MARCEL JACQUES SIMONETTE

Fatores humanos na engenharia de software

São Paulo
2015

MARCEL JACQUES SIMONETTE

Fatores humanos na engenharia de software

Tese apresentada à Escola Politécnica da
Universidade de São Paulo para a
obtenção do título de Doutor em Ciências

São Paulo
2015

MARCEL JACQUES SIMONETTE

Fatores humanos na engenharia de software

Tese apresentada à Escola Politécnica da
Universidade de São Paulo para a
obtenção do título de Doutor em Ciências.

Área de Concentração:
Engenharia de Computação

Orientador:
Prof. Dr. Edison Spina

São Paulo
2015

Catálogo-na-publicação

Simonette, Marcel Jacques

Fatores humanos na engenharia de software / M. J. Simonette -- São Paulo, 2015.
210 p.

Tese (Doutorado) - Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia de Computação e Sistemas Digitais.

1.Engenharia de software (Ética) 2.Engenharia de software (Fatores humanos) 3.Engenharia de sistemas 4.Complexidade 5.Fatores humanos I.Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia de Computação e Sistemas Digitais II.t.

A Paulo e Tiago,
pela amorosa impaciência deles
com este trabalho e comigo.

AGRADECIMENTOS

Isidor Isaac Rabi, ganhador do Prêmio Nobel de Física de 1944, relata em suas memórias que deve à mãe sua carreira científica. A mãe do Isidor Rabi, Prêmio Nobel de Física, diariamente lhe perguntava: “Você fez uma boa pergunta hoje, Izzy?”. Isidor Rabi afirma que esta diferença – fazer boas perguntas – fez dele um cientista.

Boas perguntas estimulam outras boas perguntas. E isso causa movimento. Traz novos desafios. Provoca conhecimento. Durante a elaboração desta tese eu tive a felicidade de conviver com pessoas que sabem fazer boas perguntas.

Agradeço ao Prof. Dr. Edison Spina pela orientação, apoio, paciência na realização deste trabalho, e, principalmente, pelas perguntas, que a cada conversa traziam, e trazem, novos desafios. Agradeço ao Prof. Dr. Rodrigo Filev, que a cada encontro, as perguntas, as respostas e as novas perguntas provocavam mais estudos. Agradeço ao engenheiro Mário Magalhães, que fez perguntas que me levaram a descobrir autores e conhecimentos que eu nem imaginava que existiam.

Contudo, não é apenas a busca por respostas que provoca o nosso crescimento. Conviver com quem tem história, experiência para contar, também é muito importante. Meu agradecimento especial ao Prof. Dr. Luiz Barco, o eterno professor e suas histórias que mostram diversos pontos de vista sobre um mesmo assunto. No dia que conheci o Prof. Barco, fiz uma pergunta a ele que eu jamais esquecerei a resposta.

A José Otávio Proença, que com incrível paciência respondeu as inúmeras perguntas que fiz durante os meus primeiros nove meses de formado. E que muitos anos depois, já no meu mestrado, voltei a encontrar. Mas aí, foram as histórias que me contou que fizeram a diferença, propiciaram que eu conseguisse entender as relações de nexos nos processos, e só agora, na conclusão desta tese, eu entendi algumas das histórias que ele contou.

Agradeço também a Ana Maria Badiali, nossa bibliotecária, pelas inúmeras horas pacientemente dedicadas a meus artigos e a este trabalho. A Cristina Borba, pelas várias revisões de artigos, conversas e aulas, sua ajuda foi fundamental.

Não posso deixar de agradecer ao “dream team”: Marcio de Souza Cintra, Tarcísio Tomazela, Roberto Froehlich, Gilberto Tayra, Dilson Shiney e Paulo César Corigliano. Nós temos muita história, principalmente sobre como trabalhar em equipe!

Agradeço a todos os colegas do KNOMA (Laboratório de Engenharia de Conhecimento do PCS) que, direta ou indiretamente, colaboraram nesse trabalho.

A Comissão Europeia pelo projeto VertebralCue, e ao CEST – Centro de Estudo Sociedade e Tecnologia – por, em diferentes momentos, terem suportado parcialmente este trabalho.

Tão logo completei toda a trajetória dos estudos, aos finais dos quais, em geral, alguém é incluído no rol dos sábios, mudei completamente de opinião. De fato, me vi sob o peso de tantas dúvidas e erros que me parecia ter tirado, das minhas tentativas de instruir-me, um único lucro: a crescente descoberta da minha ignorância.

(René Descartes).

RESUMO

Este trabalho identifica o Código de Ética e Prática Profissional da engenharia de software como o conjunto de práticas para consideração de fatores humanos na engenharia de software. A seguir, estende o *Kernel* da especificação *Essence*, e o utiliza para conduzir a aplicação desse conjunto de práticas. A prova de conceito indica que o conjunto de práticas identificadas não garante a consideração de fatores humanos na engenharia de software. Considerar a ética nas interações existentes na empreitada de engenharia de software não é um simples caso de utilização de checklists como forma de verificar o que deve ser feito para certificar que algo foi realizado. Considerar a ética é mais do que isso. É necessário que todas as pessoas tenham consciência da importância da ética, do respeito de um ao outro e à sociedade.

Palavras chave: Engenharia de software (Ética). Engenharia de software (Fatores humanos). Engenharia de sistemas. Sistemas sociotécnicos. Complexidade. Fatores humanos.

ABSTRACT

This text identifies the Software Engineering Code of Ethics and Professional Practice as a set of practices to consider human factors in software engineering. Afterwards, it extends the Kernel of the Essence specification, and applies it to steer the use of this set of practices. The proof of concept indicates that the identified set of practices does not ensure the consideration of human factors in software engineering. Considering ethics, in the interactions that are present in software engineering endeavor, is not a simple case of using checklists to ascertain what must be done to ensure that something was carried out. Taking ethics into consideration is more than that. It is imperative that all the people should become aware of how important ethics is, and so is the respect for each other and for the society.

Keywords: Software engineering (Ethics). Software engineering (Human factors). Systems engineering. Socio-technical systems. Complexity. Human factors.

LISTA DE ILUSTRAÇÕES

Figura 1 - Diferença entre o Pensamento Sistêmico Hard e Soft.	49
Figura 2 – Paradigma de resolução de problemas em geral.	52
Figura 3 – Combinação entre paradigma de resolução de problemas em geral e o paradigma da engenharia de sistemas para solução de problemas.....	54
Figura 4 – Modelos conceitual do método soft rigoroso, visão do processo.....	56
Figura 5 – “Coisas com as quais sempre trabalhamos”.....	68
Figura 6 – “Coisas que sempre fazemos”.....	69
Figura 7 – Cartões de estados do alpha requisitos.....	72
Figura 8 – Cartões de estados do alpha forma de trabalho alinhados.....	72
Figura 9 – Exemplo de utilização de cartões para determinar o estado dos alphas.	73
Figura 10 – Processo de dinâmica com os grupos de pessoas.....	86
Figura 11 – Figura rica.	92
Figura 12 – Representação dos fatores relacionados a satisfação utilizando linguagem sistêmica.....	94
Figura 13 – Mapa Conceitual	112
Figura 14 – Interação entre sistemas contenedores	114
Figura 15 – Interação entre sistemas contenedores considerando ética e interesses	115
Figura 16 – Interação entre sistemas, após tratamento dos desequilíbrios.....	116
Figura 17 – Estados dos alphas do Kernel para cada fase do processo da organização.....	130
Figura 18 – Estados dos alphas do Kernel no início da primeira reunião da equipe do projeto..	134
Figura 19 – Estados dos alphas do Kernel após primeira reunião.....	137

LISTA DE TABELAS

Tabela 1 – Primeira versão dos elementos para construção da matriz TOWS	97
Tabela 2 – Segunda versão dos elementos para construção da matriz TOWS.	101
Tabela 3 – Matriz TOWS.....	103
Tabela 4 – Negação dos Princípios do Código de Ética e Prática Profissional	122
Tabela 5 – Checklist suplementar para os estados do alpha forma de trabalho. ...	124

LISTA DE ABREVIATURA E SIGLAS

IEEE	Institute of Electrical and Electronic Engineers
INCOSE	International Council on Systems Engineering
ISM	Interpretative Structural Modeling
OMG	Object Management Group
SEMAT	Software Engineering Method and Theory
TI	Tecnologia da Informação
TIC	Tecnologias da Informação e Comunicação
TGN	Técnica de Grupo Nominal
RSM	Rigorous Soft Method
SSM	Soft Systems Methodology

SUMÁRIO

1	INTRODUÇÃO	27
1.1	CONTEXTO	27
1.2	PROBLEMA	27
1.1	OBJETIVO	29
1.2	MÉTODO	30
1.3	JUSTIFICATIVA.....	31
1.4	RELEVÂNCIA DA TESE.....	32
1.5	ESTRUTURA E SUMÁRIO DOS CAPÍTULOS.....	33
2	DEFINIÇÕES E CONCEITOS.....	35
2.1	CONSIDERAÇÕES INICIAIS.....	35
2.2	SISTEMAS	35
2.3	FILOSOFIA DE SISTEMAS	38
2.4	TEORIA GERAL DOS SISTEMAS.....	39
2.5	PROPRIEDADES EMERGENTES E HIERARQUIA.....	41
2.5.1	Emergência	41
2.5.2	Hierarquia	42
2.6	COMPLEXIDADE.....	43
2.6.1	Complexidade – Senso comum.....	43
2.6.2	Em busca de uma definição.....	44
2.6.3	Caracterizando a complexidade	44
2.7	PENSAMENTO SISTÊMICO	45
2.7.1	A Idade dos Sistemas.....	47
2.7.2	Pensamento Sistêmico Hard e Soft	48
2.8	SISTEMA DE MÉTODOS	50
2.8.1	A dimensão humana.....	50
2.8.2	Modelo conceitual.....	51
2.8.3	Método Soft Rigoroso	55
2.9	ENGENHARIA	58
2.9.1	Engenharia de sistemas.....	59
2.9.2	Engenharia de sistemas e complexidade.....	61
2.9.3	Engenharia de software	62

2.9.4	Integração entre engenharia de sistemas e de software	64
2.10	ESSÊNCIA DA ENGENHARIA DE SOFTWARE.....	66
2.10.1	Organização do Kernel.....	69
2.10.2	Alphas	70
2.11	SISTEMAS SOCIOTÉCNICOS.....	74
2.12	FATORES HUMANOS.....	76
2.13	CONSIDERAÇÕES FINAIS	77
3	DESENVOLVIMENTO DE SISTEMAS DE SOFTWARE E PESSOAS	79
3.1	CONSIDERAÇÕES INICIAIS.....	79
3.2	PESSOAS, PROCESSOS E FERRAMENTAS – UM SISTEMA SOCIOTÉCNICO	79
3.3	PESSOAS E AS RELAÇÕES ENTRE ELAS.....	80
3.3.1	Sistema de atividades humanas	80
3.3.2	Processos e ferramentas	81
3.3.3	Valorizar as pessoas e as relações entre elas	82
3.4	SISTEMA DE ATIVIDADES HUMANAS - PROBLEMAS DE PESSOAS ...	83
3.5	CONSIDERAÇÕES FINAIS	83
4	FATORES HUMANOS NA ENGENHARIA DE SOFTWARE.....	85
4.1	CONSIDERAÇÕES INICIAIS.....	85
4.2	PERGUNTAS GUIAS DO PROCESSO DE ANÁLISE.....	86
4.2.1	Processo de dinâmica com os grupos	86
4.2.2	Perguntas iniciais	87
4.3	A QUESTÃO E SEU DOMÍNIO.....	88
4.3.1	Questão.....	88
4.3.2	Domínio.....	88
4.4	SINTOMAS E FATORES DA QUESTÃO	89
4.4.1	Sistema sociotécnico	89
4.4.2	Fatores humanos	90
4.4.3	Identificação dos fatores humanos.....	90
4.4.4	Oportunidades, Ameaças, Pontos Fortes e Pontos Fracos	95
4.5	SISTEMAS IMPLÍCITOS.....	109
4.6	SISTEMAS CONTENEDORES.....	113
4.7	INTERAÇÃO E DESEQUILÍBRIOS DOS SISTEMAS CONTENEDORES	113

4.8	TRATAMENTO PARA DESEQUILÍBRIO E AVALIAÇÃO DO IMPACTO DA PROPOSTA.....	116
4.8.1	Impacto da proposta.....	117
4.9	SOLUÇÃO POTENCIAL	117
4.9.1	Código de ética da engenharia de software.....	118
4.10	EXTENSÃO DO KERNEL DO ESSENCE	118
4.11	CONSIDERAÇÕES FINAIS	125
5	PROVA DE CONCEITO.....	127
5.1	CONSIDERAÇÕES INICIAIS.....	127
5.2	ALINHAMENTO ENTRE O PROCESSO DA ORGANIZAÇÃO E O KERNEL 127	
5.3	CARACTERÍSTICAS DO PROJETO PARA PROVA DE CONCEITO.....	131
5.3.1	Projeto Um.....	132
5.4	CONSIDERAÇÕES FINAIS	141
6	RESULTADOS E DISCUSSÃO.....	143
6.1	CONSIDERAÇÕES INICIAIS.....	143
6.2	SOBRE A PROVA DE CONCEITO.....	143
6.3	Cultura.....	144
6.4	Ética	145
6.5	CONSIDERAÇÕES FINAIS	146
7	CONSIDERAÇÕES FINAIS.....	147
7.1	SINÓPSE DA TESE.....	147
7.2	PRINCIPAIS CONTRIBUIÇÕES.....	147
7.3	OBSERVAÇÕES DE ORDEM PRÁTICA.....	148
7.4	CUMPRIMENTO DOS OBJETIVOS	148
7.5	RECOMENDAÇÕES DE TRABALHOS FUTUROS	149
	REFERÊNCIAS.....	151
	REFERÊNCIAS CONSULTADAS	161
	GLOSSÁRIO	165
	APÊNDICES.....	169
	APÊNDICE A – Teoria Geral de Sistemas	171
	APÊNDICE B – Breve Visão Histórica	175
	APÊNDICE C – Abordagem evolucionária	181

APÊNDICE D – Métodos consensuais..... 183

1 INTRODUÇÃO

Esta pesquisa é sobre os fatores humanos presentes na engenharia de software, e a complexidade existente nesse sistema sociotécnico.

1.1 CONTEXTO

Os processos e métodos de desenvolvimento de sistemas de software tem recebido a atenção tanto da comunidade científica como da indústria (BOEHM; TURNER, 2004; BOURQUE; FAIRLE, 2014; SEI – CARNEGIE MELLON UNIVERSITY, 2013; AGILE ALLIANCE, 2013; SEMAT, 2015; SCRUM.ORG, 2015). Contudo, são raras as práticas de desenvolvimento de sistemas de software que valorizam o indivíduo e a interação ente eles, que consideram as pessoas em sua integralidade. Pessoas são os principais atores do ciclo de vida de um sistema de software, tanto de forma individual como coletiva (DEMARCO; LISTER, 1999; POPPENDIECK; POPPENDIECK, 2003; COCKBURN, 2006; SIMONETTE; SANCHES; SPINA, 2009; WEINBERG, 2011a; SIMONETTE; SPINA, 2012).

1.2 PROBLEMA

Processos de desenvolvimento de sistemas de software são dependentes tanto das pessoas que os executam como das pessoas que os gerenciam. Diversos estudos, como os trabalhos de Vosburgh et al. (1984), Walston e Felix (1997), Boehm (1981, 1987), Griffy (1996), Boehm e Turner (2004), DeMarco (1986), DeMarco e Lister (1999), Koppensteiner (2009), Fernandez-Sanz e Sanjay (2011), e Weinberg (2011a), argumentam que os fatores humanos e organizacionais influenciam diretamente a qualidade, produtividade e previsibilidade do processo utilizado. Esses estudos envolvem métricas quantitativas do trabalho realizado pelas pessoas que participam do desenvolvimento do sistema de software, e auxiliam a compreensão de como os fatores humanos e organizacionais afetam as tarefas executadas durante o desenvolvimento de um sistema de software (SCACCHI, 1984; DEMARCO; LISTER, 1999; WEINBERG, 2011a).

Os estudos referenciados indicam que uma abordagem baseada em métricas melhora o processo de desenvolvimento de sistemas de software e permite uma gestão mais adequada do trabalho realizado. Contudo, essa abordagem não é a “bala de prata”,

como comentado por Frederick Brooks em seu clássico artigo de 1987: No silver bullet: essences and accidents of software engineering (BROOKS, 1987).

As pesquisas realizadas pelo Standish Group (CHAOS Report¹) indicam que há uma melhora na taxa de sucesso dos projetos de desenvolvimento de sistemas de software, passando de uma taxa de um a cada seis projetos executados com sucesso em 1994, para um a cada três no último relatório de 2014; essa melhora na taxa de sucesso dos projetos não está relacionada a uso das métricas de fatores humanos e organizacionais, ela ocorreu devido à generalização dos paradigmas Ágeis² e Lean^{3,4} entre as empresas (STANDISH GROUP, 2014; AMBLER, 2014).

Um dos principais autores sobre métricas em projetos de sistemas de software, Tom DeMarco, argumenta em artigo de 2009 (DEMARCO, 2009) que deixou de acreditar que as métricas sejam essenciais para o sucesso de um projeto, atualmente acredita que as métricas devem estar presentes, mas que não se colocam como uma condição necessária para o sucesso de um projeto.

Projetos de sistemas de software envolvem pessoas, e não é possível se usar métricas para fatores humanos como valores, ética, lealdade, honra, dignidade, disciplina, personalidade, boa vontade, bondade ou humor. Mesmo parecendo ser "bom senso" a consideração de fatores humanos na engenharia de software, Gerald Weinberg destacou em 1971, na primeira edição de seu livro *The Psychology of Computer Programming* que: "[...] a ideia de um programador como um ser humano não irá atrair certos tipos de pessoas [...]" (WEINBERG, 2011a, posição 5921).

Ainda há muito a se aprender sobre os estudos quantitativos relacionados às atividades existentes nos processos de desenvolvimento de sistemas de software, mas não se deve fixar nesses conceitos em detrimento de estudos sobre os fatores humanos e sociais presentes nesses mesmos processos.

¹ Informações sobre o CHAOS Report estão disponíveis em: <<http://blog.standishgroup.com/post/18>>. Acesso em 12 jun. 2015.

² A definição de Ágil (ou paradigma Ágil) para este trabalho está no Glossário.

³ A definição de Lean (ou paradigma Lean) para este trabalho está no Glossário.

⁴ O autor deste trabalho preferiu manter o termo em inglês ao invés de utilizar a tradução: Enxuto(a), dada a semântica que o termo Lean possui em inglês.

1.1 OBJETIVO

O objetivo mais amplo desta pesquisa é obter um conjunto de práticas que possibilitem a consideração de fatores humanos na engenharia de software.

Os objetivos específicos desta pesquisa visam contribuir para a iniciativa SEMAT⁵ e para as pesquisas em engenharia de sistemas e software do Laboratório de Engenharia do Conhecimento do PCS, são eles:

- Extensão do Kernel⁶ da engenharia de software, definido pela especificação OMG⁷: Essence - Kernel e Linguagem para Métodos de Engenharia de Software v1.0⁸, para a consideração de fatores humanos na engenharia de software.
- Compilação da bibliografia sobre complexidade e sua aplicação na área de conhecimento de engenharia de sistemas e de software.
- Contribuição para o conhecimento sobre os fatores humanos e sociais presentes no desenvolvimento de sistemas de software.

É importante citar que este trabalho não apresenta uma receita que guiará a consideração de fatores humanos de forma inequívoca em um processo de desenvolvimento de sistemas de software, mas permite que os gestores desse processo tenham referências para as ações a serem tomadas nesse respeito.

Tratar tecnologias específicas de apoio à interação entre pessoas de uma equipe de trabalho não é objetivo deste trabalho. Também não há intenção neste trabalho de apresentar e definir um inventário da infraestrutura eletrônica disponível para apoio as equipes de desenvolvimento de sistemas de software. Este trabalho também não é um estudo metodológico que fornece um guia para projetos com sucesso, nem apresenta um método de projeto exaustivo para tratar o cenário complexo dos fatores

⁵ Sigla em inglês para: Software Engineering Method and Theory. Portal da organização disponível em: <<http://www.semat.org>>. Acesso em: 12 jun. 2015.

⁶ O autor deste trabalho preferiu manter o termo em inglês ao invés de utilizar a tradução: Núcleo, dada a semântica que o termo Kernel possui em inglês.

⁷ Sigla em inglês para: Object Management Group. Portal da organização disponível em: <<http://www.omg.org>>. Acesso em 12 jun. 2015.

⁸ Esta especificação foi desenvolvida a partir do trabalho do SEMAT⁹, ela define uma linguagem para a criação, uso e melhoria dos métodos de engenharia de software. O OMG aprovou e liberou essa especificação na sua versão 1.0 em novembro de 2014.

humanos que estão presentes em um processo de desenvolvimento de sistemas de software.

Este trabalho expõe o contexto no qual ocorre o desenvolvimento de sistemas de software e realiza uma análise e síntese de maneira específica, para determinar práticas que possuem o humano e o social como centro de atenção.

1.2 MÉTODO

Para identificar um conjunto de práticas que possibilitem a consideração de fatores humanos na engenharia de software, são realizadas diversas reuniões e entrevistas com pessoas de diferentes empresas, de diferentes níveis hierárquicos, e que trabalham com diferentes processos de desenvolvimento de sistemas de software. Em comum entre elas apenas o fato de trabalharem com o desenvolvimento de sistemas de software. A identificação dessas práticas é realizada através de um sistema de métodos⁹. A escolha de um sistema de métodos para este trabalho é determinada pela sua própria característica: um sistema de métodos é livre de contexto, sendo útil para investigar problemas complexos.

O conjunto de práticas identificadas é utilizado para estender o Kernel do Essence, o que permite a consideração de fatores humanos ao se utilizar os elementos do Kernel. A escolha do Essence ocorre por ser uma especificação do OMG que fornece uma base para definição de práticas de desenvolvimento de sistemas de software, e inclui os elementos essenciais que predominam em todo esforço de engenharia de software. É também realizada uma prova de conceito, com o Kernel estendido sendo aplicado pelo autor deste trabalho em um cenário real de desenvolvimento de sistemas de software.

⁹ Um sistema de métodos incorpora métodos, ferramentas, procedimentos, processos, práticas, e as habilidades e experiências das pessoas (HITCHINS, 2007).

1.3 JUSTIFICATIVA

Diversos fatores levam à reflexão sobre a necessidade da consideração de fatores humanos na engenharia de software, entre eles estão:

- A formação dos profissionais dessa área.
- O ambiente físico no qual os profissionais exercem suas atividades.
- O respeito à dignidade do ser humano durante a empreitada de desenvolvimento de sistemas de software.

Soma-se a esses fatores a realidade na qual vivem os profissionais da área de Tecnologia da Informação (TI):

[...] nossa pesquisa descobriu que profissionais de TI precisam de muita capacidade mental e que deles se exigem as atitudes de bombeiros, treinadores, corredores e maratonistas, diplomatas e inventores. Exige-se deles que, em um piscar de olhos, estejam prontos para servir, resolver problemas, desfazer erros, acertar de primeira [...] (EMERSON, 2012, p.2).

Se não 24 horas por dia, ao menos 119 horas por semana - praticamente três vezes o tamanho de uma jornada tradicional de trabalho. Esse é o tempo exigido, em média, que profissionais da área de tecnologia da informação (TI) estejam disponíveis para suas empresas [...] (ARCOVERDE, 2012).

Existe um volume substancial de estudos voltados ao desempenho das pessoas, e de métodos de gerenciamento de pessoas para garantir esse desempenho. Esses métodos utilizam metáforas que são adequadas à produção de produtos tangíveis, ao desenvolvimento de sistemas concretos, onde o sucesso é medido pela quantidade com qualidade; são métodos que possuem pouca eficácia quando aplicados ao desenvolvimento de sistemas abstratos, onde a própria abstração leva a diferentes índices de qualidade, onde o sucesso é medido pela qualidade, independentemente da quantidade produzida (CHERRY; ROBILLARD, 2008; APPELO, 2012). São métodos de gerência orientada ao aumento e garantia da produtividade das pessoas, se preocupam com questões como motivação e engajamento, sem valorizar os fatores humanos, sem respeitar à dignidade do ser humano.

1.4 RELEVÂNCIA DA TESE

Em todo processo utilizado para se criar sistemas de software, quer seja Ágil, Prototipação, Processo Unificado, Cascata, ou qualquer outro método orientado a um plano ou não, existem atividades que sempre estão presentes, como a formação e gerenciamento de equipes e a construção e transmissão do conhecimento através da interação entre as pessoas. Os profissionais envolvidos no desenvolvimento de sistemas de software são pessoas que lidam com informação e conhecimento. DeMarco e Lister (1999) usam termos como “trabalhadores do pensamento”, “trabalhadores do conhecimento” e “trabalhadores do intelecto” quando se referem a essas pessoas, indicando que elas necessitam pensar. Poppendieck e Poppendieck (2003) destacam que essas pessoas necessitam pensar profundamente para tratar a complexidade existente no desenvolvimento de um sistema de software. Diversos são os trabalhos – entre eles DeMarco e Lister (1999), Poppendieck e Poppendieck (2003), Cockburn (2006), Humphrey e James (2011) – que exemplificam e demonstram que profissionais que trabalham com sistemas de software são diferenciados, são “trabalhadores do conhecimento” que devem ser vistos de forma diferente dos “trabalhadores manuais”¹⁰. Essa diferenciação ocorre devido à necessidade constante da indústria em elevar a produtividade, previsibilidade e a qualidade do trabalho. Contudo, essa necessidade acaba por considerar as pessoas como componentes do processo; componentes ou ferramentas que devem executar as tarefas previstas no método que implementa o processo.

Este trabalho vai além dos aspectos motivacionais e de engajamento profissional que são relacionados à produtividade, previsibilidade e qualidade. Ele apresenta uma perspectiva da consideração de fatores humanos na engenharia de software. Uma perspectiva que considera as pessoas em sua integralidade, valorizando os indivíduos e a interação entre eles.

¹⁰ “trabalhadores manuais” e “trabalhadores do conhecimento” é uma nomenclatura adotada por Drucker (1999) para diferenciar dois grupos de trabalhadores distintos, sendo que os fatores que influenciam a produtividade, a previsibilidade e a qualidade do “trabalhador do conhecimento” não são necessariamente os mesmos que regem o “trabalho manual”.

1.5 ESTRUTURA E SUMÁRIO DOS CAPÍTULOS

Este capítulo, introdutório, delimita o assunto tratado, define o objetivo e justifica a pesquisa. O restante deste texto encontra-se organizado da seguinte forma:

- O capítulo dois apresenta as definições e conceitos utilizados neste trabalho.
- O capítulo três apresenta o trabalho em equipe no processo de desenvolvimento de sistemas de software como um sistema sociotécnico. Neste capítulo é reconhecido o valor de processos, métodos e da tecnologia presente nesse sistema sociotécnico, e é apresentada a importância das pessoas e da interação entre elas.
- O capítulo quatro apresenta o sistema de métodos utilizado para obter um conjunto de práticas que possibilita a consideração de fatores humanos na engenharia de software. A seguir, apresenta a extensão da especificação Essence da OMG para implementar a consideração de fatores humanos na engenharia de software.
- O capítulo cinco apresenta a prova de conceito, o Essence com seu Kernel estendido é aplicado em uma empreitada de engenharia de software com o objetivo de identificar se a extensão permite a consideração de fatores humanos na engenharia de software.
- O capítulo seis apresenta a discussão dos resultados obtidos através da prova de conceito, e as considerações sobre o problema e a resposta ao problema da consideração de fatores humanos na engenharia de software.
- O capítulo sete apresenta as considerações finais e é seguido pelas referências e um glossário.

2 DEFINIÇÕES E CONCEITOS

Este capítulo apresenta definições e conceitos utilizados neste trabalho, em particular, apresenta os aspectos dessas definições e conceitos que são relacionados à engenharia de sistemas e de software.

2.1 CONSIDERAÇÕES INICIAIS

No final dos anos 20 e início dos anos 30 do século XX, o conhecimento científico e sua filosofia Cartesiana, que ainda oferecem respostas que ampliam o conhecimento humano, mostrava sinais de ter alcançado algum limite. O principal exemplo desse limite ocorreu na Biologia, onde Ludwig von Bertalanffy identificou as limitações do Cartesianismo para responder questões relacionadas à Vida.

Durante a Segunda Guerra Mundial, as Forças Armadas Americanas recrutaram diversos cientistas e engenheiros para tratar os problemas relacionados com os esforços de guerra. Problemas logísticos, de comunicação e ações, como bombardeios estratégicos, estavam entre as demandas do governo para os cientistas e engenheiros. Engstrom (1957) argumenta que a experiência vivida pelos engenheiros e cientistas durante a Segunda Guerra Mundial trouxe um grande ímpeto à aplicação de uma abordagem sistêmica, uma filosofia, que permitisse entender problema e solução em sua integralidade. Engstrom prossegue comentando que a prática, durante as circunstâncias e pressões da Guerra, era pensar além dos componentes individuais e seus usos no ambiente onde esses componentes iriam operar. Era necessário considerar o objetivo final: o sistema.

2.2 SISTEMAS

A palavra sistema possui uma natureza subjetiva. Utiliza-se a palavra sistema para se referir a formas de organização ou funções, ela não se refere a um objeto que existe no mundo real. O construtivismo radical de Ernest von Glasersfeld¹¹ permite compreender a natureza subjetiva do termo sistema:

¹¹ Informações sobre o trabalho intelectual de Ernest von Glasersfeld estão disponíveis em: <<http://www.vonglasersfeld.com>>. Acesso em 12 jun. 2015.

O que é construtivismo radical? Ele é uma abordagem não convencional para problemas do saber e do conhecimento. Ele tem início com a suposição de que conhecimento, não importando como ele é definido, está na cabeça das pessoas, e que o indivíduo pensante não tem alternativa senão construir o que ele ou ela sabe com base na própria experiência dele ou dela. O que fazemos de nossa experiência constitui o único mundo em que conscientemente vivemos. Ela [a experiência] pode ser classificada em vários tipos, como coisas, o self¹², os outros e assim por diante. Mas todos os tipos de experiência são essencialmente subjetivos, e embora eu possa encontrar razões para acreditar que minha experiência pode não ser diferente da sua, eu não tenho como saber que elas são as mesmas. A experiência e a interpretação da linguagem não são exceção. (GLASERSFELD, 1995, p.1, tradução do autor).

O construtivismo radical permite afirmar que um sistema não existe no mundo real de forma independente da mente humana. Um sistema é sempre uma abstração escolhida de acordo com as experiências das pessoas. Essa abstração pode ter ênfase em aspectos do que está sendo pensado ou observado, sendo que esses aspectos podem ser estruturais, ou funcionais ou ambos. Contudo, apesar desses aspectos serem utilizados para se identificar um sistema, apenas a análise de aspectos estruturais e/ou funcionais não permite a total compreensão de um sistema. As interações entre as partes estruturais ou funcionais de um sistema e o propósito ou significado da composição das partes podem ser compreendidos apenas quando se considera o conjunto em sua integralidade. Separando suas partes, o sistema perde as características que o levaram a ser identificado como sistema. Essa é uma característica essencial de um sistema, as interações entre suas partes devem ser compreendidas da mesma forma como se busca compreender as suas partes. Meadows (2009) argumenta que quando se consegue observar a relação entre estrutura e comportamento de um sistema, é possível se entender como ele funciona. Uma definição mais científica de sistema é dada por Russell Ackoff:

[...] um sistema é um conjunto de dois ou mais elementos que satisfazem três condições:

- 1- O comportamento de cada elemento tem um efeito no comportamento do sistema como um todo.
- 2- O comportamento dos elementos e seus efeitos sobre o todo são interdependentes.

¹² O autor deste trabalho preferiu manter o termo em inglês ao invés de utilizar a tradução: próprio, ou comportamento individual, dada a semântica que o termo self possui em inglês.

- 3- Quando subgrupos de elementos são formados, cada um possui um efeito sobre o comportamento do todo e nenhum possui um efeito independente sobre o todo. Visto de outra forma, os elementos de um sistema estão tão conectados que subgrupos independentes não podem ser formados. (ACKOFF, 1981, p.15, tradução do autor).

Derek Hitchins, em seu livro *Putting Systems to Work* (HITCHINS, 1992), define sistema de uma forma mais pragmática:

Um sistema é um conjunto de entidades inter-relacionadas de tal forma que ambos, as entidades e os seus inter-relacionamentos reduzem a entropia local. (HITCHINS, 1992, p. 56, tradução do autor).

Nessa definição, Hitchins afirma que as relações recebem o mesmo grau de importância que as entidades, isso é necessário porque, ao se identificar o padrão ou rede de relações, se reduzem as incertezas sobre o sistema, da mesma forma como ocorre quando se identificam as entidades. Hitchins prossegue afirmando que a noção de entropia local está incluída na definição apenas para sugerir que um sistema pode ter limites além dos quais não existem elementos ou relações que influenciem a sua desordem.

Meadows (2009) define sistema de uma forma mais simples, contudo, acrescenta o conceito de propósito do sistema:

[...] um conjunto de elementos ou partes que estão organizados de forma coerente e interconectados de acordo com um padrão ou estrutura que produz um conjunto característico de comportamentos, frequentemente classificado como função ou propósito. (MEADOWS, 2009, p.188, tradução do autor).

Simplificando ainda mais, Maturana e Varela (1980) definem sistema como qualquer conjunto definível de componentes.

2.3 FILOSOFIA DE SISTEMAS

Esta é uma filosofia desenvolvida a partir de conceitos de sistemas. Ela apresenta uma reorganização das formas de perceber e pensar.

Em contraste com o reducionismo, determinismo, e a forma de pensar unidirecional de causa e efeito da ciência clássica, a filosofia de sistemas estabelece uma reorganização das formas de pensar. Ela cria uma nova visão de mundo, um novo paradigma de percepção e explanação, que se manifesta na integração, no pensamento holístico, na busca de propósitos, na causalidade mútua, e é centrada no processo de investigação. (BANATHY, 1991, p.31-32, tradução do autor).

A filosofia de sistemas foi descrita pela primeira vez por Ervin Laszlo no livro *Systems Philosophy: Towards a New Paradigm of Contemporary Thought*¹³, editado em 1972. Na edição revisada de seu livro *General System Theory*, publicada em 1976, Ludwig von Bertalanffy não alterou o texto da edição anterior incluindo esta filosofia, mas, em seu prefácio, Bertalanffy argumenta que esta filosofia é a reorientação do pensar e ver o mundo que surge do conceito de sistema como um novo paradigma científico, em contraste com a ciência clássica; é a nova filosofia da natureza, que considera o mundo como uma grande organização que é mais orgânica do que mecânica.

Essa analogia orgânica não implica que os sistemas nos quais estejam presentes pessoas, civilizações, sociedade, indústrias e empresas devam ser considerados como organismos. Ela sugere que, ao se analisar um sistema formado pelas interações entre suas partes, é necessário entender essas partes como complementares, desempenhando funções em nome do sistema como um todo (HITCHINS, 2003, 2007).

Hitchins (2003) argumenta que a filosofia de sistemas possui três princípios que são importantes para a engenharia de sistemas:

- **Holismo** - ou preocupação com o todo - propõem que tudo em um sistema está conectado ou relacionado - e afeta - a tudo, há, portanto, uma interdependência mútua entre as partes. Considera que não é racional tratar

¹³ Mais informações sobre Ervin Laszlo estão disponíveis em: <<http://systemsphilosophy.org/history-and-development-of-systems-philosophy.html>>. Acesso em: 12 de jun.2015.

apenas as partes de forma isolada. Os sistemas, e os problemas relacionados a eles, devem ser vistos em sua integralidade, como um todo. O holismo observa a tendência que o mundo natural tem de criar “todos”, e que um todo pode ser mais que a soma de suas partes.

- **Síntese** - elaborada a partir das partes - é o oposto de redução. A síntese propõe que as várias partes de um sistema não podem existir, ou sobreviver, ou operar, ou se comportar ou serem consideradas em isolamento mútuo. Um sistema existe quando as suas partes complementares estão juntas. Cada uma depende da existência da outra e de suas relações.
- **Analogia Orgânica** – é desenvolvida a partir de partes complementares que interagem para apoiar umas às outras e para criar capacidades e comportamentos. Apesar de nem todos os sistemas serem orgânicos, eles se parecem com organismos biológicos e se comportam como um todo unificado. As partes possuem um ciclo de vida, exibindo um crescimento, estabilidade e morte, algumas vezes repentinas, que podem levar ao colapso do sistema.

Em relação a síntese, Ackoff (1981) argumenta que ela está focada na função, revelando o por que das coisas funcionarem, favorecendo a compreensão e possibilitando a explanação.

A filosofia de sistemas reorienta o pensamento e a visão de mundo seguindo conceitos de sistemas, refletido um conceito de que tudo no mundo concreto é um sistema ou parte de um sistema, sendo que ele só pode ser entendido quando se considera que ele funciona em sua integralidade devido ao relacionamento entre as suas partes.

2.4 TEORIA GERAL DOS SISTEMAS

Klir (2001) argumenta que o holismo, e o reconhecimento crescente da existência e utilidade de isomorfismos entre disciplinas, criou uma consciência crescente de que determinados conceitos, ideias, princípios e métodos podiam ser aplicados a sistemas em geral, apesar da categorização disciplinar que eles possuíam. Isso levou à noção de Sistemas Gerais, Teoria Geral dos Sistemas, Pesquisa em Sistemas Gerais e termos semelhantes.

Não apenas os aspectos gerais e pontos de vista são parecidos em diferentes ciências; frequentemente encontramos leis formalmente idênticas ou isomórficas em diferentes campos. [...] parece existir leis gerais de sistemas que se aplicam para qualquer sistema de certo tipo, independente das propriedades particulares do sistema e dos elementos envolvidos. Essa consideração leva a se postular uma nova disciplina científica que nós chamamos de Teoria Geral dos Sistemas. Seu objetivo é a formulação de princípios que são válidos para 'sistemas' em geral, qualquer que seja a natureza de seus componentes e a relação ou 'forças' ente eles. (BERTALANFFY, 1976, p. 37, publicado pela primeira vez em 1969, tradução do autor).

A Teoria Geral de Sistemas segue a exploração científica e a compreensão de sistemas que existem em diversas áreas de conhecimento, de forma a obter uma expressão organizada de conjuntos de conceitos que se inter-relacionam e de princípios que se aplicam a todos os sistemas (BANATHY, 1991). Esta Teoria trata as propriedades gerais dos sistemas, independente de formas físicas, funcionais ou domínios de aplicação. Ela é ao mesmo tempo uma ciência básica e uma ciência aplicada (STROKES, 1997; SKYTTNER, 2005; HITCHINS, 2007; STASINOPOULOS, 2009; MEADOWS, 2009):

- Como em uma ciência básica, a Teoria Geral de Sistemas proporciona uma forma de se abstrair a realidade, simplificando e ao mesmo tempo capturando a multidimensionalidade. Como uma epistemologia, ela estrutura não apenas o pensamento sobre a realidade, mas também o pensamento sobre o próprio pensamento.
- Como uma ciência aplicada, a Teoria Geral de Sistemas se tornou uma Ciência de Sistemas, uma meta-disciplina com um conteúdo capaz de se transferir de disciplina para disciplina. Ela trata de conhecimento a respeito de conhecimento e tenta adicionar e integrar os aspectos que não parecem ser adequadamente tratados pela ciência que tem como base o Cartesianismo.

Em relação à ciência que tem como base o Cartesianismo, faz-se necessário ressaltar que a visão mecanicista e determinista que ela possui permitiu e permite o sucesso de inúmeros sistemas desenvolvidos pelo homem. Contudo, como argumentado por Hitchins (2007), o Cartesianismo apresenta dificuldades ao tratar pessoas e

instituições sociais, muitas vezes vistos apenas como parte do contexto, sem pertencer diretamente ao sistema, ou, algumas vezes, são simplesmente ignorados. Mais informações sobre a Teoria Geral de Sistemas podem ser encontradas no Apêndice A.

2.5 PROPRIEDADES EMERGENTES E HIERARQUIA

Emergência é uma propriedade de um sistema que não está associada a nenhum de seus componentes especificamente, e também a nenhuma relação específica que exista entre seus componentes. Ao se decompor um sistema para estudar uma emergência, ela deixa de existir. Jackson (2003) argumenta que emergência é um movimento de regras de comportamentos que se observa em sistemas, desde relações simples entre seus componentes, até relações sofisticadas, ou múltiplas relações, que envolvem múltiplos componentes.

2.5.1 Emergência

Ao considerar o conceito de um sistema ser um *todo*¹⁴, que não pode ser dividido em partes independentes para que seja possível um total entendimento dele, Ackoff (1981) postula duas importantes propriedades:

- Todas as partes de um sistema possuem propriedades que deixam de existir quando elas são separadas do sistema.
- Todos os sistemas possuem propriedades essenciais, propriedades que nenhuma das partes possui isoladamente.

Hitchins (2007) também argumenta sobre as propriedades essenciais, chamando-as de propriedades emergentes. Ackoff (1981) e Hitchins (2007) afirmam que, devido às propriedades emergentes (essenciais), um sistema não pode ser entendido em sua integralidade quando se utiliza apenas a análise do Reduccionismo Cartesiano. Hitchins prossegue afirmando que a engenharia pode construir sistemas combinando as suas partes de uma forma específica para que ele apresente propriedades

¹⁴ O autor deste trabalho utiliza neste texto o substantivo **todo** para traduzir do inglês o termo **whole**. **Todo** tem o significado de integralidade, significando totalidade, um conjunto completo.

emergentes específicas, que são desejadas, e minimize propriedades indesejadas; métodos para alcançar esses objetivos constituem a essência da engenharia de sistemas.

Em relação à emergência em sistemas, Hitchins (2007) destaca que ela não é uma propriedade ou conceito misterioso, nem que ela só ocorre para sistemas chamados complexos. Como exemplo, ele cita o caso da Amônia, um gás que possui um cheiro forte, mas é formado por dois gases inodoros, o Hidrogênio e o Nitrogênio. Neste caso, o cheiro resultante se deve à emergência que provem das interações dinâmicas entre os vários elementos do todo, ou sistema.

2.5.2 Hierarquia

Um conjunto de componentes de um sistema é considerado outro sistema (subsistema) se ele exibir propriedades emergentes. Essa condição permite que seja identificada uma estrutura, uma hierarquia de sistemas.

As propriedades emergentes são uma ilustração de um paradigma alternativo para sistemas, um paradigma que está relacionado com o todo e suas propriedades. Ele é holístico, mas não no sentido usual de se referir ao todo; é um conceito de sistemas que está relacionado com o todo e sua hierarquia, não apenas com o todo (CHECKLAND, 1999).

Hierarquia neste caso possui um significado diferente do usual, que se refere a estruturas administrativas ou militares. Em um sistema, quando ocorre interação entre seus componentes, de tal forma que se tenha um todo com propriedades emergentes, um nível de hierarquia é estabelecido. Quando um sistema possui subsistemas que começam a interagir formando um todo coerente com propriedades emergentes, então um novo nível de hierarquia é estabelecido. Esse novo todo coerente interage com outro complementar, e surgem novas propriedades emergentes, então um novo nível mais elevado de hierarquia é identificado, caracterizado por propriedades emergentes que não existiam no nível anterior, onde elas nem possuem significado. (CHECKLAND, 1999; HITCHINS, 2007).

Hitchins (2007) argumenta que o conceito de hierarquia é consistente tanto na natureza, como nas atividades humanas, como para os sistemas desenvolvidos pelo homem.

Considerando as atividades humanas, classificadas por Checkland (1999) como uma das classes de sistemas encontradas no universo¹⁵, pode-se iniciar uma análise e síntese das propriedades emergentes a partir do corpo humano. Os subsistemas do corpo humano, como o sistema nervoso, sistema pulmonar, sistema cardíaco, etc., interagem para formar o humano, onde as propriedades emergentes suportam e constituem a vida humana. Acrescentando-se mais um nível, vários humanos podem se juntar para formar uma equipe ou uma família, que também possui propriedades emergentes e é isso que diferencia uma equipe ou uma família de um grupo de pessoas. Isso ocorre porque os humanos em uma equipe ou família interagem uns com os outros, formando um todo que é capaz de realizações maiores que a soma dos indivíduos separados. Podemos ainda prosseguir com o cenário em que várias equipes podem se juntar para sintetizar uma organização, como uma federação, com propriedades emergentes, e assim por diante. Apenas quando se observa o próximo nível de hierarquia é possível determinar o que podem ser as propriedades emergentes e quais serão seus impactos do novo todo formado.

Mais informações sobre hierarquia podem ser encontradas no Apêndice A.

2.6 COMPLEXIDADE

Complexidade é um conceito subjetivo presente em diversas áreas de conhecimento. Essa múltipla presença torna confusa a sua utilização e compreensão.

A “complexidade” é um fenômeno não simplificável e traduz uma incerteza que não pode se erradicar no próprio seio da cientificidade. (MORIN; CIURANA; MOTTA, 2009, p. 48, aspas dos autores).

2.6.1 Complexidade – Senso comum

Frequentemente se utiliza as palavras “complexidade” e “complexo” para se referir a coisas ou situações e sistemas que são difíceis de entender ou de construir. Essa

¹⁵ Checkland (1999) afirma que existem cinco classes de sistemas: os sistemas naturais, os sistemas físicos, os sistemas abstratos, os sistemas de atividades humanas e os sistemas que estão além da compreensão humana. Sobre os sistemas de atividades humanas, sistemas onde os componentes são atividades humanas, Checkland destaca que o ato humano de projetar é por si só um exemplo desta classe de sistema.

forma de classificação depende da experiência de quem está observando a situação ou sistema, e depende também do propósito que o observador tem em relação a essa situação ou sistema. Contudo, se a familiaridade, ou conhecimento, determinam se algo é complexo ou não, então esse conceito de complexidade dado pelo senso comum não é fixo.

2.6.2 Em busca de uma definição

Vasconcellos (2003, p.104) argumenta que: “[...] não é a complexidade que é nova, mas é seu reconhecimento pela ciência que é muito recente”. Esse reconhecimento recente, e o uso do termo complexidade em distintas áreas científicas, tem levado a diferentes interpretações sobre o que é complexidade, como apresentado por Waldrop (1992), Casti (1995), Capra (1996), e Mitchell (2009). O Professor Joseph M. Sussman¹⁶ do Departamento de Engenharia Civil e Ambiental do Massachusetts Institute of Technology (MIT) apresenta vinte concepções diferentes sobre complexidade em sistemas¹⁷.

Dada às várias concepções e diferentes áreas de conhecimento que estão lidando com a complexidade, o Professor Brian Castellani, Diretor do Grupo de Complexidade na Saúde da Kent State University¹⁸ buscou traçar uma compreensão dos múltiplos conceitos relacionados à complexidade; ele elaborou um mapa¹⁹ que fornece uma perspectiva histórica da complexidade desde os anos 40 e 50 do século XX até 2015.

2.6.3 Caracterizando a complexidade

Embora complexidade tenha diferentes interpretações, Vasconcelos (2003) argumenta que ao contrário do que se pode acreditar, a complexidade não é uma

¹⁶ A página web do Prof. Joseph M. Sussman no MIT está no endereço: <<http://cee.mit.edu/sussman>>. Acesso em 12 jun. 2015.

¹⁷ O trabalho com os vinte conceitos sobre complexidade identificados pelo Prof. Joseph M. Sussman está disponível em: <<http://web.mit.edu/esd.83/www/notebook/20ViewsComplexity.PDF>>. Acesso em: 12 jun. 2015.

¹⁸ As páginas web sobre o Prof. Brian Castellani estão disponíveis nos endereços: <<http://art-sciencefactory.com/about-factory.html>> e <http://cch.ashtabula.kent.edu/b_castellani.html>. Acessos em: 12 jun. 2015.

¹⁹ O mapa da complexidade está disponível no endereço: <http://art-sciencefactory.com/complexity-map_feb09.html>. Acesso em: 12 jun. 2015.

propriedade específica dos fenômenos biológicos e sociais, ela é um pressuposto epistemológico transdisciplinar²⁰.

Não sendo um conceito fixo, Hitchins (2003) e Vasconcelos (2003) argumentam que é possível identificar aspectos que caracterizam elementos da complexidade. Para Hitchins (2003, p.91) esses aspectos são:

- **Variedade:** variedade de elementos e/ou relações.
- **Conectividade:** quantidade de conexões – relações – entre elementos.
- **Desordem:** pode ser visto como o entrelaçamento ocasionado pelo arranjo dos elementos, que pode afetar as relações e até dificultar a identificação da variedade e da conectividade.

Vasconcelos (2003, p.107) também considera três aspectos, e destaca que eles estão relacionados à transformação do paradigma da ciência, e possuem origem em questões levantadas pela física. Eles são:

- O “**problema lógico**”: Tratar as contradições, ao invés de excluí-las.
- O “**problema da desordem**”: O reconhecimento da desordem, ou da tendência a desordem; exige a consideração da indeterminação e da imprevisibilidade dos fenômenos.
- O “**problema da incerteza**”: O reconhecimento de que a relação entre o observador e o sistema observado restringe a observação, isto é, pode interferir no que o observador consegue identificar.

Os aspectos considerados por Vasconcelos são formas de pensar que buscam lidar com as limitações do Cartesianoismo, e são instrumentos fundamentais para oferecer uma resposta aos três aspectos apontados por Hitchins.

2.7 PENSAMENTO SISTÊMICO

Pensamento sistêmico é a filosofia de sistemas como um processo. É um modo de abordar problemas não apenas separando-os em suas partes menores, mas vendo

²⁰ Informações sobre o conceito de transdisciplinaridade podem ser encontradas em Nicolescu (1999).

as partes e as relações entre elas como um problema maior. Através do arcabouço dado pelas leis e princípios da teoria de sistemas os engenheiros utilizam o pensamento sistêmico para caracterizar suas observações do todo. Checkland (1999) argumenta que o Pensamento Sistêmico está apoiado em dois pares de ideias:

- **Emergência e hierarquia**, comentados nos tópicos anteriores.
- **Comunicação e controle**, a comunicação no sentido de transferência de informação.

Checkland destaca que controle é o meio pelo qual um todo retém sua identidade e/ou desempenho sobre circunstâncias que não se mantêm constantes. Por exemplo, um processo de tomada de decisão assegura que ações de controle serão tomadas considerando o propósito ou a missão do sistema, sem prejudicar seu desempenho. Hitchins (2007) afirma que, como as ideias relacionadas à teoria de sistemas podem ser aplicadas a todos os tipos de sistemas, o pensamento sistêmico permite o entendimento e modelagem do comportamento de um sistema.

O pensamento sistêmico é utilizado para a compreensão de problemas, sejam eles considerados complexos ou não. Ackoff (1981) expõe que há três formas de se tratar um problema: ele pode ser resolvido, solucionado ou dissolvido:

- **Problema Resolvido**: resolver um problema é encontrar uma resposta que seja “boa o suficiente”, uma resposta que o satisfaça. É uma abordagem mais qualitativa do que quantitativa, baseada em experiências anteriores, bom senso e julgamento subjetivo.
- **Problema Solucionado**: solucionar um problema é encontrar a resposta correta, da mesma forma como se soluciona uma equação. Utiliza métodos científicos, técnicas e ferramentas, como modelos matemáticos e simulações.
- **Problema Dissolvido**: dissolver um problema é mudar a situação de alguma forma que cause o desaparecimento do problema. É idealizar, ao invés de satisfazer ou aperfeiçoar. É alterar o sistema ou o ambiente onde ocorre o problema, para que se alcance um estado desejado onde o problema não

ocorra. Na prática pode significar não tratar o cenário como ele é apresentado.

Em relação ao problema, deve-se destacar que pode haver casos em que não é possível se compreender completamente o problema, ou, mesmo que compreendido, o problema pode não ter uma solução. Contudo, mesmo não tendo solução, um problema pode ser resolvido da melhor maneira possível em um primeiro momento, e, com a compreensão desse primeiro tratamento, é possível iniciar outro ciclo de compreensão do problema e reposta. Esse ciclo²¹ se repetirá até que se tenha todo o entendimento do problema e, eventualmente, uma solução.

2.7.1 A Idade dos Sistemas

Ackoff (1974, 1981, 1997) argumenta que na década de 40 do século XX teve início a Idade dos Sistemas. Uma Idade preocupada com os sistemas que permitem escolhas tanto de significados como de propósitos, sendo a consideração das pessoas envolvidas com o sistema um dos problemas centrais. Ackoff também argumenta que, na Idade dos Sistemas, o princípio central é a síntese através do pensamento sistêmico. Hitchins (2007) argumenta que é a síntese, em oposição ao reducionismo cartesiano, o caminho a ser utilizado pela engenharia de sistemas para o tratamento das questões que envolvem projetos de sistemas. Na Idade dos Sistemas a análise não é abandonada, análise e a síntese são utilizadas para se compreender um sistema em sua integralidade.

O pensamento sistêmico e seu processo cognitivo de análise e síntese permite a compreensão das complementaridades e da unidade dos elementos e das interações em um sistema, sem privilegiar as características específicas que possam existir ou a competitividade que possa existir entre elementos ou interações de um sistema (MORIN, 1992).

²¹ Um exemplo desse ciclo é a abordagem evolucionária apresentada no Apêndice C.

2.7.2 Pensamento Sistêmico Hard e Soft

De acordo com Engstrom (1957) e Hall (1962), a engenharia de sistemas, através do arcabouço dado pelas leis e princípios da teoria de sistemas, utiliza o pensamento sistêmico para tratar os problemas surgidos durante a Segunda Grande Guerra. E, conforme apresentado por Ackoff (1981), Checkland (1999) e Hitchins (2007) os problemas da Idade dos Sistemas trazem a preocupação com as pessoas, a humanização dos sistemas.

Em relação ao pensamento sistêmico, há duas abordagens sistêmicas: a abordagem Hard²² e a abordagem Soft²³, sendo que o uso de uma ou de outra está relacionado à percepção que se tem do sistema (CHECKLAND 1999; JACKSON, 2003). A figura 1, adaptada de Checkland (1999), é um diagrama que apresenta a diferença de percepção que se tem de um sistema.

A partir das percepções que se pode ter de um sistema, Hitchins (2007) sugere haver formas duas escolas de engenharia de sistemas para tratar um problema:

- **Escola de Sistemas Hard** utiliza o conceito de criar um novo sistema que pode ser introduzido em uma situação problemática para solucionar o problema.
- **Escola de Sistemas Soft** utiliza o conceito de olhar os sintomas do problema e tentar reparar, diminuir ou contorná-lo, visando suprimir os sintomas para resolver o problema.

A primeira escola é caracterizada pela observação Hard do problema e solução. Considera que há um estado atual e um estado desejado a ser atingido, e seleciona a melhor maneira de reduzir a diferença entre eles. Para alcançar o objetivo, a solução do problema para esta escola possui um propósito claro e será desenvolvida, entregue, colocada a funcionar, suportada e eventualmente substituída no final de seu ciclo de vida. Ao mesmo tempo em que reconhece a importância de interações e

²² O autor deste trabalho preferiu manter o termo em inglês ao invés de utilizar a tradução: Dura, dada a semântica que o termo hard possui em inglês.

²³ O autor deste trabalho preferiu manter o termo em inglês ao invés de utilizar a tradução: Flexível, dada a semântica que o termo soft possui em inglês.

processos, esta escola enfatiza os aspectos funcionais, estruturais e arquitetônicos das soluções (CHECKLAND, 1999; JACKSON, 2003; HITCHINS, 2007).

Figura 1 - Diferença entre o Pensamento Sistêmico Hard e Soft.

Fonte: Adaptado de Checkland (1999).

A segunda escola é caracterizada pela visão Soft do problema e pela busca de uma solução que o resolva, que seja uma resposta que satisfaça o problema em um determinado momento. Esta escola investiga o problema a ser tratado, buscando experiências práticas e interações com o problema, tentando entender a natureza dos sintomas do problema. Esta escola modela os sistemas tentando expressar diferentes pontos de vistas, deixando explícitas as implicações dessas diferenças, de forma a construir expectativas diferentes que possam ser exploradas sistematicamente, com o objetivo de gerar um processo de aprendizado sistêmico sobre o problema (CHECKLAND, 1999; HITCHINS, 2007). Jackson (2003) argumenta que a abordagem Soft admite que não é possível assumir que os objetivos e propósitos para o sistema sejam facilmente identificados, e que isso é devido aos múltiplos valores, desejos e interesses existentes em relação ao problema e à solução.

2.8 SISTEMA DE MÉTODOS

Sistema de métodos é uma abordagem que utiliza métodos de pensamento sistêmico e teoria de sistemas para lidar com situações onde os problemas não estão estruturados, situações em que diferentes pessoas possuem diferentes entendimentos sobre o que é o problema. Um sistema de métodos permite gerenciar e alterar sistemas. (BANATHY, 1991).

Hitchins (2007) argumenta que um sistema de métodos incorpora métodos, ferramentas, procedimentos, processos, práticas e as habilidades e experiências das pessoas, é uma “forma de fazer as coisas”. Hall (1989) afirma que um sistema de métodos é um processo criativo de resolução de problemas que pode ser usado por indivíduos, pequenos grupos, equipes multidisciplinares, ou até mesmo equipes de equipes; Hall afirma que um sistema de métodos é uma coleção de quatro conjuntos (HALL, 1989, p.27):

- Um conjunto de atividades.
- Um conjunto de ferramentas (ou técnicas).
- Um conjunto de pessoas.
- Um conjunto de relações entre as atividades. As relações são definidas por conexões lógicas, que levam a uma sequência de atividades e apresentam uma realimentação entre os elementos dos conjuntos.

Um sistema de métodos é multifacetado, trabalha com perspectivas que consideram diversos pontos de vista. Sendo uma abordagem sistêmica, um sistema de métodos lida com elementos de um problema, com as inter-relações entre esses elementos e as relações desses elementos com o meio ambiente.

2.8.1 A dimensão humana

As primeiras iniciativas em utilizar a engenharia de sistemas para lidar com problemas do setor público, como distribuição de água e energia para população, não tiveram sucesso. Essa iniciativa teve origem na sociedade americana, sob o conceito de que se a engenharia de sistemas contribuiu para a vitória americana na Segunda Grande Guerra. Se essa engenharia auxiliou a construção de sistemas de defesa no início da

guerra fria e se levou o homem à Lua, porque não utiliza-la para tratar os problemas do setor público? A resposta principal a esse fracasso está na dimensão humana dos problemas enfrentados no setor público, que são muito diferentes daqueles que os engenheiros estavam acostumados na prática da engenharia para sistemas de defesa e de missões espaciais (HALL, 1989; HUGUES, 2005; HITCHINS, 2007).

Hall (1989) argumenta que essa falha ocorreu por não ter sido considerada a cultura da população local, onde a solução²⁴ do problema dada pela engenharia de sistemas foi implementada. Hall destaca que, apesar de um sistema de métodos não ter uma forma abrangente de lidar com as questões culturais, ele permite que se seja sensível para essas questões, e deve ser utilizado pela engenharia de sistemas moderna mesmo que o fato de considerar a dimensão humana dos problemas seja considerado discutível e hipotético; um sistema de métodos é orientado a um objetivo com a intenção de satisfazer as necessidades humanas das pessoas envolvidas. Hitchins (2007) afirma que um sistema de métodos trabalha com os aspectos culturais presentes tanto no problema como na resposta ao problema.

Sistemas de métodos trabalham com aspectos culturais presentes tanto na dimensão do problema como na dimensão da resposta ao problema. Além disso, sistemas de métodos devem ter uma integridade, devem ser aderentes a princípios morais e ao código de conduta da engenharia. Eles devem ser éticos, tratando problema e resposta de forma imparcial, objetiva e culturalmente consciente (HITCHINS, 2009).

2.8.2 Modelo conceitual

Sistemas de métodos é, em sua essência, um conjunto de métodos para resolução de problemas. Hitchins (2007) argumenta que o paradigma para resolução de problemas em geral, representado na figura 2, é simples em conceito e poderoso em sua aplicação, especialmente quando combinado com uma abordagem sistêmica. Nesse paradigma, a questão²⁵ é uma preocupação a respeito de um tema ou tópico que surge como problema no mundo real. Esses problemas são utilizados para modelar um mundo ideal, onde eles não existem e, conseqüentemente, a questão não

²⁴ Solução no sentido dado por Ackoff (1981): solucionar um problema é encontrar a resposta correta, da mesma forma como se soluciona uma equação.

²⁵ No texto original em inglês Hitchins (2007) utiliza o termo **issue**. O autor deste trabalho traduziu esse termo para o português como: **questão**

surge. Identificado um mundo ideal, ele é comparado com o mundo real de uma forma em que as diferenças entre os dois possam ser identificadas e utilizadas em uma agenda de mudanças. A agenda de mudanças é utilizada para sair da situação do problema no mundo real e, de alguma forma ainda não definida, se aproximar de uma situação na qual o problema não ocorre, isto é, se aproximar do modelo do mundo ideal.

Figura 2 – Paradigma de resolução de problemas em geral.

Fonte: Adaptada de Hitchins (2007, pg. 172).

O paradigma de solução de problemas em geral em conjunto com a abordagem sistêmica permite que se lide com a agenda de mudanças de forma adaptativa, isto é, a agenda considera que o sistema possui um dinamismo próprio, resultado das interações entre seus elementos e entre esses elementos e o meio ambiente. Dessa forma, ela considera que as mudanças podem impactar tanto de forma positiva como negativa, identificando possíveis erros e voltando atrás quando necessário.

Hitchins (2007) argumenta que há um outro paradigma de resolução de problemas que é muito utilizado: O paradigma da engenharia de sistemas. Nesse paradigma há duas atividades independentes executadas logo após a definição do problema:

- Concepção das opções de solução.
- Identificação de critérios para solução ideal.

Com os dados dessas duas atividades é identificada a solução ótima, e, a seguir é formulada a estratégia para implementá-la. Esse é um paradigma que propõem encontrar a solução ótima, solucionando o problema segundo argumentado por Ackoff (1981) e apresentado no item anterior deste trabalho.

Os dois paradigmas – solução de problemas em geral e o de engenharia de sistemas – possuem uma grande diferença. Hitchins (2007) destaca que o que primeiro enfatiza o problema, enquanto o segundo enfatiza a solução. Hitchins sugere a combinação dos dois paradigmas para que se tenha um paradigma adequado para um sistema de métodos, que permita a identificação de uma resposta que soluciona o problema, ou que seja boa o suficiente; ou ainda, que permita que o problema seja dissolvido, ou alterado em seus objetivos, uma vez que questões relacionadas aos problemas podem desaparecer. A figura 3 apresenta esse diagrama.

Figura 3 – Combinação entre paradigma de resolução de problemas em geral e o paradigma da engenharia de sistemas para solução de problemas.

Fonte: Adaptado de: Hitchins (2007, p.174).

A combinação de paradigmas proposta por Hitchins possui uma característica que não é herdada de nenhum dos outros dois outros paradigmas. Essa característica é o uso de modelos de referência para auxiliar a gerar opções para resolver a questão. Modelos de referência são muito úteis quando se está à procura de opções para resolução de problemas, e é nesse ponto que a experiência das pessoas envolvidas na resolução de um problema é significativa para o sucesso de gerar opções de resolução; quando mais experiência, mas modelos de referência são conhecidos e podem auxiliar a identificar opções, quer utilizando de forma direta um modelo de referência, quer compondo esses modelos.

2.8.3 Método Soft Rigoroso

Com base na composição de paradigmas apresentada no item anterior, Hitchins (2007) desenvolveu um sistema de métodos denominado: Método Soft Rigoroso (em inglês: Rigorous Soft Method – RSM).

O RSM pode ser aplicado em qualquer contexto e possibilita tanto o entendimento do problema como a proposição de respostas que podem resolver ou solucionar o problema. Ele não é um sistema de métodos completo porque não oferece suporte à implementação das respostas aos problemas. Ele pode ser considerado como primeiro estágio de um sistema de métodos completo, no qual há a composição de uma série de métodos e ferramentas que possibilitam lidar com o problema e identificar potenciais respostas, utilizando uma abordagem sistêmica.

A qualidade das potenciais respostas obtidas é limitada pela qualidade das entradas, em particular pelo conhecimento e visão de quem está explorando a situação do problema e exposto os vários fatores.

Hitchins (2007) destaca que o RSM pode ser usado por uma pessoa ou por um grupo de pessoas, isto é, um grupo de trabalho; ele também argumenta que o RSM organiza e processa a informação, sendo que processar a informação implica na redução da desordem dos dados obtidos sobre o problema, e a transformação desses dados em informação para elaborar a resposta ao problema.

A figura 4 apresenta a forma gráfica do RSM. Pelo diagrama se percebe a característica do RSM ser um sistema de métodos que se auto verifica, ele faz isso de forma cíclica para auxiliar na identificação de respostas que nem sempre solucionam o problema, e nesse caso, resolvem o problema da melhor forma possível naquele momento.

Figura 4 – Modelos conceitual do método soft rigoroso, visão do processo.

Fonte: Adaptada de Hitchins (2007, fig. 7.4 pg. 198).

Os passos do RSM que estão representados na figura 4 representam as seguintes atividades (HITCHINS, 2003, 2007).

- 1) **Nomear a questão e o seu domínio:** o RSM tem início com a identificação de uma questão de um problema, isto é, uma preocupação a respeito de um tema ou tópico. Identifica-se o cenário em que existe o problema, realizando uma descrição da situação atual;
- 2) **Identificar sintomas e fatores da questão:** é realizada a identificação dos sintomas que apontam a natureza da questão. Não é incomum redefinir-se

a suposta natureza de uma questão depois que alguns sintomas tenham sido identificados e explorados. Os fatores a identificar são os aspectos de uma questão que a tornam especial, fora do comum ou única; não são sintomas de disfunção, mas podem contribuir com a situação do problema e devem estar nos modelos a serem construídos. Neste passo são realizadas entrevistas com várias pessoas que tenham conhecimento do domínio do problema, isto é, tenham alguma relação com o problema ou algum interesse. Nesta fase deve-se utilizar métodos consensuais²⁶, que promovem o consenso entre as pessoas sobre os sintomas e fatores da questão; entre esses métodos estão o Brainstorming, a Técnica Grupo Nominal, a Análise e Estruturação de Modelos, as figuras ricas do Método de Sistemas Soft (Soft System Methodology – SSM) de Checkland (CHECKLAND, 1981), e até mesmo a matriz TOWS²⁷ (WEIHRICH, 1982);

- 3) **Identificar os sistemas implícitos:** Cada sintoma implica na existência de pelo menos um sistema implícito na situação do problema. Esses sistemas são abertos, adaptativos e interativos. São sistemas funcionais que devem existir para o sintoma apareça; o sintoma emergente sinaliza que algo está disfuncional em um sistema ou no conjunto que eles formam. É possível utilizar figuras ricas do SSM de Checkland (1981) para identificar esses sistemas;
- 4) **Agregar sistemas implícitos, sintetizando sistemas contenedores:** Os sistemas implícitos são agregados formando conjuntos, um conjunto por sintoma. Cada conjunto é um sistema contenedor implícito. Este estágio do processo permite obter uma nova figura rica de toda a questão. Também é possível identificar possíveis agrupamentos, que podem gerar uma hierarquia de sistemas, destacando problemas relacionados à questão;
- 5) **Compreender a interação e desequilíbrios dos sistemas contenedores:** Avalia-se as interações existentes entre os sistemas contenedores, O

²⁶ O Apêndice D apresenta cinco métodos consensuais que podem ser utilizados no passo 2 do RSM.

²⁷ Matriz TOWS é uma ferramenta de análise situacional que, através da análise do presente, possibilita a formulação de uma estratégia para o futuro (WEIHRICH, 1982). TOWS é a sigla em inglês para Threats, Opportunities, Weakness e Strengths, respectivamente: Ameaças, Oportunidades, Fraquezas e Forças em português.

processo de agregação pode revelar potenciais desequilíbrios nessas interações;

- 6) **Propor tratamento para desequilíbrio entre sistemas contenedores:** Pode ser necessário tratar os desequilíbrios identificados para neutralizar os sintomas. São utilizadas as diferenças entre o mundo ideal o mundo real para propor soluções sociotécnicas aos desequilíbrios encontrados;
- 7) **Avaliar o impacto da proposta em todos os desequilíbrios dos sistemas, contidos e contenedores:** Conceber e testar tratamentos candidatos para verificar se, caso implementados, eles podem eliminar todos os sintomas identificados no passo dois, e todos os desequilíbrios do passo 6. Esses tratamentos também devem ser testados em relação a sua aceitação cultural pelas pessoas envolvidas na situação do problema;
- 8) **Nomear soluções potenciais:** Nomear um ou mais tratamentos sensatos e aceitáveis desde que exista um. É importante destacar que este passo é possível desde que se tenha identificado pelo menos uma solução ou resolução potencial, pois, segundo o próprio autor do método, identificar uma resposta para o problema não é garantido pelo método.

Sendo um método que procura obter o consenso sobre o problema e a resposta potencial para o problema, o RSM possibilita que as pessoas que participam dele se sintam acolhidas pelo processo. Naturalmente, como Hitchins (2007) argumenta, as pessoas que se sentem insatisfeitas com este tipo abordagem são aquelas que não têm interesse em um consenso, são pessoas que desejam sempre impor a sua visão de mundo.

2.9 ENGENHARIA

De acordo com o dicionário Caldas Aulete da língua portuguesa (AULETE, 2009), engenharia é a ciência e técnica das construções civis, da fabricação de máquinas e do aproveitamento dos recursos da natureza em benefício do homem e suas necessidades. Kossiakoff et al. (2011) afirmam que no trabalho da engenharia há uma preocupação com eficiência. Hitchins (2007) argumenta que a engenharia clássica é linear e, geralmente, baseada no Reduccionismo Cartesiano. A engenharia dá ênfase

a funções, formas, estruturas e arquitetura, tendo como objetivo um todo igual à soma das partes.

Durante o século XIX, a engenharia ganhou a estrutura de disciplina científica. Isso corresponde a um desenvolvimento tecnológico onde as ferramentas utilizadas desde o século XIII foram substituídas passo a passo por máquinas, as quais, durante o século XIX, começaram a se conectar a sistemas - sistemas elétricos, sistemas de telégrafo, sistemas ferroviários e a sistemas de transmissão de energia nas fábricas. A engenharia, em seu início, era entendida como uma aplicação do conhecimento das ciências naturais, uma visão que continua a dominar a reconstrução teórica das ciências tecnológicas, pelo menos até o fim da Segunda Grande Guerra (POSER, 1998).

O pós-guerra, quando inicia a Idade dos Sistemas, marca um novo entendimento da engenharia. Os sistemas começam a demandar que engenharia também tenha uma dimensão humana. Auyang (2006) argumenta que em sua ação, o engenheiro necessita ter conhecimentos das ciências naturais, e ter compreensão dos fatores humanos e socioeconômicos envolvidos, indo além das ciências naturais em sua missão de utilidade e serviço para a sociedade; engenheiros trabalham com artefatos tecnológicos e pessoas, possibilitando a harmonia entre a natureza e humanidade.

2.9.1 Engenharia de sistemas

Nos Estados Unidos, durante a Segunda Grande Guerra Mundial, e também no período do pós-guerra e início da guerra fria, houve necessidade de se desenvolver técnicas baseadas em Teoria Geral de Sistemas para a criação de grandes sistemas de defesa. Engenheiros, cientistas e administradores desenvolveram técnicas que se tornaram conhecidas por:

- **Engenharia de Sistemas**, para o projeto e desenvolvimento de sistemas.
- **Pesquisa Operacional**, para o uso de técnicas quantitativas na análise dos sistemas de armamento.
- **Análise de Sistemas**, para comparar e avaliar os projetos propostos.

Essas técnicas foram desenvolvidas porque muitos dos sistemas que estavam sendo criados não podiam ser totalmente compreendidos de acordo com o ensinado nas escolas clássicas de engenharia e de administração (HUGHES, 2005).

Desde a sua formalização, a engenharia de sistemas é uma atividade mais baseada na prática do que em uma disciplina acadêmica, é considerada, mais propriamente, como um método do que como uma disciplina fundada em livros e formalismos. Diferente das disciplinas tradicionais de engenharia, a engenharia de sistemas não segue um conjunto de fenômenos fundamentais baseados em propriedades físicas e relações. Ao invés disso, ela está associada ao conhecimento de forma a se poder orquestrar esses fenômenos e tratar as propriedades emergentes dos sistemas.

Hitchins (2007) afirma que a engenharia de sistemas trata os sistemas tecnológicos considerando-os como dinâmicos, com contextos de interação, potencialmente adaptáveis a outros sistemas em seu ambiente, e capaz de exibir propriedades emergentes, capacidades e comportamentos.

A definição dada por Hitchins (2007) para Engenharia de Sistemas é a adotada pelo autor deste trabalho:

Engenharia de Sistemas é a arte e a ciência de criar soluções como um todo para sistemas complexos. (Hitchins, 2007 p.91, tradução do autor)

Essa não é a única definição existente. Há diversas outras que, apesar apresentarem algumas palavras significativas como: Interdisciplinar, interativo, sócio, técnico e todo, parecem mais com formas de como a engenharia de sistemas deve ser executada, do que uma definição propriamente dita. Por exemplo:

Engenharia de Sistemas é uma abordagem interdisciplinar que possibilita a realização de sistemas de sucesso. (INCOSE, 2006, p. 2.1, tradução do autor).

A função da engenharia de sistemas é *guiar a engenharia de sistemas complexos*. (KOSSIAKOFF et al., 2011, p. 3, tradução do autor).

Engenharia de sistemas é a área tradicional da engenharia que auxilia os engenheiros a compreender como aperfeiçoar um sistema por completo. (STASINOPOULOS, 2009, p. 22, tradução do autor).

Engenharia de sistemas é uma abordagem multidisciplinar para o desenvolvimento de soluções para problemas complexos. (FRIEDENTHAL; MOORE; STEINER, 2008, p. XI, tradução do autor).

Hall (1962) argumenta que a engenharia de sistemas não possui uma definição clara e precisa contida em uma única frase devido as suas múltiplas particularidades. Contudo, ele identifica cinco características do engenheiro de sistemas que Rhodes e Hastings (2004) argumentam estarem presentes até hoje. Engenheiros de sistemas são pessoas que possuem:

- Afinidade com o ponto de vista sistêmico.
- Habilidade para julgamento.
- Criatividade.
- Facilidade para relações humanas.
- Um dom para se expressar.

Parnell; Driscoll e Herdeson (2015) afirmam que o ponto de vista sistêmico, o uso do pensamento sistêmico, diferencia a engenharia de sistemas das outras engenharias. Embora todas as engenharias necessitem encarar os desafios do século XXI através de abordagens sistêmicas, esses autores destacam que é a engenharia de sistemas que permite a elaboração de processos holísticos, lidando com equipes multidisciplinares para elaborar a sequência cognitiva de atividades que suportam o ciclo de vida dos projetos de engenharia.

2.9.2 Engenharia de sistemas e complexidade

Arthur David Hall, engenheiro elétrico dos Laboratórios da Bell Telefones (Bell Telephone Laboratories, em inglês), e primeiro editor da IEEE Transactions on Systems Science and Cybernetics²⁸ em 1965²⁹, lançou em 1962 o primeiro livro significativo sobre Engenharia de Sistemas: A Methodology for Systems Engineering (HALL, 1962). Nesse livro de 1962, Hall argumenta que há uma dificuldade na

²⁸ Disponível em: <http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=06593086>. Acesso em: 12 jun. 2015.

²⁹ Informação disponível em: <http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=04082039>. Acesso em: 12 jun. 2015.

definição do conceito de complexidade, e por isso ele manifesta uma preocupação com o uso de expressões como: "a crescente complexidade dos sistemas". Contudo, Hall prossegue afirmando que é a complexidade dos sistemas que leva a evolução da engenharia de sistemas, mesmo não sendo possível dar uma definição numérica ou científica para o termo. Nesse sentido, Hall argumenta que a complexidade é tratada informalmente, associando-a à quantidade de componentes e às suas relações em um sistema.

Hitchins (2007) argumenta que, mesmo com um entendimento limitado do conceito de emergência em sistemas, é possível tratar a questão da complexidade e sintetizar sistemas com as propriedades emergentes desejadas. A engenharia de sistemas trabalha com padrões de processos em projetos de sistemas que possam resultar em emergência, e assegura que o padrão identificado não perturbe o sistema. Hitchins destaca que isso é possível desde que o engenheiro, ao representar um sistema, não utilize apenas estruturas, funções, formas, etc., ele deve utilizar também uma representação das propriedades emergentes, capacidades e comportamentos.

2.9.3 Engenharia de software

A engenharia de software foi formalizada em 1968 na Conferência de Engenharia de Software³⁰ promovida pelo Comitê de Ciência do Tratado do Atlântico Norte (NATO Science Committee). Essa conferência foi organizada para discutir o que no momento se chamava de crise do software; uma crise que estava relacionada com a dificuldade para se escrever programas de computador de uma forma correta, compreensível e que fosse possível ser verificada.

A engenharia de software é definida nessa conferência para tratar três pontos básicos de todos os projetos de desenvolvimento de sistemas de software:

- Ter processos mais produtivos.
- Ter processos mais previsíveis.
- Ter melhor qualidade.

³⁰ O relatório da conferência está disponível em:
<<http://homepages.cs.ncl.ac.uk/brian.randell/NATO/index.html>>. Acesso em: 12 jun. 2015.

Seguindo o preceito de que os métodos e as técnicas de engenharia são capazes de dar ao homem processos produtivos, com previsibilidade e qualidade, os cientistas e engenheiros presentes na conferência concordaram que uma saída para a crise do software seria a aplicação de conceitos já aceitos na engenharia para o desenvolvimento de software. Entre as definições que surgem desse preceito, Pressman (2010) destaca a definição de engenharia de software proposta por Fritz Bauer:

O estabelecimento e uso de sólidos princípios de engenharia para que se possa obter economicamente um software que seja confiável e que funcione eficientemente em máquinas reais. (PRESSMAN, 2010, p.13, tradução do autor)

Sobre essa definição, Pressman (2010) argumenta que não inclui diretamente a necessidade de satisfação do cliente ou a pontualidade na entrega do software, mas define a linha básica para a construção de softwares: o uso de sólidos conhecimentos de engenharia.

A norma IEEE Std 610.12-190 (IEEE, 1990) apresenta uma definição que Pressman (2010) argumenta ser mais abrangente:

Engenharia de Software: (1) A aplicação de abordagens sistemáticas, disciplinadas e quantificáveis para o desenvolvimento, operação e manutenção de software; ou seja, a aplicação da engenharia ao software. (2) O estudo das abordagens como em (1). (IEEE, 1990, tradução do autor)

Apesar de considerar essa definição, Pressman destaca que, ao mesmo tempo em que é necessária uma disciplina, é também necessária adaptabilidade e agilidade da equipe que trabalha com software.

Sommerville (2011) define que a engenharia de software é uma disciplina da engenharia que se ocupa de todos os aspectos da produção de um sistema de software, desde os estágios iniciais de especificação do sistema até a manutenção desse sistema, depois que entrou em operação. Sommerville argumenta que nessa definição há duas frases importantes:

- **Disciplina de Engenharia:** Essa frase indica que é a engenharia que é responsável pela produção de um sistema de software. A engenharia é uma

disciplina que aplica teoria, métodos e ferramentas nas situações apropriadas e de modo seletivo.

- **Todos os aspectos da produção de software:** Essa frase indica que a engenharia não se preocupa apenas com os processos de desenvolvimento, mas também com o gerenciamento do projeto e o desenvolvimento de ferramentas, métodos e teorias que deem suporte à produção de sistemas de software.

Jacobson e Meyer (2009) afirmam que embora os conceitos de engenharia de software indiquem que ela devesse ser uma disciplina de engenharia com base científica, o que se encontra na prática é uma sucessão de métodos e processos que prometem produtividade, previsibilidade e qualidade. Os autores argumentam que esse cenário é como uma indústria da moda, com uma sucessão de métodos de desenvolvimento e gerenciamento para o desenvolvimento de sistemas de software; eles destacam que não há uma base teórica que permita comparar esses diversos métodos para que se conclua qual é o melhor. Em outro artigo de Ivar Jacobson, desta vez em parceria com Ian Spence, o autor defende a necessidade de se refundar a engenharia de software, reconstruindo-a a partir de uma teoria desenvolvida através do conhecimento e experiência das pessoas, da prática de se desenvolver sistemas de software (JACOBSON; SPENCE, 2009).

Em seu trabalho para refundar a engenharia de software, Ivar Jacobson e seus colegas criaram a iniciativa SEMAT (Software Engineering Methods & Theory), um grupo de pessoas de diferentes países que juntaram suas experiências e conhecimentos em uma cooperação internacional que conforme argumentado por Bertrand Meyer: “[...] adotou uma abordagem sistêmica para identificar um ‘kernel’ dos princípios e práticas da engenharia de software que tem resistido ao teste do tempo e do reconhecimento [...]” (MEYER, 2012, p.xxi, tradução do autor). O trabalho resultante da iniciativa SEMAT para refundar a engenharia de software é chamado Essence, é um padrão da OMG e é apresentado neste trabalho no item 2.11.

2.9.4 Integração entre engenharia de sistemas e de software

Como definido por Sommerville (2011), a engenharia de software é uma disciplina da engenharia que se ocupa de todos os aspectos da produção de um sistema de

software, desde os estágios iniciais de especificação do sistema até a manutenção desse sistema, depois que entrou em operação. Contudo, Barry Boehm destaca que é importante que exista uma integração entre a engenharia de software e a engenharia de sistemas (BOEHM, 2006a, 2006b, 2011). Boehm argumenta que sem uma visão de engenharia de sistemas, duas falhas podem ocorrer no processo de desenvolvimento de sistemas de software:

- Problemas de liderança da equipe.
- Tendência da equipe de desenvolvimento considerar soluções que sejam simples, que não considerando requisitos não funcionais como confiabilidade, disponibilidade e segurança.

Normalmente, uma equipe de desenvolvimento de sistemas de software adota uma forma de trabalho que se concentra em desenvolver a parte técnica do sistema de software, que envolve hardware e software, assumindo que os requisitos refletem uma necessidade da empresa. Uma necessidade que pode estar relacionada tanto com a integração com outros sistemas como com a necessidade de estar de acordo com normas operacionais – novas ou atualizadas – colocadas pelos processos de negócio e/ou pela sociedade. Essa abordagem resulta em um sistema que funciona como a equipe acha que deve ser, mas que na prática, ao tomar contato com o sistema, os usuários o utilizam de uma forma não prevista pelos desenvolvedores, uma utilização que é um reflexo da experiência e cultura desses usuários.

Ao se integrar com a engenharia de software, a engenharia de sistemas pode contribuir para o tratamento dos problemas colocados, pois a engenharia de sistemas utiliza o pensamento sistêmico para tratar desses problemas:

- Utiliza o pensamento sistêmico soft para entender a natureza do problema, procurando experiências práticas e interações com o problema, tentando entendê-lo e propondo soluções que de imediato podem não solucionar o problema, mas fornecem uma resposta que melhora seu entendimento, permitindo o desenvolvimento de uma solução que é a melhor possível naquele momento. Uma abordagem iterativa, que é mais qualitativa do que quantitativa, baseada em experiências anteriores dos membros da equipe.

- Utiliza o pensamento sistêmico hard para apoiar o processo de codificação e implantação do sistema de software, enfatiza os aspectos funcionais, estruturais e arquitetônicos das soluções.

Ao estar integrada com a engenharia de software em todos os aspectos da produção, manutenção e descarte de um sistema de software, a engenharia de sistemas auxilia as equipes que participam desse processo, pois sua essência, como argumentado por Hitchins (2007), está em: Selecionar as partes corretas, agrupá-las, e orquestrá-las para que ocorra uma interação da forma correta.

2.10 ESSÊNCIA DA ENGENHARIA DE SOFTWARE

A Essência da Engenharia de Software é uma resposta a um “Chamado para Ação” lançado pela iniciativa SEMAT (sigla em inglês para: Software Engineering Method and Theory – Teoria e Método da Engenharia de Software, em português).

O SEMAT foi fundado em setembro de 2009 por Ivar Jacobson, Bertrand Meyer, e Richard Soley, que, devido a experiência deles no processo de desenvolvimento de sistemas de software, sentiam que tinha chegado o momento de mudar a forma como as pessoas trabalham para desenvolver sistemas de software. Ao fundar o SEMAT eles escreveram um “Chamado para Ação”, que, em poucas linhas, identifica diversos problemas que eles consideram críticos, e que indicam que existe uma necessidade de se agir para mudar esse cenário. Ao mesmo tempo em que indica os problemas, o chamado também propõe o que precisa ser feito na visão de seus autores (JACOBSON et al., 2012, 2013).

Em fevereiro de 2010, os fundadores do SEMAT transformaram o “Chamado para Ação” em uma declaração de visão (JACOBSON; MEYER; SOLEY, 2010). Seguindo essa visão, a iniciativa SEMAT passou a centrar seus esforços em dois objetivos principais:

- Encontrar um núcleo de elementos da engenharia de software que fosse amplamente acordado.
- Definir uma sólida base teórica.

Encontrar um núcleo, uma base comum, para a engenharia de software foi o primeiro passo. Essa base comum é definida pelo SEMAT como a Essência da Engenharia de Software, um núcleo de elementos essenciais – chamado de Kernel - que são considerados como universais para todos os esforços de desenvolvimento de sistemas de software. Elementos essenciais constituem uma linguagem simples que permite descrever métodos e práticas.

O Kernel foi publicado pela primeira vez em fevereiro de 2012, quando da submissão que o SEMAT realizou em resposta a chamada de propostas feita pela OMG para uma especificação que permitisse a fundação da engenharia de software com base em princípios de práticas ágeis³¹. Em junho de 2014 os diretores da OMG anunciaram que a proposta do SEMAT foi adotada como uma especificação oficial, e o nome da especificação como sendo: Essence – Kernel Language for Software Engineering Methods³².

Essence é ao mesmo tempo um núcleo (o Kernel) e uma linguagem para os métodos de engenharia de software, sendo que o Kernel fornece os elementos essenciais para a definição de práticas de desenvolvimento de sistemas de software. Esses elementos possuem estados que representam o progresso e a saúde do que é essencial no desenvolvimento de sistemas de software; esses elementos também possibilitam a comparação de métodos quando eles são descritos utilizando a linguagem do Essence.

As figuras 5 e 6 apresentam esses elementos essenciais que constituem o Kernel da engenharia de software. Eles são apresentados como “coisas com as quais sempre trabalhamos” e “coisas que sempre fazemos” quando se desenvolve um sistema de software.

Esses elementos essenciais – denominados alphas³³ do Kernel do Essence - formam um conjunto integrado que permite que seja possível identificar o que já foi feito, o que se está fazendo, e o que ainda deve ser feito (JACOBSON et al., 2013).

³¹ A chamada da OMG denomina-se: Foundation for Agile Creation and Enactment of Software Engineering, informações sobre ela podem ser encontradas em: <<http://www.omg.org/cgi-bin/doc?ad/2011-6-26>>. Acesso em: 12 jun. 2015.

³² O anúncio da OMG está disponível no endereço: <http://www.omg.org/news/releases/pr2014/06-16-14.htm>>. Acesso em: 12 jun. 2015.

³³ O autor deste trabalho preferiu manter o termo em inglês ao invés de utilizar a tradução: alfa, pois **alpha** em inglês é um acrônimo para “Abstract-Level Progress Health Attribute”, que em português, em uma tradução direta, significa: Atributo de Nível Abstrato do Progresso e Saúde.

Essence não compete com nenhum método de desenvolvimento de sistemas de software. Também não é uma nova forma de desenvolver sistemas de software. O Essence é agnóstico em relação à abordagem escolhida para o desenvolvimento do sistema de software. Ele auxilia uma equipe a utilizar qualquer método de desenvolvimento de sistemas de software.

Figura 5 – “Coisas com as quais sempre trabalhamos”.

Fonte: Adaptado de Jacobson et al. (2013)

Figura 6 – “Coisas que sempre fazemos”.

Fonte: Adaptado de Jacobson et al. (2013).

2.10.1 Organização do Kernel

O Kernel do Essence é organizado em três áreas de interesse da engenharia de software (JACOBSON et al., 2013):

- **Cliente:** todo desenvolvimento de software envolve pelo menos um cliente que demanda uma necessidade específica. A perspectiva do cliente deve ser considerada no dia a dia de trabalho para garantir que a solução correta seja desenvolvida, esta área de interesse da engenharia de software está relacionada com o uso e valor do sistema de software desenvolvido.
- **Solução:** o objetivo principal do desenvolvimento de software é propor a solução de um problema através de sistema de software que funcione. O domínio de solução possui todos os aspectos relacionados a especificação e desenvolvimento de um sistema de software.
- **Empreitada:** o desenvolvimento de software é uma empreitada que leva tempo e trabalho para atingir a seus objetivos; ela envolve uma equipe de pessoas que realiza o desenvolvimento e afeta diferentes outras pessoas

em uma organização ou até mesmo fora dela. Esta área de preocupação da engenharia de software está relacionada aos aspectos que envolvem a equipe de desenvolvimento e a forma as pessoas que compõem essa equipe efetuam seu trabalho.

Como apresentado na Figura 5, cada uma das três áreas de preocupação da engenharia de software possui um subconjunto dos elementos essenciais presentes em todo o processo de desenvolvimento de sistemas de software. Em conjunto, os elementos das três áreas de preocupação devem ser compreendidos e controlados, como uma forma de se ter produtividade, previsibilidade e qualidade.

2.10.2 Alphas

Os alphas, elementos essenciais presentes em todos os processos de desenvolvimento de sistemas de software, constituem o Kernel do Essence e formam um conjunto integrado que permite que seja possível identificar o que já feito, o que se está fazendo, e o que ainda deve ser feito. Isso é possível porque cada alpha possui um conjunto de estados que representam o progresso daquele alpha no processo de desenvolvimento. Por exemplo, o alpha trabalho possui a sequência de estados: Iniciado – Preparado – Sobre Controle – Concluído – Encerrado. Cada estado de um alpha possui um checklist que especifica o critério que a equipe de desenvolvimento deve identificar para determinar e controlar as ações para o estado atual. Os checklists dos estados também devem ser utilizados pela equipe de desenvolvimento para identificar que ela deve fazer para alcançar cada um dos estados. Conhecer o estado de cada alpha em um processo de desenvolvimento de sistemas de software permite que se identifique o progresso e a saúde da empreitada para alcançar seu final com sucesso (JACOBSON et al., 2013; SEMAT, 2015).

Os sete alphas do Kernel do Essence estão representados na Figura 5 como: “as coisas com as quais sempre trabalhamos”. O conceito de cada alpha é (ESSENCE, 2014):

- **Oportunidade:** conjunto de circunstâncias que motivam o desenvolvimento de um sistema de software ou uma mudança em um sistema já existente.

- **Stakeholders**³⁴: pessoas, grupos ou organizações que afetam ou são afetadas por um sistema de software.
- **Requisitos**: o que um sistema de software deve fazer para tratar a oportunidade e satisfazer os stakeholders.
- **Sistema de Software**: sistema constituído de software, hardware e dados que prove seu valor primário através da execução de software.
- **Equipe**: grupo de pessoas ativamente engajadas no desenvolvimento, manutenção, entrega e suporte de um sistema de software.
- **Trabalho**: atividade envolvendo esforço físico e mental realizada por uma equipe para produzir um sistema de software.
- **Forma de Trabalho**: o conjunto de práticas e ferramentas usadas por uma equipe para guiar e suportar o trabalho que realizam.

Para o uso diário por uma equipe os alphas foram identificados em forma de cartões, para que com um simples conjunto de cartas seja possível identificar o que foi feito, o que está sendo feito e o que ainda resta a ser feito. A figura 7 apresenta os cartões que representam os estados do alpha requisitos.

³⁴ O autor deste trabalho preferiu manter o termo em inglês dada a ausência de uma palavra em português que tenha a mesma semântica.

Figura 7 – Cartões de estados do alpha requisitos.

Fonte: Produção do Autor.

Para a utilização dos cartões, o primeiro passo é colocar os cartões de estados dos alphas sobre uma mesa formando uma linha. A figura 8 apresenta os estados do alpha forma de trabalho em linha, deve-se repetir essa disposição para todos os alphas.

Figura 8 – Cartões de estados do alpha forma de trabalho alinhados.

Fonte: Produção do autor.

Após os cartões de estado dos alphas serem distribuídos em linha pela mesa, deve-se percorrer estado por estado de cada alpha, avaliando qual o estado alcançado até

o momento, então desloca os cartões com os estados alcançados para a esquerda. Os cartões que restam do lado direito indicam os próximos estados a serem alcançados. A figura 9 apresenta esse cenário para três alphas: requisitos, sistema de software e forma de trabalho. Não se utilizaram todos os estados nessa representação para não sobre carregar o conteúdo da figura.

Figura 9 – Exemplo de utilização de cartões para determinar o estado dos alphas.

Fonte: Produção do autor.

Existem outras forma de utilizar os cartões para determinar o estado atual de uma empreitada, essa formas podem ser encontradas em Jacobson et al. (2013), em SEMAT (2015) e na pagina da empresa Ivar Jacobson Internacional³⁵, que também desenvolveu aplicativos³⁶ que auxiliam a verificação dos estados dos alphas.

³⁵ A página web da empresa sobre o Essence está disponível em: <http://www.ivarjacobson.com/Software_Engineering_Method_and_Theory/>. Acesso em: 12 jun. 2015.

³⁶ Os aplicativos desenvolvidos pela Ivar Jacobson International estão disponíveis em: <http://www.ivarjacobson.com/Alpha_State_Explorer_App/>. Acesso em: 12 jun. 2015.

2.11 SISTEMAS SOCIOTÉCNICOS

Após a Segunda Grande Guerra Mundial, os sistemas tecnológicos desenvolvidos pelo homem se tornaram o principal problema para os engenheiros e outros profissionais. Na Idade dos Sistemas a indústria torna-se um sistema que possui uma complexa inter-relação entre pessoas e tecnologias, incluindo hardware, software, dados, ambiente, procedimentos, leis e regras. Administrar essas indústrias é um dos principais desafios da sociedade do pós-guerra (HUGHES, 2005).

Maté e Silva (2005) afirmam que Fred Emery e Eric Trist, pesquisadores em desenvolvimento organizacional do Tavistock Institute of Human Relations³⁷ em Londres, definiram em 1960 o termo sociotécnico; desafiando uma época baseada no determinismo tecnológico da Revolução Industrial que postulava:

- A tecnologia é autônoma.
- As condições individuais e sociais do trabalho humano deviam seguir as estruturas técnicas (ROPOHL, 1999, apud MATÉ; SILVA, 2005).
- Treinamento é um processo que adapta homens às máquinas, logo os homens podem ser facilmente substituídos, se necessário.

O taylorismo é uma consequência do determinismo tecnológico, sendo a produção em massa de Henry Ford o exemplo mais proeminente. Contra esse cenário, Emery e Trist formalizaram uma interdependência e uma relação recíproca entre homens e tecnologia. A interdependência e a relação recíproca são necessárias para que o trabalho realizado pelo homem e os aspectos sociais e técnicos envolvidos estejam em harmonia, possibilitando a humanização do trabalho (TRIST, EMERY, MURRAY, 1993).

Maté e Silva (2005), a partir de definição colocada pelo sítio Computing Cases³⁸, colocam a seguinte definição de sistemas sociotécnicos, que é a adotada pelo autor deste trabalho:

³⁷ A página web do Tavistock Institute está disponível em: < <http://www.tavistock.org> >. Acesso em: 12 jun. 2015.

³⁸ A página web do Computing Cases está disponível em: < <http://www.computingcases.org> >. Acesso em: 12 jun. 2015.

Um sistema sociotécnico é uma inter-relação complexa de pessoas e tecnologia, incluindo hardware, software, dados, ambiente físico³⁹, pessoas, procedimentos, leis e regulamentações. (MATÉ; SILVA, 2005 p. ix, tradução do autor).

A inter-relação entre pessoas e tecnologia para a realização de um trabalho implica em pessoas utilizando artefatos tecnológicos para a realização de uma tarefa, e até mesmo para uma satisfação na realização do trabalho. As pessoas também são criativas na forma de organizar e reorganizar o trabalho que precisam realizar, e usam dessa criatividade para descobrir novas formas de utilizar a tecnologia para poder concluir o trabalho e ter satisfação tanto no uso da tecnologia como na realização do trabalho em si. Esse comportamento das pessoas não é previsto por quem disponibiliza a tecnologia; o comportamento das pessoas e a relação delas com a tecnologia são incontrolláveis, não são previsíveis, são propriedades emergentes do sistema sociotécnico.

Baxter e Sommerville (2011) argumentam que atualmente o termo sociotécnico é utilizado para descrever muitos sistemas, mas que há cinco características chaves sempre presentes em um sistema sociotécnico:

- O sistema deve possuir partes interdependentes.
- O sistema deve adaptar-se aos objetivos do ambiente externo.
- O ambiente interno do sistema possui subsistemas técnicos e sociais que são interdependentes.
- O sistema possui objetivos que podem ser alcançados de mais de uma forma.
- O desempenho do sistema depende da otimização conjunta dos subsistemas técnico e social. Concentrar-se em um deles pode conduzir a degradação do sistema.

Sommerville et al. (2012) argumenta que para retratar que sistemas sociotécnicos envolvem o técnico, o humano e o organizacional, e que esses elementos interagem

³⁹ Ambiente físico: Paredes também influenciam e embasam regras sociais, e suas disposições podem influenciar as formas como a tecnologia é utilizada. A sala de um gerente que possui uma secretária na frente é um exemplo; os escritórios sem paredes é outro. Levar uma tecnologia que assume um ambiente físico para outro diferente pode causar problemas de entendimento no uso dessa tecnologia (COMPUTING CASES, 2009, tradução do autor).

e evoluem, o termo “sistemas sociotécnicos” pode ser substituído por “ecossistemas sociotécnicos”.

2.12 FATORES HUMANOS

Embora o desenvolvimento de sistemas de software seja uma atividade que envolve profissionais com conhecimento técnico, as relações entre pessoas não se restringem às atividades técnicas. As relações entre as pessoas também estão presentes na administração⁴⁰ desse processo; desenvolvedores e administradores são pessoas igualmente importantes nos processos de desenvolvimento.

A importância das pessoas e da consideração de fatores humanos na execução de um projeto, seja de um sistema de software ou não, está presente em diversas publicações (ALBUQUERQUE, 1999; GIL, 2001; VALERIANO, 2001; TURNER; BOEHM, 2003; MARRAS, 2011; CHIAVENATO, 2014). Por exemplo:

Na estratégia de controle, os empregados são vistos como números, como custo, como mais um fator de produção que, para desempenharem, devem ser mandados e controlados. Na estratégia de comprometimento, parte-se da consideração das pessoas como parceiros na produção, nos quais a empresa deve investir para conseguir melhores resultados; baseia-se na crença de que conseguir o comprometimento dos empregados levará ao aumento do desempenho. (ALBUQUERQUE, 1999, p.220)

[...] a gestão de pessoal trata do recurso humano do projeto focalizando a pessoa, como entidade viva, isolada, sob o aspecto de seu progresso profissional e também a equipe, como um conjunto harmonioso devotado ao trabalho cooperativo. (VALERIANO, 2001, p.249).

Um dos resultados mais significativos de nossas análises é a constatação de que enquanto métodos, técnicas administrativas e abordagens que usam de tecnologia são valiosas, os fatores mais críticos para o sucesso são mais propensos a estarem do reino dos fatores que envolvem pessoas. (TURNER; BOEHM, 2003, tradução do autor).

Outro exemplo da importância das pessoas está no próprio conceito de gestão de pessoas⁴¹ que, tanto em relação às pessoas, como em relação à estratégia

⁴⁰ Neste trabalho o termo administração indica ação de tratar dos problemas típicos das organizações, como: recursos financeiros, recursos humanos e recursos patrimoniais (VALERIANO, 2005, p.12).

⁴¹ Informações sobre esse conceito estão disponíveis no Glossário: Gestão de Pessoas.

organizacional, afirma que o comprometimento das pessoas que participam de um projeto é determinante para o sucesso desse projeto (WRIGHT; BOSWELL, 1999). Em diversas organizações, para obter o comprometimento das pessoas, a gestão de pessoas desenvolve estratégias relacionadas a habilidades, motivação, engajamento e resiliência⁴².

Contudo, existem fatores humanos que também devem ser considerados, caso contrário, as pessoas passam a ser consideradas como objetos, um meio, para obter o sucesso dos projetos. As pessoas possuem diferentes desejos, valores e emoções, exemplos de fatores humanos que devem ser considerados para que ocorra o respeito à dignidade humana, para que as pessoas não sejam tratadas como componentes de execução de um projeto.

2.13 CONSIDERAÇÕES FINAIS

As definições e conceitos utilizados neste trabalho possuem uma relação intrínseca com a engenharia de sistemas e de software. Desde a visão sistêmica e as propriedades emergentes, até em como a engenharia de sistemas lida com essa emergência através do pensamento sistêmico hard e soft. Um ponto importante a destacar é a argumentação de Ackoff sobre resolução de problemas: um problema pode ser resolvido, solucionado ou dissolvido.

O sistema sociotécnico, que se forma quando as pessoas iniciam um processo de desenvolvimento de sistemas de software, demanda uma atenção especial ao sistema de atividades humanas que o integra. As pessoas possuem diferentes desejos, valores e emoções, exemplos de fatores humanos que devem ser considerados para que ocorra o respeito à dignidade humana.

⁴² Resiliência neste contexto indica a capacidade de uma pessoa em absorver mudanças em seu ambiente de trabalho, mantendo sua produtividade e qualidade.

3 DESENVOLVIMENTO DE SISTEMAS DE SOFTWARE E PESSOAS

Este capítulo conceitua a empreitada de desenvolvimento de sistemas de software como um sistema sociotécnico. Reconhece o valor da tecnologia presente nesse sistema e o valor dos processos e ferramentas utilizadas, destacando que apesar do valor que esses instrumentos trazem ao processo, as pessoas e as relações entre elas são fundamentais para o sucesso da empreitada.

3.1 CONSIDERAÇÕES INICIAIS

Brooks (1987) argumenta que a quantidade elevada de elementos distintos existentes nos sistemas de software torna o desenvolvimento desses sistemas mais complexo que qualquer outro tipo de construção humana. Como evidência do número de elementos presentes no desenvolvimento de sistemas de software, Clarke e O'Connor (2012) apresentam quarenta fatores e cento e setenta subfatores que estão presentes no contexto do desenvolvimento de sistemas de software.

A complexidade existente nos processos de desenvolvimento de sistemas de software não se deve apenas ao elevado número de elementos. A existência de pessoas durante todo esse processo contribui para a existência dessa complexidade, pois o comportamento delas é influenciado por interesses, desejos, valores e emoções. Weinberg (2011a) comenta que a atividade de programação, existente no desenvolvimento de sistemas de software, não é apenas um comportamento humano, é um comportamento humano complexo.

3.2 PESSOAS, PROCESSOS E FERRAMENTAS – UM SISTEMA SOCIOTÉCNICO

Quando uma equipe é formada para uma empreitada de desenvolvimento de um sistema de software, as pessoas que formam essa equipe devem ser apresentadas umas às outras e conhecer as atividades que elas irão realizar. A Tecnologia da Informação e Comunicação (TIC) fornece ferramentas que favorecem a relação entre as pessoas de uma equipe, de forma que, mesmo quando as pessoas de uma equipe compartilham o mesmo espaço físico, elas utilizam ferramentas tecnológicas para interagir umas com as outras. Essa característica adquire proporções maiores quando

há pessoas na equipe que fazem parte da geração que nasceu e cresceu utilizando computadores, e é fascinada por novas tecnologias (SIMONETTE; SPINA, 2012).

As pessoas de uma equipe de desenvolvimento formam uma rede social. Uma rede na qual as interações entre as pessoas, e as interações entre as pessoas e as ferramentas tecnológicas que suportam o processo de desenvolvimento formam um sistema sociotécnico. Emery (1993) argumenta que a harmonia nas relações entre as pessoas, e entre as pessoas e a tecnologia, permite que as pessoas utilizem todo o potencial das ferramentas tecnológicas disponíveis, e isso permite uma evolução da compreensão pelas pessoas sobre o trabalho que elas realizam.

3.3 PESSOAS E AS RELAÇÕES ENTRE ELAS

O desenvolvimento de sistemas de software é uma atividade presente em inúmeras organizações. Essa atividade é fundamental para o sucesso e a estratégia competitiva dessas organizações. Em vista disso, os processos de desenvolvimento de sistemas de software passam por melhorias constantes para que as organizações tenham mais produtividade, previsibilidade e qualidade no desenvolvimento de seus sistemas de software. Além desses pontos, os processos de desenvolvimento também devem atender de forma rápida as necessidades por novos sistemas e alterações nos sistemas existentes dessas organizações.

Melhorias nos processos de desenvolvimento de sistemas de software podem ser obtidas através de ferramentas oferecidas pela TIC. Contudo, um sistema de software não é desenvolvido apenas a partir de ferramentas, pessoas também são necessárias.

3.3.1 Sistema de atividades humanas

Diferente de outras disciplinas de engenharia, a engenharia de software não é regida por uma ciência que estabelece princípios ou leis que permitem se estabelecer a precisão e a previsibilidade de seus processos e métodos (LEHMAN, 1989, 1991; Jacobson et al. 2012, 2013). Tanto o desenvolvimento de uma primeira versão de um sistema de software, como a evolução de um sistema já existente não é um processo governado por leis da natureza. LEHMAN (1980) argumenta que a primeira implementação e as mudanças em sistemas de software não ocorrem de forma espontânea, pessoas realizam esse trabalho interpretando as necessidades dos

usuários, consumidores, clientes, legislação, diretrizes administrativas, falhas em sistemas atuais, ou, até mesmo, por inspiração de mudar ou fazer algo que julga ser melhor; o julgamento e o raciocínio humano possuem um papel decisivo na execução de um processo de desenvolvimento de sistemas de software.

São pessoas que executam processos de desenvolvimento de sistemas de software. Elas interagem umas com as outras para interpretar as necessidades de quem tem interesse no sistema, para encontrar formas eficientes de transformar necessidades em requisitos funcionais, não funcionais ou restrições⁴³. Também são pessoas que transformam os requisitos em código e em outros componentes necessários para que o sistema atenda aos requisitos, às necessidades e aos propósitos para os quais foi desenvolvido. Essas ações são atividades que constituem um sistema dentro do sistema sociotécnico que é o processo de desenvolvimento de sistemas de software, um sistema de atividades humanas. Ao tratar as pessoas e as interações entre elas como um sistema de atividades humanas, busca-se aumentar a qualidade das interações e conseqüentemente a melhor compreensão do trabalho a ser realizado, de forma que quando do sistema de software estiver pronto, não ocorra uma discrepância entre o que é esperado e o que é percebido pelas pessoas que se relacionam com o sistema.

3.3.2 Processos e ferramentas

As pessoas que participam da empreitada de desenvolvimento de sistemas de software utilizam processos e ferramentas para realizar esse trabalho⁴⁴. Dessa forma, as pessoas e a forma como elas trabalham devem estar em conformidade com os processos e ferramentas utilizados. Um problema surge quando se considera que a forma de trabalho deve estar em conformidade os processos e ferramentas. Essa necessidade torna difícil a acomodação de novas ideias, novos requisitos, novas

⁴³ Robertson e Robertson (2006) argumentam que há três tipos de requisitos nos sistemas de software: requisitos funcionais, que são as coisas que o sistema deve fazer; requisitos não funcionais, que são as qualidades que o sistema deve ter, e as restrições, que são determinações, regras, condições de execução do projeto ou qualquer outra questão que estabeleça limites ao projeto em si ou a seus requisitos funcionais ou não funcionais. Um exemplo é ter que adotar uma determinada tecnologia para construir o sistema, outro é ter uma data fixa – que não pode ser alterada - para conclusão do desenvolvimento.

⁴⁴ Diferente do conceito de trabalho do Essence, aqui trabalho é mais do que a execução de atividades, inclui o gerenciamento da execução.

abordagens, a criatividade no julgamento e o raciocínio presentes na execução de um processo de desenvolvimento de software; é como se os processos e métodos fossem um trilho a ser seguido para o sucesso do projeto, quando de fato ele deveria ser apenas uma trilha.

Processos e ferramentas devem auxiliar as pessoas em seu trabalho, sem sobrecarregar as pessoas com novas tarefas desnecessárias ao desenvolvimento do sistema em si. Além disso, processos e ferramentas não podem dificultar ou impedir o gerenciamento das atividades, eles devem auxiliar e não direcionar as decisões.

3.3.3 Valorizar as pessoas e as relações entre elas

Dar mais valor às pessoas e às relações entre elas não implica em abandonar processos e ferramentas. Processos e ferramentas possuem sua utilidade em um projeto, eles fornecem visibilidade e permitem o uso de métricas e organização do conhecimento. Contudo, processos e ferramentas não podem prejudicar ou deixar de promover a interação entre as pessoas.

A interação entre as pessoas que executam o trabalho de desenvolver sistemas de software é por si só problemática. Essas pessoas normalmente não têm facilidade, e às vezes, nem vontade, de interagir com outras pessoas. Preferem interagir com máquinas. Contudo, a interação é essencial para ocorra comunicação de forma clara, rápida, eficiente e efetiva entre as pessoas. Essa comunicação favorece o surgimento de novas ideias e ajustes nos processos utilizados, o que permite que em conjunto as pessoas identifiquem oportunidades para melhorar tanto os processos como a satisfação em participar da empreitada de desenvolvimento.

Processos e ferramentas e a forma de trabalho de uma equipe mantém uma relação direta. Processos e ferramentas do paradigma ágil favorecem a liberdade para que as pessoas escolham como trabalhar com os problemas que elas precisam tratar, favorecendo que se tenha um ambiente como o de uma oficina de artesanato, onde se espera que cada pessoa faça o que é necessário para o sucesso da empreitada; ao passo que, se for adotado o paradigma orientado a um plano, se favorece um ambiente no qual as pessoas se sentem mais confortáveis e autorizadas a trabalhar quando há políticas claras e procedimentos que definem corretamente o papel de cada uma delas no processo (BOEHM; TURNER, 2004).

Como argumentado por Weinberg (2011b, posição 155, tradução do autor): “[...] há sempre um problema. [...] Não importa como ele se apresenta à primeira vista, ele é sempre um problema de pessoas”.

3.4 SISTEMA DE ATIVIDADES HUMANAS - PROBLEMAS DE PESSOAS

A equipe que trabalha no processo de desenvolvimento de sistemas de software é constituída por pessoas, sendo que essas pessoas e as interações entre elas constituem um sistema de atividades humanas que faz parte do sistema sociotécnico que é o processo de desenvolvimento de sistemas de software. As argumentações de Morin, Ciurana e Motta (2009, p. 51) de que “[...] se existe um âmbito ao qual corresponde por antonomásia⁴⁵ o qualitativo “complexo”, esse é o mundo social e humano”, em conjunto com as afirmações de Petraglia (2006, p.23): “[...] complexidade indica que tudo se liga a tudo, numa rede relacional e interdependente [...] Ao mesmo tempo em que o indivíduo é autônomo, é dependente, numa circularidade que o singulariza e o distingue, simultaneamente [...]”, permitem caracterizar o sistema de atividades humanas como complexo.

3.5 CONSIDERAÇÕES FINAIS

A empreitada de desenvolvimento de sistemas de software é complexa devido às diversas interações que ocorrem: a interação entre as pessoas que participam da empreitada, a interação entre essas pessoas e a tecnologia que elas utilizam para realizar a empreitada e a interação entre essas pessoas e o ambiente em que elas vivem e em que realizam a empreitada. Para lidar com essa complexidade é necessário que se compreenda as propriedades emergentes do sistema de atividades humanas que possibilita a transformação de requisitos em código e em outros componentes necessários para que o sistema de software atenda as necessidades e aos propósitos para os quais foi desenvolvido.

⁴⁵ Antonomásia é uma figura de linguagem que consiste na substituição do nome próprio por um nome comum ou por uma expressão que informa seu significado (AULETE, 2009).

4 FATORES HUMANOS NA ENGENHARIA DE SOFTWARE

O capítulo quatro apresenta o sistema de métodos utilizado para obter um conjunto de práticas que possibilita a consideração de fatores humanos na engenharia de software. A seguir, apresenta a extensão da especificação Essence da OMG para implementar a consideração de fatores humanos na engenharia de software.

4.1 CONSIDERAÇÕES INICIAIS

O primeiro passo, para obter um conjunto de práticas que possibilita a consideração de fatores humanos na engenharia de software, é reconhecer quais são os fatores humanos que devem ser considerados. O segundo passo é identificar práticas que permitam considerar esses fatores no pensar e agir existentes nos processos de desenvolvimento de sistemas software. Contudo, seria um ato de violência contra as equipes que desenvolvem sistemas de software se, em nome da consideração de fatores humanos na engenharia de software, o autor deste trabalho determinasse quais fatores humanos devem ou não ser considerados.

Também não seria correto identificar esses fatores através de um cenário que considerasse uma única organização, pois diferentes organizações possuem diferentes culturas organizacionais⁴⁶.

Para identificar os fatores humanos, e as práticas que permitem considerar esses fatores, o autor utiliza o RSM (item 2.8.3), um sistema de métodos que promove o consenso entre as pessoas; uma condição necessária visto que este trabalho demanda o levantamento e validação das informações com pessoas de diferentes organizações, de diferentes níveis hierárquicos, e que trabalham com diferentes processos de desenvolvimento de sistemas de software. Além disso, o RSM é um sistema de métodos que, dentro do ciclo de vida de um sistema, fornece informações para todas as fases que se seguem ao levantamento de requisitos (SIMONETTE, 2010).

⁴⁶ Schein (1984) argumenta que cultura organizacional é o padrão de premissas básicas que um grupo de pessoas inventa, descobre ou desenvolve no processo de aprender a lidar com seus problemas de (i) adaptação do grupo com o mundo externo, e (ii) integração interna. Quando frente a esses problemas as premissas funcionam suficientemente bem para serem consideradas válidas, ela é ensinada aos novos membros do grupo como a forma correta de perceber, pensar e sentir em relação aos problemas.

4.2 PERGUNTAS GUIAS DO PROCESSO DE ANÁLISE

As perguntas guias são utilizadas para iniciar a conversa com os grupos de pessoas que participam das fases do RSM. O objetivo é estimular as pessoas para que elas comentem sobre o conhecimento que possuem em relação a fatores humanos na engenharia de software. Além disso, as respostas a essas perguntas permitem que se alinhem os conceitos e a forma de pensar, para que todos comentem o que sabem e se identifique o que pode ser feito para esclarecer pontos em abertos.

4.2.1 Processo de dinâmica com os grupos

O processo utilizado na condução da dinâmica com os grupos segue o representado na Figura 10; no qual a organização do conhecimento, e a ética⁴⁷ na condução desse processo, guiam o pensamento tanto sobre o conteúdo como sobre as mudanças que sejam necessárias na condução da dinâmica. Os quatro grandes domínios de ação sobre a dinâmica (representados como elipses na figura 10) estão em uma relação recursiva, de forma a oferecer interpretações mais completas.

Figura 10 – Processo de dinâmica com os grupos de pessoas.

Fonte: Produção do autor.

⁴⁷ Cortella e Barros Filho (2014, p. 24) argumentam que ética é um conjunto de valores e princípios que guia a conduta das pessoas.

4.2.2 Perguntas iniciais

A princípio, a pergunta imediata que pode ser feita é “Quais fatores humanos devemos considerar na engenharia de software?”. Contudo, essa pergunta é apropriada apenas quando o grupo de pessoas já convive sobre razoável estabilidade, isto é, o conceito de fatores humanos é entendido e compartilhado por todos. Quando todos convivem em um ambiente de estabilidade, pequenos ajustes e melhorias na forma de trabalho podem ser realizados para manter as pessoas em um ambiente estável, implementando esses ajustes de forma gradual e acompanhando as lentas mudanças no ambiente. Como esta pesquisa envolve pessoas de diferentes organizações, de diferentes níveis hierárquicos e de diferentes escolas de formação superior, o conceito de fatores humanos não é disseminado entre todos, não há um entendimento compartilhado por todos. Não há um ambiente de estabilidade comum a todos. Neste caso, são necessárias outras perguntas para que se tenha um contexto inicial a ser explorado.

As perguntas utilizadas no início das dinâmicas deste trabalho foram as seguintes:

- Qual é a natureza e quais são as características do dia a dia do seu ambiente de trabalho frente ao dinamismo que os negócios possuem?
- Quais são as implicações que isso traz para a consideração de fatores humanos no processo de desenvolvimento de sistemas de software?
- Qual deve ser o papel que a gerência (liderança) ou grupo deve ter frente aos fatores humanos?
- Considerando as suas crenças e valores, que imagem você tem sobre a questão dos fatores humanos na engenharia de software?

Esse grupo de questões iniciais, elaboradas a partir da experiência deste autor e princípios de pensamento sistêmico, permitem que as pessoas pensem em suas repostas considerando os valores e aspirações delas próprias, são repostas baseadas na experiência delas. Isso torna explícito os propósitos sobre a consideração de fatores humanos, e permite a criação e validação de interpretações na busca do que seria ideal.

4.3 A QUESTÃO E SEU DOMÍNIO

O problema a ser tratado é identificar e considerar fatores humanos na engenharia de software.

4.3.1 Questão

O desenvolvimento de sistemas de software é uma atividade humana, como consequência, deve haver fatores humanos presentes nessa atividade.

A preocupação⁴⁸ sobre fatores humanos na engenharia de software é determinar quais são esses fatores, e, posteriormente, identificar práticas que permitam a consideração desses fatores no pensar e agir existentes nos processos de desenvolvimento de sistemas de software.

4.3.2 Domínio

O desenvolvimento de sistemas de software é uma atividade presente em inúmeras organizações. São pessoas que executam processos de desenvolvimento de sistemas de software. Elas interagem umas com as outras para interpretar as necessidades de quem tem interesse no sistema, para encontrar formas eficientes de transformar necessidades em requisitos funcionais, não funcionais ou restrições. Também são pessoas que transformam os requisitos em código e em outros componentes necessários para que o sistema de software atenda aos requisitos, às necessidades e aos propósitos para o qual foi desenvolvido. Essas ações são atividades que constituem um sistema dentro do sistema sociotécnico que é o processo de desenvolvimento de sistemas de software, um sistema de atividades humanas.

A complexidade existente nos processos de desenvolvimento de sistemas de software não se deve apenas ao elevado número de elementos⁴⁹. A existência de pessoas durante todo o ciclo de vida de um sistema de software contribui para a existência da

⁴⁸ Questão é uma preocupação a respeito de um tema ou tópico que surge como problemas no mundo real (item 2.8.2).

⁴⁹ Como evidência do número de elementos presentes no desenvolvimento de sistema de software, Clarke e O'Connor (2012) apresentam quarenta fatores e cento e setenta subfatores que estão presentes no contexto do desenvolvimento de sistemas de software.

complexidade, pois o comportamento delas é influenciado por interesses, desejos, valores e emoções.

4.4 SINTOMAS E FATORES DA QUESTÃO

Para identificar fatores humanos na engenharia de software realizaram-se reuniões com diversos grupos de pessoas de diferentes organizações, de diferentes níveis de experiência profissional⁵⁰ e de diferentes níveis hierárquicos.

4.4.1 Sistema sociotécnico

O sistema sociotécnico que se forma quando as pessoas iniciam uma empreitada de engenharia de software demanda uma atenção especial aos sistemas de atividades humanas que o integra, para que não se desprezem os fatores humanos existentes. A tecnologia fornece ferramentas que favorecem a relação entre as pessoas de uma equipe. Contudo, para que o potencial oferecido pela tecnologia seja aproveitado, é necessário que exista harmonia:

- Nas relações entre as pessoas que realizam a empreitada.
- Nas relações essas pessoas e a tecnologia que elas utilizam para realizar a empreitada.
- Nas relações entre essas pessoas e o ambiente em que elas vivem e realizam a empreitada.

Essas múltiplas relações, em conjunto com (i) o número de elementos presentes no ambiente de desenvolvimento de software e (ii) a imprevisibilidade característica do ser humano⁵¹, resultam na complexidade do sistema sociotécnico que tem o propósito de realizar a empreitada de engenharia de software.

⁵⁰ Entende-se por diferentes níveis de experiência profissional a classificação que indica a maturidade profissional: trainee, júnior, pleno, sênior, etc.

⁵¹ Barros Filho e Pompeu (2015) argumentam que os desejos das pessoas mudam a todo instante, que as pessoas são inconstantes. Barabási (2011, pg.8, tradução do autor) afirma: “Nós podemos prever onde um elétron estará, nós podemos ligar ou desligar um gene e nós podemos até enviar um robô para Marte, mas não temos como explicar ou prever fenômenos que deveríamos conhecer, as ações de nossos companheiros humanos”.

4.4.2 Fatores humanos

Considerar fatores humanos na engenharia de software é uma afirmação vaga. Fatores humanos na engenharia podem estar relacionados a falhas em projeto devido ao erro humano, a projetos centrados nos usuários, a fatores relacionados à gestão de pessoas durante uma empreitada de engenharia e também podem estar relacionados à consideração do humano nessa mesma empreitada.

A natureza da questão relacionada ao problema é a consideração do humano na empreitada de engenharia de software. Contudo, a natureza da questão ainda não deixa claro quais são os fatores humanos, quais são os fatores que tornam a questão única; deve-se deixar claro que não se está abordando os sintomas da disfunção, isto é, da não consideração do humano na empreitada da engenharia de software, mas sim os fatores que podem contribuir para a consideração do humano e devem estar presentes na empreitada de engenharia de software.

4.4.3 Identificação dos fatores humanos

Para identificar os fatores humanos relevantes para a questão do problema, foram realizadas duas sessões de Brainstorming com profissionais da área de Tecnologia da Informação (TI) de diversas organizações, ambas sessões tiveram participação de 24 pessoas. Como resultado dessas sessões foi produzida a figura rica apresentada na figura 11; para que o grupo iniciasse a discussão, foram apresentadas as quatro perguntas comentadas no item 4.2.2 deste trabalho.

As perguntas foram apresentadas uma a uma, e solicitado que cada um guardasse para si a sua resposta, até que houvesse a abertura para exposição das ideias e discussão. Após a última pergunta, deu-se início a exposição de ideias. E logo foi necessária uma intervenção, pois as primeiras ideias colocadas eram reclamações sobre o dia a dia; não traziam ideias que respondessem a alguma das questões colocadas inicialmente.

Foi solicitada uma pausa na exposição de ideias e que cada um refletisse de acordo com suas crenças e valores sobre o que seria ideal de ocorrer no dia a dia de trabalho para que fatores humanos fossem considerados; parafraseando Ackoff (1997, p.420): “para que não houvesse a desumanização do trabalho”.

Retomada a exposição de ideias, as pessoas começaram a debater sobre ambiente de trabalho, o que envolveu debates sobre ter ou não lugares com sofás e almofadas, ter ou não restaurantes com grandes chefes de cozinha na empresa, máquinas de refrigerantes e salgadinhos grátis, além de regras mais flexíveis que promovessem a junção de diversão e trabalho. Por fim, o grupo caminhou para um consenso: fatores humanos no trabalho diário de desenvolver e manter software são aqueles associados a “ter vontade de ir para o trabalho”. Essa ideia evoluiu durante a elaboração da figura rica (figura 11) quando o grupo começou a identificar fatores que, quando presentes em uma empreitada, proporcionam satisfação pessoal em participar dessa empreitada; e fatores que a presença, ou ausência, proporcionam insatisfação ao participar de uma empreitada.

Figura 11 – Figura rica.

Fonte: Produção do autor. Elaborada através de dinâmicas de grupo.

Os fatores citados por esse grupo como fatores que proporcionam satisfação ao participar da empreitada são:

- Presença de desafios (desafio).
- É dada autonomia (autonomia).
- Existe união entre as pessoas da equipe (união).
- Oportunidade de aprender (aprendizado).
- Saber o que se espera de cada pessoa da equipe (expectativa).
- Reconhecimento financeiro pelo trabalho realizado (bônus).
- Reconhecimento pelos superiores e pela organização (reconhecimento).
- O salário é correto – justo - em relação às responsabilidades (reconhecimento).

Os fatores citados pelo grupo de pessoas como os que geram insatisfação ao participarem da empreitada são:

- Trocas de pessoas que compõem a equipe durante a empreitada (mudança).
- Não ter desafios (desafio).
- Expectativas quanto ao trabalho não são claras (expectativa).
- Ausência de reconhecimento (reconhecimento).
- Restrições técnicas (restrição);
- Mudanças de escopo e requisitos mal definidos (mudança).
- Políticas individuais, interesses individuais relacionados a reconhecimento e/ou bônus devido ao sucesso ou fracasso da empreitada (interesse).

Em ambos os casos acima, a palavra entre parênteses sintetiza a frase.

Um sintoma que emerge quando se analisa os fatores identificados é que expectativa, reconhecimento e desafio aparecem tanto como fatores relacionados à satisfação, como nos fatores relacionados a insatisfação. Esse comportamento pode ser entendido pelo fato de que quanto mais clara é a expectativa que se tem sobre uma pessoa, mais satisfeita ela fica, ou, o contrário, quanto menos se falar sobre o que se espera de uma pessoa, menos satisfeita ela ficará. Em relação aos outros dois fatores: reconhecimento e desafio, a relação deles com a satisfação é proporcional, isto é,

quanto mais eles estiverem presentes, maior é a satisfação. A figura 12 apresenta um diagrama em linguagem sistêmica⁵² com a interpretação da relação dos fatores com a satisfação, esse diagrama foi elaborado após a conversa com o segundo grupo de pessoas, relatada no item 4.4.4. O diagrama está colocado neste item porque ele complementa os dados obtidos na elaboração da figura rica.

Figura 12 – Representação dos fatores relacionados a satisfação utilizando linguagem sistêmica.

Fonte: Produção do autor. Diagrama em linguagem sistêmica com a interpretação da relação dos fatores com a satisfação.

Através da linguagem sistêmica, a figura 12 mostra que os fatores: autonomia, união, aprendizado e bônus possuem uma relação de causa e efeito com a satisfação, a presença deles leva à satisfação; os fatores mudanças, restrições, expectativas não declaradas e políticas individuais (interesses) possuem um efeito de proporcionalidade inversa sobre a satisfação, quanto maior a presença desses fatores, menor é a satisfação; os fatores desafio, reconhecimento e expectativas claras possuem um efeito de proporcionalidade direta sobre a satisfação, quanto mais

⁵² A linguagem sistêmica é um dos instrumentos utilizados para se colocar em prática conceitos sistêmicos. Mas informações podem ser encontradas em Andrade et al. (2006).

eles estiverem presentes, maior a satisfação, ao passo que quanto menos estiverem presentes, menor é a satisfação.

4.4.4 Oportunidades, Ameaças, Pontos Fortes e Pontos Fracos

Um segundo grupo de pessoas participou de duas reuniões para elaborar a matriz TOWS⁵³. O objetivo desta atividade é identificar oportunidades e ameaças (fatores externos a uma organização) e também pontos fortes e pontos fracos (fatores internos a uma organização) para considerar fatores humanos na engenharia de software.

O grupo foi composto por 7 pessoas, em comum entre elas o fato de serem todas seniores, trabalharem com engenharia de software há mais de 15 anos e atualmente ocuparem posições de liderança em suas organizações. As questões iniciais (item 4.2.2) foram apresentadas e foi solicitado que os participantes aguardassem para se posicionarem. A seguir, foi explicado a eles como foi montada a figura rica (figura 11) e teve início um breve debate sobre as considerações de cada pessoa do grupo em relação às suas respostas para as quatro perguntas iniciais e em relação à figura rica. Os comentários sobre a figura rica reforçaram os pontos já representados na figura, com a diferença que destacaram a presença de bônus e reconhecimento como preocupação para todas as pessoas relacionadas a TI; o grupo interpretou isso como “trabalhar em função do bônus e do reconhecimento”.

Após as considerações sobre “bônus e reconhecimento” serem ou não apenas instrumentos de motivação e engajamento, e se as políticas de bônus das organizações deterioram ou não o ambiente de trabalho, o grupo chegou ao consenso de que é um direito de cada uma das pessoas envolvidas na empreitada trabalhar em função de bônus e reconhecimento, contudo elas devem exercer essa opção de forma ética⁵⁴.

⁵³ Matriz TOWS é uma ferramenta de análise situacional que, através da análise do presente, possibilita a formulação de uma estratégia para o futuro (WEIHRICH, 1982). TOWS é a sigla em inglês para Threats, Opportunities, Weakness e Strengths, respectivamente: Ameaças, Oportunidades, Fraquezas e Forças em português.

⁵⁴ Durante as entrevistas e dinâmicas realizadas, todas as pessoas que participaram se referiram a ética como um adjetivo, um indicador de qualidade de uma ação. Foi comum o autor escutar expressões como: “de forma ética”. As pessoas ouvidas davam a ética um sentido de “busca do certo”, ou “fazer o certo”, associando com o respeito às pessoas, se comportar de maneira adequada. No decorrer das atividades com essas pessoas não houve intenção de corrigi-las, apresentando a definição de ética adotada neste trabalho. Essa decisão foi tomada pois o objetivo das atividades realizadas com essas pessoas era identificar os significados que são exteriores a elas.

O grupo também comentou sobre a expectativa, fator que aparece como satisfação e insatisfação. O grupo concluiu que a expectativa está relacionada à comunicação entre pessoas. As pessoas têm a expectativa de serem reconhecidas pelo trabalho que realizam, contudo, quando o critério para avaliação do trabalho realizado não é comunicado com clareza, ou até mesmo não é comunicado, a expectativa das pessoas pelo reconhecimento – financeiro ou não – não é atendida. No outro extremo, quando há comunicação correta, isto é, as pessoas sabem o que a organização ou liderança espera que elas realizem, e sabem como serão reconhecidas se realizarem o esperado, a expectativa de cada pessoa é atendida e as pessoas ficam satisfeitas. O grupo ressaltou que o reconhecimento de um trabalho realizado muitas vezes é subjetivo, e pode considerar a forma como uma pessoa realizou um trabalho, e não apenas o resultado. Este conjunto de comentários levou a construção da Figura 12, que apresenta a relação entre os fatores identificados através da construção da figura rica e a satisfação em trabalhar em uma empreitada de desenvolvimento de sistemas de software.

A seguir, foi solicitado ao grupo que identificasse os pontos fortes, fracos, oportunidades e ameaças em relação a se considerar fatores humanos na engenharia de software. Os pontos identificados estão representados na tabela 1, eles são os elementos para elaboração da matriz TOWS.

Tabela 1 – Primeira versão dos elementos para construção da matriz TOWS

FATORES HUMANOS NA ENGENHARIA DE SOFTWARE		
AJUDA		ATRAPALHA
ORIGEM	EXTERNA (ambiente)	OPORTUNIDADES O1. Ruptura tecnológica, como a computação em nuvem e aplicativos para dispositivos móveis. O2. A TI como agente de invocação nas empresas. O3. Existência de diversos métodos e práticas para a engenharia de software.
	INTERNA (organização)	AMEAÇAS A1. Oscilações econômicas – influenciam na realização de projetos e no cumprimento de metas. A2. Restrições ao acesso à internet, diversas organizações estão restringindo o acesso à internet no ambiente de trabalho, incluindo fóruns tecnológicos (acesso a informação). A3. Políticas de uso de dispositivo móvel particular no ambiente de trabalho.
		PONTOS FORTES F1. Apoio da área de recursos humanos da organização. F2. TI como agente de inovação nas empresas. F3. Estrutura organizacional simples. F4. Existência de comunicação interna rápida e efetiva. F5. Avaliação constante dos projetos. F6. Disponibilidade de recursos tecnológicos. F7. Equipe consistente – grupo de pessoas que compartilham as mesmas habilidades e demonstram confiança e respeito mútuo. F8. Auto-organização: a equipe se organiza para o trabalho a ser feito, a equipe organiza o processo que melhor se adequa ao ambiente, a equipe organiza o cronograma para melhor cumprir a entrega do incremento de software.
		PONTOS FRACOS D1. Ausência da área de recursos humanos. D2. Assumir compromissos sem consultar a equipe. A equipe se desmotiva quando há uma pessoa que toma decisões sem consultá-la. D3. Estrutura organizacional com diversos níveis e intersecções em relação às tomadas de decisão. D4. Diferente capacitação (skills) das pessoas, diferentes níveis de conhecimento tecnológico. D5. Pressões para entregar o sistema em prazos cada vez menores. D6. Burocracia interna. D7. Limitação de recursos financeiros. D8. Lideranças não dispõem de tempo para ouvir as pessoas da equipe. D9. Não há capacitação das pessoas técnicas que passam a assumir posições de liderança. D10. Ausência de comunicação interna rápida e efetiva. D11. Avaliação dos projetos apenas na entrega. D12. Não há disponibilidade de recursos tecnológicos. D13. Equipe com grande número de pessoas recém-formadas, ou sem treinamento necessário. D14. Problema de integração entre áreas. D15. Cultura de horas extras. D16. Resistência ao uso de processos e métodos padronizados. D17. Relação de posse do desenvolvedor pelo seu código.

Fonte: Produção do autor.

Concluída a identificação dos elementos para a matriz TOWS, o grupo se preocupou com o a significativa quantidade de pontos fracos que podem atrapalhar a consideração de fatores humanos na empreitada de engenharia de software (tabela 1). Decidiu-se rever cada um dos elementos identificados para a matriz TOWS, considerando se elemento contribui ou não para tornar a empreitada de software mais humana.

Os resultados dessa revisão foram os seguintes:

- **Oportunidades**

Decidiu-se retirar o item:

O2. A TI como agente de inovação nas empresas.

A razão da retirada desse elemento é que apesar de “TI como agente de inovação” ser consenso entre as pessoas, esse papel da área de TI nas organizações depende mais da organização em si, do que do ambiente em que a organização está inserida. Contudo, esse elemento é um ponto forte de uma organização, pois traz desafios constantes para a equipe de desenvolvimento de software, e desafio é um item presente e relevante na para a satisfação das pessoas.

- **Ameaças**

Decidiu-se retirar os itens:

A2. Restrições ao acesso à internet: diversas organizações estão restringindo o acesso à internet no ambiente de trabalho, incluindo fóruns tecnológicos (acesso a informação).

A3. Políticas de uso de dispositivo móvel particular no ambiente de trabalho.

Ao revisar, o grupo acordou que esses dois elementos não possuem relação direta com a empreitada de engenharia de software, apenas tornam o ambiente mais agradável para quem depende de acesso à internet.

- **Pontos Fortes**

Decidiu-se retirar os itens:

F1. Apoio da área de recursos humanos da organização.

F3. Estrutura organizacional simples.

F8. Auto-organização: a equipe se organiza para o trabalho a ser feito, a equipe organiza o processo que melhor se adequa ao ambiente, a equipe organiza o cronograma para melhor cumprir a entrega do incremento de software.

Ao revisar os elementos, o grupo destacou que tanto “apoio” quanto “ausência” da área de recursos humanos apareceu como ponto forte e como ponto fraco, isto é, tanto pode ajudar como atrapalhar a consideração de fatores humanos na empreitada de engenharia de software. Algumas pessoas do grupo destacaram que a presença da área de recursos humanos tem relação com qualidade de vida no trabalho, com foco em proporcionar produtividade e previsibilidade; essa presença não tem preocupações com uma vida de qualidade, com as pessoas respeitadas em seus aspectos humanos. Além disso, destacaram que a presença ou ausência da área de recursos humanos junto às equipes de TI é muito mais uma questão de cultura organizacional do que propriamente um fator que pode ser julgado como pertencente a uma empreitada de engenharia de software. O grupo decidiu por retirar esse elemento tanto dos pontos fortes, quanto dos pontos fracos.

Em relação ao **F3**, o grupo avaliou que ele está também presente no **F4-Estrutura de comunicação rápida e efetiva**, pois estruturas organizacionais simples facilitam a comunicação.

Em relação ao item **F8**, o grupo avaliou que seria melhor retirá-lo, pois ele é específico de uma abordagem ágil, não é uma condição necessária para a consideração de fatores humanos.

- **Pontos Fracos**

Decidiu-se retirar os itens:

D1. Ausência da área de recursos humanos.

D13. Equipe com grande número de pessoas recém-formadas, ou sem treinamento necessário.

D17. Relação de posse do desenvolvedor pelo seu código.

Ao revisar os elementos, o grupo decidiu retirar o elemento **D1**, como já explicado. O elemento **D13** foi retirado porque se considerou que ele já está subentendido no elemento **D4 – Diferença de capacitação**. O grupo também decidiu que o elemento **D17** é uma condição que tem que ser considerada pela liderança da equipe, mas não é algo com impacto direto em se considerar fatores humanos na empreitada.

O fato de comunicação interna aparecer tanto como ponto fraco, como ponto forte também foi comentado. Contudo, diferente do apoio ou presença da área de recursos humanos, decidiu-se que este elemento deve estar presente tanto como ponto fraco como ponto forte. O grupo considerou de extrema relevância a comunicação, não apenas a comunicação escrita trocada entre as pessoas na forma de e-mails, relatórios ou memorandos, mas principalmente pela conversa informal e formal (reuniões) com as equipes.

Durante a revisão, o grupo decidiu por reescrever dois elementos: **D2** e **D14**. A nova versão ficou:

D2. Assumir compromissos sem consultar a equipe.

D14. Problema de integração entre áreas de TI.

A seguir, a tabela 2 apresenta os elementos atualizados para a matriz TOWS.

Tabela 2 – Segunda versão dos elementos para construção da matriz TOWS.

FATORES HUMANOS NA ENGENHARIA DE SOFTWARE		
ORIGEM	AJUDA	ATRAPALHA
	OPORTUNIDADES	AMEAÇAS
EXTERNA (ambiente)	<p>O1. Ruptura tecnológica, como a computação em nuvem e aplicativos para dispositivos móveis.</p> <p>O2. Existência de diversos métodos e práticas para a engenharia de software.</p>	<p>A1. Oscilações econômicas – influenciam na realização de projetos e no cumprimento de metas.</p>
INTERNA (organização)	<p>PONTOS FORTES</p> <p>F1. TI como agente de inovação nas empresas.</p> <p>F2. Existência de comunicação interna rápida e efetiva.</p> <p>F3. Avaliação constante dos projetos.</p> <p>F4. Disponibilidade de recursos tecnológicos.</p> <p>F5. Equipe consistente – grupo de pessoas que compartilham as mesmas habilidades e demonstram confiança e respeito mútuo.</p>	<p>PONTOS FRACOS</p> <p>D1. Assumir compromissos sem consultar a equipe.</p> <p>D2. Estrutura organizacional com diversos níveis e interseções em relação às tomadas de decisão.</p> <p>D3. Diferente capacitação (skills) das pessoas, diferentes níveis de conhecimento tecnológico.</p> <p>D4. Pressões para entregar o sistema em prazos cada vez menores.</p> <p>D5. Burocracia interna.</p> <p>D6. Limitação de recursos financeiros.</p> <p>D7. Lideranças não dispõem de tempo para ouvir as pessoas da equipe.</p> <p>D8. Não há capacitação das pessoas técnicas que passam a assumir posições de liderança.</p> <p>D9. Ausência de comunicação interna rápida e efetiva.</p> <p>D10. Avaliação dos projetos apenas na entrega.</p> <p>D11. Não há disponibilidade de recursos tecnológicos.</p> <p>D12. Problema de integração entre áreas de TI.</p> <p>D13. Cultura de horas extras.</p> <p>D14. Resistência ao uso de processos e métodos padronizados.</p>

Fonte: Produção do autor

Esgotado o tempo que as pessoas dispunham para esta reunião, foi agendada uma segunda reunião para estabelecer as relações entre os elementos identificados para a matriz TOWS (tabela 3). Essa segunda reunião teve início com a distribuição da tabela 2 para as pessoas do grupo e a solicitação de que respondessem as seguintes perguntas:

- **Pontos Fortes X Oportunidades:** Como usar os pontos fortes identificados para aproveitar as oportunidades existentes?
- **Pontos Fortes X Ameaças:** Como usar os pontos fortes identificados para se proteger das ameaças?
- **Pontos Fracos X Oportunidades:** Como superar os pontos fracos identificados para aproveitar as oportunidades?
- **Pontos Fracos X Ameaças:** O que fazer para superar os pontos fracos identificados que não fazem frente às ameaças?

A tabela 3 apresenta a matriz TOWS obtida. Na matriz foi colocada apenas a essência das ideias, a seguir é apresentado o relacionamento entre os elementos com mais detalhes.

Tabela 3 – Matriz TOWS

Matriz TOWS		Fatores Internos	
Fatores Externos		PONTOS FORTES F1. TI como agente de inovação nas empresas. F2. Existência de comunicação interna rápida e efetiva. F3. Avaliação constante dos projetos. F4. Disponibilidade de recursos tecnológicos. F5. Equipe consistente – grupo de pessoas que compartilham as mesmas habilidades e demonstram confiança e respeito mútuo.	PONTOS FRACOS D1. Assumir compromissos sem consultar. D2. Estrutura e tomada de decisão. D3. Diferente capacitação (skills) das pessoas. D4. Pressões por prazos. D5. Burocracia interna. D6. Limitação de recursos financeiros. D7. Lideranças sem tempo para ouvir equipe. D8. Não há capacitação para liderança. D9. Ausência de comunicação rápida e efetiva. D10. Avaliação dos projetos apenas na entrega. D11. Não há recursos tecnológicos. D12. Problema de integração entre áreas de TI. D13. Cultura de horas extras. D14. Resistência ao uso padrões.
	OPORTUNIDADES O1. Ruptura tecnológica O2. Existência de diversos métodos e práticas.	- Os pontos fortes relacionados às pessoas (F5 e F2) e o relacionado a processos internos que permitem a inovação (F1, F3 e F2) podem ser utilizados para aproveitar a oportunidade O1. - A oportunidade O2 pode ser explorada através dos pontos fortes F1, F3 e F2.	- Respeitar as opiniões da equipe, e considera-las acima de seu interesse próprio em relação ao compromisso a ser assumido, ajuda a superar D1 e D14. - Delegar decisões e comunicação para alinhar expectativas ajuda a superar D2, D7 e D8. - Desenvolver comunicação e práticas de trabalho que aproveitem a diversidade de habilidades das pessoas ajuda a superar D3, D12 e D14. - Considerando as restrições, negociar partes ou funcionalidades do sistema deverão ser entregues primeiro ajuda a superar D4, D5 e D6. - Não houve proposta para superar D9 e D13. - Promover integração entre as áreas da organização e sensibilizar as pessoas sobre vantagens da tecnologia para organização ajuda a superar D10, D11 e D12.
	AMEAÇAS A1. Oscilações econômicas.	- Os pontos fortes F5, F3 e F2 podem ser utilizados para proteger da ameaça A1.	- As mesmas ações sugeridas para superar os pontos fracos e aproveitar as oportunidades servem para fazer frente à ameaça A1.

Fonte: Produção do autor.

- **Pontos Fortes X Oportunidades:** Como usar os pontos fortes identificados para aproveitar as oportunidades existentes?

O grupo chegou ao consenso de que os pontos fortes, fatores internos de uma organização que ajudam a considerar fatores humanos na engenharia de software, podem ser agrupados da seguinte forma:

- Pessoas e as relações entre elas (**F5 e F2**).
- Disponibilidade de recursos tecnológicos para o trabalho e comunicação rápida e eficiente (**F4 e F2**).
- Processos que permitem avaliações constantes dos projetos, a comunicação e a inovação (**F1, F3 e F2**).

Os pontos fortes relacionados às pessoas (**F5 e F2**) e o relacionado a processos internos que permitem a inovação (**F1, F3 e F2**) podem ser utilizados para aproveitar a oportunidade **O1**. Essa ação traz novos desafios e oportunidades para a equipe, que possui relação direta com a satisfação das pessoas.

A oportunidade **O2** pode ser explorada através dos pontos fortes **F1, F3 e F2**. Ter processos que permitem a inovação e avaliação constante de projetos permite a identificação de métodos e práticas que possam ser mais adequadas à equipe de acordo com as características da empreitada, eliminando ou diminuindo restrições, o que favorece a satisfação, que pode vir a favorecer a união e o aprendizado de novas práticas, que também favorecem a satisfação.

- **Pontos Fortes X Ameaças:** Como usar os pontos fortes identificados para se proteger das ameaças?

As ameaças externas que atrapalham consideração de fatores humanos na engenharia de software são as oscilações econômicas (**A1**). O grupo destacou que elas causam mudanças que influenciam a realização de projetos, e por vezes afetam o cumprimento de metas, que afeta expectativas e reconhecimento, o que pode levar as pessoas a focarem mais em seus interesses próprios (políticas individuais), o que pode diminuir a

satisfação das pessoas que participam da empreitada de engenharia de software. Para se proteger das ameaças, devem-se explorar os pontos fortes **F5**, **F3** e **F2**, pois uma comunicação eficiente permite o esclarecimento de novas expectativas que se façam necessárias frente às ameaças. Uma equipe consistente, bem informada e que realiza avaliações constantes dos projetos, possui um entendimento de seus processos que permite absorção de mudanças e o mínimo, ou até mesmo ausência, de interesses individuais.

- **Pontos Fracos X Oportunidades:** Como superar os pontos fracos identificados para aproveitar as oportunidades?

O grupo identificou formas de agir com o objetivo de superar os pontos fracos e considerar fatores humanos na engenharia de software para aproveitar as oportunidades identificadas. Todas as repostas do grupo são relacionadas a ações da liderança da equipe⁵⁵. A seguir os comentários para cada ponto fraco.

- **D1. Assumir compromissos sem consultar a equipe.**

Essa é uma atitude que deve ser sempre evitada, alguns compromissos podem até ser assumidos, mas apenas quando existem cenários previsíveis e claros, onde as atividades para o objetivo são claras, o que não é o caso das oportunidades **O1** e **O2**. Para aproveitar as oportunidades **O1** e **O2** os líderes, devem inspirar⁵⁶ as pessoas a participar e opinar sobre as oportunidades tecnológicas e métodos ou práticas utilizadas. Não se trata de delegar decisões ou formar algum tipo de colegiado, mas sim de respeitar as opiniões do grupo de trabalho, e considera-las acima de seu interesse próprio em relação ao compromisso a ser assumido.

- **D2. Estrutura organizacional com diversos níveis e intersecções em relação às tomadas de decisão.**

⁵⁵ O grupo entrevistado defende o conceito que algum tipo de liderança sempre deve estar presente, mesmo em equipes que seguem paradigmas ágeis e utilizam práticas de auto-organização.

⁵⁶ O grupo entrevistado destacou o “inspirar as pessoas” como algo diferente de “motivar”, segundo o grupo, “motivar” tem uma associação a receber algo em troca, o que não esse caso. O “inspirar” está associado a “fazer surgir o entusiasmo”, fazer com as pessoas percebam uma oportunidade de aprendizado e ou participação.

Para superar este ponto fraco os diversos níveis de liderança devem saber delegar decisões. Além disso, é necessário trabalhar a comunicação para que ocorra o alinhamento de expectativas. Cada líder deve alinhar com seus liderados quem decide o que. Contudo, não se trata apenas de dizer quem faz o que, é necessário que ocorra a transmissão de conhecimento entre líder e quem recebe a missão de decidir. Não se deve dar responsabilidade de decisão sem dar condições, conhecimento, para a tomada de decisão.

- **D3. Diferente capacitação (skills) das pessoas, diferentes níveis de conhecimento tecnológico.**

Para superar este ponto fraco, a liderança deve conversar com as pessoas da equipe sobre as vantagens de uma equipe possuir pessoas com diferentes níveis de habilidades. Pessoas com habilidades diferentes podem responder de forma diferente os mesmos problemas, sendo possível utilizar essa diferença para identificar melhores respostas e oportunidades. Contudo, o líder pode ter problemas relacionados à satisfação de algumas pessoas, pois podem ocorrer casos em que uma pessoa não se sinta bem em fazer o que não é natural para ela, por exemplo, participar de constantes discussões para alinhamento de visões diferentes.

- **D4. Pressões para entregar o sistema em prazos cada vez menores.**
- **D5. Burocracia interna.**
- **D6. Limitação de recursos financeiros.**

Para superar estes pontos fracos deve-se recorrer a princípios do paradigma ágil. É necessário negociar o que pode ser feito no prazo solicitado, considerando a burocracia interna e a limitação de recursos. É estabelecer prioridades sobre que partes ou funcionalidades do sistema deverão ser entregues primeiro, e estabelecer uma sequência de revisões e entregas, caso contrário, todos irão ficar insatisfeitos, a equipe responsável pela empreita e as pessoas que necessitam do sistema. Esta ação demanda que a que a área de TI da organização pense de forma livre de preconceitos, tanto para negociar a priorização como para aproveitar as oportunidades do ambiente.

- **D7. Lideranças não dispõem de tempo para ouvir as pessoas da equipe.**

Este é um ponto fraco considerado pelo grupo como “bola de neve”: quanto menos um líder delega decisões, menos ele tem tempo para conversar com as pessoas. Para superar este ponto fraco deve-se recorrer a princípios de delegar decisões e comunicação, para que ocorra o alinhamento de expectativas, como comentado para o ponto fraco **D2**.

- **D8. Não há capacitação das pessoas técnicas que passam a assumir posições de liderança.**

Superar este ponto fraco está relacionado ao comentado para o ponto fraco **D2**. Não se deve dar responsabilidade de decisão sem dar condições, conhecimento, para a tomada de decisão.

- **D9. Ausência de comunicação interna rápida e efetiva.**

O grupo debateu se este ponto fraco é um problema organizacional ou se ele ocorre de propósito, isto é, se as pessoas não se comunicam de forma clara de propósito, se existem interesses pessoais em não ser claro, até mesmo para evitar comprometimento. Quando é um problema organizacional, cabe as lideranças identificar e procurar ferramentas e processos, por exemplo aproveitar a oportunidade **O2**, para superar este ponto fraco. Quando é relacionado a interesses pessoais, é mais complicado de ser superado, pode envolver uma disputa líderes e liderados e um interesse maior que é satisfação de todos em participar da empreitada. Este ponto fraco ficou em aberto, não se encontrou uma forma de supera-lo sem a troca de pessoas, contudo, decidir pela troca de pessoas leva a outro problema: como identificar essas pessoas e justificar a troca, sem utilizar apenas impressões pessoais.

- **D10. Avaliação dos projetos apenas na entrega.**

- **D11. Não há disponibilidade de recursos tecnológicos.**

É necessária a presença de uma liderança superior à equipe que executa a empreitada de engenharia de software para superar esses

dois pontos fracos. Para superar o ponto fraco **D10** é necessário que a organização promova processos de integração entre áreas com interesses no sistema e área de TI para que as pessoas tenham disponibilidade de tempo para participar de avaliações. Para o **D11** é necessário um trabalho conjunto entre a área de TI e as demais áreas da organização para evidenciar o impacto que restrições tecnológicas trazem ao negócio principal da organização, e não apenas para a área de TI.

- **D12. Problema de integração entre áreas de TI.**

Este ponto fraco pode ser superado de forma semelhante ao ponto fraco **D3**. As diferentes áreas de conhecimento (banco de dados, redes, infraestrutura, etc.) que compõem a área de TI de uma organização possuem conhecimentos diferentes que podem responder de forma diferente aos mesmos problemas. A liderança deve promover a integração entre as áreas para compartilhar conhecimentos, propiciando visão sistêmica para identificar as melhores respostas e oportunidades.

- **D13. Cultura de horas extras.**

O grupo considerou que este ponto fraco não é possível de ser superado. Ele envolve valores pessoais. Existem pessoas que não se importam em ter uma vida equilibrada entre trabalho e família, se preocupam com o lado financeiro da questão, ou até mesmo por não ter vontade de voltar para casa cedo. Apesar do grupo indicar que eles próprios como líderes não valorizam essa cultura de horas extras, eles admitiram conhecer pessoas com essa cultura, e que nada conseguiram fazer para mudar essa opinião. A menos quando existiam regras ou determinações que proibiam hora extra. Contudo, essa resposta ao problema deixa insatisfeitas as pessoas que adotam essa prática.

- **D14. Resistência ao uso de processos e métodos padronizados.**

O grupo sugeriu que a resistência das pessoas pode ser superada através de comunicação e participação dos componentes do grupo. Foi colocado que existe resistência quando as pessoas não participam do processo de decisão de uso do processo ou método.

A liderança da equipe deve proporcionar informações sobre o processo ou método padronizado que a organização ou área de TI deseja implementar. Contudo, para vencer este ponto fraco é necessário ir além de uma simples apresentação e distribuição de documentação, é necessário que ocorram debates sobre as vantagens e desvantagens do uso do padrão em relação organização, área de TI e equipe. Todos os membros da equipe devem ser convidados a participar dos debates. É um processo longo, mas quando feito à margem dos interesses pessoais, ocorre o aprendizado sobre os padrões. A participação de todos interessados não necessariamente reduz a resistência, mas possibilita que se identifiquem os argumentos para a resistência, de forma a não ficar apenas a ideia de mudanças não são bem vindas.

- **Pontos Fracos X Ameaças:** O que fazer para superar os pontos fracos identificados que não fazem frente às ameaças?

O grupo considerou que todas as ações sugeridas para superar os pontos fracos e aproveitar as oportunidades, também ajudam a fazer frente às ameaças. Não foi incluída nenhuma ação específica para fazer frente à ameaça.

4.5 SISTEMAS IMPLÍCITOS

A empreitada de engenharia de software constrói sistemas de software que são desenvolvidos e mantidos por seres humanos. As figuras 11 e 12 (figura rica e linguagem sistêmica) e a matriz TOWS (tabela 3) foram elaboradas utilizando as considerações de diversas pessoas que participam ativamente de empreitadas de desenvolvimento e manutenção de sistemas de software. A figura rica, a representação em linguagem sistêmica e a matriz TOWS não apresentam estratégias ou elementos que tenham por objetivo o aumento da produtividade, previsibilidade ou qualidade do trabalho das pessoas. Elas foram elaboradas a partir da proposta de considerar fatores humanos na engenharia de software; uma abordagem que promove o humanismo da empreitada de engenharia de software, que considera que as

peessoas possuem diferentes desejos, valores e emoções, que as pessoas devem ser consideradas em sua integralidade.

Para realizar a análise e síntese das informações existentes na figura rica, representação sistêmica e matriz TOWS, o autor deste trabalho desenvolveu um mapa conceitual⁵⁷ (figura 13). O mapa obtido possibilita a identificação de sistemas implícitos, que neste caso são todos sistemas de atividades humanas. Esses sistemas que devem existir para o que os sintomas relacionados à questão de considerar os fatores humanos na engenharia de software apareçam.

Os sistemas implícitos identificados são:

- **Sistema de atividades pessoais:** Este sistema é composto pelo aprendizado, que possibilita uma pessoa a superar com ética⁵⁸ os desafios que também devem ser estabelecidos de forma ética. Além disso, o aprendizado tem relação com processos de inovação, que também contribuem com novos desafios para as pessoas. A avaliação constante de projetos também possibilita novos aprendizados que auxiliam nos mecanismos de delegar decisões e são oportunidades de crescimento pessoal.
- **Sistemas de atividades de equipe:** Este sistema de atividades humanas é um sistema no qual é necessária comunicação eficiente. A comunicação clara e ética permite a compreensão das mudanças quando elas são necessárias e também a existência de uma cultura organizacional na qual as diversas habilidades das diferentes pessoas podem ser aproveitadas na forma de trabalho da equipe e também para avaliação de projetos, proporcionando crescimento profissional e autonomia das pessoas da equipe.

⁵⁷ Mapa conceitual é uma estrutura esquemática para representar um conjunto de conceitos imersos numa rede de proposições. É uma abordagem com base na teoria construtivista, em especial na teoria da aprendizagem significativa de David Ausubel, que entende que o indivíduo constrói seu conhecimento e significados a partir de sua predisposição para realizar essa construção, relacionando a informação que lhe é apresentada com o seu conhecimento prévio sobre o tema. Mapas conceituais são considerados como estruturadores do conhecimento, apresentam como o conhecimento sobre o assunto está organizado na estrutura cognitiva de seus autores, que assim podem visualizar e analisar a profundidade e extensão desse conhecimento (TAVARES, 2007; MOREIRA, 2010).

⁵⁸ Para a identificação dos sistemas implícitos ainda se mantém o conceito de ética como o utilizado pelos entrevistados: indicador de qualidade de uma ação.

- **Sistemas de atividades de forma de trabalho:** Este sistema de atividades humanas é composto pelas pessoas e as atividades que elas realizam para que se tenha uma forma de trabalho livre de preconceitos, que permita o aprendizado e a negociação das solicitações e priorização de atividades. Que favoreça a união da equipe através de comunicação clara e ética sobre quem toma as decisões necessárias na execução do trabalho. Toda a forma de trabalho deve ter uma liderança, formal ou informal.
- **Sistemas de atividades da organização:** Este sistema estabelece as políticas de reconhecimento financeiro e pessoal das pessoas que trabalham na empreitada de engenharia de software. Essas políticas devem ser comunicadas de forma clara e ética para que todos conheçam os critérios utilizados para reconhecimento, e não ocasione expectativas de reconhecimento que não irão se cumprir. Cada organização tem a sua cultura organizacional que influencia seus processos, que tem relação com a clareza e a ética na forma como (i) ocorre a comunicação dentro da organização, (ii) projetos são avaliados, e (iii) é conduzida a inovação. A cultura de uma organização também tem influência em como é feita e aceita a delegação de decisões. Podem surgir restrições em relação à empreitada de engenharia de software, e a cultura organizacional pode promover ou ajudar as pessoas a vencerem essas restrições.
- **Sistemas de atividades de comunicação:** Este sistema de atividades humanas transmite informações de forma clara e ética para que as pessoas tenham conhecimento de como o trabalho será reconhecido, como as mudanças serão tratadas, que processos deverão ser seguidos, criados ou atualizados. A comunicação não deve ser utilizada para atender interesses pessoais.

4.6 SISTEMAS CONTENEDORES

O autor deste trabalho decidiu não utilizar técnicas especiais de agrupamento como os propostos por Hitchins (2007, p. 205). Todos os sistemas implícitos são sistemas de atividades humanas que, pela própria definição, possuem características relacionadas às pessoas e as interações entre elas, o que leva à existência de padrões que podem ser identificados pela observação das características próprias de cada sistema implícito. Os sistemas contenedores identificados são:

- **Sistema de necessidades pessoais:** contém os sistemas implícitos:
 - Sistema de atividades pessoais.
- **Sistema de necessidades da equipe:** contém os sistemas implícitos:
 - Sistema de atividades de equipe.
 - Sistema de atividades de forma de trabalho.
- **Sistema de necessidades da organização:** contém os sistemas implícitos:
 - Sistema de atividades da organização.
 - Sistema de atividades de comunicação.

Os sistemas contenedores identificados representam uma perspectiva sobre a situação do problema no mundo ideal. Não significa que necessariamente eles serão projetados e implementados no mundo real. Também não significa que eles sejam os únicos sistemas existentes na situação do problema. Durante as fases seguintes podem surgir novos sintomas do problema que não foram identificados até este momento, neste ciclo de execução do sistema de métodos⁵⁹.

4.7 INTERAÇÃO E DESEQUILÍBRIOS DOS SISTEMAS CONTENEDORES

As interações entre os sistemas contenedores ocorrem durante a empreitada de engenharia de software. Essas interações estão representadas na figura 14, as linhas entre os sistemas indicam a existência de interações entre eles, que ocorrem em ambas as direções.

⁵⁹ A sequência de identificação e tratamento de sintomas segue conceito da abordagem evolucionária (Apêndice C).

Figura 14 – Interação entre sistemas contenedores

Fonte: Produção do autor

Um sintoma relevante presente em todos os sistemas implícitos é a referência à ética. A interpretação de ética como a qualidade de uma ação. Sobre a interpretação de ética, Renato Janine Ribeiro argumenta:

[...] penso que há um grande mal-entendido sobre a ética em nossos dias, que até tem um fundo positivo: é o fato de cada vez mais pessoas terem simpatia por ela, de haver um clamor pela ética na política, no âmbito social geral. Um dos grandes discursos dos últimos anos reclama por ética. Na sociedade brasileira isso corresponde a um fato social importante, a saber: não existe mais sustentação que legitime a desigualdade, a injustiça, a miséria entre nós. (RIBEIRO, 2003, p. 150).

A interpretação de ética como qualidade de uma ação permite que a intenção das pessoas, ao utilizarem a palavra ética, possa estar representada tanto no entendimento do problema como na proposição de repostas que podem resolver ou solucionar esse problema. Considerar essa interpretação na interação entre os sistemas contenedores sugere que as interações entre esses sistemas podem ser éticas ou motivadas por interesses pessoais, como representadas na figura 15.

Figura 15 – Interação entre sistemas contenedores considerando ética e interesses

Fonte: Produção do autor. Representa a interpretação de ética como a qualidade de uma ação.

Contudo, ética significa um conjunto de valores e princípios que guia a conduta das pessoas (CORTELLA; BARROS FILHO, 2014, p.24). O significado de ética implica que ela está presente mesmo quando as interações ocorrem por interesse pessoal, ou interesse de um grupo de pessoas, pois a decisão e a ação ocorrem de acordo com um conjunto de valores e princípios pessoal, ou compartilhado por um grupo de pessoas.

Sobre ética, Edgar Morin afirma:

Todo olhar sobre a ética deve perceber que o ato moral é um ato individual de religação: religação com um outro, religação com uma comunidade, religação com uma sociedade e, no limite, religação com a espécie humana. (MORIN, 2007, p. 21)

Considerando essa argumentação de Edgar Morin e observando a figura 14 é possível considerar que a forma como a palavra ética é utilizada está associada:

- A cada pessoa que a utilizou (como indivíduo).
- À relação entre as pessoas (equipe).
- À relação com a comunidade (organização), e, no limite,
- ao humano (fatores humanos).

A ética, como conjunto de valores e princípios que guia a conduta das pessoas, deve estar presente nas interações que ocorrem entre os sistemas contenedores, e inclui as interações que, na interpretação pelo senso comum (figura 15), são motivadas por interesses pessoais.

4.8 TRATAMENTO PARA DESEQUILÍBRIO E AVALIAÇÃO DO IMPACTO DA PROPOSTA

Os sintomas comentados no item anterior devem ser considerados em uma nova proposta de representação dos sistemas contenedores (figura 16), na qual a ética, como conjunto de valores e princípios que guia a conduta das pessoas, está presente nas interações.

Figura 16 – Interação entre sistemas, após tratamento dos desequilíbrios.

Fonte: Produção do autor.

4.8.1 Impacto da proposta

As interações entre os sistemas contenedores ocorrem por ação de uma pessoa, ou grupo de pessoas, e essa ação sempre possui uma intenção. Considerar fatores humanos na engenharia de software é considerar as pessoas envolvidas nessas ações, que deve ser a consequência das ações e não o resultado direto, a intenção principal da ação. No cenário representado na figura 16, a ética está relacionada ao conjunto de ações que são realizadas no dia a dia da empreitada de engenharia de software, e inclui também as ações realizadas nos sistemas implícitos que compõem os sistemas contenedores. Toda ação deve ser construída para deixar um legado positivo ao grupo, à organização, à sociedade. Bônus e reconhecimento profissional e financeiro são direitos das pessoas, contudo, o reconhecimento de que se está atendendo e servindo as pessoas que participam da empreitada é a parte fundamental, a origem para esse direito.

Os sistemas implícitos, sistemas contenedores e as interações existentes, permitem a percepção de que há problemas relacionados à ética na empreitada de engenharia de software. Principalmente, como o conjunto de valores e princípios que guiam a conduta das pessoas com respeito às outras pessoas e a sociedade.

4.9 SOLUÇÃO POTENCIAL

A empreitada de engenharia de software envolve atividades como especificação, desenvolvimento, testes, operação, suporte e manutenção do sistema. Devido à abrangência do conjunto dessas atividades, é comum ser necessário uma equipe de pessoas para realizá-las, e essas pessoas devem assumir responsabilidades em relação a essas atividades. Para isso, além dos conhecimentos técnicos necessários para realizar essas atividades, as pessoas que participam dessa empreitada também devem ser sensíveis às necessidades uns dos outros, da comunidade e da sociedade. Isso é necessário para que (i) essas pessoas se sintam satisfeitas e (ii) o sistema de software criado, ou modificado, atenda às expectativas das pessoas que irão utilizar o sistema. Contudo, no mundo real, esse cenário nem sempre ocorre.

Durante a empreitada de engenharia de software há diversos momentos em que é necessário pensar qual a ação a ser tomada e nos motivos da escolha realizada. Em geral, o ser humano possui a tendência de se defender, de defender seus interesses

próprios. O problema ocorre quando esses interesses não consideram as outras pessoas envolvidas, não consideram os fatores humanos envolvidos na ação e nos motivos da escolha da ação tomada.

4.9.1 Código de ética da engenharia de software

O raciocínio e o julgamento humano possuem um papel decisivo na execução da empreitada de engenharia de software, no respeito às pessoas no decorrer de todo o ciclo de vida de um sistema de software.

Para garantir, tanto quanto o possível, que os esforços das pessoas que participam da empreitada de engenharia considerem as pessoas envolvidas, a organização e a sociedade, as duas principais associações de profissionais que atuam com no desenvolvimento de sistemas de software, a Association for Computing Machinery (ACM) e a Computer Society of the Institute of electrical and Electronic Engineers (IEEE-CS), aprovaram em conjunto o Código de Ética e Prática Profissional na área de engenharia de software (IEEE/ACM, 1999).

O Código de Ética e Prática Profissional contem oito Princípios relacionados à tomada de decisão e ao comportamento das pessoas que participam do desenvolvimento de sistemas de software⁶⁰. No preambulo da versão estendida do Código de Ética (IEEE/ACM, 1999) é argumentado que os Princípios identificam as relações nas quais indivíduos, grupos e organizações participam, e a obrigação primária dessas relações, sendo que essas obrigações são fundadas no cuidado com as pessoas afetadas pelo trabalho.

As características dos Princípios, argumentadas no preambulo do Código, os identificam como o conjunto de práticas que devem estar presentes na empreitada de engenharia de software para a consideração de fatores humanos.

4.10 EXTENSÃO DO KERNEL DO ESSENCE

Embora seja importante reconhecer que uma conduta profissional, que obedece um Código de Ética, deva ser utilizada no processo de desenvolvimento de software, os Princípios do Código de Ética e Prática Profissional da engenharia de software não

⁶⁰ Essas pessoas são: engenheiros, praticantes, educadores, gerentes, supervisores, pessoas que tomam decisões políticas, trainees, estagiários e estudantes.

estão presentes nas “coisas com as quais sempre trabalhamos” e “coisas que sempre fazemos” do Kernel do Essence (item 2.1). Essa ausência não é um problema no conceito do Kernel, por duas razões:

- Ele é definido como sendo o conjunto dos elementos essenciais da engenharia de software, e não como os vários elementos necessários para uma equipe desenvolver um sistema de software.
- Por conter apenas o essencial, o Kernel pode ser estendido para considerar novos elementos ou práticas, de acordo com a necessidade de uma empreitada de engenharia de software.

A flexibilidade para extensão do Kernel do Essence permite que ele seja estendido para considerar os Princípios do Código de Ética e Prática Profissional da engenharia de software, permitindo que as práticas relacionadas aos Princípios sejam consideradas pelas pessoas no exercício de suas funções profissionais.

Jacobson et al. (2013) e Essence (2014) apresentam as possibilidades de extensão do Kernel. Dentre as formas possíveis de serem utilizadas, a mais simples é suplementar os checklists dos estados de um alpha. Essa forma de extensão não demanda conhecimento da linguagem Essence e é de fácil utilização pelas pessoas, que, normalmente, já utilizam alguma forma de checklist no dia a dia.

Utilizar um checklist permite que os membros da equipe debatam sobre cada item da lista e identifiquem se eles estão tendo problemas para serem aderentes aos Princípios do Código de Ética e Prática Profissional.

Das três áreas de interesse do Kernel do Essence, é na área Empreitada onde ocorrem as interações entre as pessoas que desenvolvem os sistemas de software. Os três alphas da área Empreitada e seus respectivos estados são (ESSENCE, 2014, p.64-71, tradução do autor):

- **Equipe:** grupo de pessoas ativamente engajadas no desenvolvimento, manutenção, entrega e suporte de um sistema de software. Uma, ou mais, equipes planejam e executam o trabalho necessário para criar, atualizar e/ou modificar um sistema de software específico. Seus estados são:

- Iniciada – a missão da equipe está clara e o know-how para o crescimento da equipe está disponível.
- Formada – a equipe foi formada com pessoas suficientes para começar a missão.
- Colaborando – os membros da equipe estão trabalhando juntos de forma coesa.
- Realizando – a equipe está trabalhando de forma eficaz e eficiente.
- Em Recesso – a equipe não é mais necessária para a realização da missão.
- **Trabalho:** atividade envolvendo esforço físico e mental realizada por uma equipe para produzir um sistema de software. Seus estados são:
 - Iniciado – o trabalho foi solicitado.
 - Preparado – Todas as pré-condições para iniciar os trabalhos foram satisfeitas.
 - Começado – o trabalho está a decorrer.
 - Sob Controle – O trabalho está indo bem, os riscos estão sob controle e os níveis de produtividade são suficientes para alcançar um resultado satisfatório.
 - Concluído – O trabalho necessário para se obter os resultados foi concluído.
 - Encerrado – Todas as tarefas de manutenção restantes foram concluídas e o trabalho foi oficialmente encerrado.
- **Forma de Trabalho:** conjunto de práticas e ferramentas usadas por uma equipe para guiar e suportar o trabalho que realizam. Seus estados são:
 - Princípios Estabelecidos – os princípios, e as restrições, que moldam a forma de trabalho estão estabelecidos.
 - Fundamentos Estabelecidos – as principais práticas e ferramentas, que constituem a base da forma de trabalho, estão selecionadas e prontas para uso.
 - Em Uso – alguns membros da equipe estão utilizando, e adaptando, a forma de trabalho.
 - No Lugar – todos os membros da equipe estão utilizando a forma de trabalhar para realizar seu trabalho.

- Funcionando Bem – a forma de trabalho da equipe está funcionando bem para a equipe.
- Retirada – a forma de trabalho não está mais em uso pela equipe.

Através da definição de cada alpha, e da lista de estados de cada um, identifica-se que é o alpha forma de trabalho é o que apresenta características que permitem receber uma extensão para considerar o Código de Ética, pois é esse alpha que identifica as práticas a serem utilizadas e estabelece os princípios, e as restrições, que moldam a sequência de passos que guiam o trabalho a ser realizado pela equipe. O Código de Ética e Prática Profissional da engenharia de software (IEEE/ACM, 1999) é extenso, contudo, ele possui uma versão reduzida, que apresenta de forma sucinta o preâmbulo e os oito Princípios relacionados ao comportamento e às decisões tomadas pelos profissionais que atuam na engenharia de software. A forma reduzida dos oito Princípios permite que eles sejam utilizados para elaborar um checklist. Essa possibilidade é utilizada pelo autor para suplementar os checklists, definidos pelo Kernel do Essence, para os estados do alpha forma de trabalho. Essa extensão dos checklists permite que os membros da equipe de desenvolvimento identifiquem se há ocorrências, na condução da empreitada de engenharia de software, que não estejam aderentes aos oito Princípios do Código de Ética e Prática Profissional.

Cada um dos Princípios do Código de Ética exprime uma forte afirmação ou intenção. Portanto, uma forma de se identificar a adesão dos membros de uma equipe a um Princípio, é através da negação desse Princípio. Por exemplo, o Princípio⁶¹: **1. PUBLIC – Software engineers shall act consistently with the public interest**, pode ser representado em forma de um item de checklist através da negação: “os membros da equipe não estão agindo de forma inconsistente com o interesse público”, esse formato permite que os membros da equipe identifiquem se estão aderentes ou não ao Princípio. Caso a resposta ao checklist seja afirmativa, estão aderentes, caso contrário, não estão. A tabela 4 apresenta a negação para os oito Princípios do Código de Ética e Prática Profissional da engenharia de software.

⁶¹ O texto não foi traduzido pelo autor para o português devido às restrições do IEEE/ACM que não permitem nenhuma alteração do texto original (IEEE/ACM,1999).

Tabela 4 – Negação dos Princípios do Código de Ética e Prática Profissional

Princípio ⁶²	Forma de checklist (negação)
1. PUBLIC - Software engineers shall act consistently with the public interest.	Os membros da equipe não estão agindo de forma inconsistente com o interesse público.
2. CLIENT AND EMPLOYER - Software engineers shall act in a manner that is in the best interests of their client and employer consistent with the public interest.	Os membros da equipe não estão agindo de uma forma que não é no melhor interesse de seu cliente e empregador.
3. PRODUCT - Software engineers shall ensure that their products and related modifications meet the highest professional standards possible.	Os membros da equipe asseguram que seus produtos, e as modificações de produtos, não estão seguindo padrões pobres ou inadequados
4. JUDGMENT - Software engineers shall maintain integrity and independence in their professional judgment.	As decisões dos membros da equipe não estão sendo influenciadas por outros profissionais ou outros interesses, independentemente de quaisquer vantagens pessoais.
5. MANAGEMENT - Software engineering managers and leaders shall subscribe to and promote an ethical approach to the management of software development and maintenance.	Os gerentes e líderes da equipe não estão realizando ou promovendo uma abordagem antiética no gerenciamento do desenvolvimento, ou da manutenção, do sistema de software em qualquer aspecto: técnico, profissional, comportamental ou em aspectos relativos às pessoas da equipe.
6. PROFESSION - Software engineers shall advance the integrity and reputation of the profession consistent with the public interest.	Os membros da equipe, por ação ou por omissão, não estão denegrindo a profissão.
7. COLLEAGUES - Software engineers shall be fair to and supportive of their colleagues.	Os membros da equipe não estão sendo injustos, desrespeitosos, ou não estão deixando de apoiar qualquer colega, profissional ou membro de qualquer equipe.
8. SELF - Software engineers shall participate in lifelong learning regarding the practice of their profession and shall promote an ethical approach to the practice of the profession	Os membros da equipe, por ação ou por omissão, não estão evitando ou ignorando as oportunidades de aprendizado sobre o exercício da profissão ou reforçar as abordagens éticas para o exercício da profissão.

Fonte: Produção do autor.

⁶² As informações dessa coluna foram extraídas de IEEE/ACM (1999). Copyright © 1999 by the Institute for Electrical and Electronics Engineers, Inc. and the Association for Computing Machinery, Inc..

Utilizando a negação de cada Princípio (tabela 4), o autor elaborou tabela 5, que apresenta o checklist suplementar, e a justificativa da suplementação, para cada estado do alpha forma de trabalho.

Durante a empreitada de engenharia de software há diversos momentos em que os membros da equipe, e a liderança dessa equipe, necessitam considerar qual a ação a ser tomada e os motivos da escolha realizada. A adesão dos membros da equipe e da liderança aos Princípios do Código de Ética e Prática Profissional estabelece elementos comuns a todos para escolha da ação a ser tomada e dos motivos da escolha, permitindo que as pessoas envolvidas considerem os fatores humanos envolvidos na ação e nos motivos da escolha da ação tomada. Ter um checklist durante a empreitada permite que a cada transição de estado do alpha forma de trabalho a equipe como um todo dialogue sobre a aderência ou não ao Código de Ética e Prática Profissional da engenharia de software.

Tabela 5 – Checklist suplementar para os estados do alpha forma de trabalho.

Estado	Checklist Suplementar	Justificativa
Princípios Estabelecidos	<ul style="list-style-type: none"> Os membros da equipe aderem ao Código de Ética e Prática Profissional da engenharia de software. 	Neste estado os princípios que moldam a forma de trabalho são estabelecidos, por isso há a necessidade dos membros da equipe concordarem em aderir ao Código de Ética.
Fundamentos Estabelecidos	<ul style="list-style-type: none"> Os membros da equipe concordam que as práticas e ferramentas selecionadas aderem ao Código de Ética e Prática Profissional da engenharia de software. 	Neste estado as práticas e ferramentas são definidas, é necessário que elas não estejam em desacordo com o Código de Ética.
Em Uso	<ul style="list-style-type: none"> Os membros da equipe não estão agindo de forma inconsistente com o interesse público. Os membros da equipe não estão agindo de uma forma que não é no melhor interesse de seu cliente e empregador. Os membros da equipe asseguram que seus produtos, e as modificações de produtos, não estão seguindo padrões pobres ou inadequados. As decisões dos membros da equipe não estão sendo influenciadas por outros profissionais ou outros interesses, independentemente de quaisquer vantagens pessoais. Os gerentes e líderes da equipe não estão realizando ou promovendo uma abordagem antiética no gerenciamento do desenvolvimento, ou da manutenção, do sistema de software em qualquer aspecto: técnico, profissional, comportamental ou em aspectos relativos às pessoas da equipe. Os membros da equipe, por ação ou por omissão, não estão denegrindo a profissão. Os membros da equipe não estão sendo injustos, desrespeitosos, ou não estão deixando de apoiar qualquer colega, profissional ou membro de qualquer equipe. Os membros da equipe, por ação ou por omissão, não estão evitando ou ignorando as oportunidades de aprendizado sobre o exercício da profissão ou reforçar as abordagens éticas para o exercício da profissão. 	Neste estado, as práticas e ferramentas selecionadas estão sendo utilizadas de acordo com os princípios, e restrições, estabelecidos no primeiro estado do alpha. O desenvolvimento do sistema de software está ocorrendo. É necessário checar a aderência a todos os princípios do Código de Ética.

Continua

Continuação

Estado	Checklist Suplementar	Justificativa
No Lugar	• Suplemento idêntico ao do estado Em Uso.	Mesmo do estado anterior.
Funcionando Bem	• Suplemento idêntico ao do estado Em Uso.	Mesmo do estado anterior.
Retirada	• Os membros da equipe têm a certeza de que as decisões tomadas durante a empreitada estão registradas para referência futura.	Decisões tomadas durante a empreitada de desenvolvimento podem ter consequências positivas, ou negativas, quando o sistema de software entra em operação. Apesar dos Princípios não fazerem referência, é necessário que se guarde para referência futura as decisões que podem impactar o comportamento do sistema perante determinadas condições de operação. Outra razão é a construção de uma base de conhecimento que serve de referência para decisões futuras.

Fonte: Produção do autor.

4.11 CONSIDERAÇÕES FINAIS

Este capítulo apresenta a aplicação do RSM para obter um conjunto de práticas que possibilitem a consideração de fatores humanos na engenharia de software. Esse conjunto são os oito Princípios do Código de Ética e Prática Profissional da engenharia de software. Para essas práticas estarem incorporadas no Kernel do Essence, o autor utiliza a negação para suplementar o checklist dos estados alpha forma de trabalho. A utilização do checklist permite que, a cada transição de estado do alpha forma de trabalho, a equipe como um todo dialogue sobre a aderência ou não ao Código de Ética e Prática Profissional da engenharia de software.

As conversas com as pessoas que participaram da aplicação do sistema de métodos (RSM) sempre tiveram início com o mesmo conjunto de perguntas, e prosseguiram em forma de diálogo. As perguntas iniciais serviram de ponto de partida, para que através do diálogo o autor e as pessoas iniciassem uma conversa, hora entre elas mesmas e hora entre elas e o autor. Promover um diálogo dessa forma permitiu que o autor e as pessoas se sentissem estimuladas a não seguir um roteiro, cada ideia

colocada por uma das pessoas oferecia a oportunidade de se seguir por caminhos que não eram imaginados quando do início da conversa. Apesar dessa forma de construção coletiva proporcionar diversidade de ideias sobre o tema, chegar a um consenso nem sempre é fácil.

O trabalho de engenharia possui uma dimensão técnica, na qual o engenheiro utiliza conhecimentos para embasar suas decisões, que são tanto de natureza técnica como de sua experiência profissional. Contudo, o trabalho do engenheiro também se funde com a dimensão ética de sua atividade, onde estão presentes os códigos comportamentais de um engenheiro e os valores mais amplos das relações da engenharia com a sociedade.

5 PROVA DE CONCEITO

Este capítulo apresenta a prova de conceito, o Essence com seu Kernel estendido é aplicado em uma empreitada de engenharia de software com o objetivo de identificar se as práticas identificadas através do RSM permitem a consideração de fatores humanos na engenharia de software.

5.1 CONSIDERAÇÕES INICIAIS.

A prova de conceito consiste em utilizar o Kernel estendido do Essence em uma empreitada de engenharia de software, e, concluída a empreita, utilizar as informações obtidas no item 4.4.3 para identificar, através do dialogo com os membros da equipe, se há a consideração de fatores humanos na engenharia de software.

O autor é líder técnico de diversos projetos de sistemas de software, e teve a oportunidade de utilizar o Kernel estendido em um projeto de integração entre dois sistemas de uma organização, apesar de área de TI dessa organização não utilizar a especificação Essence, e nem pretender utilizar.

5.2 ALINHAMENTO ENTRE O PROCESSO DA ORGANIZAÇÃO E O KERNEL

Quando se utiliza o Essence em uma organização que já possui um processo de desenvolvimento de sistemas de software, o primeiro passo é realizar o alinhamento entre os estados dos alphas do Kernel e o processo utilizado pela organização (Jacobson et al., 2013).

A prova de conceito é realizada em uma organização que possui um processo definido para tratar as demandas das suas áreas de negócio. Esse processo é composto pelas seguintes fases:

- **Solicitação de demanda** – quando uma das áreas de negócio da organização tem uma necessidade que demande um novo sistema de software, ou a alteração de um já existente, ela preenche um formulário de demanda que é encaminhado para a área de TI. O formulário de demanda possui campos fixos, que a área de negócio utiliza para passar informações necessárias para que a área de TI faça o orçamento da demanda.

- **Orçamentação** - a área de TI recebe as solicitações de demanda. As lideranças técnicas da área se reúnem, entre uma e duas vezes por semana, para avaliar os formulários e preparar os orçamentos para implementar cada uma das demandas. Além do orçamento, também é dada uma estimativa de prazo de execução. Tanto o orçamento como o prazo consideram atividades de desenvolvimento de software e de configuração da infraestrutura eletrônica que possa ser necessária.
- **Aprovação do orçamento** - a organização possui um comitê, constituído por representantes das diversas áreas de negócio, que avalia as demandas considerando o custo, o prazo e o retorno potencial. As demandas aprovadas são encaminhadas para área de TI, as rejeitadas, nem sempre são descartadas. Uma demanda rejeitada no comitê pode sofrer alterações e ser enviada novamente para orçamento e novamente para aprovação. Não há restrição para o número de vezes que esse ciclo pode ocorrer.
- **Especificação e design** - quando é aprovada a demanda, é formada uma equipe para tratá-la. A equipe entra em contato com as pessoas que demandaram o sistema e começa a fazer a especificação. Ao mesmo tempo em que os requisitos são identificados, a arquitetura do sistema de software é avaliada. Caso seja identificado algum problema, como mudança de escopo em relação ao escopo da orçamentação, ou restrições relacionadas à arquitetura do sistema, a equipe conclui a especificação e a liderança técnica elabora um novo orçamento. E, neste caso, a demanda retorna para o comitê para nova aprovação do orçamento. Caso não sejam identificados problemas, a especificação é concluída e é solicitado que as pessoas que demandaram o sistema a aprove. Aprovada a especificação, é iniciado o desenvolvimento. Se não for aprovada, a equipe retoma os trabalhos de especificação e design.
- **Desenvolvimento** – quando é concluída e aprovada a especificação, a equipe elabora o código do sistema de software, e, se for o caso, inicia as configurações de infraestrutura nos três ambientes de hardware da organização: desenvolvimento, homologação e produção.

- **Homologação** – quando a codificação é concluída, sistema de software é disponibilizado no ambiente de homologação para que as pessoas que demandaram o sistema possam testa-lo.
- **Produção** – quando o sistema de software produzido ou alterado é aprovado pelas pessoas que o solicitaram, é liberado para ser implantado no ambiente de produção.

Através da análise dos estados de cada um dos alphas do Kernel, é possível identificar em que estado cada alpha deve estar quando da conclusão de cada uma das fases do processo da organização. Como a **Solicitação de demanda** não é uma atividade da área de TI, ela não é considerada neste mapeamento. O resultado dessa análise está representado na figura 17. Não há uma teoria de engenharia de software ou de gerenciamento de projetos que estabeleça parâmetros para análise e associação apresentada na figura 17, ela foi elaborada através da experiência do autor em trabalhar utilizando o processo da organização e utilizando o Kernel.

Figura 17 – Estados dos alphas do Kernel para cada fase do processo da organização.

Fonte: Produção do autor.

5.3 CARACTERÍSTICAS DO PROJETO PARA PROVA DE CONCEITO

O autor utiliza dois critérios para selecionar o projeto para a prova de conceito: (i) ser um projeto de curto prazo (entre duas e três semanas de duração), (ii) ser um projeto para equipes pequenas (máximo de seis pessoas, incluindo a liderança técnica). Um projeto com tempo de duração maior que três semanas, ou que tenha mais pessoas envolvidas, pode dificultar a execução da prova de conceito, porque, para o uso do Essence, o autor precisa realizar reuniões diárias⁶³ com a equipe, atividade que não faz parte da cultura da área de TI da organização. A importância de reuniões diárias com a equipe é destacada por Jacobson et al. (2013, p.79, tradução do autor): “[...] ela [reunião diária] propicia uma maneira para que eles [membros da equipe] possam oferecer e receber assistências uns dos outros.”.

O uso do Kernel do Essence não é vinculado a nenhuma tecnologia ou linguagem de programação, por essa razão o autor não apresenta detalhes relacionados a linguagem de programação, infraestrutura eletrônica ou tecnologia do barramento de serviço. Por não se tratar de uma prova de conceito sobre o Essence, o autor também não apresenta os detalhes das transições de estado de todos os alphas do Kernel. São apresentados apenas os dados relacionados ao alpha forma de trabalho, que é o alpha relevante neste trabalho.

Para não utilizar nomes reservados da organização, o projeto da prova de conceito é chamado de Projeto Um, o escopo do projeto é implementar a integração entre dois sistemas de software da organização, utilizando um barramento de serviços corporativo⁶⁴. Os sistemas de software são chamados de: Sistema A e Sistema B.

⁶³ Reuniões diárias são encorajadas pela abordagem ágil para que a equipe discuta o que cada membro realizou desde de a reunião anterior, o que cada um irá fazer a seguir, e as potenciais dificuldades ao progresso da empreitada.

⁶⁴ Um barramento de serviços corporativo permite que ocorram trocas de informações entre dois ou mais sistemas de software. O barramento permite que o sistema que necessite da informação não precise conhecer diretamente quem está provendo a informação. Em outras palavras, a equipe que está implementando o sistema que necessita da informação não necessita conhecer os detalhes de acesso e/ou tecnologia do sistema que está provendo a informação. Mais informações sobre o conceito de barramento de serviços (enterprise service bus em inglês) podem ser encontradas em Chappell (2003).

5.3.1 Projeto Um

Este projeto é a implementação da integração entre dois sistemas: Sistema A e Sistema B. O prazo de execução é de duas semanas.

A equipe deste projeto é de quatro pessoas, além do autor que é o líder técnico:

- Uma pessoa responsável por desenvolver no Sistema A o acesso ao barramento de serviços, obter as informações, e processar essas informações de acordo com as especificações.
- Uma pessoa responsável por desenvolver no Sistema B o serviço que disponibiliza as informações necessárias para o Sistema A.
- Uma pessoa responsável por realizar as implementações necessárias no barramento de serviços, para que o serviço criado no Sistema B fique disponível no barramento para acesso pelos outros sistemas.
- Uma pessoa, com dedicação parcial, responsável pela infraestrutura eletrônica necessária para a implementação da integração nos três ambientes de trabalho na empresa: desenvolvimento, homologação e produção.

Os membros da equipe são escolhidos pelos supervisores responsáveis para cada um dos sistemas de software (Sistema A e B). No caso do representante de infraestrutura, a escolha é do gerente de infraestrutura, e no do barramento de serviço, a escolha é do supervisor de área de integração e arquitetura. Neste momento, como representado na figura 17, os alphas do Kernel estão nos seguintes estados:

- Stakeholders: estado Representado.
- Oportunidade: estado Valor Estabelecido.
- Requisitos: estado Delimitado.
- Sistema de software: estado: Arquitetura Seleccionada.
- Trabalho: estado: Preparado.
- Equipe: não atingiu nenhum estado.
- Forma de trabalho: não atingiu nenhum estado.

Na primeira reunião de equipe, o autor explica o funcionamento do Essence. Essa explicação ocorre utilizando a figura 17, que contextualiza o processo da organização e os alphas do Kernel do Essence, e os cartões de que representam os estados dos alphas. A figura 7 e a figura 8 são exemplos desses cartões e a figura 18 apresenta os cartões colocados sobre uma mesa⁶⁵, a disposição dos cartões na mesa representa os estados dos alphas neste momento do projeto.

⁶⁵ Cada linha representa um alpha, a sequencia de cartões na linha representa a sequencia de estados do alpha. Quando há um espaço entre dois cartões de uma linha, deve-se entender que os cartões à esquerda desse espaço representam os estados já atingidos, e os cartões a direita do espaço representam os estados a serem alcançados. Esse método de manipular os cartões foi utilizado durante todo o projeto, ele possibilita o entendimento de todos sobre qual checklist de qual estado se está falando a cada momento da reunião. Informações sobre a forma de utilizar os cartões estão disponíveis no item 2.10.2.

Figura 18 – Estados dos alphas do Kernel no início da primeira reunião da equipe do projeto..

Fonte: Produção do autor.

Após apresentar o Kernel e seus alphas, o autor apresenta a forma reduzida do Código de Ética e Prática profissional da engenharia de software (IEEE/ACM,1999), e a extensão do checklist do alpha forma de trabalho. A seguir, o autor explica o objetivo de utilizar o Essence e a extensão do alpha forma de trabalho: a consideração de fatores humanos na engenharia de software. O autor também esclarece que ao final do projeto ele precisa conversar com todos para identificar se o objetivo foi alcançado.

O autor conduz a reunião avaliando com a equipe o checklist do próximo estado de cada um dos alphas. Não é objetivo desta prova de conceito detalhar o uso do Kernel

do Essence, por essa razão é apresentado apenas o checklist realizado para o alpha forma de trabalho. A equipe identifica que o estado Princípios Estabelecido do alpha forma de trabalho está alcançado. Ao analisar o próximo estado: Fundamento Estabelecido, a equipe conclui que esse estado também está alcançado. O checklist de cada estado, e o motivo de cada item ser considerado como realizado, é apresentado a seguir. Em negrito é destacado o item que suplementa o checklist de cada estado

- Estado: Princípios Estabelecidos, checklist:
 - Princípios e restrições são aceitos pela equipe.
Item realizado: o processo da organização não permite que a equipe tenha liberdade de negociar princípios e restrições. Todos concordam que este item está realizado, pois não há o que fazer.
 - Princípios e restrições são acordados com os stakeholders.
Item realizado: no processo da organização só é possível negociar restrições com stakeholders quando é identificado um problema ou impossibilidade de prosseguir na fase de Especificação e Design. Todos concordam que não há problemas identificados.
 - As necessidades de ferramentas para o trabalho estão acordadas.
Item realizado: as ferramentas já estão definidas pela área de TI da organização.
 - Uma recomendação para abordagem a ser utilizada está disponível.
Item realizado: a equipe de integração e arquitetura determina um padrão para os sistemas que disponibilizam informações (serviços) e para os que consomem esses serviços pelo barramento corporativo.
 - O contexto no qual a equipe irá operar é compreendido.
Item realizado: todos os membros da equipe concordam em participar do projeto e da prova de conceito. O autor é o responsável pela associação entre: atividades da equipe e a manipulação dos cartões. Todos concordam com reuniões diárias de trinta minutos para acompanhamento das atividades e dos checklists dos estados dos alphas.
 - As restrições em relação a seleção, aquisição e uso de práticas e ferramentas são conhecidas.

Item realizado: as restrições são conhecidas e fazem parte do dia a dia da área de TI da organização.

- **Os membros da equipe aderem ao Código de Ética e Prática Profissional da engenharia de software.**

Item realizado: os membros da equipe conversam sobre os Princípios do Código e decidem aderir

- Estado: Fundamentos Estabelecidos, checklist:

- As principais práticas e ferramentas que fundamentam a forma de trabalho estão selecionadas.
- A equipe está de acordo com as práticas que serão utilizadas para início do trabalho.
- Todas as práticas e ferramentas não negociáveis foram identificadas.
- As diferenças que existem entre as práticas e ferramentas que são necessárias, e aquelas que estão disponíveis, foram analisadas e compreendidas.
- A diferença de capacidade técnica entre o que é necessário para executar a forma de trabalho desejada, e a capacidade técnica da equipe foi analisada e compreendida.
- As práticas e ferramentas selecionadas foram integradas e constituem uma forma de trabalho utilizável.

Itens acima realizados: os membros da equipe não possuem a liberdade para alterar as práticas já utilizadas no dia a dia da área de TI. Utilizar o Essence neste projeto não traz alterações na forma de trabalho que é utilizada pelos membros da equipe.

- **Os membros da equipe concordam que as práticas e ferramentas selecionadas aderem ao Código de Ética e Prática Profissional da engenharia de software.**

Item realizado: não são identificadas restrições entre o Código de Ética e Práticas Profissionais e as práticas utilizadas pela equipe.

Ao término da reunião, a disposição dos cartões está na forma apresentada na figura 19. Não há alterações nos estados dos alphas: stakeholders, oportunidade, requisitos

e sistema de software. O alpha trabalho está no estado Começado, o alpha equipe está no estado Formada e o alpha forma de trabalho está no estado Fundamentos Estabelecidos.

Figura 19 – Estados dos alphas do Kernel após primeira reunião.

Fonte: Produção do autor.

Nos dois dias seguintes, as duas reuniões ocorrem como previsto, e o projeto progride como esperado. Nessas duas reuniões o checklist do estado Em Uso do alpha forma de trabalho é percorrido item por item, para identificar se a o alpha já está nesse estado, e, se não está, identificar o que pode ser feito para alcançá-lo. Na reunião do terceiro dia, os membros da equipe concordam que todos os itens do checklist desse

estado, incluindo a extensão, estão realizados, o alpha forma de trabalho está no estado Em Uso.

No quarto dia do projeto, ocorre um imprevisto. O autor não comparece à reunião devido a um problema de lentidão em um dos principais sistemas corporativos da organização. O problema no sistema pode comprometer a operação da organização, e a direção de TI pede para o autor parar todas as atividades e auxiliar no entendimento e solução do problema. A emergência na solução do problema é necessária porquê esse dia, e o seguinte, são os dias do mês em que duas áreas de negócio da empresa, que utilizam esse sistema, não podem ter a produtividade comprometida. Caso isso aconteça, podem ocorrer perdas financeiras para a organização. Entre a identificação do problema, solução e implantação da solução, se passam três dias. O Projeto Um entra no sétimo dia, e só ocorreram as três reuniões diárias, a equipe ficou três dias sem liderança técnica para sanar possíveis dúvidas e problemas de relação entre equipes. No sétimo dia do projeto, os trinta minutos da reunião diária não são suficientes para percorrer todos os alphas, isso porquê ocorre um debate entre os membros da equipe sobre a interpretação de dois itens do checklist suplementar do estado No Lugar, do alpha forma de trabalho. Esses itens são os seguintes:

- As decisões dos membros da equipe não estão sendo influenciadas por outros profissionais ou outros interesses, independentemente de quaisquer vantagens pessoais.
- Os gerentes e líderes da equipe não estão realizando ou promovendo uma abordagem antiética no gerenciamento do desenvolvimento, ou da manutenção, do sistema de software em qualquer aspecto: técnico, profissional, comportamental ou em aspectos relativos às pessoas da equipe.

As interpretações dos membros da equipe sobre esses dois itens se resumem nas seguintes considerações:

- Se o autor, como líder técnico, não irá pressionar a equipe sobre possíveis ações que deveriam ter ocorrido, e não ocorreram, nos três dias de ausência

do líder. Ou se, como líder, o autor irá pressionar para que eles façam horas extras, para superar possíveis atrasos. Comentário comum entre os membros da equipe: “não temos culpa por ter ficado três dias sem liderança técnica”.

- É correto “surpreender” as pessoas com o deslocamento de uma atividade para outra? É correto fazer isso e nem perguntar se a pessoa quer, ou que impacto essa movimentação pode trazer ao grupo de pessoas que trabalham com ela?

Na reunião do oitavo dia, há o registro da progressão dos estados dos alphas. A equipe está na fase de Desenvolvimento. Considerando os alphas da área Empreitada, os estados são:

- Trabalho - estado: Sob Controle.
- Equipe – estado: Realizando.
- Forma de trabalho – estado: Em Uso.

Os membros da equipe ainda não conseguem chegar a um acordo sobre os dois itens do checklist complementar do estado No Lugar, do alpha forma de trabalho, repetidos abaixo:

- As decisões dos membros da equipe não estão sendo influenciadas por outros profissionais ou outros interesses, independentemente de quaisquer vantagens pessoais.
- Os gerentes e líderes da equipe não estão realizando ou promovendo uma abordagem antiética no gerenciamento do desenvolvimento, ou da manutenção, do sistema de software em qualquer aspecto: técnico, profissional, comportamental ou em aspectos relativos às pessoas da equipe

No nono dia, a equipe é surpreendida com a ausência do representante da área de integração e arquitetura. Esse é o dia previsto para o início da montagem da integração no ambiente de homologação. Após a reunião, o autor recebe a seguinte informação: a gerência da área de arquitetura e integração determinou que o membro

da equipe, que é da área de integração e arquitetura, parasse o que estava fazendo para realizar uma tarefa classificada como emergencial. Como resultado, a tarefa programada para o nono dia do projeto ocorre no décimo dia, quando os membros da equipe fazem horas extras para concluir a montagem do ambiente de homologação, além das tarefas já previstas para esse dia.

A decisão de realizar as horas extras não ocorre por decisão da equipe ou do líder. A gerência responsável pelo Sistema A não aceita as justificativas do atraso, e exige que a equipe entregue o ambiente de homologação funcionando o mais rápido possível.

Em resumo, nas demais reuniões diárias é constatado que os estados de todos os alphas progridem, menos o do alpha forma de trabalho, que não sai do estado Em Uso. Os membros da equipe decidem que os dois itens do checklist suplementar do estado No Lugar, comentados anteriormente, não foram realizados. O projeto é entregue na data acertada inicialmente, e, ao conversar com os membros da equipe, o autor recebe a seguinte realimentação:

- Não se sentiram satisfeitos em participar da empreitada. Essa afirmação ocorre quando os membros da equipe observam as informações obtidas no item 4.4.3, em especial a figura 12 (Representação dos fatores relacionados a satisfação utilizando linguagem sistêmica).
- A equipe teve de aceitar interferências de outras pessoas que afetaram o ritmo de trabalho.
- Precisaram realizar horas extras devido compromissos assumidos por outras pessoas.
- Destacaram que o aspecto positivo dessa empreitada, que a diferencia de outras que já participaram, é conhecer o Kernel do Essence, acreditam que a especificação é útil, mas que não funciona em empresas que possuem abordagens tradicionais de desenvolvimento de software.
- Sobre o Código de Ética e Prática Profissional da engenharia de software, eles comentam que saber que existe esse código faz com que se sintam desanimados com o dia a dia que enfrentam na organização.
- Concluem afirmando que não houve a consideração de fatores humanos na engenharia de software.

Questionados pelo autor, para sugerirem mudanças na prática utilizada, os membros da equipe não apresentam nenhuma sugestão, apenas concluem: “o problema é a ‘politicagem’, o interesse das pessoas que decidem”.

5.4 CONSIDERAÇÕES FINAIS

O uso das práticas identificadas com o RSM, através da extensão do Kernel do Essence, não se mostrou satisfatório para tratar o problema de consideração de fatores humanos na engenharia de software. A prova de conceito identifica que o conjunto de práticas utilizadas nem soluciona, nem resolve, nem dissolve o problema. Os objetivos esperados não foram alcançados.

6 RESULTADOS E DISCUSSÃO

Este capítulo apresenta a discussão dos resultados obtidos através da prova de conceito. Também são apresentadas considerações sobre o problema e a resposta ao problema da consideração de fatores humanos na engenharia de software

6.1 CONSIDERAÇÕES INICIAIS

A prova de conceito indica que existe pelo menos um caso em que a utilização do conjunto de práticas, obtidas para consideração de fatores humanos na engenharia de software, não alcança os objetivos esperados.

A falha pode estar relacionada ao conjunto de práticas em si ou na extensão do Kernel do Essence ou, ainda, ter sido um caso isolado.

6.2 SOBRE A PROVA DE CONCEITO

O retorno dado pelos dos membros da equipe que participaram da prova de conceito foi muito claro: “não houve a consideração de fatores humano na engenharia de software”. O checklist estendido do alpha forma de trabalho possibilitou que todos os membros da equipe avaliassem quase que diariamente se estavam aderentes aos Princípios do Código de Ética e Prática Profissional da engenharia de software. Implementar o checklist como negação permitiu que a cada item do checklist os membros da equipe refletissem sobre: **não estar agindo** ou **não estar seguindo** ou **não estar sendo influenciados**.

As razões pelas quais o checklist do estado No Lugar não foi realizado estão relacionadas a fatores externos à equipe. Interferências de pessoas externas à equipe impactaram diretamente o dia a dia dos membros. Os dois itens do checklist que os membros da equipe não consideraram como realizados foram:

- As decisões dos membros da equipe não estão sendo influenciadas por outros profissionais ou outros interesses, independentemente de quaisquer vantagens pessoais.
- Os gerentes e líderes da equipe não estão realizando ou promovendo uma abordagem antiética no gerenciamento do desenvolvimento, ou da manutenção, do sistema de software em qualquer aspecto: técnico,

profissional, comportamental ou em aspectos relativos às pessoas da equipe.

As interferências das pessoas externas à equipe, determinantes para a equipe não considerar os itens acima como realizados, foram as seguintes:

- Deslocar o líder da equipe para outra atividade de forma inesperada (pedido da direção da área de TI).
- Deslocar um dos membros da equipe para outra atividade de forma inesperada (pedidos de um gerente da área de TI).
- Exigir que os membros equipe trabalhassem horas extras (pedido de um gerente de TI).

Essas interferências, quando consideradas utilizando a interação entre os sistemas contenedores identificados na aplicação do RSM (figura 15), permite identificar que as interações entre o **Sistema de necessidade da organização** e o **Sistema de necessidade de equipe** ocorreram por interesse e não levaram em consideração as pessoas envolvidas.

6.3 CULTURA

Schein (1984) afirma que cultura organizacional é o padrão de premissas básicas que um grupo de pessoas inventa, descobre ou desenvolve no processo de aprender a lidar com seus problemas de (i) adaptação do grupo com o mundo externo, e (ii) integração interna.

Devido à possibilidade de que as interferências das pessoas externas ao grupo sejam parte da cultura da organização na qual ocorreu a prova de conceito. O autor tentou utilizar novamente o Kernel estendido. Desta vez, em um projeto de terceirização, com o intuito de diminuir a possibilidade de interferência das pessoas da organização sobre a empreitada. A equipe deste projeto teve três pessoas: o autor e uma pessoa da área de infraestrutura, representando a organização, e o representante de uma fábrica de software, para a qual foi terceirizada a codificação do sistema de software. Não há a necessidade de contextualizar todas as reuniões que ocorreram, visto que o objetivo

também não foi alcançado. Itens do checklist estendido do estado Em Uso do alpha forma de trabalho não foram realizados. Esses itens são:

- Os membros da equipe não estão agindo de uma forma que não é no melhor interesse de seu cliente e empregador.
- Os membros da equipe asseguram que seus produtos, e as modificações de produtos, não estão seguindo padrões pobres ou inadequados.

As justificativas para esses itens não serem considerados realizados são:

- Atrasos as entregas parciais por parte da fábrica;
- A inspeção de código entregue pela fábrica, em sua primeira entrega parcial, identificou problemas no uso de padrões de projetos (design patterns).

Essa segunda experiência realizada pelo autor indica a existência de outros cenários nos quais a utilização do conjunto de práticas, obtidas para consideração de fatores humanos na engenharia de software, não alcança os objetivos esperados.

6.4 ÉTICA

O autor utilizou o RSM, um sistema de métodos que possibilita o consenso entre as pessoas, para a identificação de fatores humanos na engenharia de software, condição necessária devida à iniciativa do autor de não determinar ele mesmo os fatores humanos a serem considerados para a obtenção das práticas. Através do RSM, o autor identificou que a ética, como conjunto de valores e princípios que guia a conduta das pessoas, está presente nas interações que ocorrem na empreitada de engenharia de software. Como a engenharia de software possui um Código de Ética e Prática Profissional (IEEE/ACM, 1999), o autor propôs a extensão do Kernel do Essence como forma de considerar esse código nas interações existentes na empreitada de engenharia de software, como representado na figura 16.

Contudo, o autor identificou, através da experiência prática, que seguir um Código de Ética e Prática Profissional não é uma simples ação de alinhamento entre os princípios, e restrições, que moldam o fluxo de trabalho e os princípios do Código. Cada pessoa possui seu próprio conjunto de valores e princípios, que influenciam

suas decisões e as interações com outras pessoas. Quando organizadas como uma equipe, as pessoas desenvolvem uma compreensão compartilhada de um conjunto de valores e princípios, que permite que a equipe desenvolva confiança, ou desconfiança, no comportamento de cada um individualmente e de cada um como membro da equipe.

6.5 CONSIDERAÇÕES FINAIS

A experiência adquirida na prova de conceito, e na segunda tentativa de usar o Kernel, comentada acima, permitem ao autor afirmar que considerar a ética nas interações existentes na empreitada de engenharia de software não é um simples caso de se utilizar um checklist, de verificar o que deve ser feito para certificar que realizou algo. Considerar a ética é mais do que isso. É necessário que todas as pessoas envolvidas tenham consciência da importância da ética, do respeito de um ao outro e à sociedade. Um ambiente de trabalho onde essa consciência sobre a ética esteja presente depende das pessoas que se relacionam nesse ambiente. Essas pessoas que devem considerar as outras pessoas em suas decisões, e, no limite dessas relações, a sociedade. Contudo, o início da construção desse ambiente deve estar na formação das pessoas que atuam na empreitada de engenharia de software, no ensino da ética nos cursos de engenharia.

7 CONSIDERAÇÕES FINAIS

Este capítulo apresenta a sinopse da tese, as principais contribuições deste trabalho e as observações de ordem prática sobre a consideração de fatores humanos na engenharia de software, seguidas pelas considerações sobre o cumprimento dos objetivos e recomendações de trabalhos futuros.

7.1 SINÓPSE DA TESE

O autor utilizou o RSM, um sistema de métodos que possibilita o consenso entre as pessoas, para a identificação de fatores humanos na engenharia de software. Através do RSM, o autor identificou que a ética, como conjunto de valores e princípios que guia a conduta das pessoas, está presente nas interações que ocorrem na empreitada de engenharia de software. Como a engenharia de software possui um Código de Ética e Prática Profissional (IEEE/ACM, 1999), o autor propôs a extensão do Kernel do Essence como forma de considerar esse código nas interações existentes na empreitada de engenharia de software.

A prova de conceito, que utilizou a extensão do Kernel em um projeto de desenvolvimento de sistemas de software, indicou que conjunto de práticas utilizadas nem soluciona, nem resolve, nem dissolve o problema.

O objetivo mais amplo desta pesquisa não foi alcançado.

7.2 PRINCIPAIS CONTRIBUIÇÕES

A principal contribuição deste trabalho é o resultado da aplicação do sistema de métodos, que possibilitou identificar que, para a consideração de fatores humanos na engenharia de software, é preciso que exista ética nas interações que ocorrem no sistema de atividades humanas que está presente no processo de desenvolvimento de sistemas de software.

Apenas o conhecimento técnico das pessoas não é suficiente para garantir o sucesso da empreitada de engenharia de software de forma contínua, isto é, projeto após projeto. A ética também é necessária, por possuir uma relação direta com a consciência da dimensão social das atividades de engenharia. Apenas quando essa consciência estiver presente nas empreitadas de engenharia de software haverá a consideração de fatores humanos, ou o respeito ao humano, a dignidade humana. E

as pessoas que participam dessa empreitada se sentiram satisfeitas em trabalhar e produzir novos sistemas de software.

A responsabilidade de considerar os fatores humanos na engenharia de software demanda esforços que tem início na própria formação escolar do engenheiro, através de aulas sobre ética nas grades curriculares de engenharia.

7.3 OBSERVAÇÕES DE ORDEM PRÁTICA

Este trabalho oferece uma perspectiva, dentre várias possíveis, sobre os fatores humanos na engenharia de software. É uma perspectiva que resulta de uma avaliação sobre um lugar no desenvolvimento de sistemas de software ocupado pelos entrevistados neste trabalho e pelo autor. Contribuem para essa perspectiva a trajetória de vida e profissional de cada um dos entrevistados, bem como do objetivo de cada um deles em aceitar a dialogar sobre esse tema.

Considerar a ética, nas interações que ocorrem no sistema de atividades humanas que está presente no processo de desenvolvimento de sistemas de software, vai além do “o que fazer” ou do “como fazer”. Embora “o que fazer” e “como fazer” sejam importantes, não são necessariamente ações que permitem considerar o humano como um todo. Para isso, é necessário que se considere princípios conceituais que definem a dignidade humana para toda e qualquer atividade. Este é desafio: criar uma nova cultura. Uma cultura na qual, cotidianamente, estão presentes valores humanos.

7.4 CUMPRIMENTO DOS OBJETIVOS

O objetivo mais amplo desta pesquisa não foi cumprido. A prova de conceito indica que existe pelo menos um caso em que a utilização do conjunto de práticas, obtidas para consideração de fatores humanos na engenharia de software, não alcança os objetivos esperados. Contudo, o fato de o conjunto de práticas ter sido elaborado a partir do Código de Ética e Prática Profissional da engenharia de software permitiu ao autor identificar um outro problema que necessita ser pesquisado: a ética nas interações no sistema de atividades humanas que está presente no processo de desenvolvimento de sistemas de software.

O primeiro objetivo específico desta pesquisa foi cumprido parcialmente. O Kernel do Essence foi estendido, contudo, não possibilitou a consideração de fatores humanos na engenharia de software. Os outros dois itens foram cumpridos: a bibliografia

compilada está nas referências, e nas referências consultadas, desta tese; a contribuição para o conhecimento sobre fatores humanos e sociais está presente nos itens três, quatro, cinco e seis desta tese.

7.5 RECOMENDAÇÕES DE TRABALHOS FUTUROS

A pesquisa realizada neste trabalho identificou algumas oportunidades para a sua continuidade, ou mesmo para a realização de novas pesquisas. As possibilidades identificadas são:

- **Uso da dialética para melhoria da interpretação nas dinâmicas de grupo.**

A construção coletiva através do diálogo indica que o melhor pensamento pode ocorrer de forma coletiva. Contudo, a conclusão, o encerramento, nem sempre é fácil. A construção pelo diálogo exige o consenso, e obter consenso sobre as ideias demanda humildade, de quem defende uma ideia e de quem a contesta, o que nem sempre acontece. O que por vezes acontece é que, no final, há pessoas satisfeitas por verem suas ideias progredirem, e outras pessoas descontentes porque desistiram da argumentação, ou porque não encontraram bons argumentos para defender suas ideias. Por essa razão, nem sempre o consenso é bom. As vezes é necessário que se chegue à melhor interpretação do que está sendo discutido naquele momento, e para o que se deve recorrer à dialética.

- **Extensão do Kernel do Essence para permitir a descrição de práticas que estimulem o debate de ideias.**

Existem empreitadas de engenharia de software que demandam conversas constantes entre os membros da equipe, não apenas sobre questões assertivas, mas principalmente para que ocorra o aprendizado da equipe. Esse é o cenário quando se deseja lidar com o erro humano, o erro do programador, que pode ocorrer por falta de conhecimento, pressão por prazo, falta de familiaridade com as regras que devem ser implementadas, ou ainda, quando a equipe faz parte do processo de inovação de uma organização.

- **Ética na engenharia de software.**

A ética na engenharia de software deve ser pesquisada considerando tanto as decisões de projeto, em todas as fases do ciclo de vida dos sistemas de software, como a relação entre as pessoas que participam da empreitada de engenharia de software; e, não menos importante, o impacto do sistema de software na sociedade.

- **Teoria de sistemas para a engenharia de sistemas e de software.**

A engenharia de sistemas e a engenharia de software carecem de uma base teórica. Elas não possuem uma base teórica que permita avaliar projetos diferentes, que tenham a mesma finalidade. A comparação ocorre mais pela experiência prática dos engenheiros do que por algum critério teórico comum a todos os projetos.

REFERÊNCIAS

ACKOFF, R. L. **Redesigning the Future**: a systems approach to societal problems. New York: John Wiley and Sons, 1974, 260p. ISBN: 9780471002963.

ACKOFF, R. L. **Creating the corporate future**: plan or be planned for. Nova York: John Wiley and Sons, 1981, 297p. ISBN: 978-0471090090.

ACKOFF, R. L. Systems, Messes and Interactive Planning, In: TRIST, E.; EMERY, F.; MURRAY, H. **The Social Engagement of Social Science**: a Tavistock anthology: the socio-ecological perspective. Philadelphia: University of Pennsylvania Press, 1997, v. 3, p. 417-438. ISBN: 9780812281941. Disponível em: <http://www.moderntimesworkplace.com/archives/ericssess/sessvol3/sessvol3.html>. Acesso em: 20 jun. 2015.

AGILE ALLIANCE. **Guide to Agile Practices**. Apresenta uma lista de práticas ágeis com uma descrição que inclui de forma sucinta: definição, benefícios esperados e erros comuns na aplicação da prática. Disponível em: <http://guide.agilealliance.org/#sthash.xP4cQ00s.dpuf>. Acesso em: 09 nov. 2013.

ALBUQUERQUE, L. G. Estratégias de recursos humanos e competitividade. In: VIEIRA, M. M. F., OLIVEIRA, L. M. B. (Org.) **Administração contemporânea: perspectivas estratégicas**. São Paulo: Editora Atlas, 1999, 308p. ISBN-10: 8522422133

AMBLER, S. The non-existent software crisis: debunking the Chaos Report. **Dr. Dobb's**, fev. 2014. Disponível em: <http://www.drdoobs.com/architecture-and-design/the-non-existent-software-crisis-debunk/240165910#>. Acesso em: 12 jun. 2015.

ANDRADE, A. L. **Pensamento Sistêmico**: caderno de campo: o desafio da mudança sustentada nas organizações e na sociedade. Porto Alegre: Bookman, 2006, 488p. ISBN: 8536307005.

APPELO, J. **How to Change the World**: change management 3.0. Rotterdam, The Netherlands: Jojo Ventures BV, 2012. Edição para Kindle Paperwhite v. 5.6.5.

ARCOVERDE, L. Profissionais de TI precisam estar sempre disponíveis. **Valor Econômico**, São Paulo, 16 jul. 2012. Carreira. Disponível em: <http://www.valor.com.br/carreira/2752092/profissionais-de-ti-precisam-estar-sempre-disponiveis>. Acesso em: 20 jul. 2015.

AULETE, C. **Dicionário Caldas Aulete da Língua Portuguesa** - Edição de bolso. L&PM Editores, 2009, 1072p. ISBN: 9788525418548.

AUYANG, S. Y., **Engineering**: an endless frontier. Cambridge, Mass: Harvard University, 2006. ISBN: 9780674019782

BARABÁSI, A. L. **Bursts**: the hidden patterns behind everything we do, from your email to bloody crusades. New York: Plume, 2011, 320 p. ISBN: 978-0452297180.

BARROS FILHO, C.; POMPEU, J. **Somos Todos Canalhas**: filosofia para uma sociedade em busca de valores. Rio de Janeiro: Casa da Palavra, 2015. ISBN: 978-85773455311.

BAXTER, G.; SOMMERVILLE, I., Socio-technical systems: from design methods to systems engineering. **Journal of Interaction with Computers**, v. 23, n. 1, p. 4-17, 2011. DOI=10.1016/j.intcom.2010.07.003

BANATHY, B. H., **A Systems View of Education**: concepts and principles for effective practice. Englewood Cliffs: Educational Technology, 1991, 247p. ISBN: 978-0877782452.

BERTALANFFY, L. von, **General System Theory**: foundations, development, applications. Revised ed. New York: George Braziller, 2006, 295 p., ISBN: 978-807604533.

BOEHM, B. W. **Software Engineering Economics**. Englewood Cliffs, New Jersey: Prentice-Hall, 1981. ISBN: 978-0138221225.

BOEHM, B. W. Improving software productivity. **IEEE Computer**, v. 20, n. 9, p. 43-57. Set. 1987. DOI: [10.1109/MC.1987.1663694](https://doi.org/10.1109/MC.1987.1663694)

BOEHM, B. Some Future Trends and Implications for Systems and Software Engineering Processes. **Systems Engineering**, v. 9, n. 1, p. 1-19, 2006a. DOI: [10.1002/sys.20044](https://doi.org/10.1002/sys.20044)

BOEHM, B. A view of 20th and 21st century software engineering. In: **ICSE '06 INTERNATIONAL CONFERENCE ON SOFTWARE ENGINEERING**, 28. **Proceedings...** New York: ACM, 2006b. p.12-29. DOI: [10.1145/1134285.1134288](https://doi.org/10.1145/1134285.1134288)

BOEHM, B. Some Future Software Engineering Opportunities and Challenges. In Nanz, S. (ed.), **The future of software engineering**. Heidelberg Berlin: Springer, 2011. ISBN: 978-3642151873

BOEHM, B. W.; TURNER, T. **Balancing agility and discipline: a guide for the perplexed**. Boston: Addison-Wesley, 2004. Edição para Kindle Paperwhite v. 5.6.5.

BOURQUE, P.; FAIRLEY, R. E., Eds. **Guide to the Software Engineering Body of Knowledge, Version 3.0**, IEEE Computer Society, 2014; Disponível em: www.swebok.org. Acesso em: 12 jun. 2015.

BROOKS, F. P. 1987. No silver bullet: essences and accidents of Software Engineering. **IEEE Computer**, v. 20, n. 4, p.10-19, 1987. DOI: [10.1109/MC.1987.1663532](https://doi.org/10.1109/MC.1987.1663532)

CASTI, J. L. **Complexification**: explaining a paradoxical world through the science of surprise. New York: HarperCollins, 1995. ISBN: 0060925876.

CAPRA, F. **A Teia da vida**. São Paulo: Cultrix, 1996. ISBN: 9788531605567

CHAPPELL, D. A. **Enterprise Service Bus**: theory in practice. Cambridge: O'Reilly, 2004. ISBN: 0596006756

CHAOS REPORT. **CHAOS REPORT 2014**, 2014. Apresenta as características da edição 2014 do CHAOS REPORT. Disponível em: <http://blog.standishgroup.com/post/18>>. Acesso em: 12 jun. 2014.

CHECKLAND, P. **Systems Thinking, Systems Practice: Includes a 30-Year Retrospective**. Chichester: John Wiley & Sons, 1999. Edição para Kindle Paperwhite v. 5.6.5.

CHIAVENATO, I. **Gerenciando com as Pessoas**: Transformando o Executivo em um Excelente Gestor de Pessoas, 5ª Ed. São Paulo: Editora Manole, 2014, 428 p. ISBN: 9788520439876.

CHERRY, S.; ROBILLARD, P.N. The social side of software engineering – a real ad hoc collaboration network. **International Journal of Human-Computer Studies**, v. 6, n. 7, p. 495-505. 2008. DOI= 10.1016/j.ijhcs.2008.01.002.

CLARKE, P.; O'CONNOR, R. V. The situational factors that affect the software development process: towards a comprehensive reference framework. **Information and Software Technology**, v. 54, n. 5, p. 433–447, 2012. DOI: [10.1016/j.infsof.2011.12.003](https://doi.org/10.1016/j.infsof.2011.12.003)

COCKBURN, A. **Agile Software Development**: the cooperative game. 2nd. Ed. New Jersey: Addison-Wesley, 2006. 504 p. ISBN: 978-0321482754.

CORTELLA, M. S.; BARROS FILHO, C. **Ética e Vergonha na Cara**. Campinas, SP: Papirus 7 Mares, 2014, 111 p. ISBN: 978- 8561773489.

DEMARCO, T. **Controlling Software Projects**: management, measurement, and estimation. New York: Yourdon Press, 1986. ISBN: 978-0131717114.

DEMARCO, T.; LISTER, T. R. **Peopleware**: productive projects and teams. 2nd ed. New York, NY: Dorset House, 1999. 245 p. ISBN-13: 978-0932633439.

DEMARCO, T. Software Engineering. An idea whose time has come and gone?. **IEEE Software**, v. 26, n. 4, p. 95-96, 2009. DOI: 10.1109/MS.2009.101

DRUCKER, P. **Desafios Gerenciais para o Século XXI**. São Paulo: Pioneira, 1999. 168 p. ISBN: 8522102007.

EMERY, F. Characteristics of Socio-Technical Systems, In Trist, E. & Murray, H. (Eds.) **The Social Engagement of Social Science**: a Tavistock anthology: the socio-technical perspective. Philadelphia: University of Pennsylvania Press, 1993, v. 2. ISBN: 9780812281934. Disponível em: <http://moderntimesworkplace.com/archives/ericssess/sessvol2/sessvol2.html> >. Acesso em: 12 jun. 2015.

EMERSON. **Os Empregos Mais Exaustivos de TI**. A report from the experts in business-critical continuity, Emerson Network Power, 2012. Disponível em: http://www.emersonnetworkpower.com/documentation/en-us/latest-thinking/documents/pt-br_is03897_mostalwaysonjob_fin.pdf>. Acesso em: 20 Jul. 2015.

ENGSTROM, E.W. Systems engineering: a growing concept. **Electrical Engineering**, v. 76, n.2, p. 113-116, fev. 1957. DOI: 10.1109/EE.1957.6442968.

ESSENCE. **Kernel and Language for Software Engineering Methods (Essence)**, Version 1.0, 2014. OMG Document number: formal/2014-11-02. Disponível em: <<http://www.omg.org/spec/Essence/1.0/>>. Acesso em: 12 jun. 2015

FERNANDEZ-SANZ, L.; SANJAY, M. Influence of human factors in software quality and productivity. In: **Computational Science and Its Application – ICCSA 2011**. Lecture Notes in Computer Science Series, p. 257-269, 2011. Berlin: Springer Berlin Heidelberg. DOI: 10.1007/978-3-642-21934-4_22

FRIEDENTHAL, S.; MOORE, A; STEINER, R. **A Practical Guide to SysML**. Burlington: Morgan Kaufmann, 2008. ISBN: 978-0123473794.

GIL, A. C. **Gestão de Pessoas: enfoque nos papéis profissionais**. São Paulo: Editora Atlas, 2001, 312p. ISBN: 8522429529.

GRIFFYTH, J. Human issues in software development process – modeling their influence on productivity and integrity. In **Safety-Critical Systems: The Convergence of Huge Tech and Human Factors**, Redmill, F; Anderson T, Eds., p. 105-123, 1996. London: Spring London. DOI: 10.1007/978-1-4471-1480-2_7

HALL, A.D. **A Methodology for Systems Engineering**. Princeton, New Jersey: D. Van Nostrand, 1962. 478 p. ASIN: B000L9R3P0

HALL, A. D. **Metasystems Methodology**. Oxford New York: Pergamon, 1989. 518p. ISBN: 978-0080369563.

HITCHINS, D. K. **Putting System to Work**. New York: John Wiley & Sons, 1992. 324p. ISBN: 978-0471934267.

HITCHINS, D. K. **Advanced Systems Thinking, Engineering, and Management**. Norwood: Massachusetts, 2003. 469p. ISBN: 978-1580536196.

HITCHINS, D. K. **Systems Engineering: a 21st century systems methodology**. Chichester: John Wiley & Sons, 2007. 528p. ISBN: 978-0470058565

HUGHES, T. P. **Human-Built World: how to think about technology and culture**. Chicago: The University of Chicago Press, 2005. 240p. ISBN 0226359344.

HUMPHREY, W. S.; JAMES W. O. **Leadership, teamwork, and trust: building a competitive software capability**. New Jersey: Addison-Wesley, 2011. 369 p. ISBN 9780321624505.

IEEE/ACM. Apresenta o Código de Ética e prática Profissional na área de engenharia de software. **Software Engineering Code of Ethics and Professional Practice** (v. 5.2), 1999. Disponível em: <<http://www.computer.org/web/education/code-of-ethics#>> e em: <<http://www.acm.org/about/se-code>>. Acessos em: 25 set. 2015.

INCOSE. Systems Engineering Handbook Development Team of the International Council on Systems Engineering (INCOSE). **SYSTEMS ENGINEERING HANDBOOK, v. 3**. INCOSE-TP-2003-002-03, June 2006.

JACOBSON, I.; MEYER, B. Methods Need Theory. **Dr. Dobb's**, Aug. 2009. Disponível em: <<http://www.drdobbs.com/architecture-and-design/methods-need-theory/219100242#>>. Acesso em: 12 jun. 2015.

JACOBSON, I.; SPENCER, B. Why we need a theory for software engineering. **Dr. Dobb's**, Oct. 2009. Disponível em: <<http://www.drdobbs.com/architecture-and-design/why-we-need-a-theory-for-software-engine/220300840#>>. Acesso em: 12 jun. 2015

JACOBSON, I.; MEYER, B.; SOLEY, R., **SEMAT Vision Statement**. 2010. Disponível em: < http://semat.org/?page_id=28 >. Acesso em: 12 jun. 2015.

JACOBSON I.; Ng, P. W.; McMAHON, P. E.; SPENCE, I.; LIDMAN, S.: The Essence of Software Engineering: The SEMAT Kernel. **Queue**, vol. 10, no. 10, pp. 40-52, 2012. DOI: 10.1145/2381996.2389616

JACOBSON I.; Ng, P. W.; McMAHON, P. E.; SPENCE, I.; LIDMAN, S.: **The Essence of Software Engineering – Applying the SEMAT Kernel**. New Jersey: Addison-Wesley Professional, 2013, 352p. ISBN: 978-0321885951.

JACKSON, M.C., **Systems Thinking**: creative holism for managers. Chichester: England: John Wiley & Sons, 2003, 378p.. ISBN: 0470845228.

KLIR, G. J. **Facts of systems science**. New York: Springer, 2nd ed., 2001. 748 p. ISBN 978-0306466236.

KOPPENSTEINER, S. Overview of practices and methods to influence human factors in software projects. In: **IDIMT-2009 INTERDISCIPLINARY INFORMATION AND MANAGEMENT TALKS CONFERENCE**, 17. **Proceedings**. Jindřichův Hradec, Czech Republic: Trauner, 2009, p. 159-165, 2009. Disponível em: <http://www.idimt.org/idimt_proceedings_2009.pdf>. Acesso em: 12 jun. 2015.

KOSSIAKOFF A.; SWEET W. N.; SEYMOR, S. J.; BIEMER, S M. **Systems Engineering Principles and Practice**, 2nd ed.. Hoboken: John Wiley & Sons Inc., 559p. 2011. ISBN-13: 978-0470405482.

LEHMAN, M.M. On understanding laws, evolution, and conservation in the large-program life cycle. **Journal of Systems and Software**, v. 1, p. 213-221, 1980. DOI: 10.1016/0164-1212(79)90022-0

LEHMAN, M.M. Software engineering, the software process and their support. **Software Engineering Journal**, v.6, n.5, p.243-258, set. 1991. Disponível em: <<http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=107025&isnumber=3263>>. Acesso em: 12 jun. 2015.

MATÉ, J. L.; SILVA, A. **Requirements engineering for sociotechnical systems**. London: Information Science, 2005, 373p. ISBN 9781591405078.

MARRAS, J. P. **Administração de Recursos Humanos: Do Operacional ao Estratégico**, 14^a ed. São Paulo: Saraiva, 2011, 352 p. ISBN: 978-8502125605.

MATURANA, H. R.; VARELA, F. J. **Autopoiesis and Cognition: The realization of the Living**. Dordrecht, Holland Boston: E. Reidel , 1980.

MEADOWS, D. **Thinking in Systems: A Primer**. London, Sterling, VA: Earthscan, Edited by Wright, D., 2009, 235p. ISBN: 978184477267.

MITCHELL, M. **Complexity: a guided tour**. New York, NY: Oxford University, 2009. ISBN: 978-0195124415. Edição para Kindle Paperwhite v. 5.6.5.

MOREIRA, M. A. Mapas Conceituais e Aprendizagem Significativa. São Paulo: Centauro, 2010, 80p. ISBN: 978-8579280054

MORIN, E. From the concept of system to the paradigm of complexity. **Journal of Social and Evolutionary Systems**, v. 15, n. 4, p. 371-385, 1992. DOI: 10.1016/1061-7361(92)90024-8

MORIN, E.; CIURANA, E. R.; MOTTA, R. D., **Educar na Era Planetária: O pensamento complexo como método de aprendizagem pelo erro e incerteza humana**. São Paulo: Ed. Cortez, 2009. ISBN: 978-8524909375.

NICOLESCU, B. **Manifesto da Transdisciplinaridade**. São Paulo: Trion, 1999. ISBN: 978-8585464226.

PARNELL, G. S.; DRISCOLL, P. J.; HENDERSON, D. L. **Decision making in systems engineering and management**. Hoboken, N. J: Wiley, 2011, 544p. ISBN: 978-0470900420.

PETRAGLIA, I., Sete Ideias Norteadoras da Relação Educação e Complexidade. In: **Estudos de Complexidade**, Almeida, C. R.; Petraglia, I. (org.); São Paulo: Xama, 2006.

POSER, H. On structural differences between science and engineering. **Philosophy and Technology**, 1998, v. 4, n. 2, p. 81-93. Disponível em: <<http://scholar.lib.vt.edu/ejournals/SPT/v4n2/POSER.html>>. Acesso em: 12 Jun. 2015.

POPPENDIECK, M.; POPPENDIECK, T. **Lean software development: an agile toolkit**. Upper Saddle River, NJ: Addison-Wesley, 2003. 240 p.

PRESSMAN, R. S., **Software Engineering: A Practitioner`s Approach**, 7th ed., New York: McGraw-Hill, 2010, 930 p., ISBN: 978-0073375977.

RIBEIRO, R. J. Ética e Direitos Humanos. **Interface**, Botucatu, v. 7, n. 12, p. 149-166, fev. 2003. Disponível em: <http://www.scielo.br/scielo.php?script=sci_arttext&pid=S1414-32832003000100015&lng=en&nrm=iso>. Acesso em: 26 set. 2015.

RHODES, D.; HASTINGS, D., **The case for Evolving Systems Engineering as a Field within Engineering Systems**, MIT Engineering Systems Symposium, March, 2004, Disponível em: <<http://esd.mit.edu/symposium/pdfs/papers/rhodes.pdf>>. Acesso em: 12 jun. 2015.

ROBERTSON, S.; ROBERTSON, J. **Mastering the Requirements Process**, 2nd Edition. Boston: Pearson Education., 2006, 560 p. ISBN 0321419499.

SCACCHI, W. Managing software engineering projects: a social analysis. **IEEE Transaction on Software Engineering**, v. 10, n.1, p. 49-59, 10984. DOI: [10.1109/TSE.1984.5010198](https://doi.org/10.1109/TSE.1984.5010198)

SCRUM.ORG – Scrum.org Improving the Profession of Software Development. Apresenta recursos e métodos desenvolvidos por praticantes do SCRUM e especialistas em métodos ágeis para entregar sistemas de valor utilizando SCRUM. Disponível em: <<http://www.scrum.org>>. Acesso em: 12 jun. 2015.

SEI - CARNEGIE MELLON UNIVERSITY. **SEI** - Methods & Tools. Apresenta ferramentas e métodos desenvolvidos pelo Software Engineering Institute - SEI. Disponível em: <<http://www.sei.cmu.edu/tools/?location=secondary-nav&source=1358>>. Acesso em: 09 Nov. 2013.

SEMAT - Software Engineering Method and Theory. Apresenta recursos e atividades devolvidas pela iniciativa SEMAT. Disponível em: <<http://www.semat.org>>. Acesso em: 12 jun. 2015.

SHEIN, E. H. Coming to a new awareness of organizational culture. **MIT Sloan Management Review**, v. 22, n.2, p. 3-16. Winter, 1984. Disponível em: <<http://sloanreview.mit.edu/article/coming-to-a-new-awareness-of-organizational-culture/>>. Acesso em 10 set. 2015.

SIMONETTE, M.; SANCHES, F.; SPINA, E. Beyond human factors. In **CIT'09 INTERNATIONAL CONFERENCE ON COMMUNICATIONS AND INFORMATION TECHNOLOGY**, 3. **Proceedings**., Vouliagmeni, Athens: WSEAS, 2009, p. 240-244. ISBN 9604741465. Disponível em: <<http://www.wseas.us/e-library/conferences/2009/vouliagmeni2/CIT/CIT-42.pdf>>. Acesso em: 09 set. 2015.

SIMONETTE, M. **Engenharia de Sistemas em Sistemas Sociotécnicos**. São Paulo, 2010. Dissertação (Mestrado) – Escola Politécnica, Universidade de São Paulo.

SIMONETTE, M.; SPINA, E. The Human Side of Software Development – Teamwork as Sociotechnical Systems. In ZAPATA, Carlos, et al. (Ed.) **Software Engineering: Methods, Modeling and Teaching**, v. 2, Lima: Sello Editorial - Pontificia Universidad Católica del Perú, 2012. Cap. 3. p. 21-28. ISBN 9786124057847. Disponível em: <<http://lases.cidenet.org/images/content/libro-completo-2012.pdf>>. Acesso em: 12 jun. 2015.

SKYTTNER, L. **General systems theory**: problems, perspectives, practice. New Jersey: World Scientific, 2nd ed., 2005, 524 p., ISBN: 9812564675.

SOMMERVILLE, I. **Software Engineering**, 9th ed. Boston: Addison-Wiley, 2011, 790p. ISBN 9780137035151.

SOMMERVILLE, I. et al. Large-scale complex IT systems. **Communications of the ACM**, v. 55, n. 7, p. 71-77, 2012. DOI=10.1145/2209249.2209268

STANDISH GROUP. **Big Bang Boom**, 2014. Este relatório utiliza a base de dados dos relatórios CHAOS para fazer comparações entre projetos de grandes sistemas de software. Disponível em: <
https://www.standishgroup.com/sample_research_files/BigBangBoom.pdf>. Acesso em: 12 jun. 2015.

STASINOPOULOS, P. et al. **Whole System Design**: an integrated approach to sustainable engineering. London-Sterling, VA: Earthscan, 2009, 183 p., ISBN: 9781844076437.

STROKES, D. E. **Pasteur's Quadrant**: basic science and technological innovation. Washington, D.C.: Brookings Institution Press, 1997, 228 p., ISBN: 0815781776.

SYDENHAM, P. H. **System Approach to Engineering Design**. Norwood: Artech House, 2003, 333p. ISBN: 1580534791.

TAVARES, R. Construindo mapas conceituais. **Ciências & Cognição**, v.12, p. 72-85, 2007. Disponível em: <<http://www.cienciasecognicao.org/pdf/v12/m347187.pdf>>. Acesso em: 12 jun. 2015.

TRIST, E.; EMERY, F.; MURRAY, H. **The Social Engagement of Social Science**: a Tavistock anthology: the socio-technical perspective. Philadelphia: University of Pennsylvania, 1993, v. 2, 712 p. ISBN: 9780812281934. Disponível em: <
<http://www.moderntimesworkplace.com/archives/ericssess/sessvol2/sessvol2.html> >. Acesso em: 20 jun. 2015.

TURNER, R.; BOEHM, B. People factors in software management: Lessons for comparing agile and plan-driven methods. **Crosstalk**: the journal of defense software engineering, v. 16, n. 12, p. 4-8, Dec. 2003. Disponível em: <
<http://www.crosstalkonline.org/storage/issue-archives/2003/200312/200312-0-Issue.pdf>>. Acesso em: 12 jun. 2015.

VALERIANO, D. L. **Gerenciamento Estratégico e Administração por Projetos**. São Paulo: Makron Books, 2001, 295 p. ISBN: 978-8534612081.

VALERIANO, D. L. **Moderno Gerenciamento de Projetos**. São Paulo: Pearson Prentice Hall, 2005, 272p. ISBN: 8576050390.

VOSBURGH, J.; CURTIS, B.; WOLVERTON, R.; ALBERT, B.; MALEC, H.; HOBEN, S.; LIU, Y. Productivity factors and programming environments. In: **ICSE '84 INTERNATIONAL CONFERENCE ON SOFTWARE ENGINEERING**, 7. **Proceedings**. Piscataway, NJ: IEEE, 1984, p. 143-153. Disponível em: <
<http://dl.acm.org/citation.cfm?id=801963>>. Acesso em: 12 jun. 2015.

WALDROP, M. M. Complexity: **The emerging science at the edge of order and chaos**. New York: Simon & Shcister, 1992. ISBN: 9780671767891.

WALSTON, C. E.; FELIX, C. P. A method of programming measurement and estimation. **IBM Systems Journal**, v. 16, n. 1, p. 54-73, 1997. DOI: [10.1147/sj.161.0054](https://doi.org/10.1147/sj.161.0054)

WEIHRICH, H. The TOWS Matrix – A Tool for situational analysis. **Long Range Planning**, v. 15, n. 2, p. 54-66, April 1982. DOI: 10.1016/0024-6301(82)90120-0

WEINBERG, G. M. **The psychology of computer programming: Silver Anniversary Edition**. New York: Weinberg & Weinberg, 2011a. Edição para Kindle Paperwhite v. 5.6.5.

WEINBERG, G. M. **The Secrets of Consulting: a guide to giving and getting advice successfully**. New York: Weinberg & Weinberg, 2011b. Edição para Kindle Paperwhite v. 5.6.5.

WRIGHT, P. M.; BOSWELL, W. R. Disaggregating HRM: a review and synthesis of micro and macro human resource management research. **Journal of Management**, v. 28, n. 3, p. 247-76. 2002. DOI: 10.1177/014920630202800302.

REFERÊNCIAS CONSULTADAS

AHN, L. V.; DABBISH, L. Designing games with a purpose. **Communications of the ACM**, 51 (8), 2008, 58-67. doi=10.1145/1378704.1378719

ANDERSON, D. J. **KANBAN**: successful evolutionary change for your technology business. Sequin, Washington: Blue Hole, 2011. Edição para Kindle Paperwhite v. 5.6.5.

ARANHA M. L. A.; MARTINS, M H. **Filosofando: introdução à filosofia**. 2ª ed. São Paulo: Moderna, 1993.

AVIZIENIS, A., LAPRIE, J.-C., RANDELL, B., LANDWEHR, C. Basic concepts and taxonomy of dependable and secure computing. **IEEE Transactions on Dependable and Secure Computing**, v. 1, n. 1, p.11–33, Jan.-Mar. 2004. DOI: 10.1109/TDSC.2004.2

BARROS FILHO, C; MEUCCI, A. **O Executivo e o Martelo**. São Paulo: HSM, 2013, 270p. ISBN: 978-8565482202.

BENCIVENGA, D., A Humanistic Approach to Space, **HR Magazine**, Society for Human Resource Management, March, 1998. Disponível em: <<http://www.shrm.org/Publications/hrmagazine/EditorialContent/Pages/0398COV.aspx>> Acesso em: 16 Nov. 2013.

BIRNHOLTZ, J.; GUTWIN, C.; HAWKEY, K., Privacy in the open: how attention mediates awareness and privacy in open-plan offices. In **GROUP '07 INTERNATIONAL ACM CONFERENCE ON SUPPORTING GROUP WORK. Poceedings**. New York: ACM, 2007, p. 51-60. DOI=10.1145/1316624.1316632

CAMPOLARGO, M. eInfrastructure: Changing How Research is Done. **IEEE Communication Magazine**, v.42, n. 11, p. 31-32, nov. 2004.

CCE – COMISSÃO DAS COMUNIDADES EUROPEIAS - European Commission, EGEE-II, Project; BELIEF, Project; OMII-Europe, Project; DEISA, Project; Geant2, Project. **A guide to European flagship e-Infrastructure projects**. e-Infrastructure Unit, 2007. Disponível em: <http://belief-dl.research-infrastructures.eu/OpenDLib/UI/1.0/Disseminate/eu-egge.org/2009_Publication_02705/Content/pdf?version=1&ip=146.48.87.125&port=8001>. Acesso em: 11 set. 2013.

COCKBURN, A.; HIGHSMITH, J. Agile software development, the people factor. **Computer**, v. 34, n. 11, p. 131-133, nov. 2001. DOI: 10.1109/2.963450

CROWDER, R. M., ROBINSON, M. A., HUGHES, H. P.N., YEE-WAI S. The development of an agent-based modeling framework for simulating engineering teamwork. **IEEE Transaction on Systems, Man and Cybernetics, part A: systems and humans**, v.42, n.6, p.1425-1439, nov. 2012. DOI: 10.1109/TSMCA.2012.2199304.

CURTIS, B.; KRASNER, H.; ISCOE, N. A. Field study of the software design process for large system. **Communications of the ACM**, v. 31, n. 11, p. 1268-1287, 1988. DOI: 10.1145/50087.50089

DIJKSTRA, E. Programing Considered as a Human Activity. In: **IFIP`65 INTERNATIONAL FEDERATION FOR INFORMATION CONGRESS. Proceedings**. New York: North-Holland, 1965. Disponível em: <http://www.cs.utexas.edu/~EWD/transcriptions/EWD01xx/EWD117.html>. Acesso em: 12 jun. 2015.

FREIRE, P. **Ação Cultural para a Liberdade: e outros escritos**. 5a ed. Rio de Janeiro: Paz e Terra, 1970. ISBN: 857753166X.

FREIRE, P. **Pedagogia do oprimido**. 17^a ed. Rio de Janeiro: Paz e Terra, 1987.

GALIMBERTI, U. **Psique e techne: o homem na idade da técnica**. São Paulo: Editora Paulus, 2006. 918p. ISBN: 8534923213.

JACKSON, M.C. **Systems Methodology for the Management Sciences**. New Youk: Springer, 1991. 298p. ISBN: 978-3540713593.

JUGDEV, K.; MOLLER, R., A retrospective look at our evolving understanding of project success. **IEEE Engineering Management Review**, v.34, n.3, p.110, Third Quarter 2006. DOI: 10.1109/EMR.2006.261387.

HEENEY, C., Privacy and the identity gap in socio-technical systems. In **Handbook of Research on Socio-Technical Design and Social Networking Systems**, Whitworth, B.; Moor, A., Eds., p. 110-122, 2009. Hershey, PA: Information Science Reference. DOI:10.4018/978-1-60566-264-0.ch008

HUANG, Y. **Supporting Meaningful Social Networks**. Southampton, 2009. Thesis (PhD). University of Southampton, UK. Disponível em: <http://eprints.soton.ac.uk/267180/>. Acesso em: 21 nov. 2013.

IEEE, IEEE Standard Glossary of Software Engineering Terminology. **IEEE Standard 610.12-1990**, New York, 1990.

KRAUT, R.; MAHER, M. L.; OLSON, J.; MALONE, T. W.; PIROLI, P.; THOMAS, J.C. Scientific Foundations: a case for technology- mediated social- participation theory. **Computer**, v.43, n.11, p.22-28, nov. 2010. DOI: 10.1109/MC.2010.324.

LONG, Y. & SIAU, K., Social Network Structures in Open Source Software Development Teams. In P. Zaphiris & C. S. Ang (eds.), **Human Computer Interaction: Concepts, Methodologies, Tools and Applications**. Hershey, New York. Information Science Reference. ISBN: 9781605660530.

MASCARENHAS, S. Por uma nova engenharia. Entrevista a Elton Alisson. **Agência FAPESP**, São Paulo, set. 2011. Disponível em: http://agencia.fapesp.br/por_uma_nova_engenharia/14428/. Acesso em: 10 jun. 2015.

PMBOK Guide. **A Guide to the Project Management Body of Knowledge**, 4rd Edition. Pennsylvania: Project Management Institute, Inc. 2008, ISBN 9781933890517.

RIOS, I. C. Humanização: a essência da ação técnica e ética nas práticas de saúde. **Revista Brasileira de Educação Médica**, Rio de Janeiro, v. 33, n. 2, p. 253-261, jun. 2009. Disponível em: < <http://dx.doi.org/10.1590/S0100-55022009000200013>>. Acesso em: 12 jun. 2015.

SCHRAIBER, L B. No encontro da técnica com a ética: o exercício de julgar e decidir no cotidiano do trabalho em medicina. **Interface – Comunicação, Saúde, Educação**, v. 1, n. 1, 1997. Disponível em: <<http://www.hcnet.usp.br/humaniza/pdf/No%20encontro%20da%20tecnica%20com%20a%20etica%20o%20exercicio%20de%20julgar%20e%20decidir%20no%20cotidiano%20do%20trabalho%20em%20Medicina.pdf>>. Acesso em: 10 jun. 2015.

SCOTT, J. **Social Network analysis: A Handbook**. London: Sage Publications Ltd., 2000, 208 p., ISBN: 9780761963394.

TSALLIS, C. Mais uma vez Sérgio Mascarenhas tem razão. **Notícias Academia Brasileira de Ciências**, set. 2011. Disponível em: <http://www.abc.org.br/article.php3?id_article=1425>. Acesso em: 10 jun. 2015

WASSERMAN, S.; FAUST, K., Social Network Analysis: Methods and Applications. In: **Structural Analysis in the Social Sciences**, v.8. Cambridge: Cambridge University Press, 1994, 825 p. ISBN: 978052138707

WHITWORTH, B., The social requirements of technical systems. In: **Handbook of Research on Socio-Technical Design and Social Networking Systems**, Whitworth, B.; Moor, A., Eds., p. 3–22, 2009. Hershey, PA: IGI Global. DOI: 10.4018/978-1-60566-264-0.ch001

GLOSSÁRIO

Administração – Administrar - Administrador – referem-se aos problemas típicos das organizações, como finanças, pessoal e patrimônio – Definição extraída de Valeriano (2005, p.12).

Ágil – Ver: **Paradigma Ágil**.

Contenedor – Que contém, abrangedor, do latim: Continere - Definição extraída do dicionário Caldas Aulete (AULETE,2009).

Corporação – Conjunto de pessoas que exercem a mesma atividade profissional, sujeito a mesmas regras ou estatutos, e com os mesmos deveres ou direitos; empresa ou associação de empresas de um ou mais setores de atividade econômica - Definição extraída do dicionário Caldas Aulete (AULETE,2009).

Empreitada – Uma atividade ou conjunto de atividades direcionadas a um objetivo – Definição extraída do documento Essence (2014). Tradução do autor.

Ética – Conjunto de valores e princípio que guia a conduta das pessoas – Definição extraída de Cortella e Barros Filho (2014, p. 24).

Figura rica – Representação gráfica que permite interpretar um problema. Resume uma situação real utilizando figuras que representam os componentes, idéias, relações, conexões influencias existentes na situação problema. Pode incluir elementos subjetivos que expressam dimensões humanas como emoções, aspirações, etc.

Gerenciamento – Gerenciar – Gerência - Gerente – referem-se às ações situadas em um nível específico da organização, seja um departamento, seja um projeto ou ainda em nível mais elevado voltado para a interação da organização com o ambiente – Definição extraída de Valeriano (2005, p.12).

Gestão – Gerir - Gestor – dizem respeito, tanto no âmbito da organização como no do projeto, a setores especializados das atribuições do administrador ou do gerente

do projeto. No projeto, a gestão é uma das partes da gerência, geralmente delegada pelo gerente – pode se constituir em gestão do tempo, gestão dos custos, gestão dos riscos, gestão da qualidade etc. – Definição extraída de Valeriano (2005, p.12).

Gestão de Pessoas – A gestão de pessoas possui dois aspectos, um macro e outro micro: (i) o aspecto micro está relacionado a questões funcionais da gestão de pessoas, como recrutamento, seleção, avaliação, desenvolvimento individual e remuneração. Neste aspecto, a gestão de pessoas tem preocupações com a produtividade, qualidade e satisfação de cada indivíduo; considera as habilidades, capacidades, atitudes das pessoas e utiliza avaliações de desempenho para medir o impacto de suas políticas sobre os indivíduos. (ii) o aspecto macro está relacionado a uma perspectiva estratégica, envolve práticas de gestão de pessoas que possuem a organização como elemento de análise, e não o indivíduo; avalia suas práticas segundo variáveis organizacionais relevantes, ignorando a variância que a política aplicada pode ter entre indivíduos ou grupos de indivíduos, assumindo que nesse nível há uma uniformidade – Definição extraída de Wright e Boswell (1999).

Instituição – Grupo organizado de pessoas que, em determinado local, presta serviços necessários à comunidade ou sociedade, ou que se dedica a certas atividades de interesse privado - Definição extraída do dicionário Caldas Aulete (AULETE,2009).

Lean – Ver: **Paradigma Lean**.

Matriz TOWS – Ferramenta de análise situacional que, através da análise do presente, possibilita a formulação de uma estratégia para o futuro (WEIHRICH, 1982). TOWS é a sigla em inglês para Threats, Opportunities, Weakness e Strengths, respectivamente: Ameaças, Oportunidades, Fraquezas e Forças em português.

Método – Conjunto de regras e critérios que estabelecem uma forma precisa e repetível de atingir um objetivo. Fornece a técnica de como fazer.

Organização – Maneira pela qual um ser vivo ou um sistema se constitui e/ou se organiza; reunião de pessoas com objetivos ou interesse comuns - Definição extraída

do dicionário Caldas Aulete (AULETE,2009). Outra definição relevante para este trabalho é: Um sistema que tem objetivos específicos e uma estrutura organizacional – Definição extraída de Valeriano (2005, p.8)

Paradigma Ágil – O paradigma Ágil é caracterizado por ter equipes de desenvolvimento que seguem processos iterativos e leves, os membros das equipes são colaborativos entre si, auto organizados, e focados em qualidade. Exemplos de aplicação desse paradigma são: SCRUM e Programação Extrema (XP). Nota do autor.

Paradigma Lean – O paradigma Lean é caracterizado por ter equipes de desenvolvimento focadas na filosofia do valor para o cliente, que trabalham continuamente para otimizar o valor do sistema de software para o consumidor, ao mesmo tempo em que reduz o tempo de trabalho, o custo e melhora a qualidade. Exemplos de aplicação desse paradigma são: Kanban e Scrumban. Nota do autor

Paradigma Orientado a Plano – O paradigma Orientado a Plano é caracterizado por ter equipes de desenvolvimento que seguem processos de desenvolvimento inspirados no modelo cascata (waterfall).

Prática – Uma prática é uma descrição de como lidar com um aspecto específico de uma empreitada de engenharia de software. Uma prática fornece uma maneira sistemática e verificável de tratar um aspecto particular do trabalho a ser realizado. Ela tem um objetivo claro expresso em termos de resultados que serão alcançados com sua aplicação. Além de fornecer orientação e guiar os profissionais para alcançar o objetivo, uma prática também garante que o objetivo está compreendido, e permite verificar se ele foi alcançado – Definição extraída do documento Essence (2014). Tradução do autor.

Processo – É uma sequência continua de atividades e que determina pontos de decisão e controle de sua condução. É como um roteiro que fornece estabilidade, controle e organização a uma sequência de atividades. Nota do autor.

Tecnologias da Informação e Comunicação (TIC) – Se refere ao conjunto de tecnologias disponíveis no setor da informática e no setor das telecomunicações que permitem comunicar e partilhar a informação.

APÊNDICES

APÊNDICE A – Teoria Geral de Sistemas

Klir (2001) afirma que, aparentemente, não há dúvidas de que o termo Teoria Geral dos Sistemas é devido a Ludwig Von Bertalanffy, que já o utilizava em palestras nos anos 30, mas o uso desse termo em um livro se deu apenas após a Segunda Grande Guerra Mundial.

Além de ser o idealizador do termo, Ludwig Von Bertalanffy, um biólogo, também teve presença predominante no movimento para a promoção dessa teoria. Em dezembro de 1954 fundou juntamente com Kenneth Boulding (economista), Ralph Gerard (fisiologista) e Anatol Rapoport (bio-matemático) a Sociedade para Pesquisa em Sistemas Gerais, que possuía quatro objetivos fundamentais (CHECKLAND, 1981; KLIR, 2001; ADAMS, MUN, 2005; HITCHINS, 2007):

- 1) Investigar o isomorfismo de conceitos, leis e modelos de várias áreas e auxiliar em transferências úteis de uma área para outra.
- 2) Encorajar o desenvolvimento de modelos teóricos adequados em áreas que tivessem carência deles.
- 3) Minimizar a duplicação de esforços teóricos em diferentes áreas.
- 4) Promover a unidade da ciência através da melhoria da comunicação entre os especialistas.

Hitchins (2007) informa que coube a Kenneth Boulding uma das tarefas mais difíceis desse grupo: Realizar uma classificação coerente dos sistemas. A tabela 1 apresenta essa classificação, sendo que as relações dos níveis com as áreas de conhecimento têm a seguinte interpretação:

- Os três primeiros níveis são relacionados à física, astronomia e as ciências duras. Um sistema desse tipo é considerado como fechado, Hitchins afirma que esses sistemas não possuem um contato significativo com o ambiente e não se adaptam ao ambiente, segundo Skyttner (2005) o contato com o ambiente só ocorre para o recebimento de energia.
- Os próximos três níveis são os relacionados à biologia, botânica e zoologia, considerados sistemas abertos, isto é, sistemas que estão em contato com

um ambiente com o qual trocam matéria, energia e informação, sendo capaz de se adaptar ao ambiente e manter um estado estável (SKYTTNER, 2005; HITCHINS, 2007).

- Os níveis seguintes, de sete a nove, também são sistemas abertos, o homem e os sistemas sociais, as artes, humanidades e religião.

Tabela 1 - Classificação de sistemas

Nível	Características	Exemplo
Estruturas Estáticas	Estático	Pontes, moléculas, cristais, estruturas biológicas desde o nível microscópico até o macroscópico.
Trabalho como relógio	Movimentos pré-determinados	Máquinas convencionais em geral, sistema solar
Mecanismos de Controle	Controle em malha fechada	Termostato, servomecanismo, mecanismo homeostático em organismos
Abertos	Automanutenção	Células biológicas e organismos em geral.
Organismos Inferiores	Crescimento, reprodução	Plantas
Animais	Cérebro, aprendizado	Pássaros
Homem	Conhecimento, simbolismo	Humanos
Social	Comunicação, valores	Família
Transcendental	Desconhecido	Deus

Fonte: Adaptada de Hitchins (2007, p. 10), Bertalanffy (2006, p. 28,29) e Boulding (1956).

Adams e Mun (2005) argumentam que o interesse em uma ciência universal procurou juntar diversas disciplinas para obter uma lei das leis, uma lei aplicável a todas as demais, fato que levou a gênese da Teoria Geral dos Sistemas.

Posteriormente, a Sociedade para Pesquisa em Sistemas Gerais se tornou uma organização que deu suporte ao movimento sistêmico em geral. Em 1988 o seu nome foi alterado para Sociedade Internacional para as Ciências em Sistemas (ISSS - International Society for the Systems Sciences⁶⁶).

⁶⁶ A página web da ISSS está disponível no endereço: <<http://iss.org/world/index.php>>. Acesso em: 12 jun. 2015.

Classificação de Sistemas - Hierarquia

Checkland (1981) comenta que o argumento de Aristóteles de que “o todo é maior que a soma das partes” foi considerado uma doutrina desnecessária na Revolução Científica do século XVII. A física Newtoniana fornecia uma visão mecânica do universo que sobreviveu a testes severos, e a visão teleológica de Aristóteles, na qual os objetos no mundo seguem sua natureza interna ou propósito parecia uma especulação metafísica desnecessária. Contudo, a biologia moderna recolocou o propósito como um conceito respeitável e identificou a existência de níveis de organização para sistemas, utilizando a uma analogia com o conceito de hierarquia. A teoria de hierarquia é construída sobre a verificação empírica de que há propriedades emergentes associadas a um conjunto de componentes que se inter-relacionam, essa teoria é a base da classificação de sistemas apresentada na tabela 1. Checkland (1981) destaca que não há questionamentos sobre a ordem proposta por Boulding (1956) para a classificação de sistemas (tabela 1), ela é convincente pois intuitivamente se reconhece que propriedades emergentes sinalizam um novo nível.

REFERÊNCIAS

- ADAMS, K. MacG.; MUN, J. H. The application of systems thinking and systems theory to systems engineering. In: **Proceedings of the 26th National ASEM Conference: Organizational Transformation: Opportunities and Challenges**. Oct. 2005, p. 493-500. American Society for Engineering Management, Rolla, MO, USA, Disponível em: <https://www.researchgate.net/publication/273261062_The_Application_of_Systems_Thinking_and_Systems_Theory_to_Systems_Engineering>. Acesso em: 10 jun. 2015.
- BERTALANFFY, L. von, **General System Theory: Foundations, Development, Applications**. New York: George Braziller, 2006, 295 p. ISBN: 978-0807604533.
- BOULDING, K. E. General Systems Theory - The skeleton of science. **Management Science**, v. 2, n. 3, Abr. 1956, p. 197–208. Disponível em: <http://iscepublishing.com/ECO/ECO_other/Issue_6_1-2_18_CP.pdf>. Acesso em: 01 jun. 2015.
- CHECKLAND, P. **Systems Thinking, Systems Practice**. London: John Wiley & Sons, 1981. 344p. ISBN: 0471279110.
- HITCHINS, D. K. **Systems Engineering: a 21st century systems methodology**. Chichester: John Wiley & Sons, 2007. 528p. ISBN 9780470058565.
- KLIR, G. J. **Facts of systems science**. New York: Springer, 2^a ed., 2001. 748 p. ISBN: 978-0306466236.

SKYTTNER, L. **General systems theory:** problems, perspectives, practice. New Jersey: World Scientific, 2^a ed., 2005, 524 p. ISBN: 9812564675.

APÊNDICE B – Breve Visão Histórica

O avanço que o homem vem conseguindo na construção de ferramentas, métodos e artefatos vêm promovendo uma revolução cada vez mais rápida, passando pela Idade da Pedra, Idade do Bronze, Idade do Ferro e, segundo Ackoff (1974), a Idade das Máquinas e a Idade dos Sistemas.

A Idade das Máquinas é marcada pela Revolução Industrial, com a substituição do homem por máquinas construídas pelo homem, como força de trabalho (ACKOFF, 1974). Durante essa Idade, os artefatos construídos pelo homem possuíam um número restrito de variáveis, os problemas podiam ser reduzidos a dimensões quantitativas, levando o homem a pensar que tinha total entendimento do dispositivo criado (HUGHES, 2005).

A Revolução Industrial é precedida pelo Renascimento, quando a crença no homem e em seu potencial começa a se sobrepor às crenças da Idade Média. Pensadores renascentistas como Francis Bacon, René Descartes e Issac Newton resgatam o conceito de átomo do filósofo grego Demócrito. Tudo é feito de partes indivisíveis.

No Renascimento se acreditava que era possível ter-se todo o entendimento do funcionamento do mundo (ACKOFF, 1981) e a Revolução Industrial, que emerge do Renascimento, deve muito a um dos principais pensadores do Renascimento: René Descartes.

Descartes forneceu o ferramental necessário ao entendimento do mundo, ao elaborar suas regras para busca pelo verdadeiro método para se alcançar o conhecimento de todas as coisas. O conceito que Descartes expôs foi dividir o problema, ou dificuldade, encontrada em partes menores para facilitar o entendimento de cada parte envolvida no problema. Se for possível entender e explicar as partes menores, então o todo poderá ser explicado juntando-se novamente as partes. Essa ideia e prática, chamada Reducionismo, está presente em nossas vidas até os dias de hoje. Sempre que listamos, priorizamos, desmontamos, desagregamos e decompomos estamos prestando uma homenagem a René Descartes e seu Reducionismo Cartesiano (HITCHINS, 2007). Esse método de análise permitiu ter-se um instrumento pelo qual a ciência pôde procurar o entendimento do mundo procurando por seus elementos menores, suas partes indivisíveis (SYDENHAM, 2003).

Seguindo a visão reducionista, todas as áreas da ciência procuraram pelo seu elemento indivisível. Explicações sobre o comportamento e propriedades do todo⁶⁷ eram obtidas através de explicações sobre o comportamento e propriedades das partes desse todo. Quando o todo a ser explicado não podia ser dividido em partes independentes, a relação entre as partes tinha que ser entendida, para que se entendesse o todo. Acreditava-se que toda a interação entre objetos, eventos e suas propriedades podiam ser reduzidas a uma única e simples relação fundamental: relação de Causa e Efeito. Tudo devia ser considerado como efeito de alguma Causa, caso contrário ele não poderia ser entendido. Um Efeito não ocorre a menos que sua Causa tenha ocorrido, e o Efeito deve existir se a sua Causa existiu. Essa doutrina é chamada Determinismo.

O Reduccionismo e o Determinismo permitiram acreditar que todas as coisas e todos os eventos podiam ser reduzidos a partículas de matéria e seu movimento. Isso tanto para coisas que possuíam movimento como para os que não possuíam, não havia diferenças essenciais. Assim as ciências físicas eram tidas como o suficiente para explicar a vida. Essa visão do mundo, chamada Mecanicista, não necessita de conceitos teológicos para explicar fenômenos naturais e, em seu limite, traz o conceito do universo como uma máquina. Nesse ponto, Descartes depende de seu Relojoeiro para fazer sua máquina biológica, visto que as máquinas não se autocriam na natureza (ACKOFF, 1974, 1997; HITCHINS, 2007; HUGHES, 2005).

Ackoff (1997) argumenta que a questão do universo visto como um relógio, cujo comportamento era completamente determinado por sua própria estrutura e leis causais que se aplicavam a ele, trouxe uma grande questão de que um Deus era necessário. Nesse sentido, tem-se a visão de que o mundo foi concebido por Deus como uma máquina para servir a Seus propósitos, uma máquina para fazer o Seu trabalho. Assim, nada mais natural para o homem do que tentar produzir máquinas para servir a seus propósitos, realizar o seu trabalho.

No início do século XX, alguns cientistas começaram a notar que nem tudo era respondido pela abordagem reducionista e pelo determinismo. Em 1927, Werner Heisenberg desenvolveu o Princípio da Incerteza, que abalou a estrutura do determinismo e, conseqüentemente, a Idade das Máquinas. A crença de que era

⁶⁷ O autor deste trabalho utiliza neste texto o substantivo todo para traduzir do inglês o termo whole. Todo significando o conjunto completo.

possível ter um entendimento de todo o mundo se abalou. Algumas coisas pareciam funcionar como um todo, certamente elas podiam ser divididas em partes menores, mas essas partes não explicavam o funcionamento do todo. As propriedades essenciais das coisas não podiam ser inferidas pelo conhecimento das propriedades das partes, ou das interações entre as partes, por exemplo: a personalidade ou a inteligência de um ser humano (ACKOFF, 1981). Entende-se que há coisas que podem ser divididas em partes, mas que perdem suas funções e suas propriedades essenciais ao se fazer isso, Ludwig Von Bertalanffy chama essas “coisas” de sistemas.

Bertalanffy (2006) argumenta que o problema com sistemas é essencialmente um problema de limitação do reducionismo e determinismo utilizados pela ciência, e que essa limitação costuma ser expressa por uma colocação quase metafísica, como evolução emergente ou “o todo é maior que a soma das partes”.

O todo ser maior que a soma das partes significa que a interação das partes dá ao sistema propriedades únicas. Essas propriedades, chamadas de propriedades emergentes, se referem a propriedades do sistema como um todo, que não são atribuídas exclusivamente a uma das partes. Por exemplo, o odor da amônia quando dois gases inodoros e incolores - nitrogênio e hidrogênio - são combinados. São propriedades que se originam da interação entre as partes.

Todo progresso da ciência durante a Idade das Máquinas mostra que a visão mecanicista é de grande sucesso em inúmeras aplicações. Bertalanffy (2006) argumenta que esse sucesso depende de duas condições:

- A primeira é que a interação entre as partes não exista ou possa ser fraca o suficiente para ser negligenciada para certos propósitos. Apenas sob essa condição, as partes podem estudadas verdadeiramente, tanto logicamente como matematicamente, e então serem colocadas juntas.
- A segunda condição é que as relações que descrevem os comportamentos das partes sejam lineares. Apenas se essas relações forem lineares é que há a condição aditiva, isto é, a equação que descreve o comportamento do todo é da mesma forma que as equações que descrevem o comportamento de cada uma das partes.

Essas duas condições não estão presentes no que Bertalanffy chama de sistema, que é constituído por partes em interação e pode ser descrito por um conjunto de equações diferenciais, em geral, não são lineares.

Ackoff (1974, 1981, 1997) argumenta que a percepção do fato de que sistema é um todo que não pode ser entendido pelo reducionismo é a principal fonte da revolução intelectual que levou à mudança de Idades: A passagem da Idade das Máquinas para outra, a Idade dos Sistemas. Ackoff prossegue argumentando que se torna visível a necessidade de método para o entendimento do comportamento e propriedades dos sistemas; esse método, a síntese, ou agrupamento de coisas, é a chave para o pensamento da Idade dos Sistemas, assim como a análise reducionista é chave para o pensamento na Idade das Máquinas, sendo que a síntese deve ser utilizada em conjunto com a análise reducionista, não substituí-la.

A Abordagem Sistêmica, ou Pensamento Sistêmico, é o uso do modo de pensar sintético nos problemas envolvendo sistemas, ele traz um conjunto de métodos para serem utilizados para entender as relações, a simultaneidade, o transiente e as mudanças dessas mesmas relações, buscando o tratamento de problemas para os quais apenas o uso do Pensamento Analítico não traz respostas satisfatórias. Charles François, editor da International Encyclopedia of Cybernetics and Systemics, em matéria para o Seminário Primer⁶⁸ da Sociedade Internacional para Ciências em Sistemas (ISSS - International Society for the Systems Sciences), coloca como exemplos desses problemas a clínica na medicina, as mudanças econômicas no mundo, a saúde mental individual e coletiva e os problemas ecológicos provocados pelo homem.

A DESUMANIZAÇÃO DA IDADE DAS MÁQUINAS

Pouco antes da Segunda Guerra Mundial, o homem da Idade das Máquinas e as máquinas por ele criadas, estavam organizados num processo apoteótico de produção em massa e linha de montagem. A mecanização, a substituição do homem pela máquina como fonte de trabalho físico, afetou a natureza e as tarefas deixadas pelos criadores das máquinas para o homem desempenhar. Ao homem restaram as tarefas simples e repetitivas, partes pequenas do processo de produção e quanto mais máquina eram usadas para substituir o homem, mas ele se comportava como

⁶⁸ Artigo disponível em: < <http://www.issss.org/primer/francois.htm> >. Acesso em: 27 jun. 2015.

máquina. Fato representado por Charles Chaplin em seu filme Tempos Modernos de 1936.

A mecanização levou a desumanização do trabalho humano. Ackoff (1997) argumenta que esta é a grande ironia da Revolução Industrial.

SISTEMA: O OBJETIVO FINAL

Durante a Segunda Guerra Mundial, as Forças Armadas Americanas recrutaram inúmeros cientistas e engenheiros para tratar os problemas relacionados com os esforços de guerra. Entre esses problemas estavam:

- Problemas logísticos, devido às distâncias das tropas e as diversas frentes de batalhas.
- Problemas de comunicação entre as tropas, dada a necessidade de codificações das mensagens e distância; e,
- Bombardeios estratégicos.

Adams e Mun (2005) argumentam que os esforços feitos para tratar esses problemas trouxeram uma contribuição significativa tanto para a filosofia como a para as técnicas do que foi chamado de Pesquisa Operacional. Engrstrom (1957) informa que a experiência vivida pelos engenheiros e cientistas durante a Segunda Guerra Mundial trouxe um grande ímpeto a aplicação de uma abordagem sistêmica. Engrstrom prossegue comentando que a prática, durante as circunstâncias e pressões da Guerra, era pensar além dos componentes individuais e seus usos, considerando o ambiente onde esses componentes iriam operar. Era necessário olhar o objetivo final: o Sistema.

REFERÊNCIAS

ACKOFF, R. L. **Redesigning the Future**: a systems approach to societal problems. New York: John Wiley and Sons, 1974, 260p. ISBN: 9780471002963.

ACKOFF, R. L. **Creating the corporate future**: plan or be planned for. Nova York: John Wiley and Sons, 1981, 297p. ISBN: 978-0471090090.

ACKOFF, R. L. Systems, Messes and Interactive Planning, In: TRIST, E.; EMERY, F.; MURRAY, H. **The Social Engagement of Social Science**: a Tavistock anthology: the socio-ecological perspective. Philadelphia: University of Pennsylvania Press, 1997, v. 3, p. 417-438. ISBN: 9780812281941. Disponível em:

<<http://www.moderntimesworkplace.com/archives/ericssess/sessvol3/sessvol3.html>>. Acesso em: 20 jun. 2015.

ADAMS, K. MacG.; MUN, J. H. The application of systems thinking and systems theory to systems engineering, In: NATIONAL ASEM CONFERENCE, 26th, 2005, Rolla, MO, USA. **Proceedings...** Rolla, MO: American Society for Engineering Management, p. 493-500. Disponível em: < <http://www.asem.org/conferences/2005conferenceproceedings/91.pdf> >. Acesso em: 10 abr. 2015.

BERTALANFFY, L. von, **General System Theory**: foundations, development, applications. Revised ed. New York: George Braziller, 2006, 295 p., ISBN: 978-807604533.

ENGSTROM, E.W. Systems engineering: a growing concept. **Electrical Engineering**, v. 76, n.2, p. 113-116, fev. 1957. DOI: 10.1109/EE.1957.6442968.

HITCHINS, D. K. **Systems Engineering**: a 21st century systems methodology. Chichester: John Wiley & Sons, 2007. 528p. ISBN: 978-0470058565.

HUGHES, T. P. **Human-Built World**: how to think about technology and culture. Chicago: The University of Chicago Press, 2005. 240p. ISBN 0226359344.

SYDENHAM, P. H. **System Approach to Engineering Design**. Norwood: Artech House, 2003, 333p. ISBN: 1580534791.

APÊNDICE C – Abordagem evolucionária

A abordagem evolucionária (figura 1) trata da interação entre **realidade** e **pensamento** e da interação entre **problema** e o que Soares (1986) denominou de **solução**. Soares define **solução** como superação de impossibilidades ou melhorias na realidade existente através de uma ação, considerando **solução** como também indicativa de melhorias, isto é, de uma resposta que satisfaz, mas não soluciona o problema. O autor deste trabalho prefere o uso do termo **resposta**, pois de acordo com Ackoff (1981) um problema pode ser **resolvido**, **solucionado** ou **dissolvido**. Das interações expostas acima há quatro ações que geram um ciclo a ser percorrido para resolver (nem sempre solucionar) um problema. Essas ações são as seguintes:

- **Compreensão**⁶⁹: Quando o engenheiro constrói um entendimento, uma representação abstrata para um problema real.
- **Concepção**: Quando o engenheiro cria uma resposta ao problema que satisfaz o problema no plano do pensamento.
- **Realização**: Construção da resposta ao problema no plano da realidade.
- **Utilização**: Estabelecimento da resposta ao problema no ambiente do problema.

Estabelecer a resposta ao problema pode trazer impacto à realidade, criando cenários não determinados anteriormente, o que dá origem a novas demandas levando a uma redefinição do problema. A sequência de tratamentos aos problemas origina uma espiral evolucionária (figura 1, baseada em Soares, 1986) que mantém o controle dos passos na identificação dos requisitos e das dimensões humanas e sociais do tratamento de um problema, sendo que esse controle é fundamental na engenharia de requisitos (SOARES, 1986; KOTONYA; SOMMERVILLE, 1998; INCOSE, 2006)

⁶⁹ Soares (1986) chama essa ação de apreensão, o autor deste trabalho prefere chamar de compreensão (KIERAN, 2002).

Figura 1 - Espiral Evolucionária

Fonte: Adaptada de Soares (1986).

REFERÊNCIAS

ACKOFF, R. L. **Creating the corporate future: plan or be planned for**. Nova York: John Wiley and Sons, 1981, 297p. ISBN: 978-0471090090.

INCOSE. Systems Engineering Handbook Development Team of the International Council on Systems Engineering (INCOSE). **SYSTEMS ENGINEERING HANDBOOK**, v. 3. INCOSE-TP-2003-002-03, jun. 2006.

KIERAN, E. **A Mente Educada**. São Paulo: Bertrand Brasil, 1ª edição, 406 p., 2002. ISBN: 8528609170.

KOTONYA, G.; SOMMERVILLE, I. **Requirements Engineering: Processes and Techniques**. London: J. Wiley, 1998, 294p. ISBN: 978-0471972082.

SOARES, J. O. P. **Especificação de Métodos de Desenvolvimento de Sistemas - Aplicação a Sistemas de Tempo Real**. São Paulo, 1986. Dissertação (Mestrado) - Escola Politécnica, Universidade de São Paulo.

APÊNDICE D – Métodos consensuais

Gause e Weinberg (1989) argumentam que não se pode descobrir o que as pessoas necessitam sem que se compreendam quais são os desejos delas relacionados a essa necessidade. Uma forma de construir essa compreensão é através de entrevistas e de dinâmicas com esse grupo para que seja possível construir um consenso entre todos sobre a necessidade e os desejos. A seguir, são apresentados cinco métodos que possibilitam obter o consenso entre um grupo de pessoas sobre alguma questão ou problema.

BRAINSTORMING

Brainstorming é uma técnica criativa na qual um grupo de pessoas é encorajado por um moderador para expressar suas ideias em relação a um problema ou necessidade. Isso é feito solicitando ideias sobre um determinado tópico ou sobre uma determinada questão. O resultado de uma sessão de Brainstorming pode ser uma lista de ideias, expectativas, requisitos, definições e até a identificação de um problema em relação a própria definição desses itens (TAGUE, 2005; HITCHINS, 2007).

TÉCNICA DE GRUPO NOMINAL (TGN)

A Técnica de Grupo Nominal (em inglês: Nominal Group Technique - NGT) possui semelhanças com o Brainstorming. Um grupo de pessoas se reúne em uma sala, um moderador apresenta um assunto e conduz uma discussão:

- Antes de iniciar a discussão propriamente dita, o moderador pede aos participantes para que escrevam em uma folha de papel suas ideias sobre o assunto.
- Após um tempo para que todos escrevam suas ideias, o moderador convida cada um dos participantes a ler a primeira ideia que escreveu.
- A cada leitura o moderador escreve a ideia em uma grande folha de papel (flip chart).
- Uma nova rodada é realizada com cada participante apresentando sua segunda ideia. O ciclo prossegue até que o moderador tenha anotado todas as ideias.

- O moderador distribui, a seguir, as folhas do flip chart pela sala. A intenção dessa ação é distanciar cada ideia da pessoa que a teve.
- O grupo discute as ideias, iniciando pela combinação daquelas que podem ser consideradas como duplicadas, alcançando um entendimento sobre o que cada ideia realmente significa.
- Cada participante é convidado a dar uma nota entre um e dez para cada uma das ideias: Dez para a que mais gostou, nove para a seguinte e assim por diante. O moderador agrega todas as notas. Algumas ideias poderão ficar sem nenhuma nota, e serão descartadas.
- As ideias são ordenadas e delas selecionadas as melhores, normalmente as são consideradas as vinte melhores pontuadas.

Essas ideias constituem as respostas do grupo para o assunto colocado e foram produzidas pelo grupo como um todo, daí o nome “grupo nominal” (TAGUE, 2005; HITCHINS, 2007).

ESCRITA DE IDÉIAS

A Escrita de Ideias é um aprofundamento da TGN. Depois da apresentação de um tema, esta técnica tem a seguinte estrutura:

- Os participantes são convidados a escrever suas ideias, sugestões, etc., em uma folha de papel.
- Depois de apenas dois ou três minutos, o moderador pede para os participantes passarem a folha para uma pessoa ao seu lado. O ideal é circular em um sentido excluindo algumas pessoas: passar a folha para duas pessoas à esquerda, por exemplo.
- Quem recebe a folha pode ver as ideias já escritas, o que pode levá-lo a um novo conjunto de ideias.
- Depois de um curto período de tempo, o moderador pede para a folha circular novamente, dessa vez um número diferente de pessoas. O processo se repete por uns 30 minutos, ou até que o moderador perceba que a maioria das pessoas não possui mais ideias.

Há dois propósitos nesta estratégia:

- 1) Estimular o surgimento de ideias dentro do grupo, e
- 2) Esconder a origem de uma ideia particular.

Essa estratégia é para que cada ideia possa ser tratada objetivamente durante a discussão subsequente, sem preconceitos por ter sido originada por alguém com pouca experiência no tema ou por alguém com cargo de chefia. Para atingir esses propósitos é fundamental:

- Que todos utilizem o mesmo padrão de papel e caneta.
- Que os participantes escrevam suas ideias com letras maiúsculas.
- Que o moderador varie os sentidos e o intervalo de pessoas a pular cada vez que solicitar a circulação da folha com as ideias.

As listas de ideias são trabalhadas posteriormente através de Brainstorming ou TGN para a obtenção de um plano de ação

ANÁLISE E ESTRUTURAÇÃO DE MODELOS

A Análise e Estruturação de Modelos (em inglês: Interpretative Structural Modeling - ISM) auxilia a análise de questões complexas, organizando uma visão coerente delas (WRIGHT, 1995). A técnica foi proposta pela primeira vez em 1973 por John Warfield para realizar análises de sistemas sociais, auxiliando a formulação de planos, estratégias e tomadas de decisão. (WARFIELD, 1976).

O objetivo do ISM é obter um grafo no qual os nós representam entidades sobre as quais se deseja estabelecer relações, por exemplo, as entidades podem ser objetivos e as relações indicarem qual objetivo ajuda a alcançar o outro. Antes do início de uma sessão de ISM, é necessário que se identifique o conjunto de entidades entre as quais irão se estabelecer relações. Uma forma de identifica-las é através de Brainstorming ou de outro método consensual descrito anteriormente. Todo o processo pode vir a consumir muito tempo, especialmente quando a muita divergência dentro do grupo sobre os relacionamentos entre as entidades. Contudo, Hitchins (2007) argumenta

que esse tempo é importante, pois permite aos participantes oportunidades para o entendimento, reconhecendo argumentos uns dos outros, buscando um consenso.

SISTEMA DE MÉTODOS SOFT

Sistema de Métodos Soft (em inglês: Soft Systems Methodology - SSM) é um método proposto por Checkland (1981) para a articulação de problemas pouco estruturados, isto é, problemas sobre os quais as pessoas não possuem um consenso, são problemas mal definidos. Sua origem está em um programa de pesquisa do Grupo de Engenharia de Sistemas da Universidade de Lancaster, Inglaterra, que ficava frustrado ao tentar aplicar métodos tradicionais da engenharia de sistemas aos problemas de administração de empresas. Nem sempre eles conseguiam determinar os propósitos do sistema ou da organização que eles estavam tentando auxiliar. A SSM representa um desenvolvimento do pensamento sistêmico para abordar os problemas que esses pesquisadores encontraram.

A SSM promove um entendimento do problema, ou de uma visão do problema, que surge da interação entre as pessoas envolvidas na situação estudada, promovendo uma acomodação das múltiplas visões do problema e dos múltiplos interesses em seu tratamento. Como uma metodologia desenvolvida a partir do pensamento sistêmico soft, o SSM não produz uma resposta final a questionamentos, ela busca entender a situação do problema e encontrar a melhor resposta possível (CHECKLAND, 1985).

O SSM pode ser representado por um modelo de sete estágios como o que pode ser visto na figura 1. A representação foi montada a partir de Checkland (1981, fig. 6, pg. 163); Bellini; Rech e Borenstein (2004, fig. 1, pg. 5) e Hitchins (2007, fig. 7.1, pg. 194) e destaca as características de apoio à aprendizagem e à reformulação de hipóteses da metodologia.

Hitchins (2007) argumenta que o ponto chave da abordagem é que a análise aborda a situação do problema sobre duas perspectivas: Parte do método demanda o relato do que realmente está acontecendo na situação que está sendo analisada (o mundo real), outra parte utiliza de lógica e pensamento sistêmico (buscando um modelo no mundo ideal).

Figura 1– Estágios do Sistema de Métodos Soft

Fonte: Adaptada Checkland (1981, fig. 6, pg. 163); Bellini; Rech e Borenstein (2004, fig. 1, pg. 5) e Hitchins (2007, fig. 7.1, pg. 194).

Os sete estágios que compõem o SSM são os seguintes (CHECKLAND, 1981; BELLINI; RECH; BORENSTEIN, 2004; HITCHINS, 2007):

- **Estágios 1 e 2:** Explorar a Situação Problemática Mal Definida (não estruturada) e Expressá-la.

O objetivo dos dois primeiros estágios é elaborar uma figura que represente a situação problemática, segundo a nomenclatura própria do SSM, uma: figura rica (rich picture). Essa figura é uma representação gráfica livre do entendimento sobre o problema. O uso de uma representação gráfica encoraja a formação de ideias e facilita à observação de relações e conflitos, Checkland e Scholes (1990) apresentam a figura rica como a ferramenta mais indicada para explorar os relacionamentos, em contraste

com a linguagem linear. Hitchins (2007) afirma que a figura rica possibilita identificar possíveis propósitos para o sistema, pois são identificados vários pontos de vista durante a sua construção.

Os aspectos principais a serem considerados na construção de uma figura rica são (CHECKLAND, 1981):

- 1) **A estrutura da situação:** que ajuda a identificar hierarquias formais e informais e sistemas de comunicação.
- 2) **O processo da situação:** que ajuda a compreender como as coisas funcionam e quem faz o quê.
- 3) **A relação entre estrutura e processo:** que ajuda a identificar a cultura existente.

- **Estágio 3: Definições-Raízes de Sistemas Relevantes.**

Ao iniciar este estágio ainda não é possível saber qual sistema necessita ser desenvolvido ou redefinido, contudo é possível identificar nomes de sistemas conceituais que parecem relevantes ao problema. A identificação desses sistemas relevantes deve ser realizada de forma cuidadosa e explícita. Esses sistemas representam uma perspectiva particular sobre a situação problemática, mas não significa que eles serão necessariamente projetados e implementados no mundo real (CHECKLAND, 1981).

Neste estágio, uma definição sucinta, ou **definição-raiz** (root definition) segundo a nomenclatura própria da SSM, é desenvolvida para cada sistema relevante, descrevendo seis aspectos do sistema (CHECKLAND, 1981; ANDRADE et al., 2006; HITCHINS, 2008):

C (Cliente do sistema): vítima ou beneficiário das transformações do sistema, se o sistema existisse;

A (Atores dentro do sistema): aqueles que seriam os protagonistas da transformação, que executariam as atividades do sistema;

T (Transformação): o que o sistema faria; que entradas seriam transformadas em que saídas pelo sistema;

W (Weltanschauung ou **W**ord view (visão de mundo)): que visão de mundo, no contexto do sistema, torna a transformação que seria promovida pelo sistema significativa;

O (Owners / proprietários): quem teria autoridade para parar ou modificar a transformação que seria promovida pelo sistema;

E (Environment constrains / restrições ambientais): restrições externas que o sistema teria como certas.

Essas **definições-raízes**, identificadas pelo mnemônico **CATWOE**, devem ser uma descrição concisa de um sistema de atividades humanas que captura uma visão particular sobre ele (CHECKLAND, 1981). Andrade et al. (2006) argumenta que cada uma das **definições-raízes** esclarece a respectiva visão do sistema e diferentes visões são, de fato, a maior causa de complexidade.

- **Estágio 4:** Modelos Conceituais.

Neste estágio são elaborados os modelos conceituais dos sistemas nomeados nas *definições-raízes*. São construídos de modo a incluir o mínimo necessário de atividades para as suas necessidades, são modelos e não projetos. A elaboração pode ter qualquer nível de detalhe e pode ser feita de uma única vez ou através de interações de modo que se tenha uma hierarquia de modelos. Em ambos os casos, a elaboração deve ser o resultado de pensamento sistêmico, não uma referência explícita a organizações existentes ou processos que expõem apenas atividades que são logicamente necessárias (ANDRADE et al., 2006; HITCHINS, 2007).

- **Estágio 5:** Modelos Conceituais e Situação Problemática Expressada.

Neste estágio, comparam-se os modelos conceituais com as percepções da situação real. Deve-se considerar no momento da comparação as ações e mudanças necessárias nos modelos para a transformação da situação problemática. Essa análise deve atender a dois critérios: ser sistematicamente desejável por todos e culturalmente viável dentro do contexto do sistema (CHECKLAND, 1981; ANDRADE et al. 2006).

- **Estágios 6 e 7:** Mudanças Possíveis e Desejadas e Ações para Transformação

O estágio seis desenvolve um plano de ação para as mudanças que são viáveis e desejadas. Viáveis porque podem ser iniciadas e conduzidas na cultura existente, e desejadas porque elas podem trazer mudanças benéficas à situação problemática. No estágio sete o plano de ação é posto em prática (HITCHINS, 2007).

Hitchins (2007) argumenta que é improvável que um único ciclo da SSM solucione o problema. Esse ciclo irá alterar a situação que causou o problema, deixando-o mais exposto, isto é, apenas cria uma nova situação que pode ser beneficiada por uma nova abordagem da SSM. A esse respeito, Hitchins prossegue argumentando que um ciclo da SSM pode resolver o problema, sendo necessários vários ciclos para solucionar o problema.

Kotonya e Sommerville (1998) afirmam que a essência da SSM está no fato de ela reconhecer que os sistemas estão incorporados no contexto humano e organizacional, e que, em um sentido mais amplo, ela está preocupada com os sistemas sociotécnicos.

REFERÊNCIAS

ANDRADE, A. L. et al. **Pensamento Sistêmico:** caderno de campo. Porto Alegre: Bookman, 2006, 488 p. ISBN: 8536307005.

BELLINI, C. G. P.; RECH, I.; BORENSTEIN, D. Soft Systems Methodology: Uma aplicação no "pão dos pobres" de Porto Alegre. **RAE Eletrônica**, São Paulo, v. 3, n. 1, jun. 2004. DOI: 10.1590/S1676-56482004000100007

CHECKLAND, P. **Systems Thinking, Systems Practice**. London: John Wiley & Sons, 1981. 344p. ISBN: 0471279110.

CHECKLAND, P. Soft Systems Methodology: a 30-year retrospective. In CHECKLAND, P.; SCHOLLES, J. **Soft Systems Methodology in Action**. Chichester: John Wiley & Sons, 1990. 436p. ISBN: 0471927686.

CHECKLAND, P. From optimizing to learning: a development of systems thinking for the 1990s. **Journal of the Operational Research Society**, v. 36, n. 9, p. 757-767, 1985. Disponível em: <<http://www.jstor.org/stable/2582164>>. Acesso em: 12 jun. 2015.

CHECKLAND, P.; SCHOLLES, J. **Soft Systems Methodology in Action**. Chichester: John Wiley & Sons, 1990. 436p. ISBN: 0471927686.

GAUSE, D.C.; WEINBERG, G. M. **Exploring Requirements**: quality before design. New York: Dorset House, 1989, 300 p., ISBN: 0932633137.

HITCHINS, D. K. **Systems Engineering**: a 21st century systems methodology. Chichester: John Wiley & Sons, 2007. 528p. ISBN: 978-0470058565.

KOTONYA, G.; SOMMERVILLE, I. **Requirements Engineering**: processes and techniques. London: J. Wiley, 1998, 294p. ISBN: 978-0471972082.

TAGUE, N. R. **The Quality Toolbox**. 2nd ed. Milwaukee: ASQ Quality, 2005, 559p. ISBN: 0873896394.

WARFIELD, J. N. **Societal systems**: planning, policy, and complexity. New York: John Wiley & Sons, 1976, 490p. ISBN: 978-0471015695.

WRIGHT, J. T. C. Análise e estruturação de modelos baseada em inferência lógica: objetivos para o Porto de Santos. **Revista de Administração da Universidade de São Paulo**, São Paulo, v. 30, n. 1, jan./mar. 1995. Disponível em: < <http://www.rausp.usp.br/download.asp?file=3001090.pdf> >. Acesso em 12 jan. 2015.