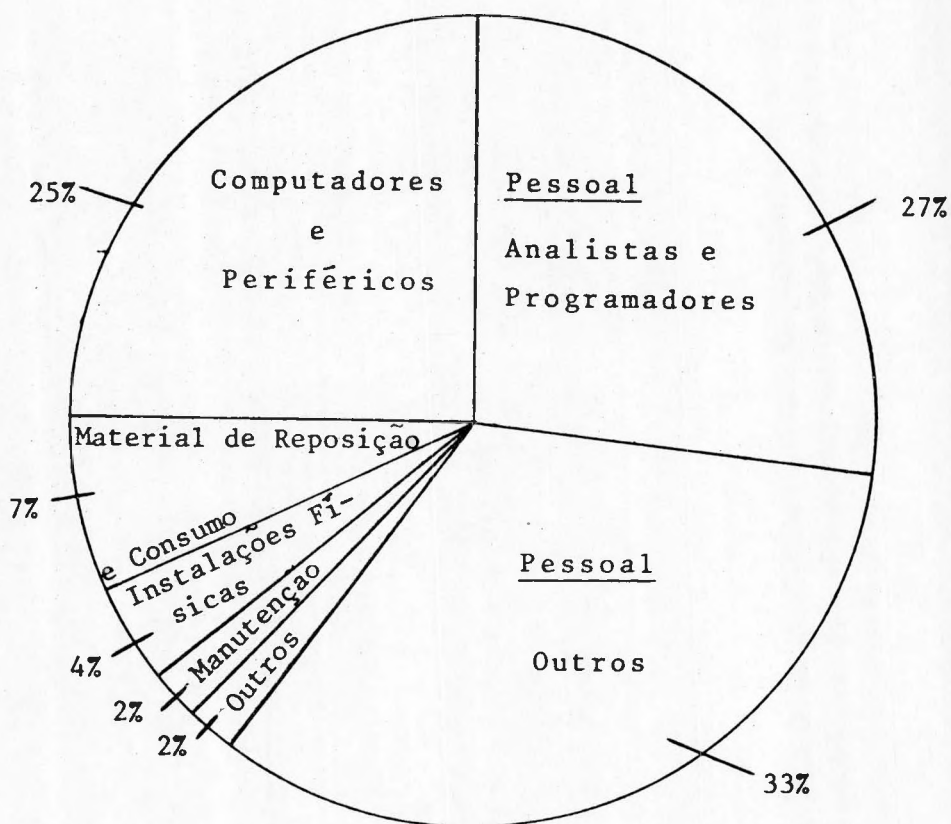
(Sociedade dos Usuários de Computadores e Equipamentos Subsidiários de S.Paulo) sobre o salário médio de cada uma dessas modalidades de profissionais, chega-se a conclusão de que apesar de programadores e analistas representarem apenas 23,2% daquele total de pessoas, em termos de salários e encargos pagos eles são responsáveis por 46% do total.

Ou seja, quase metade do componente mão de obra no setor de processamento de dados no Brasil é devido a gastos com programadores e analistas de sistemas. E a taxa média de crescimento do número de pessoas empregadas nessas funções é de cerca de 12% ao ano (2).

Tudo isso tem o objetivo de demonstrar a importância de se passar a estudar de uma forma mais científica e menos improvisada a área de desenvolvimento de sistemas de computador, já que esta só em termos de salários e encargos representa mais de um quarto de tudo que é gasto com processamento de dados no Brasil atualmente, representando aproximadamente 0,4% do nosso PIB.

O gráfico abaixo foi construído com base em dados divulgados pelos órgãos anteriormente mencionados nesta seção, constituindo-se em um resumo de tudo o que foi dito.

²Paulo Corchaki e Luiz A. P. Souza, "Estudo de desequilíbrio entre demanda e oferta", DATANews, pág. 17, 15/11/78.

PRINCIPAIS ITENS DE DISPÊNDIO EM PROC. DADOS NO BRASILVALOR DE MERCADO EM 78 : US\$ 2 BI3. DESCRIÇÃO DO PROBLEMA

Apesar do enorme crescimento da área de processamento eletrônico de dados e do alto grau de desenvolvimento tecnológico atingido pela mesma, o gerenciamento de projetos de sistemas para computador continua sendo feito praticamente da mesma forma como o vinha sendo há cerca de 20 anos. Muito pouco tem sido feito no sentido de sistematizar e organizar o conhecimento existente na área. Com a expan-

são das necessidades de serviços de processamento de dados muitos problemas tem vindo à tona.

Entre os problemas que mais afligem os gerentes de centros de processamento de dados, existem dois que ocupam a maior relevância:

1. A sua incapacidade em determinar a mão de obra necessária para desenvolver sistemas automatizados, tanto no que se refere à análise dos sistemas como na confecção de programas para os computadores. Evidentemente, estimativas criteriosas são essenciais não apenas para se efetuar melhores alocações dos recursos mas sobretudo para que se possa avaliar a viabilidade econômica dos programas e sistemas.
2. A sua incapacidade em premiar e/ou penalizar os seus funcionários de acordo com a produtividade dos mesmos. Este segundo problema é uma consequência do primeiro, já que não se pode avaliar se algo foi bem feito quando não se tem meios de dizer o que vem a ser "bem feito".

Em outras palavras, se não se sabe estimar com precisão quanto tempo uma tarefa deve demandar, então, menos ainda, quando a mesma estiver concluída, se saberá dizer se ela utilizou muitos ou poucos recursos, ou então com que grau de eficiência foi concluída.

Essas considerações conduzem ao problema básico a que se pretende responder com a presente dissertação:

- Qual o esforço necessário para se desenvolver um programa

de computador, após conhecer-se as características básicas que o mesmo deverá atender?

As principais agravantes para este problema decorrem do fato de que a circunstância de se dispor das características básicas ou mesmo de especificações precisas acerca do programa que deverá ser desenvolvido não caracterizam de imediato a quantidade de trabalho que deverá ser feito.

Dessa forma o que acontece, praticamente na totalidade das vezes, é que as estimativas são feitas com base unicamente na intuição e experiência pessoal dos supervisores, gerentes ou analistas. E, mesmo introduzindo-se altíssimos coeficientes de segurança, que de outra forma seriam dispensáveis, essas estimativas costumam falhar. E os projetos invariavelmente acabam por requerer mais recursos além do que originalmente havia sido estimado. E por que essas falhas?

Principalmente pelos seguintes motivos:

- Dificuldade em se definir exatamente o que deve ser feito.
- Muitos fatores envolvidos difíceis de serem quantificados, principalmente, sem se saber quais são os mais relevantes.
- Falta de dados históricos coletados de forma sistemática e organizada para servir de base para futuras interpretações.

Mas, o fato que talvez mais contribua para dificultar a atividade dos gerentes responsáveis pelos desenvol

vimentos dos sistemas, seja a falta de artigos e publicações que os possam orientar sobre o assunto. Isso será abor dado na próxima seção.

4. ANÁLISE DO CORPO DE CONHECIMENTOS

Entre as tentativas de explicações para a falta de trabalhos mais profundos sobre o gerenciamento do desenvolvimento de sistemas para computador, uma das mais razoáveis é a que une fatores tecnológicos e históricos.

Assim sendo, essa falta dever-se-ia ao fato de que nos primórdios do processamento eletrônico de dados os gastos com pessoal representavam uma parcela ínfima perto dos astronômicos custos dos computadores e seus periféricos; tal situação contribuía para que a atenção e atuação dos pesquisadores e cientistas da área se voltasse para essa parte de equipamentos (hardware). Tal fato provocou um volume imenso de livros, artigos e publicações na área. Como consequência dessa atenção toda, os custos de hardware começaram a baixar consideravelmente. Nessa altura, evidentemente, os custos de software e de pessoal passaram a se tornar significativos e no estágio atual eles representam, por exemplo, no Brasil 60% dos gastos totais com processamento de dados. Por isso é esperado que para os próximos anos, com a forçosa mudança de ênfase que se faz necessária, comecem a surgir um maior número de trabalhos analisando as várias funções dos profissionais que atuam em processamento de dados.

De qualquer forma atualmente é muito escasso o número de livros ou artigos em revistas especializadas, abordando o tema da medida do trabalho na área de processamento de dados, mais especificamente em programação de computadores.

Entre os livros, os principais autores são Brandon (3) e (4), Szewda (5) e Brooks (6).

Brandon é considerado, praticamente por todos que adentraram a área, o maior especialista no assunto, tendo escrito, entre outros, dois livros muito bem conceituados: Management Planning for Data Processing e Project Control Standars. Estes livros antes de tudo, pretendem servir de guias para o desenvolvimento de projetos de sistemas de computador e nesse particular aspecto eles apresentam uma metodologia bem detalhada. No entanto, no que se refere ao aspecto prático da avaliação do trabalho de analistas e programadores, os livros já não oferecem uma metodologia a ser aplicada, limitando-se a fornecer dados prontos (standars) nos quais os gerentes de projetos devem se basear para efetuar estimativas e medir produtividade. Praticamente esses dados não apresentam grande utilidade pelas seguintes razões: o

³ Dick Brandon, Management Standars for Data Processing, New York, Van Nostrand Reinhold Company, 1963.

⁴ Dick Brandon, Project Control Standars, Philadelphia, Auerbach, 1970

⁵ Ralph Szewda, Information Processing Management, Princeton, Auerbach Publishers, 1972.

⁶ Fred Brooks, The Mythical Man Month: Essays on Software Engineering, Mass. U.S.A., Addison - Wesley, 1975.

primeiro dos livros citados, que é o que apresenta o maior número de standars, é muito antigo, de 1962, e a maiorias dos conceitos e recomendações estão superados. Mas o principal problema, é que, e isso não é enfatizado, estes standards, não podem ser generalizados e ter utilização universal, já que as condições específicas variam enormemente de um Centro de Processamento de Dados para outro. Estes fatos foram levados em conta no segundo livro (Project Control Standars) o qual passa a dar uma ênfase maior aos fatores ambientais. Neste último, todavia, apenas dois capítulos são dedicados ao assunto em questão, capítulo 9 - Time Estimating and Project Planning, e capítulo 10 - Personnel Assignment, funcionando, principalmente, como um guia para efetuação de estimativas, à exemplo do primeiro livro.

Esta última função é também a ênfase principal em "Information Processing Management", de Ralph Szweda. Este não chega a apresentar uma metodologia para o estabelecimento de padrões, mas sim uma sistemática para se efetuar estimativas que, por ser de aplicação bem ampla, encontrou vários adeptos, inclusive em centros de processamento de dados, no Brasil. A amplitude é devida ao fato de vários tipos de fatores terem sido considerados, tendo o autor descido a um nível de detalhe razoável, permitindo servir como diretriz para efetuação de estimativas, principalmente na área de análise de sistemas. A exemplo dos livros de Brandon, este não é todo dedicado ao assunto Medida do Trabalho, consignando para este 2 entre 12 capítulos. A principal críti

tica que mesmo os adeptos de Szweda apontam, é a necessidade que ele impõe de as tarefas sempre deverem ser caracterizadas em simples, medianas e complexas, o que acaba impondo um alto grau de subjetividade à sua sistemática, já que é comum acontecer de a mesma tarefa ser considerada complexa para um avaliador e simples para outro.

Finalmente, o último livro acima mencionado, "The Mythical Man-Month: - Essays on Software Engineering", de Frederick Brooks Jr., é um livro dedicado principalmente a descrever a experiência pessoal do autor na administração de projetos de sistemas de computador e aos erros básicos que nesses são cometidos do ponto de vista gerencial. Realça, principalmente, que a abordagem na administração de projetos pequenos é completamente diferente da de projetos grandes e que justamente a maioria dos erros que são cometidos, devem-se ao fato de se pretender extrapolar experiências adquiridas em projetos pequenos para projetos grandes e vice-versa. Este livro, apesar de não ter a pretensão de pertencer ou adentrar à área de Medida do Trabalho, é aqui mencionado como um dos mais importantes, pelas recomendações objetivas que ele faz no sentido de alertar sobre os erros mais comuns que são cometidos ao se efetuar estimativas sobre duração de projetos.

Com relação à artigos em revistas, é enorme a predominância em quantidade e qualidade de artigos saídos na publicação norte-americana "Datamation". No mesmo nível de boa qualidade, mas com uma menor taxa de artigos relacionados

com Medida do Trabalho, está "IBM Systems Journal".

Artigos muito interessantes, principalmente versando sobre aplicações de Medida de Trabalho à atividade de engenheiros, foram encontrados em "Industrial Engineering" e estão todos enumerados na bibliografia.

Praticamente a totalidade desses artigos podem ser enquadradas em 4 categorias, conforme o tema predominante em cada um:

- I - ARTIGOS VOLTADOS PARA A REALIZAÇÃO DE ESTIMATIVAS E ANÁLISES DE PRODUTIVIDADE DE PROJETOS COMO UM TODO, dos quais os principais são os de Walston & Felix (7) e o de Peeples (8).

Esta categoria de artigos não enfatiza o aspecto programação de computadores, a qual é vista como mais uma fase dentre as várias fases do projeto todo. Para este sim são definidos macro-índices o que ocorre, principalmente, no caso do artigo de Walston e Felix. Neste, inclusive, é utilizada análise de regressão para verificar como 29 variáveis por eles definidas, podem afetar a produtividade de um projeto de sistema automatizado.

- II - ARTIGOS VOLTADOS PARA ANÁLISE DAS TAREFAS DE PROGRAMADORES, SOB O PONTO DE VISTA DE MEDIDA DO TRABALHO, des

⁷C. Felix e C. Walston, "A Method of Programming Measurement and Estimation", IBM Systems Journal, vol. 16, Nº 1: 54-73, 1977.

⁸Donald Peeples, "Measure for Productivity", Datamation, vol. 24, Nº 5: 222-230, Maio 78.

tacando-se os artigos de Shell (9), Scott (10) e Donelson (11). Estes são artigos que entraram mais a fundo nos aspectos das tarefas e do trabalho em si de programadores de computador.

O artigo de Shell tem como principal colocação, que a duração do desenvolvimento de um programa é função do respectivo tamanho do diagrama de blocos, e que a partir deste é possível efetuar estimativas de durações para o desenvolvimento do programa. Muito discutível, já que o diagrama de blocos que antecede o desenvolvimento do programa, e é normalmente feito pelo analista responsável pela descrição do mesmo, varia extremamente de tamanho dependendo do grau de detalhe que este último queira (ou saiba) dar.

Scott utiliza a técnica Delphi na avaliação do desenvolvimento de programas. O artigo é completo, pois além de caracterizar variáveis que, segundo o autor, interferem no trabalho, também descreve passo a passo a metodologia a ser aplicada, no caso, a técnica Delphi. O único porém, é a grande carga de conteúdo subjetivo que esta técnica encerra, pois as avaliações são feitas baseando-se principalmente na experiência

⁹Richard Shell, "Work Measurement for Computer Programming Operations", Industrial Engineering, vol. 4, nº 10 : 32-36, Outubro 72.

¹⁰Randall Scott, "Programmer Productivity and the Delphi Technique", Datamation, vol. 20, nº 5 : 71-73, Maio de 74.

¹¹William Donelson, "Project Planning and Control", Datamation, vol. 22, nº 6 : 73-80, Junho 76.

pessoal de um grupo de pessoas.

Finalmente o artigo de Donelson, o qual não analisa em detalhes somente o trabalho de programação, mas também faz considerações sobre o projeto como um todo. Este artigo tem como aspecto mais interessante, o enquadramento que é feito dos programas em 12 categorias, inferindo-se que a complexidade de um programa é de uma certa forma dependente da categoria em que o mesmo foi enquadrado. A metodologia que será exposta no capítulo V desta dissertação, também se baseou nesta conceituação, apesar de ter-se afeito a um número menor de categorias.

III - ARTIGOS QUE ANALISAM O TRABALHO DE PROGRAMADORES SOB O PONTO DE VISTA DE NÚMERO DE LINHAS DE CODIFICAÇÃO, principalmente os de Johnson (12) e Jones (13).

Esta é, a categoria que possui o maior número de artigos; alguns defendendo, outros criticando, o critério de aferir-se a produtividade de programadores a partir do número de linhas de codificação (e consequentemente, o número de cartões) que eles produzirem. Os entusiastas do critério afirmam que a complexidade de

¹²James Johnson, "A Working Measure of Productivity", Datamation, vol. 23, nº 2 : 106-110, Fevereiro de 77.

¹³Jones, T.C., "Measuring Programming Quality and Productivity", IBM Systems Journal, vol. 17, nº 1, 1978.

um programa é função do número de linhas de codificação escritas para o mesmo. Nesse grupo, o principal autor é Johnson; que, entretanto, coloca que a taxa

Nº de linha de codificação

homens / dia

é extremamente variável, sendo maior quanto mais simples e menos complexo for o sistema ou programa. Dentre os que atacam o critério, se destaca Jones, que pondera três problemas principais:

- dificuldade para se estabelecer exatamente o que é uma "linha de codificação".
- existência de muitas atividades dos programadores que não tem nenhuma relação com linhas de codificação.
- tendência do critério em penalizar linguagens de programação de alto nível como Cobol e PL-1, já que com estas linguagens os programas resultam maiores que, por exemplo, se estivessem em Assembler.

Esta técnica, todavia, possui um grande inconveniente, pois ela se limita a permitir análises de produtividade não sendo possível com a sua utilização efetuar-se estimativas acerca da duração do desenvolvimento dos programas. Ou em outras palavras, ela só po

de ser utilizada após a conclusão das tarefas.

IV - ARTIGOS QUE ANALISAM A APLICAÇÃO DE MEDIDA DO TRABALHO EM ATIVIDADES SEMELHANTES ÀS DE PROGRAMAÇÃO DE COMPUTADORES, realçando-se os de Van Kirk (14) e Smerilson (15). Estes dois artigos foram publicados na "Industrial Engineering" e propõem maneiras de se efetuar estimativas acerca das durações de tarefas peculiares a engenheiros.

A essência do artigo de Smerilson forneceu subsídios para a presente dissertação, principalmente pela utilização de análise de regressão para a obtenção de equações, através das quais são efetuadas estimativas de duração de tarefas. Sendo que, evidentemente, as variáveis independentes dessas equações são aquelas que afetam diretamente a tarefa analisada.

Finalmente, além de livros e artigos em revistas especializadas, foram pesquisadas teses de doutoramento e dissertação de mestrado.

Apesar do grande número de pesquisas encontradas sobre Medida do Trabalho, todas elas se voltavam à ati

¹⁴ Van Kirk, Richard, "Work Standars for Highway Engineers", Industrial Engineering, vol. 5, nº 1 : 34-39, Janeiro 73.

¹⁵ Smerilson, Harvey H., "Standars for Engineers", Industrial Engineering, vol. 7, nº 10 : 12-17, Outubro de 75.

vidades puramente fabris, fugindo do escopo desta dissertação. Por outro lado os trabalhos que analisavam funções em centro de processamento de dados não as enfocavam sob o ponto de vista de Medida do Trabalho. Desta forma não foi possível utilizar nenhuma destas teses ou dissertações.

5. RESUMO

A presente dissertação está dividida em 6 (seis) capítulos, um dos quais é esta própria introdução, com a qual se pretendeu, além de mostrar a importância do problema que será analisado, também realçar o seu papel no panorama de Processamento de Dados brasileiro de hoje.

O capítulo seguinte apresenta o trabalho alvo deste estudo, ou seja, programação de computadores, situando-o em relação ao todo da função de processamento de dados na empresa e caracterizando-se também os aspectos organizacionais dessa função além de realçar-se os aspectos básicos do ciclo de vida dos sistemas automatizados. Em seguida, no terceiro capítulo, são enumerados e analisados os vários fatores que atuam sobre a tarefa de programação de computadores os quais encontram-se separados em três categorias; Fatores Ambientais, Variáveis Inerentes a Cada Programa e Experiência do Programador. No quarto capítulo são consideradas várias técnicas de Medida do Trabalho, examinando-se cada uma

sob a ótica de uma possível aplicação com programadores de computador. A metodologia proposta para se estabelecer padrões de tempo e produtividade para esses profissionais está apresentada no quinto capítulo. Resumidamente ela consiste em se escolher e observar um grupo de programas de computador, quantificando simultaneamente suas variáveis peculiares conforme conceituadas no terceiro capítulo e o esforço dispendido em cada um. Através do uso da análise de regressão múltipla são então obtidas as equações que darão os tempos normais necessários para se produzir cada tipo de programa em função das variáveis relevantes em cada tipo.

Finalmente no último capítulo são mostrados os principais benefícios obtidos com a aplicação da metodologia sugerindo-se possibilidades para se expandir o estudo.

C A P Í T U L O I I

CARACTERÍSTICAS DO TRABALHO ANALISADO

1. A FUNÇÃO DO PROCESSAMENTO ELETRÔNICO DE DADOS

As várias funções internas da empresa têm cada uma as suas características próprias, e agem segundo seus objetivos próprios. Assim sendo, os objetivos próprios do setor de vendas de uma empresa são de natureza diversa dos objetivos do setor de produção, ou ainda dos da secção de pessoal. Além destes objetivos específicos existem também os objetivos gerais da empresa como um todo. A função da administração é coordenar os vários objetivos específicos, no sentido de que a empresa caminhe integrada e coesa na direção de seus objetivos gerais. Para cumprir essa sua função, a administração necessita de dispositivos de controle, para que se possa saber, por exemplo, quando um determinado setor não está contribuindo de forma correta para o cumprimento do objetivo geral da empresa. A cada setor deve ser mostrado o caminho correto, observando-se a sua atividade para poder corrigir-lhe os desvios. É nessa hora que surge a justificativa para a função de processamento de dados dentro da empresa: o computador é o único recurso que viabiliza um controle integrado geral nas empresas de porte, possuindo a capa-

cidade de ordenar, classificar e sintetizar grandes quantidades de dados com uma rapidez que torna o inviável em algo perfeitamente acessível.

Para se conseguir esse proveito do computador é forçoso que existam processos pelos quais o homem com ele se comunique e o comande. Ou seja, é necessário que se transmita à máquina as ordens que definam o processamento que se deseja efetuar. Essas ordens dizem à máquina como trabalhar com os dados, e devem portanto ser fornecidas de acordo com padrões pré-estabelecidos, de modo a se tornarem compreensíveis para os circuitos do computador. Estes circuitos contêm uma espécie de dicionário além de algo como um conjunto de regras gramaticais. Qualquer ordem que não atender a estas regras ou que não seja localizada no dicionário não será nem aceita e nem reconhecida. As etapas que ocorrem antes que se possa transmitir à máquina as ordens ou instruções constituem o que comumente se chama de análise de sistemas e programação de computadores.

2. DESENVOLVIMENTO DE SISTEMAS PARA COMPUTADOR. CICLO DE VIDA DE SISTEMAS.

Na verdade as funções de análise de sistemas e programação de computadores fazem parte de algo mais amplo que se convencionou chamar ciclo de vida de sistemas. Pois assim como outros tipos de sistemas usualmente conhecidos tais como, sistemas telefônicos, sistemas de transmissão de energia elétrica, sistemas de construção civil, etc., os sistemas de pro

cessamento de dados também tem que ser projetados e construídos para que possam ser operados pelo homem. Em resumo, antes que a empresa possa dispor da solução para efetuar seu processamento de dados, um sistema automatizado que abrangerá um ou vários programas de computador deverá ser concebido e desenvolvido.

E o que é um sistema de computador?

Uma pergunta muito comumente feita pelos usuários que comprem ou encomendam sistemas automatizados e que na maioria das vezes é respondida subjetivamente com uma das seguintes formas:

- É a solução dos seus problemas administrativos
- É uma nova forma de você passar a trabalhar
- Etc.

Na verdade, quando usuários comprem sistemas automatizados, eles deveriam proceder exatamente como quando compram outras mercadorias, mecanismos ou máquinas, inclusive efetuando as usuais conferências contra as "listas de materiais". E o que deve constar na lista de materiais de um sistema de computador?

Basicamente os seguintes itens, segundo Donelson (16).

1. Um número determinado de programas de computador que efetuem recepção e consistência de dados, que imprimam relató-

¹⁶William Donelson, "Project Planning and Control", Datamation, vol. 22, Nº 6 : 73-80, Junho 76.

rios, que efetuem cálculos, etc..

2. Ferramentas para implantar e operar o sistema, como programas de "job control", criação de bancos de dados, utilitários que façam conversão de arquivos, etc..
3. Instruções para uso, que incluem o manual do usuário, descrevendo passo a passo o que deve fazer o comprador (usuário) para ser bem sucedido com seu novo sistema, ou seja, como ele deve preencher os formulários de entrada e entregá-los, o que significam as mensagens impressas nos relatórios, etc..
4. Guia de Características Técnicas, que servirão para os analistas e programadores que efetuarão as prováveis modificações necessárias no futuro. Consiste dos vários manuals de I/O, processamento, operação, etc.. Servem também para auxiliar o pessoal de operação do computador a compreender melhor o sistema.
5. Treinamento, que habilita todas as pessoas envolvidas a usufruírem melhor do novo sistema.

Para se saber como essa lista de materiais pode ser fabricada e tornada disponível aos usuários, é necessário se conhecer as várias fases que compõem o desenvolvimento de um sistema automatizado. Isto será explicado através do ciclo de vida dos sistemas.

A figura abaixo mostra de uma forma sintética as várias fases do ciclo de vida de um sistema automatizado de processamento de dados, desde a hora em que ele é concebido

até o momento em que ele passa a ser operado e utilizado pela empresa.



Em seguida é feita uma descrição sucinta de cada uma dessas fases:

2.1. Fase I - Concepção do Sistema: Esta é a fase na qual surge a idéia de se desenvolver um sistema automatizado, o qual poderá ser destinado a substituir atividades que venham sendo executadas de forma manual, como poderá servir para produzir novos relatórios de informação gerencial, ou então ter alguma outra finalidade diferente. O importante é que essa é uma fase típica de criação na qual os departamentos ou filiais, autores da idéia discutem com o pessoal de Processamento de Dados (P.D.), as características gerais do futuro sistema. O pessoal de P.D. participa dessa fase muito mais em caráter de assessoria, ajudando os autores da idéia (ou usuários) a definir de forma mais precisa o futuro sistema. O documento básico que caracteriza esta fase é

um estudo de viabilidade econômica, mostrando as vantagens e desvantagens da idéia. Nessa altura é praticamente impossível se conseguir uma idéia precisa de quanto tempo haverá para chegar à data de implantação ou mesmo do esforço em homens/mês, que será necessário. Todavia estimativas globais devem ser feitas, sem as quais seria impossível a realização do estudo de viabilidade econômica. O profissional que caracteriza essa fase além do usuário é o analista de concepção, cuja função no Brasil, em geral, é desempenhada pelo analista de organização e métodos. Atualmente, vem se tornando frequente a participação de engenheiros de produção nesta etapa.

2.2 Fase II - Especificações Funcionais: Esta fase é de responsabilidade do Departamento de Processamento de Dados, embora ainda haja uma grande convivência com a equipe de usuários. Principalmente porque essa é uma fase de levantamento de dados e de análise de alternativas, sendo a ocasião em que os analistas saem a campo para conhecer como funcionam os departamentos envolvidos ou setores que serão afetados pelo novo sistema automatizado. Nesta fase devem ser definidas e especificadas todas as funções que o novo sistema abrangerá. Evidentemente, nem todas as funções serão executadas pelo computador; algumas delas serão executadas manualmente e para estas, procedimentos detalhados deverão ser definidos. Também os formatos (lay-outs) dos relatórios (outputs) bem como os formatos dos documentos de entrada

(inputs) devem estar completamente acertados entre analis-
tas e usuários. O principal documento produzido nesta fase,
deve, além de enumerar as várias funções também, conter os
objetivos à que se propõe o novo sistema, para que este te-
nha sua performance avaliada quando da implantação. É conve-
niente que um novo estudo de viabilidade econômica seja le-
vado à cabo, pois apesar de ainda não se poder contar comes-
timativas precisas acerca do prazo a decorrer, já se possui
uma idéia bem mais precisa acerca do sistema, do que aquela
que havia na Fase I. O profissional que caracteriza esta fa-
se é o analista de organização e métodos apesar de já haver
um envolvimento inicial de analistas de sistemas com conhe-
cimento de funcionamento de computador.

2.3. Fase III - Projeto Detalhado: Definidas as funções
do novo sistema surge a necessidade de se especificar como
elas serão realizadas.

Isso significa decidir:

- a - como e quais informações serão armazenadas.
- b - que equipamento será utilizado bem como qual será a for-
ma de processamento (real-time, batch, etc.).
- c - qual a forma de organização dos arquivos (sequenciais,
index, diretos, etc.).
- d - quantos programas serão necessários e o que cada um fa-
rá (descrição de programas).
- e - como os programas e arquivos se interrelacionam (fluxo-

grama do sistema).

f - que controles possuirá o sistema.

g - todas as informações para que o sistema possa vir a ser operado conforme os objetivos estabelecidos na Fase II.

Após essa fase no ciclo de vida dos sistemas, toda e qualquer necessidade de alteração passa a se tornar demasiadamente onerosa. Isto significa que nesta fase o usuário deve estar suficientemente envolvido para que se possa ter certeza de que essas necessidades de alterações não acontecerão.

O principal documento produzido nesta fase é a denominada "pasta do programa", existindo uma pasta para cada programa cuja necessidade foi detectada anteriormente. Esta pasta contém uma descrição pormenorizada dos objetivos do programa além de descrever as funções que o mesmo deve executar, como por exemplo, cálculos, verificações de validade de valores, impressão de relatórios, etc.

Esta descrição, idealmente, deve ser feita em um nível tal que dispense o contato pessoal entre o elemento que descreveu o programa (o analista) e aquele que será responsável por levá-lo a cabo (o programador). Além das descrições as pastas contém também os arquivos que são lidos e gravados pelo programa, bem como lay-outs detalhados de cada relatório que deverá ser impresso. Com relação aos arquivos, várias informações devem ser fornecidas, tais como:

a - Formatos de cada tipo diferente de registro existente no

arquivo;

- b - dispositivo onde o arquivo é lido ou gravado, ou seja, se se trata de fita magnética, disco, cartão, ou outros meios;
- c - dados quantitativos, tais como, quantidade de registros que cada arquivo deverá conter quando o sistema estiver operacional, volatilidade dos dados, etc..

É somente ao final da Fase III, que estimativas precisas acerca do esforço ainda necessário e do prazo a decorrrer até a implantação já podem ser feitos. Como faze-las é um dos temas que será predominantemente enfatizado nos próximos capítulos deste trabalho. O profissional que caracteriza a Fase III é o analista de sistemas com conhecimentos de processamentos de dados.

2.4. Fase IV - Programação: Programar é a tarefa de converter o que o analista de sistemas especificou na "pasta do programa", em uma linguagem que o computador possa entender. O profissional que caracteriza essa fase é o programador. A atividade de programação é constituída de 5 etapas a seguir descritas:

1. Estudo: Nessa altura o programador deve ler e entender em detalhes o conteúdo da "pasta do programa". É fundamental que ele entre nos mínimos detalhes, prevendo todas as possibilidades e tudo o que deve ser feito e em que sequência. Normalmente nessa altura é traçada uma representa

tação gráfica, mesmo que isso já tenha sido feito pelo analista de sistemas, a qual se convencionou chamar de diagrama de blocos e que contém o esquema geral sobre como deverá funcionar o programa. Quanto mais subdividido em detalhes estiver este diagrama de blocos, mais facilidade terá o programador para realizar a etapa seguinte.

2. Codificação: Definida na etapa anterior a lógica detalhada do programa, pode se passar a tarefa de converter a mesma em uma sequência de comandos (instruções) em uma linguagem que o computador possa entender. Isso significa que uma série de normas e regras, tais como numa gramática, devem ser obedecidas pelo programador ao escrever os comandos, ou melhor ao codificar o programa.
3. Depuração: Por melhor e mais habilitado que seja o programador, dificilmente ele deixará de incorrer em erros básicos que tornarão o programa ininteligível ao computador. São os chamados erros de sintaxe e formato. Para descobri-los o programa deve ser submetido ao computador que o lerá e apontará todas as incorreções possivelmente existentes.

O programador as corrigirá, e submeterá novamente ao computador até que o programa se encontre no estado que se convencionou chamar de "zero erro".

4. Teste: Todo programa deve ser testado para ser assegurado que funcionará de acordo com o que o analista previu na descrição do programa. Pois o simples fato de a lógica

ca e a codificação estarem perfeitos não significa que o programa funcionará conforme esperado. Casos para testes devem ser desenvolvidos, cabendo ao programador criar os dados de teste para que o programa os leia, verificando então o programador se estão sendo produzidos os resultados esperados. O programa só pode ser considerado correto quando processou sob todas as condições os dados de teste e produziu os resultados esperados.

5. Documentação: Nesta etapa, o programador deve deixar escrito tudo o que foi feito, anexando diagramas de bloco, listagens de programa, modelos dos relatórios impressos, etc.. Deve se preocupar principalmente com as instruções que devem ser dadas aos operadores do computador, quando forem executar o seu programa. Um dos objetivos principais da documentação é o de poder servir como base para futuras revisões e/ou alterações que porventura venham a se tornar necessárias.

Estas são, assim, as 5 etapas que constituem o trabalho de programação. Considerando um programador que tenha a seu dispor um computador para efetuar depurações e testes no momento que lhe convier, pode ser dito que, em média, cada etapa consome do programador as seguintes porcentagens de tempo:

Estudo - 15 a 25%

Codificação - 20 a 30%

Depuração - 10 a 20%

Testes - 25 a 40%

Documentação - 5 a 15%

2.5. Fase V - Implantação: Na fase anterior, ficaram prontos os vários componentes do sistema, tendo cada um deles sido testado, individualmente. Agora todos devem ser testados conjuntamente para que se possa perceber possíveis dissintonias entre programas. Além dos programas, os setores da empresa que se relacionam com o novo sistema também devem ser testados no sentido de se aferir se estão aptos a conviver com o mesmo, ou seja, se sabem preparar os inputs e interpretar os outputs. Além do teste geral do sistema, outra atividade importante que ocorre na implantação é a conversão, que nada mais é que a passagem dos dados existentes, sejam quais forem as formas em que estiverem armazenadas, para o novo sistema, onde passarão a figurar nos novos arquivos. Após o teste geral e a conversão, o novo sistema já pode ser implantado e tornar-se operacional. Na fase de implantação, há necessidade da atuação de todos os profissionais que atuaram nas fases anteriores, e, principalmente, da atuação intensa dos usuários do novo sistema.

2.6. Fase VI - Operação: A operação do sistema, ao contrário das fases anteriores, deixa de ser uma tarefa de desenvolvimento para se constituir numa tarefa de produção. Mais ou menos como se fosse uma linha de montagem, o sistema recebe dados e produz relatórios. Analistas e programadores só voltarão a atuar no sistema caso surja alguma neces-

cidade de correção ou de alteração no sistema.

3. ORGANIZAÇÃO DA FUNÇÃO DE PROCESSAMENTO DE DADOS

Embora esse seja um assunto muito vasto para poder ser aqui tratado resumidamente, o tema da organização de um Centro de Processamento de Dados (C.P.D.) é aqui colocado principalmente, com o intuito de situar e caracterizar os diversos tipos de funções existentes dentro de um CPD.

Evidentemente, a organização de um CPD dependerá, primordialmente, dos objetivos do mesmo. Assim um bureau de serviços de P.D. terá uma estrutura diferente de uma Divisão de Sistemas e Automação de um banco, ou de um CPD de uma indústria. Todavia existem muitas características em comum. Por exemplo, entre os objetivos de qualquer um deles, sempre existem dois aspectos básicos:

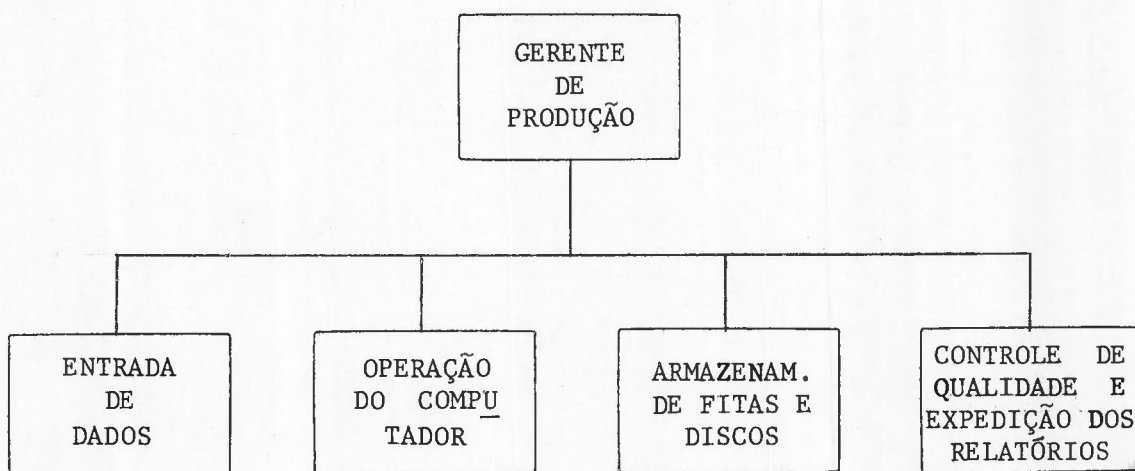
- Desenvolver novos sistemas automatizados e efetuar manutenções nos já existentes.
- Processar dados, ou seja, executar os sistemas já existentes.

Em torno desses dois objetivos, 3 grupos são sempre formados:

- Um que agrupa programadores e analistas e que normalmente tem alguma denominação semelhante a Departamento de Desenvolvimento de Sistemas.
- Outro que agrupa operadores de cartão, fitotecários, con-

feridores de dados, etc., e que normalmente tem uma denominação tal como Departamento de Produção ou Departamento de Processamento.

Relacionado com o ciclo de vida de sistemas exposto anteriormente, o primeiro grupo atua desde a Fase I até a Fase V, inclusive. Tendo o segundo grupo a incumbência total pela Fase VI, ou seja, a operação do sistema. O trabalho deste último grupo muito se assemelha ao trabalho fabril, sendo as atribuições de um gerente de produção semelhantes às atribuições de um superintendente de fábrica. O organograma típico de um departamento de produção de um CPD é representado pelo gerente, tendo abaixo dele, respondendo em geral diretamente, os vários setores produtivos. Um exemplo típico é mostrado na figura abaixo.

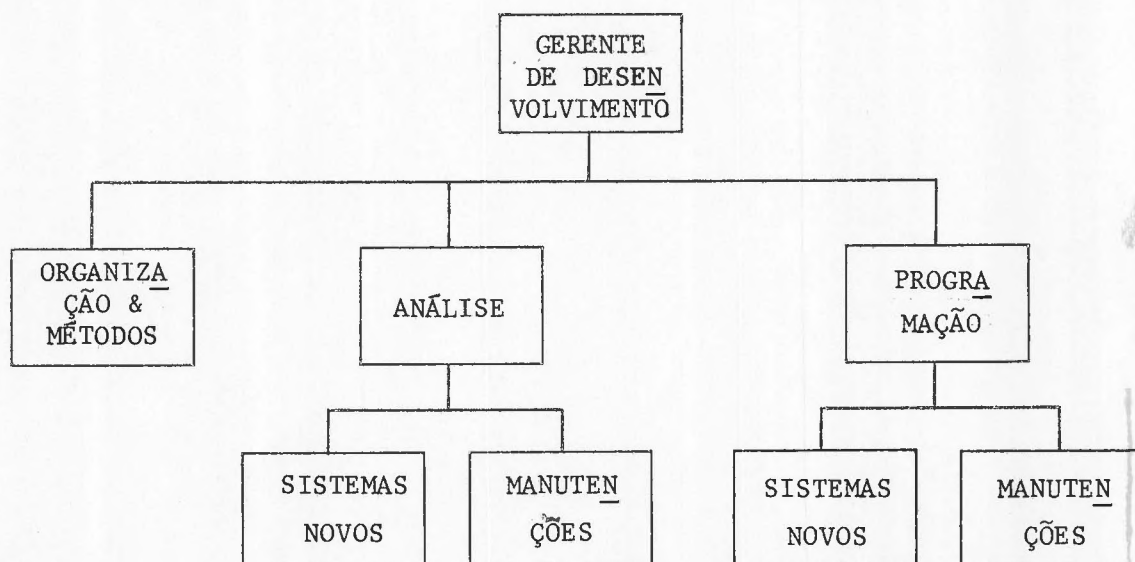


O interesse do presente trabalho é muito maior na área de desenvolvimento de sistemas, e por isso será limita

da por aqui a exposição sobre a área de produção, enfatizando que no que tange as atividades dos programadores, a principal responsabilidade desta última área, é prover as necessárias horas do computador, para que aqueles profissionais possam efetuar os testes e depuração em seus programas.

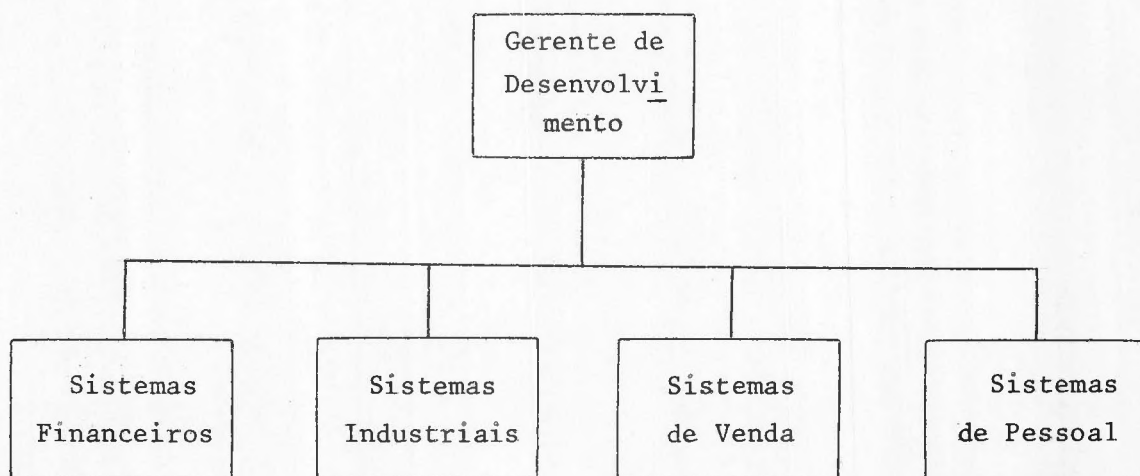
Daqui para a frente se passará, então, a examinar mais atentamente o grupo de desenvolvimento e as principais opções que este possui para organizar o seu trabalho. Existem várias alternativas, mas praticamente todas elas recaem em duas estruturas diferentes, das quais as outras todas são derivadas: estrutura por função e estrutura por aplicação.

No caso da estrutura por função, ou funcional como é mais conhecida, cada supervisor que responde ao gerente de desenvolvimento de sistemas é responsável por alguma especialidade técnica específica. Um organograma clássico é o mostrado na figura abaixo:



Um dos objetivos principais da estrutura funcional, é o de conseguir pessoas extremamente bem habilitadas nas suas especialidades. A forma de funcionamento é a seguinte: quando surge a necessidade de desenvolver um novo sistema, a área da empresa nele interessada contata o gerente de desenvolvimento, o qual incumbe ao supervisor de organização e métodos, da formação de uma equipe. Concluído o trabalho desta equipe, o mesmo é apresentado ao gerente de desenvolvimento, o qual incumbe agora o supervisor de análise também da criação de uma equipe para conduzir a Fase III do ciclo de vida dos Sistemas-Projeto Detalhado. Este fluxo assim segue até a conclusão do trabalho da secção de programação. Nesta altura o gerente de desenvolvimento solicitará a cada supervisor que indiquem pessoas para a formação do task-force, para a implantação.

O outro caso de estrutura organizacional mencionado é o da estrutura por aplicação, cujo organograma clássico é o mostrado na figura abaixo:



A característica básica deste tipo de estrutura é que cada um dos supervisores que responde ao gerente de desenvolvimento tem sob a sua responsabilidade, um grupo onde atuam conjuntamente analistas de O & M, analistas de sistemas e programadores. Além disso, cada um desses grupos é responsável por montar e desenvolver sistemas, para áreas específicas da empresa, só trabalhando em sistemas afins com estas áreas, o que possibilita com que se consiga dentro do CPD, pessoas especializadas em áreas dos usuários. Aliás, esta é a principal vantagem deste tipo de estrutura, havendo inclusive, um intercâmbio muito grande de funcionários entre o CPD e as áreas dos usuários. A grande desvantagem é que pode ocorrer (e sempre ocorre) equipes atoladas de serviços enquanto outras permanecem completamente ociosas.

Estas são resumidamente as duas estruturas organizacionais básicas nas quais atuam os programadores de computador. A escolha de cada uma delas dependerá do gerente de desenvolvimento, o qual levará em consideração, o estágio da empresa em PD (se está iniciando ou se já está numa fase madura), o tipo de profissional que possui (conhecedores ou não de áreas de usuários) e principalmente a previsão futura de novos sistemas.

Pessoalmente a maioria dos programadores prefere trabalhar numa estrutura por aplicação já que esta propicia um contato mais íntimo com analistas e com usuários, o que significa maiores possibilidades de abertura e diversificação profissional.

Apesar da tarefa de programação ser exatamente a mesma em qualquer uma das duas estruturas, observa-se um rendimento médio maior no caso de estruturas por aplicação. Isso se deve, não só ao fato de os programadores se sentirem mais motivados pelos motivos expostos acima, mas também por se tornarem mais familiarizados com os problemas específi-cos do usuário para quem atuam, nem que seja apenas pelo conhecimento da terminologia e linguagem própria deste usuá-rio.

Como se vê, motivação e estrutura organizacional são fatores que podem influir no rendimento de programado-res. Assim como esses fatores, existem vários outros que pas-sarão a ser analisados no próximo capítulo.

C A P Í T U L O I I I

FATORES QUE AFETAM O TRABALHO ANALISADO

O relacionamento dos fatores que afetam a atividade de programação de computadores será feito em 3 passos, já que os mesmos podem ser enquadrados em 3 categorias completamente distintas: Fatores ambientais, Variáveis inerentes a cada programa e Experiência do programador.

1. FATORES AMBIENTAIS

Esses fatores são aqueles que são peculiares a cada centro de processamento de dados (CPD), ou seja, eles são constantes para todos os programas desenvolvidos em um mesmo CPD, mas variam entre CPDs distintos. Como por exemplo, suponhamos o caso de 2 programas idênticos mas que devem ser desenvolvidos cada um em um CPD distinto. Neste caso específico, esses fatores deverão ser considerados. Os principais fatores ambientais incluem:

1.1. Linguagem Utilizada: Além da complexidade naturalmente diferente entre as várias linguagens, há de se considerar também o fato de que as vezes para resolver um deter-

minado problema, uma dada linguagem pode exigir muito mais linhas de codificação que outra. Um exemplo muito comum onde isso acontece, é no caso de empresas comerciais que se deparam com a necessidade de resolver problemas científicos ou matemáticos como por exemplo de Pesquisa Operacional. Como em geral esse tipo de empresa só dispõe de programadores familiarizados com linguagem Cobol, esse tipo de linguagem passa a ser empregado para resolver aquele tipo de problema. O que acaba sendo muito menos eficiente do que se fosse utilizado a linguagem Fortran, por exemplo.

1.2. Restrições de Hardware: O tipo de equipamento onde o programa será executado tem grande influência na elaboração do mesmo. Além das naturais restrições que cada computador impõe, que fazem com que o programa tenha que ser mais ou menos flexível, a própria linguagem às vezes apresenta diferenças nas suas possibilidades. Por exemplo, apesar do IBM 370/145 e NCR 251 serem computadores do mesmo porte, programas em Cobol escritos para um deles não são aceitos no outro e vice-versa.

1.3. Acessibilidade do Computador: Nos primórdios do processamento de dados, o computador era demasiado dispendioso para que ficasse parado a espera do programador; por isso as prioridades sempre favoreciam os trabalhos de produção, ficando os programadores com as sobras para efetuarem seus testes e depurações. Hoje a situação se inverteu, e programado

res ociosos representam uma carga muito onerosa para os CPDs. Por isso o gerente de produção eficaz é aquele que consegue prover aos programadores, disponibilidade de horas de máquina no exato instante em que estas se fizerem necessárias. Isto é comum hoje nas grandes empresas, mas como não é a regra geral, constitui-se em outro fator ambiental.

1.4. Metodologia Empregada: O que se quer aqui dizer com esse item é principalmente, se o CPD emprega ou não programação estruturada, sendo este, talvez, o mais importante dos fatores ambientais. Programação estruturada é a primeira tentativa séria de compreender a programação de computadores em seus vários aspectos deixando de encará-la como uma simples atividade artesanal, para localizá-la como uma disciplina científica sistematizando-a e lhe definindo um caráter metodológico. O objetivo básico dela é melhorar a qualidade de programação. O programador realiza um trabalho chamado "top-down" para dividir um problema em "n" sub-problemas, devendo verificar se essa divisão está correta. Cada sub-problema é verificado até chegar a um nível em que o problema global se considera resolvido. A codificação do programa é feita na mesma ordem hierárquica em que o problema foi resolvido e o programador poderá codificar cada subparte do programa independentemente, com a certeza de que quando tiver todas as subpartes codificadas terá o problema solucionado. Entre as vantagens da programação estruturada pode se citar a padronização que ela acarreta e a grande simplificação que

dá aos programas facilitando o entendimento de um programa por parte de outros programadores que não tenham sido seus autores. Para que isso aconteça, muitas vezes a eficiência operacional do programa (uso de memória de computador e tempo de execução) é sacrificada. Mas esse aspecto é amplamente compensado pelas reduções em custo, que acontecem quando das inevitáveis necessidades futuras de manutenção nos programas. Justamente a técnica da programação estruturada só surgiu recentemente (início da presente década) pois os primeiros computadores, muito mais desprovidos de recursos que os de hoje e também muito mais caros relativamente, acabavam por obscurecer o custo de programação; como com o decorrer dos anos, os computadores foram se tornando mais baratos (17), hoje em dia já ocorre na maioria dos CPDs, de os orçamentos das equipes de programação ultrapassarem os custos do computador de forma que a tendência é de limitar os primeiros. A médio e longo prazo isso acontece plenamente com a programação estruturada, pois em média o programador leva algum tempo a mais para codificar o programa, mas os tempos gastos posteriormente em manutenção do programa são infinitamente menores do que no caso de não se usar programação estruturada (18).

¹⁷ Martins, L. e Soares, M., "Soft: O papel do Estado", Dados e Idéias, vol. 4, nº 3 : 2-9, Janeiro 79.

¹⁸ Yourdon, E., "Making the move to structured programming", Datamation, vol. 21, nº 6 : 52-56, Junho 75.

1.5. Padrões de Documentação: Como já foi dito anteriormente, o programador deve documentar os seus programas. Esse trabalho será tão árduo quanto mais rigorosos forem os padrões instituídos pela empresa, a qual é a grande beneficiada ao poder dispor de documentação precisa e detalhada. No entanto, não é esse o único tipo de documentação a influir na tarefa de programação. A qualidade da documentação presente na pasta de programa, ou seja, a descrição do programa, a qual é feita pelo analista de sistemas, também tem uma importância muito grande. O ideal seria que essa descrição fosse tão completa que não houvesse necessidade de nenhum contato pessoal entre analistas e programadores. E, esta é a tendência que vem crescendo nos últimos anos, apesar de existirem alguns grupos de desenvolvimento onde praticamente o programador acaba descobrindo a finalidade do programa que deve codificar, somente após horas de conversa com os analistas. Evidentemente, esse aspecto é fonte de maiores problemas, quando se atua em uma estrutura funcional. Aliás, para que um grupo de desenvolvimento possa optar por esta última estrutura, o ideal seria que já tivesse atingido maturidade suficiente para poder produzir descrições de programas que dispensassem interações analista-programador.

1.6. Suportes de Software: Recursos adicionais de Software disponíveis em uma instalação tais como utilitários, librarians ou outros "enlatados" podem simplificar sobremaneira a tarefa de programação, constituindo-se pois em mais um

fator ambiental. O sistema operacional (DOS, OS, etc.) disponível, também deve ser enquadrado nesta categoria.

1.7. Procedimentos de Segurança: A segurança em processamento de dados é um dado vital e, como tal, deve ser encarada com todo rigor. Procedimentos mais ou menos flexíveis, acabam por repercutir na tarefa do programador. Muitas vezes a falta de bom senso acaba por impor procedimentos burocráticos que passam a encarar programadores como se estes fossem um bando de mal-intencionados. Evidentemente nessas circunstâncias, que não são raras, a tendência é de haver uma queda na produtividade da tarefa de programação.

1.8. Instalações Físicas: O trabalho do programador requer ambientes confortáveis e, sobretudo, silenciosos, de preferência salas com no máximo 2 pessoas. Já se foi o tempo em que apenas o computador requeria instalações sofisticadas. Como já foi dito anteriormente, tudo deve ser feito para se conseguir extrair a máxima produtividade do programador, já que este é atualmente, recurso bem escasso e dos mais caros. Instalações físicas adequadas não significam grandes dispêndios e podem ter grandes repercussões na produtividade. A experiência mostra que com instalações inadequadas (por exemplo, ambientes abertos com mais de 5 pessoas), a produtividade chega a cair à metade da que existiria com condições satisfatórias.

Estes são assim os fatores ambientais mais importantes, sendo importante realçar que a influência dos mesmos não varia de um programa para outro. Eles afetam a todos igualmente. É o contrário do que sucede com o próximo conjunto de fatores.

2. VARIÁVEIS INERENTES A CADA PROGRAMA

Esta outra categoria de fatores que afetam a tarefa de programação, abrange os fatores diretamente relacionados com o programa específico que deve ser desenvolvido.

Esta categoria será aqui abordada separadamente em duas subcategorias: Função Principal ou Tipo de Programa e Características de Entradas/Saídas do Programa. Outros autores tem aqui colocado o item complexidade, que todavia é um fator de caráter subjetivo. Um programa muito complexo para um programador pode ser extremamente simples para outro. Além disso como se verá adiante, o produto natural de uma avaliação feita com as 2 subcategorias acima definidas é a própria complexidade do programa, a qual acaba sendo inclusive quantificada ao se estimar o número de dias que o desenvolvimento do mesmo deve durar.

2.1. Tipo de Programa: Esta é uma categoria de difícil caracterização, em ambientes onde não se utiliza filosofia top-down de programação estruturada. Isso porque nessas condições, os programas, em geral, não são projetados de forma

a abrangerem uma ou poucas funções básicas, sendo comum, ao contrário, o analista definir sistemas onde um programa faz praticamente todas as funções existentes e os outros programas restantes limitam-se a ser meros impressores de relatórios. Essa filosofia de desenvolvimento de sistemas está aos poucos sendo superada, principalmente em função das dificuldades que advêm, quando da necessidade de se efetuar alterações futuras nos sistemas, já que se torna muitas vezes difícil descobrir em que trecho daquele programa-único, ou dos programas do tipo "faz-tudo", se encontra determinada função. E quando se descobre, ela, em geral, está emaranhada ou acoplada com outras, complicando sobremaneira, qualquer alteração. O advento da programação estruturada ajudou a resolver este tipo de problema, principalmente pela filosofia que esta metodologia de programação acarreta: tudo deve ser feito para se ter programas claros, nos quais seja simples se efetuar quaisquer manutenções.

Para se alcançar esse objetivo, nada melhor que realizem poucas funções cada um. Evidentemente, o número de programas por sistemas é aumentado com essa abordagem (19).

As instalações de processamento comercial (bancos, indústrias, magazines, empresas públicas, etc.) que optam por este último caminho, constataam com surpresa que, na verdade, são poucas as funções básicas que existem em processa

¹⁹ Garbassi, U. e Mc Cracken, D., "Programação Cobol", traduzido do inglês por Danilo A. Nogueira, São Paulo, Editora Atlas S.A., 1976, pág.352.

mento de dados comercial e, em consequência, também são poucas as modalidades ou tipos diferentes de programas existentes. Nessas condições, é possível se afirmar sem medo de errar que, pelo menos 80% dos programas definidos com essa conceituação enquadram-se em uma das 5 seguintes modalidades:

1. Atualização/Manutenção de Arquivos
2. Consistência/Validação de Campos
3. Impressão de Relatórios
4. Organização de Arquivos
5. Formatação de Registros

Essas são as modalidades preponderantes e encontráveis em todo Centro de Processamento de Dados. Outras modalidades, como por exemplo, programas extratores de data base, processadores de cálculos, etc., não serão aqui consideradas por não serem tão comuns.

A seguir, descreve-se o que vem a ser cada uma das 5 modalidades supra-referidas:

2.1.1. Programas de Atualização/Manutenção de Arquivos: Antes de entrar propriamente na descrição do que faz esse tipo de programa, é necessário que sejam inicialmente feitos alguns esclarecimentos sobre o armazenamento de informações em sistemas automatizados e mais especificamente sobre as modalidades de arquivos que este tipo particular de programa manuseia.

Para facilitar a utilização dos dados e garantir a

precisão e correção dos resultados, os sistemas automatizados utilizam os dados de forma agrupada. Arquivo é o nome que se dá a esses agrupamentos. Dois conceitos devem ser colocados para que se entenda melhor o que vem a ser um arquivo:

CAMPO : É o conjunto sequencial de um ou mais caracteres considerados como uma unidade de informação. Exemplos de campos utilizados em sistemas bancários: nome do cliente, data de nascimento do mesmo, saldo médio da conta corrente, juros pagos, etc.

REGISTRO: É um conjunto de campos agrupados com a finalidade de permitir o processamento, tendo cada um, um campo identificador também chamado de chave. Por exemplo, o conjunto de dados referentes a um dado cliente (nº da conta, nome, saldo atual, saldo médio, etc.), constitui um exemplo de registro que tem como chave o campo nº da conta.

Os programas de atualização/manutenção de arquivos, fundamentalmente, manuseiam duas modalidades de arquivos: Arquivo Mestre e Arquivo Transações.

O arquivo Mestre, também conhecido por Permanente, Cadastro ou Principal, possui duas categorias de campos de dados: estáticos, que são aqueles que perduram sem modificação por um longo período de tempo (em geral de 6 a 12 meses), e campos de dados dinâmicos, que correspondem aos valores que são frequentemente alterados. Por exemplo, em um arqui-

vo mestre de um sistema de contas correntes de um banco, da dos estáticos seriam o número da conta corrente e o nome do cliente, enquanto dados dinâmicos seriam o saldo da sua conta ou o número de cheques emitidos até uma determinada data. O Arquivo de Transações é aquele que é informado a cada novo processamento, contendo informações que se destinam a atualizar o arquivo mestre.

Programas de Atualização/Manutenção de arquivos são basicamente aqueles que leem um arquivo mestre e um arquivo de transações utilizando os dados do último para acertar ou ajustar os dados do primeiro.

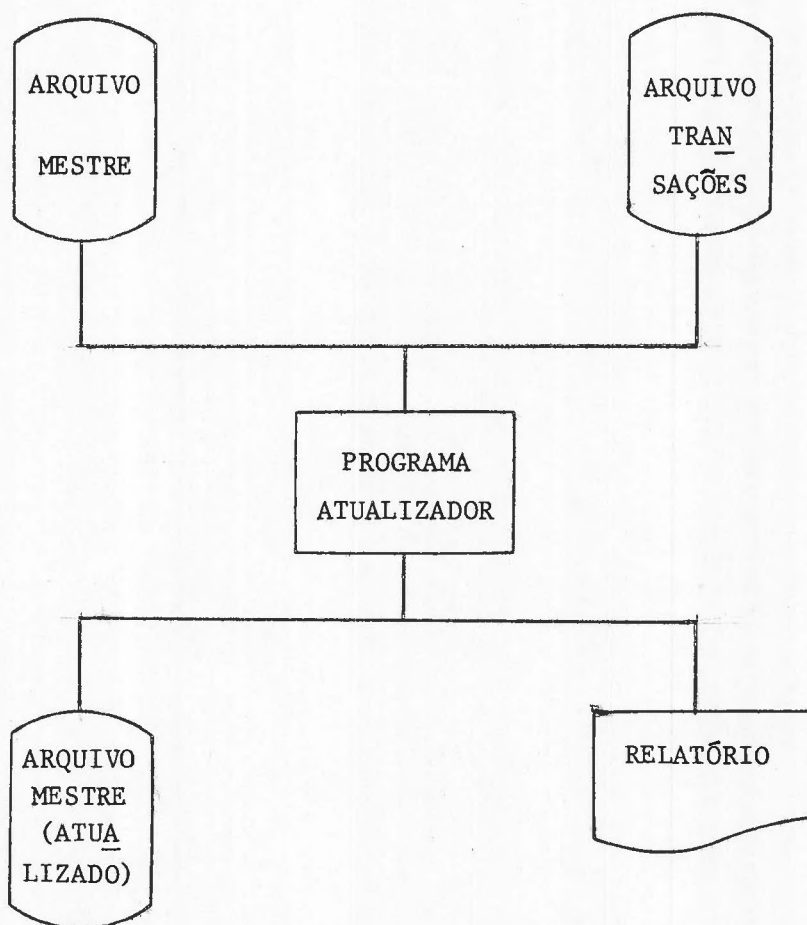
Na verdade existe uma diferença entre atualização e manutenção. Em geral, a palavra atualização é usada exclusivamente em relação aos campos dinâmicos, reservando-se o termo manutenção para os campos estáticos. Como na prática muitas vezes essas duas funções se confundem, os termos serão aqui usados indistintamente. Em essência, a função de um programa destes é relativamente simples. Ele deve ler um a um os registros do arquivo transações e identificar os correspondentes (se houverem) no arquivo mestre, efetuando as devidas alterações nos campos destes últimos.

Dentro dessa conceituação, um programa de atualização sempre lê dois arquivos. No entanto, o número de arquivos gravados pode variar. Na grande maioria dos casos, realmente é um único arquivo, que é evidentemente o arquivo mestre atualizado ou mantido. No entanto algumas vezes poderá ser necessário gravar algum outro tipo de arquivo, con-

tendo alguma espécie particular de informação.

Também na grande maioria das vezes, os programas atualizadores emitem um único tipo de relatório, o qual em geral é uma relação dos registros que se encontravam no arquivo de transações, mencionando se foi efetuada a atualização/manutenção, e, se não o foi, qual teria sido o motivo.

Um fluxograma simplificado representando o modelo de programa atualizador é o que está esquematizado a seguir:



2.1.2. Programas de Consistência/Validação de Cam-

pos: Também chamados de programas de crítica, eles resultam da necessidade de se ter sempre a certeza de que os dados de entrada estão completos e são válidos. Isso deve ser feito, pois erros podem ser cometidos, principalmente em duas ocasiões:

- quando o usuário (ou seus representantes) preenche os documentos de entrada;
- quando esses documentos são perfurados em cartão, digitados em fita ou convertidos para algum outro meio de entrada de dados.

Para se saber se os dados estão completos, é necessário que todos os registros e lotes entrados no sistema sejam contados. Uma das funções dos programas de consistência é efetuar essas contagens e emitir relatórios apontando possíveis irregularidades. Estas contagens devem corresponder às cifras obtidas manualmente quando da confecção e coleta dos documentos originais. Usualmente não se faz apenas a contagem do número de registros (em geral, um documento de entrada corresponde a um registro de entrada e vice-versa), mas também escolhem-se alguns campos internos aos documentos que terão seus valores aferidos manualmente (e preenchidos nas fichas de lote) e pelo computador (através do programa de consistência). Claro está, que essa acumulação de campos não se destina apenas a constatar falta de registros, mas principalmente, permite constatar o correto preenchimento dos campos cuja acumulação venha a ser feita.

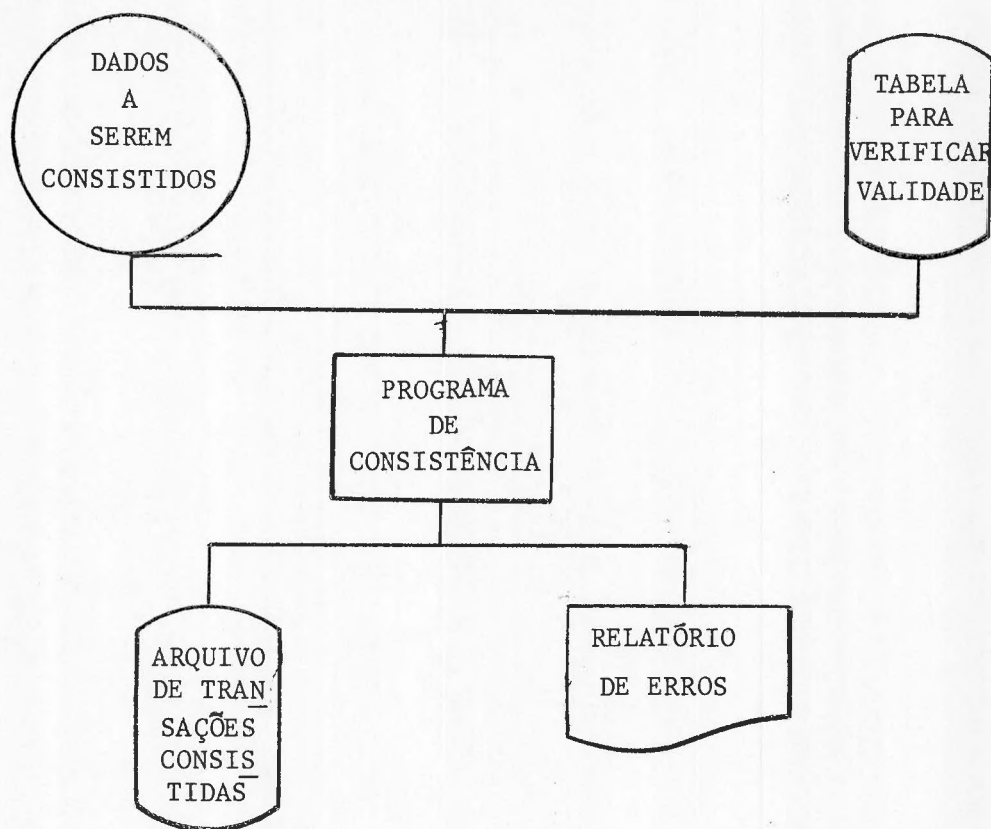
Com relação a validade de campos, normalmente uma série de testes devem ser projetados, o que pode acarretar o exame de campos peculiares nos registros, para ver se estão de acordo com as regras especificadas. Costuma-se também efetuar verificações nas relações lógicas entre os campos dentro dos registros, quando possível.

Entre os vários tipos de provas, a mais simples é a de verificar que os campos especificados como campos numéricos só contenham efetivamente caracteres numéricos e que os campos definidos como alfabéticos só contenham caracteres alfabéticos. Outra verificação simples é a efetuada para campos que só podem conter determinados valores, ou então valores dentro de determinada faixa.

As verificações vão se tornando mais complexas, à medida que elas não podem ser efetuadas por confrontações contra valores ou tabelas internas ao programa, mas outros arquivos devem passar a ser consultados. É o caso, por exemplo, que sucede em um sistema de controle de estoques onde possivelmente, caso se queira saber da validade de um campo como "código de peças", dever-se-á consultar o "arquivo mestre de peças", onde todas elas estarão relacionadas, provavelmente constituindo cada uma um registro.

Por isso em programas de consistência, o número de arquivos de entrada e o número de arquivos de saída é extremamente variável.

É o seguinte o esquema gráfico representativo desse tipo de programa.

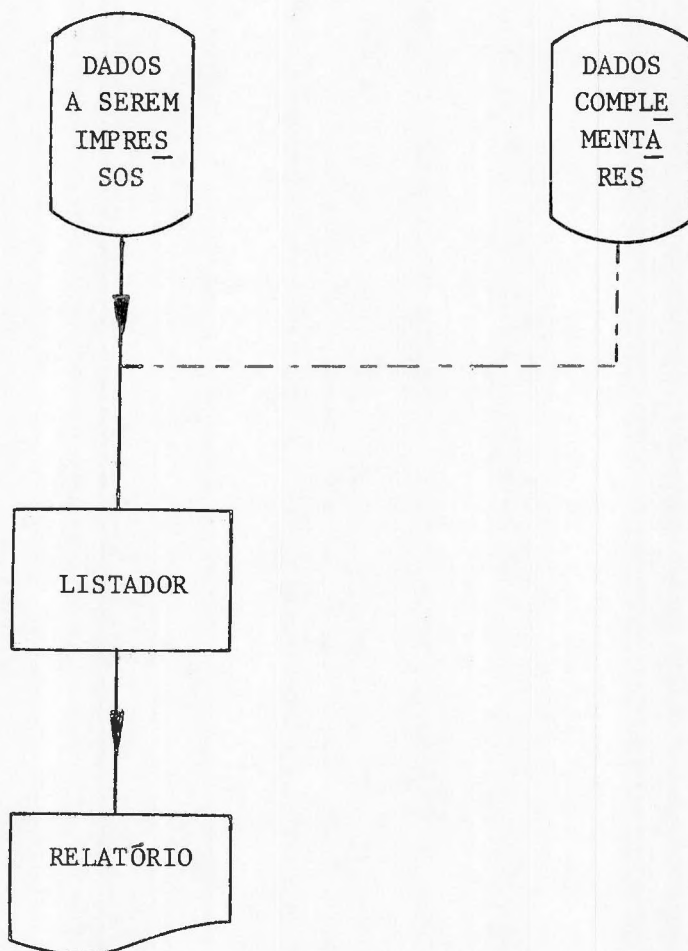


2.1.3. Programas de Impressão de Relatórios: Apesar da função impressão de relatórios existir na maioria dos programas, principalmente em atualizadores e programas de consistência, aqui enquadraram-se nessa categoria aqueles programas cujo objetivo primordial é o de imprimir relatórios. Ou seja, estes são programas que lêem um arquivo básico o qual ou será listado pura e simplesmente, ou então servirá de base para a impressão de algum relatório. Este último caso acontece quando alguma coisa deve ser feita com o conteúdo do arquivo antes que este seja colocado no papel. Esta alguma coisa pode ser, por exemplo, algum cálculo simples, uma

eventual consulta a outros arquivos para obter dados complementares, ou então uma consulta a cartões contendo parâmetros definidores do relatório a ser impresso.

Uma característica das mais importantes, se bem que não aconteça absolutamente em todos os casos, mas em boa parte, é que esse tipo de programa não gera como output nenhum arquivo em fita, disco ou cartão, mas unicamente um relatório que, aliás, é o objetivo máximo do mesmo.

O fluxograma básico representativo desse tipo de programa é o seguinte:



Como se ve, esses programas se situam sempre nas extremidades dos fluxogramas dos sistemas, já que em geral eles não costumam fornecer inputs para outros programas.

2.1.4. Programas que Organizam Arquivos: Quase nunca as transações que entram em um sistema já estão ordenadas na forma em que serão lidas pelos programas do mesmo. Nesse caso pode ser necessário classificá-las antes de efetuar algum processamento. Este é um tipo de programa organizador de arquivos. Algumas vezes inclusive, o programa organizador pode ser um programa utilitário (do tipo general purpose) desenvolvido pelo próprio fabricante do equipamento disponível no C.P.D. Quando esta disponibilidade não existir, será necessário desenvolver um programa especialmente para este fim. Além de programas que efetuam classificações, outros exemplos desta modalidade são programas intercaladores, ou seja, programas que para um dado grupo de registros inserem nesse grupo outros registros, mediante certas regras básicas. Outra espécie ainda, são os programas que expandem arquivos, ou seja, para cada registro lido de um dado arquivo, mediante o cumprimento de certas regras definidas no programa ou então mediante consulta a parâmetros definidos em outros arquivos, é efetuada uma criação de novos registros, normalmente semelhantes aos que serviram de base. Um último exemplo dentro desta categoria é o dos programas agregadores, os quais são programas que lêem um único arquivo, agregando os campos de valor dos registros que

possuem a mesma chave. Evidente que para esse tipo de programa, os arquivos output sempre conterão menos registros que o arquivo input.

Um exemplo de agregação é dado a seguir para o caso de um arquivo de número de contas correntes em um banco:

A - arquivo de entrada

| tipo | registro | nº conta corrente | valor débito em cheque | nº cheques no depósito |
|------|----------|-------------------|------------------------|------------------------|
| | 07 | 10547 | 1.000,00 | 1 |
| | 07 | 10547 | 500,00 | 2 |
| | 07 | 11239 | 1.230,00 | 4 |
| | 07 | 11239 | 1.000,00 | 2 |
| | 07 | 11239 | 10.000,00 | 3 |
| | 07 | 14598 | 8.975,00 | 2 |

B - arquivo de saída (após agregação)

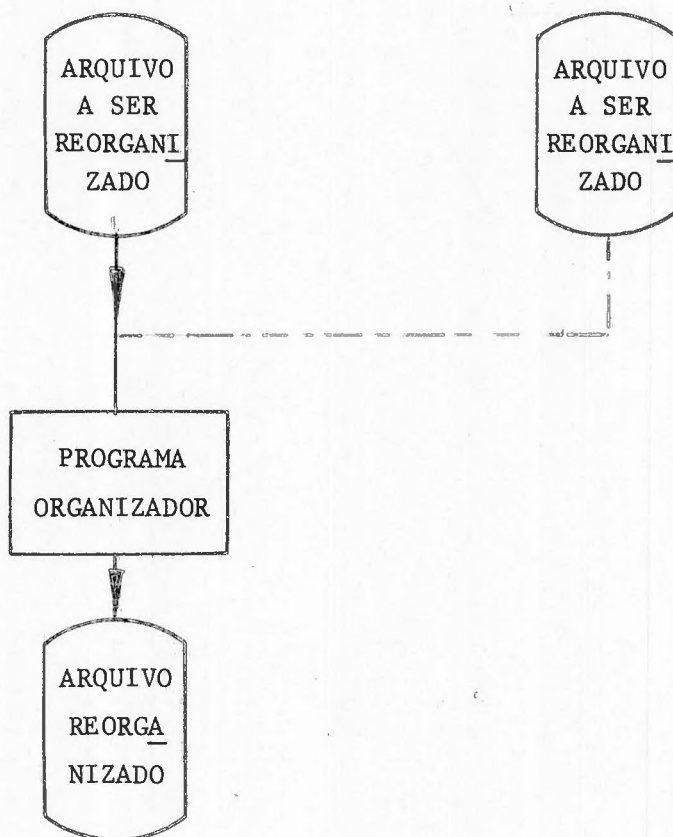
| | | | |
|----|-------|-----------|---|
| 07 | 10547 | 1.500,00 | 3 |
| 07 | 11239 | 12.230,00 | 9 |
| 07 | 14598 | 8.975,00 | 2 |

A finalidade pois, destes programas todos é a de mudar arquivos em uma dada organização (ou nenhuma organização) para uma outra pré definida. Os exemplos acima mencionados são apenas alguns poucos, existindo vários outros entre eles, além das combinações dos casos citados.

Uma característica que aparece na maioria dos ca-

so para este tipo de programa é a deles dispensarem relatórios.

O fluxograma que representa a maioria desses tipos de programas é o seguinte:

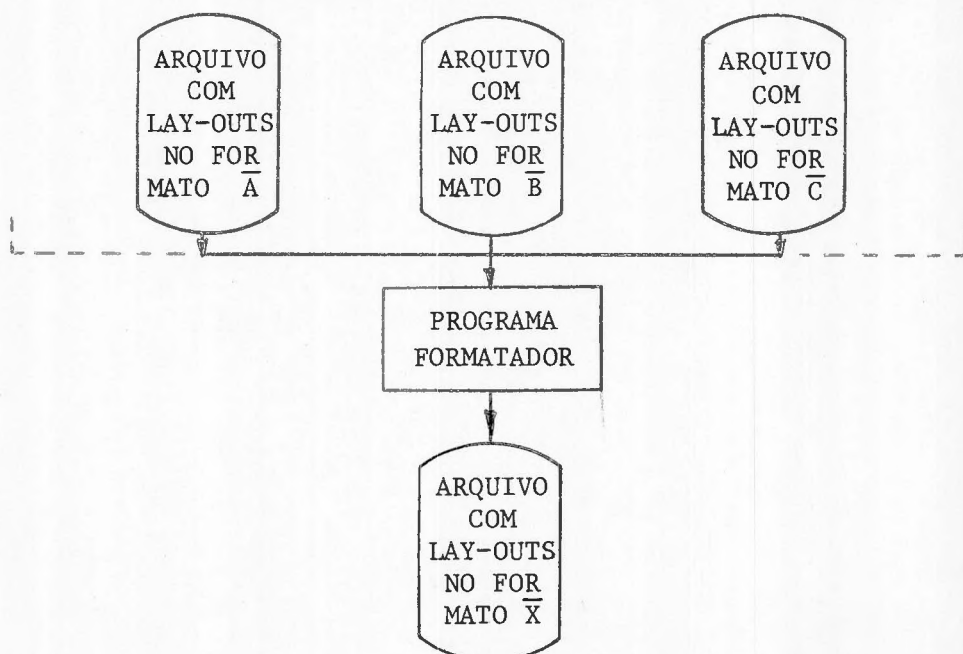


2.1.5. Programas que Formatam Registros: Este é o tipo mais simples e dos mais comuns de programa. Comum porque é muito frequente em Processamento de Dados a necessidade de se alterar o lay-out (distribuição de campos) de determinados registros dentro do fluxo de um mesmo sistema. Assim sendo, esses programas formatadores suprem essa necessidade ou seja, lêem registros em um dado formato para soltá-lo em ou

tro completa ou parcialmente diferente. Algumas vezes campos são suprimidos, outros são transformados e outros são repetidos identicamente na saída. É comum, embora não seja regra geral, também para esse tipo de programa, a inexistência de relatórios de saída.

Outra denominação que se costuma aplicar a este tipo de programa é a de interfaces, já que eles costumam servir de pontes entre dois ou mais sistemas. Em outras palavras, uma situação muito comum é a de um dado sistema automatizado necessitar receber informações de outro sistema automatizado, sendo pouco provável nesses casos que os registros nesses dois sistemas possuam o mesmo formato. Então algum programa deverá ler algum arquivo do sistema fornecedor de informações e formatar os seus registros no formato aceito pelo sistema captador de informações. Este subcaso de programa formatador é também denominado de programa de interface.

O fluxograma mais representativo é o seguinte:



Os cinco tipos de programas acima definidos não esgotam todos os tipos possíveis de programas. Todavia, por abrangerem, como já foi dito, a grande maioria das funções (pelo menos 80%) serão os únicos a serem aqui considerados.

2.2. Características das Entradas e Saídas: Além do tipo do programa em si, um outro fator peculiar a cada programa e que afeta a tarefa de programação, é a configuração das entradas e saídas dos mesmos. Por configuração de entradas e saídas, se quer denominar o conjunto de arquivos lidos e gravados, além dos diferentes formatos de registro existentes nesses arquivos. Inclui-se também neste item, os diferentes formatos de relatórios que o programa deve emitir.

É entendimento da totalidade dos autores que abordaram o presente assunto (20) e (21), que a complexidade de um programa aumenta à medida em que ele deve manipular um número de arquivos crescente. Todavia um arquivo não apresenta sempre um único tipo de registro. Existem arquivos que chegam a apresentar 30 ou mais diferentes tipos de formatos de registros (lay-outs). Evidentemente o aumento do número de registros manuseados por um programa também é um fator acarretador de maior complexidade.

Não é apenas o número de arquivos, registros e re

²⁰ Dick Brandon, Project Control Standars, Philadelphia, Auerbach, 1970, pág. 94.

²¹ Ralph Szveda, Information Processing Management, Princeton, Auerbach, 1972.

latórios que afetará o trabalho do programador. Outro aspecto vital a ser considerado é o de em um mesmo arquivo haverem ou não registros de tamanhos variáveis, fato este que tornará mais complexa a tarefa do programador.

Apenas a título de complementação, já que não é opinião aqui endossada, mas é a de alguns conceituados autores (22) e (23), também deveriam ser considerados dentro do presente item o fato de um arquivo estar contido em disco, fita ou cartão; bem como a maneira em que o mesmo fosse lido ou gravado, se sequencialmente, index-sequencialmente ou diretamente.

Esta opinião não está sendo aqui defendida, pois através de levantamentos e observações por nós efetuados, não foi possível evidenciar a influência deste último aspecto.

O que se pode comprovar é que existem 8 variáveis, essas sim, cuja influência na tarefa de programação e, portanto, na sua complexidade merece ser analisada. São as seguintes:

1. Nº de arquivos de entrada: Tratam-se dos arquivos lógicos que são lidos pelo programa, não interessando o meio em que estejam acondicionados, cartão, disco ou fita. Esta informação pode ser obtida diretamente do fluxo de sis

²²Dick Brandon, Project Control Standards, op. cit.

²³Richard Shell, "Work Measurement for Computer Programming Operations", Industrial Engineering, vol. 4, nº 10 : 32-36, Outubro 72.

tema no início da Fase III do ciclo de vida dos sistemas.

2. Nº de arquivos de saída: Tratam-se dos arquivos lógicos que são gravados pelo programa, não devendo ser considerados os relatórios. Somente, arquivos que posteriormente poderão ser usados por outros programas e que então podem estar em cartão, disco ou fita. Também obtido do fluxo do sistema.
3. Nº de tipos diferentes de relatórios: Tratam-se dos diferentes lay-outs de relatório que existem para o programa imprimir. Também obtido do fluxo do sistema.
4. Somatória das variáveis 1, 2 e 3: Autoexplicativo.
5. Nº de diferentes tipos de registros de entrada: Nesse caso, cada arquivo lido pelo programa deve ser examinado para se constatar quantos lay-outs diferentes de registros ele possui. Além disso devem ser excluídos os registros que se repetem em dois arquivos ou mais. Alguns exemplos, a título de esclarecimento:

a) 3 arquivos de entrada X, Y, Z com os seguintes formatos de registro cada um

X possuindo formatos A

B

Y possuindo formato C

Z possuindo formatos B

D

Nesse caso o valor da presente variável será 4 (A, B,

C, D).

b) 4 arquivos de entrada X, Y, W, Z, com os seguintes formatos de registro cada um:

X possuindo A

B

C

Y possuindo A

D

E

Z possuindo A

B

C

W possuindo F

G

H

Nesse caso, o valor da variável será 8 (A, B, C, D, E, F, G, H).

OBS: Ainda com relação a este item, ressalte-se que não devem ser considerados em cada programa, como uma sua variável aqueles tipos de registros que figuram nos arquivos, mas que não são manuseados pelo programa. Se um arquivo possuir 3 tipos diferentes de registros mas apenas dois desses são lidos pelo programa, o número a considerar deve ser dois e não três.

6, Nº de diferentes tipos de registros de saída: Neste caso

devem ser computados os registros existentes nos arquivos de saída, incluindo-se "linha detalhe" de relatórios. Os exemplos e observações da variável anterior, nº 5, são totalmente aplicáveis aqui.

7. Soma das variáveis 5 e 6: Autoexplicativo.
8. Nº de tipos diferentes de registros manuseados pelo programa: Ao se efetuar a contagem de registros para este parâmetro, deve-se evitar repetições, ou seja, se algum formato já foi considerado na entrada, não deve ser computado na saída.

Exemplo: Arquivos de Entrada X e W

e

Arquivos de Saída Y e Z

X possui os formatos A

B

W possui os formatos A

C

D

Y possui os formatos A

E

F

Z possui o formato G

O número a ser considerado é 7 (A, B, C, D, E, F, G), apesar de que a variável 5, nº de registro de entrada, seria igual a 4 e a variável 6, nº de registros de saída se

ria também igual a 4, produzindo, portanto, uma variável soma igual a 8.

Finalmente pode-se passar para o exame da última categoria de fatores, que tem influência sobre o trabalho de programação.

3. EXPERIÊNCIA DO PROGRAMADOR

Em qualquer tipo de trabalho, a experiência e habilidade do executante podem fazer com que o mesmo seja concluído em mais ou menos tempo. Por isso, inclusive em atividades fabris, quando do estabelecimento de tempos padrões para execução de atividades, é rotineiro que se efetue avaliações de ritmo, para que as pessoas que serviram como base para obtenção dos dados não permitam o desvirtuamento dos resultados, devido a influência de seus ritmos próprios de trabalho.

O mesmo sucede em programação de computadores. Quando se tenta estabelecer padrões, quando se avalia produtividade, ou quando se fazem estimativas, é preponderante que seja conhecida e levada em conta a experiência, tanto de programadores que estejam sendo observados, como de programadores que serão incumbidos de determinadas tarefas.

O ideal seria que, para cada programador, fosse conhecida exatamente a sua experiência e conhecimento, mas isso é impossível de acontecer na prática, e por isso, tanto para fins de efetuar observações como para estimativas, os

programadores tem sido enquadrados em três categorias, conforme a sua experiência:

1. Programador Senior (Sr.)
2. Programador Pleno (Pl.)
3. Programador Junior (Jr.)

Infelizmente, não existe um consenso geral das empresas sobre o que se entende por cada uma das denominações Junior, Pleno e Senior. Aliás, nem mesmo essas denominações são utilizadas por todas as empresas. Algumas definem somente duas categorias, programadores trainee, e programador, apenas para citar um outro exemplo. Todavia, como Jr., Pleno e Sr. são as mais utilizadas, tratar-se-á de conceituá-las tentando estabelecer o julgamento predominante.

1. Programador Junior: São programadores com cursos de programação recém concluídos, e que normalmente só conhecem um tipo de equipamento. Existem programas (os mais complexos) que para este programador seriam de conclusão impossível. Em geral sua experiência é inferior a dois anos e dificilmente dominam a utilização de cartões controle.
2. Programador Senior: São programadores que já desenvolveram praticamente todo tipo de programa, não existindo para eles nenhum tipo que não possa ser levado a cabo. Conhecem pelo menos dois equipamentos diferentes. Conseguem ter boa compreensão de sistemas automatizados, sendo bem sucedidos em localizar os papéis de seus progra

mas específicos dentro dos sistemas em que estão envolvidos. Possuem total domínio do uso de cartões controle . Em geral, sua experiência é superior a quatro anos.

3. Programador Pleno: Apresenta as qualidades intermediárias entre as do programador Junior e programador Senior.

Mais difícil ainda do que tentar definir cada categoria de programador, é caracterizar a velocidade de trabalho de cada um. Na bibliografia existem alguns comentários (24), mas aqui será necessário nos basearmos na nossa experiência. Esta foi conseguida solicitando que o mesmo programa fosse desenvolvido por um programador Junior, um Pleno e um Senior. Essa experiência foi feita várias vezes, com programas de diferentes graus de complexidade, de forma a se poder apurar as performances atingidas.

Dessa forma, chegamos à conclusão que, considerando o ritmo do programador Pleno igual a 1 (hum), os ritmos dos programadores Junior e Senior, respectivamente, variarão dentro das seguintes faixas:

Junior 0,4 a 0,8

Senior 1,1 a 1,5

²⁴ Ralph Szweda, Information Processing Management, op. cit.

C A P Í T U L O I V

ANÁLISE DAS TÉCNICAS EXISTENTES PARA MEDIR O TRABALHO ANALISADO

1. CONCEITOS BÁSICOS DE MEDIDA DO TRABALHO

Definindo-se da forma mais abrangente, Medida do Trabalho é simplesmente o meio de se determinar uma relação de equivalência entre uma quantidade de trabalho realizada e o número de homens/hora necessário para realizá-la. Essa relação, muitas vezes, é identificada por um padrão unitário de tempo, o qual, em geral, constitui-se no tempo requerido por um trabalhador qualificado trabalhando em um ritmo normal para produzir uma unidade de uma determinada tarefa previamente definida. Em suma, escolhida a unidade, pode-se responder à pergunta: "Quanto tempo levar-se-á para executar a tarefa X?"

O objetivo principal da Medida de Trabalho é o de substituir avaliações subjetivas por avaliações objetivas a cerca das durações que devem demandar as tarefas.

Todo gerente ou supervisor, desde o chefe da seção de tornos, até o diretor industrial, tem suas próprias impressões acerca de quanto tempo devem dispender os seus

funcionários na execução das respectivas tarefas, ou seja, todos já dispõem dos seus próprios padrões, os quais, sem dúvida, foram determinados de forma subjetiva. Assim sendo, o objetivo da Medida do Trabalho é o de substituir esses critérios subjetivos por valores obtidos de uma forma mais científica.

Justamente é a área de desenvolvimento de programas de computador, onde atualmente se faz um uso intenso de critérios subjetivos para se efetuar estimativas de tempos, havendo pois, um terreno fértil para a aplicação de técnicas de Medida do Trabalho.

É importante que fique bem claro que a Medida do Trabalho não objetiva fazer com que as pessoas trabalhem mais depressa, ou simplesmente que trabalhem mais. Ela objetiva responder sobre o que seria um bom dia de trabalho, tanto do ponto de vista dos empregados, como dos empregadores. Trata-se, pois, de um ferramenta que, a exemplo de outras ferramentas de uso gerencial, quando temperada com bom senso e discernimento na sua utilização, pode incrementar sobremaneira a produtividade.

Dentro da área de programação de computadores, as aplicações de Medida do Trabalho se encaixam em duas grandes categorias:

- Fornecimento de dados para planejamento das tarefas;
- Fornecimento de dados para o controle das mesmas.

Em essência, os padrões obtidos com a aplicação de

Medidas do Trabalho permitem que se diga com grande chance de acerto, quantos dias ou horas um determinado programador levará para concluir um programa de computador, além de permitir que se estabeleçam avaliações precisas de produtividade para se conhecer quais os programadores que podem ou devem, por exemplo, receber incentivos. Especificando-se um pouco mais essa caracterização, pode-se dizer que o gerente de sistemas que dispõe dos padrões de Medida do Trabalho pode estimar futuras necessidades de mão-de-obra, estabelecer preços para serviços prestados a usuários ou terceiros, produzir orçamentos mais precisos, estabelecer gols de performance, programar de forma racional as várias tarefas do seu departamento, etc.

Percebe-se, pois, que a Medida do Trabalho atua tanto no planejamento a curto prazo, como planejamento a longo prazo. Mas Medida do Trabalho também é uma ferramenta de controle, permitindo ao gerente, detectar eventuais pontos críticos ou problemas que necessitem sua intervenção, como por exemplo, excessos no quadro de funcionários ou no número de horas extras, custos excessivos ou baixa performance de algumas áreas, de supervisores ou de empregados, etc..

Existem, portanto, características diferenciadoras entre as duas grandes funções da aplicação das técnicas de Medida do Trabalho: o planejamento é preventivo, e o controle é corretivo.

No entanto, existe também um interrelacionamento

muito grande entre as duas funções, que pode ser enfatizado pelo seguinte aspecto: Muitas situações que requerem controle apurado, poderiam dispensá-lo se Medida do Trabalho tivesse sido usada inicialmente para conseguir um planejamento meticoloso.

Não é possível se falar em Medida do Trabalho sem se definir tempo normal e tempo padrão. Considere-se o conceito de Barnes (25) sobre estes dois aspectos:

"O Tempo Normal representa o tempo que um operador qualificado e treinado, trabalhando com um ritmo normal, levaria para completar um ciclo da operação. Ele não é o tempo padrão para a tarefa, desde que é necessário adicionar-se as tolerâncias ao tempo normal, a fim de se obter o tempo padrão".

O problema que aqui se coloca é o de transplantar esta definição para a área de programação de computadores. Apesar da definição acima estar muito precisa e clara, ela envolve como boa parte das definições de estudo de tempos, alguns aspectos de caráter subjetivo, principalmente no que diz respeito a:

O que é ritmo normal?

Segundo Riggs (26):

²⁵ Ralph Barnes, Estudo de Movimentos e de Tempos: Projeto e Medida do Trabalho, traduzido do inglês por Sérgio O. Assis e outros, S. Paulo, Editora Edgard Blucher LTDA., 1963, pág. 419.

²⁶ James Riggs, Administração da Produção: Planejamento, Análise e Controle, traduzido do inglês por Eda Quadros, São Paulo, Editora Atlas, 1976, pág. 323.

"É aquele que pode ser obtido e mantido por um trabalhador médio, durante um dia típico de trabalho sem fadiga indevida".

Ninguém melhor para caracterizar esse trabalhador médio na área de programação de computadores, do que o programador Pleno conforme anteriormente caracterizado. Assim sendo, reunindo-se tudo que foi dito até aqui, serão adotados as seguintes definições para tempo normal e tempo padrão:

Tempo Normal - É o tempo necessário para que um programador desenvolva um programa, realizando-o de acordo com procedimentos e técnicas previamente definidas, em uma velocidade de trabalho costumeira à um programador de habilidades medianas normalmente denominado como programador Pleno. (V. capítulo III, item 3)

Tempo Padrão - É o tempo normal com previsão de demoras e atrasos, independentes do controle do programador, ou, em outras palavras, é o tempo normal acrescido das tolerâncias pessoais, de fadiga e de espera.

Uma atividade fundamental antes de se iniciar qualuquer estudo de tempos é a de deixar bem caracterizada para a tarefa que será observada, qual é o seu início e o seu fim. No presente trabalho as durações serão sempre conside-

radas em número de dias e os conceitos adotados para início e fim do desenvolvimento de um programa são os seguintes:

Início - É o dia em que o programador recebe de algum seu superior as especificações do programa que deverá desenvolver. Em geral estas especificações estão na "pasta do programa", conforme mencionada anteriormente. (V. capítulo II, item 2.3.)

Fim - É o dia em que o programador entrega a algum seu superior a documentação e a listagem do programa sem erros com todas as evidências de que o mesmo foi devidamente testado. Todavia, o dia em que isto ocorre só deverá mesmo ser considerado como o último dia do desenvolvimento, caso até a implantação do sistema aonde esse programa se integrará, não vier a ser detectada nenhuma irregularidade que poderia ter sido evitada pelo programador. Caso ainda haja alguma necessidade de acerto ou correção, todo o tempo que vier a ser dispendido com esta atividade deverá ser adicionado à duração do programa.

Ainda com relação ao cálculo das durações do desenvolvimento dos programas, um cuidado especial deverá ser tomado, no caso de programadores que tem sob si a responsabilidade simultânea por mais de um programa. Isto porque, o tempo dispendido pelos mesmos deverá ser devidamente computado e dividido por entre esses vários programas.

2. TÉCNICAS DE MEDIDA DO TRABALHO

Como já foi dito anteriormente, a Medida do Trabalho é consubstanciada com a definição de um tempo padrão para a concretização de uma determinada tarefa.

Para se chegar aos tempos padrões das atividades, existem várias técnicas que sempre aparecem descritas na bibliografia disponível sobre Medida do Trabalho. No entanto, elas nunca deixam de ser abordadas sob o ponto de vista das operações industriais, ou tipicamente fabris. No presente capítulo, pretende-se sumariar essas técnicas, mas sob o prisma da sua possível utilização, para avaliar tempos padrões de atividades de programação de computadores.

Ao todo existem seis técnicas mais usuais, que se distribuem por 3 categorias, as quais baseando-se em Krick (27), são as seguintes:

- a - Aproveitamento da experiência anterior
 - 1. Técnica dos dados históricos
- b - Observação Direta
 - 1. Técnica da auto-avaliação
 - 2. Técnica da amostragem
 - 3. Técnica da cronometragem
- c - Síntese
 - 1. Técnica dos dados-padrão
 - 2. Técnica dos tempos predeterminados

Uma a uma, estas técnicas passam a ser descritas

²⁷ Edward V. Krick, Métodos e Sistemas, traduzido de inglês por Roberto Verdussen, Rio de Janeiro, Livros Técnicos e Científicos Editora LTDA, 1971, 2 volumes, pág. 219.

ênfatizando-se em cada uma, os aspectos positivos e negativos das mesmas com vistas a serem aplicadas em tarefas de programação de computadores.

2.1. Aproveitamento da Experiência Anterior: Esta categoria que de relevante só apresenta uma técnica, se caracteriza pelo fato de se dever confiar em registros ou lembranças de experiências anteriores semelhantes à operação (ou programa) que está sendo estudada e, dessa forma, inferir-se um tempo padrão de desempenho.

2.1.1. Técnica dos Dados Históricos: Esta técnica é a mais simples das técnicas de Medida do Trabalho e, por sinal, é a mais usada para estimar-se durações no desenvolvimento de programas de computador. Basicamente, ela envolve nada mais que o correlacionamento entre volumes de serviço e horas/homem (ou outra unidade), gastas no passado, para executá-las. Isso pode e, em geral é feito, consultando-se registros da empresa. No entanto as estimativas podem ser feitas, não consultando-se registros, mas simplesmente baseando-se no espírito prático e experiência geral sobre o assunto.

Vantagens:

1. É uma técnica fácil de ser implantada e que dispensa necessidades especiais de treinamento
2. Dados Históricos são de fácil manutenção

3. Não interfere com a rotina normal de trabalho
4. Não há necessidade de equipamentos especiais de medição. Inclusive um mínimo de anotações são realizadas.

Desvantagens:

1. Os padrões que são desenvolvidos com base em dados históricos incluem todas as ineficiências do passado, ou seja, eles mostram quanto tempo levou a execução das tarefas no passado, mas não quanto tempo deveriam ter levado. Na verdade, é o mesmo que assumir-se que os processos e métodos do passado são os melhores.
2. No caso de variações no tipo de atividade executada, como por exemplo, a metodologia empregada, deve se esperar até que novamente se disponham de dados acumulados sobre as mesmas, para se poder dispor de informações que possibilitem inferências.

2.2. Observação Direta: As técnicas enquadradas nesta categoria requerem observação direta da tarefa que está sendo executada.

2.2.1. Técnica da Auto-Avaliação: Esta é uma técnica participativa, na qual os tempos padrões são obtidos a partir de mensurações de tempo e de volumes efetuados pelos próprios executores das tarefas. Dessa forma, cada funcionário preenche um formulário pré-impresso, onde ele registra cada tarefa executada, o seu volume (se for o caso) e o tem

po dispendido. A Figura I (capítulo V, item 1.1.2), mostra um modelo deste formulário, que, por sinal, é o mesmo utilizado na fase de levantamento de dados que será mencionada posteriormente. Esta é outra técnica bem utilizada na área de programação e análise de sistemas, na qual a responsabilidade pelo levantamento das informações é atribuída aos programadores e analistas. Os tempos padrões são obtidos após alguns dias ou meses, conforme o caso, dividindo-se os tempos totais consumidos em cada tarefa pelo número de unidades da mesma que foi realizada durante o período.

Vantagens:

1. Muito fácil de ser implantada, principalmente por ser fácil de ser entendida pelos programadores e por envolvê-los, tornando mais fácil a sua aceitação.
2. Dispensa o uso de equipamentos.
3. Esta técnica é voltada para tarefas de ciclo longo, identificando-se com as tarefas ligadas ao desenvolvimento de sistemas automatizados.

Desvantagens:

1. Os padrões desenvolvidos por esta técnica mostram o tempo que tem sido gasto com as tarefas, mas não indicam o tempo que deveria ter sido gasto com as mesmas.
2. Falhas no preenchimento dos formulários costumam acontecer, tanto deliberadamente como não, na maioria dos casos devido ao fato do funcionário não registrar o evento imediatamente após o mesmo ter acontecido, fazendo-o quan

do já não se recorda mais da tarefa.

Não são feitos ajustes para considerar diferenças de ritmo/capacidade individuais.

A tabulação dos dados é trabalhosa. Imagine-se um setor com 10 programadores. Em um mês (20 dias úteis), serão gerados 200 formulários que deverão ser computados e calculados um a um.

2.2.2. Técnica da Amostragem: A técnica da amostragem foi desenvolvida para resolver problemas de Medida do Trabalho em situações heterogêneas, quando vários tipos de tarefas devem ser analisadas ao mesmo tempo, sendo ou não interdependentes entre si, que é justamente o caso que se dá nos projetos de sistemas automatizados, quando o supervisor ou gerente necessita saber com razoável precisão, o que ocorre com cada membro de suas equipes em termos de proporção de tempo que está sendo gasta em cada tipo de atividade.

A teoria da amostragem baseia-se na premissa de que uma amostra aleatória extraída de um universo maior, tenderá a comportar-se exatamente como esse universo, se a amostra extraída tiver um tamanho adequado. Extrapolando para o campo da Medida do Trabalho, a amostragem do trabalho baseia-se no fato de que o tempo gasto em trabalhos pode ser considerado como constituído de instantes individuais, nos quais um estado particular de atividade ou inatividade prevalece.

Resumindo, o método de amostragem do trabalho envolve uma estimativa de proporção do tempo dispendido em um dado tipo de atividade, em um certo período, através de observações instantâneas, efetuadas ao acaso.

Vantagens:

1. Rapidez na obtenção de informações.
2. Não interfere na rotina normal de trabalho de programadores e analistas, desde que anteriormente tenha sido feito um esclarecimento adequado aos mesmos.
3. Não requer equipamentos especiais.
4. Não requer pessoas especialmente treinadas e habilitadas para efetuar as medições, o que implica num baixo custo para esta técnica.
5. Simultaneamente, pode-se estudar vários tipos de tarefas diferentes entre si.

Desvantagens:

1. É difícil explicar a validade da técnica às pessoas que estão sendo observadas. Se bem que no caso de programadores e analistas, em função de que, muitos tiveram formação estatística nos seus cursos superiores, este problema é atenuado.
2. Em geral, esta técnica não fornece subsídios para que as tarefas observadas venham a ter seus procedimentos de execução melhorados, além disso, quando estas últimas são modificadas, novas observações devem voltar a ser feitas.

3. Muitas vezes distorções ocorrem, pois os funcionários alteram seu comportamento diante da presença de observadores. Este fato pode ser atenuado usando-se recursos especiais para cada caso.

2.2.3. Técnica de Cronometragem: Esta é, sem dúvida, a técnica mais conhecida e mais utilizada para efetuar Medida do Trabalho, em áreas fabris. Pressupõe a utilização de um cronômetro, bem como a avaliação do ritmo de trabalho do funcionário observado, para que se estabeleça o ritmo normal. No entanto, para a avaliação da atividade de desenvolvimento de sistemas que aqui é o problema central, esta técnica não apresenta nenhum ponto positivo, mas sim, sérias limitações, das quais as principais são as seguintes:

1. Esta técnica é apropriada para medir tarefas de ciclo curto, o que não é o que acontece no trabalho de programadores e analistas.
2. A necessidade de avaliação de ritmos de trabalho, característica primordial desta técnica, somente é possível de ser feita, em tarefas relacionadas com movimentos físicos do corpo humano. Isto, absolutamente, não é o caso nas tarefas que estão sendo analisadas. É absolutamente impossível estimar-se ritmos de trabalho para atividades mentais ou intelectuais. Apesar de que, como se verá adiante, correções devem ser feitas para compensar as experiências individuais dos programadores.

2.3. Síntese: As técnicas constantes nesta categoria se baseiam no fato de que é possível obter os tempos padrões das tarefas, sem necessidade de observá-las constantemente, mas através do uso de tabelas, gráficos e fórmulas.

2.3.1. Técnica dos Dados-Padrão: Também conhecida por tempos estatísticos, justamente por ser uma técnica puramente estatística, que se baseia na aplicação de regressão linear. Dessa forma, para se chegar aos tempos -padrão, é necessário inicialmente, especificar quais as variáveis que afetam significativamente o tempo normal para um determinado tipo de tarefa (como se verá adiante, tipos de tarefas seriam cada tipo de programa de computador). Em seguida, coletar os tempos gastos para um mesmo tipo de trabalhos semelhantes, e então, usar estes dados para determinar as relações entre o tempo normal e cada variável que foi significativa.

Vantagens:

1. Dispensa pessoal e equipamento especializado para efetuar medições, às quais, aliás, não existem. Não é necessário sequer ver a execução da atividade.
2. Pode corrigir erros e melhorar precisão dos tempos obtidos por outros métodos.
3. Extrema rapidez na obtenção de dados.

Desvantagens:

1. Os dados obtidos através do uso desta técnica tornam-se

inúteis tão logo haja uma modificação no método da operação, para a qual ele pode estabelecer padrões. No presente caso, seria quando, por exemplo, houvesse alguma mudança nos fatores externos que afetam o trabalho de programador, como linguagem utilizada, padrões de documentação, acessibilidade do computador para execução de testes, etc..

2. Esta técnica se baseia nos dados obtidos no passado para efetuar suas extrapolações. Portanto, se, porventura tiverem havido imperfeições nos dados coletados anteriormente, esta técnica as incorporará e assumirá nos novos dados que estão sendo obtidos desta vez, através da técnica.

2.3.2. Técnicas dos Tempos Pré-Determinados: A hipótese básica destas técnicas, é a de que todo trabalho consiste de certos movimentos básicos do corpo humano, para os quais já existem tempos mensurados e catalogados. Assim sendo, decompondo-se os trabalhos analisados nestes movimentos já medidos e catalogados, consegue-se, por síntese, a obtenção do tempo padrão daqueles trabalhos. Todavia, esta técnica é absolutamente incompatível com o tipo de tarefa realizado pelos programadores, já que o trabalho destes não envolve movimentos físicos do corpo humano, mas sim atividades mentais e intelectuais. Além disso, os tempos pré-determinados só tem aplicação válida em tarefas de ciclo curto, o que também não é o presente caso.

Tendo-se agora uma visão geral de todas as técnicas disponíveis, é possível de se discernir quais devem ser utilizadas. Esta análise é feita no próximo capítulo, no item Esforço de Programação (1.1.).

C A P Í T U L O V

DESCRIÇÃO DA METODOLOGIA PARA OBTENÇÃO DOS TEMPOS PADRÕES DAS TAREFAS DE PROGRAMAÇÃO DE COMPUTADORES. UM CASO PRÁTICO.

Este capítulo tem por objetivo, descrever como um grupo de desenvolvimento pode passar a dispor de parâmetros que lhe darão condições de estimar datas de conclusão de programas de uma forma científica, dando condições para que a produtividade dos programadores também passe a ser aferida com bastante rigor.

A descrição da metodologia está dividida em 4 partes:

Na primeira parte é descrita a forma como deve ser feito o levantamento de dados, tanto dos tempos dispendidos pelos programadores como das características e variáveis dos programas por eles desenvolvidos.

A segunda parte descreve como fazer a análise dos dados levantados, concluindo com equações de regressão para cada categoria de programa definida.

Na terceira parte se descreve como a partir daquelas equações é possível se obter os tempos padrões das tarefas, bem como efetuar as estimativas acerca das durações dos

programas e calcular os índices de eficiência.

E, finalmente, a última parte, descreve um caso real de aplicação dessa metodologia.

1. LEVANTAMENTO DE DADOS

O levantamento de dados, apesar de dever ser feito a um só tempo, apenas com o intuito de facilitar a compreensão do texto, está dividido em 2 subpartes:

- Esforço de programação, que é a determinação do esforço em mão de obra envolvido em cada programa;
- Variáveis de cada programa, que é a caracterização e quantificação das variáveis que afetam cada programa.

Realçando um esclarecimento importante: Esforço de programação e Variáveis dos programas devem ser pesquisadas a um só tempo, para que uns correspondam aos outros, mas todo este trabalho só deverá ocorrer, provavelmente, uma única vez, pois após eles terem sido levantados e correlacionados estarão disponíveis as equações necessárias para efetuar estimativas e calcular eficiências. Isso até que algum novo fator ambiental surja ou algum dos antigos fatores sejam modificados.

1.1. Esforço de Programação: Analisando as seis técnicas de Medida do Trabalho mencionadas no capítulo III, é possí-

vel de se verificar que nem todas elas se prestam a permitir a quantificação do tempo gasto por programadores em suas tarefas.

Se não, veja-se. A técnica dos dados históricos destina-se à efetuação de estimativas, pressupondo para isso a existência de informações do passado, nas quais devase basear. Não se trata do presente caso, já que a maioria dos CPDs no Brasil não tem por hábito coletar e armazenar informações do passado. A técnica da cronometragem é voltada para tarefas de ciclo curto e repetitivo, o que também não é o caso, sem contar que seria temerária a introdução de um cronômetro em um ambiente de programadores.

Com relação à técnica dos dados padrão, a sua essência será utilizada mais adiante, quando do estabelecimento das equações, que são um dos objetivos deste trabalho. No entanto, nesta altura, fase de levantamento de dados, sua utilização é inviável, pois ainda não há nada em que se basear. Finalmente a técnica dos tempos pré-determinados é a que reúne maiores motivos para ser rejeitada de imediato, já que ela é voltada para tarefas de cunho manual, o que não é absolutamente o que acontece com programação de computadores.

Restam, portanto, duas técnicas: Auto-Avaliação e Amostragem. As duas devem ser utilizadas no levantamento de dados, principalmente, com o espírito de uma complementar a outra. Aos programadores deve ser dada a incumbência de registrarem diariamente como o seu tempo é dispendido e aos

supervisores compete efetuarem as observações para se obter a amostragem do trabalho e que permitirá não só validar os dados registrados pelos programadores, mas também complementá-los com outras informações. A seguir, nos 3 sub itens, está descrito como cada uma dessas técnicas deve ser utilizada, e a maneira de validar e complementar as informações obtidas.

1.1.1. Aplicação de Amostragem do Trabalho: O principal intuito ao se aplicar esta técnica de Medida do Trabalho é o de deixar bem caracterizada a porcentagem do tempo do programador que é consumido com atividades produtivas. Inclusive, na maioria das vezes que se aplica amostragem do trabalho em outras atividades fabris ou não, costuma-se pesquisar apenas 2 valores: porcentagem do tempo produtivo ($X\%$) e porcentagem do tempo ocioso ($100 - X\%$). Maiores quebras nessas informações são possíveis de serem obtidas, mas com o risco de comprometer a operacionalidade da técnica, além de normalmente implicarem na necessidade de aumento no número de observações. Essa é a situação que também ocorre quando da aplicação da técnica para avaliar programadores de computador. Caso se consiga resolver satisfatoriamente o problema de quem fará as observações, e como as fará, pode ser interessante e válido, promover um levantamento de um maior número de estados, além de ativo e ocioso. Esse é um problema que aqui surge: Quem fará as observações? Se esse alguém for dotado de conhecimentos específicos inferior ao dos pro

gramadores, dificilmente o estudo será aceito por estes, que alegarão falta de conhecimento por parte do observador para poder caracterizar as atividades que eles estão executando.

Se for alguém de maior nível, como, por exemplo, um analista, além de se encarecer o estudo, pode-se incompartibilizar esse alguém junto aos programadores. Essas duas consequências são indesejáveis, e como se vê, o problema não é tão simples como talvez possa aparentar. A solução é entregar ao chefe de programação ou ao supervisor da equipe, a incumbência pelas observações. Também se aumentará o custo do estudo, alegarão alguns. Nem tanto, pois observar seus subordinados e saber o que eles fazem, já é obrigação de todo chefe ou supervisor. Simplesmente estes passarão a fazer as observações que de alguma maneira, já devem ser feitas, de uma forma sistematizada, para permitir um tratamento estatístico das mesmas. Evidentemente, o número de observações não poderá ser exagerado, sob o risco de comprometer as outras responsabilidades do chefe/supervisor.

A experiência tem demonstrado que para grupos de tamanho médio (8 a 12 pessoas), um supervisor consegue efetuar duas observações de cada funcionário por hora, sem chegar a causar comprometimento nas suas outras tarefas habituais. Com essa taxa se consegue chegar a 1000 observações em 3 meses (62 dias úteis).

Assim sendo, ao se aplicar amostragem do trabalho em tarefas de programação, surge a necessidade de promover uma ligeira mudança no "figurino" habitual de aplica-

ção da mesma, no sentido de que o número de observações, em geral, não poderá ser estabelecido a priori, mas sim deverá ser uma consequência da disponibilidade de tempo do encarregado pelos que estão sendo observados, da duração a decorrer do projeto que servirá de amostra e da pressa com que se pretenda conhecer os resultados. Sem esquecer, evidentemente, do erro relativo admissível para o estudo.

Decidido o número diário de observações que será efetuado, o ideal seria que o observador sorteasse os instantes em que as deverá efetuar. Para isso poderá usar as tabelas de tempos ocasionais (28).

E o que observar?

O mais simples é, simplesmente, constatar os dois estados:

1. produzindo
2. não produzindo

Mas o ideal seria um detalhamento maior abrangendo as 5 etapas da programação, mencionadas no Capítulo II, item 2.4., de forma que fossem observados os 6 seguintes estados:

1. estudando
2. codificando
3. depurando
4. testando

²⁸ Ralph Barnes, Estudo de Movimentos e de Tempos: Projeto e Medida do Trabalho, op. cit., pag. 562.

5. documentando
6. não produzindo

Evidentemente, esta última alternativa não permite que o chefe/supervisor faça as observações sentado em sua mesa, simplesmente olhando para os seus programadores, como no primeiro caso.

Ele deverá se aproximar de cada um, para constatar de fato, o que cada programador está fazendo.

A escolha de uma das duas alternativas ou de alguma outra intermediária, dependerá dos fatores já citados anteriormente, e principalmente, dos objetivos do estudo.

1.1.2. Aplicação da Técnica da Auto-Avaliação: Aos programadores deve ser dada a incumbência de preencherem um formulário semelhante ao mostrado a seguir (Figura I), onde especificarão em detalhes, quanto tempo dispenderam em cada uma de suas tarefas. Isso deve ser feito diariamente, e, ao fim do dia, o formulário deve ser entregue ao chefe/supervisor.

FIGURA I

| | | | | | |
|----------------------------|---|--------------------------|-------|--------------------------|-------|
| PROGRAMADOR: | | Dia: | | | |
| | | <input type="checkbox"/> | Manhã | <input type="checkbox"/> | Tarde |
| Programa Atividade (hs) | 1 | 2 | 3 | 4 | Total |
| Estudo | | | | | |
| Codificação | | | | | |
| Depuração | | | | | |
| Testes | | | | | |
| Documentação | | | | | |
| Sub-Total | | | | | |
| Espera | | | | | |
| Pessoal | | | | | |
| TOTAL | | | | | |

O ideal é que cada programador preencha 2 desses formulários por dia: ao final do período matutino e ao final do período vespertino. Isso pode evitar esquecimentos e possíveis distorções.

O formulário possui 4 colunas, pois esse é considerado o número máximo de programas, por mais simples que eles sejam, que um programador pode ter, simultaneamente, sob a sua responsabilidade, sem correr o risco de perder o controle dos mesmos.

Examinando-se o formulário, percebe-se que ele pos

sui 5 linhas onde são indicadas atividades produtivas e que já foram descritas anteriormente e possui 2 linhas para atividades não produtivas: "espera" e "pessoal". Na linha "espera", o programador deve ser instruído a preencher o número de horas em que ele não realizou atividade por estar esperando alguma modificação na "pasta do programa", cuja evidência tenha surgido após o início do estudo do programa, o que é comum acontecer. Ou então, ainda nessa linha, ele deverá anotar o número de horas que não teve nada para fazer por estar esperando a concessão de horas de computador para depurar ou testar seus programas, o que, entre vários outros motivos, pode ter acontecido por quebra ou avaria no computador ou em algum equipamento periférico. No caso de instalações em que o programador dispõe de terminais on-line, deve ser colocado o tempo em que ele, por exemplo, ficou na fila esperando para usar o equipamento. Enfim, nesta linha, devem ser lançados todos os tempos consumidos, com esperas totalmente independentes da vontade do programador.

Na linha "pessoal", o programador deve ser incentivado a lançar as horas que consumiu com pausas para "bate-papo" ou "cafezinhos", com "telefonemas pessoais", com necessidades fisiológicas, com pausas para, efetivamente, descansar, com atrasos para voltar do almoço, enfim, com todo tipo de atividade não produtiva, mas que ocorreu por algum outro motivo que não a "espera" nos moldes em que foi especificada acima.

É muito importante esclarecer aos programadores,

que é normal e praticamente obrigatório que, durante o dia, alguns minutos sejam dispendidos com essas atividades não produtivas. Simplesmente elas devem ficar restritas a alguns limites. Entre autores que abordaram o assunto (29) e (30) de tolerâncias pessoais e de fadiga para atividades semelhantes (as atividades de engenheiros foram as mais estudadas dentro da categoria de atividades pensantes) existe praticamente um consenso de que as mesmas devem representar cerca de 15%. Isto significa que em um dia de trabalho de 8 horas, é normal que 1,2 hs. ou 72 minutos, sejam consumidos daquela forma.

Uma das principais recomendações para que a técnica da Auto-Avaliação seja bem sucedida, é que haja obrigatoriedade, aos programadores de entregarem os formulários preenchidos ao supervisor pelo menos uma vez por dia. Este, por sua vez, deverá proceder às tabulações dos valores, diariamente, para isso, utilizando um mapa com o mesmo formato básico do formulário utilizado pelos programadores, somente que com um maior número de colunas: uma para cada programa que estiver sendo observado.

1.1.3. Determinação do Esforço Havido. Correções: Por mais esclarecidos que estejam os programadores, a experiên-

²⁹ Harvey Smerilson, "Standars for Engineers", Industrial Engineering, vol. 7, nº 10 : 12-17, Outubro 75.

³⁰ William S. Donelson, "Project Planning and Control", Datamation, vol. 22, nº 6 : 73-80, Junho 76.

cia comprova que eles tendem a subestimar, quando preenchem o formulário de auto registro, as horas consumidas em atividades não produtivas e normalmente registram um número de horas trabalhadas bem superior ao realmente havido (em alguns casos até 50% a mais). A comprovação desse fato é facilmente obtível pela aplicação simultânea de amostragem do trabalho. E também, caso se aplique esta última técnica com nível de detalhamento maior (os mesmos estados mencionados no formulário de auto-avaliação, Figura I) se constatará que as horas gastas em cada atividade produtiva obtidas pela amostragem, serão inferiores aos mostrados nos formulários de auto-avaliação, mas as proporções relativas são praticamente as mesmas.

Exemplificando uma situação típica, suponha-se um programador observado durante um mês a quem foi solicitado também que se auto-avaliasse (os dados são reais e a amostragem envolveu 504 observações).

| | Dados obtidos p/ auto-avaliação | | Dados obtidos p/ amostragem | |
|--------------|------------------------------------|-----------|--------------------------------|-----------|
| | hs | %s /Total | hs | %s /Total |
| Estudo | 29,0 | 17 | 24,0 | 14 |
| Codificação | 34,0 | 20 | 31,0 | 18 |
| Depuração | 18,0 | 11 | 15,0 | 9 |
| Testes | 45,0 | 27 | 39,0 | 23 |
| Documentação | 12,0 | 7 | 9,5 | 6 |
| Sub Total | 138,0 | 82 | 118,5 | 70 |
| Espera | 13,0 | 8 | 9,5 | 6 |
| Pessoal | 17,0 | 10 | 40,0 | 24 |
| TOTAL | 168,0 | 100 | 168,0 | 100 |

Analisando-se o quadro acima, é fácil de se constatar diferenças significativas entre os dados obtidos por um e pelo outro processo. O mais importante deles, que seria o sub-total de horas produtivas mostra uma diferença de 16% ($138/118,5 \times 100$) a mais nos valores obtidos pela auto-avaliação.

Agora analise-se o seguinte quadro que realça, para os mesmos dados, apenas a distribuição relativa das horas produtivas:

| | Dados obtidos p/ auto-avaliação | | Dados obtidos p/ amostragem | |
|--------------|------------------------------------|-----------|--------------------------------|-----------|
| | hs | %s /Total | hs | %s /Total |
| Estudo | 29 | 21 | 24 | 20 |
| Codificação | 34 | 25 | 31 | 26 |
| Depuração | 18 | 13 | 15 | 13 |
| Testes | 45 | 32 | 39 | 33 |
| Documentação | 12 | 9 | 9,5 | 8 |
| Sub Total | 138 | 100 | 118,5 | 100 |

Percebe-se que a distribuição relativa das horas dispendidas com as atividades produtivas é praticamente a mesma como resultado de qualquer dos processos de levantamento de dados. Portanto, o que varia basicamente em cada um dos processos é a informação de total de horas produtivas, bem como, e conseqüentemente, as informações sobre tempo pessoal e tempo com esperas. Apesar de que a experiência foi realizada com um único programador (por sinal aquele que foi

escolhido o representante médio do setor) e apenas durante um mês, os dados obtidos foram praticamente os esperados e, além disso, posteriormente, quando foi levado a cabo um estudo mais profundo, só fizeram por confirmar essas tendências.

Assim sendo, a conclusão que se chegou a partir desta experiência, é que as duas técnicas seriam conjugadas em 2 etapas simultaneamente aplicadas:

1. Da amostragem devem ser extraídos 3 estados. Tempo pessoal, tempo de espera e tempo produtivo. Este último, deve ser observado na sua forma bruta, não sendo necessário descer a níveis de detalhe, tais como tipo de tarefa ou programa específico que esteja sendo executado.
2. Dos registros da auto-avaliação, deve se obter a distribuição relativa das horas produtivas, cujo total é obtido simultaneamente pela técnica da amostragem. Além disso, é a partir das informações fornecidas pela auto-avaliação, que se obtém também como o tempo de cada programa se distribui por entre os vários programas que o mesmo tem sob a sua responsabilidade.

Cumpridas as 2 etapas e tabulando-se os dados mensurados estará obtida a informação mais importante: o tempo efetivamente consumido com o desenvolvimento de cada programa.

Nesta hora, deve se passar a considerar aquilo que na fábrica seria chamado de avaliação de ritmo para obten-

ção de tempos normais.

Na área de programação de computadores, a correção do ritmo individual da pessoa que está sendo observada deve ser feita com base na experiência da mesma, conforme foi inicialmente abordado no item Experiência do Programador, no capítulo III, item 3. Nesse item inclusive, está descrita uma experiência através da qual se constatou que atribuindo-se o valor 1,0 (hum) para a velocidade média de trabalho de um programador pleno, se inferiu as seguintes faixas de velocidade para as outras modalidades:

| | |
|--------------------|-----------|
| Programador Junior | 0,4 a 0,8 |
| Programador Senior | 1,1 a 1,5 |

Em função desses dados, na presente metodologia, a recomendação é que sejam utilizados os seguintes fatores de correção de ritmo:

para programadores juniors: 0,6

para programadores plenos : 1,0

para programadores seniors: 1,3

Todavia, é importante salientar que esses valores só poderão ser utilizados por empresas que também agreguem os seus programadores nas 3 categorias acima, e, principalmente, que aquilo que a empresa entende por cada um dos cargos se assemelhe ao descrito no capítulo III, supramencionado.

Caso contrário, a empresa deverá escolher vários programadores de cada categoria que possui e entregar a eles a realização do mesmo programa, procedendo conforme aquela

experiência já descrita anteriormente.

1.2. Quantificação das Variáveis de Cada Programa: Este assunto também já foi abordado anteriormente, quando se salientou que na tarefa de programação, atuam 3 categorias de fatores:

1. Ambientais
2. Variáveis Inerentes a Cada Programa
3. Experiência do Programador

Os ambientais não devem ser aqui considerados, pois esta é uma metodologia a qual visa poder ser aplicada em qualquer CPD, mas os resultados obtidos em um CPD raramente poderão ser transplantados para outro, justamente devido ao fato da influência dos fatores ambientais, que não são levados em conta pela aplicação da metodologia em si. A influência dos fatores ambientais é analisada de forma ampla por Felix & Walston (31).

Sobre como considerar a experiência do programador, o terceiro conjunto de fatores, se falará mais adiante, já que este é um fator que afeta a tarefa de programação, mas não o programa em si. Este, evidentemente, não mudará o seu grau de complexidade, em função do indivíduo que o estará desenvolvendo.

³¹C. Félix e C. Walston, "A method of programming measurement and estimation", IBM Systems Journal, vol. 16, nº 1 : 54-73, 1977.

Portanto, no momento, serão analisados apenas os fatores diretamente relacionados com cada programa, ou seja, aqueles da segunda categoria: Variáveis Inerentes a Cada Programa.

Para que se possa aplicar a presente metodologia, deve ser lembrado que ela pressupõe uma abordagem fundamental por parte do CPD que pretenda utilizá-la: A orientação no sentido de sempre desenvolver sistemas que contenham programas simples com o menor número possível de funções em cada um. Isto significa um maior número de programas por sistemas. O que não quer dizer que o esforço global para desenvolver o sistema virá a ser maior, já que o número de funções será sempre o mesmo, adotando-se ou não esta orientação. Simplesmente estas passam a estar pulverizadas por um número maior de programas menores. O grande benefício desta abordagem se faz sentir quando, após a implantação do sistema, começam a aparecer as manutenções, as quais nunca deixarão de existir, pela própria dinâmica das empresas que requerem que os seus sistemas automatizados, já operacionais, acompanhem-nas modificando-se.

Definido, então, o ambiente básico em que os programas são desenvolvidos, pode-se passar a verificar suas características intrínsecas.

A certeza acerca das características de um programa só se obtém mesmo, quando o programa está pronto, e, às vezes, só depois que o sistema também já está implantado, porque não é raro aparecerem necessidades de alterações nas

características dos programas, entre o momento em que eles já estão testados unitariamente e o instante em que o sistema se torna operacional.

No entanto, muito antes disso, às vezes na fase de especificações funcionais, sempre na fase de projeto detalhado, os analistas já tem definido quantos programas o sistema conterá e quais as funções e características internas a cada um. Como já salientado antes, a presente metodologia visa atuar nessa hora. Além disso ela visa ser acima de tudo, facilmente operacionalizável, para incentivar os gerentes a usarem-na. E para usá-la, basta ter à mão o fluxo automatizado, onde constem todos os programas do sistema e também uma descrição de todos os arquivos com os diferentes formatos de registros que cada um agrega. Não necessariamente, portanto, as descrições de programa devem estar concluídas. Isto permite que os administradores do projeto possam saber de quantos programadores necessitarão, alguns meses (no mínimo 1 mês para projetos pequenos e até 6 meses para projetos grandes, em geral) antes de terem que estar com os mesmos já incorporados em suas equipes. Ou seja, terão o tempo que durar a confecção das descrições de programas, podendo fazer as estimativas necessárias já na primeira metade da Fase III do ciclo de vida do sistema. (V. capítulo II, item 2.3.)

De posse então daquelas informações, deverão proceder a 2 passos:

1º Passo - Quantificar, para cada programa, os dados de ar-

quivos e registros.

Mais precisamente os seguintes, já explicados no capítulo III, item 2.2. - Características das Entradas e Saídas dos Programas:

1. Número de Arquivos de Entrada
2. Número de Arquivos de Saída
3. Número de tipos diferentes de relatórios
4. Somatória dos parâmetros 1, 2 e 3
5. Número de diferentes tipos de registros de entrada
6. Número de diferentes tipos de registros de saída
7. Somatória dos parâmetros 5 e 6
8. Número de diferentes tipos de registros manuseados

2º Passo - Enquadrar cada programa em uma das 5 categorias definidas

E que são as seguintes:

1. Atualização/Manutenção de Arquivos
2. Consistência/Validação de Campos
3. Impressão de Relatórios
4. Organização de Arquivos
5. Formação de Registros

Aqui se faz necessário um esclarecimento importante: a experiência mostra que no caso de projetos desenvolvidos sob a orientação já exposta (pulverizar as funções por um número grande de programas relativamente pequenos) pelo menos 80% dos programas podem ser enquadrados nas cinco categorias de programas definidos. Os programas que ficam de

fora, normalmente são programas que abrangem funções totalmente particulares aos sistemas em que se situam, variando o seu grau de complexidade, desde programas muito simples (que demandam, por exemplo, 5 dias de um programador junior) até programas muito complexos (40 dias de um programador senior, por exemplo).

No caso de centros de processamento de dados, em que a ocorrência de alguma modalidade de programa que não uma das 5 mencionadas, seja frequente, recomenda-se que essa modalidade seja definida e dados acerca dela passem a ser levantados, para que então, essa nova modalidade de categoria possa também integrar a presente metodologia.

Com esse esclarecimento, fica encerrada a etapa de levantamento de dados.

2. ANÁLISE ESTATÍSTICA DOS DADOS LEVANTADOS E OBTENÇÃO DO TEMPO NORMAL

Separados os programas em cada uma das categorias definidas, deve-se tabular os dados utilizando-se um formulário igual ou semelhante ao mostrado das páginas 123 até 127. Utiliza-se uma (ou mais) dessas folhas para cada categoria, evitando-se colocar programas de categorias distintas na mesma folha.

A análise estatística é feita em três passos explicados adiante:

a. Análise dos dados brutos

b. Análise de correlação

c. Análise de regressão propriamente dita

Apesar de que os passos b e c possam se dar através de cálculos efetuados manualmente, recomenda-se que eles sejam efetuados através do uso de computador, principalmente para evitar a realização de cálculos, não raro enfadonhos, que podem por isso mesmo redundar em erros.

2.1. Análise dos Dados Brutos: Cada modalidade de programa deve ter os seus dados analisados individualmente. Nesta altura deve se tentar descobrir alguma relação entre os valores que simplifique as etapas posteriores.

Por exemplo:

- Pode acontecer que uma das colunas de variáveis apresente sempre o mesmo valor, ou então pequenas variações. Isto significa que determinada variável é constante, não afetando a duração dos programas. Evidentemente, esta variável poderá ser desprezada para os próximos passos.
- É muito comum que algumas variáveis estejam altamente correlacionadas entre si, a tal ponto que seja dispensável efetuar os respectivos cálculos de coeficientes de correlação. Nesse caso escolhe-se uma das variáveis para continuar no estudo, desprezando as outras que com ela estão altamente correlacionadas. Evidentemente, será escolhida a que contiver maior explicação lógica.

Ao analisar os dados brutos, é importante não es-

quecer que o valor representativo da duração do programa, última coluna à direita daquele formulário, já deve estar corrigido do fator de ritmo.

2.2. Análise da Correlação das Variáveis com o Esforço em Dias: Cada variável que permanecer após as eliminações da etapa anterior, deve ter calculado o respectivo coeficiente de correlação entre ela e os dados da coluna de duração do programa. Nos casos em que os coeficientes de correlação forem baixos, a variável deixará de ser considerada como válida para figurar em uma equação de regressão. Nesta etapa, também devem ser verificados as variáveis independentes que são dependentes entre si. Nesses casos somente uma delas deve permanecer, a qual será a que tiver a maior correlação com a variável dependente, ou seja, o esforço em dias.

2.3. Análise de Regressão Propriamente Dita: Definidas as variáveis que devem ser consideradas em cada caso, resta calcular para cada categoria de programa a respectiva equação de regressão.

Não é objetivo deste trabalho enumerar os passos necessários para se calcular as equações de regressão ou os coeficientes de correlação, mesmo porque isso seria mera cópia da bibliografia existente (31) e (32).

³¹Norman Draper, Applied Regression Analysis, New York, Wiley, 1966.

Todavia, e como já foi realçado anteriormente, as equações devem ser calculadas através do uso de um computador.

Nessa circunstância, caso a instalação onde serão efetuadas esses cálculos, não contar com um "package" de análises estatísticas, o mesmo poderá ser comprado ou facilmente alugado. No caso prático que será descrito adiante, foi utilizado um "package" de propriedade do Centro de Processamento de Dados da Escola de Administração de Empresas de São Paulo da Fundação Getúlio Vargas. Esse "package", bem como a grande maioria deles, calcula as equações de regressão múltipla utilizando o procedimento Stepwise o qual em resumo consiste das seguintes etapas:

1. É calculada uma matriz que dá os coeficiente de correlação simples entre todas as variáveis envolvidas, dependentes ou independentes.
2. A variável independente que apresentar o mais alto coeficiente de correlação simples com a variável dependente é escolhida no primeiro passo e dessa forma surge a primeira equação de regressão, evidentemente em função de uma única variável independente.
3. No próximo passo são analisadas todas as variáveis não escolhidas no passo anterior para se decidir qual será a próxima a ser considerada. Escolhendo-se aquela que tiver o mais alto coeficiente de correlação parcial com a

³³Karl Fox & William Merrill, Estatística Econômica: Uma Introdução, trazido do inglês por Alfredo Alves de Faria, São Paulo, Editora Atlas, 1977.

variável dependente. O produto final deste passo é uma equação de regressão com duas variáveis independentes.

4. Da mesma forma que anteriormente, é calculado o coeficiente de correlação parcial entre cada variável remanescente (que ainda não entrou na equação) e a variável dependente, escolhendo-se a que contiver o valor mais alto.
5. Esses passos seguem sem interrupção, produzindo-se a cada novo passo uma equação com mais uma variável, até que em um determinado passo o coeficiente de determinação múltipla da equação que seria resultante deste passo, forme-se maior que o coeficiente de determinação múltipla da equação que havia resultado no passo imediatamente anterior, a qual fica sendo então a última equação definida pelo processo.

3. EFETUAÇÃO DE ESTIMATIVAS E CÁLCULO DOS ÍNDICES DE EFICIÊNCIA

O objetivo básico da metodologia apresentada é o de permitir, através do uso das fórmulas/equações obtidas pela análise de regressão, que sejam efetuadas estimativas precisas acerca das durações que devem ter os desenvolvimentos dos programas, bem como também permitir com que a produtividade dos programadores incumbidos pelos desenvolvimentos possa ser aferida. No entanto, a duração que se obtém pela aplicação das fórmulas é aquela relativa ao tempo normal, res-
tanto a necessidade de serem efetuados alguns ajustes para

que sejam obtidos os respectivos tempos-padrões.

Esses ajustes referem-se às tolerâncias que devem ser considerados em estudos de tempo e são enquadrados em três categorias:

- Tolerâncias pessoais, de fadiga e de espera.

Conforme já mencionado neste mesmo capítulo (v. item 1.1.2.) as tolerâncias pessoais e de fadiga, somadas, situam-se em cerca de 15% (34) e (35). Com relação às tolerâncias de espera, o ideal seria que fosse possível lhes atribuir o valor 0 (zero). Mas isso não é possível já que esperas independentes da vontade do programador ocorrem em todas as instalações.

Portanto, o valor mais conveniente a ser utilizado é aquele que resultou do estudo de amostragem, além de que neste caso generalizações não são permitidas. Estimadas as tolerâncias pode se partir para o cálculo do tempo padrão o qual é obtido da seguinte forma:

$$T_{\text{PADRÃO}} = T_{\text{NORMAL}} + (\text{TOLERÂNCIAS}) \times T_{\text{NORMAL}}$$

onde

T_{NORMAL} , é a fórmula/equação obtida pela análise de regressão.

³⁴Harvey Smerilson, "Standars for Engineers", Industrial Engineering, op. cit.

³⁵William S. Donelson, "Project Planning and Control", Datamation, op. cit.

Após a obtenção das fórmulas que dão o tempo padrão para cada modalidade de programa, para se chegar as estimativas de duração dos programas restam tarefas puramente operacionais que podem ser cumpridas por elementos que possuam as qualificações de um programador junior. O conhecimento de programação, se bem que em nível apenas rudimentar, é justificado devido a necessidade de interpretação dos fluxos de programas, no intuito de caracterizar-se e quantificar-se as variáveis em cada programa, o qual deve ter sua duração estimada. Estas tarefas estão descritas nos passos seguintes:

1. Enquadrar o programa em uma das categorias definidas, escolhendo a respectiva fórmula/equação de regressão.
2. Verificar quais os tipos de variáveis que a mencionada equação considera, examinando a descrição de programa com o objetivo de quantificar a cada uma.
3. Entrar com os valores de cada uma das variáveis quantificadas na fórmula que estiver sendo utilizada, obtendo o tempo normal para a duração do desenvolvimento do programa.
4. Aplicar as taxas de tolerâncias sobre o tempo normal obtendo o respectivo tempo padrão.
5. Caso seja possível, nesta altura, caracterizar que espécie de programador (júnior, pleno, senior) será responsável pelo programa, então dever-se-á efetuar os devidos ajustes sobre o tempo padrão obtido no passo anterior, pa

ra levar este fato em consideração.

Para isso deverão ser conhecidos os fatores utilizados os quais correspondem aos ritmos de cada modalidade de programador conforme explicado neste capítulo, no item 1.1.3. Multiplicando o tempo padrão por este fator de ritmo estará obtida a estimativa da duração do desenvolvimento do programa a qual denominamos tempo padrão ajustado.

Feita a estimativa da duração do desenvolvimento do programa, estar-se-á em condições de, ao fim do mesmo, a ferir-se a produtividade havida. Isto é feito através do cálculo do seguinte quociente:

$$\text{ÍNDICE DE EFICIÊNCIA} = \left[\frac{\text{DIAS GASTOS PARA CONCLUIR O PROGRAMA}}{\text{TEMPO PADRÃO AJUSTADO (EM DIAS)}} \right]^{-1}$$

Evidentemente, a fórmula acima que dá a eficiência de um programador desenvolvendo um único programa pode ser estendida para outros casos, inclusive para o cálculo de eficiência de equipes completas de programadores.

4. UM CASO PRÁTICO

Os dados aqui apresentados são reais e foram coletados em um CPD de grande porte que atua para várias empresas da área financeira. Trata-se de um CPD cujo grupo de desenvolvimento optou pela estrutura aplicacional nos moldes já descritos no capítulo II. Ao todo as observações referem

-se a um período pouco superior a dois anos. Foi escolhida uma equipe de analistas e programadores cujo tamanho médio, foi de 10 (dez) pessoas, dos quais 8 (oito) eram programadores. Essa equipe nesse período produziu três sistemas que ao todo resultaram 120 programas com um esforço envolvido de 15,7 programadores/ano. Destes 120 programas, foram utilizados na amostra da pesquisa 109 programas, ou seja, 91% do número de programas desenvolvidos. Esses 109 programas envolveram um esforço de 14,4 programadores/ano o que significa 92% do esforço total de programação havido. Os programas desprezados da amostra original, o foram por três motivos principais:

1. Houve variação de algum fator ambiental, principalmente não utilização de programação estruturada.
2. Programas que não se enquadravam em nenhuma das cinco categorias de programas previamente definidas.
3. No caso de existirem dois programas semelhantes entre si, tendo o mesmo programador ficado incumbido dos dois, apenas o primeiro deles entrou na amostra. Isso para evitar possíveis influências da lei do conhecimento.

Com relação aos projetos em si, sempre que possível se procurou desenvolver programas simples e com poucas funções cada um, o que evidentemente acabou por aumentar o número de programas mas simplificou enormemente as tarefas de manutenções que se fizeram necessárias após as implantações. Todos os programas foram desenvolvidos para IBM 370/

145 em DOS/VS e a acessibilidade desse equipamento pode ser considerada como boa, já que o mesmo atuando em multiprocessamento quase sempre reservava uma partição para fins de depuração e testes de programas.

Além desse aspecto, todos os programadores sempre tinham responsabilidade simultânea por no mínimo dois programas, de tal forma que, se por algum motivo, um dos programas se encontrasse em um estágio de teste, por exemplo, e, conseqüentemente, pudesse ficar na dependência de disponibilidade de computador, havia o outro ou outros programas para o programador se ocupar. Os padrões de documentação da empresa, apesar de serem muito rigorosos, não chegaram a demandar muito tempo dos programadores individualmente, já que por orientação gerencial estes preenchiam apenas alguns itens mínimos da documentação necessária e transmitiam oralmente informações básicas para um programador que assim centralizava o trabalho de documentação. No decorrer dos meses da pesquisa, também melhorou muito o padrão das descrições de programas, de tal forma que, se no início houve necessidade de grandes interações entre analistas e programadores, com o decorrer do tempo elas foram gradativamente se reduzindo, apesar de nunca terem chegado a deixar de existir. Esses contatos eram classificados para fins da amostragem do trabalho na categoria "estudo". Aliás, aos programadores sempre foi concedido o benefício da dúvida nos casos de incerteza, em que as observações não permitiam visualizar com clareza se se tratava de tempo produtivo ou improdutivo.

Por exemplo, se um programador era observado conversando, admitia-se que a conversa era sobre problemas técnicos de suas áreas, a não ser que fosse evidente que a mesma era sobre assuntos pessoais.

Todos os quadros e tabelas resultantes do levantamento de dados estão mostrados desde a página 120 até a página 132.

A partir da página 133 são mostrados os resultados da análise estatística onde aparecem para cada um dos cinco tipos de programas definidos, a respectiva equação de regressão. Notar-se-á que todas as equações possuem apenas duas variáveis independentes; isto é proposital pois, apesar de em alguns casos o processo stepwise utilizado para calculá-las ter indicado até cinco variáveis para figurarem na equação, decidiu-se restringir a escolha das variáveis apenas até o segundo passo do stepwise, em todas as categorias de programas. Isto foi feito principalmente com o intuito de se obter equações facilmente manipuláveis e portanto de fácil operacionalização, considerando sobretudo que a partir do 3º passo do stepwise os coeficientes de determinação múltipla que vinham sendo obtidos eram muito poucos superiores aqueles obtidos no segundo passo. Apenas para elucidar melhor esse aspecto, são os seguintes os coeficientes de determinação múltipla que foram obtidas para as equações de regressão, relativas à categoria de "programas de consistência".

- Após o 1º passo : $R^2 = 0,79$

- Após o 2º passo : $R^2 = 0,87$
- Após o 3º passo : $R^2 = 0,89$
- Após o 4º passo : $R^2 = 0,90$
- Após o 5º passo : $R^2 = 0,91$.

Apesar de que em termos puramente estatísticos a 5a. equação seja mais poderosa que a 2a. equação, preferiu-se ficar com esta, pois além dos motivos já alegados, a medida que um maior número de variáveis vai sendo colocado na equação esta vai perdendo seu poder lógico de explicar o fenômeno a que se destina. Entre outras agravantes este último aspecto tem o inconveniente de tornar difícil a aceitação da validade da equação por parte das pessoas que não dominam o instrumental teórico estatístico envolvido, e que não raro, por este último fato, acabam dificultando a implantação da metodologia como um todo.

A seguir são mostrados todos os valores nas fases de levantamento de dados e análise estatística, sendo que no anexo aparecem para o caso específico de programas listadores, a título de exemplo, cópias das listagens referentes a todos os passos do stepwise.

É a seguinte a relação de quadros e tabelas que aparecem nas páginas seguintes:

- | | |
|--------------|---|
| Página 120 : | Resultados obtidos com a aplicação de amostragem do trabalho. |
| Página 121 : | Distribuição relativa do tempo útil dos programadores. |

- Página 122 : Características Globais da Amostra de 109 programas utilizados na pesquisa.
- Pág. 123 à Pág. 127 : Características das variáveis (obtidas das descrições de programas) de cada um dos programas das 5 categorias estudadas.
- Pág. 128 à Pág. 132 : Matriz dos coeficientes de correlação simples entre as variáveis envolvidas para cada uma das categorias estudadas.
- Página 133 : Resultados da Análise de Regressão : Equações para cada tipo de programa dando os tempos normais.
- Página 134 : Equações para obtenção dos tempos-padrão para cada modalidade de programa.

RESULTADOS OBTIDOS COM A APLICAÇÃO DE AMOSTRAGEM

| | Nº de OBS: | % |
|------------|---------------|-------|
| PRODUZINDO | 912 | 69,4* |
| ESPERANDO | 150 | 11,4 |
| INATIVO | 252 | 19,2 |
| TOTAL | 1.314 | 100,0 |

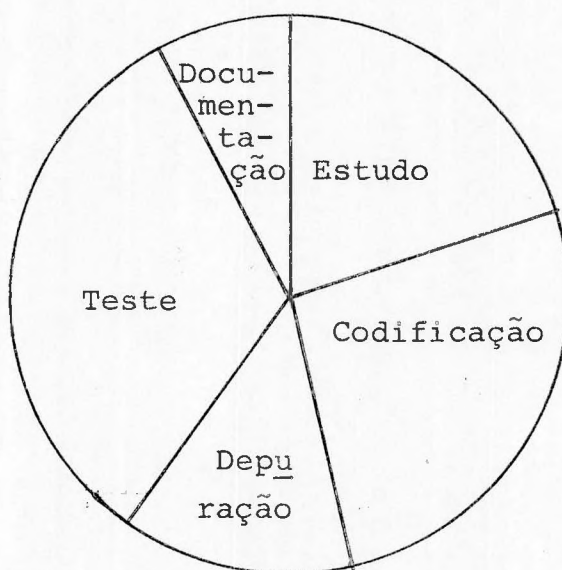
* erro relativo de 3,7%, para um nível de confiança de 95% (conforme obtido utilizando-se a tabela 64 do livro Estudo de Movimentos e de Tempos, de Ralph Barnes (36)).



³⁶ Ralph Barnes, Estudo de Movimentos e de Tempos: Projeto e Medida do Trabalho, op. cit.

DISTRIBUIÇÃO RELATIVA DO TEMPOPRODUTIVO DOS PROGRAMADORES (*)(CONFORME OBTIDO ATRAVÉS DOS FORMULÁRIOS DE AUTO-AVALIAÇÃO)

| FASE | % |
|--------------|------|
| ESTUDO | 19,7 |
| CODIFICAÇÃO | 26,4 |
| DEPURAÇÃO | 13,3 |
| TESTE | 32,6 |
| DOCUMENTAÇÃO | 8,0 |



(*) Computadas 7.744 horas de trabalho

CARACTERÍSTICAS GLOBAIS DA AMOSTRA DE 109 PROGRAMASUTILIZADOS NA PESQUISA

| CATEGORIA | Nº DE PROGRAMAS | DURAÇÃO MÉDIA (*) |
|------------------|-----------------|-------------------|
| 1. Atualizadores | 17 | 35,8 |
| 2. Consistência | 18 | 27,1 |
| 3. Listadores | 27 | 20,5 |
| 4. Organizadores | 25 | 12,5 |
| 5. Formatadores | 22 | 10,6 |
| TOTAL | 109 | 20,1 |

(*) Já corrigida pelo fator de ritmo, ou seja, ajustados pela experiência do programador. Unidade: dias.

CARACTERÍSTICAS DAS VARIÁVEIS DE CADA UM DOS

PROGRAMAS DA CATEGORIA ATUALIZADORES

| VAR. PROG. | X1 | X2 | X3 | X4 | X5 | X6 | X7 | X8 | DURA ÇÃO |
|---------------|----|----|----|----|----|----|----|----|-------------|
| A-1 | 2 | 1 | 1 | 4 | 4 | 3 | 7 | 6 | 25 |
| A-2 | 2 | 1 | 1 | 4 | 4 | 3 | 7 | 6 | 22 |
| A-3 | 2 | 1 | 1 | 4 | 5 | 3 | 8 | 5 | 18 |
| A-4 | 2 | 2 | 2 | 6 | 4 | 5 | 9 | 6 | 30 |
| A-5 | 2 | 1 | 1 | 4 | 2 | 4 | 6 | 4 | 19 |
| A-6 | 2 | 1 | 1 | 4 | 4 | 2 | 6 | 3 | 21 |
| A-7 | 2 | 1 | 3 | 6 | 16 | 8 | 24 | 22 | 79 |
| A-8 | 2 | 2 | 1 | 5 | 3 | 5 | 8 | 6 | 20 |
| A-9 | 2 | 1 | 2 | 5 | 6 | 14 | 20 | 20 | 59 |
| A-10 | 2 | 1 | 2 | 5 | 12 | 5 | 17 | 15 | 67 |
| A-11 | 2 | 1 | 1 | 4 | 6 | 4 | 10 | 6 | 23 |
| A-12 | 2 | 1 | 1 | 4 | 2 | 4 | 6 | 4 | 17 |
| A-13 | 2 | 1 | 1 | 4 | 7 | 6 | 13 | 12 | 49 |
| A-14 | 2 | 1 | 1 | 4 | 3 | 9 | 12 | 11 | 39 |
| A-15 | 2 | 1 | 2 | 5 | 4 | 4 | 8 | 7 | 34 |
| A-16 | 2 | 2 | 1 | 5 | 5 | 4 | 9 | 9 | 37 |
| A-17 | 2 | 1 | 4 | 7 | 6 | 6 | 12 | 12 | 50 |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |

X1 = Nº de Arquivos de Entrada

X2 = Nº de Arquivos de Saída

X3 = Nº de Tipos de Relatórios

X4 = X1 + X2 + X3

X5 = Nº de Registros de Entrada

X6 = Nº de Registros de Saída

X7 = X5 + X6

X8 = Nº de diferentes tipos de Registros

CARACTERÍSTICAS DAS VARIÁVEIS DE CADA UM DOS PROGRAMAS

DA CATEGORIA CONSISTÊNCIA

| VAR. PROGR. | X1 | X2 | X3 | X4 | X5 | X6 | X7 | X8 | DURAÇÃO |
|----------------|----|----|----|----|----|----|----|----|---------|
| C-1 | 1 | 1 | 1 | 3 | 3 | 2 | 5 | 5 | 17 |
| C-2 | 3 | 1 | 1 | 5 | 6 | 6 | 12 | 10 | 21 |
| C-3 | 4 | 1 | 1 | 6 | 8 | 5 | 13 | 8 | 27 |
| C-4 | 3 | 1 | 1 | 5 | 5 | 4 | 9 | 8 | 19 |
| C-5 | 4 | 1 | 1 | 6 | 4 | 1 | 5 | 5 | 22 |
| C-6 | 5 | 1 | 1 | 7 | 15 | 8 | 23 | 23 | 39 |
| C-7 | 4 | 1 | 2 | 7 | 19 | 19 | 38 | 22 | 58 |
| C-8 | 1 | 1 | 1 | 3 | 4 | 4 | 8 | 6 | 24 |
| C-9 | 2 | 1 | 1 | 4 | 5 | 3 | 8 | 4 | 19 |
| C-10 | 1 | 1 | 1 | 3 | 3 | 3 | 6 | 6 | 13 |
| C-11 | 1 | 1 | 1 | 3 | 12 | 9 | 21 | 12 | 16 |
| C-12 | 3 | 1 | 1 | 5 | 4 | 3 | 7 | 5 | 17 |
| C-13 | 3 | 1 | 1 | 5 | 5 | 3 | 8 | 8 | 28 |
| C-14 | 1 | 1 | 1 | 3 | 4 | 4 | 8 | 7 | 18 |
| C-15 | 6 | 2 | 1 | 9 | 20 | 11 | 31 | 28 | 49 |
| C-16 | 2 | 1 | 1 | 4 | 5 | 4 | 9 | 7 | 23 |
| C-17 | 4 | 1 | 2 | 7 | 15 | 9 | 24 | 21 | 43 |
| C-18 | 3 | 1 | 1 | 5 | 11 | 11 | 22 | 18 | 34 |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |

X1 = Nº de Arquivos de Entrada X5 = Nº de Registros de Entrada
 X2 = Nº de Arquivos de Saída X6 = Nº de Registro de Saída
 X3 = Nº de Tipos de Relatórios X7 = X5 + X6
 X4 = X1 + X2 + X3 X8 = Nº de diferentes tipos de Registros

CARACTERÍSTICAS DAS VARIÁVEIS DE CADA UM DOS PROGRAMAS
DA CATEGORIA LISTADORES

| VAR. PROGR. | X1 | X2 | X3 | X4 | X5 | X6 | X7 | X8 | DURA ÇÃO |
|----------------|----|----|----|----|----|----|----|----|-------------|
| L-1 | 1 | 0 | 1 | 2 | 4 | 1 | 5 | 5 | 17 |
| L-2 | 4 | 1 | 2 | 7 | 6 | 3 | 9 | 8 | 28 |
| L-3 | 5 | 0 | 2 | 7 | 6 | 2 | 8 | 8 | 36 |
| L-4 | 4 | 2 | 1 | 7 | 8 | 4 | 12 | 9 | 39 |
| L-5 | 2 | 1 | 1 | 4 | 7 | 2 | 9 | 8 | 26 |
| L-6 | 1 | 0 | 1 | 2 | 1 | 1 | 2 | 2 | 14 |
| L-7 | 1 | 0 | 1 | 2 | 1 | 1 | 2 | 2 | 7 |
| L-8 | 1 | 0 | 1 | 2 | 1 | 1 | 2 | 2 | 9 |
| L-9 | 1 | 0 | 1 | 2 | 2 | 1 | 3 | 3 | 14 |
| L-10 | 1 | 1 | 1 | 3 | 2 | 2 | 4 | 3 | 15 |
| L-11 | 1 | 0 | 1 | 2 | 1 | 1 | 2 | 2 | 7 |
| L-12 | 1 | 0 | 1 | 2 | 2 | 2 | 4 | 3 | 15 |
| L-13 | 2 | 1 | 1 | 4 | 3 | 2 | 5 | 4 | 18 |
| L-14 | 3 | 0 | 1 | 4 | 5 | 1 | 6 | 6 | 23 |
| L-15 | 3 | 1 | 1 | 5 | 3 | 3 | 6 | 4 | 31 |
| L-16 | 3 | 0 | 1 | 4 | 3 | 3 | 6 | 4 | 32 |
| L-17 | 1 | 0 | 1 | 2 | 1 | 1 | 2 | 2 | 13 |
| L-18 | 1 | 0 | 1 | 2 | 4 | 1 | 5 | 5 | 13 |
| L-19 | 1 | 0 | 1 | 2 | 1 | 1 | 2 | 2 | 10 |
| L-20 | 5 | 1 | 2 | 8 | 9 | 4 | 13 | 12 | 42 |
| L-21 | 2 | 1 | 1 | 4 | 3 | 2 | 5 | 4 | 18 |
| L-22 | 3 | 2 | 1 | 6 | 5 | 3 | 8 | 8 | 26 |
| L-23 | 1 | 0 | 1 | 2 | 1 | 1 | 2 | 2 | 10 |
| L-24 | 2 | 0 | 1 | 3 | 3 | 1 | 4 | 4 | 17 |
| L-25 | 6 | 1 | 2 | 9 | 11 | 4 | 15 | 14 | 48 |
| L-26 | 1 | 0 | 1 | 2 | 1 | 1 | 2 | 2 | 9 |
| L-27 | 2 | 0 | 1 | 3 | 2 | 1 | 3 | 3 | 16 |

X1 = Nº de Arquivos de Entrada X5 = Nº de Registros de Entrada
 X2 = Nº de Arquivos de Saída X6 = Nº de Registros de Saída
 X3 = Nº de Tipos de Relatórios X7 = X5 + X6
 X4 = X1 + X2 + X3 X8 = Nº de diferentes tipos de Registros

CARACTERÍSTICAS DAS VARIÁVEIS DE CADA UM DOS PROGRAMAS
DA CATEGORIA ORGANIZADORES

| VAR. PROGR. | X1 | X2 | X3 | X4 | X5 | X6 | X7 | X8 | DURA ÇÃO |
|----------------|----|----|----|----|----|----|----|----|-------------|
| O-1 | 2 | 1 | 1 | 4 | 8 | 8 | 16 | 9 | 19 |
| O-2 | 1 | 1 | 0 | 2 | 2 | 2 | 4 | 2 | 6 |
| O-3 | 1 | 1 | 0 | 2 | 2 | 2 | 4 | 2 | 7 |
| O-4 | 2 | 1 | 1 | 4 | 4 | 4 | 8 | 4 | 22 |
| O-5 | 5 | 2 | 1 | 8 | 8 | 8 | 16 | 9 | 23 |
| O-6 | 1 | 1 | 0 | 2 | 2 | 2 | 4 | 2 | 10 |
| O-7 | 2 | 2 | 1 | 5 | 2 | 2 | 4 | 2 | 14 |
| O-8 | 1 | 1 | 0 | 2 | 2 | 2 | 4 | 2 | 9 |
| O-9 | 2 | 1 | 0 | 3 | 7 | 7 | 14 | 8 | 22 |
| O-10 | 2 | 1 | 0 | 3 | 1 | 1 | 2 | 1 | 7 |
| O-11 | 4 | 2 | 0 | 6 | 5 | 5 | 10 | 5 | 17 |
| O-12 | 1 | 1 | 0 | 2 | 2 | 2 | 4 | 3 | 9 |
| O-13 | 2 | 2 | 1 | 5 | 4 | 4 | 8 | 5 | 16 |
| O-14 | 1 | 1 | 0 | 2 | 1 | 1 | 2 | 2 | 6 |
| O-15 | 1 | 1 | 0 | 2 | 1 | 1 | 2 | 2 | 5 |
| O-16 | 2 | 1 | 0 | 3 | 5 | 5 | 10 | 5 | 18 |
| O-17 | 4 | 1 | 0 | 5 | 4 | 4 | 8 | 4 | 10 |
| O-18 | 2 | 1 | 0 | 3 | 2 | 2 | 4 | 3 | 7 |
| O-19 | 1 | 1 | 0 | 2 | 4 | 4 | 8 | 4 | 13 |
| O-20 | 2 | 2 | 0 | 4 | 1 | 1 | 2 | 1 | 7 |
| O-21 | 5 | 1 | 0 | 6 | 5 | 5 | 10 | 6 | 16 |
| O-22 | 2 | 1 | 1 | 4 | 4 | 4 | 8 | 5 | 13 |
| O-23 | 1 | 1 | 0 | 2 | 4 | 4 | 8 | 4 | 16 |
| O-24 | 2 | 1 | 0 | 3 | 3 | 3 | 6 | 3 | 12 |
| O-25 | 2 | 2 | 1 | 5 | 2 | 2 | 4 | 3 | 8 |
| | | | | | | | | | |
| | | | | | | | | | |

X1 = Nº de Arquivos de Entrada

X2 = Nº de Arquivos de Saída

X3 = Nº de Tipos de Relatórios

X4 = X1 + X2 + X3

X5 = Nº de Registros de Entrada

X6 = Nº de Registros de Saída

X7 = X5 + X6

X8 = Nº de diferentes tipos de Registros

CARACTERÍSTICAS DAS VARIÁVEIS DE CADA UM DOS PROGRAMAS

DA CATEGORIA FORMATADORES

| VAR. PROGR. | X1 | X2 | X3 | X4 | X5 | X6 | X7 | X8 | DURA ÇÃO |
|----------------|----|----|----|----|----|----|----|----|-------------|
| F-1 | 1 | 1 | 0 | 2 | 1 | 1 | 2 | 2 | 6 |
| F-2 | 1 | 1 | 1 | 3 | 2 | 1 | 3 | 3 | 8 |
| F-3 | 2 | 1 | 0 | 3 | 2 | 1 | 3 | 2 | 8 |
| F-4 | 1 | 2 | 1 | 4 | 1 | 2 | 3 | 3 | 7 |
| F-5 | 5 | 1 | 0 | 6 | 5 | 2 | 7 | 2 | 10 |
| F-6 | 4 | 2 | 0 | 6 | 4 | 2 | 6 | 3 | 9 |
| F-7 | 1 | 2 | 1 | 4 | 8 | 6 | 14 | 14 | 11 |
| F-8 | 2 | 8 | 0 | 10 | 3 | 12 | 15 | 4 | 19 |
| F-9 | 1 | 3 | 0 | 4 | 2 | 6 | 8 | 8 | 9 |
| F-10 | 2 | 3 | 0 | 5 | 2 | 3 | 5 | 2 | 8 |
| F-11 | 2 | 2 | 0 | 4 | 2 | 2 | 4 | 4 | 7 |
| F-12 | 2 | 1 | 0 | 3 | 4 | 1 | 5 | 4 | 11 |
| F-13 | 2 | 6 | 0 | 8 | 5 | 5 | 10 | 7 | 15 |
| F-14 | 2 | 1 | 1 | 4 | 4 | 1 | 5 | 4 | 12 |
| F-15 | 4 | 3 | 2 | 9 | 4 | 5 | 9 | 6 | 11 |
| F-16 | 6 | 2 | 1 | 9 | 3 | 3 | 6 | 4 | 10 |
| F-17 | 5 | 5 | 1 | 11 | 5 | 6 | 11 | 8 | 14 |
| F-18 | 4 | 1 | 3 | 8 | 4 | 2 | 6 | 4 | 9 |
| F-19 | 1 | 3 | 1 | 5 | 4 | 2 | 6 | 4 | 12 |
| F-20 | 1 | 4 | 0 | 5 | 4 | 4 | 8 | 5 | 13 |
| F-21 | 4 | 5 | 1 | 10 | 5 | 8 | 13 | 13 | 12 |
| F-22 | 3 | 6 | 2 | 11 | 6 | 9 | 15 | 9 | 13 |
| | | | | | | | | | |
| | | | | | | | | | |

X1 = Nº de Arquivos de Entrada X5 = Nº de Registros de Entrada

X2 = Nº de Arquivos de Saída X6 = Nº de Registros de Saída

X3 = Nº de Tipos de Relatórios X7 = X5 + X6

X4 = X1 + X2 + X3

X8 = Nº de diferentes tipos de Registro

MATRIZ DOS COEFICIENTES DE CORRELAÇÃO SIMPLES ENTRE AS
VARIÁVEIS ENVOLVIDAS NO CASO DOS PROGRAMAS DA CATEGORIA

A T U A L I Z A D O R E S

| | DIAS | X1 | X2 | X3 | X4 | X5 | X6 | X7 | X8 |
|------|-------|------|-------|-------|------|-------|-------|-------|-------|
| DIAS | 1,00 | 0,00 | 0,17 | 0,67 | 0,56 | 0,86 | 0,64 | 0,94 | 0,96 |
| X1 | 0,00 | 1,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 |
| X2 | -0,17 | 0,00 | 1,00 | -0,11 | 0,33 | -0,20 | -0,09 | -0,19 | -0,18 |
| X3 | 0,67 | 0,00 | -0,11 | 1,00 | 0,91 | 0,54 | 0,37 | 0,57 | 0,59 |
| X4 | 0,56 | 0,00 | 0,33 | 0,91 | 1,00 | 0,43 | 0,31 | 0,46 | 0,49 |
| X5 | 0,86 | 0,00 | -0,20 | 0,54 | 0,43 | 1,00 | 0,29 | 0,85 | 0,78 |
| X6 | 0,64 | 0,00 | -0,09 | 0,37 | 0,31 | 0,29 | 1,00 | 0,75 | 0,79 |
| X7 | 0,94 | 0,00 | -0,19 | 0,57 | 0,46 | 0,85 | 0,75 | 1,00 | 0,98 |
| X8 | 0,96 | 0,00 | -0,18 | 0,59 | 0,49 | 0,78 | 0,79 | 0,98 | 1,00 |

X1 = Nº de Arquivos de Entrada X5 = Nº de Registros de Entrada

X2 = Nº de Arquivos de Saída X6 = Nº de Registros de Saída

X3 = Nº de Tipos de Relatórios X7 = X5 + X6

X4 = X1 + X2 + X3

X8 = Nº de diferentes tipos de Registros

MATRIZ DOS COEFICIENTES DE CORRELAÇÃO SIMPLES ENTRE AS
VARIÁVEIS ENVOLVIDAS NO CASO DOS PROGRAMAS DA CATEGORIA :

C O N S I S T Ê N C I A

| | DIAS | X1 | X2 | X3 | X4 | X5 | X6 | X7 | X8 |
|------|------|------|-------|-------|------|------|------|------|------|
| DIAS | 1,00 | 0,74 | 0,43 | 0,68 | 0,82 | 0,88 | 0,83 | 0,89 | 0,88 |
| X1 | 0,74 | 1,00 | 0,53 | 0,28 | 0,98 | 0,68 | 0,44 | 0,59 | 0,71 |
| X2 | 0,43 | 0,53 | 1,00 | -0,09 | 0,57 | 0,52 | 0,28 | 0,43 | 0,55 |
| X3 | 0,66 | 0,28 | -0,09 | 1,00 | 0,42 | 0,57 | 0,65 | 0,62 | 0,49 |
| X4 | 0,82 | 0,98 | 0,57 | 0,42 | 1,00 | 0,76 | 0,54 | 0,68 | 0,78 |
| X5 | 0,89 | 0,68 | 0,52 | 0,57 | 0,76 | 1,00 | 0,89 | 0,98 | 0,96 |
| X6 | 0,83 | 0,44 | 0,28 | 0,65 | 0,54 | 0,89 | 1,00 | 0,96 | 0,82 |
| X7 | 0,89 | 0,59 | 0,43 | 0,62 | 0,68 | 0,98 | 0,96 | 1,00 | 0,93 |
| X8 | 0,88 | 0,71 | 0,55 | 0,49 | 0,78 | 0,96 | 0,82 | 0,93 | 1,00 |

X1 = Nº de Arquivos de Entrada

X5 = Nº de Registros de Entrada

X2 = Nº de Arquivos de Saída

X6 = Nº de Registros de Saída

X3 = Nº de Tipos de Relatórios

X7 = X5 + X6

X4 = X1 + X2 + X3

X8 = Nº de diferentes tipos de Registros

MATRIZ DOS COEFICIENTES DE CORRELAÇÃO SIMPLES ENTRE AS
VARIÁVEIS ENVOLVIDAS NO CASO DOS PROGRAMAS DA CATEGORIA:

L I S T A D O R E S

| | DIAS | X1 | X2 | X3 | X4 | X5 | X6 | X7 | X8 |
|------|------|------|------|------|------|------|------|------|------|
| DIAS | 1,00 | 0,95 | 0,58 | 0,68 | 0,94 | 0,90 | 0,89 | 0,94 | 0,90 |
| X1 | 0,95 | 1,00 | 0,51 | 0,80 | 0,97 | 0,87 | 0,81 | 0,90 | 0,89 |
| X2 | 0,58 | 0,51 | 1,00 | 0,20 | 0,68 | 0,60 | 0,78 | 0,68 | 0,60 |
| X3 | 0,68 | 0,80 | 0,20 | 1,00 | 0,77 | 0,69 | 0,56 | 0,68 | 0,74 |
| X4 | 0,94 | 0,97 | 0,68 | 0,78 | 1,00 | 0,89 | 0,88 | 0,93 | 0,91 |
| X5 | 0,90 | 0,87 | 0,60 | 0,69 | 0,89 | 1,00 | 0,78 | 0,98 | 0,99 |
| X6 | 0,89 | 0,81 | 0,78 | 0,56 | 0,88 | 0,79 | 1,00 | 0,88 | 0,79 |
| X7 | 0,94 | 0,90 | 0,68 | 0,68 | 0,93 | 0,98 | 0,88 | 1,00 | 0,98 |
| X8 | 0,90 | 0,89 | 0,60 | 0,74 | 0,91 | 0,99 | 0,79 | 0,98 | 1,00 |

X1 = Nº de Arquivos de Entrada

X5 = Nº de Registros de Entrada

X2 = Nº de Arquivos de Saída

X6 = Nº de Registros de Saída

X3 = Nº de Tipos de Relatórios

X7 = X5 + X6

X4 = X1 + X2 + X3

X8 = Nº de diferentes tipos de Registros

MATRIZ DOS COEFICIENTES DE CORRELAÇÃO SIMPLES ENTRE AS
VARIÁVEIS ENVOLVIDAS NO CASO DOS PROGRAMAS DA CATEGORIA:

O R G A N I Z A D O R E S

| | DIAS | X1 | X2 | X3 | X4 | X5 | X6 | X7 | X8 |
|------|------|------|------|------|------|------|------|------|------|
| DIAS | 1,00 | 0,48 | 0,17 | 0,45 | 0,53 | 0,88 | 0,88 | 0,88 | 0,83 |
| X1 | 0,48 | 1,00 | 0,38 | 0,20 | 0,90 | 0,56 | 0,55 | 0,56 | 0,55 |
| X2 | 0,17 | 0,38 | 1,00 | 0,48 | 0,68 | 0,07 | 0,07 | 0,07 | 0,08 |
| X3 | 0,45 | 0,20 | 0,48 | 1,00 | 0,56 | 0,36 | 0,36 | 0,36 | 0,41 |
| X4 | 0,53 | 0,90 | 0,68 | 0,56 | 1,00 | 0,53 | 0,53 | 0,53 | 0,54 |
| X5 | 0,88 | 0,56 | 0,07 | 0,36 | 0,53 | 1,00 | 1,00 | 1,00 | 0,98 |
| X6 | 0,88 | 0,56 | 0,07 | 0,36 | 0,53 | 1,00 | 1,00 | 1,00 | 0,98 |
| X7 | 0,88 | 0,56 | 0,07 | 0,36 | 0,53 | 1,00 | 1,00 | 1,00 | 0,98 |
| X8 | 0,83 | 0,55 | 0,08 | 0,41 | 0,54 | 0,98 | 0,98 | 0,98 | 1,00 |

X1 = Nº de Arquivos de Entrada X5 = Nº de Registros de Entrada

X2 = Nº de Arquivos de Saída X6 = Nº de Registros de Saída

X3 = Nº de Tipos de Relatórios X7 = X5 + X6

X4 = X1 + X2 + X3

X8 = Nº de diferentes tipos de Registros

MATRIZ DOS COEFICIENTES DE CORRELAÇÃO SIMPLES ENTRE AS
VARIÁVEIS ENVOLVIDAS NO CASO DOS PROGRAMAS DA CATEGORIA:

F O R M A T A D O R E S

| | DIAS | X1 | X2 | X3 | X4 | X5 | X6 | X7 | X8 |
|------|------|------|-------|-------|------|------|------|------|------|
| DIAS | 1,00 | 0,13 | 0,81 | 0,03 | 0,64 | 0,53 | 0,73 | 0,77 | 0,37 |
| X1 | 0,13 | 1,00 | 0,06 | 0,32 | 0,68 | 0,29 | 0,11 | 0,21 | 0,48 |
| X2 | 0,81 | 0,06 | 1,00 | -0,03 | 0,72 | 0,29 | 0,89 | 0,79 | 0,43 |
| X3 | 0,03 | 0,32 | -0,03 | 1,00 | 0,45 | 0,31 | 0,11 | 0,21 | 0,28 |
| X4 | 0,64 | 0,68 | 0,72 | 0,45 | 1,00 | 0,45 | 0,71 | 0,73 | 0,41 |
| X5 | 0,53 | 0,29 | 0,29 | 0,31 | 0,45 | 1,00 | 0,40 | 0,73 | 0,70 |
| X6 | 0,73 | 0,11 | 0,89 | 0,11 | 0,71 | 0,40 | 1,00 | 0,92 | 0,62 |
| X7 | 0,77 | 0,21 | 0,79 | 0,21 | 0,73 | 0,73 | 0,92 | 1,00 | 0,76 |
| X8 | 0,37 | 0,48 | 0,43 | 0,28 | 0,41 | 0,70 | 0,62 | 0,76 | 1,00 |

X1 = Nº de Arquivos de Entrada

X5 = Nº de Registros de Entrada

X2 = Nº de Arquivos de Saída

X6 = Nº de Registros de Saída

X3 = Nº de Tipos de Relatórios

X7 = X5 + X6

X4 = X1 + X2 + X3

X8 = Nº de diferentes tipos de Registros

EQUAÇÕES PARA OBTENÇÃO DOS TEMPOS NORMAIS COM OS RES-
PECTIVOS COEFICIENTES DE DETERMINAÇÃO MÚLTIPLA

ATUALIZADORES

$$Y = 4,36 + 3,22 X_3 + 2,93 X_8$$

$$R^2 = 0,93$$

CONSISTÊNCIA

$$Y = 1,40 + 2,86X_4 + 0,80X_7$$

$$R^2 = 0,87$$

LISTADORES

$$Y = 3,89 + 4,01X_1 + 1,45X_7$$

$$R^2 = 0,94$$

ORGANIZADORES

$$Y = 4,44 + 1,89X_3 + 2,21X_6$$

$$R^2 = 0,80$$

FORMATADORES

$$Y = 5,42 + 1,09X_2 + 0,58X_5$$

$$R^2 = 0,75$$

onde,

X1 = Nº de Arquivos de Entrada

X5 = Nº de Registros de Entrada

X2 = Nº de Arquivos de Saída

X6 = Nº de Registros de Saída

X3 = Nº de Tipos de Relatórios

X7 = X5 + X6

X4 = X1 + X2 + X3

X8 = Nº de diferentes tipos de Registros

EQUAÇÕES PARA OBTER DIRETAMENTE O TEMPO - PADRÃO

(considerando-se uma taxa de tolerâncias de 25%)

ATUALIZADORES

$$Y = 5,45 + 4,03X_3 + 3,66X_8$$

CONSISTÊNCIA

$$Y = 1,75 + 3,58X_4 + 1,00X_7$$

LISTADORES

$$Y = 4,86 + 5,01X_1 + 1,81X_7$$

ORGANIZADORES

$$Y = 5,55 + 2,36X_3 + 2,76X_6$$

FORMATADORES

$$Y = 6,78 + 1,36X_2 + 0,73X_5$$

onde,

X1 = Nº de Arquivos de Entrada

X5 = Nº de Registros de Entrada

X2 = Nº de Arquivos de Saída

X6 = Nº de Registros de Saída

X3 = Nº de Tipos de Relatórios

X7 = X5 + X6

X4 = X1 + X2 + X3

X8 = Nº de diferentes tipo de Registros

C A P Í T U L O V I

C O N C L U S Õ E S

Um dos resultados a que se chegou com este trabalho, foi o de que o desenvolvimento de programas para computador não é uma tarefa imprevisível, mas que, ao contrário, desde que sejam conhecidas algumas características básicas dos programas, previsões poderão vir a ser feitas com boas chances de acerto. Esta, aliás, é a hipótese básica assumida na presente dissertação.

A possibilidade de se passar a dispor de um instrumental para a consecução do planejamento e controle da tarefa de programação de computadores é o resultado mais relevante que advém da implantação da presente metodologia. No entanto, tão importante quanto este aspecto, mas talvez não tão evidente, é um outro produto derivado, qual seja o de permitir uma melhor compreensão do processo de programação de computadores por parte de todos os envolvidos, programadores, analistas e supervisores, e principalmente o de se criar um clima em que o aspecto produtividade passe a se constituir em uma das preocupações mais importantes desses agrupamentos de pessoas.

Aqueles que trabalham em centros de processamento

de dados sabem como é difícil a implantação de tal clima.

Inclusive, como recomendação importante para futuras implantações desta metodologia, deve se procurar conseguir o total envolvimento de programadores que serão alvo de observações. Isso pode ser conseguido de diversas formas, entre elas esclarecendo-lhes exatamente os objetivos do estudo e solicitando-lhes subsídios, como, por exemplo, a opinião de cada um sobre quais as variáveis que afetariam cada modalidade de programa, ou então que tipos de categorias de programas deveriam ser consideradas. O ideal, é que todos os programadores de um centro de processamento de dados participem do estudo e que o andamento e as conclusões do mesmo, sejam discutidas com todos eles, pelo menos quinzenalmente. Isso facilitará sobremaneira a aceitação dos padrões que vierem a ser estabelecidos.

Com relação aos resultados da pesquisa, o fato de terem sido obtidos altos coeficientes de determinação múltipla é importante, mas não significa que a pesquisa deva cessar. Ela deve continuar obedecendo principalmente os seguintes aspectos:

- Observar e levantar dados sobre um maior número de programas, procurando-se inclusive identificar novas modalidades.
- Revisar a lista de variáveis, propondo-se e analisando-se outras diferentes daquelas que já foram apresentadas.

A separação dos programas em categorias, resultou

acertada, o que pode ser constatado por uma tentativa que foi feita de estabelecer-se uma única equação genérica, independente do tipo de programa. Assim sendo, tomou-se a amostra completa de 109 programas e utilizando-se o mesmo processo step-wise se calculou as equações de regressão, obtendo-se coeficientes de determinação múltipla bem abaixo dos que haviam sido obtidos para as equações individualmente obtidas por modalidades de programas, ou seja, enquanto para estas últimas haviam sido obtidos em média, coeficientes de 0,81 e 0,86 respectivamente no primeiro e segundo passo do step-wise, para as equações relativas à amostra global os valores dos coeficientes de determinação múltipla foram respectivamente de 0,63 e 0,71.

Ainda com relação a sistemática para obtenção das equações é importante frisar que ela não se esgota com a obtenção pura e simples das mesmas, pois um aspecto fundamental é mantê-las sempre atualizadas.

Portanto, uma recomendação muito importante é que a coleta de informações nunca cesse, de tal forma que seja constituído um banco de dados em contínua expansão, contendo informações acerca de todos os programas que venham a ser desenvolvidos mesmo após as fórmulas já estarem obtidas. Periodicamente, por exemplo, a cada seis meses, esse banco de dados deve servir de amostra para obtenção de novas equações de regressão que substituirão, então, as antigas.

Além do que já foi mencionado, uma consequência da implantação da metodologia proposta no capítulo anterior

é que ela permite para o CPD que adotá-la, uma nova abordagem em matéria de possibilidades de melhorias nos procedimentos administrativos e nas técnicas de planejamento e controle. O benefício mais imediato são as facilidades que passam a existir para o estabelecimento do tamanho do quadro de programadores necessários, bem como para a efetuação de projetos sobre futuras necessidades. Sendo importante se atentar para o fato de que dentro do ciclo de vida dos sistemas, no instante em que se iniciam as atividades dos programadores ainda se deverá gastar 70 a 80% do esforço total em homens/hora requerido pelo projeto todo, valores estes que sobem para cerca de 90% se se passar a incluir também os gastos com horas de computador para fins de teste. Portanto, é vital para o gerente de sistemas que ele possa contar com instrumentos eficazes para planejar, pelo menos as fases IV e V, Programação e Implantação, do ciclo de vida dos sistemas. Ainda com relação ao planejamento da mão de obra, a fixação de padrões de tempo, dá ao administrador de projetos a possibilidade de decidir com antecedência entre empregar horas-extras ou contratar mais programadores estudando a conveniência econômica de cada alternativa.

Do ponto de vista de melhoria de controles, o mais significativo é a possibilidade de se avaliar a eficiência da mão-de-obra o que dá flexibilidade inclusive para a implantação de um sistema de incentivos salariais, se a empresa assim o pretender. (37).

³⁷S. Green and R. Greene, "Overtime Pay for D.P. Employees?", Datamation, vol. 23, nº 5 : 215-216, Maio de 1977.

Aliás com a utilização dos índices de eficiência, passa a ser possível o exato enquadramento dos programadores dentro das suas várias categorias profissionais: junior, pleno e senior. Isso pode ser feito através da observação sistemática dos índices atingidos individualmente pelos programadores. Por exemplo, um programador que constantemente apresente altos índices de eficiência (acima de 1,20) está a indicar que deve ser promovido, o que não pode ocorrer quando a constância forem baixos índices (inferiores a 0,75).

Além dos benefícios já mencionados, vários outros existem apesar de não serem tão diretos e imediatos. Entre estes destaca-se também o de poder medir-se o custo unitário dos programas, o que é fundamental para centros de processamento de dados que atuam principalmente dentro de uma filosofia de prestação de serviços, como por exemplo bureaux de serviços, e as grandes empresas estatais de processamento de dados. Nesses casos inclusive, fica facilitada a tarefa de estabelecer-se o preço dos serviços de processamento de dados a serem cobrados dos usuários e/ou consumidores. Ficam assim claras, portanto, as perspectivas de melhorias no instrumental dos administradores de CPD, mencionando-se para finalizar as possibilidades que esses passam a ter de fixar gols e de trabalhar com base em orçamentos realistas.

Antes de concluir a presente dissertação é importante que fiquem caracterizados alguns outros assuntos relacionados com a área de desenvolvimento de sistemas para com

putador, para os quais se faz necessária a realização de estudos e pesquisas análogas, que objetivem estimar durações de tarefas e avaliar índices de produtividade. Visualizam-se principalmente três áreas de estudos:

1. Atividades de Analistas de Sistemas: As tarefas dos Analistas de Sistemas e Analistas de O&M também poderiam receber estudos à luz das técnicas de Medida do Trabalho, recomendando-se para a realização destes, a aplicação de uma metodologia semelhante aquela que aqui foi apresentada.
2. Padrões para efetuação de manutenções em programas de computador: Inegavelmente mais complexo que o estabelecimento de padrões para o desenvolvimento de novos programas, esse é um problema que também precisa ser atacado sob um enfoque de Medida do Trabalho. A maior dificuldade está na caracterização das variáveis que afetam o trabalho de manutenção. A sugestão aos gerentes de manutenção é para que inicialmente passem a coletar dados sobre os esforços de mão-de-obra sob sua responsabilidade e numa etapa posterior tentem explicar porque algumas manutenções foram mais demoradas que outras, procurando, assim, descobrir possíveis variáveis. Uma delas já pode ser antecipada: O grau de conhecimento que o programador que irá efetuar a manutenção possui sobre o programa a ser alterado.
3. Análise da influência da lei do conhecimento: No caso apresentado no capítulo V, sempre se procurou impedir ain

fluência da lei do conhecimento. Isso foi feito, evitando-se com que o mesmo programador, viesse a produzir um determinado programa, se anteriormente ele já houvesse produzido algum outro programa semelhante, mesmo que apenas parcialmente. Todavia uma sugestão para a realização de estudos posteriores é a análise da influência desta lei sobre a atividade de programação de computadores.

Finalizando, acreditamos ter atingido o objetivo exposto no primeiro capítulo desta dissertação, qual seja o de ter contribuído para o incremento dos recursos e técnicas utilizadas na administração de projetos de sistemas para computador. Acreditamos, também que ficaram patentes as grandes possibilidades com que se pode contar através da utilização das técnicas de Medida do Trabalho na área de desenvolvimento de sistemas de processamento de dados.

REFERÊNCIAS

BIBLIOGRÁFICAS

LIVROS:

1. ATKINS, W., DITRI, A., and SHAW, J., Managing the EDP Function, New York, Mc Graw Hill Book Company, 1971.
2. ATKINS, W. and SHAW, J. Managing Computer System Projects, New York, Mac Graw Hill Book Company, 1970.
3. BARNES, Ralph M., Estudo de Movimentos e de Tempos: Projeto e Medida do Trabalho, traduzido do ingles por Sergio O. Assis, Jose S. G. Azevedo e Arnaldo Pallotta, São Paulo, Editora Edgard Blücher LTDA, 1963.
4. BASTOS, Alex C., Programação Cobol, Rio de Janeiro, Livros Técnicos e Científicos Editora S.A., 1977.
5. BRANDON, Dick H., Management Standars for Data Processing, New York, Van Nostrand Reinhold Company, 1963.
6. _____, Project Control Standars, Philadelphia, Auerbach, 1970.
7. BROOKS, Fred, The Mythical Man - Month: Essays on Software Engineering, Mass., U.S.A., Addison-Wesley, 1975.
8. CLIFTON, H.D., Systems Analysis for Business Data Processing, New York, Petrocelli Books, 1974.
9. DRAPER, Norman, Applied Regression Analysis, New York, Wiley, 1966.
10. FONSECA, J.S., MARTINS, G.A., TOLEDO, G.L., Estatística

Aplicada, São Paulo, Editora Atlas, 1976.

11. FOX, Karl e MERRIL, William, Estatística Econômica: Uma Introdução, traduzido do inglês por Alfredo Alves de Faria, São Paulo, Editora Atlas, 1977.
12. FRIELINK, A., Economics of Automatic Data Processing, Amsterdam, A.B. Frielink, 1965.
13. GARBASSI, U. e Mc CRACKEN, Daniel D., Programação Cobol, traduzido do inglês por Danilo A. Nogueira, São Paulo, Editora Atlas S.A., 1976.
14. KRICK, Edward V., Métodos e Sistemas, traduzido do inglês por Roberto Verdussen, Rio de Janeiro, Livros Técnicos e Científicos Editora LTDA., 1971, 2 volumes.
15. LEME, Ruy A.S., Controles na Produção, São Paulo, Departamento de Engenharia de Produção da E.P.U.S.P., 1967.
16. _____, Curso de Estatística: Elementos, 3a. edição, Rio de Janeiro, Ao Livro Técnico S.A., 1969.
17. MACHLINE, C., SÁ MOTA, I., SCHOEPS, W., WEIL, K., Manual de Administração da Produção, 2a. edição, Rio de Janeiro, Fundação Getúlio Vargas, 1974, 2 volumes.
18. MAYNARD, H.B., Manual de Engenharia de Produção: Seção 3 - Técnicas de Medida do Trabalho, traduzido do inglês por Itiro Iida e vários outros, São Paulo, Editora Edgard Blucher LTDA., 1970.

19. MUNDEL, M. E., Estudos de Movimentos e Tempos: Princípios e Prática, 3a. edição, São Paulo, Mestre Jou, 1966.
20. REIS, Dayr A., Sistemas de Produção: Projeto e Controle, Michigan, CSI - Copy Graph Services, 1975.
21. RIGGS, James L., Administração da Produção: Planejamento, Análise e Controle, traduzido do inglês por Eda Quadros, São Paulo, Editora Atlas S.A., 1976.
22. STARR, Martin K., Administração da Produção: Sistemas e Sínteses, traduzido do inglês por Miguel Cezar Santo ro, São Paulo, Editora Edgard Blucher LTDA., 1971.
23. SZWEDA, Ralph, Information Processing Management, Princeton, Auerbach Publishers Ind., 1972.
24. YEARSLEY, R.B. and GRAHAM, G.M.R., Handbook of Computer Management, Great Britain, Gower Press Limited, 1973.
25. WEINBERG, Gerald M., The Psychology of Computer Programming, New York, Van Nostrand Reinhold Company , 1971.

ARTIGOS

1. ALBRECHT, D. and KARGER, D., "The Measurement of Professional Work", Industrial Engineering, vol. 3, nº 8 : 26-31, Agosto de 1971.

2. ARVEY, R. and HOYLE, J., "Evaluating Computer Personnel", Datamation, vol. 19, nº 7 : 69-73, julho de 1973.
3. AVOTS, Ivars, "Making Project Management Work", Datamation, vol. 19, nº 1 : 42-45, Janeiro de 1973.
4. BAKER, F. T., "Chief Programmer Team: Management of Production Programming", IBM Systems Journal, Vol. II, nº 1 : 56-73, 1972.
5. BELADY, L.A. and LEHMAN, M.M., "A Model of Large Program Development", IBM Systems Journal, vol. 15, nº 3 : 225-252, 1976.
6. BOEPPLE Jr, E. and KELLY, L.A., "How to Measure Thinking?", Industrial Engineering, vol. 3, nº 7 : 8-16, Julho de 1971.
7. BOHEM, Barry W., "Software and its Impact: A Quantitative Assesment", Datamation, vol. 19; nº 5 : 48-59, Maio de 1973.
8. BRYAN, B. and HOLTON J., "Structured Top - down Flowcharting", Datamation, vol. 21, nº 5 : 80-84, Maio de 1975.
9. CANNING, Richard, "The Benefits of Standard Practice", EDP Analyzer, vol. 13, nº 8 : 1-12, Agosto de 1975.
10. _____, "Progress in Software Engineering:

- Part 1", EDP Analyzer, vol. 16, nº 2 : 1-13, Fevereiro de 1978.
11. _____, "Progress in Software Engineering: Part 2", EDP Analyzer, vol. 16, nº 3 : 1-13, Março de 1978.
 12. CHRIST, Charles, "An Analytical Way to Measure Indirect Work", Industrial Engineering, vol. 1, nº 6 : 36-38, Junho de 1969.
 13. CONSTANTINE, L., MYERS, G. and STEVENS, W., "Structured Desing", IBM Systems Journal, vol. 13, nº 2: 115-139, 1974.
 14. COOKE Jr, Lawrence H., "Programming Time vs Running Time", Datamation, vol. 20, nº 12 : 56-58, Dezembro de 1974.
 15. CORCHAKI, P. e SOUZA, L.A.P., "Estudo Ve Desequilíbrio entre Demanda e Oferta", Datanews, págs. 17 a 22, de 15/11/78.
 16. COSGROVE, John, "Needed: A New Planning Framework", Datamation, vol. 17, nº 23 : 37-39, 15 de dezembro de 1971.
 17. COUGER, and ZAWACKI, "What Motivates D.P. Professionals?", Datamation, vol. 24, nº 9 : 116-123, Setembro de 1978.
 18. DALY, Edmund B., "Management of Software Development",

IEEE Transaction on Software Engineering, 229-242,
Maio de 1977.

19. DONALDSON, James, "Structured Programming", Datamation,
vol. 19, nº 12 : 52-54, Dezembro de 1973.
20. DONELSON, William S., "Project Planning and Control",
Datamation, vol. 22, nº 6 ; 73-80, Junho de 1976.
21. DONEY, L., and GELB, T., "Regression Short-Cuts Cycle
Time Estimating", Industrial Engineering, vol. 3, nº
2 : 22-23, Fevereiro de 1971.
22. ERSHOV, Andrei P., "Aesthetics and the Human Factor in Pro-
gramming" Datamation, vol. 18, nº 7 : 62-67, Julho
de 1972.
23. FAGAN, M. E., "Desing and Code Inspection to Reduce
Errors in Program Development", IBM Systems Journal,
vol. 15, nº 3 : 182-211, 1976.
24. FELIX, C.P. and WALSTON C.E., "A Method of Programming
Measurement and Estimation", IBM Systems Journal, vol.
16, nº 1 : 54-73, 1977.
25. FITZ-ENZ, Jac, "Who Is the D.P. Professional?", Datama-
tion, vol. 24, nº 9 : 125-128, Setembro de 1978.
26. FRANK, E., "Low Cost Standars for Indirect Labor", In-
dustrial Engineering, vol. 2, nº 8 : 27-28 Agosto
de 1970.

27. GILDERSLEEVE, Thomas R., "Organizing the Data Processing Function", Datamation, vol. 20, nº 11 : 46-50, Novembro de 1974.
28. _____, "The Time-Estimating Myth", Datamation, vol. 19, nº 1 : 47-48, Janeiro de 1973.
29. GREEN, S. and GREENE, R., "Overtime Pay for D.P. Employees?", Datamation, vol. 23, nº 5 : 215-216, Maio de 1977.
30. GREENBAUN, J., "The Division of Labor in the Computer Field - part 1", Computers and People, vol. 25, nº 11 : 17-22, Novembro de 1976.
31. _____, "The Division of Labor in the Computer Field - part 2", Computers and People, vol. 25, nº 12 : 19-21, Dezembro de 1976.
32. HELENA, S., "Daqui para Frente, Liberdade para Definir e Aplicar Critérios", Datanews, pág. 2, 20.12.78.
33. JOHNSON, James R., "A Working Measure of Productivity", Datamation, vol. 23, nº 2 : 106-110, Fevereiro de 1977.
34. JOHNSON R.R., "Needed : A Measure for Measure", Datamation, vol. 16, nº 17 : 22-30, 15 de dezembro de 70.
35. JONES, T.C., "Measuring Programming Quality and Productivity", IBM Systems Journal, vol. 17, nº 1, 1978.

36. LARREN, G., "Software : A Qualitative Assesment", Data-mation, vol. 19, nº 11 : 60-66, Novembro de 1973.
37. LEVITT, Theodore, "Las Prestaciones de Servicios Vistos con Criterio Fabril", Administracion Empresas, vol. VI, nº 67 : 587-603, Outubro de 1975.
38. LIU, Chester C., "A Look at Software Maintenance", Datamation, vol. 22, nº 11 : 51-55, Novembro de 1976.
39. LONG Jr, C. and SADOSKY, T., "Applying Scientific Management to Creativity", Industrial Engineering, vol. 3, nº 1 : 20-22, Janeiro de 1971.
40. LOSOVIZ, E.A., "El Centro de Computos como Organization Productiva", Administracion Empresas, vol. IV-B, nº 43 : 653-662, Outubro de 1973.
41. MARTIN, D., "Instant Time Standars", Industrial Engineering, vol. 3, nº 11 : 24-27, Novembro de 1971.
42. MARTINS, L., e SOARES, M., "Soft: o Papel do Estado", Dados e Idéias, vol. 4, nº 3 : 2-9, Janeiro de 79.
43. MATTOS, Antonio C.M., "O Impacto do Computador na Empresa", Revista de Administração de Empresas, vol. 18, nº 4 : 53-58, out/dezembro de 1978.
44. NOLAN, R., "Experts Discuss Clerical Work Measurement Needs", Industrial Engineering, vol. 3, nº 5 : 17 - 20, Maio de 1971.

45. OLIVEIRA, Carlos A.A., "Recursos Brasileiros em Computação: Valor de Mercado", Dados e Idéias, vol. 3, nº 3: 63-73, Janeiro de 1978.
46. PATRICK, R.L., "Software Engineering and Life Cycle Planning", Datamation, vol. 22, nº 12 : 79-80, Dezembro de 1976.
47. PEEPLES, Donald E., "Measure for Productivity", Datamation, vol. 24, nº 5 : 222-230, Maio de 1978.
48. RAU, P., "Evaluating the EDP Function", Datamation, vol. 18, nº 9 : 72-73, Setembro de 1972.
49. ROGERS, Lloyd A., "Guidelines for Project Management Teams", Industrial Engineering, vol. 6, nº 12 : 12 - 17, Dezembro de 1974.
50. SCOTT, Randall F., "Programmer Productivity and the Delphi Technique", Datamation, vol. 20, nº 5 : 71-73, Maio de 1974.
51. SHELL, Richard L., "Work Measurement for Computer Programming Operations", Industrial Engineering, vol. 4, nº 10 : 32-36, Outubro de 1972.
52. SLAYTON, Raymond D., "Work Measurement on a Grand Scale", Industrial Engineering, vol. 6, nº 11 : 38-45, Novembro de 1974.
53. SMERILSON, Harvey H., "Standars for Engineers", Indus-

trial Engineering, vol. 7, nº 10 : 12-17, Outubro de 1975.

54. SOBCZAK, J., "Pricing Computer Usage", Datamation, vol. 20, nº 2 : 61-64, Fevereiro de 1974.

55. SOLOMON, Martin B., "Economies of Scale and Computer Personnel", Datamation, vol. 16, nº 3 : 107-110, Março de 1970.

56. SUCESU, "Pesquisa Salarial", Procesu, nº 3, pág. 4, Junho de 1978.

57. VAN KIRK, Richard L., "Work Standars for Highway Engineers", Industrial Engineering, vol. 5, nº 1 : 34-39, Janeiro de 1973.

58. YOURDON, Edward, "Symposium on Structured Programming in Cobol", Datamation, vol. 21, nº 6 : 97, Junho de 1975.

59. _____, "Making the Move to Structured Programming", Datamation, vol. 21, nº 6 : 52-56, Junho de 1975.

A N E X O

(listagens referentes a aplicação do package de step-wise para o caso de programas da categoria listadores).

PAGE 1

// JGB 1130 0019

LCG DRIVE CART SPEC CART AVAIL PHY DRIVE

| | | | |
|------|------|------|------|
| 0000 | 1130 | 1130 | 0001 |
| 0001 | 0019 | 0C19 | 0002 |
| | | 2C00 | 0003 |
| | | 3C00 | 0004 |
| | | 4C00 | 0005 |

V2 M11 ACTUAL 16K CONFIG 16K

*EQUAT(R2501,READ),(READZ,CAR02),(D2501,D1442),(PRNTZ,PRNZ)

// XEQ TAB 1

*FILES(1,TBX01,0019),(606,TBX02,0019)

SISTEMA T A B

ERROS
(NO.PARAM)

CARTOES DE CONTROLE

```
INI
PRC 'PROGRAMAS LISTADORES'
ARC -1
VAR 1,1,2 'ARQ.ENT.'
VAR 2,3,4 'ARQ.SAI.'
VAR 3,5,6 'RELATOR.'
VAR 4,7,8 'S 1+2+3'
VAR 5,9,10 'FORM ENT'
VAR 6,11,12 'FORM SAI'
VAR 7,13,14 'S 5+6'
VAR 8,15,16 'FORM DIF'
VAR 9,17,18 'DIAS'
REG 9,1,2,3,4,5,6,7,8
FIM
/*
```

INICIO DA EXECUCAO

```
1 0 1 2 4 1 5 517
4 1 2 7 6 3 9 828
5 0 2 7 6 2 8 836
4 2 1 7 8 4 12 939
2 1 1 4 7 2 9 826
1 0 1 2 1 1 2 214
1 0 1 2 1 1 2 2 7
1 0 1 2 1 1 2 2 9
1 0 1 2 2 1 3 314
1 1 1 3 2 2 4 315
1 0 1 2 1 1 2 2 7
1 0 1 2 2 2 4 315
2 1 1 4 3 2 5 418
3 0 1 4 5 1 6 623
3 1 1 5 3 3 6 431
3 0 1 4 3 3 6 432
1 0 1 2 1 1 2 213
1 0 1 2 4 1 5 513
1 0 1 2 1 1 2 210
5 1 2 8 9 4 13 1242
2 1 1 4 3 2 5 418
3 2 1 6 5 3 8 826
1 0 1 2 1 1 2 210
2 0 1 3 3 1 4 417
6 1 2 9 11 4 15 1448
1 0 1 2 1 1 2 2 9
2 0 1 3 2 1 3 316
/*
```

T A B

PROGRAMAS LISTA00RES

VARIAVEIS UTILIZADAS

| | | | | | | |
|--------|--------|-------|-------|-------|--------|-------|
| 17.000 | 1.000 | 0.000 | 1.000 | 2.000 | 4.000 | 1.000 |
| 5.000 | 5.000 | | | | | |
| 28.000 | 4.000 | 1.000 | 2.000 | 7.000 | 6.000 | 3.000 |
| 9.000 | 8.000 | | | | | |
| 36.000 | 5.000 | 0.000 | 2.000 | 7.000 | 5.000 | 2.000 |
| 8.000 | 8.000 | | | | | |
| 39.000 | 4.000 | 2.000 | 1.000 | 7.000 | 8.000 | 4.000 |
| 12.000 | 9.000 | | | | | |
| 26.000 | 2.000 | 1.000 | 1.000 | 4.000 | 7.000 | 2.000 |
| 9.000 | 8.000 | | | | | |
| 14.000 | 1.000 | 0.000 | 1.000 | 2.000 | 1.000 | 1.000 |
| 2.000 | 2.000 | | | | | |
| 7.000 | 1.000 | 0.000 | 1.000 | 2.000 | 1.000 | 1.000 |
| 2.000 | 2.000 | | | | | |
| 9.000 | 1.000 | 0.000 | 1.000 | 2.000 | 1.000 | 1.000 |
| 2.000 | 2.000 | | | | | |
| 14.000 | 1.000 | 0.000 | 1.000 | 2.000 | 2.000 | 1.000 |
| 3.000 | 3.000 | | | | | |
| 15.000 | 1.000 | 1.000 | 1.000 | 3.000 | 2.000 | 2.000 |
| 4.000 | 3.000 | | | | | |
| 7.000 | 1.000 | 0.000 | 1.000 | 2.000 | 1.000 | 1.000 |
| 2.000 | 2.000 | | | | | |
| 15.000 | 1.000 | 0.000 | 1.000 | 2.000 | 2.000 | 2.000 |
| 4.000 | 3.000 | | | | | |
| 18.000 | 2.000 | 1.000 | 1.000 | 4.000 | 3.000 | 2.000 |
| 5.000 | 4.000 | | | | | |
| 23.000 | 3.000 | 0.000 | 1.000 | 4.000 | 5.000 | 1.000 |
| 6.000 | 6.000 | | | | | |
| 31.000 | 3.000 | 1.000 | 1.000 | 5.000 | 3.000 | 3.000 |
| 6.000 | 4.000 | | | | | |
| 32.000 | 3.000 | 0.000 | 1.000 | 4.000 | 3.000 | 3.000 |
| 6.000 | 4.000 | | | | | |
| 13.000 | 1.000 | 0.000 | 1.000 | 2.000 | 1.000 | 1.000 |
| 2.000 | 2.000 | | | | | |
| 13.000 | 1.000 | 0.000 | 1.000 | 2.000 | 4.000 | 1.000 |
| 5.000 | 5.000 | | | | | |
| 10.000 | 1.000 | 0.000 | 1.000 | 2.000 | 1.000 | 1.000 |
| 2.000 | 2.000 | | | | | |
| 42.000 | 5.000 | 1.000 | 2.000 | 8.000 | 9.000 | 4.000 |
| 13.000 | 12.000 | | | | | |
| 18.000 | 2.000 | 1.000 | 1.000 | 4.000 | 3.000 | 2.000 |
| 5.000 | 4.000 | | | | | |
| 26.000 | 3.000 | 2.000 | 1.000 | 6.000 | 5.000 | 3.000 |
| 8.000 | 8.000 | | | | | |
| 10.000 | 1.000 | 0.000 | 1.000 | 2.000 | 1.000 | 1.000 |
| 2.000 | 2.000 | | | | | |
| 17.000 | 2.000 | 0.000 | 1.000 | 3.000 | 3.000 | 1.000 |
| 4.000 | 4.000 | | | | | |
| 48.000 | 6.000 | 1.000 | 2.000 | 9.000 | 11.000 | 4.000 |
| 15.000 | 14.000 | | | | | |
| 9.000 | 1.000 | 0.000 | 1.000 | 2.000 | 1.000 | 1.000 |
| 2.000 | 2.000 | | | | | |
| 16.000 | 2.000 | 0.000 | 1.000 | 3.000 | 2.000 | 1.000 |
| 3.000 | 3.000 | | | | | |

MATRIZ DOS PRODUTOS CROZADOS

| VARIÁVEL | DIAS | ARQ.ENT. | ARQ.SAI. | RELATOR. | S 1+2+3 | FORM ENT | FORM SAI | S 5+6 |
|----------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|
| DIAS | 0.14657E 05 | 0.16250E 04 | 0.35600E 03 | 0.70700E 03 | 0.26880E 04 | 0.26960E 04 | 0.13020E 04 | 0.39960E 04 |
| ARQ.ENT. | 0.16250E 04 | 0.18700E 03 | 0.39000E 02 | 0.79000E 02 | 0.30500E 03 | 0.30300E 03 | 0.14300E 03 | 0.44600E 03 |
| ARQ.SAI. | 0.35600E 03 | 0.39000E 02 | 0.16000E 02 | 0.15000E 02 | 0.70000E 02 | 0.70000E 02 | 0.36000E 02 | 0.10600E 03 |
| RELATOR. | 0.70700E 03 | 0.79000E 02 | 0.15000E 02 | 0.39000E 02 | 0.13300E 03 | 0.12800E 03 | 0.63000E 02 | 0.19100E 03 |
| S 1+2+3 | 0.26880E 04 | 0.30500E 03 | 0.70000E 02 | 0.13300E 03 | 0.50800E 03 | 0.50100E 03 | 0.24200E 03 | 0.74300E 03 |
| FORM ENT | 0.26960E 04 | 0.30300E 03 | 0.70000E 02 | 0.12800E 03 | 0.50100E 03 | 0.53800E 03 | 0.23700E 03 | 0.77500E 03 |
| FORM SAI | 0.13020E 04 | 0.14300E 03 | 0.36000E 02 | 0.63000E 02 | 0.24200E 03 | 0.23700E 03 | 0.12200E 03 | 0.35900E 03 |
| S 5+6 | 0.39960E 04 | 0.44600E 03 | 0.10600E 03 | 0.19100E 03 | 0.74300E 03 | 0.77500E 03 | 0.35900E 03 | 0.11340E 04 |
| FORM DIF | 0.35450E 04 | 0.39900E 03 | 0.91000E 02 | 0.17300E 03 | 0.66300E 03 | 0.69600E 03 | 0.31400E 03 | 0.10100E 04 |

MATRIZ DOS PRODUTOS CRUZADOS

VARIÁVEL FORM DIF

| | |
|----------|-------------|
| CIAS | 0.35450E 04 |
| ARC.ENT. | 0.39900E 03 |
| ARC.SAI. | 0.91000E 02 |
| RELATOR. | 0.17300E 03 |
| S 1+2+3 | 0.66300E 03 |
| FORM ENT | 0.65600E 03 |
| FORM SAI | 0.31400E 03 |
| S 5+6 | 0.10100E 04 |
| FORM DIF | 0.91100E 03 |

MATRIZ DOS PRODUTOS CRUZADOS DOS RESÍDUOS

| VARIÁVEL | DIAS | ARO.ENT. | ARO.SAI. | RELATOR. | S 1+2+3 | FORM.ENT | FORM.SAI | S 5+6 |
|----------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|
| DIAS | 0.33307E 04 | 0.41659E 03 | 0.11022E 03 | 0.72074E 02 | 0.59888E 03 | 0.72977E 03 | 0.27792E 03 | 0.10077E 04 |
| ARO.ENT. | 0.41659E 03 | 0.58074E 02 | 0.12777E 02 | 0.11259E 02 | 0.82111E 02 | 0.93222E 02 | 0.33740E 02 | 0.12696E 03 |
| ARO.SAI. | 0.11022E 03 | 0.12777E 02 | 0.10666E 02 | 0.12222E 01 | 0.24666E 02 | 0.27333E 02 | 0.13777E 02 | 0.41111E 02 |
| RELATOR. | 0.72074E 02 | 0.11259E 02 | 0.12222E 01 | 0.34074E 01 | 0.15888E 02 | 0.17777E 02 | 0.55925E 01 | 0.23370E 02 |
| S 1+2+3 | 0.59888E 03 | 0.82111E 02 | 0.24666E 02 | 0.15888E 02 | 0.12266E 03 | 0.13833E 03 | 0.53111E 02 | 0.19144E 03 |
| FORM.ENT | 0.72977E 03 | 0.93222E 02 | 0.27333E 02 | 0.17777E 02 | 0.13833E 03 | 0.19666E 03 | 0.59222E 02 | 0.25588E 03 |
| FORM.SAI | 0.27792E 03 | 0.33740E 02 | 0.13777E 02 | 0.55925E 01 | 0.53111E 02 | 0.59222E 02 | 0.29407E 02 | 0.88629E 02 |
| S 5+6 | 0.10077E 04 | 0.12696E 03 | 0.41111E 02 | 0.23370E 02 | 0.19144E 03 | 0.25588E 03 | 0.88629E 02 | 0.34451E 03 |
| FORM.DIF | 0.86192E 03 | 0.11274E 03 | 0.32777E 02 | 0.22592E 02 | 0.16811E 03 | 0.23022E 03 | 0.71407E 02 | 0.30162E 03 |

MATRIZ DOS PRODUTOS CRUZADOS DOS RESIDUOS

VARIÁVEL

FORM DIF

| | |
|----------|-------------|
| DIAS | 0.86192E 03 |
| ARC.ENT. | 0.11274E 03 |
| ARC.SAI. | 0.32777E 02 |
| RELATOR. | 0.22592E 02 |
| S 1+2+3 | 0.16811E 03 |
| FORM ENT | 0.23022E 03 |
| FORM SAI | 0.71407E 02 |
| S 5+6 | 0.30162E 03 |
| FORM DIF | 0.27540E 03 |

T A B

PROGRAMAS LISTA00RES

MATRIZ DE VARIANCA-COVARIANCA

| VARIAVEL | DIAS | ARQ.ENT. | ARQ.SAI. | RELATOR. | S 1+2+3 | FORM ENT | FORM SAI | S 5+6 |
|----------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|
| DIAS | 0.12810E 03 | 0.16022E 02 | 0.42393E 01 | 0.27720E 01 | 0.23034E 02 | 0.28068E 02 | 0.10689E 02 | 0.38757E 02 |
| ARQ.ENT. | 0.16022E 02 | 0.22336E 01 | 0.49145E 00 | 0.43304E 00 | 0.31581E 01 | 0.35854E 01 | 0.12977E 01 | 0.48831E 01 |
| ARQ.SAI. | 0.42393E 01 | 0.49145E 00 | 0.41025E 00 | 0.47008E-01 | 0.94871E 00 | 0.10512E 01 | 0.52991E 00 | 0.15811E 01 |
| RELATOR. | 0.27720E 01 | 0.43304E 00 | 0.47008E-01 | 0.13105E 00 | 0.61111E 00 | 0.68376E 00 | 0.21509E 00 | 0.89886E 00 |
| S 1+2+3 | 0.23034E 02 | 0.31581E 01 | 0.94871E 00 | 0.61111E 00 | 0.47179E 01 | 0.53205E 01 | 0.20427E 01 | 0.73632E 01 |
| FORM ENT | 0.28068E 02 | 0.35854E 01 | 0.10512E 01 | 0.68376E 00 | 0.53205E 01 | 0.75641E 01 | 0.22777E 01 | 0.98418E 01 |
| FORM SAI | 0.10689E 02 | 0.12977E 01 | 0.52991E 00 | 0.21509E 00 | 0.20427E 01 | 0.22777E 01 | 0.11310E 01 | 0.34088E 01 |
| S 5+6 | 0.38757E 02 | 0.48831E 01 | 0.15811E 01 | 0.89886E 00 | 0.73632E 01 | 0.98418E 01 | 0.34088E 01 | 0.13250E 02 |
| FORM DIF | 0.33150E 02 | 0.43361E 01 | 0.12606E 01 | 0.86894E 00 | 0.64658E 01 | 0.88547E 01 | 0.27464E 01 | 0.11601E 02 |

MATRIZ DE VARIANCA-COVARIANCA

| VARIAVEL | FORM DIF |
|----------|----------|
|----------|----------|

| | |
|----------|-------------|
| CIAS | 0.33150E 02 |
| ARG.ENT. | 0.43361E 01 |
| ARG.SAI. | 0.12606E 01 |
| RELATOR. | 0.86894E 00 |
| S 1+2+3 | 0.64658E 01 |
| FORM ENT | 0.88547E 01 |
| FORM SAI | 0.27464E 01 |
| S 5+6 | 0.11601E 02 |
| FORM DIF | 0.10592E 02 |

| ESTATISTICA DAS VARIÁVEIS | | OBSERVAÇÕES 27 | | | | |
|---------------------------|-----------|----------------|---------|-------------|-------------|-------------|
| VARIÁVEL | V. MINIMO | V. MAXIMO | MEDIA | D. PADRAO | C. VARIACAO | VARIANCA |
| CIAS | 7.0000 | 48.0000 | 20.4814 | 0.11318E 02 | 0.55261 | 0.12810E C3 |
| ARC. ENT. | 1.0000 | 6.0000 | 2.1851 | 0.14945E 01 | 0.68393 | 0.22336E 01 |
| ARQ. SAI. | 0.0000 | 2.0000 | 0.4444 | 0.64051E 00 | 1.44115 | 0.41025E C0 |
| RELATOR. | 1.0000 | 2.0000 | 1.1481 | 0.36201E 00 | 0.31530 | 0.13105E C0 |
| S 1+2+3 | 2.0000 | 9.0000 | 3.7777 | 0.21720E 01 | 0.57496 | 0.47179E C1 |
| FORM ENT | 1.0000 | 11.0000 | 3.5555 | 0.27502E 01 | 0.77351 | 0.75641E C1 |
| FORM SAI | 1.0000 | 4.0000 | 1.8518 | 0.10635E 01 | 0.57429 | 0.11310E C1 |
| S 5+6 | 2.0000 | 15.0000 | 5.4074 | 0.36401E 01 | 0.67317 | 0.13250E C2 |
| FORM DIF | 2.0000 | 14.0000 | 4.8518 | 0.32546E 01 | 0.67080 | 0.10592E C2 |

MATRIZ DOS COEFICIENTES DE CORRELACAO

| VARIAVEL | DIAS | ARQ. ENT. | ARQ. SAI. | RELATOR. | S 1+2+3 | FORM ENT | FORM SAI | S 5+6 |
|-----------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|
| CIAS | 0.10000E 01 | 0.94721E 00 | 0.58476E 00 | 0.67654E 00 | 0.93694E 00 | 0.90168E 00 | 0.88803E 00 | 0.94071E 00 |
| ARQ. ENT. | 0.94721E 00 | 0.10000E 01 | 0.51339E 00 | 0.80039E 00 | 0.97285E 00 | 0.87229E 00 | 0.81646E 00 | 0.89759E 00 |
| ARQ. SAI. | 0.58476E 00 | 0.51339E 00 | 0.10000E 01 | 0.20273E 00 | 0.68191E 00 | 0.59677E 00 | 0.77792E 00 | 0.67916E 00 |
| RELATOR. | 0.67654E 00 | 0.80039E 00 | 0.20273E 00 | 0.10000E 01 | 0.77717E 00 | 0.68675E 00 | 0.55869E 00 | 0.69209E 00 |
| S 1+2+3 | 0.93694E 00 | 0.97285E 00 | 0.68191E 00 | 0.77717E 00 | 0.10000E 01 | 0.89063E 00 | 0.88428E 00 | 0.93126E 00 |
| FORM ENT | 0.90168E 00 | 0.87229E 00 | 0.59677E 00 | 0.68675E 00 | 0.89063E 00 | 0.10000E 01 | 0.77873E 00 | 0.98305E 00 |
| FORM SAI | 0.88803E 00 | 0.81646E 00 | 0.77792E 00 | 0.55869E 00 | 0.88428E 00 | 0.77873E 00 | 0.10000E 01 | 0.88053E 00 |
| S 5+6 | 0.94071E 00 | 0.89759E 00 | 0.67816E 00 | 0.68209E 00 | 0.93126E 00 | 0.98305E 00 | 0.88053E 00 | 0.10000E 01 |
| FORM DIF | 0.89993E 00 | 0.89146E 00 | 0.60475E 00 | 0.73750E 00 | 0.91462E 00 | 0.98922E 00 | 0.79346E 00 | 0.97921E 00 |

T A B

PROGRAMAS LISTACORES

MATRIZ DOS COEFICIENTES DE CORRELAÇÃO

VARIÁVEL FORM DIF

| | |
|----------|-------------|
| DIAS | 0.85993E 00 |
| ARC.ENT. | 0.89146E 00 |
| ARC.SAI. | 0.60475E 00 |
| RELATOR. | 0.73750E 00 |
| S 1+2+3 | 0.91462E 00 |
| FORM ENT | 0.98922E 00 |
| FORM SAI | 0.79346E 00 |
| S 5+6 | 0.97921E 00 |
| FORM DIF | 0.10000E 01 |

T A B

PROGRAMAS LISTADOES

ANALISE DE REGRESSAO

| | |
|--------------------------|--------|
| VARIABEL DEPENDENTE | DIA5 |
| C. PADRAO DOS RESIDUOS | 3.7004 |
| ERRO PADRAO DA MEDIA | 0.7121 |
| COEFICIENTE - R | 0.9472 |
| RCUADRADO | 0.8972 |
| RQUADRADO AJUSTADO(G.L.) | 0.8972 |

| | |
|-------------|--------|
| F P/RETIRAR | 0.0500 |
| F P/INSERIR | 0.0300 |
| TOLERANCIA | 0.0001 |

VARIABEL INSERIDA ARQ. ENT.

| VARIABEL | COEF - B | ERRO DE B | TESTE - T | R-PARCIAL | COEF - BETA | ERRO DE BETA | ANTI-LOG DE B |
|-----------|----------|-----------|-----------|-----------|-------------|--------------|---------------|
| ARQ. ENT. | 7.1734 | 0.4855 | 14.7730 | 0.9472 | 0.9472 | C.0641 | 1304.3621 |

| | | | |
|-----------------|--------|----------|----------|
| TERMO CONSTANTE | 4.8061 | ANTI-LOG | 122.2566 |
|-----------------|--------|----------|----------|

ANALISE DE VARIANCA

| CRIGEM | G.L. | SOMA DOS QUADRADOS | MEDIA DOS QUADRADOS | TESTE - F |
|-----------|------|--------------------|---------------------|----------------|
| MEDIA | 1 | 0.11326259E 05 | 0.11326259E 05 | |
| REGRESSAO | 1 | 0.29884142E 04 | 0.29884142E 04 | 0.21824296E 03 |
| ERRO | 25 | 0.34232652E 03 | 0.13693060E 02 | |

T A B

PROGRAMAS LISTADORES

ANALISE DE REGRESSAO

| | |
|-------------------------|--------|
| VARIAVEL DEPENDENTE | DIAS |
| C.PACRAO DOS RESIDUOS | 2.9C08 |
| ERRO PACRAO DA MEDIA | 0.5582 |
| COEFICIENTE - R | 0.9692 |
| QUADRADO | 0.9393 |
| QUADRADO AJUSTADO(G.L.) | 0.9369 |
| F P/RETIRAR | 0.0500 |
| F P/INSERIR | 0.0300 |
| TOLERANCIA | 0.0001 |

VARIAVEL INSERIDA S 5+6

| VARIAVEL | COEF - B | ERRO DE B | TESTE - T | R-PARCIAL | COEF - BETA | ERRO DE BETA | ANTI-LOG DE B |
|----------|----------|-----------|-----------|-----------|-------------|--------------|---------------|
| ARG.ENT. | 4.0079 | 0.8635 | 4.6413 | 0.6877 | 0.5292 | 0.1140 | 55.0325 |
| S 5+6 | 1.4479 | 0.3545 | 4.0840 | 0.6403 | 0.4656 | 0.1140 | 4.2543 |

| | | | |
|-----------------|--------|----------|---------|
| TERMO CONSTANTE | 3.8937 | ANTI-LOG | 49.0954 |
|-----------------|--------|----------|---------|

ANALISE DE VARIANCA

| CRIGEM | G.L. | SOMA DOS QUADRADOS | MEDIA DOS QUADRADOS | TESTE - F |
|-----------|------|--------------------|---------------------|----------------|
| MEDIA | 1 | 0.11326259E 05 | 0.11326259E 05 | |
| REGRESSAO | 2 | 0.31287767E 04 | 0.15643883E 04 | 0.18590102E 03 |
| ERRO | 24 | 0.20196402E 03 | 0.84151678E 01 | |

T A B

PROGRAMAS LISTADORES

ANALISE DE REGRESSAO

| | |
|--------------------------|--------|
| VARIABEL DEPENDENTE | DIAS |
| D.PADRAO DOS RESIDUOS | 2.4485 |
| ERRO PADRAO CA MEDIA | 0.4712 |
| COEFICIENTE - R | 0.9790 |
| QUADRADO | 0.9586 |
| QUADRADO AJUSTADO (G.L.) | 0.9551 |
| F P/RETIRAR | 0.0500 |
| F P/INSERIR | 0.0300 |
| TOLERANCIA | 0.0001 |

VARIABEL INSERIDA FORM DIF

| VARIABEL | COEF - B | ERRO DE B | TESTE - T | R-PARCIAL | COEF - BETA | ERRO DE BETA | ANTI-LOG DE B |
|----------|----------|-----------|-----------|-----------|-------------|--------------|---------------|
| FORM DIF | -2.4020 | 0.7347 | -3.2691 | -0.5632 | -0.6907 | 0.2112 | 0.0905 |
| ARG-ENT. | 4.3449 | 0.7361 | 5.9025 | 0.7761 | 0.5737 | 0.0972 | 77.0863 |
| S 5+6 | 3.4267 | 0.6752 | 5.0750 | 0.7268 | 1.1020 | 0.2171 | 30.7765 |

| | | | |
|-----------------|--------|----------|---------|
| TERMO CONSTANTE | 4.1114 | ANTI-LOG | 61.0330 |
|-----------------|--------|----------|---------|

ANALISE DE VARIANCA

| ORIGEM | G.L. | SOMA DOS QUADRADOS | MEDIA DOS QUADRADOS | TESTE - F |
|-----------|------|--------------------|---------------------|----------------|
| MEDIA | 1 | 0.11326259E 05 | 0.11326259E 05 | |
| REGRESSAO | 3 | 0.31928510E 04 | 0.10642836E 04 | 0.17752256E 03 |
| ERRO | 23 | 0.13788965E 03 | 0.59952022E 01 | |

T A B

PROGRAMAS LISTAORES

ANALISE DE REGRESSAO

VARIAVEL DEPENDENTE DIAS
 O. PADRAO DOS RESIDUOS 2.2794
 ERRO PADRAO DA MEDIA 0.4386
 COEFICIENTE - R 0.9826
 RCUADRADO 0.9656
 RCUADRADO AJUSTADO (G.L.) 0.9612

F P/RETIRAR 0.0500
 F P/INSERIR 0.0300
 TOLERANCIA 0.0001

VARIAVEL INSERIDA S 1+2+3

| VARIAVEL | COEF - B | ERRO DE B | TESTE - T | R-PARCIAL | COEF - BETA | ERRO DE BETA | ANTI-LOG DE B |
|-----------|----------|-----------|-----------|-----------|-------------|--------------|---------------|
| FORM DIF | -2.6378 | 0.6929 | -3.8068 | -0.6301 | -0.7585 | 0.1992 | 0.0715 |
| ARC. ENT. | 6.7683 | 1.3280 | 5.0963 | 0.7358 | 0.8937 | 0.1753 | 869.8465 |
| S 1+2+3 | -2.3336 | 1.0954 | -2.1302 | -0.4135 | -0.4478 | 0.2102 | 0.0969 |
| S 5+6 | 4.0369 | 0.6907 | 5.8439 | 0.7798 | 1.2983 | 0.2221 | 56.6513 |

TERMO CONSTANTE 5.4765 ANTI-LOG 239.0300

ANALISE DE VARIANCA

| ORIGEM | G.L. | SOMA DOS QUADRADOS | MEIA DOS QUADRADOS | TESTE - F |
|-----------|------|--------------------|--------------------|----------------|
| MEDIA | 1 | 0.11326259E 05 | 0.11326259E 05 | |
| REGRESSAO | 4 | 0.32164302E 04 | 0.80410757E 03 | 0.15475720E 03 |
| ERRO | 22 | 0.11431045E 03 | 0.51959298E 01 | |

ANALISE DE REGRESSAO

| | |
|--------------------------|--------|
| VARIABEL DEPENDENTE | DIAS |
| C.PADRAO DOS RESIDUOS | 2.2398 |
| ERRO PADRAO DA MEDIA | 0.4310 |
| COEFICIENTE - R | 0.9940 |
| RCUACRACC | 0.9683 |
| RCUACRACC AJUSTADO(G.L.) | 0.9626 |
| F P/RETIRAR | 0.0500 |
| F P/INSERIR | 0.0300 |
| TOLERANCIA | 0.0001 |

VARIABEL INSERIDA FORM ENT

| VARIABEL | COEF - B | ERRO DE B | TESTE - T | R-PARCIAL | COEF - BETA | ERRO DE BETA | ANTI-LOG DE B |
|----------|----------|-----------|-----------|-----------|-------------|--------------|---------------|
| FORM DIF | -1.4906 | 1.0958 | -1.3603 | -0.2845 | -0.4286 | 0.3151 | 0.2252 |
| ARC.ENT. | 7.4977 | 1.4145 | 5.3002 | 0.7564 | 0.9900 | 0.1867 | 1803.9624 |
| S 1+2+3 | -3.4081 | 1.3437 | -2.5363 | -0.4842 | -0.6540 | 0.2578 | 0.0331 |
| FORM ENT | -2.1995 | 1.6463 | -1.3360 | -0.2798 | -0.5344 | 0.4000 | 0.1108 |
| S 5+6 | 4.9945 | 0.9871 | 5.0593 | 0.7411 | 1.6063 | 0.3174 | 147.6091 |

| | | | |
|-----------------|--------|----------|----------|
| TERMO CONSTANTE | 5.0183 | ANTI-LOG | 151.1648 |
|-----------------|--------|----------|----------|

ANALISE DE VARIANCA

| ORIGEM | G.L. | SOMA DOS QUADRADOS | MEDIA DOS QUADRADOS | TESTE - F |
|-----------|------|--------------------|---------------------|----------------|
| MECIA | 1 | 0.11326259E 05 | 0.11326259E 05 | |
| REGRESSAO | 5 | 0.32253850E 04 | 0.64507701E 03 | 0.12857988E 03 |
| ERRO | 21 | 0.10535565E 03 | 0.50169357E 01 | |

ANALISE DE REGRESSAO

| | |
|--------------------------|--------|
| VARIABEL DEPENDENTE | DIAS |
| C.PADRAO DOS RESIDUOS | 2.2378 |
| ERRO PADRAO DA MEDIA | 0.4306 |
| COEFICIENTE - R | 0.9848 |
| RQUADRACO | 0.9699 |
| RQUADRACO AJUSTADO(G.L.) | 0.9627 |

| | |
|-------------|--------|
| F P/RETIRAR | 0.0500 |
| F P/INSERIR | 0.0300 |
| TOLERANCIA | 0.0001 |

VARIABEL INSERIDA ARQ.SAI.

| VARIABEL | COEF - B | ERRO DE B | TESTE - T | R-PARCIAL | COEF - BETA | ERRO DE BETA | ANTI-LOG DE B |
|----------|----------|-----------|-----------|-----------|-------------|--------------|---------------|
| FCRM DIF | -0.9800 | 1.2042 | -0.8138 | -0.1790 | -0.2818 | 0.3462 | 0.3752 |
| ARC.ENT. | 10.2969 | 3.0904 | 3.3318 | 0.5974 | 1.3596 | 0.4080 | 29641.9838 |
| ARQ.SAI. | 2.4319 | 2.3877 | 1.0185 | 0.2220 | 0.1376 | 0.1351 | 11.3813 |
| S 1+2+3 | -5.7303 | 2.6459 | -2.1657 | -0.4358 | -1.0997 | 0.5077 | 0.0032 |
| FORM ENT | -2.3993 | 1.6565 | -1.4483 | -0.3081 | -0.5830 | 0.4025 | 0.0907 |
| S 5+6 | 4.6645 | 1.0381 | 4.4930 | 0.7087 | 1.5001 | 0.3338 | 106.1180 |

| | | | |
|-----------------|--------|----------|----------|
| TERMO CONSTANTE | 6.6109 | ANTI-LOG | 743.2036 |
|-----------------|--------|----------|----------|

ANALISE DE VARIANCA

| CRIGEM | G.L. | SOMA DOS QUADRADOS | MEDIA DOS QUADRADOS | TESTE - F |
|-----------|------|--------------------|---------------------|----------------|
| MEDIA | 1 | 0.11326259E 05 | 0.11326259E 05 | |
| REGRESSAO | 6 | 0.32305802E 04 | 0.53843004E 03 | 0.10751344E 03 |
| ERRO | 20 | 0.10016049E 03 | 0.50080249E 01 | |

VALORES ESTIMADOS

| OBS | ATUAL | ESTIMADO | RESIDUO |
|-----|------------|------------|-------------|
| 1 | 0.170CE 02 | 0.1427E 02 | 0.2727E 01 |
| 2 | 0.280CE 02 | 0.2986E 02 | -0.1862E 01 |
| 3 | 0.360CE 02 | 0.3306E 02 | 0.2937E 01 |
| 4 | 0.3900E 02 | 0.4050E 02 | -0.1509E 01 |
| 5 | 0.260CE 02 | 0.2406E 02 | 0.1939E 01 |
| 6 | 0.140CE 02 | 0.1041E 02 | 0.3583E 01 |
| 7 | 0.700CE 01 | 0.1041E 02 | -0.3416E 01 |
| 8 | 0.900CE 01 | 0.1041E 02 | -0.1416E 01 |
| 9 | 0.1400E 02 | 0.1170E 02 | 0.2298E 01 |
| 10 | 0.1500E 02 | 0.1306E 02 | 0.1931E 01 |
| 11 | 0.700CE 01 | 0.1041E 02 | -0.3416E 01 |
| 12 | 0.150CE 02 | 0.1636E 02 | -0.1366E 01 |
| 13 | 0.180CE 02 | 0.1891E 02 | -0.9197E 00 |
| 14 | 0.2300E 02 | 0.2469E 02 | -0.1690E 01 |
| 15 | 0.310CE 02 | 0.2815E 02 | 0.2849E 01 |
| 16 | 0.3200E 02 | 0.3144E 02 | 0.5506E 00 |
| 17 | 0.130CE 02 | 0.1041E 02 | 0.2583E 01 |
| 18 | 0.130CE 02 | 0.1427E 02 | -0.1272E 01 |
| 19 | 0.100CE 02 | 0.1041E 02 | -0.4168E 00 |
| 20 | 0.420CE 02 | 0.4196E 02 | 0.3108E-01 |
| 21 | 0.180CE 02 | 0.1891E 02 | -0.9197E 00 |
| 22 | 0.260CE 02 | 0.2546E 02 | 0.5373E 00 |
| 23 | 0.100CE 02 | 0.1041E 02 | -0.4168E 00 |
| 24 | 0.1700E 02 | 0.1755E 02 | -0.5536E 00 |
| 25 | 0.4800E 02 | 0.4910E 02 | -0.1105E 01 |
| 26 | 0.900CE 01 | 0.1041E 02 | -0.1416E 01 |
| 27 | 0.1600E 02 | 0.1626E 02 | -0.2685E 00 |

COEFICIENTE DE DURBIN-WATSON 2.3013

EXECUCAO TERMINADA