KAREN SANDHOF

**FATORES HUMANOS NO PROCESSO DE DESENVOLVIMENTO DE
SOFTWARE: UM ESTUDO VISANDO QUALIDADE**

Dissertação apresentada à Escola
Politécnica da Universidade de São Paulo
para obtenção do título de Mestre em
Engenharia

Edição Revisada

São Paulo

2004

KAREN SANDHOF

**FATORES HUMANOS NO PROCESSO DE DESENVOLVIMENTO DE
SOFTWARE: UM ESTUDO VISANDO QUALIDADE**

Dissertação apresentada à Escola
Politécnica da Universidade de São Paulo
para obtenção do título de Mestre em
Engenharia

Área de Concentração:
Computação e Sistemas Digitais

Orientadora:
Profa. Dra. Lucia Vilela Leite Filgueiras

Edição Revisada

São Paulo

2004

Este exemplar foi revisado e alterado em relação à versão original, sob responsabilidade única do autor e com a anuência de seu orientador.

São Paulo, 03 de dezembro de 2004

Assinatura do autor

Assinatura do orientador

FICHA CATALOGRÁFICA – EDIÇÃO REVISADA

Sandhof, Karen

**Fatores Humanos no Processo de Desenvolvimento de Software: Um Estudo Visando Qualidade. São Paulo, 2004.
185p.**

**Dissertação (Mestrado) – Escola Politécnica da Universidade de São Paulo.
Departamento de Engenharia de Computação e Sistemas Digitais.**

1. Qualidade de Software 2. Fatores Humanos

I. Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia de Computação e Sistemas Digitais. II. t

AGRADECIMENTOS

A Deus por ter me dado saúde física e mental para concluir este trabalho.

A minha orientadora Profa. Dra. Lucia, pela orientação, dedicação e paciência ao longo destes anos de trabalho.

A minha família por ter me dado apoio e compreensão durante esta longa jornada.

Aos amigos que se mantiveram sempre prontos e dispostos a colaborar e compreender minhas ausências.

A Itautec Philco pelo apoio em permitir que a pesquisa fosse realizada em suas dependências, com seus desenvolvedores bem como pela possibilidade de divulgação destes dados.

A todos aqueles que, direta ou indiretamente, colaboraram na execução deste trabalho.

SUMÁRIO

RESUMO

ABSTRACT

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE GRÁFICOS

1	INTRODUÇÃO	18
1.1	MOTIVAÇÃO.....	18
1.2	PERSPECTIVA DE CONTRIBUIÇÃO	23
1.3	METODOLOGIA DE TRABALHO	24
1.3.1	<i>Créditos</i>	24
1.3.2	<i>Pesquisa de Campo</i>	26
1.4	TRABALHOS CORRELATOS	26
1.5	ESTRUTURA DO TRABALHO.....	27
2	FATORES HUMANOS NA MELHORIA DA QUALIDADE DE SOFTWARE 29	
2.1	MODELOS DE MELHORIA DE QUALIDADE	30
2.2	ISO 9000-3	31
2.3	CAPABILITY MATURITY MODEL FOR SOFTWARE (SW-CMM).....	32
2.4	SYSTEMS ENGINEERING CAPABILITY MATURITY MODEL (SE-CMM)	34
2.5	ISO/IEC TR 15504 - SPICE	35
2.6	PEOPLE CAPABILITY MATURITY MODEL (P-CMM)	36
2.7	PSP	39
2.8	CMMI	42
2.9	ESTUDO COMPARATIVO	44
3	FATORES CONDICIONANTES NO AMBIENTE PROFISSIONAL	46
3.1	FATORES INDIVIDUAIS E ORGANIZACIONAIS	46
3.1.1	<i>Peopleware</i>	47
3.1.2	<i>Inteligência Emocional</i>	51
3.1.3	<i>Comunicação</i>	55
4	ERRO HUMANO	57
4.1	ERRO HUMANO	57
4.2	NATUREZA DO ERRO HUMANO.....	58
4.3	CLASSIFICAÇÃO DO ERRO	59
4.4	ERRO HUMANO E DEFEITOS NO SOFTWARE	64
5	METODOLOGIA DE ANÁLISE.....	68
5.1	METODOLOGIA DE COLETA E ANÁLISE DE DADOS	69
5.2	DESCRIÇÃO DO PROCEDIMENTO.....	69
5.2.1	<i>Elaboração do questionário</i>	69

5.2.2	Questionário	70
5.2.3	Divulgação	75
5.2.4	Coleta	75
5.2.5	Análise	75
5.2.6	Obtenção dos Resultados	76
5.2.7	Detalhamento do questionário	76
6	APRESENTAÇÃO DE RESULTADOS	83
6.1	APRESENTAÇÃO DA AMOSTRA	83
6.2	RESULTADOS.....	83
6.3	CRUZAMENTO DE RESULTADOS	98
6.4	CRUZAMENTO DE RESULTADOS BASEADOS NOS TIPOS DE ERRO	106
6.5	RESULTADOS POR INDIVÍDUO	111
6.6	ANÁLISE CONCLUSIVA.....	157
7	CONCLUSÕES	159
8	ANEXOS	161
	ANEXO I – QUESTIONÁRIO E RESULTADOS DA ETAPA PRELIMINAR	161
	Questionário da Etapa Preliminar	161
	Obtenção dos Resultados	166
	Resultados da Etapa Preliminar	166
	ANEXO II - DECLARAÇÃO DE AUTORIZAÇÃO DOS DADOS DA PESQUISA COLETADA	178
9	REFERÊNCIAS BIBLIOGRÁFICAS	179
10	BIBLIOGRAFIA PESQUISADA	182

R E S U M O

Os modelos para melhoria da qualidade de software são fundamentados na tríade processo-pessoas-tecnologia. Este trabalho busca a relação entre dois destes pilares da qualidade de software: o processo de desenvolvimento e o ser humano. Este trabalho avalia como os modelos de qualidade de software consideram o ser humano e caracteriza a inserção de defeitos no software por seus desenvolvedores como erros humanos cometidos no processo de desenvolvimento. Fatores humanos condicionantes do desempenho são identificados e, através de pesquisa de campo cujo objetivo é explorar a realidade do processo de desenvolvimento de software, analisam-se os fatores humanos que podem afetar a qualidade de software, principalmente no que diz respeito às características de funcionalidade e confiabilidade.

ABSTRACT

Software quality improvement models are based on the triplet process-people-technology. This research works on the relationship between two of these fundamental structures of software quality: human beings and the development process. This work evaluates how software quality models consider human beings and characterizes defect insertion in software products by developers as human errors throughout the development process. Some human factors are identified as performance-shaping and, by the means of a field study intended to explore the reality of software development, they are analyzed as influences on software quality, mainly respective to functionality and reliability characteristics.

LISTA DE FIGURAS

FIGURA 1 – OS TRÊS COMPONENTES BÁSICOS DA QUALIDADE.....	2
FIGURA 2 – DIMENSÕES CRÍTICAS DA CAPACIDADE ORGANIZACIONAL.....	12
FIGURA 3 – DESENVOLVIMENTO DA AVALIAÇÃO DO PROCESSO DE SOFTWARE.....	18
FIGURA 4 – NÍVEIS DE MATURIDADE E CATEGORIAS DE PROCESSOS ASSINALADAS PELO P-CMM.....	21
FIGURA 5 – NÍVEIS DE MATURIDADE DO PSP.....	24
FIGURA 6 – MODELOS DE ABORDAGENS DO CMMI	27
FIGURA 7 – QUADRO COMPARATIVO DOS MODELOS DE QUALIDADE	28
FIGURA 8 – FATORES INDIVIDUAIS E ORGANIZACIONAIS QUE AFETAM O DESEMPENHO	30
FIGURA 9 – ESTRUTURA DO PEOPLEWARE.....	31
FIGURA 10 – MODELO DE ROTATIVIDADE DE BARTOL	34
FIGURA 11 - ALGORITMO PARA DISTINGUIR A VARIEDADE DE COMPORTAMENTO INTENCIONAL	42
FIGURA 12 - CLASSIFICAÇÃO DOS TIPOS DE ERRO PRIMÁRIO DE ACORDO COM OS ESTÁGIOS COGNITIVOS EM QUE ELAS OCORREM.	43
FIGURA 13 – ESQUEMA DA DINÂMICA DO ERRO NAS TRÊS DIMENSÕES DEFINIDAS POR REASON	45
FIGURA 14 – RESUMO DOS PRINCIPAIS TÓPICOS DOS MODOS DE FALHA PARA CADA UM DOS TRÊS NÍVEIS DE DESEMPENHO	46
FIGURA 15 – INJEÇÃO DE ERROS DE DESENVOLVIMENTO E DEPURAÇÃO DE SOFTWARE	49
FIGURA 16 – FATORES HUMANOS NAS CAUSAS DE ERROS DE SOFTWARE.....	50
FIGURA 17 – FATORES DE CONTEXTUALIZAÇÃO INDEPENDENTES DO TEMPO	55
FIGURA 18 – FATORES DIÁRIOS	56
FIGURA 19 – FATORES RELACIONADOS A CADA ERRO ENCONTRADO	57

LISTA DE TABELAS

TABELA 1 – TOTAIS ANALISADOS	66
TABELA 2 – DISTRIBUIÇÃO DAS POSSIBILIDADES DE ESTADO EMOCIONAL EM POSITIVO E NEGATIVO	75
TABELA 3 – TOTAIS OBTIDOS SOBRE O ESTADO EMOCIONAL DOS DESENVOLVEDORES	75
TABELA 4 – CAUSAS DE ERROS DISTRIBUÍDAS PELA CARGA DE TRABALHO.....	84
TABELA 5 – TOTAIS DE ERROS PELA CARGA DE TRABALHO	85
TABELA 6 – DISTRIBUIÇÃO DE ERROS PELA CAUSA EM PERCENTUAIS	86

LISTA DE GRÁFICOS

GRÁFICO 1 – GRAU DE CONTROLE DE PROCESSOS	67
GRÁFICO 2 – COMUNICAÇÃO ENTRE OS DESENVOLVEDORES	68
GRÁFICO 3 – POLÍTICAS E SISTEMAS DE RECURSOS HUMANOS	68
GRÁFICO 4 – CAPACITAÇÃO GERENCIAL	69
GRÁFICO 5 – ABERTURA À OPINIÃO DE FUNCIONÁRIOS.....	69
GRÁFICO 6 – RECONHECIMENTO PROFISSIONAL AO DESENVOLVEDOR	70
GRÁFICO 7 – RELACIONAMENTO ENTRE A EQUIPE	71
GRÁFICO 8 – MOTIVAÇÃO DO DESENVOLVEDOR NO PROJETO	71
GRÁFICO 9 – DIFICULDADES ENCONTRADAS NO PROJETO	72
GRÁFICO 10 – ETAPAS DE PROJETO DE SOFTWARE.....	73
GRÁFICO 11 – PAPÉIS E RESPONSABILIDADES ATRIBUÍDAS AO DESENVOLVEDOR.....	73
GRÁFICO 12 – PROBLEMAS ENFRENTADOS PELO DESENVOLVEDOR	74
GRÁFICO 13 – ESTADO EMOCIONAL APRESENTADO PELOS DESENVOLVEDORERS	75
GRÁFICO 14 – CARGA DE TRABALHO.....	76
GRÁFICO 15 – RESPONSÁVEL PELA IDENTIFICAÇÃO DO ERRO	76
GRÁFICO 16 – FASE DO DESENVOLVIMENTO EM QUE O ERRO FOI COMETIDO	77
GRÁFICO 17 – TEMPO DE EXISTÊNCIA DO ERRO DETECTADO.....	78
GRÁFICO 18 – MOTIVO DA IDENTIFICAÇÃO DO ERRO.....	78
GRÁFICO 19.1 – TIPO DE ERRO IDENTIFICADO	79
GRÁFICO 19.2 - OUTROS TIPOS DE ERRO IDENTIFICADOS E RELATADOS E IDENTIFICADOS.....	80
GRÁFICO 20 – CAUSAS DO ERRO IDENTIFICADO	80
GRÁFICO 21 – CARACTERÍSTICAS DE QUALIDADE DE SOFTWARE PREJUDICADAS PELA PRESENÇA DE ERROS	81

GRÁFICO 22 – CONTROLE DE PROCESSOS VERSUS QUANTIDADE DE ERROS IDENTIFICADOS.....	82
GRÁFICO 23 – DIFICULDADES ENCONTRADAS VERSUS A ETAPA DE DESENVOLVIMENTO	83
GRÁFICO 24 – DIFICULDADES ENCONTRADAS VERSUS A ETAPA DE DESENVOLVIMENTO	84
GRÁFICO 25 – CARGA DE TRABALHO VERSUS A CAUSA DE ERROS	85
GRÁFICO 26 – ERRO DETECTADO VERSUS CAUSA	86
GRÁFICO 27 – CARACTERÍSTICAS DE QUALIDADE QUE SOFRERAM IMPACTOS	87
GRÁFICO 28 – TEMPO DE EXISTÊNCIA DO ERRO VERSUS A FASE EM QUE FOI COMETIDO	88
GRÁFICO 29 – TEMPO DE EXISTÊNCIA DO ERRO VERSUS A FASE EM QUE FOI COMETIDO	89
GRÁFICO 30 – TIPO DE ERRO VERSUS A COMUNICAÇÃO	90
GRÁFICO 31 – TIPO DE ERRO VERSUS SATISFAÇÃO COM AS POLÍTICAS ORGANIZACIONAIS.....	91
GRÁFICO 32 – TIPO DE ERRO VERSUS A LIDERANÇA	91
GRÁFICO 33 – TIPO DE ERRO VERSUS RECONHECIMENTO PROFISSIONAL	92
GRÁFICO 34 – TIPO DE ERRO VERSUS RELACIONAMENTO ENTRE DESENVOLVEDORES	93
GRÁFICO 35 – MOTIVAÇÃO DA DESENVOLVEDORA JOANA	95
GRÁFICO 36 – DIFICULDADES DA DESENVOLVEDORA JOANA.....	95
GRÁFICO 37 – FASES DE TRABALHO DA DESENVOLVEDORA JOANA.....	95
GRÁFICO 38 – PAPÉIS E RESPONSABILIDADES DA DESENVOLVEDORA JOANA.....	96
GRÁFICO 39 – PROBLEMAS DA DESENVOLVEDORA JOANA.....	96
GRÁFICO 40 – ESTADO EMOCIONAL DA DESENVOLVEDORA JOANA	96
GRÁFICO 41 – CARGA DE TRABALHO DA DESENVOLVEDORA JOANA	97
GRÁFICO 42 – MOTIVAÇÃO DA DESENVOLVEDORA MARIA	97

GRÁFICO 43 – DIFICULDADES DA DESENVOLVEDORA MARIA	98
GRÁFICO 44 – FASES DE TRABALHO DA DESENVOLVEDORA MARIA	98
GRÁFICO 45 – PAPÉIS E RESPONSABILIDADES DA DESENVOLVEDORA MARIA	98
GRÁFICO 46 – PROBLEMAS DA DESENVOLVEDORA MARIA	99
GRÁFICO 47 – ESTADO EMOCIONAL DA DESENVOLVEDORA MARIA.....	99
GRÁFICO 48 – CARGA DE TRABALHO DA DESENVOLVEDORA MARIA	99
GRÁFICO 49 – IDENTIFICADOR X MOTIVO - DESENVOLVEDORA MARIA	100
GRÁFICO 50 – FASE DO ERRO - DESENVOLVEDORA MARIA.....	100
GRÁFICO 51 – TEMPO DE EXISTÊNCIA DO ERRO - DESENVOLVEDORA MARIA	101
GRÁFICO 52 – TIPO DE ERRO - DESENVOLVEDORA MARIA.....	101
GRÁFICO 53 – CAUSAS DE ERRO - DESENVOLVEDORA MARIA	101
GRÁFICO 54 – CARACTERÍSTICAS DE QUALIDADE AFETADAS PELOS ERROS - DESENVOLVEDORA MARIA	102
GRÁFICO 55 – MOTIVAÇÃO DA DESENVOLVEDORA ELISA	102
GRÁFICO 56 – DIFICULDADES DA DESENVOLVEDORA ELISA.....	103
GRÁFICO 57 – FASE DE TRABALHO DA DESENVOLVEDORA ELISA	103
GRÁFICO 58– PAPÉIS E RESPONSABILIDADES DA DESENVOLVEDORA ELISA	103
GRÁFICO 59– PROBLEMAS DA DESENVOLVEDORA ELISA.....	104
GRÁFICO 60 – ESTADO EMOCIONAL DA DESENVOLVEDORA ELISA	104
GRÁFICO 61 – CARGA DE TRABALHO DA DESENVOLVEDORA ELISA	104
GRÁFICO 62 – IDENTIFICADOR DO ERRO X MOTIVO - DESENVOLVEDORA ELISA	105
GRÁFICO 63 – FASE DO ERRO - DESENVOLVEDORA ELISA.....	105
GRÁFICO 64 – TEMPO DE EXISTÊNCIA DO ERRO- DESENVOLVEDORA ELISA	105
GRÁFICO 65 – TIPOS DE ERRO - DESENVOLVEDORA ELISA.....	106
GRÁFICO 66 – CAUSAS DO ERRO - DESENVOLVEDORA ELISA.....	106

GRÁFICO 67 – CARACTERÍSTICAS DE QUALIDADE AFETADAS PELOS ERROS - DESENVOLVEDORA ELISA	106
GRÁFICO 68 – MOTIVAÇÃO DO DESENVOLVEDOR JOÃO	107
GRÁFICO 69 – DIFICULDADES DO DESENVOLVEDOR JOÃO	107
GRÁFICO 70 – FASE DE TRABALHO DO DESENVOLVEDOR JOÃO.....	108
GRÁFICO 71– PAPÉIS E RESPONSABILIDADES DO DESENVOLVEDOR JOÃO	108
GRÁFICO 72 – PROBLEMAS DO DESENVOLVEDOR JOÃO	108
GRÁFICO 73 - ESTADO EMOCIONAL DO DESENVOLVEDOR JOÃO	109
GRÁFICO 74 – CARGA DE TRABALHO DO DESENVOLVEDOR JOÃO	109
GRÁFICO 75 – IDENTIFICADOR X MOTIVOS - DESENVOLVEDOR JOÃO.....	109
GRÁFICO 76 – FASE EM QUE O ERRO FOI COMETIDO - DESENVOLVEDOR JOÃO	110
GRÁFICO 77– TEMPO DE EXISTÊNCIA DO ERRO - DESENVOLVEDOR JOÃO	110
GRÁFICO 78– TIPOS DE ERRO - DESENVOLVEDOR JOÃO	110
GRÁFICO 79 – CAUSAS DE ERRO - DESENVOLVEDOR JOÃO	111
GRÁFICO 80 – CARACTERÍSTICAS DE QUALIDADE AFETADAS PELOS ERROS - DESENVOLVEDOR JOÃO	111
GRÁFICO 81 – MOTIVAÇÃO DO DESENVOLVEDOR RAFAEL	112
GRÁFICO 82 – DIFICULDADES DO DESENVOLVEDOR RAFAEL.....	112
GRÁFICO 83 – FASES DE TRABALHO DO DESENVOLVEDOR RAFAEL.....	112
GRÁFICO 84 – PAPÉIS E RESPONSABILIDADES DO DESENVOLVEDOR RAFAEL.....	113
GRÁFICO 85 – PROBLEMAS DO DESENVOLVEDOR RAFAEL.....	113
GRÁFICO 86 – ESTADO EMOCIONAL DO DESENVOLVEDOR RAFAEL	113
GRÁFICO 87 – CARGA DE TRABALHO DO DESENVOLVEDOR RAFAEL	114
GRÁFICO 88 – IDENTIFICAR X MOTIVO - DESENVOLVEDOR RAFAEL	114
GRÁFICO 89 – FASES EM QUE O ERRO FOI COMETIDO - DESENVOLVEDOR RAFAEL ...	114

GRÁFICO 90 – TEMPO DE EXISTÊNCIA DO ERRO -DESENVOLVEDOR RAFAEL	115
GRÁFICO 91 – TIPOS DE ERRO - DESENVOLVEDOR RAFAEL.....	115
GRÁFICO 92 – CAUSAS DE ERRO - DESENVOLVEDOR RAFAEL	115
GRÁFICO 93 – CARACTERÍSTICAS DE QUALIDADE AFETADAS PELOS ERROS - DESENVOLVEDOR RAFAEL	116
GRÁFICO 94 – MOTIVAÇÃO DO DESENVOLVEDOR FERNANDO	116
GRÁFICO 95 – DIFICULDADES DO DESENVOLVEDOR FERNANDO	117
GRÁFICO 96 – FASES DE TRABALHO DO DESENVOLVEDOR FERNANDO	117
GRÁFICO 97 – PAPÉIS E RESPONSABILIDADES DO DESENVOLVEDOR FERNANDO	117
GRÁFICO 98 – PROBLEMAS DO DESENVOLVEDOR FERNANDO	118
GRÁFICO 99 – ESTADO EMOCIONAL DO DESENVOLVEDOR FERNANDO	118
GRÁFICO 100 – CARGA DE TRABALHO DO DESENVOLVEDOR FERNANDO.....	118
GRÁFICO 101 – IDENTIFICADOR X MOTIVO - DESENVOLVEDOR FERNANDO	119
GRÁFICO 102 – FASE EM QUE O ERRO FOI COMETIDO - DESENVOLVEDOR FERNANDO	119
GRÁFICO 103 – TEMPO DE EXISTÊNCIA DO ERRO - DESENVOLVEDOR FERNANDO	119
GRÁFICO 104 – TIPOS DE ERRO - DESENVOLVEDOR FERNANDO	120
GRÁFICO 105 – CAUSAS DE ERRO - DESENVOLVEDOR FERNANDO.....	120
GRÁFICO 106 – CARACTERÍSTICAS DE QUALIDADE AFETADAS PELOS ERROS - DESENVOLVEDOR FERNANDO	120
GRÁFICO 107– MOTIVAÇÃO DA DESENVOLVEDORA RITA.....	121
GRÁFICO 108 – DIFICULDADES DA DESENVOLVEDORA RITA	122
GRÁFICO 109 – FASES DE TRABALHO DA DESENVOLVEDORA RITA	122
GRÁFICO 110 – PAPÉIS E RESPONSABILIDADES DA DESENVOLVEDORA RITA	122
GRÁFICO 111 – PROBLEMAS DA DESENVOLVEDORA RITA	123
GRÁFICO 112 – ESTADO EMOCIONAL DA DESENVOLVEDORA RITA.....	123

GRÁFICO 113– CARGA DE TRABALHO DA DESENVOLVEDORA RITA.....	123
GRÁFICO 114 – IDENTIFICADOR X MOTIVO - DESENVOLVEDORA RITA.....	124
GRÁFICO 115 – FASE EM QUE O ERRO FOI COMETIDO - DESENVOLVEDORA RITA	124
GRÁFICO 116– TEMPO DE EXISTÊNCIA DO ERRO - DESENVOLVEDORA RITA.....	124
GRÁFICO 117– TIPOS DE ERRO - DESENVOLVEDORA RITA	125
GRÁFICO 118 – CAUSAS DE ERRO - DESENVOLVEDORA RITA.....	125
GRÁFICO 119 – CARACTERÍSTICAS DE QUALIDADE AFETADAS PELOS ERROS- DESENVOLVEDORA RITA.....	125
GRÁFICO 120– MOTIVAÇÃO DA DESENVOLVEDORA ANTÔNIA	126
GRÁFICO 121– DIFICULDADES DA DESENVOLVEDORA ANTÔNIA.....	126
GRÁFICO 122– FASES DE TRABALHO DA DESENVOLVEDORA ANTÔNIA	127
GRÁFICO 123– PAPÉIS E RESPONSABILIDADES DA DESENVOLVEDORA ANTÔNIA	127
GRÁFICO 124– PROBLEMAS DA DESENVOLVEDORA ANTÔNIA	127
GRÁFICO 125 – ESTADO EMOCIONAL DA DESENVOLVEDORA ANTÔNIA.....	128
GRÁFICO 126– CARGA DE TRABALHO DA DESENVOLVEDORA ANTÔNIA.....	128
GRÁFICO 127 – IDENTIFICADOR X MOTIVO - DESENVOLVEDORA ANTÔNIA.....	128
GRÁFICO 128 – FASE EM QUE O ERRO FOI COMETIDO - DESENVOLVEDORA ANTÔNIA.....	129
GRÁFICO 129 – TEMPO DE EXISTÊNCIA DO ERRO DA DESENVOLVEDORA ANTÔNIA ..	129
GRÁFICO 130 – TIPOS DE ERRO - DESENVOLVEDORA ANTÔNIA.....	129
GRÁFICO 131 – CAUSAS DE ERRO - DESENVOLVEDORA ANTÔNIA	130
GRÁFICO 132 – CARACTERÍSTICAS DE QUALIDADE AFETADAS PELOS ERROS - DESENVOLVEDORA ANTÔNIA.....	130
GRÁFICO 133 – MOTIVAÇÃO DA DESENVOLVEDORA BEATRIZ	131
GRÁFICO 134 – DIFICULDADES DA DESENVOLVEDORA BEATRIZ.....	131
GRÁFICO 135 – FASES DE TRABALHO DA DESENVOLVEDORA BEATRIZ.....	131

GRÁFICO 136 – PAPÉIS E RESPONSABILIDADES DA DESENVOLVEDORA BEATRIZ.....	132
GRÁFICO 137 – PROBLEMAS DA DESENVOLVEDORA BEATRIZ.....	132
GRÁFICO 138 –ESTADO EMOCIONAL DA DESENVOLVEDORA BEATRIZ.....	132
GRÁFICO 139 – CARGA DE TRABALHO DA DESENVOLVEDORA BEATRIZ.....	133
GRÁFICO 140 – IDENTIFICADOR X MOTIVO - DESENVOLVEDORA BEATRIZ.....	133
GRÁFICO 141 – FASE EM QUE O ERRO FOI COMETIDO - DESENVOLVEDORA BEATRIZ	133
GRÁFICO 142 – TEMPO DE EXISTÊNCIA DO ERRO - DESENVOLVEDORA BEATRIZ.....	134
GRÁFICO 143 – TIPOS DE ERRO - DESENVOLVEDORA BEATRIZ.....	134
GRÁFICO 144 – CAUSAS DE ERRO - DESENVOLVEDORA BEATRIZ	134
GRÁFICO 145 – CARACTERÍSTICAS DE QUALIDADE AFETADAS PELOS ERROS - DESENVOLVEDORA BEATRIZ.....	135
GRÁFICO 146 – MOTIVAÇÃO DO DESENVOLVEDOR CÉSAR	135
GRÁFICO 147 – DIFICULDADES DO DESENVOLVEDOR CÉSAR.....	136
GRÁFICO 148 – FASES DE TRABALHO DO DESENVOLVEDOR CÉSAR.....	136
GRÁFICO 149 – PAPÉIS E RESPONSABILIDADES DO DESENVOLVEDOR CÉSAR	136
GRÁFICO 150 – PROBLEMAS DO DESENVOLVEDOR CÉSAR.....	137
GRÁFICO 151 – ESTADO EMOCIONAL DO DESENVOLVEDOR CÉSAR	137
GRÁFICO 152– CARGA DE TRABALHO DO DESENVOLVEDOR CÉSAR	137
GRÁFICO 153 – IDENTIFICADOR X MOTIVO - DESENVOLVEDOR CÉSAR	138
GRÁFICO 154 – FASE EM QUE O ERRO FOI COMETIDO - DESENVOLVEDOR CÉSAR	138
GRÁFICO 155 – TEMPO DE EXISTÊNCIA DO ERRO - DESENVOLVEDOR CÉSAR	138
GRÁFICO 156 – TIPOS DE ERRO - DESENVOLVEDOR CÉSAR.....	139
GRÁFICO 157 – CAUSAS DE ERRO - DESENVOLVEDOR CÉSAR	139
GRÁFICO 158 – CARACTERÍSTICAS DE QUALIDADE AFETADAS PELOS ERROS - DESENVOLVEDOR CÉSAR	139

1 INTRODUÇÃO

Este trabalho visa o estudo dos fatores humanos que têm influência no desenvolvimento de software e impacto na qualidade deste software. Esta introdução apresenta o tema ao leitor e orienta a leitura através dos capítulos do texto. Para tanto, este capítulo tem as seguintes seções:

- Discussão da Motivação para realizar este trabalho
- Apresentação da Perspectiva de Contribuição
- Breve descrição da Metodologia de Análise usada para desenvolver a pesquisa
- Apresentação da estrutura do trabalho, para os capítulos seguintes

1.1 Motivação

O tema da Qualidade de Software e os modelos existentes que gerenciam e indicam esta qualidade têm sido objeto de muitos estudos dada a importância atual da indústria de software. Um dos fatores intrigantes a respeito deste assunto é saber como o ser humano, que é o elemento que gerencia e promove esta qualidade, é tratado por estes modelos.

A qualidade de um produto de software está fortemente relacionada ao ser humano: ele é o responsável pelas etapas do desenvolvimento de software desde a identificação de requisitos até a homologação final do produto. Desta forma acredita-se que as características inerentes ao ser humano e o seu comportamento como elemento da produção de software devam ser levados em consideração ao se estabelecerem as regras para manter o processo de software numa curva ascendente de qualidade. No desenvolvimento de software, o erro humano pode ser traduzido numa ação errônea que se manifesta como um defeito de software, e é neste Universo que Nakamura et al concluem em seu artigo sobre atividades de prevenção de

defeitos de software que “é importante lidar com o erro humano para garantir a Qualidade de Software.” (NAKAMURA, 1991, p. 360)

Os processos utilizados para desenvolver e manter um software precisam de pessoas para trabalhar efetivamente pela definição das tarefas necessárias, para a execução das tarefas, para o controle do seu andamento efetivo e na incorporação de lições aprendidas.

Fatores como a má remuneração, falta de reconhecimento, alta rotatividade, ambiente de trabalho ruim, falta de motivação, mau uso do poder, mau gerenciamento e má distribuição de tarefas atingem a produção de software de forma invisível aos olhos das empresas.

A maturidade de uma empresa em todas as tarefas essenciais ao sucesso de um projeto “deve ter foco em três componentes interrelacionados - pessoas, processos e tecnologia” (CURTIS, 1995, xiii). Na figura 1 a seguir, a relação entre os três componentes representa um ciclo e evidencia que o bom entrosamento entre os três alicerces determina um produto ou um serviço com qualidade.

Fonte: Adaptado de CURTIS, 1995, p xiii

Fig. 1 – Os três componentes básicos da qualidade

O elemento que pode ser o diferencial nas práticas de engenharia de software é o elemento humano. Este elemento humano, como cume da tríade processo-tecnologia-pessoas, baseia-se nos conhecimentos adquiridos no processo de desenvolvimento de software. Por esta razão o mercado de trabalho neste segmento se tornou mais competitivo e acirrado. As empresas já sabem que a qualidade de um produto de software está diretamente relacionada com a qualidade do processo

utilizado para desenvolvê-lo e também já perceberam que os programas de melhoria são voltados apenas aos processos e à tecnologia, não às pessoas. Existe atualmente a visão de que, para que as práticas adotadas atinjam o resultado esperado, mudanças também deverão ser feitas na forma como as pessoas são gerenciadas.

Por exemplo, equipes que desenvolvem produtos similares, ou simplesmente implementam componentes de software de igual funcionalidade de um já existente na empresa, produzido por outra equipe, mostram que falta de comunicação ou problemas de relacionamento podem ser fatores de desperdício de tempo e dinheiro dentro de uma organização. Segundo Katz (1978) toda organização humana depara-se com a tarefa de tentar reduzir a variabilidade, a instabilidade e a espontaneidade de atos humanos individuais.

Fatores como estes, comuns ao dia-a-dia de qualquer empresa de tecnologia, foram motivadores de ações visando a estruturação dos processos e a melhoria da qualidade de software.

O “processo de software é estabelecido pela definição de um pequeno número de atividades estruturadas que são aplicáveis para todos os projetos de software independente de seu tamanho ou complexidade.” (PRESSMAN, 2001, p. 23)

O processo de software é definido por seres humanos e em grande parte também executado por eles, portanto, os limites de cada um dos sub-processos, seus objetivos principais, seu modo de execução, seus parâmetros de qualidade, tudo isto é passível de ser afetado pelo ser humano. Um processo bastante formal, visto como diretrizes que regem o envolvimento do ser humano, proporciona uma melhor confiança na execução, que assim permitirá a concepção de um produto mais confiável, consistente e de melhor qualidade. Numa linha contrária a esta afirmação, modelos de desenvolvimento ágeis e flexíveis estabelecem premissas que atribuem à capacidade humana a condição de garantia da qualidade. (COCKBURN, 2001) Em qualquer um dos casos a relevância do fator humano é evidente.

As regras são vistas normalmente pelos desenvolvedores de software como imposições intransigentes que devem ser seguidas independentemente de seus propósitos, e que servem para definir formatos de execução de determinadas tarefas, por isso, a forma de estabelecimento destas regras é fundamental para sua eficiência dentro de uma organização, pois depende de fatores culturais e humanos.

É fato que a utilização de parâmetros e metodologias padroniza o trabalho de desenvolvedores de software, permitindo assim, que existam fatores comparáveis e possíveis de serem medidos. Por este motivo, uma organização começa a seguir indicadores de qualidade visíveis e passíveis de melhoria. No entanto, isso não é suficiente: “O processo pode proporcionar um *framework* vantajoso para grupos e indivíduos trabalharem em conjunto, mas o processo por si só não pode se sobrepor à falhas de competência, enquanto que a competência pode superar as extravagâncias do processo”. (COCKBURN, 2001, p.132)

Por outro lado, as pessoas são vulneráveis, criativas, inteligentes - e essenciais. Pessoas talentosas são mais do que simples recursos que se pode ter para realizar tarefas no processo de software e na garantia de sua qualidade. Elas são determinantes para a qualidade. São elas as responsáveis por controlar e desenvolver todas as tarefas cruciais no processo de desenvolvimento de software, desde a captura de requisitos, desenvolvimento, instalação e manutenção. Sendo estas as tarefas determinantes no sucesso de um projeto de software, bem como na qualidade que ele apresenta, temos a relação explícita, pouco levada em consideração: pessoas atreladas à qualidade. “Pessoas talentosas são o bem mais importante de uma organização.” (HUMPHREY, 1995, p. 92)

O processo de desenvolvimento de software depende do conhecimento e da experiência, e até da habilidade que se tem para resolver certos tipos de problemas. “Aspectos intelectuais e criativos do desenvolvimento de software somente podem ser executados pelo talento, habilidade, e pessoas motivadas.” (DEIMEL, 1996, p. 1) O que não se pode confundir é talento e habilidade com um processo criativo semelhante ao dos artistas. O desenvolvedor de software não é um artista que depende de inspiração e de estímulos para realizar seu trabalho no momento em que estas coisas acontecem, contudo “o primeiro passo para a inovação de um processo é através da criatividade”. (HUMPHREY, 1995, p. 149)

Se analisada como qualquer outra produção artística, a habilidade de desenvolvimento por si só fica atrelada à inspiração, ao momento, à vontade de um único indivíduo. A criatividade “é vista como um poder mágico vindo de uma inspiração divina.” (HUMPHREY, 1995, p. 149) Isto na produção de software torna-se inviável, pois uma única pessoa não pode determinar os prazos pela sua

vontade. “A criatividade é meramente um modo de pensamento que parece surgir de *insights*¹ repentinos, mas são precedidos de pressentimentos e indicações anteriores que são desenvolvidos através de um processo lógico ordenado.”(HUMPHREY, 1995, p. 149-50) Não se pode esperar que o comportamento de uma equipe desenvolvedora de software seja igual a de máquinas, pois toda criação depende de criatividade e talento, incluindo a de software.

Se analisado cuidadosamente, o processo de desenvolvimento de software é iniciado na geração e captura de idéias, o que é essencial ao processo. Segundo a teoria de Piaget² o pensamento não é o resultado automático de reflexos ou da intuição, mas sim uma operação flexível, que se desenvolve através da tentativa e do erro. (PIAGET, 1975). Baseando-se nesta captura de idéias, o analista começa o processo com a atividade essencial da Engenharia de Software: a abstração da informação. A compatibilidade daquilo que é produzido com o que se abstrai é fundamental, e, muitas vezes, torna-se difícil pelo grau de complexidade imposto pela abstração. Lidar com as restrições requer a avaliação de muitos conceitos e decisões. (BRADY, 1986). O que não se pode deixar de lado é o fato de existirem influências e interferências externas no trabalho do desenvolvedor que podem afetar o resultado final.

Este trabalho visa contribuir sobre como identificar e dimensionar fatores internos e externos ao ser humano que podem trazer impacto no desenvolvimento de software. Fatores externos à produção de software como motivação, reconhecimento, comunicação, organização e gerenciamento devem ser analisados e gerenciados para benefício das empresas. Também fatores internos como estado emocional, problemas e dificuldades enfrentados pelo desenvolvedor podem ser influenciadores na qualidade do que é produzido.

A falta de dimensionamento destes fatores cria problemas como: alta rotatividade de profissionais, perda do sistema crítico de conhecimento, crescimento do trabalho extra e também do estresse, trabalhos sem finalização, aumento do custo de serviços e do produto, entre outros. Atitudes e satisfação não são os únicos

¹ Do inglês - pensamentos

² Jean Piaget (1896-1980), psicólogo, pedagogo e filósofo. Através de uma psicologia do desenvolvimento da inteligência estabeleceu uma teoria sobre as quatro fases através das quais o indivíduo adquire aptidões para o raciocínio lógico.

identificadores necessários para se saber quais práticas devem ser introduzidas numa empresa.

A vulnerabilidade e a sensibilidade do ser humano a estes fatores podem ser determinantes na qualidade daquilo que se produz.

O desempenho humano é dependente de contexto, ou seja, há fatores que condicionam, favorável ou desfavoravelmente este desempenho e que podem modificar a probabilidade do erro humano.

Estes fatores foram estudados de forma extensa nas aplicações críticas do ponto de vista de segurança, como por exemplo em aeronaves e plantas nucleares, já que nestas aplicações o erro humano pode provocar catástrofes.

Embora o desempenho humano seja, como visto anteriormente, essencial para a qualidade de software, não se conhece um estudo que relacione o erro humano no processo de desenvolvimento a seus fatores condicionantes.

O objetivo deste trabalho é, portanto, identificar e analisar os fatores humanos que podem afetar a qualidade de software, principalmente no que diz respeito às características de funcionalidade e confiabilidade.

Deseja-se, com este trabalho, gerar um mecanismo para identificação destes fatores. As soluções para remediar as possíveis causas destes fatores estão fora deste escopo e acredita-se que as empresas, de posse do mecanismo de diagnóstico, tenham condições de solucionar os problemas apontados.

1.2 Perspectiva de Contribuição

Este trabalho pretende, portanto, analisar os fatores humanos na produção de software como contribuição para aumentar a qualidade e servir de base para uma possível investigação de mecanismos para que o ser humano não introduza defeitos no software que está produzindo. Desta forma, o trabalho propõe uma visão alternativa do assunto qualidade de software.

Pretende-se que os resultados obtidos nas investigações de campo realizadas na empresa Itautec Philco S/A contribuam substancialmente com fatores de qualidade que serão passíveis de análise crítica, já que a mesma está empenhada na melhoria do processo de desenvolvimento de software.

Este trabalho vem a contribuir também como uma fusão das pesquisas em Confiabilidade Operacional e Engenharia de Software no Laboratório de Tecnologia de Software da EPUSP.

1.3 Metodologia de Trabalho

Esta seção apresenta de forma resumida a metodologia usada para realizar a pesquisa. Os detalhes da pesquisa são apresentados no Capítulo 5.

1.3.1 Créditos

Como forma de atingir os objetivos desejados, o cumprimento dos créditos necessários foi realizado em áreas relacionadas à pesquisa.

A disciplina de “*Qualidade de Software: Conceitos e Paradigmas*” veio a contribuir com os conceitos de Qualidade de Software: modelos utilizados, normas estabelecidas, entre outros, o que permitiu a formação básica no assunto. As disciplinas, então obrigatórias, “*Tópicos sobre Fundamentos da Engenharia da Computação*” e “*Arquitetura de Computadores*” trouxeram a base fundamental do raciocínio lógico e operacional de sistemas computacionais. “*Princípios de Projeto de Software e Metodologia de Programação*” e “*Modelagem de Software Orientado a Objetos*” permitiram resgatar e aprofundar conceitos na área de Software. Isto levou ao questionamento das tecnologias hoje existentes, bem como o ponto de vista sob o qual são estudadas. A concepção e implementação da modelagem de software orientado a objetos sofreram uma visão crítica diferenciada do ponto de vista da garantia de qualidade através de atributos como reusabilidade, confiabilidade, disponibilidade, entre outros.

A disciplina “*Metodologia de Desenvolvimento de Interfaces Homem-Computador*” foi realizada numa fase mais avançada do projeto, o que permitiu a visão e o questionamento de um dos pontos mais cruciais do projeto de software que é a Interface Homem-Computador. E, por fim, a disciplina de “*Confiabilidade e Segurança de Sistemas Computacionais*” permitiu o contato com questões atuais de confiabilidade do ponto de vista de *segurança* e proporcionou a abordagem, discussão e aprofundamento do tema de Confiabilidade Humana, tanto na produção de software como na utilização de sistemas digitais.

Com o auxílio destas disciplinas, a pesquisa bibliográfica do assunto girou em torno dos seguintes tópicos:

1. *Comportamento Humano*: neste tópico, diversos assuntos relacionados a Neuroanatomia, Neurofisiologia e Psicologia foram pesquisados e estudados com o objetivo de identificar comportamentos humanos relevantes à produção de software, bem como os modos de observá-los. Conceitos como emoções, conhecimento e lógica foram estudados na tentativa de elucidar quais comportamentos interferem na produção de software e como identificá-los e quantificá-los.
2. *Modelos da Qualidade*: o estudo contemplou modelos e processos importantes para a qualidade de software, com o objetivo de compreender como tais modelos que buscam melhorar a qualidade tratam os fatores humanos.
3. *Confiabilidade e Erro Humano*: o objetivo do estudo deste tópico foi entender os mecanismos de erros humanos e como eles afetam a confiabilidade.
4. *Injeção de Defeitos X Qualidade de Produto*: aqui foram estudados mecanismos humanos de injeção de defeitos e mecanismos de prevenção dos mesmos.

Além destes assuntos, ao longo do desenvolvimento deste trabalho, uma bibliografia variada foi coletada e utilizada no enriquecimento de assuntos relevantes ao estudo realizado. As referências utilizadas trouxeram alicerces não só com relação

aos assuntos já enumerados e de domínio público, mas também para a concepção da pesquisa de campo.

1.3.2 Pesquisa de Campo

O envolvimento do aspecto humano nesta pesquisa fez necessária uma pesquisa de campo com o objetivo de explorar a realidade do processo de desenvolvimento de software. A pesquisa de campo foi realizada em duas etapas. Uma pesquisa preliminar teve o objetivo de identificar quais variáveis de contexto afetavam o desempenho humano, esta primeira pesquisa foi feita através de um questionário submetido aos desenvolvedores de software da empresa Itautech Philco S/A, localizada em São Paulo, Capital, no período de outubro a dezembro de 2002 e se apresenta na íntegra no Anexo I. Os dados desta versão mostraram, no entanto, que a verdadeira influência dos fatores no desempenho requeria um acompanhamento mais minucioso do processo de desenvolvimento. Assim, a segunda versão, mais focada e elaborada do questionário, foi feita no formato de páginas HTML e aplicada diariamente aos desenvolvedores da referida empresa por um período de 19 dias úteis que aconteceram entre julho e agosto de 2004.

1.4 Trabalhos Correlatos

A pesquisa bibliográfica identificou alguns trabalhos correlatos nas seguintes áreas de concentração: confiabilidade humana (DOUGHERTY, KENCKLUND, LASALA, JAVAUX, SMIDTS, VANDERHAGEN, SHARIT), prevenção de defeitos (NAKAMURA, SUTZKE), erro humano (REASON) e fatores humanos na engenharia de software (THOMAS et al., WIDMAIERidmaier).

Dos trabalhos pesquisados, os que foram estudados com maior ênfase foram os da Dra. Carol Smidts³ professora da Universidade de Maryland que tem como sua principal área de interesse a confiabilidade. Seus trabalhos giram em torno da busca de modelos matemáticos que adicionem o erro humano como uma das variáveis básicas, passíveis de enumeração e de influência nos números obtidos acerca da confiabilidade humana em tempo de desenvolvimento.

O trabalho aqui proposto difere dos estudados, principalmente dos trabalhos da Dra. Smidts, pois o foco deste é estudar os erros humanos cometidos no processo de desenvolvimento, visando avaliação de confiabilidade de software. A perspectiva do presente trabalho é a identificação e análise dos erros humanos visando melhoria do processo baseado no reconhecimento dos fatores de influência.

1.5 Estrutura do Trabalho

Este trabalho possui oito capítulos.

Capítulo 1 – Introdução

Na Introdução é feita uma breve descrição das questões que se pretende tratar, discutir e validar ao longo do trabalho.

Capítulo 2 – Fatores Humanos na Melhoria da Qualidade de Software

O segundo capítulo apresenta a abordagem do fator humano como item relevante à Qualidade de Software feita pelos principais modelos e normas de qualidade hoje existentes. O objetivo deste capítulo é mostrar como estes modelos tratam o fator humano e identificar neles elementos influenciadores do desempenho reconhecidos como tal na literatura de Qualidade de Software.

Capítulo 3 – Fatores Condicionantes no Ambiente Profissional

Este capítulo trata dos aspectos gerais do comportamento humano. Serão levados em consideração assuntos referentes ao gerenciamento de pessoas na área de tecnologia, carreira, motivação, maturidade organizacional, entre outros.

³ Profa. do Depto. De Materiais e Eng. Nuclear e do programa de Confiabilidade da Universidade de Maryland no College Park. Títulos de mestrado e doutorado em Eng. Física nos anos de 86 e 91 respectivamente, ambos na Université Libre de Bruxelles, na Bélgica

O objetivo deste capítulo é apresentar, com base na literatura sobre comportamento humano em ambientes profissionais, alguns fatores influenciadores de desempenho.

Capítulo 4 – Erro Humano

O quarto capítulo discorre sobre o assunto do Erro Humano. O intuito é descrever os principais conceitos de erro e suas divisões no que se refere ao ser humano, bem como identificar modelos e fatores que contribuam com a qualidade.

Capítulo 5 – Metodologia de Análise

Neste capítulo é apresentada a metodologia da pesquisa de campo, realizada através de um questionário. Ele é apresentado na íntegra com detalhes e justificativas.

Capítulo 6 – Resultados Obtidos

Neste capítulo são apresentados os resultados obtidos na pesquisa de campo realizada.

Capítulo 7 – Conclusões

Neste capítulo são apresentadas as conclusões do trabalho, na forma de elucidar as questões levantadas.

Capítulo 8 – Anexos

O capítulo 8 apresenta os documentos de referência do trabalho.

Capítulo 9 – Referências Bibliográficas

Neste capítulo são apresentadas as referências utilizadas.

Capítulo 10 – Bibliografias Pesquisadas

Neste capítulo são apresentadas as referências consultadas durante o processo de estudo dos temas abordados neste trabalho.

2 FATORES HUMANOS NA MELHORIA DA QUALIDADE DE SOFTWARE

Empresas produtoras de software vêm passando pelo desafio da melhoria da qualidade. Com o mercado consumidor de software cada vez mais exigente, muitas empresas perceberam que o real benefício está na melhoria do processo de desenvolvimento. A consciência de necessidade de melhoria contínua, da definição de processos, do gerenciamento, entre outros atributos, pode levar uma empresa a sair do caos e da desordem, e passar a um nível de maturidade significativo qualquer que seja o modelo de melhoria de qualidade adotado como referência.

A qualidade de software é baseada na tríade processo-tecnologia-pessoas. Esta triangulação envolve tudo aquilo que é relevante na produção de um software. Normas e modelos de qualidade abordam os processos de desenvolvimento, o mercado consumidor de software dita a tecnologia a ser utilizada, e não há regras ou padrões que tenham uma ênfase específica no ser humano.

A junção destes três itens fundamentais ao desenvolvimento de software conduz uma organização a um patamar de capacitação que eleva seus processos e produtos à qualidade desejada, como mostra o esquema da figura.2.

Fonte: Adaptada de ARMITAGE, 1993, p 2-2

Fig. 2 – Dimensões críticas da capacidade organizacional

Não é intenção dos modelos de melhoria de qualidade tratar o fator humano como item relevante da qualidade de software, assim como não é de obrigação dos mesmos ditar as tecnologias que devem ser incorporadas.

Por isso, o objetivo deste capítulo não é discutir os modelos e normas de qualidade, mas relacionar o processo de desenvolvimento que é o foco destes modelos, com o ser humano como fator que pode interferir neste processo.

Para elucidar esta relação faz-se necessário estudar estas normas e modelos mais utilizados com o foco em compreender como elas consideram o ser humano .

Este capítulo irá discutir os modelos de qualidade mais conhecidos (SW-CMM, SE-CMM, P-CMM, ISO 9000-3, SPICE (ISO/IEC TR 15504), PSP e CMMI) sob o foco dos fatores humanos. Com isto pretende-se identificar como eles tratam a importância e relevância do ser humano no processo de obtenção da qualidade de software.

2.1 Modelos de Melhoria de Qualidade

Os modelos a serem abordados surgiram porque o produto de software requer um processo, o que tornou evidente a necessidade de padrões que gerissem e viabilizassem da melhor forma possível estes processos.

Todos os modelos referidos têm em comum o estabelecimento de formas para garantia e gerenciamento da qualidade do software.

Cada um deles coloca suas regras e recomendações em exposição sem especificar as técnicas a serem seguidas, prevendo a adaptação a características específicas das organizações.

Pretende-se a seguir apresentar as normas e os modelos de qualidade mais conhecidos, sob a análise do fator humano. Não é parte do escopo deste trabalho detalhar os modelos propostos, visto que são bastante conhecidos. Para que este estudo tivesse base sólida, foram estudados todos os modelos referenciados.

2.2 ISO 9000-3

A norma NBR ISO 9000 trata as normas de gestão e garantia da qualidade. A parte 3 desta norma traz as diretrizes para a aplicação da NBR 19001 ao desenvolvimento, favorecimento e manutenção de software. Esta norma técnica é estabelecida pela ABNT – Associação Brasileira de Normas Técnicas.

A ISO 9000-3 é uma das normas mais conhecidas para a gestão e a garantia de qualidade de software. Seu objetivo é “fornecer orientação para a garantia da qualidade de software” (ABNT, 1993, p. 1). A estrutura do sistema de qualidade é abordada, bem como documentação, auditoria, atividades de ciclo de vida e itens de contrato.

A revisão da norma ISO 9000 feita em 2000 trouxe a abordagem de modelo por processos, e centrou-se na gestão de melhoria destes processos onde é reconhecido o ciclo de melhoria contínua que monitora e potencializa as evoluções. Esta revisão se estende para a norma ISO 9000-3.

Nesta norma, software é definido como sendo uma “criação intelectual compreendendo os programas, procedimentos, regras e qualquer documentação correlata à operação de um sistema de processamento de dados”. (ABNT, 1993, p. 2) “Criação intelectual”, nesta norma, é um termo vago e sem definição, mas que pressupõe a dependência do elemento humano.

Baetjer (1998) *apud* Pressman (2001) reafirma o conceito de que o processo de software é dependente justamente desta criação intelectual, pois “o software, como todo capital, é incorporado ao conhecimento, e porque este conhecimento é inicialmente disperso, implícito, oculto, e incompleto em larga escala, o desenvolvimento de software é um processo de aprendizado social.” (PRESSMAN, 2001, p. 19) Mesmo o trabalho técnico pode ser criativo, baseado no conhecimento, na análise de situações e alternativas e, conseqüentemente, na composição de soluções.

A norma ISO 9000-3 toma como base diretrizes contratuais da produção de software. fornecedor e cliente, contrato, administração e auditoria são algumas

palavras importantes e que mostram sua particular preocupação na abrangência do processo.

Como prerrogativas de qualidade, além da documentação que é comum a todos os modelos aqui levantados, tem-se o desenvolvimento contínuo da qualidade ao longo de todo o desenvolvimento do processo, ao invés de tentar medi-la no produto final.

A norma não referencia o ser humano para conquista da qualidade. A única situação em que se atrela qualidade de software a pessoas é no que diz respeito ao treinamento, pois o treinamento deve ser dado a “todo pessoal que executa atividades que influenciem a qualidade” (ABNT, 1993, p. 12). A norma explicita que determinadas tarefas devam ser realizadas por um pessoal qualificado com base na educação adequada, treinamento e experiência, sem que para isto existam maiores definições ou especificações.

Outro aspecto relacionado a seres humanos da ISO 9000-3 é o foco no usuário do produto de software. Um sistema gerado para um tipo específico de usuário torna-se melhor se suas necessidades e características típicas para seu tipo de perfil forem analisadas e levadas em consideração.

A menos destes aspectos, não há menção da possível influência do ser humano na qualidade, e muito menos do grau desta influência.

2.3 Capability Maturity Model for Software (SW-CMM)

O CMM é um modelo de maturidade, de origem no Software Engineering Institute (SEI) da Carnegie Mellon University (CMU) dos EUA, que surgiu como prática para melhorar a qualidade e a produtividade de software em resposta à pouca habilidade de gerenciar processos de software apresentada por inúmeras empresas. Nestas empresas, muitos sucessos eram atribuídos somente a esforços de equipes dedicadas, mas que não possuíam processos de softwares maduros.

O CMM é dividido em cinco níveis de maturidade: Inicial, Repetível, Definido, Gerenciado e Otimizado. Cada um dos níveis possui áreas chave de

processo (KPAs) que devem ser bordadas e garantidas pelas empresas que queiram atingir um determinado nível de maturidade.

“O CMM foi desenhado para guiar as empresas produtoras de software na escolha de estratégias para melhoria de processos pela determinação do processo corrente de maturidade e identificando algumas tarefas críticas para a qualidade do software e para a melhoria do processo” (PAULK, 1993, p.78).

O CMM é um modelo descritivo no que se refere aos importantes atributos que podem caracterizar uma empresa num determinado nível de maturidade. A maturidade de um processo de software “é a garantia de que um processo específico é explicitamente definido, gerenciado, dimensionado, controlado e efetivado” (PAULK, 1993, p. 97). O nível de maturidade estabelecido identifica a capacidade da empresa, seu potencial de crescimento e também indica a melhoria da produtividade e da qualidade como resultado da organização dos processos de software melhorados através do tempo.

O CMM não diz a uma empresa como melhorar nem especifica exatamente como conseguir um novo nível de maturidade. Ele deve ser adequado à realidade de cada situação empresarial, e para isto, é necessário um julgamento profissional adequado.

As melhorias ocorrem através de inovações nos processos existentes e pela inovação adquirida através da utilização de novas tecnologias e métodos.

Mas existem problemas culturais que muitas vezes se opõem à prática de estratégias e utilização de metodologias, à necessidade de disciplinar processos de software e à obtenção gradual da maturidade dos processos.

Quando o assunto é ser humano o CMM se posiciona claramente ao afirmar que: “Selecionar, contratar, desenvolver e/ou reter pessoas competentes são tarefas significantes para as empresas em todos os níveis de maturidade, mas que estão amplamente fora do escopo do CMM” (PAULK, 1993, p. 102).

Pôde-se perceber que este modelo também não prevê se os fatores humanos têm impacto na qualidade do software e nem como medir este impacto.

Mas quando se procura a causa dos problemas da adoção do CMM, geralmente a atribuição é dada muito mais a “problemas relacionados a pessoas” do que qualquer tipo de barreira tecnológica ou de implementação. (DEIMEL, 1996)

“O CMM por si enfatiza que as pessoas decretam o processo, logo isto significa que as pessoas, de alguma maneira, devem adquirir recursos necessários e treinamento para a melhoria do processo de software.” (DEIMEL, 1996, p. 1)

Todo processo precisa de pessoas que trabalhem na sua incorporação, construção e melhoria, esta é uma verdade às vezes esquecida na elaboração e execução de determinados processos.

Questões como habilidade e experiência são fundamentais neste modelo para a produção de software com altos índices de confiabilidade, fatores estes, portanto, que merecem atenção.

2.4 Systems Engineering Capability Maturity Model (SE-CMM)

O SE-CMM, proveniente do SEI/CMU, também segue a premissa de que a qualidade de um produto é uma função direta dos processos e da tecnologia utilizada para desenvolvê-lo e também, da capacidade das pessoas de realizarem este trabalho.

Ele se baseia na engenharia de sistemas, que é uma aplicação seletiva que concentra esforços para transformar uma necessidade operacional numa descrição para configuração de um sistema que melhor a satisfaça. Também procura integrar todos os parâmetros técnicos e torná-los compatíveis com a estrutura física funcional de maneira a otimizar a definição total do sistema e de seu design.

O SE-CMM descreve os estágios que cada processo atinge, como ele é definido, implementado e melhorado.

“O SE-CMM acredita que a qualidade de um produto é uma função direta do processo e da tecnologia usados para desenvolver o produto e a capacidade das pessoas de fazer o trabalho” (ARMITAGE, 1995, p. 13).

A base para o SE-CMM é o processo, ele é a primeira dimensão da capacidade organizacional. O foco no processo proporciona o prognóstico de desempenho. Mas o processo também é colocado como uma função de integração entre pessoas e tecnologia.

O SE-CMM não especifica a engenharia com a inclusão dos humanos, ou seja, ele se restringe às especialidades da engenharia consideradas necessárias e apropriadas para o desenvolvimento particular de um produto.

2.5 ISO/IEC TR 15504 - SPICE

Também o SPICE (ISO/IEC TR 15504:1998) é um modelo para avaliação de processos de software. Ele surgiu como forma das organizações se envolverem no planejamento, gerenciamento, implantação e evolução do software através da análise contínua do processo de software empregado por essas organizações nas etapas evolutivas. O esquema de avaliação dos processos no SPICE está relacionado na figura 3, onde a melhoria do processo é motivada pela determinação da capacidade organizacional e identificação de riscos.

Fonte: Adaptado de <http://www.sqi.gu.edu.au/spice/>

Fig. 3 – Desenvolvimento da avaliação do processo de software

O SPICE é uma iniciativa internacional de dar apoio ao desenvolvimento de um padrão internacional de avaliação do processo de software. Ele se tornou um *draft* em 1998 e padrão em 2003 – ISO/IEC TR 15504.

Os métodos introduzidos pelo SPICE geraram ganhos substanciais em qualidade e produtividade. Estes ganhos chamaram atenção para este modelo de avaliação. (ISO/IEC TR 15504, 2001)

O SPICE se concentra na análise de resultados gerados ao final da avaliação de processos, isto ajuda a identificar os pontos fortes e fracos numa empresa e os riscos existentes em seu processo. Também identifica causas significantes de qualidade baixa e limites extrapolados de tempo e custo. Com isto é possível determinar prioridades de maneira correta.

Pessoas são incluídas apenas no que diz respeito ao treinamento.

2.6 People Capability Maturity Model (P-CMM)

Após a publicação do CMM percebeu-se que a melhoria contínua pregada necessitava de mais alguns fatores até então não levados em consideração. Mudanças na maneira de como se gerencia, desenvolve-se e se retém pessoas numa organização passaram a ser discutidas, pois estavam fora do escopo dos demais modelos de maturidade. Esta foi a motivação que levou o SEI/CMU a desenvolver o P-CMM.

O P-CMM é um modelo que tenta guiar as empresas no que diz respeito às vantagens humanas na produção de software. Ele é um guia que fala sobre as vantagens de atrair, desenvolver, motivar, organizar e reter os talentos necessários para proporcionar o melhor desenvolvimento de software. Ele foi desenvolvido após a estruturação do CMM, por isso pode ser integrado a ele ou pode ser usado por si só.

Com a evolução proporcionada pelo CMM muitas empresas viram suas práticas e processos transformados e produzindo resultados significativos, mas mesmo assim, mudanças fundamentais implicariam em transformações no modo como encaravam o fator humano.

A partir disto, a força de trabalho passou a ser analisada como passível de melhorias, e o P-CMM foi desenvolvido para embutir princípios de maturidade no desenvolvimento e aperfeiçoamento desta força de trabalho. Ele procura aprimorar a capacidade da força de trabalho, da mesma maneira que o CMM procura melhorar a capacidade do processo de software numa empresa.

Uma das premissas do P-CMM é a de que haja continuidade (assim como outras práticas de CMM, o P-CMM está dividido em 5 níveis), pois senão, o processo de melhoria se acaba. Isto promove disciplina no ambiente de trabalho e o estabiliza. A base para novos projetos passa a ser melhor tendo as premissas básicas já atendidas e estando dentro de um ciclo de melhoria contínuo.

O P-CMM coloca o conhecimento como sendo “a matéria bruta para o desenvolvimento de software” (CURTIS, 1995, p. 7), e, portanto, ter pessoal capacitado para o desenvolvimento de atividades é a maior vantagem que uma empresa produtora de software pode ter.

Por este motivo, muitas empresas perceberam que as habilidades individuais e de equipes passaram a ser diferenciais competitivos.

Toda empresa deve ser competitiva sob dois aspectos: seus produtos e serviços devem ser bons o suficiente para conquistar o mercado, e também, deve ter pessoas talentosas para desenvolver e vender estes produtos.

O P-CMM, como os demais modelos de maturidade em estágios, tem também os 5 níveis: Inicial, Repetível, Definido, Gerenciado e Otimizando. Ele também apresenta categorias de processos que indicam quais atividades devem ser estabelecidas em cada um dos níveis de maturidade.

Por ser este um modelo que trata fatores relacionados ao ser humano, ele será apresentado com mais detalhe. A seguir serão apresentadas as *Categorias de Processo* assinaladas pelo P-CMM:

Desenvolvimento de Capacidades: o nível em que se dá início esta categoria é no Repetível, pois é aí que se identifica a necessidade imediata de treinamento das pessoas em cada unidade. Outras habilidades procuram ser trabalhadas nesta categoria, são elas: comunicação, conhecimento e habilidades para desenvolver o negócio da organização, competências de equipes e liderança.

Construção de Equipes e Cultura: esta categoria de processo se inicia no nível Repetível, pois é aí que se inicia a organização e a interação das pessoas com a organização. O foco desta categoria é proporcionar a comunicação interpessoal e formal das pessoas com a organização. Também se procura desenvolver uma cultura de participação pelo aumento do envolvimento das pessoas nas decisões que afetam

seu trabalho. A partir de então, o foco passa a ser a construção de equipes com níveis apropriados de autonomia.

Motivação e Gerenciamento de Desempenho: esta categoria tem como objetivos a motivação, o desempenho, o ambiente de trabalho propício para o desenvolvimento de cada atividade, os benefícios, o sistema de remuneração e sua administração, a carreira.

Moldando a Força de Trabalho: esta categoria visa moldar a força de trabalho às necessidades do negócio através do estabelecimento de práticas básicas de recrutamento, seleção e orientação.

Na figura 4 têm-se os níveis de maturidade relacionados a categorias de processos apresentadas pelo P-CMM. Para cada categoria de processo são relacionadas as atividades necessárias para obtenção do nível de maturidade.

Níveis de Maturidade	Categoria de Processos			
	Desenvolvimento de capacidades	Construção de equipes e cultura	Motivação e gerenciamento de desempenho	Moldando a força de trabalho
5 Otimização	- Treinamento - Desenvolvimento da competência pessoal	Inovações Contínuas de Desempenho		
4 Gerenciamento	Direcionamento	Construção de equipes	- Alinhamento do desempenho organizacional - Práticas baseadas em equipes	Gerenciamento da competência organizacional
3 Definição	- Desenvolvimento da competência - Análise de conhecimento e habilidades	Cultura participativa	- Práticas baseadas na competência - Desenvolvimento de carreira	Planejamento de trabalho
2 Repetição	Treinamento Comunicação	Comunicação	- Compensação - Gerenciamento de desempenho - Ambiente de trabalho	Prover pessoal
1 Inicial				

Fonte: Adaptada de CURTIS, 1995, p 29

Fig. 4 – Níveis de maturidade e categorias de processos assinaladas pelo P-CMM

2.7 PSP

Práticas individuais mostram-se eficazes e diretas no que diz respeito à qualidade de software. É o caso do PSP (Personal Software Process) que traz disciplina à prática individual de engenheiros de software. O PSP é um conjunto de métodos, formulários e documentos. “O foco do PSP é o gerenciamento dos defeitos de software produzidos individualmente” (HUMPHREY, 1995, p. 271). Idealizado por Watts Humphrey o PSP foi desenvolvido para guiar, planejar e desenvolver módulos de software ou pequenos programas. “O PSP é um modelo de maturidade assim como o CMM” (HUMPHREY, 1995, p. 9), e como tal é baseado no princípio da melhoria do processo. Enquanto o CMM é focado na melhoria da capacidade organizacional, o foco do PSP é o indivíduo engenheiro. “O PSP aplica ao desenvolvimento de software métodos de qualidade que têm sido provados em muitos outros campos.” (HUMPHREY, 1995, p. 81)

Os objetivos principais do PSP são: melhorar as estimativas, melhorar o planejamento e o acompanhamento de cronogramas, proteger o indivíduo contra o excesso de compromissos, criar um comprometimento pessoal para a qualidade, estabelecer o envolvimento do engenheiro na melhoria contínua do processo. (HUMPHREY, 1995) Com isto deve-se “produzir maior qualidade dos produtos de software o mais cedo possível no ciclo de desenvolvimento.” (HUMPHREY, 1995, p. 81)

O PSP provê um método de trabalho que ensina um processo de software aos engenheiros, e o ponto de partida é fazer com que o engenheiro se envolva em seu próprio processo. O PSP tem sido adaptado para atender áreas da engenharia de software, como a requisição de software, especificação de software e caso de testes. Ele é altamente flexível podendo ser utilizado na construção de módulos indo desde um projeto completo a um pequeno sistema. Seus princípios e conceitos podem ser aplicados e adaptados em qualquer estrutura organizacional.

O PSP veio como uma boa alternativa para expandir o conhecimento e a capacidade pessoal, usando os limites pessoais para alcançar objetivos com mais efetividade.

O PSP ajuda os engenheiros e seus gerentes a aprenderem a prática da quantificação do gerenciamento do processo. “Processos definidos são amplamente reconhecidos como necessários em grandes projetos, mas geralmente não são entendidos como forma de ajuda a pequenas equipes e indivíduos.” (HUMPHREY, 1995, p. 27). Eles aprendem o uso do processo definido e aprendem também a coletar dados para gerenciar, controlar e melhorar o trabalho. Portanto, ele introduz um aprendizado e uma prática para a melhoria do processo. Pequenos ciclos de desenvolvimento e os dados pessoais tornam fácil o entendimento e aumenta a experiência dos desenvolvedores. Os dados obtidos com o PSP melhoram o planejamento e o gerenciamento do cronograma de projetos de software, resultando em um produto de melhor qualidade, com redução de custos nos testes e diminuição do tempo de desenvolvimento. (HUMPHREY, 1995)

Assim como no CMM, no PSP existem diversos níveis com características próprias. Esta prática pessoal pode melhorar drasticamente a qualidade, a previsibilidade e o ciclo de tempo dos produtos desenvolvidos, pois os engenheiros “praticam métodos efetivos de qualidade pessoal”. (HUMPHREY, 1995, p. 81) Melhorias podem ser notadas na capacidade adquirida de expandir a mente para o que deve ser feito e não para o que se está acostumado a fazer; também a capacidade de fazer e gerenciar planos, bem como reduzir defeitos através de prevenção e acompanhamento de cada fase do processo de software. “O foco está em identificar e remover defeitos existentes.” (HUMPHREY, 1995, p. 301) A remoção de defeitos é favorecida pelo planejamento de ações, pois “antes de começar cada trabalho, o engenheiro deve planejar e garantir os dados nos passos do seu trabalho.” (HUMPHREY, 1995, p. 81)

Isto se dá através do conhecimento que é passado para cada indivíduo de uma equipe dentro do padrão do PSP. Dentro dos sete passos progressivos ensinados, o conhecimento do processo torna-se mais visível e consistente aos olhos do engenheiro de software. Através da mensuração de defeitos, codificação, planejamento, qualidade e processo cíclico pessoal pode-se ter um maior controle sobre como cada indivíduo afeta e/ou contribui para a qualidade do software.

A implantação do PSP é dividida em 7 níveis: o Ponto de Partida do Processo Pessoal compreende 2 níveis, o PSP 0 que trata o processo atual e as medições

básicas e o PSP 0.1 que trata os padrões de codificação, propostas de melhoria e estimativas. O Planejamento Pessoal compreende outros 2 níveis, o PSP 1 que trata as estimativas e os relatórios relacionados a testes, e o PSP 1.1 que trata o planejamento de tarefas e tempo. A qualidade pessoal tem 2 níveis, o PSP 2 que é responsável pela garantia de revisões de código e projeto, e o PSP 2.1 que trata dos modelos de projeto. Por fim o processo pessoal cíclico compreende o último nível, o PSP 3 que garante o processo cíclico. Todos os níveis devem ser obtidos de maneira incremental. Os níveis superiores adicionam características aos níveis já implantados. Isto minimiza o impacto na mudança no processo do engenheiro, na qual ele somente tem que adaptar novas técnicas às já existentes. Os 7 níveis são apresentados de maneira incremental na figura 5:

Fig. 5 – Níveis de maturidade do PSP

No *Ponto de Partida do Processo Pessoal*, doutrinas de registro são instituídas, como o registro do tempo gasto em cada etapa do ciclo do desenvolvimento e registro dos defeitos encontrados. Isto é conseguido através do uso de formulários adequados. O nível PSP 0.1 inclui o uso de um padrão de codificação, de medidas padronizadas e do formulário de proposta de melhoria do processo.

No *Planejamento Pessoal*, o foco é o planejamento. A idéia geral é obter a capacidade de estimar quanto tempo se levará para realizar uma tarefa baseada nas

medições feitas em tarefas semelhantes realizadas anteriormente, como é o caso dos testes (PSP 1). Neste nível aprende-se a assumir compromissos que podem realmente ser cumpridos. O nível PSP1.1 inclui o planejamento de tarefas e a elaboração de cronogramas.

Na *Qualidade Pessoal* é introduzido o conceito de ensinamento a respeito dos erros cometidos por cada desenvolvedor. Deve-se ter uma idéia precisa de quantos erros em média são cometidos, em cada fase do ciclo de desenvolvimento. O PSP mostra que a forma mais adequada para tratar erros é evitá-los desde a sua origem. Devem ser utilizados os dados sobre defeitos já coletados para criar uma lista de verificação a ser utilizada em revisões de projeto e de código – PSP 2. O nível PSP 2.1 inclui a criação de padrões de projeto, bem como métodos de análise e prevenção de defeitos.

O *Processo Pessoal Cíclico* é a última etapa do PSP. Neste nível (PSP 3), o PSP sai do desenvolvimento de pequenos programas para tratar do desenvolvimento de projetos maiores, embora ainda em nível pessoal. A idéia é dividir os grandes projetos em pequenos projetos. Neste caso, o desenvolvimento acontece em passos incrementais.

2.8 CMMI

O CMM, como prática comum na adoção de critérios que possibilitam a uma organização o amadurecimento de seus processos, teve por seus desdobramentos específicos (SA-CMM, SE-CMM, SW-CMM, P-CMM) um avanço em sua adoção e também um fator de complicação, pois cada modelo exige uma certificação específica.

A utilização destes diversos desdobramentos do CMM fez com que surgissem algumas inconsistências entre eles, no que diz respeito ao processo de avaliação e ao modo de implementação.

Organizações que implementaram mais de um CMM tiveram problemas com confusão de termos e conceitos e altos custos de treinamento e avaliação, já que, embora com inspiração comum, os modelos eram independentes. Assim, uma

empresa que implantasse o SW-CMM e o P-CMM, por exemplo, era obrigada a realizar processos de avaliação separados.

Outro fator importante foi a necessidade, em parte de natureza política, de compatibilizar o SW-CMM com a norma ISO 15504 (SPICE). Novamente, inconsistências de método e terminologia exigiram que algo fosse feito.

Estes fatores levaram o SEI a abandonar a abordagem de lançar o SW-CMM 2.0, estabelecendo em seu lugar um projeto de integração dos CMMs e compatibilização com a norma ISO. Daí o nome CMMI: Capability Maturity Model Integration.

O projeto CMMI é, portanto, um esforço colaborativo para propiciar um modelo de melhoria de processos. “O propósito do CMMI é proporcionar direção na melhoria dos processos das organizações e a habilidade que possuem de gerenciar o desenvolvimento, a aquisição e a continuidade dos produtos ou serviços.” (CMMI, 2002, p. 1)

O resultado do projeto CMMI é um conjunto de produtos, que cumprem os requisitos do projeto, ao passo que reduzem a redundância, complexidade e custos resultantes do uso separado de vários CMMs. (CMMI, 2002)

A maior contribuição do CMMI é o significativo aumento da flexibilidade na implantação de projetos de melhoria dos processos de software. O foco de qualidade do CMMI está na qualidade dos produtos, de serviços, assim como no desempenho dos processos. (CMMI, 2002)

A principal mudança do CMMI em relação ao SW-CMM é a possibilidade de utilização de duas diferentes abordagens para a melhoria de processos. Estas duas abordagens são conhecidas como o “modelo contínuo” e o “modelo em estágios”. O SW-CMM, como se mostrou na seção 2.3, é um modelo em estágios. Alguns dos modelos do CMM com foco diferente de software, bem como o SPICE, usam uma abordagem diferente, o chamado “modelo contínuo”. Neste caso, cada área de processo (PAs) – diferente do CMM que trata áreas chave de processo (KPAs) - possui características relativas em mais de um nível. Assim, uma PA que, no modelo em estágios, pertence exclusivamente ao nível 2, no modelo contínuo pode ter características que a coloquem em outros níveis.

A flexibilidade do CMMI permite as duas abordagens dando liberdade para que cada organização com realidades e necessidades distintas escolham a mais adequada. A figura 6 mostra o retrato destas duas abordagens. No modelo em estágios o foco é a melhoria organizacional, que é obtida com o cumprimento de cada PA pertencente ao nível de maturidade da organização. Já no modelo contínuo o foco é o processo que leva a empresa a atingir seus objetivos, independente a qual nível de maturidade este processo se referencie.

O modelo contínuo é mais flexível, embora mais complexo de administrar.

Fonte: Adaptada de <http://www.choose.com.br/consultoria/cmmi.asp>

Fig. 6 – Modelos de abordagens do CMMI

Uma das práticas gerais do CMMI é o treinamento de pessoas. Seu propósito é “garantir que as pessoas tenham as habilidades necessárias e a experiência para desenvolver e garantir o processo.” (CMMI, 2002, p. 42) Mais uma vez a garantia da realização dos processos de maneira adequada é de responsabilidade de profissionais, os quais devem seguir um treinamento adequado.

2.9 Estudo Comparativo

Como forma de avaliação dos modelos até aqui apresentados com o foco nos fatores humanos, tem-se o quadro comparativo que é mostrado na figura 7. O conjunto de itens selecionados para compor este quadro foi escolhido com base no estudo das categorias de processo do P-CMM e no estudo da relevância de cada fator passível de influência no desenvolvimento de software sob o foco humano.

	ISO 9000-3	SW-CMM	SE-CMM	P-CMM	SPICE	PSP	CMMI
Regras para treinamento	√	√	√	√	√	√	√
Recomendações individuais						√	
Motivação				√			
Recompensa				√			
Retenção de talentos				√			

Fig. 7 – Quadro Comparativo dos modelos de qualidade

As regras para treinamento são a indicação que cada modelo faz para que esta prática seja estabelecida e dentro de que padrão ela deve acontecer; como o treinamento deve ser realizado e quais pessoas devem ser envolvidas.

Recomendações individuais são regras para cada indivíduo como prática habitual dentro do processo de desenvolvimento de software. São práticas que devem ser seguidas independentemente das práticas de equipe. Na verdade são metas a serem atingidas por cada membro de uma equipe.

A motivação é um fator difícil de ser dimensionado, portanto na análise deste item se procurou verificar como cada modelo tratou o assunto. Para isto, foi verificado se o foco do modelo neste assunto é direto ou indireto, se o trata como fator de influência a qualidade, se pondera outros fatores que possam aumentar ou diminuir a motivação de uma equipe.

Também foi analisada a preocupação em estabelecer e gerenciar uma política adequada de recompensas. De maneira direta, ou indireta (como fator de influência sobre a motivação) este item foi levado em consideração.

A preocupação em reter pessoas mostra como cada modelo trata o fato de que pessoas são importantes no processo de desenvolvimento de software.

Como forma de analisar cuidadosamente como o ser humano, que é falível, passível de erro, influencia a qualidade de software, mostra-se necessário o estudo do comportamento e do ser humano, assunto do próximo capítulo.

3 FATORES CONDICIONANTES NO AMBIENTE PROFISSIONAL

Neste capítulo será feito um estudo da relevância de cada fator passível de influência no desenvolvimento de software sob o foco humano. Isto exigiu detalhamento e estudos sobre a teoria do erro humano, já que alguns erros atribuídos ao ser humano possuem evidências no processo de desenvolvimento do software.

O objetivo deste capítulo é extrair fatores condicionantes do desempenho humano na produção de software. Para tanto, o capítulo enfatiza fatores individuais e organizacionais.

3.1 Fatores Individuais e Organizacionais

Vários fatores individuais e organizacionais relacionados aos seres humanos são influenciadores de suas atividades rotineiras, sejam elas relacionadas a software ou não. Estes fatores são geralmente estudados por áreas de concentração que não a Engenharia. Mas quando o assunto é a interferência destes fatores na produção intelectual do indivíduo, o tema torna-se relevante para o estudo da Qualidade de Software.

Como forma de identificar os fatores condicionantes presentes no dia-a-dia de trabalho de profissionais da área de software, alguns fatores foram objeto de estudo e designados como relevantes para no processo de desenvolvimento de software; eles são os apresentados na figura 8 e na sequência serão discutidos. Os fatores enumerados nesta figura foram escolhidos com base no trabalho de Reason (1990) e aperfeiçoados com os conhecimentos adquiridos nas áreas de conhecimentos da Psicologia e da Sociologia.

Fatores Organizacionais
Comunicação
Relacionamento
Controle sobre os processos
Peopleware
Políticas e Sistemas de RH
Papéis e Responsabilidades
Reconhecimento
Fatores Individuais
Dificuldade no desempenho de tarefas
Problemas pessoais (financeiros, profissionais, saúde)
Carga de trabalho
Inteligência Emocional
(aspectos cognitivos do comportamento)
Motivação
Liderança
Estado Emocional

Fig. 8 – Fatores Individuais e Organizacionais que afetam o desempenho

3.1.1 Peopleware

Segundo DeMarco (1990) *peopleware* é um conceito que engloba as pessoas de uma empresa e os diversos elementos da estrutura organizacional ligados a elas: políticas e sistemas de recursos humanos (remuneração, premiação, carreira), papéis e responsabilidades e lógica dos objetivos da empresa. O fator de maior relevância deste conceito para este trabalho é o fato do *peopleware* ser um conjunto de fatores individuais (vide figura 8).

No caso de software vale destacar que o conceito de *peopleware* abrange dois tipos de pessoas envolvidas: os **desenvolvedores**, profissionais que trabalham numa organização e os **usuários** que são os destinatários do produto. Em ambos os casos, a forma como o *peopleware* afeta o desenvolvimento de software é uma preocupação do engenheiro de software. Quando o *peopleware* se refere ao usuário, esta preocupação está no domínio da Engenharia de Usabilidade (Nielsen, 1993) e não é escopo deste trabalho.

Mas quando o *peopleware* se refere aos desenvolvedores, este é um problema que afeta os processos visando qualidade de software.

Peopleware é um conceito multidimensional e pode ser definido da seguinte forma: em seu núcleo estão os modelos mentais, valores e crenças coletivas da empresa e na extremidade estão os elementos estruturais de uma organização, conforme mostra a figura 9:

Adaptado de DeMarco, 1990

Fig. 9 – Estrutura do Peopleware

Com a visão deste modelo percebe-se que fatores intrínsecos e também os incorporados pelo desenvolvedor de software, afetam a sua capacidade de produtividade. Mais que isto, o conceito de *peopleware* atrela-se a aspectos comportamentais subjetivos.

Mesmo com todo conhecimento e capacidade técnica que um desenvolvedor possa ter, fatores como má remuneração, mau gerenciamento e delegação de tarefas, ambiente de trabalho ruim, e maus relacionamentos interpessoais interferem diretamente na sua concentração e criatividade, e, conseqüentemente, no produto que está desenvolvendo.

Processos contínuos que acontecem no mundo interno e que permitem pensar e sentir o mundo, determinar estados comportamentais das mais diferentes formas e as adaptar à realidade constituem a chamada subjetividade. “A subjetividade é portanto, o mundo construído internamente pelo sujeito, a partir de suas relações sociais, de suas vivências no mundo e da sua constituição biológica; é também fonte de suas manifestações afetivas e comportamentais” (BOCK, 1995, p. 23)

Os processos subjetivos são observáveis e podem ser comprovados segundo a própria Psicologia. “O comportamento é um objeto observável, mensurável, que pode ser reproduzido em diferentes condições e em diferentes sujeitos”. (BOCK,

1995, p. 42) Katz (1978) exemplifica alguns destes processos: a constância nas chegadas e partidas dos empregados de uma empresa, os horários de pausas para descanso ou para o café e a persistência dos ciclos de comportamento são passíveis de observação.

Assim, pessoas não são módulos e não podem ser tratadas igualmente. Este é um dos maiores erros cometidos nas empresas. Os perfis devem ser levados em consideração, as diferentes aspirações e capacidades também. As pessoas são diferentes pela presença das variáveis inatas e, pelas variáveis adquiridas pelas diversas experiências passadas ao longo da vida.

Para que estas metas sejam atingidas é preciso ter consciência de que a produção de um software não pode ser equiparada a uma linha de produção, pois seu desenvolvimento não é algo mecânico, mas sim um processo criativo que exige certa liberdade e tempo.

O ser humano é movido pela emoção⁴ e pelo reconhecimento. (FUGIWARA, São Paulo, Correspondência pessoal, 2001). “As emoções referem-se aos sentimentos e humores e à maneira pela qual estes são expressos tanto no comportamento quanto nas respostas fisiológicas”. (KENDEL, 1997, p. 475) Algumas emoções estão tão fortemente atreladas a algumas características profissionais essenciais que passaram a ser premissas ao desenvolvedor de software, por isso, devem ser observadas, analisadas e, se possível, aperfeiçoadas pelas empresas.

As emoções de um desenvolvedor são aparentes em seu trabalho em situações como a da depressão, por exemplo, que envia a pessoa ao estado mais profundo do isolamento. A depressão pode facilmente ser percebida, pois o indivíduo não consegue concentração necessária ao trabalho e nem consegue se relacionar para o desenvolvimento do “trabalho em equipe”. A frustração é uma das maiores conseqüências para a vida profissional de um indivíduo nesta situação. Conseqüentemente ficam comprometidos prazos, custos e principalmente a qualidade. Isto tudo passa a ser atribuído pela empresa ao mau profissionalismo ou a

³ Emoções são reações fisiológicas e psicológicas que influem na percepção, aprendizagem e no desempenho. Elas exercem efeitos motivadores sobre o comportamento humano. FUGIWARA, P. (Psicóloga, São Paulo). Correspondência pessoal, 2001

falta de interesse do desenvolvedor, quando na verdade deveria ter consciência de que abalos emocionais como este são comuns e podem ser resolvidos.

Manter pessoas competentes é extremamente difícil para as organizações de software, pois vários fatores são determinantes para atrair e reter este tipo de talento. Segundo Bartol (1983) a combinação de atrativos e condições de trabalho, entre outros, são fatores determinantes para as taxas de rotatividade deste tipo de profissional.

A rotatividade está relacionada com os sentimentos que o funcionário consegue atrelar à empresa na qual trabalha. Estes sentimentos são provocados e cultivados por vários aspectos do seu ambiente de trabalho como supervisão, remuneração e o próprio trabalho em si.

Existe uma forte ligação entre o comprometimento organizacional, que significa o grau de identificação individual, o envolvimento na empresa e a rotatividade. Também o profissionalismo está intrinsecamente ligado a esta rotatividade, pois aquele indivíduo que tem seu comportamento profissional levado em consideração pela empresa em que trabalha, vê-se mais disposto a permanecer, e se sente mais comprometido. Mas para isto é preciso que a empresa possibilite que o indivíduo se desenvolva como profissional, ou seja, permita a seu funcionário desenvolver características como autonomia profissional, comprometimento com a profissão, ética, crença em padrões, e outras que são conseguidas com o apoio da empresa na forma de apoio, liberdade e incentivo.

Bartol (1983) propõe um modelo baseado em 4 variáveis que devem ser analisadas como causas da rotatividade de profissionais. São elas: satisfação com o emprego, comprometimento organizacional, profissionalismo e critérios de recompensa organizacional. Na figura 10 este modelo é apresentado com a relação existente entre cada uma das variáveis, como cada uma delas pode aumentar a incidência de outra e como todas contribuem para a diminuição da rotatividade.

Fonte: Adaptado de Bartol, 1983, p. 808

Fig.10 – Modelo de rotatividade de Bartol

Benefícios e bônus são extremamente relevantes no momento da decisão sobre um novo emprego, o que mostra que as empresas devem se preocupar com eles mesmo que por interesse de atrair talentos e não por uma política de reconhecimento do funcionário como fator chave para o sucesso.

3.1.2 Inteligência Emocional

A Inteligência Emocional é o conjunto de aptidões básicas necessárias para lidar adequadamente com as diferentes situações da existência e com relacionamentos inter-pessoais e grupais familiares, sociais e no trabalho em todo o seu espectro (GOLEMAN, 1995).

No âmbito do desenvolvimento de software, algumas das aptidões básicas da Inteligência Emocional são visíveis e merecem atenção, são elas:

- **Motivação:** capacidade de automotivar-se para a ação na busca de realização de objetivos.
- **Criatividade:** capacidade de encontrar alternativas para as diferentes situações.
- **Liderança:** capacidade de conduzir pessoas na busca de objetivos comuns.
- **Estado Emocional:** conjunto de emoções humanas.

Mesmo estas aptidões sendo perceptíveis no desenvolvedor de software em seu ambiente de trabalho, é preciso ter a consciência de que a produção e o estímulo

destas aptidões têm origem fisiológica e são relevantes no processo de desenvolvimento de software se visto sob o ponto de vista de produtividade e confiabilidade daquilo que é fruto do processo estabelecido.

Fato é que os estímulos provenientes do Sistema Límbico (emoções) podem causar um ruído que atrapalha o funcionamento intelectual em diferentes graus de acordo com fatores e condicionamentos adquiridos; como se existissem dois cérebros: o racional e o emocional. O funcionamento mental adequado depende de uma harmonia entre ambos. Portanto, o objetivo deve ser o equilíbrio entre razão e emoção e não a supressão da emoção. (KENDEL, 1997)

O Estado Emocional de um desenvolvedor de software também precisa de atenção. Nele tem impacto direto fatores como dificuldades encontradas pelos desenvolvedores na realização de suas tarefas diárias, além de problemas pessoais, profissionais, financeiros e de saúde vividos por cada indivíduo.

3.1.2.1 Liderança

O gerenciamento de pessoas é um dos fatores de maior consideração para se obter qualidade, mas é também o mais negligenciado. O gerente, ou líder é aquela pessoa que deve tornar possível o trabalho do indivíduo ou da equipe. É ele que deve se assegurar de que nada externo interfira no rendimento pessoal de cada desenvolvedor. Isto deve ser conseguido através de empatia e coragem e não através de imposições e pressões. “A liderança é sempre uma função combinada dos fatores sociais estruturais das características particulares dos indivíduos que formam a estrutura” (KATZ, 1978, p. 350). O termo liderança ainda é muito relacionado ao cargo ocupado na hierarquia organizacional e mesmo quando uma pessoa exerce influência sobre outra, não se diz que há liderança.

Uma relação difícil entre gerentes de várias áreas que precisam trabalhar em conjunto também pode gerar problemas para o projeto, e em certas ocasiões, exigir que exista um coordenador para gerenciar os chefes destas áreas para que o projeto flua como desejado. A influência que um gerente deve exercer sobre sua equipe pode se dar através de sugestões, solicitações, ordens ou comandos. De qualquer maneira

“a influência se dá quando uma pessoa age de modo a modificar o comportamento de uma outra” (KATZ, 1978, p. 252). Mas o poder exercido nos níveis hierárquicos superiores é que determinam a influência e o controle que se tem sobre uma equipe, e é este controle que “envolve a distinção entre tentativas de influência que obtêm êxito e as que não tem sucesso” (KATZ, 1978, p. 253).

O maior desafio torna-se, portanto, o conhecimento por parte do gerente, do material humano que tem nas mãos. Segundo Bergamini (1986) o conhecimento que o gerente adquire do seu pessoal torna-o capaz de :

1. “Ajudar cada um a conhecer-se e valorizar-se mais adequadamente, utilizando-se de seus pontos fortes e minorando suas deficiências;
2. Orientar convenientemente a mão-de-obra em função de aptidões, capacidades e interesses;
3. Melhorar os níveis de supervisão;
4. Levantar novas necessidades de treinamento ou aprimorá-lo quando insuficiente;
5. Readaptar profissionalmente;
6. Aproveitar e melhorar o potencial humano na empresa;
7. Informar a administração superior de fatores importantes;
8. Planejar o futuro conforme os fatos levados ao conhecimento da administração.” (BERGAMINI, 1986, p. 51)

Mesmo organizações que atingiram certo nível de maturidade em seus processos de desenvolvimento de software necessitam de líderes, pois existem também condições no ambiente que podem interferir no que diz respeito à liderança, pois mudanças tecnológicas, legais, culturais podem atingir um estado sistêmico e levar o que anteriormente era muito favorável a uma situação completamente inviável. (KATZ, 1978)

3.1.2.2 Motivação

“Motivação é o fenômeno humano responsável pelo dinamismo do indivíduo em situações de trabalho e nele determina um movimento no sentido de evoluir do

menor para o maior grau, do amadurecer pessoalmente e estar sempre exibindo um comportamento de busca” (BERGAMINI, 1986, p. 71). Ela implica num objetivo a ser alcançado, e “está sempre presente como desencadeadora da ação, quer seja por necessidades afetivas ou intelectuais, quer seja por necessidades fisiológicas” (BOCK, 1995, p. 81). Ela depende da individualidade de cada um, pois cada um tem seu estilo, sua filosofia de vida, sua maneira de agir e isto, é o que direciona seus objetivos.

Sob o ponto de vista da Engenharia de Software os aspectos cognitivos de percepções e ações são fundamentais em desenvolvedores de software e estão diretamente relacionadas com a motivação pessoal de cada um.

“Os estados de motivação, como os processos cognitivos, são estados internos inferidos que são postulados para explicar a intensidade e a direção de uma variedade de comportamentos.” (KENDEL, 1997, p. 490)

A motivação é algo que deve ser trabalhado continuamente dia após dia, pois a partir dela sobressaem-se as melhores idéias e os melhores profissionais. Com o passar do tempo “um profissional adquire competência, não necessariamente adquire motivação” (HUMPHREY, 1995, p. 65). A competência é técnica, já a motivação é intrínseca ao ser humano.

A motivação está ligada a incentivos, mas é mais do que isto. Além de recompensas, faz parte da motivação a busca pela aceitação do grupo no qual o indivíduo trabalha e quer se estabelecer, e a busca pelo reconhecimento da capacidade profissional.

Existem meios através dos quais uma empresa pode tentar buscar a motivação de seus funcionários, um destes meios é a adoção do marketing de incentivos.

A adoção de um marketing de incentivo, que é a política adotada para determinar premiações para certos projetos numa empresa, não pode ser feito isoladamente. Ele deve vir acompanhado de reformas na estrutura organizacional, pois o contrário não produzirá o efeito desejado. (DeMARCO, 1990)

Pessoas são movidas por desafios, e eles são os instrumentos através dos quais elas se juntam. O ponto chave, entretanto, é que muitos dos fatores motivacionais ao desenvolvedor são diretamente controlados pelos líderes. (HUMPHREY, 1995)

A motivação através de um direcionamento de ações em desenvolvedores de software pode ser obtida através de práticas individuais como o PSP. (HUMPHREY, 1995)

Fato é que a “motivação é ágil , mas ela depende da pessoa, da tarefa, do ambiente”. (HUMPHREY, 1995, p. 65)

3.1.3 Comunicação

“A comunicação é um dos fatores mais relevantes nos relacionamentos de trabalho, ela é um processo social da mais ampla relevância no funcionamento de qualquer grupo, organização de sociedade” (KATZ, 1978, p. 257). O profissional de software passa cerca de 50% do seu dia se comunicando com pessoas (LAGO, 2000), por isso falhas neste processo podem trazer perdas significativas ao processo de desenvolvimento de software. A comunicação pode atrapalhar ou ajudar, direta ou indiretamente qualquer problema existente, pois “as organizações humanas são sistemas de informação” (KATZ, 1978, p. 256)

A comunicação interpessoal tem três fatores essenciais: pessoas, linguagem e realidade. Ela é um processo que envolve um emissor e um receptor, e isto gera perda de informações. O que deve se fazer é tentar minimizar estas perdas. “Muitos de nossos problemas individuais e sociais são o resultado de comunicação inadequada e falha.” (KATZ, 1978, p. 257)

A comunicação entre áreas de desenvolvimento em uma empresa pode ser um foco crucial. Esta comunicação pode gerar ruídos, que podem ser traduzidos como erros de definição, especificação, entre outros. A comunicação deve ser a mais abrangente possível para permitir a decodificação correta. O problema maior para se gerar esta comunicação limpa é que o ser humano, apesar de agente neste processo, é passível de interpretações diferentes, por isso podem ocorrer ambigüidades nas definições. A comunicação entre os membros de uma equipe desenvolvedora de software é espelho da comunicação que ela tem com o gerente da área. Por isso uma “comunicação aberta entre gerentes e equipe pode ajudar a melhorar a comunicação entre os membros desta equipe.” (HUMPHREY, 1995, p. 176)

Uma das conseqüências da má comunicação é a sobrecarga de informações que se dá quando a entrada de informações geradas pela comunicação é maior do que algumas partes da organização podem absorver. E ela tem um fator agravante: além

das informações passadas normalmente pela comunicação e influência acerca de um projeto, muitas informações chegam via chamada telefônica, por exemplo, o que acarreta no mínimo, constantes interrupções de raciocínio. Estas interferências podem ser causa de baixa na produtividade. Outros tipos de interrupções constantes podem levar à queda na produtividade e também à desatenção que pode levar ao erro.

A incapacidade de absorver uma grande quantidade de informações associadas aos prazos impostos, muitas vezes irreais, faz com que o indivíduo deixe de realizar tarefas. Como forma de controle mais efetivo e com metas na produtividade, a formalização de requisitos e aprovações devem ser documentais, ou seja, deve haver um documento correspondente aos fatos para que o processo não se perca com as ambigüidades e dependências individuais.

A pressão é outro fator que pode gerar problemas na comunicação e, por consequência, pode-se chegar ao erro.

A boa comunicação entre os membros de uma equipe é o fator chave para que o grupo trabalhe de forma integrada.

4 ERRO HUMANO

Este capítulo discorrerá sobre o assunto do Erro Humano. O intuito é apresentar os conceitos, a natureza e apresentar as classificações dos erros conforme a literatura.

O Erro Humano pode ser causado por diversos fatores que foram tratados no capítulo anterior, portanto, este capítulo tem como finalidade apresentar a fundamentação da teoria do Erro Humano para que a correlação entre causas e conseqüências possa ser estabelecida. Além disto, a identificação de erros provenientes e dependentes dos seres humanos são importantes no que tange a confiabilidade do ser humano no processo de desenvolvimento de software.

Para ajudar nesta análise, torna-se de fundamental importância, estudar o que é o erro humano. As classificações apresentadas são genéricas em relação ao ser humano, ao erro cometido em qualquer que seja a área que tenha o envolvimento do ser humano. Sendo a área de desenvolvimento de software uma destas áreas, pode-se abstrair estes conceitos e assim se traçar um paralelo para os erros do desenvolvedor de software.

4.1 Erro Humano

Há séculos filósofos e psicólogos centram suas pesquisas na capacidade do cognitivo humano ser passível de erro.

As situações em que os erros se manifestam são semelhantes no âmbito de atividades mentais, sendo assim, torna-se possível identificar formas comparáveis de erros, como: linguagem, percepção, recordação, reconhecimento, julgamento, solução de problemas, tomada de decisão e formação de conceitos.

Segundo Reason (1990) os erros podem ser classificados de acordo com sua forma de ocorrência em variáveis (constantemente mudando, não existindo um

parâmetro para ampará-lo) e os constantes (são facilmente comparáveis a erros já cometidos por um mesmo indivíduo).

Os erros tratados como constantes são erros pré-julgados como previsíveis, e estes tipos de erro são dados através de probabilidade. A maioria das falhas humanas previsíveis pode ser mapeada nas propriedades adaptativas da cognição humana. Já os erros variáveis não possuem uma forma de ocorrência definida, o que acaba fazendo com que esta categoria seja uma espécie de guarda-chuva a todos os outros erros. (REASON, 1990).

O erro humano pode ser atrelado a operações e situações mal planejadas, a má distribuição do papel desempenhado por quem opera o sistema, má definição da interface utilizada, deficiências organizacionais e gerenciais, e nem sempre aos fatores tecnológicos.

Cada tarefa desempenhada por um ser humano dentro de um ambiente computacional tem suas premissas e regras estabelecidas, mas cada um dos passos pode proporcionar vários caminhos alternativos, e conseqüentemente levar ao erro. Os erros ocorrem se os elementos envolvidos no processo estão incorretos, incompletos ou ausentes. (SMIDTS, 1995)

4.2 Natureza do Erro Humano

Muitas teorias a respeito da identificação e previsibilidade do erro humano levam em consideração quatro grandes elementos na produção de um erro: a natureza da tarefa, as circunstâncias do ambiente, o mecanismo que gerencia o desempenho e a natureza do indivíduo. (REASON, 1990) Mas, para muitos autores, o erro previsível não existe, pois se isto fosse possível conseguiríamos nos precaver com inúmeros passos para cercá-los e evitá-los. Prevenir erros, especialmente os humanos, depende de uma variedade de fatores que dão sentido à sua ocorrência, o que os torna inteligíveis.

Sendo a informação o fundamento no qual os desenvolvedores de software se baseiam e, a partir do qual é feita a abstração do sistema, a confiabilidade e a

legibilidade destas informações podem ser a diferença entre um software com mais ou menos defeitos embutidos.

A informação pode levar a tomadas de decisão e, conseqüentemente, a efetivação de ações sobre determinadas tarefas, como mostra a figura 11. Na figura em questão há a evidência de erro por uma ação não intencional, o que é chamado de lapso ou deslize, ou ainda por uma ação intencional, mas com engano de execução. Por este motivo, a fonte da informação tem papel fundamental na veracidade da informação disponível, pois é utilizada como base para o fluxo de decisões e ações no desenvolvimento de software. (SMIDTS, 1995)

Fonte: Adaptado de REASON, 1990, p 6

Fig. 11 - Algoritmo para distinguir a variedade de comportamento intencional

4.3 Classificação do Erro

Segundo Reason (1990), é possível classificar um erro humano, para isto, é preciso levar em consideração a variedade do comportamento intencional que o ser humano tem perante as situações.

Um erro tem sempre embutido em si a intenção da realização de uma tarefa. Esta intenção pode ser mal definida ou conter distorções da tarefa a ser realizada, por isso o erro muitas vezes é cometido, sem a consciência do que se está gerando.

Os tipos de erros, segundo Reason (1990), têm sua origem nos estágios cognitivos de efetivação de uma ação: planejamento, armazenamento e execução, como apresentado na figura 12.

Estágio Cognitivo	Tipo de Erro Primário
Planejamento	Enganos
Armazenamento	Lapsos
Execução	Deslizes

Fonte: Adaptada de REASON, 1990, p 13

Fig. 12 - Classificação dos tipos de erro de acordo com os estágios cognitivos em que elas ocorrem.

Os enganos são difíceis de serem detectados, pois “enganos podem ser definidos como uma deficiência ou falha no julgamento e/ou envolvem o processo de dedução da seleção de um objetivo ou a especificação do significado de sua conclusão” (REASON, 1990, p. 8). Os enganos podem ocorrer por dois motivos: falha ou falta de conhecimento. No caso de desenvolvimento de software, podemos considerar os enganos como defeitos de software, ou seja, defeitos que são latentes, provenientes de erros de implementação, lógica, etc.

Já os deslizes são aqueles erros decorrentes de uma ação não planejada, e os lapsos concentram a maioria das formas de erro. No desenvolvimento de software, podemos considerar os lapsos e deslizes como sendo os defeitos de software, que são detectados e retirados.

Deslizes e lapsos envolvem falta de atenção e omissão ao realizar o acompanhamento da maneira devida. Eles estão associados com falhas normais no que diz respeito à execução e intenção de ação, enquanto que os enganos ocorrem no nível de formação da intenção.

Reason (1990) enumera os erros em três dimensões: erros baseados em habilidade (SB), erros baseados em regras (RB) e erros baseados em conhecimento (KB).

Erros baseados em habilidade e baseados em regras podem gerar defeitos, que são latentes no caso de desenvolvimento de software. Estes defeitos podem ser provenientes da aplicação da regra errada, ou incompletas.

Já os erros baseados em conhecimento ocorrem pelo grau de envolvimento do indivíduo na solução do problema.

Um exemplo de como se caracterizam estas três dimensões de erro está representada na figura 13. As atividades humanas relatadas na figura 13 mostram-se generalizadas, mas são perfeitamente cabíveis ao desenvolvimento de software. Desde as tarefas mais rotineiras, passando pela definição de modelos e regras a serem seguidos até a decisão por modelos abstratos capazes de solucionar o problema em questão, todas estas são atividades presentes no ciclo de vida de um software. Desta maneira, o erro humano também se mostra presente em cada uma das etapas mencionadas.

Fonte: Adaptada de REASON, 1990, p 64

Fig. 13 – Esquema da dinâmica do Erro nas três dimensões definidas por Reason

Com esta concepção de dimensões de erros, tem-se que os erros baseados em (falta de) regras e (in) habilidades são mais visíveis e presentes que os erros baseados em (falta de) conhecimentos. Por isso é preciso saber mais sobre a natureza da tarefa a ser realizada para que assim sejam previstas as regras individuais necessárias para que se evite o erro. (REASON, 1990)

Estes erros podem acontecer em qualquer atividade humana e são muito evidentes na área de desenvolvimento de software. Falta de especificações, falta de especialização de profissionais nas tecnologias exigidas pelo mercado, falta de conhecimento do tipo de sistema desenvolvido, tudo isto são evidências do dia-a-dia de empresas produtoras de software.

A maioria dos erros observados no nível SB está atrelada ao fator atenção. Grande parte da limitação dos recursos da atenção se dá pelo fato de que algumas preocupações internas ou distrações externas nos momentos críticos de investigação bastam para levar a ação para um caminho totalmente diferente. Muitos dos erros que envolvem problemas com atenção são, na verdade, gerados por omissões que estão fortemente relacionadas com interrupção.

A maioria dos erros observados no nível RB gira em torno do tratamento de exceções, sobrecarga de informações e redundância. Já os erros no nível KB giram em torno dos fatores pertinentes ao conhecimento e capacidade do indivíduo.

A seguir, na figura 14, estão representados alguns dos fatores pertinentes a cada nível de erro humano segundo Reason (1990)

Desempenho Baseado em Habilidade (SB)	
Desatenção	Excesso de Atenção
Capturadores duplos de lapsos	Omissões
Omissões seguidas de interrupção	Repetições
Redução de intencionalidade	Reversões
Confusões de percepção	
Erros de interface	
Desempenho Baseado em Regras (RB)	
Falta da aplicação de regras boas	Aplicação de regras ruins
Primeiras exceções	Deficiência de codificação
Sinais opostos e falta de sinais	Ações deficientes
Sobrecarga de informações	Regras contra indicadas
Regra de força	Regras erradas
Regras gerais	Regras irrelevantes
Redundância	
Rigidez	
Desempenho Baseado em Conhecimento (KB)	
Seletividade Limitações da área de trabalho Fora da vista, fora da mente Confirmação de ponto de vista Superconfiança Revisão preconceituosa Correlação ilusória Problemas com casualidade Problemas com complexidade Problemas com atraso no feed-back Considerações insuficientes do processo Dificuldades com o desenvolvimento exponencial Pensamento em série e não em rede Temática errante	

Fonte: Adaptada de REASON, 1990, p 69

Fig. 14 – Resumo dos principais tópicos dos modos de falha para cada um dos três níveis de desempenho

O desenvolvedor tende a simplificar as causas dos erros, porque ele é seguido com as recorrências do passado próximo refletido por suas ações e reações, e é inclinado a subestimar as irregularidades do futuro. Em sua maioria, as pessoas não são boas em fazer diagnósticos sobre tarefas. Erros são inevitáveis e geralmente é pouco aceitável que o ser humano arque com as conseqüências por sua inabilidade em lidar com tarefas de informações difíceis.

O primeiro passo para combater o erro humano nas tecnologias de alto risco é considerar o que é conhecido sobre o significado dos deslizes e lapsos e enganos. O sucesso é um conjunto do pensamento de cada indivíduo, juntamente com as indicações e repostas do meio no qual ele está inserido.

4.4 Erro Humano e Defeitos no Software

A probabilidade que um sistema tem de desempenhar sua missão sob condições definidas para um período de tempo deve ser levado em consideração, já que assim se tem a caracterização da eficiência de um sistema de software . A robustez e a tolerância a falha de um sistema também devem ser analisados e projetados de maneira a garantir que seus objetivos sejam atingidos. “O software ainda é uma atividade humana, conseqüentemente, falhas no processo que se traduzem em defeitos no software são resultado de erros humanos.” (SMIDTS, 1998, p. 270)

Segundo Smidts et al (1998), os erros genéricos de um analista são: omissão de requisitos, erros cognitivos como má interpretação de requisitos e falha ao notificar a existência de conflito entre dois ou mais requisitos, bem como os erros topológicos, ou seja, troca na posição dos requisitos. Em todos os erros enumerados por Smidts o conhecimento pode ser a diferença entre condições aceitáveis e inaceitáveis, pois a detecção do erro pode ser determinante para a tomada de decisões.

Técnicas como a de prognóstico da taxa de erro humano, podem prever e avaliar a probabilidade da ocorrência dos erros humanos que são causados

isoladamente ou em conjunto com erros de máquina e sistema. Este é um grande desafio que pode melhorar a interação Homem-Máquina e evitar a degradação deste ambiente.

Segundo Smidts et al (1998) as condições importantes para que as atividades humanas sejam consideradas confiáveis são a habilidade desenvolvida e a capacidade em se lidar com situações de pressão. Estes fatores podem, se controlados, ajudar a minimizar os enganos e lapsos humanos, relacionados a não identificação e correção dos defeitos de software.

Muitas barreiras existem para que os erros humanos sejam analisados e quantificados, mas sabe-se que eles estão (e estarão) presentes em todas as fases de desenvolvimento de software. Fato é que os “erros foram (ou serão) feitos em todos os estágios do ciclo de vida” de um software (SMIDTS, 1998, p. 271).

Os erros mais comumente atribuídos aos desenvolvedores de software são originados durante o ciclo de desenvolvimento de software e podem ser divididos em duas categorias: “erros de desenvolvimento feitos durante análise de requisitos, projeto e codificação; e erros de depuração feitos durante tentativa de remoção de defeitos identificados durante a inspeção e testes dinâmicos de software.” (STUTZKE, 2001, p. 184)

A figura 15 ilustra o processo de erro e remoção de falhas por conta do desenvolvedor.

Fonte: Adaptado de STUTZKE, 2001, p 186

Fig. 15 – Introdução de erros de desenvolvimento e depuração de software

Quando um erro é identificado e um desenvolvedor se engaja na solução deste erro, três coisas podem acontecer: o erro pode ser removido com sucesso, o erro pode não ser removido e/ou novas falhas podem ser introduzidas. De qualquer maneira, sabe-se que “um único erro pode injetar múltiplos defeitos no software” (STUTZKE, 2001, p. 185).

Um experimento de Nakamura (1992) realizado durante um mês com um grupo de desenvolvedores mostrou um total de 120 erros sendo que os fatores humanos são a maioria dentre as causas de erros detectados. A figura 16 mostra o percentual dos fatores humanos nas causas de erros de software. Segundo Vanderhaegen (2001) para se determinar o Erro Humano é preciso uma análise de diferentes valores estimados por julgamentos subjetivos de especialistas assim como o controle destas informações em bancos de dados. O agravante é que estes raramente existem nas empresas, quer seja por falta de conhecimento, ou a existência de dados incompletos, ou ainda a falta de hábito de armazenar estas informações, por isso os números atribuídos a erros humanos no desenvolvimento e manutenção de software podem ser ainda maiores.

Fonte: Adaptado de NAKAMURA, 1992, p 360

Fig. 16 – Fatores Humanos nas causas de erros de software

O processo de desenvolvimento de software vem sendo acelerado cada vez mais com a busca incessante pela redução de custos, e nesta engrenagem a susceptibilidade do ser humano perante as inúmeras condições já citadas até aqui, é cada vez maior. Por isso, é cada vez mais “importante aumentar os mecanismos de detecção dos erros humanos no software.” (NAKAMURA, 1992, p. 361)

5 METODOLOGIA DE ANÁLISE

Através dos estudos apresentados a respeito do Erro Humano e dos fatores identificados no capítulo 3, a pesquisa de campo realizada teve por objetivo formar um mecanismo de identificação dos fatores relacionados como condicionantes de desempenho.

A pesquisa de campo procurou investigar individualmente como cada desenvolvedor se envolve no processo de identificação e recuperação de defeitos de software, bem como sua atuação sobre estes defeitos, e sua própria análise de causas destes defeitos. Para isto, a pesquisa foi realizada em duas etapas.

A primeira etapa foi realizada nas diretorias responsáveis pelas áreas de Automação Bancária e Internet da Itautec Philco S/A no período de outubro a dezembro de 2002 e não foi baseada num grupo específico de controle. A área de Automação Bancária é responsável pela produção de hardware e software para bancos, seguradoras e financeiras, enquanto que a área de Internet tem como objetivo prestar serviços nesta área e vender soluções de compra e venda pela Web. Para este experimento, foram descartadas as áreas envolvidas no processo de desenvolvimento de hardware. As aplicações de software são típicas para as áreas determinadas e são produtos próprios da empresa.

Por não estar atrelada a evidências estatísticas esta primeira etapa foi uma pesquisa exploratória. A pesquisa foi feita na forma de um questionário, distribuído aos desenvolvedores pertencentes a diversas áreas produtoras de software da Itautec Philco S. A., o qual era preenchido quando o desenvolvedor localizava um defeito. Embora seus resultados não tenham sido úteis para a determinação de quais fatores afetavam o aparecimento dos defeitos, esta pesquisa foi útil para levantar tipos de defeitos e caracterizar fatores adicionais que foram incorporados à segunda pesquisa. Esta pesquisa e os dados preliminares deste trabalho encontram-se no Anexo II deste trabalho.

A segunda etapa da pesquisa realizada entre os meses de julho a agosto de 2004 com um grupo de 10 desenvolvedores da área de Automação Bancária, teve o foco mais bem definido a partir das observações feitas pelos resultados da primeira

etapa. Esta fase contou com um grupo menor que permitiu o acompanhamento e observação das variáveis desejadas.

A base para a concepção de ambas as etapas da pesquisa foram os fatores humanos condicionantes de desempenho e a própria origem do erro humano.

Em ambas as versões foram feitas análises dos dados que procuraram caracterizar erros cometidos e identificados no processo de desenvolvimento de software em função de características dos projetos e dos indivíduos.

5.1 Metodologia de coleta e análise de dados

A seguir serão apresentados o procedimento utilizado para a pesquisa de campo, desde a elaboração do questionário até a forma de análise de resultados, além da apresentação na íntegra do questionário final aplicado.

5.2 Descrição do procedimento

Este item visa detalhar o procedimento de concepção do questionário utilizado, bem como as formas de análise dos resultados encontrados.

5.2.1 Elaboração do questionário

Como forma de coletar os dados necessários capazes de contribuir no objetivo deste trabalho elaborou-se um questionário a ser aplicado em equipes de desenvolvimento de software. O questionário visa investigar individualmente como cada desenvolvedor se envolve no processo de identificação e recuperação de

defeitos, bem como sua atuação sobre estes defeitos, e sua própria análise de causas destes defeitos.

A elaboração deste questionário teve sua base de formação na versão exploratória preliminar, o que permitiu ajustes e acertos nos objetivos almejados.

Não só questões organizacionais foram investigadas, mas também os fatores individuais como motivação, carga de trabalho, estado emocional e demais fatores organizacionais e individuais enumerados no capítulo 3.

Pelo fato do primeiro questionário ter sido distribuído no formato texto, a resposta obtida não foi a esperada. Em entrevista com os desenvolvedores, a aparente dificuldade na alteração da rotina diária para o preenchimento do questionário foi o fator determinante na ausência da participação de alguns. Fato este que indicou a necessidade de alteração no formato de apresentação para a segunda etapa da pesquisa.

As facilidades oferecidas pelos recursos da Internet foram determinantes para a decisão do formato da nova pesquisa. Somado a isto a análise sobre as atividades diárias dos desenvolvedores e possíveis impactos na produtividade pela colaboração no andamento desta pesquisa, ficou evidente que nem mesmo a publicação de um Site traria o comprometimento desejado. Por esta razão ficou definido o formato de publicação HTML enviado diariamente por email ao grupo de desenvolvedores, garantindo-se desta maneira um comprometimento superior a 95% de respostas diárias obtidas.

5.2.2 Questionário

O questionário desenvolvido, apresentado nas figuras 17, 18 e 19, foi apresentado aos desenvolvedores numa reunião realizada com todos para explicação dos objetivos, motivos da escolha individual de cada um pelo superior imediato, e esclarecimento de todas as questões, alternativas e forma de envio.

O questionário foi distribuído a desenvolvedores de software alocados em projetos diversos e em fase de evolução no ciclo de vida do projeto distintos, o que não os restringiu a atuar em apenas um projeto e apenas uma fase do ciclo de

desenvolvimento no período da pesquisa. Todos do grupo de desenvolvedores fazem implementação de software, mas nem todos participam de fases como especificação e projeto, por isso, a questão 1 de contextualização trata do início do processo de implementação, já que esta é uma etapa comum a todos.

O grupo foi selecionado com base no perfil individual do desenvolvedor pelo gerente da área. A premissa para a participação deste grupo era o engajamento na pesquisa, sem que isto causasse impacto nas atividades diárias de cada um. Contou-se com 10 desenvolvedores, que foram indicados dentro da empresa por seus superiores imediatos. Este número permitiu que o acompanhamento pessoal se mantivesse com o mesmo grau de qualidade do começo ao fim do processo de coleta de dados. Esta quantidade de desenvolvedores e a quantidade de dias em que o processo se baseou produzem resultados válidos.

Como forma de tornar a análise e o preenchimento mais fácil e consistente, o questionário foi dividido em três partes. A primeira delas, que teve preenchimento único, teve por objetivo identificar os fatores, independentes de tempo e de projeto, que são variáveis condicionantes de desempenho. Esta primeira etapa está identificada na figura 17.

Acompanhamento do projeto

1. A especificação do que você deve implementar está em que forma (selecione apenas a mais frequentemente usada no seu projeto)?

- ☐ Especificação Funcional
- ☐ Especificação Técnica
- ☐ e-mail
- ☐ Documento simples técnico
- ☐ Documento simples gerencial
- ☐ Conversa com usuário/técnicos

2. A equipe do projeto é:

- ☐ Excelente: coesa e todos os membros se dão bem
- ☐ Bom: coesa, mas nem todos os membros se dão bem
- ☐ Regular: individualista e cada um trabalha "na sua"

3. Em relação à política de remuneração e benefícios praticada pela empresa:

- ☐ Sinto-me plenamente satisfeito
- ☐ Sinto-me parcialmente satisfeito
- ☐ Sinto-me insatisfeito

4. Em relação às chefias com as quais você tem contato:

Capacitação: os líderes são:

- ☐ Plenamente capacitados para o cargo
- ☒ precisam de treinamento gerencial

Abertura: os líderes são:

- ☐ Abertos às opiniões de subordinados
- ☒ Não aceitam idéias ou sugestões

5. Você se sente valorizado e reconhecido?

- ☐ Sim
- ☐ Não

6. Em relação a sua equipe de trabalho, você identifica a maioria das pessoas como:

- ☐ Amigos
- ☐ Colegas de Trabalho
- ☐ Desafetos

Fig. 17 – Fatores de contextualização independentes do tempo

A segunda parte teve por objetivo contextualizar as variáveis organizacionais e individuais dependentes do tempo. Esta etapa é demonstrada na figura 18.

Acompanhamento diário – Projeto / Pessoal

7. O projeto é:

☐ Extremamente desafiador
☐ Interessante
☐ Normal
☐ Entediante

8. Dificuldades sentidas no projeto (Marque todas as que você considerar aplicáveis):

☐ Usuário muda muito de idéia
☐ Usuário não conhece bem o assunto
☐ Desconhecimento técnico para execução
☐ Não vejo dificuldades

9. Em que etapa o projeto se encontra atualmente?

☐ Especificação
☐ Manutenção
☐ Documentação
☐ Implementação
☐ Testes

10. No projeto você:

☐ Trabalha em grupo, dividindo tarefas e responsabilidades
☐ Trabalha sozinho porque ninguém mais tem conhecimento técnico para isto

11. Pessoalmente você hoje tem:

☐ Problemas particulares
☐ Problemas de saúde
☐ Problemas profissionais
☐ Problemas financeiros
☐ Não tenho problemas

12. Como você está hoje?

☐ Bem
☐ Alegre
☐ Confiante
☐ Doente
☐ Animado
☐ Mal
☐ Triste
☐ Nervoso
☐ Disposto
☐ Desanimado

13. Como você considera sua carga de trabalho?

☐ Excessiva
☐ Normal
☐ Baixa demais

Fig. 18 – Fatores diários

A terceira parte teve por objetivo contextualizar a identificação e recuperação dos erros de software. Esta última parte também trata de variáveis dependentes de tempo, e é demonstrada na figura 19.

Acompanhamento diário – por erro encontrado

14. O erro encontrado foi cometido:

☐ Por mim

☐ Por outra pessoa. Quem?

15. Este erro foi cometido em que fase do projeto segundo a sua visão?

☐ Especificação

☐ Manutenção

☐ Documentação

☐ Implementação

☐ Testes

16. Há quanto tempo este erro foi cometido? (aproximadamente)

☐ Hoje

☐ Há 2 semanas

☐ Há 1 mês

☐ Nesta última semana

☐ Há 3 semanas

☐ Há mais de 1 mês. Quando?

17. O erro foi encontrado por que motivo?

☐ Estou realizando testes

☐ Houve reclamações do usuário

☐ A equipe de testes (ou outra pessoa) testou e identificou o erro

18. Assinale o tipo de erro identificado:

☐ Erro de digitação

☐ Erro de integração com outros sistemas

☐ Erro de navegação

☐ Erro de interface

☐ Programação redundante para tratar o mesmo requisito

☐ Implementação incompleta

☐ Implementação errada

☐ Outro tipo. Qual?

19. Quais as causas para este erro em sua opinião?

☐ Falta de documentação

☐ Interrupções frequentes (telefonemas, reuniões, etc)

☐ Omissão de requisitos conhecidos

☐ Falta de conhecimento técnico para realizar a tarefa de forma segura e completa

☐ O requisito está errado

☐ Outras:

20. Quais as características de Qualidade de Software que deixam de ser atendidas por conta do erro encontrado?

☐ **Confiabilidade:** o produto de software é capaz de manter seu nível de desempenho, ao longo do tempo, nas condições estabelecidas

☐ **Funcionalidade:** as funções e propriedades específicas do produto, satisfazem as necessidades do usuário

☐ **Manutenibilidade:** refere-se ao esforço necessário para a realização de alterações específicas, no produto de software

☐ **Usabilidade:** esforço necessário para a utilização do sistema, baseado em um conjunto de implicações e de condições do usuário

☐ **Eficiência:** os recursos e os tempos envolvidos são compatíveis com o nível de desempenho requerido pelo software

☐ **Portabilidade:** facilidade de rodar em múltiplas plataformas

Fig. 19 – Fatores relacionados a cada erro encontrado

5.2.3 Divulgação

A divulgação do questionário utilizado foi feita na empresa Itaotec Philco S/A através da diretoria de automação bancária. Com a ajuda e apoio do diretor executivo da área, bem como de seus imediatos, o questionário foi divulgado e atingiu as áreas de desenvolvimento de software.

Como forma de oficializar a utilização destes dados como fonte de pesquisa, o diretor executivo da área de Automação Bancária consentiu, em nome da empresa, na divulgação dos dados da empresa para fins acadêmicos, conforme mostra a carta apresentada no Anexo II.

5.2.4 Coleta

A coleta de dados foi feita pela resposta automática diária no momento do envio do email pelos desenvolvedores. Os dados foram inseridos e trabalhados num banco de dados e em planilhas eletrônicas. Este trabalho foi realizado ao longo de 19 dias (de 13/julho/2004 a 06/agosto/2004) úteis e contou com um grupo de 10 desenvolvedores.

5.2.5 Análise

Como forma de análise destes questionários foi montado um banco de dados e planilhas de acompanhamentos individuais onde as variáveis pertinentes foram levantadas e foram cruzadas de maneira a permitir várias visões sobre os dados obtidos.

A análise individual de algumas perguntas trará benefícios e índices a serem contabilizados, outros fatores relevantes dependem do cruzamento de várias informações.

5.2.6 Obtenção dos Resultados

Os resultados serão apresentados de forma qualitativa das dimensões do ser humano envolvidas na produção de software. Para isso, elaborou-se uma série de consultas cruzadas a este banco de dados, que permitiram a coleta dos dados parciais apresentados no próximo capítulo.

5.2.7 Detalhamento do questionário

O questionário contou com o total de 20 questões sendo distribuídas da seguinte maneira: seis questões de contextualização, sete questões de contextualização organizacionais e individuais dependentes de projeto e tempo e seis questões de caracterização dos erros identificados.

A seguir, apresenta-se uma explicação individual de cada questão pertencente ao questionário, junto com os fatores humanos condicionantes de desempenho que formaram a base para a concepção das perguntas.

As questões de 1 a 6 estão agrupadas em “Acompanhamento do projeto” por serem questões que foram respondidas uma única vez e por tratarem de variáveis que são constantes no decorrer do tempo.

As questões de 7 a 12 estão agrupadas em “Acompanhamento diário – Projeto/Pessoal” por se tratarem de questões relativas ao projeto trabalhado no dia assim como as características pessoais que variam em função de vários aspectos.

As questões de 13 a 20 estão agrupadas em “Acompanhamento diário – por erro encontrado” por se tratarem de questões relativas ao erro identificado, suas causas, consequências e impacto sob a avaliação do desenvolvedor.

Questão 1-) Controle de processos.

A especificação do que você deve implementar está em que forma (selecione apenas a mais freqüentemente usada no seu projeto)? Respostas possíveis: Especificação Funcional, Especificação Técnica, e-mail, Documento simples técnico, Documento simples gerencial, Conversa com usuário/técnicos.

Esta questão faz referência ao controle dos processos. Como todos os desenvolvedores passam por esta situação, ela faz parte da contextualização. Para tornar a análise destes fatores mais clara, as respostas foram agrupadas da seguinte maneira: Especificação Funcional e Especificação Técnica, os quais foram considerados como processo estruturado. Documento simples técnico, Documento simples gerencial, que foram considerados um processo semi-estruturado por não compor a formalização adequada, mas sim um que indica a presença de um processo controlado. Conversa com usuário/técnicos e e-mail foram considerados como um processo desorganizado por evidenciarem a falta de controle.

Questão 2-) Comunicação.

A equipe do projeto é: Excelente: coesa e todos os membros se dão bem, Bom: coesa, mas nem todos os membros se dão bem, Regular: individualista e cada um trabalha isoladamente.

O critério adotado foi o de excelente, bom e regular respectivamente. Não se tratou o relacionamento como ruim pelo histórico da empresa, onde o regular mencionado acima é o pior caso.

Questão 3-) Sistemas de Recursos Humanos.

Em relação à política de remuneração e benefícios praticada pela empresa: Sinto-me plenamente satisfeito, Sinto-me parcialmente satisfeito, Sinto-me insatisfeito

As alternativas para esta questão estão relacionadas com a satisfação a respeito das políticas organizacionais como remuneração, benefícios, perspectivas profissionais, etc. Desta maneira, a abertura para qualquer uma das respostas fez-se viável.

Questão 4-) Liderança.

Em relação às chefias com as quais você tem contato. **Capacitação:** os líderes são: plenamente capacitados para o cargo, precisam de treinamento gerencial.

Abertura: os líderes são: Abertos às opiniões de subordinados, Não aceitam idéias ou sugestões.

Esta questão teve o objetivo de analisar a liderança sob o ponto de vista do desenvolvedor e não da organização.

Questão 5-) Reconhecimento.

Você se sente valorizado e reconhecido? Sim ou Não.

O sentimento individual de cada desenvolvedor pode ser expresso nesta.

Questão 6-) Relacionamento.

Em relação a sua equipe de trabalho, você identifica a maioria das pessoas como: amigos, colegas de trabalho, desafetos.

Nesta questão o desenvolvedor pode avaliar o seu relacionamento com os colegas de trabalho dizendo se considera as pessoas do seu convívio como amigos, colegas ou pessoas de desagrado. O critério usado nesta questão foi o de bom para amigos, razoável para colegas e ruim para desafetos.

Questão 7-) Motivação.

O projeto é: extremamente desafiador, interessante, normal ou entediante.

O intuito desta questão foi a análise de como cada desenvolvedor vê o projeto como trabalha. Classificaram-se os projetos como desafiador, interessante, entediante e normal.

Questão 8-) Dificuldades esperadas no projeto.

Dificuldades sentidas no projeto (Marque todas as que você considerar aplicáveis): usuário muda muito de idéia, usuário não conhece bem o assunto, desconhecimento técnico para execução, não vejo dificuldades.

Estas dificuldades foram escolhidas pelo fato de poderem se evidenciar em várias fases do ciclo de desenvolvimento do software.

Questão 9-) Fase do projeto.

Em que fase o projeto se encontra atualmente? especificação, documentação, implementação, testes e manutenção.

O intuito desde o início da pesquisa foi de não restringi-la aos desenvolvedores atuantes na fase de implementação.

Questão 10-) Papéis e responsabilidades.

No projeto você: trabalha em grupo, dividindo tarefas e responsabilidades, trabalha sozinho porque ninguém mais tem conhecimento técnico para isto.

Questão 11-) Problemas.

Pessoalmente você hoje tem: problemas particulares, problemas de saúde, problemas profissionais, problemas financeiros, não tenho problemas.

O fato do desenvolvedor apresentar problemas foi segmentado em áreas comuns a qualquer indivíduo.

Questão 12-) Estado Emocional.

Como você está hoje? Bem, alegre, confiante, doente, animado, mal, triste, nervoso, disposto e desanimado.

Nesta questão, que permite mais de uma resposta, o desenvolvedor pode retratar seus sentimentos. Para efeito de contabilização, estes sentimentos ficaram divididos em positivos e negativos, sendo que a resposta diária pôde variar entre -5 e +5, pois entende-se que a caracterização de mais de um estado indica a valorização do mesmo. O valor 5 se refere ao fato de existirem 5 variáveis que representam aspectos emocionalmente positivos e outras 5 que representam os aspectos emocionalmente negativos.

Questão 13-) Carga de Trabalho.

Como você considera sua carga de trabalho? Excessiva, normal, baixa demais.

A determinação deste item é subjetiva não havendo parâmetro para o desenvolvedor a não ser seu próprio julgamento.

Questão 14-) Responsável pela identificação do erro.

O erro encontrado foi cometido: por mim, por outra pessoa. Quem?

Para que a análise das causas do erro pudessem ser realmente atreladas a quem os cometeu, a identificação do responsável faz-se necessária.

Questão 15-) Fase em que o erro foi cometido.

Este erro foi cometido em que fase do projeto segundo a sua visão? Especificação, Documentação, Implementação, Testes, Manutenção.

Nesta questão, o desenvolvedor ao se deparar com o erro, teve que fazer uma análise da origem do erro segundo a sua visão e conhecimento sobre o projeto e sobre o erro identificado.

Questão 16-) Tempo de existência do erro.

Há quanto tempo este erro foi cometido? (aproximadamente): Hoje, Há 2 semanas, Há 3 semanas, Há 1 mês, Há mais de 1 mês.

O tempo de existência do erro, ou seja, a latência do erro, pode estar relacionada ao tipo de erro cometido, como sendo um simples deslize ou um engano.

Questão 17-) Motivo de se encontrar o erro.

O erro foi encontrado por que motivo? Estou realizando testes, Houve reclamações do usuário, A equipe de testes (ou outra pessoa) testou e identificou o erro.

O identificador do erro pode indicar a relação com o tipo de erro cometido, bem como a latência deste erro.

Questão 18-) Tipo de erro.

Assinale o tipo de erro identificado: Erro de digitação, Erro de integração com outros sistemas, Erro de navegação, Erro de interface, Programação redundante para tratar o mesmo requisito, Implementação incompleta, Implementação errada, Outro tipo. Qual?

A decisão pelos tipos básicos de erro teve origem na análise do erro humano. Reason (1990) propõe uma tabela genérica de erros humanos como mostra a figura 14 no capítulo 4. Aliado a isto foi feita uma análise de quais tipos são mais adequados ao desenvolvimento de software.

A seguir uma breve descrição do que é cada um dos tipos de erro propostos como alternativa, como foram explicados aos desenvolvedores que participaram da pesquisa:

- Erro de digitação: erro de sintaxe na linguagem de programação;
- Erro de integração com outros sistemas: erro identificado na entrada ou saída do módulo em execução, quando submetido a comunicação com outro sistema ou módulo, que gera resultados diferentes dos esperados;
- Erro de navegação: erros no encadeamento de telas do sistema;
- Erro de interface: erros na interface com o usuário;
- Programação redundante para tratar o mesmo requisito: tratamento excessivo ou não objetivo para o tratamento de um requisito;

- Implementação incompleta: erro identificado pela falta de completude ao se tratar um requisito, o que proporciona um resultado parcialmente correto;
- Implementação errada: erro que identifica o não cumprimento de um requisito do sistema.
- Outro tipo: erro não descrito entre as opções anteriores.

Questão 19-) Causas do erro.

Quais as causas para este erro em sua opinião? Falta de documentação, Interrupções frequentes (telefonemas, reuniões, etc), Omissão de requisitos conhecidos, Falta de conhecimento técnico para realizar a tarefa de forma segura e completa, O requisito está errado, Outras.

Para a identificação deste grupo de causas foram tomados como base a bibliografia sobre a Engenharia de Software, as reclamações dos desenvolvedores em análise anterior ao desenvolvimento do questionário e na experiência profissional adquirida. A possibilidade de indicação de outras causas de erro é essencial, já que só as mais frequentes estão na forma de alternativas.

A questão é de múltipla escolha, pois a combinação de fatores de causa pode ser evidenciado.

Questão 20-) Características de qualidade de software prejudicadas pela presença do erro.

Quais as características de Qualidade de Software que deixam de ser atendidas por conta do erro encontrado?

- *Confiabilidade*: o produto de software é capaz de manter seu nível de desempenho, ao longo do tempo, nas condições estabelecidas;
- *Funcionalidade*: as funções e propriedades específicas do produto, satisfazem as necessidades do usuário;
- *Manutenibilidade*: refere-se ao esforço necessário para a realização de alterações específicas, no produto de software;
- *Usabilidade*: esforço necessário para a utilização do sistema, baseado em um conjunto de implicações e de condições do usuário;
- *Eficiência*: os recursos e os tempos envolvidos são compatíveis com o nível de desempenho requerido pelo software;

- *Portabilidade*: facilidade do software ser executado em múltiplas plataformas.

Nesta questão foram utilizadas as características e suas definições de acordo com a norma ISO 9126. Cada característica foi seguida de uma breve definição para evitar interpretações equivocadas pelos desenvolvedores.

6 APRESENTAÇÃO DE RESULTADOS

Os resultados obtidos serão apresentados a seguir:

6.1 Apresentação da Amostra

Os totais obtidos para esta versão do questionário são apresentados na tabela 1 a seguir:

Totais	
Desenvolvedores	10
Erros	84

Tabela 1 – Totais analisados

6.2 Resultados

A seguir, serão apresentados os resultados de cada questão bem como a análise de cada um dos fatores aos quais elas se referem.

As análises e conclusões sobre cada questão têm o fator de relevância a ser considerado que é o conhecimento minucioso da empresa na qual a pesquisa foi realizada, já que a autora é funcionária desta mesma empresa.

As questões de 1 a 6 tem por base a totalização de valores idêntica ao número de desenvolvedores já que as respostas foram dadas um única vez, uma por desenvolvedor.

Questão 1-) Grau de controle dos processos.

O grau de controle dos processos organizacionais mostrou-se bem equilibrado, e indica um controle parcial sobre os processos na empresa. Esta parcialidade era esperada já que a empresa passa atualmente por uma reestruturação

em todas as suas áreas, de maneira a unificar seus processos, para adoção de um modelo de qualidade, o CMMI.

Dos 10 desenvolvedores participantes da pesquisa, 5 evidenciaram a formalização dos processos nos quais estão envolvidos enquanto que os outros 5 mostraram a falta de formalização nestes processos, o que para efeito de classificação foi considerado como desorganização. Não houve respostas que indicassem um processo semi-estruturado. Os resultados obtidos em percentual são mostrados no gráfico 1.

Gráfico 1 – Grau de controle de processos

Questão 2-) Comunicação.

A comunicação entre os membros das equipes de desenvolvimento é boa. Este fator se deve ao ambiente de trabalho que é bom e a competição profissional é comedida.

A comunicação entre os componentes das equipes desenvolvedoras de software foi analisado do ponto de vista de coesão o que mostrou que em 60% dos casos a comunicação é boa e apenas 10% considera a comunicação regular, como mostra o gráfico 2.

Gráfico 2 – Comunicação entre os desenvolvedores

Questão 3-) Sistemas de Recursos Humanos.

A maioria dos desenvolvedores envolvidos na pesquisa mostrou-se insatisfeito com as políticas e sistemas de recursos humanos adotados pela empresa, mesmo sendo a satisfação um dos fatores organizacionais mais discutidos e presentes em qualquer área de concentração. Como pode se observar no gráfico 3, no grupo de desenvolvedores a metade dos desenvolvedores se mostrou insatisfeita com o sistema e as políticas da empresa enquanto que 40% se disseram parcialmente satisfeitos e apenas 10% se disseram plenamente satisfeitos.

Gráfico 3 – Políticas e Sistemas de Recursos Humanos

Questão 4-) Liderança.

A maioria dos gerentes avaliados pelos desenvolvedores precisam de treinamento para liderar. Este era um resultado esperado já que a política de promoção da empresa vincula-se ao desempenho técnico de cada indivíduo. Neste ponto a empresa é mesmo carente de líderes bem treinados para lidar com situações gerenciais, salvo algumas exceções.

Sob o ponto de vista de capacitação gerencial, a grande maioria dos desenvolvedores, 70%, identificou a necessidade de treinamento dos gerentes com os quais tem contato, enquanto que apenas 30% identificou que os gerentes têm plena capacitação gerencial. Estes resultados são visíveis no gráfico 4.

Em contrapartida, a grande maioria, 90% dos desenvolvedores indicaram que os líderes são abertos a opiniões de subordinados, como se observa no gráfico 5. Mais uma vez o bom relacionamento se faz presente nas respostas obtidas.

Gráfico 4 – Capacitação Gerencial

Gráfico 5 – Abertura à opinião de funcionários

Questão 5-) Reconhecimento.

O reconhecimento parcial assinalado pelos desenvolvedores é reflexo da insatisfação com as políticas e sistemas de recursos humanos, já que para a maioria o reconhecimento se dá por benefícios e/ou remuneração diferenciada.

A metade dos desenvolvedores se disse reconhecida na empresa e a outra metade não, como pode se observar no gráfico 6. O sentimento de reconhecimento particular a cada desenvolvedor é genérica e bastante abrangente, mas se mostra equilibrada.

Gráfico 6 – Reconhecimento profissional ao desenvolvedor

Questão 6-) Relacionamento.

O relacionamento não foi considerado ruim, apenas bom, ou seja, indica a falta de excelência.

50% dos desenvolvedores consideraram o relacionamento bom e os outros 50% razoável como mostra o gráfico 7. Não há evidências de desafetos entre os desenvolvedores

Gráfico 7 – Relacionamento entre a equipe

Acompanhamento Diário - Projeto/Pessoal

As questões de 7 a 13 são questões com respostas diárias e por isso têm sua totalização com valores superiores ao mostrados até agora. A quantidade de respostas obtidas nesta parte equivale aos 19 dias úteis (período de 13/julho/2004 a 06/agosto/2004) em que a pesquisa foi realizada.

Questão 7-) Motivação.

A motivação nos projetos mostrou-se alta. Estas respostas mostraram que mesmo os desenvolvedores que se dizem insatisfeitos com as políticas organizacionais têm uma visão positiva dos projetos nos quais estão inseridos. No gráfico 8 vê-se que 90% dos desenvolvedores classificaram os projetos nos quais estão envolvidos como motivantes, e apenas 10% consideram o projeto entediante.

Gráfico 8 – Motivação do desenvolvedor no projeto

Questão 8-) Dificuldades esperadas no projeto.

A maioria dos desenvolvedores não encontrou dificuldades nos projetos trabalhados na empresa no período em que esta pesquisa foi realizada. Isto pode ser observado pelos percentuais do gráfico 9 que mostra 62% dos desenvolvedores sem dificuldades para realizarem suas tarefas nos projetos, enquanto que cerca de 18% se disseram despreparados tecnicamente para desempenhar sua função e cerca de outros 18% disseram que a constante mudança de idéia por parte dos usuário é que trazem as grandes dificuldades para o projeto. Cerca de 2% atribuiu as dificuldades à falta de conhecimento do usuário acerca do assunto. É importante notificar que a mudança de foco por parte da organização não foi considerada uma dificuldade por nenhum dos desenvolvedores.

Gráfico 9 – Dificuldades encontradas no projeto

Questão 9-) Fases do projeto.

A maioria dos desenvolvedores estava envolvida na etapa de implementação dos projetos realizados na época desta pesquisa, embora se tenha buscado a diversidade de fases de desenvolvimento de software nos projetos selecionados. No gráfico 10 se observa que esta ampla maioria corresponde a 68% do total. As fases de especificação, manutenção, testes e documentação mostram um percentual bem inferior: 12%, 10%, 9% e 2% respectivamente.

Gráfico 10 – Etapas de Projeto de Software

Questão 10-) Papéis e responsabilidades.

As responsabilidades distribuídas e o papel bem definido no trabalho em equipe são evidenciados nas respostas a esta questão. No gráfico 11 vê-se que o trabalho em equipe com divisão de tarefas e responsabilidades é realidade em 81% dos casos, enquanto que em apenas 19% dos casos o trabalho e as responsabilidades ficam a cargo de uma única pessoa.

Gráfico 11 – Papéis e responsabilidades atribuídas ao desenvolvedor

Questão 11-) Problemas.

A maioria dos desenvolvedores relatou algum tipo de problema durante o período da realização desta pesquisa.

Das possibilidades de problemas apresentados aos desenvolvedores,, no questionário 46% afirmou não ser afligido por nenhum deles, ou seja, não têm problemas. Mas como pode ser percebido no gráfico 12, em 20% dos casos os desenvolvedores se mostraram com problemas particulares e, em 24% dos casos, com problemas profissionais. Outros 9% afirmaram passar por problemas financeiros e 2% mostrou passar por problemas de saúde pessoal ou familiares. A soma destes percentuais com referências a problemas totaliza 54%.

Gráfico 12 – Problemas enfrentados pelo desenvolvedor

Questão 12-) Estado Emocional.

A análise do estado emocional baseou-se no seguinte critério: foram apresentados 10 estados emocionais ao desenvolvedor, sendo que cinco deles representam estados positivos (bem, alegre, confiante, disposto, animado) e outros cinco representam estados negativos (mal, triste, nervoso, doente, desanimado) como mostra a tabela 2. A questão permitiu respostas múltiplas. Desta forma, a quantidade de questões assinaladas indica o maior ou menor grau do estado emocional no dia em questão. Por exemplo, o desenvolvedor que assinalou apenas ‘bem’ no dia tem seu estado positivo menor que o desenvolvedor que assinalou ‘bem’, ‘confiante’ e ‘disposto’. Todas as alternativas assinaladas foram somadas e computadas como respostas e o total obtido é mostrado na tabela 3.

Não evidenciou-se estados emocionais contraditórios, ou seja, um único desenvolvedor identificar estado positivo e negativo no mesmo dia.

O estado emocional se mostrou positivo na grande maioria das vezes, correspondendo a 90% dos casos, o que pode ser observado no gráfico 13.

Gráfico 13 – Estado emocional apresentado pelos desenvolvedores

Estado Emocional	
Positivo	Negativo
Bem	Mal
Alegre	Triste
Confiante	Nervoso
Disposto	Doente
Animado	Desanimado

Tabela 2 – Distribuição das possibilidades de Estado Emocional em positivo e negativo

Totais	
Positivo	345
Negativo	40

Tabela 3 – Totais obtidos sobre o Estado Emocional dos desenvolvedores

Questão 13-) Carga de Trabalho.

Nenhum desenvolvedor trouxe a evidência de carga de trabalho baixa. Surpreendentemente, como mostra o gráfico 14, a carga de trabalho mostrou-se acima do normal em apenas 26% dos casos enquanto que a carga de trabalho considerada normal foi relatada por 74% dos desenvolvedores.

Gráfico 14 – Carga de trabalho

Acompanhamento Diário – por erro encontrado

As questões de 14 a 20 são questões com respostas a cada erro encontrado sem que isto possuísse uma regra de tempo ou quantidade. O total de respostas obtidas neste trecho do questionário foi de 84, o que evidencia o número de erros relatos no período da pesquisa.

Questão 14-) Responsável pela identificação do erro.

Em 67% das ocorrências, o próprio desenvolvedor é quem comete e detecta o erro e apenas em 33% o erro é identificado por outras pessoas, como mostra o gráfico 15. Quando a identificação do erro ocorre por parte de outras pessoas, poucos relatos identificaram o usuário do sistema como sendo o real identificador do problema. Na maioria das situações estas respostas estiveram relacionadas à equipe de testes ou outro desenvolvedor.

Gráfico 15 – Responsável pela identificação do erro

Questão 15-) Fase em que o erro foi cometido.

A maioria dos erros identificados foram cometidos na fase de implementação, o que corresponde a 71% dos erros identificados como aponta o gráfico 16. Em 17% dos casos os erros foram apontados como provenientes da fase de especificação, 6% provenientes da fase de testes e apenas 4% da fase de manutenção.

Gráfico 16 – Fase do desenvolvimento em que o erro foi cometido

Questão 16-) Tempo de existência do erro.

A maioria dos erros foi cometido e identificado no mesmo dia, o que corresponde a 51% das ocorrências. Além deste número, o gráfico 17 mostra o outro extremo que é a identificação de erros existente a mais de um mês, o que equivale a 24% do total.

Gráfico 17 – Tempo de existência do erro detectado

Questão 17-) Motivo de se encontrar o erro.

Os testes, realizados pelos próprios desenvolvedores, mostraram-se como as maiores causas de identificação de erros (72%), como se pode verificar no gráfico 18. Em 17% dos casos relatados, os testes por outras pessoas, quer sejam de uma área específica de testes ou não, foram os motivadores desta identificação, enquanto que em apenas 11% dos casos o relato de um erro partiu do usuário.

Gráfico 18 – Motivo da identificação do erro

Questão 18-) Tipo de erro.

Dos oito tipos de erro postulados, a serem identificados pelos desenvolvedores, aqueles que se apresentaram com maior intensidade foram os erros de implementação, quer sejam provenientes de implementação incompleta ou errada.

Ambas somam 70% dos erros cometidos e identificados. O gráfico 19.1 apresenta que em 47% dos casos os erros foram de implementação errada e outros 23% foram de implementação incompleta.

Como a questão possibilitava ao desenvolvedor responder que o erro encontrado seria outro que não aqueles estabelecidos na pesquisa, as respostas variaram muito, como é apresentado no gráfico 19.2.

De acordo com o que os desenvolvedores apresentaram se percebe uma confusão entre causa e consequência dos erros, já que os erros relatados puderam ser lidos como causa dos erros.

Gráfico 19.1 – Tipo de erro identificado

Gráfico 19.2 - Outros tipos de erro identificados e relatados e identificados

Questão 19-) Causas do erro.

Pelos valores obtidos como resposta, se percebe o equilíbrio entre as causas identificadas pelos desenvolvedores. O trato dos requisitos foram os menores causadores de erro, sendo que 10% dos erros foram relacionados a omissão de requisitos e apenas 3% dos erros foram atrelados a existência de requisitos errados, como mostra o gráfico 20.

O relato de outros tipos de causa para os erros identificados mostrou-se alto, contando com 23% das ocorrências, mas na justificativa destas causas se pode perceber que os desenvolvedores relataram formas de identificação distintas e não novas causas.

Gráfico 20 – Causas do erro identificado

Questão 20-) Características de qualidade de software prejudicadas pela presença do erro.

A característica de qualidade com maior impacto pelos erros identificados é a funcionalidade com 43%, seguida da confiabilidade com 19%, como mostra o gráfico 21. A portabilidade foi a característica com menor impacto pelos erros, segundo os desenvolvedores, com apenas 5% das evidências.

Gráfico 21 – Características de Qualidade de Software prejudicadas pela presença de erros

6.3 Cruzamento de Resultados

Na busca pelas evidências, o cruzamento de alguns dos resultados, até agora mostrados individualmente, mostra-se necessário. A seguir é feita uma análise destes cruzamentos.

Neste item alguns cruzamentos são interessantes para que se tenha uma visão geral da relação entre os dados. A seguir uma relação dos cruzamentos que serão relatados:

Cruzamento 1 -) Controle de Processos por Erros detectados

Cruzamento 2-) Dificuldades encontradas pelo desenvolvedor pela etapa de trabalho.

Cruzamento 3-) Motivação pela Carga de Trabalho

Cruzamento 4-) Carga de Trabalho pelas Causas de Erro

Cruzamento 5-) Erro pela Causa

Cruzamento 6-) Erros pelas Características de qualidade sobre as quais tem impacto.

Cruzamento 7-) Tempo de existência do erro pela Fase em que foi cometido.

Cruzamento 8-) Tempo de existência do erro pelo Tipo de erro encontrado

Cruzamento 1 -) Controle de Processos por Erros detectados

O gráfico 22 mostrado a seguir dá a quantidade de erros encontrados em cada um dos tipos de processo. Pelo apresentado, 25% dos erros encontrados por aqueles desenvolvedores que seguem um processo formal e estruturado de desenvolvimento são erros de implementação incompleta e 49% são erros de implementação errada, o que totaliza 74% do total de erros. Já dos erros detectados nos processos considerados desorganizados 14% corresponde a erros de implementação incompleta e 38% a implementação errada, o que totaliza 52% do total de erros. Portanto, não há evidências para esta pesquisa, na empresa estudada de que a falta de controle no processo possa influenciar em maior quantidade de erros de implementação.

Gráfico 22 – Controle de processos versus quantidade de erros identificados

Cruzamento 2-) Dificuldades encontradas pelo desenvolvedor pela fase de trabalho.

Na fase de implementação a maioria dos desenvolvedores, 57%, não encontrou dificuldades, e em 21% o desconhecimento técnico é apontado como dificuldade nesta fase como se pode observar no gráfico 23. Nas demais fases a ausência de dificuldade para o cumprimento das tarefas é a maior evidência.

Gráfico 23 – Dificuldades encontradas versus a etapa de desenvolvimento

Cruzamento 3-) Motivação pela Carga de Trabalho

Este cruzamento não mostra grandes novidades, pois nos projetos considerados entediantes a carga de trabalho se mostra equilibrada entre normal e excessiva, como se observa no gráfico 24. Um fato interessante é que em cerca de 30% dos casos a carga é excessiva o desenvolvedor continua considerando o projeto como interessante.

Gráfico 24 – Dificuldades encontradas versus a etapa de desenvolvimento

Cruzamento 4-) Carga de Trabalho pelas Causas de Erro

O número de erros cometidos e identificados com carga de trabalho normal é superior aos cometidos com carga de trabalho excessiva como mostra a tabela 5.

A tabela 4 mostra a diferença de percentuais entre as causas de erros quando a carga de trabalho é excessiva ou normal. Nota-se um equilíbrio entre os vários fatores de causa de erro quando a carga é excessiva, mas os requisitos são os fatores de menor incidência com 11% de omissão de requisitos e apenas 3% de requisitos errados. Já quando a carga de trabalho é normal o desenvolvedor identificou em 34% dos relatos com outras causas de erro como se observa no gráfico 25. Nesta atribuição de carga de trabalho se percebe a variação da incidência de determinadas causas de erro, mas um fator a ser destacado é o valor de 24% das causas a serem atribuídas a interrupções, fator que acontece com menos incidência (20%) quando a carga de trabalho é excessiva. Nenhuma ocorrência de carga de trabalho baixa foi relatada.

	Normal	Excessiva
Falta de documentação	17%	23%
Interrupções	24%	20%
Omissão de requisitos	5%	11%
Falta de conhecimento técnico	17%	23%
Requisito errado	3%	3%
Outros	34%	20%

Tabela 4 – Causas de erros distribuídas pela carga de trabalho

Normal	55
Excessiva	29
Baixa	0

Tabela 5 – Totais de erros pela carga de trabalho

Gráfico 25 – Carga de trabalho versus a causa de erros

Cruzamento 5-) Erro pela Causa.

Na análise deste cruzamento algumas evidências já eram esperadas, como é o caso da causa interrupção para o erro de digitação e de implementação incompleta. A tabela 6 mostra a distribuição de deslizes e lapsos ressaltados em azul e os enganos em vermelho.

O gráfico 26 mostra que 43% dos erros de implementação incompleta são provenientes de interrupções e outros 33% atribuídos a outros fatores. Já nos erros de implementação errada 33% são atribuídos a falta de conhecimento técnico.

Nos erros de navegação, interface e erros que não tiveram enquadramento possível nas categorias apresentadas por julgamento do desenvolvedor, tiveram como causa apontada em pelo menos 50% dos casos a falta de documentação.

Os erros de redundância foram apontados como provenientes da falta de documentação em 100% dos casos. Pela baixa incidência deste tipo de erro - apenas um erro foi relatado - conclusões ficam comprometidas.

	Digitação	Integração	Navegação	Interface	Redundância	Implementação incompleta	Implementação errada	Outros
Falta de documentação	0%	0%	67%	50%	100%	14%	20%	40%
Interrupções	60%	21%	0%	0%	0%	41%	12%	20%
Omissão de requisitos	0%	14%	33%	25%	0%	5%	5%	20%
Falta de conhecimento técnico	0%	14%	0%	25%	0%	5%	34%	0%
Requisito errado	0%	7%	0%	0%	0%	5%	2%	0%
Outros	40%	43%	0%	0%	0%	32%	27%	20%

Enganos

Deslizes e Lapsos

Tabela 6 – Distribuição de erros pela causa em percentuais

Gráfico 26 – Erro detectado versus causa

Cruzamento 6-) Erros pelas Características de qualidade sobre as quais tem impacto.

A característica de qualidade de software que tem mais impacto nos erros indicados, segundo os desenvolvedores, é a funcionalidade com um total de 43% das respostas como mostra o gráfico 27. Logo após esta característica, vem a

confiabilidade, prejudicada em quase todos os tipos de erros com 19% dos apontamentos.

Gráfico 27 – Características de Qualidade que sofreram impactos

Cruzamento 7-) Tempo de existência do erro pela Fase em que foi cometido.

O gráfico 28 relata que erros provenientes da fase de especificação, em sua maioria - 53% dos relatos - existem há mais de um mês, características dos enganos ou erros de conceito, de mais difícil diagnóstico por estarem associados a uma construção mental enganada. Os erros da fase de implementação se concentram mais no próprio dia da identificação - 60% dos casos. Na fase de testes a diversidade é maior, pois 40% dos erros foram cometidos no dia da identificação, outros 40% no tempo aproximado de duas semanas e 20% no período do último mês. Esta diversidade da fase de Testes pode ser explicada pelo fato de existir uma equipe focada em testes que trabalha paralelamente a outras fases de todos os projetos. Por ser esta uma fase que retorna ao desenvolvimento quando algum erro é encontrado a periodicidade mostra-se mais próxima à introdução do erro.

Não foram relatados erros associados à documentação.

Gráfico 28 – Tempo de existência do erro versus a fase em que foi cometido

Cruzamento 8-) Tempo de existência do erro pelo Tipo de erro encontrado

O gráfico 29 relata que do total de respostas obtidas, 67% dos erros de digitação, 21% dos erros de integração, 67% dos erros de interface, 78% dos erros de implementação incompleta e 47% dos erros de implementação errada, tem como tempo de latência do erro um dia.

Em 33% dos erros de digitação, 29% dos erros de integração, 50% dos erros de navegação, 33% dos erros de interface, 100% dos erros de redundância, 22% dos erros de implementação incompleta e 29% dos erros de implementação errada, aconteceram com um tempo de latência superior a um mês.

Os erros que são provenientes de deslizes e lapsos são os que têm latência predominantemente no dia em que ocorrem, como é o caso dos erros de digitação e implementação incompleta. Os erros que se mostram latentes após um mês de terem sido cometidos são enganos, e por isso, mais difíceis de serem detectados. O equilíbrio em torno de 30% mostra que um engano não está atrelado somente a erros de implementação.

O fato de 100% dos erros de redundância serem latentes por mais de um mês não é expressivo nesta amostra, pois se tratou de um único erro.

Gráfico 29 – Tempo de existência do erro versus a fase em que foi cometido

6.4 Cruzamento de Resultados baseados nos Tipos de Erro

A seguir tem-se nos gráficos de 30 a 34 a apresentação do cruzamento dos resultados baseados nos tipos de erro e nas variáveis de processo que são independentes do tempo.

Tipo de Erro versus a comunicação

Os resultados obtidos eram esperados já que o ambiente de coleguismo existente na empresa proporcionam esta facilidade. Além disto, a boa comunicação nas atividades cruciais do projeto pode ser explicada pelo maior grau de especialização dos desenvolvedores envolvidos nestas fases do projeto – como encontra-se destacado no gráfico 30 – e também, pelo fato de que estes desenvolvedores são aqueles que possuem cargos mais altos, e portanto, mais próximos às chefias.

Gráfico 30 – Tipo de erro versus a comunicação

Tipos de Erro *versus* satisfação com as políticas organizacionais, ou seja, o sistemas de Recursos Humanos

Neste cruzamento os resultados obtidos também eram esperados já que a satisfação é indicada pela maior ou menor remuneração que é a base da política de recursos humanos da empresa. O grau de especialização, bem como o cargo dos desenvolvedores envolvidos nas fases de integração e implementação nos projetos pesquisados é alto. Disto pode-se concluir que a insatisfação com salários e benefícios destes desenvolvedores é maior do que a dos desenvolvedores com cargos inferiores, o que se mostra com evidência no gráfico 31.

Gráfico 31 – Tipo de erro versus satisfação com as políticas organizacionais

Tipos de Erro versus a liderança

Como nos resultados demonstrados pelo gráfico 30, o resultado para este cruzamento, demonstrados no gráfico 32, era esperado já que os desenvolvedores com maior cargo, e portanto, mais próximos às lideranças estão envolvidos nas fases de integração e implementação.

Gráfico 32 – Tipo de erro versus a liderança

Tipos de Erro *versus* o reconhecimento profissional

A maioria dos desenvolvedores que não se sente reconhecida está envolvida com tipos de erros complexos e cruciais nos projetos de software, os enganos. Esta evidência se dá principalmente nos erros de implementação errada e incompleta como ressaltado no gráfico 33.

Os resultados do cruzamento para os erros de interface não produzem uma análise consistente pela baixa quantidade de erros deste tipo cometidos e relatados.

Gráfico 33 – Tipo de erro versus reconhecimento profissional

Tipos de Erro *versus* o relacionamento entre os desenvolvedores

Mais uma vez o ambiente de coleguismo da empresa se evidencia neste cruzamento mostrado no gráfico 34. O relacionamento é melhor nas atividades onde o grau de proximidade do desenvolvedor à chefia é maior.

Gráfico 34 – Tipo de erro versus relacionamento entre desenvolvedores

A maior parte dos erros relatados diz respeito a implementação de software, quer seja uma implementação errada ou uma implementação incompleta. Notavelmente, temos acentuação de alguns fatores na presença destes tipos de erro.

Para tornar estes dados mais evidentes é preciso classificar algumas respostas. No caso de questões em que as opções de resposta eram excelente, bom e razoável, percebe-se a maioria das respostas como bom. Isto, em termos de análise, evidencia que o fator analisado poderia ser melhor do ponto de vista do desenvolvedor.

Pelos gráficos apresentados, percebe-se que a comunicação analisada pelo desenvolvedor poderia ser melhor, e esta recorrência está presente em todos os tipos de erro. Da mesma maneira a necessidade de capacitação dos líderes é uma evidência relevante, sendo assim julgada pelos desenvolvedores na maioria dos erros, sendo que em todos eles esta incidência é superior a 50%. O mesmo se verifica ainda sobre o relacionamento entre os desenvolvedores e a equipe onde trabalham. A possibilidade de se ter este fator com melhores indicações é evidente em mais de 50% em todos os erros identificados.

Alguns fatores têm relação direta com alguns tipos de erros, como é possível se verificar no caso da insatisfação. Ela está atrelada especialmente a erros de integração (31%), implementação errada (38%) e incompleta (27%). Ainda nesta

linha, a falta de reconhecimento se mostra diretamente relacionada com os erros de implementação incompleta (24%) e errada (51%).

Somado a estes fatores, também a identificação de problemas profissionais com percentual maior que os problemas pessoais reforça a idéia de insatisfação e o sentimento de falta de reconhecimento.

6.5 Resultados por indivíduo

Para uma análise criteriosa da relação entre os diversos fatores condicionantes aos quais todos os indivíduos estavam suscetíveis, a apresentação dos resultados individuais ao longo do tempo se mostra de grande valor.

Para preservar a identidade dos desenvolvedores participantes da pesquisa, seus nomes serão substituídos por nomes fictícios.

Joana

Esta desenvolvedora mostrou-se na normalidade, pois como se pode observar nos gráficos a seguir, os projetos são considerados por ela normais, ela não apresenta dificuldades no trabalho, está implementando atualmente, não apresentou problemas no trabalho em equipe, na maioria dos dias mostrou uma carga de trabalho normal e o estado emocional teve picos muito negativos seguidos de picos positivos. Apesar dela se encontrar na fase de implementação, ela não reportou erros. Pelo exposto por ela ao longo do decorrer da pesquisa, a razão para o fato aconteceu por ela estar desenvolvendo sempre com alguém ao seu lado o que não permitiu tempo e, principalmente, privacidade para as respostas pedidas.

Gráfico 35 – Motivação da desenvolvedora Joana

Gráfico 36 – Dificuldades da desenvolvedora Joana

Gráfico 37 – Fases de trabalho da desenvolvedora Joana

Gráfico 38 – Papéis e responsabilidades da desenvolvedora Joana

Gráfico 39 – Problemas da desenvolvedora Joana

Gráfico 40 – Estado Emocional da desenvolvedora Joana

Gráfico 41 – Carga de trabalho da desenvolvedora Joana

Maria

Mostrou alta motivação na maioria dos dias, as maiores mudanças e variações ocorreram entre 13 e 23 de julho com picos de relato dos erros entre os dias 19 e 21 de julho. Neste período ela se mostrou motivada, estava implementando, sem problemas, e seu estado emocional atingiu um grande pico. A carga de trabalho estava normal, os erros apontados existiam há cerca de uma semana e provenientes de problemas de implementação e causados por falta de testes específicos.

Gráfico 42 – Motivação da desenvolvedora Maria

Gráfico 43 – Dificuldades da desenvolvedora Maria

Gráfico 44 – Fases de trabalho da desenvolvedora Maria

Gráfico 45 – Papéis e responsabilidades da desenvolvedora Maria

Gráfico 46 – Problemas da desenvolvedora Maria

Gráfico 47 – Estado Emocional da desenvolvedora Maria

Gráfico 48 – Carga de trabalho da desenvolvedora Maria

Maria – variáveis de Erro

Gráfico 49 – Identificador x Motivo - desenvolvedora Maria

Gráfico 50 – Fase do erro - desenvolvedora Maria

Gráfico 51 – Tempo de existência do erro - desenvolvedora Maria

Gráfico 52 – Tipo de erro - desenvolvedora Maria

Gráfico 53 – Causas de erro - desenvolvedora Maria

Gráfico 54 – Características de Qualidade afetadas pelos erros - desenvolvedora Maria

Elisa

Mostrou-se motivada nos projetos que foram relatados como interessantes em sua maioria, ela apresentou dificuldades pela mudança constante de idéia por parte do usuário, e estava implementando na maior parte do tempo. No período de 16 a 22 de julho houve uma mudança em seus gráficos que apresentam uma baixa na motivação, erros apresentados pelo usuário o que a levou a fase de manutenção e ao trabalho em equipe por um dia. Ela apresentou problemas profissionais na grande maioria dos dias, um estado emocional que oscilou pouco se mantendo a maior parte do tempo bom e uma carga de trabalho, em 100% dos dias excessiva.

Esta foi a desenvolvedora que mais relatou erros ao longo da pesquisa, totalizando 28 erros. Um dos picos de relato de erros é o referido acima, com poucas identificações sendo feita pelo usuário. No mesmo período ela estava envolvida na especificação de outros projetos e se percebe que os erros relatados tinham data originária de mais de um mês. Os erros relatados eram de integração e implementação errada e tendo como causa omissão de requisitos e falta de documentação o que prejudicou a confiabilidade e a funcionalidade do sistema.

Gráfico 55 – Motivação da desenvolvedora Elisa

Gráfico 56 – Dificuldades da desenvolvedora Elisa

Gráfico 57 – Fase de trabalho da desenvolvedora Elisa

Gráfico 58 – Papéis e responsabilidades da desenvolvedora Elisa

Gráfico 59– Problemas da desenvolvedora Elisa

Gráfico 60 – Estado emocional da desenvolvedora Elisa

Gráfico 61 – Carga de trabalho da desenvolvedora Elisa

Elisa – variáveis por Erro

Gráfico 62 – Identificador do erro x Motivo - desenvolvedora Elisa

Gráfico 63 – Fase do erro - desenvolvedora Elisa

Gráfico 64 – Tempo de existência do erro- desenvolvedora Elisa

Gráfico 65 – Tipos de erro - desenvolvedora Elisa

Gráfico 66 – Causas do erro - desenvolvedora Elisa

Gráfico 67 – Características de Qualidade afetadas pelos erros - desenvolvedora Elisa

João

Este desenvolvedor se apresentou motivado, trabalhou parte do tempo na fase de manutenção do sistema o qual considerou o projeto interessante. Na fase de implementação, em que esteve envolvido a maior parte do tempo, o projeto mostrou-se desafiador. Por apresentar problemas particulares praticamente todos os dias relatados, o estado emocional se mostrou bastante instável. Os erros relatados por ele foram identificados, em sua maioria, pela equipe de testes e variaram muito. Os erros considerados como implementação errada tiveram sua origem na falta de teste específico ou falta de documentação. Por todos os erros identificados por ele, e por sua análise, o impacto maior destes erros foi quanto à confiabilidade do sistema.

Gráfico 68 – Motivação do desenvolvedor João

Gráfico 69 – Dificuldades do desenvolvedor João

Gráfico 70 – Fase de trabalho do desenvolvedor João

Gráfico 71– Papéis e responsabilidades do desenvolvedor João

Gráfico 72 – Problemas do desenvolvedor João

Gráfico 73 - Estado Emocional do desenvolvedor João

Gráfico 74 - Carga de trabalho do desenvolvedor João

João – variáveis por Erro

Gráfico 75 - Identificador x Motivos - desenvolvedor João

Gráfico 76 – Fase em que o erro foi cometido - desenvolvedor João

Gráfico 77– Tempo de existência do erro - desenvolvedor João

Gráfico 78– Tipos de erro - desenvolvedor João

Gráfico 79 – Causas de erro - desenvolvedor João

Gráfico 80 – Características de Qualidade afetadas pelos erros - desenvolvedor João

Rafael

As oscilações de motivação no projeto mostram-se relacionadas com o fato deste desenvolvedor desconhecer tecnicamente como executar suas funções e estar fazendo especificação e documentação de sistema. Estes picos possíveis de serem relacionados acontecem entre os dias 20 e 22 de julho e também de 26 a 30 de julho. Apesar de não apresentar problemas, o desenvolvedor apresentou um estado emocional positivo, porém baixo. Isto significa que ele identificou sempre uma das cinco possibilidades que caracterizam estado emocional positivo, restringindo-se sempre a apenas uma opção.

Nestes dois períodos foram relatados a maioria dos erros provenientes das fases de especificação e implementação, sendo grande parte erros de implementação

errada provenientes de erros de lógica e da falta de documentação, o que trouxe impacto direto na funcionalidade e eficiência em sua maioria.

Gráfico 81 – Motivação do desenvolvedor Rafael

Gráfico 82 – Dificuldades do desenvolvedor Rafael

Gráfico 83 – Fases de trabalho do desenvolvedor Rafael

Gráfico 84 – Papéis e responsabilidades do desenvolvedor Rafael

Gráfico 85 – Problemas do desenvolvedor Rafael

Gráfico 86 – Estado emocional do desenvolvedor Rafael

Gráfico 87 – Carga de trabalho do desenvolvedor Rafael

Rafael – variáveis por Erro

Gráfico 88 – Identificar x Motivo - desenvolvedor Rafael

Gráfico 89 – Fases em que o erro foi cometido - desenvolvedor Rafael

Gráfico 90 – Tempo de existência do erro -desenvolvedor Rafael

Gráfico 91 – Tipos de erro - desenvolvedor Rafael

Gráfico 92 – Causas de erro - desenvolvedor Rafael

Gráfico 93 – Características de Qualidade afetadas pelos erros - desenvolvedor Rafael

Fernando

Motivação alta, ausência de dificuldades, poucas evidências de problemas particulares, carga de trabalho adequada e estado emocional em alta mostraram a estabilidade deste desenvolvedor. Os erros relatados por ele foram identificados pela equipe de testes e vinculados com as fases de implementação e testes. Os erros cometidos na fase de implementação são latentes há mais de um mês, provenientes de implementação incompleta e tem como causa erros de lógica. Já os erros provenientes da fase de testes são latentes em uma semana e são erros de implementação errada. Por análise do desenvolvedor, estes erros foram ocasionados por falta de atenção. A característica de qualidade do sistema que mais teve impacto com estes erros foi a funcionalidade.

Gráfico 94 – Motivação do desenvolvedor Fernando

Gráfico 95 – Dificuldades do desenvolvedor Fernando

Gráfico 96 – Fases de trabalho do desenvolvedor Fernando

Gráfico 97 – Papéis e responsabilidades do desenvolvedor Fernando

Gráfico 98 – Problemas do desenvolvedor Fernando

Gráfico 99 – Estado emocional do desenvolvedor Fernando

Gráfico 100 – Carga de trabalho do desenvolvedor Fernando

Fernando – variáveis por Erro

Gráfico 101 – Identificador x Motivo - desenvolvedor Fernando

Gráfico 102 – Fase em que o erro foi cometido - desenvolvedor Fernando

Gráfico 103 – Tempo de existência do erro - desenvolvedor Fernando

Gráfico 104 – Tipos de erro - desenvolvedor Fernando

Gráfico 105 – Causas de erro - desenvolvedor Fernando

Gráfico 106 – Características de Qualidade afetadas pelos erros - desenvolvedor Fernando

Rita

No período de 26 de julho a 05 de agosto muitos fatores mostram-se atrelados. A desmotivação total, somados a algumas mudanças no rumo do projeto por conta do usuário, implementação pesada, problemas profissionais e carga de trabalho excessiva tiveram forte impacto para a determinação do estado emocional negativo do período. Esta desenvolvedora encontra-se totalmente desmotivada com a empresa.

A maior parte dos erros relatados foram identificados pela própria e associados às fases de implementação e testes. Foram cometidos no próprio dia da identificação. A maioria dos erros foi considerada como erros de implementação errada e incompleta e a falta de documentação e a falta de tempo foram a causa da ocorrência dos mesmos. Novamente o maior impacto ficou na funcionalidade do sistema, mas teve impacto também na usabilidade.

Gráfico 107– Motivação da desenvolvedora Rita

Gráfico 108 – Dificuldades da desenvolvedora Rita

Gráfico 109 – Fases de trabalho da desenvolvedora Rita

Gráfico 110 – Papéis e responsabilidades da desenvolvedora Rita

Gráfico 111 – Problemas da desenvolvedora Rita

Gráfico 112 – Estado emocional da desenvolvedora Rita

Gráfico 113– Carga de trabalho da desenvolvedora Rita

Gráfico 114 – Identificador x Motivo - desenvolvedora Rita

Gráfico 115 – Fase em que o erro foi cometido - desenvolvedora Rita

Gráfico 116– Tempo de existência do erro - desenvolvedora Rita

Gráfico 117– Tipos de erro - desenvolvedora Rita

Gráfico 118 – Causas de erro - desenvolvedora Rita

Gráfico 119 – Características de Qualidade afetadas pelos erros- desenvolvedora Rita

Antônia

A desenvolvedora em questão teve um pico de desmotivação atrelado a outros fatores como problemas profissionais e carga de trabalho excessiva. Este também foi o período em que ela relatou erros. Surpreendentemente, mesmo neste pico o estado emocional esteve muito alto e ela não reportou dificuldades.

Os erros relatados da fase de implementação foram identificados no próprio dia em que foram feitos. Estes foram erros de implementação e provenientes de interrupções. Para ela, a funcionalidade e a usabilidade do sistema foram os itens mais prejudicados pelos erros.

Gráfico 120– Motivação da desenvolvedora Antônia

Gráfico 121– Dificuldades da desenvolvedora Antônia

Gráfico 122– Fases de Trabalho da desenvolvedora Antônia

Gráfico 123– Papéis e Responsabilidades da desenvolvedora Antônia

Gráfico 124– Problemas da desenvolvedora Antônia

Gráfico 125 – Estado emocional da desenvolvedora Antônia

Gráfico 126– Carga de trabalho da desenvolvedora Antônia

Gráfico 127 – Identificador x Motivo - desenvolvedora Antônia

Gráfico 128 – Fase em que o erro foi cometido - desenvolvedora Antônio

Gráfico 129 – Tempo de existência do erro da desenvolvedora Antônio

Gráfico 130 – Tipos de erro - desenvolvedora Antônio

Gráfico 131 – Causas de erro - desenvolvedora Antônio

Gráfico 132 – Características de Qualidade afetadas pelos erros - desenvolvedora Antônio

Beatriz

Desenvolvedora com satisfação parcial de sua situação na empresa, mostrou estabilidade em relação ao projeto como carga de trabalho e motivação. Ela apresentou muitos problemas em toda a fase de realização da pesquisa, o que se afirma na oscilação do estado emocional da mesma. A dificuldade que ela encontra para desempenhar sua função no projeto parece ser um agravante desta instabilidade.

Dos erros relatados por ela a maioria veio de reclamações do usuário e na latência de uma semana. Os erros se mostraram provenientes de implementação

errada e de integração e tem como principal causa a falta de comunicação. A eficiência foi a grande prejudicada, segundo ela.

Gráfico 133 – Motivação da desenvolvedora Beatriz

Gráfico 134 – Dificuldades da desenvolvedora Beatriz

Gráfico 135 – Fases de trabalho da desenvolvedora Beatriz

Gráfico 136 – Papéis e responsabilidades da desenvolvedora Beatriz

Gráfico 137 – Problemas da desenvolvedora Beatriz

Gráfico 138 – Estado emocional da desenvolvedora Beatriz

Gráfico 139 – Carga de trabalho da desenvolvedora Beatriz

Beatriz – variáveis por Erro

Gráfico 140 – Identificador x Motivo - desenvolvedora Beatriz

Gráfico 141 – Fase em que o erro foi cometido - desenvolvedora Beatriz

Gráfico 142 – Tempo de existência do erro - desenvolvedora Beatriz

Gráfico 143 – Tipos de erro - desenvolvedora Beatriz

Gráfico 144 – Causas de erro - desenvolvedora Beatriz

Gráfico 145 – Características de Qualidade afetadas pelos erros - desenvolvedora Beatriz

César

Funcionário insatisfeito. Ausentou-se da participação da pesquisa por um período de uma semana, relatou pouco do seu dia e poucos erros.

O desconhecimento técnico apresentado como dificuldade, está associado à fase de especificação de sistema na qual ele se mostrou envolvido. Detectou-se a ausência de relato de problemas bem como o estado emocional positivo, porém baixo, ou seja, isto significa que ele identificou sempre uma das cinco possibilidades que caracterizam estado emocional positivo, restringindo-se sempre a apenas uma opção. A carga de trabalho excessiva reafirma a insatisfação.

Os poucos erros identificados foram cometidos no próprio dia e atrelados a interrupções constantes.

Gráfico 146 – Motivação do desenvolvedor César

Gráfico 147 – Dificuldades do desenvolvedor César

Gráfico 148 – Fases de trabalho do desenvolvedor César

Gráfico 149 – Papéis e responsabilidades do desenvolvedor César

Gráfico 150 – Problemas do desenvolvedor César

Gráfico 151 – Estado emocional do desenvolvedor César

Gráfico 152 – Carga de trabalho do desenvolvedor César

César – variáveis por Erro

Gráfico 153 – Identificador x Motivo - desenvolvedor César

Gráfico 154 – Fase em que o erro foi cometido - desenvolvedor César

Gráfico 155 – Tempo de existência do erro - desenvolvedor César

Gráfico 156 – Tipos de erro - desenvolvedor César

Gráfico 157 – Causas de erro - desenvolvedor César

Gráfico 158 – Características de Qualidade afetadas pelos erros - desenvolvedor César

6.6 Análise Conclusiva

Na busca pela resposta de quais os fatores humanos que influenciam a qualidade de software, percebe-se que a comunicação, o controle sobre os processos, a satisfação, a liderança, problemas profissionais, carga de trabalho e motivação são os fatores em que se percebe relação com os erros no desenvolvimento de software para esta equipe, nesta empresa.

Alguns fatores não mostraram relação evidente nos dados coletados, são eles: relacionamento, papéis e responsabilidades e reconhecimento. O fato de não se relacionar à alteração dos demais fatores na presença de erros não os invalida como fatores condicionantes de desempenho. Outros fatores não se podem afirmar nem sua relação nem ausência de sua influência, sendo eles: problemas pessoais, financeiros e de saúde; estado emocional e dificuldades no desempenho de tarefas.

Estas conclusões são qualitativas e baseadas nos dados e gráficos apresentados neste capítulo, não possuindo uma análise estatística. O instrumento de coleta de dados funciona e pode ser aplicado continuamente, e, a partir da análise dos dados obtidos neste instrumento, percebe-se que deslizos e lapsos mostraram evidências em relação a problemas (financeiros, profissionais e pessoais) e motivação. Já os enganos mostraram evidências diretas com o controle sobre os processos, liderança, políticas de RH, comunicação e carga de trabalho. Na pesquisa realizada, a possibilidade de se tratar desenvolvedores engajados em projetos distintos e em etapas distintas do desenvolvimento trouxeram riqueza para este trabalho. De desenvolvedores muito insatisfeitos, com motivação baixa a desenvolvedores com alto estado emocional e motivação elevada, alguns fatores pertinentes mostraram-se com maior evidência. Alguns são discutidos a seguir.

A queda na motivação associada à insatisfação tem reflexo nos problemas profissionais destacados e nos erros cometidos. Esta conclusão é possível já que isto é visível nos relatos de erros que foram cometidos pelo próprio desenvolvedor no dia do relato. Esta análise ficou restrita aos relatos não sendo feita ao longo do tempo como se queria. Estas observações podem ser feitas nos gráficos 107 a 110, 112 e 113 pertencentes a uma desenvolvedora e 120 a 123, 125 e 126 de outra desenvolvedora. Em ambos os casos estas relações se evidenciam.

A maioria dos erros relatados é proveniente de implementação, quer sejam erradas ou incompletas.

Fatores como a insatisfação e carga de trabalho mostram-se ligadas a problemas profissionais e mostram-se causadores de implementações errôneas. Esta evidência pode ser vista nos gráficos 108 e 112 e, também, 125 e 126.

É comum ao ser humano escutar aquilo que lhe é conveniente, de acordo com suas próprias experiências, paradigmas e pré-julgamentos e nisto evidenciam-se dois fatores de impacto direto sobre todo o ciclo de desenvolvimento de um software: comunicação e documentação. A comunicação é a fonte e a base para qualquer desenvolvimento e perpetua-se na vida do desenvolvedor de software como conhecimento e relevância de fatores incorporados no sistema a ser desenvolvido. A documentação é o fruto de uma comunicação, e, além de existir deve guiar os passos do desenvolvedor. Fato é que as pessoas são diferentes, e, portanto têm visões e ações diferentes sobre um mesmo fato, ou assunto.

Erros de integração e implementação (errada e incompleta) estão, em sua maioria, presentes com a latência de um dia a uma semana no máximo. Na maioria dos casos os erros de integração foram atribuídos a omissão de requisitos relevantes, e os erros de implementação têm como maior causa falta de documentação. Evidências podem ser vistas nos gráficos de 51 a 53 e 64 a 66.

O cruzamento dos fatores individuais associados ao despreparo da liderança e a insatisfação da vida profissional no geral são fatores presentes no relato dos indivíduos. Isto pode ser notado pela grande incidência de repostas de despreparo da liderança no gráfico 32 e em casos especiais de desenvolvedores como Elisa (gráficos de 55 a 67) e Rita (gráficos de 107 a 114)

7 CONCLUSÕES

Com o estudo dos modelos e padrões existentes que tratam da qualidade de software fica claro que estes são estabelecidos com foco em processos e na sua aplicação. Pessoas são recursos necessários para colocá-los em prática, mas não compete a estes modelos a determinação de fatores humanos capazes de implicar na qualidade do desenvolvimento de software.

O ser humano é imprevisível, é frágil, vulnerável, contraditório e ambíguo. Sabe-se que as emoções humanas influenciam diretamente nos resultados de uma empresa. Por este motivo empresas produtoras de software devem se preocupar com a qualidade das pessoas que para ela trabalham, e também nas condições em que trabalham para que assim tenham maior controle sobre a qualidade daquilo que se produz.

O ser humano passível de erro, falível, influencia na Qualidade de Software através da sua capacidade de resolução de problemas e síntese de soluções. Nisto influenciam os mecanismos organizacionais aos quais ele está exposto e as circunstâncias pessoais vividas. Mas os modelos de Qualidade de software vêem o ser humano somente do ponto de vista de treinamento.

Os defeitos introduzidos pelas pessoas no processo de desenvolvimento de software podem ser explicados pelas teorias do Erro Humano, pois o desenvolvedor de software sofre influências de fatores pessoais e organizacionais como qualquer outro ser humano e os enganos, deslizes e lapsos fazem parte do processo de desenvolvimento de software. Como forma de identificar e prever defeitos de software, deve-se levar em conta os 3 maiores elementos que influenciam a produção de software, que são a natureza da tarefa e as circunstâncias do ambiente, o mecanismo que gerencia o desempenho e a natureza do indivíduo, ou seja, os mecanismos cognitivos envolvidos na produção do defeito.

A preocupação das empresas produtoras deve girar em torno do momento em que o erro acontece, quer seja o momento do desenvolvimento do software ou a fase

de manutenção, porque este é o ponto onde uma cadeia muitas vezes pouco visível de erros pode se instaurar.

Práticas individuais e coletivas associadas ao que diz respeito às características individuais profissionais, se associadas e controladas de forma conjunta, podem levar ao controle mais eficiente e efetivo da qualidade de software.

Da análise dos modelos de qualidade existentes e dos trabalhos já enumerados em diversas áreas que pretendem quantificar as relações do erro humano ao processo de desenvolvimento produtivo, percebe-se que a determinação e o controle de quais são os fatores humanos que realmente causam erros é complexa.

Como forma de continuar este trabalho visa-se a expansão do mecanismo para outras empresas, o que terá grande valia, já que a análise de outros ambientes e características organizacionais podem enriquecer a coleta de dados.

Posteriormente, deve-se fazer o aperfeiçoamento do mecanismo para que ele permita identificar o desenvolvedor com melhores condições emocionais para lidar com determinados projetos e situações, evitando desta forma que o ser humano introduza defeitos no software que está produzindo.

Pelo exposto, os fatores discutidos neste trabalho, tanto organizacionais quanto individuais, mostram-se determinantes para o erro humano e para qualidade de software e precisam ser analisados para a melhoria dos processos de desenvolvimento de software. Isto implica diretamente na relação custo/benefício e se reflete na produtividade individual de cada membro componente de uma equipe desenvolvedora de software.

Fica a lição de que nesta era da informação, a evolução da ciência e da tecnologia está fazendo com que o homem olhe sempre para fora de si e não perceba que nele residem as respostas da verdadeira evolução.

8 ANEXOS

ANEXO I – Questionário e Resultados da Etapa Preliminar

Este questionário foi feito com o objetivo de identificar se os fatores propostos para estudo eram válidos, bem como erros na pesquisa. Além disto, a análise deste questionário identificou a necessidade de se tratar alguns fatores sob enfoque diferenciado para que as conclusões desejadas fossem obtidas.

Questionário da Etapa Preliminar

O questionário desenvolvido, apresentado nas figuras abaixo, foi formalizado através de uma carta de apresentação cujo intuito era posicionar o desenvolvedor com relação aos objetivos que se procurava atingir, e como aquilo estaria inserido no cenário acadêmico. Também procurou-se elucidar a importância da veracidade das informações e a garantia da confidencialidade das informações. Nesta carta de apresentação procurou também identificar as partes componentes do questionário.

Como forma de tornar a análise mais fácil e consistente, o questionário foi dividido em duas partes: contextualização e identificação de erros.

A *Contextualização* pretendeu dar uma idéia mais profunda com relação à complexidade do projeto conforme percebida pelo desenvolvedor. Em função do conhecimento da pesquisadora do contexto da empresa, a análise desta contextualização leva a conclusões tais como: como é a liderança do projeto e o grau de importância do projeto na empresa. Além disto, saber se a liderança releva características individuais no gerenciamento do processo de desenvolvimento de software que podem ser encaradas como boas práticas.

A *Identificação dos Erros* pretendeu situar os tipos de erros identificados pelos desenvolvedores, bem como suas possíveis causas e ações a serem tomadas mediante seu acontecimento, assim como a identificação da caracterização do estado emocional do desenvolvedor .

O público alvo que se procurou atingir foi o de desenvolvedores de software, quer atuem ou não dentro de padrões de desenvolvimento de software e/ou modelos de garantia de qualidade de software.

O anonimato pretendeu garantir aos desenvolvedores que nenhum tipo de informação poderia ser utilizada pela empresa em questão, e que, de nenhuma forma, isto seria uma avaliação individual de desempenho no trabalho sendo executado.

O questionário é apresentado na íntegra a seguir:

Contextualização

1. Nome do Projeto: _____
2. Quantas áreas estão envolvidas no projeto?

☒ Somente a minha
☐ A minha e mais uma
☐ Mais de duas
3. Qual a linguagem de programação que você utiliza? _____
4. Fase atual do projeto

☒ Especificação
☐ Implementação
☐ Teste

☒ Prototipação
☐ Implantação
☐ Manutenção
5. Existe documentação formal com relação ao que deve ser implementado?

☒ Sim
☐ Não
6. Os conceitos exigidos pelo projeto são claros para você?

☒ Sim
☐ Não
7. Você faz apenas o que lhe é solicitado?

☒ Sempre
☐ Apenas às vezes
☐ Nunca, sempre faço mais
8. Como você começa a implementação?

☒ Idéia de alguém
☐ Rotina parecida

☒ Idéia própria
☐ Leituras a respeito do assunto
9. Depois de terminar sua implementação, você desconfia dos pontos que podem "dar problemas"?

☒ Sim
☐ Não

Se Sim, por quê? _____
10. Considere o papel que você tem no projeto e classifique:

a. O grau de dificuldade de suas atribuições

☒ Baixa
☐ Média
☐ Alta

b. A severidade da(s) tarefa(s) que você está desempenhando

☒ Baixa
☐ Média
☐ Alta

c. As condições operacionais que você tem para desempenhar a tarefa

☒ Regular
☐ Bom
☐ Ótimo
☐ Excelente

Fig. I – Folha de Contextualização do questionário

Identificação de Erros

ATENÇÃO: Preencher este trecho a cada erro detectado

Data ____/____/2002 Horário da identificação: _____

A. Como você está hoje? – Assinale quantas quiser...

- ☒ Bem
 ☐ Alegre
 ☐ Confiante
 ☐ Doente
☒ Mal
 ☐ Triste
 ☐ Nervoso(a)

B. Como se sente dentro do projeto? ☒ Motivado ☐ Desmotivado

C. Como tem sido a colaboração dos colegas?

- ☒ Ruim
 ☐ Boa
 ☐ Ótima
 ☐ Excelente

D. Como se sente com relação a carga de trabalho?

- ☒ Sobrecarregado
 ☐ Tranquilo

E. Quem identificou o erro?

- ☒ Eu
 ☐ Um colega
 ☐ Meu chefe
 ☐ Usuário

F. Qual a gravidade do erro encontrado? ☒ Baixa ☐ Média ☐ Alta

G. Qual o tipo de erro encontrado?

- ☒ Esquecimento de alguma tarefa que deveria ter sido feita
☒ Falta de equipamento/ambiente para testar
☒ Programação excessiva para o mesmo trecho
☒ Raciocínio interrompido por algum motivo
☒ Implementação de código desnecessário para o escopo do sistema
☒ Má definição das regras a serem implementadas
☒ Integração com outros sistemas
 ☐ Erro de digitação
☒ Outro _____

H. Descreva o erro: _____

I. Qual a sua ação sobre este erro?

- ☒ Correção imediata
☒ Necessidade de rever conceitos de projeto
☒ Necessidade de ajuda técnica (pessoas ou bibliográfica)

Fig. II – Folha de Identificação dos Erros do questionário

Piloto do questionário

Como forma de validação do questionário elaborado, foi aplicado um piloto num grupo de 11 pessoas. Este grupo foi composto por 11 analistas de sistemas desenvolvedores de software da Itautec Philco S/A, sendo que eles pertencem as áreas de desenvolvimento de aplicações bancárias e desenvolvimento de soluções e-Business. Sete destes analistas atuam com linguagem Visual C++ e se dedicam à área de segurança, e os outros quatro atuam com linguagem Java e HTML e se dedicam à área de B2B.

O objetivo deste piloto era a validação e consistência das questões para o grupo analisado, bem como a identificação de eventuais discordâncias e dúvidas causadas pelas perguntas e possíveis respostas.

O resultado obtido com este piloto foi o questionário analisado e melhorado, pronto para ser aplicado em larga escala.

Divulgação

A divulgação do questionário utilizado foi feita na Itautec Philco S/A através das diretorias que cuidam das áreas de automação bancária e Internet. Com a ajuda e apoio dos diretores destas áreas, o questionário foi divulgado na empresa e atingiu as áreas de desenvolvimento de software, passando pelos gerentes e sendo centralizadas pela pesquisadora.

Como forma de oficializar a utilização destes dados como fonte de pesquisa, o diretor da área de Automação Bancária consentiu, em nome da empresa, na divulgação dos dados da empresa mediante carta apresentada no Anexo A.

Coleta

A coleta de dados foi feita pelas áreas e repassado à pesquisadora. Este trabalho foi realizado ao longo de duas semanas e mostrou um resultado de 24 questionários respondidos.

Como o questionário foi distribuído num arquivo em formato de documento Word, eles retornaram um a um para análise, sendo assim, todos foram contabilizados individualmente.

Análise

Como forma de análise destes questionários foi montado um banco de dados onde as variáveis pertinentes foram levantadas e foram cruzadas para atingir o objetivo do experimento.

A análise individual de algumas perguntas trará benefícios e índices a serem contabilizados, outros fatores relevantes dependem do cruzamento de várias informações.

A seguir, tem-se a apresentação do modelo do banco de dados utilizado:

Fig. III – Modelo do Banco de Dados montado para armazenar as informações relativas à Pesquisa de Campo

Obtenção dos Resultados

Os resultados serão apresentados de forma qualitativa das dimensões do ser humano envolvidas na produção de software. Para isso, elaborou-se uma série de consultas cruzadas a este banco de dados, que permitiram a coleta dos dados parciais apresentados no próximo capítulo.

No capítulo apresentado a seguir, o detalhamento das questões será feito já com a apresentação de resultados obtidos.

Resultados da Etapa Preliminar

Os resultados obtidos até o presente momento serão apresentados juntamente com a explicação dos objetivos de cada questão formulada.

Apresentação da Amostra

Os totais obtidos para esta versão do questionário são apresentados na tabela a seguir:

Total analisado	
Questionários	24
Projetos	13
Erros	41

Tabela I – Totais analisados

Detalhamento das Questões e Resultados

A seguir, será apresentado o detalhamento das questões, bem como seus resultados.

Contextualização do Projeto

Questão 1-) Nome do Projeto. Classificou-se cada projeto em relação ao critério complexidade, levando em consideração ritmo e pressão de prazos, e qualificando-o em alta, média e baixa. Para isto, a pesquisadora contou com a análise subjetiva dos gerentes de cada projeto, bem como sua própria análise subjetiva de acordo com seu conhecimento do contexto onde estão inseridos estes projetos na empresa.

Critério Adotado	
Ritmo / Pressão	Complexidade
Alto / Alto	Alta
Alto / Baixa	Média
Baixo / Baixa	Baixa

Tabela II – Critério de Complexidade adotada por Projeto

Como forma de auxiliar o critério adotado, foi levado em consideração o tipo de linguagem de desenvolvimento com a qual o desenvolvedor tem contato e os prazos estabelecidos por cada gerente para a execução das tarefas individuais.

Complexidade do projeto	
Alta	2
Média	7
Baixa	4

Tabela III – Complexidade do Projeto

Questão 2-) Classificou-se o projeto em relação a quantidade de áreas envolvidas, com o objetivo de relacionar estes projetos com erros devidos a problemas de integração.

Áreas envolvidas	
Uma	6
Duas	2
Mais de duas	5

Tabela IV – Quantidade de áreas envolvidas por projeto

Questão 3-) Classificou-se cada projeto em relação à complexidade exigida do programador de acordo com a linguagem de programação assinalada por ele.

Compl. Linguagem	
Alta	2
Média	7
Baixa	4

Tabela V – Complexidade inerente ao projeto pela linguagem utilizada

Questão 4-) Classificou-se a fase em que os projetos estão.

Fase do Projeto	
Implementação	6
Manutenção	6
Testes	1

Tabela VI – Fases dos Projetos analisados

Questão 5-) Classificou-se a indicação da existência ou não da especificação de requisitos de software do projeto.

Por uma questão de terminologia, foi perguntado ao desenvolvedor se existia a documentação formal para o projeto, que para eles, corresponde à especificação de requisitos de software.

Documentação	
SIM	7
NÃO	6

Tabela VII – Documentação dos projetos

Questão 6-) Classificou o projeto em relação ao comportamento do gerente no envolvimento da área desenvolvedora com os requisitos do projeto, sua prioridade e contexto.

Conceitos do projeto: para todos os projetos enumerados, os desenvolvedores disseram conhecer ao menos os conceitos básicos necessários.

Contextualização do Indivíduo

Questão 7-) Classificou-se o indivíduo segundo o estado pessoal de comprometimento no projeto. Esta análise, ainda que subjetiva, baseou-se nas respostas de acordo com o que o desenvolvedor realiza, se faz o que lhe é designado ou se procura melhores saídas para suas obrigações mesmo que isto signifique realizar mais tarefas. Adotou-se os seguintes critérios: baixo, médio e alto.

Critério Adotado	
Realização de tarefas	Comprometimento
Somente o solicitado	Baixo
Pouco mais que o solicitado	Médio
Sempre mais	Alto

Tabela VIII – Critério adotado para determinar o comprometimento individual

Comprometimento no Projeto	
Pouco	7
Médio	3
Alto	3

Tabela IX – Comprometimento Demonstrado no Projeto

Questão 8-) Classificou-se o indivíduo segundo a indicação de desempenho. Esta análise mostra-se subjetiva, mas baseada nas respostas de iniciativas de implementação do indivíduo. Adotou-se os seguintes critérios: ruim, satisfatório e bom.

Critério Adotado	
Iniciativas de Implementação	Desempenho
Idéias de alguém / Rotinas parecidas	Ruim
Leituras a respeito do assunto	Satisfatório
Idéias próprias	Bom

Tabela X – Critério adotado para determinar o desempenho individual

Desempenho	
Ruim	1
Satisfatório	7
Bom	5

Tabela XI– Desempenho Demonstrado no Projeto

Questão 9-) Classificou-se a existência ou não de técnicas de prevenção de erros. Entende-se por técnicas de prevenção a execução de testes, previamente estabelecidos, de verificação de processos de software desenvolvidos.

Prevenção	
SIM	9
NÃO	4

Tabela XII – Prevenção a Erros

Questão 10-) Classificou-se como o desenvolvedor se vê dentro do projeto. Para isto foram feitas 3 sub-questões nesta questão que buscaram compor a motivação que o desenvolvedor vê de acordo com suas ações no projeto. Estas sub-questões tiveram o foco no grau de dificuldade das tarefas exercidas, severidade e condições para desempenhar a tarefa. Foram adotados os seguintes critérios: regular, bom, ótimo e excelente.

Critério Adotado	
Papel desempenhado no projeto	Motivação do Projeto
Tarefa de baixa dificuldade / Baixa severidade / Poucas condições operacionais	Ruim
Tarefa de média dificuldade / Média severidade / Boas condições operacionais	Bom
Tarefa de alta dificuldade / Alta severidade / Boas condições operacionais	Ótimo
Tarefa de alta dificuldade / Alta severidade / Excelentes condições operacionais	Excelente

Tabela XIII– Critério adotado para determinar a motivação do projeto corrente

Motivação do Projeto	
Ruim	0
Bom	7
Ótima	4
Excelente	2

Tabela XIV– Motivação do projeto corrente

Identificação dos Erros

Serão apresentados os dados gerais relacionados aos erros encontrados.

Classificou-se quantos erros ocorrem por período do dia.

Erros por período do dia	
manhã	24
tarde	17

Tabela XV – Período do dia em que os erros foram detectados

Questão A-) Caracterizou-se o estado emocional do desenvolvedor no dia, segundo determinado pelo próprio desenvolvedor. As opções encontradas pelo desenvolvedor eram: bem, mal, alegre, triste, confiante, nervoso, doente, podendo haver mais de uma opção assinalada. Para que possamos ter uma visualização efetiva, foi adotado o seguinte critério: positivo, negativo e conflitante.

Proj. / Estado Emocional	Positivo	Negativo	Conflitante
Projeto 1		1	
Projeto 2	11		
Projeto 3		1	
Projeto 4		1	
Projeto 5	3		
Projeto 6	1		
Projeto 7	3		
Projeto 8	1	2	
Projeto 9	2		
Projeto 10	2		
Projeto 11	1		
Projeto 12	6		
Projeto 13			6

Tabela XVI – Estados emocionais, por projeto, apresentados no momento da identificação do erro

Gráfico I – Estado Emocional do desenvolvedor no dia da identificação do Erro

Questão B-) Caracterizou-se a situação emocional com relação à motivação, segundo o critério: motivado ou desmotivado.

Motivação Pessoal	
Motivado	34
Desmotivado	7

Tabela XVII – Motivação Pessoal no momento da identificação do erro

Gráfico II – Motivação pessoal no momento da identificação do erro

Questão C-) Identificou-se as relações interpessoais. O critério adotado foi ruim, bom, ótimo e excelente.

Colaboração entre áreas	
Boa	13
Ótima	10
Excelente	18

Tabela XVIII – Colaboração entre as áreas participantes de um mesmo projeto

Questão D-) Identificou-se a motivação com relação à carga de trabalho, segundo o critério: boa ou excessiva.

Carga de Trabalho	
Satisfatória	22
Excessiva	19

Tabela XIX – Carga de trabalho

Gráfico III – Carga de trabalho do ponto de vista do desenvolvedor

Questão E-) Identificou-se quem foi o responsável pela identificação do erro. O critério adotado foi: o próprio, gerência e usuário.

Identificação do Erro	
Desenvolvedor	23
Colega	9
Usuário	9

Tabela XX – Agente identificador do erro

Questão F-) Identificou-se o grau de severidade do erro encontrado, segundo o critério: baixa, média ou alta. O próprio desenvolvedor estabeleceu este critério.

Severidade	
Alta	9
Média	13
Baixa	19

Tabela XXI – Grau de severidade dos erros encontrados

Questões G e H-) Contabilizou-se o tipo de erro assinalado pelo desenvolvedor. O critério adotado foi a caracterização do erro em grupos: omissão, redundância, digitação, regras irrelevantes, integração com outros sistemas, falta de estrutura, interrupções, regras erradas.

Tipo de Erro	
Omissão	4
Redundância	4
Digitação	1
Regras Irrelevantes	1
Integração	9
Falta de Estrutura	4
Interrupção	1
Regras Erradas	17

Tabela XXII – Tipos de Erros Analisados

Gráfico IV – Tipos de Erro encontrados

Questão I-) Indicou-se o desenvolvimento técnico pessoal do desenvolvedor. Critérios: apto, com pouco envolvimento na concepção do projeto e necessidade de treinamento, foram analisadas as respostas dos indivíduos sobre sua ação possível no erro identificado.

Critério Adotado	
Ação sobre o erro	Desenvolvimento Pessoal
Correção imediata	Apto
Necessidade de revisar conceitos de projeto	Pouco envolvido
Necessidade de ajuda técnica	Necessidade de treinamento

Tabela XXIII – Critério adotado para determinar o desenvolvimento pessoal

Desenvolvimento Pessoal	
Apto	28
Pouco envolvido	4
Necessidade de treinamento	9

Tabela XXIV – Desenvolvimento Pessoal

Cruzamento de Resultados

A caracterização dos Projetos, Indivíduos e Erros será mostrada individualmente em quadros, de forma a evidenciar a relação dos dados obtidos com esta pesquisa nos quadros de correlação.

Projeto	Complexidade	Núm. Áreas	Fase	Documentação	Equipe	Erros Identificados
Gerenciador de Comportamento	Baixa	1	Implementação	NAO	3	1
lhcScript 2.0/lhcScript - 3.0	Média	1	Manutenção	NAO	20	11
Cheques sem Provisão	Média	1	Manutenção	SIM	5	1
Plataforma de Negócios	Média	3	Manutenção	SIM	3	1
Plataforma de Negócios - Banco ABN/Real	Média	3	Manutenção	SIM	20	3
e-Tracking	Média	3	Teste	NAO	4	1
EProcurement	Média	2	Manutenção	SIM	8	3
Duraflor	Média	2	Implementação	SIM	3	3
DuraflorNet	Média	1	Manutenção	NAO	2	2
Legacy Navigator	Média	1	Implementação	NAO	15	2
Terminal caixa em servidor WEB	Média	3	Implementação	SIM	8	1
Vistoria em PDA	Baixa	1	Implementação	SIM	3	6
InfoMovie	Média	3	Implementação	NAO	2	6

Tabela XXV – Mapa de caracterização dos Projetos

Indivíduo	Projeto	Comprometimento	Desempenho	Motivação	Aptidão	Núm. Erros
Edgard	Gerenciador de Comportamento	Médio	Satisfatório	Bom	Apto	1
Newton	lhcScript 2.0/lhcScript - 3.0	Médio	Satisfatório	Bom	Apto	11
Tárcio	Cheques sem Provisão	Baixo	Satisfatório	Ótimo	Necessidade de treinamento	1
Andréia	Plataforma de Negócios / Plataforma de Negócios - Banco ABN/Real	Baixo	Satisfatório	Bom	Necessidade de treinamento	4
Marcos	e-Tracking	Baixo	Satisfatório	Excelente	Apto	
Roberto	EProcurement	Baixo	Bom	Ótimo	Apto	1
Alan	Duraflor	Médio	Bom	Ótimo	Apto	3
Alan	DuraflorNet	Baixo	Ruim	Bom	Necessidade de treinamento	3
Carmen	Legacy Navigator	Baixo	Bom	Bom	Apto	2
Solange	Terminal caixa em servidor WEB	Alto	Bom	Bom	Pouco envolvido	2
Tatiana	Vistoria em PDA	Alto	Satisfatório	Ótimo	Apto	3
Tatiana	Vistoria em PDA	Alto	Satisfatório	Ótimo	Pouco envolvido	1
Tatiana	Vistoria em PDA	Alto	Satisfatório	Ótimo	Necessidade de treinamento	2
Marcelo	InfoMovie	Alto	Bom	Excelente	Apto	4
Marcelo	InfoMovie	Alto	Bom	Excelente	Pouco envolvido	1
Marcelo	InfoMovie	Alto	Bom	Excelente	Necessidade de treinamento	1

Tabela XXVI – Mapa de caracterização dos indivíduos

Tipo	Distribuição														
	Hora		Carga Trab.		Relacionamento				Identificador				Severidade		
	Manhã	Tarde	Boa	Excessiva	Ruim	Bom	Ótimo	Excelente	Desenv.	Colega	Usuário	Gerente	Baixa	Média	Alta
Omissão	2	2	4	0	0	2	1	1	2	1	1	0	3	0	1
Redundância	2	2	4	0	0	1	3	0	2	0	2	0	2	0	2
Digitação	1	0	0	1	0	0	0	1	0	1	0	0	1	0	0
Regra Irrelevante	1	0	1	0	0	0	1	0	1	0	0	0	1	0	0
Integração	7	2	6	3	0	4	1	4	7	0	2	0	5	2	2
Falta de Estrutura	2	2	3	1	0	0	3	1	3	0	1	0	1	3	0
Interrupção	0	1	0	1	0	0	0	1	0	0	1	0	0	1	0
Regra Errada	10	7	4	13	0	5	2	10	8	7	2	0	6	7	4

Tabela XXVII – Mapa de caracterização dos erros

Tipo	Fatores do Indivíduo											
	Desempenho			Motivação		Aptidão de Resolução			Estado Emocional			
	Ruim	Satisf.	Bom	Motivado	Desm.	Apto	Poc. Env.	Treina.	Pos.	Neg.	Conflitante	
Omissão	0	3	1	3	1	3	0	1	3	1		0
Redundância	0	1	3	4	0	3	0	1	3	1		0
Digitação	0	1	0	1	0	1	0	0	1	0		0
Regra Irrelevante	0	1	0	1	0	0	1	0	1	0		0
Integração	1	3	5	6	3	4	1	4	4	1		4
Falta de Estrutura	0	0	4	4	0	2	1	1	2	1		1
Interrupção	0	1	0	1	0	1	0	0	1	0		0
Regra Errada	1	14	2	14	3	14	1	2	15	1		1

Tabela XXVIII – Mapa de relação dos erros com os indivíduos

Tipo	Fatores do Projeto											
	Complexidade			Áreas			Fase					
	Baixa	Média	Alta	Uma	Duas	Mais	Esp.	Prot.	Implem.	Implan.	Teste	Manut.
Omissão	2	2	0	2	1	1	0	0	2	0	0	2
Redundância	1	3	0	1	3	0	0	0	2	0	0	2
Digitação	0	1	0	1	0	0	0	0	0	0	0	1
Regra Irrelevante	1	0	0	1	0	0	0	0	1	0	0	0
Integração	2	7	0	4	0	5	0	0	7	0	0	2
Falta de Estrutura	0	4	0	1	2	1	0	0	4	0	0	0
Interrupção	0	1	0	0	0	1	0	0	0	0	1	0
Regra Errada	1	16	0	13	0	4	0	0	3	0	0	14

Tabela XXIX – Mapa de relação dos erros com os projetos

ANEXO II - Declaração de autorização dos dados da pesquisa coletada

São Paulo, 13 de maio de 2003

DECLARAÇÃO

Declaro para fins de publicação da Dissertação de Mestrado de KAREN SANDHOF que os dados obtidos em pesquisa realizada na ITAUTEC PHILCO S/A são verídicos e autorizo sua publicação.

José Carlos Lopes de Almeida
Diretor de Automação Bancária

Func. 251124
DAB 03-122

9 REFERÊNCIAS BIBLIOGRÁFICAS

ABNT – Associação Brasileira de Normas e Técnicas. **NBR ISO 9000-3 Parte 3: Diretrizes para a aplicação da NBR 19001 ao desenvolvimento, fornecimento e manutenção de “software”**. São Paulo, 1993

ARMITAGE, J. et al. **A System Engineering Capability Maturity Model – version 1.1**, Pittsburgh, Software Engineering Institute of Carnegie Mellon University, Maturity Model CMU/SEI-95-MM-01, 1995

BARTOL, K.M. **Turnover Among DP Personnel: a Causal Analysis**. Communications of the ACM, v.26, n.10, p. 807-11, out. 1983

BERGAMINI, C. W. **Avaliação de Desempenho Humano na Empresa**. 3ª edição, Ed. Atlas, São Paulo, 1986

BOCK, Ana M.B.; FURTADO, O.; TEIXEIRA, Maria L. **Psicologias**. 8ª edição, São Paulo, editora Saraiva, 1995

BRADY, J.T. **A Theory of Productivity in the Creative Process**. IEEE Computer Graphics & Applications, p. 25-34, mai. 1986

CMMI Product Team, **Capability Maturity Model Integration**, Pittsburgh, Software Engineering Institute of Carnegie Mellon University, Continuous Representation CMU/SEI-2002-TR-011, ESC-TR-2002-011, mar. 2002

COCKBURN, A.; HIGHSMITH, J.; **Agile software development: the people factor**. Computer, v. 34, Issue: 11, nov 2001 p. 131-133

CURTIS, B.; HEFLEY, W.E.; MILLER, S. **Overview of the People Capability Maturity Model**, Pittsburgh, Software Engineering Institute of Carnegie Mellon University, Maturity Model CMU/SEI-95-MM-01, 1995

DEIMEL, B.; MOGILENSKY, J. **People Issues in the Capability Maturity Model (CMM)**, Working Paper, ver. 1.1, 1996 Disponível em www.sei.cmu.edu Acesso em maio de 2003

DeMARCO, T.; LISTER, T. **Peopleware – Como Gerenciar Equipes e Projetos Tornando-os Mais Produtivos**. São Paulo, McGraw-Hill, 1990

FUGIWARA, P. correspondência pessoal, 2002 a 2003

GOLEMAN, D. **Inteligência Emocional**. 82ª edição, Rio de Janeiro, Objetiva, 1995

HUMPHREY, Watts S. **The complete PSP Book – A Discipline for Software Engineering**. Addison-Wesley, 1995

ISO/IEC TR 15504: Software Process Assessment. Disponível em <http://www.sqi.gu.edu.au/spice/> Acesso em junho de 2001

JAVAUX, D. **Human error, safety, and systems development in aviation.** Reliability Engineering and System Safety, v. 75, n. 2, p. 115-19, fev. 2002

JAVAUX, D. **A method for predicting errors when interacting with finite state systems. How implicit learning shapes the user's knowledge of a system.** Reliability Engineering and System Safety, v. 75, n. 2, p. 147-65, fev. 2002

KATZ, D.; KAHN, R.L. **Psicologia Social das Organizações.** 2ª edição, São Paulo, Editora Atlas SA, 1978

KECKLUND, L. J.; SVENSON, O. **Human errors and work performance in a nuclear power plant control room: associations with work-related factors and behavioral coping.** Reliability Engineering and System Safety, v. 56, p. 5-15, 1997

KENDEL, Eric R.; SCHWARTZ, James H.; JESSEL, Thomas M. **Fundamentos da Neurociência e do Comportamento.** Rio de Janeiro, Prentice-Hall do Brasil, 1997

LAGO, A. P. Comunicação – uma perspectiva abrangente. Artigo disponível em www.rh.matrix.com.br/cgi-rh/banco/ Acesso em maio de 2001

LASALA, K. P. **Human Performance Reliability: A Historical Perspective.** IEEE Transactions on Reliability, v. 47, n. 3, set. 1998

NAKAMURA, K.; SUGAWARA, M.; TOYAMA, M.; **Defect prevention activities and tools** Communications, ICC '91, Conference Record. IEEE International Conference on , 1991, v. 1, p. 360-363

NIELSEN, Jakob. **Usability Engineering.** Academic Press, Chestnut Hill, 1993.

PAULK, M.C.; CURTIS, B.; CHRISSIS, M.B.; WEBER, C.V. **Capability Maturity Model for Software – version 1.1**, Pittsburgh, Software Engineering Institute of Carnegie Mellon University, Technical Report CMU/SEI-93-TR- 024, 1993

PIAGET, J. **A Epistemologia Genética.** Coleção os Pensadores, 1ª edição, p.127-190, 1975

PRESSMAN, R. S. **Software Engineering: A Practitioner's Approach.** 5ª edição, McGraw-Hill, 2001

REASON, J. **Human Error,** Nova York, Cambridge University press, 1990

SHARIT, J. and M.; DAVID, M. **Incorporating the Effects of Time Estimation into Human-Reliability Analysis for High- Risk Situations.** IEEE Transactions on Reliability, v. 40, n. 2, jun. 1991

SMIDTS, C.; SONG-HUA SHEN; MOSLEH, A.; **A taxonomy and root-cause analysis of human cognitive behavior based on a cognitive model** Reliability and Maintainability Symposium, Proceedings, Annual , 1995, p. 303-314

SMIDTS, Carol; STUTZKE, Martin; STODDARD, Robert W. **Software Reliability Modeling: An Approach to Early Reliability Prediction.** IEEE Transactions on Reliability, v. 47, n. 3 - parte I, p. 268-78, set. 1998

SMIDTS, C. **A Stochastic Model of Human Errors in Software Development: Impact of Repair Times.** Proceedings of the 10th International Symposium on Software Reliability Engineering, 1999, p. 94–103

STUTZKE, M. A.; SMIDTS, C. S.; **A stochastic model of fault introduction & removal during software development,** IEEE Transactions on Reliability, v. 50 Issue: 2, jun 2001, p. 184-193

VANDERHAEGEN, F. **A non-probabilistic prospective and retrospective human reliability analysis method – application to railway system.** Reliability Engineering and System Safety, v. 71, n. 1, p. 1-13, jan. 2001

10 BIBLIOGRAFIA PESQUISADA

ABRAHAMSSON, P.; **Is management commitment a necessity after all in software process improvement?** Euromicro Conference, 2000. Proceedings of the 26th , v. 2, 2000, p. 246-53

AGARWAL, R. FERRAT, T.W. **Crafting an HR Strategy to Meet the Need for IT Workers.** Communications of the ACM, v.44, n.6, p. 59-64 jul. 2001

ARNOLD, D.; NIEDERMAN, F. **The Global IT Work Force.** Communications of the ACM, v.44, n.6, p. 31-3, jul. 2001

BARON, J.; **The people side of software. A lesson plan for establishing a successful training program** Software Engineering Education, Proceedings., Ninth Conference on , 1996, p. 184-198

BEVAN, N.; AZUMA, M.; **Quality in use: incorporating human factors into the software engineering lifecycle** Software Engineering Standards Symposium and Forum. Emerging International Standards. ISESS 97., Third IEEE International , 1997, p. 169-179

BIFFL, S.; **Analysis of the impact of reading technique and inspector capability on individual inspection performance** Software Engineering Conference, 2000. APSEC 2000. Proceedings. Seventh Asia-Pacific , 2000, p. 136-45

BOEHM, B.W; PENEDO, M.H.; STUCLE, E.D.; WILLIAMS, R.D.; PYSTER, A.B. **A Software Development Enviroment for Improving Productivity.** Computer, v.17, n.6, p.30-42, jun. 1984

BROOKS, Frederick P, Jr. **The Mythical Man-Month: essays on Software Engineering.** 20ª edição, Addison-Wesley, 1995

CROSBY, P.B. **Qualidade é Investimento.** Nova York, José Olympio Editora, 1979

DE JOURS, C. **Uma Nova Visão do Sofrimento Humano nas Organizações, in O Indivíduo na Organização.** v. 1, São Paulo, editora Atlas, p. 150-73, 1993

DeMARCO, T.; LISTER, T. **Programmer Performance and the Effects of the Workplace.** Procedente da 8ª Conferência Internacional de Engenharia de Software, Nova York, IEEE, p. 168-72, 1985

DOUGHERTY, Ed M. **Is human failure a stochastic process.** Reliability and System Safety, v. 5, p. 209 –15, 1997

DUPIN, F.; REER, B.; STRATER, O.; GERDES, V.; SALIOU, G.; ULLWER W. **Human-cetered modeling in human reliability analysis: some trends based on case studies.** Reliability Engineering and System Safety, v. 58, p. 249-74, 1997

EMAM, K.E.; DROUIN, J.; MELO, W. **SPICE: The Theory and Practice of Software Process Improvement and Capability Determination**. IEEE CS Press, 1998

EL-EMAM, K.; GARRO, I. **Estimating the Extent of Standards Use: the Case of ISO/IEC 15504** National Research Council Canada, Institute for Information Technology, Ottawa, 1999

FIELDS, R. E.; WRIGHT, P. C.; HARRISON, M. D.; **A task centered approach to analysing human error tolerance requirements** Requirements Engineering, Proceedings of the Second IEEE International Symposium on , 1995, p.18-26

GARCIA, S.M. **Evolving Improvement Paradigms: Capability Maturity Models & ISO/IEC 15504 (PDTR)**. Pittsburgh, Software Engineering Institute of Carnegie Mellon University, 1999

GARDNER, W. **A Nova Ciência da Mente**. 2ª edição, São Paulo, EDUSP, 1996

HAZELHURST, S. **Developing It Skills Internationally: Who's Developing Whom?** Communications of the ACM, v.44, n.6, p. 27-8, jul. 2001

HERZBERG, F. **Novamente: Como Se Faz Para Motivar Funcionários?** Biblioteca Harvard de Administração de Empresas, Boston, artigo 13, v.1, 1975

HODGES, P. **Salary Survey: Small Change for DP Pros**. Datamation, v.32, n.18, p. 72-87, set. 1986

HOWARD, A. **Software Engineering Project Management**. Communications of the ACM, v.44, n.5, p. 23-4, mai. 2001

HUGHES, J.; O'BRIEN, J.; RODDEN, T.; ROUNCEFIELD, M.; SOMMERVILLE, L.; **Presenting ethnography in the requirements process** Requirements Engineering, Proceedings of the Second IEEE International Symposium on, 1995, p. 27 -34

HUMPHREY, W.S. **Managing Technical People**. Adison Wesley, 1997

JONES, B. S.; EARTHY, J. V.; **Human-centred process integration: the process maturity perspective** Making User-Centred Design Work in Software Development (Ref. No. 1999/010), IEE colloquium on , 1999, v. 2, p. 1-15

KLAWE, M. **Refreshing the Nerds**. Communications of the ACM, v.44, n.6, p. 67-8, jul. 2001

KOONO, Z.; FAR, B. H.; **Structural way of thinking for attaining reliable software** Communications, 1994. ICC '94, SUPERCOMM/ICC '94, Conference

Record, Serving Humanity Through Communications.' IEEE International Conference on, 1994, v. 3, p. 1772-1778

KOONO, Z.; SOGA, M.; **Structural way of thinking as applied to quality assurance management** Selected Areas in Communications, IEEE Journal on , v. 8 Issue: 2 , Feb. 1990, p. 291-300

KOZAK, D.V.; BORTOLOZZI, F.; EBERSPACHER, H.F.; ELEUTERIO, M. A. **Uma Proposta de Metodologia de Desenvolvimento de Aplicativos para Treinamento Baseado em Computador.** Departamento de Informática PUC-PR, Curitiba, disponível em <http://www.minerva.uevora.pt/simposio/comunicacoes/Eberspacher2/ArtigoCBT>, Acesso em agosto de 2001

LEVINSON, H. **Atitudes asininas em Relação à Motivação.** Biblioteca Harvard de Administração de Empresas, Boston, artigo 6, v.2, 1976

MANASCO, B. **Unleashing Human Capital to Drive the Next Wave of Growth.** Ubiquity. Artigo disponível em http://www.acm.org/ubiquity/views/b_manasco_1.html Acesso em junho de 2001

MILIONI, B. **Comportamento Gerencial – O Poder Em Questão.** São Paulo, Ed.Nobel, 1990

MORISIO, M.; **Applying the PSP in industry** IEEE Software , v. 17 Issue: 6 , nov.-dec. 2000, p. 90-95

MOSCOVICI, F. **Equipes dão Certo.** José Olympio, 1995

MOORE, J.E.; YAGER, S.E.; SUMNER, M.; CROW, G.B. **Facilitating Career Changes Into IT.** Communications of the ACM, v.44, n.6, p. 71-3, jul. 2001

PAULK, M.C. **Analyzing the Conceptual Relationship between ISSO/IEC 15504 (Software Process Assessment) and the Capability Maturity Model for Software.** 1999 International Conference on Software Quality in Cambridge, Pittsburgh, Software Engineering Institute of Carnegie Mellon University, 1999

PINTO, J.K.; MILLET, I. **Successful Information System Implementation – The Human Side.** 2ª edição, Newtown Square, Project Management Institute, 1999

POTOSNAK, K.; **Human factors-management: the key to success** IEEE Software v. 6 Issue: 2 , mar. 1989, p. 86-88

RODRIGUES, M. V. C. **Qualidade de Vida no Trabalho.** Ed. Vozes, 1994

SANCHES, R.; FABRI, S.; MALDONADO, J.C. **Qualidade de Software: da Engenharia de Software aos Modelos de Qualidade.** ANAIS VI Escola Regional de Informática de São Paulo, São Carlos, p. 219-39, 2001

SCHWARCZ, L.K.M. **História da Etnologia. Levi-Strauss e os Embates em Região de Fronteira.** São Paulo, Revista de Antropologia, v.42, n.1 e 2, p.199-222, 1999

SEAMAN, C. B.; **Qualitative methods in empirical studies of software engineering** Software Engineering, IEEE Transactions on , v. 25 Issue: 4 , July-Aug. 1999, p. 557-72

SHENDIL, K.; MADHAVJI, N. H.; **Personal 'progress functions' in the software process** Software Process Workshop, Proceedings., Ninth International , 1994, p. 117 –121

STENSRUD, E.; MYRTVEIT, L.; **Human performance estimating with analogy and regression models: an empirical validation** Software Metrics Symposium, Metrics 1998. Proceedings. Fifth International , 1998, p. 205-213

THOMAS, S.; HURLEY, S.; BARNES, D. **Looking for the human factors in software quality management.** Proceedings of the 1996 International Conference on Software Engineering: Education and Practice (SE: E&P '96)

TOLRA, P.L.; WARNER, J.P. **Etnologia Antropologia.** 2ª edição, Rio de Janeiro, Editora Vozes, 1997

TRAUTH, E. M. **Mapping Information – Sector Work to the Work Force** Communications of the ACM, v.44, n.6, p. 74-5, jul. 2001

VINTER, R.; LOOMES, M.; KORNROT, D.; **Applying software metrics to formal specifications: a cognitive approach** Software Metrics Symposium, Metrics 1998. Proceedings. Fifth International , 1998, p. 216-223

WEST, L.A.; BOGUMIL, W.A **Immigration and the Global IT Work Force.** Communications of the ACM, v.44, n.6, p. 34-8, jul. 2001

WIDMAIER, J. C.; SMIDTS, C.; XIN, H.; **Producing more reliable software: mature software engineering process vs. state-of-the-art technology?** Software Engineering, Proceedings of the 2000 International Conference on, 2000, p. 88-93

WOOD, R.L.; **Recognising how humans make process modelling difficult** Systems Dependency on Humans (Ref. No. 2000/020), IEE One-day Seminar on, 1999, v. 7, p. 1-5

ZHENG, Y.; QIAN J. **Human errors in programming with interactive programming environment** Systems, Man, and Cybernetics. Proceedings of the 1988 IEEE International Conference on, v. 1, 1988, p.181-84