COLÉGIO LUTERANO CONCÓRDIA

CURSO TÉCNICO EM INFORMÁTICA

POLÍGRAFO DE ANÁLISE DE SISTEMAS

Prof. Marcos Ricardo Kich

2001/01

ÍNDICE

1. ANÁLISE DE SISTEMAS

1.1 Sistemas

Definição

- ✓ Um grupo de itens que interagem entre si ou que sejam interdependentes, formando um todo unificado;
- ✓ Um conjunto organizado de doutrinas, idéias ou princípios, habitualmente previsto para explicar a organização ou funcionamento de um conjunto sistemático;
- ✓ Conjunto particular de instrumentos e convenções adotadas com o fim de dar uma informação;

1.2 Tipos mais comuns de Sistemas

- ✓ Sistemas Naturais
  - ➢ Sistemas Estelares: Galáxia, Sistemas Solares, etc.;
  - ➢ Sistemas Geológicos: Rios, Cadeias de Montanhas, etc.;
  - ➢ Sistemas Moleculares: Organizações Complexas de Átomos;

- ✓ Sistemas Feitos pelo Homem
  - ➢ Sistemas Sociais: Organizações de leis doutrinais, costumes, etc.;
  - ➢ Sistemas de Transporte: Rede Rodoviária, Canais, Linhas Aéreas, Petroleiros, e Semelhantes;
  - ➢ Sistemas de Comunicação: Telefone, Telex, etc.;
  - ➢ Sistemas de Manufatura: Fábricas, Linhas de Montagem, etc.;
  - ➢ Sistemas Financeiros: Contabilidade, Controle de Estoques

* Hoje, a maioria desses sistemas usa computadores.

Por que alguns Sistemas de Informação não devem ser automatizados?

Pode haver muitas razões, mas aqui estão algumas das mais comuns:

✓ **Custo:** pode ser mais barato continuar executando as funções do sistema e armazenamento as informações manualmente. Nem sempre é verdade que os computadores sejam mais rápidos e mais baratos do que o modo “antigo”;

✓ **Conforto:** Um sistema automatizado pode ocupar espaço em demasia, excessivo ruído, geram muito calor ou consumir eletricidade, geram muito calor ou consumir eletricidade, demais ou, em termos gerais, a presença de um deles pode ser uma dor de cabeça. Isso está se tornando menos verdadeiro com a penetrante influência dos microprocessadores mas ainda é um fator a considerar;

✓ **Segurança:** se o sistema de informação mantém dados importantes e confidenciais, o usuário pode achar que um sistema automatizado não seja suficiente seguro, podendo preferir manter a capacidade de conservar as informações protegidas debaixo de chave;

✓ **Manutenibilidade:** O usuário pode argumentar que um sistema computadorizado de informações poderia ser econômico, só que não há ninguém na organização que possa mantém o hardware e/ou o software de computador, de forma que ninguém estaria capacitado a reparar o sistema no caso de um defeito e não haveria quem pudesse efetuar modificações e melhoramentos.

✓ **Política:** A comunidade de usuários pode achar que os computadores são uma ameaça a seus empregados ou que tornam seu trabalho tedioso e “mecânico”, ou podem ter uma dúzia de outras razões que o analista de sistemas pode considerar como irracionais. Mas como todo sistema pertence aos usuários, a opinião deles está acima de tudo. Se ele não desejam um sistema automatizado, farão o que for possível para que o sistema fracasse caso ele lhes seja imposto.

Embora haja muitos tipos diferentes de sistemas automatizados, todos têm componentes comuns:

✓ **Hardware de componentes:** CPU, Terminais, Impressoras, Unidades de Fitas Magnéticas.

✓ **Programas de Sistemas tais como:** Sistemas Operacionais, Sistemas de Banco de Dados e Programas de Controle de Telecomunicações, além dos programas aplicativos que executam as funções desejadas pelo usuário;

✓ **Pessoas:** aqueles que operam o sistema, que fornecem as entradas e utilizam as saídas, e as que desempenham atividades de processamento manual em um sistema.

✓ **Dados:** as informações que o sistema conserva por um período de tempo;

- ✓ **Procedimentos:** determinações e instruções formais para a operação do sistema.

Para pensarmos em um sistema devemos considerar o seguinte:

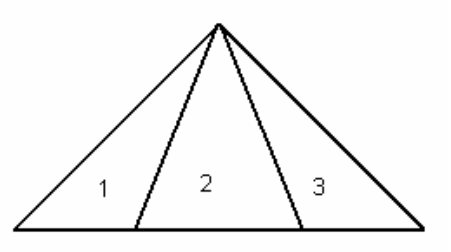
- ✓ Os objetivos totais do sistema (Realmente declarado);

- ✓ O ambiente do sistema: responsável pela elaboração, implantação e acompanhamento dos planos, que em função dos planos, que em funções do sistema existente, alocação aos diversos componentes os recursos disponíveis, de modo a que os objetivos do sistema sejam alcançados com o máximo rendimento.

1.3 Sistema de Informação

Se desenvolvem em uma empresa segundo duas dimensões: os componentes da empresa e o nível de decisão da empresa.

Os componentes são os diversos setores que executam funções diferentes e necessárias ao funcionamento da empresa (produção, pesquisa, marketing, etc.).



Níveis de decisão obedecem a hierarquia existente na empresa e são conhecidos como estratégicos, táticos e operacionais.

A divisão mais prática dos sistemas automatizados:

- ✓ Sistemas Batch (em lotes);
- ✓ Sistemas on-line;
- ✓ Sistemas de Tempo-Real;
- ✓ Sistemas de Apoio à Decisão;
- ✓ Sistemas Baseados no Conhecimento.

Alguns princípios “gerais” de particular interesse para os que constróem sistemas automatizados de informações são os seguintes:

✓ Quanto mais específico é um sistema, menos capaz ele é de se adaptar a circunstâncias diferentes;

✓ Quanto maior for um sistema, maior o número de seus recursos que serão destinados à manutenção diária;

✓ Os Sistemas sempre fazem parte de sistemas maiores e sempre podem ser divididos em sistemas menores;

✓ Os Sistemas Crescem

1.3.1 Funções Relacionadas com Sistemas de Processamento de Dados

- ✓ Usuários;
- ✓ Gerentes;
- ✓ Auditores, Pessoal do Controle de Qualidade e Mantenedores do Padrões;
- ✓ Analistas de Sistemas;
- ✓ Projetistas de Sistemas;
- ✓ Programadores;
- ✓ Operadores.

1.4 Usuários

Usuário é a pessoa (ou grupo de pessoas) para quem o sistema é constituído. Ele ou ela é a pessoa a quem o Analista de Sistemas entrevistará, muitas vezes detalhadamente, para saber que características o novo sistema deverá ter para ser bem-sucedido.

Assim como em tantas outras atividades, “o cliente sempre tem razão”, não importando quão exigente, desagradável ou ilógico ele ou ela possa parecer e o cliente, afinal de contas, é quem paga pelo sistema e habitualmente tem o direito ou a possibilidade de se recusar a pagar se ele ou ela não ficam satisfeito com o produto.

Sempre que possível, o Analista de Sistemas deve tentar estabelecer contato direto com usuário. É importante que haja reuniões regulares, frente a frente com a pessoa que irá afinal receber o sistema. Na verdade, é até melhor se o usuário for um membro de tempo integral da equipe de projeto.

Se não for possível a comunicação direta com o usuário, então a documentação produzida pelo analista de sistema torna-se ainda mais crítica.

Dois modos de classificar os usuários:

- ✓ Por tipo de função, ou por nível de supervisão;
- ✓ Por nível de experiência com processamento de dados.

Classificação dos usuários por tipo de função:

- ✓ Usuário Operativo:
  - ➢ Normalmente tem visão local;
  - ➢ Executa a função do sistema;
  - ➢ Tem visão física do sistema.

- ✓ Usuário Supervisor
  - ➢ Pode ou não Ter visão local;
  - ➢ Normalmente conhecer a operação;
  - ➢ Orientando por considerações orçamentárias;
  - ➢ Muitas vezes age como intermediário entre os usuários e os níveis mais elevados da direção.

- ✓ Usuário Executivo
  - ➢ Tem visão global;
  - ➢ Tem iniciativa sobre o projeto;
  - ➢ Não tem experiência operativa;
  - ➢ Tem preocupações estratégicas.

1.4.1 Classificação dos Usuários

- ✓ Usuário Amador;
- ✓ Usuário “Novato Arrogante”;
- ✓ Usuário Experiente;

Veja o seguinte algoritmo

Regra Nº1

O Usuário tem sempre a razão

Razão Nº2

Se o usuário não tiver razão, volta para regra Nº1

1.5 Analista de Sistemas

É o profissional responsável capaz de preparar especificações para sistemas de P.D. e realizar estudos de viabilidade. Os analistas envolvidos na fase de Concepção e Projeto Lógico são mais orientados para a empresa e comportamento humano. Os analistas envolvidos na fase de Projeto Físico são mais orientados para os equipamentos de P.D.

1.5.1 Perfil do Analista

- ✓ Profundo conhecedor da tecnologia de Processamento de Dados;
- ✓ Político hábil para projeto intersetorial;
- ✓ Capaz de avaliar as necessidades de Processamento de Dados;
- ✓ Intermediário dos desejos megalomaníacos dos usuários versus interesses do CPD;
- ✓ Receptivo as mudanças;
- ✓ Líder na equipe do projeto;
- ✓ Enérgico e perseverante;
- ✓ Mil Faces;
- ✓ Afável (fácil e cortês nas relações);
- ✓ Responsável;
- ✓ Capaz de sintetizar várias opiniões, inclusive a sua.

1.5.2 Papel do Analista

- ✓ Ver o todo e a relação entre as partes;
- ✓ Identificar como vive a organização;
- ✓ Ensinar ao superior a solicitar trabalhos exeqüíveis (seguir até o fim, que se pode executar);
- ✓ Compatibilizar conflitos políticos versus tecnocrata;
- ✓ Enxergar falhas;
- ✓ Transmitir idéias e atividades aos outros;
- ✓ Entender o que os outros estão tentando dizer;

1.6 Técnicas de Análise

- ✓ Identificação dos fatores causais dos fenômenos;
- ✓ Reunião das informações para análise;

- ✓ Desenvolvimento das interpretações sobre a situação apresentada;

1.6.1 Tipos de Técnicas

- ✓ Questionário
  - ➢ Menos dispendioso
  - ➢ Universo maior
  - ➢ Resultados rápidos
  - ➢ Fácil manipulação e tratamento estatístico

- ✓ Observação
  - ➢ Validar informações

- ✓ Entrevista
  - ➢ Instrumento de integração analista usuário;
  - ➢ Exige preparação e desibinição;
  - ➢ Padronizada ou não, aberta ou fechada

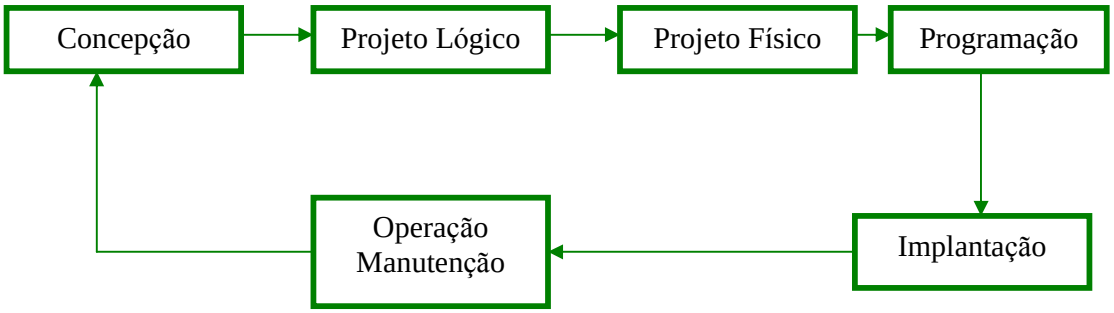
Funções / *Fases*	Analista	Programador	Operador
Concepção	X		
Projeto Lógico	X		
Projeto Físico	X		
Programação		X	
Implementação	X	X	X
Operação			X
Manutenção	X	X	X

Tecnocrata - político, administrador ou funcionário que procura soluções meramente técnicas e/ou racionais, desprezando os aspectos humanos e sociais dos problemas

2. CICLOS DE VIDA DE UM SISTEMA

Os sistemas possuem “Ciclos de Vida” Bem definidos, todos eles passam por estágios de concepção, desenvolvimento e vida útil.

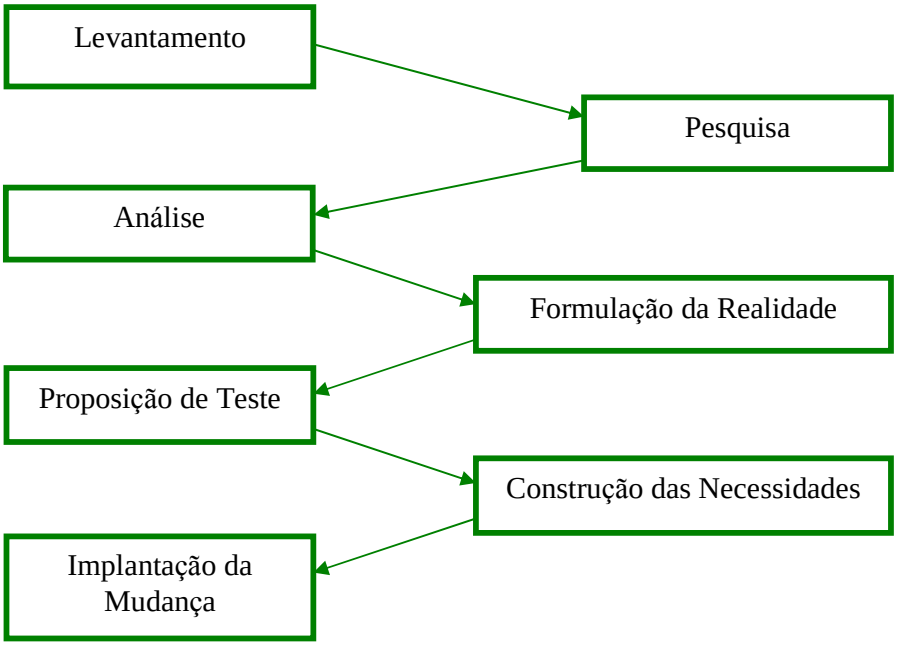
Ciclo de Vida de um sistema de P.D.



2.1 Ciclo de Vida Dialético

- ✓ Processo de descrição exata do real;

- ✓ Desenvolvimento de Processos gerados por oposições que provisoriamente se resolvem em unidades;



2.1.1 Concepção

É analisada a conveniência de se desenvolver um novo sistema ou de se introduzir modificações no sistema existente, de forma a melhor atender à organização.

A fase de concepção pode ser ainda divida em duas subfases:

✓ **Percepção da Necessidade** - é a identificação precisa dos problemas que determinam o desenvolvimento dos estudos de reformulação do sistema atual, bem como a fixação dos objetivos que deverão ser alcançados com a introdução dos novos procedimentos. A necessidade de um novo sistema pode ter sua origem:

- Análise dos planos da empresa (P.D.I.);
- No reconhecimento da existência de um problema que necessite atenção;
- Na descoberta de uma oportunidade de ser melhorado o rendimento.

✓ **Estudo de viabilidade** - onde se estuda as vantagens e desvantagens decorrentes das diversas alternativas propostas para a solução dos problemas organizacionais, escolhendo-se, a alternativa mais conveniente, sob os pontos de vista econômico, técnico e operacional são os seguintes subfatores:

- Viabilidade econômica;
- Viabilidade técnica;

- Viabilidade financeira;
- Viabilidade de mão-de-obra;
- Viabilidade de recursos de suporte;
- Viabilidade de cronograma;
- Viabilidade social;

Fase de ação na concepção do sistema

- ✓ Consultoria de processos;
- ✓ Metodologias de resolução de problemas complexos;
- ✓ Questionários orientados para problemas;
- ✓ Levantamento de dados;
- ✓ Entrevistas;
- ✓ Reuniões e seminários;

2.1.2 Projeto Lógico

Nesta fase é especificado detalhadamente o sistema que vai ajudar a melhorar o funcionamento da organização. Está dividido em duas subfases:

- ✓ Analise de Sistemas Existentes (o que é, como é, porque);
- ✓ Projeto do Novo Sistema;

Quando concluído o Projeto Lógico do Sistema, deve atender aos seguintes requisitos:

- ✓ Satisfazer aos objetivos da organização;
- ✓ As suas especificações devem ter sido aceitas pelos usuários;
- ✓ A lógica de processamento do sistema deve estar bem definida;

O resultado final desta fase é um documento que apresenta o sistema lógico proposto. A maior ênfase desta fase se refere aos benefícios que o sistema proposto terá para a organização. O principal talento do Projeto Lógico é identificar os benefícios e projetar o sistema possível de maneira a alcançar estes benefícios.

Fases do Projeto Lógico:

- ✓ Técnicas de organização e métodos (levantamento e análise de procedimentos);
- ✓ Técnicas de elaboração, análise e avaliação de projetos;
- ✓ Técnicas de contabilidade, finanças, produção, estoques, etc.;
- ✓ Técnicas de documentação;
- ✓ Consultoria de processo.

2.1.3 Projeto Físico

O projeto físico consiste em definir, dentro das restrições que possam existir, os seguintes pontos:

- ✓ A organização do processamento (on-line, batch, etc.);
- ✓ O equipamento utilizado;
- ✓ O sistema operacional utilizado;
- ✓ Os softwares utilizados e de suporte necessários;
- ✓ As especificação dos programas do sistema;
- ✓ A organização dos bancos de dados;
- ✓ Os controles do sistema;

Nesta fase são produzidas as especificações para os componentes do sistema físico que processará as informações.

Idealmente, este Projeto Físico é um detalhamento do sistema usado no estudo de viabilidade.

Fase de Projeto Físico:

- ✓ Técnicas de seleção de hardware;
- ✓ Técnicas de documentação de sistemas e programas;
- ✓ Técnicas de estruturação de arquivos;
- ✓ Técnicas de database e comunicação de dados.

2.1.4 Programação

Na fase de programação são constituídas as microestruturas que compõem o sistema de dados, isto é, elaborando os programas de computador que implementam a estrutura de dados, definida na fase anterior.

2.1.4.1 A programação engloba:

- ✓ A revisão da especificação dos programas;
- ✓ O desenvolvimento das lógicas dos programas;
- ✓ A codificação dos programas;
- ✓ A construção dos arquivos;
- ✓ Os testes dos programas;

- ✓ A elaboração dos manuais de operação;

2.1.4.2 Fases da Programação

- ✓ Técnicas de programação, avaliação e teste de programas;
- ✓ Técnicas de gerência de equipes de programação
- ✓ Técnicas de acompanhamento e avaliação de atividades de programação.

2.1.5 Implantação

A fase de implantação tem início quando os componentes do sistema, já desenvolvidos e testados individualmente, são agora reunidos para teste e simulação do sistema como um todo.

Este fase envolve atividades ligadas ao treinamento de pessoal, ao teste e simulação do novo sistema, à revisão dos procedimentos operacionais que foram anteriormente estabelecidos e, finalmente, à conversão do antigo para o novo sistema.

Esta fase termina quando o sistema é aceito pela organização e é entregue ao C.P.D.

Fase de Implantação

- ✓ Técnicas de simulação;
- ✓ Técnicas de implantação de sistemas;
- ✓ Técnicas de avaliação e performance de sistemas.

2.1.6 Operação

O maior cuidado nesta fase é no sentido de verificar se as instruções de operação estão sendo correntemente seguidas. Um controle gerencial é necessário para assegurar que o sistema está produzindo resultados corretos e está sendo operado corretamente.

Subfase

Produção

Manutenção : Principais razões que podem gerar uma manutenção em um sistema são :

- Alterações internas e externas;
- Alterações na configuração do equipamento;
- Correção de erros;
- Modificações para melhoria da “Performance” do sistema.

3. AJUDAS DE DIAGRAMAS

Diagramas bons e claros desempenham um papel muito importante no projeto se sistemas e programas complexos.

Os filósofos freqüentemente têm mencionado que a nossa capacidade de pensar depende da linguagem usada para pensar. Enquanto a humanidade usava apenas os algarismos Romanos, as pessoas não podiam multiplicar ou dividir. A adoção dos algarismos arábicos veio contribuir para ampliar essa capacidade. Com o computador podemos criar processos mais complexos que aqueles executados manualmente. Diagramas apropriados podem nos ajudar na visualização e no desenvolvimento de tais processos.

Se somente uma pessoa estiver desenvolvimento um projeto de sistema ou programa, os diagramas se tornam um auxílio para a clareza de pensamento. Uma má escolha no tipo de técnica de diagramação pode inibir seu pensamento, enquanto que uma boa escolha pode acelerar seu trabalho e melhorar a qualidade dos resultados.

Quando várias pessoas trabalham em um sistema ou programa, os diagramas são essenciais para a comunicação. Uma técnica formal de diagramação é necessária para possibilitar aos projetistas o entendimento e a troca de idéias, além de permitir o entrosamento entre os diversos componentes de um mesmo projeto.

Quando os sistemas são modificados, diagramas compreensíveis são uma ajuda essencial para a manutenção. Eles possibilitam a uma nova equipe entender o funcionamento dos programas e projetar as modificações. Estas modificações normalmente afetam outras partes do sistema. Os diagramas da estrutura do programa ou do sistema possibilitam a visualização dos efeitos de uma modificação.

Ao se depurar programas ou sistemas, os diagramas novamente desempenham um papel importante, auxiliando no acompanhamento das funções e na determinação de falhas.

Quanto maior a equipe, maior a necessidade de precisão nos diagramas. Ë difícil ou impossível para membros de uma grande equipe entender em detalhe o trabalho dos outros. Mas é possível e necessário que cada membro da equipe se familiarize com uma visão geral do sistema e conheça onde os componentes de cada trabalho se encaixam e se completam. Ele deve estar apto a desenvolver o seu componente em perfeita harmonia e compatibilidade com o restante da equipe. Ele possui diagrama precisos do inter-relacionamento do seu componente. Quando um programador modifica o seu projeto, este não deve afetar o projeto dos outros programadores, a menos que isto seja inevitável. As ligações entre os trabalhos de diferentes programadores devem ser imutáveis para empreender com sucesso as necessidades de precisão do todo o sistema.

As técnicas de diagramação estruturada oferecem muitas vantagens. Primeiro, elas combinam notações gráficas e narrativas para melhorar o entendimento. Os gráficos são especialmente úteis porque tendem a ser menos ambíguas que uma descrição narrativa. Também, porque tendem a ser mais precisos e podem ser desenhados em menor tempo do que a elaboração de uma descrição narrativa.

Segundo, as técnicas de diagramação estruturada suportam uma abordagem de desenvolvimento top-down. Elas podem descrever um sistema ou um programa em vários níveis de detalhe durante o processo de decomposição funcional. Elas esclarecem as etapas e os resultados deste processo através do fornecimento de uma forma padronizada de descrição da lógica de procedimento e da estrutura de dados.

As técnicas de diagramação estruturada ajudam os projetistas a manipular o grande volume de detalhes que se originam do processo de desenvolvimento de programas.

As técnicas de diagramação aplicadas à área de processamento de dados ainda estão evoluindo; isto é necessário porque encontramos sérias deficiências nas técnicas utilizadas atualmente. Os fluxogramas estão saindo de uso porque não fornecem uma visão estruturada do programa. Algumas das técnicas primitivas de diagramação necessitam ser modificadas porque falham na tentativa de representar algumas construções importantes. Nós estamos inventando

métodos mais rigorosos para melhorar as especificações. Vários melhoramentos são necessários e possíveis neste processo. Estes melhoramentos trazem novos métodos de diagramação.

Os diagramas são usados em quatro áreas principais :

✓ **Análise do sistema de informações gerais -** Um modelo completo de empresa com seus sistemas pode ser desenhado. As funções e os processos são decompostos hierarquicamente e esquematiza-se o fluxo global de dados entre as funções e processos.

✓ **Arquitetura do programa -** A arquitetura completa do programa ou de um conjunto de programas é desenhada, mostrando os módulos separados.

✓ **Detalhe do programa -** A lógica detalhada dentro de um módulo de programa é desenvolvida.

✓ **Estrutura de dados -** Uma estrutura completa dos dados é desenhada. Ë importante distinguir modelos de banco de dados e representação de arquivo.

As áreas que são aplicáveis as técnicas de diagramação:

Técnicas de Diagramação	**Análise do Sistema Geral**	**Arquitetura do Programa**	**Lógica Detalhada do Programa**	**Estrutura de Arquivo**	**Estrutura de Banco de Dados**
Diagramas de Fluxo de Dados	X				
Decomposição Funcional	X				
Quadros de Estrutura	X	X			
HIPO	X	X	X		
Warnier-orr	X	X	X	X	
Michael Jackson		X		X	
Fluxogramas			X		
Linguagem Estruturadas e Pseudocódigo			X		
Nassi-Shneiderman			X		
Diagramas de Ação	X	X	X	X	
Árvores de Decisão	X		X		
Tabelas de Decisão			X		
Diagramas em Bolha					X
Quadros de Relacionamento-Entidades					X

Mapa de Acesso aos Dados		X			
Diagramas de Ação de Banco de Dados	X	X	X	X	
HOS	X	X	X	X	

O Envolvimento do Usuário Final

Atualmente é muito importante o envolvimento dos usuários finais, para melhorar a comunicação com os analistas de sistemas e para entender e discutir os diagramas.

De forma crescente, alguns usuários finais têm criado suas próprias aplicações através do uso de linguagens de quarta geração amigáveis. Onde acontece isto, seria interessante que eles esquematizassem suas necessidades e trabalhos juntamente com o analista que irá desenvolver a aplicação, talvez no centro de informações. A computação dirigida aos usuários é uma tendência muito importante para capacitar os usuários finais a resolverem os seus problemas através dos computadores.

Por estas razões, as técnicas de diagramação devem ser amigáveis aos usuários. Elas devem ser projetadas para encorajar o entendimento, participação e esquematização do usuário. Muitas técnicas de diagramação de PD foram projetados apenas para uso dos profissionais da área. Para que um diagrama seja amigável ao usuário, ele deve ser mais óbvio possível, evitando símbolos e códigos mnemônicos de entendimento restrito.

Ferramentas para Documentação de Programa

As técnicas de diagramação estruturada são muito importantes como ferramentas de documentação.

Elas são usadas para definir as especificações de programa e também para representar o seu projeto. Elas fornecem as diretrizes para transformar o projeto em codificação de programa.

Uma visão geral e/ou detalhada do programa é importante dependendo do objetivo do leitor. Se estiver procurando um erro, então um documentação detalhada poderá guiá-lo para o local exato de origem do erro. Se a intenção, é determinar em qual dos vários programas uma certa função é executada, a melhor forma é através de uma documentação em alto nível.

Não é desejável o uso de técnicas diferentes e incompatíveis para os projetos de alto e baixo níveis. Isto acontecia quando o arquiteto do programa e o codificador eram pessoas diferentes. Hoje é mais apropriado que se decomponha o projeto em alto nível, em componentes de baixo nível

usando a mesma técnica de diagramação. Esta decomposição deve ser realizada por uma pessoa, preferencialmente na tela de um computador. Á medida que as linguagens de quarta geração se tornem mais populares, a era do codificador isolado irá desaparecendo. Nós necessitamos de uma técnica de diagramação que possibilite a uma pessoa esquematizar uma visão geral do programa e em seguida decompor esta visão em uma lógica detalhada. Os diagramas de ação fornecem esta habilidade. O processo de projeto de programa deve estar ligado ao projeto do banco de dados.

As técnicas de diagramação produzem tanto a documentação interna como a externa dos programas. A documentação interna está embutida na codificação fonte do programa ou é a gerada durante a compilação. Os comentários do programa e a listagem de referência cruzada são exemplos da documentação interna.

A documentação externa está separada da codificação fonte. Diagramas Warnier-orr e HIPO são exemplos da documentação externa de programa.

Funções dos Diagramas Estruturados

As técnicas de diagramação estruturada possuem muitas funções importantes:

- ✓ Auxiliam o pensamento claro.
- ✓ Melhoram a comunicação entre os membros da equipe de desenvolvimento.
- ✓ Padronizam as ligações entre os módulos.
- ✓ Permitem o cumprimento de uma boa estruturação.
- ✓ Auxiliam a depuração.
- ✓ Auxiliam a manutenção dos sistemas.
- ✓ Auxiliam o desenvolvimento.
- ✓ Disciplinam as especificações(quando ligadas às ferramentas de especificação computadorizada).
- ✓ Possibilitam aos usuários finais a revisar o projeto.
- ✓ Encorajam os usuários finais a esquematizar suas necessidades claramente.
- ✓ Possibilitam a ligação de rotinas de verificação automática e geração de programas.

3.1 Diagramas de Fluxo de Dados (DFD)

Um diagrama de fluxo de dados mostra os processos e o fluxo de dados entre estes processos. Em um nível mais alto costuma-se mostrar eventos do negócio e a transações resultantes destes eventos, sejam transações manuais ou computadorizadas. Em um nível mais baixo costuma-se mostrar programas ou módulos do programa e o fluxo de dados entre estas rotinas.

É utilizado como o primeiro passo em um formulário de projeto estruturado. Mostra o fluxo de dados global através de um sistema ou programa. É principalmente uma ferramenta de análise de sistemas utilizada para desenhar os componentes de procedimentos básicos e os dados que possam entre eles.

O diagrama de fluxo de dados mostra como os dados fluem através de um sistema lógico, mas ele não fornece informações de acompanhamento ou controle.

Um diagrama de fluxo de dados(DFD) é uma representação em forma de rede de um sistema, mostrando os processos e as ligações de dados entre eles. O DFD é constituído de quatro componentes básicos: o fluxo de dados, o processo, o arquivo de dados e o terminal.

3.2 Decomposição Funcional

A maioria dos projetos estruturados emprega uma forma de decomposição funcional. Uma função de alto nível é decomposta para o mais baixo nível de funções e estas por sua vez, são decompostos novamente. Uma árvore é um exemplo de decomposição.

Uma decomposição hierárquica pode ser utilizada em estruturas organizacionais, de sistemas, de programas, de arquivos e de relatórios. O termo “decomposição funcional” é mais utilizado em funções do que em dados. Entretanto, diagramas similares são as vezes, desenhados tanto pela decomposição de dados quanto de funções.

Nós temos três categorias diferentes de decomposição funcional, três espécies distintas como diria um botânico à respeito das árvores.

ESPÉCIE I

O tipo mais comum de decomposição funcional é uma estrutura em árvore que está relacionada com as funções e não com os dados utilizados por estas funções. É apropriada para mostrar a estrutura de um corporação. Seu mais alto nível de decomposição funcional é aplicada em uma empresa. Um diagrama estruturado em árvore é utilizado para mostrar uma organização.

ESPÉCIE II

A segunda espécie nos mostra os tipos de dados que são inseridos e extraídos de cada função. Este tipo pode ser muito mais detalhado quando manejado pelo computador, pois a máquina pode conferir os dados consumidos e produzidos por cada nó funcional.

ESPÉCIE III

A terceira espécie é ainda mais detalhada. Permite somente certos tipos de decomposição os quais têm que obedecer regras definidas por axiomas matemáticos. A estrutura resultante pode ser verificada completamente para garantir o que está compatível e correto internamente.

Defendemos o detalhamento não só porque queremos evitar erros nas especificações de programas mas também porque as técnicas detalhadas têm demonstrado que na pratica economizam tempo e dinheiro. As técnicas detalhadas serão conduzidas à um nível mais alto de automação.

Uma corporação complexa deve ter 30 áreas funcionais e de 150 a 300 processos. O gráfico deve ser desenhado verticalmente ao invés de horizontalmente.

3.3 Quadros de Estrutura

Os gráficos de estrutura são uma forma de decomposição funcional. Juntamente com os diagramas de fluxo de dados, eles constituem uma metodologia de projeto estruturado comumente utilizada na fase de codificação.

O bloco básico de construção de um programa é o módulo. Os programas estruturados são organizados como uma hierarquia de módulos. O gráfico estruturado é uma árvore ou um diagrama hierárquico, que define a arquitetura completa do programa mostrando seus módulos e relacionamentos.

O programa é representado como um conjunto de módulos hierarquicamente ordenados. Os módulos que executam tarefas de alto nível são colocados na parte superior da hierarquia, enquanto que os módulos executam funções em baixo nível, tarefas detalhadas aparecem nos níveis mais baixos. Olhando de cima para baixo na hierarquia, os módulos em cada nível sucessivo contêm tarefas que melhor definem a de precedente.

3.4 Diagramas

3.4.1 Diagramas de HIPO

HIPO vem do inglês da expressão Hierarchical Input Process Output (Entradas Processamento e Saídas Hierárquicas). Um diagrama que utiliza um conjunto de diagramas para mostrar entradas, saídas e funções de um sistema ou programa. Ele pode dar uma visão geral ou detalhada de um sistema (ou programa).

Como um gráfico de estrutura, HIPO mostra O QUE o sistema faz e , não COMO. Existem três tipos básicos de diagramas HIPO:

- ✓ Índice visual
- ✓ Diagramas gerais
- ✓ Diagramas detalhadas

O índice visual é muito semelhante a um gráfico de estrutura. Sua finalidade é mostrar os componentes funcionais gerais e referenciar diagramas HIPO gerais e detalhados. Note-se, entretanto, que ele não mostra o fluxo de dados entre componentes funcionais ou quaisquer informações de controle.

Os diagramas HIPO gerais e detalhados dão informações adicionais sobre cada componente funcional mostrado no índice visual. A distinção entre eles está na quantidade de detalhes que eles mostram. Diagramas gerais descrevem as entradas , processamento e saídas dos componentes funcionais maiores. De acordo com a IBM sua finalidade é fornecer “conhecimento geral de uma função particular” . Por outro lado, diagramas detalhados fornecem todas as informações necessárias para entender a função especificada no diagrama de nível mais alto. Diagramas gerais são usados tipicamente para descrever de nível mais alto no índice visual, enquanto diagramas detalhados são usados para descrever componente de nível mais baixo.

3.4.2 Diagramas Warnier-orr

Os diagramas Warnier-orr são uma alternativa para os diagramas HIPO ou Quadros de Estrutura . Eles receberam a denominação de seus proponentes : Jean-dominique Warnier e Ken Orr. Do mesmo modo o HIPO e os quadros de Estrutura, eles auxiliam no projeto de programas estruturados. São fáceis de aprender e usar porque são compostos de apenas quatro técnicas de diagramação.

Um diagrama Warnier-orr representa graficamente a estrutura hierárquica de um programa., de um sistema ou, de dados. E desenhado horizontalmente através da página, ao invés de seguir para baixo com blocos.

Cada chave no diagrama representa um quebra funcional do item mencionado na sua parte superior. É similar ao quadro de estrutura ou à tabela de conteúdo do diagrama HIPO. Além de mostrar a estrutura hierárquica, o diagrama Warnier-orr também ilustra o fluxo de controle através da estrutura. É similar à visão geral ou detalhada do diagrama HIPO.

Os quadros Warnier-orr podem representar as estruturas de dados, relatórios ou estruturas de programas.

O Warnier-orr é mais poderoso que o quadro de estrutura porque ilustra as construções de controle do programa: seqüência, seleção e repetição.

3.4.3 Diagramas Michael Jackson

Os diagramas Warnier-orr têm a vantagem de representar tanto estruturas de dados como de programas. As técnicas Michael Jackson possuem a mesma vantagem. Além disso as duas técnicas enfatizam que as estruturas de programas devem de derivar das estruturas de dados. Os dados de input e de output de um programa são usados para criar a estrutura do programa.

A técnica Jackson visualiza a estrutura do programa e dos dados hierárquica. Ela usa um diagrama de estrutura para representar tanto os dados como o programa.

3.4.4 Diagramas de Ação

O Diagramas de Ação fornecem uma forma natural de desenhar a visão geral como os quadros HIPO e , também de desenhar a lógica detalhada de forma a facilitar o aprendizado do usuário final e também para introduzi-los nas linguagens de quarta geração.

Esta técnica de diagramação pode ser transformada de uma visão em alto nível, diretamente para uma linguagem de quarta geração. Quando se usa a tela do computador, os projetistas podem editar e ajustar o diagrama, completando-o com detalhes até chegar à codificação do programa.

Atualmente, uma técnica de diagramação deve ser manipulada tanto manualmente quanto por computador. Os usuários e analistas desejam racionar com a ajuda dos desenhos feitos à mão ou em quadro. Também desejam construir diagramas complexos na tela do computador, além de usar este recurso para validar, editar e manter os próprios diagramas. Esta ferramenta funciona como um editor de textos para diagramação, tornando fácil o trabalho de modificação do diagrama.

Os diagramas de ação foram projetados para resolver os problemas das outras técnicas de diagramação. Eles foram projetados com as seguintes propriedades :

- ✓ São rápidos e fáceis de desenhar
- ✓ São bons para o esboço manual e para edição computadorizada.
- ✓ A técnica se estende desde a visão geral até os detalhes a nível de codificação.
- ✓ São fáceis de ensinar aos usuários; eles encorajam os usuários ao projeto da lógica do processo. São também projetados como uma ferramenta para o centro de informações.

- ✓ Podem ser impressos em papel de largura normal, possibilitando a impressão em computadores pessoais.
- ✓ Possibilitam a construção de programas estruturados e são mais gráficos que pseudocódigos.
- ✓ Os diagramas de ação no banco de dados são projetados para diagramar as operações no banco de dados inclusive as operações relacionais.
- ✓ Os diagramas de ação no banco de dados são projetados para ligação com o modelo de dados.
- ✓ Trabalham bem com as linguagens de quarta geração.
- ✓ São projetados para verificação cruzada por computador.

Alguns aspectos da lista acima não existem nas primeiras técnicas estruturadas:

- ✓ Edição computadorizada de diagramas.
- ✓ Verificação cruzada automática.
- ✓ Modelos de dados.
- ✓ Composição das operações do banco de dados relacional.
- ✓ Uso de computador pessoais para projeto.
- ✓ Linguagem de quarta geração.
- ✓ Forte envolvimento do usuário.
- ✓ Gerenciamento do centro de informações.

3.4.5 Diagramas de Ação no Banco de Dados

Estes diagramas são chamados de diagramas de ação no banco de dados (DABS). Eles usam as mesmas construções dos diagramas de ação, mas adicionando rombóides para o desenho de operações de banco de dados, inclusive com operações relacionais compostas.

Os diagramas de ação no banco de dados foram originalmente projetados com a ajuda do psicólogo Dr. K. Winter para tornar o desenho de procedimentos orientados para transações ; tanto quanto possível orientado para o usuário. A técnica pode ser ensinada para a maioria dos não profissionais de PD em um dia. Os diagramas são fáceis para ler e verificar, e são quase auto explicativos. Como os simples diagramas de ação, eles podem ser expandidos de uma visão geral de alto nível até a codificação da linguagem de quarta geração executável.

Os diagramas de ação no banco de dados são similares aos diagramas de ação e manuseiam os colchetes e a lógica condicional do mesmo modo; em adição eles mostram as ações no banco de dados como rombóides:

As linguagens tradicionais usam ação no banco de dados simples; algumas linguagens de quarta geração usam ações compostas.

Uma ação simples no banco de dados é uma operação aplicada a uma ocorrência de um registro em um modelo de dados (geralmente um registro totalmente normalizado). Existem quatro tipos de ações simples: CRIAR, LER, ATUALIZAR E DELETAR. A sigla CLAD pode ajudar o leitor a lembrar destes quatro tipos. As linguagens de banco de dados têm operadores que permitem estes tipos de ações no banco de dados.

O rombóide pode apresentar o número de ação no banco de dados. Este pode incluir o número do registro usado.

Uma descrição da ação no banco de dados é escrita dentro do rombóide. Geralmente deve ser uma sentença de comando começando com um verbo CRIAR, LER, ATUALIZAR E DELETAR. A sentença deve incluir o nome do dado em questão.

Uma ação composta no banco de dados leva uma ação simples contra o banco de dados, mas a ação pode usar múltiplos registros do mesmo tipo. Ela pode pesquisar ou classificar um arquivo lógico. Pode ser uma operação relacional que usa relação inteira. Pode ser uma operação relacional querendo uma ligação de duas ou mais relações. As linguagens de quarta geração têm instruções para uma grande variedade de ações compostas no banco de dados. A maioria das ações no banco de dados, em processamento de dados tradicional são ações simples. À medida que os bancos de dados relacionais e as linguagens não-procedurais se difundirem, as ações compostas se tornarão mais comuns.

3.5 Fluxogramas

Os fluxogramas foram um dos primeiros métodos de diagramação. Foram usados pela maioria dos analistas e programadores antes da era das técnicas estruturadas e ainda, são encontrados em uso na maioria dos CPD's . Os fluxogramas podem ser substituídos por diagramas mais claros, menores e mais estruturados, hoje o seu uso é recomendável. Existem dois tipos de fluxogramas : fluxogramas de sistema e de programas.

O fluxograma do sistema serve como documentação de operação, mostra ao operador a forma de executar o sistema.

Um fluxograma de programa é basicamente uma ferramenta de codificação. Mostra na forma gráfica a seqüência na qual os comandos ou blocos de processo serão executados e a lógica de controle que governa esta execução. Tradicionalmente, os fluxogramas de programa têm servido para dois propósitos. Primeiro, ele têm sido usados como uma ferramenta ou projeto de programa para planejar detalhadamente uma lógica complicada. Segundo, eles têm usados como documentação de programa.

Devido os fluxogramas fornecem um representação seqüencial, ao invés de hierárquica do programa, eles não podem mostrar claramente sua estrutura e o inter-relacionamento entre os componentes procedurais. Os blocos com setas desenhadas , tendem a encorajar o uso de sentenças GO TO, indesejadas quando se trata de um método estruturado.

3.6 Linguagem Estruturada e Pseudocódigo

A linguagem estruturada é uma notação narrativa usada para definir a lógica procedural.

A linguagem estruturada, assim como qualquer outro meio de representar as estruturas de programa, deve ser projetada para mostrar quatro construções básicas:

- ✓ Seqüência
- ✓ Condição
- ✓ Caso
- ✓ Repetição

A linguagem estruturada tem sido uma ferramenta útil para descrição da lógica do programa. Acreditamos que ferramentas mais diagramáticas são melhores.

3.7 Quadros

3.7.1 Quadros Nassi-Shneiderman

I.Nassi e B.Shneiderman substituíram o fluxograma tradicional por um quadro que oferece uma visão estruturada, hierárquica da lógica do programa. Os quadros Nassi-Shneiderman são usados para detalhar o projeto do programa e também para sua documentação . W. chapin usa um tipo muito similar de diagrama que é conhecido como QUADROS CHAPIN.

Os quadros de Nassi-Shneiderman(N-S) representam as estruturas de programa que possuem um ponto de entrada e um ponto de saída, e são compostas de construções de controle de seqüência, seleção e repetição. É fácil converter um quadro N-S para codificação estruturada.

Um quadro N-S consiste de um bloco retangular representando a lógica do modelo do programa. Usa-se representar o conteúdo de um bloco através de um quadro de estrutura.

O quadro N-S deve ser desenhado em uma página. Não deve possuir mais de 15 a 20 subdivisões. Quando um quadro N-S se torna muito grande, as sub funções são separadas e desenhadas em outro quadro.

Várias estruturas são agrupadas dentro do quadro N-S, para mostrar a lógica do processo. A instrução de desvio não é permitida.

3.7.2 Quadro de Relacionamento de Entidades

Os analistas de dados necessitam de muita ajuda dos usuários finais e executivos para entender os dados de uma organização , e para projetar os dados que serão mais úteis na gerência da organização. Eles necessitam de formas claras de visualização. Os Quadros de Relacionamento de Entidades são essências para planejamento dos recursos de informação da empresa.

Uma entidade é algo(Real ou abstrato) a respeito da qual nós armazenamos dados.

Exemplos de tipos de entidades: CLIENTE, PEÇA, EMPREGADO, FATURA, MÁQUINA, VENDEDOR etc.

O nome de cada tipo de entidade deve ser um substantivo, em alguns casos com um qualificador.

Um tipo de entidade pode ser imaginado como possuidor das propriedades de um substantivo.

Uma entidade possui vários atributos, os quais podemos registrar, tais com : cor, valor monetário, utilização percentual, ou nome.

Um TIPO DE ENTIDADE é a denominação de uma classe de entidades que possui o mesmo conjunto de tipos de atributos: por exemplo, EMPREGADO é um tipo de entidade.

OCORRÊNCIA DE ENTIDADE é uma incidência de um tipo de entidade: por exemplo, B.J.WATKINGS é uma incidência do tipo de entidade EMPREGADO.

Usa-se abreviar o TIPO DE ENTIDADE como ENTIDADE. Quando o leitor encontrar CLIENTE como uma entidade, ele deve estar ciente de que se refere ao tipo de entidade.

Usando esta abreviação, descrevemos os dados em termos de ENTIDADE E ATRIBUTOS. A diferença entre eles é que armazenamos informações sobre uma entidade em vários tipos de itens de dados. Não armazenamos informações sobre um atributo em vários tipos de itens dados. Se um tipo de item de dado que estivermos chamado de ATRIBUTO requerer informações além de VALOR, então ele é realmente uma ENTIDADE.

Uma caixa retangular é desenhada para representar uma entidade. Para a maioria das entidades armazenamos registros, por exemplo : registro de CLIENTES, PEÇAS, EMPREGADOS etc. A caixa é algumas vezes também usada para mostrar um tipo de registro de entidade. Entretanto, o objetivo de nosso modelo de dados é representar a realidade dos dados sem no entanto pensar em como será representado no computador. Podemos decidir representá-lo sem uma estrutura convencional de registro.

3.8 Árvores e Tabelas de Decisão

As árvores e tabelas de decisão não se originaram das técnicas estruturadas nem da programação. Elas possuem várias outras aplicações. São usadas em biologia, ciência do computador, teoria da informação e dos circuitos. Existem quatro principais campos de aplicação:

- ✓ Taxinomia, diagnósticos e reconhecimento de padrões
- ✓ Projeto de circuito (lógica) e teste de confiança
- ✓ Análise dos algoritmos
- ✓ Programação e banco de dados apoiados em tabelas de decisão.

Uma árvore de decisão é um modelo de função discreta, na qual uma variável é determinada, e a partir deste valor alguma ação é executada. A ação pode ser : executar outra função ou simplesmente fornecer um output. Portanto, cada ação tomada depende do valor corrente da variável em teste e de todas as ações prévias terem sido executadas. Em uma árvore de decisão formalmente definida, uma variável é testada somente uma vez qualquer ramo da árvore. Esta restrição visa testes redundantes.

As árvores de decisão são normalmente constituídas a partir da descrição de um problema. Elas fornecem uma visão da tomada de decisões necessária. Especificam que variáveis são testadas, que ações devem ser tomadas, e a ordem em que a tomada de decisões ocorre. Dependendo do valor de uma variável seque-se um caminho na árvore de decisão que se inicia com a raiz e segue até a folha da árvore.

3.9 Gráficos

3.9.1 Gráficos de Bolhas para Dados

Gráfico de bolhas fornecem um modo de desenhar e entender as associações entre itens de dados. Este entendimento é necessário a fim de criar registros que sejam claramente estruturados. Os gráficos de bolhas formam a entrada de um processo de montagem de dados que cria estruturas estáveis de dados. Este processo é automatizado.

Os gráficos de bolhas são úteis para ensinar usuários finais e analistas sobre associações de dados. Quando eles começam a criar estruturas de bancos de dados lógicas eles devem empregar os gráficos de bolhas. À medida que eles se tornam mais experientes eles podem deixar de desenhá-los e representar as mesmas informações como entrada de um meio automatizado que sintetiza a estrutura de dados.

A parte mais elementar dos dados é chamada de item de dado. Algumas vezes é chamado também de campo ou elemento de dado.

É o átomo, que não pode ser subdividido em tipos de dados menores e ainda manter algum significado para os seus usuários. Você não pode dividir o item de dado denominado SALÁRIO, por exemplo, em itens de dados menores que sejam por si significativos para os usuários finais.

3.9.2 Gráficos HOS

Na prática convencional, as funções são decompostas de algum modo que pareça conveniente para o projetista. Nesta técnica descrita, somente formas de descrição rigorosamente definidas são permitidas. Estas formas de decomposição são precisamente definidas, com regras matemáticas, a cada passo, portanto a decomposição é comprovadamente correta. A decomposição continua até que sejam conseguidos blocos a partir dos quais a codificação de programas executável possa ser gerada.

A técnica aqui descrita é denominada “higher-order-software” (HOS). Ela foi criada por Hamilton e Zeldin, e é implementada com um software de gráficos chamados USE-II, fornecido por uma companhia chamada Higher Order Software. O software gera automaticamente a codificação de programa executável. Assim temos a decomposição final. Ao contrário de muitos geradores de codificação este processo pode gerar a codificação para sistemas muito complexos.

HOS, como outros tipos de decomposição funcional, emprega estruturas em árvore. Em sua forma mais primitiva ele usa árvore binárias nas quais cada nó pode ser decomposto de três maneiras. A partir destas premissas precisamente definidas são construídas estruturas de controle de nível mais alto, que não são necessariamente binárias.

Como Yourdon ou Michael Jackson, HOS elabora estruturas em árvore verticais com raiz em cima. Ao contrário de Yourdon, Michael Jackson ou outros, cada decomposição é um tipo específico denominado estrutura de controle. O tipo de estrutura de controle usado é escrito no gráfico. A decomposição tem que obedecer regras para chegar a essa estrutura de controle, que forçam a decomposição matematicamente correta.

Cada nó subordinado é desenhado abaixo de seu superior nos diagramas HOS. Convencionou-se dizer que um ramo está “entrando” em um nó se ele vem de cima. Diz-se que ele está “saindo” de um nó se ele vai para um nó subordinado.

Cada nó de uma árvore binária HOS representa uma função.

Uma função tem um ou mais objetos como entrada e um ou mais objetos como saída.

Um objeto pode ser um item da dado, uma lista, uma tabela, um relatório um arquivo, um banco de dados, ou ele pode ser uma entidade física tal como um circuito, um míssil, um produto submetido a um programa de manufatura, um trem, trilhos, desvios ferroviários, etc.

Mantendo-se a notação matemática o objeto ou objetos de entrada são escritos do lado direto da função; o objeto ou objetos de saída são escritos à esquerda da função.

3.10 Mapa de Acesso aos Dados

Uma vez que exista um modelo de dados estável e completamente normalizado, a tarefa do analista de sistema torna-se mais fácil. O projetista deve definir como navegar através do banco de dados. Ele necessita de uma técnica clara de diagramação para isto.

Um princípio básico do projeto estruturado é : DIVIDIR E CONQUISTAR. Projetos complexos devem ser reduzidos para possibilitar a visão simplificada dos modelos. A existência de um modelo de dados normalizado possibilita a redução de aplicações complexas até projetos mais simples, como: ENTRAR E VALIADAR, ATUALIZAR, EXECUTAR CÁLCULOS, MANIPULAR ACESSOS, GERAR RELATÓRIOS, etc. Algumas vezes estes projetos usam dados em formas complexas com muitas referências cruzadas enter eles.

Muitos destes projetos ser executados por uma só pessoa, especialmente linguagens de Quarta geração são usadas. A principal comunicação entre os diversos analistas é através do modelo de dados. A maioria dos problemas de comunicação humana nos grandes projetos tende a desaparecer. Os projetos de uma só pessoa são muito atraentes. A gerência pode selecionar a pessoa para o trabalho e motivá-lo para a rapidez e qualidade. Ele é o senhor de seu próprio sucesso. Ele não será apenas uma engrenagem na máquina. Ele não terá de se atrasar porque teve de rescrever o seu programa devido aos erros terceiros.

Os dados são projetados separadamente usando técnicas corretas, preferivelmente automatizadas. Os usuários ou analistas, que elaboram os projetos empregam o modelo de dados para considerar os procedimentos relacionados. Freqüentemente, o modelo de dados é projetado por um administrador de dados. Em outros casos isto é realizado pelo próprio analista.

Os dados possuem propriedades próprias, independentes dos procedimentos, as que levam à uma estruturação estável. O mapa de acesso aos dados é desenhado dentro de um modelo de dados e tem a finalidade de ligá-lo aos programas. Ele forma ima ponte, fácil de usar, entre o modelo de dados e o projeto dos procedimentos. Qualquer projeto de um programa de banco de dados deve se iniciar pela esquematizado do mapa de acesso aos dados.

O primeiro passo na criação de procedimentos que utilizam um modelo de dados é identificar a seqüência na qual os registros devem ser acessados. Esta seqüência pode ser desenhada no modelo de dados. Nos referidos a ela como mapa de acesso aos dados (MAD). A partir do MAD podem ser criados informações para o projeto físico do banco de dados e para o projeto dos procedimentos da aplicação. O mapa de acesso aos dados leva ao projeto de procedimentos com diagramas de ação no banco de dados (DADS). Eles por sua vez, levam à codificação estruturada de programas (que são diretamente aplicadas às linguagens de quarta-geração).

O modelo completo de dados é freqüentemente muito complexo para o desenho das seqüências de acesso. Somente certas entidades do modelo de dados são necessárias. O projetista dos procedimentos especifica quais entidades são de interesse.

Do modelo de dados pode-se imprimir uma VIZINHANÇA. A VIZINHANÇA de uma entidade é um conjunto de entidades que podem ser obtidas atravessando ligações do modelo da dados. Geralmente isto significa: examinando as próximas portas abertas no modelo de dados.

O projetista examina a vizinhança. Ele enxerga os registros que seus procedimentos devem utilizar, e mais alguns. Ele elimina os não desejados. Existem algumas ligações MANDATORIAS nos registros especificados, que devem ser mantidos. Por exemplo, quando um registro de requisição de material entregue é criado, deve-se atualizar o registro de inventário do material.

O projetista estabelece o grupo de registros que seu procedimentos irá utilizar. Também conhece o subconjunto do modelo de dados necessário para isto. Geralmente este subconjunto pode ser desenhado em uma página.

Agora o projetista toma decisões sobre a seqüência na qual seu procedimento irá utilizar os registros. Ele desenha esta seqüência, talvez com caneta de tinta vermelha no subconjunto do modelo de dados. Ele usa setas de cabeça simples para indicar que uma ocorrência do registro é acessada e usa setas de cabeça dupla para indicar o acesso a mais de uma ocorrência.

4. ANÁLISE DE SISTEMA EXISTENTE

O objetivo da análise do sistema é determinar em que grau ele atende as necessidades da empresa. São verificadas a qualidade e a rapidez das informações fornecidas, bem como a eficiência em que estas informações são geradas.

Nesta subfase, o analista está principalmente interessado em conhecer o sistema atual o que ele faz e como funciona.

Para podermos avaliar um sistema existentes devemos reunir as seguintes informações :

- ✓ Os objetivos do sistema ;
- ✓ A organização da empresa;
- ✓ A documentação utilizada na empresa : relatórios, arquivos;
- ✓ Os procedimentos existentes.

4.1 Determinação dos Objetivos do Sistema

O funcionamento do sistema de P.D. deve ir de encontro as necessidades do sistema maior que é a empresa. Portanto é preciso conhecer :

- ✓ Os objetivos da empresa;
- ✓ Como é medida a eficiência da empresa;
- ✓ As decisões tomadas pelos executivos da empresa.

4.2 Estudo da Organização

O entendimento da estrutura organizacional da empresa ajuda a ver claramente os seus diversos componentes;

A determinar as necessidades de informação destes componentes;

E a decidir que pessoas da organização deverão ser consultados com relação a estas necessidades. Portanto para conhecermos a estrutura da empresa devemos :

- ✓ Coletar e estudar organogramas, descrições de funções
- ✓ Conhecer a organização 'informal' da empresa.
- ✓ Estudar o fluxo de dados através da organização.

4.3 Documentação Utilizada

A documentação de um determinado setor é composta pelos relatórios que ele emite e pelos arquivos que ele mantém.

Relatório é qualquer documento gerado por um determinado setor, como por exemplo: documentos de gerência, sumários, cheques de pagamentos, notas fiscais, etc. ...

E de cada um devemos observar :

- ✓ Sua finalidade
- ✓ Que informações contém
- ✓ Origem e destino de cada informação
- ✓ Recursos requeridos para a sua preparação
- ✓ Número de cópias
- ✓ Volume dos documentos emitidos

- ✓ Erros : freqüência, natureza e implicações

Arquivo é qualquer dispositivo onde sejam armazenados temporariamente documentos que tenham entre si uma relação , como por exemplo : Arquivo de contas a pagar , arquivos de contas pagas, etc. ...

E de cada um devemos observar :

- ✓ A sua finalidade
- ✓ Que informações ele contém
- ✓ Origem e destino de cada informação
- ✓ Volume de atualização das informações
- ✓ Freqüência destas atualizações
- ✓ Recursos disponíveis para essa atualização
- ✓ Tempo em que os documentos devem permanecer no arquivo

OBS. :

- ➢ Revisar toda a documentação colhida para atualizar o sistema atual. Se necessário, completar o processo de obtenção de informações coletando toda a documentação pertinente que descreva as operações detalhadas do sistema atual.

- ➢ Se necessário, criar um inventário da documentação detalhada obtida do sistema atual.

- ➢ Analisar as informações do sistema atual e consolidar os problemas, oportunidades e necessidades dos usuários que foram documentados durante a pesquisa dos documentos.

- ➢ Discutir com os usuários os principais pontos fortes e fracos do sistema atual. Se apropriado, identificar as características essenciais do sistema atual que os usuários gostariam de manter no novo sistema.

- ➢ Conferir a documentação do sistema atual para garantir que reflita adequadamente o ambiente existente e validar os problemas, oportunidades e necessidades consolidadas dos usuários.

4.4 Procedimentos Existentes

As operações efetuadas sobre as informações podem ser manuais ou mecanizadas. Além das descrições das operações devemos observar :

- ✓ Fluxo lógico destas operações
- ✓ Volume das operações e tempo necessário para executá-los
- ✓ Procedimentos de exceção
- ✓ Precisão de chegada dos documentos de entrada e os prazos para a preparação dos relatórios de saída.

5. A CULTURA CASE

"Computer-Aided Software Engineering"

O desenvolvimento de sistemas tradicional é lento e rígido e de um modo geral os procedimentos normais não funcionam de maneira adequada . As soluções existem mas requerem novos softwares, novos procedimentos e novas estruturas gerenciais, muitas vezes estando em choque com a própria Cultura da organização.

As ferramentas CASE, por serem um conjunto integrado de ferramentas que atuam em todas as fases de desenvolvimento de software, têm um impacto profundo, exigindo novas metodologias e impactos Culturais profundos a nível Organizacional.

CASE é automação de software. A tecnologia CASE é talvez a mais profunda transformação corrida na comunidade de software. CASE é automação da automação. Ele oferece uma resposta prática aos problemas de produtividade de software dos últimos vinte cinco anos, tempo da utilização dos computadores.

A tecnologia CASE é uma combinação de ferramentas de software com metodologia. Deste modo, CASE é diferente das primeiras tecnologias de software, porque enfoca o problema da produtividade de software e não só os problemas de implementação.

O termo foi criado no começo dos anos oitenta, quando a idéia de que ferramentas gráficas, como a dos diagramas de fluxo de dados, os diagramas de entidades e relacionamentos e gráficos estruturais, poderiam ser úteis na análise e projeto de sistemas.

Uma definição talvez mais simples e completa é a que nos é dada por McCLURE, 1989': "CASE é a automação do desenvolvimento de software". Nesta concepção, CASE é uma nova abordagem para o ciclo de vida de software, que é baseada na automação. A idéia básica é que CASE proveria um conjunto integrado de ferramentas de economia de trabalho, ligando e automatizando todas as fases do ciclo de vida de software.

Isso evidencia que a tecnologia de ferramentas automáticas continuará avançando até o fim da década. Os do analista em meados da década de 1990, certamente terão características sofisticadas de verificação de erro, tão bem como habilidade em gerar código e mesmo (usando técnicas de inteligência artificial) possibilidades de código reutilizável a partir de Bibliotecas de Software.

Muitas ferramentas CASE incluem poderosos editores gráficos que implementam uma interface com o usuário, transformando o desenvolvimento de software num processo visual.

CASE promove novas técnicas como engenharia da informação, protipagem rápida e sessões JAD com usuários finais, assim como suporte a técnicas estruturadas tradicionais estruturadas tradicionais como o projeto estruturado.

CASE altera e prevê o ciclo de vida do sistema . Com CASE, o projetista do sistema pode focalizar mais o trabalho da análise e projeto e menos a codificação e teste. Programas codificados manualmente e alterados por geradores de código que automaticamente produzem código a partir das especificações do projeto.

Especificações do projeto tomam a forma de diagramas estruturados, como DFD e DER, e modelos de protótipos. Como as especificações são armazenadas no repositório CASE, elas são automaticamente verificadas para a correção e referência cruzada com outro sistema de informação. Toda a informação do sistema armazenada em repositório CASE onde ele pode ser automaticamente gerenciados e distribuído.

Finalmente , CASE reduz o sério, e custoso, problema de manutenção de sistemas, pois as ferramentas CASE geram a documentação automaticamente, analisa e reestrutura os programas existentes.

5.1 Categorização

Há poucos anos atrás, dificilmente algum analista de sistemas teria ouvido o termo CASE. Agora é o centro das atenções, e o que há de mais recente e atual para reportar os problemas próprios de desenvolvimento de sistemas. O mercado de ferramentas CASE tem um dos mais altos graus de crescimento das empresas que atuam no setor de desenvolvimento de software. Como o mercado nasce, a definição de CASE continua a mudar e não há um padrão definido para sua categorização, no entanto os tempos abaixo são os que melhor o identificam.

Front End ou Upper CASE . São aquelas que apoiam as etapas iniciais de criação dos sistemas, são elas as fases de planejamento, análise e projeto do programa ou aplicação(ou seja , a parte lógica).

Back End ou Lower CASE. São aquelas que dão apoio à parte física, isto é, à codificação testes e manutenção da aplicação, normalmente encontram-se produtos de ambos os lados, mas o difícil é se obter uma compatibilidade entre ambas, isto é, conseguir com auxílio de uma única ferramenta (ou mais de uma , de forma integrada).

I-CASE ou Integrated CASE. Classifica os produtos que cobrem o ciclo de vida do software, desde os requisitos do sistema até o controle final da qualidade.

Outra classificação nos é dada segundo a fase do Ciclo de Vida que apoia:

Planning Workstation - Uma estação de trabalho que automatiza planejamento de sistemas da empresa. Consiste de editores de matrizes e dicionário de dados que apoiam técnicas de planejamento estratégico de informações.

Analysis Workstation - Apoia o trabalho do analista de sistemas, auxiliando-o no trabalho de especificação, rigorosa e detalhadamente, sistemas de aplicação segundo as técnicas estruturadas de análise de sistemas. Este é o tipo de estação de trabalho com maior concorrência no mercado, é que, por isso mesmo, tem tido o maior desenvolvimento e inovação. O próprio termo CASE está, em grande parte, ligado a esse tipo de software.

Design Workstation - Voltada para o projetista de sistemas, possui editores gráficos capazes de auxiliar no projeto da arquitetura de código dos sistemas, de acordo com métodos que variam entre Yourdon-Constantine, Warnier, Jackson E James Martin.

Ferramentas de Programação - São ferramentas de programação CASE para suportar a implementação de programas. Muitas destas ferramentas são familiares e amplamente usadas pelos programadores. A diferença é que esses pacotes de ferramentas são feitos para se integrarem com outros.

5.2 Critérios

5.2.1 Critério de Seleção de Ferramentas CASE

- ✓ Quais as direções para o futuro e funcionalidade da ferramenta.

- ✓ fabricante da ferramenta tem uma filosofia de arquitetura aberta.

- ✓ A ferramenta produz software utilitário e procedimentos de leitura, bibliotecas de fontes e cria especificações de componentes CASE para sistemas existentes.
- ✓ A ferramenta tem uma interface efetiva com outras ferramentas de projeto CASE anterior à aquisição ou avaliação inferior.

- ✓ A ferramenta dispõe de metodologias gráficas, capazes de explodir os diagramas do projeto e especificações do dicionário, para um sistema de dimensões moderadas.

- ✓ É capaz de executar tarefas com capacidade de trabalhar com janelas.

- ✓ É provido de capacidade de Prototipação.

- ✓ É capaz de gerar automaticamente um escopo das especificações do projeto físico a partir das especificações do projeto.

- ✓ A ferramenta tem capacidade de gerar automaticamente relatórios das especificações do projeto.

- ✓ A ferramenta permite a distribuição de responsabilidades pelo projeto de desenvolvimento.

- ✓ Pode a ferramenta de interface do projeto e especificações de desenvolvimento para um DBMS funcional, ser usada para manter dados de uma companhia.

- ✓ Tem capacidade de processamento de texto.

- ✓ A ferramenta destaca o gerenciamento do projeto.

- ✓ É possível modificar o projeto CASE e ferramentas de desenvolvimento relativas ao ambiente interno da empresa ou metodologia existente.

- ✓ Pode a ferramenta gerar automaticamente projeto, operação e documentação do usuário final.

- ✓ A ferramenta tem facilidades para manter o projeto tão bem quanto sistemas.

- ✓ Pode a ferramenta gerar programas para uma ampla faixa de sistemas.

5.2.2 Critérios para Comparação

- ✓ Compatibilidade com as metodologias adotadas na empresa.
- ✓ Aplicabilidade no ciclo de vida.
- ✓ Recursos gráficos.

Outros aspectos são : Dicionário de Dados, Prototipação, Documentação e integração com outras ferramentas.

5.2.3 Critérios para Implementação de Ferramentas CASE

- ✓ Estabelecimento e aplicação de critérios para avaliação das Ferramentas.

- ✓ Inclusão de um Treinamento extensivo.

- ✓ Seleção de um grupo experiente para treinamento e acompanhamento durante a implementação.

- ✓ Formulação de um conjunto de padrões e procedimentos.

- ✓ Grupo de Consultores customizando o treinamento e padrões de documentação e procedimentos.

- ✓ Seleção de um grupo experiente (líderes de projeto), como primeiro grupo a receber treinamento.

- ✓ Casa haja treinamento interno, deve haver a inclusão de treines.

- ✓ projeto piloto deve ser relativamente simples, de 3 a 6 meses, para não haver desgaste.

- ✓ grupo diretamente ligado ao piloto deve ser treinado por consultores.

- ✓ Após o término do primeiro projeto, deverão ser reavaliados e alterados os padrões e procedimentos da empresa de acordo com a experiência alcançada.

- ✓ Deverá ser feita uma reavaliação do treinamento após o primeiro projeto.

6. PROJETO DO NOVO SISTEMA

Projetar um novo sistema exige uma grande dose de criatividade, neste ponto o analista começa a sentir o resultado realizado.

É muito difícil traçar um caminho rígido a ser seguido no novo projeto do novo sistema. Uma vez conhecido em detalhes o sistema existente, estamos em condições de especificar um novo sistema que atenda melhor as necessidades da empresa.

6.1 Definição do Novo Sistema

O sistema de PD deve ser definido em função do sistema maior, que é a empresa. Portanto o novo sistema deve “respeitar” o todo do qual é parte, integrando os seus objetivos e os seus elementos básicos com os da organização a que pertence, a fim de atingir as metas estabelecidas para a empresa.

A fim de definirmos um novo sistema devemos considerar os seguintes parâmetros :

- ✓ Objetivos ==> Exemplos de objetivos que podem levar ao desenvolvimento de novos sistemas :

- Reduzir os custos operacionais da empresa;
- Absorver volumes crescentes, permanentes ou temporários de trabalho;
- Dar a administração informações mais corretas, mais atualizadas e mais adequadas, de maneira a melhorar a qualidade de decisão e do controle gerencial;
- Dar maior eficiência operacional á empresa através do aumento da velocidade de execução das tarefas;

✓ Ambientes ==> O ambiente do sistema de PD abrange os setores da empresa e do público em geral com os quais se comunica.

✓ Recursos ==> Correspondem aos orçamentos existentes na empresa para deu desenvolvimento e operação.

Ex..: Contratação, treinamento, obtenção de equipamentos, etc..

O alcance do sistema (novo) dependerá dos recursos colocados a disposição para seu desenvolvimento e operação.

✓ Componentes ==> Os componentes do sistema de PD consistem nos diversos subsistemas que o constituem. Estes subsistemas aparecem estruturados em sucessivos níveis hierárquicos, até se chegar aos componentes básicos do sistema, que são: A informação, as decisões tomadas em função destas informações e as operações executadas em função destas decisões.

✓ Modularidades ==> Os detalhes que surgem durante o desenvolvimento de um sistema de PD são tão numerosos, que enfoque, no qual todas as funções do sistema fossem desenvolvidas simultaneamente, provocaria grande dificuldade de implantação e necessária uma equipe técnica não disponível na maioria das empresas. É necessário, portanto, distinguir subsistemas dentro do sistema principal, que tenham o menor intercâmbio possível de informações entre si, de modo que o projeto possa ser desenvolvido de forma modular. Os subsistemas deverão ser projetados de maneira a atender as necessidades de uma área restrita da organização, mas tendo sempre em mente as suas necessidades globais.

O desenvolvimento de um sistema por módulos oferece diversas vantagens:

✓ Reduz a complexidade inerente no desenvolvimento de um sistema grande e complexo.

✓ Minimiza o risco de desenvolver um sistema massivo e monolítico apenas para descobrir muito tardiamente que ele não atende a todos os requisitos dos usuários.

✓ Fornece aos usuários as funções que eles precisam com mais urgência antes de o sistema como todo ser implementado.

- ✓ Simplifica a distribuição das tarefas e otimização dos recursos disponíveis. Por outro lado, o conceito de módulos pode implicar alguma ampliação do processo de desenvolvimento, já que talvez seja necessário um esforço extra para a equipe de desenvolvimento garantir que os diferentes módulos permaneçam consistentes e sincronizados uns com outros.

7. MÉTODOS DE COLETA DE INFORMAÇÕES

As informações necessárias ao conhecimento do sistema geralmente não estão a disposição do analista sob a forma que ele necessita. É preciso obter estas informações junto aos diversos setores da empresa. E os mais importantes métodos de coleta de informações sobre o sistema atual são :

- ✓ Entrevista
- ✓ Observação
- ✓ Questionário
- ✓ Demonstrações feitas pelos fornecedores
- ✓ Visitas a outras instalações
- ✓ Coleta de dados
- ✓ Pesquisa externa

7.1 Entrevista

Por que fazemos entrevistas durante a análise de sistemas ? Os motivos são estes :

✓ Precisamos coletar informações sobre o comportamento de um sistema atual ou sobre os requisitos de um novo sistema de pessoas que têm essas informações armazenadas em algum lugar em suas cabeças.

✓ Precisamos verificar nossa própria compreensão, como analistas de sistemas, do comportamento de um sistema atual ou dos requisitos de um novo sistema. Essa compreensão deve ter sido adquirida através de entrevistas prévias em combinação com informações coletadas de modo independente.

✓ Precisamos coletar informações sobre o(s) sistema(s) atual(is) para executarmos os estudos de custo/benefício.

A entrevista é , geralmente , a maneira mais produtiva de obtenção de informações , é necessário que o entrevistado confie no entrevistador e que a entrevista comece nos escalões mais altos da empresa, seguido de gerentes, chefes de setor e operários. Ao se planejar uma entrevista , alguns pontos devem ser considerados(lembrados) :

- ✓ Defina claramente os objetivos
- ✓ Planeje a entrevista com antecedência
- ✓ Não prolongar a entrevista
- ✓ Aprender a separar opiniões de fatos
- ✓ Estar seguro de estar entrevistando pessoas, representando ambos os lados de cada operação
- ✓ Questionar o que o entrevistado sugere para melhorar o desempenho da empresa
- ✓ Fazer as entrevistas de uma maneira informal
- ✓ Marcar a entrevista com antecedência.
- ✓ Certifique-se de que você tem autorização para falar com os Usuários.

7.2 Observação

A análise de observação é uma técnica de obtenção de fatos muito efetiva. Ela pode ser usada para diversas finalidades, como processamento e confirmação dos resultados de uma entrevista, identificação de documentos que devem ser coletados para análise posterior, esclarecimento do que está sendo feito no ambiente atual, e tarefas similares. Durante algum tempo, o analista observa os usuários em seu ambiente enquanto eles executam suas atividades diárias. Apesar de o analista poder observar sem intervir diretamente no processo, na maioria das vezes, ele interagirá com a pessoa que está sendo observada. Freqüentemente fará perguntas para conhecer o funcionamento da operação. No limite do possível, o analista deve executar as atividades do usuário, para entender melhor como o usuário opera em seu próprio ambiente.

7.3 Questionários

O questionário de pesquisa é usado com freqüência para obter informações. Essa ferramenta é muito conveniente quando há um grande número de usuários, o que geralmente ocorre em grandes organizações, em que diversos grupos de usuários podem estar em diversos locais diferentes do país.

Os questionários de pesquisa têm algumas desvantagens. Uma desvantagem séria é que a comunicação com os usuários é seriamente restringida; não há uma real troca de informações face a face.

Podem ser usados diversos formatos para o questionário; múltipla escolha, lista de verificação, questões com espaços em branco são alguns dos exemplos.

7.4 Demonstrações feitas pelos Fornecedores

Os fornecedores de hardware e os fornecedores de software podem já haver desenvolvido sistemas para a aplicação em que você esteja interessado. Solicitando-lhes uma demonstração dos recursos desses sistemas pode não somente auxiliá-lo a decidir se o produto é uma boa solução, mas também revelar funções e dados armazenados que você pode não ter percebido.

7.5 Visitas a outras Instalações

Procure outras empresas que estejam no mesmo ramo de atividades ou que tenham sistemas semelhantes àquele em que você esteja trabalhando. Combine uma visita à instalação para obter informações diretas sobre as características e aptidões do sistema.

7.5.1 Coleta de Dados

Procure formulários, relatórios, manuais, procedimentos escritos, registros, imagens de tela de terminais e listagens de programas que já existam na organização usuária. Lembre-se, todavia, que esses recursos normalmente estão relacionados com a implantação atual do sistema, isso costuma incluir informações redundantes e/ou contraditórias e/ou obsoletas. Não obstante, isso muitas vezes é um bom ponto de partida para você familiarizar-se com o terreno antes de iniciar as entrevistas pessoais com o usuários.

7.5.2 Pesquisa Externa

Se você estiver construindo um sistema para uma nova aplicação, para a qual o usuário não dispões de qualquer experiência para descrever os requisitos, talvez seja necessário tentar obter informações em sociedades profissionais, ou em periódicos e livros técnicos e em relatórios de pesquisas.

8. AS CRISES DO SOFTWARE

A área de processamento de dados passa por sucessivas "crises". A administração não tem as informações gerenciais de que necessita ou na hora que necessita.

Um sentimento reinante nas empresas, principalmente por aqueles que tratam dos investimentos e controles financeiros, é a sensação de que muito está sendo gasto em computação, e ainda assim, o resultado da área de processamento nunca é o desejado.

Uma relação dos principais problemas de processamento de dados, vistos pela administração e pelos usuários.

- ✓ "Os usuários não conseguem obter aplicações quando precisam. Existe mesmo uma demora de anos".

- ✓ "É difícil, ou impossível, obter modificações que a administração precisa em tempo razoável" .

- ✓ "Algumas vezes os programas têm erros ou não funcionam".

- ✓ "Os sistemas entregues dificilmente atendem às reais necessidades dos usuários".

- ✓ "É difícil processamento de dados e comunicar os requerimentos precisos."

- ✓ "As especificações que os usuários tiveram que aprovar são difíceis de verificar."

- ✓ "Os sistemas custam muito para desenvolver e manter do que o previsto."

- ✓ "Devido ao longo tempo necessário à obtenção de resultados, a maioria dos importantes sistemas de suporte de decisão nunca são implementados. O suporte é necessário mais rapidamente do que o tempo necessário para criar os programas."

O problema principal não está com a máquina em si, mas nos métodos de trabalho. O ciclo de vida de desenvolvimento de sistemas tradicional é lento e rígido.

Seu uso existe nas empresas através de padrões e procedimentos, tais como abertura de uma ordem de serviço, estudos de tempos e custos de desenvolvimento, análise, programação, implementação e testes. De um modo geral, esses procedimentos não funcionam de modo adequado, principalmente em grandes organizações, onde é necessário que haja uma auditoria de sistemas forte para o controle dos processos; no entanto, esse controle é necessário e formas viáveis para se automatizar o ciclo de vida e manterem-se os controles a nível de trilhas de auditoria devem ser encontradas.

As soluções existem, mas requerem novos softwares, novos procedimentos e novas estruturas gerenciais, muitas vezes entrando em choque com a própria cultura da organização.

A hora é de um rompimento com os antigos métodos, ainda mais que a revolução gerada pelos microcomputadores e sua ligação aos Mainframes permitem a automação das antigas metodologias e incentiva a criação de novas técnicas para a automação do ciclo de desenvolvimento de software. No entanto, a maioria das empresas e dos profissionais de processamento de dados não acompanham a velocidade da proliferação das novas tecnologias.

Exemplos são os bancos de dados relacionais e as linguagens de quarta geração, em relação ao Assembler ou mesmo ao Cobol. As ferramentas CASE, por serem um conjunto integrado de ferramentas que atuam em todas as fases do ciclo da vida de desenvolvimento de software, têm um impacto profundo, exigindo novas metodologias.

Ao longa fila de espera "back-log" e a incapacidade dos departamentos de processamento em responder às necessidades rapidamente, não é o desejo do usuário; este por sua vez teria uma

propensão em obter novas máquinas, descentralizando o processamento e, por muitas vezes, transformando a organização em uma verdadeira torre de Babel tecnológica, com cada usuário utilizando as suas próprias ferramentas e computadores.

Dentre os principais problemas dos sistemas de informação temos :

- ✓ Usuários insatisfeitos.
- ✓ Alto custo de manutenção.
- ✓ Donos de sistemas.
- ✓ Back_log.
- ✓ Baixa confiabilidade
- ✓ Excessiva necessidade de retrabalho.

Como expectativas para os anos 90 :

- ✓ Novas tecnologias
- ✓ Sistemas cada vez mais complexos.
- ✓ Sistemas voltados para fora da empresa.
- ✓ Competição mais acirrada.
- ✓ Demanda por soluções integradas.
- ✓ Maior velocidade na mudanças.

9. DOCUMENTAÇÃO DE SOFTWARE

9.1 Informação Introdutória

A Informação Introdutória é a documentação normalmente oferecida a um cliente em perspectiva, quando ele começa a pensar na possibilidade de adquirir o software. A partir desta pesquisa inicial, ele pode resolver se vai ou não adquirir o software, ou pode decidir que necessita de informações adicionais. Estas informações poderão ser obtidas através de contatos com outros usuários do software, ou através da Documentação do Usuário, ou ainda pelos dois meios.

A informação introdutória cumpre dois papéis principais. Em primeiro lugar ela deve expor claramente o que o software irá fazer e qual o equipamento que será necessário. Isto evitará que aqueles para quem o software não é pertinente percam seu tempo. Em segundo lugar, a Informação Introdutória deve atrair os possíveis clientes e encorajá-los a adquirir o software.

Para maior facilidade, a Informação Introdutória é dividida em três conjuntos de documentação.

✓ A Informação Publicitária, que dá uma visão geral do software: mostra como o software pode ajudar o usuário e o relaciona com a sua área de aplicação.

✓ A Descrição Funcional oferece informações mais detalhadas sobre as funções do software e seus resultados.

✓ Resumo dos Detalhes Técnicos indica qual o hardware de computador e o software necessário para a utilização deste software.

9.2 Informação Publicitária

9.2.1 Uso

A Informação Publicitária é a primeira documentação que um cliente em perspectiva de software irá receber. Para despertar seu interesse, ela deverá descrever a finalidade do software de forma facilmente compreensível. As informações devem ser genéricas, porém suficientes para tornar sua aplicação clara e evitar a maioria das dúvidas que causam perda de tempo. Particularidades técnicas de computação devem ser evitadas, a menos que o software se destine a utilizado apenas por profissionais do ramo.

9.2.2 Quando Preparar

A Informação Publicitária pode ser inicialmente editada quando o fornecedor tiver concluído as especificações do software e providenciado para que este seja codificado. Novas edições devem ser publicadas sempre que houver modificações nas especificações básicas.

9.2.3 Conteúdo

✓ Nome do software

✓ Descrição dos objetos do software e das opções disponíveis, mostrando as vantagens para o usuário.

✓ Descrição geral do hardware onde o software será processado e de quaisquer outros pré-requisitos.

✓ Contato para maiores (ou mais recentes) informações e para consultas referentes à atualidade das informações.

✓ Data da publicação desta informação.

9.3 DESCRIÇÃO FUNCIONAL

9.3.1 Uso

A Descrição Funcional oferece mais detalhes sobre as funções do software ao usuário em perspectiva que, depois de ler a Informação Publicitária, deseja saber mais a respeito do software, antes de tomar a decisão de adquiri-lo. As informações apresentadas no Resumo dos Detalhes Técnicos podem ser suficientes para que o usuário em perspectiva venha a se decidir. Caso essas informações sejam ainda insuficientes para o usuário em perspectiva, este irá estudar a Descrição Funcional, para ver se o software fornece o que ele deseja, e o faz de uma maneira satisfória.

9.3.2 Quando Preparar

Os aspectos do software descritos na Descrição Funcional são determinados logo no início do projeto. Sendo assim, este tipo de documentação pode ser produzido durante os primeiros estágios, mesmo que seja necessário alterá-lo mais tarde.

9.3.3 Conteúdo

✓ Nome do software

✓ Objetivos do software
- ➢ Pré-requisitos para usar o software

✓ Descrição esquematizada, identificando as funções mais importantes e os recursos disponíveis , indicando se são obrigatórios ou opcionais.

✓ Para cada função :
- ➢ Entrada requerida (em termos bem gerais)
- ➢ Saída fornecida pelo software(em termos bem gerais)

- Descrição de quaisquer regras de processamento significativas
- Lista de manuais e outras documentações disponíveis

✓ Contato para consultas

✓ Data da publicação desta informação.

9.4 Resumo dos Detalhes Técnicos

9.4.1 Uso

O Resumo dos Detalhes Técnicos deve fornecer mais detalhes que a Informação Publicitária, a respeito do Hardware, dos requisitos para o software, dos custos e tempos de processamento. Se o usuário em perspectiva for tecnicamente competente (no que diz respeito a computação), a partir desta documentação estará apto a julgar se o software pode ser utilizado pela instalação em que ele o deseja usar, e qual será o seu custo. Um usuário inexperiente deve procurar orientação, provavelmente da equipe de profissionais da instalação em questão, para saber se o software é tecnicamente exeqüível e economicamente justificável.

9.4.2 Quando Preparar

Normalmente esta documentação será preparada depois que o software tiver sido completamente testado, em pelo menos uma configuração, de modo que nela possam ser incluídos detalhes sobre os testes.

9.4.3 Conteúdo

✓ Nome do software.

✓ Objetivos do software.

✓ Processadores nos quais o software foi utilizado e aqueles nos quais ele também poderá vir a ser usado.

✓ Requisitos de memória

✓ Equipamentos periféricos utilizados

✓ Todo equipamento adicional necessário

✓ Modalidade de processamento (se for iterativo, as necessidades de terminais e de equipamentos de comunicação devem ser estabelecidas).

✓ Arquivos mestres necessários (quantidade, tamanho e meio de armazenamento).

✓ Arquivos de trabalho necessários (quantidade, tamanho e meio de armazenamento).

✓ Sistemas Operacionais com os quais o software foi projetado para funcionar.

✓ Linguagem(s), compilador(s) e outros itens de software necessários

Atividades para recuperação do processamento e dos dados, em caso de defeitos (em termos gerais)

Controles para o processo

✓ Projeção e segurança dos dados (em termos gerais)

Custo e condições de venda de todas as configurações disponíveis do software

Suporte e manutenção fornecidos dentro do preço-base

Disponibilidade e custo de cursos de treinamento (se existirem) com detalhes sobre o contrato.

✓ Contato para consultas

✓ Data da publicação

9.5 Documentação Do Usuário

9.5.1 Objetivo da Documentação do Usuário

Antes de ser possível a utilização do software, três etapas devem ser cumpridas. Em primeiro lugar o usuário precisa saber como usar o software. Para isto ,há a necessidade de um treinamento formal. Em segundo lugar, a equipe de operação da instalação do computador em que o software será utilizado, deve saber como operar este software. Em terceiro lugar, o software deve ser instalado no computador. A documentação do Usuário deve ser projetada para ajudá-lo nessas três etapas, não só durante a instalação, como também mais tarde, como fonte de referência.

9.5.2 Tipos de Documentação do Usuário

A documentação básica do usuário é a Documentação de Referência do Usuário. Ela oferece uma visão geral do "ponto de vista do usuário". A documentação de Treinamento do Usuário cobre em grande parte os mesmos assuntos, mas é projetada para ser usada como auxílio no treinamento de um novo usuário e não como documentação de referência. A Documentação da Operação fornece os detalhes necessários para operar o software na instalação do computador e a documentação para Instalação descreve, essencialmente, como instalar o software no computador.

9.6 Documentação de Referência do Usuário

A principal documentação de um usuário de software é a documentação de Referência do Usuário. Ela deve conter todas as informações necessárias para que todos os recursos e opções possam ser utilizados . Também deve constar da Documentação de Referência do Usuário a orientação para que o software seja utilizado da melhor maneira possível. Todas as possibilidades de entrada e saída do sistema (inclusive de erros), como exemplos , também devem constar dessa documentação. As mensagens de erros devem ser bem claras, assim como as várias etapas a serem seguidas para identificá-los e corrigi-los. O formato da documentação deve ser adequada para que possa ser usada como fonte de referência.

Um usuário em perspectiva pode utilizar a Documentação de Referência do Usuário para verificar se o software executa as tarefas que ele tem em mente e o faz adequadamente. Ele também pode querer certificar-se de que a documentação , particularmente esta, é satisfatória.

9.6.1 Quando Preparar

A maior parte deste material, ou seu rascunho, deve ser preparada enquanto o software estiver sendo projetado e codificado. Esta documentação deve estar pronta ao mesmo tempo que o software. Deste modo, ambos serão testados simultaneamente. Isto será útil, especialmente se for feito um teste no campo, seja para um software inteiramente novo, seja para um componente adicional. Em qualquer hipótese o manual deve chegar às mãos do usuário o mais cedo possível e de preferência, até mesmo antes do software.

9.6.2 Conteúdo

- ✓ Nome do software e referência da versão
- ✓ Índice
- ✓ Informação Introdutória

- Descrição
- Problemas que o software pode resolver
- Descrição do sistema quanto aos programas, componentes e módulos
- Requisitos de Hardware
- Requisitos de software (inclusive sistema operacional)
- Método de acesso
- Rotinas de segurança e técnicas utilizadas
- Relacionamento entre os problemas resolvidos e seus componentes

Componentes do sistema. Para cada componente deve ser descrito em detalhe :

- ✓ Problema(s) que o componente ajuda a resolver
- ✓ Informação de entrada que entrada é necessária formato meio
- ✓ Processamento efeito da entrada e efeitos colaterais erros que podem ser detectados , saída impressa resultante e como corrigir o erro proteção e segurança dos dados
- ✓ Saída formato meio exatidão opções
- ✓ Exemplos práticos que incluam amostras de entrada e saída.

Uso do terminal se o software é usado por meio de um terminal, há necessidade de incluir as instruções completas de como operar o terminal (telas e menus) .

Informações Administrativas

- ✓ Contatos para consultas
- ✓ Procedimentos para atualizar a documentação
- ✓ Procedimentos para relatar erros na documentação
- ✓ Procedimentos para informar sobre erros no software
- ✓ Referência da Documentação de Treinamento do Usuário (se houver)
- ✓ Referência da Documentação de Operação (se houver)
- ✓ Data da publicação
- ✓ Glossário
- ✓ Índice
- ✓ Data da publicação.

9.7 Documentação de Treinamento do Usuário

A Documentação de treinamento do Usuário tem como objetivo ajudá-lo a aprender a usar o software o mais rapidamente possível. Essa documentação pode ser utilizada como uma

alternativa em lugar de um treinamento direto. Normalmente serve como complemento da instrução formal. A documentação deve fornecer instruções claras sobre como deve ser usada.

9.7.1 Quando Preparar

Assim como a Documentação de Referência do Usuário, esta documentação deve ser testada “no campo” ao mesmo tempo que o software. Deve estar pronta a tempo que o usuário possa estudá-la completamente antes de receber o software.

9.7.2 Conteúdo

As mesmas áreas cobertas pela Documentação de Referência do Usuário devem constar da Documentação de Treinamento. Os primeiros parágrafos devem apresentar, claramente, os objetivos e instruções sobre o método de utilização. Do ponto de vista didático, é preferível que cada um dos componentes do software seja chamado separadamente, iniciando pelo mais simples. O ideal é que a Documentação de Treinamento apresente apenas casos simples, deixando explicações sobre usos mais complexos do software para a Documentação de Referência do Usuário. Para que possa ser usada sem supervisão, todas as etapas devem ser seguidas de exemplos.

Todos os manuais de auto-aprendizagem devem trazer questões que o leitor deve tentar responder em outra parte do manual. Se o software pode vir a ser utilizado por pessoas pouco experientes, os conceitos devem ser simples e objetivos. Se o software de destina a pessoas com experiência em computação, pode-se assumir que o leitor tenha algum conhecimento técnico; mesmo assim, é aconselhável que os problemas técnicos sejam explicados para evitar quaisquer possíveis mal-entendidos. O grau de conhecimento necessário para que o manual possa ser compreendido deve ser estabelecido com clareza.

9.8 Documentação da Operação

A Documentação da Operação deve apresentar as instruções necessárias para a operação normal do software em uma configuração específica. Também deve abordar a transcrição dos dados da entrada para meios legíveis para o sistema, caso seja necessária tal transcrição.

Inicialmente, mesmo para a utilização normal do software, a equipe de operação precisará consultar esta documentação. Com o tempo, a experiência fará com que as consultas sejam cada vez menos freqüentes. No entanto, ela deve estar sempre à mão para ser utilizada caso apareça uma mensagem pouco utilizada pelo sistema.

A equipe encarregada do preparo dos dados também utilizará a documentação durante o período em que estiver aprendendo a transformar os dados em meios legíveis pelo sistema, e mais tarde, quando totalmente treinados, no caso de dúvidas.

Um usuário em perspectiva pode querer ver esta documentação antes de decidir se irá ou não usar o software. Ele pode desejar que sua equipe de operações verifique se a documentação é adequada e se os procedimentos de preparo e operação dos dados são razoáveis.

9.8.1 Quando Preparar

Esta documentação deve estar pronta a tempo de ser testada por instalações pioneiras.

9.8.2 Conteúdo

Nome dos software e referência da versão do Manual Introdutório

- ✓ Detalhes resumidos sobre a aplicação
- ✓ Sumário dos procedimentos

Requisitos para cada uma das principais opções.

- ✓ Requisitos do computador
  - ➢ Tamanho de memória (mínimo e aconselhável)
  - ➢ Periféricos

- ✓ Requisitos de software
  - ➢ Sistema operacional
  - ➢ Outros requisitos

- ✓ Arquivos usados - para cada arquivo
  - ➢ Objetivo
  - ➢ Meio
  - ➢ Tamanho

Entrada

- ✓ Dados de entrada
  - ➢ Formulário original
  - ➢ Meio de leitura pelo computador
  - ➢ Requisitos para segurança dos dados

Saída

- ✓ Dados de saída
  - ➢ Meio
  - ➢ Formato de operação.

Informações Administrativas

- ✓ Contato para consultas e maiores informações
- ✓ Procedimentos para atualizarão da documentação
  - ➢ Procedimentos para relatórios sobre erros na documentação
  - ➢ Procedimentos para relatórios sobre erros no software
  - ➢ Referências para a Documentação de referência do Usuário

- ✓ Data da publicação
  - ➢ Glossário
  - ➢ Índice

9.9 Documentação para Instalação

A documentação para Instalação é usa da quando o software é colocado no sistema do computador e deve especificar, em detalhe, as ações necessárias para a instalação. Pode ser utilizada pelo usuário ou pelo fornecedor do software, de acordo com o que for estabelecido entre os mesmos. Se o software for carregado no sistema de computação em todas as vezes que for utilizado, os procedimentos para esta tarefa devem ser incluídos na documentação da Operação, e não há necessidade de uma Documentação para instalação, em separado. Caso contrário, a Documentação para Instalação deve cobrir todos os procedimentos necessários, antes que os procedimentos normais especificados na Documentação da Operação possam ser executados . O procedimento da instalação pode ser efetuado apenas uma vez, ou pode haver necessidade de repetir a operação quando nova versões do software forem emitidas.

9.9.1 Quando Preparar

A documentação para Instalação (ou pelo menos seu rascunho) deve estar pronta para ser testada ao mesmo tempo que o software.

9.9.2 Conteúdo

* Nome do software e referência da versão

Requisitos

- ✓ Tipo do computador
- ✓ Sistema operacional

Outros software necessários

- ✓ Hardware necessário para implantação
- ✓ Material fornecido para implantação

Procedimentos

- ✓ Procedimentos passo a passo
- ✓ Arquivos e/ou bibliotecas a serem criados
- ✓ Cartões de controle para execução do serviço ou outras instruções para o computador

Saída que deve ser produzida

- ✓ Referência para a documentação dos testes necessários para o processamento, inclusive resultados

Informações Administrativas

- ✓ Contato(s) para o caso de surgirem problemas e para verificar se as informações ainda estão atualizadas
- ✓ Procedimentos para relatar erros encontrados na documentação
- ✓ Data de publicação

Glossário

Índice

9.10 Documentação da Manutenção

9.10.1 Aspectos Gerais da Documentação da Manutenção

O conjunto mais importante da Documentação da Manutenção é o da Documentação do Sistema e da Programação. Esta documentação fornece detalhes completos sobre todos os programas e arquivos em suas versões atualizadas. A Documentação dos Testes é um registro dos testes projetados para verificar se o software funciona corretamente. A Documentação do Histórico, como o próprio nome indica, contém informações sobre a criação do software, inclusive planos da estratégia geral, que podem ser úteis para a manutenção. O último conjunto desta documentação é a Documentação de Atualização dos Manuais, através da qual os usuários são informados das últimas novidades e atualizam sua documentação.

A Documentação da Manutenção difere da Documentação do Usuário e da Informação Introdutória porque, enquanto as duas últimas permanecem em mãos do usuário, a Documentação da Manutenção fica em poder do fornecedor e será utilizada apenas internamente. Há pouquíssimas cópias dessa documentação - talvez apenas duas : uma em usa e outra em back-up e guardada em lugar seguro.

10. METODOLOGIAS E TÉCNICAS

Os problemas que afetam as empresas em razão da crises por que passam o desenvolvimento de software, ou seja , as preocupações com produtividade , manutenabilidade e qualidade de software, além dos problemas gerados pela falta de credibilidade nos prazos e estimativas de custos na área. Dentro desse enfoque e como parte da revolução que surge em desenvolvimento de sistemas, muito bem descrito por MARTIN, 1983, há a presença obrigatória das metodologias estruturadas e das suas respectivas técnicas. Essas metodologias não são novas, no entanto, durante décadas seu uso estava restrito a poucas organizações e em métodos baseados no papel, o que tornava cansativa e exaustiva a tarefa de desenvolver sistemas, utilizando-se das técnicas estruturadas.

Com a revolução dos computadores pessoais, houve um grande impacto quando surgiu no ano de 1982, ano em que o computador foi eleito a "Máquina do Ano" pela revista Time ("Machine of the Year, The Computer Moves In", 01/01/1993 n.1.), quebrando uma tradição da revista. Da mesma forma, no início de 1984 , a mesma revista Time ("Computer Software - The Wizard the Machine", 16/04/84 n.16) trazia como reportagem de capa "The Wizard inside the Machines", enfatizando o

software como sendo o "tapete mágico do futura", e nessa época, os software eram apenas "pré-históricos" se compararmos com a evolução surgida na segunda metade da década de 80 e início dos anos 90 e hoje com os ambientes gráficos, onde o destaque fica por conta da Multimídia.

Dentro deste contexto, o desenvolvimento de software, era feito ainda com as mesmas ferramentas de décadas atrás, enquanto a indústria e diversos outros setores passavam a viver uma verdadeira revolução, com as máquinas de controle numérico computadorizado, as ferramentas de CAD/CAM("CAD - Computer-Aided Design, CAM - Computer-Aided Manufacturing"), a robótica e a indústria de Computer Graphics.

Hoje, devido a todos esses avanços surge a automação das metodologias e , principalmente as interfaces gráficas, tão necessárias ao desenvolvimento de software através do uso de ferramentas.

Podemos definir metodologia como :

✓ Um conjunto de procedimentos capazes de orientar um grupo de usuários, quanto aos passos que eles deverão executar, informando aos mesmos o que fazer a seguir.

✓ Um conjunto de métodos juntamente com um conjunto de ferramentas, técnicas de gerência e normas, que auxiliam no processo de desenvolvimento do sistema.

Deste modo, uma metodologia deve possuir atributo de fornecer suporte para a realização das atividades envolvidas no processo de desenvolvimento. Uma metodologia, portanto, deve possuir suporte para a construção, verificação e validação de especificações, suporte para gerência(por exemplo, suporte para a criação do Plano do Projeto de modo a facilitar a avaliação da viabilidade), suporte para modificação etc.

Uma metodologia, para ser um apoio eficaz à qualidade da especificação, deve portando:

✓ Possuir suporte (de preferência, automatizado) para realização das tarefas envolvidas no processo de desenvolvimento (evolução, verificação , validação, modificação, reconstrução e gerência);

✓ Ter um amplo campo de aplicabilidade, isto é, poder ser aplicada a diversos tipos de problemas e durante as diferentes fases do ciclo de vida. Isto, sem dúvida, favorece a comunicação com o usuário que não teria de aprender várias metodologias.

Uma evolução cronológica aproximada, das metodologias de desenvolvimento de sistemas pode ser dada como :

1965 - Metodologia estruturada.

1970 - Técnicas de modelagem de dados.

1975 - Projeto de banco de dados.
Banco de dados e linguagens de quarta geração.

1980 - Especificação do projeto.
Ferramentas de software.
Modelagem de dados.
Linguagens de Quarta Geração.
Prototipação.

1985 - Interface com o usuário.
Ferramentas de prototipação.
Automação das metodologias.

1990 - Ferramentas de geração de código.
Metodologias Orientadas a Objetos.

10.1 Metodologia e CASE

Com a implementação de ferramentas CASE ("Computer-Aided Software Engineering") sem uma forte e fundamental metodologia, é improvável que se obtenha os benefícios desejados.

Entretanto, ferramentas e metodologias são tão ligadas que implementam primeiro uma metodologia que pode levar a problemas.

Uma escolha por metodologias pode revelar-se inapropriada para automação com ferramentas CASE, caso se escolha uma metodologia manual, o tédio envolvido pode desviar-se para a metodologia e conseqüentemente para CASE.

Por causa do refletido processo de automação, inevitavelmente mudanças no processo, tecnologia CASE e metodologia são melhor implementados juntos. Com uma estratégica bem definida e um projeto piloto cuidadosamente planejado ou uma série de projetos pilotos.

Problemas com o Desenvolvimento Tradicional	***Os Efeitos da Metodologia***
O desenvolvimento leva muito tempo.	**O desenvolvimento é muito mais rápido.**
Os custos de desenvolvimento são muito	**Uma redução importante nos custos de**

altos.	desenvolvimento.
A programação é manual.	A programação pode ser automatizado.
A maioria dos erros são achados após a codificação.	A maioria dos erros são encontrados antes da implementação.
A maioria dos erros são encontrados manualmente ou através de operações dinâmicas.	A maioria dos erros são encontrados por análise automática e estática.
Alguns erros nunca são encontrados.	A maioria dos erros são encontrados.
Especificações inconsistentes.	Completas e consistentes são reforçadas.
Muitos erros de interface de desencontros entre sistemas.	Interfaces rigorosas e comprovadamente corretas entre os subsistemas.
Nenhuma garantia de integridade da função após a implementação.	Garantia da integridade da função após a implementação.
Grandes equipes de desenvolvimento problemas sérios de comunicação.	Equipes menores ou de uma pessoa; menos problemas de comunicação.
Grande papelada para o controle administrativo.	Eliminação da maioria da papelada.
Muito desenvolvimento de código.	Identificação de módulos comuns. Utilização de módulos comuns feita mais facilmente.
Pessoal de desenvolvimento separado reinventando a roda.	Mecanismo de biblioteca de interfaces rigorosas encoraja construção de estruturas reutilizáveis.
Dificuldade de manutenção.	Facilidade de manutenção.
As modificações disparam reações em cadeia de novos erros.	Os efeitos de todas as modificações são feitos de forma clara.
Manutenção sucessiva deteriora a qualidade do código.	Código de alta qualidade e regenerado após cada modificação.
Problemas na portabilidade.	Um projeto pode ser regenerado para diferentes ambientes.

11. PROJETO DE INTERFACES

Fatores humanos para avaliação da interfaces :

- ✓ Tempo para aprender
- ✓ Desempenho
- ✓ Taxa de erros na utilização
- ✓ Satisfação subjetiva
- ✓ Retenção do conhecimento

Alguns princípios a serem considerados no projeto da interface com o usuário

- ✓ A função principal da interface com o usuário é a comunicação
- ✓ A interface deve manter o computador como estando entre o usuário e o trabalho
- ✓ A interface deve ser consistente por todo o programa .
- ✓ A interface deve ser flexível para satisfazer uma gama variada de usuários.

- ✓ A interface deve manter o usuário distante de como a tarefa está sendo realizada.
- ✓ usuário deve ter acesso a ajuda interativa ("HELP").
- ✓ programa não deve ter erros de execução.
- ✓ programa deve trabalhar mais para que o usuário trabalhe menos.
- ✓ Buscar fazer com que seu programa trabalhe como outros similares a ele.

"OITO REGRAS DE OURO" no projeto de interfaces

- ✓ Busca pela consistência :
  - ➢ Terminologia idêntica por todo o programa, comandos e resultados idênticos.

- ✓ Oferecer "atalhos" para usuários freqüentes (experientes) :
  - ➢ Teclas especiais, comandos ocultos, macros.

- ✓ Oferecer "feedback" (realimentação) informativo:
  - ➢ Toda ação executada deve mostrar alguma informação da execução

- ✓ Projetar diálogos que guiem a conclusão de tarefas :
  - ➢ Ações devem ser organizadas de forma que o usuário saiba que há um início, meio e fim.

- ✓ Oferecer meios simples para manipulação de erros :
  - ➢ Conduzir o usuário a não cometer erros muito sérios, oferecer soluções simples para erros.

- ✓ Permitir desfazer ações ("UNDO") :
  - ➢ Diminuir a ansiedade do usuário, incentivar a exploração.

- ✓ Suportar facilidades de controle direto pelo usuário :
  - ➢ Permitir que usuários experientes controlem o programa e não o inverso.

- ✓ Reduzir a necessidade de memorização:
  - ➢ Manter telas simples, telas múltiplas conectadas, evitar movimentação constante na tela.

11.1 Interação por "Menus"

- ✓ Utilizar estrutura semântica para montar "menus"
  - ➢ simples, árvore , etc.
- ✓ Indicar posição atual na estrutura:
  - ➢ numeração, títulos, sobreposição de menu

- ✓ Opções tornam-se títulos ao se selecioná-las
- ✓ Criar grupos e seqüências de itens significativos
- ✓ Itens devem possuir estilo consistente
- ✓ Permitir atalhos
- ✓ Permitir retorno rápido ao menu principal ou anterior
- ✓ Considerar: - tempo de resposta
  - ➢ tamanho da tela
- ✓ Oferecer ajuda interativa

11.2 Interação por Preenchimento de Lacunas

- ✓ Títulos significativos
- ✓ Instruções significativas/compreensíveis
- ✓ Agrupamento lógico de campos
- ✓ Disposição uniforme das lacunas
- ✓ Terminologia e abreviaturas consistentes
- ✓ Facilidade de correção de caracteres e campos
- ✓ Apresentação de espaço disponível
- ✓ Facilidade de ajuda interativa

11.3 Interação por Linguagem (de Comandos/Natural)

- ✓ Criar modelos de ações e objetos claros
- ✓ Escolher nomes específicos, significativos e distintos
- ✓ Prover estrutura (de ações/objetos) consistente
- ✓ Criar abreviações consistentes
- ✓ Permitir ao usuário criação de macros
- ✓ Limitar o número de comandos e modos para atingir um objetivo.

11.4 Estilos de Interação com o Usuário

Estilo	*Vantagens*	*Desvantagens*
Seleção por "MENU" (cardápio) de opções	❖ **Acelera o conhecimento sobre o programa;** ❖ **Facilidade para suporte a erros;**	❖ **Cansa usuário freqüentes;** ❖ **Gasta muito espaço em tela;** ❖ **Necessita de rapidez na**

	❖ Reduz combinações de teclas.	apresentação.
Preenchimento de lacunas	❖ Simplifica entrada de dados; ❖ Requer treinamento simples; ❖ Uso de gerenciadores de lacunas.	❖ Consome muito espaço em tela.
Linguagem de comando	❖ Flexibilidade; ❖ Dá "poder" ao usuário; ❖ Incentiva a iniciativa do usuário.	❖ Dificuldade na manipulação de erros; ❖ Requer treinamento e memorização.
Linguagem natural	❖ Liberdade na escrita do comando (sintaxe).	❖ Necessita diálogos para especificar ações; ❖ Pode requerer combinações de teclas.
Manipulação direta	❖ Fácil de aprender; ❖ Fácil de se lembrar; ❖ Encoraja exploração; ❖ Satisfação subjetiva elevada.	❖ Dificuldade na implementação; ❖ Necessita hardware específico.

11.5 Chamando a Atenção do Usuário

- ✓ Intensidade do texto: dois níveis somente
- ✓ Marcação de texto/opção : - sublinhado
- ✓ Tamanho do fonte(máximo) : 4 tamanhos
- ✓ Tipos de fontes(máximo) : 3 fontes
- ✓ Vídeo reverso : normal ou reverso
- ✓ Piscante : não piscante ou piscante (2 a 4 Hz)
- ✓ Áudio : suaves para ações regulares
  - ➢ ativo para emergência
- ✓ Mensagens ("feedback") 1 a 3 segundos após a início da ação

11.6 Mensagens de Erro ao Usuário :

- ✓ Ser o mais específico e preciso possível
- ✓ Ser construtivo : indicar o que deve ser feito
- ✓ Usar tom positivo : evitar condenação
- ✓ Escolher mensagens centradas no usuário
- ✓ Manter consistência gramatical, de terminologia e abreviaturas
- ✓ Manter consistência visual de forma e localização

12. TRABALHO DE QUALIDADE DE SOFTWARE

A sociedade da segunda metade do século XX está , marcada pelo aparecimento e uso do computador. São inúmeras suas aplicações atuais e estas tendem aumentar. Esta disseminação de uso do computador afeta a um grande número de pessoas que, cada vez mais, são sensíveis a suas implicações sociais e exigentes quando à qualidade dos sistemas de aplicação.

A necessidade de um cuidadoso controle da qualidade surge com a questão do desenvolvimento e manutenção do que se convencionou chamar software produto.

Atualmente, ao desenvolvermos um software produto , já não podemos mais tolerar o uso de métodos artesanais de desenvolvimento de programas por uma série de razões, entre elas :

- ✓ É necessário assegurar elevada confiabilidade e manutenibilidade aos programas.
- ✓ É necessário redução dos custos de desenvolvimento e manutenção dos programas.

- ✓ É necessário aumentar a produtividade das equipes de desenvolvimento de programas.

Nesta sociedade repleta de informação, o uso da informática não é a melhor opção: É a única.

Os sistemas de informações são organizados nas empresas e instituições de forma a espelhar as suas estruturas e os seus valores gerenciais. Em razão disso, o sucesso na utilização das ferramentas e recursos de informática não deve ser medido pelo domínio desses instrumentos mas pela efetividade no apoio e suporte aos processos gerenciais, que têm evoluído sensivelmente nos anos.

Cada vez mais o bom desempenho de corporações depende de um departamento de informática o mais livre possível de falhas. Na busca da qualidade, é importante que tudo seja documentado e que haja integração total da equipe de desenvolvimento para que o resultado final seja um produto de fácil utilização e com o mínimo de defeitos.

Com a distribuição dos dados, a garantia de um produto de bom desempenho é crucial para o segmento corporativo. E para isso, entidades e consultorias do mundo todo vêm desenvolvendo metodologias que leve à criação de aplicativos o grau mais próximo da perfeição. "O controle de qualidade está se tornando mais importante, porque hoje os aplicativos são mais críticos ao ambiente corporativo de que eram no passado", afirma o consultor Cezar Taurion.

Mas o que é qualidade de software? Basicamente, é o software que funciona sem defeitos, apresentando uma imagem de confiabilidade para seus usuários. A arquitetura cliente/servidor apresenta algumas características peculiares e é inerentemente mais complexo que o ambiente centralizado. A interface gráfica e seu paradigma de "objeto-ação" e a parafernália de software que envolve uma aplicação geram desafios maiores para a qualidade.

Este ambiente se caracteriza por apresentar um potencial maior de defeitos, com menor eficiência na sua remoção. Como conseqüência, mais bug’s são entregues aos usuários. Uma recente pesquisa demonstrou que, em média, uma aplicação cliente/servidor é entregue aos usuários com pelo menos 20% dos seus possíveis bug’s.

Implementar qualidade de software exige uma mudança na própria cultura da organização. O nível adequado de maturidade organizacional e tecnológica é extremamente importante para que abordagem voltada para qualidade tenha sucesso.

De maneira geral, cerca de 50% do custo de um projeto está relacionado com atividades de teste. Entretanto, o custo do não-teste(ou não qualidade) pode ser muitas vezes superior, com grande

insatisfação do cliente, inconformado com a instabilidade e conseqüentemente falta de confiabilidade no sistema.

É, portanto, de suma importância que um processo de qualidade de software, através de um método eficiente de teste, seja adotado pela organização. Este método deve ser iterativo, pois não mais se considera adequado um processo de desenvolvimento estático, em cascata, com etapas bem definidas. Os métodos iterativos, de prototipação, produzem aplicações através de refinamentos iterativos de desenho e programação. Neste contexto, os teste também devem ser iterativos, sucessivamente refinados.

Os processos que buscam qualidade do software têm de ser embutidos desde o início do projeto. Deve incluir uma fase planejamento que envolva considerações sobre a equipe de teste e quais aspectos específicos da aplicação devem ser testados mais detalhadamente. Deve incluir as fases de projeto e desenvolvimento do teste, que vão especificar e produzir os procedimentos que serão aplicados no desenrolar dos testes em si, e finalmente, com certeza deverão existir medições para avaliar-se e eficiência dos processos de teste adotados.

Os testes devem objetivar não só a busca da diminuição do número de defeitos, mas também o máximo de aderência às regras de negócio, além fazer garantir que a performance se mantenha dentro das expectativas, mesmo em situações de stress da aplicação.

Embora as técnicas e sistemas de informações para gerenciamento das empresas e instituições não produzam resultados por si (os resultados são obtidos pelo seu efetivo pelas pessoas), melhores informações contribuem decisivamente para melhorar a qualidade das decisões.

O funcionamento de uma grande empresa pode requerer a execução de uma quantidade considerável de atividades, o que dificulta o planejamento e o controle de suas operações.

Devidamente complementado e adaptado às necessidades de cada empresa, e utilizado nas fases subseqüentes ao ciclo de desenvolvimento e de implantação de sistemas, servirá de base para a construção de um instrumento que proporcionará aos administradores uma visão global das operações significativas de uma empresa, de seus custos e receitas. Facilitará, portanto, a adoção de medidas que resultem em maiores benefícios para o conjunto da organização.

Com o uso adequado de tal instrumento, será possível detectar e corrigir desvios entre volumes e custos previstos e realizados, contribuindo-se, dessa forma, para atingir o retorno previsto para os investimentos.

O citado instrumento poderá apoiar um processo de planejamento e controle flexível, que estimule a aproveitamento de oportunidades não previstas de aumento de lucros e de melhoria de resultados. A empresa passará a dispor de facilidade para planejar e acompanhar a contribuição de

cada unidade da organização, de cada atividade significativa e de cada produto para atingir os objetivos empresariais.

Facilitará, também, a implantação de controles compatíveis com que está sendo controlado e sua adaptação às mudanças necessárias ao negócio. Poderá proporcionar, ainda, informações por cliente ou por grupo de clientes, relativas à receita obtida e aos custos dos produtos e serviços, e dados comparativos de diferentes fontes de obtenção de recursos(por exemplo, fornecedores internos x externos à empresa).

Os administradores poderão dispor tanto do dados dos custos associados diretamente à geração dos produtos e serviços para os clientes da organização, quanto dos custos associados aos serviços de apoio.

"Os custos de uma organização podem ser gerenciados de forma mais eficaz através de modelo de sistemas de informação bem preparados e detalhados. Para domar os imprevistos e enfrentar melhor o próximo desafio, ponha os dados na manga e vá à luta! ".

Agora vamos ver alguma experiência em algumas empresas com os seus software .

A empresa Klabin do setor de papel , que já definiu que o caminho da companhia é ter um sistema totalmente integrado. A disputa, no caso, está entre os produtos da SAP e da Baan, também os favoritos entre as corporações estrangeiras. O processo vai consumir a maior parte do investimento de US$ 22,5 milhões projetado para 1997.

"O objetivo não é só colocar o software no ar. A empresa quer resultados práticos em custo e no faturamento. A monitoração não é apenas saber se ele está funcionando mas que ele faz dentro da gestão da companhia", argumenta Antônio J. R. Almeida, CIO da Klabin.

A Infraero descobriu duas coisas importantes este ano. A primeira é que os aeroportos são uma grande fonte de negócio - a empresa deve faturar US$ 1 bilhão em 96. A outra precisa melhorar o serviço prestado.

"O objetivo é colocar o que existe de mais moderno em operação nos aeroportos estrangeiros aqui no Brasil. Pesquisamos o que é feito na Alemanha, Suíça e França e trouxemos esta concepção moderna de check-in", garante Flávio da Rocha. O carro-chefe do novo formato do check-in, o Cais, são os cartões de embarque magnéticos, desenvolvidos pela Infraero em parceria com a UNB - Universidade de Brasília. Criado com a ferramenta Case da Oracle, o aplicativo levou 8 meses para ser concluído e tem interface gráfica de última geração.

A estimativa inicial é que o sistema reduza o tempo de check-in em 30 ou mesmo 40%.

Todo o processo da feitura deste aplicativo, assim como a própria migração de sistemas, está sendo feito com a ferramenta Case da Oracle. A empresa, por meio de licitação, também é a fornecedora do banco de dados, da informatização de sistemas de cargas e a responsável por todo o suporte técnico à Infraero.

Seguindo a mesma filosofia do Cais, a Infraero partiu para desenvolvimento de aplicativos modernos, como o Teca Plus para cargas aéreas e o SIV, como sistema informativo de vôo. O Teca, também realizado com o auxílio da UNB, está em fases de teste nos aeroportos de Porto Alegre, Manaus e Garulhos, em São Paulo. Já o SIV, feito internamente, está em piloto em Brasília e Manaus, com a instalação de quiosques para os usuários destes aeroportos. As pessoas já podem consultar uma série de informações, como o horário dos vôos, saída e chegada previstas, além de informes mais gerais.

"Este é o caminho que será percorrido este ano, mas a modernização será perene. No próximo ano vamos disseminar os sistemas em todas as bases da Infraero e partir para a Internet", revela Flávio da Rocha.

O Grupo Ticket é pioneiro no uso de smart cards.

"A substituição integral do papel pelos cartões de última geração é que vai direcionar a modernização do grupo, que passa pela informatização de todos os processos. Do atendimento ao cliente final até o controle maior dos nossos dados", garante Luiz Peduti, gerente de Marketing da Ticket.

"A receptividade foi muito boa, com o serviço ligado às transportadoras de cargas. Os resultados preliminares, indicam a redução de 15% nos custos, de corrente do maior controle de gastos que o cartão possibilita" garante Henrique Valério Silva Jr., gerente de informática da Ticket.

"Os restaurantes vão ganhar muito, porque não precisarão preencher guias de recolhimento. É só passar o cartão nas nossas máquinas que o crédito será depositado em três dias úteis" explica Peduti. Esta dinâmica no reembolso será obtida mediante o pagamento de uma taxa de 2% sobre o valor do gasto. Hoje o prazo é de 14 dias.

A Prefeitura de Osasco na Grande São Paulo, está informatizando o setor de saúde. O projeto vai otimizar as consultas feitas pela população nas 42 unidades hospitalares da cidade. A previsão é de que, a partir de agosto, mais de 150 mil pessoas sejam beneficiadas mensalmente com o novo sistema. A proposta é melhorar o atendimento e o controle dos pacientes.

Os resultados da informatização, no entanto, surgem em vários níveis nos locais já conectados. O primeiro aspecto positivo foi a racionalização do atendimento, com consultas

marcadas diretamente no computador, até mesmo quando pedidas pelo telefone. O impacto mais evidente foi a diminuição das longas filas nos postos de saúde e hospitais em algo suportável pelos pacientes.

Outras aplicações serão implementadas com o crescimento do sistema, como o prontuário único. O objetivo é permitir que qualquer paciente com ficha na rede pública possa se consultar em uma dasc 42 unidades, com seu histórico disponível em toda a rede . Além disto, qualquer campanha do município, como as de vacinações, poderá ser planejada e gerenciada com informações abrangentes e on-line.

"Ele vai permitir ainda a visualização rápida das patologias mais freqüentes em cada uma das regiões da cidade, possibilitando mais dinamismo nas ações direcionadas aos problemas", entusiasma-se Garroti, diretor de Apoio Operacional da Prefeitura de Osasco e responsável pela implantação do sistema.

13. OS REQUISITOS DE HARDWARE/SOFTWARE/REDE

13.1 Descrição das Tarefas

✓ Analisar detalhadamente os requisitos funcionais e de dados descritos nos modelos funcionais de processo e de dados do sistema.

✓ Modificar ou complementar os requisitos de hardware/software/rede identificados durante a fase de análise preliminar. Incluir quaisquer requisitos adicionais necessários para suportar os futuros processos de conversão, teste e treinamento.

✓ Consolidar os requisitos detalhados de hardware/software/rede. Garantir que estejam completos e sejam compatíveis.

✓ Descrever as características físicas dos equipamentos e facilidades de hardware/software/rede necessárias para suportar o novo sistema.

✓ Desenvolver a estratégia de aquisição e instalação de hardware/software/rede, se necessário.

✓ Testar os requisitos detalhados de hardware/software/rede.

Os requisitos preliminares de hardware/software/rede do sistema são revistos à luz das informações obtidas até o momento. Agora a lista dos principais requisitos dessa área deve ser detalhada e completada na medida do possível. Alguns itens a serem cobertos são:

✓ A identificação de equipamentos e facilidades de hardware/software/rede que podem ser usadas para satisfazer os requisitos do novo sistema.

✓ A identificação de novos equipamentos e facilidades de hardware/software/rede que devem ser adquiridas para desenvolver e operar eficientemente o novo sistema em seu ambiente de produção.

O refinamento dos requisitos de hardware/software/rede exigirá um esforço mínimo se os novos requisitos de processamento e armazenamento do novo sistema puderem ser atendidos pela configuração de hardware/software rede existente. E, tal situação, a tarefa presente pode simplesmente identificar o conjunto adicional de terminais e impressoras necessário para suportar o novo sistema. Além disso, se a organização já tiver prescrito alguns padrões básicos para dar diretrizes ao pessoal de desenvolvimento durante e escolha de equipamento de hardware/software/rede, este processo será mais facilmente executado e minimizará o risco de adquirir equipamentos ou recursos inadequados.

Quando definir o equipamento e os recursos de hardware/software/rede, a equipe de desenvolvimento deve sempre examinar essas três entidades aparentemente distintas como sendo única, porque, em geral, elas são muito independentes. Na realidade, um dos aspectos mais cruciais do processo de tomada de decisões sobre a aquisição de equipamento de computação e recursos de software é o fator de compatibilidade. Por exemplo, as decisões sobre a aquisição de uma linha específica de equipamento de hardware de um dado fabricante, por padrão, podem ditar a escolha de alguns recursos de software muito específicos para manter um nível aceitável de confiabilidade e eficiência para o sistema em pauta.

Mesmo que diversos recursos de software possam rodar em diversos tipos de *Mainframes* ou microcomputadores, o mesmo software pode não oferecer necessariamente o mesmo nível de funcionalidade e eficiência quando trabalhar nesses diferentes ambientes computadorizados. Por último, o equipamento de computação e os recursos de software que serão adquiridos para o novo sistema devem ser completamente compatíveis com o ambiente automatizado existente. Eles

também devem ser compatíveis com qualquer futura aquisição de computador/software que possa ser considerada dentro dos próximos cinco anos.

13.2 Recursos de hardware/software.

Algumas categorias de equipamentos de hardware e recursos de software que devem ser descritas neste ponto incluem:

13.2.1 Hardware

- ✓ Processadores de grande porte.
- ✓ Microcomputadores.
- ✓ Unidades de fita.
- ✓ Unidades de disco.
- ✓ Plotadoras gráficas.
- ✓ Impressoras.
- ✓ Terminais de vídeo.
- ✓ Leitoras (óticas, código de barra, cartões).
- ✓ Dispositivos de armazenamento de arquivos.
- ✓ Máquinas automáticas de caixa.
- ✓ Leitoras de caracteres magnéticos (MICR).

As dimensões físicas de cada nova peça de hardware devem ser especificadas, juntamente com suas principais características.

13.2.2 Software

- ✓ Monitores *batch.*
- ✓ Software de sistema operacional.

- ✓ Rede de comunicações.
- ✓ Sistema de gerenciamento de bancos de dados.
- ✓ Utilitários gerais de sistema para cópia de segurança (*backup*)/recuperação (*recovery*), classificação/intercalação (*sort/merge*), descarga/recarga (*unload/reload*), conversão de arquivos/bancos de dados e assim por diante.
- ✓ Ferramentas estáticas e interativas de testes.
- ✓ Geradores de dados de teste.
- ✓ Geradores de código de aplicação.
- ✓ Otimizadores de código de aplicação.
- ✓ Linguagem de programação e compiladores.
- ✓ Recursos de controle de segurança de software.
- ✓ Ferramentas de depuração de programas.
- ✓ Monitor de teleprocessamento.
- ✓ Recursos de comunicações.
- ✓ Ferramentas de gerenciamento de projetos.
- ✓ Facilidades de métodos de acesso.
- ✓ Ferramentas de administração de bibliotecas de programas.
- ✓ Ferramentas automatizadas para suportar sessões interativas tipo JAD.
- ✓ Ferramentas de entrada de dados de uso geral.
- ✓ Recursos de geradores de consultas e relatórios.

A versão desejada de cada componente de software deve ser claramente descrita pela equipe de desenvolvimento.

Deve-se dar especial atenção à descrição das características físicas dos periféricos que serão utilizados pelos usuários como interface com o sistema. Na criação do modelo funcional de processo, os relatórios e telas do sistema foram prototipados pela equipe de desenvolvimento e usuários. Esse esforço foi obtido, concentrando, primariamente, a atenção do usuário sobre as características funcionais e de uso do sistema para manter a maior flexibilidade possível no projeto dessas interfaces. Agora, é o momento de atender a algumas questões relacionadas com as características físicas detalhadas do equipamento de interface homem-máquina.

Para terminais de vídeo, algumas das características que devem ser documentadas são:

- ✓ Quantidades de unidades necessárias por categoria de usuários.
- ✓ Tamanho das telas.
- ✓ Recursos de teclado desejados (por exemplo, teclado adaptado especificamente para funções especializadas, caracteres de outras línguas).
- ✓ Necessidade de telas coloridas.
- ✓ Quantidade de linhas e colunas da tela (24 ou 30 linhas, 80 ou 132 colunas).
- ✓ Recursos básicos de edição de campos (validação alfabética/numérica, conferência de faixa).
- ✓ Capacidades gráficas.
- ✓ Nível de resolução de tela (baixo, médio ou alto).
- ✓ Capacidade de formatação do terminal.

Além disso, os tipos de terminais selecionados podem ser influenciados por suas localizações, como um escritório ou uma linha de produção. Por exemplo, se as condições de trabalho nas fábricas forem extremas (calor, frio, poeira, vibração, atmosfera corrosiva), escolha terminais que possam resistir a elas.

Outro fator a ser considerado são as diferentes categorias de pessoas que utilizarão os terminais. Por exemplo, os terminais serão usados por pessoas que executam constantemente algumas funções muito especializadas ou por pessoas diferentes que devem executar diversas funções de caráter geral? O uso de terminais com finalidade especial pode ser muito vantajoso quando é necessário o máximo de velocidade e precisão, já que podem minimizar a necessidade de

digitar dados e, dessa forma, a ocorrência de erros. Por outro lado, terminais de uso geral podem aumentar a flexibilidade dos usuários, especialmente quando mudam freqüentemente de métodos de trabalho ou são chamados para executar diversas outras funções não especializadas. Todavia, pode ser vantajoso garantir algum grau de consistência ao longo de toda a aplicação quando se escolhem terminais ou impressoras.

Para terminais de impressão, algumas das informações que devem ser coletadas incluem:

- ✓ Nível desejado de ruído.
- ✓ Qualidade da impressão.
- ✓ Quantidade máxima de caracteres por linha.
- ✓ Suporte a caracteres especiais de impressão.
- ✓ Tipos de formulários que devem ser suportados e largura máxima do papel.
- ✓ Velocidade de impressão.
- ✓ Características especiais de impressão desejadas (formatação de microficha, identificação de código de barras).
- ✓ Quantidade máxima de cópias que podem ser produzidas com uma impressora de impacto.
- ✓ Quantidade de impressoras necessárias, com base na quantidade de usuários a serem atendidos e na carga de impressão estimada (tanto para impressoras locais como centralizadas).

13.3 Recursos de rede.

O tópico seguinte é o ambiente de comunicação de dados ou de rede. Um esboço preliminar, ilustrando as diferentes localizações geográficas onde os computadores e seus periféricos serão instalados, deve ser feito neste estágio. Ao mesmo tempo, diversas questões devem ser respondidas:

- ✓ As instalações atuais satisfazem os requisitos de hardware em termos de espaço, temperatura, umidade e energia?
- ✓ Que mudanças devem ser feitas nos diversos locais para acomodar os novos equipamentos de hardware?

✓ Qual será o impacto dessas mudanças sobre o ambiente atual?

Os requisitos básicos para conectar diversas peças de hardware em diversos locais diferentes também devem ser mostrados no fluxograma. Mas, antes disso, os seguintes pontos devem ser finalizados:

✓ Identificar os locais exatos onde os periféricos (terminais de vídeo/impressoras) e processadores (*Mainframes* ou micros) serão instalados. Cada usuário terá uma estação de trabalho dedicada ou os terminais ficarão em uma área comum? Enquanto a maioria dos usuários estiver ligada ao sistema em locais predeterminados, podem ocorrer situações específicas em que os usuários precisarão de acesso ao sistema quando estiverem em outro local, como vendedores ou representantes de marketing. Eles usarão terminais portáteis para se comunicar com o sistema? Os recursos atuais de comunicação podem suportar essas necessidades especiais do sistema?

✓ Estimar o número médio de horas por dia e dias por semana durante os quais cada combinação de conexões deve permanecer em operação.

✓ Determinar o volume de dados transmitidos em períodos de pico, requisitos de velocidade de resposta associados a cada principal categoria de transmissão e à quantidade máxima de usuários conectados em horas de pico.

✓ Identificar o nível de degradação que pode ser aceito pela aplicação em caso de sobrecarga. Definir os requisitos de *backup* de rede nos casos de falhas.

✓ Analisar as necessidades decorrentes da evolução da aplicação durante um futuro previsível ao mesmo tempo em que se avaliam as diferentes alternativas de redes de comunicação disponíveis (redes públicas de dados, redes públicas discadas, redes mistas, redes locais, redes de pacotes). Os custos associados a cada opção também devem ser levados em consideração.

A complexidade dos requisitos de rede determinará o nível de experiência necessário para desenvolver configurações ótimas de rede. Uma coisa é projetar uma rede local para acomodar dois ou três usuários de micros instalados no mesmo andar. Outra é projetar uma estratégia otimizada de rede necessária para acomodar uma aplicação complexa que troque informações entre diversos estados.

Dependendo do nível de experiência disponível na organização, pode ser mais prático contratar os serviços de uma empresa especializada em projetos de redes complexas, para se obter a melhor configuração adaptada e atender às necessidades de comunicação de sua empresa. Além disso, a tecnologia de rede, em geral, evolui constantemente ao longo tempo, e somente um grupo

interno dedicado em tempo integral ao projeto de grandes redes pode acompanhar essa rápida evolução. Isso não deve impedir que o pessoal de desenvolvimento de sistema se familiariza com os conceitos básicos relacionados com esse campo especializado de processamento de dados. Isso sensibilizará a equipe para a importância de obter as informações essenciais relacionadas à aplicação que são necessárias como entrada para o projeto de estruturas de rede de qualidade.

13.4 Estratégia de aquisição e instalação

Assim que os requisitos detalhados de hardware/software/rede de aplicação tiverem sido identificados com os usuários, deve ser desenvolvido um plano de ação para adquirir e instalar esses recursos. Esse plano deve atender aos seguintes itens:

✓ Interdependências que podem existir entre os componentes relacionados de hardware/software/rede.

✓ Seqüência em que eles devem ser adquiridos e instalados para cada módulo.

✓ Papéis e responsabilidades de cada grupo envolvido na aquisição e instalação desses equipamentos.

✓ Dependências existentes entre o módulo atual e outros módulos do sistema.

✓ Datas previstas para entrega dos equipamentos de computação e recursos de software e datas previstas para instalação.

Dependendo da importância do sistema para o bem-estar financeiro da organização, em alguns casos é preferível considerar alternativas adicionais para substituir itens de hardware, caso surjam problemas de última hora. Por exemplo, essa abordagem preventiva pode ser necessária para cobrir situações em que o equipamento não pode ser entregue pelo vendedor na data prevista ou quando o equipamento não funciona da forma esperada anteriormente.