

JAYLSON TEIXEIRA

MÉTODO PARA GERENCIAMENTO DE PEQUENOS PROJETOS DE SOFTWARE

Dissertação apresentada como requisito parcial à obtenção do grau de Mestre, Curso de Pós-Graduação em Informática, Setor de Ciências Exatas da Universidade Federal do Paraná.

Orientador: Prof. Dr. Martin A. Musicante
Co-orientador: Prof. MSc. Nelson Suga

**CURITIBA
2000**



Ministério da Educação
Universidade Federal do Paraná
Mestrado em Informática

PARECER

Nós, abaixo assinados, membros da Banca Examinadora da defesa de Dissertação de Mestrado em Informática do aluno *Jaylson Teixeira*, avaliamos o trabalho intitulado "*Método para Gerenciamento de Pequenos Projetos de Software*", cuja defesa foi realizada no dia 29 de novembro de 2000. Após a avaliação, decidimos pela aprovação do Candidato.

Curitiba, 29 de novembro de 2000.

Prof. Dr. Martin Alejandro Musicante
Presidente - DINF/UFPR

Prof. MSc. Nelson Suga
DINF/UFPR

Prof. Dr. Sílvio Romero de Lemos Meira
CIn/UFPE

Profª. Dra. Sílvia Regina Vergilio
DINF/UFPR

DEDICATÓRIA

Aos meus pais pelo exemplo e apoio na difícil empreitada
de escrever trabalhos científicos.

A minha esposa pelo apoio, paciência, dedicação e
companheirismo que foram fundamentais para
este trabalho, como tem sido para minha vida.

À minha avó Virgilina e meu filho Ícaro por me ensinarem
a viver melhor aceitando novas idéias.

A todos nós que trabalhamos para fazer um mundo um
pouco melhor.

AGRADECIMENTOS

Agradeço ao Professor **Nelson Suga** que contribuiu em muito para esse trabalho disponibilizando bibliografia e com comentários relevantes.

Agradeço ao Professor **Martin Alejandro Musicante** pela atenção e compreensão viabilizando esse trabalho apesar das muitas atribuições do orientador e do orientado.

Agradeço ao Professor **Uirapuru Florido** por ter acreditado no trabalho e ajudado a aplicar o método de gerenciamento de pequenos projetos de software com os seus alunos da Escola Técnica da Universidade Federal do Paraná. Sem esse apoio o estudo de casos não seria possível.

Agradeço ao senhor **José Simão de Paula Pinto** por tomar a frente do gerenciamento segundo o método 5E+3P na sua equipe de alunos da Escola Técnica da Universidade Federal do Paraná que estagiavam no Hospital das Clínicas de Curitiba.

Agradeço aos amigos **Luís Gonçales Bueno de Camargo e Patricia da Silva Cardoso**, professores de literatura da UFPR, pela revisão da escrita.

Agradeço aos alunos da Escola Técnica da Universidade Federal do Paraná que contribuíram para a realização do estudo de casos com os seus projetos. Obrigado **Cristiane Pereira de Andrade, Aldo Monteiro do Nascimento, Mauro Kaliu Souza, Fernando Ferraz, Flávio Luiz da Silva, Roberto Marini Steck, Fernando Augusto Starepravo, Ricardo Köhler Costa, Fábio Mascarello**.

SUMÁRIO

DEDICATÓRIA	II
AGRADECIMENTOS	III
RESUMO	VI
ABSTRACT	VII
1.INTRODUÇÃO	1
2.CRISE DO SOFTWARE	5
3.REVISÃO BIBLIOGRÁFICA	11
3.1 - MODELOS DE MELHORIAS DO PROCESSO DE SOFTWARE	11
3.1.1 – ISO 9000-3	12
3.1.2 - CAPABILITY MATURITY MODEL – CMM	14
3.1.3 – SPICE (SOFTWARE PROCESS IMPROVEMENT AND CAPABILITY DETERMINATION).	17
3.2 - CLASSIFICAÇÃO DE PROJETOS	21
3.2.1 - CLASSES DE INCERTEZAS TECNOLÓGICAS	23
3.2.2 - CLASSES DO ESCOPO DO SISTEMA	26
3.2.3 - PESQUISA DE SHENHAR [SHENHAR 1998]	30
3.2.4 - VARIÁVEIS DE GERENCIAMENTO.	33
3.3 - PLANEJAMENTO: ALCANCE, CRONOGRAMA E ESTIMATIVA DE CUSTOS	34

3.3.1 - DOCUMENTO DE REQUISITOS	35
3.3.2 – EDT: ESTRUTURA DE DIVISÃO DO TRABALHO	39
3.3.3 - CRONOGRAMA	42
3.3.4 - ESTIMATIVA DE CUSTOS	44
3.4 - GERENCIAMENTO DE RISCOS	45
3.4.1 - VISÃO DE GRADY BOOCH	45
3.4.2 - VISÃO DE MARTYN OULD	47
3.4.3 - GRÁFICO DE HUBBARD.	51
<u>4.MÉTODO 5E+3P.</u>	<u>52</u>
4.1.ANÁLISE DE REQUISITOS	57
4.2.EDT–ESTRUTURA DE DIVISÃO DO TRABALHO	59
4.3 MODELO DE PROCESSO EM V SIMPLIFICADO	61
4.4.RELATÓRIO DE CUSTOS	63
4.5.CRONOGRAMA	64
4.6.ANÁLISE DE RISCOS	65
4.7.CONCLUSÃO	65
<u>5.ESTUDO DE CASOS.</u>	<u>67</u>
5.1.PROJETOS	69
5.2.RESULTADOS	70
5.3 CONTRIBUIÇÃO AOS MODELOS DA QUALIDADE.	74
5.4.CONCLUSÃO	76
<u>6.CONCLUSÃO</u>	<u>78</u>
<u>REFERÊNCIA BIBLIOGRÁFICA</u>	<u>83</u>
<u>ANEXOS</u>	<u>87</u>

RESUMO

Projetos de software costumam ultrapassar o orçamento previsto, alongar-se nos prazos e apresentar funcionalidades que não satisfazem os clientes. Muitos desses casos poderiam ser evitados com o mínimo de planejamento e controle.

Este trabalho tem como finalidade definir um método para planejamento e controle de pequenos projetos de software. O planejamento e o controle são etapas fundamentais ao gerenciamento de projetos.

O método está baseado em cinco evidências e três princípios. Através das evidências o método gerencia as variáveis custo, prazo e funcionalidade. Os princípios servem de guia para minimizar riscos aos quais o projeto está sujeito.

O método 5E+3P foi definido a partir da bibliografia especializada e aplicado em três projetos, bem sucedidos, que serviram de base para um estudo de casos.

ABSTRACT

Software projects use to be over the previewed budget, to be longer in schedule and have functionalities that do not satisfy the customer. A minimum of planning and control could avoid these cases.

The goal of this work is to define a method to plan and control small software projects. Both planning and controlling are fundamental issues in project management.

The method is based on five evidences and three principles. Evidences manage the variables cost, schedule and functionality. Principles are used as a guide to minimize risks which the project is subjected.

The method proposed was defined from specialized literature and was applied to three projects, well succeeded, which serve as cases in point.

1. INTRODUÇÃO

Gerenciamento de Projetos de software é o tema escolhido. “Projeto é um empreendimento a ser realizado dentro de determinado esquema: projeto administrativo; projetos educacionais etc”[Ferreira 1986]. Existem muitos relatos na literatura de projetos de software nos quais os prazos são ultrapassados, os custos ficam acima do orçado e as funcionalidades do sistema não atendem totalmente às necessidades do usuário. Os pequenos projetos de software, embora com maiores possibilidades de sucesso, também estão sujeitos a essa realidade.

Sabendo disso, muitas iniciativas têm sido tomadas em diversas partes do mundo para enfrentar esse desafio. Sob o título geral de engenharia de software, várias técnicas, métodos, metodologias, hipóteses e estudos são feitos com a finalidade de se desenvolver software disciplinadamente, com planejamento e previsibilidade.

Mais recentemente, com as empresas se voltando à qualidade total, a área de software ganhou um programa de qualidade semelhante aos modelos aplicados no gerenciamento de

outras áreas do conhecimento, mas com um perfil próprio influenciado pela engenharia de software.

Depois de passar por vários projetos bem sucedidos e alguns mal sucedidos, como coordenador de projeto de software em variados campos de atividade, com equipes de uma a dez pessoas, o autor passa a acreditar que um pouco de organização e antecipação aos problemas são receitas simples que aumentam a probabilidade de sucesso. Esta percepção é intuitiva. E a intuição tem tido importância crescente na tomada de decisão pelos gerentes. [Sauter 1999]. Baseado nesta intuição, vinda da experiência profissional, identificada na literatura e confirmada pela prática, foi elaborado um guia com idéias simples que auxiliam o planejamento e o controle de pequenos projetos de software. O planejamento e o controle são etapas fundamentais ao gerenciamento de projetos [Koontz 1978] [Castor e Suga 1989]. O objetivo desse trabalho é estabelecer um método simples com base científica para o gerenciamento de pequenos projetos de software.

Seria esse método simples possível de ser derivado da literatura científica? Caso seja possível a existência de tal método, seria ele aplicável? Se aplicável, seria bem aceito pelos desenvolvedores? Este trabalho pretende demonstrar que se pode responder sim a essas perguntas.

O planejamento e o controle exigem uma certa disciplina, porém acredita-se que um pequeno projeto exija pouca burocracia para o gerenciamento de custos, prazos e funcionalidades. Controlando essas variáveis se aumentaria a possibilidade de sucesso do projeto. Essas impressões devem ser confirmadas por textos científicos e por teste de campo.

Um método fácil de seguir pode se tornar uma ferramenta muito útil para gerentes de pequenos projetos de software. Ele poderia dar previsibilidade ao gerente, facilitando o

relacionamento comercial com o seu cliente. A partir de um planejamento em que custos, prazos e funcionalidades estão relacionados, é possível negociar, por exemplo, as funcionalidades, adequando o projeto ao limite de tempo e orçamento do cliente.

Um método é um caminho bem definido para alcançar um objetivo. Por isso mesmo é menos abstrato que os programas de qualidade que sugerem os procedimentos a serem estabelecidos, mas deixa a definição do caminho em aberto.

A estrutura do trabalho é composta por 6 capítulos que são Introdução, Crise do Software, Revisão Bibliográfica, Método 5E+3P, Estudo de Caso e Conclusão.

Nesta **Introdução** define-se o tema da dissertação e o objetivo do trabalho. São levantadas hipóteses que devem ser confirmadas ou rejeitadas após o estudo. É apresentada uma justificativa da importância do tema. Finalmente uma introdução aos capítulos que o leitor irá passar.

No capítulo **Crise do Software** é feito um levantamento dos problemas encontrados nos projetos de software. Os pontos levantados como críticos serviram de base para um estudo mais aprofundado que compõe o capítulo **Revisão Bibliográfica**. Com base no estudo realizado no capítulo Revisão Bibliográfica, no capítulo **Método 5E+3P** é estabelecido um método para atuar nos pontos levantados no capítulo Crise do Software. O método 5E+3P proposto serve para gerenciamento de pequenos projetos de software. Esse processo, no qual um texto científico é feito a partir de outros textos científicos, correlacionando e fazendo inferências, é denominado pela metodologia científica como Método Dedutivo de Abordagem Científica.

No capítulo **Estudo de Casos** é feita uma prova de campo com o método 5E+3P proposto. Essa experiência foi realizada com alunos do último ano do ensino médio do curso de computação da Escola Técnica da Universidade Federal do Paraná. Nesse capítulo são descritos o procedimento, o material didático utilizado para exposição do método e os resultados obtidos. O estudo de casos é conhecido na metodologia científica como um Método de Procedimento Científico.

Na **Conclusão** é retomado o roteiro desse trabalho com as conclusões parciais de cada capítulo que servem de base para os capítulos seguintes. No final, tem-se um trabalho original, desenvolvido com base científica, onde se propõe um método. Além de proposto, o método foi aplicado e seus resultados documentados. Esse trabalho tem como produto um **Método para Gerenciamento de Pequenos Projetos de Software**.

2. CRISE DO SOFTWARE

Em junho de 1996 a revista *Fortune* publica um artigo intitulado “The Trouble With Software Is ... It Sucks” [Alsop 1996]. Nesse artigo, Stewart Alsop torna claro o seu descontentamento quanto à qualidade dos softwares. As reclamações apresentadas não são caso isolado. Essa deficiência na qualidade se deve ao fato de o software ser uma mercadoria abstrata, intangível, por isso mesmo difícil de quantificar e controlar.

Inúmeros são os relatos de insucesso de software. O caso do sistema de automação de bagagem do aeroporto de Denver é o mais citado da bibliografia [Rational 1997] [Diaz 1995] [Lyu 1996]. Esse projeto de software sofreu um atraso de 6 meses e consumiu 1,1 milhão de dólares por dia com juros e custos de operação.

Aqui estão outros casos citados:

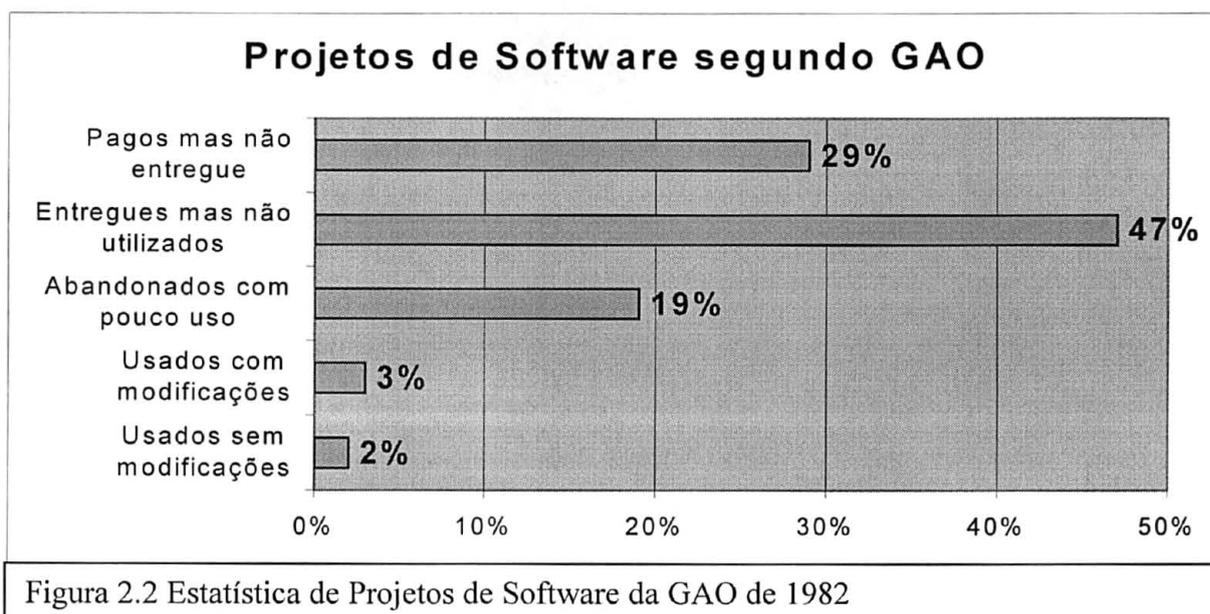
- O departamento de veículos motorizados da Califórnia gastou mais de US\$ 43 milhões em um sistema para combinar os sistemas estaduais de registro de veículos e de motorista. O sistema foi abandonado sem chegar a ser utilizado [Rational 1997].
- A American Airlines gastou US\$ 165 milhões em um projeto cancelado para interligar a reserva de vôos aos sistemas de reserva das redes Marriott, Hilton e Bugget [Rational 1997] [Diaz 1995].

- No DoD (Department of Defense) dos Estados Unidos, o primeiro voo do jato AFTI/F-16 sofreu um atraso de mais de um ano devido ao projeto de software e nenhuma nova característica inicialmente planejada foi disponibilizada [Lyu 1996].
- A comunidade científica sabia com maior antecedência da existência do buraco na camada de ozônio sobre a Antártida se o programa de análise de dados não suprimisse os dados considerados anômalos [Lyu 1996].
- Em 15 de julho de 1994 a bolsa de valores NASDAQ, que tem o seu pregão totalmente eletrônico, abriu seu pregão com 2 horas de atraso devido a um problema que comprometia o sistema. Por conta desse atraso muitos negócios deixaram de ser realizados. [Birman 1999] .
- Em Janeiro de 1990 um chaveador eletrônico no sistema de telefonia da AT&T falhou. As chamadas foram automaticamente desviadas para um outro chaveador que deu uma pane devido a um bug de software. As chamadas foram desviadas a um terceiro chaveador que apresentou o mesmo bug de software e assim por diante. Durante 9 horas 60 mil pessoas ficaram sem os serviços de telefonia e 70 mil ligações telefônicas deixaram de ser completadas. [Birman 1999].
- O equipamento de terapia por radiação Therac-25, de produção em massa, aparentava ser bastante seguro até que um erro de software vitimou vários pacientes entre 1985 e 1986 [Lyu 1996].
- Em 26 de outubro de 1992, o sistema do computador que controlava a logística do despacho de ambulância de Londres travou o computador logo após a sua instalação,

paralizando o maior serviço de ambulância do mundo com 5 mil requisições de transporte de pacientes por dia. [Lyu 1996].

Os casos expostos e afirmações de diversos autores levam a concluir que projetos de software são iniciados e nem sempre são concluídos, e que os concluídos muito provavelmente estouraram os limites do orçamento e prazos inicialmente previstos. Além disso, esses sistemas podem estar sendo usados parcialmente ou até mesmo não estarem sendo usados.

A quantificação desses fatos pode ser estimada por uma estatística da referência bibliográfica. A estatística é de 1982 realizada pelo GAO – General Accounting Office (GAO é equivalente ao nosso TCU – Tribunal de Contas da União). GAO é a unidade de auditoria do congresso americano (Congress investigative arm) (figura 2.2)



Porque estes fatos, aparentemente restritos à área de produção de software, são tão relevantes?

Se considerarmos que o software só afeta equipamentos onde ele está instalado, isso já seria problema suficiente para assustar a sociedade, mas, além dos equipamentos, serviços e informações gerenciais também estão cada vez mais dependentes do software. Hoje é inadmissível uma empresa de médio porte sem um sistema corporativo informatizado. A bolsa de NASDAQ foi a primeira a fechar negócios de compra e venda de ações por meio totalmente eletrônico, em 1971, deixando para trás a velha prática de compra e venda de ações no balcão das bolsas[Lyu 1996].

Microondas, compras e pagamentos via telefone e via internet, carros, trens, aviões com software embarcados: cada vez mais a sociedade coloca seus produtos e serviços dependentes de software. Esta dependência faz com que os problemas do software afetem a sociedade em

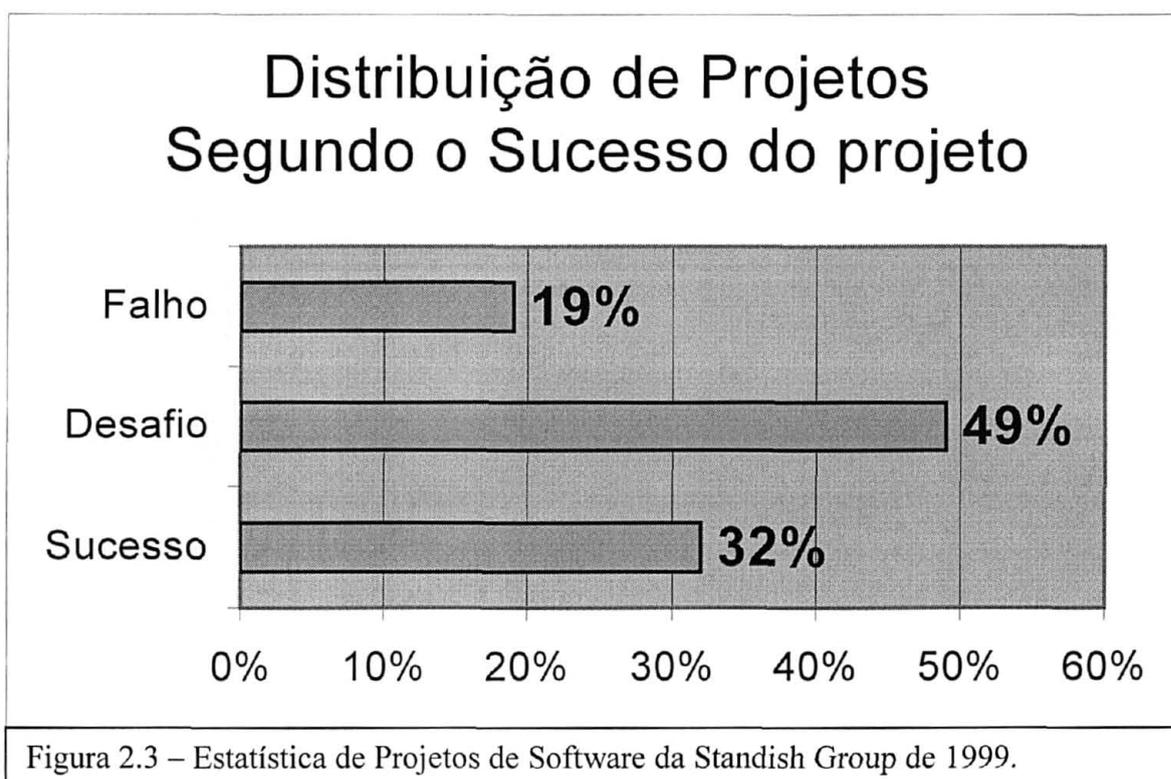
geral. Um exemplo, hoje bem popular, desse efeito é o bug do milênio, notado como Y2K (year two thousand).

Dados mais recentes foram publicados pelo Standish Group em setembro de 1999. Standish Group é um grupo internacional de pesquisa de mercado, consultor especializado em software de missão crítica e comércio eletrônico. O grupo foi fundado nos Estados Unidos em Dennis – MA, no ano de 1985.

Um relatório, com o título de *Perils in Gas & Electricity do Standish Group* trata de um caso no qual a desregulamentação dos requisitos para as companhias de gás e eletricidade modificou a maneira como essas companhias faziam negócios. Por conta dessa desregulamentação, as companhias tiveram que modificar seus sistemas de informação de missão crítica. Outras empresas além dos setores de gás e eletricidade também entraram na pesquisa, totalizando 23 mil projetos entre 1994 e 1998.

Os projetos foram classificados em três categorias que são **sucesso**, **desafio**, **falho**. Na categoria **sucesso**, estão aqueles projetos completados no prazo estabelecido e dentro do orçamento, com todas as funções inicialmente especificadas. Na categoria **desafio**, estão aqueles projetos completados e operacionalizados mas acima do orçamento ou acima do prazo estipulado, com poucas funções e características definidas inicialmente. Na categoria **falho**, estão os projetos cancelados em determinado ponto do ciclo de vida.

Para empresas com faturamento entre 100 e 200 milhões de dólares por ano encontrou-se a distribuição da Figura 2.3



Pode-se concluir que o software é vital para a sociedade moderna. Apesar de importante, encontram-se problemas de qualidade no produto entregue, além do desperdício de tempo e dinheiro em projetos atrasados, com custos acima do orçado e muitas vezes cancelados.

Para se conseguir projetos de software de sucesso, deve-se manter as variáveis custo, prazo e funcionalidade sob controle. Esse é o grande desafio da produção de software.

3.REVISÃO BIBLIOGRÁFICA

Neste capítulo serão expostos alguns tópicos importantes da bibliografia que estão embasando o trabalho. Serão apresentados modelos de melhoria de processos de software, uma classificação de projetos em geral, documentos que auxiliam o planejamento de projetos de software e visões de autores quanto ao controle de riscos em projetos de software

3.1 - MODELOS DE MELHORIAS DO PROCESSO DE SOFTWARE

Entre os modelos mais conhecidos de melhoria do processo de software estão o estabelecido pela norma ISO (International Organization for Standardization) 9000-3, o CMM (Capability Maturity Model) e o SPICE (Software Process Improvement and Capability Determination).

3.1.1 – ISO 9000-3

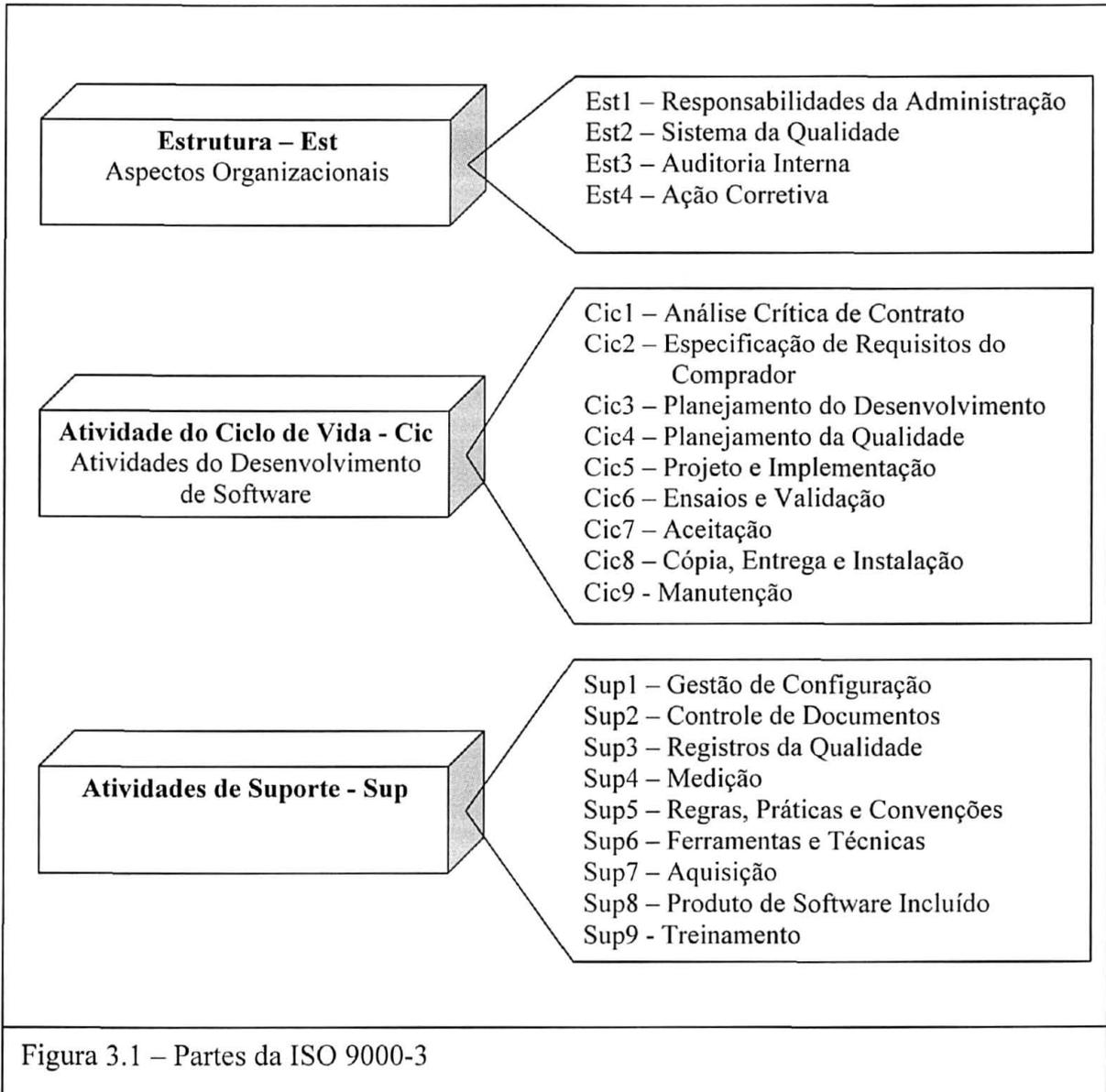
A série ISO 9000 é um conjunto de normas criado em 1987 para gerenciamento da qualidade elaborada pela ISO (International Organization for Standardization). É o primeiro modelo de gerência da qualidade a adquirir abrangência mundial. Essas normas são genéricas e podem ser aplicadas a produtos e serviços de qualquer natureza. A norma ISO 9001 trata de um modelo para garantia da qualidade em projeto, desenvolvimento, produção, instalação e assistência técnica. A norma ISO 9000-3 é um guia para adaptação da norma ISO 9001 para empresas que desenvolvem, fornecem e mantêm software. [Antonioni 1995]

A ISO 9000-3 é composta de 3 partes nas quais são sugeridos processos que devem ser descritos e auditados para garantir sua execução. As três partes são: estrutura, atividades do ciclo de vida e atividades de suporte. [Antonioni 1995]

A parte referente à estrutura descreve os aspectos organizacionais relativos ao sistema da qualidade. Detalham-se responsabilidades e ações que devem ser tomadas pelo fornecedor e pelo comprador.

A parte referente às atividades do ciclo de vida descreve as atividades do desenvolvimento de software. A norma define que a empresa deve adotar um ciclo de vida que contemple o planejamento e implementação das atividades de software. Estas atividades devem ser parte de uma das categorias definidas na Figura 3.1.

A parte das atividades de suporte se refere às atividades que apoiam as atividades do ciclo de vida. Essas atividades são catalogadas conforme a Figura 3.1.

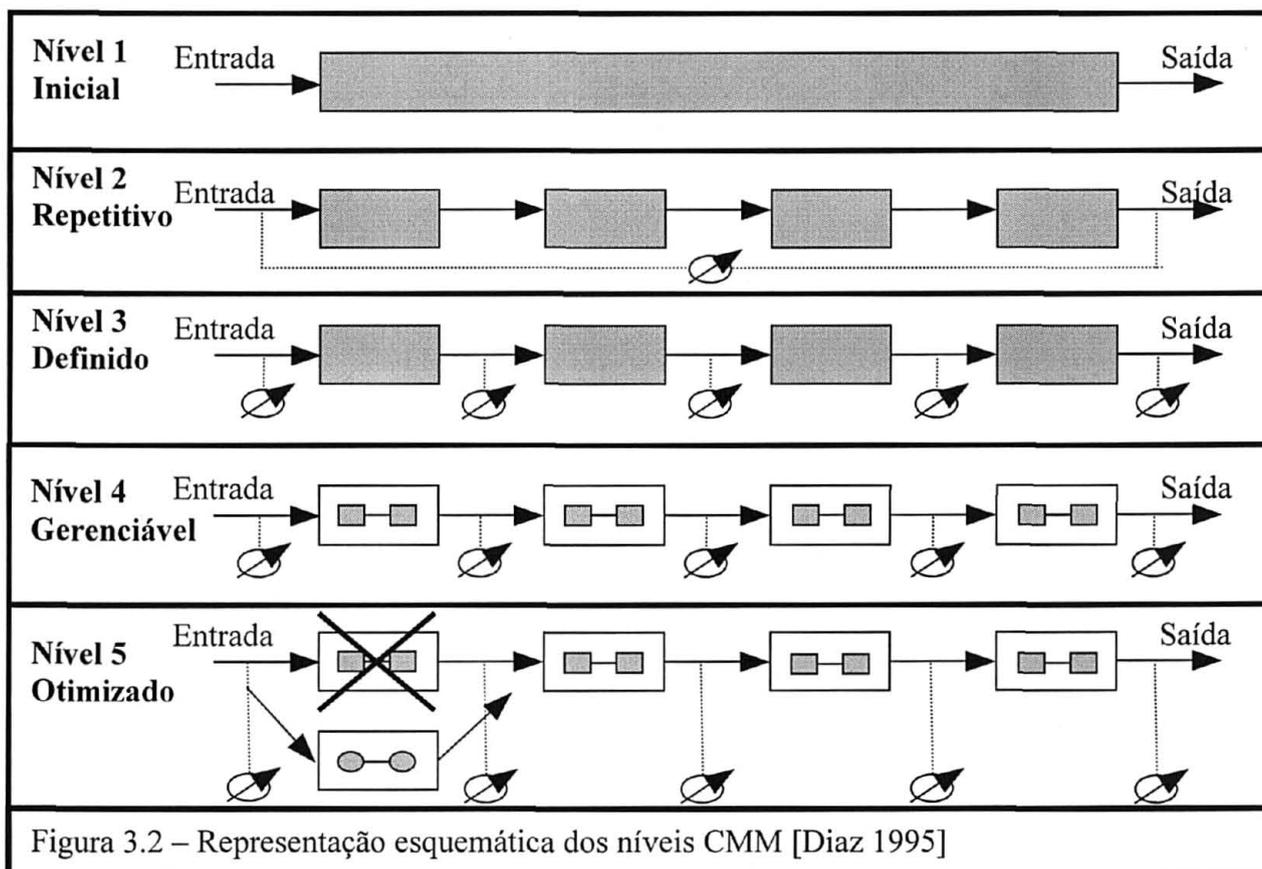


As normas ISO 9000 têm como objetivo definir e fazer cumprir o processo estabelecido. A norma recomenda ações preventivas e corretivas dos processos, mas não trata da qualidade dos processos de forma direta.

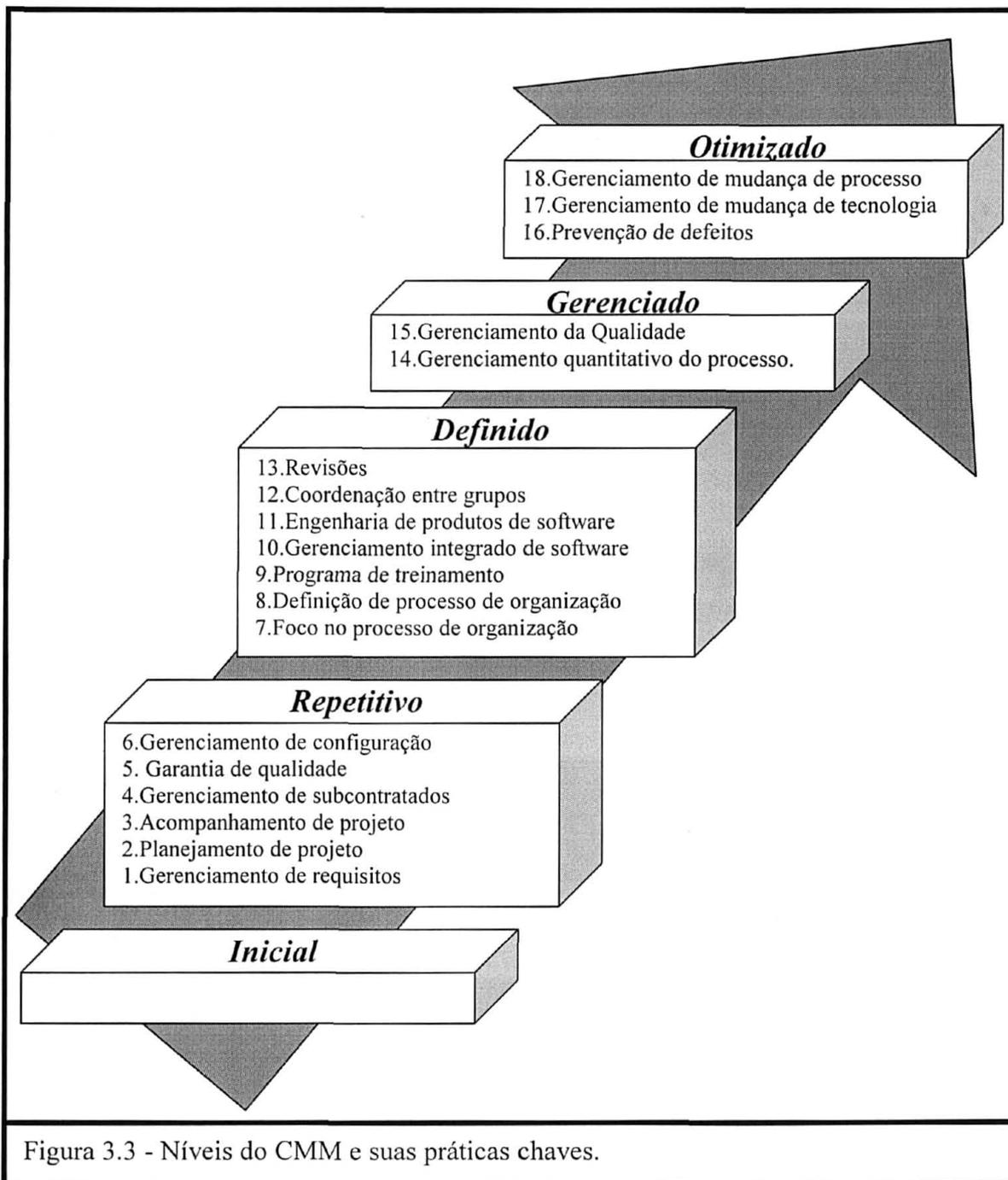
3.1.2 - Capability Maturity Model – CMM

É um modelo que indica a maturidade que a empresa tem em desenvolver software. Criado pelo SEI-CMU (Instituto de Engenharia de Software da Carnegie Mellon University) [SEI-CMU 1994]. O SEI foi fundado pelo DoD (Departamento de Defesa do governo americano) com a missão de promover a transferência tecnológica de software. CMM foi criado para melhorar a qualidade da produção de software dos Estados Unidos, em especial, das empresas que recebem fundos do DoD [SEI-CMU 1994]. Segundo este modelo existem cinco níveis de maturidade representados na Figura 3.2.

- **Nível 1 - Inicial:** O trabalho é feito *ad hoc* sem muitos processos definidos. É basicamente uma entrada com requisitos e uma saída com o software pronto. No meio, um processo nebuloso. O processo depende de esforços individuais e de heróis. Orçamento e cronograma não são confiáveis.
- **Nível 2 - Repetitivo:** O processo antes nebuloso é dividido em vários processos consecutivos. As experiências bem sucedidas são repetidas. Neste nível o sucesso do projeto depende muito do empenho individual do gerente e sua equipe.
- **Nível 3 - Definido:** O processo de desenvolvimento e o gerenciamento estão bem definidos e padronizados. Medidas do produto e do gerenciamento são possíveis.
- **Nível 4 - Gerenciável:** Medições detalhadas são possíveis. Tanto o produto quanto o gerenciamento são entendidos e controlados pelos números.
- **Nível 5 - Otimizado:** Processo de melhoria contínua, baseado nas medições, conduzido por novas idéias e tecnologias.



O que vai caracterizar a empresa ou a equipe como sendo de um determinado nível é o fato de o processo de software possuir determinadas práticas chaves. Essas práticas chaves são acumulativas, isto é, para obter o Nível 2 deve-se ter todas as práticas do Nível 1 e mais as práticas específicas que caracterizam o Nível 2. Veja Figura 3.3.



Se uma organização possui as práticas chaves de 1 a 6 ela tem o nível 2. Se uma outra empresa possui as práticas chaves de 1 a 13 exceto a prática 3, essa outra empresa está no nível 1. Embora esse exemplo seja teoricamente possível, na prática é difícil que isso aconteça. As práticas chaves são dispostas de tal modo que no nível 1 elas dependam do indivíduo, no nível 2 as práticas dependem de uma cultura implantada até a gerência, do nível 3 em diante as práticas dependem de uma cultura organizacional.

3.1.3 – SPICE (*Software Process Improvement and Capability dEtermination*).

Em 1993, a ISO (*International Organization for Standardization*) e a IEC (*International Electrotechnical Commission*) lançaram o projeto SPICE (*Software Process Improvement and Capability dEtermination*)[Rosa 1997][ISO 15504]. O projeto é um trabalho voluntário a partir de um compromisso internacional, coordenado por técnicos da Inglaterra, Canadá, Estados Unidos e Austrália, com a intenção de criar um padrão de avaliação do processo de software. Este padrão serviria para compradores, fornecedores e avaliadores. Os compradores poderiam avaliar os seus fornecedores. Os fornecedores poderiam se avaliar e melhorar seu processo de software baseando-se neste padrão e os avaliadores teriam um roteiro estruturado que conduziria a avaliação[Rosa 1997] [ISO 15504]. A versão completa da ISO 15504 está prevista para 2001. O projeto tinha três objetivos:

1. Acelerar o processo de definir uma norma de avaliação para software em uma versão preliminar (*draft*).
2. Comprometer-se com a realização de testes práticos (*trails*) para avaliação e revisão da norma.

3. Alertar ao mercado do aparecimento de um novo padrão.

A norma 15504 sofreu influência de outros modelos, tentando unificá-los. Entre eles tem-se Trillium, STD (Software Technology Diagnostic), Bootstrap, TQM (Total Quality Management), Malcolm Baldrige, PDCA (Plan, Do, Chek, Act) e principalmente CMM (Capability Maturity Model).

A norma prevê uma estrutura de avaliação de processos. Considerando um modelo em duas dimensões, pode-se listar os processos a serem avaliados na posição horizontal e a cada um dos processos avaliados atribuir um valor de capacidade do processo de 0 a 5 na dimensão vertical. Como mostra na Figura 3.4

Nível 5							●				
Nível 4				●			●		●		
Nível 3	●			●	●		●	●	●		
Nível 2	●		●	●	●		●	●	●	●	
Nível 1	●	●	●	●	●	●	●	●	●	●	●
	CUS.1	CUS.2	CUS.3	SUP.2	SUP.3	SUP.6	SUP.7	SUP.8	MAN.1	MAN.2	MAN.4
<p>Figura 3.4 – Avaliação do Desenvolvimento de Software em duas dimensões (Processo e Capacidade)</p>											

Os processos podem ser eleitos a partir do **modelo de referência**. O modelo de referência que consta na norma é composto por 29 processos distribuídos em 5 categorias (Tabela 3.1). Estes processos foram baseados e ampliados a partir da norma ISO 12207[ISO 15504].

Tabela 3.1 – Processos e Categoria de Processos do Modelo de Referência da ISO 15504.			
Categoria de Processos		Processos	
Id.	Título	Id.	Título
Processos do Ciclo de Vida Primário			
CUS	Cliente Fornecedor		
		CUS.1	Aquisição
		CUS.1.1	Preparação para Aquisição
		CUS.1.2	Seleção de Fornecedor
		CUS.1.3	Monitoração de Fornecedor
		CUS.1.4	Aceitação do Consumidor
		CUS.2	Fornecimento
		CUS.3	Elicitação Requerimentos
		CUS.4	Operação
		CUS.4.1	Uso Operacional
		CUS.4.2	Suporte ao Cliente
ENG	Engenharia		
		ENG.1	Desenvolvimento
		ENG.1.1	Análise de Projeto do Sistema de Requerimentos
		ENG.1.2	Análise de Requerimentos de Software
		ENG.1.3	Projeto de Software
		ENG.1.4	Construção do Software
		ENG.1.5	Integração do Software
		ENG.1.6	Teste de Software
		ENG.1.7	Integração e Teste de Software
		ENG.2	Manutenção de Software e Sistema
Suporte ao Ciclo de Vida			
SUP	Suporte		
		SUP.1	Documentação
		SUP.2	Gestão da Configuração
		SUP.3	Garantia da Qualidade
		SUP.4	Verificação
		SUP.5	Validação
		SUP.6	Revisão Conjunta
		SUP.7	Auditoria
		SUP.8	Resolução de Problemas
Ciclo de Vida Organizacional			
MAN	Gestão		
		MAN.1	Gestão
		MAN.2	Gestão de Projeto
		MAN.3	Gestão da Qualidade
		MAN.4	Gestão de Riscos
ORG	Organização		
		ORG.1	Alinhamento Organizacional
		ORG.2	Melhoria de Processo
		ORG.2.1	Estabelecimento de Processo
		ORG.2.2	Medição de Processo
		ORG.2.3	Melhoria de Processo
		ORG.3	Gestão de Recursos Humanos
		ORG.4	Infraestrutura
		ORG.5	Medição
		ORG.6	Reuso

Na Tabela 3.2 estão os cinco níveis de capacitação dos processos. O Nível 0 (zero) não consta da tabela e é considerado como um processo que não está implementado e falha na tentativa de atender seus objetivos.

Tabela 3.2 – Níveis de Capacitação de Processos			
Nível	Processo	Idéia	Descrição
Nível 1	Executado	Nós fizemos, aqui estão os produtos, mas não pergunte como.	O processo implementado atinge o seu objetivo definido.
Nível 2	Gerenciado	Nós podemos mostrar que estamos no prazo na maioria das vezes e que o produto atende os requisitos.	O processo executado entrega produtos de trabalho, de definida qualidade, dentro do cronograma e com recursos definidos.
Nível 3	Estabelecido	Você gostaria de ver nossos procedimentos?	O processo gerenciado é executado usando um processo definido, baseado em bons princípios de engenharia de software.
Nível 4	Previsível	Aqui estão nossas medidas, mostrando quão bem estamos fazendo.	O processo estabelecido é executado consistentemente dentro dos limites definidos de controle para atingir seus objetivos.
Nível 5	Otimizando	Aqui estão as melhorias que fizemos e aqui estão as suas medidas.	O processo previsível é otimizado o seu desempenho para atender as necessidades de negócio atuais e futuras e atinge repetibilidade em atender seus objetivos definidos de negócios.

O modelo SPICE é dividido em 9 partes que são:

Tabela 3.3 – Partes da Norma 15.504	
Parte 1	Conceitos e Guia Introductório (informativo)
Parte 2	Modelo Idealizado de Processo (normativo)
Parte 3	Classificação dos Processos (normativo)
Parte 4	Guia de Orientação para Condução da Avaliação (informativo)
Parte 5	Diretrizes para Elaboração de um Instrumento de Avaliação (informativo)
Parte 6	Qualificação e Treinamento de Avaliadores (informativo)
Parte 7	Guia de Orientação para Melhoria de Processos (informativo)
Parte 8	Guia de Orientação para Determinação da Capacidade de Processo (informativo)
Parte 9	Vocabulário (informativo)

3.2 - CLASSIFICAÇÃO DE PROJETOS

O termo projeto não é exclusivo da área de engenharia de software. A palavra projeto é utilizada em todas as atividades humanas. Esses projetos podem variar em tamanho, tempo utilizado, nível de automação, necessidade do usuário etc. Porém, missões tão diferentes quanto a reforma de uma casa e o lançamento de um novo foguete precisam de uma gerência de projeto.

Aaron J. Shenhar no seu artigo *From Theory to Practice: Toward a Typology of Project-Management Styles* [Shenhar 1998] propõe um modelo de classificação de projetos, em dois eixos, para verificar as diferentes abordagens de gerenciamento. Esses eixos são: Incerteza Tecnológica e Escopo do Sistema. Veja Figura 3.5.

Incerteza tecnológica representa o grau de tecnologia utilizada no projeto. Quanto mais se aproxima do estado da arte, mais incerteza ter-se-á inicialmente quanto a custos, prazos e funcionalidades do sistema. Shenhar propõe 4 classes de incerteza que são: Baixa Tecnologia, Média Tecnologia, Alta Tecnologia e Altíssima Tecnologia.

Segundo o dicionário Aurélio [Ferreira 1986], o sistema é composto de partes e elementos de um todo, coordenados entre si, que funcionam como uma estrutura organizada. Esta definição comporta diferentes sistemas, como sistema penitenciário ou sistema de refrigeração. Uma das partes do sistema pode ser um outro sistema. Baseado nessa hierarquia

de sistema e subsistema é que Shenhar classifica o escopo do sistema. As classes de projeto quanto ao escopo do sistema são: Montagem, Sistema, Matriz.

Escopo do Sistema ↑	<i>Matriz</i> (3)				
	<i>Sistema</i> (2)				
	<i>Montagem</i> (1)				
		<i>Baixa Tecnologia</i> (A)	<i>Média Tecnologia</i> (B)	<i>Alta Tecnologia</i> (C)	<i>Altíssima Tecnologia</i> (D)

Figura 3.5 - Classificação Bidimensional de Shenhar.

3.2.1 - Classes de Incertezas Tecnológicas

As classes de Shenhar para classificar um projeto quanto a sua incerteza tecnológica são: baixa tecnologia, média tecnologia, alta tecnologia e altíssima tecnologia.

Projetos de baixa tecnologia (Tipo A) utilizam tecnologia bem conhecida. Podem ser grandes projetos mas possuem um baixíssimo grau de incerteza. Tipicamente, edificações, construções de estradas e projetos de utilidade pública se encaixam nesta classificação. Nesta classe os projetos não requerem um trabalho de desenvolvimento. Geralmente o planejamento da arquitetura, esquemas e recursos são estabelecidos por um grupo e implementado por outro, em um único ciclo de projeto. O planejamento é feito, revisado e congelado. Como não existem novas tecnologias envolvidas, a margem de lucro é baixa e o planejado tem de ser cumprido rigorosamente.

A comunicação entre o gerente e a equipe é conduzida por canais formais, isto é, documentos, formulários e reuniões. A frequência das reuniões é baixa, uma vez a cada mês ou a cada dois meses.

O gerente de projetos do tipo A é firme, rígido e formal. O gerente é julgado pela execução do projeto no prazo, dentro do orçamento e de acordo com o planejado.

Projetos de média tecnologia (Tipo B) são os mais utilizados. Tem-se pelo menos um elemento com uma nova tecnologia, mas sempre numa proporção menor que 50% do projeto total. Essa nova tecnologia é incorporada geralmente para obter vantagem competitiva em relação ao competidor. A inovação é incremental. É um novo módulo em uma indústria de

automóvel ou eletrodomésticos, por exemplo. Uma melhoria ou derivação de um produto já existente seria outro exemplo.

O planejamento do projeto é congelado no primeiro ou segundo quarto de tempo do período de duração do projeto, após um ou dois ciclos de planejamento.

A comunicação entre os membros da equipe de gerenciamento é mais freqüente que nos projetos do tipo A. Reuniões acontecem em um período de uma semana ou 15 dias. O gerente de projeto do tipo B costuma evitar mudanças desnecessárias e está sempre preocupado com o excesso de gastos. O estilo de gerência é considerado moderadamente firme.

Projetos de alta tecnologia (Tipo C) são tipicamente utilizados em um novo produto ou mesmo uma nova linha de produto. No projeto, 50 % ou mais da tecnologia utilizada é nova. Os primeiros videocassetes desenvolvidos pela Sony e a JVC em meados dos anos 70 são exemplos de projetos do tipo C. Outro exemplo clássico é a criação pela Apple da linha Macintosh, com uma tecnologia pouco conhecida na época, explorando o conceito desenvolvido pela Xerox de um computador visual baseado em ícones e mouse.

Projetos do tipo C possuem longos períodos de desenvolvimento, testes e replanejamento. Requerem 2 ou 3 ciclos de planejamento. O planejamento só é congelado durante a ocorrência do segundo ou terceiro quarto do projeto.

A participação do cliente é sempre benéfica ao projeto, mas em projetos de alta tecnologia é muito intensa, chegando a ser mais uma cabeça pensante na realização do projeto. Shenhar conta dois casos onde o cliente abre mão da especificação inicial ao perceber as dificuldades e custos da especificação original.

A comunicação no projeto tipo C é muito intensa e feita de maneira formal e informal. A comunicação inclui relatórios, minutas e memorandos. A intensa comunicação oral parte do princípio que os detalhes mudam tão rapidamente que os meios formais não conseguem acompanhá-los em tempo real. A gerência procura encorajar interações adicionais e discussões informais. A atmosfera é de uma comunicação aberta. Os profissionais desses projetos têm uma educação elevada e alta qualificação.

O gerente tem comprovada experiência tecnológica. A escolha do gerente é baseada na combinação das habilidades administrativa e técnicas. O estilo de gerenciamento é baseado na flexibilidade, porém a uma certa altura do projeto se faz necessário o congelamento das especificações e a rejeição de novas mudanças. O estilo passa a ser mais rígido. Pode-se definir o estilo como sendo moderadamente flexível.

Projetos de altíssima tecnologia (Tipo D) se caracterizam por utilizar uma tecnologia emergente ou simplesmente não existente no início do projeto. Tem uma forte ênfase em pesquisa e desenvolvimento. Testes e escolhas entre alternativas tecnológicas são constantes. Projetos de altíssima tecnologia são raros. Geralmente são conduzidos por grandes organizações ou agências governamentais. Possuem grandes riscos e oportunidades competitivas. O projeto Apollo de pouso na lua é um exemplo desta categoria. Outro exemplo é o telescópio espacial Hubble. Esses projetos, embora tivessem uma missão clara, a tecnologia necessária para seu desenvolvimento não estava totalmente disponível no início do projeto. O ambiente é de grande incerteza tecnológica e busca constante da resolução dessa incerteza. Recursos adicionais e atenção gerencial extra são necessários para resolver a nova tecnologia, além do planejamento construção e testes do produto.

Para resolver incertezas lança-se mão de um projeto intermediário. Cria-se um modelo (uma maquete) para provar a validade dos novos conceitos que possibilite fazer testes tendo a nova tecnologia como objeto de estudo.

A comunicação é intensa. Mudanças são constantes e existe muita troca de informações sob a forma de relato de problemas. A procura dos problemas a serem solucionados é uma questão importante. Cada equipe compartilha informações sem esperar reuniões ou reportes formais.

O gerente de projetos é um bem conceituado líder técnico, geralmente PhD em tecnologia. A concorrência e seleção para todas as posições são extremamente elevadas. Acredita-se que uma equipe de elite faz a diferença e cria o futuro. Cada líder de projeto tem um alto grau de liberdade para testar novos conceitos e seguir em direção ao estado da arte em seu campo de atuação. Muitas idéias são testadas e muitas mudanças e melhoras são feitas. O estilo do gerente é altamente flexível.

3.2.2 - Classes do Escopo do sistema

O escopo do sistema é dividido em Projeto de Montagem, Projeto de Sistema e Projeto Matricial, com base na posição hierárquica na divisão sucessiva entre sistema e subsistema.

Projeto de Montagem (Tipo 1) é caracterizado por um único componente ou montagem de componentes em uma única unidade. A função desta unidade pode ser bem definida em um grande sistema, sendo assim um subsistema ou um produto independente, de uma única função. Um receptor de radar, um direcionador de míssil, uma unidade de controle

ou um driver de disco rígido em um computador são exemplos de subsistemas de função bem definida. Um *compact disc player*, um televisor e uma máquina de lavar roupa são exemplos de um produto independente. Esses projetos costumam ser tocados por uma única organização, muitas vezes por um único grupo. O projeto do vídeo cassete desenvolvido pela Sony e pela JVC é um exemplo de projeto de montagem de alta tecnologia.

A atmosfera é informal com fácil acesso a todos do grupo. O planejamento é feito manualmente ou por pequenos pacotes de software. Existem poucas atividades a serem executadas, geralmente da ordem de algumas centenas. Faz-se um diagrama de árvore do produto e em seguida uma divisão por módulos e um cronograma (diagrama de Gantt ou PERT). O controle é simples através do orçamento e pontos de verificação. O diagrama de árvore é denominado em inglês por WKS – Work Breakdown Structure, que traduz-se por EDT – Estrutura de Divisão do Trabalho. Essa técnica é muito utilizada nos projetos em geral. [Shenhar 1990] [Ould 1990].

A comunicação é informal. As reuniões diárias e informais nem sempre têm o seu conteúdo registrado. O direcionamento do trabalho é igualmente informal. A equipe tem de superar a inércia da informalidade registrando as principais decisões e procedimentos mais importantes.

A informalidade do gerente contribui para o sucesso técnico. Às vezes essa mesma informalidade prejudica aspectos gerenciais gerando desorganização e gastos desnecessários.

Projeto de sistema (Tipo 2) é caracterizado pela existência de elementos interativos que funcionam juntos como um produto. Diferentemente dos projetos de montagem, os projetos de sistema possuem várias funções para atingir uma necessidade operacional. Componentes e subsistemas atuam juntos para cumprir a missão do sistema. Radares,

computadores, mísseis, projetos de comunicação, aeronaves, embarcações e automóveis são exemplos de sistemas.

A completude do projeto está vinculada a uma série de atividades de suporte como treinamento, equipamentos de teste, manutenção de equipamentos, suporte logístico, partes sobressalentes e documentação.

A gerência é regulada por um contrato principal que regulamenta o desempenho e a qualidade do produto, prazo e orçamento. O esforço da realização do projeto é dividido entre contratos secundários que podem ser internos ou externos à organização.

O projeto do Macintosh é um exemplo de projeto de sistema de alta tecnologia, enquanto o telescópio espacial Hubble é um exemplo de projeto de sistema de altíssima tecnologia.

A divisão do projeto em subprojetos requer um esforço extra do gerente. Os problemas mais comuns são os de integração. Partes que funcionam em separado nem sempre funcionam juntas, principalmente em projetos de média, alta e altíssima tecnologia. Problemas de interfaces, desperdício de energia e lacunas requerem montagens, testes e decisões conciliatórias. Vários ciclos são necessários.

Os gerentes costumam reclamar da falta de ferramentas para suas necessidades administrativas. Há necessidade de burocratizar. Sistemas de gerenciamento incluem procedimentos, documentos, ferramentas gerenciais, reuniões, revisões e estrutura organizacional específica. Para utilizar ferramentas já existentes, adotam-se pacotes de software com os níveis de controle exigidos. O número de atividades na rede de planejamento chega a centenas e às vezes a milhares. Os gerentes devem conciliar a complexidade do projeto à complexidade do software de gerenciamento. A complexidade do software pode se

tornar fonte de atrasos e reclamações. O estilo das relações é formal entre contrato principal e subcontratos.

O projeto matricial (Tipo 3) é uma coleção de sistemas funcionando juntos para um propósito comum. Os sistemas da matriz costumam estar geograficamente espalhados. A matriz é composta por sistemas, sendo um supersistema. Por exemplo, o Sistema de Defesa Aérea Nacional é um supersistema que consiste de radares de alerta, centro de comando e controle, naves de combate aéreo e mísseis terra-ar. Outro exemplo é o sistema de transporte de uma grande cidade.

Matrizes são sistemas muito grandes construídos gradualmente, adicionando novos sistemas. Gerenciar um projeto matricial é gerenciar diferentes projetos de diferentes sistemas. A organização que leva em frente um projeto matricial serve como guarda-chuva central, coordenando ações de outras organizações.

O Programa Mestre da Delegacia de Trânsito da cidade de Nova York é um exemplo de um projeto matricial de baixa tecnologia, e possuía mais de 350 projetos para construção de vias, estações e pontes. A Iniciativa de Defesa Estratégica dos Estados Unidos, projeto conhecido como Guerra nas Estrelas, é um exemplo de projeto matricial de altíssima tecnologia.

Projetos matriciais não possuem os níveis de preocupação com integração existente em projetos sistêmicos. Uma matriz não é um único produto entregue de uma só vez ao usuário. As partes são finalizadas separadamente e entregues progressivamente em diferentes lugares, em instantes diferentes.

O guarda-chuva que resguarda a integridade da matriz é relativamente pequeno. Geralmente a preocupação principal é por finanças, fluxo de caixa e problemas de controle. Aspectos técnicos são deixados para os subprojetos.

Extensas subcontratações e dispersão dos sistemas levam o gerenciamento a ser altamente formal, com ênfase em aspectos legais dos vários contratos. Ferramentas de planejamento são menos relevantes. Cada projeto encontra uma maneira própria para coordenar contratos e controlar projetos. O número de atividades envolvidas é da ordem de dezenas de milhares. O estilo de gerenciamento é remoto e altamente formal.

3.2.3 - Pesquisa de Shenhar [Shenhar 1998]

Escopo do Sistema ↑	Matriz (3)	AYL	BIS TEV		
	Sistema (2)	SPN SWB JHH	NVR , ADM FBL , VHL	RBA , GWI PAL , BAT TSM , ILL	TMD COR
	Montagem (1)		LBD MDL GON	TAD PSA MXE	ABR
		Baixa Tecnologia (A)	Média Tecnologia (B)	Alta Tecnologia (C)	Altíssima Tecnologia (D)

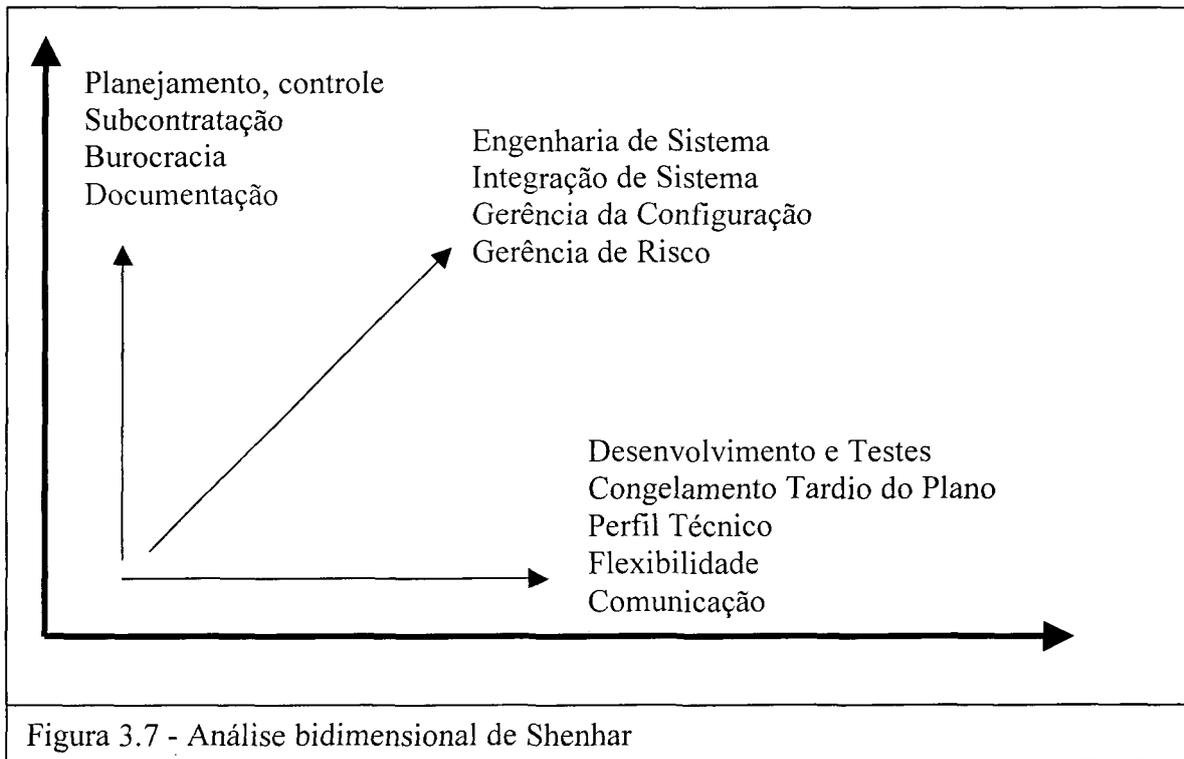
Figura 3.6 - Mapa dos Projetos Analisados por Shenhar.

Os projetos analisados por Shenhar estão na tabela 3.4 e na figura 3.6.

Tabela 3.4 – Projetos analisados por Shenhar			
Projeto	Sigla	Classificação	Descrição
1	SPN	A2	SPN foi o desenho e a construção de uma piscina que se tornou parte dos recursos de treinamento de tropas militares.
2	SWB	A2	SWB refere-se a um novo prédio para o departamento de ciências sociais de uma universidade.
3	JHH	A2	JHH refere-se à construção de um novo escritório principal regional para uma grande companhia.
4	LBD	B1	LBD foi um projeto de desenvolvimento interno de uma nova bateria de lítio para um propósito especial.
5	MDL	B1	MDL refere-se ao desenho e manufatura de uma cabine de proteção contra uma peça pesada de um equipamento.
6	GON	B1	GON foi um projeto interno de um sensor ótico para um sistema de extinção de incêndio para ser instalado em um veículo especial.
7	NVR	B2	NVR foi um projeto financiado por uma companhia de investimentos para um sistema de supervisão a ser instalado em uma plataforma militar.
8	AYL	A3	AYL refere-se à construção e melhoria no sistema de rodovias para uma grande área urbana.
9	ADM	B2	ADM refere-se ao desenvolvimento e instalação de um sistema para coleta e análise de dados de um problema de desempenho e confiabilidade em um grande sistema militar distribuído.
10	FBL	B2	FBL foi um projeto para desenhar, construir e instalar uma fábrica de produção em massa de dispositivos semicondutores.
11	VHL	B2	VHL foi um projeto de atualização e melhoria de uma aeronave militar.
12	BIS	B3	BIS foi um programa que envolve melhoria, inspeção e reorganização de uma esquadra aérea incluindo aeronave, sistema de armas e um sistema de suporte aerotransportado.
13	TEV	B3	TEV era um programa para desenhar, desenvolver,

			construir e instalar uma rede de comunicação geograficamente espalhada.
14	TAD	C1	TAD foi um projeto de auto-investimento para desenvolver um pacote de software para transformar uma linguagem em outra.
15	PSA	C1	PSA era uma subcontratação para desenvolver uma coleção de subsistemas de suprimento de energia com um requerimento específico para servir um grande sistema eletrônico.
16	MKR	C1	MKR foi o desenvolvimento de um subsistema sensor que era parte de um esforço de desenvolvimento de uma arma avançada.
17	MXE	C1	MXE refere-se ao desenvolvimento de uma unidade de multiplexação usada em um cabo submarino transatlântico de transmissão telefônica.
18	RBA	C2	RBA era um projeto de desenvolvimento de um novo sistema de radar.
19	GWI	C2	GWI era uma melhoria e atualização feita para um sistema de armas existentes para uso da marinha.
20	PAL	C2	PAL era o desenvolvimento de um sistema de guerra eletrônico.
21	BAT	C2	BAT era o desenvolvimento de um sistema de controle de fogo para uso em uma plataforma militar.
22	TSM	C2	TSM era um projeto de investimento próprio para um sistema de visão e alvo, envolvendo tecnologia a laser, um sistema ótico, um sistema estabilizador e mostradores.
23	ILL	C2	ILL era um programa para desenvolver uma nova plataforma aerotransportada incluindo sua estrutura, aerodinâmica, controles e muitos outros subsistemas.
24	ABR	D1	ABR era um projeto para desenvolver um novo módulo eletrônico e computacional que se tornaria parte de um grande sistema.
25	TMD	D2	TMD era um programa para desenvolver um novo sistema de armas para ser utilizada pela Força Aérea.
26	COR	D2	COR refere-se a um sistema eletrônico totalmente novo que teria de funcionar em um ambiente de comunicação desconhecido com uma inamistosa e larga banda de frequência do espectro eletromagnético.

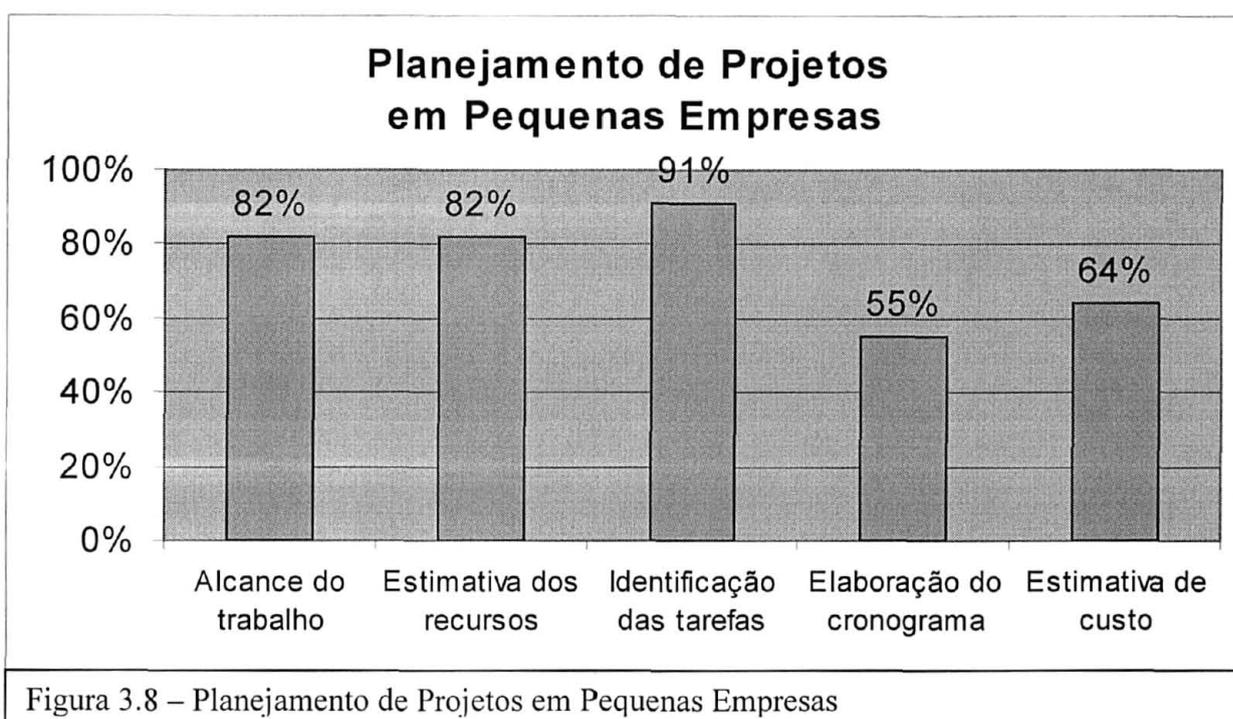
3.2.4 - Variáveis de Gerenciamento.



Shernhar tomou incerteza tecnológica e escopo do sistema como variáveis independentes na análise administrativa de um projeto. A partir desse modelo bidimensional ele fez análises quantitativas e qualitativas de algumas variáveis de administração. Shenhar notou que algumas dessas variáveis, como planejamento, controle, subcontratação, burocracia e documentação, variam apenas com a dimensão do escopo do sistema. Variáveis como desenvolvimento e testes, congelamento tardio do plano do projeto, perfil técnico, flexibilidade e comunicação variam apenas com a dimensão da incerteza tecnológica. Variáveis como engenharia de sistema, integração de sistema, gerência da configuração e gerência de risco variam com ambas as dimensões.

3.3 - PLANEJAMENTO: ALCANCE, CRONOGRAMA E ESTIMATIVA DE CUSTOS

Paola G. Rosa [Rosa 1997], em sua dissertação de mestrado, analisa pequenas empresas na região de São Carlos, no interior do estado de São Paulo. Para verificar o que era feito no planejamento do projeto de software, ela analisa se as etapas da fase de planejamento, sugeridas por Pressman, estão sendo realizadas. Obteve-se a estatística apresentada no figura 3.8.



O alcance do trabalho é o resumo dos requisitos na fase de análise, o alcance funcional. Estimativa dos recursos se refere aos recursos humanos, hardware e software necessários ao projeto. Identificação das tarefas refere-se às atividades a serem realizadas no projeto. Elaboração do cronograma refere-se às tarefas associadas a pessoas e datas. Estimativa de custos é o resultado da estimativa de recursos com associação de valores.

Ao realizar estas etapas quatro documentos seriam suficientes como evidência dessas etapas: **Documento de requisitos**, que evidenciaria a etapa de alcance do trabalho; a **EDT-Estrutura de Divisão do Trabalho** (*WBS – Work Breakdown Structure*) que evidenciaria a etapa de identificação das tarefas; o **cronograma**, que evidenciaria a etapa de elaboração de cronograma, e **relatório de custos**, que evidenciaria as etapas de estimativa de recursos e estimativa de custos. Mas como se produzem estes documentos?

No livro de engenharia de software de Sommerville & Duffie [Sommerville 1990], encontram-se sugestões de como estes cinco documentos devem ser gerados.

3.3.1 - Documento de Requisitos

No documento de requisitos, Sommerville [Sommerville 1990] recomenda a existência de uma definição de requisitos e uma especificação de requisitos.

Definição de requisitos é um texto descritivo auxiliado por diagramas que serve para especificar os serviços esperados do sistema a ser desenvolvido e as condições sob as quais será operado. A linguagem utilizada é familiar ao usuário. Deve refletir o que o usuário deseja do sistema. Sommerville cita como exemplo a definição de requisito:

- *Requisito 1) O software deve prover meios para representação e acesso de arquivos externos criados por outras ferramentas.*

Especificação de Requisitos é um documento estruturado em que constam os detalhes dos serviços do sistema. Também é chamado de especificação funcional. Serve como um

contrato entre o desenvolvedor e o comprador do projeto. Sommerville cita como exemplo as seguintes especificações de requisito:

- *Requisito 1.1) O usuário terá facilidades para definir o tipo de arquivo externo.*
- *Requisito 1.2) Cada tipo arquivo externo estará associado a uma ferramenta para manusear o arquivo.*
- *Requisito 1.3) Cada tipo de arquivo externo será representado por um ícone específico na tela do usuário.*
- *Requisito 1.4) Facilidades devem ser disponibilizadas para que o usuário associe um ícone ao tipo de arquivo externo.*
- *Requisito 1.5) Quando o usuário seleciona um ícone de arquivo externo na sua tela, a ferramenta associada àquele tipo de arquivo é utilizada para abrir o arquivo externo associado ao ícone.*

Em um projeto pequeno, a definição de requisitos e a especificação de requisitos podem estar em um único documento, chamado documento de requisitos. O documento de requisitos não é um projeto (*design*) do software, por isso não deve especificar **como** deve ser feito o software mas sim **o que** o software tem de fazer. Segundo Sommerville, Heninger em 1980 sugere seis recomendações para o documento de requisitos:

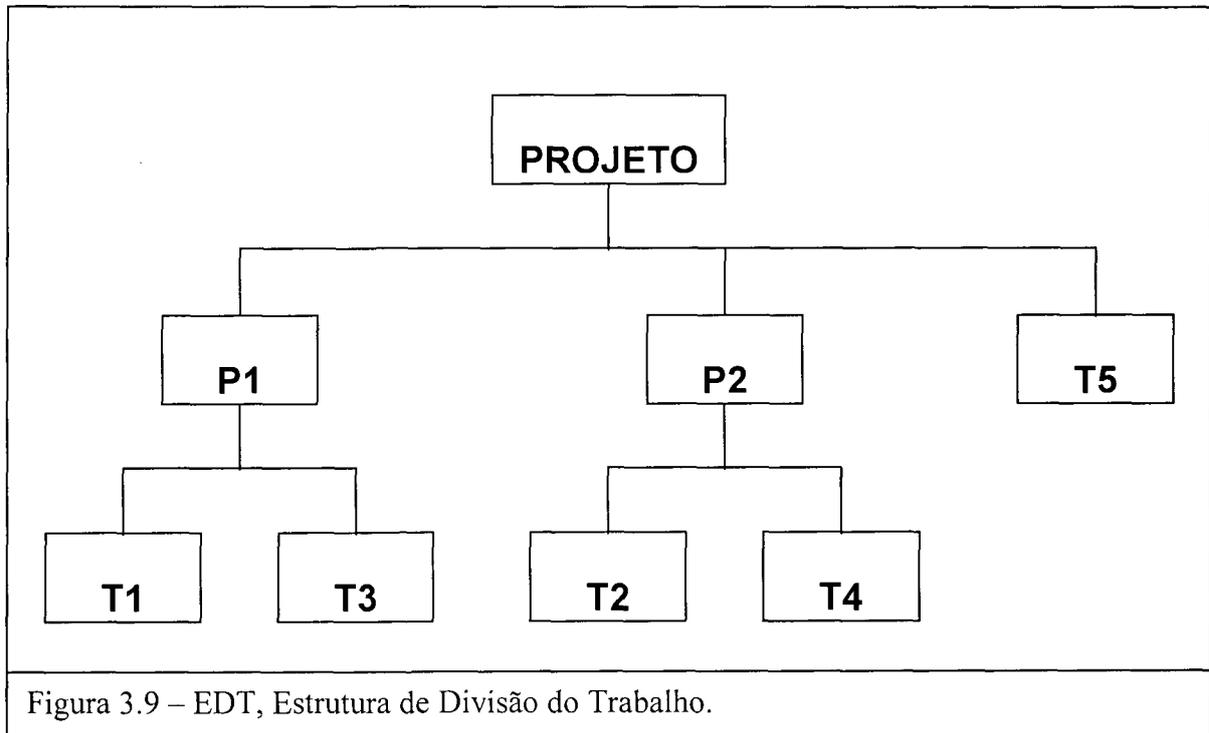
- Deve especificar o comportamento externo do sistema.
- Deve especificar as restrições para implementação.
- Deve ser fácil de alterar.
- Deve servir de referência para os mantenedores do sistema.
- Deve registrar o planejamento sobre o ciclo de vida adotado para o sistema.
- Deve caracterizar respostas inaceitáveis do sistema e eventos indesejáveis.

Sommerville também sugere um roteiro para o documento de requisitos composto por introdução, glossário, modelos do sistema, definição de requisitos funcionais, definição de requisitos não funcionais, evolução do sistema, especificação de requisitos. A descrição desse roteiro está na tabela 3.5.

Tabela 3.5 – Descrição do Documento de Requisitos	
Capítulo	Descrição
Introdução	Justifica a necessidade do sistema. Descreve rapidamente suas funções e ilustra seu relacionamento com outros sistemas. Contextualiza o software nos objetivos estratégicos e de negócio da organização cliente.
Glossário	Este espaço é para ser utilizado para explicar termos técnicos que o leitor comum não esteja familiarizado.
Modelos do Sistema	Neste capítulo é mostrada a relação entre os componentes do sistema e o ambiente do negócio. Podem ser utilizado modelo de objetos, fluxo de dados e modelo semântico de dados.
Definição dos Requerimentos Funcionais	Os serviços fornecidos ao usuário devem ser descritos neste capítulo. A descrição deve ser um texto descritivo com diagramas explicativos de modo a facilitar o entendimento por parte do usuário. O uso de diagrama explicativo é recomendado.
Definição dos Requerimentos Não Funcionais	As restrições impostas ao software e suas funções devem constar deste capítulo. Restrições quanto a memória, especificação de desempenho e representação dos dados podem ser incluídos. Produtos e processos padrões são descritos.
Evolução do Sistema	Neste capítulo é feita uma revisão do contexto do negócio. Procura-se antecipar as mudanças das necessidades do usuário, a natural evolução do sistema e do ambiente, com vistas para as alterações e melhorias futuras.
Especificação de Requisitos	Neste capítulo os requerimentos funcionais e não funcionais devem ser detalhados.

3.3.2 – EDT: Estrutura de Divisão do Trabalho

EDT – Estrutura de Divisão do Trabalho é uma tradução para *WBS – Work Breakdown Structure*. Consiste de uma série de tarefas agrupadas em pacotes de trabalho em forma de árvore. Veja a figura 3.9.



Na figura 3.9, existe uma EDT com três níveis. No nível 2, têm-se três itens, que são P1, P2 e T5. A partir de um item de qualquer nível pode-se caracterizar um pacote de trabalho. P2 é um pacote de trabalho composto pelas tarefas T2 e T4. T5 é um pacote de trabalho composto apenas pela tarefa T5. PROJETO é um pacote de trabalho composto pela tarefa T5 e os pacotes P1 e P2. Os pacotes de trabalho têm uma carga de trabalho associada. A carga de trabalho do pacote é igual à soma das cargas de trabalho dos subpacotes que o compõem. Assim, tomando a figura 3.9 como exemplo, se a tarefa T2 tem uma carga de trabalho igual a 10 dias e a tarefa T4 tem uma carga de 15 dias, o pacote P2 terá a carga de 25 dias.

Se o projeto total for muito grande, é comum se quebrar o projeto em subprojetos. Neste caso um determinado pacote se torna um projeto independente.

O uso de EDT é muito freqüente nos projetos em geral [Ould 1990] [Shenhar 1998]. Em 25 de março de 1993 a EDT é descrita pelo estabelecimento de defesa americano no documento *Military Standard (MIL-STD) 881B*. Neste documento lê-se:

Uma Estrutura de Divisão do Trabalho é uma árvore orientada ao produto, composta por hardware, software, serviços, dados e facilidades... EDT mostra e define o(s) produto(s) a ser(em) desenvolvido(s) ou produzido(s) e relaciona os elementos do trabalho a serem realizados. Relacionamento dos elementos uns com os outros e com o produto final.

Para elaborar uma EDT, costuma-se iniciar por uma abordagem orientada à tarefa, com uma lista superficial de afazeres. O projeto é uma coleção de tarefas. Essas tarefas são agrupadas logicamente por diferentes critérios, dependendo da escolha do projetista. A divisão lógica pode ser por fases do projeto, áreas funcionais, produtos principais etc.

O projetista deve tentar fazer com que a EDT atinja as seguintes metas:

- Ser compatível com a forma que o trabalho será executado e com os custos e prazos que serão gerenciados.
- Dar visibilidade ao esforço de resolução de riscos.
- Mapear o modelo do processo utilizado.
- Promover clara divisão de responsabilidades para gerentes e líderes de tarefas.
- Prover uma estrutura para obtenção de dados históricos do processo.

- Fazer sentido para os executores dos projetos e os coletores de dados. Os dados coletados são de desempenho e financeiros.

Ould [Ould 1990] sugere que os pacotes de atividades compostos por mais de uma tarefa usem rótulos que indiquem a classe das tarefas (exemplo: Relatórios, Definição de Requisitos), enquanto as tarefas devem se iniciar com verbos indicando o que deve ser feito (exemplo: Fazer Relatório, Definir Requisitos).

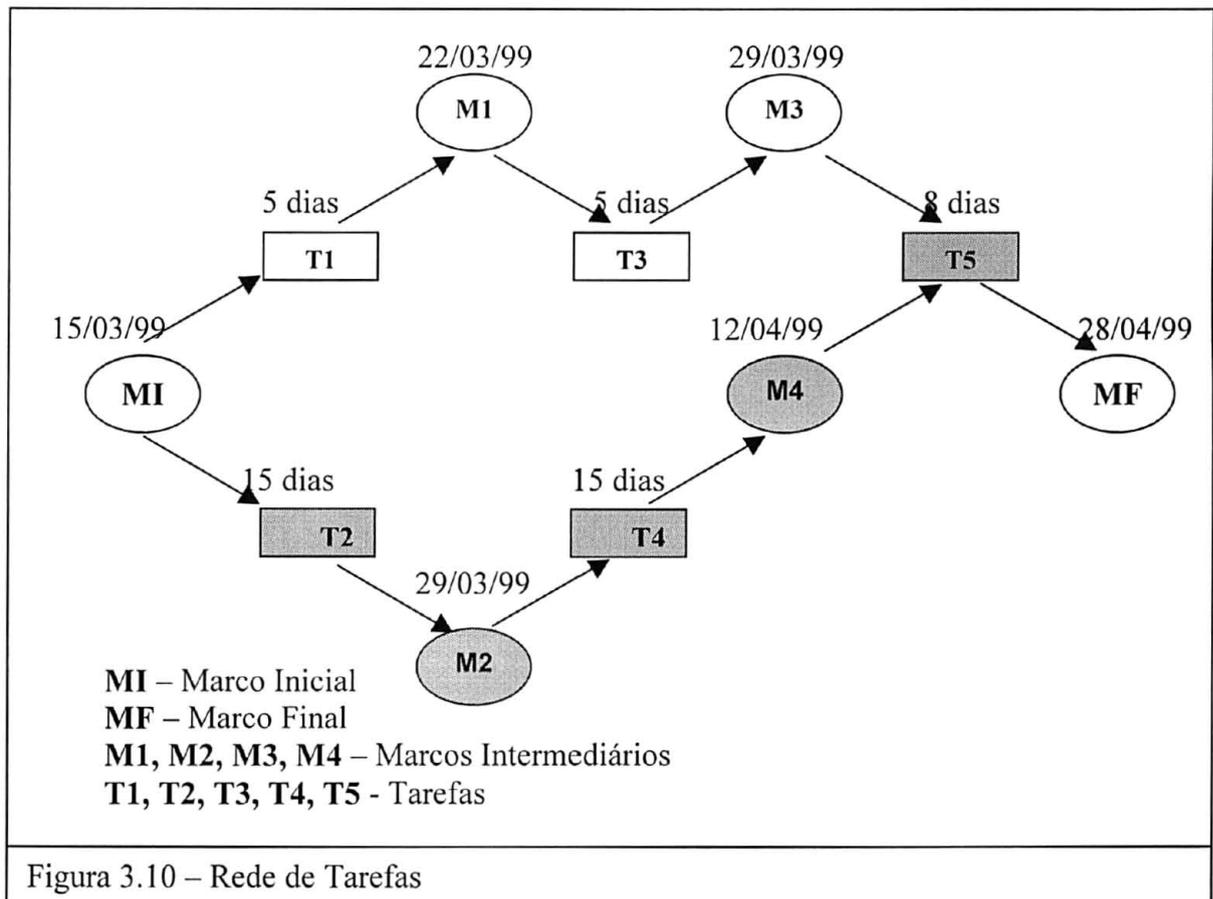
A completude é muito importante para a EDT. Na EDT deve constar tudo o que deve ser feito no projeto. Caso isso não aconteça de maneira satisfatória, o planejamento do projeto estará comprometido. Se uma atividade não consta da EDT, ela deverá ser realizada em tempo zero e a custo zero.

Cada atividade na EDT deve ser dimensionada quanto ao esforço necessário para executá-la. Existe um certo consenso de que os tempos das tarefas são mais bem estimados quando feitos por semanas [Ould 1990] [Sommerville 1990]. As tarefas devem ter prazos de 5 a 15 dias. Caso a organização exija apresentação em meses ou horas, em vez de semanas ou dias, deve-se converter-las após finalizado o trabalho de estimativa de prazos. Esta estimativa de prazos é a parte mais subjetiva do planejamento. Os tempos de execução das tarefas influenciam bastante no custo da mão-de-obra e, conseqüentemente, no custo final do projeto. Ould [Ould 1990] recomenda com bastante ênfase que se utilizem pessoas com experiência no tipo de aplicação e no negócio ao qual a aplicação servirá.

Neste trabalho serão apresentadas técnicas e abordagens para definição de custos que envolvem estimativa de prazos no item 3.3.4.

3.3.3 - Cronograma

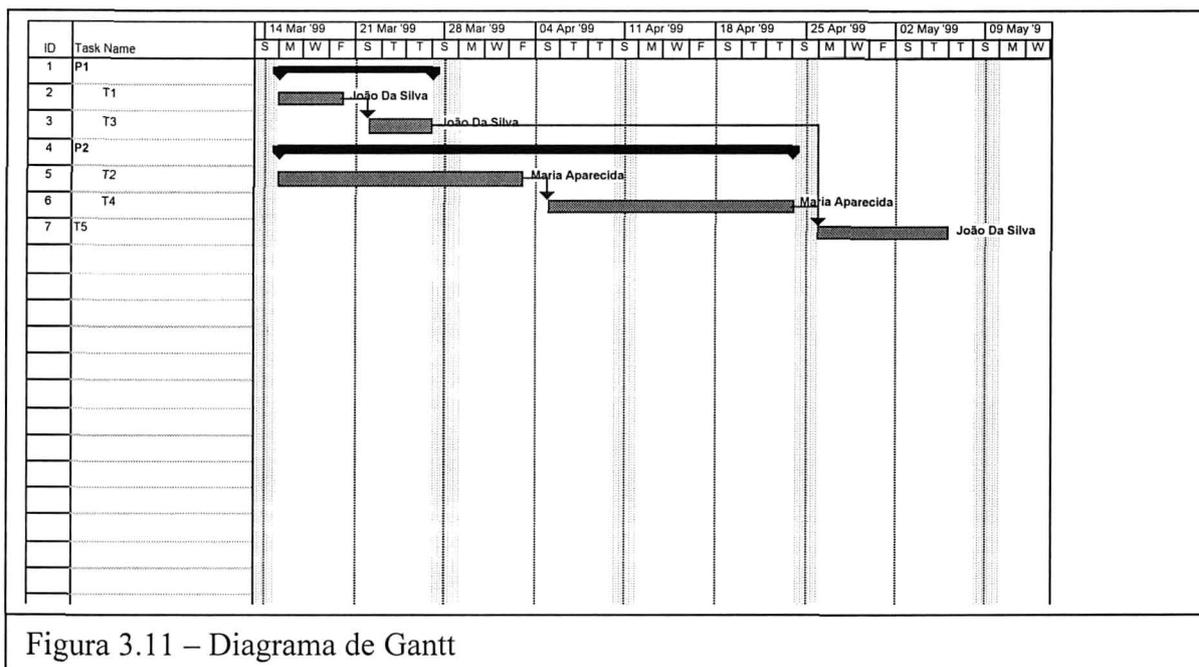
Para se realizar um projeto é necessário quebrar o total do trabalho a ser realizado em tarefas menores. Sommerville recomenda que estas tarefas tenham a duração de uma ou duas semanas. Para cada tarefa é estabelecido um prazo de execução. As tarefas expostas de maneira ordenada, com relação de dependência, geram marcos no tempo, como mostra a Figura 3.10.



O caminho do marco inicial ao final que está em cinza na figura 3.10 é o caminho crítico. Caminho crítico é o caminho do marco inicial ao final que possui o maior prazo. Este é o prazo mínimo de entrega do projeto [Sommerville 1990].

No exemplo da figura 3.10, como as atividades T1 e T3 podem ser realizadas em paralelo com T2 e T4, duas pessoas seriam o número ideal de profissionais para este projeto. Se apenas uma pessoa fosse alocada neste projeto, o prazo final não seria o prazo do caminho crítico.

Na figura 3.10 encontram-se as mesmas tarefas da figura 3.11 distribuídas entre duas pessoas na forma conhecida como diagrama de Gantt [Sommerville 1990].



3.3.4 - Estimativa de Custos

No aspecto orçamento, tem-se os componentes de custo e as técnicas de estimativa de custos. Sommerville classifica os componentes de custo como: Hardware, Software, Treinamento, Viagens e Esforços. O esforço geralmente é medido em pessoas-mês ou pessoas-hora.

As técnicas para estimar custos são definidas por Boehm [1981] como sendo: modelo algoritmo de custos, julgamento por especialista, estimativa por analogia, lei de Parkinson, lance para ganhar.

Modelo Algoritmo de Custos: Usa algoritmo para fornecer custo em função de uma medida de software (geralmente tamanho). É estimada a métrica e o algoritmo prediz o esforço requerido.

Julgamento por Especialista: Um ou mais especialistas são chamados. São especialistas nas técnicas de desenvolvimento de software que serão utilizadas ou no domínio da aplicação. Chega-se a um custo por consenso.

Estimativa por Analogia: É utilizado quando um projeto semelhante em tecnologia ou no domínio da aplicação já foi desenvolvido.

Lei de Parkinson: O trabalho deve ser expandido para alcançar o tempo disponível e o custo deve ser aquele disponível para gastar.

Lance para Ganhar: O preço deve ser o que o usuário tem disponível para gastar.

Existem também duas abordagens: abordagem *top-down* e abordagem *bottom-up*.

Abordagem Top-Down: O custo é estimado por uma propriedade geral do projeto e depois repartido entre os componentes do projeto.

Abordagem Bottom-up: O custo é estimado para cada componente do projeto e a estimativa geral é a soma dessas partes.

3.4 - GERENCIAMENTO DE RISCOS

Serão apresentadas duas visões para redução de riscos em projetos de software. Uma de Grady Booch [Booch 1995] e outra de Martyn Ould [Ould 1990].

3.4.1 - Visão de Grady Booch

Grady Booch [Booch 1995], em seu livro **Object Solutions – Managing the Object – Oriented Project** comenta que o gerenciamento de software deve seguir três regras básicas:

- Manter os desenvolvedores focados nas características essenciais mínimas.
- Proteger a equipe de qualquer distração e evitar que desperdice seu potencial.
- Tomar as decisões necessárias para resolver conflitos de negócio e de engenharia.

Estas regras são acompanhadas de responsabilidades que são:

- Gerenciar riscos;
- Planejar e agendar atividades;
- Levantar custos e gerenciar pessoal;
- Monitorar, medir e testar;
- Documentar.

Sobre Gerenciamento de riscos, Booch [Booch 1995] recomenda que o desenvolvimento de software seja baseado em objetos e voltado à arquitetura. Segundo ele, com esta abordagem, os riscos a que os projetos estão sujeitos vêm a tona.

Uma vez identificado os riscos, recomenda-se atacá-los no início do projeto, projetando, codificando e testando pequenos modelos nos quais os riscos estão contidos. Assim se há dúvidas quanto a utilização de uma nova biblioteca de comunicação, é recomendável que seja feito um pequeno programa nas mesmas condições do projeto inicial para dar cabo das incertezas.

Segundo Booch, cada projeto tem sua coleção única de riscos. Não existe projeto com risco zero. Os riscos aumentam à medida que aumenta a complexidade do software. Recomenda-se que se mantenha uma lista dos 10 maiores riscos.

Ele comenta que a responsabilidade de cada risco é, em última análise, uma decisão de negócios, e não necessariamente de engenharia.

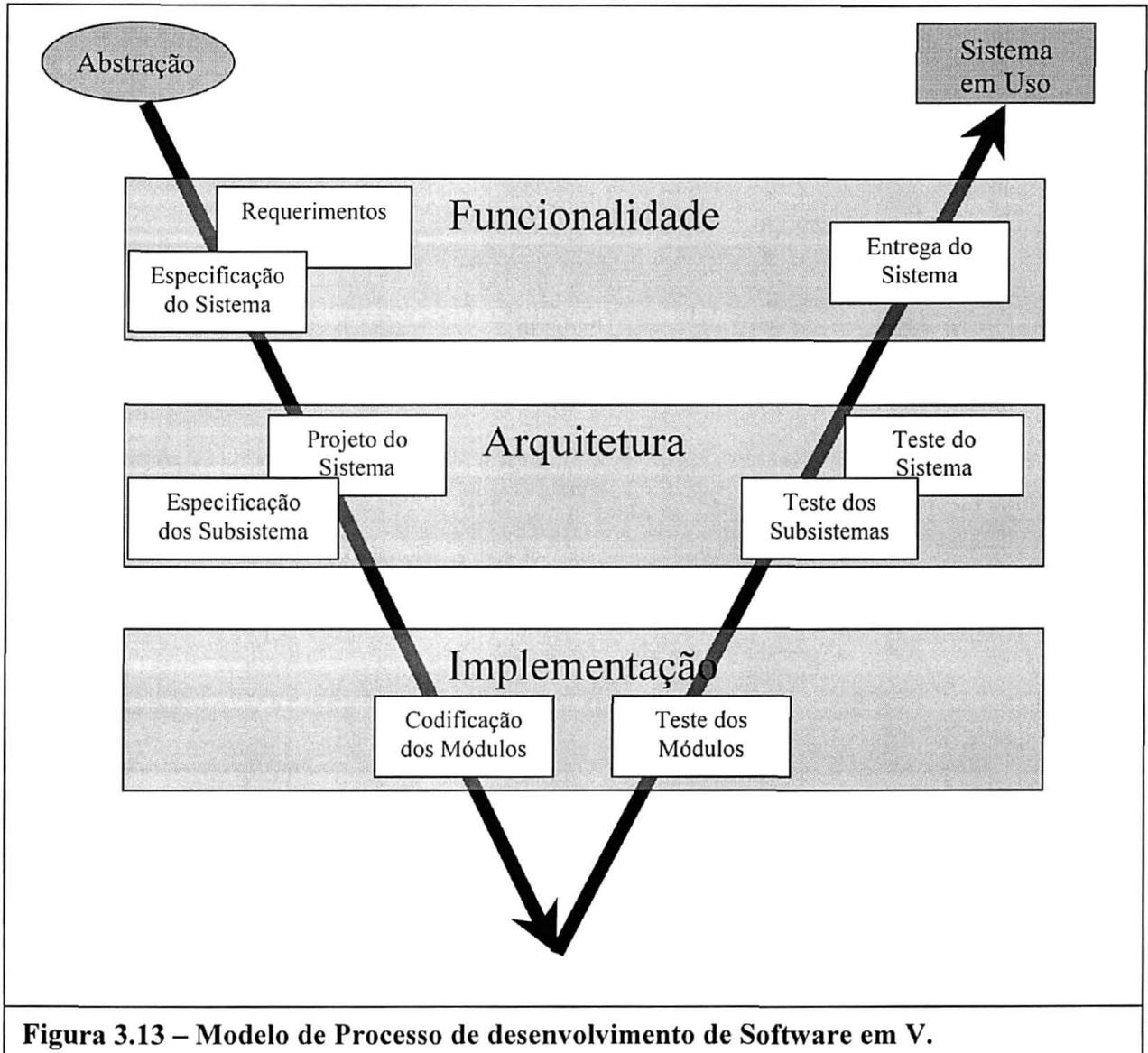
Booch define o termo artefatos como sendo os materiais que o gerente usa para exercer controle sobre o projeto, são os marcos de início e fim de atividades (*millestones*) e os materiais entregues após a execução das atividades que a evidenciam (*deliverables*). O gerente deve estabelecer marcos e evidências plausíveis, realizáveis. Afirma que, em projetos de sucesso, o constante planejamento é necessário. Este planejamento é feito com o controle dos artefatos chaves e motivado pelos fatores de risco.

3.4.2 - Visão de Martyn Ould

Ould, em seu livro **Strategies for Software Engineering – The Management of Risk and Quality** [Ould 1990], diz que se deve fazer o planejamento baseado em redução de riscos. Para isso, o modelo do processo de desenvolvimento de software deve ser compatível com o risco enfrentado. O modelo do processo não é um método. Para um projeto adota-se apenas um modelo do processo e vários métodos inseridos no processo. Segundo Ould, o principal motivo para a adoção de um modelo de processo é dar ao projeto de desenvolvimento de software uma estrutura de redução de riscos. Um projeto sem estrutura não é gerenciável. Ele não pode ser planejado, não é possível fazer estimativas, não se pode estabelecer marcos, não é possível se monitorá-lo e não se pode oferecer ao cliente nenhuma perspectiva de custos e prazos. Com uma estrutura fornecida pelo modelo de processo, planejada para reduzir riscos, reduz-se a incerteza e torna-se o projeto mais gerenciável.

Ould ilustra alguns modelos utilizados e os relaciona com o nível de incerteza dos requisitos. Os modelos de processos descritos são: Modelo de processo em V, modelo de processo em V com protótipo, modelo de processo de desenvolvimento evolucionário, modelo de processo de entrega incremental, modelo de processo exploratório. Os modelos de processo de software estão esquematizados na Figura 3.12.

O modelo de processo em V é semelhante ao tradicional modelo cascata, porém, no modelo em V, as atividades de testes e validação são ressaltadas e relacionadas com o que foi estabelecido nos requisitos e definições de implementação.



O **modelo de processo em V** é utilizado quando se tem um baixo nível de incerteza. Os requisitos são bem definidos e inteligíveis pelos desenvolvedores no início do projeto

O **modelo de processo em V com protótipo** é utilizado quando existe uma pequena incerteza a respeito dos requisitos ou da sua implementação. Neste caso um protótipo é realizado. As atividades necessárias ao protótipo são adicionadas ao modelo de processo em V.

No **modelo de processo evolucionário**, vários sistemas são desenvolvidos com base nos sistemas anteriores. Cada sistema é completo em funcionalidade, mas as funcionalidades são dependentes da mudança natural dos conceitos. Ao longo do tempo, sucessivas versões são entregues. Este processo pode ter fim ou não, dependendo de quando e como o sistema vai se estabilizar pelas intervenções com o ambiente de negócio.

O **modelo de processo incremental** é utilizado quando apenas algumas funcionalidades são requeridas ou estão bem definidas. Deste modo, o sistema é entregue em várias etapas ficando com um número maior de funções após cada incremento.

O **modelo de processo exploratório** é utilizado quando se tem uma definição fraca do produto final. A tônica é aprender os requerimentos durante o processo. Tem um nível alto de incerteza. O processo pode ou não chegar a um produto final.

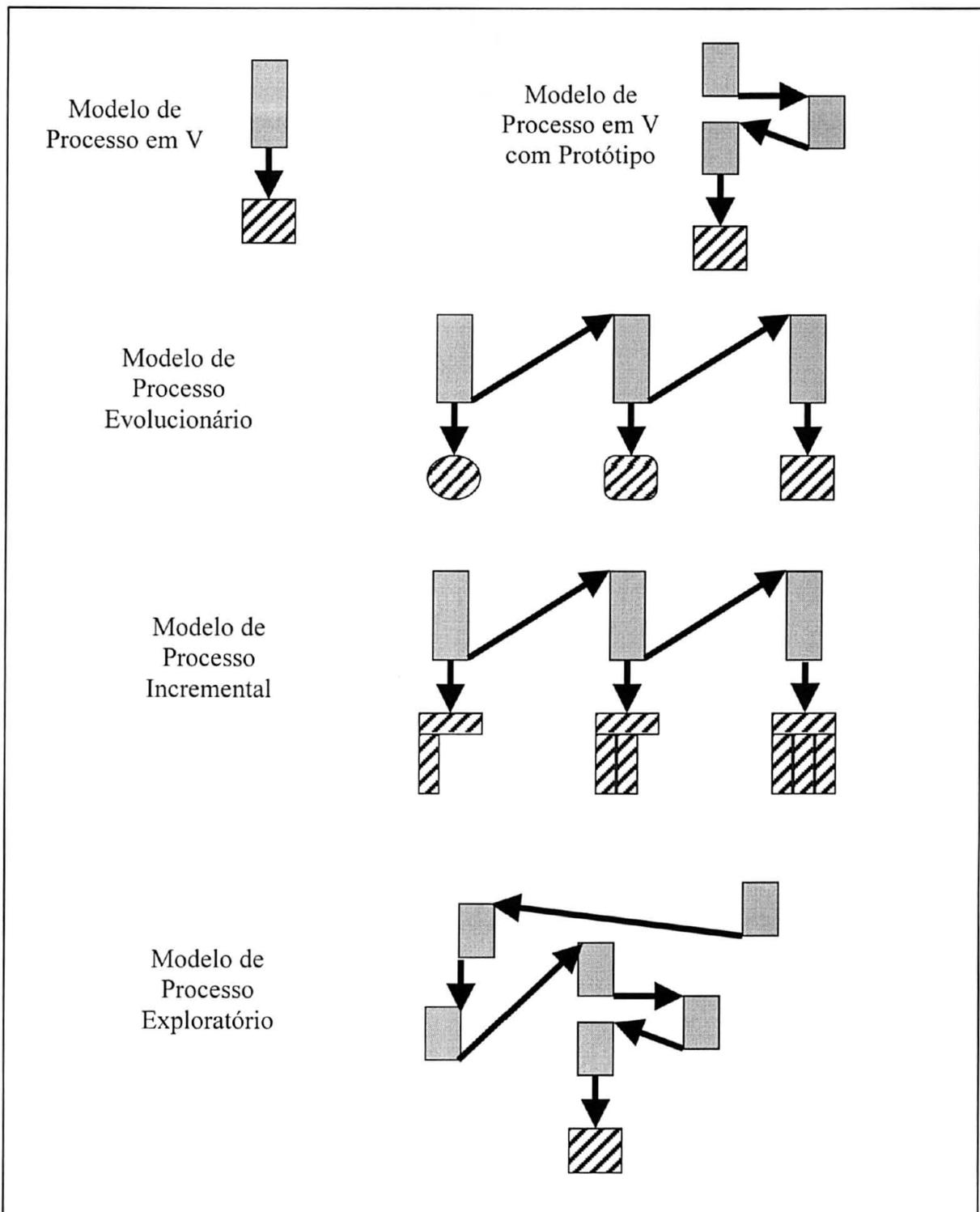
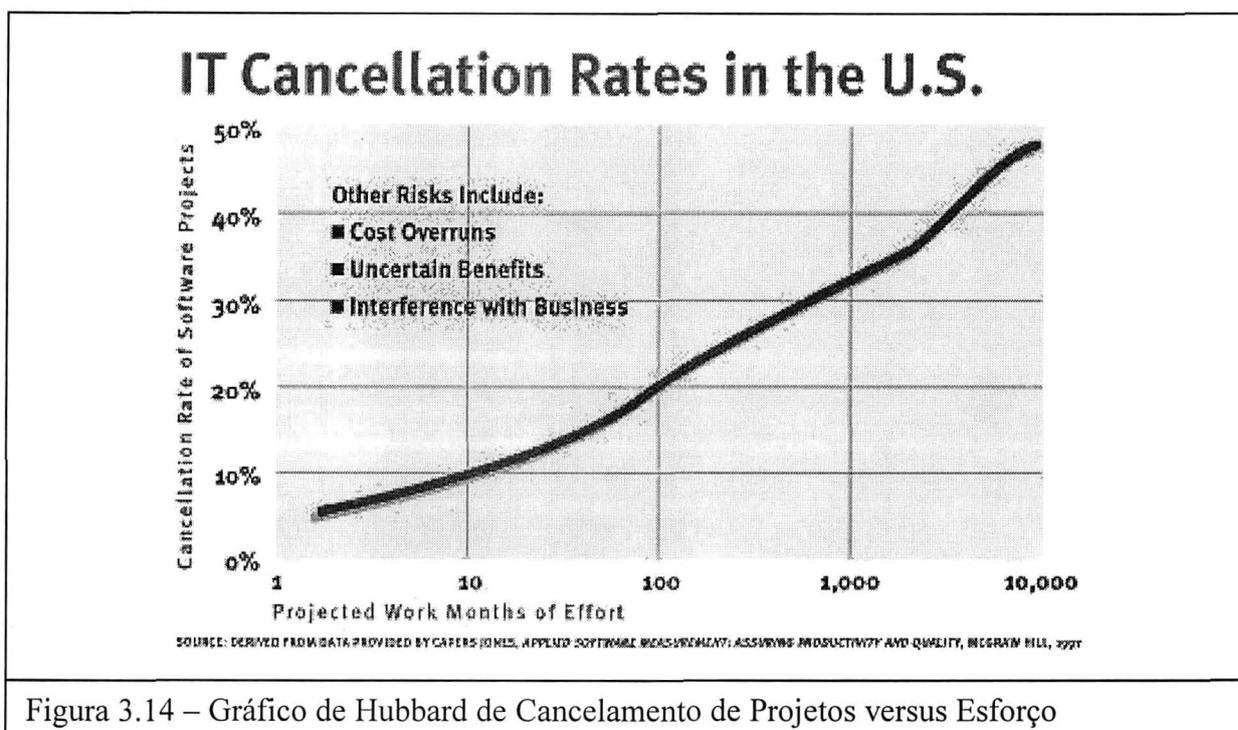


Figura 3.12 – Modelos de Processo segundo Ould.

3.4.3 - Gráfico de Hubbard.

Muitos autores comentam a relação entre complexidade e riscos [Booch 1995] [Ould 1990] [Sommerville 1990] [Rosa 1997]. Hubbard [Hubbard 1998] traz uma relação numérica entre essas duas variáveis, ao relacionar percentual de projetos cancelados com o esforço em homens-mês. Neste gráfico confirma-se o crescimento dos riscos em função da complexidade. Nota-se a vantagem relativa de pequenos projetos de software, os quais possuem uma menor taxa de cancelamentos. Veja Figura 3.14.



4.MÉTODO 5E+3P.

No capítulo anterior apresentamos o estudo de Shenhar em diversas organizações, no qual os projetos são classificados em duas dimensões: Incerteza Tecnológica e Escopo do Sistema. A incerteza tecnológica vem com a inovação tecnológica, com a novidade para o mercado. Quando se aumenta a incerteza, nota-se o crescimento de atividades como desenvolvimento e testes, congelamento tardio do projeto, perfil técnico, flexibilidade e comunicação. Quanto maior o escopo do projeto, mais intensamente acontecem as atividades de planejamento, controle, subcontratação, burocracia e documentação. Outras variáveis, como engenharia de sistema, integração de sistema, gerência da configuração e gerência de risco, aumentam com a incerteza e com o escopo do projeto. Para incerteza tecnológica são caracterizados 4 tipos de projetos (A - Baixa Tecnologia, B - Média Tecnologia, C - Alta Tecnologia, D - Altíssima Tecnologia) e para o escopo do projeto são caracterizados 3 tipos de projetos (1 – Montagem ou Componente, 2 – Sistema, 3 – Matriz). [Shenhar 1998].

Os projetos do tipo A (exemplo: construção de uma piscina) geralmente não possuem fase de planejamento. Geralmente são aproveitados esquemas e cálculos de um projeto

anterior. Por isso os projetos de desenvolvimento de software não se encaixam na descrição do tipo A (Baixa Tecnologia).

Os pequenos projetos de software são de componentes de software (exemplo: rotina de acesso a banco de dado, componente Corba para verificação de CPF digitado etc) ou pequenos sistemas (exemplo: controle de locação de fitas para vídeo locadora, controle de fluxo de caixa etc). Em muitos projetos se faz necessário desenvolver os componentes e o pequeno sistema. Nestes pequenos projetos de software existe a informalidade, característica do tipo 1 (Montagem), e a integração de componentes, característica do tipo 2 (Sistema). O nível de comunicação, complexidade tecnológica e o fechamento do planejamento após um ou dois ciclos caracterizam projetos do tipo B (Média Tecnologia).

Neste trabalho será proposto um método para gerenciamento de pequenos projetos que se encontram caracterizados com B1 (Montagem de Média Tecnologia) ou B2 (Sistema de Média Tecnologia) pela classificação de Shenhar, porém o método tratará apenas dos projetos de software, como indica o círculo de Pequenos Projetos de Software na Figura 4.1. Quanto ao número de profissionais envolvidos, os pequenos projetos de software não chegam a ultrapassar 10 pessoas.

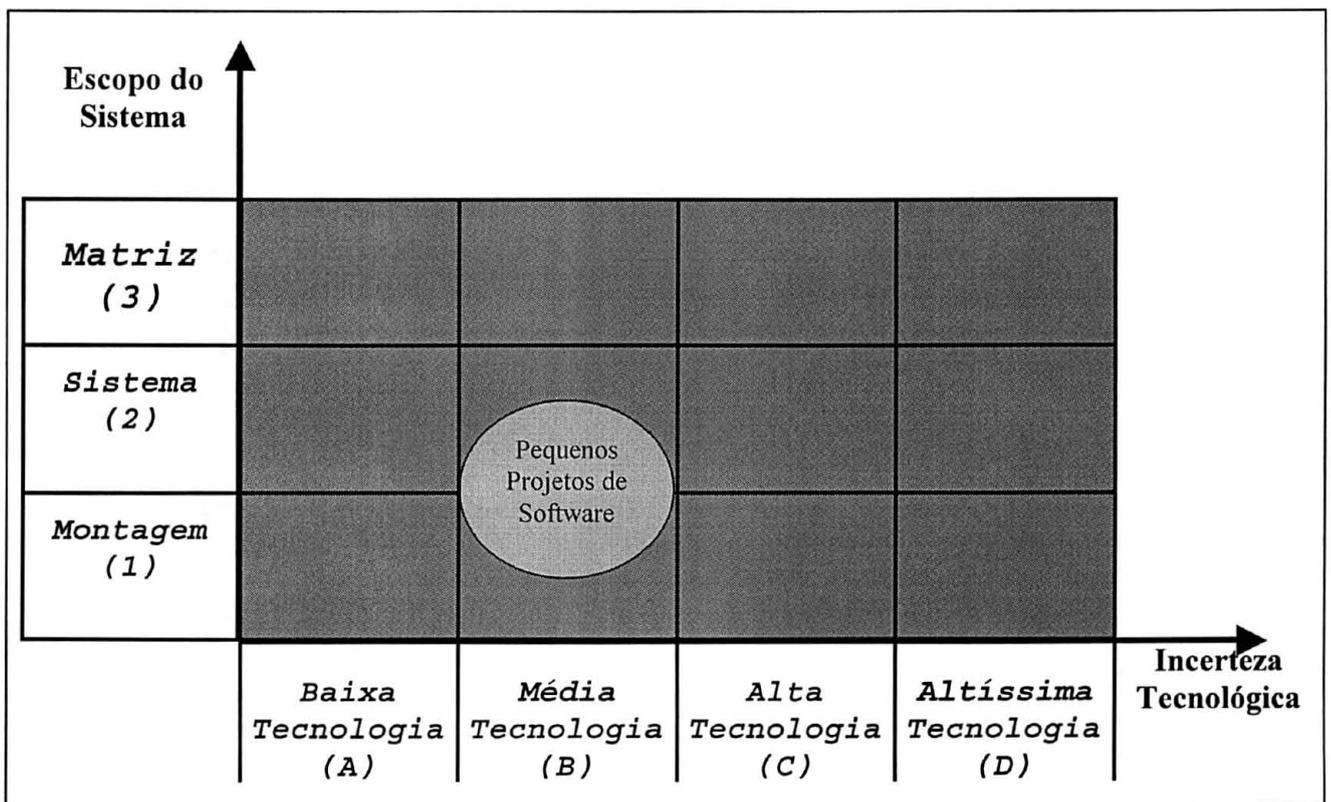


Figura 4.1 - Classificação Bidimensional de Shenhar.

Os modelos de qualidade de software são largamente aplicáveis por serem genéricos. As empresas podem entender os princípios e adaptá-los à sua cultura específica. Para que a adaptação seja possível tem-se um grupo de pensadores que entendem os modelos e a cultura da empresa, criando, adaptando e divulgando um modelo aplicável.

As pequenas empresas não dispõem de recursos humanos e financeiros para criar um modelo aplicável. Um método já pronto, ao invés de um modelo, parece ser mais adequado à necessidade das pequenas empresas. O método ajuda a pequena empresa a entender o modelo. O método é um exemplo aplicável do modelo.

Neste trabalho serão focadas as atividades de planejamento e controle de pequenos projetos de software que são atividades características do gerenciamento de projetos.

A informalidade em pequenos projetos é uma característica positiva e negativa. É positiva à medida que existe fácil acesso a todos no grupo e podem-se corrigir rapidamente os rumos do projeto, e negativa à medida que as decisões não são registradas, prejudicando o andamento por falha na comunicação oral.

Certa burocracia deve ser introduzida, mas na medida em que supere as desvantagens da informalidade, deixando espaço para a informalidade apresentar suas vantagens. Em outras palavras, a medida da burocracia em pequenos projetos deve ser pequena.

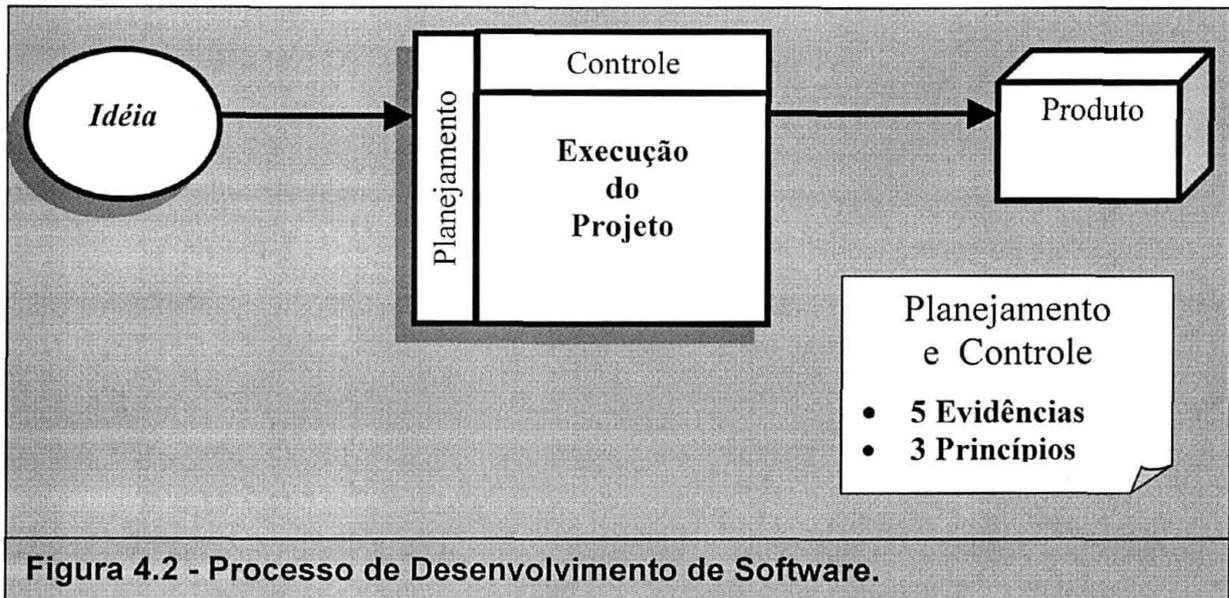
Para facilitar a absorção desse conhecimento, será dada ênfase às evidências do processo, isto é, documentos que provam a intenção de planejamento do desenvolvimento de software. Foram eleitos neste trabalho apenas 5 documentos para planejar e controlar com pouca burocracia, 5 evidências(5E). As evidências são:

- Relatório de Requisitos
- Estrutura de Divisão do Trabalho
- Relatório de Custos
- Cronograma
- Lista dos Riscos mais Importantes

O risco de insucesso em pequenos projetos, apesar de ser menor que nos grandes projetos, ainda existe. Para minimizar os riscos, foram escolhidos 3 princípios(3P) a partir da bibliografia:

- Modelo de Processo em V Simplificado
- Protótipo quando possível
- Antecipar no cronograma a resolução dos riscos

Essas 5 evidências e 3 princípios (5E+3P) devem conduzir o projeto de software como indica a Figura 4.2



O produto de software tem origem em uma idéia. Mas para que a idéia vire realidade é necessária a execução de um projeto de software. O planejamento do projeto é uma etapa que se deve realizar para o sucesso do projeto. No método 5E+3P proposto, em um ambiente de desenvolvimento de software, a elaboração das cinco evidências caracterizam o planejamento realizado.

Descrevendo grosseiramente o planejamento e o controle em pequenos projetos de software, pode-se dizer que, quando se produzem as evidências, está-se planejando, e quando se mantêm as evidências atualizadas está-se controlando o projeto.

O método é destinado a pequenos projetos de software. Tem como objetivo aumentar a probabilidade de sucesso do projeto resolvendo riscos e controlando os custos, prazos e funcionalidades do projeto. É um método para planejar e controlar o projeto com relação a estas variáveis.

O planejamento do sistema com relação à arquitetura e a implementação (*design*), a partir de um determinado conjunto de requisitos, não diz respeito a este método.

O planejamento é particular de cada projeto. Depende da cultura e informação de que dispõem os projetistas. O método vê o processo de planejamento como uma caixa preta nas quais, sejam quais forem as escolhas feitas pelos projetistas, deve-se gerar evidências. Por isso este método está definido em função de cinco evidências que devem ser geradas levando em conta três princípios. Assim, tem-se um padrão sem reprimir o processo de criação inerente ao planejamento.

Deve-se começar o planejamento pelo relatório de requisitos, definindo o **que** deve ser feito. Depois se faz a EDT (Estrutura de Divisão do Trabalho), definindo o **como** deve ser feito. A partir da EDT pode-se gerar o Cronograma, o Relatório de Custos e a Lista de Riscos.

4.1. ANÁLISE DE REQUISITOS

O relatório de requisitos é uma evidência do processo de análise de requisitos. Nesta fase deve-se documentar o **que** o usuário deseja receber, sem se preocupar em documentar **como** será implementado.

O relatório de requisitos servirá como um contrato entre o cliente do projeto de software e os desenvolvedores. Neste trabalho abordar-se-ão projetos onde apenas o documento, estruturado segundo a recomendação de Sommerville [Sommerville 1990], seja suficiente para conter todas os requisitos necessários. Esse documento deve ter de 2 a 20 folhas.

A Tabela 4.1 descreve o conteúdo do documento de requisitos, mas um exemplo como o do anexo A.1 dá uma idéia melhor da simplicidade do documento.

Tabela 4.1 – Descrição do Documento de Requisitos	
Capítulo	Descrição
Introdução	Justifica a necessidade do sistema. Descreve rapidamente suas funções e ilustra seu relacionamento com outros sistemas. Contextualiza o software nos objetivos estratégicos e de negócio da organização cliente.
Glossário	Este espaço é para ser utilizado para explicar termos técnicos com as quais o leitor comum não esteja familiarizado.
Modelos do Sistema	Neste capítulo é mostrada a relação entre os componentes do sistema e o ambiente do negócio. Pode ser utilizado modelo de objetos, fluxo de dados e modelo semântico de dados.
Definição dos Requerimentos Funcionais	Os serviços fornecidos ao usuário devem ser descritos neste capítulo. Deve ser um texto descritivo com diagramas explicativos de modo a facilitar o entendimento por parte do usuário.
Definição dos Requerimentos Não Funcionais	As restrições impostas ao software e suas funções devem constar deste capítulo. Restrições quanto a memória, especificação de desempenho e representação dos dados podem ser incluídos. Produtos e processos padrões são descritos.
Evolução do Sistema	Neste capítulo é feita uma revisão do contexto do negócio. Procura-se antecipar as mudanças das necessidades do usuário, a natural evolução do sistema e do ambiente, com vistas para as alterações e melhorias futuras.
Especificação de Requisitos	Neste capítulo os requerimentos funcionais e não funcionais devem ser detalhados.

4.2. EDT – ESTRUTURA DE DIVISÃO DO TRABALHO

A EDT é a evidência intermediária entre o relatório de requisitos e as evidências cronograma, relatório de custos e lista de riscos. A primeira idéia ao se fazer o trabalho de planejamento é respeitar a seqüência: relatório de requisitos, EDT, cronograma, relatório de custos e lista de riscos. Porém os documentos interagem, fazendo com que tal seqüência linear dificilmente aconteça. Além da interação há sempre um fato novo ou uma lembrança que nos força a rever os documentos. Seja como for, a EDT é o documento chave, o elo de ligação entre as evidências.

Todas as atividades que constam da EDT (Estrutura de Divisão do Trabalho), e somente estas, podem ser acompanhadas pelo gerente do projeto e entrarão na compilação de custos e prazos. Se uma atividade não consta da EDT, ela não consta do planejamento do projeto. Veja Figura 4.2.

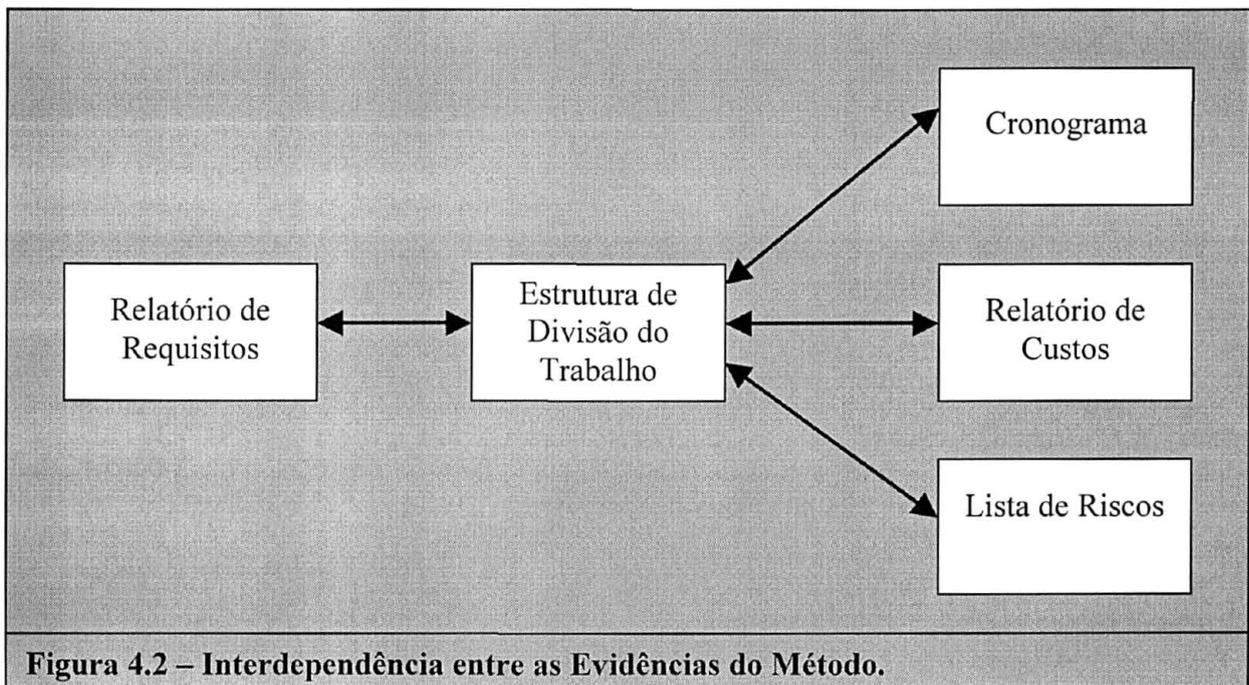


Figura 4.2 – Interdependência entre as Evidências do Método.

A EDT é uma tradução de *WBS – Work Breakdown Structure*. Consiste de uma série de atividades necessárias à realização do projeto. As atividades são agrupadas em pacotes de trabalho segundo uma divisão adotada. Para cada atividade ou pacote de atividades é estabelecido um responsável, e o responsável deve atribuir um prazo para a realização da atividade.

Os dois primeiros níveis da divisão do trabalho em pequenos projetos costumam ser uma divisão relativa ao ciclo de vida e outra relativa a aplicação. O exemplo do anexo A.2 ilustra essa divisão.

A atribuição de prazos é uma tarefa de previsão incerta e subjetiva. As técnicas e abordagens apresentadas no capítulo anterior podem auxiliar nesta tarefa. As técnicas para estimar custos são definidas por Boehm [Boehm 1981] como sendo: Modelo algoritmo de custos, julgamento por especialista, estimativa por analogia, lei de Parkinson e preço para ganhar. As abordagens são estimativa top-down e estimativa bottom-up. Recomenda-se que sejam utilizadas pessoas experientes [Booch 1995], [Ould 1990], [Sommerville 1990].

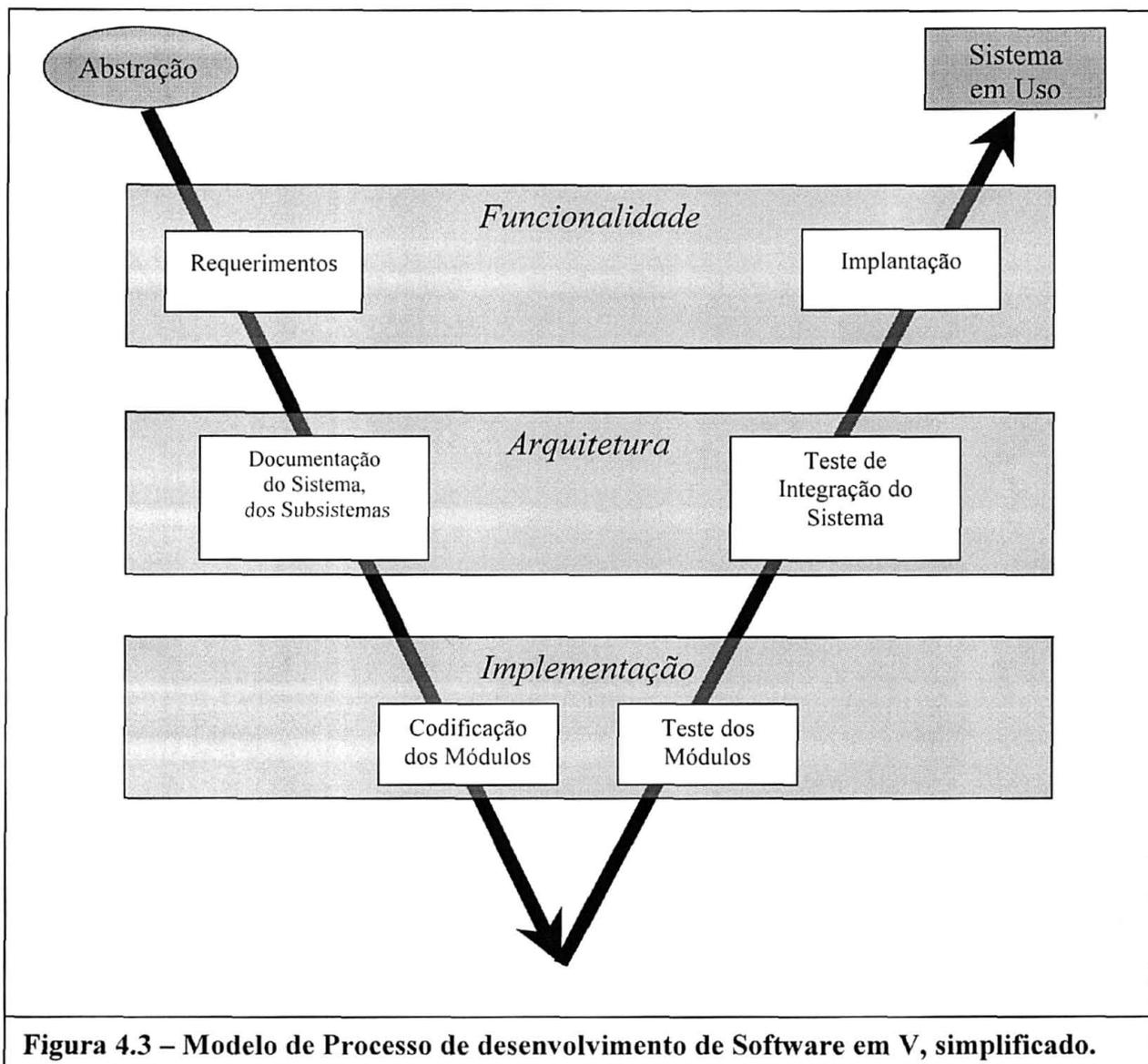
4.3 MODELO DE PROCESSO EM V SIMPLIFICADO

O pequeno projeto está exposto a menos e menores riscos [Booch 1995] [Ould 1990] [Sommerville 1990] [Rosa 1997]. A principal recomendação de Ould quanto à redução de riscos é que se tenha um processo [Ould 1990]. Os projetos que são objetos deste estudo serão definidos como os que se encaixam nos dois primeiros modelos de Ould, o modelo de processo em V e o modelo de processo em V com protótipo, que são os projetos com os menores índices de incerteza.

Para simplificar o modelo V, propõem-se algumas generalizações e mudança de nomenclatura que são:

- Onde há Requisitos e Especificação de Sistema no modelo de Ould que seja substituído por Requisitos.
- Onde há Projeto do Sistema e Especificação dos Subsistemas que seja substituído por Documentação do Sistema e Subsistemas.
- Onde há Teste do Sistema e Teste dos Subsistemas que seja substituído por apenas Teste de Integração do Sistema
- Onde há Entrega do Sistema que seja substituído por Implantação.

Veja Figura 4.3.



Quando se faz necessário um protótipo, é interessante que o trabalho realizado seja reaproveitado no projeto definitivo. O protótipo pode ser um módulo da aplicação, um módulo componente ou um subsistema. O importante do protótipo é que seja útil para resolver as incertezas do projeto.

4.4.RELATÓRIO DE CUSTOS

O relatório de custos deve ter como meta levantar o custo total do projeto para definição de um orçamento. A divisão dos custos proposta por Sommerville [Sommerville 1990] é uma maneira de classificá-los. Esta divisão é:

- **Hardware e Software.** Custos fixos e de manutenção.
- **Viagens.** Custos fixos e diários.
- **Treinamento.**
- **Esforço** em Homens-Mês.

Os itens de cada uma dessas divisões podem ser precificados com um valor unitário e uma quantidade especificando a unidade adotada. É como se costuma preencher uma nota fiscal. Veja Anexo A.4.

Os custos geralmente têm uma parte fixa formada pelos valores que se paga uma vez durante o projeto, como o custo do hardware necessário. Outra parte é variável formada pelos valores que aumentam em função do tempo de duração do projeto, como o valor da mão-de-

obra empregada. Esses dois aspectos devem ser lembrados na composição do custo total do projeto.

Neste trabalho recomenda-se que na divisão de Hardware e Software de Sommerville sejam incluídos outros custos com materiais, como, por exemplo, a compra de papel especial necessário ao sistema.

4.5.CRONOGRAMA

O cronograma fica mais fácil de ser feito tendo-se a EDT (Estrutura de Divisão do Trabalho) com os prazos e responsáveis determinados. A partir da EDT, com prazos atribuídos a cada atividade, deve-se levantar a relação de dependência entre as atividades e estabelecer início e fim de cada atividade. Com a data início e data fim de cada atividade tem-se o cronograma pronto.

O cronograma é costumeiramente apresentado na forma gráfica conhecida como diagrama de Gantt. Veja Anexo A.3.

É importante lembrar que estimar não significa predizer. Por mais experiência que se tenha na hora de estimar prazos e realizar cronogramas, há sempre a possibilidade de um engano ou fato novo que altere a data de entrega do projeto. Quem trabalha com projetos de software sabe que a data final do cronograma costuma ser postergada. A intenção principal do cronograma é justificar a data da entrega. Com o planejamento do projeto tem-se mais certeza da data final do projeto.

4.6. ANÁLISE DE RISCOS

A análise de riscos pode ter início na observação da EDT, verificando se segue o modelo de processo em V simplificado.

A lista de riscos, como conta Booch [Booch 1995], é particular de cada projeto. Dessa forma, não há receita e as melhores pessoas para fazer esta lista são as que estão diretamente envolvidas no projeto. Pessoas experientes na aplicação e em desenvolvimento de sistemas são bastantes úteis e deve-se ter preferência por elas. De posse da lista de riscos, observa-se se as atividades de resolução dos riscos estão no início do cronograma. Veja Anexo A.5.

4.7. CONCLUSÃO

O método 5E+3P de gerenciamento de pequenos projetos se baseia em 5 evidências e 3 princípios. As evidências servem para comprovar o planejamento realizado. As evidências devem apresentar os 3 princípios propostos. As evidências e princípios do método são citados na Tabela 4.2.

Tabela 4.2 – Evidências e Princípios do Método 5E+3P	
Evidências	Princípios
1. Relatório de Requisitos	1. Modelo de Processo em V Simplificado
2. EDT	2. Protótipo quando possível
3. Cronograma	3. Antecipar no cronograma a resolução dos riscos
4. Relatório de Custos	
5. Lista de Riscos	

A EDT (Estrutura de Divisão do Trabalho), o cronograma, o relatório de custos e a lista de riscos vão se refinando, assim como todo o planejamento. O ponto de partida é o relatório de requisitos. O processo de refinamento existe porque os documentos se influenciam mutuamente. Até mesmo o relatório de requisitos pode ser modificado depois de uma negociação com o cliente. Os requisitos mudam devido a dificuldades e/ou facilidades técnicas descobertas a posteriori. Shenhar conta casos onde isso aconteceu [Shenhar 1998].

As evidências indicam que houve o planejamento do projeto. A manutenção das evidências durante o decorrer do projeto caracteriza o controle do projeto. Planejamento e controle indicam que o projeto está sendo gerenciado.

O planejamento não garante as estimativas, mas alerta com antecedência sobre as dificuldades que o projeto enfrentará.

5. ESTUDO DE CASOS.

No capítulo Revisão Bibliográfica foram apresentados alguns textos científicos que tratam de gerenciamento de projetos em geral e na área de software em particular. No capítulo Método 5E+3P houve uma reflexão sobre a literatura apresentada e a proposta de um método baseado em 5 evidências e 3 princípios. O método surgiu de um processo dedutivo a partir dos textos científicos do capítulo anterior. Neste capítulo é estudada a aplicação do método 5E+3P em pequenos projetos de software.

O processo indutivo utilizado para justificar o método é descrito por Lakatos e Marconi [Lakatos 1992] [Lakatos 1995] como Método de Abordagem Científica. O método de abordagem é amplo com um alto nível de abstração.

O método foi aplicado em 3 projetos com a finalidade de observar sua aplicabilidade e a aceitação. Este procedimento empírico é conhecido como Estudo de Caso. Segundo Lakatos & Marconi [Lakatos 1992] [Lakatos 1995] o Estudo de Caso é um Método de Procedimento Científico. Método de Procedimento é um método menos abstrato que o Método de Abordagem. Ele propõe atividades concretas limitadas a um domínio em particular.

Os projetos estudados foram realizados por alunos do último ano da Escola Técnica da Universidade Federal do Paraná durante a disciplina de Projetos.

Foram realizadas três apresentações. A primeira explicando o Método de uma maneira geral e como as evidências se complementam. A segunda para explicar o documento de requisitos e uma terceira para explicar os demais documentos. Para que os alunos de nível médio pudessem ter um texto para rever e estudar a proposta, foi elaborado um documento explicando o método 5E+3P, as evidências e os princípios. Este documento se encontra no Anexo B.

Além das apresentações foram feitas visitas de acompanhamento da elaboração dos documentos uma vez por semana. Nas visitas constatou-se a dificuldade que os alunos tinham em gerar o Relatório de Requisitos. Para completar esta lacuna didática criou-se um projeto fictício de Divulgação de Notas que se encontra especificado no Anexo A. As evidências desse projeto foram apresentadas aos alunos que imediatamente se sentiram seguros em criar as evidências equivalentes para seus projetos. Foram-se aí dois meses de trabalho. Após as férias de julho os alunos retomaram os projetos.

As equipes que apresentaram as evidências se mostraram mais preocupadas em conciliar o prazo de entrega e as funcionalidades propostas. Nos três projetos foram refeitos os requisitos com a intenção de se adequar aos prazos.

O professor da disciplina conseguiu a concessão da escola para que o método fosse aplicado. O professor sugeriu ao conselho escolar que os documentos, chamados no método de evidências, passassem a ser condição necessária para a aprovação dos alunos e que recomendava a aprovação das propostas de projeto final que apresentassem esses documentos. O professor argumentou que através do método o aluno teria oportunidade de conhecer aspectos de gestão de projetos de software. O conselho acatou a recomendação.

5.1.PROJETOS

Os projetos de software desenvolvidos pelos alunos da Escola Técnica Federal estão descritos na Tabela 5.1. As evidências geradas constam do Anexo C. Foram ao todo nove alunos e dois coordenadores.

Tabela 5.1 – Projetos do Estudo de Casos		
Nome do Projeto	Objetivo	Participantes
Namoro OnLine	Criar um site na internet para encontros virtuais baseados em perfis captados e analisados com intermediação de um psicólogo especializado.	Cristiane Pereira de Andrade Aldo Monteiro do Nascimento Mauro Kaliu Souza Coordenador: Irapuru Florido
Boletim-Net	Criar um site para que os alunos, professores e secretaria se comuniquem. Serão publicadas Notas Bimestrais, horários, calendário escolar, avisos, editais e comunicados.	Fernando Ferraz Flávio Luiz da Silva Roberto Marini Steck Coordenador: Irapuru Florido
Protocolo Eletrônico	Disponibilizar na rede interna do Hospital das Clínicas de Curitiba um sistema de informações onde seria possível consultar todos os protocolos de coleta de dados.	Fernando Augusto Starepravo Ricardo Köhler Costa Fábio Mascarello Coordenador: José Simão

5.2.RESULTADOS

A partir da experiência vivida junto aos alunos da Escola Técnica da Universidade Federal do Paraná, alguns fatos foram observados, que estão descritos a seguir:

O Relatório de Requisitos serviu para definir e controlar funcionalidades do sistema a ser definido. Nas primeiras reuniões de acompanhamento, os alunos traziam questões e idéias misturando a fase de requisitos com a fase de projeto, muitas vezes desviando o foco da conversa. Por diversas vezes foi mencionado que o Relatório de Requisitos define **o que** deve ser feito e não o **como** será feito. Ao concluir o relatório de requisitos os alunos conseguiram entender e separar esses dois conceitos, modificando o seu comportamento nas reuniões. Ao final os alunos passaram a modificar o relatório de requisitos por força de prazos ou riscos que foram mencionados em outras evidências.

A EDT mostrou-se um instrumento eficiente de transição entre o Relatório de Requisitos e as demais evidências (Cronograma, Lista de Riscos e Relatório de Custos). A discussão de **como** seria feito o projeto, ansiosamente aguardada pelos alunos, teve seu espaço. Neste momento a realização do projeto foi discutida sem se preocupar com prazos ou custos. Nessas discussões alguns dos riscos foram mencionados.

As tarefas que aparecem com uma estimativa de prazo no cronograma eram antes detalhadas na EDT. Esse caso é mais evidente no projeto Namoro Online onde, por exemplo, o item Telas do Sistema aparece apenas uma vez no cronograma, mas na verdade é um pacote de trabalho detalhado na EDT. Na EDT é possível especificar todas as telas que seriam desenvolvidas.

O cronograma se mostrou um instrumento efetivo na gestão de prazos. Mesmo durante a fase de planejamento ele apontou a impossibilidade de realizar tudo que constava na primeira versão do Relatório de Requisitos. Isso aconteceu nos três projetos estudados.

O Relatório de Custos serve para definir e controlar o orçamento de projeto. No caso dos alunos da Escola Técnica da UFPR não havia necessidade da preocupação com custos. Foi proposto que eles imaginassem todos os recursos necessários e fizessem um orçamento hipotético. Surgiram questões semelhantes às que se discutem nas empresas hoje em dia. Discutiu-se o valor do trabalho dos informatas, forma de pagamento (salário ou por hora trabalhada), amortização de custos de compiladores e equipamentos utilizados em vários projetos etc.

A Lista de Riscos e os três princípios (prototipação, adiantamento das atividades de risco e o processo em V) servem para focar os esforços de resolução dos riscos que ameaçam o sucesso do projeto. Os riscos já eram comentados durante a elaboração da EDT. As maiores preocupações das equipes se referiam à tecnologia de desenvolvimento para Web. O coordenador do projeto Protocolo Eletrônico tomou uma ação no sentido de minimizar os riscos, providenciando um curso sobre Java. A iniciativa do coordenador só foi possível ao saber claramente dos riscos apontados pela equipe através dessa evidência do método proposto.

Nos projetos Namoro Online e Boletim-Net foram utilizados diagramas da UML (Unified Model Language) antes da fase de codificação. Já no projeto de Protocolos Eletrônicos foram feitas uma análise de requisitos e uma definição da base de dados. Em ambos os casos o método cumpriu o objetivo de gerir pequenos projetos independentemente do ciclo de vida ou do processo de desenvolvimento adotado.

Além dos três projetos que foram minuciosamente estudados havia mais dois que não seguiram o método 5E+3P proposto. Dentre esses dois, um foi bem sucedido e outro não. Observou-se que esse projeto bem sucedido que não seguiu o método 5E+3P teve um alto nível de estresse ao se aproximar a data final de entrega, o que não aconteceu com os projetos bem sucedidos apoiados pelo método.

Nesse conjunto de cinco projetos, os que usaram o método foram bem sucedidos, e o que não foi bem sucedido não usou o método. Embora cinco projetos sejam uma amostra pequena, o experimento mostrou que o método influencia positivamente o sucesso do projeto.

O que se pode perceber é que as evidências foram suficientes para controlar custos, prazos e funcionalidades. As evidências, uma vez superada a falta de exemplos, se mostraram fáceis de entender e reproduzir. Os alunos, depois de passar pelas dificuldades iniciais, passam a acreditar na validade do método.

Essas afirmações vêm da observação do experimento. Para transformar esses fatos observados em dados científicos Lakatos e Marconi [Lakatos 1992] [Lakatos 1995] sugerem técnicas como entrevistas, questionário, formulário, teste etc. Estas técnicas são utilizadas de maneira sistemática com os envolvidos no experimento. Informalmente foram feitas entrevistas confirmando o observado, mas para formalizar os fatos optou-se por usar um questionário com os envolvidos confirmando ou refutando diretamente o observado.

Os alunos que responderam o questionário foram os que entregaram as evidências até o mês de setembro. Os alunos que não entregaram todas as evidências não foram incluídos na pesquisa. Acredita-se que para julgar o método deva-se conhecê-lo e vivenciá-lo em primeiro lugar.

Na Tabela 5.2 estão as perguntas e o percentual de respostas.

Tabela 5.2 – Questionário e Respostas da Escola Técnica da Universidade Federal do Paraná		
	Pergunta	Estatística Resposta
1	Tem-se por hipótese que a resolução dos riscos e o controle dos custos, prazos e funcionalidades aumentam a probabilidade de sucesso do projeto de software ?	7 (100%) – Sim 0 (0%) – Não
2	O Relatório de Requisitos serve para definir e controlar o conjunto das funcionalidades do projeto ?	6 (86%) – Sim 1 (14%) – Não
3	O Cronograma é um instrumento efetivo para controle de prazos ?	7 (100%) – Sim 0 (0%) – Não
4	O Relatório de Custos serve para definir e controlar o orçamento de projeto ?	7 (100%) – Sim 0 (0%) – Não
5	A Lista de Riscos e os 3 princípios (Prototipação, Adiantamento das atividades de risco e o processo em V) serve para focar os esforços de resolução dos riscos que ameaçam o sucesso do projeto ?	7 (100%) – Sim 0 (0%) – Não
6	Você usou mais algum documento, além dos sugeridos pelo método, para o planejamento e gerenciamento dos projetos quanto a custos, prazos e funcionalidade ?	1 (14%) – Sim 6 (86%) – Não
7	Tomando como base a sua experiência, como você classifica o método proposto quanto a complexidade ?	0 (0%) – Muito Simples 4 (57%) – Simples 3 (43%) – Regular 0 (0%) – Complexo 0 (0%) – Muito Complexo
8	Tomando como base a sua experiência, como você classifica o método proposto quanto a eficiência ?	0 (0%) – Ótimo 7 (100%) – Bom 0 (0%) – Regular 0 (0%) – Ruim 0 (0%) – Péssimo
9	Você recomendaria o uso do método a seus colegas ?	7 (100%) – Sim 0 (0%) – Não

5.3 CONTRIBUIÇÃO AOS MODELOS DA QUALIDADE.

O método 5E+3P pode atender total ou parcialmente a processos definidos pelos modelos da qualidade como ISO 9000-3 (International Standardization Organization) , CMM (Capability Maturity Model) e SPICE (Software Process Improvement and Capability dEtermination)

Veja nas tabelas 5.3, 5.4 e 5.5 os processos onde o método pode contribuir.

Tabela 5.3 – Processos ISO 9000-3 nos quais o Método 5E+3P contribui	
Parte	Descrição do Processo
Atividade do Ciclo de Vida	Cic2 – Especificação de Requisitos do Comprador
	Cic3 – Planejamento do Desenvolvimento
	Cic4 – Planejamento da Qualidade
Atividades de Suporte	Sup5 – Regras, Práticas e Convenções
	Sup6 – Ferramentas e Técnicas

Tabela 5.4 – Práticas Chaves CMM nas quais o Método 5E+3P contribui	
Nível	Descrição do Processo
Nível 2 – Repetitivo	1. Gerenciamento de Requisitos.
	2. Planejamento de Projetos.
	3. Acompanhamento de Projetos.
	4. Gerenciamento de Subcontratados

Tabela 5.5 – Processos SPICE nos quais o Método 5E+3P contribui	
Categoria	Descrição do Processo
Cliente Fornecedor	CUS.3 – Elicitação de Requisitos.
Engenharia	ENG.1.1 – Análise e Projeto do Sistema de Requisitos.
	ENG.1.2 – Análise de Requisitos de Software.
Suporte ao Ciclo de Vida	SUP.5 – Validação.
Ciclo de Vida Organizacional	MAN.2 – Gestão de Projeto
	MAN.4 – Gestão de Riscos.

5.4. CONCLUSÃO

O método 5E+3P no capítulo anterior foi criado a partir de um processo Dedutivo. O processo Dedutivo é um Método de Abordagem Científica.

O método 5E+3P foi testado pelo Método de Procedimento Científico conhecido como Estudo de Caso.

A experiência do Estudo de Casos foi realizada com o apoio da Escola Técnica da Universidade Federal do Paraná com alunos do último ano. Através do uso empírico do método notou-se uma falha na forma de divulgação por falta de exemplos. Corrigida essa deficiência, o método se mostrou aplicável e de boa aceitação para pequenos projetos de software.

Nos casos estudados, esse método simples se mostrou uma ferramenta eficiente para planejar e controlar custos, prazo e funcionalidade, independente do ciclo de vida adotado ou do processo de desenvolvimento de software.

O ambiente da experiência que fez a passagem da teoria à prática não foi o ideal. O método foi proposto para atender a pequenos projetos de software em organizações produtoras de software. Quando o utilizamos em projetos cuja principal finalidade é didática relevam-se algumas dificuldades enquanto se permitem outras. Os custos, no nosso estudo de casos, foram relevados, enquanto usar desenvolvedores que não dominam as ferramentas de desenvolvimento foi uma permissão que não cabe em uma organização profissional produtora de software. Desse modo pode-se dizer que o Método de Procedimento Científico foi feito com projetos concretos porém em um domínio que não era o esperado.

Embora a massa de projetos estudada seja pequena e o domínio dos experimentos não tenham sido o ideal, notou-se que o método influenciou positivamente para o sucesso dos projetos.

6.CONCLUSÃO

Na introdução desse trabalho foi citada a opinião do autor, baseada na experiência profissional, de como se deve fazer projetos. Um pequeno projeto de software, com um time de até 10 pessoas, precisa apenas de um pouco de organização e antecipação aos possíveis problemas. Essa foi a motivação que o levou a realizar esse trabalho. Essa intuição deveria se mostrar como fato comprovado ao final desse trabalho. Para chegar a essa conclusão o trabalho desenvolvido em seis:

Introdução: Na introdução é descrita a intenção do trabalho de se obter um método para planejamento e controle de pequenos projetos de software. Esse instrumento é de grande importância para gerentes de projetos de software, correlacionando custos, prazos e funcionalidades. Com isso tem-se uma melhor previsibilidade e um aumento da possibilidade de sucesso de um projeto.

Crise do Software: Neste capítulo são vistos relatos de problemas recorrentes em projetos de software. Ao final desse capítulo fica claro que os projetos costumam ultrapassar

prazos, custar acima do que foi orçado e as funcionalidades costumam não atender totalmente as necessidades do cliente.

Revisão Bibliográfica: Neste capítulo é feito um apanhado da bibliografia que trata dos pontos identificados no capítulo anterior. São vistos três modelos de qualidade de software: ISO 9000-3 (International Standardization Organization), CMM (Capability Maturity Model) e SPICE (Software Process Improvement and Capability dEtermination). É vista uma classificação geral de projetos proposta por Shenhar. Neste estudo Shenhar usa um modelo bidimensional onde uma dimensão é a incerteza tecnológica e a outra dimensão é o escopo do projeto. É visto ainda o que a literatura propõe para controlar funcionalidades, prazos e custos. Complementando o estudo, é feito um apanhado sobre gerência de riscos em um projeto de software. Esse capítulo é essencialmente expositivo.

Método 5E+3P: Neste capítulo é utilizado o material bibliográfico do capítulo anterior. O capítulo se inicia definindo o que é um pequeno projeto de software. Para isso é utilizada a classificação geral de projetos de Shenhar. O pequeno projeto de software é definido como sendo de média tecnologia, quanto à incerteza tecnológica, e de montagem e sistema, quanto ao escopo. Foram eleitos cinco documentos, recomendados por diversos autores, que servem para planejar e controlar de custos, prazos, funcionalidade e riscos. Esses documentos são chamados de evidências da gestão do projeto. Além desses documentos foram recolhidas três recomendações para diminuir o risco a que o projeto está exposto. Essas recomendações são chamadas de princípios pelo método 5E+3P. Ao final é proposto um Método para Gerenciamento de Pequenos Projetos de Software baseado em cinco evidências e três princípios. As evidências são: Relatório de Requisitos, EDT – Estrutura de Divisão do Trabalho, Cronograma, Relatório de Custos e Lista de Riscos. Os princípios são: Seguir o

modelo em V, fazer protótipo quando possível e antecipar no cronograma a resolução dos riscos.

Estudo de Casos: Essa experiência se passou com os alunos do último ano do curso de computação da Escola Técnica da Universidade Federal do Paraná. O método foi exposto com o material didático do anexo B. Depois complementado pelo material do anexo A. Os projetos realizados até setembro de 1999 foram utilizados neste estudo. Foram três projetos bem sucedidos cujas evidências se encontram no anexo C. Após essa experiência o método foi considerado simples, de bom desempenho e com boa aceitação. Essas afirmações podem ser comprovadas pelo questionário aplicado aos participantes da experiência.

O trabalho pronto, logicamente encadeado, esconde as dificuldades e as dúvidas do processo de criação. Alguns desses aspectos serão apresentados aqui. A começar pela estrutura do trabalho onde se deve expor um tema, levantar hipóteses, rever a bibliografia especializada, propor algo novo e testar. Essa estrutura veio da metodologia científica. A partir dela foi possível estruturar o trabalho e a sua condução.

Não foi possível validar o método com um rigor matemático. A área de gestão não é exata, depende de pessoas. Os métodos exatos não se aplicavam a comprovação da hipótese desse trabalho. Os procedimentos utilizados pelas ciências humanas, se valendo de estatísticas e questionários como ferramentas, se mostraram mais adequados.

Outro momento importante foi a aplicação do método. O método retirado da bibliografia estava pronto e bem descrito, porém a experiência junto aos alunos do segundo grau mostrou que a compreensão era mais fácil a partir de exemplos. Isso modificou o material didático e criou o anexo A deste trabalho. A compreensão do método pelos alunos e a discussão dos

prazos versus funcionalidades, usando as evidências do método como base, deram a certeza de que esse trabalho seria bem sucedido.

Evidências como EDT, Cronograma e Relatório de Custos vêm de vários estudos de projetos e são ferramentas consagradas. A existência dessas ferramentas é anterior a existência da engenharia de software e até mesmo a invenção do computador.

O estudo da gestão de riscos e da possibilidade de sucesso é dependente de uma massa de projetos significativa. Isso dificulta a comprovação das propostas. Melhor seria se existisse uma base de projetos significativa que seguiu o método e outra base significativa que não seguiu. Assim, seria possível quantificar a probabilidade de sucesso e o quanto ela seria afetada pelo método. Com muita dificuldade conseguimos três projetos seguindo o método e sete pessoas para responder o questionário. Essas sete pessoas passaram a conhecer o método. O que foi feito neste trabalho é a compilação das opiniões dessas pessoas quanto à influência do método no fator de risco.

O relatório de requisitos se mostrou uma excelente ferramenta para definição do alcance funcional do sistema. Essa evidência em especial pode ser melhorada com um estudo para sistematizar o encontro dos requisitos funcionais. Essa técnica é proposta por Jacobson em seu livro *OOSE (Object Oriented Software Engineering)* e denominada de técnica de Casos de Uso (*Use Cases*). Assim é possível melhorar os itens Modelos do Sistema e Requisitos Funcionais do Relatório de Requisitos a partir do estudo de Jacobson. O item Requisitos Não Funcionais também pode ser melhorado a partir de estudos feitos por Chung em seu livro *Dealing with Non-Functional Requirements: Three Experimental Studies of a Process-Oriented Approach*.

Na introdução desse trabalho foram colocadas três perguntas que agora têm respostas:

Seria possível se obter um método simples derivado da literatura científica? Sim, uma possibilidade é o método que se encontra no capítulo Método 5E+3P e é resultado da bibliografia apresentada na Revisão Bibliográfica.

Caso seja possível a existência de tal método, seria ele aplicável? Sim, o Método Proposto foi aplicado em três projetos com os alunos da Escola Técnica da Universidade Federal do Paraná com sucesso.

Se aplicável, seria bem aceito pelos desenvolvedores? Sim, a prova disso é que 100% dos entrevistados responderam afirmativamente a seguinte pergunta do questionário: ***Você recomendaria o uso do método a seus colegas?***

Espero que esse trabalho seja também o ponto de partida para outros estudos científicos de interesse e relevância para a comunidade científica e para a sociedade em geral.

REFERÊNCIA BIBLIOGRÁFICA

- [Alsop 1996] ALSOP, Stewart. **The Trouble With Software Is... It Sucks.** Fortune. 10 de junho de 1996, em <http://www.fortune.com>.
- [Antonioni 1995] ANTONIONI, José A; ROSA, Newton Braga, ASSUMPÇÃO FILHO, Milton Mira de. **Qualidade em Software.** Makron Books. 1995.
- [Bello 1998] BELLO, José Luiz de Paiva. Metodologia Científica: Estrutura de Apresentação do Trabalho. In: Pedagogia On Line, 1998. (<http://home.iis.com.br/~jbello/bestrutu.htm>).
- [Birman 1999] BIRMAN, Kenneth P.; RENESSE, Robbert van. **Software Reliable Networks.** <http://www.sciam.com/0596birman.html>. Fevereiro 1999.
- [Boehm 1981] BOEHM, Barry W. **Software Engineering Economics.** Englewood Cliffs, Prentice-Hall, 1981. 767 p.
- [Booch 1995] BOOCH, Grady. **Object Solutions – Managing the Object-Oriented Project.** 4^a ed. Santa Clara, Addison-Wesley, 1995. 323p.

- [Brodman 1994] BRODMAN, Judith G. ; JOHNSON, Donna. L. What Small Businesses and Small Organizations Say About the CMM. In: 16TH INTERNATIONAL CONFERENCE ON SOFTWARE ENGINEERING (1.:1994:s.l.). **Anais.** s.l.:IEEE Computing Society Press, 1994. P.331-40. Association for Computing Machinery Library.
- [Castor e Suga 1989] CASTOR, Belmiro V. J. e SUGA, Nelson, Planejamento e Ação Planejada: O Dificil Binômio. **Revista de Administração Pública**, Rio de Janeiro: Fundação Getúlio Vargas, v. 22, n. 1, p. 102-122, janeiro/março 1989.
- [Diaz 1995] DIAZ, Ramon. **Metodologia e Ferramentas de Orientação a Objetos.** In: 4^o CONGRESSO NACIONAL SOBRE DESENVOLVIMENTO DE SISTEMAS E ORIENTAÇÃO A OBJETO (2.:1995:São Paulo). **Anais.** São Paulo: Mantel: 1995. 50p.
- [Fairley 1985] FAIRLEY, Richard. **Software Engineering Concepts.** 2^a ed. New York, McGraw-Hill, 1985. 364p.
- [Ferreira 1986] FERREIRA, Aurélio Buarque de Holanda. **Novo Dicionário da Língua Portuguesa.** 2^aEdição. Rio de Janeiro, Editora Nova Fronteira, 1986. 1838p.
- [Hubbard 1998] HUBBARD, Douglas. **Hurdling Risk.** CIO Enterprise Magazine. 15 de junho de 1998, em <http://www.cio.com> .
- [Humphrey 1989] HUMPHREY, Watts S. **Managing the Software Process.** 1^oed. Massachusetts, Addison-Wesley, 1989. 494p.
- [ISO 15504] PDTR ISO 15504. In: **Software Quality Institute**, Novembro 1996. (<http://www.sqi.gu.au/spice>)
- [Koontz 1978] KOONTZ, Harold, Management theory, science, and approaches. In: **Encyclopedia of Professional Management**, New York: McGraw-Hill, 1978, v1 , pp.650-664.

- [Lakatos 1992] LAKATOS, Eva Maria, MARCONI, Maria Andrade **Metodologia do Trabalho Científico**. 4^a ed. São Paulo, Editora Atlas, 1992. 216p.
- [Lakatos 1995] LAKATOS, Eva Maria, MARCONI, Maria Andrade. **Metodologia Científica**. 2^a ed. São Paulo, Editora Atlas, 1995. 256p.
- [Lyu 1996] LYU, Michel R. **Handbook of Software Reliability Engineering**. 2^a ed. New York, McGraw-Hill, 1996. 850p.
- [Ould 1990] OULD, Martyn A. **Strategies for Software Engineering – The Management of Risk and Quality**. 1^a ed. Grã Betanha, John Wiley & Sons Ltd, 1990. 243p.
- [Rational 1997] RATIONAL. **Análise e Projeto Orientados a Objetos Usando UML**. Apostila. 1997.
- [Rosa 1997] ROSA, Paola Goulart. **Modelo de Avaliação de Processo de Software para Pequena Empresa**. Dissertação de Mestrado. Instituto de Ciências Matemáticas de São Carlos. Universidade de São Paulo, 1997, 90p.
- [Sauter 1999] SAUTER, Vick L. Intuitive Decision-Making, **Communications of The ACM**, volume 42, number 6, June 1999, pp. 109-115.
- [SEI-CMU 1994] SOFTWARE ENGINEERING INSTITUTE – CARNEGIE MELLON UNIVERSITY. **The Capability Maturity Model**. Addison-Wesley. 1994
- [Shenhar 1998] SHENHAR, Aaron J. **From Theory to Practice: Toward a Typology of Project-Management Styles**. IEEE Transactions on Engineering Management. Vol. 45, n. 1, p. 33-48, February 1998.
- [Sommerville 1990] SOMMERVILLE, Ian; DUFFIE, N. A., **Software Engineering**. 3^a ed. Wokinghan, Addison-Wesley, 1990. 653 p.

[Weber 1994] WEBER, Kival Chaves; MILLET, Paulo Barreira; BRANDÃO JUNIOR, Dorgival. **Qualidade e Produtividade em Software**. Editado por QA&T Consultores Associados LTDA. 1994.

ANEXOS

ANEXO A – EVIDÊNCIAS DE EXEMPLO DO MÉTODO 5E+3P.

Anexo A.1 – Relatório de Requisitos.

Anexo A.2 – EDT – Estrutura de Divisão do Trabalho.

Anexo A.3 – Cronograma.

Anexo A.4 – Relatório de Custos.

Anexo A.5 – Lista de Riscos

Anexo B – Explicação do Método para os Alunos da Escola Técnica da UFPR

Anexo C – Projetos Utilizados no Estudo de Casos

ANEXO C.1 – NAMORO ONLINE

ANEXO C.2 – BOLETIM NET

ANEXO C.3 – PROTOCOLO ELETRÔNICO

ANEXO D – QUESTIONÁRIOS

ANEXO A

EVIDÊNCIAS DE EXEMPLOS DO MÉTODO PROPOSTO

ANEXO A.1

RELATÓRIO DE REQUISITOS

Projeto de Divulgação de Notas

1) Introdução

Este sistema servirá para divulgação de notas dos alunos de uma escola. O sistema funcionará em uma rede local, possibilitando que os alunos tenham acesso as notas através de um software de acesso.

Com o uso deste sistema, pretende-se limpar os murais de notas, feitos com papeletas, comuns em finais de semestres.

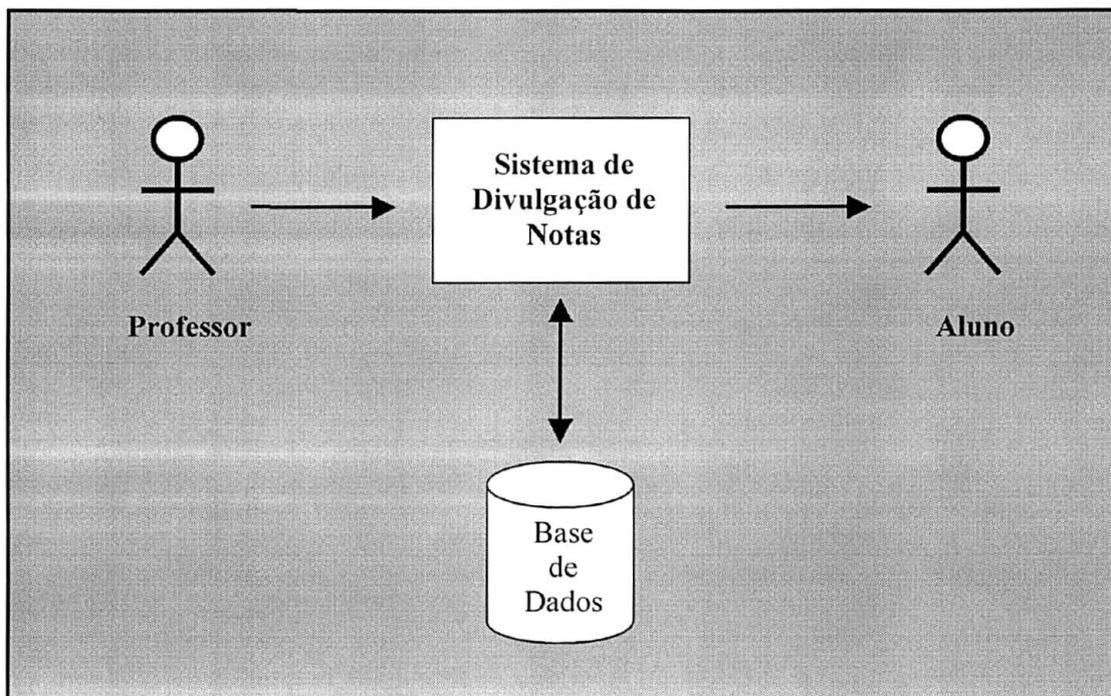
O sistema será informal como os murais. Cabendo aos professores dispor as notas e comentários e aos alunos será dada apenas a permissão de consulta.

2) Glossário

Delphi – Sistema de desenvolvimento de programas.

Classe – Conjunto de alunos que terão têm aula de uma determinada disciplina com um determinado professor.

3) Módulos do Sistema



4) Definição de Requisitos Funcionais

Dois serão os usuários do Sistema de Divulgação de Notas, Os professores e os alunos. Administração do sistema ficará a cargo dos professores, enquanto os alunos poderão apenas consultar as notas.

Embora o sistema permita que qualquer professor altere qualquer nota, espera-se que apenas o professor daquela classe específica manipule suas notas.

As funções disponíveis no sistemas são:

4.1) Cadastro de Aluno

Qualquer professor poderá cadastrar um aluno, determinando a sua senha de acesso.

4.2) Cadastro de Professor

Qualquer professor poderá cadastrar um outro professor, determinando a sua senha de acesso.

4.3) Inserção de Notas

Qualquer professor poderá registrar, na tabela de notas, uma nota e um comentário para uma combinação de aluno, classe.

4.4) Verificação de Notas pelos Alunos

Aluno terá acesso a um relatório de notas e comentários contendo todas as classes nas quais ele é mencionado.

4.5) Verificação de Notas pelo Professor.

O professor terá acesso a um relatório de notas e comentários de todos os alunos de uma determinada classe.

5) Requisitos não Funcionais.

A identificação do aluno no sistema será feito exclusivamente pela sua matrícula, que é um campo numérico.

O professor será identificado no sistema por um campo alfanumérico.

O sistema será desenvolvido em Delphi.

O banco de dados ficará em uma área de diretório da rede comum aos usuários. O formato utilizado será o Paradox.

6)Evolução do sistema

O sistema poderá evoluir para divulgar média e desvio padrão das notas em função de uma classe, um professor ou um aluno.

O sistema pode ser estendido também para divulgar o número de faltas de cada aluno.

7)Especificação de Requisitos.

7.1)Mapa do menu do sistema.

- Professor (Pede Senha de Acesso)
 - Cadastra Aluno
 - Cadastra Professor
 - Tabela de Notas
 - Varifica Notas por Classe

- Aluno(Pede Senha de Acesso)
 - Tabela de Notas por Aluno

7.2) Relatório de Notas

No relatório deve conter as informações:

- Classe
- Aluno
- Nota
- Comentário

ANEXO A.2

EDT - ESTRUTURA DE DIVISÃO DO TRABALHO

EDT - Estrutura de Deposição do Trabalho

Id	Task Name
1	Projeto
2	Protótipo
3	Gerar um Relatório Teste
4	Definição da Base de Dados
5	Definir Tabela Professor Senha
6	Definir Tabela Aluno Senha
7	Definir Tabela das Classes
8	Documentação
9	Módulo Menu
10	Módulo Cadastro Alunos
11	Módulo Cadastro de Professores
12	Módulo Relatório de Notas para Alunos
13	Módulo Relatório de Notas para Professores
14	Codificação
15	Menu
16	Cadastro de Alunos
17	Manutenção Na tabela de Alunos
18	Manutenção Na tabela de Professores
19	Módulo Relatório de Notas para Alunos
20	Módulo Relatório de Notas para Professores
21	Testes
22	Menu
23	Manutenção Na Tabela De Alunos
24	Manutenção na tabela de Professores
25	Relatório de Notas para Alunos
26	Relatório de Notas para Professores
27	Integração
28	Codificação
29	Testes
30	Implantação

ANEXO A.3

CRONOGRAMA

Id	Task Name	Duração	Nomes de reat	Predecessor	Início	Jul 99					18 Jul 99					25 Jul 99					01 Ago 99					08 Ago 99									
						S	T	Q	Q	S	S	D	S	T	Q	Q	S	S	D	S	T	Q	Q	S	S	D	S	T	Q	Q	S	S	D	S	T
1	Projeto	24 dias			Seg 12/07/99	[Gantt bar for Task 1]																													
2	Protótipo	5 dias			Seg 12/07/99	[Gantt bar for Task 2]																													
3	Gerar um Relatório Teste	5 dias	Analista 1;Analista 2		Seg 12/07/99	[Gantt bar for Task 3]																													
4	Definição da Base de Dados	3 dias	Analista 2		Seg 19/07/99	[Gantt bar for Task 4]																													
5	Definir Tabela Professor Senha	1 dia	Analista 2	2	Seg 19/07/99	[Gantt bar for Task 5]																													
6	Definir Tabela Aluno Senha	1 dia	Analista 2	5	Ter 20/07/99	[Gantt bar for Task 6]																													
7	Definir Tabela das Classes	1 dia	Analista 2	6	Qua 21/07/99	[Gantt bar for Task 7]																													
8	Documentação	5 dias			Seg 19/07/99	[Gantt bar for Task 8]																													
9	Módulo Menu	1 dia	Analista 1	2	Seg 19/07/99	[Gantt bar for Task 9]																													
10	Módulo Cadastro Alunos	1 dia	Analista 1	9	Ter 20/07/99	[Gantt bar for Task 10]																													
11	Módulo Cadastro de Professores	1 dia	Analista 1	10	Qua 21/07/99	[Gantt bar for Task 11]																													
12	Módulo Relatório de Notas para Alunos	1 dia	Analista 2	7	Qui 22/07/99	[Gantt bar for Task 12]																													
13	Módulo Relatório de Notas para Professores	1 dia	Analista 2	12	Sex 23/07/99	[Gantt bar for Task 13]																													
14	Codificação	8 dias			Qui 22/07/99	[Gantt bar for Task 14]																													
15	Menu	2 dias	Analista 1	11	Qui 22/07/99	[Gantt bar for Task 15]																													
16	Cadastro de Alunos	2 dias	Analista 1	15	Seg 26/07/99	[Gantt bar for Task 16]																													
17	Manutenção Na tabela de Alunos	2 dias	Analista 1	16	Qua 28/07/99	[Gantt bar for Task 17]																													
18	Manutenção Na tabela de Professores	2 dias	Analista 1	17	Sex 30/07/99	[Gantt bar for Task 18]																													
19	Módulo Relatório de Notas para Alunos	2 dias	Analista 2	13	Seg 26/07/99	[Gantt bar for Task 19]																													
20	Módulo Relatório de Notas para Professores	2 dias	Analista 2	19	Qua 28/07/99	[Gantt bar for Task 20]																													
21	Testes	5 dias			Sex 30/07/99	[Gantt bar for Task 21]																													
22	Menu	1 dia	Analista 1	18	Ter 03/08/99	[Gantt bar for Task 22]																													
23	Manutenção Na Tabela De Alunos	1 dia	Analista 1	22	Qua 04/08/99	[Gantt bar for Task 23]																													
24	Manutenção na tabela de Professores	1 dia	Analista 1	23	Qui 05/08/99	[Gantt bar for Task 24]																													
25	Relatório de Notas para Alunos	1 dia	Analista 2	20	Sex 30/07/99	[Gantt bar for Task 25]																													
26	Relatório de Notas para Professores	1 dia	Analista 2	25	Seg 02/08/99	[Gantt bar for Task 26]																													
27	Integração	4 dias	Analista 1;Analista 2		Sex 06/08/99	[Gantt bar for Task 27]																													
28	Codificação	2 dias	Analista 1;Analista 2	24;26	Sex 06/08/99	[Gantt bar for Task 28]																													
29	Testes	2 dias	Analista 1;Analista 2	28	Ter 10/08/99	[Gantt bar for Task 29]																													
30	Implantação	1 dia	Analista 1	27	Qui 12/08/99	[Gantt bar for Task 30]																													

ANEXO A.4

RELATÓRIO DE CUSTOS

Relatório de Custos

Jaylson Teixeira

22/08/99

<i>Projeto</i> Divulgação de Notas
<i>Data: 22/08/1999</i>

Descrição	Preço Unitário	Quantidade	Total
1. Hardware, Software e Outros Materiais			
1.1. Ambiente Delphi (US\$ 3.000,00)	R\$ 6.000,00	1	R\$ 6.000,00
2. Viagens			
3. Treinamento			
3.1. Livro: BORLAND DELPHI 4 CURSO COMPLETO. Autor: LEAO, MARCELO. Editora: AXCEL BOOKS	R\$ 149,00	1	R\$ 149,00
4. Esforços			
4.1. Analista Junior			
4.2. Analista Pleno			
TOTAL			R\$ 6.149,00

ANEXO A.5

LISTA DE RISCOS

Lista de Riscos

Jaylson Teixeira

15/08/99

Projeto
Divulgação de Notas

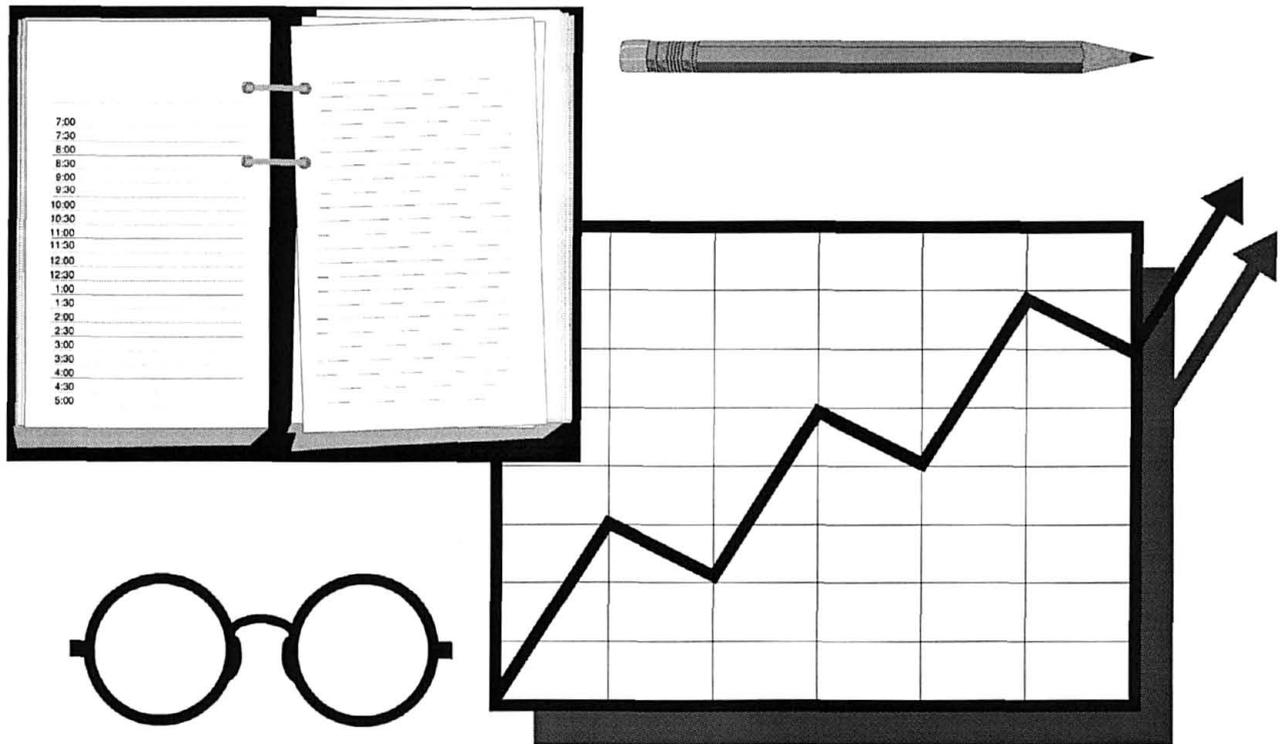
Identificação do Risco	
Relatório via Quick Report	
<i>Data Abertura</i>	<i>Descrição do Problema</i>
02/08/1999	Ninguém envolvido com o projeto utilizou o Quick Report antes
<i>Data Fechamento</i>	<i>Solução Proposta</i>
09/08/1999	Vamos procurar ajuda com pessoas que já usaram o Quick Report antes.

Identificação do Risco	
Rede Local	
<i>Data Abertura</i>	<i>Descrição do Problema</i>
09/08/1999	Para que o Projeto funcione a rede local deve está instalada a tempo
<i>Data Fechamento</i>	<i>Solução Proposta</i>
	Apesar da área de suporte garantir as datas, estaremos acompanhando o cronograma de implantação com o coordenador da área de suporte.

ANEXO B

EXPLICAÇÃO DO MÉTODO PARA OS ALUNOS DA ESCOLA TÉCNICA DA UFPR

**MÉTODO PARA
GERENCIAMENTO DE
PEQUENOS PROJETOS DE SOFTWARE**



Jaylson Teixeira

Curitiba

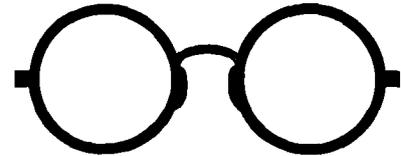
Junho de 1999

Método para Gerenciamento de Pequenos Projetos de Software

Jaylson Teixeira

09/06/1999

ÍNDICE



1. CRISE DO SOFTWARE.....	3
2. O MÉTODO.....	5
3. RELATÓRIO DE REQUISITOS.....	7
4. ESTRUTURA DIVISÃO DO TRABALHO.....	10
5. CRONOGRAMA.....	13
6. RELATÓRIO DE CUSTOS	15
6.1. TÉCNICAS DE ESTIMATIVAS DE CUSTO	15
6.1 ABORDAGEM.....	15
6.3. COMPONENTES DO CUSTO.....	16
7. LISTA DE RISCOS	17
8. MODELO DE PROCESSO EM V SIMPLIFICADO.....	18

Método para Gerenciamento de Pequenos Projetos de Software

Jaylson Teixeira

09/06/1999



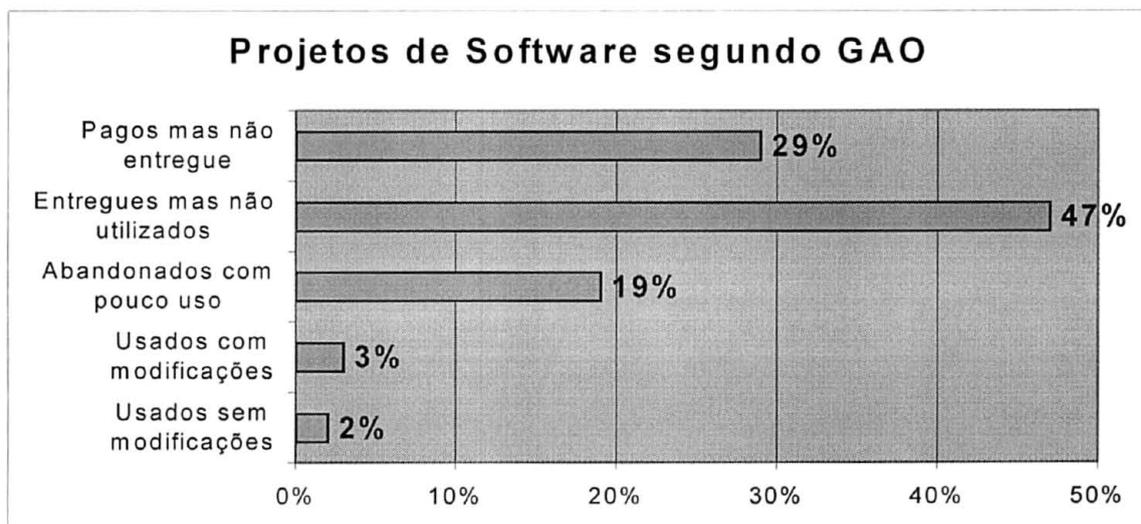
1. CRISE DO SOFTWARE

São inúmeros os relatos de projetos ultrapassando prazos, gastando acima do orçamento e gerando software de qualidade duvidosa.

Dados que ilustram o insucesso dos projetos podem ser encontrados na revista American Programmer de 1995 e no GAO – General Accounting Office de 1982 (GAO é equivalente ao nosso TCU – Tribunal de Contas da União).

- 31% dos projetos serão cancelados antes de serem completados.
- 53% dos projetos ultrapassarão seus custos iniciais estimados em 189%
- Companhias americanas e agências governamentais gastarão 81 bilhões em projetos de software cancelados.

American Programmer, July 95



Método para Gerenciamento de Pequenos Projetos de Software

Jaylson Teixeira

09/06/1999

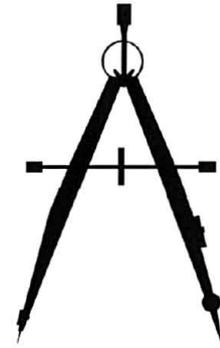
O método a ser apresentado tem como objetivo balizar o compromisso que um projeto tem entre o que vai ser feito (funcionalidade), quanto vai custar (orçamento) e quanto tempo vai levar (prazo). O planejamento e o gerenciamento de um projeto tem como objetivo controlar as variáveis de custo, prazo e funcionalidade.

Método para Gerenciamento de Pequenos Projetos de Software

Jaylson Teixeira

09/06/1999

2. O MÉTODO



O método é destinado a pequenos projetos de software. Tem como objetivo aumentar a probabilidade de sucesso do projeto resolvendo riscos e controlando os custos, prazos e funcionalidades do projeto. É um método para planejar e controlar o projeto com relação a estas variáveis.

O planejamento do sistema com relação a arquitetura e a implementação, a partir de um determinado conjunto de requisitos, não diz respeito a este método.

O planejamento é particular de cada projeto. Depende da cultura e informação de que dispõem os projetistas. O método vê o processo de planejamento como uma caixa preta, onde sejam quais forem as escolhas feitas pelos projetistas deve-se gerar evidências. Por isso este método está definido em função de 5 evidências que devem ser geradas levando em conta 3 princípios. Assim tem-se um padrão sem reprimir o processo de criação inerente planejamento.

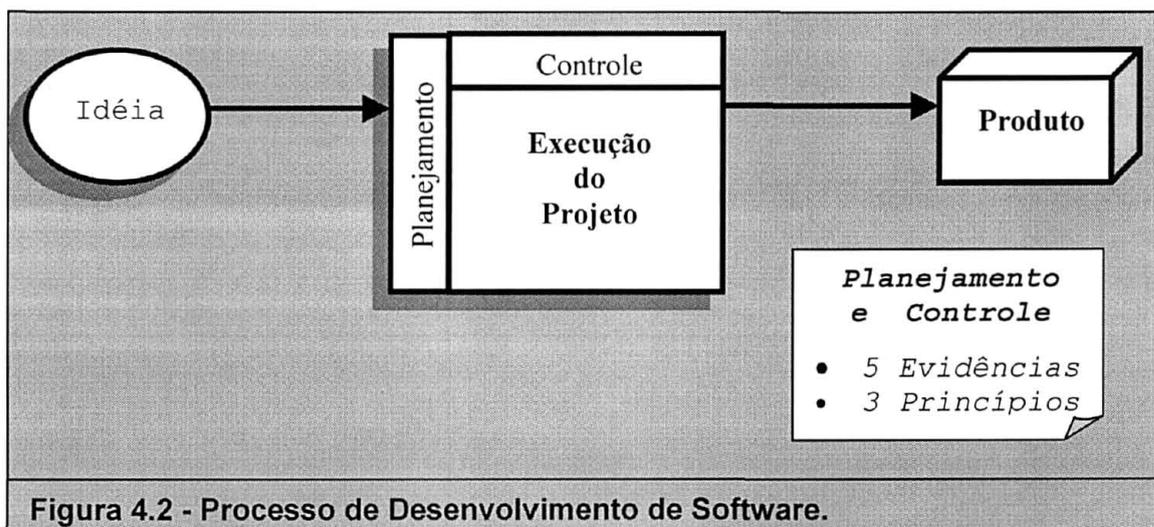


Figura 4.2 - Processo de Desenvolvimento de Software.

Método para Gerenciamento de Pequenos Projetos de Software

Jaylson Teixeira

09/06/1999

Deve-se começar o planejamento pelo relatório de requisitos, definindo o **que** deve ser feito. Depois faz-se a EDT(Estrutura de Divisão do Trabalho), definindo o **como** deve ser feito. A partir do EDT pode-se gerar o Cronograma e Relatório de Custos e Lista de Riscos.

Em anexo, tem-se exemplos de cada um dessas evidências facilitando o entendimento do texto descritivo.

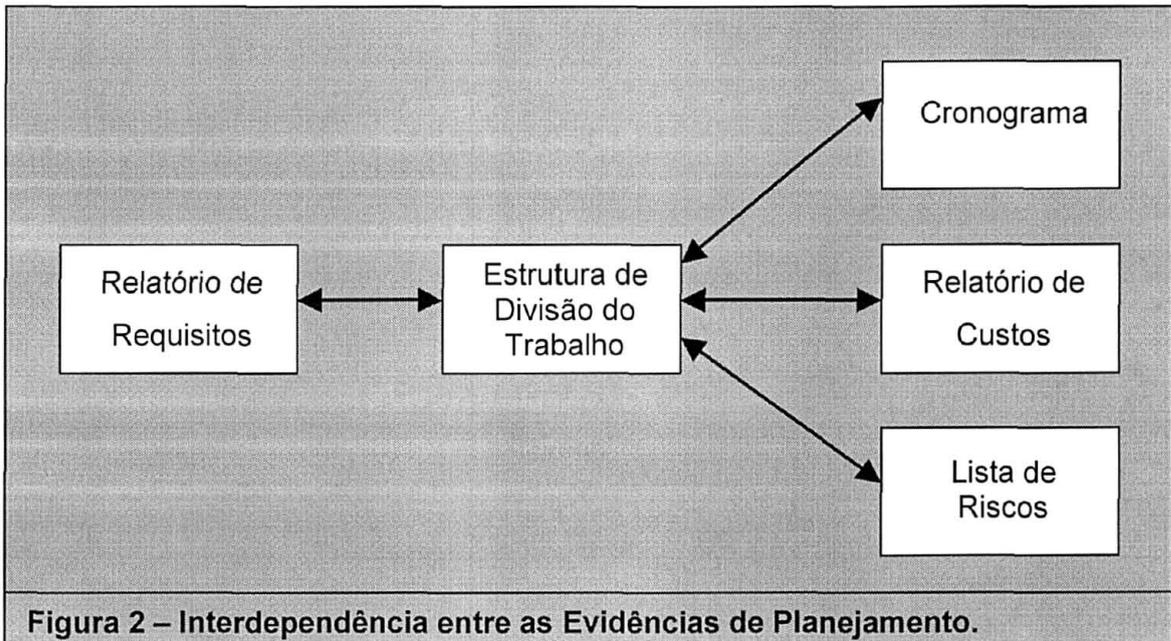


Figura 2 – Interdependência entre as Evidências de Planejamento.

As 5 evidências que caracterizam o planejamento são:

- Relatório de Requisitos
- Estrutura de Divisão do Trabalho
- Cronograma
- Relatório de Custos
- Lista de Riscos

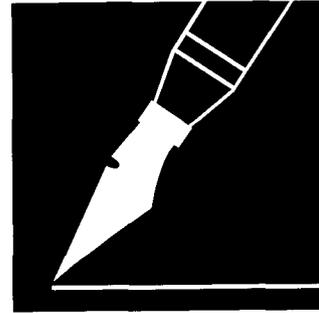
Os 3 princípios que deve-se observar nas evidências são:

- Utilização do Processo em V Simplificado
- Uso de protótipo quando possível
- Resolução de riscos no início do projeto.

Método para Gerenciamento de Pequenos Projetos de Software

Jaylson Teixeira

09/06/1999



3. Relatório de Requisitos.

No documento de requisitos, Sommerville recomenda a existência de uma definição de requisitos e uma especificação de requisitos.

Definição de requisitos é um texto descritivo auxiliado por diagramas que serve para especificar os serviços esperados do sistema a ser desenvolvido e as condições sob as quais será operado. A linguagem utilizada é familiar ao usuário. Deve refletir o que o usuário deseja do sistema. Sommerville cita como exemplo a definição de requisito:

- *Requisito 1) O software deve prover meios para representação e acesso de arquivos externos criados por outras ferramentas.*

Especificação de Requisitos é um documento estruturado que consta os detalhes dos serviços do sistema. Também é chamado de especificação funcional. Serve como um contrato entre o desenvolvedor e o comprador do projeto. Sommerville cita como exemplo as seguintes especificações de requisito:

- *Requisito 1.1) O usuário terá facilidades para definir o tipo de arquivo externo.*
- *Requisito 1.2) Cada tipo arquivo externo estará associado a uma ferramenta para abrir o arquivo.*
- *Requisito 1.3) Cada tipo de arquivo externo será representado por um ícone específico na tela do usuário.*
- *Requisito 1.4) Facilidades devem ser disponibilizadas para que o usuário associe um ícone ao tipo de arquivo externo.*
- *Requisito 1.5) Quando o usuário seleciona um ícone de arquivo externo na sua tela, a ferramenta associada àquele tipo de arquivo é utilizada para abrir o arquivo externo associado ao ícone.*

Método para Gerenciamento de Pequenos Projetos de Software

Jaylson Teixeira

09/06/1999

Em um projeto pequeno, a definição de requisitos e a especificação de requisitos podem está em um único documento, chamado documento de requisitos. O documento de requisitos não é um desenho do software, por isso não deve especificar **como** deve ser feito o software mas sim **o que** o software tem de fazer. Segundo Sommerville, Heringer em 1980 sugere seis recomendações para o documento de requisitos:

- Deve especificar o comportamento externo do sistema.
- Deve especificar as restrições para implementação.
- Deve ser fácil de alterar.
- Deve servir de referência para os matenedores do sistema.
- Deve registrar o ciclo de vida adotado para o sistema.
- Deve caracterizar respostas inaceitáveis do sistema e eventos indesejáveis.

A técnica criada por Jacobson conhecida como Casos de Uso pode ser utilizada para fazer um Modelo do Sistema e levantar as funcionalidade.

Na técnica de Casos de Uso começa-se identificando quais os atores do seu sistema. Atores são tipos diferentes de usuários vão interagir com o sistema. Atores também podem ser outros sistemas que interagem com o que se está desenvolvendo. Um bloco representado o sistema com os atores em volta já é uma boa representação do modelo do sistema. Para melhorar o modelo do sistema, pode-se discriminar os componentes ou subsistemas internos e posicionar os atores segundo a sua distribuição geográfica.

A funcionalidade do sistema poder ser obtida perguntando para cada ator de que maneira ele interage com o sistema. Cada função levantada está relacionada com um caso de uso.

Sommeville também sugere um roteiro para o documento de requisitos composto por introdução, glossário, modelos do sistema, definição de requisitos funcionais, definição de requisitos não funcionais, evolução do sistema, especificação de requisitos. A descrição desse roteiro está na tabela 1 e em anexo tem-se um exemplo do Relatório de Requisitos do Projeto de Divulgação de Notas.

Método para Gerenciamento de Pequenos Projetos de Software

Jaylson Teixeira

09/06/1999

Tabela 1 – Descrição do Documento de Requisitos	
Capítulo	Descrição
Introdução	Justifica a necessidade do sistema. Descreve rapidamente suas funções e ilustra seu relacionamento com outros sistemas. Contextualiza o software nos objetivos estratégicos e de negócio da organização cliente.
Glossário	Este espaço é para ser utilizado para explicar termos técnicos que o leitor comum não esteja familiarizado.
Modelos do Sistema	Neste capítulo é mostrado a relação dos componentes do sistema e o ambiente do negócio de forma gráfica. Uma figura costuma ser suficiente para representar o sistema com um nível de abstração alto que sirva para contextualizar os requisitos. Os usuários do sistema devem ser representados.
Definição dos Requerimentos Funcionais	Os serviços fornecidos aos usuários e a outros sistemas devem ser descritos neste capítulo. A definição devem ser um textos descritivos para cada função do sistema. A descrições devem não devem especificar a tecnologia que será utilizada.
Definição dos Requerimentos Não Funcionais	As restrições imposta ao software e suas funções devem constar deste capítulo. Restrições quanto a memória, especificação de desempenho e representação dos dados podem ser incluídos. Produtos, processos padrões e tecnologia são descritos.
Evolução do Sistema	Neste capítulo é feita uma revisão do contexto do negócio. Procura-se antecipar as mudanças das necessidades do usuário, a natural evolução do sistema e do ambiente, com vistas para as alterações e melhorias futuras.
Especificação de Requisitos	Neste capítulo os requerimentos funcionais e não funcionais devem ser detalhados.

Método para Gerenciamento de Pequenos Projetos de Software

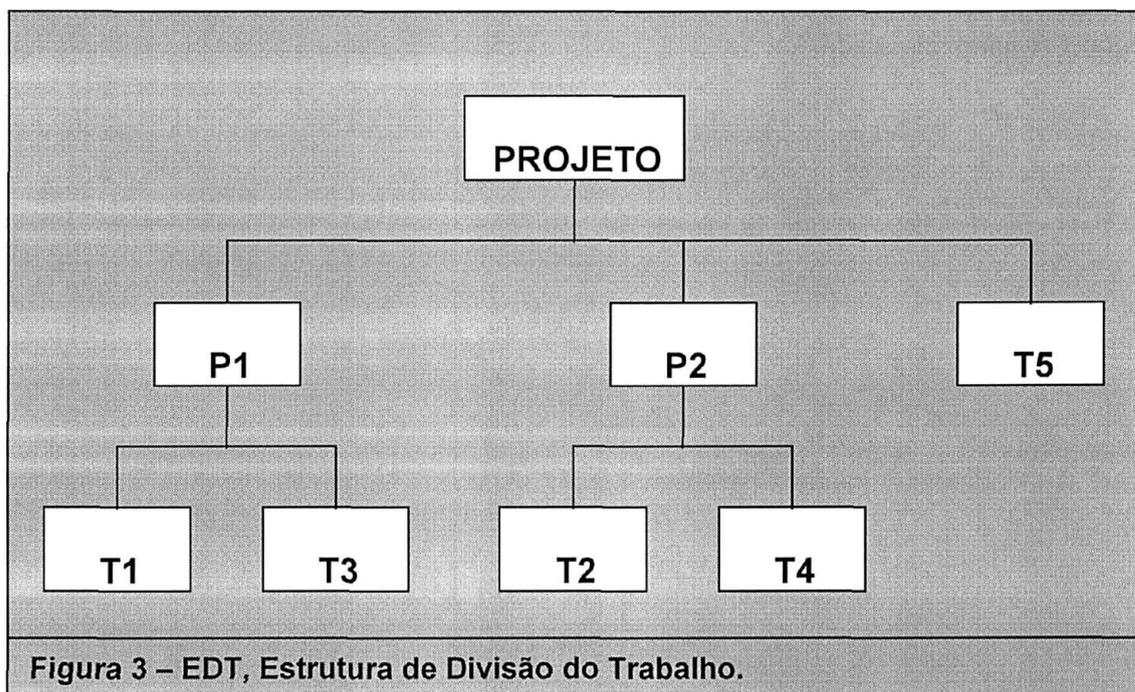
Jaylson Teixeira

09/06/1999



4. ESTRUTURA DIVISÃO DO TRABALHO.

EDT – Estrutura de Divisão do Trabalho é uma tradução para *WBS – Work Breakdown Structure*. Na prática, EDT é uma lista de atividades com itens e sub-itens que caracteriza uma árvore a partir de um nó comum que é o projeto. Consiste de uma série de tarefas agrupadas em pacotes de trabalho em forma de árvore. Veja a figura 1.



Na figura 1, existe um EDT com três níveis. No nível 2 tem-se três itens que são P1, P2 e T5. A partir de um item de qualquer nível podemos caracterizar um pacote de trabalho. P2 é um pacote de trabalho composto pelas tarefas T2 e T4. T5 é um pacote de

Método para Gerenciamento de Pequenos Projetos de Software

Jaylson Teixeira

09/06/1999

trabalho composto apenas pela tarefa T5. PROJETO é um pacote de trabalho composto pela tarefa T5 e os pacotes P1 e P2. Os pacotes de trabalho tem uma carga de trabalho associada. A carga de trabalho do pacote é igual a soma das cargas de trabalho dos subpacotes que o compõem. Assim, tomando a figura 1 como exemplo, se a tarefa T2 tem uma carga de trabalho igual a 15 dias e a tarefa T4 tem uma carga de 15 dias o pacote P2 terá a carga de 30 dias.

Se o projeto total for muito grande, é comum se quebrar o projeto em subprojetos. Neste caso um determinado pacote se torna um projeto independente.

O uso de EDT é muito freqüente na indústria [Ould 1990] [Shenhar 1998]. Em 25 de março de 1993 a EDT é descrito pelo estabelecimento de defesa americano no documento *Military Standard (MIL-STD) 881B*. Neste documento le-se:

Uma Estrutura de Divisão do Trabalho é uma árvore orientada ao produto, composta por hardware, software, serviços, dados e facilidades... EDT mostra e define o(s) produto(s) a ser(em) desenvolvido(s) ou produzido(s) e relaciona os elementos do trabalho a serem realizados. Relacionamento dos elementos uns com os outros e com o produto final.

Para elaborar uma EDT, costuma-se iniciar por uma aborgagem orientada a tarefa, com uma lista superficial de a fazeres. O projeto é uma coleção de tarefas. Essas tarefas são agrupadas logicamente por diferentes critérios dependendo da escolha do projetista. A divisão lógica pode ser fases do projeto, áreas funcionais, produtos principais etc.

O projetista deve tentar fazer com que a EDT atinja as seguintes metas:

- Ser compatível com a forma que o trabalho será executado e com os custos e prazos que serão gerenciados.
- Dar visibilidade ao esforço de resolução de riscos.
- Mapear o modelo do processo de processo utilizado.
- Promover clara divisão de responsabilidades para gerentes e líderes de tarefas.
- Prover uma estrutura para obtenção de dados históricos do processo.

Método para Gerenciamento de Pequenos Projetos de Software

Jaylson Teixeira

09/06/1999

- Fazer sentido para os executores do projeto e os coletores de dados. Os dados coletados são de desempenho e financeiros.

Ould [Ould 1990] sugere que os pacotes de atividades compostos por mais de uma tarefa usem rótulos que indiquem a classe das tarefas (exemplo: Relatórios, Definição de Requisitos), enquanto as tarefas devem iniciar com verbos de indiquem o que devem ser feito (exemplo: Fazer Relatório, Definir Requisitos).

A completude é muito importante para a EDT. Na EDT deve constar tudo que deve ser feito no projeto. Caso isso não aconteça de maneira satisfatória o planejamento do projeto estará comprometido. Se uma atividade não consta do EDT, ela deve ser realizada em tempo zero e a custo zero.

Cada atividade na EDT deve ser dimensionada quanto ao esforço necessário para executa-las. Existe um certo consenso de que os tempos das tarefas são melhor estimados quando feitos por semanas [Ould 1990] [Sommerville 1990]. As tarefas devem ter prazos de 5 a 15 dias e caso a organização exija apresentação em meses ou horas, deve-se converter após finalizado o trabalho de estimativa de prazos. Esta estimativa de prazos é a parte mais subjetiva do planejamento. Os tempos de execução das tarefas influenciam bastante no custo da mão-de-obra e conseqüentemente no custo final do projeto. Ould [Ould 1990] recomenda com bastante ênfase que se utilize pessoas com experiência no tipo de aplicação e no negócio ao qual a aplicação servirá.

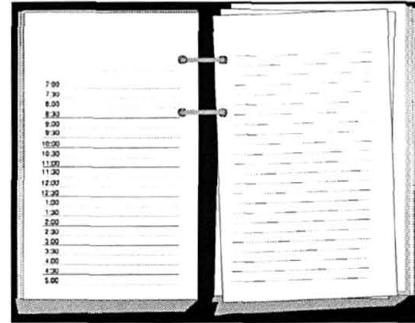
Neste trabalho serão apresentadas técnicas e abordagens para definição de custos que envolvem estimativa de prazos.

A EDT é a evidência intermediária entre o relatório de requisitos e as evidências cronograma, relatório de custos e lista de riscos. A primeira idéia ao fazer o trabalho de planejamento é seguir a seqüência: relatório de requisitos, EDT, cronograma, relatório de custos e lista de riscos. Porém os documentos se interagem fazendo com que essa seqüência linear dificilmente aconteça. Além da interação há sempre um fato novo ou uma lembrança que nos força a rever os documentos. Seja como for a EDT é o documento chave, o elo de ligação entre as evidências

Método para Gerenciamento de Pequenos Projetos de Software

Jaylson Teixeira

09/06/1999



5. Cronograma

O cronograma fica fácil de ser feito tendo em mãos a EDT(Estrutura de Divisão do Trabalho). A partir da EDT deve-se estabelecer responsáveis e prazos para as atividades. Estabelecido início e fim de cada atividade tem-se o cronograma.

Para se realizar um projeto é necessário quebrar o total do trabalho a ser realizados em tarefas menores. Sommerville recomenda que estas tarefas tenham a duração de uma ou duas semanas. Para cada tarefa é estabelecido um prazo de execução. As tarefas expostas de maneira ordenada, com relação de dependência, geram marcos no tempo, como mostra a figura 4.

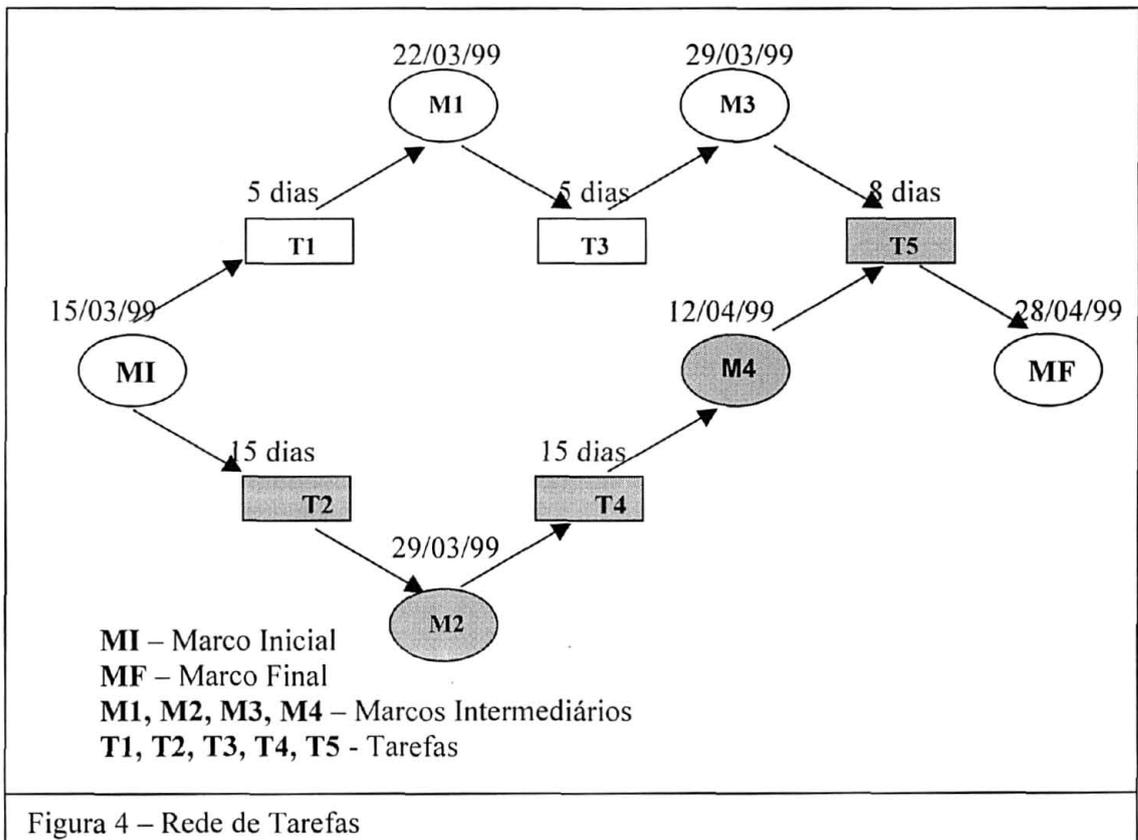


Figura 4 – Rede de Tarefas

Método para Gerenciamento de Pequenos Projetos de Software

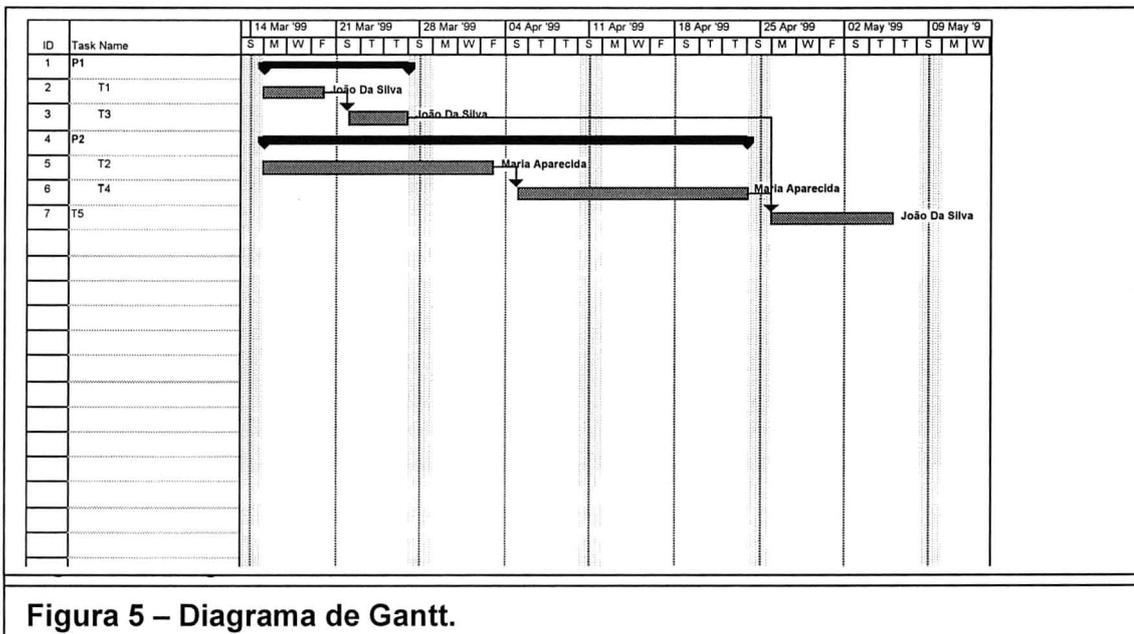
Jaylson Teixeira

09/06/1999

O caminho do marco inicial ao final que está hachurado na figura 2 é o caminho crítico. Caminho crítico é o caminho do marco inicial ao final que possui o maior prazo. Este prazo é o prazo mínimo de entrega do projeto.

No exemplo da figura 4, como as atividades T1 e T3 podem ser realizadas em paralelo com T2 e T4, duas pessoas seria o número ideal de profissionais para este projeto. Se apenas uma pessoa fosse alocada neste projeto, o prazo final não seria o prazo do caminho crítico.

Na figura 5 encontra-se as mesmas tarefas da figura 4 distribuídas entre duas pessoas na forma conhecida como diagrama de Gantt.

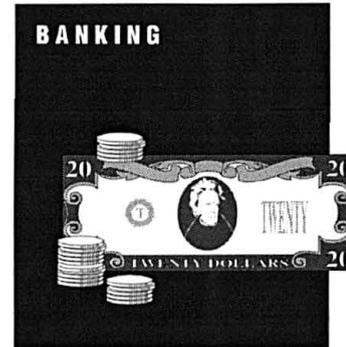


Método para Gerenciamento de Pequenos Projetos de Software

Jaylson Teixeira

09/06/1999

6. Relatório de Custos



No aspecto orçamento têm-se os componentes de custo e as técnicas de estimativa de custos e as abordagens.

6.1. Técnicas de Estimativas de Custo

Modelo Algoritmo de Custos: Usa algoritmo para fornecer custo em função de uma medida de software (geralmente tamanho). É estimada a métrica e o algoritmo prediz o esforço requerido.

Julgamento por Especialista: Um ou mais especialistas são chamados. São especialistas nas técnicas de desenvolvimento de software que serão utilizadas ou no domínio da aplicação. Chega-se a um custo por consenso.

Estimativa por Analogia: É utilizado quando um projeto semelhante em tecnologia ou no domínio da aplicação já foi desenvolvido.

Lei de Parkinson: O trabalho deve ser expandido para alcançar o tempo disponível e o custo deve ser aquele disponível para gastar.

Preço para Ganhar: O preço deve ser o que o usuário tem disponível para gastar.

6.1 Abordagem

Estimativa Top-Down: O custo é estimado por uma propriedade geral do projeto e depois repartido entre os componentes do projeto.

Estimativa Bottom-up: O custo é estimado para cada componente do projeto e a estimativa geral é a soma dessas partes.

Método para Gerenciamento de Pequenos Projetos de Software

Jaylson Teixeira

09/06/1999

6.3. Componentes do Custo

O relatório de custos deve ter como meta levantar o custo total do projeto para definição de um orçamento. A divisão dos custos proposta por Sommerville [Sommerville 1990] é uma maneira de classifica-los. Esta divisão é:

- **Hardware & Software.**
- **Viagens.**
- **Treinamento.**
- **Esforço** (medido, por exemplo, em Homens-Mês.)

Cada uma dessas divisões podem ser precificada com por um valor unitário e uma quantidade especificando a unidade adotada. É como se costuma preencher uma nota fiscal.

Os custos geralmente têm uma parte fixa formada pelos valores que se paga uma vez durante o projeto, como o custo do hardware necessário. Outra parte é variável formada pelos valores que aumentam em função do tempo de duração do projeto, como o valor da mão de obra empregada. Esses dois aspectos devem ser lembrados na composição do custo total do projeto principalmente aqueles custos fixos que podem ser utilizado em outros projetos, como a compra de um compilador.

Neste trabalho recomenda-se a divisão de Hardware e Software de Sommerville outros custos materiais como por exemplo a compra de papel especial com marca d'água para impressão de um Certidão.

Método para Gerenciamento de Pequenos Projetos de Software

Jaylson Teixeira

09/06/1999



7. Lista de Riscos

A lista de riscos como conta booch [Booch 1995] é particular de cada projeto. Dessa forma não há receita para identificação dos riscos. As melhores pessoas para fazer esta lista são as que estão diretamente envolvidas no projeto. Dê preferência a pessoas experientes.

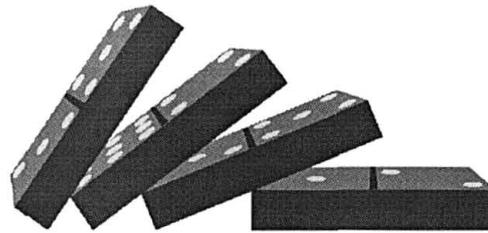
Observe o formulário utilizado no exemplo Divulgação de Notas. Nele é registrado a caracterização do problema e a solução proposta, além da data que foi caracterizado o risco e a data na qual o risco foi resolvido.

O risco é um problema em potencial. Resolução de risco pode ser a resolução do problema caso ele se manifeste. Pode ser também uma forma de evitar o problema.

Método para Gerenciamento de Pequenos Projetos de Software

Jaylson Teixeira

09/06/1999



8. Modelo de Processo em V Simplificado.

Ould, em seu livro **Strategies for Software Engineering – The Management of Risk and Quality** [Ould 1990], diz que deve-se fazer o planejamento baseado em redução de riscos. Para isso o modelo do processo de desenvolvimento de software deve ser compatível com o risco enfrentado. O modelo do processo não é um método. Para um projeto adota-se apenas um modelo do processo e vários métodos inseridos no processo. Segundo Ould, o principal motivo para adoção de um modelo de processo é dar ao projeto de desenvolvimento de software uma estrutura de redução de riscos. Um projeto sem estrutura não é gerenciável. Ele não pode ser planejado, não se pode fazer estimativas, não se pode estabelecer marcos, não é possível se monitorar e não se pode oferecer ao cliente nenhuma perspectiva de custos e prazos. Com uma estrutura fornecida pelo modelo de processo, planejada para reduzir custos, reduz-se a incerteza e torna o projeto mais gerenciável.

Ould ilustra alguns modelos utilizados e os relaciona com o nível de incerteza dos requisitos. Os modelos de processos descritos para projetos de baixa complexidade são: Modelo de processo em V e modelo de processo em V com protótipo.

Método para Gerenciamento de Pequenos Projetos de Software

Jaylson Teixeira

09/06/1999

O **modelo de processo em V** é utilizado quando se tem um baixo nível de incerteza. Os requisitos são bem definidos e inteligíveis pelos desenvolvedores no início do projeto

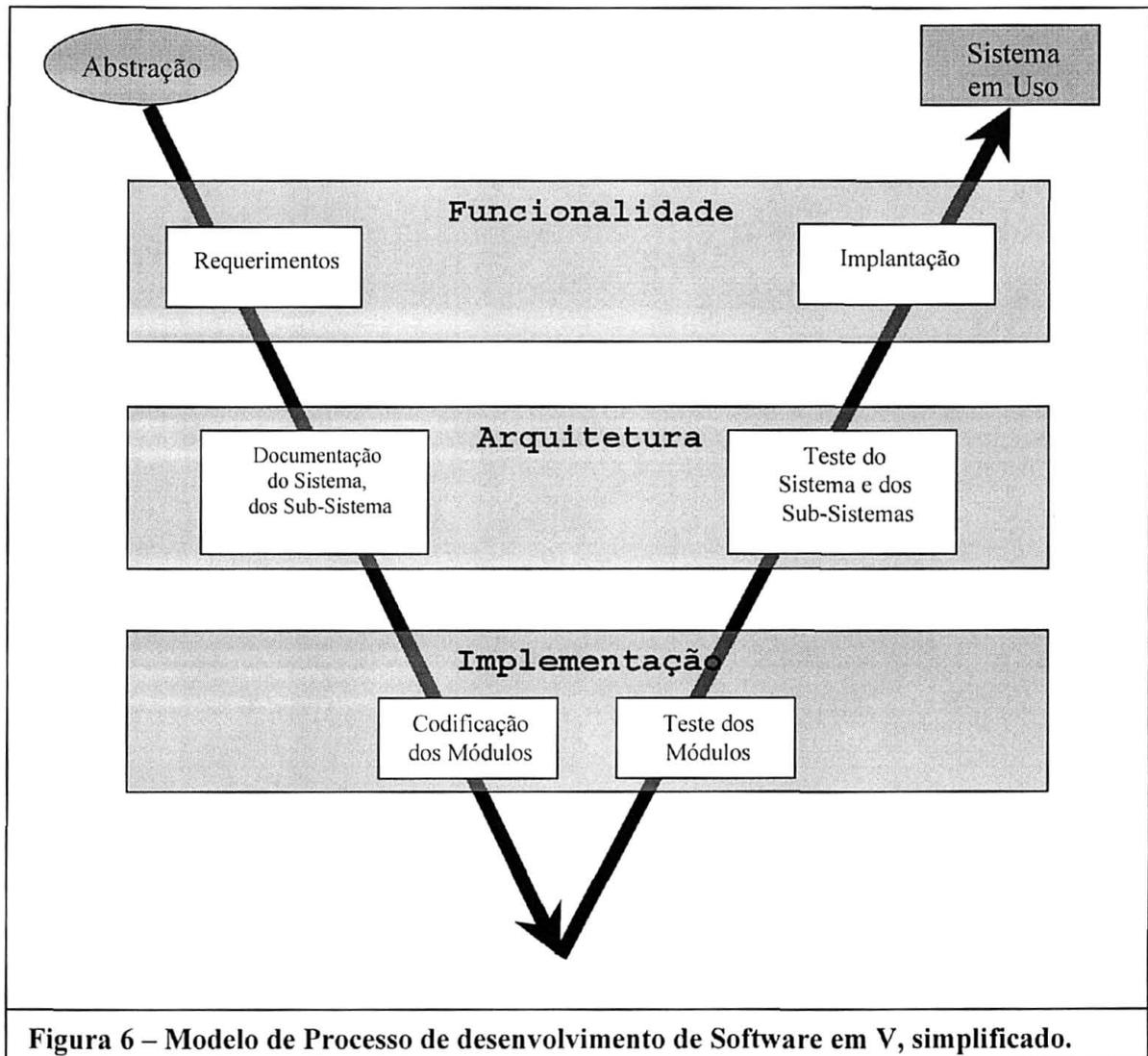
O **modelo de processo em V com protótipo** é utilizado quando existe uma pequena incerteza a respeito dos requisitos ou da sua implementação. Neste caso um protótipo é realizado. As atividades necessárias ao protótipo são adicionadas ao modelo de processo em V.

O modelo em V é semelhante ao modelo em Cascata mas com uma forte ênfase nos testes. Cada módulo produzido deve ser testado. Ao se obter os subsistemas, e depois o sistema por completo, eles devem ser verificados, isto é, deve-se verificar se a arquitetura proposta está de acordo com o que foi documentado na fase de planejamento do sistema. Uma vez que o sistema esteja pronto deve ser validado, isto é, verificar se o sistema funciona de acordo com o que foi definido nos requisitos.

Método para Gerenciamento de Pequenos Projetos de Software

Jaylson Teixeira

09/06/1999



ANEXO C

PROJETOS UTILIZADOS NO ESTUDO DE CASOS

ANEXO C.2

BOLETIM NET

PROJETO BOLETIM-NET

Fernando Ferraz
Flavio Luiz da Silva
Roberto Marini Steck

Orientador: Irapuru Florido

RELATÓRIO DE REQUISITOS

INTRODUÇÃO:

O Sistema Boletim-NET é um meio bem mais rápido e prático de pôr a disposição informações úteis à alunos, pais de alunos e a sociedade em si. Implantado em escolas, o Sistema permite ao Professor digitar as notas de alunos, mandar avisos, consultar seu horário e de outros professores, além de muitas outras funções. Para o aluno, ficará muito mais fácil saber de suas notas bimestrais, horários, calendário escolar, avisos da secretaria, editais, comunicados, além de outros. Para os funcionários da secretaria tudo ficará mais fácil também, pois não precisarão se preocuparem com divulgação das informações. Por exemplo, os boletins escolares serão disponibilizados via Internet e a secretaria não precisará imprimir e enviar boletins para os alunos, pois os mesmos que vão ter acesso as suas notas. Assim evita o desperdício, por que somente os alunos que tiverem interesse é que vão abrir seus boletins. E para os alunos que não tem Internet em casa, poderão acessar via terminais que estarão à disposição deles na Biblioteca da Escola, incentivando até mesmo o aluno a freqüentar a própria.

MODELOS DO SISTEMA:



Basicamente, terá um Sistema Client que irá alimentar os dados que vão estar armazenados no Servidor. O Servidor, por sua vez, terá um sistema que vai gerar páginas dinâmicas com as informações solicitadas pelo Browser.

OBJETIVO GERAL DO SISTEMA

Informatizar a divulgação de notas e informações gerais do colégio para os alunos e outros interessados, usando uma rede local e a Internet.

ESCOPO DO PROJETO (OBJETIVOS ESPECÍFICOS)

Permitir que o aluno consulte suas notas bimestrais (boletim), calendário, provas, trabalhos marcador, horários de turmas e horários de professores da escola bem como se informe de eventos extra classe e outras atividades programadas na escola. Isso a partir de qualquer terminal da escola ou pela Internet.

Possibilitar à secretaria: manter cadastro de alunos atualizado; manter cadastro de disciplinas; manter calendário de eventos atualizado; manter cadastro de turmas atualizado; manter cadastro de Professores atualizado.

Permitir ao professor divulgar as notas bimestrais de cada aluno por turma e disciplina, através da rede local ou Internet. O professor poderá divulgar também as faltas e observações em relação ao comportamento do aluno, bem como divulgar datas de provas e entrega de trabalhos.

DEFINIÇÃO DOS REQUERIMENTOS FUNCIONAIS

1 - Consulta do boletim -> O aluno via Browser terá acesso dinâmico às suas notas, pois assim que o professor registra as notas, automaticamente as mesmas já estão à disposição na Internet. Ele entra com sua matrícula e senha e o Sistema traz o Boletim do aluno com as notas que já foram apresentadas pelos professores, deixando em branco os espaços das notas que ainda não foram preenchidas, faltas referente ao mesmo e observações feitas pelo seu professor.

2 - Consulta do Calendário Escolar -> Via Browser, o aluno ou qualquer outro interessado terá acesso ao Calendário Escolar através de um link que terá na Página da escola, onde qualquer alteração no calendário será atualizada automaticamente.

3 - Consulta do Horário das aulas -> O aluno via Browser informará sua matrícula e sua senha, isso feito, o Sistema apresenta o horário das disciplinas que o aluno está matriculado; podendo também o aluno consultar o horário de seus professores, para saber onde se encontram em casos diversos.

4 - Consulta do Horário do Professor -> O professor, devidamente identificado no Sistema, terá seu horário e dos demais professores para consulta quando for preciso, também mostrara quais horários o professor estará no colégio para revisões, dúvidas e outros problemas.

5 - Consulta de Datas -> O aluno via Browser poderá consultar datas importantes como provas, palestras e entregas de trabalhos que serão informados ao sistema pelo seu professor. Ele entrará com sua matrícula e senha e automaticamente o Sistema busca essas informações atualizadas, isto é, se o Professor acabou de marcar tais datas, elas já estarão à disposição do aluno.

6 - Inclusão de Notas -> O professor poderá incluir as notas bimestrais de qualquer terminal nas dependências da escola. Ele informará seu login e senha e terá acesso ao Sistema com suas limitações. O professor poderá alterar as notas que incluiu até a data de fechamento do bimestre. Após isso, as notas não podem ser mais alteradas, com exceção de casos extremos, que serão acertados pela secretaria.

7 - Divulgação de datas e informações -> O Professor também terá um espaço para divulgar datas de provas, palestras, trabalhos e outros eventos em sua disciplina. Terá também, se desejar, fazer observações referentes aos seus alunos para os mesmos consultarem. Ele terá que entrar no Sistema, informar seu login e senha para ter acesso a esse serviço.

8 - Cadastro de Aluno -> Os funcionários da secretaria também serão devidamente identificados com seus respectivos login's. Eles poderão Incluir, Alterar e Consultar informações que dizem respeito aos alunos. Os dados dos alunos poderão ser acessados por outros sistemas, os de Controle Acadêmico por exemplo.

9 - Cadastro do Professor -> A secretaria terá acesso aos dados do professor através do login e senha. Poderá incluir professores, alterar dados e fazer consulta. Esses dados de professores também poderão ser acessados por outros sistemas que utilizem essas informações.

10 - Cadastro de Turmas -> A secretaria que definirá as turmas a serem formadas com suas respectivas disciplinas, número de alunos e informações adicionais. Os dados da turma são fixados no início do período e não podem ser alterados até o final do mesmo.

11 - Cadastro de Disciplinas / Aluno -> A secretaria cadastrará no sistema as disciplinas por aluno. O sistema terá um controle onde o aluno só poderá ser matriculado nas disciplinas compatíveis com o seu curso e seu período.

12 - Cadastro de Disciplinas / Professor -> A secretaria cadastrará no sistema as

13 - Cadastro do Calendário Escolar -> A secretaria cadastrara o calendário escolar que a escola ira adotar durante Ano/Período. Esses dados podem ser alterados durante o ano, mas somente as datas que ainda estão por vir.

DEFINIÇÃO DOS REQUERIMENTOS NÃO FUNCIONAIS

HARDWARE:

Servidor:	Processador AMD K6-3 400mhz Memória RAM DIMM 128MB HD Quantum 12.6 GB
Cliente:	Processador AMD K6 233mhz Memória RAM 32MB HD Quantum 3.2 GB
LAN:	Rede NT Ponto a Ponto

SOFTWARE:

Servidor:	Windows NT Server SQL 7
Cliente:	Windows 98

EVOLUÇÃO DO SISTEMA

- O Operador poderá configurar as características da página da escola pelo Sistema, indicando figuras, parâmetros, configurações, etc.
- A Secretaria poderá mandar e-mail's individuais ou grupais para os alunos, sendo mensagens padrões os especificadas pela própria secretaria.
- Cada aluno poderá Ter uma conta de e-mail no domínio da Escola e também um link na página Principal, dependendo do conteúdo de sua página.
- Poderão ser criados murais de recados, informações sobre teste de seleção, achados e perdidos, vendas de livros, etc; dependendo da necessidade da Escola.

Projeto Boletim-NET

- Lista de Tarefas (EDT)

Projeto Piloto

- 1 Definição do Projeto
- 2 Definição do Escopo
- 3 Diagramação Básica
- 4 Teste de Implementação
- 5 Adequação de Projeto

Diagramação Completa

- 6 Análise Orientada ao Objeto
 - Caso de Uso
 - Cenário
 - Seqüência / Colaboração
 - Classes
 - Estados / Atividades
- 7
- 8
- 9
- 10 Organização dos Dados
 - Diagrama de Persistência
 - Relacionamentos
- 11
- 12 Telas
 - Seqüência de Telas
- 13

Implementação

- 14 Cliente:
 - 15 Construção das Telas
 - 16 Implementação dos Módulos Individualmente
 - 17 Testes dos Módulos Separados
 - 18 Agregação dos Módulos
 - 19 Testes do Sistema como um todo
- 20 Servidor
 - 21 Construção das Telas
 - 22 Implementação dos Módulos Individualmente
 - 23 Testes dos Módulos Separados
 - 24 Agregação dos Módulos
 - 25 Testes do Sistema como um todo
- 26 Banco de Dados
 - 27 Instalação
 - 28 Testes

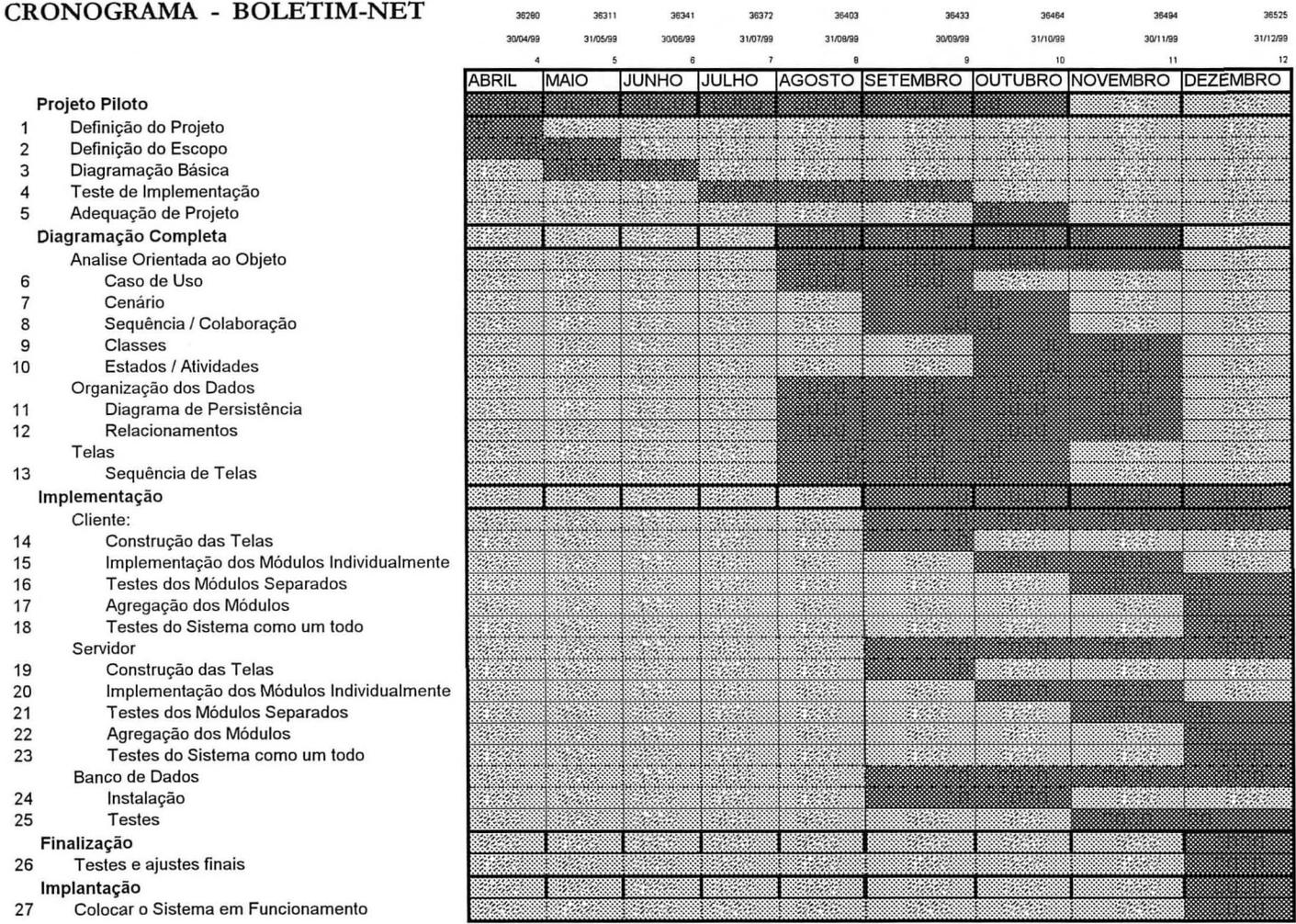
Finalização

- 26 Testes e ajustes finais

Implantação

- 27 Colocar o Sistema em Funcionamento

CRONOGRAMA - BOLETIM-NET



Legenda: =Tempo passado =Tempo restante =Primeira quinzena
 Bordas: =Segunda quinzena =Mês completo =Terceiro nível
 =Primeiro nível =Segundo nível

Relatório de Custos

Projeto Boletim-NET

Descrição	Preço Unitário	Quantidade	Total
1. Hardware, Software e Outros Materiais			
1.1. Servidor NT	R\$ 3000,00	1	R\$ 3000,00
1.2. Rede NT ponto á ponto	R\$ 1200,00	1	R\$ 1200,00
1.3. Ms SQL Server	R\$ 1250,00	1	R\$ 1250,00
1.4. Livros: "Delphi 4 Guia Completo" e "Delphi 4 A Bíblia"	R\$ 350,00	1	R\$ 350,00
1.5. Broser e Editor HTML	Free		
2. Viagens			
2.1.			
2.2.			
3. Treinamento			
3.1.			
3.2.			
4. Esforços			
4.1. Custo hora/trabalho dos desenvolvedores	R\$ 20,00	720	R\$14400,00
4.2.			

TOTAL:

R\$ 20200,00

***OBS.: Sujeito à alterações.**

Lista de Riscos

Projeto
Boletim-Net

<i>Identificação do Risco:</i> HTML dinâmicas pelo Delphi.	
<i>Data Abertura</i> 15/04/1999	<i>Descrição do Problema</i> Falta de conhecimento de uma nova tecnologia que permite gerar páginas no formato HTML dinâmicas pelo Delphi
<i>Data Fechamento</i> Em Aberto	<i>Solução Proposta</i> Os componentes da equipe estão pesquisando e procurando a solução

<i>Identificação do Risco:</i> Tempo	
<i>Data Abertura</i> 09/09/1999	<i>Descrição do Problema</i> Percebemos através da elaboração do Cronograma percebemos que o tempo que temos para concluir o projeto é curto.
<i>Data Fechamento</i> 18/09/1999	<i>Solução Proposta</i> Elaboração de um novo Cronograma onde várias tarefas serão desenvolvidas simultaneamente para concluir à tempo.

<i>Identificação do Risco</i>	
<i>Data Abertura</i>	<i>Descrição do Problema</i>
<i>Data Fechamento</i>	<i>Solução Proposta</i>

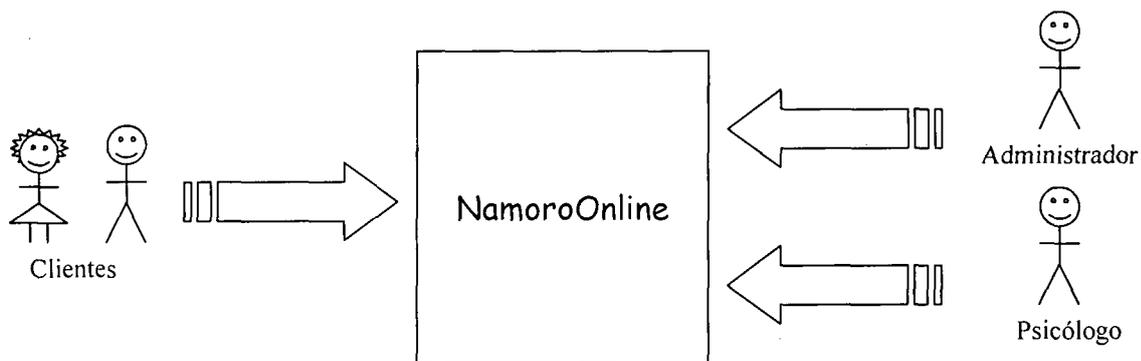
ANEXO C.1

NAMORO ONLINE

Introdução

Devido ao fato de muitas pessoas procurarem, na internet, outras pessoas para conversar, namorar, ou até mais, a agência NamoroOnline está sendo desenvolvida. O software, como uma agência de namoro *online*, poderá ser utilizado em um provedor, atraindo, assim, a visita de mais clientes e aumentando o número de patrocínios, assim como em uma agência de casamentos, aumentando o número de clientes atendidos e facilitando o atendimento ao cliente, que poderá ser feito de sua própria casa.

Modelos do Sistema



Definição dos Requerimentos Funcionais

1. O cliente poderá:
 - 1.1. Cadastrar seus dados pessoais, juntamente com suas características e preferências;
 - 1.2. Solicitar uma pesquisa de compatibilidade;
 - 1.3. Utilizar o mural de recados e o sistema de mensagens;
 - 1.4. Entrar no *chat* para conversas com os demais clientes que o estão utilizando;
 - 1.5. Mandar página e/ou foto;
2. Quanto ao administrador, ele poderá:
 - 2.1. Realizar pesquisas de dados estatísticos gerais e por pessoa, entre os clientes do *site*;
 - 2.2. Solicitar relatórios, tanto das pessoas mais assíduas quanto das menos assíduas, com a data do último acesso ao *site*;
 - 2.3. Aprovar a página e/ou foto do cliente, ativando-a ou não;
 - 2.4. Gerenciar o cadastro dos clientes;

- 2.5. Gerenciar o tempo limite de cadastro ativo, sem acesso ao site;
 - 2.6. Configurar o intervalo de geração da compatibilidade entre os clientes;
 - 2.7. Cadastrar outros administradores e psicólogos.
3. O psicólogo poderá:
- 3.1. Cadastrar as preferências, que serão selecionadas pelos clientes ao preencherem o questionário de cadastro;
 - 3.2. Cadastrar as questões, também respondidas pelos clientes, no questionário de cadastro;
 - 3.3. Cadastrar as combinações, tanto entre preferências e questões, quanto entre questões somente.

Definição dos Requerimentos Não Funcionais

Para o cadastro do cliente, será utilizado CGI e as informações serão armazenadas em uma base de dados SQL.

O chat será desenvolvido em Perl.

Servidor

Hardware

Pentium III 400
64 MB RAM
HD SCSI 6 GB

Software

SQL Server
Windows NT 4

Cliente

Hardware

Pentium 100
16 MB RAM

Software

Windows 95
IEExplorer 3 ou Netscape Navigator 2

Evolução do Sistema

Futuramente o sistema poderá ser acrescido de:

- Um sistema de *web mail*;
- Montagem da página através do *browser*;
- Montagem das características retirada da foto enviada pelo cliente.

Especificação de Requisitos

1. O cliente poderá:

- 1.1. **Cadastrar seus dados pessoais, juntamente com suas características e preferências:** para se beneficiar dos serviços prestados pela agência, o cliente terá de cadastrar-se, respondendo a um questionário, através do qual será definido o seu perfil e pesquisada as suas compatibilidades. Já estando cadastrado no sistema, poderá alterar seus dados no *site*, assim como excluí-los (desativar);
- 1.2. **Solicitar uma pesquisa de compatibilidade:** após o processo de geração da compatibilidade, executado automaticamente, em intervalo pré-definido pelo administrador, o cliente poderá solicitar uma pesquisa das suas compatibilidades, onde o resultado será uma lista dos usuários compatíveis à ele, com o respectivo percentual de compatibilidade entre eles;
- 1.3. **Utilizar o mural de recados e o sistema de mensagens:** o cliente poderá mandar mensagens a um cliente específico através do sistema de mensagens (recados) ou para todos os clientes, através do mural de recados;
- 1.4. **Entrar no chat para conversas com os demais clientes que o estão utilizando:** o *chat* é formado por três diferentes tipos de mensagens: a geral, que será vista por todos os usuários do *chat*; a reservada, vista somente pelo destinatário da mensagem e a compatível, vista, somente, pelos usuários compatíveis ao remetente da mesma, que estão utilizando o *chat*;
- 1.5. **Enviar página e/ou foto:** a página do cliente é enviada em substituição da página padrão mostrada pelo sistema nas consultas de compatibilidade e a foto é incluída na página padrão.

2. Quanto ao administrador, ele poderá:
 - 2.1. **Realizar pesquisas de dados estatísticos gerais e por pessoa, entre os clientes do site:** são dados estatísticos gerais o número de pessoas que acessam o site diariamente, o número de pessoas cadastradas, o número de pessoas que usaram o chat ou realizaram a pesquisa de perfil. Dados estatísticos por pessoa são: há quanto tempo está cadastrado, há quanto tempo não acessa o sistema, o número de clientes indicados por ele;
 - 2.2. **Solicitar relatórios, tanto das pessoas mais assíduas quanto das menos assíduas, com a data do último acesso ao site:** o administrador fornece o intervalo para pesquisa e o número de clientes referentes ao relatório. Após, faz uma lista dos clientes ordenada pelo número de acessos ao sistema.
 - 2.3. **Aprovar a página do cliente, ativando-a ou não:** dependendo de critérios estabelecidos pelo administrador e não controlados pelo sistema, a página será aprovada ou não, sendo, na primeira hipótese, ativada;
 - 2.4. **Gerenciar o cadastro dos clientes:** o sistema lhe fornecerá a lista dos clientes que têm o cadastro desativado, que então poderá excluídos ou reativá-los;
 - 2.5. **Gerenciar o tempo limite de cadastro ativo, sem acesso ao site:** o administrador poderá determinar o número de dias em que o cliente poderá permanecer sem acessar o *site*, após o que, terão seu cadastro desativado.
 - 3.4. **Configurar o intervalo de geração da compatibilidade entre os clientes:** O administrador pode configurar a geração da compatibilidade para ocorrer em determinado intervalo de horas, já que não é gerada automaticamente a cada inclusão ou alteração de clientes, devido ao grande fluxo cadastral previsto, não sobrecarregando, assim, o servidor.
 - 3.5. **Cadastrar outros administradores e psicólogos:** um administrador com permissão de acesso ao cadastro, poderá cadastrar outros administradores e/ou psicólogos, sendo que um administrador pode ter acesso de psicólogo, e vice-versa.
4. O psicólogo poderá:
 - 4.1. **Cadastrar as preferências, que serão selecionadas pelos clientes ao preencherem o questionário de cadastro:** o questionário é composto de dois grandes grupos, sendo um deles as preferências do cliente. Aqui, então, o psicólogo poderá cadastrá-las, para que sejam selecionadas pelos clientes ao cadastrarem-se no *site*;

- 4.2. **Cadastrar as questões, também respondidas pelos clientes, no questionário de cadastro:** o outro grande grupo do questionário são as questões, com respostas pré-definidas, que devem ser selecionadas pelos clientes, também no cadastro.
- 4.3. **Cadastrar as combinações, tanto entre preferências e questões, quanto entre questões somente:** esta parte da atividade do psicólogo compreende definir quais são os usuários combinam com quais, através de compatibilização das preferências pré-definidas com as respostas das questões e, se for o caso, de uma resposta com outra. É a parte mais importante do sistema, pois é aqui que se define como será o processo de geração de compatibilidade entre os clientes.

Atividades do Sistema Namoro Online (EDT)

<i>Nº</i>	<i>Descrição</i>	<i>Responsável</i>	<i>Tempo Estimado (semanas)</i>	<i>Semana</i>
1.	Definições do Sistema	Todos	7	3-9
2.	Diagrama de Casos de Uso	Todos	4	9-12
3.	Estudo de Java/CGI/Perl	Todos	8	17-24
4.	Elaboração do Questionário de Cadastro	Todos		15-19
5.	Cenários	-	5	11-15
5.1.	Cadastrar Usuário	Cristiane		
5.2.	Cadastrar Psicologo	Mauro		
5.3.	Cadastrar Administrador	Mauro		
5.4.	Cadastrar Questionário	Aldo		
5.5.	Gerar Compatibilidade	Cristiane		
5.6.	Emitir Relatório	Aldo		
5.7.	Participar do Chat	Cristiane		
5.8.	Mensagens	Cristiane		
5.9.	Escrever Recado no Mural	Cristiane		
6.	Diagrama de Sequência	-	5	14-18
6.1.	Cadastrar Usuário	Cristiane		
6.2.	Cadastrar Psicologo	Aldo		
6.3.	Cadastrar Administrador	Aldo		
6.4.	Gerenciar Cadastro de Usuários	Cristiane		
6.5.	Cadastrar Questões	Mauro		
6.6.	Cadastrar Preferências	Mauro		
6.7.	Registrar Combinações	Mauro		
6.8.	Desativar Usuários	Cristiane		
6.9.	Gerar Compatibilidade	Cristiane		
6.10.	Avisar Tempo Limite Usuário Ativo	Cristiane		
6.11.	Pesquisar Compatibilidade	Cristiane		
6.12.	Enviar Página	Mauro		
6.13.	Enviar Foto	Mauro		
6.14.	Aprovar Página	Mauro		
6.15.	Aprovar Foto	Mauro		
6.16.	Emitir Relatório	Aldo		
6.17.	Participar do Chat	Cristiane		
6.18.	Enviar Mensagem	Aldo		
6.19.	Ler Mensagem	Aldo		
6.20.	Escrever Recado no Mural	Aldo		
6.21.	Verificar Acesso Cliente	Cristiane		
6.22.	Verificar Acesso Psicologo	Cristiane		
6.23.	Verificar Acesso Administrador	Cristiane		
7.	Telas do Sistema	-	5	17-21
7.1.	Cadastrar Usuário	Cristiane		
7.2.	Cadastrar Psicologo	Aldo		
7.3.	Cadastrar Administrador	Aldo		
7.4.	Gerenciar Cadastro de Usuários	Cristiane		
7.5.	Cadastrar Questões	Mauro		
7.6.	Cadastrar Preferências	Mauro		
7.7.	Registrar Combinações	Mauro		

7.8.	Desativar Usuários	Cristiane		
7.9.	Gerar Compatibilidade	Cristiane		
7.10	Avisar Tempo Limite Usuário Ativo	Cristiane		
7.11	Pesquisar Compatibilidade	Cristiane		
7.12	Enviar Página	Mauro		
7.13	Enviar Foto	Mauro		
7.14	Aprovar Página	Mauro		
7.15	Aprovar Foto	Mauro		
7.16	Emitir Relatório	Aldo		
7.17	Participar do Chat	Cristiane		
7.18	Enviar Mensagem	Aldo		
7.19	Ler Mensagem	Aldo		
7.20	Escrever Recado no Mural	Aldo		
7.21	Verificar Acesso Cliente	Cristiane		
7.22	Verificar Acesso Psicologo	Cristiane		
7.23	Verificar Acesso Administrador	Cristiane		
8.	Diagrama de Estados	Todos	5	19-23
8.1.	TConfiguração	Aldo		
8.2.	TPsicologo	Aldo		
8.3.	TAdministrador	Aldo		
8.4.	TEstatisticas	Aldo		
8.5.	TMsgChat	Mauro		
8.6.	TChat	Mauro		
8.7.	TGeral	Mauro		
8.8.	TCompativel	Mauro		
8.9.	TReservado	Mauro		
8.10	TColecaoCliente	Cristiane		
8.11	TCliente	Cristiane		
8.12	TColecaoCompatibilidade	Cristiane		
8.13	TCompatibilidade	Cristiane		
8.14	TQuestionário	Cristiane		
8.15	TQuestao	Cristiane		
8.16	TPreferencia	Cristiane		
8.17	TOpcaoPreferencia	Cristiane		
8.18	TColecaoMural	Mauro		
8.19	TMural	Mauro		
8.20	TColecaoMensagem	Mauro		
8.21	TMensagem	Mauro		
8.22	TColecaoPagina	Aldo		
8.23	TPagina	Aldo		
8.24	TColecaoFoto	Aldo		
8.25	TFoto	Aldo		
9.	Diagrama de Classes	Cristiane	4	16-19
10.	Tabelas do Sistema	Aldo	4	20-23
11.	Diagrama de Entidade e Relacionamento	Mauro	2	22-23
12.	Aprovação Geral dos Diagramas	Todos	3	23-25
13.	Implementação	Todos	6	25-32

Relatório de Custos

<i>Projeto</i> NamoroOnline

Descrição	Preço Unitário	Quantidade	Total
1. Hardware, Software e Outros Materiais			
1.1. Borland Delphi 4.0	R\$ 6000,00	1	R\$ 6000,00
1.2. Crystal	R\$ 1500,00	1	R\$ 1500,00
1.3. Microsoft SQL Server	R\$ 2355,00	1	R\$ 2355,00
1.4. Broser e Editor HTML	Free		
2. Viagens			
2.1.			
3. Treinamento			
3.1. Livros: "Delphi 4 Curso Completo"	R\$ 149,00	1	R\$ 149,00
4. Esforços			
4.1. Hora/Análise	R\$ 50,00	360	R\$ 18000,00
4.2. Hora/Programação	R\$ 30,00	240	R\$ 7200,00

TOTAL:

R\$ 35.204,00

***OBS.: Sujeito à alterações.**

Relatório de Custos

Namoro Online

- Implantação -

Descrição	Preço Unitário	Quantidade	Total
1. Hardware, Software e Outros Materiais			
1.1. Servidor		1	
1.2. Windows NT Server		1	
1.3. Microcomputador Pentium III		1	
1.4. Browser		2	
1.5. Microsoft SQL Server 7.0		1	
2. Viagens			
2.1.			
2.2.			
3. Treinamento			
3.1.			
3.2.			
4. Esforços			
4.1. Hora/Técnico			
4.2.			

Lista de Riscos

Projeto
NamoroOnline

<i>Identificação do Risco</i>	
Questionário	
<i>Data Abertura</i> 20/05	<i>Descrição do Problema</i> O projeto é dependente de um questionário para a análise das combinações.
<i>Data Fechamento</i> 27/08	<i>Solução Proposta</i> Deixamos de lado a formação de um questionário, deixando o sistema mais flexível, e o questionário à cargo de um psicólogo, que o cadastrará

<i>Identificação do Risco</i>	
Ferramentas de Programação	
<i>Data Abertura</i> 01/07	<i>Descrição do Problema</i> Nenhum dos integrantes da equipe tem conhecimento nas linguagens propostas para o desenvolvimento do sistema, sendo elas: <ul style="list-style-type: none">▪ Delphi (Interface com a internet);▪ Java, Java/CGI;▪ Perl, Perl/CGI.
<i>Data Fechamento</i>	<i>Solução Proposta</i> Estudo das ferramentas

ANEXO C.3

PROTOCOLO ELETRÔNICO

Gabriel Kalil
Leoncio José de Almeida Reis
Renato Afonso Zatta

GEM - Sistema de Compatibilidade Genética

Curitiba, 1999

Índice

1. INTRODUÇÃO	1
2. GLOSSÁRIO	1
3. MÓDULOS DO SISTEMA.....	3
4. DEFINIÇÃO DE REQUISITOS FUNCIONAIS.....	3
5. REQUISITOS NÃO FUNCIONAIS.	4
6. EVOLUÇÃO DO SISTEMA	4
7. ESPECIFICAÇÃO DE REQUISITOS.....	5
7.1. MAPA DO MENU DO SISTEMA.	5
7.2. RELATÓRIOS DE EXAMES	5
8. DIAGRAMA DE RISCOS.....	6
9. ESTRUTURA DE DECOMPOSIÇÃO DO TRABALHO (EDT)	7
10. RELATÓRIO DE CUSTOS	8
11. CRONOGRAMA.....	10

1. Introdução

Atualmente, é realizado no Hospital de Clínicas da UFPR, Seção de Histocompatibilidade do Setor de Imunogenética do Laboratório Central de Análises Clínicas, um exame para identificar a compatibilidade genética de um paciente, e através disto, definir se é possível ou não um transplante de órgão.

O exame é composto de vários processos, sendo o último deles, a comparação entre o alelo do paciente e os padrões genéticos pré-existentes. Esta comparação é realizada manualmente e, devido a grande complexidade, está sujeito a erros de imprecisão.

Este projeto objetiva automatizar este processo do exame, acelerando a obtenção dos resultados e extinguindo as chances de erro. Esta automatização tornará o exame mais rápido, seguro, e com os custos operacionais reduzidos.

Antes do processo de comparação dos resultados, existe um outro processo que consta na leitura e interpretação de uma folha de raio-x. O resultado dessa interpretação é o alelo do paciente que será usado no processo posterior. Existe então a possibilidade de automatizar esta parte do exame, que é a leitura e interpretação da folha de raio-x através da utilização de um scanner, agilizando ainda mais o exame.

2. Glossário

Alelo: uma das formas alternativas (variantes) de um gene, que ocupa um determinado lugar (loco) num cromossomo.

Genômico: referente ao genoma. Sendo que genoma é o conjunto de genes de um ser vivo.

HLA-DRB: conjunto de genes do complexo HLA (gene DRB1, DRB3, DRB4 e DRB5), ditos de classe II, que participam da resposta imunológica contra agentes patogênicos (bactérias, vírus, etc.) através da apresentação de fragmentos de

proteínas destes agentes infecciosos aos linfócitos T auxiliares (um tipo de glóbulo branco).

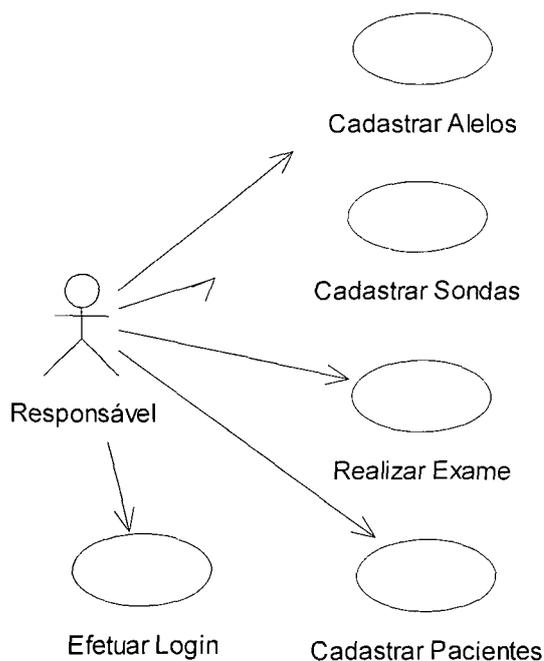
Primer: um curto segmento de DNA ou RNA que hibridiza-se (alinha-se) com a seqüência complementar de uma fita de DNA, e desta forma permite que a enzima DNA Polimerase inicie a síntese de uma nova fita de DNA. Os primers que existem nos organismos vivos são moléculas de RNA e permitem a duplicação (replicação) das moléculas de DNA de todos os seres vivos. Os primers utilizados “in vitro” (em laboratório), são seqüências de DNA sintético, permitem a amplificação de genes através da técnica de reação em cadeia da polimerase (PCR). Após 30 ciclos de PCR obtem-se aproximadamente 1 milhão de segmentos de DNA a partir de 1 segmento de DNA alvo (do gene escolhido para análise).

Sonda: um curto segmento de DNA (oligonucleotídeo) sintético utilizado para detectar (investigar ou sondar) seqüências variáveis (polimórficas) numa amostra de DNA que está sendo testada.

PCR-SSO (Polymerase Chain Reaction – Sequence Specific Oligonucleotide): este é um método de laboratório que permite, dentre outras coisas, a identificação (tipificação) dos alelos HLA existentes num determinado indivíduo. O método emprega a técnica da reação em cadeia da polimerase (PCR) para a amplificação de um gene (ou segmento do gene) seguida da hibridização com sondas de oligonucleotídeos seqüência específicas (SSO).

Tipificação HLA: identificação (designação) dos alelos HLA de um indivíduo através do emprego de um método molecular de análise como, por exemplo, a PCR-SSO.

3. Módulos do Sistema



4. Definição de Requisitos Funcionais

Sempre a interpretação e análise dos resultados é feita por duas pessoas para que haja segurança na emissão dos laudos. Existem 6 pessoas no laboratório capacitadas para fazer a análise e interpretação dos resultados.

A administração do sistema poderá ser realizada por dois funcionários responsáveis da Imunofenotipagem. A realização do exame poderá ser feito pelos bioquímicos e biólogos do Setor.

O tempo decorrido entre o recebimento da amostra de sangue e a entrega do resultado é de 4 a 5 dias. O laboratório trabalha com uma bateria de aproximadamente 48 amostras (pacientes) por vez.

A análise dos resultados, que inclui a leitura das reações nos filmes de raio-X e a posterior comparação do padrão de reatividade de cada amostra com o conjunto de sondas utilizados, leva de 4 a 6 horas, e permite a análise dos 48 indivíduos incluídos numa bateria.

As funções disponíveis no sistemas são:

Efetuar login: O biólogo digita seu login e senha para poder efetuar o eame ou cadastrar alelos, sondas ou pacientes.

Cadastrar Alelos: Os biólogos cadastram os alelos para a execução da pesquisa.

Cadastrar Sondas: É cadastrado também para a pesquisa de compatibilidade genética.

Realizar Exame: É feito a pesquisa com base no Banco de dados onde estão cadastrados os alelos e sondas e é dado a compatibilidade genética do paciente como resultado.

Cadastrar Pacientes: Cadastro dos pacientes e seus rtesultados para ser feito uma estatística posteriormente.

5. Requisitos não Funcionais.

Descrever a funcionalidade do sistema.

Como será feito o cadastro.

Como será feito a busca

O sistema será sendo desenvolvido em JAVA.

O banco de dados será em SQL.

6. Evolução do sistema

Existe a possibilidade de automatizar o exame, através da leitura e interpretação da folha de raio-x com a utilização de um scanner.

O usuário colocaria o raio-X no scanner, esperaria alguns segundos a leitura do raio-X pelo scanner, o Sistema de Compatibilidade Genética interpretaria o mesmo raio-X digitalizado, compararia com os dados armazenados previamente(Anexo 2) e logo após daria o resultado do exame do paciente .

7. Especificação de Requisitos.

7.1. Mapa do menu do sistema.

- Biólogo/Bioquímico (Pede Senha de Acesso)
 - Cadastra Sondas
 - Cadastra Alelos
 - Cadastrar paciente
 - Realizar pesquisa no BD

7.2. Relatórios de exames

No relatório deve conter as informações:

- Nome do paciente
- Resultado do exame
- Bioquímico responsável pelo exame

8. Diagrama de Riscos

Identificação do Risco	
Linguagem JAVA	
<i>Data Abertura</i> 03/05/1999	<i>Descrição do Problema</i> Ninguém da equipe tinha utilizado o JAVA antes deste projeto.
<i>Data Fechamento</i> 12/05/1999	<i>Solução Proposta</i> Fizemos um treinamento para começarmos a programar nesta linguagem.
Identificação do Risco	
Metodologia Orientada a Objetos	
<i>Data Abertura</i> 05/04/1999	<i>Descrição do Problema</i> Nós nunca tínhamos utilizado esta metodologia.
<i>Data Fechamento</i> 10/08/1999	<i>Solução Proposta</i> Foi ensinado para a nossa turma, os principais componentes desta metodologia e ainda está sendo dado um suporte.
Identificação do Risco	
Exame HLA-DRB 1*	
<i>Data Abertura</i> 27/04/1999	<i>Descrição do Problema</i> A equipe não tinha nenhum conhecimento de como era realizado este exame.
<i>Data Fechamento</i> 28/10/1999	<i>Solução Proposta</i> O Hospital das Clínicas da UFPR ensinou para nós todo o processo de execução do exame.

9. Estrutura de Decomposição do Trabalho (EDT)

1	Projeto
2	Protótipo
3	Automatizar parte do exame HLA-DRB1*
4	Definição da Base de Dados
5	Definir tabela Paciente
6	Definir tabela Exame
7	Definir tabela Sondas
8	Definir tabela Alelos
9	Definir tabela Usuários
10	Documentação
11	Módulo Menu
12	Módulo Cadastro de Pacientes
13	Módulo Cadastro de Sondas
14	Módulo Cadastro de Alelos
15	Módulo Pesquisa
16	Implementação
17	Módulo Menu
18	Módulo de Cadastro de Pacientes
19	Módulo de Cadastro de Sondas
20	Módulo de Cadastro de Alelos
21	Módulo de Execução do Exame

10. Relatório de Custos

Custos Pessoais

Tarefa	Horas	Custos/Hora	Custo Total
Análise/Projeto	640	R\$ 60,00	R\$ 38.400,00
Programa	320	R\$ 15,00	R\$ 4.800,00
TOTAL	960	R\$ 45,50	R\$ 43.200,00

Custo dos Programas e Equipamentos utilizados

Programas	Custos
Microsoft Windows NT Server 4.0	R\$ 1200,00
Erwin/ERX 2.5	Gratuito
Microsoft Office 97	R\$ 800,00
Visio Professional 5.0 for Microsoft Windows	R\$ 560,00
Together J 3	Gratuito
Rational Rose 98 Enterprise Edition	R\$ 8.100,00
SQL Server 7.0	R\$ 4.500,00
Jbuilder 3 Enterprise	R\$ 4.900,00
Subtotal programas	R\$ 19.630,00
Equipamentos	Custos
Microcomputador com processador Pentium II 333 MHz e 128 megabytes de memória RAM com kit Multimídia	R\$ 2.600,00
Impressora HP Laserjet 5M	R\$ 2.500,00
Subtotal equipamentos	R\$ 5.100,00
TOTAL	R\$ 25.430,00

Custos de Implantação do Sistema

Programas

Especificação	Preço Total
Microsoft Windows NT Server 4.0	R\$ 870,00
SQL Server	R\$ 4.500,00
Microsoft Windows 95	R\$ 300,00
Microsoft Internet Explorer 5	Gratuito
Virtual Machine Java	Gratuito
TOTAL	R\$ 5.670,00

Configuração Mínima

Especificação	Preço Unitário	Preço Total
Microcomputador com processador Pentium 166 MHz e 32 megabytes de memória RAM	R\$ 800,00	R\$ 800,00
TOTAL		R\$ 800,00
TOTAL (Programas+Config. Mínima)		R\$ 6.470,00

ANEXO D

QUESTIONÁRIOS

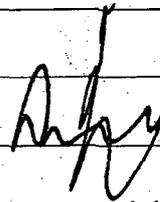
Questionário de Avaliação do Método de Planejamento e Gerenciamento de Pequenos Projetos de Software

Entrevistado: ALDO MONTEIRO DO NASCIMENTO

Cargo: ALUNO

Instituição: ET-UFPR

Projeto: Numero On line

Data: 29 / 09 / 99 Assinatura: 

Resolução de Riscos: Significa resolver um problema em potencial ou contorna-lo.

1) Tem-se por hipótese que a resolução dos riscos e o controle dos custos, prazos e funcionalidades aumentam a probabilidade de sucesso do projeto de software.

Você concorda com esta afirmação ?

Sim () Não

2) O Relatório de Requisitos serve para definir e controlar o conjunto das funcionalidades do projeto ?

Sim () Não

3) O Cronograma é um instrumento efetivo para controle de prazos ?

Sim () Não

4) O Relatório de Custos serve para definir e controlar o orçamento de projeto ?

Sim () Não

5) A Lista de Riscos e os 3 princípios (Prototipação, Adiantamento das atividades de risco e o processo em V) servem para focar os esforços de resolução dos riscos que ameaçam o sucesso do projeto ?

() Sim Não

6) Você usou mais algum documento, além dos sugeridos pelo método, para o planejamento e gerenciamento dos projeto quanto a custos, prazos e funcionalidade ?

() Sim Não

Se sim, qual : _____

7) Tomando como base a sua experiência, como você classifica o método proposto quanto a Complexidade ?

() Muito Simples Simples () Regular () Complexo () Muito Complexo

8) Tomando como base a sua experiência, como você classifica o método proposto quanto a Eficiência ?

() Ótimo Bom () Regular () Ruim () Péssimo

9) Você recomendaria o uso do método aos seus colegas?

Sim () Não

Questionário de Avaliação do Método de Planejamento e Gerenciamento de Pequenos Projetos de Software

Entrevistado: Cristiane Pereira de Andrade

Cargo: Aluna

Instituição: Bocda Técnica da UFPR

Projeto: Names Online

Data: 29/09/99 Assinatura: J. Andrade

Resolução de Riscos: Significa resolver um problema em potencial ou contorna-lo.

1) Tem-se por hipótese que a resolução dos riscos e o controle dos custos, prazos e funcionalidades aumentam a probabilidade de sucesso do projeto de software.

Você concorda com esta afirmação ?

Sim Não

2) O Relatório de Requisitos serve para definir e controlar o conjunto das funcionalidades do projeto ?

Sim Não

3) O Cronograma é um instrumento efetivo para controle de prazos ?

Sim Não

4) O Relatório de Custos serve para definir e controlar o orçamento de projeto ?

Sim Não

5) A Lista de Riscos e os 3 princípios (Prototipação, Adiantamento das atividades de risco e o processo em V) servem para focar os esforços de resolução dos riscos que ameaçam o sucesso do projeto ?

Sim Não

6) Você usou mais algum documento, além dos sugeridos pelo método, para o planejamento e gerenciamento dos projeto quanto a custos, prazos e funcionalidade ?

Sim Não

Se sim, qual : _____

7) Tomando como base a sua experiência, como você classifica o método proposto quanto a Complexidade ?

Muito Simples Simples Regular Complexo Muito Complexo

8) Tomando como base a sua experiência, como você classifica o método proposto quanto a Eficiência ?

Ótimo Bom Regular Ruim Péssimo

9) Você recomendaria o uso do método aos seus colegas?

Sim Não

Questionário de Avaliação do Método de Planejamento e Gerenciamento de Pequenos Projetos de Software

Entrevistado: Fernando Ferraz

Cargo: Aluno

Instituição: Escola Técnica da UFPR

Projeto: Boletim-NET

Data: 23 / 09 / 99

Assinatura: _____



Resolução de Riscos: Significa resolver um problema em potencial ou contorná-lo.

1) Tem-se por hipótese que a resolução dos riscos e o controle dos custos, prazos e funcionalidades aumentam a probabilidade de sucesso do projeto de software.

Você concorda com esta afirmação ?

Sim Não

2) O Relatório de Requisitos serve para definir e controlar o conjunto das funcionalidades do projeto ?

Sim Não

3) O Cronograma é um instrumento efetivo para controle de prazos ?

Sim Não

4) O Relatório de Custos serve para definir e controlar o orçamento de projeto ?

Sim Não

5) A Lista de Riscos e os 3 princípios (Prototipação, Adiantamento das atividades de risco e o processo em V) servem para focar os esforços de resolução dos riscos que ameaçam o sucesso do projeto ?

Sim Não

6) Você usou mais algum documento, além dos sugeridos pelo método, para o planejamento e gerenciamento dos projeto quanto a custos, prazos e funcionalidade ?

Sim Não

Se sim, qual : _____

7) Tomando como base a sua experiência, como você classifica o método proposto quanto a Complexidade ?

Muito Simples Simples Regular Complexo Muito Complexo

8) Tomando como base a sua experiência, como você classifica o método proposto quanto a Eficiência ?

Ótimo Bom Regular Ruim Péssimo

9) Você recomendaria o uso do método aos seus colegas?

Sim Não

Questionário de Avaliação do Método de Planejamento e Gerenciamento de Pequenos Projetos de Software

Entrevistado: FLAVIO LUIZ DA SILVA.

Cargo: ALUNO

Instituição: ESCOZA TÉCNICA DA UFPR.

Projeto: BOLETIM-NET

Data: 29/09/1999 Assinatura: Flávio L. Silva

Resolução de Riscos: Significa resolver um problema em potencial ou contorna-lo.

1) Tem-se por hipótese que a resolução dos riscos e o controle dos custos, prazos e funcionalidades aumentam a probabilidade de sucesso do projeto de software.

Você concorda com esta afirmação ?

Sim Não

2) O Relatório de Requisitos serve para definir e controlar o conjunto das funcionalidades do projeto ?

Sim Não

3) O Cronograma é um instrumento efetivo para controle de prazos ?

Sim Não

4) O Relatório de Custos serve para definir e controlar o orçamento de projeto ?

Sim Não

5) A Lista de Riscos e os 3 princípios (Prototipação, Adiantamento das atividades de risco e o processo em V) servem para focar os esforços de resolução dos riscos que ameaçam o sucesso do projeto ?

Sim Não

6) Você usou mais algum documento, além dos sugeridos pelo método, para o planejamento e gerenciamento dos projeto quanto a custos, prazos e funcionalidade ?

Sim Não

Se sim, qual : _____

7) Tomando como base a sua experiência, como você classifica o método proposto quanto a Complexidade ?

Muito Simples Simples Regular Complexo Muito Complexo

8) Tomando como base a sua experiência, como você classifica o método proposto quanto a Eficiência ?

Ótimo Bom Regular Ruim Péssimo

9) Você recomendaria o uso do método aos seus colegas?

Sim Não

Questionário de Avaliação do Método de Planejamento e Gerenciamento de Pequenos Projetos de Software

Entrevistado: IRAPURU HASKVO FLOREDO

Cargo: PROFESSOR

Instituição: Escola Técnica da UFPR

Projeto: Nome On-line / Boletim Net

Data: 29 / 9 / 99

Assinatura: _____

Resolução de Riscos: Significa resolver um problema em potencial ou contorná-lo.

1) Tem-se por hipótese que a resolução dos riscos e o controle dos custos, prazos e funcionalidades aumentam a probabilidade de sucesso do projeto de software.

Você concorda com esta afirmação ?

Sim Não

2) O Relatório de Requisitos serve para definir e controlar o conjunto das funcionalidades do projeto ?

Sim Não

3) O Cronograma é um instrumento efetivo para controle de prazos ?

Sim Não

4) O Relatório de Custos serve para definir e controlar o orçamento de projeto ?

Sim Não

5) A Lista de Riscos e os 3 princípios (Prototipação, Adiantamento das atividades de risco e o processo em V) servem para focar os esforços de resolução dos riscos que ameaçam o sucesso do projeto ?

Sim Não

6) Você usou mais algum documento, além dos sugeridos pelo método, para o planejamento e gerenciamento dos projeto quanto a custos, prazos e funcionalidade ?

Sim Não

Se sim, qual : _____

7) Tomando como base a sua experiência, como você classifica o método proposto quanto a Complexidade ?

Muito Simples Simples Regular Complexo Muito Complexo

8) Tomando como base a sua experiência, como você classifica o método proposto quanto a Eficiência ?

Ótimo Bom Regular Ruim Péssimo

9) Você recomendaria o uso do método aos seus colegas?

Sim Não

Questionário de Avaliação do Método de Planejamento e Gerenciamento de Pequenos Projetos de Software

Entrevistado: JOSE SIMÃO DE PAULA PINTO

Cargo: ANALISTA DE SUPORTE

Instituição: HOSPITAL DE CLÍNICAS DA UNIV. FEDERAL DO PR

Projeto: PROTÓTIPOS ELETRÔNICOS DE COLETA DE DADOS

Data: 26/11/1988

Assinatura: SIMÃO

Resolução de Riscos: Significa resolver um problema em potencial ou contorná-lo.

1) Tem-se por hipótese que a resolução dos riscos e o controle dos custos, prazos e funcionalidades aumentam a probabilidade de sucesso do projeto de software.

Você concorda com esta afirmação ?

Sim Não

2) O Relatório de Requisitos serve para definir e controlar o conjunto das funcionalidades do projeto ?

Sim Não

3) O Cronograma é um instrumento efetivo para controle de prazos ?

Sim Não

4) O Relatório de Custos serve para definir e controlar o orçamento de projeto ?

Sim Não

5) A Lista de Riscos e os 3 princípios (Prototipação, Adiantamento das atividades de risco e o processo em V) servem para focar os esforços de resolução dos riscos que ameaçam o sucesso do projeto ?

Sim Não

6) Você usou mais algum documento, além dos sugeridos pelo método, para o planejamento e gerenciamento dos projeto quanto a custos, prazos e funcionalidade ?

Sim Não

Se sim, qual : FERRAMENTA CASE, PLANILHA ELETRÔNICA

7) Tomando como base a sua experiência, como você classifica o método proposto quanto a Complexidade ?

Muito Simples Simples Regular Complexo Muito Complexo

8) Tomando como base a sua experiência, como você classifica o método proposto quanto a Eficiência ?

Ótimo Bom Regular Ruim Péssimo

9) Você recomendaria o uso do método aos seus colegas?

Sim Não

Questionário de Avaliação do Método de Planejamento e Gerenciamento de Pequenos Projetos de Software

Entrevistado: Marcos Kellin de Souza

Cargo: Aluno

Instituição: Escola Técnica da UFPR

Projeto: Manuseio OnLine

Data: 29/09/99 Assinatura: Marcos Kellin

Resolução de Riscos: Significa resolver um problema em potencial ou contorná-lo.

1) Tem-se por hipótese que a resolução dos riscos e o controle dos custos, prazos e funcionalidades aumentam a probabilidade de sucesso do projeto de software.

Você concorda com esta afirmação ?

Sim Não

2) O Relatório de Requisitos serve para definir e controlar o conjunto das funcionalidades do projeto ?

Sim Não

3) O Cronograma é um instrumento efetivo para controle de prazos ?

Sim Não

4) O Relatório de Custos serve para definir e controlar o orçamento de projeto ?

Sim Não

5) A Lista de Riscos e os 3 princípios (Prototipação, Adiantamento das atividades de risco e o processo em V) servem para focar os esforços de resolução dos riscos que ameaçam o sucesso do projeto ?

Sim Não

6) Você usou mais algum documento, além dos sugeridos pelo método, para o planejamento e gerenciamento dos projeto quanto a custos, prazos e funcionalidade ?

Sim Não

Se sim, qual : _____

7) Tomando como base a sua experiência, como você classifica o método proposto quanto a Complexidade ?

Muito Simples Simples Regular Complexo Muito Complexo

8) Tomando como base a sua experiência, como você classifica o método proposto quanto a Eficiência ?

Ótimo Bom Regular Ruim Péssimo

9) Você recomendaria o uso do método aos seus colegas?

Sim Não