VALTER FERNANDES AVELINO

**MERUSA: METODOLOGIA DE
ESPECIFICAÇÃO DE REQUISITOS DE
USABILIDADE E SEGURANÇA ORIENTADA
PARA ARQUITETURA**

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

São Paulo
2005

VALTER FERNANDES AVELINO

**MERUSA: METODOLOGIA DE
ESPECIFICAÇÃO DE REQUISITOS DE
USABILIDADE E SEGURANÇA ORIENTADA
PARA ARQUITETURA**

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

Área de Concentração:
Engenharia Elétrica

Orientadora:
Prof.^a Dra. Selma Shin Shimizu
Melnikoff

São Paulo

2005

FICHA CATALOGRÁFICA

Avelino, Valter Fernandes

MERUSA: metodologia de especificação de requisitos de usabilidade e segurança orientada para arquitetura / V.F. Avelino. -- São Paulo, 2005.

262 p.

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

1.Engenharia de programação 2.Descrição de sistemas (Arquitetura de computadores) 3.Segurança em computador 4.Interface homem-computador I.Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia de Computação e Sistemas Digitais II.t.

*À minha amada esposa Maria Aparecida
e meus adorados filhos André e Renato.*

AGRADECIMENTOS

A Prof.^a Dra. Selma Shin Shimizu Melnikoff pela grande amizade, orientação e permanente incentivo, que foram essenciais não só para a o desenvolvimento deste trabalho como também para o desenvolvimento da minha vida profissional.

A Prof.^a Dra. Lúcia Vilela Filgueiras pela amizade, apoio e constante incentivo, que muito contribuíram para a realização deste trabalho.

Aos colegas do Departamento de Engenharia de Computação e Sistemas Digitais da Escola Politécnica da Universidade de São Paulo pelo apoio e incentivo.

À Companhia de Transmissão de Energia Elétrica Paulista – CTEEP pela permissão da utilização de informações operativas de suas subestações, pelo apoio técnico fornecido, bem como pela colaboração na discussão dos resultados obtidos.

A minha esposa pela valiosa colaboração, contínuo estímulo e grande paciência.

Aos meus filhos pela compreensão e estímulo.

A todas as pessoas que de alguma forma contribuíram para a obtenção deste importante objetivo na minha carreira acadêmica.

RESUMO

Esta tese objetiva propor uma metodologia de especificação de requisitos de usabilidade e segurança orientada para arquitetura (MERUSA) aplicável a sistemas digitais de automação distribuídos e abertos.

A estratégia utilizada para atingir esse objetivo consiste no desenvolvimento de uma metodologia focalizada nos processos de elicitação e análise dos requisitos de usabilidade e segurança, sincronizada com a evolução da arquitetura do sistema computacional.

O modelo de referência ODP (*Open Distributed Process*) é utilizado como uma referência básica desse processo, fornecendo conceitos, procedimentos e modelos adequados à especificação e ao desenvolvimento de sistemas distribuídos e abertos de automação (orientados a objeto ou não), permitindo obter uma infra-estrutura coerente e gerenciável, com exigentes requisitos de integração.

A eficácia da metodologia é verificada com a sua aplicação a um meta-modelo de especificação de projeto para sistemas digitais de controle distribuído de transmissão de energia para o sistema elétrico brasileiro.

ABSTRACT

This thesis purposes one new architecture oriented safety and usability requirements specification methodology (MERUSA) applied to open and distributed automation digital systems.

The strategy used to achieve this goal consists on the development of a methodology centered on safety and usability requirement elicitation and analysis processes, synchronized with computational system architectural evolution.

The RM-ODP (Reference Model of Open Distributed Process) is used as basic reference of this process. This pattern. supplies concepts, procedures and models appropriated to open and distributed system specification and development (object oriented or not), providing one coherent and manageable framework, with strong integration requirements.

The methodology effectiveness is verified with its application to a project specification meta-model for energy transmission distributed control digital systems of Brazilian electrical system.

SUMÁRIO

LISTA DE TABELAS

LISTA DE FIGURAS

GLOSSÁRIO DE ABREVIATURAS E SIGLAS

1	INTRODUÇÃO	1
1.1	MOTIVAÇÃO	1
1.2	ABRANGÊNCIA DO TRABALHO	2
1.3	JUSTIFICATIVA	4
1.4	METODOLOGIA DE TRABALHO	5
1.5	ESTRUTURA DA TESE	7
2	ENGENHARIA DE REQUISITOS.....	10
2.1	FUNDAMENTOS DA ENGENHARIA DE REQUISITOS	10
2.2	REQUISITOS FUNCIONAIS (RF)E NÃO FUNCIONAIS (RNF).....	16
2.3	ENGENHARIA DE REQUISITOS E O PROJETO DA ARQUITETURA	18
2.4	MÉTODO DE ANÁLISE DE COMPROMISSOS DE ARQUITETURA (ATAM)	23
2.5	TÉCNICAS PARA MODELAGEM DE REQUISITOS	31
2.5.1	<i>Linguagem de requisitos orientada a objetivos (GRL)</i>	32
2.5.2	<i>Mapas de casos de uso (UCM)</i>	34
2.5.3	<i>Integração entre GRL e UCM</i>	36
3	MODELO DE SISTEMAS ABERTOS COM ARQUITETURA DE OBJETOS	
	DISTRIBUÍDOS.....	40
3.1	PONTO DE VISTA DA EMPRESA	42
3.2	PONTO DE VISTA DA INFORMAÇÃO	45
3.3	PONTO DE VISTA DA COMPUTAÇÃO	46
3.4	PONTO DE VISTA DA ENGENHARIA	48
3.5	PONTO DE VISTA DA TECNOLOGIA.....	51
3.6	SERVIÇOS DE TRANSPARÊNCIA	52
3.7	FUNÇÕES ODP.....	54
3.7.1	<i>Funções de gerenciamento</i>	54
3.7.2	<i>Funções de coordenação</i>	55
3.7.3	<i>Funções de repositórios</i>	55
3.7.4	<i>Funções de segurança de acesso</i>	56
3.8	CONCEITOS BÁSICOS E NOTAÇÃO	57

3.9	NÍVEIS DE ABSTRAÇÃO DO MODELO DO NEGÓCIO.....	61
3.10	RESUMO DOS PONTOS DE VISTA ODP	63
4	TÉCNICAS DE AVALIAÇÃO DE DEPENDABILIDADE.....	65
4.1	DEFINIÇÕES	65
4.2	PROJETO DE SISTEMAS RELACIONADOS COM A SEGURANÇA	67
4.3	TÉCNICAS DE ANÁLISE DE REQUISITOS DE SEGURANÇA	71
4.4	SEGURANÇA X SEGURANÇA DE ACESSO	76
4.5	TÉCNICAS DE PROJETO RELACIONADOS COM SEGURANÇA	78
4.5.1	<i>Princípios de prevenção de falha.....</i>	<i>79</i>
4.5.2	<i>Princípios de tolerância a falha.....</i>	<i>82</i>
5	TÉCNICAS DE AVALIAÇÃO DE USABILIDADE.....	89
5.1	CONCEITOS DE ENGENHARIA DE USABILIDADE.....	90
5.2	MODELO DE CICLO DE VIDA DE USABILIDADE.....	95
5.2.1	<i>Ciclo de vida da engenharia de usabilidade.....</i>	<i>97</i>
5.2.2	<i>Métodos de avaliação de usabilidade.....</i>	<i>102</i>
5.3	AVALIAÇÃO DE ERROS HUMANOS	112
5.3.1	<i>Métodos de análise de erros humanos.....</i>	<i>114</i>
5.3.2	<i>Modelos de análise de risco.....</i>	<i>115</i>
5.3.3	<i>Conclusões sobre a avaliação de erros humanos.....</i>	<i>122</i>
6	MERUSA: METODOLOGIA DE ESPECIFICAÇÃO DE REQUISITOS DE USABILIDADE E SEGURANÇA ORIENTADA PARA ARQUITETURA.....	124
6.1	ADEQUAÇÃO DAS TÉCNICAS DE SEGURANÇA AO RM-ODP.....	125
6.2	METODOLOGIA DE ELICITAÇÃO E ANÁLISE DE REQUISITOS DE SEGURANÇA.....	137
6.3	ADEQUAÇÃO DAS TÉCNICAS DE USABILIDADE AO RM-ODP	157
6.4	METODOLOGIA DE ELICITAÇÃO E ANÁLISE DE REQUISITOS DE USABILIDADE	164
6.5	DESCRIÇÃO DA MERUSA	178
6.6	CRITÉRIOS DE AVALIAÇÃO DE SENSIBILIDADE E PRIORIZAÇÃO DE REQUISITOS	182
6.7	CRITÉRIOS DE SINCRONISMO.....	185
6.8	RESUMO DAS CARACTERÍSTICAS DA MERUSA.....	188
7	AVALIAÇÃO DA METODOLOGIA.....	191
7.1	META-ARQUITETURA PARA SISTEMA DE TRANSMISSÃO DO SIN.....	192
7.1.1	<i>Modelo de negócio do SIN.....</i>	<i>194</i>
7.1.2	<i>Meta-Modelo ODP para um Sistema de Transmissão de Energia</i>	<i>197</i>
7.2	APLICAÇÃO DA MERUSA	201
7.2.1	<i>Atividade 1.1 - Identificação do negócio e da organização.....</i>	<i>202</i>

7.2.2	<i>Atividade 1.2 - Definição dos requisitos do negócio</i>	204
7.2.3	<i>Atividade 2.1 - Identificação dos objetivos e cenários do sistema.....</i>	204
7.2.4	<i>Atividade 2.2 - Elicitação dos requisitos do domínio</i>	205
7.2.5	<i>Atividade 2.3 - Descrição da arquitetura</i>	206
7.2.6	<i>Atividade 2.4 - Construção de cenários.....</i>	207
7.2.7	<i>Atividade 2.5 - Construção dos modelos de arquiteturas</i>	209
7.2.8	<i>Atividade 2.6 - Avaliação da sensibilidade da arquitetura.....</i>	210
7.2.9	<i>Atividade 3.1 - Identificação dos perigos e objetivos de segurança.....</i>	213
7.2.10	<i>Atividade 3.2 - Elicitação dos requisitos globais e cenários de segurança.....</i>	213
7.2.11	<i>Atividade 3.3 - Elicitação dos requisitos específicos de segurança.....</i>	215
7.2.12	<i>Atividade 3.4 - Construção dos cenários de segurança.....</i>	215
7.2.13	<i>Atividade 3.5 - Análise de riscos.....</i>	216
7.2.14	<i>Atividade 3.6 - Análise de segurança (independente).....</i>	217
7.2.15	<i>Atividade 3.7 - Análise de segurança integrada</i>	217
7.2.16	<i>Atividade 4.1 - Identificação do perfil do usuário e análise de tarefas</i>	219
7.2.17	<i>Atividade 4.2 - Princípios de IHC e restrições ambientais.....</i>	220
7.2.18	<i>Atividade 4.3 - Elicitação dos requisitos de usabilidade</i>	220
7.2.19	<i>Atividade 4.4 - Elaboração do guia de estilo de IHC</i>	222
7.2.20	<i>Atividade 4.5 - Reengenharia do trabalho.....</i>	222
7.2.21	<i>Atividade 4.6 - Projeto da IHC</i>	223
7.2.22	<i>Atividade 4.7 - Prototipação da IHC.....</i>	223
7.2.23	<i>Atividade 4.8 - Avaliação de usabilidade (independente).....</i>	223
7.2.24	<i>Atividade 4.9 - Avaliação de usabilidade integrada.....</i>	223
7.2.25	<i>Atividade 5.1 - Análise dos requisitos integrados.....</i>	225
7.2.26	<i>Atividade 5.2 - Análise de compromissos</i>	226
7.3	RESUMO DOS RESULTADOS.....	238
8	CONCLUSÃO.....	241
8.1	CONTRIBUIÇÕES DA TESE	242
8.2	TRABALHOS SEMELHANTES.....	246
8.3	TRABALHOS FUTUROS	247
9	BIBLIOGRAFIA DE REFERÊNCIA.....	248

LISTA DE TABELAS

<u>TABELA 3.1: CONCEITOS BÁSICOS DO MODELO ODP (ISO-10746-2, 1995)</u>	58
<u>TABELA 3.2: REGRAS DE ESPECIFICAÇÃO DO MODELO ODP (ISO-10746-2, 1995)</u>	59
<u>TABELA 3.3: ELEMENTOS DA LINGUAGEM DE DESCRIÇÃO DA EMPRESA (ISO-15414, 2002)</u>	60
<u>TABELA 3.4: RESUMO DAS CARACTERÍSTICAS DOS PONTOS DE VISTA DO MODELO ODP</u>	64
<u>TABELA 4.1: RESUMO DAS ESTRATÉGIAS DE PREVENÇÃO DE FALHA</u>	81
<u>TABELA 4.2: RESUMO DAS ESTRATÉGIAS DE TOLERÂNCIA A FALHA</u>	87
<u>TABELA 5.1: PRINCÍPIOS GERAIS DE USABILIDADE PARA IHC (DIX, 1998)</u>	93
<u>TABELA 5.2: CLASSIFICAÇÃO DOS MÉTODOS DE AVALIAÇÃO DE USABILIDADE</u>	103
<u>TABELA 6.1: ADEQUAÇÃO ENTRE AS TÉCNICAS DE SEGURANÇA E MODELO ODP</u>	136
<u>TABELA 6.2: ADEQUAÇÃO DAS TÉCNICAS DE ANÁLISE À METODOLOGIA DE SEGURANÇA</u>	156
<u>TABELA 6.3: REPRESENTAÇÃO DOS ATRIBUTOS DE USABILIDADE NO MODELO ODP</u>	158
<u>TABELA 6.4: ASSOCIAÇÃO DOS ATRIBUTOS DE USABILIDADE E O MODELO ODP</u>	159
<u>TABELA 6.5: ADEQUAÇÃO DAS TÉCNICAS DE AVALIAÇÃO À METODOLOGIA DE USABILIDADE</u>	177
<u>TABELA 6.6: RESUMO DAS CARACTERÍSTICAS DE CADA FASE DA MERUSA</u>	189
<u>TABELA 7.1: RELAÇÃO DOS CASOS DE USO BÁSICOS DO SSC-STE</u>	206
<u>TABELA 7.2: REQUISITOS DO SSC-STE E SEU NÍVEL DE SENSIBILIDADE</u>	210
<u>TABELA 7.3: REQUISITOS EXCLUÍDOS DO SSC-STE</u>	212
<u>TABELA 7.4: REQUISITOS DE SEGURANÇA DO SSC-STE E SEU NÍVEL DE SENSIBILIDADE</u>	215
<u>TABELA 7.5: RECOMENDAÇÕES DA ANÁLISE DE SEGURANÇA DO SSC-STE</u>	218
<u>TABELA 7.6: REQUISITOS DE USABILIDADE DO SSC-STE</u>	221
<u>TABELA 7.7: AVALIAÇÃO DE USABILIDADE DO SSC-STE</u>	224
<u>TABELA 7.8: ANÁLISE DE COMPROMISSOS PARA O SSC-STE</u>	227
<u>TABELA 7.9: REQUISITOS DO SSC-STE DO PONTO DE VISTA DA EMPRESA</u>	228
<u>TABELA 7.10: REQUISITOS DO SSC-STE DO PONTO DE VISTA DA INFORMAÇÃO</u>	229
<u>TABELA 7.11: REQUISITOS DO SSC-STE DO PONTO DE VISTA DA COMPUTAÇÃO</u>	231
<u>TABELA 7.12: REQUISITOS DO SSC-STE DO PONTO DE VISTA DA ENGENHARIA</u>	233
<u>TABELA 7.13: REQUISITOS DO SSC-STE DO PONTO DE VISTA DA TECNOLOGIA</u>	236

LISTA DE FIGURAS

FIGURA 1.1: ORGANIZAÇÃO DO PROCESSO DE DESENVOLVIMENTO DO TRABALHO DA TESE	7
FIGURA 2.1: MODELO DO PROCESSO DE ENGENHARIA DE REQUISITOS (LEITE, 1994)	13
FIGURA 2.2: ATRIBUTOS DE QUALIDADE DE SOFTWARE (ISO-9126-1, 2000)	17
FIGURA 2.3: PROCESSO DE DESENVOLVIMENTO DE SISTEMAS (KOTONYA, 1998)	19
FIGURA 2.4: ETAPAS DO MÉTODO ATAM (KAZMAN, 1998)	25
FIGURA 2.5: INTEGRAÇÃO DOS MODELOS GRL E UCM (LIU, 2001)	38
FIGURA 3.1: PONTOS DE VISTA DO MODELO ODP (PUTMAN, 2001)	41
FIGURA 3.2: EXEMPLO DE RECURSIVIDADE DO MODELO ODP (PUTMAN, 2001)	63
FIGURA 5.1: ATRIBUTOS DE QUALIDADE EM USO DE SOFTWARE (ISO-9126-1, 2000)	92
FIGURA 5.2: CICLO DE VIDA DE USABILIDADE (MAYHEW, 1999)	98
FIGURA 5.3: INTEGRAÇÃO DOS FATORES HUMANOS A AVALIAÇÃO DE RISCO DO SISTEMA	117
FIGURA 5.4: FLUXOGRAMA DO MÉTODO ATHEANA (NEA, 1998)	121
FIGURA 6.1: ETAPAS DE DESENVOLVIMENTO DA METODOLOGIA MERUSA	124
FIGURA 6.2: METODOLOGIA DE ELICITAÇÃO E ANÁLISE DE REQUISITOS DE SEGURANÇA	143
FIGURA 6.3: METODOLOGIA DE SEGURANÇA ASSOCIADA AO CICLO DE VIDA DO SISTEMA	152
FIGURA 6.4: HIERARQUIA DOS REQUISITOS ENTRE AS VISÕES DO MODELO ODP	154
FIGURA 6.5: METODOLOGIA DE ELICITAÇÃO E ANÁLISE DE REQUISITOS DE USABILIDADE	166
FIGURA 6.6: METODOLOGIA DE USABILIDADE ASSOCIADA AO CICLO DE VIDA DO SISTEMA	174
FIGURA 6.7: ETAPAS DA MERUSA CONSIDERANDO VÁRIOS CICLOS DE ITERAÇÃO	181
FIGURA 6.8: EXEMPLO DE SEQUÊNCIA TEMPORAL ENTRE ITERAÇÕES DA MERUSA	187
FIGURA 7.1: COMUNIDADES DO SETOR ELÉTRICO BRASILEIRO	195
FIGURA 7.2: META-ARQUITETURA ODP PARA O STE DO SISTEMA INTERLIGADO NACIONAL	199
FIGURA 7.3: ATIVIDADES DO CASO DE ESTUDO DA MERUSA	203

GLOSSÁRIO DE ABREVIATURAS E SIGLAS

ABAS	- ATTRIBUTE-BASED ARCHITECTURAL STYLES
ABNT	- ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS
AEA	- ACTION ERROR ANALYSIS
ALARP	- AS LOW AS IS REASONABLY PRACTICABLE
ANA	- AGÊNCIA NACIONAL DE ÁGUAS
ANEEL	- AGÊNCIA NACIONAL DE ENERGIA ELÉTRICA
ANSI	- AMERICAN NATIONAL STANDARDS INSTITUTE
AQD	- AQUISIÇÃO DE DADOS
ATAM SM	- ARCHITECTURE TRADEOFF ANALYSIS METHOD
ATHEANA	- A TECHNIQUE FOR HUMAN ERROR ANALYSIS
BD	- BASE DE DADOS
CCA	- CAUSE-CONSEQUENCE ANALYSIS
CCEE	- CÂMARA DE COMERCIALIZAÇÃO DE ENERGIA ELÉTRICA
CEMIG	- COMPANHIA ENERGÉTICA DE MINAS GERAIS
CHESF	- COMPANHIA HIDROELÉTRICA DO SÃO FRANCISCO
CMD	- COMUNICAÇÃO DE DADOS
CN	- CENÁRIOS
CNOS	- CENTRO NACIONAL DE OPERAÇÃO DO SISTEMA
CNS	- CENÁRIO DE SEGURANÇA
CNU	- CENÁRIO DE USABILIDADE
COL	- CENTRO DE OPERAÇÃO LOCAL
COPEL	- COMPANHIA PARANAENSE DE ENERGIA
CORBA	- COMMON OBJECT REQUEST BROKER ARCHITECTURE
COS	- CENTRO DE OPERAÇÃO DO SISTEMA
COSR	- CENTRO REGIONAL DE OPERAÇÃO DO SISTEMA
COTS	- COMMERCIAL OFF-THE-SHELF
CPFL	- COMPANHIA PIRATINGA DE FORÇA E LUZ S/A
CRO	- CENTRO REGIONAL DE OPERAÇÃO
CTEEP	- COMPANHIA DE TRANSMISSÃO DE ENERGIA ELÉTRICA PAULISTA
DE	- REQUISITO TIPO DESEMPENHO
DISP	- DISPONIBILIDADE
EAT	- EXTRA-ALTA TENSÃO
ELETROSUL	- ELETROSUL CENTRAIS ELÉTRICAS S/A
EPTE	- EMPRESA DE PAULISTA DE TRANSMISSÃO DE ENERGIA
ER	- DENSIDADE DE ENERGIA
ETA	- EVENT TREE ANALYSIS

FMECA	- FAILURE MODES, EFFECTS AND CRITICALITY ANALYSIS
FTA	- FAULT TREE ANALYSIS
FURNAS	- FURNAS CENTRAIS ELÉTRICAS S/A
GPS	- GLOBAL POSITIONING SYSTEM
GRL	- GOAL-ORIENTED REQUIREMENT LANGUAGE
HAZOP	- HAZARDS AND OPERABILITY ANALYSIS
HEI	- HUMAN ERROR IDENTIFICATION
HEP	- HUMAN ERROR PROBABILITY
H-HAZOP	- HUMAN HAZARD AND OPERABILITY ANALYSIS
HRA	- HUMAN RELIABILITY ASSESSMENT
IBAMA	- INSTITUTO BRASILEIRO DO MEIO AMBIENTE
IEC	- INTERNATIONAL ELECTROTECHNICAL COMMISSION
IHC	- INTERFACE HOMEM COMPUTADOR
ISO	- INTERNATIONAL ORGANIZATION FOR STANDARDIZATION
ITU	- INTERNATIONAL TELECOMMUNICATIONS UNION
MERUSA	- METODOLOGIA DE ESPECIFICAÇÃO DE REQUISITOS DE USABILIDADE E SEGURANÇA ORIENTADA PARA ARQUITETURA
MME	- MINISTÉRIO DAS MINAS E ENERGIA
MORT	- MANAGEMENT OVERSIGHT AND RISK TREE ANALYSIS
MTBF	- MEAN TIME BETWEEN FAILURES
MTTR	- MEAN TIME TO REPAIR
OB	- REQUISITO TIPO OBRIGAÇÃO
OBE	- OBJETO BÁSICO DE ENGENHARIA
ODP	- OPEN DISTRIBUTED PROCESSING
OG	- OBJETIVO GERAL
OGS	- OBJETIVO GERAL DE SEGURANÇA
ONS	- OPERADOR NACIONAL DO SISTEMA
OOSE	- OBJECT ORIENTED SOFTWARE ENGINEERING
PE	- REQUISITO TIPO POSSIBILIDADE DE EXPANSÃO
PR	- REQUISITO TIPO PROIBIÇÃO
PRA	- PROBABILISTIC RISK ASSESSMENT
PTA	- PROCEDURE OR TASK ANALYSIS
QoS	- QUALITY OF SERVICE
RA	- REQUISITO ADICIONAL
RDP	- REQUISITO DE DESEMPENHO
RE	- REQUISITO TIPO RESTRIÇÃO ESPECÍFICA
RF	- REQUISITO FUNCIONAL
RGD	- REGISTRO DE DADOS

RM-ODP	- REFERENCE MODEL OF OPEN DISTRIBUTED PROCESSING
RNF	- REQUISITO NÃO FUNCIONAL
RSA	- REQUISITOS DE SEGURANÇA ASSOCIADO A ANÁLISE
RSG	- REQUISITO DE SEGURANÇA
RUC	- REQUISITO DE USABILIDADE COMPLEMENTAR
RUS	- REQUISITO DE USABILIDADE
SAA	- SISTEMAS ABERTOS DE AUTOMAÇÃO
SE	- SUBESTAÇÃO (DE ENERGIA ELÉTRICA)
SEI	- SOFTWARE ENGINEERING INSTITUTE
SFMECA	- SOFTWARE FAILURE MODES, EFFECTS AND CRITICALITY ANALYSIS
SFTA	- SOFTWARE FAULT TREE ANALYSIS
SHELL	- SOFTWARE, HARDWARE, ENVIRONMENT AND LIVEWARE
SIN	- SISTEMA INTERLIGADO NACIONAL
SMHA	- STATE MACHINE HAZARD ANALYSIS
SOE	- SEQUENCE OF EVENTS
SSC	- SISTEMA DE SUPERVISÃO E CONTROLE
SSC-STE	- SISTEMA DE SUPERVISÃO E CONTROLE PARA SUBESTAÇÕES DE TRANSMISSÃO DE ENERGIA
STE	- SISTEMAS DE TRANSMISSÃO DE ENERGIA
TA	- TEMPERATURA AMBIENTE
TTD	- TRATAMENTO DE DADOS
UC	- CASO DE USO
UCM	- USE CASE MAPS
UDI	- UNIVERSO DA INFORMAÇÃO
UML TM	- UNIFIED MODELING LANGUAGE
UR	- UMIDADE RELATIVA
URN	- USER REQUIREMENT NOTATION
UUC	- CASO DE USO DE USABILIDADE
V&V	- VERIFICAÇÃO E VALIDAÇÃO
WEB	- THE WORLD WIDE WEB
WSA	- WORK SAFETY ANALYSIS
XML	- W3C'S EXTENSIBLE MARKUP LANGUAGE
Σ _{Er}	- ENERGIA MÉDIA

1 INTRODUÇÃO

O objetivo desta pesquisa é apresentar uma **Metodologia de Especificação de Requisitos de Usabilidade e Segurança orientada para Arquitetura** do sistema, denominada **MERUSA**, aplicável a processos de desenvolvimento de sistemas digitais de automação distribuídos e abertos.

Este capítulo apresenta a motivação para este trabalho, a sua abrangência, a sua relevância e o processo utilizado para a avaliação da adequação da metodologia à eliciação e análise de requisitos de usabilidade e segurança para processos de desenvolvimento de sistemas digitais distribuídos e abertos.

1.1 Motivação

É inegável a dependência do mundo moderno com relação aos sistemas baseados em software. A crescente utilização de sistemas baseados em computador, nas mais diversas aplicações industriais e domésticas (desde brinquedos e eletrodomésticos a sistemas de transportes, telecomunicações, controle de processos e suporte de vida), faz com que a saúde e o bem estar das pessoas sejam dependentes desses sistemas. Concomitantemente, a funcionalidade e a sofisticação desses sistemas tem crescido, sendo que esses incrementos de funcionalidade são geralmente implementados através de software.

Apesar dessa participação essencial, o software como produto ainda utiliza processos de produção bastante insatisfatórios. São vários os prejuízos financeiros e também de vidas humanas que foram causados devido a erros gerados no processo de desenvolvimento do software (LUTZ, 2000).

Quando a crise do software foi identificada nos anos sessenta, muito esforço foi dirigido para encontrar as causas da síndrome dos problemas. As investigações determinaram que as deficiências nos requisitos estariam entre os mais importantes contribuintes para o problema: em quase todos os projetos de software que falharam em atender aos objetivos de desempenho e às metas de custo, as deficiências de

requisitos representaram um papel principal e caro no fracasso do projeto (STANDISH, 1995). O desenvolvimento da especificação de requisitos em muitos casos parece trivial, mas provavelmente é a parte do processo que conduz a mais fracassos que qualquer outro (SOMMERVILLE, 1993).

A necessidade da obtenção de bons requisitos aumenta com o tamanho e a complexidade do sistema a ser desenvolvido. Adicionalmente, sistemas que contém um grande volume de software geralmente apresentam uma maior complexidade inerente, ou seja, são mais difíceis de serem entendidos do que sistemas que não contém grande dependência de software; assim, esses sistemas são mais sensíveis à qualidade dos seus requisitos (DORFMAN, 1997).

Associada à crescente utilização de sistemas baseados em computador está a expansão da infra-estrutura de redes de comunicação distribuídas e abertas, viabilizando a integração de vários sistemas de automação (de diversas tecnologias) e fornecendo recursos mais eficazes de supervisão, controle e comando remoto.

Com o avanço tecnológico e a expansão das redes de comunicação, existe uma busca incessante de integração, interoperabilidade, segurança, disponibilidade entre os mais diversos domínios, eminentemente distribuídos e heterogêneos.

Outra tendência associada à crescente expansão das redes de comunicação distribuídas é a adoção de arquiteturas de sistemas abertos com objetos distribuídos.

Como decorrência dessas tendências, surgiu a motivação desta pesquisa, cujo interesse é elaborar uma metodologia de especificação de requisitos, focada em aspectos não funcionais, para o desenvolvimento de sistemas computacionais com objetos distribuídos.

1.2 Abrangência do Trabalho

Neste trabalho focalizam-se duas classes de requisitos em especial, devido à sua relevância para sistemas críticos, bem como à prioridade em relação às necessidades de integração de sistemas de tempo real com diversas tecnologias e que

possuem componentes com diferentes níveis de criticidade. As classes de requisitos focalizadas em detalhe são segurança (considerando-se tanto a segurança relacionada com a falha do produto como a relacionada com acesso aos dados do sistema) e usabilidade (especialmente os requisitos relacionados com a operabilidade, a avaliação e a redução de erros humanos), uma vez que estas categorias de requisitos são as que apresentam maior necessidade de aprimoramento para assegurar uma operação confiável e segura dos sistemas interconectados por redes de comunicação atuais (PITAC, 1999); (NITRD, 2001).

Uma especificação de requisitos de boa qualidade deve produzir, além da definição correta do que é exigido do sistema, uma documentação adequada das funções, do desempenho, das interfaces internas e externas e dos atributos de qualidade do produto, bem como quaisquer restrições válidas para o projeto do sistema ou para o seu processo de desenvolvimento.

O termo especificação de requisitos, no contexto deste trabalho, engloba basicamente as etapas de elicitação (termo utilizado na literatura nacional, cujo original em inglês é: *elicitation*, tendo como sentido: extração, dedução, derivação) e análise de requisitos. Nesse sentido, a metodologia concentra-se no processo inicial do desenvolvimento do sistema, abrangendo a identificação dos requisitos de usabilidade e segurança e dos critérios para sua análise e sua validação, em relação à arquitetura dos sistemas de automação distribuídos.

A metodologia de elicitação e análise de requisitos desenvolvida nesta pesquisa está fortemente ligada à evolução da arquitetura do sistema, sendo pressuposto que exista um processo de controle do desenvolvimento da mesma.

Neste trabalho é considerado que a arquitetura do sistema é representada segundo o modelo de referência para o processamento distribuído e aberto (RM-ODP - *Reference Model of Open Distributed Processing*), que atualmente é considerado o mais completo e atualizado padrão para um modelo de arquitetura para sistemas distribuídos e abertos (ISO-10746-1, 1995); (BECERRA, 1998).

O processo de avaliação da evolução da arquitetura em função dos requisitos é baseado no método de análise de compromissos de arquitetura (ATAM -*Architecture Tradeoff Analysis Method*), desenvolvido no *Software Engineering Institute* (SEI) da *Carnegie Mellon University* (KAZMAN, 2000).

Para uma avaliação da aplicabilidade dessa metodologia a sistemas reais é desenvolvida, nesta pesquisa, uma meta-arquitetura que pode ser utilizada para a especificação de sistemas digitais de supervisão e controle distribuído de transmissão de energia elétrica, pertencentes ao Sistema Interligado Nacional (SIN) (ONS, 2002).

1.3 Justificativa

Várias metodologias de elicitação e análise de requisitos já foram elaboradas, porém a maioria destas, bem como as ferramentas já disponíveis para controle do processo de requisitos, aborda os aspectos funcionais do software. Os requisitos não funcionais (RNF) do sistema, relativos a desempenho, confiabilidade, disponibilidade, segurança, manutenibilidade, usabilidade, portabilidade, requisitos culturais e políticos, requisitos legais, também chamados de atributos de qualidade, ainda não são tratados de forma sistemática pelas ferramentas disponíveis, uma vez que estes são difíceis de serem quantificados e geralmente interagem de maneira conflitante entre eles e com os demais requisitos funcionais (RF) do sistema (VAN LAMSWEERDE, 2000); (GROSS, 2001).

Nesse contexto, a relevância da correta elicitação e análise dos RNF dos sistemas na fase inicial de projeto, que recai basicamente na avaliação da qualidade de produto de software, torna-se essencial para o sucesso do empreendimento (ABNT, 1999).

Adicionalmente, a metodologia desenvolvida neste trabalho abrange vários aspectos citados como desafios na literatura da área de engenharia de requisitos: o desenvolvimento de novas técnicas de modelagem e análise do ambiente do sistema, a aproximação entre técnicas de elicitação e técnicas de análise formal, a necessidade

de novos modelos de análise para RNF, o melhor entendimento do impacto das decisões de arquitetura na evolução dos requisitos, os modelos de reuso de requisitos e a necessidade de multidisciplinaridade na abordagem dos engenheiros de requisitos (VAN LAMSWEERDE, 2000); (NUSEIBEH, 2000).

A dinâmica da sociedade atual já é dependente de sistemas de grande escala baseados em software, que fazem parte da sua infra-estrutura. A grande complexidade, heterogeneidade, distribuição e integração desses grandes sistemas interconectados exigem o desenvolvimento de novas ferramentas de projeto, implementação e operação, apropriadas para a escala desses empreendimentos. As metodologias atuais devem ser aprimoradas em função da abrangência dos sistemas e da inerente heterogeneidade dos seus componentes (NITRD, 2001).

Dessa forma, esta tese justifica-se pela necessidade da elaboração de metodologias que contribuam para melhoria da qualidade do software nesse ambiente emergente de interoperabilidade entre sistemas heterogêneos altamente distribuídos e críticos do ponto de vista da infra-estrutura da sociedade.

A utilização do modelo de referência ODP deve-se também aos trabalhos de pesquisa já realizados nos Laboratórios de Tecnologia de Software e de Sistemas Abertos do Departamento de Engenharia de Computação e Sistemas Digitais da Escola Politécnica da Universidade de São Paulo. Nesses laboratórios têm-se desenvolvido tecnologias para identificar um modelo de arquitetura aberta para projetos de automação, também baseada no modelo ODP (BECERRA, 1998).

1.4 Metodologia de Trabalho

Com o objetivo de controlar o processo da pesquisa foi utilizada uma metodologia de trabalho que consiste nas seguintes etapas principais:

- Pesquisa bibliográfica: caracterizada pela análise da literatura existente e fundamentação dos conceitos de engenharia de requisitos, métodos de

desenvolvimento de arquiteturas, técnicas de avaliação de confiabilidade e segurança, conceitos de usabilidade e modelo ODP;

- Elaboração da metodologia: caracterizada pela definição da proposta de metodologia de elicitação e análise de requisitos, determinação de suas etapas e respectivos procedimentos;
- Elaboração da arquitetura de referência: caracterizada pela elaboração de uma meta-arquitetura para sistemas de supervisão e controle de transmissão de energia do sistema elétrico interligado brasileiro. Essa meta-arquitetura é utilizada como uma plataforma de avaliação para a metodologia proposta;
- Avaliação da metodologia: caracterizada pela aplicação da metodologia à meta-arquitetura de referência, para a avaliação da sua adequação a um sistema distribuído e heterogêneo.

A organização dos conhecimentos envolvidos com o processo de desenvolvimento do trabalho da tese está esquematizada na figura 1.1. Nessa figura pode ser observado que o modelo de referência básico da metodologia é o modelo ODP. Como a metodologia é focada nos requisitos de segurança e usabilidade, em uma primeira etapa é verificada a adequação do modelo ODP às ferramentas de avaliação desses requisitos, que consiste na verificação de potenciais conflitos entre os conceitos propostos no modelo ODP e os conceitos utilizados na avaliação desses requisitos. Os conceitos e ferramentas básicas utilizados no desenvolvimento da metodologia são oriundos da engenharia de requisitos e da engenharia de usabilidade. Dentro da engenharia de requisitos adota-se, como referência, a estratégia de desenvolvimento centrada em arquitetura (baseado no método ATAM). A engenharia de usabilidade fornece, como referência, a estratégia de desenvolvimento centrada em fatores humanos. A harmonização dessas estratégias de desenvolvimento, juntamente com os conceitos do modelo ODP, tem como resultado, uma proposta de metodologia de especificação de requisitos de usabilidade e segurança orientada para a arquitetura, denominada MERUSA.

Figura 1.1: Organização do processo de desenvolvimento do trabalho da tese

Finalmente, essa metodologia é avaliada utilizando-se uma meta-arquitetura para sistemas de supervisão e controle distribuídos de transmissão de energia para o sistema elétrico brasileiro (denominado SIN - Sistema Interligado Nacional).

1.5 Estrutura da Tese

A tese está subdividida em oito capítulos, cujos conteúdos são os seguintes:

Capítulo 1 – Introdução: neste capítulo são apresentados os objetivos da tese, a motivação, as justificativas para sua realização e a metodologia utilizada para o seu desenvolvimento (BECERRA, 1998); (NUSEIBEH, 2000); (VAN LAMSWEERDE, 2000); (NITRD, 2001); (KNIGHT, 2002);

Capítulo 2 – Processo de Engenharia de Requisitos: neste capítulo são apresentadas as noções básicas do processo de engenharia de requisitos, enfatizando-se as etapas de elicitação e análise de requisitos, sua modelagem, sua relação com o ciclo de vida do sistema e com o desenvolvimento da arquitetura. São descritos também as características básicas do método ATAM e as técnicas mais relevantes para modelagem dos requisitos (LEITE, 1994); (DORFMAN, 1997); (KOTONYA, 1998); (KAZMAN, 2000); (CYSNEIROS, 2001); (LIU, 2001); (AMYOT, 2002);

Capítulo 3 – Modelo de Sistemas Abertos com Arquitetura de Objetos Distribuídos: neste capítulo são apresentados os conceitos do modelo ODP, focalizando principalmente as suas funções e as cinco visões de especificação previstas no mesmo (ISO-10746-1, 1995); (BECERRA, 1998); (PUTMAN, 2001);

Capítulo 4 – Técnicas de Avaliação de Dependabilidade: neste capítulo são apresentados os critérios e as técnicas utilizados para a avaliação dos requisitos relacionados com a confiabilidade e segurança em sistemas críticos de controle (LEVESON, 1996); (IEC-1508-1, 1997); (EAMES, 1999); (HUSSEY, 2000);

Capítulo 5 – Técnicas de Avaliação de Usabilidade: neste capítulo são apresentados os critérios e as técnicas utilizados para a avaliação de usabilidade, sendo também discutidos os modelos de avaliação de erros humanos (MAYHEW, 1999); (FILGUEIRAS, 2000); (ISO-9126-1, 2000); (USABILITY, 2003); (HARTSON, 2004);

Capítulo 6 – MERUSA: Metodologia de Especificação de Requisitos de Usabilidade e Segurança orientada para Arquitetura: neste capítulo é realizada a verificação da adequação das técnicas de análise de segurança e de usabilidade ao modelo ODP, bem como é apresentada a metodologia de especificação de requisitos de usabilidade e segurança orientada para arquitetura, detalhando suas fases, procedimentos e modelos (KOTONYA, 1995); (EAMES, 1999); (ROLLAND, 1999); (ALLEMAN, 2000); (KAZMAN, 2000); (DAOUK, 2001);

Capítulo 7 – Avaliação da Metodologia: neste capítulo é elaborada uma meta-arquitetura, baseada no modelo ODP, aplicável a Sistemas de Supervisão e

Controle para Sistemas de Transmissão de Energia (SSC-STE). Neste capítulo também são apresentados os resultados de um caso de estudo conduzido com a aplicação da MERUSA ao SSC-STE, para a avaliação da aplicabilidade dessa metodologia (PUTMAN, 2001); (ONS, 2002); (MME, 2003); (AVELINO, 2004);

Capítulo 8 – Considerações Finais: neste capítulo são apresentadas as conclusões gerais da tese, os comentários finais, a contribuição acadêmica e os trabalhos que poderão ser realizados no futuro, relacionados com esses resultados.

2 ENGENHARIA DE REQUISITOS

Os sistemas computacionais com grande volume de software apresentam maior complexidade inerente, ou seja, são mais difíceis de serem compreendidos, sendo mais sensíveis à qualidade dos seus requisitos, assumindo aspectos particularmente críticos quando a função do software tem alguma relação com a segurança do processo controlado (DORFMAN, 1997).

Fontes históricas têm confirmado que o problema de definição de requisitos atinge grande quantidade de sistemas. Um levantamento de cerca de 8000 projetos, em 350 companhias norte-americanas, revelou que cerca de um terço destes projetos não foram completados ou foram com sucesso parcial, apresentando funcionalidade parcial, custo excessivo ou atrasos significativos. Essa pesquisa mostrou que 47% dos executivos responsáveis por esses projetos atribuíram essas falhas a problemas com requisitos, mais especificamente: falta de envolvimento do usuário (13%), requisitos incompletos (12%), mudança de requisitos (11%), expectativas não realistas (6%) e objetivos mal definidos (5%) (STANDISH, 1995).

As próximas seções apresentam como o processo de definição e gerenciamento de requisitos contribui com o ciclo de vida de desenvolvimento do sistema, assim como os conceitos fundamentais associados a esses processos, destacando-se as atividades de elicitação, modelagem e análise de requisitos.

2.1 Fundamentos da Engenharia de Requisitos

O processo de desenvolvimento de sistema tem seu início quando se tem estabelecido o que se quer produzir. O estabelecimento desses objetivos é realizado através da identificação dos requisitos do sistema.

A IEEE Std-610.12:1990 apresenta a seguinte definição para requisito de sistema: Requisito é: (1) uma condição ou capacidade exigida pelo usuário para resolver um problema ou atingir um objetivo; (2) uma condição ou capacidade que deve ser atendida ou possuída por um sistema ou componente do sistema para

satisfazer um contrato, norma, especificação ou outro documento formalmente imposto; (3) uma representação documentada de uma condição ou capacidade como em (1) ou (2) (IEEE-STD-610.12, 1990).

A definição dessas condições ou capacidades para um sistema complexo é uma tarefa difícil, sendo que, desempenhar esta tarefa sem recursos adequados pode torná-la impossível. Durante muito tempo, as pesquisas em engenharia de software centraram esforços relacionados com os requisitos na modelagem do sistema de software, desenvolvendo linguagens e técnicas de representação que foram se tornando mais abstratas e, ao mesmo tempo, mais completas (podendo representar mais elementos) e mais formais. Em função desse processo, muito pouca atenção foi dada para a tarefa de coletar ou descobrir o que o sistema tinha que realmente fazer, pois o mais importante era a representação do software do sistema (LEITE, 1994).

A engenharia de requisitos é a disciplina que procura sistematizar o processo de definição de requisitos, definindo métodos, técnicas e ferramentas que facilitam esse trabalho de definição. Essa sistematização se faz necessária na medida em que os sistemas atuais tornam-se cada vez mais complexos, exigindo uma grande atenção na correta compreensão do problema antes da elaboração de uma solução, sob pena do seu insucesso (NUSEIBEH, 2000).

Neste trabalho adota-se a definição de engenharia de requisitos de Leite (1994): A engenharia de requisitos estabelece o processo de definição de requisitos como um processo no qual o que deve ser feito é elicitado, modelado e analisado. Este processo deve lidar com diferentes pontos de vista, e usar uma combinação de métodos, ferramentas e pessoal. O produto desse processo é um modelo, do qual um documento, chamado de especificação de requisitos é produzido. Este processo acontece num contexto previamente definido, denominado de Universo de Informações (UdI) (LEITE, 1994).

Esta definição é adequada para este trabalho porque estabelece a engenharia de requisitos como um processo dinâmico, envolvendo vários pontos de vista e

considerando o contexto onde o sistema está envolvido (UdI). Essas características são os pontos básicos da metodologia proposta neste trabalho.

O UdI é o contexto geral no qual o sistema (em especial o software) deve ser desenvolvido e operado. Esse universo contempla todas as fontes de informação e todas as pessoas envolvidas com o sistema. Essas pessoas são os atores desse universo (denominados de *stakeholders*). Dessa forma o UdI é a realidade circunstanciada pelo conjunto de objetivos definidos por quem demanda o sistema. No contexto do desenvolvimento de software o UdI é também denominado como domínio da aplicação (LEITE, 1992).

Os processos de elicitação, modelagem e análise podem ser definidos da seguinte forma (LEITE, 1994):

- Elicitação: é a tarefa de identificação dos fatos que compõem os requisitos do sistema (inclui-se neste processo a análise do domínio), de forma a prover o mais correto e mais completo entendimento do que é exigido do sistema;
- Modelagem: consiste na forma de representação dos fatos extraídos, de um modo sistemático, segundo as técnicas de modelagem utilizadas (inclui-se neste processo a elaboração do documento de especificação de requisitos);
- Análise: consiste do processo de verificação e validação do resultado da modelagem (inclui-se neste processo a análise da especificação).

É importante ressaltar que existe um processo de gerenciamento de requisitos que está implícito no controle dos processos de elicitação, modelagem e análise. O gerenciamento de requisitos é definido como o processo de controlar as mudanças nos requisitos durante o processo de engenharia de requisitos e o desenvolvimento do sistema (SOMMERVILLE, 1993).

A gestão dos requisitos é essencial em qualquer processo de desenvolvimento de software (PAULK, 1993). Essa necessidade decorre dos requisitos serem, inevitavelmente, incompletos e inconsistentes; é uma característica comum que

novos requisitos sejam identificados durante o processo de desenvolvimento, quando os negócios necessitam de mudanças, ou devido a uma melhor compreensão do sistema; também é freqüente que pontos de vista diferentes possuam requisitos diferentes e que sejam, muitas vezes, contraditórios (SOMMERVILLE, 1993).

O enfoque de investigação da engenharia de requisitos é o processo de obtenção e controle dos mesmos. Os engenheiros de requisitos executam a análise e a definição de requisitos, bem como investigam em teoria como os requisitos deveriam ser desenvolvidos. Para facilitar a compreensão dos elementos envolvidos na definição dos requisitos é apresentado na figura 2.1 um modelo, proposto por Leite (1994), para o processo de engenharia de requisitos.

Figura 2.1: Modelo SADT do processo de engenharia de requisitos (LEITE, 1994)

Neste modelo o processo recebe, como entrada, as informações do UdI e seleciona os participantes, os métodos e as ferramentas a serem utilizados. O ciclo

Elicitar → Modelar → Analisar se repete até que, por consenso dos participantes, chega-se à conclusão de que os requisitos já estão definidos. Como saídas principais desse modelo existem os requisitos e o próprio modelo. Adicionalmente, os resultados das análises, denominados de deltas, representam a avaliação do modelo. Os requisitos podem ser representados em linguagem natural e o modelo procura espelhar o mais corretamente possível sua semântica (LEITE, 1994).

Essa modelagem auxilia o entendimento do processo de obtenção de requisitos como um todo. Entretanto, as divisões entre elicitação, modelagem e análise são geralmente difíceis de serem distintas na prática, uma vez que existe um forte vínculo entre elas (CHRISTEL, 1992); (DORFMAN, 1997).

A execução do processo, apresentado na figura 2.1, envolve a realização de múltiplas atividades interrelacionadas (VAN LAMSWEERDE, 2000):

- Análise do domínio: consiste no estudo do sistema existente no qual o software e o sistema a ser desenvolvido devem ser instalados. Nessa atividade os envolvidos com o sistema devem ser identificados e entrevistados; os problemas e deficiências do sistema existente devem ser considerados; novas oportunidades devem ser investigadas; os objetivos gerais do sistema alvo devem ser identificados;
- Elicitação de requisitos: nesta atividade devem ser explorados modelos alternativos para que o sistema alvo atinja seus objetivos; requisitos e hipóteses sobre os componentes desses modelos são identificados, possivelmente com a ajuda de cenários hipotéticos de interação; devem ser definidas as fronteiras entre o sistema e seu ambiente;
- Negociação e acordos: os requisitos e hipóteses alternativas devem ser avaliados, bem como devem ser realizadas análises de risco, para chegar-se ao melhor compromisso entre as características conflitantes e que sejam aceitas por todas as partes envolvidas;

- Especificação: os requisitos e as hipóteses sobre o sistema devem ser formuladas de uma forma precisa;
- Análise da especificação: as especificações devem ser verificadas quanto às suas deficiências (tais como: inadequação, falta de completeza ou inconsistências) e quanto a sua exequibilidade (em termos de recursos necessários, custos de desenvolvimento, prazos, etc.);
- Documentação: as várias decisões tomadas durante o processo devem ser documentadas, juntamente com suas hipóteses e justificativas, visando principalmente a sua rastreabilidade;
- Gerenciamento dos requisitos: os requisitos podem ser modificados durante o processo de desenvolvimento e mesmo durante o período de operação e manutenção do sistema (para acomodar correções, mudanças no ambiente ou novos objetivos), assim, a rastreabilidade da evolução dos requisitos deve garantir a identificação da origem dos requisitos, permitindo a identificação da sua necessidade e a influência da sua modificação face aos demais requisitos (provavelmente alguns derivados do requisito original).

A definição de processo de engenharia de requisitos, bem como as realimentações do modelo da figura 2.1, sugerem que o processo de engenharia de requisitos seja bastante complexo. Dentre os fatores que levam a essa complexidade pode-se citar (VAN LAMSWEERDE, 2000):

- O escopo do UdI é extremamente amplo, abrangendo desde o mundo das organizações humanas ou das leis físicas até os artefatos técnicos que devem ser integrados e dos objetivos de alto nível do sistema até os seus procedimentos operacionais;
- Existem múltiplas restrições que devem ser consideradas além dos aspectos funcionais, por exemplo, características de segurança, usabilidade, interoperabilidade, desempenho, custo, dentre outras. Observa-se que as características não funcionais são geralmente conflitantes;

- Existem múltiplos participantes envolvidos no processo de engenharia de requisitos, cada um com seu conhecimento específico, habilidades, histórico, restrições, percepções e meios de expressão, sendo que geralmente esses partícipes possuem pontos de vista conflitantes;
- As especificações de requisitos podem apresentar uma grande gama de deficiências. Algumas dessas deficiências são erros que podem levar a desastrosos efeitos nas fases subsequentes de desenvolvimento ou na qualidade do produto de software resultante.

Em função dessa complexidade, são necessárias técnicas rigorosas, envolvendo múltiplas disciplinas (incluindo-se psicologia e sociologia), para um efetivo suporte ao processo de engenharia de requisitos (NUSEIBEH, 2000).

2.2 Requisitos Funcionais (RF) e Não Funcionais (RNF)

As técnicas e as ferramentas da engenharia de requisitos geralmente são relativas à elicitación e análise das características funcionais do sistema. Entretanto, os requisitos do sistema incluem outras necessidades que podem não estar diretamente relacionadas com os aspectos funcionais do mesmo.

Uma característica comum a todos os tipos de sistemas é que eles possuem um conjunto de propriedades que dependem da integração de seus componentes. Essas características, chamadas de emergentes, são propriedades do sistema global e que só são identificadas depois que todos os subsistemas individuais tenham sido integrados. Exemplos desse tipo de propriedades para sistemas baseados em computadores são confiabilidade, disponibilidade, manutenibilidade, desempenho, usabilidade, segurança, entre outras. Os requisitos de sistema geralmente têm essas propriedades, não sendo restritos a requisitos funcionais (RF) (NUSEIBEH, 2000).

Os requisitos não funcionais (RNF) são definidos como sendo as qualidades ou atributos globais do sistema resultante, que não estão especificamente relacionados com a funcionalidade do sistema, mas que se caracterizam como restrições

associadas ao modo de melhor atender às necessidades do usuário (KOTONYA, 1998).

Este trabalho é direcionado para sistemas fortemente baseados em software e adota a classificação de RNF estabelecida na ISO/IEC 9126:2000, que tem sido amplamente utilizada na área de engenharia de software (ABNT, 1999).

Para os requisitos de produto de software, a ISO/IEC 9126:2000 apresenta um modelo de qualidade que categoriza os atributos de qualidade de software em seis características principais (funcionalidade, confiabilidade, usabilidade, eficiência, manutenibilidade e portabilidade), as quais são subdivididas em atributos secundários, visando facilitar o processo de avaliação do software através de métricas de qualidade. A figura 2.2 apresentada esta classificação, sendo apresentados com fundo cinza os RNF focalizados neste trabalho (ISO-9126-1, 2000).

Figura 2.2: Atributos de qualidade de software (ISO-9126-1, 2000)

Observando-se a classificação de atributos de qualidade de software da figura 2.2 verifica-se que o atributo de segurança (denominado de *safety*) não está explicitado nesse modelo, tendo como justificativa o fato de que a segurança está ligada à totalidade do sistema, sendo qualificado como atributo de qualidade em uso

(não apenas associado ao produto de software). Observa-se também que o atributo de segurança de acesso (denominado de *security*) está relacionado como um atributo funcional.

Apesar dessas diferenças de classificação, os atributos de segurança e segurança de acesso, bem como os aspectos de confiabilidade, disponibilidade, manutenibilidade e tolerância a falhas, são tratados neste trabalho de um modo integrado, sendo esse conjunto de requisitos denominado de requisitos de dependabilidade. Assim, a dependabilidade pode ser definida como o conjunto de propriedades de um sistema que justifica a confiança atribuída ao mesmo (STOREY, 1997).

Um modelo mais detalhado dos atributos de qualidade voltados para a segurança, propondo métricas, pode ser encontrado em (PÁSCOA, 2002).

Os processos de desenvolvimento de software geralmente tratam os RNF de modo secundário e muito informal com relação a etapa de elicitação de requisitos, fazendo com que os modelos de RNF raramente sejam integrados na representação do domínio (normalmente estes são representados apenas como formas de restrição e condições de contorno, não associados entre si, nem associados explicitamente às demais características do sistema). Adicionalmente, os RNF geralmente são conflitantes e interagem com outros requisitos do sistema, tal que, muitas vezes, os RF tenham que ser sacrificados para atendimento das restrições não funcionais (especialmente se estas forem restrições relacionadas com a segurança). Como resultado, freqüentemente estes requisitos são de difícil verificação durante o desenvolvimento e de difícil validação no final desse processo (ROLLAND, 1999); (CYSNEIROS, 2001).

2.3 Engenharia de Requisitos e o Projeto da Arquitetura

Apesar dos processos de desenvolvimento de sistemas variarem muito de tamanho e complexidade, bem como dependerem do tipo de sistema e das

organizações envolvidas no seu desenvolvimento, existem algumas etapas principais que são comuns a todos eles. A figura 2.3 apresenta um processo genérico de desenvolvimento de sistemas, onde são destacadas as etapas que envolvem o processo de engenharia de requisitos (KOTONYA, 1998).

Figura 2.3: Processo de desenvolvimento de sistemas (KOTONYA, 1998)

As etapas da figura 2.3 tem as seguintes definições (KOTONYA, 1998):

- Definição dos requisitos de sistema: nesta etapa os requisitos globais do sistema são estabelecidos. Usualmente estes requisitos são expressos em um alto nível de abstração, geralmente utilizando uma linguagem natural, gerando a especificação de requisitos do sistema. Algumas restrições (consideradas críticas para o sucesso do sistema) podem estar incluídas nesta etapa, contribuindo para a eliciação dos RNF;
- Partição da arquitetura: nesta etapa é realizado o projeto da arquitetura do sistema, sendo o mesmo decomposto em um conjunto de subsistemas;

- Alocação de requisitos: nesta etapa os requisitos são distribuídos nos vários subsistemas, sendo definidos quais requisitos serão atribuídos ao hardware e quais serão atribuídos ao software;
- Decomposição de requisitos: nesta etapa os requisitos de alto nível de software (e de hardware) são decompostos em um conjunto mais detalhado de requisitos que serão aplicados aos componentes do sistema;
- Desenvolvimento dos subsistemas: nesta etapa os subsistemas de hardware e software são projetados e implementados em paralelo, prosseguindo o detalhamento dos requisitos em função do detalhamento destes;
- Integração do sistema: nesta etapa os subsistemas de hardware e software são unidos e testados em conjunto para comporem o sistema completo;
- Validação do sistema: nesta etapa o sistema é validado contra seus requisitos globais (definidos na especificação de requisitos do sistema e detalhados durante o processo de desenvolvimento do sistema).

As etapas da figura 2.3 estão representadas numa sequência progressiva, entretanto, existe uma interação entre cada etapa e a etapa anterior (interação especialmente forte entre as etapas de partição, alocação e decomposição) de modo que, se algum erro for detectado em relação à especificação de requisitos, a etapa anterior terá que ser revisada, evidenciando um mecanismo iterativo no processo da engenharia de requisitos em relação ao desenvolvimento do sistema.

Dessa forma, os modelos de ciclo de vida tradicionais (que pressupõem uma sequência de etapas, onde uma só inicia-se quando a anterior termina) são modelos muito simplificados, na medida em que eles não consideram a forte interação entre as etapas de decomposição e integração dos vários elementos que compõem o sistema. Na prática, tem sido observado que o modelo de desenvolvimento de sistema em espiral é o aquele que melhor modela essa realidade (DORFMAN, 1997).

O mecanismo iterativo de refinamento dos requisitos faz com que o processo de manutenção de requisitos do sistema possa ser uma atividade complexa, exigindo

uma documentação adequada e um processo de gerenciamento de requisitos extremamente organizado, para permitir a sua rastreabilidade em qualquer fase do ciclo de vida do sistema (FIORINI, 1998); (VAN LAMSWEERDE, 2000).

Na figura 2.3 evidencia-se um envolvimento entre a definição de requisitos e a partição da arquitetura do sistema, logo no início do processo de desenvolvimento. Neste contexto, o desenvolvimento da arquitetura do sistema associado à definição dos requisitos torna-se estratégica, uma vez que a necessidade de descrição da arquitetura do sistema define a estrutura do projeto, as estratégias de desenvolvimento e pode limitar o atendimento dos requisitos do sistema. Assim, os requisitos do sistema poderão ser mais robustos, se a definição da arquitetura do sistema for associada à elicitação dos requisitos, considerando-se a participação dos vários envolvidos nos processos de aquisição, definição, desenvolvimento, utilização e manutenção do sistema (*stakeholders*), bem como considerando-se os vários cenários de utilização (especialmente nas etapas de alocação e decomposição de requisitos) (IEEE-STD-1471, 2000).

Finalmente é importante ressaltar o entendimento do que seja a arquitetura do sistema. Neste trabalho adotada-se a definição proposta no modelo ODP, onde:

- Arquitetura: refere-se aos conceitos e regras que definem a estrutura, comportamento semântico e relacionamento entre partes de um sistema, podendo ser entendido como o planejamento de algo a ser construído. Essa definição inclui os elementos que compreendem as entidades do sistema, o relacionamento entre esses elementos, as restrições associadas a esses relacionamentos, focalizando as partes das entidades e focalizando as entidades como um todo (ISO-10746-2, 1995).

Essa definição de arquitetura, que compreende o software e o hardware do sistema, possibilita variados graus de abstração e de detalhamento, podendo a arquitetura representar uma parte ou o todo de um sistema. Assim, pode ser chamada de arquitetura do componente o inter-relacionamento (incluindo sua estrutura, comportamento semântico e restrições associadas) dos elementos que compõem esse

componente; a arquitetura de um subsistema pode ser o inter-relacionamento entre componentes que compõem este subsistema; a arquitetura de sistema pode ser o inter-relacionamento dos subsistemas que compõem esse sistema; e assim sucessivamente, permitindo uma visão hierárquica das diversas partes do sistema.

O resultado da aplicação dos processos de partição, alocação e decomposição de um sistema, representados na figura 2.3, é conhecido como projeto de arquitetura ou projeto de alto nível do sistema, que leva a uma definição da hierarquia até um certo nível de abstração, com a geração dos requisitos para todos os elementos e determinação das interfaces entre eles. Embora essa atividade seja chamada de projeto, a engenharia de requisitos está envolvida ao longo desse processo.

Assim, o desenvolvimento do projeto de arquitetura é um processo no qual os passos de análise de requisitos e projeto da arquitetura se alternam, com um detalhamento maior a cada iteração do ciclo de desenvolvimento. A saída da análise de requisitos é fornecida como entrada para a próxima fase de projeto, e a saída do projeto é utilizada como entrada da próxima fase de análise de requisitos.

Pode-se considerar que, no processo inicial de elicitação, os requisitos são inerentemente ambíguos, intuitivos e informais. Já o software é logicamente não intuitivo (por exemplo, é difícil de ser decifrado sem o conhecimento da sua linguagem) e deve ser interpretado pela máquina de uma forma não ambígua. A arquitetura do sistema pode ser considerada como a ponte semântica entre os requisitos e o software (ALLEMAN, 2000).

As ferramentas e métodos utilizados na engenharia de requisitos fornecem suporte a todos os aspectos de desenvolvimento do projeto da arquitetura (partição, alocação e decomposição), permitindo que os processos de análise de requisitos e desenvolvimento da arquitetura sejam integrados.

A qualidade dos sistemas de grande porte baseados em software são determinadas principalmente pelas características da arquitetura do sistema. Dessa forma, em sistemas de grande porte, a satisfação dos RNF podem depender mais da arquitetura global do sistema do que de boas práticas no nível de código (como

escolha de linguagem, detalhamento do projeto, algoritmos, estruturas de dados, testes, etc.). Assim, a elicitação e a análise dos RNF no início do processo de desenvolvimento são de extrema importância para a determinação da viabilidade e adequação da arquitetura ao sistema a ser desenvolvido. Entretanto, poucos trabalhos tem sido realizados para sistematizar técnicas que permitam extrair a descrição da arquitetura do sistema a partir da sua especificação de requisitos (VAN LAMSWEERDE, 2000).

2.4 Método de Análise de Compromissos de Arquitetura (ATAM)

Os trabalhos de pesquisa na área de engenharia de requisitos tem enfatizado a necessidade do gerenciamento dos compromissos (*tradeoffs*) entre requisitos, especialmente aqueles relacionados aos RNF (VAN LAMSWEERDE, 2000).

Os compromissos entre requisitos geralmente afloram a partir dos pontos de vista dos diversos usuários envolvidos nos processos de aquisição, definição, desenvolvimento, utilização e manutenção do sistema (*stakeholders*). Assim, para que o processo de elicitação de requisitos seja bem sucedido, é necessário que os pontos de vista de todas as partes envolvidas com o sistema sejam capturados e, eventualmente, integrados de um modo consistente (VAN LAMSWEERDE, 2000).

Observa-se que os conflitos entre requisitos não necessitam ser obrigatoriamente resolvidos no momento em que surgem, pois diferentes pontos de vista podem trazer outras informações relevantes durante o processo de elicitação de requisitos, mesmo que estes sejam conflitantes em algum aspecto (HUNTER, 1998).

A elicitação dos requisitos, a partir de questionários e entrevistas com os usuários envolvidos com o sistema, é uma das tarefas mais difíceis, principalmente porque as pessoas geralmente tem uma dificuldade natural de expressar suas necessidades de forma abstrata (VILLER, 1998).

A utilização de cenários operacionais, que consideram um sistema hipotético, tem sido aplicada como uma ferramenta eficaz na obtenção dos requisitos iniciais do

sistema, ainda que não se tenha uma compreensão detalhada do futuro sistema. Tipicamente um cenário é uma sequência temporal de eventos de interação entre o sistema de software a ser desenvolvido e o seu ambiente, em um contexto restrito, para a obtenção de um objetivo específico. Várias técnicas baseadas em cenários tem sido propostas para elicitación e validação de requisitos (IEEE-STD-1471, 2000)

Neste trabalho adota-se um método de desenvolvimento de arquitetura, baseado em cenários e nos compromissos dos requisitos de qualidade do sistema, desenvolvido no *Software Engineering Institute* (SEI) da *Carnegie Mellon University*, denominado ATAM (*Architecture Tradeoff Analysis Method*).

Esse método é direcionado para a análise dos requisitos da arquitetura do sistema; dessa forma, o modelo do processo de engenharia de requisitos (representado na figura 2.1) é executado por diversas vezes: no início do processo de desenvolvimento, para analisar as várias opções de arquitetura e, ao longo do ciclo de vida do sistema, quando a sua arquitetura é modificada (KAZMAN, 1998); (BASS, 1999).

O objetivo do ATAM é avaliar as conseqüências das decisões de arquitetura à luz dos requisitos de qualidade. A utilização do método permite revelar quão bem a arquitetura satisfaz aos objetivos de qualidade particulares (tais como desempenho e manutenibilidade), fornece uma indicação de como esses objetivos de qualidade interagem entre si (compromissos entre requisitos), e permite compreender as conseqüências das decisões de arquitetura com relação aos atributos de qualidade do sistema, através de um processo de avaliação de riscos.

O ATAM é um método baseado na identificação de riscos das decisões de arquitetura, onde não há a necessidade de se predizer, precisamente, o comportamento dos requisitos de qualidade, uma vez que isso pode ser impossível num estágio inicial de projeto (KAZMAN, 2000).

O ATAM utiliza um modelo de desenvolvimento em espiral (representado na figura 2.4), onde são consideradas várias opções de arquitetura para o sistema, denominadas de arquiteturas candidatas. Nessas arquiteturas são realizadas análises

de risco em relação à satisfação dos requisitos de qualidade e são avaliados os compromissos resultantes entre esses requisitos. Esse processo é iterativo, levando à escolha da arquitetura mais adequada aos requisitos, à análise dos compromissos entre os requisitos e ao refinamento sucessivo destes (BASS, 2001).

Figura 2.4: Etapas do método ATAM (KAZMAN, 1998)

Existem conceitos importantes utilizados neste método (KAZMAN, 2000):

- ❑ Caracterização dos requisitos de qualidade;
- ❑ Utilização de cenários;
- ❑ Utilização de estilos de arquitetura baseados em atributos;
- ❑ Utilização de diagrama de árvores de atributos;
- ❑ Definição de ponto de sensibilidade;
- ❑ Definição de pontos de compromisso (*tradeoff*).

A utilização do ATAM leva à caracterização dos requisitos, através da melhor compreensão e da concretização dos requisitos de qualidade. Isso é obtido em parte através da elicitacão de cenários a partir dos *stakeholders*, que devem claramente estabelecer os requisitos de qualidade em termos de estímulo e resposta do sistema.

A utilização de cenários, através de técnicas do tipo tempestade cerebral (*brainstorming*), também contribui para a comunicação e o consenso sobre os requisitos de qualidade.

Um estilo de arquitetura é um conjunto de restrições de arquitetura (restrições em componentes e nas suas interações), tal que essas restrições definam um conjunto ou uma família de arquiteturas que as satisfazem. Identificando estilos de arquitetura que compõem uma determinada arquitetura, podem ser definidas quais as estratégias que o arquiteto pode utilizar para atender a determinados objetivos do sistema. Analogamente, conhecendo-se os requisitos que uma determinada arquitetura deve atender, pode-se identificar qual estilo de arquitetura é o mais adequado para satisfazer esses requisitos. Por exemplo, um estilo cliente/servidor deve ser definido para assegurar que o sistema possa ter seu desempenho facilmente ampliado, de modo que o tempo médio de latência seja minimamente alterado, quando for dobrado o número de usuários do sistema de software da empresa. Os estilos de arquitetura baseados em atributos fornecem padrões de avaliações de atributos específicos que ilustram como cada decisão de arquitetura altera a obtenção da satisfação de determinado requisito de qualidade. O ATAM utiliza uma especialização desse processo denominado de ABAS (*Attribute-based architectural Styles*) (KLEIN, 1999).

Para o detalhamento dos atributos de qualidade, o ATAM utiliza uma árvore de atributos. Essa técnica é um mecanismo de descrição de refinamentos sucessivos, do nível geral para o detalhado (*top-down*), que permite a tradução das restrições do negócio nos cenários de avaliação dos atributos de qualidade (KAZMAN, 2000).

O ponto de sensibilidade é representado por uma propriedade de um ou mais componentes (e/ou relações entre componentes) que é crítica para a obtenção da

resposta de um particular atributo de qualidade. Por exemplo, o nível de confiabilidade em uma rede privada virtual pode ser sensível ao número de bits de encriptação, portanto, nesse caso, o número de bits de encriptação é um ponto de sensibilidade. Os pontos de sensibilidade indicam ao analista onde deve ser focada sua atenção, quando da verificação da satisfação de um determinado atributo de qualidade. Em muitos casos, os pontos de sensibilidade tornam-se riscos quando são dependentes da arquitetura. Por exemplo, para a satisfação do nível de confiabilidade em uma rede privada virtual assume-se um nível de encriptação de 32 bits, o que pode representar um risco para a arquitetura.

Um ponto de compromisso (*tradeoff point*) é uma propriedade que pode alterar a satisfação de mais de um atributo e representa um ponto de sensibilidade para esses atributos. Por exemplo, a mudança no nível de encriptação pode ter impacto tanto na segurança de acesso como no desempenho: o aumento do nível de encriptação melhora o predicado de segurança de acesso, porém requer mais tempo de processamento; se o processamento de uma mensagem confidencial importante tem um requisito de latência de tempo real reduzido, então o nível de encriptação pode ser um ponto de compromisso (eventualmente conflitante), entre os requisitos de segurança de acesso e latência de tempo real. A análise dos pontos de compromisso é a atividade mais crítica a ser considerada no processo de definição da arquitetura do sistema (KAZMAN, 2000).

Conforme a figura 2.4, o ATAM divide-se em quatro fases: identificação de cenários e elicitação de requisitos (Fase I), descrição das possíveis arquiteturas e compreensão dos cenários (Fase II), implementação e análise dos modelos de arquitetura (Fase III) e análise dos compromissos entre requisitos (Fase IV).

A implementação de um ciclo desse método consiste de um conjunto de reuniões (tipicamente com duração total de três dias), envolvendo o grupo de avaliação da arquitetura, o grupo de desenvolvimento (arquitetos) e os *stakeholders*.

Nessas reuniões realiza-se a seguinte seqüência de atividades (KAZMAN, 2000):

1. Apresentação do ATAM: o grupo de avaliação apresenta, ao conjunto dos *stakeholders*, os passos do ATAM, as técnicas utilizadas e as saídas do processo (essa atividade só é realizada na primeira iteração do ciclo de desenvolvimento para compreensão do método pelos envolvidos);
2. Apresentação das motivações do negócio: o gerente do projeto descreve os objetivos do negócio que estão motivando o esforço do desenvolvimento, apresentando o conjunto de requisitos funcionais mais importantes, os objetivos de qualidade que devem orientar a definição da arquitetura, juntamente com as limitações impostas ao sistema (técnicas, gerenciais, econômicas) e o contexto do sistema; também são identificados os cenários básicos que devem operacionalizar os RNF, visando facilitar a comunicação entre *stakeholders* e criar uma visão comum das atividades importantes que o sistema deve executar;
3. Apresentação da arquitetura: o arquiteto apresenta a arquitetura proposta, no nível de detalhe disponível, focalizando como ela atende às restrições do negócio. Essa apresentação deve incluir: as restrições técnicas dos componentes previstos (sistema operacional, hardware, *Commercial Off-The-Shelf* - COTS), os outros sistemas que interagem com a arquitetura e os princípios de projeto de engenharia de uma arquitetura inicial (ou um conjunto de arquiteturas candidatas); essa arquitetura deve ser descrita em termos de elementos e propriedades que sejam relevantes para cada um dos atributos de qualidade importantes. Se o nível de detalhe permitir devem ser apresentados os mais importantes cenários previstos;
4. Identificação dos estilos de arquitetura: o arquiteto apresenta os estilos de arquitetura considerados. O grupo de avaliação realiza a identificação e catalogação dos padrões de arquitetura e dos estilos que são apresentados, mas não os analisa. Essa atividade objetiva identificar as características da arquitetura que asseguram que os requisitos mais críticos podem ser atendidos de um modo previsível;

5. Geração da árvores dos atributos de qualidade: o grupo de avaliação elicita, prioriza e refina os atributos de qualidade mais importantes no formato de uma árvore, chegando até o nível dos cenários. A geração dessa árvore de atributos permite focalizar a atenção dos *stakeholders* na compreensão dos cenários com relação aos aspectos da arquitetura que são mais críticos para o sucesso do sistema. Como resultado dessa etapa, tem-se a priorização dos requisitos de qualidade e o levantamento dos cenários de avaliação dos mesmos;
6. Análise dos estilos de arquitetura: o grupo de avaliação examina os estilos de arquitetura (apresentados no passo 4) à luz dos atributos de qualidade para identificar riscos, pontos de sensibilidade e pontos de compromisso (que emergem nesta atividade); essa análise é baseada nos cenários de maior prioridade identificados no passo 5. Nessa etapa elicita-se um conjunto de informações sobre cada estilo de arquitetura: seus mais importantes aspectos, suas deficiências, seus pontos de sensibilidade, suas interações e conflitos com outros estilos de arquiteturas. Ao final do processo, esse material é utilizado para fornecer uma descrição dos riscos associados, cujo objetivo é permitir uma avaliação inicial de que a instanciação desse estilo de arquitetura no sistema viabiliza o atendimento dos requisitos de qualidade mais prioritários, com baixo risco;
7. Tempestade cerebral (*brainstorming*) e priorização de cenários: o grupo completo de *stakeholders* elicita um conjunto de cenários, que atende às suas necessidades específicas. Esses cenários podem cobrir as situações antecipadas de uso do sistema (cenários de casos de uso), as situações de modificações futuras previstas para o sistema (cenários de crescimento) e as situações limítrofes, não previstas para o sistema atual, mas que podem potencialmente ocorrer no futuro (cenários exploratórios). Uma vez que o conjunto dos cenários é identificado, ele deve ser priorizado por um processo de votação, permitindo a compreensão da importância relativa

entre estes. Finalmente comparam-se os resultados da priorização de cenários com os resultados gerados na árvore de atributos de qualidade do passo 5. Qualquer diferença entre os cenários mais prioritários de ambas as classificações deve ser reconciliada ou ao menos explicada, permitindo verificar que todas as necessidades dos *stakeholders* estão bem compreendidas e não apresentam conflitos evidentes. Após essa atividade a árvore de atributos de qualidade deve representar em detalhe todos os requisitos de mais alta prioridade oriundos da discussão com todos os *stakeholders*;

8. Análise dos estilos de arquitetura: esse passo repete o passo 6, porém a priorização dos cenários estabelecida no passo 7 é considerada na análise dos modelos de arquitetura obtidos no passo 4. Esses casos de teste podem descobrir novos riscos, pontos de sensibilidade e pontos de compromisso, registrando-se os estilos e os componentes dentro dos estilos que são alterados através de cada cenário. Se necessário, em função dos novos riscos ou de pontos de sensibilidade descobertos, pode-se retornar aos passos 4, 5 e 6 até que não haja nenhuma nova informação a ser analisada. Nessa atividade determina-se a sensibilidade individual de cada atributo em relação aos elementos da arquitetura;
9. Apresentação dos resultados: o grupo de avaliação recapitula a execução dos passos do ATAM, registra os resultados (os estilos, os cenários e sua priorização, as questões específicas de atributos, a árvore de atributos de qualidade, os riscos, os pontos de sensibilidade e os compromissos identificados), apresenta esses resultados ao grupo completo dos *stakeholders* e elabora um relatório com essa informação, incluindo alguma estratégia de mitigação de conflitos.

Após terem sido completados os passos anteriores, os resultados das análises de requisitos são comparados. Quando a análise mostrar que os requisitos são atendidos com a arquitetura proposta os projetistas podem prosseguir o desenvolvimento do

sistema; entretanto, na prática, é usual continuar a monitorar a arquitetura com modelos analíticos durante as etapas de desenvolvimento, instalação e manutenção. Se a análise revelar que existe um problema no atendimento dos requisitos então deve-se iniciar um novo ciclo, onde podem ser alterados os cenários, os requisitos, a arquitetura e os modelos. Havendo a identificação de outros riscos e pontos de compromisso caracteriza-se a necessidade de uma nova iteração no método, até que os compromissos sejam adequadamente solucionados (KAZMAN, 2000).

O presente trabalho está focado na elicitação, na análise e nas potenciais interações entre os requisitos de segurança e usabilidade sobre um modelo de arquitetura para sistemas distribuídos e abertos; assim, utilizam-se os conceitos básicos do método ATAM, porém considerando-se as visões de arquitetura características do modelo ODP em substituição aos estilos de arquitetura descritos no passo 4 do método (ALLEMAN, 2000).

2.5 Técnicas para Modelagem de Requisitos

Um dos objetivos deste trabalho é sistematizar não só a identificação dos requisitos de usabilidade e segurança, mas também permitir uma racionalização do seu processo de verificação durante o desenvolvimento do sistema, uma vez que é conhecido o problema de interdependência entre RNF (a adição ou modificação da especificação de um RNF força decisões que podem afetar, positiva ou negativamente, outros RNF). Para essa racionalização, podem ser utilizados modelos que permitem a identificação das interdependências entre os requisitos e a associação destes ao detalhamento da arquitetura do sistema (LIU, 2001).

Nesta seção discutem-se as técnicas consideradas relevantes para a modelagem de RNF, que se baseiam em linguagem natural complementada por diagramas de representação (como diagramas de classes, seqüência e colaboração).

A opção do uso da linguagem natural como forma de representação dos requisitos é justificada porque, geralmente, esta é a única notação que é

compreendida por todos os envolvidos com o sistema. Este tipo de representação tem como ponto negativo o risco de eventuais ambigüidades, que podem ser minimizadas com o uso de heurísticas ou de modelos complementares de representação (CYSNEIROS, 2001).

Os modelos conceituais tradicionais, utilizados na engenharia de requisitos, podem ser subdivididos em dois grupos (em função do enfoque do relacionamento do mundo real com o ambiente do sistema): os orientados a objetivos e os orientados a cenários.

Neste trabalho foram selecionadas técnicas de modelagem de objetivos e de cenários que permitem tanto uma representação em linguagem natural (para facilitar a comunicação com os *stakeholders*), como uma representação gráfica e formal (para permitir a sistematização e eventual reuso dos resultados). As técnicas que atendem a esses critérios são:

- GRL (*Goal-oriented Requirement Language*): que permite a descrição dos objetivos do empreendimento, de alternativas de atendimento para os requisitos, de hipóteses assumidas e de requisitos não funcionais no nível da empresa (AMYOT, 2003);
- UCM (*Use Case Maps*): que permite a representação dos requisitos funcionais do sistema, através da análise de cenários causais (podendo ser facilmente associada com a arquitetura do sistema) (DE BRUIN, 2001).

Adicionalmente, a ITU (*International Telecommunications Union*) está padronizando tanto a representação da GRL como da UCM em uma linguagem de representação formal denominada URN (*User Requirement Notation*), que é utilizada nos exemplos apresentados (AMYOT, 2002), (ITU-T-Z.150, 2003).

2.5.1 Linguagem de requisitos orientada a objetivos (GRL)

A GRL é uma linguagem que fornece suporte à modelagem orientada a objetivos e permite a representação de requisitos, especialmente RNF. Ela oferece

elementos de construção gráfica para representação para vários conceitos que são utilizados na análise de requisitos e na análise de alto nível da arquitetura do sistema (LIU, 2001).

Existem quatro categorias de conceitos envolvidos nessa linguagem: atores, elementos intencionais, relacionamentos e elementos não intencionais. Os atores são os elementos que contém as intenções, sendo as entidades ativas do ambiente, que necessitam que os objetivos sejam alcançados, as tarefas sejam executadas, os recursos estejam disponíveis e os objetivos difusos sejam satisfeitos. Os elementos intencionais são objetivos (*goals*), tarefas (*tasks*), objetivos difusos (*softgoals*), recursos e hipóteses assumidas (*beliefs*). Os elementos intencionais são utilizados para responder a questões do tipo: quais comportamentos particulares, de determinados aspectos da informação ou da estrutura, foram incluídos nos requisitos do sistema; quais alternativas foram consideradas; quais critérios foram utilizados para a escolha de opções alternativas; e quais as razões que levaram a escolha de uma alternativa no lugar da outra. Os relacionamentos são utilizados para conectar elementos isolados do modelo de requisitos, representando diferentes tipos de relações intencionais, tais como: associação, decomposição, contribuição, dependência, correlação. Os elementos não intencionais são mecanismos para referenciar objetos fora do modelo GRL (AMYOT, 2003).

É importante observar que, devido à sua própria natureza, um requisito não funcional (geralmente caracterizado como objetivo difuso) raramente pode ser considerado totalmente satisfeito. Na literatura inglesa é utilizado o termo *satisfice* ao invés de *satisfy* para simbolizar essa satisfação aceitável. Neste trabalho, quando utiliza-se o termo satisfeito para um RNF, este deve ser entendido como satisfeito dentro de limites razoáveis de utilização.

Um grafo GRL é uma representação formal dos RNF do sistema, utilizando a notação URN. Para complementar a descrição, essa notação pode utilizar a linguagem natural ou formal, baseada na linguagem XML (*W3C's eXtensible Markup Language*) (ITU-T-Z.150, 2003).

2.5.2 Mapas de casos de uso (UCM)

A técnica de mapas de casos de uso - UCM é utilizada para a representação visual dos aspectos comportamentais do sistema num alto nível de abstração. Essa técnica é baseada em cenários mostrando o comportamento do sistema através de relações de causa e efeito, quando se percorrem os caminhos de cada cenário. A representação de múltiplos cenários em um diagrama, juntamente com a interação entre eles, facilita a compreensão do comportamento dos aspectos gerais do sistema pelos *stakeholders* em uma fase inicial de projeto (devido ao elevado nível de abstração nesses diagramas não são representados os detalhes do projeto) (DE BRUIN, 2001).

Os diagramas UCM (denominados de hiper-grafos) fornecem uma visão integrada do comportamento e da estrutura do sistema, através da superposição de caminhos de cenários numa estrutura de componentes abstratos. Essa combinação de comportamento e estrutura permite associar aspectos da arquitetura do sistema com os objetos da representação UCM. Assim, o processo de detalhamento da arquitetura permite o refinamento da representação UCM em modelos mais detalhados (como por exemplo, em diagramas de interação na representação UML - *Unified Modeling Language*). Esses modelos detalhados podem ser transformados em diagramas de estado (em UML) e finalmente podem gerar implementações concretas (com a eventual geração automática de código). Além disso, esses modelos possibilitam a verificação, validação, análise de desempenho e geração de testes em vários estágios do desenvolvimento do sistema. Dessa forma, a técnica UCM permite a transição entre o universo informal e formal, podendo cobrir o hiato normalmente existente entre os requisitos (aqui representados pelos casos de uso) e o projeto do sistema de uma forma explícita e visual (DE BRUIN, 2001).

A linguagem de especificação utilizada na notação UCM tem, como objetivo, atender não somente às necessidades dos responsáveis pela especificação e desenvolvimento do sistema (especialistas), mas também permitir que o sistema seja entendido por não especialistas (usuários, administradores, gerentes) devido à

natureza visual, simples e intuitiva da representação gráfica utilizada. Para atingir esses objetivos, a especificação em UCM identifica as fontes de informação de entrada, suas saídas, descreve as necessidades (em termos de entrada e saída) de cada cenário e integra os cenários (ou os casos de uso relacionados) em um diagrama tipo mapa. Nesses mapas, os cenários são estruturados e integrados incrementalmente, permitindo a detecção de interações potencialmente indesejáveis entre cenários e componentes. Os vários cenários podem ser integrados já que cada cenário complexo pode ser representado como um objeto UCM simples (um sub-mapa) dentro de um novo diagrama UCM (LIU, 2001).

A notação UCM é baseada numa estrutura gráfica, denominada hiper-grafo, onde são especificados todos os elementos que compõem os caminhos e interconexões relativos aos cenários causais (mapas). Os elementos básicos do hiper-grafo são os pontos de partida, pontos de chegada, responsabilidades, componentes e caminhos. Os pontos de partida são círculos preenchidos representando as pré condições ou causas de disparo da atividade. Os pontos de chegada são barras representando as pós-condições ou efeitos finais. As responsabilidades são cruzes (em formato de **X**) representando ações, tarefas ou funções a serem realizadas. Os componentes são retângulos representando entidades ou objetos que compõem o sistema. Os caminhos são linhas irregulares que conectam pontos de partida, responsabilidades e pontos de chegada. Uma responsabilidade é caracterizada como ligada a um componente quando a cruz está dentro do componente, sendo que neste caso o componente é responsável por executar a ação, tarefa ou função representada pela mesma.

Assim como a GRL, um hiper-grafo UCM também pode ser complementado utilizando-se uma linguagem natural ou formal, como por exemplo a linguagem XML (*W3C's eXtensible Markup Language*) (ITU-T-Z.150, 2003).

2.5.3 Integração entre GRL e UCM

Apesar da UCM poder representar formalmente as alternativas de projeto da arquitetura do sistema num alto nível de abstração, não representa explicitamente as vantagens e desvantagens entre essas alternativas (ou mesmo eventuais compromissos), bem como não representa as características que levaram a uma específica decisão de projeto. Além disso, como todas as técnicas baseadas em cenários, os modelos UCM representam apenas parcialmente o sistema (não há como garantir que todos os cenários ou casos de uso foram considerados).

Por outro lado, o modelo GRL representa explicitamente as alternativas e os potenciais compromissos entre RNF. A maioria dos RNF está relacionada com alternativas de arquitetura, que evoluem conforme a arquitetura é detalhada. Assim, a modelagem GRL pode complementar os modelos UCM, desde que sejam estabelecidas as relações entre os elementos intencionais GRL e os componentes funcionais e as responsabilidades nos modelos de cenário UCM. Modelando-se objetivos e cenários de modo complementar pode-se incrementar tanto a identificação de objetivos como a identificação cenários adicionais (e segmentos de cenários) relevantes para a arquitetura do sistema, contribuindo assim para a completeza e precisão da eliciação de requisitos, bem como para a qualidade do projeto da arquitetura (ROLLAND, 1999); (LIU, 2001); (AMYOT, 2003).

A figura 2.5 ilustra o processo integrado proposto por Liu e Yu (2001), onde inicialmente elaboram-se os modelos GRL, através dos quais os objetivos do negócio e os RNF são refinados e operacionalizados (análise do nível de satisfação dos requisitos em relação aos objetivos do negócio). A análise dos requisitos gera tarefas (decisões de projeto) que são modeladas, gerando cenários UCM. Nessa primeira modelagem de cenários não há a preocupação de localizar onde é realizada cada função, sendo apenas modelado como os cenários devem ser satisfeitos, ou seja, não são interligadas às responsabilidades do modelo UCM com os componentes que as deverão executar (essa alocação ocorrerá naturalmente com o refinamento da arquitetura) (LIU, 2001).

Simultaneamente, os cenários UCM são utilizados para descrever as características comportamentais e a arquitetura do sistema. Nesse processo, as responsabilidades são atribuídas aos componentes da arquitetura e as alternativas de arquitetura podem ser explicitadas, podendo surgir novos RNF (derivados das opções de arquitetura analisadas ou simplesmente do melhor conhecimento do sistema).

O refinamento do modelo UCM contribui para a eliciação de novos requisitos que, por sua vez, podem implicar na necessidade de novos cenários e assim sucessivamente. A combinação dos modelos GRL e UCM auxilia a eliciação, o refinamento e a operacionalização dos requisitos de modo incremental, até que sejam geradas especificações satisfatórias, tanto para os requisitos como para a arquitetura do sistema.

Figura 2.5: Integração dos modelos GRL e UCM (LIU, 2001)

O modelo GRL e os hiper-grafos do modelo UCM podem ser representados textualmente, tanto em linguagem natural, como utilizado a linguagem XML. A notação gráfica semi-formal pode ser a mais adequada durante as etapas iniciais de projeto da arquitetura, para permitir a exploração e a avaliação de alternativas de projeto. Com o decorrer do desenvolvimento a notação formal pode ser introduzida para permitir um maior controle e rastreabilidade no processo (ITU-T-Z.150, 2003).

O processo integrado dos modelos GRL e UCM é adotado neste trabalho como referência para a metodologia de especificação de requisitos, porque esse enfoque atende a várias necessidades da engenharia de requisitos, entre elas: a necessidade de técnicas para formalmente modelar e analisar as propriedades dos RNF; a diminuição da distância entre a elicitação de requisitos, baseada em enfoques contextuais, e técnicas formais de especificação e análise; a utilização de modelos mais detalhados para a captura e análise de RNF; um melhor entendimento do impacto das escolhas de arquitetura de software na priorização e evolução dos requisitos; e a possibilidade de reuso de modelos de requisitos (NUSEIBEH, 2000).

Exemplos de aplicação desse processo integrado podem ser encontrados em Liu e Yu (2001) e De Bruin e Van Vliet (2001) (LIU, 2001); (DE BRUIN, 2001).

3 MODELO DE SISTEMAS ABERTOS COM ARQUITETURA DE OBJETOS DISTRIBUÍDOS

Um processamento distribuído é caracterizado quando este é realizado por entidades de software e hardware que estão fisicamente separadas. Um sistema de processamento distribuído é composto de componentes de software separados que são interconectados com o propósito de compartilhar informação e processamento. Se esses componentes exibem características de abertura (ou seja, se estes são padronizados e permitem que outros sistemas interajam com os mesmos sem a necessidade de modificações na sua infra-estrutura), então a interação entre eles pode ser caracterizada como um processamento distribuído e aberto (FAROOQUI, 1995).

Um sistema (de processamento) distribuído e aberto ideal deve permitir que diferentes aplicações distribuídas, desenvolvidas em diferentes ambientes, possam interagir. Para que esse objetivo possa ser alcançado, é necessário que esses ambientes tenham um modelo conceitual (ou arquitetura) comum. Nesse sentido, dois organismos internacionais de padronização, a ISO/IEC (*International Organization for Standardization / International Electrotechnical Commission*) e a ITU (*International Telecommunication Union*), uniram-se para padronizar um modelo de referência para o processamento distribuído e aberto (*Reference Model of Open Distributed Processing - RM-ODP*), que atualmente é considerado o mais completo e atualizado padrão para um modelo de arquitetura para sistemas distribuídos e abertos (ISO-10746-1, 1995).

O modelo ODP apresenta um conjunto de conceitos padronizados (*framework*) que permite definir uma arquitetura de sistema que tenha integrado aos mesmos recursos de suporte a distribuição, inter-relacionamento, interoperabilidade e portabilidade para aplicações distribuídas.

A metodologia de projeto, definida no modelo ODP, estabelece que o sistema distribuído deve ser descrito em cinco níveis de abstração, denominados pontos de vista (*viewpoints*).

A idéia da utilização de pontos de vista provém do fato de que, em qualquer sistema de processamento de informação, existe um certo número de categorias de usuários (cada um desempenhando seu papel) que tem interesse nesse sistema (*stakeholders*).

Para tratar dessa diversidade de interesses, o modelo ODP preconiza a utilização de cinco pontos de vista (ou projeções), cada um refletindo um conjunto de restrições (que poderão se tornar requisitos) que o sistema deve atender, permitindo assim a modelagem do sistema em cinco visões que se complementam, sem a necessidade da criação de um grande modelo global (FAROOQUI, 1995).

O modelo ODP prevê os seguintes cinco pontos de vista (figura 3.1):

Figura 3.1: Pontos de vista do modelo ODP (PUTMAN, 2001)

- Ponto de vista da empresa: descreve as necessidades dos usuários do sistema; deve trazer a visão dos objetivos da organização na qual o sistema será utilizado, focalizando os requisitos da organização e a função do sistema dentro do empreendimento;

- Ponto de vista da informação: representa, em termos de objetos abstratos, os elementos de informação (estruturas de dados, restrições, fluxo de dados) manipulados no empreendimento;
- Ponto de vista da computação: representa a visão dos projetistas e programadores do sistema; nele são representados o particionamento lógico dos componentes (objetos de computação) distribuídos, suas interfaces e interações entre eles;
- Ponto de vista da engenharia: descreve as estruturas de gerenciamento e comunicação, que fornecem suporte para as aplicações distribuídas, incluindo os serviços e os mecanismos genéricos necessários para as aplicações e as interações descritas no modelo computacional;
- Ponto de vista da tecnologia: concentra-se na identificação dos componentes (hardware e software) e objetos do mundo real que compõem o sistema distribuído que será construído; realiza o mapeamento dos modelos de computação e engenharia em tecnologias específicas.

A especificação do modelo ODP em cinco pontos de vista descreve os elementos e a estrutura do sistema distribuído. Para completar a especificação do sistema, existem outros requisitos relacionados a funções de integração e cooperação, através da definição dos conceitos de transparência e funções ODP. Os requisitos associados a esses conceitos são resumidos nas seções seguintes, dando-se ênfase àqueles aspectos relacionados com a confiabilidade e a segurança presentes na arquitetura de sistemas distribuídos proposta por esse modelo.

3.1 Ponto de Vista da Empresa

Esse ponto de vista descreve os objetivos da organização na qual o sistema será utilizado, focalizando os requisitos da organização e a função do sistema distribuído no contexto do empreendimento.

O ponto de vista da empresa é estabelecido a partir das necessidades dos vários usuários do sistema de informação, sendo entendidos como usuários todos os agentes que vão interagir externamente com o sistema durante seu ciclo de vida (por exemplo: operadores do processo, gerentes do empreendimento, diretores da corporação, responsáveis pelo licenciamento da instalação, responsáveis pela manutenção do sistema).

Essa visão é o ponto de vista mais abstrato dentro do modelo ODP, sendo utilizada para especificar os requisitos de alto nível do sistema (seus objetivos gerais), as regras de funcionamento e estruturas organizacionais envolvidas. É importante ressaltar que, apesar de ser um ponto de vista de elevada abstração, a sua especificação pode ser descrita de forma formalizada, através de uma linguagem de representação de empresa (*The Enterprise Language*), definida para o modelo ODP (ISO-15414, 2002); (STEEN, 1999).

Em termos de engenharia de software esse ponto de vista está relacionado com a elicitação dos requisitos do sistema e o projeto inicial do sistema (sua arquitetura do ponto de vista de interfaces com o domínio). As decisões de projeto adotadas com esta visão referem-se ao que o sistema deve fazer (e não à forma como essas funções serão implementadas). Dessa forma, nesta etapa o projetista desenvolve um modelo que representa os requisitos do mundo real que devem ser incorporados, futuramente, na estrutura do sistema (FAROOQUI, 1995).

A especificação do sistema do ponto de vista da empresa deve representar os seguintes aspectos (BECERRA, 1998):

- Escopo da empresa: representa as características do negócio da empresa (seus objetivos internos e externos, a curto e longo prazo dentro do ambiente do negócio);
- Funções empresariais: representam as funções realizadas por objetos dentro da empresa, em função das políticas da mesma (exemplos dessas funções são o presidente, o administrador, o gerente, o vendedor, o fornecedor, etc.);

- Políticas: são as regras definidas interna ou externamente à empresa e que devem ser obedecidas pelas atividades nela realizadas (exemplos: normas, obrigações, restrições);
- Procedimentos: representa os conjuntos de atividades padronizados que estabelecem ações específicas a serem realizadas pelos objetos da empresa (exemplos: procedimentos operacionais, procedimentos da qualidade, procedimentos de verificação e validação);
- Comunidade: representa os grupos de objetos da empresa ou de funções empresariais que realizam suas atividades dentro de políticas definidas e que visam um objetivo comum (um exemplo de uma comunidade dentro de um sistema de informação hospitalar poderia ser representado pelo conjunto das entidades que participam do serviço de radiologia);
- Federação: representa a união de várias especificações de empresas, subsidiárias ou domínios, cuja estrutura deve definir o relacionamento entre esses elementos (um exemplo de federação é o sistema de informação corporativo de uma empresa que possui várias subsidiárias distribuídas).

O modelo ODP fornece ferramentas descritivas para a realização dessa modelagem (ISO-10746-1, 1995).

A modelagem do ponto de vista da empresa visa a obtenção de um diagrama de objetos empresa, onde são representadas as interações destes, segundo o contexto definido pelas regras da empresa. Como ferramenta de modelagem podem ser utilizados os modelos de estrutura estática definidos pela UML (UML, 1997). Alguns exemplos da aplicação dessa linguagem de representação pode-se citar: a utilização de diagrama de classes para a visão de empresa, mostrando suas associações e atributos; a utilização de diagramas de casos de uso (*use cases*) para definição das regras de interação entre os objetos.

3.2 Ponto de Vista da Informação

Esse ponto de vista descreve o conteúdo da informação do empreendimento. É definida a semântica da informação e a semântica do processamento da informação no sistema (estruturas de dados, restrições, fluxo de dados), isto é, descrevem-se quais os tipos de informações que são manipulados, quais são as restrições ao acesso às informações, quais são as restrições das interfaces de comunicação.

O modelo de informação é representado por objetos abstratos que expressam os elementos de informação manipulados pelo sistema. A atividade de modelagem da informação consiste em identificar: as estruturas (ou elementos) de informação do sistema, restrições e transformações que devem ocorrer nessas estruturas e o fluxo dessa informação. Essas informações devem ser completamente independentes da forma como serão implementadas, bem como não devem haver restrições relacionadas com a forma de como essas informações são representadas no sistema real, nem aos meios utilizados para manipulação dessas informações.

Para a representação da semântica da informação são utilizados três tipos de esquemas (BECERRA, 1998):

- Esquema invariante: conjunto de condições ou predicados que são sempre verdadeiros e que se aplicam a um ou a um conjunto de objetos de informação. Essas condições controlam os possíveis estados ou mudanças de estado dos objetos de informação;
- Esquema estático: especifica o estado de um objeto ou conjunto de objetos em um ponto específico dentro do tempo de processamento. Esse esquema está sujeito às restrições do esquema invariante;
- Esquema dinâmico: define os estados de um objeto ou conjunto de objetos de informação dentro do tempo de processamento. Esse esquema é controlado pelo esquema invariante.

A modelagem do ponto de vista da informação apresenta os requisitos do sistema em função da informação armazenada e manipulada pelos objetos empresa e

pelo sistema. Para especificar as regras de manipulação da informação utilizam-se, como ferramentas de apoio, os contratos (documentos de especificação que descrevem como a operação chega ao seu resultado final) (BECERRA, 1998).

Para modelar os esquemas invariantes utilizam-se diagramas de estruturas estáticas definidos pela UML, como os diagramas de classes que podem representar os objetos informação com suas respectivas associações. Na elaboração dos esquemas estáticos e dinâmicos podem ser utilizados os diagramas de colaboração da UML (que podem representar os objetos informação com as informações recebidas e transmitidas entre os objetos) e os diagramas de estado para representar a sequência de estados de um diagrama dinâmico (UML, 1997).

3.3 Ponto de Vista da Computação

Esse ponto de vista descreve o sistema distribuído para os projetistas e programadores. Ele reflete o particionamento lógico da aplicação distribuída, baseado nos tipos de serviços oferecidos pelas partes do sistema e no fluxo de informações entre suas interfaces. Nessa visão surge o importante conceito de componentes da aplicação (objetos de computação) e suas funções dentro de uma interação do tipo cliente-servidor (FAROOQUI, 1995).

A visão da computação retrata a distribuição do sistema em termos de componentes da aplicação e suas interações, independente do ambiente distribuído (sistema operacional, infra-estrutura de comunicação), onde esse sistema será implementado, ou seja, essa visão não considera quais mecanismos serão utilizados para viabilizar a execução da aplicação.

Nessa visão, os elementos processadores de informação são definidos como objetos computação, que são interligados através de interfaces computacionais, estruturando assim um sistema distribuído. Esses objetos encapsulam suas atividades e estados, podendo ser executados em paralelo.

A modelagem computacional de uma aplicação distribuída consiste na estruturação da aplicação nos seus objetos computação, identificação e especificação das interfaces computacionais, identificação da comunicação entre interfaces computacionais no nível da aplicação (denominada de interações computacionais) e identificação das restrições do ambiente relacionadas com essas interações (muitas das quais vem do modelo de empresa).

Os objetos computação são, geralmente, distribuídos; assim, as interações computacionais representam uma abstração da transparência do sistema distribuído em termos de operações no nível da aplicação (denominadas de operações computacionais) ao invés de representarem a troca física de mensagens remotas (FAROOQUI, 1995).

Os objetos computação possuem os seguintes elementos (BECERRA, 1998):

- Interface computacional: definida por uma especificação da interface, onde são identificados as formas de interação, os comportamentos das interface e os contratos de interação externa;
- Especificação do comportamento: definem-se os estados e os serviços executados pelos objetos computação;
- Requisitos externos: requisitos impostos pelo ambiente externo, normalmente definidas em contratos e oriundas do modelo de empresa.

As operações computacionais são utilizadas pelos objetos computação para realizar seus pedidos de serviço. O tipo de pedido de serviço entre objetos pode ser classificado em três formas de interface:

- Interface para operações: é a interface entre objetos do tipo cliente-servidor que fornece recursos para as interações, denominadas de operações (estruturas semelhantes a procedimentos ou chamadas de rotinas). Existem dois tipos de operações: o tipo interrogação, onde o servidor retorna uma resposta ao cliente, e o tipo anúncio, onde não existe resposta ao cliente;

- Interfaces para fluxos: é uma interface entre objetos do tipo produtor e consumidor, sendo suas interações caracterizadas por fluxos contínuos de informação (como fluxo de áudio e vídeo numa aplicação multimídia);
- Interfaces para sinais: é uma interface entre objetos do tipo iniciador e de resposta, sendo suas interações caracterizadas por sinais que representam invocações como eventos e interrupções.

A interação entre dois objetos de computação é realizada entre suas interfaces, que são conectadas através de uma via de comunicação. Essa ligação entre interfaces é denominada de conexão, sendo que no modelo ODP são definidos os seguintes tipos de conexão:

- Conexão explícita: é caracterizada pela existência de fluxo contínuo de informação entre duas interfaces ou pela capacidade de manipulação de operações entre essas interfaces;
- Conexão implícita: é caracterizada pela interfaces para operação, sendo especialmente utilizada em linguagens procedurais, onde não é necessário explicitar a ação de conexão.

Para especificar a visão computacional podem ser utilizadas linguagens de programação, linguagens de definição de interfaces, técnicas de descrição formal. Considerando-se a linguagem UML podem ser utilizados diagramas de classes para representar os objetos computação com suas respectivas interfaces, bem como podem ser utilizados os diagramas de casos de uso (*use cases*) para definição das regras de interação entre os objetos (UML, 1997).

3.4 Ponto de Vista da Engenharia

Esse ponto de vista descreve as estruturas de gerenciamento e comunicação que fornecem recursos para as aplicações distribuídas, incluindo os serviços e os mecanismos genéricos necessários para as aplicações e as interações descritas no modelo computacional.

O modelo de engenharia especifica a configuração concreta da aplicação, a distribuição física dos componentes, os protocolos de comunicação remota, os mecanismos de transparência da distribuição e os serviços de suporte específicos da aplicação distribuída (gerenciamento, controle, acesso). As restrições do ambiente, que são visíveis no modelo de computação, são utilizadas, no modelo de engenharia, para a seleção dos mecanismos necessários para o seu processamento distribuído. Além disso, são identificadas as interfaces específicas para os objetos de computação, de modo a permitir a implementação dos mecanismos de distribuição independente da tecnologia (definida no ponto de vista da tecnologia) (FAROOQUI, 1995).

Os mecanismos visíveis no modelo de engenharia são os recursos de processamento e armazenamento, mecanismos de transparência de distribuição (tipicamente transparência de: acesso, localização, concorrência, migração, replicação, recursos, falha e federação), mecanismos que fornecem suporte (protocolos de comunicação) e outros mecanismos específicos da aplicação.

Nesse ponto de vista, as técnicas de projeto voltadas para dependabilidade são mais evidentes; a verificação dos atributos de segurança pode ser realizada pelas ferramentas tradicionais de análise de segurança.

As noções de processador, memória, sistema operacional têm um papel secundário neste ponto de vista (fornecendo o suporte no nível de sistema); assim, o termo modelo de engenharia deve ser utilizado de uma forma mais específica para descrever um conjunto de recursos (*architectural framework*) orientado para a organização da infra-estrutura do sistema, consistindo de estruturas e mecanismos que viabilizam e controlam a distribuição do sistema (FAROOQUI, 1995).

A descrição da arquitetura do modelo de engenharia é baseada em três elementos principais: o objeto de engenharia, a estrutura gerencial e a estrutura de comunicação.

O objeto básico de engenharia (OBE) é a representação, em tempo de execução, dos objetos computação (obtida através da compilação, interpretação ou

outra forma de transformação dos objetos computação), que encapsula a funcionalidade da aplicação. Um objeto básico de engenharia corresponde a um objeto computação (funcionalmente) enriquecido com interfaces adicionais para interagir com os objetos de um canal de comunicação (FAROOQUI, 1995).

A estrutura de gerenciamento do ponto de vista do modelo de engenharia tem, como objetivo, organizar a infra-estrutura do sistema para otimizar a forma de gerenciamento dos objetos distribuídos. Os elementos que compõem essa estrutura são (BECERRA, 1998):

- Grupo: representa uma configuração de objetos básicos de engenharia relacionados (pertencentes a mesma aplicação), que tem localização específica, e que tem, como propósito, executar funções como ativação, desativação, reativação, recuperação e migração;
- Gerenciador de grupo: é um objeto de engenharia que controla os objetos básicos do seu grupo;
- Cápsula: corresponde a uma configuração de objetos de engenharia de forma a criar uma estrutura simples e compacta com o objetivo de encapsular o processo e o armazenamento;
- Gerenciador de cápsula: é um objeto de engenharia que realiza o gerenciamento dos objetos de engenharia da respectiva cápsula;
- Nó: representa a configuração de objetos de engenharia de forma a compor um processador de informação, onde são executadas as funções de processamento, armazenamento e comunicação;
- Núcleo: é o objeto de engenharia encarregado da coordenação das funções de processamento, armazenamento e comunicação dentro do nó, controlando essas funções para os objetos básicos de engenharia, objetos de transparência e objetos protocolo, colocados juntos nas cápsulas; um núcleo pode conter mais que uma cápsula e pode interagir com outros núcleos.

A estrutura de comunicação distribuída, padronizada no modelo ODP, estabelece os elementos de uma conexão distribuída entre diferentes cápsulas ou nós. Para implementar essa conexão, configura-se uma estrutura denominada canal, cujo objetivo é oferecer, de forma transparente, serviços para comunicação entre objetos de engenharia que estão interagindo.

Os objetos básicos de engenharia estão sempre diretamente ligados a objetos adaptadores (*stubs*). Os objetos adaptadores dentro do canal podem interagir utilizando outros objetos do canal (conectores, protocolos, interceptores) ou através de objetos suporte fora do canal (BECERRA, 1998).

3.5 Ponto de Vista da Tecnologia

Esse ponto de vista descreve a tecnologia selecionada para implementar os mecanismos, funções e estruturas definidos no modelo de engenharia. Neste contexto, são identificados os recursos de hardware e software que serão utilizados na implementação dos objetos de engenharia, podendo ser padronizados, proprietários ou ambos, em função da política da empresa.

Cabe ressaltar que, apesar da especificação dos outros pontos de vista ODP ser independente da tecnologia, as seleções realizadas nesta etapa devem ser analisadas de modo estratégico, uma vez que, dependendo do tipo de aplicação (ou de restrições estratégicas do domínio), as outras visões do modelo ODP podem ser influenciadas pelas decisões adotadas no ponto de vista da tecnologia (BECERRA, 1998).

Além disso, na escolha das tecnologias devem ser consideradas também as características dos sistemas legados, que podem impor restrições à escolha de tecnologia, visando a facilidade de integração do sistema a infra-estrutura existente.

3.6 Serviços de Transparência

Os sistemas distribuídos apresentam uma série de propriedades, inerentes à sua distribuição, que não são encontradas nos sistemas centralizados. Consequentemente, sua aplicação deve levar em consideração essas propriedades, que envolvem uma maior complexidade no tratamento das interações entre os objetos do sistema distribuído.

O conceito de transparência no modelo ODP está relacionado com a noção de abstração, onde os detalhes irrelevantes da aplicação são ignorados. Os serviços de transparência de distribuição são mecanismos que visam mascarar, dos sistemas e dos usuários, a complexidade dos serviços oferecidos pela estrutura distribuída, facilitando a execução de tarefas cooperativas entre os elementos desse sistema (BECERRA, 1998).

O modelo ODP define os seguintes serviços de transparência de distribuição (ISO-10746-1, 1995):

- Transparência de acesso: são os mecanismos que manipulam (mascaram) as diferenças relativas às representações dos dados e às formas de invocação, quando da interação entre dois objetos; esses serviços mascaram a diferença entre serviços locais e remotos, bem como permitem a interoperabilidade entre computadores de diferentes arquiteturas, sistemas operacionais e linguagens de programação;
- Transparência de falha: são os mecanismos de tolerância à falha, que mascaram certas falhas (de objetos ou de comunicação) e possuem mecanismos de recuperação dos objetos servidores a partir dos objetos clientes;
- Transparência de localização: refere-se ao uso da informação de localização no espaço de um objeto do sistema, mascarando para o usuário (cliente) a localização estática do objeto (servidor) sendo acessado;

- Transparência de migração: são os mecanismos que permitem a mudança de localização de um objeto dentro do sistema, mesmo durante a interação entre dois objetos, sem a perda da integridade dessa interação; esses serviços mascaram para o usuário (objeto cliente) a localização dinâmica do objeto com o qual ele está interagindo (objeto servidor);
- Transparência de persistência: são os mecanismos que permitem a ativação e desativação dos objetos nos processos distribuídos, segundo a demanda de processamento; esses serviços mascaram a transferência automática de grupos de uma cápsula para um objeto de armazenamento e vice-versa, para otimizar a utilização de recursos de um núcleo (memória, processador, etc.); com esse tipo de mecanismos, os clientes podem invocar operações nos servidores independentemente se o servidor está ativo ou desativo;
- Transparência de replicação: são os mecanismos que permitem a manipulação de objetos idênticos como se fossem um único objeto, mantendo a consistência dos dados entre as várias cópias dos objetos, sendo também conhecida como transparência de grupo; esses serviços mascaram, dos usuários, as diferenças entre um provedor de serviços replicado e não replicado, bem como mascaram, dos provedores de serviço, se os usuários são replicados ou não replicados;
- Transparência de realocização: são os mecanismos que manipulam o restabelecimento da conexão de uma interface que realizou uma migração;
- Transparência de transacional: são os mecanismos que coordenam as transações (atividades de configuração de um objeto com o objetivo de manter a consistência dos seus estados), podendo mascarar efeitos de operações entre múltiplas fronteiras administrativas entre os usuários (transparência de federação), permitindo a interoperabilidade entre diferentes domínios administrativos e tecnológicos.

3.7 Funções ODP

A implementação dos serviços de transparência descritos na seção anterior é realizada utilizando-se objetos computacionais, denominados de funções ODP.

O modelo ODP apresenta quatro categorias de funções: funções de gerenciamento, funções de coordenação, funções de repositórios e funções de segurança de acesso, que são descritas sucintamente nas seções seguintes (BECERRA, 1998); (ISO-10746-1, 1995).

3.7.1 Funções de gerenciamento

Essas funções correspondem ao controle da estrutura de gerenciamento descrita no modelo de engenharia (seção 3.4), sendo compostas dos seguintes serviços (BECERRA, 1998):

- Funções de gerenciamento de nó: trata do controle das funções de processamento, armazenamento e comunicação implantadas no nó, compreendendo o gerenciamento das linhas de controle dos processos, a criação de canais de engenharia, a localização de interfaces entre objetos, a criação e a desativação de cápsulas e o gerenciamento da sincronização dentro do nó;
- Funções de gerenciamento de cápsula: corresponde ao gerenciamento da criação e da desativação dos grupos dentro da cápsula e da desativação de todos os elementos da cápsula;
- Funções de gerenciamento de grupo: trata do gerenciamento do armazenamento da configuração do grupo, da recuperação do grupo, da reativação do grupo, da migração do grupo e da desativação dos elementos dentro do grupo;
- Funções de gerenciamento do objeto: constitui-se no gerenciamento da configuração do objeto e da sua desativação.

3.7.2 Funções de coordenação

Essas funções implementam os serviços de coordenação das atividades dos objetos de engenharia, oferecendo suporte às funções de gerenciamento, sendo compostas dos seguintes serviços (BECERRA, 1998):

- Função de notificação de eventos: serviço responsável pela estruturação do histórico dos eventos dos objetos;
- Função de controle da configuração: serviço que trata da estruturação da configuração dos grupos, realizando o armazenamento dos estados dos objetos e das suas interfaces;
- Função de recuperação: serviço responsável pelos mecanismos de recuperação, quando ocorre uma falha no grupo;
- Função de desativação e reativação: serviço que executa os mecanismos de desativação e de reativação dos grupos;
- Função de migração: serviço que executa os mecanismos que coordenam a migração dos grupos entre cápsulas diferentes;
- Função de replicação: serviço que executa os mecanismos que permitem a manipulação de um grupo de objetos, cujos comportamentos são idênticos;
- Função transacional: serviço que executa os mecanismos que asseguram a consistência das transações entre objetos.

3.7.3 Funções de repositórios

As funções de repositórios realizam a manipulação das informações referentes às interfaces, objetos e outros elementos do sistema. Os serviços deste grupo são (BECERRA, 1998):

- Função de armazenamento: esse serviço possibilita o armazenamento de informações em repositórios de dados;
- Função de organização da informação: esse serviço permite o gerenciamento da organização da informação, conforme definido nos esquemas do ponto de vista da informação;
- Função de realocização e tipos: esse serviço realiza o gerenciamento da informação referente à localização das interfaces e aos tipos de especificações;
- Função de localização de serviços: esse serviço gerencia o repositório de serviços que são oferecidos aos elementos do sistema, podendo os serviços estarem em domínios diferentes.

3.7.4 Funções de segurança de acesso

Esse grupo de funções é de especial interesse neste trabalho, uma vez que nele se encontram os serviços mais diretamente relacionados com as características de segurança do sistema, mais especificamente relacionadas com os requisitos de segurança de acesso não autorizado ao sistema (*security*).

As funções de segurança de acesso do modelo ODP oferecem os serviços que permitem implantar diferentes níveis de segurança de acesso relacionados à interação (comunicação) entre os objetos distribuídos no sistema. Os serviços que fazem parte deste grupo são (BECERRA, 1998):

- Função de controle de acesso: controla o acesso de elementos não autorizados (intrusos) aos objetos do domínio;
- Função de auditoria: realiza a monitoração, a análise e a coleta de informações dos eventos relacionados com a segurança de acesso na comunicação entre os objetos da aplicação;

- Função de autenticação: valida a identidade dos objetos, confirmando a configuração do objeto dentro de um domínio específico;
- Função de integridade: detecta e impede a manipulação não autorizada das informações dos objetos do sistema;
- Função de confidencialidade: impede a divulgação de informação não autorizada do sistema;
- Função de gerenciamento de senhas: realiza o controle de senhas, por exemplo, utilizando recursos de criptografia.

3.8 Conceitos Básicos e Notação

Esta seção descreve os conceitos básicos e as regras da linguagem para a descrição dos pontos de vista do modelo ODP, conforme recomendações da ISO/IEC15414:2002 (ISO-10746-2, 1995); (ISO-10746-3, 1995); (ISO-15414, 2002).

As tabelas 3.1, 3.2 e 3.3 apresentam os conceitos básicos, as regras de especificação e os elementos básicos da linguagem de descrição de empresa (*The Enterprise Language*) definidos para o modelo ODP (ISO-10746-2, 1995); (ISO-15414, 2002).

É importante observar que o modelo ODP fornece um conjunto de conceitos e regras de estruturação, bem como define de uma linguagem para a descrição do modelamento. Entretanto, esse modelo não estabelece um método específico de desenvolvimento do sistema ou uma notação para representar esses conceitos.

Atualmente, ainda não existem ferramentas de representação que suportem integralmente o modelo ODP. A ferramenta de representação mais utilizada para representar os conceitos do modelo ODP é a UML (*Unified Modeling Language*). Apesar de existirem conceitos do ODP, que não são representados em UML, esta ferramenta pode ser utilizada se o desenvolvedor do sistema contornar certas limitações. As principais limitações são: ausência de distinção entre tipo e classe, as

interfaces não podem ser diretamente instanciadas, um objeto não pode ter múltiplos tipos, uma interface não pode ter múltiplos tipos, etc.. A alternativa para contornar essas limitações é replicar conceitos utilizando a representação UML (por exemplo, as interfaces são especificadas como classes, as restrições são representadas como classes de associação, um objeto com múltiplos tipos é representado como múltiplas classes), permitindo uma consistência entre a representação UML e o modelo ODP (UML, 1997); (OMG, 1999); (PUTMAN, 2001); (ISO-15414, 2002).

Tabela 3.1: Conceitos básicos do modelo ODP (ISO-10746-2, 1995)

Conceito	Descrição
Conceitos Básicos do Modelo ODP	
entidade	É qualquer elemento abstrato ou concreto.
objeto	É o modelo de uma entidade. É definido pelo seu estado e comportamento. Ele muda de estado como resultado de ação externa ou interna. O estado de um objeto é único. Não é uma instância concreta de uma classe como no caso de orientação a objetos.
estado (de um objeto)	É a condição de um objeto que determina seu comportamento: o conjunto de ações restritivas das quais o objeto pode participar.
classe	É o conjunto (eventualmente nulo) de todos os membros que satisfazem um tipo específico, sendo que neste caso cada entidade é membro dessa classe.
tipo	É o conceito de uma propriedade de uma ou mais entidades. O tipo aplica-se ao menos a um objeto, interface e ação. Tipo é um predicado: um conjunto de condições e restrições que podem classificar um objeto e são válidas para um objeto.
padrão	É uma especificação que contém detalhes suficientes de uma entidade para instanciá-la.
ação	É o que uma entidade executa. É uma ocorrência ou acontecimento.
interação	É uma ação entre dois ou mais objetos ou comunidades e seu ambiente na interface; conjunto de serviços que são oferecidos através de uma interface simples e são conectados a outros objetos com uma ligação (<i>binding</i>).
interface	É um comportamento de um objeto, em relação a um subconjunto de interações do objeto, limitado pelas circunstâncias de quando elas ocorrem. Uma interação é associada com um a interface simples.
comunicação	É a forma de como a informação é transferida de um objeto a outro
locação	É a especificação de um intervalo, de tamanho arbitrário, no tempo ou no espaço quando ou onde uma ação pode ocorrer.
ambiente (de um objeto)	São todas as entidades que não fazem parte do modelo do objeto, orientando a especificação do que um objeto espera de seu ambiente e quais restrições que o ambiente impõe ao objeto.

Tabela 3.2: Regras de especificação do modelo ODP (ISO-10746-2, 1995)

Conceito	Descrição
Regras de Especificação do Modelo ODP	
abstração	É um processo onde uma entidade é especificada em alto nível, suprimindo-se os detalhes da mesma para um posterior refinamento.
composição	É um processo de agrupamento de dois ou mais objetos (resulta num objeto composto). O objeto que faz parte da composição é chamado de componente.
compatibilidade comportamental	Significa que um objeto pode substituir outro objeto porque ambos têm o mesmo comportamento.
refinamento	É um processo de detalhar a especificação de uma entidade, que se encontra em um certo nível de abstração, para um nível de maior granulosidade.
rastreabilidade	É o processo de registrar o histórico das interações de um objeto.
instanciação	É o processo de criação de uma instância de um padrão específico. Um objeto é instanciado por padrão de objeto para o seu estado inicial.
papel	Identifica o comportamento de uma entidade.
criação	É um processo similar a instanciação, porém como resultado de uma ação.
introdução	É um processo similar a instanciação, porém não é resultado de ação de objeto.
remoção	É um processo remover ou destruir algo que foi instanciado, como um objeto ou interface.
instância	Em ODP é o elemento <X> que satisfaz a um tipo.
tipo padrão	É um predicado no padrão que define os requisitos para instanciação do padrão.
invariante	É um predicado que deve ser verdadeiro para um conjunto de objetos ao mesmo tempo.
pré-condição	É um predicado que deve ser verdadeiro antes do início de uma ação e deve resultar em sucesso para que essa ação ocorra.
pós-condição	É um predicado que deve ser verdadeiro e ocorre após a ação.
Regras de Estruturação do Modelo ODP	
comunidade	É um conjunto de objetos que interagem com o propósito de atender a um objetivo e que tem um contrato definindo como o objetivo pode ser atingido.
grupo	É um conjunto de objetos relacionados por uma estrutura específica ou por um comportamento específico.
domínio	É um conjunto de objetos relacionados por uma estrutura específica ou por um comportamento específico e que possui um objeto de controle.
época	É um período de tempo definido.
ponto de referência	É um ponto em que uma interação pode ser usada como ponto de teste de conformidade para observação de determinado comportamento.
comportamento	É um conjunto de ações e indicações de época para essas ações (que são restrições). As restrições definem as dependências que existem entre ações.
atividade	É um conjunto ordenado de ações, onde cada ação é determinada pelas ações predecessoras.
contrato	É um acordo entre objetos em relação a um certo comportamento (para restringir esse comportamento).
ligação (binding)	É um contrato entre duas ou mais interfaces de objetos que é resultado de um acordo entre eles.
consistência	É a relação entre os modelos de uma especificação que garante a inexistência de compromissos entre as propriedades dos vários pontos de vista do modelo.
conformidade	É o processo de determinação da consistência por teste ou por verificação de modelo. Os testes de conformidade asseguram que a implementação do sistema corresponde à especificação da arquitetura.

Tabela 3.3: Elementos da linguagem de descrição da empresa (ISO-15414, 2002)

Conceito	Descrição
Conceitos de Sistema	
escopo	É o comportamento esperado do sistema ou da empresa. É expresso pelo conjunto de papéis das entidades do negócio e por seu comportamento associado.
objeto de empresa	Pode modelar um ser humano, uma entidade legal, um sistema de informação, um recurso ou uma coleção de qualquer uma destas entidades.
campo de aplicação	São as propriedades que o ambiente deve ter para que a especificação do sistema possa ser utilizada.
Conceitos de Comunidade	
objetivo	É a vantagem prática ou efeito desejado, expresso como preferencial para estados futuros. Define o propósito do negócio, expresso por papéis, políticas e relacionamento entre os papéis.
objeto comunidade	É um objeto de empresa que representa a comunidade. Componentes de um objeto comunidade são objetos da comunidade representada.
Conceitos de Comportamento	
ator	É um objeto de empresa que participa da ação (inicia ou responde a ações).
artefato (de ação)	É um objeto de empresa que é utilizado na ação e não executa essa ação isoladamente.
recurso	É um objeto de empresa que é essencial para algum comportamento e que requer alocação ou pode estar indisponível.
papel de interface	É um papel (função) de uma comunidade identificando um comportamento que depende da participação de objetos que não são membros da comunidade.
processo	É um conjunto de passos que ocorre de uma maneira predeterminada e possui um objetivo.
passo	É uma abstração de uma ação, usado em um processo, que pode deixar que objetos não especificados participem dessa ação.
Conceitos de Política	
política	É um conjunto de regras destinado a um propósito particular. Uma regra pode expressar obrigação, autorização, permissão ou proibição.
autorização	É a determinação de que um particular comportamento não deve ser impedido.
violação	É uma ação contrária a uma regra.
Conceitos de Responsabilidade	
party	É um objeto de empresa que modela uma pessoa real ou outra entidade que tenha direitos, poderes ou funções de uma pessoa real.
proprietário	É um objeto que modela uma pessoa real que tem autoridade para controlar algo ou um contrato de negócio que impõe os aspectos legais de uma empresa.
comprometimento	É uma ação resultando em obrigação de um ou mais participantes em cumprir uma regra ou executar um contrato.
declaração	É uma ação que estabelece um estado de negociação no ambiente em que o objeto está realizando a declaração (definição do contrato).
delegação	É uma ação que atribui autoridade, responsabilidade ou função a outro objeto.
avaliação	É uma ação que estima um valor de algo.
prescrição	É uma ação que estabelece uma regra.
agente	É um objeto de empresa que foi delegado (recebendo uma autoridade, responsabilidade, função, etc.) por outro objeto da empresa e que atua por ele.
principal	É uma <i>party</i> (pessoa) que delega (autoridade, uma função, etc.) para outro.
party contratada	É uma <i>party</i> (pessoa) que concorda com o contrato.

3.9 Níveis de Abstração do Modelo do Negócio

Um dos maiores problemas encontrados na especificação de sistema (incluindo a definição de sua arquitetura e seus respectivos requisitos) é que os projetistas responsáveis por esse documento esquecem que o sistema existe para realizar os objetivos do negócio, deixando assim de considerar requisitos essenciais para o sucesso do empreendimento.

Assim, existe a necessidade de se elaborar uma especificação do negócio para a definição dos RF e RNF do sistema, uma vez que, essencialmente, as necessidades do negócio devem orientar a melhor solução para o sistema. Nesse sentido, a especificação do negócio não deve estabelecer apenas quais as funções que o sistema deve possuir, devendo incluir também as regras e padrões segundo os quais o negócio deve operar. Na maioria das vezes essas regras não estão totalmente registradas em documentos, estando implícitas no conhecimento das pessoas que operam e comandam o negócio (*stakeholders*) (PUTMAN, 2001).

Neste trabalho, o negócio é focado no seu sentido mais amplo. Inclui as pessoas envolvidas (administradores, comerciais, técnicos, operadores, usuários, verificadores, mantenedores, etc.), os equipamentos envolvidos (máquinas que pertencem ao sistema e máquinas existentes no ambiente onde o sistema será instalado), as regras e as políticas (internas da organização e externas que devem ser seguidas pelo negócio) de ordem técnica, administrativa, financeira ou social. Dessa forma, a especificação do negócio deve ir além das características da tecnologia da informação, considerando outros aspectos que não fazem parte do sistema, mas interagem com ele, e que, se não forem considerados, podem invalidar a aplicação do sistema no ambiente para o qual foi desenvolvido.

Adicionalmente, a realização do modelo do negócio visa facilitar a comunicação entre o projetista do sistema e os *stakeholders*. Esse modelo deve ser uma via de duas mãos, devendo permitir que o projetista do sistema defina suas características (como o sistema deve ser distribuído para dar suporte ao negócio, como o sistema está posicionado no empreendimento e quais entidades do mundo

real fazem interface com ele), assim como deve permitir que os *stakeholders* tenham uma visão correta do posicionamento do sistema em relação ao negócio. Se esses objetivos forem cumpridos, os projetistas juntamente com os envolvidos com o empreendimento podem identificar as reais necessidades do sistema, quais regras aplicam-se ao mesmo, quais as restrições externas que o sistema deve atender, bem como podem especificar testes de aceitação que permitam verificar se o sistema resultante satisfaz às especificações do negócio (PUTMAN, 2001).

Com a utilização da linguagem de descrição da empresa (*The Enterprise Language*), definida para o modelo ODP, pode-se representar a especificação do negócio. Nessa especificação identificam-se os requisitos do negócio, que podem orientar a especificação da arquitetura do sistema (ISO-15414, 2002).

No modelo ODP a especificação do negócio é realizada através da elaboração da arquitetura do sistema nos pontos de vista da empresa e da informação. O processo de identificação das necessidades e das interfaces do negócio é auxiliado pela utilização dos conceitos de abstração e composição desse modelo.

Devido à generalidade dos conceitos do modelo ODP, estes podem ser utilizados para especificação de um sistema composto de sistemas de processamento de informação distribuída, para um sistema de processamento de informação distribuída singelo, para um domínio específico de objetos ou para uma coleção de objetos que compõem um componente maior em um sistema. Essa recursividade permite que um objeto, presente na especificação de um ponto de vista, possa ser um componente principal de um sistema, um subsistema ou um sistema (composto de subsistemas) em outro conjunto de especificações de pontos de vista, sendo que todos esses elementos podem fazer parte da mesma especificação de arquitetura, conforme representado na figura 3.2.

Em função da recursividade do modelo ODP, o modelo do negócio pode ser decomposto em vários níveis de abstração concatenados, permitindo o refinamento da visão do negócio até o nível de detalhe necessário para a identificação das interfaces do sistema a ser desenvolvido dentro da comunidade na qual ele está

contido. A identificação dessas interfaces permite a elicitação dos requisitos que o ambiente (ou a comunidade) impõe ao sistema.

Figura 3.2: Exemplo de recursividade do modelo ODP (PUTMAN, 2001)

3.10 Resumo dos Pontos de Vista ODP

Nas seções anteriores foram apresentados, resumidamente, os tópicos mais importantes do modelo de referência ODP, visando identificar as características que estão relacionadas com o propósito principal deste trabalho. Assim, não é apresentado no texto a parte formal da proposta ODP, podendo-se obter informações mais detalhadas nas normas que contém sua especificação (ISO-10746-1, 1995); (ISO-10746-2, 1995); (ISO-10746-3, 1995); (ISO-10746-4, 1995).

A tabela 3.4 resume as principais características de cada uma das visões do modelo ODP, procurando focalizar especialmente os aspectos que envolvem a elicitação de requisitos em cada uma dessas visões (FAROOQUI, 1995).

Tabela 3.4: Resumo das características dos pontos de vista do modelo ODP

Visão	Empresa	Informação	Computação	Engenharia	Tecnologia
Tipos de requisitos envolvidos	Necessidades do empreendimento; Objetivos e função do sistema na organização	Modelo de informação; Estruturas de informação; Fluxo de informação; Transformação da informação	Partição lógica da aplicação; Componentes (de software) da aplicação; Interfaces dos componentes	Infra-estrutura da plataforma distribuída Suporte de comunicação; Mecanismos de transparência da distribuição; Mecanismos genéricos de gerenciamento, controle e acesso	Recursos tecnológicos necessários para a implementação dos mecanismos da engenharia e dos componentes da computação; Recursos tecnológicos legados
Conceitos principais	Agentes (atores); Artefatos (recursos); Comunidades; Funções dos agentes; Domínio	Modelos; Relacionament o entre atores; Integridade da informação; Funções da aplicação	Objetos e interfaces computacionais; Restrições do ambiente computacional; Interações computacionais	Objetos básicos da engenharia; Objetos de transparência; <i>Stubs, binders</i> ; Objetos de protocolo; Núcleo (sistemas operacionais)	Soluções tecnológicas correspondentes aos mecanismos e estruturas da visão de engenharia
Quem estabelece os requisitos	<i>Stakeholders</i> do empreendimento; Gerentes da corporação; Normas e regras do domínio	Analistas de informação; Analistas de sistema; Engenheiros de informação	Projetistas da aplicação; Programadores	Projetistas de sistemas operacionais; Projetistas de sistemas de comunicação; Projetistas de sistema (aplicações do domínio)	Integradores de sistema Engenheiros de produto
Recursos de linguagem e notação	Linguagem de descrição de requisitos; Especificação de requisitos; Casos de uso	Modelo de entidade-relacionamento Modelos conceituais (estrutural, de objetos, dinâmico)	Ambiente de programação; Ferramentas de projeto de software; Linguagens de programação	Plataformas distribuídas; Ambiente de suporte de engenharia	Mapeamento de tecnologias; Identificação de recursos tecnológicos
Função na engenharia de software (em particular na engenharia de requisitos)	Elicitação dos requisitos iniciais do sistema (principalmente não funcionais); Projeto (arquitetura) preliminar do sistema; Definição dos casos de uso	Projeto conceitual da arquitetura; Modelo de informação; Especificação funcional do sistema; Detalhamento dos casos de uso	Projeto e desenvolvimento de software; Definição da arquitetura de software; Definição dos requisitos de software (consolida os requisitos da informação)	Projeto e desenvolvimento do sistema; Definição da arquitetura do sistema; Definição dos requisitos da arquitetura do sistema (consolida os requisitos das visões empresa, informação e computação)	Identificação de tecnologias; Identificação de recursos de instalação e manutenção; Detalhamento dos requisitos do modelo de engenharia

4 TÉCNICAS DE AVALIAÇÃO DE DEPENDABILIDADE

O objetivo deste capítulo é discutir os conceitos envolvidos no projeto de sistemas críticos (focalizando especialmente aqueles relacionados com a segurança), identificando as técnicas de projeto e as ferramentas de análise tipicamente empregadas para assegurar a adequação do sistema aos seus requisitos de dependabilidade (confiabilidade, disponibilidade e segurança).

Um sistema computacional pode ser definido como crítico em relação à segurança, quando sua operação (ou a falha desta) pode resultar em morte, prejuízo ou danos ambientais. Um comportamento inseguro pode resultar de requisitos errôneos ou falha do sistema em atender aos requisitos de segurança (HUSSEY, 2000).

Os princípios de um projeto de um sistema computacional com requisitos de segurança têm como prioridade eliminar as falhas durante o processo de desenvolvimento. Quando existirem falhas plausíveis que não possam ser eliminadas, o sistema deve conter mecanismos (tais como recursos de tolerância a falhas) para continuar operando com segurança mesmo na presença das mesmas.

As seções seguintes abordam as características de um projeto de sistema com requisitos relacionados com a segurança, apresentam as ferramentas mais utilizadas nas técnicas de análise de segurança e apresentam as técnicas de projeto que podem ser utilizadas para se garantir o atendimento dos requisitos de segurança e confiabilidade (seja para um sistema computacional distribuído ou centralizado).

4.1 Definições

Nesta seção definem-se termos específicos utilizados no domínio de sistemas relacionados com a segurança, tendo como referência básica a norma IEC-1508-4: 1997 - Functional Safety of Electrical/Electronic/ Programmable Electronic Safety-Related Systems – Part 4: Definitions and abbreviations (IEC-1508-4, 1997).

Confiabilidade:	Probabilidade de um componente ou um sistema estar funcionando corretamente dentro de um período de tempo, dentro de um conjunto de condições operacionais (STOREY, 1997);
Dano (<i>Harm</i>):	Ferimento físico ou prejuízo à saúde da população, direta ou indiretamente causado por prejuízo à propriedade ou ao meio ambiente (IEC-1508-4, 1997);
Disponibilidade:	Probabilidade de um componente ou sistema estar funcionando corretamente em qualquer instante de tempo (STOREY, 1997);
Estado errôneo:	Estado interno de um componente que pode levar ao insucesso, através de uma sequência de transições válidas (HUSSEY, 2000);
Estado seguro:	Estado de um sistema controlado, quando a segurança é atingida (IEC-1508-4, 1997);
Erro (<i>Error</i>):	Discrepância entre uma condição ou um valor computado, observado ou medido, e a condição ou o valor verdadeiro, especificado ou determinado teoricamente (contaminação da informação) (IEC-1508-4, 1997);
Falha (<i>Fault</i>):	Condição anormal que pode causar a redução ou a perda, da capacidade de uma unidade funcional executar uma função requerida (IEC-1508-4, 1997);
Disfunção (<i>Failure</i>):	Término da capacidade de uma unidade funcional de executar uma função requerida, com a manifestação da falha no ambiente em que o sistema está inserido (IEC-1508-4, 1997);
Perigo (<i>Hazard</i>):	Estado do sistema com potencial de causar danos (IEC-1508-4, 1997);

Risco (*Risk*): Combinação da probabilidade de ocorrência de um perigo e a severidade do mesmo (IEC-1508-4, 1997);

Segurança (*Safety*): Estado livre de riscos inaceitáveis (IEC-1508-4, 1997);

Segurança de acesso (*Security*): Prevenção de acesso não autorizado ou de modificação não autorizada da informação (STAVRIDOU, 1998);

Sistema: Conjunto de componentes que interagem, juntamente com um projeto que prescreve e controla o padrão dessa interação, podendo incluir o hardware, o software e a interação humana (IEC-1508-4, 1997);

Sistema relacionado com segurança: Sistema projetado para atender a ambos os seguintes critérios:

- Implementar as funções de segurança necessárias para atender ou manter um estado seguro para o sistema;
- Atender, ele próprio ou em conjunto com outros sistemas relacionados com a segurança ou em conjunto com facilidades de redução de riscos externos, a necessária integridade de segurança para as funções de segurança (IEC-1508-4, 1997).

4.2 Projeto de Sistemas Relacionados com a Segurança

Existe uma considerável superposição entre as técnicas de projeto utilizadas para sistemas críticos de segurança e aquelas utilizadas para sistemas que não são de segurança, mas cujo mau funcionamento pode ter conseqüências de elevado custo. Essas similaridades manifestam-se principalmente quando se comparam os enfoques da engenharia de segurança, confiabilidade, segurança de acesso (*security*) e sistemas de tempo real.

Um sistema de elevada confiabilidade (ou disponibilidade) é aquele no qual toda confiança pode ser colocada em certos aspectos da qualidade de serviços (QoS)

que ele fornece. A confiabilidade está envolvida primordialmente, com técnicas de tolerância a falhas (por exemplo, fornecendo um nível de serviços adequado, mesmo na ocorrência de falhas). A engenharia de segurança está concentrada nas conseqüências dessas falhas (disfunções) que devem ser evitadas, considerando explicitamente o contexto do sistema. Entretanto, algumas vezes não existe alternativa segura em relação ao serviço normal, sendo que neste caso o sistema deve ser confiável para ser seguro, fazendo com que ambos os conceitos estejam correlacionados.

Os sistemas de segurança de acesso (*security systems*) concentram-se em aspectos de prevenção de alteração da informação de forma não autorizada, integridade da informação, bloqueio de serviços e técnicas de não interferência. Como os sistemas de segurança também são dependentes da integridade do fluxo de informações, várias técnicas de segurança de acesso podem, potencialmente, ser utilizadas em sistemas críticos de segurança (*safety*) (LUTZ, 2000).

Em sistemas baseados fortemente em hardware, as técnicas de redundância e diversidade são geralmente empregadas para a redução dos perigos associados ao seu mau funcionamento. Em sistemas envolvendo fortemente software, o projeto voltado para a segurança deve considerar a prevenção dos perigos ou a sua redução e controle quando eles ocorrerem. O projeto voltado para a prevenção dos perigos inclui mecanismos tais como bloqueios de hardware para proteção contra erros de software, intertravamentos, temporização de vigilância (*watchdog timer*), isolamento dos módulos de segurança críticos, verificadores de sanidade (verificação de que o comportamento do software está de acordo com o esperado). Esses verificadores são assertivas que analisam as pré-condições nos dados de entrada (tipo de variável, faixa de valores), as pós-condições nos dados de saída ou se os dados estão invariantes tal que um estado perigoso contínuo possa ser evitado (LUTZ, 2000).

Essas similaridades contribuem para que as técnicas de projeto utilizadas em sistemas críticos de segurança tenham uma difusão em diversos tipos de sistemas.

Entretanto, ainda existe uma série de obstáculos na realização de projetos de sistemas relacionados com a segurança (LUTZ, 2000), dentre os quais destacam-se:

- Compromissos de projeto: as decisões de projeto usualmente envolvem compromissos entre os requisitos de segurança e os outros atributos desejados para o sistema. Os métodos de projeto visando a tolerância a falhas podem contribuir para a segurança do sistema (contribuindo, por exemplo, para determinar um comportamento temporal previsível para o sistema); entretanto, essas técnicas também criam interações adicionais entre os componentes e entre os níveis do sistema (por exemplo, para coordenar a recuperação de um estado perigoso), situação indesejável para um sistema crítico de segurança. Na medida em que o número de aplicações críticas de segurança aumenta, fatores comerciais e de mercado (como tempo de colocação no mercado e disponibilidade) tornam-se fatores importantes nas decisões de projeto (LEVESON, 1996);
- Consistência de modelo: as diferenças entre o real comportamento do sistema e o modelo mental do operador (ou do responsável pela especificação do sistema) relativo a esse comportamento são causas frequentes de geração de estados perigosos (por exemplo, confusão em relação ao modo de operação em pilotos); essa situação ocorre especialmente em sistemas de elevada complexidade, sendo caracterizada também por ser de difícil identificação na etapa de elicitação de requisitos e de difícil detecção na etapa de testes do sistema (RUSHBY, 1999);
- Uso limitado de técnicas de projeto conhecidas: o registro de diversos incidentes, envolvendo sistemas críticos de segurança, tem evidenciado como as técnicas conhecidas de boa prática de projeto de sistemas de segurança (como por exemplo, verificação de valores de entrada fora da faixa, verificação de limites da área de memória, etc.) são muitas vezes ignoradas. Adicionalmente, muitas lições podem ser obtidas a partir da análise desses incidentes, entretanto, é relativamente comum erros

repetirem-se em aplicações semelhantes. A utilização de técnicas de projeto seguro deve ser mais extensivamente divulgada, sendo seu uso plenamente justificado pelos custos envolvidos no insucesso da missão que as mesmas podem evitar (SLABODKIN, 1998); (ISBELL, 1999); (KNIGHT, 2002);

- Vulnerabilidade a erros simples de projeto: normalmente, tende-se a considerar o projeto voltado para segurança como um gerenciamento da complexidade do sistema, mas existe uma série de acidentes causados por erros simples de projeto. O paradigma de que pequenos erros tem pequenas conseqüências não é verdadeiro em software. O limite da habilidade em desenvolver sistemas seguros está relacionado com o quanto se conhece das características inatas do software e do ambiente onde é utilizado. Esse ponto torna-se extremamente mais importante quando se considera a possibilidade do uso de componentes de mercado (COTS - *Commercial Of-The-Shelf*) como parte de um projeto relacionado com a segurança, uma vez que a maioria desse tipo de componentes é vendida como caixa preta, sem o código fonte ou mesmo documentação detalhada (GARDINER, 1998); (ISBELL, 1999); (DEVANBU, 2000).

O objetivo da metodologia desenvolvida neste trabalho é contribuir, principalmente, para a melhoria dos três primeiros obstáculos citados, na medida em que esta visa: identificar e analisar os requisitos de segurança e a sua interação com os demais requisitos do sistema; identificar o modelo mental do usuário sistema através de técnicas de usabilidade, melhorando a consistência da especificação de requisitos; promover o uso de técnicas de análise de segurança ao longo do projeto, difundindo a cultura de segurança no desenvolvimento do sistema.

Nas seções seguintes discutem-se técnicas de análise de segurança e técnicas para projetos relacionados com a segurança que orientam a definição da metodologia proposta neste trabalho.

4.3 Técnicas de Análise de Requisitos de Segurança

Os perigos no nível de sistema são definidos como aqueles estados que podem conduzir a um acidente. Alguns desses perigos são evitáveis e podem ser eliminados (por exemplo, alterando o projeto do sistema ou alterando o ambiente onde o sistema deve interagir), ao passo que outros perigos inaceitáveis não podem ser evitados e o sistema deve conviver com eles. Os requisitos de segurança do sistema devem ser especificados para assegurar que o sistema possa conviver com esses perigos inevitáveis. No desenvolvimento de sistemas relacionados com a segurança a técnica associada com o processo de especificação e quantificação desses perigos é a análise de perigos (*Hazard Analysis*) (STOREY, 1997); (RUSHBY, 1999); (LUTZ, 2000).

Nesta seção apresentam-se, resumidamente, as técnicas mais freqüentemente utilizadas para análise de perigos em sistemas relacionados com a segurança, focalizando, principalmente, as ferramentas de análise de segurança destinadas ao software e a avaliação de erros humanos na segurança do sistema.

Os requisitos de segurança e confiabilidade do sistema são classificados como RNF por relacionarem-se eminentemente com as características do ambiente onde o sistema está inserido. Entretanto, quando esses requisitos são decompostos, surgem requisitos de segurança que são aplicáveis ao hardware e outros que são aplicáveis ao software do sistema. Este trabalho está concentrado na avaliação dos requisitos de segurança e confiabilidade de sistema, envolvendo tanto o hardware como o software. Apesar disso, o maior enfoque é dado para a análise de segurança de software, uma vez que a análise e o tratamento de requisitos de hardware já possui ferramentas e metodologias consolidadas, bem como técnicas de avaliação de confiabilidade padronizadas (por exemplo: modelos de Markov e redes de Petri) (SIEWIOREK, 1995).

As técnicas mais utilizadas para identificação de componentes críticos de segurança e elicitação de requisitos de segurança de software derivam das técnicas já largamente utilizadas na análise de perigos de hardware, porém com o enfoque ampliado para o software (LEVESON, 1996); (IEC-1508-1, 1997); (LUTZ, 1999).

A seguir apresentam-se, sucintamente, as técnicas de análise de segurança de hardware e suas respectivas extensões para o software:

- Análise de Árvore de Falhas (FTA - *Fault Tree Analysis*): é uma técnica de análise de perigos estruturada, partindo do nível mais alto do sistema (de um perigo ou evento indesejado) que vai sendo detalhado dentro da estrutura do sistema, até que sejam identificadas as possíveis causas do evento indesejado (*top-down analysis*). Essa técnica utiliza uma lógica booleana (operações E, OU, NÃO, OU-Exclusivo, etc.) para decompor um evento indesejado nas possíveis pré-condições que levam à sua ocorrência;
- Análise de Árvore de Falhas de Software (SFTA - *Software Fault Tree Analysis*): utiliza o mesmo método da FTA para analisar o código do software ou o projeto detalhado, para mostrar que o projeto de software não produz falhas inseguras no sistema ou, considerando a falha deste, para determinar as condições do ambiente que podem causar essa falha (RUSHBY, 1999);
- Análise de Árvore de Risco e Falha de Gerenciamento (MORT - *Management Oversight and Risk Tree analysis*): consiste, basicamente, de uma FTA acrescida de análise de funções gerenciais, comportamento humano e fatores ambientais. Seu objetivo é identificar problemas, defeitos e falhas em procedimentos que podem gerar perigos ou impedir sua identificação prematura, devido a falhas de planejamento, verificação operacional inadequada ou limites na troca de informações dentro da organização;
- Análise de Árvore de Eventos (ETA - *Event Tree Analysis*): pode ser utilizada quando a seqüência temporal dos eventos de falha é importante para a determinação dos cenários de acidente. Essa técnica consiste em uma árvore de decisão, onde, a partir de um evento iniciador, se verifica a seqüência temporal dos possíveis eventos subseqüentes, associados à sua probabilidade condicional. Sua utilidade principal é a avaliação

probabilística dos efeitos de funcionamento ou falha dos sistemas de proteção na sequência de um acidente, particularmente quando esses eventos tiverem uma ordenação temporal. Pode ser útil também para identificar o evento de topo da FTA ou na determinação dos cenários de acidente decorrentes de um evento iniciador;

- Análise de Causa-Consequência (CCA - *Cause-Consequence Analysis*): é um diagrama de sequência de eventos (blocos), mostrando a dependência temporal e a sua relação causal. A CCA é similar à FTA, iniciando-se com um evento crítico, determinando as causas desse evento e as consequências que podem resultar do mesmo. Sua aplicação principal é permitir estudos de análise de perigos em sequências de partida, desligamento e outros controles sequenciais;
- Análise de Modos de Falha, Efeitos e Criticidade (FMECA - *Failure Modes, Effects and Criticality Analysis*): é uma técnica consagrada de análise de perigos do hardware, sendo estruturada, no formato de uma tabela, que parte das causas das falhas (modos de falha) e analisa as consequências das mesmas no sistema (efeitos). Assim, essa técnica parte dos componentes mais básicos do sistema. e percorre a estrutura do sistema para verificar seu efeito global (*bottom-up analysis*);
- Análise de Modos de Falha, Efeitos e Criticidade do Software (SFMECA - *Software Failure Modes, Effects and Criticality Analysis*): é a extensão da FMECA para software. É um método para analisar os efeitos dos possíveis modos de falha do software no sistema, sendo mais frequentemente utilizada durante a fase de projeto, mas também tem sido utilizada na etapa de elicitação de requisitos. A SFMECA é orientada por uma série de modos de falhas já padronizados (por exemplo: datação errada de dados, término anormal do processo), sendo analisadas as consequências destes modos de falha para o sistema, sua criticidade (alta, média, baixa), sua probabilidade de ocorrência e a severidade das consequências;

- Análise de Perigos e Operabilidade (HAZOP - *Hazards and Operability Analysis*): é uma técnica consagrada de análise de perigos, estruturada, no formato de uma tabela, que considera desvios potenciais no ambiente ou na operação do sistema, para identificar suas possíveis causas e consequências. Similarmente a FTA, é realizada uma análise de cima para baixo do sistema (*top-down analysis*); entretanto, essa análise envolve mais elementos que a FTA, uma vez que é considerado fortemente o ambiente onde o sistema está inserido, bem como as ações do mundo externo (considera os perigos oriundos de problemas operacionais). Essa técnica é eminentemente qualitativa, sendo geralmente o primeiro método utilizado para a identificação sistemática de problemas de segurança oriundos do meio ambiente e que devem ser tratados pelo sistema. O HAZOP é muito útil na etapa inicial de elicitação de requisitos de segurança do sistema, na medida em que permite priorizar a ação da análise de segurança naqueles componentes que apresentam maior vulnerabilidade, fornecendo um guia para identificação de quais aspectos ou subsistemas merecem análise mais intensiva (por exemplo utilizando-se métodos formais para sua verificação). Essa técnica também é utilizada durante a implementação e o teste do sistema, onde deve haver um processo contínuo de verificação, para assegurar que o projeto está sendo corretamente implementado, assim como para remover ou mitigar os perigos envolvidos (normalmente novos requisitos decompostos surgem durante esse processo). Mesmo quando um sistema crítico é validado e colocado em operação, deve haver uma análise de perigos contínua, para assegurar que não surgiram novos perigos e que o sistema continua com riscos aceitáveis. Na prática, o uso do HAZOP é iterativo ao longo do processo de desenvolvimento do sistema, permitindo que requisitos adicionais possam ser descobertas e que suas consequências para o sistema sejam controladas (LUTZ, 2000).

Além dessas técnicas de análise de segurança de software, que são derivadas das técnicas de hardware, existe uma importante técnica que foi desenvolvida especificamente para a identificação de perigos relacionados ao software:

- Análise de Perigos por Máquina de Estados (SMHA - *State Machine Hazard Analysis*): esta técnica consiste em modelar o sistema através da representação de seus estados e das transições entre eles. Esse modelo pode torna-se extremamente complexo se todos os estados do sistema forem representados; assim, na prática, inicia-se a modelagem a partir de um estado perigoso e identificam-se os caminhos (ou estados) que levam a este, tendo como objetivo determinar como alterar o modelo do sistema, de modo que este estado perigoso não seja atingido. Com essa técnica o comportamento do software e de outros componentes do sistema pode ser modelado em um alto nível de abstração, permitindo que as falhas e as disfunções possam ser modeladas nas interfaces entre o hardware e o software. É, portanto, uma técnica que deve ser utilizada no início do processo de desenvolvimento do software relacionado com a segurança, uma vez que pode permitir a identificação das funções críticas de segurança do software. Durante o processo de desenvolvimento do sistema essa técnica auxilia no projeto de detecção de falhas, procedimentos de recuperação e identificação dos requisitos de falha-segura do sistema (RUSHBY, 1999).

As técnicas anteriores possuem um enfoque muito mais voltado para as falhas em equipamentos do que para falhas causadas por erros humanos; entretanto, já é bem conhecido que uma grande proporção dos acidentes são causados por erros humanos. Na literatura existem várias técnicas procurando identificar e prevenir erros humanos, sendo que as mais usualmente utilizadas são (LEVESON, 1996):

- Análise de Procedimentos ou Tarefas (PTA - *Procedure or Task Analysis*): é uma técnica estruturada, no formato de uma tabela, similar à FMECA, que parte de eventos de falhas em procedimentos operacionais e analisa as consequências das mesmas no sistema (efeitos);

- Análise de Erros de Ação (AEA - *Action Error Analysis*): é uma técnica estruturada, no formato de uma tabela, similar à FMECA, que parte de desvios de ações operacionais e analisa as conseqüências das mesmas no sistema (efeitos);
- Análise de Segurança de Tarefas (WSA - *Work Safety Analysis*): é uma técnica similar ao HAZOP, porém voltada para a pesquisa de falhas, sendo aplicada às etapas de trabalho do operador ou usuário.

O objetivo dessas técnicas de análise de erros humanos é identificar a informação necessária para o projeto da interface com o operador (IHC), tal que haja uma redução da probabilidade de ocorrência de falha por ação do operador e uma melhoria na habilidade deste na intervenção com sucesso no caso de acidentes. Como essas análises são baseadas no comportamento externo do operador, elas não dependem do sistema a ser desenvolvido, mas sim do processo no qual o sistema está inserido. Dessa forma, essas técnicas podem ser aplicadas antes do início do processo de desenvolvimento do sistema, para auxiliar na identificação dos cenários de acidente. Na etapa de verificação de requisitos, essas técnicas são úteis na determinação quantitativa das falhas de origem humana. Na seção 5.3 são discutidos modelos de erros humanos e métodos utilizados para avaliação e prevenção destes (LEVESON, 1996).

A proposta da metodologia desenvolvida neste trabalho considera a associação das ferramentas de análise de requisitos de segurança, apresentadas nesta seção, ao processo de desenvolvimento do sistema, de modo a permitir a verificação destes e assegurar a sua rastreabilidade durante todo o ciclo de vida do sistema.

4.4 Segurança x Segurança de Acesso

Um aspecto importante para ser discutido neste capítulo é a interação entre os requisitos de segurança do sistema (*safety*), referentes às funções de mitigação de falhas que podem levar o sistema a causar acidentes (com possíveis perdas materiais

ou humanas), com os requisitos de segurança de acesso (*security*), referentes às funções de inviolabilidade, integridade e disponibilidade de dados. Essa discussão é importante na medida em que os modelos de desenvolvimento de sistemas distribuídos estão sendo concebidos considerando-se os aspectos de *security* e não os de *safety* (STEEN, 1999); (OMG, 2001).

Apesar de uma contribuição comum para o aumento da confiabilidade dos sistemas, as pesquisas nas áreas de segurança e segurança de acesso têm seguido caminhos separados. Por outro lado, algumas pesquisas tem proposto a integração das análises de risco de segurança e de segurança de acesso, buscando uma metodologia unificada de tratamento para ambas, baseado na similaridade entre os dois processos (EAMES, 1999); (LUTZ, 2000); (KNIGHT, 2002).

A busca da integração dos dois processos baseia-se no fato de que o tratamento desses requisitos de forma independente pode gerar conflitos entre eles, que são difíceis de serem identificados e tratados sem uma visão integrada de ambos. Entretanto, também tem sido observado que na unificação de ambos os processos pode haver uma redução do poder de cobertura de cada análise, uma vez que as técnicas de análise de cada uma delas são especializadas (STAVRIDOU, 1998); (EAMES, 1999).

Uma investigação na literatura mostra que existem uma série de problemas quando se tenta unificar os tratamentos dos requisitos de segurança e de segurança da acesso, destacando-se as seguintes características (EAMES, 1999):

- Diferenças de modelos de desenvolvimento para segurança e segurança de acesso;
- Diferentes estruturas de análise e documentação de resultados;
- Interferência entre os requisitos de segurança e segurança de acesso (que se sobrepõe, mas não necessariamente se complementam (STAVRIDOU, 1998));

- Isolação entre os processos de engenharia de requisitos de segurança e segurança de acesso.

Apesar desses pontos negativos, tem sido investigadas formas de integração entre esses processos na medida em que:

- Apesar de possuírem modelos diferentes, o mapeamento entre os resultados dos dois tipos de modelagem pode aumentar a cobertura de falhas do sistema;
- Os requisitos de segurança tipicamente contém uma série de requisitos de segurança de acesso derivados, que devem ser comparados com os requisitos iniciais de segurança de acesso;
- O processo de elicitação e análise de requisitos é iterativo ao longo do ciclo de vida do sistema; como os requisitos de segurança podem interferir nos requisitos de segurança de acesso (e vice-versa), deve haver um processo para correlacionar, ao longo do ciclo de vida do sistema, os processos de análise de ambos os tipos de requisitos para resolver seus potenciais conflitos.

Nesta trabalho entende-se, como mais apropriado, o tratamento de ambos os processos de análise de modo independente, o que potencialmente pode gerar requisitos conflitantes. Adicionalmente, devem ser estabelecidos pontos de cruzamento de informações e análise conjunta, caracterizando-se assim, uma atividade de identificação e resolução de conflitos e inconsistências entre os dois tipos de requisitos. Essa harmonização de ambos os processos é similar à proposta realizada por Eames e Moffett (1999) (EAMES, 1999).

4.5 Técnicas de Projeto Relacionados com Segurança

Nesta seção, são apresentadas as linhas gerais que norteiam os projetos relacionados com segurança. O conjunto de técnicas apresentadas é estruturado de

forma hierárquica, partindo dos princípios de segurança e relacionando as estratégias, técnicas e requisitos típicos de segurança associados.

A classificação discutida é baseada nos princípios de arquitetura de segurança apresentado no trabalho de Hussey e Atchinson (2000), bem como nas orientações da norma IEC-1508-1:1997 (IEC-1508-1, 1997); (HUSSEY, 2000).

Nessa classificação, são considerados dois princípios de projeto de mais alto nível: a prevenção de falha e a tolerância a falha. Ambos os princípios são complementares na medida em que a prevenção de falha estabelece estratégias para assegurar que não ocorram a falha durante a operação do sistema (procurando eliminar as falhas na concepção e durante o desenvolvimento do projeto), enquanto que o princípio de tolerância a falha estabelece estratégias para que, na presença de falhas, o sistema possa detectá-las e recuperar-se, evitando a sua entrada em algum estado perigoso (uma vez que nem todas as falhas podem ser eliminadas no projeto) (LEVESON, 1996).

4.5.1 Princípios de prevenção de falha

As estratégias para a prevenção de falhas podem ser classificadas em três tipos (HUSSEY, 2000):

- Técnicas estruturadas de projeto: o uso de metodologias de desenvolvimento para o projeto de sistemas de segurança é essencial. Uma técnica estruturada para o projeto estabelece uma forma sistemática de particionamento e resolução de problema, estabelecendo uma nomenclatura (formal ou semi-formal), critérios e ferramentas para a decomposição lógica do sistema, recursos de análise, rastreabilidade, teste e verificação de consistência. Apenas com a utilização de uma estrutura para o projeto é possível a aplicação de técnicas de verificação e validação (V&V) ao projeto, para garantir a consistência e a completeza do mesmo em relação a segurança;

- Projeto padronizado: a padronização do projeto fornece critérios de qualidade e métricas que permitem avaliar as alternativas de projeto e os resultados da aplicação das técnicas adotadas (as métricas incluem medidas de complexidade, modularidade, correção, confiabilidade, eficiência, manutenibilidade, testabilidade, flexibilidade, portabilidade, reusabilidade e interoperabilidade). As métricas permitem ainda avaliar alternativas de estratégias de segurança e seus impactos nos outros requisitos (ABNT, 1999).
- Padrões de código: em projetos envolvidos com segurança, devem ser evitadas construções de código que não sejam especificadas na linguagem ou que sejam mau definidas (por exemplo, as estruturas que se aplicam a um compilador específico ou cujo resultado dependa da implementação) ou que dificultam a análise do comportamento do algoritmo. Essas linguagens devem ser padronizadas por normas, podendo ser linguagens padrão de mercado, mas limitada a um subconjunto de estruturas bem definidas (que permitam uma prova matemática da sua correção).

A tabela 4.1 apresenta um resumo das estratégias de prevenção de falha citadas, relacionando também um conjunto de técnicas e requisitos de segurança e confiabilidade típicos associados às mesmas.

Tabela 4.1: Resumo das estratégias de prevenção de falha

Princípio de segurança	Estratégia de projeto	Técnica relacionada	Requisitos típicos de segurança e/ou confiabilidade aplicáveis
Prevenção de falha	Estratégia estruturada de projeto	<p>Notação definida;</p> <p>Identificação dos limites e descrição do ambiente do sistema;</p> <p>Decomposição lógica do sistema;</p> <p>Recursos para verificação da consistência, completeza e correção do projeto (validação);</p> <p>Gerenciamento dos requisitos do sistema</p>	<p>Uso de metodologia de projeto (estruturado ou orientado a objetos);</p> <p>Uso de métodos semi formais (diagramas lógicos, diagramas de estados, redes de Petri, tabelas de decisão);</p> <p>Uso de métodos formais (notação matemática para descrição e análise da especificação do sistema);</p> <p>Uso de ferramentas para identificação e análise de requisitos de segurança (HAZOP, FTA, FMECA);</p> <p>Testabilidade (planos e procedimentos de inspeção e testes, revisão de projeto, testes automáticos);</p> <p>Reusabilidade (reutilização de componentes de segurança, limitações para uso de COTS);</p> <p>Clareza na especificação e no projeto</p>
	Projeto padronizado	<p>Critérios de qualidade de software;</p> <p>Medição dos critérios de qualidade</p>	<p>Simplicidade do software (testável, portátil, compreensível, interfaces diretas e limitadas entre componentes, tempo de resposta limitado verificável no código, determinístico, evitar o uso de interrupções, código deve conter somente as características mínimas requeridas);</p> <p>Modularidade (confinamento funcional com elevada coesão, baixo acoplamento e padronização de interfaces);</p> <p>Métricas para verificação de requisitos não funcionais (correção, confiabilidade, eficiência, manutenibilidade, testabilidade, flexibilidade, portabilidade, reusabilidade e interoperabilidade)</p>
	Codificação padronizada	<p>Evitar construções de códigos não especificadas na linguagem;</p> <p>Evitar construções de código dependentes da implementação</p>	<p>Uso de códigos simples e padronizados por normas;</p> <p>Uso de um subconjunto de instruções de código que assegure a verificação e eventual prova matemática da correção da codificação</p>

4.5.2 Princípios de tolerância a falha

As técnicas de tolerância a falhas devem garantir um funcionamento adequado do sistema (dentro de limites aceitáveis de segurança), mesmo com a presença de falhas no sistema. Esses princípios estão relacionados aos requisitos de confiabilidade e disponibilidade, mas são aqui tratados sob a óptica de sistemas críticos que envolvam segurança.

Geralmente uma falha não pode ser detectada pelo sistema até que alguma manifestação dessa falha gere um erro ou uma disfunção no sistema. Uma vez que o estado errôneo seja detectado, pode ser necessária uma avaliação da sua extensão em relação aos danos causados ao sistema. O sistema pode ser projetado para confinar os danos a uma área restrita, mantendo a sua operação normal ou o sistema pode ser projetado para, na presença do erro, retornar a um estado anterior seguro, eventualmente retomando a operação normal do sistema.

De uma forma geral, todas as técnicas de tolerância a falha dependem da utilização de alguma forma de redundância, o que implica em aumento da complexidade do sistema. O grau de redundância utilizado irá depender de uma série de fatores, destacando-se os requisitos de confiabilidade, disponibilidade, limitações de custo, limitações do ambiente onde o sistema irá operar (espaço, peso, potência, interfaces com outros sistemas).

Os princípios de tolerância a falhas podem ser subdivididos em quatro classes de estratégias de segurança (HUSSEY, 2000):

- Detecção de erros: a detecção de erros deve ser realizada o mais cedo possível, assim que ele se manifesta, para assegurar que uma ação efetiva possa ser tomada antes que o sistema atinja algum estado perigoso. A detecção de erros ocorre interceptando-se as saídas produzidas pelo sistema e verificando se as mesmas estão de acordo com a especificação do sistema. Essa função de detecção é realizada pelos verificadores (*checkers*) que, idealmente, devem ser independentes do sistema que está sendo verificado. As técnicas utilizadas pelos verificadores podem ser subdivididas em:

- ❑ Verificadores por replicação: duplicam as funções do sistema, através de redundância ou por repetição da operação. Essa técnica é mais aplicável a falhas aleatórias, uma vez que as falhas sistemáticas não ocorrem de forma independente nas redundâncias. Uma forma de replicação é a geração de N-versões de software, cujo funcionamento é replicado, mas cuja implementação é diversa. Entretanto, como essas réplicas tem implementação diferente, podem ser gerados modos de falha diferentes em ambas, reduzindo a confiabilidade do sistema; dessa forma, deve haver um balanço entre o grau de diversidade desejado e a redução de confiabilidade aceitável (POPOV, 1999);
- ❑ Verificadores por temporização: quando as condições de temporização são conhecidas utilizam-se verificadores de tempo máximo (*timeout*) ou verificadores de guarda de tempo (*watchdogs*) para garantir os limites temporais de execução;
- ❑ Verificadores por reversão: quando a computação da função inversa é única pode-se utilizar um verificador que calcula a entrada que deveria ter produzido a saída específica;
- ❑ Verificadores por codificação: usa informação redundante nos dados armazenados ou transferidos por comunicação, para detectar formas de corrupção dos mesmos. Exemplos comuns desse tipo de técnica são as verificações por paridade, verificações por somatória (*checksum*), e outros mecanismos mais sofisticados como marcas d'água nas mensagens transmitidas (DEVANBU, 2000);
- ❑ Verificadores por razoabilidade: determinam a aceitabilidade das saídas baseada nas informações do domínio, onde os resultados da computação são examinados quanto à satisfação de condições especificadas do sistema. Um exemplo simples é a verificação de que o valor obtido encontra-se em uma faixa aceitável, ou que a sua variação está de acordo com a dinâmica do processo. Esses mecanismos

geralmente não exigem a replicação do processo de geração dos dados, podendo ser mais simples que o algoritmo que ele protege;

- ❑ Verificadores por integridade estrutural: examina a integridade da estrutura dos dados. Por exemplo, a corrupção de duas listas ligadas pode ser detectada se a ligação entre elas for alterada; outros mecanismos mais sofisticados alteram a estrutura das mensagens, como criptografia e particionamento de código, onde a recuperação dos dados só é permitida com o uso de chaves cifradas (DEVANBU, 2000);
- ❑ Verificadores por diagnóstico: aplicam testes aos componentes do sistema para verificar sua adequação a resultados já estabelecidos. Os mecanismos de auto-teste já estão presentes na maioria dos componentes, podendo a diagnose ser executada *on-line* ou *off-line*;
- Confinamento de falha: o confinamento de falhas objetiva estruturar o sistema de modo a minimizar a propagação dos erros. O mecanismo básico para obtenção desse confinamento é o desacoplamento entre os elementos do sistema, que pode ser obtida pelos seguintes tipos de técnicas:
 - ❑ Particionamento físico: o desacoplamento entre os sistemas é obtido utilizando-se hardware distinto para a implementação de cada componente do sistema;
 - ❑ Particionamento lógico estático: a separação entre os componentes é obtida pela estrutura estática do projeto. Esse particionamento deve ser verificado para assegurar que os fluxos de dados e controle não violem esse particionamento, bem como devem ser eliminadas as interações não intencionais dos componentes oriundas do uso dos recursos comuns (memória, processador, etc.);
 - ❑ Particionamento lógico dinâmico: neste caso o acesso entre componentes desacoplados é controlado por mecanismos executados em tempo de execução (*run-time*). Esse mecanismo constitui-se

basicamente de controles de acesso, característicos das técnicas de segurança de acesso, onde a interação entre os componentes depende de um mecanismo de autorização gerenciado pelo próprio meio de acesso (DEVANBU, 2000);

- ❑ Intertravamento: os erros também podem ser confinados erguendo-se barreiras entre eventos ou estados do sistema incompatíveis, através de intertravamentos que limitam as seqüências de eventos. Esse processo difere das técnicas de partição na medida em que ele não impede a interação entre os componentes, mas restringe quando essa interação pode ocorrer;
- Recuperação direta de erros: após a detecção do erro, os mecanismos de recuperação direta de erros corrigem o erro e permitem continuar o processamento sem retorno a estados anteriores. Esse tipo de mecanismo depende de uma previsão antecipada das falhas que podem ocorrer e são, geralmente, específicos para a aplicação. Os tipos de mecanismos de recuperação direta de erros podem ser divididos em:
 - ❑ Reparo de dados: implementado quando existe suficiente redundância nos dados ou na estrutura de dados;
 - ❑ Reconfiguração: essa técnica utiliza o desligamento parcial do sistema ou a sua reconfiguração (por exemplo ativando um sistema de retaguarda), para eliminar o componente em falha do sistema;
 - ❑ Mascaramento: quando os erros são transitórios, como no caso de corrupção da comunicação, pode ser possível ignorar a ocorrência do erro e continuar a operação normal até que o erro seja corrigido (ou desapareça). Esse mecanismo é particularmente apropriado em sistemas de tempo real onde os dados são periodicamente atualizados;
 - ❑ Falha segura: se um estado seguro pode ser definido para o sistema, então pode ser possível a recuperação de erro forçando o sistema a ficar

permanentemente nesse estado seguro. Quando isso é possível não existe a continuidade do processamento, devendo ainda ser verificado se esse estado é seguro em todas as circunstâncias;

- Recuperação retroativa de erros: o tratamento de erros não previstos exige mecanismos mais sofisticados de recuperação, cujo objetivo é retornar o sistema a um estado anterior livre do erro e antes de ocorrência da falha. Esse tipo de mecanismo não depende de uma previsão antecipada das falhas, sendo, geralmente, aplicáveis a todos os sistemas. Entretanto, penalizam a velocidade do sistema e exigem mais recursos para o armazenamento e gerenciamento da informação; além disso, podem produzir efeitos físicos no processo devido ao retorno a um estado anterior. Os mecanismos de recuperação retroativa de erros podem ser divididos em:
 - ❑ Recuperação e retentativa: utiliza a redundância temporal ou entre componentes para recuperação de um estado errôneo. A recuperação do sistema para um estado seguro anterior pode ser realizada retornando ao último ponto de verificação registrado (*checkpoint*), ou retornando a um estado anteriormente armazenado. A forma mais simples de recuperação de estado é o retorno ao estado de início (*reset*). Uma vez que o estado seguro é recuperado a retentativa de operacionalização do sistema pode ocorrer repetindo as operações realizadas com os mesmos componentes, utilizando uma redundância temporal do sistema (se a falha for transitória é improvável que ela volte a ocorrer durante a retentativa). Outra forma de retentativa, para falhas permanentes, é a remoção do componente defeituoso do sistema e sua substituição por outro através de um chaveamento para um componente redundante (essa substituição pode ser realizada a quente - *hot-standby* ou a frio - *cold-standby*);
 - ❑ Replicação: esse é um caso especial de recuperação onde múltiplos componentes são executados em paralelo (redundantemente) de modo

que a recuperação para um estado anterior não é necessária. Na ocorrência de uma discrepância entre os elementos duplicados, os seus resultados são votados para determinar como deve seguir o processamento. Uma atenção especial tem sido dada na literatura com relação aos mecanismos de votação, especialmente em relação aos que dependem de redes de comunicação entre os elementos redundantes, devido ao problema de falha bizantina entre votadores (WANG, 2000);

- **Compensação:** mecanismo de recuperação pela realização do cancelamento da última operação, anterior a ocorrência do erro, sem a necessidade de recuperação de estado.

A tabela 4.2 apresenta um resumo das estratégias de tolerância a falha citadas neste parágrafo, relacionando também um conjunto de técnicas e requisitos de segurança e confiabilidade típicos associados às mesmas.

Tabela 4.2: Resumo das estratégias de tolerância a falha

Princípio de segurança	Estratégia de projeto	Técnica relacionada	Requisitos típicos de segurança e/ou confiabilidade aplicáveis
Tolerância a falha	Recuperação direta de erros (tratamento de erros a partir de falhas previstas)	Reparo de dados	Redundância nos dados ou na estrutura de dados
		Reconfiguração	Mecanismos de <i>backup</i> de sistema Mecanismos de reconfiguração
		Mascaramento (para falhas de comunicação)	Mecanismos de atualização de dados Mecanismos para tolerar falhas transitórias
		Falha segura	Mecanismos de condução do sistema a um estado seguro, sem continuidade de processamento

	Recuperação retroativa de erros (tratamento de erros a partir de falhas não previstas)	Recuperação e retentativa	Redundância no tempo ou nos componentes Mecanismos de condução do sistema a um estado seguro, com continuidade de processamento Mecanismos de repetição de operações anteriores Mecanismos de reconfiguração a quente (<i>hot-standby</i>) Mecanismos de reconfiguração a frio (<i>cold-standby</i>)
		Replicação	Mecanismos de execução de múltiplos sistemas em paralelo com votação de resultados
		Compensação	Mecanismos de retorno à operação anterior, sem a necessidade de recuperação de estado
	Confinamento de falha (isolação e desacoplamento dos componentes críticos de segurança)	Partição física	Desacoplamento dos componentes críticos de segurança utilizando hardware distinto
		Partição lógica estática	Desacoplamento dos componentes de segurança utilizando estruturas específicas do sistema (fluxo de dados e controle independente dos demais elementos do sistema)
		Partição lógica dinâmica	Mecanismos de autorização de acesso
		Bloqueio de intertravamento	Mecanismos de intertravamentos (restrições limitando interações entre componentes)
	Detecção de erros	Replicação	Mecanismos de verificação de erros por redundância ou por repetição de operação (geração de N versões do mesmo componente)
		Temporização	Mecanismos de <i>timeout</i> e <i>watchdog</i>
		Reversão	Mecanismos de computação da função reversa (a partir da saída obter a entrada)
		Codificação	Mecanismos de inserção de dados redundantes (paridade, <i>checksum</i> , marcas d'água)
		Razoabilidade	Mecanismos de aceitação de resultados de saída em função das características do domínio
		Integridade estrutural	Mecanismos de verificação da estrutura de dados (criptografia, partição de código)
		Diagnóstico	Mecanismos de auto-teste

5 TÉCNICAS DE AVALIAÇÃO DE USABILIDADE

O aumento da complexidade dos sistemas de controle, a exigência de elevados níveis de desempenho, combinados com a evolução tecnológica dos sistemas computacionais, vem aumentando cada vez mais a responsabilidade do papel da automação no controle dos mais diversos tipos de processos (desde sistemas voltados ao entretenimento até sistemas críticos de segurança). Normalmente a responsabilidade do controle desses processos recai tanto para controladores automáticos quanto para controladores humanos. Dessa forma, a eficácia e a segurança do controle pode depender fortemente da interação entre o computador e o ser humano (BROWN, 1998).

Adicionalmente, a interação entre homem e máquina está tornando-se crescentemente mais abstrata e complexa, fazendo com que surjam novas fontes de erro e novos tipos de perigos sejam introduzidos durante a utilização do sistema (especialmente do ponto de vista de segurança). A necessidade de reduzir os erros humanos, sem sacrifício das vantagens da automação, tem levado a idéia do projeto de sistema centrado no homem (*human-centered system design*) (DAOUK, 2001).

Uma análise recente mostrou que 63% dos grandes projetos de software tem um significativo aumento de duração em relação às estimativas de prazo iniciais. Os quatro maiores fatores considerados responsáveis por esse problema estão todos relacionados com usabilidade (LARSEN, 2000):

- Freqüentes solicitações de mudanças pelos usuários;
- Sobrecargas de tarefas;
- Dificuldade dos usuários entenderem seus próprios requisitos;
- Comunicação e entendimento insuficientes entre usuários e projetistas do sistema.

O objetivo deste capítulo é discutir os conceitos envolvidos com a determinação dos requisitos de usabilidade do sistema a ser desenvolvido, determinar os métodos de análise que podem ser empregados para verificação do atendimento

desses requisitos, bem como para o acompanhamento de sua evolução ao longo do ciclo de vida do sistema.

5.1 Conceitos de Engenharia de Usabilidade

A engenharia de usabilidade é uma disciplina que fornece métodos estruturados para a obtenção de usabilidade no projeto das interfaces com o usuário durante o desenvolvimento do produto. Essa disciplina envolve conceitos relacionados com psicologia cognitiva (que estuda a percepção e a cognição humanas para a orientação do projeto das interfaces homem-computador), psicologia experimental (que utiliza métodos empíricos para a medida do desempenho do usuário e de sua satisfação), etnografia (que estuda aspectos culturais para a descrição das características dos usuários), ergonomia (que estuda os esforços físicos para maximizar o desempenho e reduzir a fadiga e o desconforto na utilização de sistemas) e engenharia de software (que orientam integração da engenharia de usabilidade ao ciclo de vida dos sistemas) (MAYHEW, 1999).

Deve ser observado também que a engenharia de usabilidade não é representada apenas por técnicas, processos, métodos e procedimentos para o projeto de sistemas e produtos usáveis, mas inclui como aspecto mais importante, a filosofia que considera o usuário no centro do processo (RUBIN, 1994); (ISO-13407, 1999).

Dentro desse contexto insere-se a usabilidade, definida pela NBR 9241-11:2002 como: medida na qual um produto pode ser utilizado por usuários específicos para alcançar objetivos específicos com eficácia, eficiência e satisfação, em um contexto específico de uso. A eficácia da aplicação reflete a completeza e a precisão com as quais os usuários alcançam objetivos específicos. A eficiência mede a quantidade de recursos utilizados, em relação a precisão e abrangência, necessários para os usuários atingirem seus objetivos. A satisfação reflete a medida da ausência de desconforto e presença de atitudes positivas para com o uso de um produto. O contexto de uso abrange os usuários, tarefas, equipamentos (hardware, software e

materiais), e o meio físico e social no qual o produto é utilizado (NBR-9241-11, 2002).

Na comunidade de engenharia de software, o termo usabilidade foi inicialmente tratado de modo mais restrito, como sendo associado ao projeto da interface homem-computador (IHC). Mais recentemente, a revisão na norma ISO/IEC 9126-1:2000 ampliou essa visão reconhecendo que a usabilidade desempenha dois papéis no desenvolvimento de produtos de software: é uma atividade associada ao projeto de detalhamento do software, bem como está associada a objetivos globais de atendimento das necessidades do usuário que dependem do software, denominadas de qualidade em uso (BEVAN, 1999).

A qualidade em uso é a capacidade do produto de software de permitir a usuários específicos, atingir as metas especificadas, com eficácia, produtividade, segurança e satisfação, em um contexto de uso específico (ISO-9126-1, 2000).

A norma ISO/IEC 9126-1:2000 estabelece que a qualidade em uso é o efeito combinado de seis categorias de atributos de qualidade de software quando o produto está em uso (figura 5.1). O objetivo global do sistema deve atingir a qualidade em uso tanto para o usuário final (por exemplo, consumidor, operador, usuário do produto) como para o usuário de suporte (por exemplo, equipes de gerenciamento e desenvolvimento de projeto, fabricação, manutenção e reparo do produto) (ISO-9126-1, 2000).

Conforme representado na figura 5.1, os atributos de funcionalidade, confiabilidade, eficiência e usabilidade determinam a qualidade em uso para o usuário final num contexto particular. Os atributos de manutenibilidade e portabilidade estão associados com a qualidade em uso dos usuários de suporte. Nessa figura observa-se também que, do ponto de vista do usuário final de um produto de software, a usabilidade pode ser expressa pela capacidade desse produto ser compreendido, apreendido, usado e atrativo para o usuário final, quando utilizado sobre condições especificadas (ISO-9126-1, 2000).

Figura 5.1: Atributos de qualidade em uso de software (ISO-9126-1, 2000)

Assim, a usabilidade de um produto pode ser melhorada pela incorporação de características e atributos considerados capazes de beneficiar os usuários em um contexto particular de uso. Um produto pode ter níveis significativamente diferentes de usabilidade, quando usados em diferentes contextos (NBR-9241-11, 2002).

Não há consenso sobre o conjunto ideal de atributos de usabilidade que devem ser considerados no projeto de um sistema. Diversos autores e normas apresentam

conjuntos de atributos considerados essenciais para a definição da interface homem-computador (NIELSEN, 1993); (DIX, 1998); (FILGUEIRAS, 2000); (ISO-9126-4, 2000); (HFDG, 2001); (NBR-9241-11, 2002); (USABILITY, 2003).

Neste trabalho, adota-se, como guia para orientação do projeto da IHC um conjunto de princípios gerais de usabilidade elaborado por Dix et al. (1998). Este autor divide os princípios de usabilidade em três categorias: facilidade de aprendizado (capacidade de novos usuários iniciarem uma interação eficaz e obterem desempenho máximo), flexibilidade (capacidade do sistema interagir com o usuário de diversas formas) e robustez (capacidade do sistema apoiar o usuário na obtenção das suas metas com sucesso e de fornecer o reconhecimento desse estado). A tabela 5.1 apresenta resumidamente os atributos correspondentes a cada uma dessas categorias, bem como uma descrição das características associadas a cada atributo (DIX, 1998); (FILGUEIRAS, 2000).

Tabela 5.1: Princípios gerais de usabilidade para IHC (DIX, 1998)

Cat.	Atributo	Características
Facilidade de Aprendizado	Previsibilidade	Determinação do comportamento futuro do sistema com base no histórico; visibilidade das operações realizadas.
	Capacidade de síntese	Indução de um modelo mental a partir de ações passadas; avaliação de efeito das operações passadas no estado atual; notificação do estado interno do sistema.
	Familiaridade	Correlação entre conhecimento prévio do usuário e conhecimento necessário para uso eficiente do sistema; intuição da forma de uso do sistema por usuário novo.
	Capacidade de generalização	Aplicação de conhecimento válido numa certa situação em outra situação similar.
	Consistência	Semelhança de comportamento em tarefas com objetivos similares.
Flexibilid	Iniciativa de diálogo	Determinação do controle da diálogo (as ações podem ser restritas pelo sistema ou o usuário pode ativar qualquer comando do sistema); Em ações de segurança devem ser balanceados os modos de operação e os bloqueios de segurança.
	Paralelismo	Início de várias seqüências de diálogo relacionadas a diferentes tarefas do usuário (comunicação simultânea com vários processos, escalonamento de tarefas, utilização de múltiplos canais de comunicação).
	Capacidade de migração	Transferência de controle entre usuário e sistema, em função do contexto; Em aplicações de automação críticas essa característica ajuda o balanceamento de atividades de operação.

Robustez	Capacidade de substituição	Utilização de múltiplos formatos de representação para grandezas e estados equivalentes.
	Capacidade de configuração	Adaptabilidade da interface ao usuário (tanto o usuário pode ajustar o formato da interação com o computador, como a interface pode adaptar-se automaticamente em função do tipo de usuário).
	Observabilidade	Percepção do estado interno do sistema a partir da sua representação observável, com recursos de: <ul style="list-style-type: none"> ➤ <u>visibilidade</u>: identificação do estado do objeto e alternativas de ação através de informação visual; o usuário recebe realimentação contínua e total sobre o resultado das suas ações; ➤ <u>padronização</u>: padronização de valores normais (<i>defaults</i>); redução da necessidade de memorização; ➤ <u>alcance</u>: possibilidade de navegação entre estados observáveis; capacidade de recuperação de erros; ➤ <u>persistência</u>: duração do efeito de uma ação com possibilidade da utilização desse efeito na entrada de outra ação; possibilidade de escalonamento de tarefas; ➤ <u>navegabilidade</u>: exploração do estado do sistema através de janela limitada da interface; formatada de acordo com a necessidade de informação da tarefa.
	Capacidade de recuperação	Possibilidade de recuperação de erros do usuário; para tolerância a falhas e segurança a recuperação deve ser iniciada pelo sistema; princípio do esforço condizente (se for difícil desfazer deverá ser difícil fazer).
	Capacidade de resposta	Taxa de comunicação entre sistema e usuário; tempo de resposta do sistema para exibir efeito das ações do usuário; estabilidade (invariância nos tempos de resposta para recursos computacionais semelhantes).
	Conformidade às tarefas	Apoio a todas as tarefas do usuário; cobertura (mapeamento de todas as necessidades do usuário indicadas na análise de tarefas); compreensão do usuário em relação ao mapeamento dos recursos do sistema.

A escolha dessas heurísticas como o conjunto de atributos de usabilidade considerados neste trabalho deve-se a três aspectos: a frequência com que essas heurísticas são referenciadas na literatura; a sua associação direta a IHC de sistemas de supervisão e controle; e a sua objetividade em termos de caracterização desses atributos.

Deve-se ressaltar que, no projeto da IHC, além dessas heurísticas gerais, devem ser consideradas outras orientações de usabilidade específicas, relacionadas com o domínio da aplicação. Entretanto, recomenda-se que essas orientações devem ser desenvolvidas especificamente para cada tipo de projeto, baseando-se em

observações do que funciona e o que não funciona naquele específico domínio de aplicação, através de testes em produtos competitivos ou em protótipos iniciais do sistema (NIELSEN, 1993).

Como recomendação básica para a correta aplicação dos conceitos de usabilidade, Nielsen (1992) indica as seguintes ações: visitar as instalações do cliente antes de iniciar o projeto, realizar o projeto de forma iterativa com a participação do usuário, utilizar protótipos e testes empíricos com a participação dos usuários reais e iniciar a utilização de engenharia de usabilidade o mais cedo possível (NIELSEN, 1992).

As seções seguintes apresentam considerações sobre estratégias e técnicas que podem ser utilizadas para a inserção das atividades de engenharia de usabilidade no ciclo de vida do sistema, bem como métodos para avaliação dos atributos de usabilidade do sistema.

5.2 Modelo de Ciclo de Vida de Usabilidade

O objetivo desta seção é analisar as características do ciclo de vida de usabilidade, procurando a identificação destas com as atividades do ciclo de vida de software. Essa associação tem, como objetivo final, a integração do tratamento dos requisitos de usabilidade ao mesmo processo de tratamento de outros requisitos não funcionais do sistema (em especial os requisitos de segurança).

A engenharia de usabilidade tem, como filosofia básica, a colocação do homem no centro do processo de desenvolvimento. No caso dos sistemas interativos, baseados em computador, o foco principal do projeto centrado no homem é o desenvolvimento da interface homem-computador (IHC), sendo essa atividade parte integrante do ciclo de vida do software do sistema (ISO-13407, 1999); (FILGUEIRAS, 2000).

Pesquisas recentes tem buscado conciliar os modelos de desenvolvimento da IHC e modelos de desenvolvimento de software, propondo estratégias de

desenvolvimento que consideram os dois processos, bem como procurando simplificar os métodos de avaliação de usabilidade (DAOUK, 2001); (DRORI, 2003); (GRANOLLERS, 2003); (PYLA, 2003); (HARTSON, 2004).

O ciclo de desenvolvimento da IHC tem atividades análogas ao ciclo de desenvolvimento de sistema, contando com uma forte dependência dos aspectos comportamentais dos usuários (que determinam os requisitos de projeto da interação homem-computador). Comparando-o com um processo genérico de desenvolvimento de sistema (conforme representado na figura 2.3), a atividade de elaboração da arquitetura do sistema (partição de arquitetura) corresponde ao projeto da interação homem-computador (elaboração de telas e protótipos); a atividade de desenvolvimento dos subsistemas (projeto do software e do hardware) relaciona-se fortemente com o projeto do software (e do hardware) da IHC, na medida em que esta é um componente de software do sistema; analogamente a implementação de software e hardware e integração do sistema depende da implementação de software e hardware da IHC; finalmente as atividades de testes de validação do sistema devem contemplar os testes de usabilidade que envolvem a avaliação da IHC. Dessa forma, verifica-se que as atividades de desenvolvimento da IHC podem influenciar o processo de desenvolvimento de sistema como um todo (HARTSON, 2004).

Em termos temporais, o sincronismo entre as atividades de desenvolvimento da IHC e do restante do sistema não é um processo simples, na medida em que o processo de desenvolvimento da interação homem-computador é eminentemente iterativo, fazendo com que a evolução do projeto da IHC tenha seu ritmo dependente de vários fatores, que transcendem o grupo de desenvolvimento (tais como: disponibilidade dos usuários para seções de avaliação, a prototipação pode exigir recursos de software e hardware externos ao sistema, o número de ciclos de refinamento da IHC pode ser elevado). Para contornar esses problemas deve ser estabelecido um processo de gerenciamento dos ciclos de desenvolvimento da IHC.

A principal preocupação relativa ao gerenciamento do ciclo de desenvolvimento da interface homem-computador recai nos mecanismos de controle

das iterações. As recomendações para assegurar esse controle envolvem: estabelecer especificações de usabilidade (metas quantitativas), avaliar sempre com relação às especificações de usabilidade (para identificar áreas problemáticas), executar análise de impacto e custo/benefício (para determinar quais problemas atacar, em ordem de importância), decidir as mudanças a realizar no projeto da interação homem-computador, decidir quando parar o processo iterativo (os últimos 10% do processo de desenvolvimento podem consumir 90% do tempo) (HARTSON, 2004).

5.2.1 Ciclo de vida da engenharia de usabilidade

O modelo de ciclo de vida de engenharia de usabilidade considerado como referência neste trabalho é elaborado por Mayhew (1999). Esse ciclo de vida divide-se em três fases: análise de requisitos, projeto/desenvolvimento/teste e instalação, conforme representado na figura 5.2 (MAYHEW, 1999).

Na figura 5.2, as atividades de engenharia de usabilidade são representadas nos retângulos e as setas indicam a ordenação básica das tarefas. Observa-se que muitas atividades são iterativas, indicadas pelas setas de retorno a um estágio anterior do ciclo de vida. Os textos não envolvidos em caixas indicam o contexto de engenharia de software com o qual aquele ponto do ciclo de vida de usabilidade está relacionado. Nessa figura, são indicadas as etapas relativas a duas metodologias de desenvolvimento de software (como referência são utilizadas a metodologia OOSE - *Object Oriented Software Engineering* e a metodologia de prototipação rápida): na primeira linha de texto é indicada a atividade típica da metodologia de prototipação rápida, na segunda linha de texto é indicada a atividade correspondente à metodologia OOSE (JACOBSON, 1992); (MAYHEW, 1999); (SCHACH, 1999).

A fase 1 (análise de requisitos) envolve as seguintes atividades (MAYHEW, 1999):

- Obtenção do perfil do usuário: descrição das características do usuário (conhecimento, frequência de uso, habilidade com computadores);

Figura 5.2: Ciclo de vida de usabilidade (MAYHEW, 1999)

- Análise de tarefas contextual: descrição das tarefas, padrões de fluxo de trabalho e na compreensão e especificação dos objetivos dos usuários;
- Estabelecimento de metas de usabilidade: especificação qualitativa dos objetivos, resultantes dos requisitos elicitados do perfil do usuário e da análise de tarefas contextual, definindo o desempenho mínimo aceitável do usuário e critérios de satisfação;
- Restrições e potencialidades da plataforma: especificação dos recursos e restrições existentes na plataforma tecnológica de desenvolvimento da IHC;
- Guia de estilo: registro dos resultados obtidos nas quatro atividades anteriores referentes à análise de requisitos (orienta o projeto da IHC);
- Princípios gerais de projeto: revisão dos guias de orientação e dos princípios gerais de projeto da IHC disponíveis na literatura de engenharia de usabilidade, sendo registrados no guia de estilo.

A fase 2 (projeto/desenvolvimento/teste) é dividida em três níveis: no nível 1 é realizado o projeto de alto nível da interface, no nível 2 são estabelecidas as normas de projeto da interface e no nível 3 o projeto é detalhado. Cada um desses níveis de projeto é iterativo, em função da avaliação de usabilidade inserida em cada ciclo (MAYHEW, 1999).

O nível 1 (projeto de alto nível) envolve as seguintes atividades:

- Reengenharia do trabalho: reprojetado das atividades do usuário, no nível da organização e do fluxo de trabalho, para organizar e simplificar o trabalho explorando as vantagens da automação;
- Projeto do modelo conceitual: geração das alternativas de projeto de alto nível da interface. Nesse nível são identificados os caminhos de navegação, os recursos de visualização de informação principais, e as regras para apresentação consistente dos produtos, processos e ações do trabalho;

- Maquete conceitual: geração das maquetes (implementadas com papel e lápis ou como protótipos executáveis) com as idéias de projeto de alto nível, que representam a organização funcional e o projeto do modelo conceitual;
- Avaliação iterativa do modelo: avaliação das maquetes através de técnicas de avaliação de usabilidade. Essa atividade, juntamente com as duas anteriores, são realizadas em ciclos iterativos até que as falhas de usabilidade sejam identificadas e corrigidas no projeto do modelo conceitual. Quando esse projeto estiver relativamente estável pode-se iniciar o projeto da arquitetura do sistema (ou o modelo de análise do OOSE);
- Guia de estilo: ao final dos ciclos iterativos de projeto/avaliação o projeto do modelo conceitual é estabilizado e validado, sendo que esse resultado deve ser registrado no guia de estilo.

O nível 2 (estabelecimento das normas de projeto da interface) envolve as seguintes atividades:

- Definição de normas de projeto de telas: estabelecimento dos padrões específicos do produto e as convenções referentes a todos os aspectos do projeto detalhado das telas, baseando-se nos dados da análise de requisitos, no projeto do modelo conceitual e nos padrões estabelecidos pelo negócio e/ou pelos padrões dos produtos corporativos. Essas normas devem assegurar a coerência e consistência de todas as interfaces do usuário;
- Prototipação de normas de projeto de telas: aplicação das normas de projeto de telas e das definições do projeto do modelo conceitual, no projeto detalhado de interfaces do usuário, utilizando um protótipo executável;
- Avaliação iterativa de telas: avaliação iterativa dos protótipos das normas de telas, que são refinados e reprojatados, até que uma norma robusta de projeto de telas seja validada. Os ciclos iterativos são realizados até que as falhas de usabilidade sejam eliminadas e os seus objetivos atendidos. Com as normas de projeto de telas validadas, pode-se completar o projeto de

modelos e iniciar o desenvolvimento da aplicação (utilizando-se a metodologia de prototipação rápida) ou pode-se completar o modelo de projeto e iniciar o modelo de implementação (na metodologia OOSE);

- Guia de estilo: atualização do guia de estilo com o resultado final dos ciclos iterativos de projeto/avaliação das normas de projeto de telas. Esse documento é utilizado durante o detalhamento do projeto da interface do usuário para assegurar a qualidade, coerência e consistência dessa interface.

O nível 3 (detalhamento do projeto da interface do usuário) envolve as seguintes atividades:

- Projeto detalhado da interface: realização do projeto detalhado completo do produto de interface do usuário, baseado nos resultados validados do modelo conceitual e das normas de projeto de telas, documentados no guia de estilo do produto;
- Avaliação iterativa do projeto detalhado da interface do usuário: avaliação do projeto detalhado da interface do usuário a até o desenvolvimento do produto final, para verificar os critérios de usabilidade com relação a subconjuntos de funções ainda não testadas e a novas categorias de usuários. Esse processo deve ser gerenciado com eficácia, utilizando-se critérios qualitativos de avaliação de usabilidade, de modo a evitarem-se ciclos iterativos desnecessários.

A fase 3 (instalação) envolve basicamente a atividade de realimentação do usuário, que consiste no retorno de informações do usuário com relação a dificuldades de uso encontradas após a instalação e algum tempo de entrada em produção. Essa informação pode ser utilizada para melhoria do projeto, para o projeto de novas versões de interface e/ou para novos projetos de produtos relacionados com a aplicação (MAYHEW, 1999).

Esse modelo de ciclo de vida de usabilidade é utilizado como referência neste trabalho, devido à sua adequação ao ciclo de desenvolvimento espiral e devido à sua

adequação aos conceitos que são adotados no modelo ODP. A análise dessa adequação é realizada na seção 6.3.

5.2.2 Métodos de avaliação de usabilidade

Nesta seção, discutem-se as características básicas dos métodos para a avaliação de usabilidade mais usualmente referenciados na literatura. Não se pretende entrar em detalhes procedurais destes, nem apresentar uma lista exaustiva de métodos, uma vez que a frequência de proposição de novos métodos é muito elevada. O objetivo dessa apresentação é identificar categorias de métodos que podem ser utilizados nas várias etapas de desenvolvimento da metodologia de especificação de requisitos de usabilidade.

O elemento central de todas as etapas do ciclo de vida de usabilidade é a avaliação de usabilidade do produto que está sendo desenvolvido. Para se determinar o nível de usabilidade alcançado, é necessário medir o desempenho e a satisfação dos usuários, trabalhando com um produto. A medição de usabilidade é particularmente importante para visualizar a complexidade das interações entre o usuário, os objetivos, as características da tarefa e os outros elementos do contexto de uso (NBR-9241-11, 2002).

A seguir são apresentados os métodos de avaliação de usabilidade mais rotineiramente utilizados. O enfoque da discussão concentra-se na determinação de quais métodos são mais adequados para cada uma das fases do ciclo de desenvolvimento da IHC. Visando facilitar essa identificação, os métodos de avaliação de usabilidade são subdivididos em três categorias principais: teste, inspeção e questionamento. Além disso, cada método também é classificado, com relação ao envolvimento do usuário e seu ambiente, em três tipos: método especialista, empírico ou contextual. A tabela 5.2 apresenta um resumo dessa classificação (MAYHEW, 1999); (ZHANG, 1999); (USOR, 2002); (USABILITY, 2003).

Tabela 5.2: Classificação dos métodos de avaliação de usabilidade

Método de Avaliação	Fase do Ciclo de Vida				Envolvimento do Usuário		
	Análise de Requisitos	Projeto	Codificação e Teste	Operação	Método Especialista	Método Empírico	Método Contextual
Protocolo pensando alto		X	X	X		X	
Protocolo perguntando-questões		X	X	X		X	
Aprendizado de co-descoberta		X	X	X		X	
Método de ensino		X	X	X		X	
Método do treinador		X	X	X		X	
Teste retrospectivo		X	X	X		X	X
Medida de desempenho			X	X	F	X	
Walkthrough pluralista		X			H		
Walkthrough cognitivo		X	X	X	H/F		
Avaliação heurística		X	X	X	H		
Lista de verificação baseada em cenários		X	X	X	H		
Inspeção de características			X	X	H		
Estudo de campo pró-ativo	X	X					X
Observação de campo	X	X	X	X			X
Entrevista	X	X	X	X		X	X
Entrevista contextual	X	X	X	X			X
Questionário / user survey	X	X	X	X		X	X
Grupo de foco	X	X	X	X		X	X
Tempestade Cerebral	X	X	X	X		X	X
Análise de registro de uso			X	X	F	X	X
Legenda: os métodos especialistas são classificados em Formais - F ou Heurísticos - H							

TESTE

INSPEÇÃO

QUESTÃO

Em relação ao envolvimento do usuário os métodos classificam-se em:

- Métodos especialistas: são aqueles onde a avaliação é realizada por um especialista do domínio no lugar do usuário real. Esses métodos são aplicados essencialmente em avaliações do tipo inspeção e podem ter um carácter formal ou heurístico. Os métodos formais procuram demonstrar a

usabilidade através de alguma técnica abstrata, baseando-se em técnicas matemáticas, cognitivas ou analíticas, sendo geralmente de aplicação complexa, mas apresentam resultados mais precisos. Os métodos heurísticos, devido à sua simplicidade, são mais práticos, pois baseiam a avaliação do produto no entendimento intuitivo dos projetistas a respeito da qualidade, do conceito de boas praticas de projeto ou de listas de verificação de usabilidade;

- Métodos empíricos: são aqueles onde a avaliação é realizada utilizando-se os usuários reais do sistema, porém em condições de laboratório (ao invés do seu ambiente real de trabalho). Esses métodos são aplicados para a avaliação da forma e para o reprojeito da IHC. Os métodos empíricos são derivados da psicologia cognitiva e social, e avaliam características específicas da interface em laboratório (o que facilita o processo de avaliação em relação a situações mais complexas observadas em campo) e permite medidas de desempenho geralmente estatísticas (o que possibilita a verificação da evolução do projeto da interface, bem como a comparação entre soluções alternativas);
- Métodos contextuais: são aqueles onde a avaliação é realizada utilizando-se o contexto (ambiente de trabalho) onde os usuários atuam. Esse tipo de método apresenta como vantagem, em relação aos dois tipos anteriores, o fato de levar em consideração o trabalho verdadeiramente executado pelos usuários no seu ambiente de trabalho. Os métodos contextuais são pragmáticos e poderosos do ponto de vista da cobertura das reais características de uso. Entretanto, tem como pontos negativos, o fato de serem de mais difícil realização (exigem acesso às instalações do usuário, podem alterar as rotinas operacionais, geram um grande volume de dados para ser analisado), são menos conclusivos que estudos empíricos e geralmente não fornecem números e resultados de fácil interpretação.

Em relação ao tipo de avaliação os métodos classificam-se em:

- Métodos baseados em teste: nesses métodos, os usuários (ou seus representantes) trabalham nas suas tarefas típicas, utilizando o sistema (ou protótipos), enquanto que os avaliadores observam os usuários trabalhando e verificam se a IHC atende às necessidades destes para a realização das tarefas. Os métodos de avaliação desse tipo incluem:
 - ❑ Protocolo pensando alto: este método tem, como objetivo, identificar o modelo mental do usuário sobre o sistema, seus sentimentos e opiniões enquanto interage com o sistema; além disso permite identificar a terminologia (vocabulário) utilizada pelo usuário e pode identificar novas funções necessárias ao sistema. Em uma seção de teste de usabilidade, os usuários de teste devem verbalizar o que estão pensando (quais considerações estão sendo feitas mentalmente durante a execução de cada tarefa), na execução de cenários específicos (NIELSEN, 1993);
 - ❑ Protocolo perguntando-questões: este método tem, como objetivo, identificar o modelo mental do usuário sobre o sistema e as tarefas a serem realizadas, bem como identificar suas dificuldades no uso ou na compreensão do sistema. Em uma seção de testes, os analistas de usabilidade fazem perguntas diretas aos usuários sobre o sistema, enquanto os usuários de teste executam um conjunto de tarefas especificado. Esse processo pode ser mais natural que o método pensando alto, uma vez que estimula o usuário a verbalizar seus pensamentos (ZHANG, 1999);
 - ❑ Aprendizado de co-descoberta: este método tem, como objetivo, identificar o modelo mental dos usuários sobre o sistema e as tarefas a serem realizadas. Em uma seção de teste, dois usuários de teste utilizam o sistema para executar, em conjunto, algumas tarefas sobre cenários especificados. Ambos os usuários devem se ajudar mutuamente para atingir o objetivo das tarefas, sendo encorajados a verbalizar o que estão

pensando durante a execução das tarefas. Esse processo pode ser mais natural que o método pensando alto uma vez que o trabalho em grupo facilita a verbalização dos usuários (NIELSEN, 1993);

- ❑ Método de ensino: este método tem, como objetivo, verificar as características de usabilidade através da observação do aprendizado, da eficácia e da facilidade de operação com que o usuário de teste ensina as tarefas utilizando-se do sistema. Inicialmente, um usuário de teste é familiarizado com a interface do sistema e adquire habilidade para a realização de tarefas utilizando o sistema. A seguir, um novo usuário é colocado junto com cada usuário de teste. O usuário de teste deve explicar ao novo usuário como o sistema trabalha e apresentar a este um conjunto predeterminado de tarefas (VORA, 1995);
- ❑ Método do treinador: este método tem, como objetivo, identificar as informações necessárias para os usuários, de modo a prover um melhor treinamento e documentação, bem como identificar as necessidades de reprojeção da interface para evitar as questões formuladas durante o teste. Na seção de teste de usabilidade, um treinador, especialista do sistema, responde a qualquer pergunta dos usuários. O analista de usabilidade observa a interação entre o treinador e os usuários, bem como a interação entre o usuário e o sistema (NIELSEN, 1993);
- ❑ Teste retrospectivo: este método tem, como objetivo, identificar o modelo mental do usuário sobre o sistema, a eficácia, eficiência e satisfação do usuário. Uma seção de teste de usabilidade é gravada em vídeo (tanto o comportamento do usuário como a sua interação com o sistema) e reproduzida para os usuários que participaram da mesma. São realizadas perguntas sobre o comportamento do usuário durante o teste e o mesmo descreve o que estava fazendo e porque (essa técnica é especialmente útil nos casos onde a interação entre o analista de usabilidade e o usuário é restrita na seção de teste) (NIELSEN, 1993);

- ❑ Medida de desempenho: este método é utilizado para a quantificação do desempenho dos participantes durante as seções de teste de usabilidade realizadas em laboratórios de usabilidade formais (de modo que possam ser coletados dados precisos de desempenho, com um mínimo de interferências externas). Para a obtenção de resultados confiáveis são necessários no mínimo 5 participantes. Os resultados quantitativos são utilizados em testes comparativos ou testes relacionados a um padrão de referência (*benchmark*) (NIELSEN, 1993).
- Métodos baseados em inspeção: nesses métodos, os especialistas em usabilidade verificam aspectos relacionados com a usabilidade da interface com o usuário, utilizando os cenários de tarefas ou um conjunto de heurísticas. Essas heurísticas são derivadas de áreas como: estudos da interação homem-computador, ergonomia, projeto gráfico, projeto de informação e psicologia cognitiva. Os métodos de inspeção mais usuais são:
 - ❑ Walkthrough pluralista: neste método, o grupo que realiza a avaliação da usabilidade deve ser composto por três tipos de participantes: representante dos usuários, representante do desenvolvimento do produto e engenheiro de usabilidade. A atividade consiste em repassar passo a passo um conjunto de tarefas, avaliando individualmente e discutindo em grupo a usabilidade do sistema para cada tarefa (NIELSEN, 1994);
 - ❑ Walkthrough cognitivo: neste método, a avaliação envolve uma ou mais pessoas que executam uma série de tarefas, avaliando a compreensão e a facilidade de aprendizado da IHC. Esse método não envolve o usuário do sistema, entretanto possui como atividade inicial a identificação do perfil do usuário. Os avaliadores normalmente são engenheiros de usabilidade, desenvolvedores de software ou pessoal do marketing e da documentação. A análise consiste em examinar cada ação do usuário realizada no caminho da solução de um problema, procurando identificar

se a interface com o usuário facilita a realização das tarefas (NIELSEN, 1994);

- ❑ Avaliação heurística: a heurística é um guia, com regras práticas de usabilidade, que pode orientar as decisões de projeto ou pode ser utilizado para criticar um projeto já executado. Nesse método, um conjunto de 3 a 5 avaliadores (engenheiros de usabilidade) procuram identificar individualmente os problemas de usabilidade da IHC (orientados pelas regras heurísticas), sendo que os testes são geralmente realizados sobre maquetes ou protótipos do sistema. Após a avaliação individual ser completada, os inspetores reúnem-se e comparam suas observações, procurando um consenso sobre os problemas de usabilidade do sistema (NIELSEN, 1994);
- ❑ Lista de verificação baseada em cenários: neste método, a inspeção ocorre em três tipos de cenários: usuário novato, usuário especialista e condições anormais. Para cada um dos cenários é previamente elaborada uma lista de itens a serem verificados. Além dos inspetores (usualmente três desenvolvedores de software) existe um moderador (usualmente um engenheiro de usabilidade) que prepara o material, coordena a inspeção e coleta os resultados (ZHANG, 1999);
- ❑ Inspeção de características: este método analisa a disponibilidade, compreensibilidade e outros aspectos de usabilidade de um conjunto de características do produto. Os inspetores (engenheiros de usabilidade) utilizam os casos de uso do produto, verificando se os resultados obtidos são os esperados com a utilização do sistema. Cada uma das características do produto é analisada durante a realização de cada caso de uso (ZHANG, 1999).
- Métodos baseados em questionamento: nesses métodos, os especialistas em usabilidade procuram identificar as necessidades do usuário, o que o agrada (ou desagrada) o usuário, bem como procuram compreender o sistema

conversando com o usuário, observando-o na utilização do sistema em operação real (sem o objetivo de teste de usabilidade), fazendo questionamento verbais ou utilizando questionários escritos. Os métodos de avaliação desse tipo incluem:

- ❑ Estudo de campo pró-ativo: este método tem, como objetivo, identificar os usuários, suas atividades e o ambiente de trabalho. O estudo é realizado por um grupo de engenheiros de usabilidade que visitam e conversam com os usuários no seu ambiente de trabalho, focalizando os seguintes aspectos: características individuais dos usuários, tipos de tarefas realizadas, objetivos a serem atingidos e possibilidades de evolução da forma de trabalho (NIELSEN, 1993);
- ❑ Observação de campo: neste método, o engenheiro de usabilidade deve ir ao local de trabalho dos usuários para observá-los na execução de suas tarefas, procurando compreender como os usuários estão utilizando o sistema na realização de suas tarefas e qual o modelo mental que os usuários possuem sobre o sistema. Parte das observações podem ser entrevistas com os usuários; parte podem ser observações dos usuários utilizando o sistema ou levantamentos do ambiente de trabalho (NIELSEN, 1993);
- ❑ Entrevista: este método tem, como objetivo, obter informações detalhadas sobre o produto, baseando-se em questões de interesse do usuário. O entrevistador (normalmente um engenheiro de usabilidade) formula questões orientadas para usuários representativos do sistema, que respondem verbalmente expressando suas opiniões sobre o tema, sendo registradas essas respostas pelo entrevistador (normalmente a entrevista é gravada para registro) (NIELSEN, 1993);
- ❑ Entrevista contextual: este método tem, como objetivo, obter informações detalhadas sobre os problemas de usabilidade do produto experimentados por usuários em seu ambiente natural de trabalho. Essa

técnica é similar a entrevista, porém voltada para a obtenção de informações dos usuários sobre as suas práticas de trabalho e suas experiências, objetivando a compreensão da natureza do trabalho do usuário para ser utilizada na identificação de requisitos de usabilidade. Esse tipo de método também propicia um forte envolvimento participativo do usuário no processo de desenvolvimento (o entrevistador deve fazer o papel de aprendiz), uniformizando a linguagem e estrutura utilizada no trabalho, permitindo uma visão consistente do sistema entre os vários membros do grupo de desenvolvimento (usuários, projetistas, gerentes, etc.), na medida em que estes podem estar envolvidos nos processos de entrevista e análise dos dados (BANNON, 2000);

- ❑ Questionário / *user survey*: este método tem como objetivo identificar características específicas de usabilidade (no início do projeto pode ser utilizado para obtenção de informações básicas sobre os requisitos de usabilidade do sistema), sendo que já existem questionários padronizados, cujos aspectos de validade e confiabilidade são estabelecidos (alguns exemplos são: QUIS – *Questionnaire for User Interface Satisfaction*, PUEU – *Perceived Usefulness and Ease Use*, NAU – *Nielsen's Attributes of Usability*, NHE – *Nielsen's Heuristic Evaluation*, CSUQ – *Computer System Usability Questionnaire*, ASQ – *After-Scenario Questionnaire*, PHUE – *Practical Heuristics for Usability Evaluation*, PUTQ – *Purdue Usability Testing Questionnaire*). Atualmente já existem sítios na WEB onde esses questionários podem ser configurados e enviados via e-mail para usuários representativos do sistema (PERLMAN, 1999);
- ❑ Grupo de foco: este método tem, como objetivo, reunir um grupo de 6 a 9 usuários representativos para discutir as características de usabilidade do sistema, capturando as reações espontâneas dos usuários e as idéias que surgem no processo de dinâmica de grupo. Na etapa inicial de projeto,

enquanto se determinam-se as formas normais de uso do sistema com a análise de tarefas, este método permite identificar as regras de exceção. É mais utilizado para verificar a opinião dos usuários, não gerando propriamente uma avaliação (NIELSEN, 1993);

- Tempestade cerebral (*brainstorming*): este método é um dos mais antigos métodos destinado ao estímulo da criatividade em grupo. Seu objetivo é reunir um grupo de 2 a 12 usuários representativos para discutir um problema ou idéia a ser explorada, relacionada com as características de usabilidade do sistema, capturando as idéias que surgem no processo de discussão. Esse processo permite que todos os participantes da atividade obtenham um melhor entendimento do espaço do problema e cria um sentimento comum sobre os resultados obtidos pelo grupo. Neste método, o facilitador (engenheiro de usabilidade) expõe o problema ou idéia a ser discutida e a seqüência de eventos do método, que consiste basicamente em duas fases: na primeira fase todos apresentam suas idéias, sem que sejam criticadas; na segunda fase as idéias são agrupadas, descartando as similares ou incorretas, ordenadas e priorizadas. O facilitador registra os comentários resultantes da discussão para posterior utilização (USABILITY, 2003);
- Análise de registro de uso: este método tem como objetivo analisar as seqüências das ações do operador, obtendo dados estatísticos sobre a freqüência de uso dos recursos da IHC, de erros de operação, de eventos de interesse e de uso de recursos de ajuda *on-line*. O registro dessas operações deve ser realizado automaticamente pelo sistema computacional, sem a necessidade de intervenção do usuário. Como esse registro contém apenas as ações realizadas pelo usuário, e não o motivo o levou a realizá-las, é comum que esse método seja combinado com outros (como entrevistas, observação de campo ou questionários) para avaliação de eficiência e eficácia no uso da IHC (NIELSEN, 1993).

Em complemento aos métodos de avaliação de usabilidade citados anteriormente é importante destacar dois métodos de modelagem que são freqüentemente utilizados nos projetos de IHC: análise de tarefas e técnicas de prototipação.

O método de análise de tarefas avalia como os usuários executam suas tarefas e qual o fluxo das atividades relacionadas com o seu trabalho, permitindo determinar os objetivos e as tarefas que devem ser executadas pelo usuário para melhoria da eficiência e da satisfação do usuário, constituindo-se numa referência para as funções e as características do sistema que devem ser testadas (MAYHEW, 1999).

As técnicas de prototipação envolvem o desenvolvimento de representações do sistema para efeito de testes e avaliação, podendo variar desde simples desenhos (protótipos de baixa fidelidade) até sistemas que possuam uma funcionalidade próxima a do sistema final (protótipo de alta fidelidade), tendo como objetivo permitir uma avaliação formal por parte do usuário (facilitada pela utilização da linguagem de representação visual da própria IHC) (TORRES, 2002).

5.3 Avaliação de Erros Humanos

Nesta seção, discute-se um tópico especial de usabilidade relacionado com a avaliação de erros humanos, que pode ter impacto decisivo sobre os requisitos de segurança do sistema, especialmente no caso de sistemas de controle relacionados com segurança (LEVESON, 1996); (LUTZ, 2000); (GROZDANOVIC, 2001).

As técnicas de análise de erros humanos podem caracterizar a ponte entre os requisitos de segurança e de usabilidade, permitindo uma análise do balanceamento entre o nível de automação e o nível de risco. Nas seções seguintes discute-se a modelagem dos erros humanos, procurando identificar o relacionamento entre a avaliação de usabilidade (especificamente nos aspectos relacionadas com os erros humanos) e a análise de segurança, visando uma harmonização de ambos os processos na metodologia proposta neste trabalho (EAMES, 1999); (NATO, 2001).

Para a avaliação dos erros humanos no projeto da IHC de sistemas de supervisão e controle, devem ser considerados os seguintes aspectos referentes aos cenários de incidentes (GROZDANOVIC, 2001):

- Erros de manutenção e teste que afetam a disponibilidade dos sistemas de segurança (erros latentes);
- Erros de operação que iniciam o incidente;
- Ações de recuperação pelas quais o operador pode terminar o incidente;
- Erros pelos quais os operadores podem prolongar ou eventualmente agravar o incidente;
- Ações pelas quais os operadores podem recuperar equipamentos e sistemas inicialmente indisponíveis.

Alguns modelos de análise de erros humanos permitem uma estimativa da probabilidade de cada um desses tipos de erros. Essa quantificação permite o cálculo do risco do sistema (levando em consideração os fatores humanos), o qual pode ser comparado com valores aceitáveis para determinar se há necessidade de sua melhoria. Se os erros humanos não puderem ser reduzidos a níveis aceitáveis, então podem ser necessárias medidas adicionais de hardware, software, reprojetado do sistema e/ou da sua forma de operação. A experiência tem mostrado que uma combinação efetiva de mudanças de hardware e/ou software e redução de erros humanos tem atingido níveis aceitáveis de risco (GROZDANOVIC, 2001).

No caso de redução de risco para erros considerados críticos existe uma série de aspectos que podem ser melhorados (KIRWAN, 1994):

- Prevenção por mudanças de hardware e/ou software: desacoplamento de módulos de segurança, uso de intertravamentos, automação de tarefas freqüentes, uso de mecanismos de falha segura, etc.;
- Aumento na tolerância a falhas do sistema: tornando o hardware e software mais flexível ou auto-corrigível (por exemplo, com mecanismos de

reconfiguração, mascaramento de falhas), permitindo maior variabilidade de recursos para o usuário atingir seus objetivos;

- Melhoria na recuperação de erros: melhoria da detecção e correção de erros por meio de aumento de realimentação, procedimentos de verificação, supervisão e monitoramento do desempenho do sistema;
- Redução de erros na origem: redução de erros por melhoria de procedimentos, treinamento e projeto da IHC.

Os dois primeiros aspectos citados exigem a uma forte interação entre o especialista de fatores humanos, analista de segurança e a equipe de projeto, sendo ações de custo normalmente elevado. A melhoria da probabilidade da recuperação de erros pode ser obtida com medidas de implementação relativamente simples, porém geralmente não são suficientes para a redução do risco a níveis aceitáveis, e nem sempre são possíveis para todos os erros. A redução de erros na origem é a medida primária mais eficaz na melhoria da confiabilidade humana, uma vez que exige menos esforço, tem uma cobertura mais global e apresenta menos efeitos colaterais no sistema (KIRWAN, 1994).

5.3.1 Métodos de análise de erros humanos

Existem várias formas de taxonomia de erros humanos na literatura; uma das mais adequadas para a identificação de erros humanos relacionados com controle de processos e automação é o modelo baseado no enfoque de processamento da informação. Esse modelo apresenta quatro categorias de erros humanos: enganos (associados a erros de interpretação e falhas na formulação das intenções), deslizos (associados a execução incorreta de ações pretendidas, apesar das intenções estarem corretas), lapsos (associados a falha na execução das ações devido a ausência de determinados passos, causada pelo esquecimento de determinadas ações) e erros de modo (associados a execução de ações supostamente corretas em contextos incorretos) (WICKENS, 1992).

Na literatura são encontrados vários métodos que permitem a identificação das causas dos erros humanos e, em alguns casos, a sua predição. Dois relatórios técnicos do Programa de Gerenciamento de Tráfego Aéreo Europeu apresentam um resumo atualizado de vários métodos de predição e análise de erros humanos (SHELL – 1972: *Software, Hardware, Environment and Liveware*, SHERPA – 1986: *Systematic Human Error Reduction and Prediction*, GEMS – 1987: *Generic Error-Modeling System*, PHECA – 1988: *Potential Human Error Cause Analysis*, CREAM – 1993: *Cognitive Reliability Analysis Method*, ATHEANA – 1996: *A Technique for Human Error Analysis*, MERMOS – 1998: *Méthode d’Evaluation de la Réalisations des Mission Opérateur pour la Sûreté*), bem como considerações sobre combinações das mesmas e ferramentas disponíveis (HRS, 2002); (EEC, 2004).

Através desses métodos é possível determinar a contribuição individual das causas e dos cenários que contribuem para os erros humanos analisados, permitindo a determinação da estratégia de redução de erros mais adequada.

A estratégia adotada neste trabalho objetiva utilizar as informações fornecidas pelos métodos de análise de erros humanos (particularmente pelo método ATHEANA) para avaliar a contribuição de erros humanos ao longo do desenvolvimento do sistema (associado a aspectos de segurança e usabilidade). Para a avaliação desses erros no desenvolvimento do sistema são discutidos na próxima seção modelos de análise e quantificação de risco de erros humanos.

5.3.2 Modelos de análise de risco

Em áreas de aplicação envolvendo sistemas críticos de segurança (tais como área nuclear, aeronáutica, metro-ferroviária, plantas químicas), os órgãos reguladores de segurança (órgãos licenciadores) exigem que sejam realizadas avaliações probabilísticas de risco (PRA – *Probabilistic Risk Assessment*) para quantificação dos perigos envolvidos com a instalação e a operação do sistema. Qualquer análise de risco é baseada nos conceitos de frequência (ou probabilidade) dos eventos de perigo e sua severidade, para avaliar se o risco é aceitável dentro de níveis de aceitação

padronizados, como por exemplo a escala ALARP (*As Low As is Reasonably Practicable*) da norma IEC1508-1:1997 (IEC-1508-1, 1997); (STOREY, 1997).

Com a ampliação do escopo da aplicação dos sistemas de segurança, existe uma tendência de que esse tipo de análise seja necessário também para outras aplicações, que envolvam sistemas relacionados com a segurança. Além disso, a avaliação de risco é um critério essencial para a solução de conflitos entre requisitos (especialmente entre requisitos de segurança). A avaliação de risco tem ainda, como subproduto, diretivas para a redução do risco e conseqüente melhoria da segurança global do sistema.

Nas áreas de aplicação, onde tradicionalmente são aplicados os sistemas críticos de segurança, já existem procedimentos bem estabelecidos para as avaliações probabilísticas de risco. Entretanto, esses procedimentos, via de regra, consideram dentro do processo de análise de risco apenas os eventos associados a falhas dos componentes físicos da planta controlada e/ou dos sistemas computacionais envolvidos, partindo de eventos iniciadores internos e externos ao ambiente controlado. A partir do acidente na usina nuclear de Tree-Mile Island (1979), a indústria nuclear passou a considerar, com grande ênfase, o efeito dos fatores humanos nas suas análises de risco, surgindo as técnicas de avaliação da confiabilidade humana (HRA – *Human Reliability Assessment*), que modelam e prevêem o erro humano associado à operação (em situações normais e de emergência) do sistema (NEA, 1998); (HRS, 2002).

Na última década ocorreu a maturação dessas técnicas e uma maior profundidade da compreensão dos erros humanos, incluindo suas causas, manifestações e conseqüências. Atualmente as técnicas de HRA possuem modelos de erros humanos que consideram as influências organizacionais, bem como falhas de manutenção e teste, erros associados a automação e violações operacionais intencionais, cometidas pelos usuários do sistema ou por elementos estranhos ao sistema (por exemplo, *hackers*). Entretanto, ainda não existem procedimentos de avaliação probabilística de risco, bem estabelecidos nas áreas tradicionais de

aplicação de sistemas críticos, que integrem os fatores humanos às outras técnicas tradicionais de análise de segurança (PROMAIN, 2001); (HRS, 2002); (EEC, 2004).

A HRA, juntamente com a avaliação de outros riscos já considerados tradicionalmente (tais como computacionais, estruturais, mecânicos e ambientais), deve ser utilizada para a avaliação probabilística de risco (PRA), obtendo-se uma avaliação do nível de segurança global do sistema. O diagrama da figura 5.3 representa, esquematicamente, a integração entre HRA e PRA.

Figura 5.3: Integração dos fatores humanos a avaliação de risco do sistema

Conforme indicado na figura 5.3, a PRA utiliza resultados das avaliações de segurança de hardware e de software, cujas técnicas, discutidas anteriormente na seção 4.3, possuem métodos bem estabelecidos (FAA, 2000).

A PRA identifica as causas dos perigos (eventos iniciadores dos acidentes), identifica os cenários onde esses perigos podem ocorrer, determina as consequências resultantes dos perigos identificados, define os critérios de segurança a serem adotados para eliminar ou minimizar os efeitos dos perigos, quantifica as probabilidades de ocorrência dos eventos críticos de segurança, considera a frequência de ocorrência dos eventos e sua severidade. Pode avaliar o nível de segurança obtido no projeto do sistema e/ou o nível de risco para a sociedade, relacionado com os perigos que não podem ser eliminados ou evitados. A PRA avalia os riscos associados com cada aspecto do ciclo de vida do sistema relacionado com a

segurança, fazendo parte das atividades de concepção, projeto, construção, operação e descomissionamento da instalação (LEVESON, 1996); (STOREY, 1997).

As técnicas de HRA têm um enfoque múltiplo, derivado das disciplinas de engenharia, psicologia e ergonomia. Como a HRA tem o objetivo de identificar e quantificar a probabilidade de ocorrência de erros humanos, ela também permite identificar quais tipos de erros devem ser reduzidos, bem como pode fornecer opções de como isso pode ser realizado. Existem vários métodos para realização da HRA; entretanto, estes apresentam um conjunto de atividades similares (KIRWAN, 1994):

- Definição do escopo da análise: definição do escopo da HRA no contexto da PRA (por exemplo: especificando tipos de cenários, tarefas, usuários);
- Especificação e/ou análise de tarefas: especificação dos objetivos do sistema e das tarefas de interesse relacionadas com a segurança;
- Identificação dos erros humanos: aplicação de um método sistemático para a identificação dos potenciais erros em cada componente das tarefas;
- Integração da representação: representação das interações humanas em um formato que possa ser integrado com a representação da PRA do sistema (por exemplo, análise de tarefas, FTA, ETA), e que permita a compreensão do impacto das ações humanas no contexto do sistema, podendo incluir potenciais ações de recuperação de falha;
- Quantificação da probabilidade dos erros: para cada ação humana é atribuída uma probabilidade de ocorrência, que é considerada na quantificação da confiabilidade humana em cada cenário de acidente;
- Redução de erros: especificação de medidas para redução ou recuperação dos erros humanos e sua eficácia estimada;
- Documentação: registro dos resultados e realização de uma análise de sensibilidade ou garantia da qualidade do processo.

Dentre as atividades anteriores, as duas consideradas críticas para o sucesso da avaliação da confiabilidade humana são a identificação dos erros humanos (HEI – *Human Error Identification*) e a probabilidade de erro humano (HEP – *Human Error Probability*) atribuída a cada tipo de erro identificado (figura 5.3).

A HEI pode ser obtida a partir dos métodos de análise relacionados na seção 5.3.1, bem como de técnicas de análise de segurança consagradas, como a FMECA e o HAZOP, discutidas na seção 4.3. Essas técnicas, originalmente desenvolvidas para análise de todas as possíveis falhas de equipamentos, podem ser utilizadas para identificação dos erros humanos com a utilização de um conjunto de diretivas de investigação adequado. No caso do HAZOP, por exemplo, existe o variante H-HAZOP (*Human HAZOP - Human Hazard and Operability Analysis*), no qual se constrói uma tabela onde cada cenário de operação é investigado, analisando-se as consequências dos possíveis tipos de erros humanos que podem ser cometidos, durante a realização das ações necessárias para a conclusão de cada tarefa (os erros tipicamente considerados são: a não realização, a ação a menor, a ação a maior, a ação igual, a troca de ações, a reversão de ação e a ação parcial) (PROMAIN, 2001).

A quantificação da HEP também pode utilizar ferramentas clássicas de análise de segurança, como a FTA e a ETA. Entretanto, o elemento crítico desse processo é a necessidade de informações históricas sobre erros humanos. As áreas nuclear, metro-ferroviária e aeronáutica vêm, a algumas décadas, montando bases de dados com registros de erros humanos sobre incidentes e acidentes. Essas referências podem ser utilizadas pelas outras indústrias com as devidas adaptações de cenários (PROMAIN, 2001); (HRS, 2002); (EEC, 2004).

Os métodos atuais de avaliação de confiabilidade humana consideram detalhadamente os processos cognitivos na predição dos erros humanos, fornecendo base de dados com os valores de HEP e com as características dos cenários onde esses valores foram obtidos. Um desses métodos, considerado de última geração, é o ATHEANA (*A Technique for Human Error Analysis*) (PARRY, 1995); (HRS, 2002).

O método ATHEANA é baseado na compreensão dos motivos da falha da interação homem-sistema. Esse método amplia a avaliação dos erros, considerando não só os cenários normais que conduzem ao erro, mas também verificando as condições forçadas de erro, ações potencialmente inseguras e as condições da planta (NATO, 2001). O seu modelo de processo requer que os eventos de erro humano e procedimentos associados sejam identificados na PRA (por exemplo, o operador falha em acionar uma bomba de emergência). As ações inseguras (por exemplo, um equipamento em operação é desligado) são então identificadas como aquelas que podem afetar ou causar esses eventos. As condições associadas ao erro forçado (por exemplo, procedimentos mal estruturados ou indicações errôneas) são identificadas, para explicar porque essas ações inseguras podem ocorrer. Essas condições forçadas são baseadas no contexto do sistema que está sendo analisado. A figura 5.4 apresenta as atividades principais desse método (NEA, 1998).

Um evento de erro humano pode resultar de várias ações inseguras. No método ATHEANA, para cada evento de erro humana são identificadas as ações inseguras e os contextos de erro forçado associados. As cinco tarefas associadas ao método são (as atividades representadas com retângulos de fundo claro são relacionadas à PRA):

1. Familiarização com o modelo da PRA, especialmente com os cenários de acidente identificados;
2. Identificação dos potenciais eventos de erro humano (HEI) e identificação dos tipos de ações inseguras, que são potencialmente importantes para a produção do efeito caracterizado em cada um desses eventos;
3. Para cada tipo de ação insegura, devem ser identificadas as causas mais importantes para sua ocorrência e devem ser identificados os contextos de erro forçado mais significativos;
4. A identificação dos contextos forçados de erro, associados com as respectivas ações inseguras, fornecem a orientação para o detalhamento da HEI, bem como para o detalhamento dos cenários, considerados no modelo

da PRA. Dessa forma, refina-se o modelo da PRA ou selecionam-se cenários mais específicos para as ações inseguras identificadas;

5. Para cada tipo de ação insegura, e sua causa associada, devem ser quantificadas as probabilidades de ocorrência dos contextos de erro forçado (nos cenários considerados na PRA). A seguir, estima-se a probabilidade condicional dos erros humanos em função da probabilidade de ocorrência dos contextos erro forçado associados.

Figura 5.4: Fluxograma do método ATHEANA (NEA, 1998)

Finalmente, as probabilidades de ocorrência de erros calculadas nesse método podem ser incorporadas no modelo da PRA, para a revisão da quantificação do riscos de segurança do sistema, em função da consideração dos erros humanos.

Apesar desse processo parecer muito abrangente (uma vez que pode existir um grande número de combinações de eventos de erro, ações inseguras e contextos de erro forçado na ocorrência de cada acidente), o método incorpora orientações detalhadas para priorização e seleção dos eventos de erro mais significativos. Como orientação de alto nível, o método considera as ações inseguras de maior interesse como aquelas que surgem de uma base racional (mas incorreta), ou seja, as ações irracionais, espontâneas e arbitrárias não são consideradas (NEA, 1998).

Um aspecto importante desse tipo de enfoque é que, na medida em que são identificadas as causas dos erros humanos (forçados ou não), pode-se atuar na sua redução, por exemplo, alterando características de usabilidade da IHC. Dessa forma, esse processo pode criar um forte relacionamento entre a análise de requisitos de segurança e a avaliação dos requisitos de usabilidade do sistema.

5.3.3 Conclusões sobre a avaliação de erros humanos

O estudo das características dos modelos de análise de risco e dos métodos de avaliação dos erros humanos discutidos permite concluir que:

- Existem métodos e informações de referência padronizadas sobre as probabilidades de ocorrência de erros humanos, em áreas de aplicação críticas, que podem ser adaptados para utilização em outros domínios através da correção dos fatores cognitivos e das características específicas dos sistemas a serem analisados. Em particular as ferramentas clássicas, já utilizadas na análise de segurança de hardware e software (como HAZOP, FMECA, FTA, ETA), podem ser utilizadas também na identificação e quantificação da probabilidade de erro humano, incluindo-se as classes de falhas identificadas nos modelos de erro cognitivos;

- A avaliação quantitativa dos riscos é um processo iterativo, uma vez que esta depende do detalhamento do erro humano. Nesse processo a elaboração da PRA vai sendo refinada (em função da identificação de novos cenários de acidente, novos eventos iniciadores, novas ações inseguras e novos contextos forçados de erro) ao longo do ciclo de vida do sistema. Dessa forma, é razoável considerar que esta avaliação possa caminhar em paralelo com o processo evolutivo da engenharia de requisitos e com o ciclo de vida de usabilidade (inclusive acompanhando o processo de detalhamento da arquitetura do sistema);
- A análise de risco de erros humanos, realizada nos métodos de avaliação mais recentes, caracteriza-se pelo relacionamento da origem dos erros humanos aos processos cognitivos envolvidos, fornecendo diretivas de prevenção de erro associadas às características de usabilidade do sistema. Dessa forma, evidencia-se uma interação, possivelmente forte, entre os processos de análise de requisitos de usabilidade e de análise de segurança;
- Outro aspecto convergente entre a avaliação de erros humanos e a análise de requisitos de usabilidade é a forte dependência de ambos os processos dos cenários de operação; considerando como cenário de operação não apenas a análise de tarefas, mas incluindo-se também os aspectos ambientais envolvidos (fatores ergonômicos, organizacionais, treinamento, hábitos, relacionamentos do grupo de trabalho);
- Como modelo básico de método de análise de erro humano adota-se neste trabalho o método ATHEANA (figura 5.4) por tratar-se de uma técnica de última geração, que evidencia a influência dos aspectos cognitivos e ambientais na ocorrência de ações inseguras, e por ser direcionada ao refinamento da análise probabilística de risco.

6 MERUSA: METODOLOGIA DE ESPECIFICAÇÃO DE REQUISITOS DE USABILIDADE E SEGURANÇA ORIENTADA PARA ARQUITETURA

O objetivo deste trabalho é elaborar uma metodologia que permita a elicitação e análise de requisitos de usabilidade e segurança, orientada para o desenvolvimento da arquitetura de sistemas distribuídos e abertos, denominada MERUSA.

Foi adotado, neste trabalho, o modelo de referência ODP como arquitetura padrão para os sistemas de objetos distribuídos, uma vez que esse processo, além de ter sido padronizado pelo ISO, já teve a sua aplicabilidade para sistemas abertos de automação (SAA) verificada por Becerra (1998) (BECERRA, 1998).

A elaboração dessa metodologia considera vários dos conceitos discutidos nos capítulos anteriores, que foram integrados, conforme apresentado na figura 6.1. Essa figura representa um detalhamento do elemento central da figura 1.1, mostrando que a MERUSA é obtida a partir integração das metodologias de especificação de requisitos de usabilidade e segurança.

Figura 6.1: Etapas de desenvolvimento da metodologia MERUSA

Inicialmente, considerando-se as discussões dos capítulos 2, 3 e 4, é realizada uma avaliação da adequação dos critérios segurança ao processo de desenvolvimento de sistemas distribuídos baseados no modelo ODP. Com base nesta avaliação, é proposta uma metodologia de especificação para os requisitos de segurança do sistema.

Em seguida é realizada uma avaliação similar para as características de usabilidade do sistema (considerando-se as discussões dos capítulos 2, 3 e 5). Com base nesta avaliação, é proposta uma metodologia para a especificação dos requisitos de usabilidade do sistema.

Finalmente, procura-se harmonizar ambas as metodologias anteriores, de modo a identificar uma proposta comum, denominada MERUSA, cujo foco é a especificação de requisitos de segurança e de usabilidade do sistema.

No capítulo 7 a MERUSA é avaliada através da sua aplicação ao processo de desenvolvimento de um Sistema de Supervisão e Controle de Sistema de Transmissão de Energia para o Sistema Elétrico Interligado Nacional (SIN).

6.1 Adequação das Técnicas de Segurança ao RM-ODP

Comparando-se os serviços de transparência previstos no modelo ODP (seção 3.6) e as suas funções de segurança de acesso (seção 3.7.4) com as técnicas de projetos relacionados com segurança (seção 4.5), nota-se que existem vários elementos comuns, o que induz considerar a possibilidade da adequação do modelo ODP ao projeto de sistemas relacionados com a segurança.

Entretanto, observam-se também alguns potenciais conflitos entre as características dos sistemas relacionados com a segurança e as características previstas do sistema previstas no ODP (representadas pelas funções ODP), especialmente no que tange a simplicidade de projeto requerida pelos sistemas relacionados com a segurança, havendo portanto a necessidade de uma comparação mais minuciosa entre esses aspectos.

Uma avaliação mais precisa da adequação de requisitos de segurança ao ODP exige um conhecimento detalhado do sistema e do ambiente em que ele está inserido. No entanto, a proposta deste trabalho é fazer a avaliação para um ambiente computacional distribuído, contemplando obrigatoriamente as funções previstas nesse modelo, sem entrar no mérito das características específicas do domínio e da aplicação. O resultado dessa avaliação é qualitativo, procurando identificar os pontos de potencial conflito entre os requisitos e os serviços previstos no modelo ODP com relação às técnicas típicas observadas em sistemas de segurança.

Entende-se que essa abordagem é válida porque muitas das características próprias dos sistemas relacionados com a segurança estão associadas com os aspectos básicos da estrutura do sistema (tais como redundância, tolerância a falhas, confinamento de falhas, recuperação de falhas). Essas características podem ser comparadas (sob aspectos de completeza, conflitos e adequação de conceitos) com os recursos fornecidos pelo modelo ODP (especialmente suas funções) para a modelagem e implementação destas, sem que haja a necessidade da definição de um domínio de aplicação específico.

A seguir, as técnicas de segurança relacionadas nas tabelas 4.1 e 4.2 são revisadas sistematicamente, analisando-se a sua adequação ou não a cada uma das visões do modelo ODP (tabela 3.4).

Em relação às técnicas das estratégias de prevenção de falhas para projetos de segurança (tabela 4.1), tem-se a seguinte avaliação:

- Notação de especificação definida: essa característica é atendida pelo modelo ODP como um todo, uma vez que o mesmo estabelece a sintaxe de representação das informações para cada um dos pontos de vista. O uso de metodologia de projeto já é pressuposto no modelo. O uso de métodos de representação semi-formais (diagramas lógicos, diagramas de estado, redes de Petri, tabelas de decisão) não é explicitamente previsto, mas pela estrutura da sintaxe essas representações podem ser facilmente incluídas; já o uso de representações formais (notação matemática para descrição e

análise da especificação) não é previsto e, devido à complexidade das representações, pode ter sua aplicação limitada a componentes específicos do sistema. Entretanto, já existem trabalhos abordando a formalização dessas representações (TAYLOR, 2000);

- Identificação dos limites e descrição do domínio do sistema: essa característica pode ser atendida pela visão de empresa do ODP, onde são caracterizadas as necessidades do empreendimento, suas interfaces com o domínio, bem como são identificados os casos de uso do sistema. Considerando-se a inclusão dos requisitos de segurança, podem ser identificados também, no ponto de vista da empresa, os cenários específicos de segurança para viabilizar a análise desses requisitos;
- Decomposição lógica do sistema: a decomposição lógica do sistema é a característica principal do modelo ODP, abrangendo todos os cinco pontos de vista, em vários níveis de abstração. Em relação aos requisitos de segurança, essa decomposição deve objetivar a facilidade de testes, o reuso de componentes e a clareza das especificações, sendo esses objetivos cobertos pela decomposição das visões previstas do modelo ODP;
- Recursos para verificação da consistência, completeza e correção do projeto (V&V): esses recursos são apenas parcialmente explicitados no modelo ODP. Nos pontos de vista de informação, computação e engenharia (onde esses recursos deveriam ser efetivamente aplicados), os recursos de verificação de consistência e completeza definidos no modelo ODP limitam-se ao estabelecimento do conceito de regras de consistência e pontos de verificação de conformidade entre os requisitos das cinco visões do modelo, não especificando métodos para a realização desse tipo de atividades. Adicionalmente, o modelo não apresenta recursos para a verificação da consistência entre os requisitos, o que pode não ser adequado para a verificação e validação dos requisitos de segurança (um requisito do modelo computacional pode ser influenciado por um mecanismo definido no modelo de engenharia e vice-versa). Além disso, existe uma grande

possibilidade de existirem compromissos entre os requisitos de segurança e outros requisitos, que só são detectados com análises específicas de segurança (não previstas no modelo). O aspecto de completeza dos requisitos também não é coberto pelos conceitos do modelo, ficando a cargo do projetista a especificação e a forma de aplicação de métodos para essa finalidade. O aspecto de correção do projeto não é abordado no modelo ODP, uma vez que as técnicas empregadas nessa verificação normalmente exigem representações formais, não previstas no modelo;

- Gerenciamento dos requisitos do sistema: Os recursos para gerenciamento dos requisitos do sistema são apenas parcialmente atendidos no modelo ODP. Apesar da existência da especificação de características de qualidade de serviço (QoS) nos contratos do modelo ODP, não existem mecanismos específicos para controle e rastreabilidade desses requisitos. Dessa forma, o gerenciamento dos requisitos do sistema exige cuidados de modelagem e a implementação de mecanismos adicionais em todos os pontos de vista;
- CrITÉrios de qualidade de software: os dois principais critérios de qualidade de software, aplicados a sistemas relacionados com a segurança, são a simplicidade e a modularidade, sendo que ambos estão relacionados essencialmente com os pontos de vista de computação e engenharia. A simplicidade não é uma característica típica de sistemas distribuídos, uma vez que a mesma se traduz em: uso de interfaces diretas e limitadas entre componentes (incompatível com o uso de sistemas de comunicação distribuídos); tempo de resposta limitado verificável no código (de difícil obtenção quando os caminhos de interação entre componentes podem ser variados); código com as características mínimas requeridas (conflitante com as características de interoperabilidade, que exigem códigos adicionais para sua realização); código compreensível (conflitante com a tendência do uso de COTS, que são considerados como caixas pretas no sistema); facilidades de teste (que não é uma meta do modelo ODP) e portabilidade, sendo esta última a única característica dentre as anteriores que não conflita

com o modelo ODP. A modularidade é uma característica parcialmente atendida no modelo ODP, na medida em que, do ponto de vista da segurança, essa característica deve ter como consequências: confinamento funcional com elevada coesão e baixo acoplamento (a distribuição das funções entre nós diferentes da rede, com a necessidade de troca de informações entre eles - para prover transparência, distribuição de carga, gerenciamento de replicação - faz com que a interdependência entre os módulos seja elevada); além disso, as interfaces entre os módulos devem ser padronizadas (esta última característica é atendida no modelo ODP, na medida em que a mesma é necessária para a interoperabilidade);

- Medição dos critérios de qualidade: as métricas dos critérios de qualidade de software podem ser aplicadas a um sistema distribuído segundo o modelo ODP, sendo especialmente úteis nos pontos de vista de computação e engenharia. Entretanto, como alguns requisitos específicos de segurança (tais como: nível de correção, probabilidade de falha insegura e nível de criticidade) não são explícitos no modelo, o atendimento dessa característica é parcial, necessitando de complemento com ferramentas de análise de segurança específicas. Uma análise dos riscos e perigos envolvidos com o sistema deve ser realizada em todos os pontos de vista da arquitetura, em especial no ponto de vista da empresa e de informação, onde devem ser identificados os elementos críticos de segurança do sistema e estabelecida sua criticidade (prioridade para análise);
- Utilização exclusiva de construções especificadas na linguagem: essa característica aplica-se com mais relevância no ponto de vista de tecnologia, onde as linguagens são definidas. O critério de segurança, a ser atendido neste caso, é o uso de código simples e padronizado por normas. No modelo ODP não existe restrição à escolha de tipos de códigos, que é realizada no ponto de vista da tecnologia, uma vez que os demais pontos de vista não devem depender dessa definição. A tendência do projetista é usar códigos padronizados, já utilizados na organização a que o sistema

pertence, visando manter a consistência de recursos na plataforma de desenvolvimento e facilitar o reuso dos componentes. Entretanto, essa definição pode apresentar conflito com as características de segurança dos demais pontos de vista (por exemplo: necessidade de verificação formal de código, necessidade de prova de correção de função), devendo ser realizada uma análise da definição da tecnologia com relação aos requisitos de segurança dos outros pontos de vista do modelo. Como essa análise não está prevista no modelo, o atendimento a essa característica é parcial e pode ser conflitante com os outros critérios de definição de tecnologias;

- Utilização de construções de código independentes da implementação: essa característica também depende do ponto de vista da tecnologia, porém, o modelo ODP já preconiza a independência do ponto de vista da computação com relação a tecnologia, garantindo essa independência;

Em relação às técnicas das estratégias de tolerância a falha para projetos de segurança (tabela 4.2), tem-se a seguinte avaliação:

- Reparo de dados: essa característica (utilização de redundância nos dados) está ligada aos pontos de vista da computação e engenharia, sendo já coberta no modelo ODP pelo serviço de transparência de falhas e pelas funções de recuperação e transacional;
- Reconfiguração: essa característica está ligada aos pontos de vista da computação e engenharia, sendo já coberta no modelo ODP pelo serviço de transparência de falhas, transparência de persistência e pelas funções de gerenciamento de grupo, de recuperação, de replicação e transacional;
- Mascaramento de falhas: essa característica está associada aos pontos de vista da computação e engenharia, sendo coberta no modelo ODP pelo serviço de transparência de falhas, transparência de persistência, transparência transacional e pelas funções de recuperação, transacional e de integridade;

- Falha segura: no modelo ODP não existe a previsão de funções de falha segura. Os requisitos para esse tipo de funcionalidade deveriam estar presentes nos pontos de vista de empresa (definindo as características do domínio que caracterizam a falha segura), da informação (definindo o fluxo de dados nas condições de falha segura), da computação (definindo a lógica de detecção e condução do sistema à condição segura), da engenharia (definindo os mecanismos que garantem que o sistema é levado a essa condição) e eventualmente o de tecnologia (definindo o recurso que garante a condição de segurança). A inserção dessa característica no modelo exigiria a criação de funções específicas, não previstas na funcionalidade do modelo atual, sendo eventualmente conflitantes com os requisitos de disponibilidade e confiabilidade;
- Recuperação e retentativa: esses mecanismos estão previstos no modelo ODP, porém não atendem a todas às características normalmente exigidas nos mecanismos de segurança. Essas funções estão associadas, principalmente, aos pontos de vista de computação e de engenharia. Nas funções previstas no modelo já existem aquelas que cobrem a redundância temporal (transparência de persistência, função de replicação), mecanismos de repetição de operações anteriores (função de notificação de eventos, de recuperação, de auditoria) e mecanismos de reconfiguração a quente e a frio (de transparência de falhas, de replicação, transacional, funções de gerenciamento de grupo, de controle de configuração, de recuperação e transacional). Entretanto, a condução do sistema a um estado seguro não é garantida por esses mecanismos, uma vez que essa análise não está prevista no modelo (mecanismos do tipo recuperação de falha para condução do sistema a um estado seguro não existem);
- Recuperação por replicação: esses mecanismos são caracterizados pela execução de múltiplas entidades do sistema, com a verificação cruzada dos resultados (votação) para recuperação de estado. Esse tipo de mecanismo não está explicitamente previsto nas funções do ODP, podendo

eventualmente ser incluído entre as funções de coordenação (funções de replicação e de controle da configuração), resultando em requisitos mais específicos para os modelos de computação e engenharia;

- Recuperação por compensação: esses mecanismos objetivam cancelar a última ação que levou a um estado inseguro, considerando que o efeito da falha é transitório. Esse tipo de mecanismo é um caso particular do processo de recuperação e retentativa, sendo portanto parcialmente coberto pelo modelo ODP (associada às funções de notificação de eventos, de recuperação e de auditoria);
- Partição física: a alocação dos componentes críticos de segurança em hardware distinto é possível dentro de um sistema distribuído, portanto o modelo ODP pode atender a esse tipo de requisito. Entretanto, devem ser previstas, nas visões de empresa, informação, computação e engenharia, os requisitos para definir os critérios dessa alocação, bem como mecanismos para seu controle. Algumas funções existentes podem ser conflitantes com esse controle de alocação, como por exemplo a transparência de localização, transparência de migração, função de migração e função de realocação, que mascaram e tornam dinâmica a alocação física dos componentes para a aplicação. Dessa forma, a funcionalidade desses serviços devem ser orientados por requisitos de segurança;
- Partição lógica estática: o desacoplamento dos fluxos de dados e controle dos componentes críticos de segurança em relação aos demais componentes do sistema é um requisito potencialmente conflitante em relação a interoperabilidade e ao uso de *middleware* comuns entre sistemas distribuídos. Nessa medida o modelo ODP não prevê mecanismos para controle da segregação desses fluxos;
- Partição lógica dinâmica: os mecanismos de partição dinâmica são baseados em controle da autorização de acesso entre os componentes de segurança. Essas funções já estão previstas no modelo ODP (funções de

controle de acesso, de autenticação, de integridade, de gerenciamento de senhas), tanto na visão de computação como na de engenharia;

- Bloqueio de intertravamento: os mecanismos de restrições para limitação das interações entre os componentes não são funções previstas no modelo ODP. Para sua inclusão são necessários requisitos no ponto de vista da empresa (para identificação das restrições do domínio), da informação (restrições relativas ao fluxo de dados), da computação (definição das lógicas de intertravamento) e engenharia (definições dos mecanismos de intertravamento). Esse tipo de requisito é normalmente conflitante com a segregação de componentes, uma vez que os intertravamentos podem exigir diversas interações entre componentes distintos, ou mesmo podem depender do histórico de estados anteriores do sistema;
- Verificação por replicação: os mecanismos de detecção de erros por redundância ou repetição de operação já são considerados nas funções do modelo ODP. As funções de replicação e de autenticação já fornecem esses tipos de mecanismos;
- Verificação por temporização: os mecanismos de tempo máximo de função (*timeout*) e cão de guarda (*watchdog*) já são previstos na funcionalidade do ODP, uma vez que esses são utilizados nas funções de segurança de acesso (funções de controle de acesso, de autenticação e de integridade). Para atender aos mecanismos de segurança, essas funções devem ser estendidas para todos os objetos críticos de segurança;
- Verificação por reversão: os mecanismos de computação reversa não estão explicitamente presentes nas funções do modelo ODP. A inclusão desse tipo de funcionalidade pode ser conflitante com requisitos de eficiência e tempo de resposta do sistema distribuído e deve ser considerada nos modelos de informação (para definição dos pontos de verificação do fluxo de dados), no modelo de computação (para a definição dos algoritmos de

reversão) e no modelo de engenharia (para definição dos mecanismos genéricos associados);

- Verificação por codificação: os mecanismos de verificação de erros baseados em redundância de dados (marca d'água, paridade, *checksum*) já são previstos nas funções de integridade de informação do modelo ODP;
- Verificação por razoabilidade: a inclusão de mecanismos de verificação de erros por razoabilidade não está prevista na funções do modelo ODP. A definição desse tipo de mecanismo exige a elicitación das características das saídas do processamento em função do domínio (envolvendo as visões de empresa e de informação), bem como exige componentes de software especiais (visão de computação) e mecanismos associados (visão de engenharia). Esse tipo de requisito pode ser conflitante com o desempenho e tempo de resposta desejado para o sistema;
- Verificação por integridade estrutural: os mecanismos de verificação da estrutura de dados (criptografia, partição de código) já são previstos nas funções de controle de acesso, de autenticação, de integridade e de confidencialidade do modelo ODP;
- Verificação por diagnóstico: a verificação de erros por auto-teste já é considerada nas funções de transparência de falha, funções de replicação, de migração e de integridade do modelo ODP.

A tabela 6.1 apresenta um resumo das características discutidas neste capítulo, onde são comparadas as técnicas de segurança com a funcionalidade do modelo ODP. Nesta tabela foram preenchidas apenas as colunas correspondentes aos pontos de vista do modelo ODP que estão relacionados à técnica em questão (os outros pontos de vista podem até ser influenciados pela técnica, mas não são determinantes para sua aplicação). No preenchimento da tabela foi utilizada a seguinte notação:

- A: A técnica de segurança é aderente ao modelo ODP, já estando prevista na funcionalidade do mesmo;

P: A técnica de segurança é parcialmente prevista pelo modelo ODP ou sua adequação não causa potenciais compromissos com outras características do modelo;

NC: A técnica de segurança não está considerada no modelo ODP ou pode estar parcialmente prevista, mas a consideração da mesma pode ser potencialmente conflitante com outras características do modelo.

As células não preenchidas da tabela 6.1 (células em branco) correspondem aos pontos de vista que tem pouca (ou nenhuma) influência em relação ao requisito ou a técnica de segurança considerada.

A análise dos resultados da tabela 6.1 mostra que a maioria das técnicas de segurança pode ser adequadamente modelada no ODP. Entretanto, cerca de 26% das técnicas de segurança discutidas não são explicitamente consideradas no modelo ODP, na medida em que estas não estão previstas nos conceitos desse modelo (ou nos serviços previstos). Além disso, quando estas forem adequadamente modeladas, podem gerar compromissos com relação a outros mecanismos ou requisitos que já existem no modelo.

Observa-se na tabela que as técnicas de prevenção de falha são quase que totalmente satisfeitas pelo modelo ODP, a exceção dos critérios de qualidade de software, onde a simplicidade adequada a segurança não é observada; a escolha de uma tecnologia, pode ser eventualmente conflitante com os critérios de segurança, se uma análise destes não for considerada nessa definição.

Tabela 6.1: Adequação entre as técnicas de segurança e modelo ODP

<div><div></div><div>Ponto de Vista ODP</div></div>		Empresa	Informação	Computação	Engenharia	Tecnologia
Requisito ou Técnica de Segurança						
PREVENÇÃO DE FALHA	Notação de especificação definida	A	A	A	A	A
	Identificação dos limites do domínio	A				
	Decomposição lógica do sistema	A	A	A	A	A
	Recursos para verificação da consistência, completeza e correção do projeto (V&V)		P	P	P	
	Gerenciamento dos requisitos	P	P	P	P	P
	Critérios de qualidade de software (especialmente simplicidade e modularidade)			NC	NC	
	Medição dos critérios de qualidade	P	P	P	P	
	Utilização de construções especificadas na linguagem (uso de linguagens padronizadas)					NC
	Utilização de construções de código independentes da implementação			A		A
TOLERÂNCIA A FALHA	Reparo de dados			A	A	
	Reconfiguração			A	A	
	Mascaramento de falhas			A	A	
	Falha segura	NC	NC	NC	NC	NC
	Recuperação e retentativa			P	P	
	Recuperação por replicação			P	P	
	Recuperação por compensação			P	P	
	Partição física	P	P	P	P	
	Partição lógica estática			NC	NC	
	Partição lógica dinâmica			A	A	
	Bloqueio de intertravamento	NC	NC	NC	NC	
	Verificação por replicação			A	A	
	Verificação por temporização			A	A	
	Verificação por reversão		NC	NC	NC	
	Verificação por codificação			A	A	
	Verificação por razoabilidade	NC	NC	NC	NC	
	Verificação por integridade estrutural			A	A	
Verificação por diagnóstico			A	A		
Legenda: A: Adequado, P: Parcialmente adequado, NC: Não Considerado						

Esses resultados podem ser discutíveis, pois foram consideradas várias técnicas de segurança que não são necessariamente aplicáveis simultaneamente em um mesmo projeto e, portanto, um sistema relacionado com a segurança talvez possa ser adequadamente modelado com ODP sem a necessidade de grandes alterações no modelo. Entretanto, em função da discussão realizada nesta seção (independente do número de técnicas consideradas), pode-se concluir que a modelagem de sistemas relacionados com a segurança exige que os pontos de vista, principalmente o de empresa e de informação, sejam analisados com relação aos requisitos segurança, pois o tratamento completo desse tipo de requisito não é explicitamente previsto no modelo ODP. Além disso, a funcionalidade prevista nesse modelo precisa ser expandida para contemplar esse tipo de requisitos, potencialmente conflitantes com outros RNF típicos de sistemas abertos e distribuídos (tais como: interoperabilidade, disponibilidade e tolerância a falhas).

Dessa forma, pode-se considerar que é possível a adequação do modelo ODP a sistemas relacionados com a segurança, desde que sejam utilizadas técnicas de análise de segurança adicionais (não previstas no modelo ODP) para avaliação dos requisitos de segurança e seus eventuais conflitos com outros requisitos do sistema.

6.2 Metodologia de Elicitação e Análise de Requisitos de Segurança

Em função dos resultados observados na seção anterior, propõe-se nesta seção uma metodologia de elicitação e análise de requisitos de segurança. Essa metodologia considera um ciclo de desenvolvimento em espiral, tendo como bases: o método ATAM, para o tratamento de compromissos entre os requisitos de segurança e os demais requisitos do sistema; o modelo ODP para a modelagem da arquitetura dos sistemas abertos e distribuídos; e a utilização de um conjunto de atividades específico para elicitação e análise dos requisitos de segurança, sendo essas atividades independentes do processo de elicitação dos demais requisitos do sistema.

A escolha do modelo ODP para a descrição da arquitetura é consistente com as recomendações da norma IEEE-Std 1471:2000 (sobre práticas recomendadas para

descrição da arquitetura de sistemas com grande volume de software) (IEEE-STD-1471, 2000). Além disso, a escolha do modelo ODP para descrever a arquitetura do sistema é motivada pelas seguintes considerações:

- Descrição da arquitetura do sistema com refinamentos sucessivos, permitindo o desenvolvimento do sistema em espiral evolutiva, com avaliação das características em cada iteração do ciclo de desenvolvimento da arquitetura, facilitando a elicitação e análise dos requisitos do sistema;
- Descrição da arquitetura através de múltiplos pontos de vista, cada um associado a um grupo de *stakeholders* específico; realizando, assim, um detalhamento mais preciso das diversas necessidades do sistema (devido à focalização de cada descrição em um conjunto específico de necessidades), que se complementam, sem a necessidade da criação de um grande modelo global. Esse enfoque deve aumentar a cobertura do processo de elicitação dos requisitos, bem como facilitar a identificação de conflitos entre requisitos, quando esses diferentes pontos de vista são comparados;
- Processo de modelagem de arquitetura fortemente baseado na identificação inicial dos requisitos (modelo de negócio) antes das definições tecnológicas, facilitando o crescimento evolutivo da arquitetura em torno dos requisitos;
- Como o foco de cada ponto de vista é voltado para um tipo de *stakeholder* (utilizando representações e linguagem mais próximas de cada tipo de usuário), a utilização desse modelo facilita a comunicação entre o cliente e os desenvolvedores do sistema. Com a redução das falhas de comunicação, os requisitos são melhor definidos e melhor compreendidos, reduzindo o custo do retrabalho e o tempo de desenvolvimento do sistema;
- Finalmente, o ODP possui um conjunto de conceitos padronizados e consistente (definido em normas), cuja consistência já foi formalmente verificada (TAYLOR, 2000). Assim, a modelagem obtida (padrões de arquitetura, soluções adotadas) pode ser reutilizada, permitindo a construção

de meta-modelos, que podem ser utilizados em sistemas similares (geração de uma família de produtos) (STEEN, 1999).

Neste trabalho, todas essas características são consideradas na metodologia proposta que, nas suas diversas etapas, utiliza os produtos do modelo ODP para elicitação e análise dos requisitos (sejam eles funcionais ou não funcionais). Para a elicitação dos requisitos de segurança, o enfoque do modelo ODP é especialmente útil porque, além da adequação funcional, os pontos de vista de empresa e de informação permitem obter informações sobre as interfaces do sistema com o processo, que são difíceis de serem observadas pelo analista de segurança. Esses pontos de vista são essencialmente baseados na concepção dos usuários do processo que, normalmente, possuem um excelente conhecimento do domínio do sistema.

Apesar do modelo ODP fornecer os conceitos para descrição dos vários pontos de vista da arquitetura, bem como para seu sucessivo refinamento, este modelo não apresenta critérios para análise das características dessas arquiteturas, que possam orientar o seu detalhamento em função dos requisitos identificados. Neste trabalho, propõe-se a utilização de análises de cenários e avaliação de sensibilidade para o detalhamento e verificação dos requisitos (no caso dos requisitos de segurança essa avaliação de sensibilidade é baseada em análises de risco com relação à segurança).

Para a realização da análise de cenários e de sensibilidade dos requisitos da arquitetura, é utilizado como referência o método de análise de compromissos de arquitetura ATAM (*Architecture Tradeoff Analysis Method*), proposto pelo SEI (*Software Engineering Institute - Carnegie Mellon University*), apresentado na seção 2.4 (KAZMAN, 1998); (KAZMAN, 2000); (BASS, 2001).

A escolha desse método deve-se aos seguintes aspectos:

- O ATAM utiliza um método espiral de desenvolvimento de sistema, centrado na sua arquitetura. Esse método considera um conjunto inicial de possibilidades de arquitetura (arquiteturas candidatas), que vão sendo submetidas a avaliações e refinamentos sucessivos. Esse processo é aderente

ao modelo ODP, que permite a modelagem de arquiteturas de modo incremental;

- A cada iteração do ATAM, as características de cada arquitetura candidata são avaliadas, de modo qualitativo. Essa avaliação tem, como objetivos, identificar dentre as arquiteturas consideradas qual que possui menor risco em relação à satisfação dos requisitos de qualidade e verificar a existência de compromissos (*tradeoff analysis*) entre esses requisitos, que devam ser analisados. Essa avaliação qualitativa é realizada estimando-se a sensibilidade de cada arquitetura em relação aos requisitos de qualidade. Na metodologia proposta neste trabalho, utiliza-se o mesmo princípio de análise de sensibilidades para a identificação dos compromissos entre os requisitos, porém, o cálculo dessas sensibilidades é realizado de modo diferente, não tendo como objetivo direto a verificação qualitativa da satisfação dos requisitos (que é realizada por técnicas de análise de cenários ou ferramentas de análise específicas para cada RNF);
- A avaliação qualitativa da sensibilidade dos requisitos de qualidade de cada arquitetura é baseada em análises de tendências, que podem ser realizadas nas etapas iniciais do ciclo de desenvolvimento do sistema. Essa característica também é utilizada na metodologia proposta neste trabalho, uma vez que esta considera que a análise dos requisitos deve ocorrer a partir dos primeiros níveis de abstração das arquiteturas, obtidos nas etapas iniciais do desenvolvimento do sistema;
- No ATAM, a análise dos compromissos entre os requisitos, em cada iteração do método, é realizada através de uma série de avaliações, individualizadas por requisitos que, ao final de cada ciclo de iteração, são reunidas e analisadas em conjunto para identificação dos compromissos entre os diferentes requisitos, levando ao refinamento sucessivo da arquitetura do sistema, bem como dos seus requisitos. Essa mesma filosofia é adotada na metodologia proposta neste trabalho, sendo que são utilizados

os mesmos tipos de etapas do ATAM para a análise dos requisitos não relacionados com a segurança;

- A execução de cada iteração do ATAM é realizada em curto espaço de tempo (tipicamente 2 a 3 dias), no qual são realizadas reuniões de apresentação das características do sistema, das arquiteturas candidatas e de discussão e análise dos cenários (*walkthrough*), envolvendo todos os *stakeholders*. Na metodologia proposta é utilizado o mesmo enfoque, porém, diferentemente do ATAM, o envolvimento dos vários *stakeholders* ocorre não apenas nas etapas de análise, mas também nas atividades de detalhamento das visões de arquitetura do sistema. Dessa forma, cada iteração da metodologia proposta tem, tipicamente, duração maior que a do ATAM (dependendo do porte do projeto pode chegar a meses);
- Para facilitar a análise de sensibilidade dos requisitos das arquiteturas candidatas o ATAM utiliza estilos de arquitetura baseados em atributos de qualidade, que fornecem padrões de avaliações de satisfação de requisitos para cada estilo de arquitetura. O ATAM utiliza os estilos de arquitetura fornecidos por um processo denominado de ABAS (*Attribute-based architectural Styles*). Como a metodologia proposta neste trabalho está orientada para arquiteturas distribuídas e abertas, o estilo básico da arquitetura já está definido; assim, no lugar dos estilos de arquitetura, são utilizados os pontos de vista do modelo ODP como arquiteturas candidatas.

A metodologia proposta deste trabalho utiliza uma extensão do ATAM, considerando a descrição da arquitetura do sistema, conforme os cinco pontos de vista do modelo ODP, e incorporando uma etapa explícita (e independente) de análise dos requisitos de segurança, cujo resultado pode modificar (em função da análise integrada e da análise de compromissos) os demais requisitos do sistema.

A figura 6.2 apresenta uma representação gráfica da metodologia de elicitação e análise de requisitos de segurança, onde no círculo central existem as sete etapas básicas de elicitação e análise dos requisitos não relacionados com a segurança

(similares às do método ATAM), contornadas por uma elipse de fundo acinzentado, que representa as atividades específicas para os requisitos de segurança.

A abordagem da associação de uma atividade específica de análise de segurança ao ciclo de desenvolvimento, para a identificação dos compromissos dos requisitos de segurança durante o detalhamento do sistema, é consistente com as orientações das normas que tratam do desenvolvimento de sistemas computacionais relacionados com a segurança (IEC-1508-1, 1997).

Esse mesmo enfoque já foi utilizada com vantagens na integração de um método de análise de requisitos baseado em pontos de vista, denominado VORD (*Viewpoint-Oriented Requirements Definition*), onde uma etapa de análise de segurança foi associada em paralelo com a avaliação de requisitos funcionais (KOTONYA, 1995).

Conforme representado na figura 6.2, a metodologia de elicitação e análise de requisitos de segurança proposta neste trabalho é composta de quatro fases principais, similares às atividades propostas no ATAM, que caracterizam-se por: identificação de cenários e elicitação de requisitos (fase I); descrição da arquitetura, segundo cada um dos cinco pontos de vista ODP, e construção dos cenários, baseados em casos de uso (fase II); construção dos modelos de arquitetura sobre os cenários e avaliação individualizada dos requisitos, com a determinação da sensibilidade de cada tipo de requisito em relação a cada visão de arquitetura (fase III); análise integrada da satisfação dos requisitos pela arquitetura, envolvendo os requisitos de todos os pontos de vista considerados, identificação e análise dos seus compromissos (ou conflitos) e solução desses compromissos (fase IV). Ao final dessas quatro fases, pode haver a necessidade de uma nova iteração dessa metodologia para o detalhamento da arquitetura e/ou a elicitação e análise de novos requisitos resultantes das avaliações realizadas na fase IV da metodologia.

Deve ser ressaltado na figura 6.2 que as atividades externas (de fundo acinzentado) são independentes das atividades do círculo central (etapas 1 a 7), ou seja não há necessidade de sincronismo entre cada uma das sete etapas de

desenvolvimento da arquitetura e as atividades de especificação e análise de requisitos de segurança. Entretanto, o final de ambos os grupos de atividades é sincronizado na etapa 7, onde realiza-se a análise de compromissos juntamente com a análise de segurança para a verificação da conformidade de uma iteração da metodologia.

Figura 6.2: Metodologia de elicitação e análise de requisitos de segurança

As atividades propostas nessa metodologia apresentam algumas diferenças importantes em relação às atividades do método ATAM, as quais consistem em:

- Utilizar os cinco pontos de vista do modelo ODP no lugar dos estilos de arquitetura baseados em atributos de qualidade (ABAS). Inicialmente, todos os pontos de vista são detalhados individualmente (fases I, II e III), e ao final do ciclo de desenvolvimento são analisados de modo conjunto (fase IV). Essa abordagem viabiliza a integração do modelo ODP ao método ATAM;
- Elicitar e analisar todos os tipos de requisitos funcionais do sistema utilizando os mesmos tipos de atividades (o método ATAM é focado apenas em requisitos de qualidade: desempenho, manutenibilidade, disponibilidade e segurança de acesso). Dessa forma, pode-se analisar os compromissos entre todos os tipos de requisitos (aumentando a cobertura da metodologia) e uniformiza-se a análise (agregando repetibilidade e rastreabilidade ao processo de elicitação de requisitos);
- Analisar as sensibilidades dos requisitos, para a identificação dos compromissos entre eles, utilizando um método de cálculo específico, orientado para identificação da criticidade do requisito em relação aos vários pontos de vista de arquitetura, sem o objetivo de verificação qualitativa da satisfação dos requisitos. Essa estratégia permite focalizar a metodologia nos requisitos mais importantes (mais críticos em relação aos detalhamento do conjunto das arquiteturas), e viabiliza que sejam utilizadas técnicas ou ferramentas de análise específicas mais adequadas para cada tipo de requisito (aspecto importante para análise de RNF);
- Envolver os vários *stakeholders* não apenas nas etapas de análise, mas também nas atividades de detalhamento dos pontos de vista de arquitetura, resultando em ciclos de elicitação e análise com duração maiores que os utilizados no ATAM. Apesar dessa estratégia aumentar o tempo de realização de cada ciclo de iteração, ganha-se na redução de erros de omissão e no aumento da compreensão das necessidades dos usuários (especialmente no início do ciclo de desenvolvimento do sistema);

- Modelar os RNF elicitados utilizando-se a modelagem de requisitos por objetivos (GRL), juntamente com os mapas de caso de uso (UCM) para representar os requisitos funcionais e os cenários causais. A modelagem integrada de objetivos e cenários melhora a completeza e robustez do processo de elicitação de requisitos (LIU, 2001); (AMYOT, 2002).

As atividades específicas de elicitação e análise de requisitos de segurança (representadas por uma elipse com fundo acinzentado na figura 6.2) seguem as mesmas quatro fases já discutidas. Essas atividades devem ser realizadas em paralelo com as atividades de elicitação e análise dos requisitos não relacionados com a segurança, porém de forma independente (sem mútua interferência), especialmente nas fases II e III. Na fase IV unificam-se os processos de análise, de modo que os vários requisitos analisados individualmente em cada ponto de vista na fase III são avaliados em conjunto, incluindo-se os requisitos de segurança. Essa análise integrada é essencial do ponto de vista da segurança, uma vez que o atendimento desse tipo de requisito não é função apenas dos componentes de software (visão computacional), dos mecanismos de infra-estrutura (visão de engenharia) ou do tipo de tecnologia utilizada, mas sim do conjunto de todas essas visões integradas, que podem apresentar requisitos conflitantes.

Assim, as quatro fases da metodologia para a elicitação e análise de requisitos de segurança caracterizam-se da seguinte forma:

- FASE I: paralelamente às atividades de identificação de objetivos, de cenários e da elicitação de requisitos do domínio (atividades para os requisitos não relacionados com a segurança), realiza-se a identificação dos perigos, dos objetivos e dos cenários de segurança, bem como a elicitação dos requisitos globais de segurança (que estabelecem as restrições básicas relacionadas com a segurança do sistema, geralmente provenientes da análise de falhas do processo ou de restrições do modelo do negócio);
- FASE II: paralelamente às atividade de descrição da arquitetura e construção dos cenários (para os requisitos não relacionados com a segurança),

realizam-se as atividades de identificação dos requisitos específicos de segurança (derivados dos requisitos globais de segurança e decompostos entre os pontos de vista do modelo ODP) e construção dos respectivos cenários de segurança (utilizando-se modelos de requisitos por objetivos, juntamente com os mapas de caso de uso, para a verificação da consistência do conjunto de requisitos de segurança);

- FASE III: paralelamente à construção do modelo das arquiteturas e avaliação dos seus requisitos, realizam-se as atividades de identificação de riscos, priorizando-se os cenários de segurança com relação à sua criticidade, e a análise de segurança para cada ponto de vista da arquitetura. Um aspecto importante das atividades dessa fase é a sua independência em relação às atividades de modelagem e avaliação dos requisitos não relacionados com a segurança. Essa independência permite que sejam utilizadas ferramentas dedicadas para análise de segurança, bem como permite a concentração dos analistas de segurança exclusivamente nos eventos inseguros, possibilitando um maior detalhamento dos cenários de segurança e maior cobertura na identificação de falhas inseguras;
- FASE IV: nessa fase deve ocorrer a análise conjunta de todos os requisitos (segurança e não segurança) para a verificação da consistência dos modelos, identificação dos compromissos e a subsequente análise e resolução dos potenciais conflitos. A avaliação desses compromissos, ou a identificação de requisitos importantes não atendidos, pode levar à modificação dos requisitos originais de cada ponto de vista, à criação de novos requisitos ou à modificações na arquitetura do sistema. Como decorrência desse processo, o ciclo de análise deve ser feito novamente (nova iteração) até que seja verificada a adequação de todos os requisitos de segurança.

A seqüência de atividades envolvidas nessas quatro fases pode ser subdividida nas sete etapas seguintes (numeradas no círculo interno da figura 6.2):

Etapa 1. Inicialmente devem ser identificados os objetivos do negócio e um conjunto de cenários que deve operacionalizar os requisitos funcionais e não funcionais (de qualidade) do sistema. Nessa etapa, é utilizado o modelo de negócio do sistema baseado nos pontos de vista de empresa e informação do modelo ODP. Esse processo deve contar com a intensiva participação dos *stakeholders*, visando facilitar a comunicação e criar uma visão comum a todos os envolvidos (especialmente entre os usuários, especialistas do domínio e os analistas de segurança), em relação às atividades importantes que o sistema deve executar. Nessa etapa, a visão do analista de segurança deve estar concentrada na identificação dos perigos e dos objetivos de segurança;

Etapa 2. Nesta etapa, executa-se o processo de elicitação dos requisitos iniciais do sistema (funcionais e não funcionais), tendo como base os contratos do modelo ODP, as definições dos envolvidos com o sistema e as limitações impostas pelo domínio do sistema (extraído do modelo de negócios do sistema). Novamente é essencial a intensiva participação dos *stakeholders*. Eventualmente, os cenários devem ser revisados para que os mesmos levem em consideração os atributos de qualidade que emergem no processo de elicitação e/ou devido ao refinamento dos objetivos importantes inicialmente identificados e que podem ser refinados, utilizando-se o detalhamento integrado entre objetivos e cenários (pode-se utilizar a descrição de mapas de caso de uso – UCM para a modelagem dos cenários que serão utilizados). Paralelamente, o processo de análise dos requisitos de segurança deve identificar os requisitos globais de segurança e os cenários de segurança associados, sendo que nesse processo também deve haver uma intensa interação entre o grupo de análise de segurança e os envolvidos com a definição do sistema, de modo a garantir a correta compreensão do problema e do ambiente de operação do sistema (especialmente do modelo mental do usuário);

- Etapa 3.** A partir dos requisitos, dos cenários e dos princípios de modelo ODP, geram-se as arquiteturas iniciais do sistema para cada ponto de vista desse modelo (podendo haver mais de uma arquitetura candidata para cada ponto de vista); essas arquiteturas candidatas são descritas em termos de elementos e propriedades que sejam relevantes para cada um dos atributos de qualidade, dentro dos cenários considerados. Paralelamente a essa atividade, realiza-se a identificação (elicitacão) dos requisitos específicos de segurança (aplicado a cada módulo ou a cada sub-módulo do sistema), sendo que esta atividade deve ser, preferencialmente, realizada por um grupo independente do grupo de desenvolvimento do projeto (visando o incremento da cobertura da análise de segurança);
- Etapa 4.** Esta etapa corresponde ao final da fase II e caracteriza-se pela construção dos cenários utilizando os modelos de mapas de casos de uso (UCM), associados aos modelos de objetivos (GRL), para verificação da cobertura desses cenários; nesse processo podem também ser utilizados protótipos ou recursos de simulação para teste desses cenários. Similarmente, na atividade de especificação de requisitos de segurança, os analistas de segurança constroem e verificam a validade dos cenários de segurança (nessa atividade de verificação também podem ser utilizados os modelos GRL e UCM aplicados aos requisitos específicos de segurança);
- Etapa 5.** Uma vez que os requisitos iniciais e os cenários estejam elicitados, e as arquiteturas iniciais de cada ponto de vista estejam detalhadas, analisa-se individualmente cada um dos atributos de qualidade, com relação a cada ponto de vista de arquitetura; essa análise tem como resultado a quantificação da satisfação desses atributos em relação a cada arquitetura candidata. Paralelamente, realizam-se as análises de risco e de segurança (utilizando-se ferramentas do tipo FTA, HAZOP e PRA) pelo grupo de análise de segurança, considerando-se os objetivos de segurança e as características das arquiteturas propostas. Nessa etapa, não é realizada

nenhuma crítica sobre a interação entre requisitos de classes diferentes (por exemplo, entre critérios de segurança e de disponibilidade), de modo a otimizar o entendimento e a avaliação individual de cada tipo de especialista envolvido na análise dos requisitos do sistema;

Etapas 6. Nesta etapa, realiza-se uma análise integrada dos requisitos elicitados em todos os pontos de vista, incluindo-se os requisitos e as recomendações de segurança, provenientes da respectiva análise realizada na etapa anterior. A análise é realizada utilizando-se ferramentas clássicas de revisão de projeto (*design review*, *walkthrough*) e objetiva identificar os compromissos entre os requisitos dos cinco pontos de vista do modelo, bem como pode ser utilizada para a escolha de alternativas de arquitetura. Para facilitar a análise integrada, avalia-se a sensibilidade de cada requisito em relação aos elementos da arquitetura, ou seja, o grau de interferência das alterações de cada modelo de arquitetura em relação à satisfação de cada requisito. Qualquer requisito que seja significativamente alterado por mudança na arquitetura é considerado um ponto de sensibilidade. Utilizam-se essas avaliações de sensibilidade para determinarem-se as possíveis mudanças de arquitetura, necessárias para a satisfação dos respectivos requisitos, bem como para identificarem-se os requisitos que possuem maiores compromissos entre todos os pontos de vista de arquitetura, permitindo a definição de uma priorização para sua análise. Nessa etapa, o grupo de análise de segurança deve atuar de modo integrado com o grupo de desenvolvimento do projeto, de modo a maximizar a identificação da interdependência entre os requisitos de segurança e os demais atributos do sistema. A modelagem dos requisitos por objetivos (utilizando o modelo GRL) pode auxiliar na identificação dessas sensibilidades e na identificação das interações entre os RNF. Do ponto de vista do gerenciamento do projeto, nesta etapa devem-se avaliar as opções de projeto e desenvolvimento, identificando-se os pontos estratégicos para o desenvolvimento do sistema, avaliando-se as dependências, custos e

benefícios de cada opção de arquitetura e elaborando-se as estratégias de desenvolvimento do sistema e migração a partir do sistema atual (quando existem sistemas legados);

Etapas 7. Nesta etapa, as arquiteturas modeladas na etapa 5 são verificadas para detectarem-se eventuais compromissos entre os requisitos de cada ponto de vista, focalizando-se os pontos de interação entre esses requisitos (por exemplo: identificando-se os elementos da arquitetura para os quais múltiplos requisitos são sensíveis), bem como para verificar-se a consistência entre os pontos de vista (a verificação dos requisitos comuns a vários pontos de vista é utilizada como teste de conformidade entre estes pontos de vista, conforme previsto no modelo ODP). A sensibilidade dos requisitos determinada na etapa anterior auxilia na solução dos conflitos. Essa sensibilidade indica, ao analista, quais mudanças de arquitetura podem ser mais efetivas, assim como auxilia na negociação dos conflitos entre os requisitos, quando existe a necessidade da avaliação do custo/benefício da mudança de algum requisito e/ou característica de arquitetura do sistema. Nesse sentido, mesmo os requisitos de segurança, que tipicamente são tratados como não negociáveis, precisam ser avaliados quanto a custo/risco/benefício, de modo a não inviabilizar o empreendimento (muitas vezes a viabilização de um requisito de segurança depende mais de ações compensatórias externas ao sistema do que de modificações na arquitetura do mesmo). Do ponto de vista do gerenciamento do projeto, nesta etapa deve-se elaborar o planejamento das atividades de desenvolvimento, o planejamento da migração do sistema (quando existir sistemas legados), estabelecendo-se recomendações, regras, contratos e evidências para assegurar que os requisitos do sistema estão sendo satisfeitos.

Se a análise de compromissos resultar na necessidade de alteração de requisitos (ou de mudanças na arquitetura), então deve iniciar-se um novo ciclo, onde podem

ser alterados os cenários, os requisitos, a arquitetura e os modelos. Pode ainda haver a identificação de outros pontos de compromisso ou conflito entre requisitos, caracterizando-se uma nova iteração no processo até que todos os conflitos sejam adequadamente solucionados e os requisitos estejam satisfeitos.

Para uma melhor compreensão da interação da metodologia proposta com o ciclo de vida de desenvolvimento do sistema, são representados na figura 6.3 as várias etapas da metodologia associadas a um ciclo de vida genérico de sistema. Nessa figura os retângulos numerados correspondem às etapas da metodologia (figura 6.2); os retângulos brancos representam as atividades do ciclo de vida, que não estão presentes explicitamente no núcleo na metodologia discutida, mas que são essenciais para o início do processo (como a identificação dos requisitos do negócio). Nessa figura são representadas linhas pontilhadas que identificam atividades que podem contribuir com mudanças de requisitos ao longo do ciclo de vida do sistema (tanto nas etapas de desenvolvimento como de utilização e manutenção do sistema), resultando em novo ciclo iterativo de elicitação e análise de requisitos.

As duas primeiras atividades do ciclo de desenvolvimento da figura 6.3 consistem na identificação do negócio (objetivos do negócio, comunidades envolvidas, atores, agentes, regras e práticas do negócio) e na representação desses requisitos (utilizando, por exemplo, os modelos de empresa e informação).

Nas atividades de desenvolvimento, utilização e manutenção, o processo de gerenciamento dos requisitos deve continuar sendo executado, uma vez que podem ocorrer mudanças de requisitos durante essas atividades (especialmente durante o desenvolvimento do sistema), que podem implicar em mudanças de arquitetura ou que podem exigir uma reanálise das soluções adotadas (esta última podendo ser necessária apenas em função do maior grau de conhecimento sobre o sistema, decorrente do detalhamento e/ou da utilização do mesmo).

Figura 6.3: Metodologia de segurança associada ao ciclo de vida do sistema

Na figura 6.3 pode ser observado que os resultados das diversas atividades de verificação e análise da metodologia podem gerar mudanças de requisitos. Além da etapa 7, onde provavelmente a atividade de análise de compromissos deve gerar mudanças nos requisitos (ou na arquitetura), outras atividades também podem motivar uma nova iteração na especificação de requisitos. Por exemplo, nas atividades de elicitação de requisitos, a verificação da completeza dos cenários e do atendimento dos objetivos do negócio pode gerar novos requisitos (utilizando-se, por exemplo, a técnica de acoplamento entre cenários e objetivos, conforme descrito na seção 2.5) (LIU, 2001); a construção de cenários pode revelar a necessidade de novos requisitos, decorrentes da evolução da arquitetura ou do detalhamento do sistema; as atividades de análise de cenários de segurança e a própria análise de segurança também são atividades que tipicamente geram alterações em requisitos (tanto nos RF como nos RNF).

Essa metodologia de elicitação e análise de requisitos considera que os cinco pontos de vista do modelo ODP devem ser desenvolvidos de modo individual, uma vez que cada um deles possui o seu próprio modelo de arquitetura. Essa estratégia focaliza a análise dos requisitos em cada ponto de vista, assegurando um maior detalhamento e precisão na especificação destes. Considerando-se o modelo ODP, o desenvolvimento dos cinco pontos de vista pode ocorrer em paralelo; entretanto, o processo de detalhamento dos requisitos sugere a existência de uma hierarquia entre esses pontos de vista, uma vez que os requisitos específicos de cada visão devem ser coerentes entre si, partindo dos requisitos globais do sistema, que são decompostos nos diversos pontos de vista, conforme representado na figura 6.4.

No processo de elicitação dos requisitos, a visão da empresa é a mais abstrata e define as necessidades do sistema dentro da organização. Essa visão estabelece os requisitos globais do sistema. As demais visões do modelo podem ter seus requisitos próprios, mas devem estar vinculados aos requisitos globais da organização, representando um refinamento destes.

Figura 6.4: Hierarquia dos requisitos entre as visões do modelo ODP

As visões de informação e computação consolidam (detalham) essencialmente os requisitos funcionais globais, definindo, respectivamente, o modelo de informação (projeto conceitual da arquitetura, fluxo de informações, detalhamento de casos de uso) e a arquitetura de software do sistema (objetos, interfaces, interações computacionais).

A visão de engenharia deve contemplar os requisitos funcionais e não funcionais do sistema, detalhando os recursos necessários para atender aos requisitos das visões anteriores (definição dos elementos de infra-estrutura computacional para atender às características da aplicação).

A visão de tecnologia identifica as técnicas e os recursos necessários para atender ao modelo de engenharia (detalhamento dos requisitos de engenharia).

Deve ser observado que a existência de uma hierarquia entre os requisitos não implica que os pontos de vista devam ser desenvolvidos como se fossem níveis sucessivos de detalhamento. Esse tipo de enfoque caracterizaria um ciclo de desenvolvimento em cascata, onde cada ponto de vista seria apenas um nível de abstração diferente. A metodologia proposta considera que todas as visões devem ser desenvolvidas em paralelo, sendo refinadas de modo incremental e independente, seguindo um modelo espiral de desenvolvimento do sistema.

Como cada ponto de vista ODP pode ser conduzido por um grupo de atores diferentes, é necessária a consolidação dos requisitos através da análise integrada dos mesmos. Essa análise é facilitada pela hierarquia observada entre os requisitos, uma vez que esta permite a verificação da consistência entre os diversos pontos de vista (através dos pontos de teste de conformidade do modelo ODP) e a resolução dos potenciais conflitos entre requisitos (fase IV do modelo proposto).

Na figura 6.4 observa-se que cada fase da metodologia proposta pode ser associada ao maior detalhamento de um ponto de vista ODP específico: a fase I da metodologia está mais associada ao ponto de vista da empresa; a fase II está mais associada aos pontos de vista de Informação e Computação; na fase III, os pontos de vista de Engenharia e Tecnologia são essenciais; na fase IV, todos os pontos de vista devem ser considerados em detalhe para a integração de todos os requisitos.

Na análise dos compromissos entre requisitos de segurança de visões distintas, podem ser utilizadas várias das técnicas específicas apresentadas na seção 4.3, dependendo do nível de detalhamento da arquitetura do sistema ou do tipo de objetivo a ser verificado. A tabela 6.2 apresenta um resumo com essas técnicas de análise e sua respectiva aplicabilidade em cada uma das fases da metodologia.

O modelo de análise apresentado está fortemente baseado no uso de cenários. Várias pesquisas tem mostrado que a utilização de cenários na determinação dos requisitos é extremamente recomendada, sendo verificado que, em muitos casos, o cenário pode ser reutilizado em domínios diferentes (RIDAO, 2001).

Tabela 6.2: Adequação das técnicas de análise à metodologia de segurança

Fase	Técnicas	Objetivos
I	FTA	☐ Identificação de cenários
	SFTA	☐ Identificação de interfaces críticas de segurança de software
	MORT	☐ Identificação de perigos associados ao software
	HAZOP	☐ Identificação de cenários
	AEA	☐ Identificação de perigos (especialmente em novos projetos)
	WSA	☐ Identificação de procedimentos operacionais de segurança
II	SMHA	☐ Modelagem do sistema ☐ Identificação de perigos associados ao software
	CCA	☐ Análise das seqüências temporais de eventos de segurança
	ETA	☐ Construção de cenários de acidente ☐ Identificação de eventos de topo para FTA
III e IV	HRA	☐ Análise de confiabilidade humana
	PRA	☐ Análise probabilística de risco
	FTA	☐ Análise qualitativa e quantitativa de segurança
	SFTA	
	FMEA	☐ Análise quantitativa de segurança
	FMECA	
	SFMECA	
	Check-List	☐ Verificação da completeza e precisão das especificações

Juntamente com os cenários, a utilização de casos de uso para identificação de perigos também é recomendada; entretanto, deve ser observado que sua utilização aplica-se bem aos requisitos funcionais, mas geralmente é deficiente para os requisitos não funcionais, que devem ser complementados com outros recursos de modelagem (como os modelos GRL) e com outras ferramentas de análise (como as listadas na tabela 6.2 para o caso de requisitos de segurança).

Se a análise mostrar que todos os compromissos entre requisitos estão solucionados, e que os requisitos estão atendidos em todos os pontos de vista da arquitetura, os projetistas podem prosseguir o desenvolvimento do sistema; entretanto, deve-se continuar a utilização do método de análise para o gerenciamento

(controle e rastreabilidade) dos requisitos com modelos analíticos durante as etapas de desenvolvimento, instalação e manutenção, sendo essa recomendação especialmente válida para os requisitos de segurança (KOTONYA, 1995).

Essa visão integrada dos requisitos de todos os pontos de vista permite a verificação dos compromissos com relação a cada ponto de vista do modelo ODP, assim como permite uma verificação cruzada entre os requisitos dos cinco pontos de vista. Adicionalmente, esse processo garante a rastreabilidade dos RNF, registrando a sua decomposição e correlação com os demais requisitos do sistema.

6.3 Adequação das Técnicas de Usabilidade ao RM-ODP

A satisfação dos atributos de usabilidade não está diretamente associada à arquitetura do sistema, dependendo muito mais dos aspectos da IHC do que das características internas do sistema que possui essa interface. Entretanto, um modelo de desenvolvimento que tenha como objetivo contemplar a satisfação dos requisitos de usabilidade, deve possuir recursos de modelagem que permitam representar e controlar esses atributos, e deve ser aderente ao ciclo de vida de usabilidade.

Nesta seção, é analisada a adequação dos atributos de usabilidade (seção 5.1) e do ciclo de vida de usabilidade (seção 5.2) em relação ao modelo ODP (capítulo 3).

Essa análise é realizada considerando-se um ambiente computacional distribuído, com as características previstas no modelo ODP, sem entrar no mérito das características específicas do domínio e da aplicação. Dessa forma, essa análise é qualitativa, identificando a existência de recursos de modelagem para a representação dos requisitos de usabilidade, bem como verificando a aderência do modelo ODP a um ciclo de vida de usabilidade típico. A tabela 6.3 apresenta a relação de atributos de usabilidade (proveniente da tabela 5.1), revisada de modo sistemático em relação ao modelo ODP, identificando como cada tipo de atributo de usabilidade pode ser representado nos pontos de vista desse modelo.

Tabela 6.3: Representação dos atributos de usabilidade no modelo ODP

Princípio	Atributo de Usabilidade	Representação no modelo ODP
Facilidade de Aprendizado	Previsibilidade	Depende da forma de apresentação das informações e dos tipos de respostas do sistema ao usuário, características especificadas nos pontos de vista de empresa e informação
	Capacidade de síntese	Depende do tipo de informação e do tipo de usuário, características especificadas nos pontos de vista empresa e de informação
	Familiaridade	Depende da forma de apresentação das informações do tipo de usuário, características relacionadas aos pontos de vista de empresa, informação e tecnologia
	Capacidade de generalização	Depende do tipo de informação e do tipo de usuário, características especificadas nos pontos de vista de empresa e informação
	Consistência	Depende da consistência da forma e do tipo de apresentação entre os vários objetos de informação (entre tarefas similares), características relativas aos pontos de vista de informação, empresa e computação (pode ser associada a característica de reutilização de objetos de software)
Flexibilidade	Iniciativa de diálogo	Depende do tipo de diálogo com o usuário e do balanceamento entre modos de operação e bloqueios de segurança, características especificadas nos pontos de vista de empresa, informação e computação
	Paralelismo	Depende da forma da forma de diálogo com o usuário, do escalonamento de tarefas e dos canais de comunicação disponíveis, características relacionadas a todos os pontos de vista ODP
	Capacidade de migração	Depende do tipo de diálogo com o usuário e do balanceamento entre modos de operação e bloqueios de segurança, características especificadas nos pontos de vista de empresa, informação e computação
	Capacidade de substituição	Depende da forma de apresentação das informações, característica especificada no ponto de vista de informação
	Capacidade de configuração	Depende da forma e do tipo de diálogo com os usuários e dos recursos dos objetos de informação, características relacionadas com os pontos de vista de empresa, informação e computação
Robustez	Observabilidade	Depende da forma e do tipo de diálogo com o usuário e dos recursos dos objetos de informação e de engenharia (para assegurar persistência e alcance), características relacionadas com os pontos de vista de empresa, informação, computação e engenharia
	Capacidade de recuperação	Depende da forma e do tipo de diálogo com o usuário e dos recursos dos objetos de informação e dos objetos de engenharia (para assegurar a recuperação de erros), características relacionadas com os pontos de vista de empresa, informação, computação e engenharia
	Capacidade de resposta	Depende das características dinâmicas da interface com o usuário e da estabilidade do sistema, características relacionadas com os pontos de vista de computação, engenharia e tecnologia
	Conformidade às tarefas	Depende da forma e do tipo de diálogo com o usuário, características relacionadas com os pontos de vista de empresa e informação

A tabela 6.4 apresenta um resumo das características apresentadas na tabela 6.3, onde, para cada atributo de usabilidade considerado, são identificados os pontos de vista do modelo ODP que podem especificá-lo. Essa tabela também pode ser interpretada como a relação de características que devem ser especificadas, em cada ponto de vista do modelo ODP, para que a usabilidade do sistema possa ser verificada.

Tabela 6.4: Associação dos atributos de usabilidade e o modelo ODP

Princípio de Usabilidade	Ponto de Vista ODP	Empresa	Informação	Computação	Engenharia	Tecnologia
	Atributo de Usabilidade					
Facilidade de Aprendizado	Previsibilidade	X	X			
	Capacidade de síntese	X	X			
	Familiaridade	X	X			X
	Capacidade de generalização	X	X			
	Consistência	X	X	X		
Flexibilidade	Iniciativa de diálogo	X	X	X		
	Paralelismo	X	X	X	X	X
	Capacidade de migração	X	X	X		
	Capacidade de substituição		X			
	Capacidade de configuração	X	X	X		
Robustez	Observabilidade	X	X	X	X	
	Capacidade de recuperação	X	X	X	X	
	Capacidade de resposta			X	X	X
	Conformidade às tarefas	X	X			

A análise dos resultados da tabela 6.4 mostra que todos os atributos de usabilidade podem ser adequadamente modeladas no ODP, em especial nos pontos de vista de informação e de empresa.

Essa conclusão pode ser facilmente justificada, considerando-se que a especificação do ponto de vista da informação deve apresentar quais informações devem ser apresentadas ao usuário, assim como o tratamento dessas informações para apresentação ao usuário (através da definição dos objetos de informação e das ações permitidas de mudança de estado desse objeto).

O ponto de vista da empresa complementa a especificação da IHC definindo as características dos usuários, do seu ambiente de operação, quais ações devem ocorrer entre os objetos para fornecer as informações necessárias à operação do sistema e quais as ações que o usuário pode executar (os objetos de empresa podem incluir os usuários e suas relações com os outros objetos do modelo).

O ponto de vista de computação também contribui para as características de usabilidade, na medida em que determinados atributos de usabilidade (principalmente os relacionados aos princípios de flexibilidade e robustez) exigem que a arquitetura de software tenha objetos de computação com características específicas (por exemplo: suporte ao escalonamento de tarefas, comunicação simultânea entre vários processos, restrições dinâmicas de sinalização e comando, mudanças de modo de operação, sinalização de erros e falhas, estruturas para recuperação de estados anteriores, estabilidade de tempos de resposta).

O ponto de vista da engenharia é menos exigido pelos atributos de usabilidade; entretanto, algumas características de usabilidade (tais como o paralelismo de execução de tarefas, a observabilidade dos estados internos do sistema, a capacidade de recuperação de falhas do sistema e a capacidade de resposta do sistema) podem exigir que os objetos de engenharia tenham características especiais (tais como: multiprocessamento, múltiplos canais de comunicação, registro de falhas, tolerância a falhas, tempos reduzidos de chaveamento, iniciação e desligamento, requisitos de resposta em tempo real).

O ponto de vista da tecnologia também pode contribuir para alguns atributos de usabilidade, especialmente nos aspectos relacionados a familiarização do usuário com novas tecnologias (numa migração de sistema pode ser necessário manter

alguma tecnologia existente para facilitar o aprendizado do usuário), a realização de várias tarefas em paralelo e a capacidade de resposta do sistema (características que podem impor restrições à escolha de tecnologias).

Dessa forma, os atributos de usabilidade podem ser especificados através dos pontos de vista do modelo ODP, permitindo assegurar que é possível uma completa caracterização dos atributos de usabilidade através desse modelo. Apesar disso, o modelo ODP não especifica critérios ou métodos para a avaliação dessas características.

A avaliação das características de usabilidade deve ser realizada em todos os pontos de vista do modelo (conforme observado nos resultados da tabela 6.4), sendo que nesse processo podem ser utilizados os métodos apresentados na seção 5.2.2 (tabela 5.2). Entretanto, a aplicação desses métodos deve seguir um ciclo de vida de usabilidade, uma vez que a escolha do método depende dos objetivos da avaliação e do nível de detalhamento da interface homem-computador que evolui juntamente com a arquitetura do sistema.

Neste trabalho adota-se, como referência, o ciclo de vida de usabilidade elaborado por Mayhew (1999) (apresentado na seção 5.2.1). Esse modelo é utilizado como referência neste trabalho, porque o mesmo evidencia algumas características que facilitam a visualização da integração do tratamento dos requisitos de usabilidade às atividades de tratamento de outros requisitos não funcionais do sistema (MAYHEW, 1999). Essas características são:

- Sistematização da elicitação de requisitos de usabilidade desde o início do processo de desenvolvimento: o ciclo de vida (apresentado na figura 5.2) parte da análise do contexto do negócio, com o foco centrado nos usuários e suas tarefas, identificando os objetivos globais de usabilidade do empreendimento. Essa sistematização pode (e deve) levar em conta outros tipos de requisitos que podem influenciar na operação do sistema (como requisitos de segurança, confiabilidade, manutenibilidade, desempenho),

que podem ter impacto na análise de tarefas e/ou podem fazer parte dos princípios gerais de projeto;

- Processo de desenvolvimento com refinamentos sucessivos e análise de custo/benefício, baseado em testes quantitativos de usabilidade: essa característica é aderente ao modelo de desenvolvimento de arquitetura, em espiral evolutiva, com avaliação das características em cada iteração do ciclo, adotado como referência neste trabalho (ATAM) (KAZMAN, 1998);
- Processo de desenvolvimento fortemente baseado no esforço inicial de análise de requisitos, orientado pelos requisitos do usuário e pelo projeto da IHC: esse enfoque é perfeitamente aderente aos métodos e modelos de desenvolvimento já discutidos (tanto o ATAM como o modelo ODP) (FAROOQUI, 1995); (KAZMAN, 1998); (HARTSON, 2004);
- Ajuste flexível da integração das atividades de desenvolvimento de software com a engenharia de usabilidade: no ciclo de vida de usabilidade discutido, a interface com a engenharia de desenvolvimento de software não é claramente definida; algumas atividades ocorrem em paralelo (como a análise de tarefas, a maquete do modelo conceitual, as avaliações iterativas de usabilidade) e outras podem ser consideradas sobrepostas (como o projeto do modelo conceitual, que pode fazer parte do modelo de análise ou da arquitetura da empresa, a definição de normas de projeto de telas que pode fazer parte do modelo de projeto). É evidente que ambos os processos estão intimamente ligados, sendo que esse ajuste pode ser específico para cada tipo de projeto e metodologia utilizada. Assim, essa característica facilita a adequação desse processo a uma metodologia de especificação de requisitos mais abrangente (MAYHEW, 1999), (HARTSON, 2004);
- Processo de desenvolvimento estruturado de cima para baixo (modelo *top-down*): o processo de projeto, desenvolvimento e teste é estruturado em três níveis. Inicialmente são tratadas as definições de alto nível do projeto (projeto do modelo conceitual da interface), seguido do detalhamento de

padrões (definição de normas de projeto de telas), finalmente complementado com o detalhamento do projeto. Essa mesma estratégia é adotada no modelo ODP, utilizando-se níveis de abstração progressivamente mais detalhados (PUTMAN, 2001);

- Divisão do desenvolvimento em subconjuntos de funções: o ciclo de vida de usabilidade apresentado propõe que cada iteração do ciclo projeto/desenvolvimento/teste seja conduzido em relação a um subconjunto de funções, sendo na próxima iteração aplicado a outro conjunto de funções (até que toda a funcionalidade esteja coberta). Esse enfoque de desenvolvimento em camadas funcionais é similar ao adotado no modelo ODP, onde as análises são conduzidas em diversos níveis de abstração, utilizando visões diferentes para simplificar a análise e aumentar a cobertura de identificação de compromissos, de modo que os requisitos do sistema vão sendo detalhados e analisados de maneira progressiva (iterativamente) (PUTMAN, 2001);
- Distinção entre tarefas e técnicas: no ciclo de vida apresentado, há uma clara distinção entre tarefas (atividades a serem executadas) e a variedade de técnicas que podem ser utilizadas (o ciclo de vida não estabelece métodos exclusivos, apresentando algumas técnicas alternativas para cada atividade a ser executada). Com isso, esse ciclo de vida pode ser adaptado ao tamanho do projeto, à sua complexidade e ao tipo de metodologia de desenvolvimento utilizada. É importante ressaltar que todas as atividades do ciclo de vida são realizadas (seja com métodos robustos ou simplificados), dentro da filosofia de que é melhor fazer uma análise simplificada do que nenhuma. Essa característica de flexibilidade facilita a adaptação do ciclo de vida a uma nova metodologia de desenvolvimento (MAYHEW, 1999);
- Participação de grupos de projeto multidisciplinares durante todo o ciclo de desenvolvimento: o ciclo de vida de usabilidade apresentado propõe que todas as atividades previstas sejam realizadas por grupos, que contenham

especialistas de diversas modalidades, especialmente os grupos de usuários, na tomada das decisões principais do projeto (nos processos de avaliação de usabilidade, por exemplo), de modo a facilitar a comunicação e o entendimento entre as várias áreas envolvidas com o projeto. Essa estratégia integrada das várias especialidades é aderente ao objetivo de propor-se uma metodologia de especificação de requisitos que envolva vários tipos de requisitos de modo integrado, porém elicitando os requisitos do sistema a partir de visões específicas. A multidisciplinaridade é especialmente importante na análise de requisitos, para maximizar o grau de cobertura da identificação dos mesmos (por exemplo, seria interessante ter um especialista de usabilidade participando da análise de segurança, assim como seria interessante ter um especialista de segurança participando de avaliações de usabilidade).

Essas características permitem concluir que o ciclo de vida de usabilidade apresentado na figura 5.2 é aderente aos princípios básicos do modelo ODP e pode ser integrado a outros processos de análise de requisitos não funcionais.

6.4 Metodologia de Elicitação e Análise de Requisitos de Usabilidade

Em função dos resultados observados na seção anterior, propõe-se nesta seção uma metodologia de elicitação e análise de requisitos de usabilidade, utilizando-se um ciclo de desenvolvimento em espiral (conforme método ATAM) e tendo o modelo ODP como referência para a descrição da arquitetura do sistema.

A estratégia adotada para a proposição da metodologia de elicitação e análise de requisitos de usabilidade é utilizar, para os requisitos não associados com a usabilidade, as mesmas atividades propostas no núcleo da metodologia de elicitação e análise de requisitos de segurança (atividades enumeradas de 1 a 7 na figura 6.2), uma vez que estas podem atender a todos os RF, assim como àqueles RNF que não exigirem ferramentas de análise específicas.

Partindo-se das mesmas considerações discutidas na seção 6.2 (especialmente o fato do modelo ODP ser orientado pela arquitetura e o desenvolvimento do sistema ser espiral e centrado na arquitetura), a proposta para a metodologia de elicitação e análise de requisitos de usabilidade também consiste em expandir o método ATAM, considerando a descrição da arquitetura conforme o modelo ODP e incorporando um ciclo de atividades externo (e independente), para a avaliação dos requisitos de usabilidade, cujo resultado pode modificar (em função da análise integrada de requisitos e da análise de compromissos) os demais requisitos iniciais do sistema.

A figura 6.5 apresenta uma representação gráfica da metodologia de elicitação e análise de requisitos de usabilidade, onde no círculo central existem as sete etapas básicas de elicitação e análise dos requisitos não associados com a usabilidade, contornadas por uma elipse de fundo acinzentado, que representa as atividades específicas de elicitação e análise dos requisitos de usabilidade.

Conforme representado na figura 6.5, a metodologia de elicitação e análise de requisitos de usabilidade proposta neste trabalho é composta de quatro fases principais (análogas às da metodologia de elicitação e análise de requisitos de segurança), que se caracterizam por: identificação de cenários e elicitação de requisitos (fase I); descrição da arquitetura, segundo cada um dos cinco pontos de vista ODP, e construção dos cenários, baseados em casos de uso (fase II); construção dos modelos de arquitetura sobre os cenários e avaliação individualizada dos requisitos, com a determinação da sensibilidade de cada tipo de requisito em relação a cada visão de arquitetura (fase III); análise integrada da satisfação dos requisitos pela arquitetura, envolvendo os requisitos de todos os pontos de vista considerados, identificação e análise dos seus compromissos (ou conflitos) e solução desses compromissos (fase IV). Ao final dessas quatro fases, pode haver a necessidade de uma nova iteração, para o detalhamento da arquitetura e/ou a elicitação e a análise de novos requisitos resultantes das avaliações realizadas na fase IV da metodologia.

Deve ser ressaltado na figura 6.3 que as atividades externas (de fundo acinzentado) são independentes das atividades do círculo central (etapas 1 a 7), ou

seja não há necessidade de sincronismo entre cada uma das sete etapas de desenvolvimento da arquitetura e as atividades de especificação e análise de requisitos de usabilidade. Entretanto, o final de ambos os grupos de atividades é sincronizado na etapa 7, onde realiza-se a análise de compromissos juntamente com a avaliação de usabilidade para a verificação da conformidade de uma iteração da metodologia

Figura 6.5: Metodologia de elicitação e análise de requisitos de usabilidade

As atividades específicas de elicitación e análise de requisitos de usabilidade (representadas por uma elipse com fundo acinzentado na figura 6.5) seguem as mesmas quatro fases já discutidas. Essas atividades devem ser realizadas em paralelo com as atividades de elicitación e análise dos requisitos não relacionados com a usabilidade, porém de forma independente (sem mútua interferência), especialmente nas fases II e III, para assegurar a concentração dos analistas na avaliação desse tipo de requisitos. Na fase IV, unificam-se os processos de análise, de modo que os vários requisitos analisados individualmente em cada ponto de vista na fase III são avaliados em conjunto, incluindo os requisitos de usabilidade. Essa análise unificada permite a efetiva identificação dos compromissos dos requisitos de usabilidade em relação aos demais requisitos do sistema, principalmente os demais RNF, que usualmente apresentam conflitos entre si (especialmente os requisitos de segurança, disponibilidade e desempenho).

Assim, as quatro fases da metodologia para a elicitación e análise de requisitos de usabilidade caracterizam-se da seguinte forma:

- FASE I: paralelamente às atividades de identificação de objetivos, de cenários e da elicitación de requisitos do domínio (atividades para os requisitos não relacionados com a usabilidade), realizam-se as atividades de: identificação do perfil dos usuários do sistema (primários ou secundários), análise de tarefas (atividade essencial para compreensão do processo e para priorização dos critérios de avaliação de usabilidade) e especificação dos princípios gerais de projeto de usabilidade (baseada nas restrições e potencialidades características do ambiente de desenvolvimento disponível e em padrões e recomendações adotados pelo negócio);
- FASE II: paralelamente às atividades de descrição da arquitetura e construção dos cenários, realizam-se as atividades de: elicitación dos requisitos de usabilidade (para cada visão do modelo ODP), elaboração de um guia de estilo de usabilidade (único para todo o sistema, onde não só são registrados os princípios que devem orientar o projeto de usabilidade do

sistema, como também devem ser estabelecidas métricas para avaliação da aplicação desses princípios durante o desenvolvimento da IHC) e a atividade de reengenharia do trabalho (caracterizada pela análise da organização das tarefas dos usuários de modo a otimizar a inserção dos recursos de automação no ambiente de trabalho);

- FASE III: em paralelo com a construção dos modelos das arquiteturas e avaliação dos seus requisitos, realizam-se as atividades de: modelagem da IHC, prototipação da IHC e avaliação da usabilidade desses protótipos para cada ponto de vista da arquitetura (utilizando técnicas adequadas ao nível de detalhamento das arquiteturas);
- FASE IV: nesta fase, deve ocorrer a análise conjunta de todos os requisitos (funcionais e não funcionais) para a identificação dos potenciais compromissos e a subsequente análise e resolução dos potenciais conflitos. A resolução desses compromissos, ou a identificação de requisitos importantes não atendidos, pode levar à modificação dos requisitos originais de cada ponto de vista, à criação de novos requisitos ou à necessidade de modificações na arquitetura do sistema. Como decorrência desse processo, o ciclo de análise deve ser refeito novamente (nova iteração), até que seja verificada a adequação de todos os requisitos de usabilidade.

Analogamente à metodologia proposta para os requisitos de segurança, a seqüência de atividades envolvidas nas quatro fases da metodologia de elicitação e análise de requisitos de usabilidade pode ser subdividida nas sete etapas seguintes (numeradas no círculo interno da figura 6.5):

Etapas 1. Inicialmente são identificados os objetivos do negócio e um conjunto de cenários para operacionalização dos requisitos funcionais e não funcionais do sistema, sendo utilizado como referência, o modelo de negócio do sistema (baseado nos pontos de vista de empresa e informação do modelo ODP). Nessa etapa, o analista de usabilidade deve realizar a identificação do perfil dos usuários do sistema (nível de conhecimento, frequência de

uso, habilidades com utilização de computadores) e realizar a análise de tarefas contextual (descrição das tarefas usuais, padrões do fluxo de trabalho, identificação dos objetivos dos usuários), tendo como meta identificar os objetivos de usabilidade e o ambiente de operação. Nota-se que essas atividades do analista de usabilidade estão diretamente relacionadas com o objetivo principal de identificação de objetivos e cenários de operação, evidenciando a forte interação do analista de usabilidade com os demais envolvidos nas definições do sistema (*stakeholders*). Essa etapa do ciclo de vida de usabilidade é suportada por um conjunto de métodos de avaliação de usabilidade contextuais (estudo de campo pró-ativo, entrevista, questionário, grupo de foco e tempestade cerebral) que auxiliam na identificação do perfil do usuário e do ambiente de operação, e na compreensão do modelo mental do usuário;

Etapa 2. Nesta etapa, realiza-se o processo de elicitação dos requisitos iniciais do sistema (funcionais e não funcionais), tendo como base os contratos do modelo ODP, as definições dos envolvidos com o sistema e as limitações impostas pelo domínio do sistema (extraído do modelo de negócios do sistema). Eventualmente os cenários devem ser revisados para que os mesmos levem em consideração os atributos de qualidade que emergem no processo de elicitação e/ou devido ao refinamento dos objetivos importantes inicialmente identificados (por exemplo em decorrência da atividade de análise de tarefas ter detectado situações ainda não previstas nos cenários). As atividades do analista de usabilidade seguem em paralelo, identificando as restrições e potencialidades do ambiente de desenvolvimento (especificação dos recursos e limitações da IHC decorrentes da tecnologia selecionada para a plataforma do sistema e de recursos do ambiente de operação) e especificando os princípios gerais de projeto de usabilidade (recomendações da literatura, regras impostas pelo ambiente de operação, normas, padrões e recomendações adotados pelo negócio, contratos), considerando-se cada um dos cinco pontos de vista do

modelo ODP (apesar dessas informações estarem mais diretamente associadas às visões de empresa e de tecnologia);

Etapa 3. A partir dos requisitos, dos cenários e dos conceitos do modelo ODP, descrevem-se as arquiteturas iniciais do sistema para cada ponto de vista (podendo haver mais que uma arquitetura candidata para cada ponto de vista). Em paralelo, realiza-se a atividade de elicitación dos requisitos de usabilidade (definição dos objetivos de usabilidade associados a cada visão ODP), a partir da especificação do perfil do usuário e da análise de tarefas contextual. Essa atividade pode interagir com o processo de descrição da arquitetura, na medida em que os atributos de usabilidade podem ser dependentes das definições de arquitetura, assim como as características da arquitetura podem ser alteradas em função de atributos de usabilidade (especialmente nos pontos de vista de informação e tecnologia). Nesta etapa, inicia-se a consolidação dos requisitos de usabilidade através da elaboração de um guia de estilo de usabilidade (único para todas as visões do sistema), onde se definem os princípios que orientam o projeto de usabilidade e as métricas para sua avaliação;

Etapa 4. Esta etapa corresponde ao final da fase II e caracteriza-se pela construção dos cenários utilizando os modelos de mapas de casos de uso (UCM), associados aos modelos de objetivos (GRL), para verificação da cobertura desses cenários; nesse processo podem também ser utilizados protótipos ou recursos de simulação para teste desses cenários. Similarmente, na atividade de especificação de requisitos de usabilidade os analistas realizam a atividade de reengenharia do trabalho, que consiste na revisão da organização das tarefas e fluxo de trabalho, buscando simplificá-lo e otimizá-lo em função de novas características de automação fornecidas pelo sistema. Esta atividade pode ter um forte relacionamento com a construção dos cenários de operação, na medida que a definição dos casos de uso e a estrutura dos diagramas de seqüência de tarefas dependem da

organização do trabalho do usuário. Adicionalmente, em função de modificações realizadas na sequência do trabalho, pode ser necessária uma revisão da própria análise de tarefas, implicando na revisão da etapa 1 (nova iteração do ciclo de usabilidade);

Etapas 5. Nesta etapa realiza-se a avaliação individual de cada atributo de qualidade com relação a cada ponto de vista de arquitetura, tendo como resultado a quantificação da satisfação desses atributos em relação às arquiteturas candidatas. Paralelamente, o analista de usabilidade realiza a modelagem da IHC que, dependendo do estágio de desenvolvimento, pode ser um simples modelo conceitual, um projeto de telas padrão ou ainda um projeto detalhado dessa interface. Na sequência, realiza-se a prototipação do modelo da IHC, que também pode variar entre um protótipo de baixa fidelidade (caracterizado por sequências de cenários ou maquetes), para os estágios iniciais de desenvolvimento, e um protótipo de alta-fidelidade, para as etapas mais avançadas de desenvolvimento. Nessa etapa, inicia-se também a avaliação da usabilidade para cada ponto de vista do sistema, procurando identificar o atendimento dos objetivos de usabilidade através da utilização de métodos de inspeção e teste (tabela 5.3);

Etapas 6. Nesta etapa, realiza-se a análise integrada dos requisitos elicitados em todos os pontos de vista, incluindo-se os requisitos funcionais, não funcionais e as recomendações provenientes da análise de outros RNF. Nessa análise integrada, procura-se identificar a sensibilidade de cada requisito em relação aos elementos da arquitetura, ou seja, qual o impacto das alterações os modelos de arquitetura em relação a cada requisito; qualquer requisito analisado que é significativamente alterado por mudança na arquitetura é considerado um ponto de sensibilidade. Utilizam-se essas avaliações de sensibilidade para se determinar as possíveis mudanças de arquitetura, necessárias para a satisfação dos respectivos requisitos, bem como para se identificar os requisitos que

possuem maiores compromissos entre todos os pontos de vista de arquitetura, permitindo a definição de uma priorização para sua análise. A contribuição dos analistas de usabilidade nessa atividade consiste em avaliar, através de técnicas de inspeção de usabilidade (por exemplo: *walkthrough* pluralista, avaliação heurística ou inspeção de características), qual a sensibilidade da satisfação dos objetivos de usabilidade em relação às mudanças na arquitetura do sistema. Nessa etapa, o grupo de análise de usabilidade deve atuar de modo integrado com os demais grupos de desenvolvimento do projeto, de modo a maximizar a identificação de interdependência entre os requisitos de usabilidade e os demais atributos do sistema. Do ponto de vista do gerenciamento do projeto, nesta etapa devem ser avaliadas as opções de projeto e desenvolvimento, identificados os pontos estratégicos para o desenvolvimento do sistema, avaliadas as dependências, custos e benefícios de cada opção de arquitetura e elaboradas as estratégias de desenvolvimento do sistema (incluindo o desenvolvimento da IHC), e migração a partir do sistema atual (quando existem sistemas legados);

Etapa 7. Nesta etapa, as arquiteturas modeladas na etapa 5 são verificadas para detectar eventuais compromissos entre os requisitos de cada ponto de vista (focalizando-se os pontos de interação entre esses requisitos), bem como, para verificar a consistência entre os pontos de vista (utilizando-se testes de conformidade entre estes pontos de vista, conforme previsto no modelo ODP). A determinação da sensibilidade dos requisitos auxilia na solução dos conflitos, indicando ao analista quais mudanças de arquitetura podem ser mais efetivas, assim como auxilia na negociação dos conflitos entre os requisitos, quando existe a necessidade da avaliação do custo/benefício da mudança de algum requisito e/ou característica da arquitetura do sistema. Deve ser observado que o resultado das avaliações de usabilidade normalmente não interfere diretamente com os outros requisitos e, portanto, essa análise dificilmente gera conflitos entre requisitos.

Entretanto, se a avaliação identificar deficiências na satisfação dos objetivos de usabilidade, podem ser necessárias modificações na arquitetura do sistema (especialmente nos pontos de vista de informação e de computação), cujas conseqüências precisam ser verificadas, uma vez que estas podem gerar compromissos com outros requisitos (ou mesmo com outros atributos de usabilidade). Do ponto de vista do gerenciamento do projeto, nesta etapa devem-se elaborar o planejamento das atividades de desenvolvimento (por exemplo: para um novo ciclo de desenvolvimento da IHC), o planejamento da migração do sistema (para sistemas legados), as recomendações de evolução, as regras e os contratos de desenvolvimento, a atualização no guia de estilo e o registro de evidências para assegurar que os requisitos do sistema estão sendo satisfeitos.

Em função do resultado da análise de compromissos (que pode gerar a necessidade de alteração de requisitos ou de arquitetura), pode ser necessário o início de um novo ciclo de desenvolvimento, onde podem ser alterados os cenários, os requisitos, a arquitetura e os modelos, podendo haver a identificação de outros pontos de conflito, caracterizando-se uma nova iteração no processo até que todos os compromissos sejam adequadamente solucionados.

Durante o ciclo de vida de usabilidade, mesmo que não existam mudanças de requisitos ou arquitetura, podem ser necessárias novas iterações no ciclo de desenvolvimento para a evolução da IHC, podendo ainda ser realizadas várias iterações dentro de um mesmo estágio de desenvolvimento (por exemplo, a realização um protótipo com algumas funções da IHC para uma primeira avaliação de usabilidade, seguida do incremento do protótipo com mais funções e nova avaliação, até que um conjunto completo de atributos seja avaliado). Esse processo iterativo é apresentado na figura 6.6, onde a metodologia de elicitação e análise de requisitos de usabilidade é representada associada ao ciclo de vida do sistema.

Figura 6.6: Metodologia de usabilidade associada ao ciclo de vida do sistema

Essa representação é similar a da figura 6.3, onde os retângulos numerados correspondem às etapas da metodologia (figura 6.5); os retângulos brancos representam as atividades do ciclo de vida que não estão presentes explicitamente no núcleo na metodologia. As duas primeiras atividades consistem na identificação do negócio (objetivos do negócio, comunidades envolvidas, atores, agentes, regras e práticas) e na sua representação (utilizando, por exemplo, as visões de empresa e informação do modelo ODP). Nas atividades de desenvolvimento, utilização e manutenção, o processo de gerenciamento dos requisitos (envolvendo seu controle e rastreabilidade) deve continuar sendo executado, uma vez que podem ocorrer mudanças de requisitos durante essas atividades (por exemplo, alterações das necessidades dos usuários em função do desenvolvimento do sistema, necessidades de ampliação do sistema em uso ou alterações de tecnologia para evitar a obsolescência do sistema), que podem implicar em mudanças de arquitetura, gerando novas iterações no ciclo de vida de usabilidade.

As linhas pontilhadas finas da figura 6.6 (associadas a atividade de avaliação de usabilidade) representam o processo evolutivo de desenvolvimento da IHC, sem a alteração dos requisitos de usabilidade. As linhas pontilhadas grossas representam as possíveis mudanças de requisitos decorrentes das atividades de verificação e análise da metodologia. Além da etapa 7, onde a atividade de análise de compromissos usualmente gera mudanças nos requisitos (ou na arquitetura), outras atividades também podem motivar alterações nos requisitos, tais como: durante o processo de elicitação de requisitos a verificação da completeza dos cenários e do atendimento aos objetivos do negócio podem modificar os requisitos; a construção de cenários pode revelar a necessidade de novos requisitos (decorrentes da evolução da arquitetura ou do detalhamento do sistema); as atividades de análise de tarefas, identificação de restrições do ambiente de desenvolvimento e a reengenharia do trabalho podem modificar os cenários de operação; e a avaliação de usabilidade pode determinar a necessidade de novas iterações para evolução da IHC (levando ao detalhamento do projeto da IHC ou à revisão do manual de estilo).

As atividades mais críticas da metodologia apresentada são a de elicitação dos requisitos e de avaliação da usabilidade. Em ambos os casos, as ferramentas que auxiliam nessas atividades são os métodos de avaliação de usabilidade relacionados na tabela 5.3. A escolha do método mais adequado para cada fase da metodologia proposta é função do tipo de objetivo e dos recursos disponíveis (tais como: possibilidades de acesso aos usuários, nível de informação disponível, ferramentas de análise); entretanto, é possível estabelecer para cada fase um conjunto de métodos de usabilidade de mais provável aplicação (em função dos objetivos específicos da fase), conforme representado na tabela 6.5.

O processo de elicitação e análise de requisitos representado na figura 6.6 deve ser aplicado a cada um dos pontos de vista do modelo ODP, uma vez que cada ponto de vista possui o seu próprio modelo de arquitetura. As cinco visões do modelo de referência ODP são complementares, devendo portanto ser desenvolvidas paralelamente e de modo individual para assegurar a melhor escolha em relação aos requisitos específicos daquele ponto de vista. Entretanto, observa-se nas associações apresentadas na tabela 6.4 que o ponto de vista de informação é aquele que possui maior relevância para a definição dos atributos de usabilidade, seguido pelo ponto de vista de empresa. Assim, os processos iniciais de identificação de usuários, análise de tarefas, princípios de projeto de IHC e restrições do ambiente estão fortemente baseados nas visões de empresa e informação, confirmando a hierarquia entre os requisitos dos pontos de vista discutida na seção 6.2.

O modelo de especificação de requisitos de usabilidade está fortemente baseado no uso de cenários (de modo similar ao processo descrito na seção 6.2), o que é compatível com o ciclo de vida de usabilidade discutido, uma vez que este se baseia fortemente na análise de tarefas e na reengenharia do trabalho. Além disso, a utilização de cenários é compatível com os métodos de avaliação de usabilidade utilizados nas etapas iniciais de especificação de requisitos, que são métodos contextuais.

Tabela 6.5: Adequação das técnicas de avaliação à metodologia de usabilidade

Fase	Técnicas	Objetivos
I	Estudo de campo pró-ativo	☐ Identificação de usuários
	Observação de campo	☐ Análise de tarefas
	Entrevista	☐ Identificação de restrições do ambiente
	Entrevista contextual	☐ Identificação de procedimentos operacionais
	Questionário / <i>user survey</i>	☐ Identificação da linguagem do domínio
	Grupo de foco	
	Tempestade cerebral	
II	Estudo de campo pró-ativo	☐ Elicitação dos requisitos de usabilidade
	Observação de campo	☐ Reengenharia do trabalho
	Entrevista	
	Entrevista contextual	
	Questionário / <i>user survey</i>	
	Grupo de foco	
	Tempestade cerebral	
III e IV	Walkthrough pluralista	☐ Avaliação de requisitos em protótipos (estágio inicial do desenvolvimento do projeto da IHC)
	Walkthrough cognitivo	
	Avaliação heurística	☐ Avaliação de requisitos em protótipos (estágios intermediário e avançado do desenvolvimento do projeto da IHC)
	Lista de verificação	☐ Avaliação de requisitos em produtos
	Protocolo pensando alto	☐ Avaliação de requisitos em protótipos (estágio avançado do desenvolvimento do projeto da IHC)
	Protocolo perguntando-questões	☐ Avaliação de requisitos em produtos
	Aprendizado e co-descoberta	
	Inspeção de características	
	Método de ensino	☐ Avaliação de requisitos em protótipos (estágio final do desenvolvimento do projeto da IHC)
	Método treinador	☐ Avaliação de requisitos em produtos
	Teste retrospectivo	
	Medida de desempenho	
	Análise de registro de uso	

6.5 Descrição da MERUSA

Em função das similaridades observadas entre as metodologias de elicitação e análise de requisitos de segurança (seção 6.2) e de requisitos de usabilidade (seção 6.4) apresenta-se, nesta seção, a integração desses processos em uma metodologia unificada, denominada MERUSA (**M**etodologia de **E**specificação de **R**equisitos de **U**sabilidade e **S**egurança orientada para **A**rquitetura).

Os princípios básicos da MERUSA são os mesmos adotados para as metodologias relativas aos requisitos de segurança e de usabilidade:

- Processo de elicitação e análise dos requisitos, considerando um ciclo de desenvolvimento em espiral, centrado no desenvolvimento da arquitetura do sistema, com refinamentos sucessivos e análise de custo/benefício, baseado em análises quantitativas e qualitativas;
- Processo de evolução da arquitetura, identificação e análise de compromissos, tendo como referência o método ATAM (*Architecture Tradeoff Analysis Method*), proposto pelo SEI;
- Representação da arquitetura do sistema conforme os cinco pontos de vista do modelo de referência ODP (*Reference Model of Open Distributed Processing*), padronizado pela ISO/IEC 10746-1:1995;
- Atividade iterativa de elicitação de requisitos ao longo do processo de desenvolvimento, considerando a sincronização sistemática dos diversos tipos de requisitos (RF e RNF), que têm um ciclo de vida próprio, para identificação de eventuais conflitos entre estes;
- Modelagem dos requisitos de usabilidade e segurança elicitados, utilizando-se os modelos de requisitos por objetivos (GRL), juntamente com os mapas de caso de uso (UCM), para representar os requisitos funcionais e os cenários causais, de modo a permitir uma verificação cruzada entre objetivos e cenários, visando a melhoria da completeza e robustez do processo de elicitação de requisitos como um todo;

- Atividade de avaliação da satisfação de cada tipo de RNF, de modo individual realizada por especialistas. Essa atividade é independente do processo principal de avaliação de requisitos funcionais e utiliza métodos de avaliação próprios para cada tipo de RNF, de modo que o especialista de cada área possa maximizar a cobertura da sua análise específica;
- Atividade de análise integrada de todos os requisitos, baseada na verificação da sensibilidade dos requisitos às mudanças de arquitetura, utilizando métodos de avaliação específicos para cada tipo de RNF. A análise é realizada por especialistas, mas considerando as possíveis interferências dos demais requisitos nessa análise (análise de risco); dessa forma, fornece diretrizes para a comparação entre opções de arquitetura em relação ao atendimento de todos os requisitos em conjunto, bem como para o processo de análise de eventuais compromissos entre requisitos;
- Atividade de análise de compromissos baseada nas análises de risco, processo no qual eventuais conflitos entre requisitos são verificados; como resultado dessa crítica, os modelos de arquitetura podem ser refinados, modificados, revisados, ou eventualmente alguma mudança nos requisitos pode ser negociada em função do balanço custo/risco/benefício, resolvendo eventuais compromissos e levando ao refinamento sucessivo desses requisitos, juntamente com a definição da arquitetura do sistema;
- Processos iterativos de análise de requisitos centrados em tarefas e cenários de operação, tendo como ponto de partida o modelo de negócio do sistema, elaborado no início do processo, e considerando cada um dos cinco pontos de vista do modelo ODP;
- Participação de grupos de projeto multidisciplinares durante todo o ciclo de desenvolvimento, onde os especialistas das diversas modalidades (por tipo de RNF por exemplo) desenvolvem a maior parte de suas análises de modo independente. Entretanto, esses especialistas participam em conjunto das atividades relacionadas com a tomada de decisões principais do projeto,

tanto nas atividades iniciais de cada ciclo iterativo (elicitação de requisitos), como nas atividades finais de cada iteração (análise integrada e análise de compromissos). Desse modo, facilita-se a comunicação e o entendimento entre as várias áreas envolvidas com o projeto, maximizando o grau de cobertura de cada ferramenta de análise, promovendo a análise da inter-relação entre os requisitos para identificação dos compromissos e garantindo a consistência do conjunto da especificação de requisitos.

De modo análogo aos processos descritos nas seções 6.2 e 6.4 a MERUSA pode ser dividida em quatro fases principais (conforme representado na figura 6.7):

- FASE I: atividade de identificação de cenários e elicitação de requisitos do domínio (segundo cada uma das cinco visões ODP). Em paralelo, para cada tipo de RNF, existe a identificação das características específicas do ambiente (identificação dos perigos para segurança, identificação do usuário para usabilidade), bem como a identificação dos objetivos e cenários específicos de cada tipo de RNF;
- FASE II: atividade de descrição da arquitetura e construção dos cenários (baseados em casos de uso). Em paralelo, para cada tipo de RNF, realizam-se as atividades de elicitação dos requisitos específicos (referentes a cada uma das cinco visões do modelo ODP) e o registro das diretivas de projeto e de análise específicas (construção dos cenários de segurança, elaboração do guia de estilo de usabilidade, reengenharia do trabalho);
- FASE III: atividade de construção do modelo das arquiteturas e avaliação individualizada da sensibilidade dos seus requisitos (determinação da sensibilidade de cada tipo de requisito em relação às alternativas de arquitetura). Em paralelo, para cada tipo de RNF, realizam-se as atividades de modelagem, prototipação e análise específicas (identificação e análise de riscos, análise de segurança, modelagem e prototipação da IHC, análise de usabilidade) para cada ponto de vista da arquitetura;

Figura 6.7: Etapas da MERUSA considerando vários ciclos de iteração

- **FASE IV:** atividade de análise integrada de todos os tipos de requisitos para a identificação das características conflitantes e a subsequente análise dos compromissos e resolução de conflitos. Nessa fase, os grupos de análise de cada RNF específico realizam uma análise conjunta para verificação das interferências entre os diferentes modelos utilizados nas análises, visando a identificação de potenciais conflitos. A solução desses conflitos, ou a identificação de requisitos importantes não atendidos, pode levar à modificação dos requisitos originais de cada ponto de vista, à criação de

novos requisitos ou à necessidade de modificações na arquitetura do sistema, tendo como consequência a necessidade de nova iteração do ciclo de elicitação e análise de requisitos, até que seja verificada a adequação de todos os RNF do sistema.

6.6 Critérios de Avaliação de Sensibilidade e Priorização de Requisitos

Na fase III da MERUSA realiza-se uma avaliação sistemática dos requisitos originais em relação às características apresentadas nas várias visões da arquitetura. Na primeira iteração do ciclo de desenvolvimento, pode ser realizada apenas uma avaliação qualitativa dos requisitos em relação a cada ponto de vista de arquitetura (em função do elevado nível de abstração das primeiras iterações do processo de desenvolvimento). Essa avaliação deve considerar o nível de sensibilidade (denominado de **NS**) de cada requisito, em relação a possíveis alterações que podem ocorrer no detalhamento do específico ponto de vista de arquitetura do sistema.

Na MERUSA, propõe-se que a avaliação do nível de sensibilidade (**NS**) de cada requisito deve ser ponderado com relação ao grau de dependência da satisfação do mesmo em relação a cada específico ponto de vista de arquitetura, sendo essa avaliação realizada conforme os seguintes critérios:

- **Faixa de sensibilidade:** adota-se a faixa de **0** (pouco sensível) a **5** (muito sensível) para a avaliação qualitativa da sensibilidade;
- **Critérios básicos de sensibilidade:**
 - Um determinado requisito é considerado fortemente associado (**NS = 5** ou **4**) a um específico ponto de vista, se a sua implementação e avaliação dependerem essencialmente do detalhamento desse ponto de vista;
 - Um determinado requisito é considerado marginalmente associado (**NS = 3, 2** ou **1**) a um específico ponto de vista, se a sua implementação e avaliação não dependerem de maneira crítica do detalhamento deste

ponto de vista, porém, as definições adotadas no desenvolvimento da arquitetura desse ponto de vista podem ser relevantes para a satisfação desse requisito, exigindo que o requisito seja considerado no detalhamento do ponto de vista;

- Um determinado requisito é considerado independente (**NS = 0**) de um específico ponto de vista, se a sua implementação e avaliação não dependerem, de maneira alguma, do detalhamento desse ponto de vista, dispensando sua análise no detalhamento desse ponto de vista;

➤ **Corolário:** um requisito com nível de sensibilidade zero em relação a determinado ponto de vista de arquitetura não será considerado na análise desse ponto de vista.

Na fase IV da MERUSA os requisitos são analisados de modo conjunto, considerando os resultados das análises de cada RNF específico, realizadas nas etapas anteriores, bem como considerando-se todos os pontos de vista de arquitetura. O objetivo é se identificar os compromissos, potenciais conflitos ou lacunas no conjunto da especificação de requisitos, através da verificação cruzada dos resultados das diversas análises realizadas.

Esse trabalho de análise conjunta é orientado por requisito, de modo que cada requisito deve ser analisado contra todos os pontos de vista (tendo como objeto de verificação o projeto da arquitetura de cada ponto de vista). Dessa forma, é importante ordenar os requisitos por ordem de prioridade, de modo que sejam discutidos inicialmente aqueles requisitos de maior sensibilidade em relação aos vários pontos de vista do sistema.

Para definir uma priorização entre os requisitos a serem analisados utiliza-se na MERUSA uma classificação associada ao nível de sensibilidade (**NS**), definido anteriormente. Os critérios utilizados para priorização dos requisitos, tendo como referência o seu nível de sensibilidade, são os seguintes:

- **Definição:** Considera-se que nível de sensibilidade associado a cada determinado ponto de vista ($0 \leq NS \leq 5$) representa numericamente o valor da *densidade de energia* de cada requisito em relação a um determinado ponto de vista. Considerando-se que cada sensibilidade é uma variável aleatória, e utilizando-se a teoria de sistemas de comunicações, pode-se calcular a *densidade de energia* de cada requisito em relação a um ponto de vista, utilizando-se uma função de auto-correlação do tipo: $R_v(\tau) = \langle v(t), v(t-\tau) \rangle$ (onde $\langle v(t), v(t-\tau) \rangle$ representa o produto escalar dos sinais aleatórios $v(t)$ e $v(t-\tau)$) (CARLSON, 1975);
- **Cálculo:** Calcula-se a *densidade de energia* de cada requisito, utilizando-se a equação:

$$Er_i = (N_i + Z) * \sum_{j=1 (j \neq i)}^5 (N_j + Z)$$

Onde: Er_i = *densidade de energia* do requisito em relação ao ponto de vista i ;

N_i = nível de sensibilidade normalizado ($NS/5$) em relação ao ponto de vista i ($0 \leq N_i \leq 1$);

N_j = nível de sensibilidade normalizado ($NS/5$) em relação ao ponto de vista j ($0 \leq N_j \leq 1$);

Z = valor residual utilizado para evitar a multiplicação por zero, permitindo a discriminação de pequenos valores de energia, valendo 1% da variação de N_j ($Z = 0,01$);

$\sum$ = somatória dos níveis de sensibilidade normalizados em relação aos pontos de vista j (exceto quando $j = i$);

i, j = índices variando de 1 a 5 representando os cinco pontos de vista (1 = Empresa, 2 = Informação, 3 = Computação, 4 = Engenharia, 5 = Tecnologia).

- **Energia Média:** Para cada requisito, somam-se as *densidades de energia* (Er_i) obtidas em cada ponto de vista, resultando numa *energia média* do requisito ($\sum Er_i$). Esse valor é proporcional ao valor médio do nível de sensibilidade do requisito em relação aos cinco pontos de vista, ponderado

com o peso do requisito para cada ponto de vista. Dessa forma, destacam-se os requisitos que tem maior dependência em relação a maioria dos pontos de vista;

- **Priorização:** Os requisitos com maior *energia média* são aqueles prioritariamente analisados, uma vez que estes tem um maior potencial de interferência em todas as visões do sistema, sendo portanto potencialmente mais propensos a gerar compromissos entre as cinco visões do sistema

Assim, essa classificação da sensibilidade permite identificar quais são os requisitos mais dependentes do detalhamento de cada visão de arquitetura (não sendo estes, necessariamente, os mais importantes para uma visão específica, mas provavelmente sendo os dominantes do ponto de vista do conjunto das decisões de detalhamento), bem como permite identificar as interdependências entre as várias visões de arquitetura, em relação ao atendimento de cada requisito.

6.7 Critérios de Sincronismo

Cada elipse representada na figura 6.7 corresponde a uma iteração do ciclo de vida de cada tipo de requisito, que deve envolver todos os cinco pontos de vista do modelo ODP. Cada um desses processos pode evoluir independentemente, não existindo a obrigatoriedade de que todos os ciclos estejam perfeitamente sincronizados (por exemplo, enquanto se elaboram os requisitos de segurança, podem estar ocorrendo ciclos de prototipação de usabilidade). Podem ocorrer também várias iterações em um mesmo ciclo de vida (caso de um requisito que tenha um período de iteração reduzido), enquanto ocorre apenas uma iteração no ciclo de vida de outros requisitos. Entretanto, é obrigatório que, periodicamente, exista um ponto de verificação de consistência de todas os pontos de vista (por exemplo, tomando-se como referência o ciclo de vida de maior duração).

A fase IV da MERUSA caracteriza-se como um ponto de convergência dos diversos ciclos de desenvolvimento. Essa convergência é obtida através da análise

integrada dos requisitos, da identificação de compromissos e da solução de conflitos entre requisitos (garantindo a coerência do processo), da verificação da consistência do modelo entre os vários pontos de vista do sistema e da verificação da conformidade entre o modelo e os requisitos elicitados, gerando assim um sincronismo fraco, mas necessário, entre os vários ciclos de desenvolvimento.

O modelo ODP prevê a inclusão de pontos de referência em cada ponto de vista, para a verificação da consistência entre estes, denominados de pontos de testes de conformidade. Existem quatro tipos de pontos de teste de conformidade: ponto de referência programado, ponto de referência perceptível, ponto de referência de intercomunicação, ponto de referência de armazenamento (BECERRA, 1998). Na MERUSA, esses pontos de teste de conformidade são verificados ao final da fase IV, caracterizando o ponto de sincronismo entre os ciclos de elicitação e análise dos vários requisitos.

A figura 6.8 representa, esquematicamente, um exemplo da seqüência temporal que pode existir entre as iterações dos vários ciclos de vida envolvidos na MERUSA, considerando níveis progressivos de detalhamento do sistema até a sua entrada em operação (ou comissionamento para sistemas críticos de segurança). Deve ser observado que, após a entrada em operação, o sistema inicia também seu ciclo de manutenção. Mesmo nesta etapa do ciclo de vida, pode existir a necessidade de nova iteração no processo de especificação de requisitos e desenvolvimento da arquitetura, causada devido à necessidade de evolução do sistema, ou mesmo devido à detecção de erros ocultos no sistema.

Dependendo da profundidade das alterações necessárias no sistema, podem ser necessárias várias iterações na MERUSA. Certamente ao menos uma iteração é necessária para avaliação dos impactos das mudanças necessárias no sistema em relação aos requisitos já verificados, bem como para manter o controle (gerenciamento) dos requisitos do sistema (até a desativação ou descomissionamento do sistema).

Figura 6.8: Exemplo de sequência temporal entre iterações da MERUSA

6.8 Resumo das Características da MERUSA

Conforme discutido na seção 5.3, existe uma forte correlação entre a avaliação de erros humanos e a análise de requisitos de usabilidade (devido à dependência de ambos os processos em relação aos cenários de operação, análise de tarefas, fatores ergonômicos, organizacionais, treinamento, hábitos, relacionamentos do grupo de trabalho e outros fatores cognitivos). Por outro lado, a análise de risco de erros humanos está fortemente associada a aspectos de segurança, uma vez que as análises de acidentes têm mostrado que é crescente a percentagem de acidentes cuja origem está relacionada com os erros humanos. Dessa forma, é evidente que existe uma sinergia entre os processos de especificação de requisitos de usabilidade e de segurança.

Essa sinergia manifesta-se na elaboração de diretivas de prevenção de erro, associadas às características de usabilidade do sistema. Existe a possibilidade da utilização de ferramentas clássicas de análise de segurança de hardware e software (como HAZOP, FMECA, FTA, ETA) também para a identificação e quantificação da probabilidade de erro humano. Adicionalmente, a elaboração a análise probabilística de risco (PRA) representa uma ponte entre a análise de segurança e a avaliação de usabilidade.

A tabela 6.6 apresenta um resumo da seqüência de atividades envolvidas nas quatro fases da MERUSA, considerando os métodos, técnicas e os produtos obtidos, visando a especificação dos requisitos de segurança e usabilidade.

Os métodos e as técnicas citados representam os recursos típicos que podem ser utilizados em cada uma das atividades, sendo que, para cada tipo de sistema o especialista em determinado requisito poderá definir a melhor ferramenta a ser utilizada. O relacionamento entre os processos de especificação de requisitos de usabilidade e segurança pode ser observado pela similaridade dos tipos de técnicas e métodos utilizados, bem como pelos objetivos comuns nas fases III e IV.

Tabela 6.6: Resumo das características de cada fase da MERUSA

Fase	Req. Funcionais	Metodologia de Especificação de Requisitos de Segurança			Metodologia de Especificação de Requisitos de Usabilidade		
	Objetivos	Atividades	Técnicas	Produtos	Atividades	Técnicas	Produtos
I	<ul style="list-style-type: none"> ❑ Identificação de objetivos; ❑ Identificação de cenários; ❑ Identificação de restrições do ambiente; ❑ Elicitação dos requisitos do domínio; ❑ Revisão de cenários 	<ul style="list-style-type: none"> ❑ Identificação dos perigos; ❑ Identificação dos objetivos de segurança; ❑ Elicitação dos requisitos globais e cenários de segurança 	<ul style="list-style-type: none"> ❑ GRL; ❑ UCM; ❑ FTA; ❑ SFTA; ❑ MORT; ❑ HAZOP; ❑ AEA; ❑ WSA; ❑ SMHA 	<ul style="list-style-type: none"> ❑ Especificação dos requisitos globais de segurança; ❑ Especificação dos cenários de segurança; ❑ Registro da decomposição dos requisitos de segurança 	<ul style="list-style-type: none"> ❑ Identificação dos usuários; ❑ Análise de tarefas; ❑ Identificação dos princípios de projeto de IHC e restrições de ambiente 	<ul style="list-style-type: none"> ❑ Análise de tarefas; ❑ Estudo e observação de campo; ❑ Entrevista; ❑ Questionário; ❑ Grupo de foco; ❑ Tempestade cerebral 	<ul style="list-style-type: none"> ❑ Registros de análise de tarefas; ❑ Especificação da perfil do usuário; ❑ Especificação do modelo mental do usuário; ❑ Especificação dos princípios gerais de projeto
II	<ul style="list-style-type: none"> ❑ Descrição dos pontos de vista da arquitetura; ❑ Construção dos cenários 	<ul style="list-style-type: none"> ❑ Elicitação dos requisitos específicos de segurança para cada ponto de vista de arquitetura; ❑ Revisão dos objetivos de segurança; ❑ Construção dos cenários de segurança 	<ul style="list-style-type: none"> ❑ GRL; ❑ UCM; ❑ CCA; ❑ ETA; ❑ Simulação 	<ul style="list-style-type: none"> ❑ Especificação dos requisitos específicos de segurança para cada ponto de vista; ❑ Registro e validação dos cenários de segurança; ❑ Registro da decomposição dos requisitos de segurança 	<ul style="list-style-type: none"> ❑ Elicitação dos requisitos gerais e específicos de usabilidade para cada ponto de vista de arquitetura; ❑ Elaboração do guia de estilo; ❑ Reengenharia do trabalho; ❑ Revisão de tarefas 	<ul style="list-style-type: none"> ❑ GRL; ❑ UCM; ❑ Simulação; ❑ Análise de tarefas; ❑ Estudo e observação de campo; ❑ Entrevista; ❑ Questionário; ❑ Grupo de foco; ❑ Tempestade cerebral 	<ul style="list-style-type: none"> ❑ Especificação dos requisitos específicos de usabilidade para cada ponto de vista; ❑ Guia de estilo de usabilidade; ❑ Reengenharia do trabalho; ❑ Registro da decomposição dos requisitos de usabilidade

III	☐ Construção dos modelos das arquiteturas; ☐ Avaliação de sensibilidade	☐ Análise de risco (independente); ☐ Análise de segurança (independente)	☐ HRA; ☐ PRA; ☐ FTA; ☐ HAZOP; ☐ FMECA; ☐ <i>CHECK-LIST</i>	☐ Análise probabilística de risco; ☐ Análise de segurança (independente)	☐ Modelagem da IHC; ☐ Prototipação da IHC; ☐ Avaliação de usabilidade (independente); ☐ Revisão do guia de estilo	☐ Prototipação; ☐ Métodos de inspeção de usabilidade; ☐ Métodos de teste de usabilidade	☐ Protótipo da IHC; ☐ Registro de testes de usabilidade
IV	☐ Análise integrada dos requisitos; ☐ Análise de compromissos	☐ Avaliação de sensibilidade dos requisitos de segurança; ☐ Análise de segurança; ☐ Verificação de conformidade entre requisitos	☐ Testes de conformidade; ☐ GRL; ☐ PRA; ☐ FTA; ☐ FMECA	☐ Registro da satisfação e conformidade entre requisitos; ☐ Análise de risco do desenvolvimento do sistema; ☐ Especificação de estratégias de desenvolvimento	☐ Avaliação de sensibilidade dos requisitos de usabilidade; ☐ Análise de usabilidade; ☐ Verificação de conformidade entre requisitos	☐ Testes de conformidade; ☐ GRL; ☐ Métodos de inspeção de usabilidade; ☐ Métodos de teste de usabilidade	☐ Registro da satisfação e conformidade entre requisitos; ☐ Análise de risco do desenvolvimento do sistema; ☐ Especificação de estratégias de desenvolvimento

7 AVALIAÇÃO DA METODOLOGIA

A estratégia de avaliação da eficácia da Metodologia de Especificação de Requisitos de Usabilidade e Segurança orientada para Arquitetura (MERUSA), necessita de uma plataforma de testes onde a sua eficácia possa ser avaliada.

O enfoque da MERUSA são os requisitos de segurança e usabilidade, sendo que a sua definição, análise e validação são específicos para cada domínio de aplicação. Dessa forma, para que a metodologia possa ser avaliada deve ser definido um domínio de aplicação para essa plataforma de testes.

Idealmente, essa plataforma pode ser caracterizada como um ciclo de desenvolvimento de um sistema digital distribuído e heterogêneo, para o qual as características do ambiente onde o sistema deve ser inserido (que pode ser entendido como a especificação do domínio) sejam conhecidas, de modo que, com a utilização da metodologia, possam ser determinados os requisitos do sistema.

Em função das características necessárias para essa plataforma, das prioridades de investimento do governo federal em sistemas relacionadas com a infra-estrutura e considerando-se a experiência profissional do autor deste trabalho, adota-se como domínio de aplicação o Setor Elétrico Brasileiro, focalizando especificamente os sistemas digitais distribuídos responsáveis pela supervisão e controle de sistemas de transmissão de energia elétrica. Esses sistemas de transmissão são geograficamente distribuídos e possuem uma interdependência operativa, compondo uma ampla rede elétrica denominada de Sistema Interligado Nacional – SIN (ONS, 2002).

Dessa forma, a avaliação da MERUSA é realizada através de um caso de estudo focalizado no domínio do Sistema Elétrico Brasileiro. Esse estudo consiste de duas atividades principais: inicialmente é elaborado um meta-modelo de especificação de projeto para os Sistemas de Supervisão e Controle para Subestações de Transmissão de Energia (SSC-STE) associados ao Sistema Interligado Nacional (SIN); em seguida utiliza-se a MERUSA para a elicitação e análise dos requisitos de segurança e usabilidade para um sistema desse domínio. Os resultados obtidos na

primeira iteração do ciclo de desenvolvimento da arquitetura do SSC-STE são avaliados, para a verificação da eficácia da MERUSA nesse domínio.

Com o objetivo de não sobrecarregar este documento, optou-se por detalhar esse caso de estudo em um documento a parte, denominado: “MERUSA: Caso de Estudo – Sistema de Supervisão e Controle de Subestação de Transmissão de Energia do Sistema Interligado Nacional” (AVELINO, 2004).

Nas próximas seções são apresentados e comentados os resultados principais obtidos nas atividades desse caso de estudo. Os resultados parciais, bem como o detalhamento das técnicas utilizadas, para a obtenção desses resultados podem ser encontrados no documento detalhado desse caso de estudo (AVELINO, 2004).

7.1 Meta-arquitetura para Sistema de Transmissão do SIN

A plataforma para o teste da metodologia é desenvolvida considerando-se a utilização do modelo ODP ao longo de todo o ciclo de vida do empreendimento, desde a especificação de necessidades do negócio, definição de arquitetura, projeto, implementação, teste e manutenção do sistema (PUTMAN, 2001).

Utilizando-se os conceitos do modelo ODP, pode-se definir a arquitetura do sistema a partir das características do modelo do negócio do sistema, que consiste na identificação das necessidades do negócio, das características operacionais e funcionais do sistema (definição do seu comportamento) e das características das informações do negócio tratadas pelo sistema. Para a obtenção dessas informações referentes aos Sistemas de Transmissão de Energia (STE) do Sistema Interligado Nacional (SIN), utiliza-se a descrição da comunidade do Setor Elétrico (onde o SIN está contido), seguida da descrição da comunidade do SIN e, finalmente, da especificação do negócio para os Sistemas de Transmissão de Energia integrantes do SIN. Essa estratégia representa a visão do ambiente do sistema em níveis de abstração sucessivos, conforme representado na figura 3.2.

No modelo ODP, a especificação do negócio consiste da elaboração da especificação do sistema sobre os pontos de vista da empresa e da informação. A especificação do ponto de vista da empresa consiste na definição do escopo do negócio em relação ao empreendimento, seus objetivos, seus processos, seus objetos do negócio, suas políticas do negócio e os objetos que representam as entidades do mundo real ou envolvidos com os objetos de tecnologia da informação. A especificação do ponto de vista da informação consiste da definição das estruturas de fluxo de dados, dos objetos de informação e das características estáticas e dinâmicas do sistema (PUTMAN, 2001).

Se forem consideradas características suficientemente genéricas para o modelo de negócio, ou seja, características que se apliquem, de um modo geral, a toda uma família de sistemas pertencentes ao domínio, então, o modelo ODP resultante poderá ser genérico o suficiente para ser aplicável a todos os membros dessa família. Esse modelo genérico pode ser chamado de meta-modelo, se o mesmo possuir um nível de abstração tal que a sua especialização, para um sistema particular, possa ser interpretada como um processo de instanciação. Como esse meta-modelo representa a estrutura das entidades do sistema, o relacionamento entre essas entidades e as restrições associadas, esse meta-modelo também pode ser considerado a meta-arquitetura do sistema modelado.

Dessa forma, define-se o modelo de negócio dos Sistemas de Transmissão de Energia (STE) com especificações abrangentes, de modo que possa ser produzida uma meta-arquitetura do negócio para os Sistemas de Transmissão de Energia do Sistema Interligado Nacional.

As seções seguintes descrevem o modelo de negócio e o meta-modelo para descrição da arquitetura para os Sistemas de Transmissão de Energia do Sistema Interligado Nacional (ONS, 2002); (MME, 2003).

7.1.1 Modelo de negócio do SIN

O modelo desenvolvido nesta seção enfoca uma visão do negócio aplicada ao domínio do Setor Elétrico Brasileiro, dentro do qual existem várias comunidades, sendo uma delas o Sistema Interligado Nacional (SIN).

Para se determinar, de modo abrangente, todas as entidades associadas ao domínio do SIN, é elaborado um diagrama que apresenta as principais entidades envolvidas no negócio do Setor Elétrico Brasileiro. A elaboração desse diagrama é baseada na estrutura atual do Setor Elétrico Nacional, considerando-se o Modelo Institucional do Setor Elétrico, proposto em dezembro de 2003. Essas especificações são baseadas nas informações técnicas divulgadas pelo gestor principal do Sistema Interligado Nacional (Operador Nacional do Sistema - ONS) e por informações de políticas de negócio divulgadas pelo Ministério de Minas e Energia (ONS, 2002); (MME, 2003).

A partir da experiência do autor deste trabalho no domínio do Setor Elétrico Brasileiro, as especificações técnicas do ONS são estudadas em detalhe de modo a identificar todas as entidades envolvidas na comunidade do Setor Elétrico. Nessa análise são considerados principalmente: os papéis desempenhados pelas entidades, os relacionamentos entre essas entidades (suas interfaces) e a sua influência em relação ao objeto a ser especificado (neste caso um Sistema de Supervisão e Controle para Subestações do SIN).

A recursividade do modelo ODP permite que o modelo do negócio possa ser decomposto em vários níveis de abstração, com refinamentos sucessivos, até o nível de detalhe necessário para a identificação das interfaces do sistema dentro da comunidade na qual ele está contido, conforme representado na figura 7.1.

A descrição detalhada de todas as entidades identificadas nos três níveis de abstração da figura 7.1 é apresentada na referência (AVELINO, 2004). Nesta seção destacam-se apenas os elementos principais das diversas comunidades.

Figura 7.1: Comunidades do Setor Elétrico Brasileiro

No primeiro nível de abstração do diagrama da figura 7.1, são apresentadas as entidades que compõem a comunidade do Setor Elétrico. O Sistema Interligado é elemento de principal interesse, uma vez que este representa o conjunto de equipamentos e sistemas que geram, transmitem e distribuem a energia aos consumidores, integrando vários sistemas de geração (usinas), transmissão (linhas de transmissão e subestações de transmissão) e distribuição (subestações de distribuição) de energia distribuídos pelas regiões Sul, Sudeste, Centro-Oeste, Nordeste e Norte do Brasil, através da rede básica de transmissão de energia. Fora da comunidade do Setor Elétrico estão representadas duas comunidades (Seguradora e Meio Ambiente) que estão diretamente envolvidas com o Setor Elétrico, sendo consideradas na definição das interfaces do Setor Elétrico.

No segundo nível de abstração do diagrama apresentado na figura 7.1, são apresentadas as entidades que compõem a comunidade do Sistema Interligado Nacional. Deve ser observado que o objeto Sistema Interligado, do primeiro nível de abstração, é uma comunidade no segundo nível de abstração (aplicando-se os conceitos de abstração e composição do modelo ODP). Neste nível de abstração destaca-se o Sistema de Transmissão, que representa os sistemas que transportam energia (linhas de transmissão) e condicionam a energia para transferências da mesma entre linhas ou distribuidores (subestação de transmissão e distribuição). No Sistema Elétrico Brasileiro, são caracterizados como distribuidores aqueles que compram a energia dos geradores e distribuem pela Rede Básica de Transmissão (podem ser empresas estatais ou privadas, que operam a partir de uma concessão da ANEEL, tais como CTEEP, ELETROSUL, CHESF, FURNAS, COPEL).

No terceiro nível de abstração, o objeto Sistema de Transmissão, do nível de abstração 2, é tratado como uma nova comunidade, sendo decomposto nos seus objetos mais elementares. Dentre esses objetos, identifica-se o Sistema de Supervisão e Controle que se deseja especificar, o qual representa um conjunto de equipamentos que, mediante aquisição automática e processamento de dados, fornece informações exatas e constantemente atualizadas que permitem, aos operadores, a supervisão e o

controle dos dispositivos de manobra, a transformação e o condicionamento de energia das linhas de transmissão que pertencem ao SIN. Além da supervisão e controle local, esse sistema tem, como responsabilidade, a comunicação com sistemas externos, que permitam a supervisão e controle remoto dos principais dispositivos do sistema de transmissão (através dos operadores do sistema).

Para complementar o diagrama da figura 7.1, é necessária a descrição das comunidades envolvidas. Para essa descrição é utilizada a linguagem de especificação da empresa do modelo ODP (tabela 3.3). A identificação desses conceitos para cada uma das comunidades é baseada no estudo das informações técnicas divulgadas pelos gestores principais do Sistema Interligado Nacional (ONS), por informações de políticas de negócio divulgadas pelo Ministério de Minas e Energia e pela experiência do autor deste trabalho no domínio do Setor Elétrico Brasileiro (ONS, 2002); (MME, 2003); (AVELINO, 2004).

A partir das descrições das comunidades identificadas na figura 7.1, analisam-se as interfaces entre as principais entidades do Setor Elétrico Brasileiro, identificando-se as entidades que interagem com os Sistemas de Transmissão do Sistema Interligado Nacional (impondo portanto requisitos a estes sistemas). Com essa informação, são elaborados novos diagramas representando o inter-relacionamento das entidades da comunidade do Setor Elétrico com o SIN e com os Sistemas de Transmissão, permitindo a identificação da origem dos requisitos e facilitando a sua rastreabilidade (AVELINO, 2004).

7.1.2 Meta-Modelo ODP para um Sistema de Transmissão de Energia

Analisando-se os atributos das entidades envolvidas na comunidade do negócio dos Sistemas de Transmissão do SIN, considerando-se os relacionamentos dessas entidades e adotando-se como referência a estrutura de meta-modelo ODP, elabora-se uma meta-arquitetura para um Sistema de Transmissão de Energia Elétrica pertencente ao SIN (PUTMAN, 2001); (AVELINO, 2004).

Na figura 7.2 é apresentado o resultado desse trabalho (utilizando-se a notação de diagrama de classes em UML), para representação dos conceitos de empresa do modelo ODP (OMG, 1999).

Os diversos tipos de entidades apresentadas nesta figura não precisam necessariamente existir em todas as aplicações do modelo, cabendo ao arquiteto do sistema analisar a possibilidade da existência de cada entidade para a instanciação do meta-modelo na sua aplicação específica.

Da mesma forma, para cada tipo de entidade são indicados os seus atributos típicos, os quais podem não representar a totalidade das características que devem ser consideradas. Os atributos representados foram os considerados mais significativos para um Sistema de Supervisão e Controle de um Sistema de Transmissão de Energia – STE (considerando-se os Procedimentos de Rede do ONS e a experiência do autor deste trabalho) (ONS, 2002).

A meta-arquitetura para o Sistema de Transmissão de Energia do SIN, representada na figura 7.2, mostra diversos objetos como membros da comunidade do Sistema de Supervisão e Controle da SE, sendo que essas entidades podem representar os elementos reais do processo (como equipamentos de manobra, equipamentos de proteção, linhas de transmissão, estruturas metálicas e construções de alvenaria). Dentre essas entidades, destaca-se o Sistema de Supervisão e Controle (SSC), que está sendo considerado neste trabalho como uma instância de um objeto da comunidade dos Sistemas de Supervisão e Controle da SE e que se caracteriza como um sistema de objetos de processamento distribuído (ODP). Nessa figura, são exemplificados os papéis que esse objeto pode assumir e os objetivos a serem alcançados por esse objeto.

Apesar do SSC ser apenas um objeto dentro da meta-arquitetura do Sistema de Transmissão, este pode ser novamente decomposto (utilizando-se a recursividade do modelo ODP), podendo ser caracterizado como uma comunidade de novos objetos.

Figura 7.2: Meta-arquitetura ODP para o STE do Sistema Interligado Nacional

O objetivo central dessa meta-arquitetura é possibilitar uma visão global do empreendimento que permita estabelecer uma Especificação de Empresa para um Sistema de Transmissão de Energia Elétrica. O fato do SSC ser apenas um objeto dentro do contexto do empreendimento é uma visão incomum para os projetistas de sistema, onde a Especificação do Sistema é centrada no sistema digital a ser desenvolvido, não sendo considerados, de maneira sistemática, o ambiente onde o mesmo está inserido. Assim, a visão de empresa permite que as definições iniciais da estrutura do projeto do sistema digital (sua arquitetura), seus objetivos, suas estratégias de desenvolvimento e seus requisitos estejam extremamente bem alinhados com as necessidades do ambiente.

Um ponto importante desse enfoque é permitir uma visão ampla dos cenários de utilização do sistema e uma participação mais efetiva dos vários *stakeholders*, tornando a definição dos requisitos do sistema consistente com as necessidades reais da comunidade em que mesmo está inserido, e melhorando a clareza e a completeza dessa definição.

Outro aspecto importante da constituição de um meta-modelo é que essa representação permite uma semi-formalização das características do empreendimento (através da Linguagem de Especificação da Empresa), o que facilita o diálogo entre os *stakeholders* e os projetistas do sistema digital, bem como assegura a rastreabilidade dos requisitos de sistema para os requisitos do ambiente, diminuindo a taxa de erros dos processos de validação e manutenção do sistema.

As considerações anteriores aplicam-se tanto aos RF como aos RNF; entretanto, tem sido observado na prática que os problemas de completeza, rastreabilidade, validação e consistência são maiores para os RNF, uma vez que estes são geralmente mais correlacionados com as características do ambiente em que o sistema está inserido, sendo portanto de mais difícil detecção no início do desenvolvimento do sistema.

Neste trabalho, o SSC é tratado como um objeto da comunidade do Sistema de Transmissão de Energia, para o qual são identificados os requisitos impostos em

função do mesmo pertencer a esta comunidade. Para detalhar a identificação dos requisitos do objeto SSC, é elaborado um contrato entre o objeto SSC e o seu ambiente (comunidade do Sistema de Transmissão), denominado contrato ambiental do SSC (AVELINO, 2004).

Este contrato (descrito utilizando os conceitos ODP) contém: os objetivos do SSC, as responsabilidades dos papéis nele existentes, os objetos externos e suas atividades, as pré-condições para relacionamento com o SSC, as pós-condições após relacionamento com o SSC, as condições invariantes, as obrigações do SSC, as possibilidades de expansões (exceções), as proibições de comportamento, o desempenho exigido e outras restrições específicas. Estas últimas características representam os requisitos iniciais do sistema, derivados do modelo de negócio do SIN (ONS, 2002), (AVELINO, 2004).

Para complementar a especificação do negócio, conforme o modelo ODP, deve ainda ser elaborada a especificação do ponto de vista da informação, constituída dos modelos invariante, estático e dinâmico do SSC.

O esquema invariante identifica as propriedades (restrições) que se mantém constantes mesmo com a mudança de estado dos objetos. As condições invariantes do SSC podem ser apresentadas por diagramas de classes UML (AVELINO, 2004).

Os diagramas dinâmico e estático identificam os estados do objeto e as ações dinâmicas que causam as mudanças de estado. Os estados e as ações dinâmicas do SSC podem ser representadas por diagramas de atividades UML (AVELINO, 2004).

7.2 Aplicação da MERUSA

Nesta seção, apresentam-se os resultados de um caso de estudo da aplicação da MERUSA na especificação dos requisitos de segurança e usabilidade de um sistema digital, desenvolvido a partir da meta-arquitetura dos Sistemas de Transmissão do SIN (AVELINO, 2004).

O caso de estudo considerado, neste trabalho, restringe-se à primeira iteração do ciclo de desenvolvimento desse sistema digital. Essa primeira iteração envolve as atividades iniciais do ciclo de vida, associadas à elaboração da especificação de requisitos do sistema. O detalhamento dessas atividades é apresentado na figura 7.3.

As informações de referência dos Sistemas de Transmissão do SIN são aquelas representadas pelo modelo de negócio desenvolvido na seção 7.1 (em especial a descrição da meta-arquitetura e o contrato ambiental do Sistema de Supervisão e Controle). Além dessas informações, obtidas a partir de documentos do ONS, foram utilizados também dados de campo (entrevistas com operadores, desenvolvedores e especialistas do domínio) de uma Subestação de Transmissão da Companhia de Transmissão de Energia Elétrica Paulista (CTEEP), para as atividades relacionadas com análise de tarefas e identificação dos cenários de operação.

As atividades apresentadas na figura 7.3 aplicam-se a qualquer ciclo de desenvolvimento de sistema, não sendo exclusivas para o desenvolvimento de sistemas de supervisão e controle de sistemas elétricos.

As seções seguintes descrevem, sucintamente, os resultados obtidos pelo autor deste trabalho, em cada uma das atividades apresentadas na figura 7.3, para a primeira iteração da especificação de requisitos de um Sistema de Supervisão e Controle de Subestação de Transmissão de Energia (SSC-STE) para o Sistema Interligado Nacional, utilizando a MERUSA.

7.2.1 Atividade 1.1 - Identificação do negócio e da organização

A identificação das necessidades básicas do SIN é baseada na análise de normas reguladoras, modelos institucionais e procedimentos estabelecidos pelo ONS, editais de subcontratação de empresas estatais do setor elétrico e normas de outros órgãos reguladores associados à comunidade do setor elétrico (EPTE, 1998); (ANA, 2001); (CEMIG, 2002); (CPFL, 2002); (FURNAS, 2002); (ONS, 2002); (IBAMA, 2003); (MMA, 2003); (MME, 2003).

Figura 7.3: Atividades do caso de estudo da MERUSA

Os resultados dessa atividade constituem-se na identificação de todas as comunidades envolvidas com a definição, desenvolvimento, operação e manutenção de sistemas associados ao SIN (Setor Elétrico, Sistema Interligado Nacional e Sistema de Transmissão), representada, basicamente, pela figura 7.1 e por diagramas de relacionamento entre essas comunidades.

7.2.2 Atividade 1.2 - Definição dos requisitos do negócio

A identificação dos requisitos do negócio para um Sistema de Supervisão e Controle de Subestação de Transmissão de Energia (SSC-STE) integrado ao SIN é realizada, dentro do modelo de negócio desse empreendimento, a partir da meta-arquitetura do STE (figura 7.2) e da descrição do contrato ambiental do SSC.

Neste caso são identificados os seguintes requisitos para um SSC-STE integrado ao SIN: 28 obrigações (**OB01** a **OB28**), 4 possibilidades de expansões (**PE01** a **PE04**), 7 proibições (**PR01** a **PR07**), 23 requisitos de desempenho (**DE01** a **DE23**) e 17 restrições específicas (**RE01** a **RE17**) (AVELINO, 2004).

7.2.3 Atividade 2.1 - Identificação dos objetivos e cenários do sistema

Os objetivos globais (**OG**) do SSC-STE são os seguintes (identificados a partir do contrato ambiental do SSC):

- **OG01:** supervisionar e controlar o fluxo de energia elétrica através das linhas de transmissão e subestações;
- **OG02:** fornecer informações a Centros Remotos de Supervisão e Controle para permitir a coordenação regional e nacional do fluxo eletroenergético no Sistema Interligado Nacional;
- **OG03:** assegurar a segurança, confiabilidade, disponibilidade, qualidade e custo do suprimento da energia ao mercado consumidor dentro de padrões técnicos e regras de mercado estabelecidas pela ANEEL.

Os cenários globais de operação do SSC-STE podem ser identificados a partir dos Procedimentos de Rede estabelecidos pelo Operador Nacional do Sistema. Esses cenários podem ser classificados como (ONS, 2002):

- **CN01:** execução do programa diário de operação;
- **CN02:** controle da transmissão em operação normal;
- **CN03:** operação em contingência;
- **CN04:** recomposição da rede de operação após perturbação;
- **CN05:** gerenciamento da carga.

7.2.4 Atividade 2.2 - Elicitação dos requisitos do domínio

Nesta primeira iteração, os requisitos de negócio são considerados os requisitos iniciais do SSC-STE. Assim, nesta iteração, realiza-se apenas o reagrupamento dos requisitos do negócio em classes e identificam-se os casos de uso iniciais do sistema.

Os requisitos do negócio podem ser subdivididos nos seguintes classes: políticas de aquisição de dados (**AQD**), políticas de tratamento de dados (**TTD**), políticas de registro de dados (**RGD**), políticas de interface homem-computador (**IHC**), políticas de comunicação de dados (**CMD**), políticas de qualidade de serviço (**QoS**), requisitos de desempenho (**RDP**) e requisitos de segurança (**RSG**).

Os casos de uso (**UC**) do SSC-STE são identificados a partir das responsabilidades dos papéis do objeto SSC, especificadas no modelo de negócio do sistema de transmissão para o SIN. A tabela 7.1 apresenta os dez casos de uso básicos identificados, com os respectivos atores envolvidos.

Tabela 7.1: Relação dos casos de uso básicos do SSC-STE

Caso de Uso	Descrição	Atores
UC01	Supervisão da operação dos equipamentos	Operador do SSC, CNOS, COSR/COS, Engenharia de Manutenção
UC02	Controle e comando dos equipamentos	Operador do SSC, CNOS, COSR/COS
UC03	Análise de informações e estatísticas de operação das linhas de transmissão	Operador do SSC, CNOS, COSR/COS, Engenharia de Manutenção
UC04	Supervisão das ocorrências de proteção	Engenharia de Proteção, CNOS, COSR/COS
UC05	Análise de distúrbios elétricos	Engenharia de Proteção, CNOS, COSR/COS
UC06	Coordenação da operação do SSC	CNOS, COSR/COS
UC07	Coordenação da manutenção dos equipamentos	Engenharia de Manutenção
UC08	Coordenação da proteção dos equipamentos	Engenharia de Proteção
UC09	Parametrização da proteção de equipamentos	Engenharia de Proteção
UC10	Gerenciamento de recursos do SSC	Administrador do SSC

7.2.5 Atividade 2.3 - Descrição da arquitetura

A primeira versão da arquitetura do sistema, considerando os cinco pontos de vista do modelo ODP, pode ser obtida a partir do modelo do negócio do SIN apresentado na seção 7.1.

A arquitetura do ponto de vista da empresa para o SSC-STE é representada pelas entidades que interagem com o SSC (nível de abstração 3 da figura 7.1), complementada pela descrição do contrato ambiental do SSC (AVELINO, 2004).

A arquitetura do ponto de vista da informação é composta pelo esquema invariante do modelo do negócio do SSC, onde são apresentadas as restrições contratuais associadas (podendo ser representado por um diagrama de classes em UML) e pelo esquema estático e dinâmico do modelo do negócio do SSC, onde são

representados os estados do sistema e as ações dinâmicas associadas (podendo ser representado por um diagrama de atividades em UML) (AVELINO, 2004).

A arquitetura do ponto de vista da computação descreve o SSC em termos dos objetos das interfaces do sistema e das restrições das suas ações e suas interações. Esse diagrama pode ser representado utilizando-se diagramas de casos de uso em UML (AVELINO, 2004).

A arquitetura do ponto de vista da engenharia descreve o SSC em termos dos mecanismos e funções, que fornecem suporte para a interação entre objetos distribuídos e os objetos de transparência distribuída, que mascaram a complexidade das interações. Na representação dessa arquitetura são identificados os objetos básicos de engenharia – OBE associados aos específicos nós e os canais de comunicação entre os nós (AVELINO, 2004).

A arquitetura do ponto de vista da tecnologia define quais tecnologias, padrões e produtos serão utilizados para a implementação do SSC. Essa definição é associada às características da organização, sendo difícil a descrição de uma arquitetura genérica. Para permitir uma avaliação inicial dos requisitos do sistema adota-se uma arquitetura que considera, como critério básico, a utilização de produtos padronizados, em sua maioria com aplicação já consagrada na indústria ou no próprio processo elétrico (AVELINO, 2004).

7.2.6 Atividade 2.4 - Construção de cenários

Nesta atividade é elaborado um modelo representando os objetivos do sistema e seus respectivos requisitos funcionais e não funcionais, utilizando a linguagem de representação GRL. Neste modelo incluem-se os cenários (com o refinamento dos casos de uso já identificados), utilizando-se a representação UCM. A representação conjunta permite uma verificação cruzada entre objetivos e cenários, visando detectar inconsistências na especificação inicial dos requisitos.

Dessa análise conclui-se que os casos de uso **UC07**, **UC08**, **UC09** e **UC10** não estão associados com nenhum cenário e, além disso, os três últimos casos de uso não estão correlacionados com nenhum tipo de requisito, evidenciando uma inconsistência no conjunto de requisitos iniciais (AVELINO, 2004).

A causa dessa inconsistência advém do fato dos requisitos iniciais terem sido derivados dos procedimentos de rede do ONS (que é o ator principal dos cenários elicitados). Já os casos de uso **UC07**, **UC08**, **UC09** e **UC10** têm, como atores, os papéis internos à subestação (engenheiro de manutenção, engenheiro de proteção e administrador do sistema) que executam tarefas de suporte para o SIN e não estão diretamente associados aos cenários globais especificados pelo ONS.

Dessa forma, é necessário especificar cenários adicionais que contemplem os casos de uso **UC07**, **UC08**, **UC09** e **UC10**. Adicionalmente, verifica-se também a necessidade da elicitação de novos requisitos relacionados com esses casos de uso.

Os novos cenários identificados são os seguintes:

- **CN06**: manutenção de equipamentos em operação normal;
- **CN07**: parametrização de equipamentos.

Os novos requisitos elicitados, identificados como requisitos adicionais (**RA**), são os seguintes:

- **RA01**: O SSC deve permitir a supervisão do histórico de operação dos equipamentos principais;
- **RA02**: O SSC deve permitir a alteração dos parâmetros de configuração da BD;
- **RA03**: O SSC deve possuir recursos de parametrização através de terminais remotos de comunicação;
- **RA04**: O SSC deve permitir o acesso remoto registrado às suas bases de dados, com identificação e validação através do Administrador do sistema.

Esses novos requisitos podem ser subdivididos nas classes existentes da seguinte forma: **RA01** incluído nas políticas de aquisição de dados (**AQD**); **RA02** incluído nas políticas de qualidade de serviço (**QoS**); **RA03** incluído nas políticas de comunicação de dados (**CMD**); **RA04** incluído nos requisitos de segurança (**RSG**).

7.2.7 Atividade 2.5 - Construção dos modelos de arquiteturas

Os cinco pontos de vista de arquitetura para o SSC-STE já foram modelados na atividade 2.3, sendo representados, respectivamente, pelos seguintes diagramas:

- **Ponto de vista da empresa:** representado pelas entidades da figura 7.1 e complementada pela descrição do contrato ambiental (AVELINO, 2004);
- **Ponto de vista da informação:** representado pelo diagrama de classes do esquema invariante do modelo do negócio e pelo diagrama de atividades do esquema estático e dinâmico do modelo do negócio (AVELINO, 2004);
- **Ponto de vista da computação:** representado pelos objetos de computação e suas interfaces (AVELINO, 2004);
- **Ponto de vista da engenharia:** representado pelos objetos básicos de engenharia associados aos nós e canais de comunicação (AVELINO, 2004);
- **Ponto de vista da tecnologia:** representado pelas tecnologias e padrões a serem utilizados na implementação (AVELINO, 2004).

Nesta primeira iteração do processo de desenvolvimento da arquitetura do SSC-STE, o alto nível de abstração da descrição do sistema ainda não permite construir protótipos ou modelos de simulação para essas visões de arquitetura; assim, a atividade de construção de arquiteturas, nesta iteração, resume-se ao registro documental dos modelos de arquitetura já definidos.

7.2.8 Atividade 2.6 - Avaliação da sensibilidade da arquitetura

Nesta atividade, realiza-se uma avaliação qualitativa dos requisitos iniciais em relação às características apresentadas nas várias visões da arquitetura. Nesta primeira iteração, essa avaliação consiste da distribuição dos requisitos iniciais entre os cinco pontos de vista da arquitetura do SSC-STE e da avaliação o nível de sensibilidade (**NS**) de cada requisito em relação a possíveis alterações da arquitetura que podem ocorrer no detalhamento do específico ponto de vista de arquitetura do sistema, conforme os critérios descritos na seção 6.5.1.

Na tabela 7.2 apresentam-se os requisitos elicitados (a partir da tabela 7.1) para o SSC-STE, bem como o nível de sensibilidade (**NS**) em relação a cada ponto de vista da arquitetura, após uma avaliação qualitativa individual baseada na experiência do autor deste trabalho (arquiteto do sistema).

Tabela 7.2: Requisitos do SSC-STE e seu nível de sensibilidade

Requisitos do SSC-STE (Descrição Mnemônica)	NS por Ponto de Vista				
	E	I	C	G	T
OB01. Integração dos sistemas da subestação com acesso local ou remoto	3	3	3	3	3
OB02. Utilização de tecnologia digital numérica	0	0	1	1	3
OB03. Compatibilização com sistemas legados, utilizando tecnologia digital	2	3	2	1	3
OB04. Automonitoramento e autodiagnóstico, com bloqueio por defeito	1	2	3	3	2
OB05. Arquitetura aberta e utilização de protocolos normalizados	1	1	3	3	3
OB06. Intervenção de manutenção sem desligamento de equipamentos	1	1	2	2	2
OB07. Proteções com supervisão de tensão para bloqueio de operação e alarme	0	2	1	1	0
OB08. Seleção de modo de operação através de chave seletora e/ou do SSC	0	2	1	1	0
OB09. Registrador de Perturbações com contato para sinalização de eventos	0	2	1	1	1
OB10. Registro de sinais da teleproteção para análise de perturbações	0	2	1	1	0
OB11. Recursos para aquisição, registro, interconexão com ONS e GPS	2	2	3	3	3
OB12. Comunicação de dados analógicos parametrizável	0	2	3	3	3
OB13. Comunicação de dados digitais por exceção	0	2	3	3	3
OB14. Registro SOE monitora principais pontos da instalação	0	2	0	0	0
OB16. Testes de conectividade entre subsistemas	0	0	2	2	2
OB17. Supervisão do Controle Automático de Tensão – CAT	1	2	1	1	1
OB18. Coordenação, supervisão e controle das instalações acessórias	2	2	1	1	1
OB20. Supervisão SOE sem filtragem na mesma interligação com ONS	1	1	2	2	1

OB21. Informações para ONS em valor de engenharia sem filtragem	0	1	1	1	0
OB22. Informações de estado com indicadores de qualidade do dado	0	2	2	2	1
OB23. Compatibilidade de protocolos de comunicação com ONS	0	0	2	2	3
OB24. Protocolo com identificação do estado operacional dos sistemas	0	2	2	2	3
OB26. Supervisão de grandezas analógicas e sinalizações de estado com tempo	0	3	2	2	0
OB27. Supervisão da potência ativa trifásica em pontos de interligação	0	3	1	1	0
OB28. Supervisão e armazenamento de estados e grandezas do processo	0	3	1	1	0
PE01. Recursos de supervisão e controle para interconexão com ONS	1	2	1	3	3
PE02. Recursos de supervisão e controle para sala de comando remota	1	2	1	3	3
PE03. Evolução tecnológica com outros protocolos padrões	0	0	0	0	3
PE04. Período de varredura de comunicação configurável por tipo de grandeza	0	2	2	2	3
PR01. Não geração de comandos espúrios em caso de falha ou manutenção	0	0	3	3	3
PR02. Não enviar informações inválidas aos Operadores	0	2	3	3	3
PR03. Automatismos de controle sem condições inseguras	0	2	3	3	3
PR04. Automatismos de controle não causam indisponibilidade operacional	0	2	3	3	3
PR05. Informações de IHC não podem induzir Operador a erros	0	2	3	3	3
PR06. Acesso bloqueado a pessoas não autorizadas (privilegio parametrizável)	1	2	3	3	3
PR07. Mecanismos para assegurar integridade da comunicação externa	0	1	3	3	3
DE01. Uniformidade de desempenho entre sistemas integrados e equipamentos	0	0	2	3	3
DE02. Projeto, fabricação e ensaio de equipamentos conforme normas ABNT	0	0	1	1	3
DE03. Atendimento aos requisitos de compatibilidade eletromagnética	0	0	0	0	3
DE05. Taxas de aquisição de dados (supervisão de estado a cada 4s)	0	0	1	2	3
DE06. Aquisição de dados com selo de tempo (exatidão e resolução <1ms)	0	0	1	2	3
DE07. Atualização de mudança de estado no centro de operação remoto <4s	0	0	1	2	3
DE08. Relógio do sistema e selo de tempo com a mesma exatidão e resolução	0	0	1	2	3
DE09. Habilitação dos Operadores para as atividades de tempo real	2	0	0	0	0
DE10. Atualização das medições analógicas (parametrizável) <4s	0	0	1	2	3
DE11. Sinalizações de estado reportadas por exceção	0	0	2	2	2
DE12. Varredura de integridade parametrizável, sob demanda ou por evento	0	0	2	2	3
DE14. Período de aquisição de grandezas de estado pelo ONS típico de 2s	0	0	1	2	3
DE15. Medições obtidas a partir de fonte única de informação	0	1	1	1	0
DE16. Medições de tensão com precisão <1% (demais analógicas <2%)	0	0	0	0	2
DE17. Idade máxima do dado <tempo varredura +4 ou +10s (por exceção <8s)	0	0	2	2	2
DE18. Períodos de aquisição parametrizáveis	0	1	2	2	2
DE19. Filtragem por banda morta definida de acordo com ONS	2	0	0	0	0
DE22. Disp. crítica >98,5%; Disp. geral >97,5%; Disp. concentradores >99%	0	0	2	3	3
RE01. Condições ambientais (TA <40C, UR 50 a 90%)	0	0	0	0	3
RE02. Consoles de IHC redundantes	0	0	1	3	2
RE03. Capacidade registro em tempo real >300 alarmes e >300 eventos	0	0	1	3	2
RE04. Disp. funções essenciais >99,99%; Disp. sistema completo >99,95%	0	0	2	3	3
RE05. MTBF > 20.000horas	0	0	2	3	3

RE06. MTTR < 2horas	0	0	1	1	3
RE09. Atualização de estados na IHC da SE <1s	0	0	1	2	3
RE11. Atualização de telas na SE <1s	0	0	1	2	3
RE12. Execução de comando remoto <2,5s	0	0	1	2	3
RE13. Expansibilidade física e lógica >20%	0	0	2	2	3
RE14. Ocupação máxima dos processadores <70%	0	0	1	1	3
RE15. Reserva mínima de memória >30%	0	0	0	0	2
RA01: Supervisão do histórico de operação dos equipamentos principais	1	2	2	2	0
RA02: Alteração remota dos parâmetros de configuração da BD	1	2	2	2	3
RA03: Parametrização através de terminais remotos em redes de comunicação	1	2	1	3	3
RA04: Acesso remoto registrado (identificação e validação do Administrador)	1	2	3	3	3
Legenda: E: Empresa, I: Informação, C: Computação, G: Engenharia, T: Tecnologia e respectivo nível de sensibilidade (NS)					

Durante essa avaliação, alguns requisitos iniciais são identificados como redundantes em relação a outros requisitos já existentes (geralmente mais abrangentes). Esses requisitos não estão incluídos na tabela 7.2 (e consequentemente são excluídos da especificação de requisitos do sistema). A relação dos requisitos iniciais excluídos com essa avaliação é apresentada na tabela 7.3.

Tabela 7.3: Requisitos excluídos do SSC-STE

Requisitos Iniciais Excluídos do SSC-STE (Descrição Mnemônica)	Requisitos Redundantes
OB15. Integração de sistemas legados	OB03
OB19. Supervisão SOE sem filtragem ou cálculo	OB20
OB25. Protocolo com identificação do estado hierarquizado	OB24
DE04. Operação segura e eficiente no modo local ou remoto	PR05, PR06, PR07 e DE01
DE13. Sinalizações com o selo de tempo (exatidão >1ms), via GPS	DE06
DE20. Resolução entre eventos SOE <5ms; exatidão <1ms	DE06
DE21. Base de tempo registro SOE igual a de sinalização de estado	DE08
DE23. Disp. crít.>98,5%; Disp. geral >97,5%; Disp. conc. >99%	DE22
RE07. Atualização de grandezas analógicas no ONS <4s	DE10
RE08. Atualização de estados no ONS <2 s	DE14
RE10. Atualização de pontos analógicos na SE <5s	DE10
RE16. Medidas analógicas enviadas com atributo de validade	OB22 e PR02
RE17. Falha de unidades não causa acionamento incorreto na saída	PR01

7.2.9 Atividade 3.1 - Identificação dos perigos e objetivos de segurança

A partir do contrato ambiental do SSC e da revisão dos Procedimentos de Rede estabelecidos pelo Operador Nacional do Sistema, é identificado o seguinte objetivo global de segurança para o SSC-STE (ONS, 2002):

- **OGS01:** coordenar, operar e controlar o fluxo de energia elétrica através das linhas de transmissão e subestações mantendo a estabilidade do SIN.

Para a identificação dos perigos associados ao sistema, realiza-se uma análise preliminar de segurança do processo elétrico, considerando-se as características dos equipamentos típicos de uma subestação de transmissão de energia. Nessa análise preliminar de segurança, identifica-se que o perigo principal para o sistema elétrico é o controle da sua estabilidade. A partir desse perigo são analisados os equipamentos principais de uma subestação EAT, identificado-se quais são os perigos associados a cada um desses equipamentos. Verifica-se nesta análise que, do ponto de vista do SSC de uma subestação EAT, os perigos do sistema elétrico podem ter origem externa (mudanças de carregamento do sistema, flutuações nos fluxos de energia provenientes de outras subestações, defeitos nas linhas de transmissão, incidência de fenômenos da natureza com raios, ventania, etc.) ou origem interna a subestação (falhas de equipamentos e operações indevidas) (AVELINO, 2004).

7.2.10 Atividade 3.2 - Elicitação dos requisitos globais e cenários de segurança

A análise preliminar de segurança, realizada nesta atividade, tem como objetivo elicitar os requisitos de segurança associados ao SSC-STE, bem como os respectivos cenários de acidente. Assim, na análise do SSC-STE são consideradas apenas as fontes internas de perturbação da estabilidade do sistema (as fontes externas podem ser consideradas na análise de segurança do sistema elétrico interligado, onde é verificada a função dos sistemas de proteção, cujo objetivo é proteger os equipamentos e manter a estabilidade do sistema dentro de limites aceitáveis mesmo na ocorrência desses eventos).

A partir das considerações anteriores, e do estudo do funcionamento do sistema simplificado realizado na atividade anterior, elabora-se uma análise preliminar de segurança. Essa análise baseia-se nos casos de uso e cenários iniciais aplicados ao método de análise de perigos e operacionalidade (HAZOP). Como resultado dessa análise preliminar são identificados os cenários de segurança (**CNS**) considerados (qualitativamente) catastróficos ou críticos, bem como são elicitados os requisitos para garantia da integridade do sistema. Também é elaborada uma árvore de falhas inseguras do SSC-STE (FTA), cujo evento de topo é a perda de estabilidade do sistema elétrico, partindo de três perigos de segurança (de origem interna à subestação): chaveamento incorreto de circuitos principais, regulação incorreta de tensão e regulação incorreta de fase (AVELINO, 2004).

Como resultado da análise preliminar de segurança são identificados oito cenários de segurança (**CNS01 a CNS08**) e dez requisitos de segurança associados a análise(**RSA01 a RSA10**) (AVELINO, 2004).

Para cada cenário de segurança são identificados os cenários globais relacionados (**CN**), os casos de uso associados (**UC**) e os requisitos de segurança associados à análise. Essas associações auxiliam a rastreabilidade dos requisitos de segurança (AVELINO, 2004).

Dentre os cenários que envolvem violação da segurança de acesso (*security*), são considerados apenas aqueles onde o efeito dessa violação pode levar a consequências catastróficas ou críticas para o sistema elétrico, que são decorrentes de ações de intrusos externos do tipo sabotagem ou terrorismo (OMAN, 2001).

É importante observar que os requisitos **RSA06, RSA07, RSA08 e RSA10** são requisitos de segurança de acesso (*security*) e devem ser analisados com ferramentas específicas, mas simultaneamente aos demais requisitos de segurança (*safety*), uma vez que os cenários de segurança estão associados a ambos os tipos de requisitos.

7.2.11 Atividade 3.3 - Elicitação dos requisitos específicos de segurança

Cada um dos requisitos de segurança associados à análise e elicitados na atividade anterior, tem uma repercussão diferente em cada um dos cinco pontos de vista do modelo ODP. Durante o processo de desenvolvimento da arquitetura do sistema, esses requisitos gerais são decompostos em requisitos específicos aplicáveis a cada ponto de vista.

Esta primeira iteração do ciclo de desenvolvimento limita-se a identificar o nível de sensibilidade (NS) dos requisitos de segurança da análise para cada ponto de vista, conforme representado na tabela 7.4.

Tabela 7.4: Requisitos de segurança do SSC-STE e seu nível de sensibilidade

Requisitos de Segurança Associados a Análise do SSC-STE (Descrição Mnemônica)	NS por Ponto de Vista				
	E	I	C	G	T
RSA01. Característica de falha-segura nos controladores do SSC	0	0	1	4	5
RSA02. Supervisão de estados confiável com acesso à validade da informação	0	5	4	3	2
RSA03. Redes de comunicação com protocolo de comunicação seguro	0	1	2	3	5
RSA04. Replicação de lógicas de intertravamento em vários nós do SSC	0	0	1	5	2
RSA05. Lógica de intertravamento local operacional mesmo na falha do SSC	0	0	1	3	4
RSA06. Redes de comunicação (SSC e relês) com controle de acesso	1	3	4	4	5
RSA07. Mecanismos de detecção de violação de barreira de acesso remoto	1	2	4	5	1
RSA08. Limitação de acesso remoto para parametrização do SSC ou relês	1	1	3	2	2
RSA09. Treinamento dos operadores para evitar geração de eventos inseguros	5	0	0	0	0
RSA10. Treinamento dos operadores para detecção de falhas e intrusos	5	0	0	0	0
Legenda: E: Empresa, I: Informação, C: Computação, G: Engenharia, T: Tecnologia e respectivo nível de sensibilidade (NS)					

7.2.12 Atividade 3.4 - Construção dos cenários de segurança

Nesta atividade, são analisados os cenários de segurança e a sua completeza em relação aos requisitos de segurança elicitados. Para essa análise, é elaborado um modelo representando os objetivos de segurança do sistema e seus respectivos requisitos de segurança, utilizando a linguagem de representação GRL. Neste

modelo, incluem-se os cenários de segurança (com o refinamento dos casos de uso já identificados), utilizando-se a representação UCM. Essa representação conjunta permite uma verificação cruzada entre objetivos de segurança e cenários de segurança, visando detectar inconsistências na especificação inicial desses requisitos.

A avaliação crítica do grafo resultante, realizada pelo autor deste trabalho, evidencia a adequada correlação entre os requisitos de segurança da análise e os cenários de segurança. Assim, conclui-se que há consistência entre o conjunto de requisitos de segurança e os cenários de segurança nesta atividade do desenvolvimento(AVELINO, 2004).

7.2.13 Atividade 3.5 - Análise de riscos

O objetivo dessa atividade é realizar uma Avaliação Probabilística de Risco (PRA) que permita identificar os principais riscos relacionados aos objetivos de segurança do sistema, considerando-se os vários pontos de vista de arquitetura do sistema. Entretanto, na primeira iteração do processo de desenvolvimento a elaboração de uma PRA é dificultada pelo alto nível de abstração em que se encontra o detalhamento do sistema. Assim, neste primeiro ciclo de desenvolvimento, esta atividade consiste da identificação dos principais riscos do sistema, de modo a permitir a focalização da análise de segurança nos aspectos mais críticos.

A classificação de risco, para cada um dos oito cenários de segurança, é realizada utilizando os critérios estabelecidos na tabela B.1 da norma IEC1508-5:1997 (IEC-1508-5, 1997). Como resultado dessa análise, verifica-se que os cenários de segurança **CNS02**, **CNS05** e **CNS06** devem ser tratados com prioridade em relação aos demais, sendo estes dois últimos associados a características de *security* (AVELINO, 2004).

7.2.14 Atividade 3.6 - Análise de segurança (independente)

Nesta atividade, deve-se elaborar a análise de segurança do sistema, independente da análise dos demais requisitos do sistema, para a verificação da satisfação dos requisitos de segurança. Entretanto, nesta primeira iteração do ciclo de desenvolvimento, os diversos pontos de vista da arquitetura do sistema ainda estão em um elevado nível de abstração, bem como os requisitos de segurança ainda não foram confrontados com os demais requisitos do sistema para identificação de redundâncias, inconsistências ou conflitos, tornando ineficaz essa análise.

Para uma análise de risco do ponto de vista da segurança de acesso (*security*), existe a necessidade de uma avaliação das vulnerabilidades do sistema associadas a cada um dos cenários considerados. Para a realização dessa avaliação também é necessário um maior nível de detalhamento da arquitetura do sistema (especialmente do detalhamento do ponto de vista da tecnologia e da engenharia), de modo que os componentes e serviços essenciais para a segurança do sistema sejam identificados e as vulnerabilidades da tecnologia associada sejam consideradas (ALBERTS, 2003).

7.2.15 Atividade 3.7 - Análise de segurança integrada

Nesta atividade os resultados e considerações da análise preliminar de segurança são confrontados com os resultados da análise de outros requisitos, para a identificação de eventuais redundâncias ou conflitos. A partir dos requisitos do modelo de negócios relacionam-se todos os requisitos classificados inicialmente como de segurança (**RSG**) que devem ser verificados com relação aos requisitos associados a análise (**RSA**). O resultado dessa verificação conjunta constata que sete requisitos do modelo do negócio devem ser incluídos na análise de segurança (**OB04**, **OB07**, **OB22**, **PR01**, **PR03**, **PR04** e **PR05**), assim como, quatro requisitos do modelo do negócio são identificados como redundantes em relação aos requisitos de segurança já identificados (**PR02**, **PR06**, **PR07** e **RA04**) (AVELINO, 2004).

A seguir, realiza-se uma análise qualitativa de cada ponto de vista em relação a satisfação dos requisitos de segurança elicitados, gerando-se recomendações que devem ser observadas no detalhamento desses pontos de vista, conforme tabela 7.5.

Tabela 7.5: Recomendações da análise de segurança do SSC-STE

Ponto de Vista	Referência	Requisitos Avaliados	Recomendações Preliminares da Análise de Segurança
EMPRESA	Figura 7.1 Contrato Ambiental (AVELINO, 2004)	OB04, RSA06, RSA07, RSA08, RSA09, RSA10	O requisito RSA06 deve substituir o PR06; Os requisitos RSA07, RSA08, RSA09 e RSA10 devem ser incluídos na relação de obrigações do contrato ambiental do SSC
INFORMAÇÃO	Esquema Invariante; Esquema Estático e Dinâmico (AVELINO, 2004)	OB04, OB07, OB22, PR03, PR04, PR05, RSA02, RSA03, RSA06, RSA07, RSA08	Os requisitos OB07, RSA02, RSA06, RSA07 e RSA08 devem ser incluídos como requisitos de segurança no modelo invariante do SSC; O requisito OB22 deve ser classificado como de segurança no modelo invariante
COMPUTAÇÃO	Arquitetura do ponto de vista da Computação (AVELINO, 2004)	OB04, OB07, OB22, PR01, PR03, PR04, PR05, RSA01, RSA02, RSA03, RSA04, RSA05, RSA06, RSA07, RSA08	Para satisfação de OB04, PR01, PR03, PR04, RSA01 e RSA02, no mínimo os objetos: controle de algoritmos, gerenciador de comandos e gerenciador de aquisição devem ter autodiagnóstico (verificável por inspeção) e o seu monitoramento deve ser realizado em objeto(s) independente(s) para ativação dos mecanismos de falha-segura
ENGENHARIA	Arquitetura do ponto de vista da Engenharia (AVELINO, 2004)	OB04, OB07, OB22, PR01, PR03, PR04, PR05, RSA01, RSA02, RSA03, RSA04, RSA05, RSA06, RSA07, RSA08	Para satisfação de OB04, PR01, PR03, PR04, RSA01 e RSA02, cada nó deve ter mecanismos de autodiagnóstico, que deve ativar mecanismos de falha-segura nos nós de interface com processo; Para satisfação de PR05 e RSA02, no mínimo os nós de IHC e de base de dados devem ter mecanismos de persistência e tolerância a falhas; Para satisfação de RSA04 e RSA05, ao menos os nós de controle e interface devem ser replicados; Para satisfação de RSA06 e RSA07, o nó de comunicação deve ter mecanismos de controle de acesso e detecção de intrusos
TECNOLOGIA	Arquitetura do ponto de vista da Tecnologia (AVELINO, 2004)	OB04, OB22, PR01, PR03, PR04, PR05, RSA01, RSA02, RSA03, RSA04, RSA05, RSA06, RSA07, RSA08	Para satisfação de OB04, OB22 e RSA02, os nós de interface, controle e base de dados devem ter capacidade de tratar a qualidade dos dados; Para satisfação de PR01, PR03 e PR04, os códigos dos nós de interface, controle devem ser abertos e verificáveis; Para satisfação de PR04, RSA01 e RSA02, o hardware do nó de interface deve ser verificável; Para satisfação de RSA03, RSA06, RSA07 e RSA08, as redes locais devem ser isoladas das redes externas e o nó de comunicação deve possuir detecção de violação e controle de acesso

7.2.16 Atividade 4.1 - Identificação do perfil do usuário e análise de tarefas

Nesta atividade, identificam-se as características dos usuários do sistema e realiza-se a análise de tarefas contextual do processo. Para a obtenção de resultados práticos, são considerados, neste caso de estudo, os usuários típicos dos sistemas de supervisão e controle do processo elétrico pertencentes à Companhia de Transmissão de Energia Elétrica Paulista – CTEEP (Subestação Nordeste).

Nessa atividade, analisam-se os manuais de operação de uma subestação para a identificação das tarefas dos operadores. Essa análise é complementada pelo levantamento do ambiente de operação, realizado com base nas respostas dos usuários a questionários, bem como através de entrevistas com os mesmos. A partir desse conjunto de informações, identificam-se o perfil do usuário Operador de Subestação, os casos de uso, os respectivos cenários principais e a forma de organização das tarefas, obtendo-se o modelo de organização de tarefas do ponto de vista do Operador de Subestação.

Como resultado, são identificados seis casos de uso para a avaliação de usabilidade, tendo-se o setor de 345kV da SE Nordeste como referência (AVELINO, 2004).

Em função das entrevistas com os operadores e das observações em campo (especialmente focalizando as tarefas do tipo isolamento de equipamento), identificam-se não apenas as tarefas realizadas, mas também uma série de dificuldades, expectativas e possibilidades de melhorias na visão dos Operadores de Subestação. Essas informações permitem um melhor conhecimento do modelo mental desses usuários e devem ser consideradas na elicitação dos requisitos do sistema (AVELINO, 2004).

7.2.17 Atividade 4.2 - Princípios de IHC e restrições ambientais

Nesta atividade, especificam-se as restrições e as potencialidades relacionadas com o ambiente de desenvolvimento da interface homem-computador e definem-se os princípios de projeto que devem nortear o desenvolvimento dessa interface.

A avaliação das definições da plataforma de desenvolvimento (representadas no ponto de vista da tecnologia) não representa, em princípio, nenhuma grande restrição do ponto de vista da usabilidade do sistema, uma vez que essas tecnologias suportam todos os recursos normalmente utilizados em salas de comando e controle (AVELINO, 2004).

Já os princípios de projeto devem ser baseados nas recomendações da literatura de usabilidade, nas normas de interface para o domínio, na filosofia de interface utilizada pela organização e contratos associados com a operação do sistema. As referências utilizadas no capítulo 5 deste trabalho fornecem uma visão geral de vários desses princípios (AVELINO, 2004).

7.2.18 Atividade 4.3 - Elicitação dos requisitos de usabilidade

A partir da análise do perfil dos usuários, análise de tarefas (centrada no foco do Operador de Subestação) e nos princípios de projeto da IHC, são identificados quinze requisitos de usabilidade (**RUS01** a **RUS15**) para o SSC-STE (relacionados na tabela 7.6 com o respectivo nível de sensibilidade para cada ponto de vista) (AVELINO, 2004).

As entrevistas realizadas na análise de tarefas e no levantamento do perfil do usuário permitem identificar outros requisitos, associados com características funcionais do sistema, mas também relacionados com a usabilidade do sistema, denominados requisitos de usabilidade complementares. Nessa análise identificam-se onze requisitos de usabilidade complementares (**RUC01** a **RUC11**, na tabela 7.6), a serem considerados na avaliação integrada dos requisitos (AVELINO, 2004).

Tabela 7.6: Requisitos de usabilidade do SSC-STE

Requisitos de Usabilidade do SSC-STE (Descrição Mnemônica)	NS por Ponto de Vista				
	E	I	C	G	T
RUS01. Facilidade de aprendizado da IHC	1	2	0	0	0
RUS02. Facilidade de uso da IHC	0	1	2	0	3
RUS03. Simplicidade de operação da IHC	1	4	3	0	1
RUS04. Utilização de representações gráficas na IHC	0	0	3	2	5
RUS05. Conforto da representação visual da IHC	0	0	3	1	2
RUS06. Uso limitado de cores na IHC	0	2	1	0	0
RUS07. Coerência e padronização da IHC	2	5	4	0	3
RUS08. Taxa de erros de operação da IHC menor que 1%	1	5	3	0	2
RUS09. Evitar cansaço visual, desatenção e sobrecarga de sinalização na IHC	0	3	2	0	0
RUS10. Sinalização contínua da origem de comando e comunicação com ONS	0	2	1	2	0
RUS11. Priorização de alarmes na IHC	3	2	4	1	0
RUS12. Otimização da IHC para algoritmos de controle	1	2	3	0	0
RUS13. Otimização da IHC para manutenção e parametrização de equipamentos	0	2	3	1	3
RUS14. Padronização e facilidade de formatação de relatórios de operação	0	1	3	2	4
RUS15. Treinamento de usuários de acesso pouco freqüente	3	0	0	0	0
RUC01. Funções para execução otimizada das tarefas do operador	5	4	3	2	0
RUC02. Registro e sinalização passo a passo das ações automáticas	0	3	2	1	0
RUC03. Tempo de resposta de ações críticas do operador maior que 30 min.	2	3	4	4	0
RUC04. Sinalização da causa de bloqueio por intertravamento	0	2	3	3	0
RUC05. Sinalização da fronteira com outras subestações	2	1	0	0	2
RUC06. Ordenação da seqüência de eventos	1	2	2	2	0
RUC07. Integração de parametrização remota sem degradar desempenho	0	0	1	3	4
RUC08. Sinalização contínua de falha do SSC que cause bloqueio operacional	0	1	2	3	0
RUC09. Utilização de senhas pessoais para acesso a supervisão e controle	0	1	2	1	3
RUC10. Sinalização independente de estado e bloqueio de equipamento	1	2	3	4	5
RUC11. Possibilidade de exportação de informações adquiridas em tempo real	0	0	2	4	3
Legenda: E: Empresa, I: Informação, C: Computação, G: Engenharia, T: Tecnologia e respectivo nível de sensibilidade (NS)					

7.2.19 Atividade 4.4 - Elaboração do guia de estilo de IHC

Nesta atividade, definem-se os princípios de usabilidade que devem orientar o projeto da interface homem-computador, elaborando-se um documento denominado Guia de Estilo de Usabilidade do SSC-STE. Este documento é único para todos os pontos de vista do sistema, servindo como uma referência uniforme para todos os desenvolvimentos de IHC relativos ao projeto, permitindo assim a obtenção de uniformidade e coerência na interface que é desenvolvida através das várias visões do sistema.

Como os padrões de projeto da IHC são particulares para cada organização e como o conteúdo inicial deste guia consiste, basicamente, do registro dos resultados das avaliações de usabilidade realizadas nas atividades anteriores, neste caso de estudo este documento não é desenvolvido em detalhe. Entretanto, a título de orientação do trabalho para outras iterações do processo de desenvolvimento, no documento de descrição detalhada deste caso de estudo apresenta-se um padrão para descrição desse guia, aplicável a sistemas de supervisão e controle de subestações de transmissão de energia. Esse padrão é baseado em experiências anteriores do autor deste trabalho na elaboração de especificações de projeto de IHC (AVELINO, 2004).

7.2.20 Atividade 4.5 - Reengenharia do trabalho

Nesta atividade, o modelo de organização de tarefas (elaborado na atividade 4.1) é revisado com o objetivo de que o projeto do SSC-STE possa ser otimizado para satisfação dos objetivos gerais da empresa (**OG01**, **OG02** e **OG03**), bem como possam ser exploradas as potencialidades da automação.

Para essa revisão realizam-se observações de campo e entrevistas com os usuários, que resultam nos agrupamentos de tarefas representados em um modelo de organização de tarefas (AVELINO, 2004).

7.2.21 Atividade 4.6 - Projeto da IHC

Neste primeiro estágio do processo de desenvolvimento da IHC, o nível de detalhe do projeto do sistema é muito reduzido, sendo elaboradas maquetes das telas básicas do sistema, suficientes apenas para a realização de uma avaliação inicial de usabilidade dessa interface pelos usuários do sistema (AVELINO, 2004).

7.2.22 Atividade 4.7 - Prototipação da IHC

Nesta primeira iteração do ciclo de desenvolvimento, a implementação do projeto da IHC é realizada na forma de protótipos em papel e lápis (protótipos de baixa fidelidade) (AVELINO, 2004).

7.2.23 Atividade 4.8 - Avaliação de usabilidade (independente)

Nesta atividade, é realizada a avaliação de usabilidade sobre os protótipos em papel implementados na atividade anterior. Para a verificação da satisfação dos requisitos de usabilidade, utiliza-se o método de avaliação por inspeção e teste denominado de *walkthrough* pluralista.

Nesse processo de avaliação, são envolvidos, numa reunião de avaliação, os representantes da engenharia de sistema e de operação, que simulam a execução dos cenários utilizando sinóticos, janelas e caixas de diálogo representadas em papel. O resumo dos resultados dessa reunião de avaliação de usabilidade é apresentado na tabela 7.7, onde cada requisito de usabilidade é avaliado segundo os critérios: **SA**: Satisfatório, **IS**: Insatisfatório e **NA**: Não Avaliado (AVELINO, 2004).

7.2.24 Atividade 4.9 - Avaliação de usabilidade integrada

Nesta atividade as conclusões da avaliação de usabilidade independente são comparadas com os resultados da análise de outros requisitos para a identificação de eventuais redundâncias, complementaridades ou conflitos entre estes.

Tabela 7.7: Avaliação de usabilidade do SSC-STE

Requisitos de Usabilidade do SSC-STE (Descrição Mnemônica)	Avaliação de Usabilidade		
	SA	IS	NA
RUS01. Facilidade de aprendizado da IHC	X		
RUS02. Facilidade de uso da IHC			X
RUS03. Simplicidade de operação da IHC	X		
RUS04. Utilização de representações gráficas na IHC	X		
RUS05. Conforto da representação visual da IHC			X
RUS06. Uso limitado de cores na IHC		X	
RUS07. Coerência e padronização da IHC	X		
RUS08. Taxa de erros de operação da IHC menor que 1%	X		
RUS09. Evitar cansaço visual, desatenção e sobrecarga de sinalização na IHC			X
RUS10. Sinalização contínua da origem de comando e comunicação com ONS		X	
RUS11. Priorização de alarmes na IHC			X
RUS12. Otimização da IHC para algoritmos de controle	X		
RUS13. Otimização da IHC para manutenção e parametrização de equipamentos			X
RUS14. Padronização e facilidade de formatação de relatórios de operação	X		
RUS15. Treinamento de usuários de acesso pouco freqüente			X
RUC01. Funções para execução otimizada das tarefas do operador	X		
RUC02. Registro e sinalização passo a passo das ações automáticas	X		
RUC03. Tempo de resposta de ações críticas do operador maior que 30 min.			X
RUC04. Sinalização da causa de bloqueio por intertravamento		X	
RUC05. Sinalização da fronteira com outras subestações			X
RUC06. Ordenação da seqüência de eventos			X
RUC07. Integração de parametrização remota sem degradar desempenho			X
RUC08. Sinalização contínua de falha do SSC que cause bloqueio operacional			X
RUC09. Utilização de senhas pessoais para acesso a supervisão e controle	X		
RUC10. Sinalização independente de estado e bloqueio de equipamento		X	
RUC11. Possibilidade de exportação de informações adquiridas em tempo real			X
Legenda: SA: Satisfatório, IS: Insatisfatório e NA: Não Avaliado			

Para a verificação do relacionamento entre os requisitos elicitados a partir do modelo de negócios do sistema e os requisitos de usabilidade, é realizada uma atividade de tempestade cerebral, onde ambos os conjuntos de requisitos são sistematicamente comparados. O resultado dessa verificação conjunta constata que doze requisitos do modelo do negócio devem ser incluídos na avaliação de

usabilidade (**OB06, OB14, OB26, OB27, OB28, DE09, DE15, RE02, RA01, RA02, RA03 e RA04**), assim como, sete requisitos do modelo do negócio são identificados como redundantes em relação aos requisitos de usabilidade já identificados (**OB01, OB22, OB24, PE02, PR02, PR05 e PR06**) (AVELINO, 2004).

A identificação de novos requisitos de usabilidade (especialmente os ligados a outros tipos de usuários como o Engenheiro de Manutenção) gera a necessidade de novas avaliações de usabilidade (e consequentemente de novos protótipos) para considerar cenários ainda não analisados. Como resultado têm-se a identificação de dois novos casos de uso de usabilidade (AVELINO, 2004).

7.2.25 Atividade 5.1 - Análise dos requisitos integrados

Nesta atividade, os requisitos são analisados de modo conjunto, considerando os resultados das análises específicas de segurança e usabilidade realizadas nas atividades anteriores, bem como considerando-se todos os pontos de vista de arquitetura. O objetivo é identificar potenciais conflitos ou lacunas no conjunto da especificação de requisitos.

Essa atividade é realizada utilizando-se ferramentas de inspeção do tipo reuniões de revisão de projeto e análises tipo *walkthrough* cognitivo, onde cada grupo de especialistas apresenta seus resultados que são discutidos pelos outros participantes que não estiveram envolvidos nessa análise específica, obtendo-se uma verificação cruzada dos resultados das diversas análises realizadas.

Para facilitar o trabalho de análise, os requisitos devem ser ordenados por prioridade, de modo que sejam discutidos inicialmente aqueles requisitos de maior sensibilidade em relação aos vários pontos de vista do sistema. Os critérios de prioridade, descritos na seção 6.5.1, baseiam-se no cálculo da *densidade de energia* (E_r_i) e a *energia média* ($\sum E_r_i$) de cada requisito, utilizando-se como referência o nível de sensibilidade (**NS**) já determinado nas análise anteriores (AVELINO, 2004).

Os dez requisitos considerados como os mais prioritários para análise integrada são compostos de quatro requisitos de segurança (**RSA06**, **RSA02**, **RSA07** e **PR06**), quatro requisitos de usabilidade (**RUC10**, **RUC01**, **RUS07** e **RUC03**) e dois requisitos de obrigações funcionais (**OB01** e **OB11**), confirmando a hipótese de que os RNF são aqueles que apresentam maior influência em todos os pontos de vista.

O trabalho de análise integrada desse conjunto de requisitos é realizado a partir dos resultados das análises independentes realizadas anteriormente (tabelas 7.2, 7.5 e 7.7), resultando na identificação dos problemas apresentados na tabela 7.8.

7.2.26 Atividade 5.2 - Análise de compromissos

Os resultados verificados na análise integrada dos dez requisitos de maior abrangência identificam alguns problemas de conflito e de redundância, que devem ser analisados em relação aos seus casos de uso e criticidade, bem como em relação aos objetivos do empreendimento. O resultado dessa análise crítica, realizada pelo autor deste trabalho, é apresentado na tabela 7.8, onde são identificadas as ações que devem ser realizadas para mitigação dos conflitos ou cobertura das lacunas detectadas na especificação de requisitos do SSC-STE.

A verificação da conformidade entre os vários pontos de vista é obtida a partir da avaliação dos requisitos que são comuns a essas visões. Nesta primeira iteração do ciclo de desenvolvimento, não são identificadas discrepâncias entre os cinco pontos de vista modelados, mesmo porque, o nível de detalhe considerado ainda não permite a verificação minuciosa de características específicas do sistema.

O detalhamento do sistema prossegue com o início de um novo ciclo de desenvolvimento, sendo que cada um dos pontos de vista de arquitetura deve ser detalhado com o seu conjunto específico de requisitos a ser satisfeito. Nas tabelas 7.9 a 7.13 são relacionados, para cada ponto de vista de arquitetura, os seus requisitos, com o respectivo nível de sensibilidade e seu estado de satisfação, ao final da primeira iteração aplicando-se o ciclo de desenvolvimento da MERUSA.

Tabela 7.8: Análise de compromissos para o SSC-STE

REQUISITO	DESCRIÇÃO DO PROBLEMA	ALTERA	AÇÕES
RSA06	Pode exigir mecanismos complexos que reduzem desempenho; Pode dificultar acesso a manutenção remota; Redundante com PR06	RUC07 RUS13, RA02, RA03 PR06	Pesquisar técnicas de bloqueio de acesso com desempenho; Priorizar segurança de acesso em detrimento a usabilidade; Substituir PR06 por RSA06A (RSA06 com inclusão de acesso a redes internas)
RUC10	Exige informação de origem duplicada	DE15	Evitar conflito de duplicidade de origem mantendo processamento independente para cada origem
RUC01	Redundante com relação a OB01	OB01	Substituir RUC01 por OB01A (OB01 com inclusão de referência a modelo de tarefas)
RUS07	Cuidados na escolha da tecnologia para desenvolvimento da IHC		Incluir padrões do ONS no Manual de Estilo
RSA02	Sinalização de invalidez pode reduzir disponibilidade e limitar opções de tecnologia; Redundante com OB22, OB24, PR02 e RUC08	DE22, RE04, RE05 OB22, OB24, PR02, RUC08	A sinalização de invalidez do dado é essencial mesmo em detrimento da disponibilidade; Substituir OB22, PR02 por RSA02A (RSA02 com inclusão das condições de invalidez e envio a centros remotos)
RUC03	Pode exigir mecanismos complexos que reduzem desempenho;	DE22, RE04, RE05	Evitar automatismos complexos. A prioridade é disponibilidade que representa a estabilidade do sistema
RSA07	Pode exigir mecanismos complexos que reduzem desempenho; Pode dificultar acesso a manutenção remota;	RUC07 RUS13, RA02, RA03	Pesquisar técnicas de detecção de violação com desempenho; Priorizar a segurança de acesso mesmo em detrimento a usabilidade
PR06	Redundante com RSA06	RSA06	Substituir PR06 por RSA06A
PR07	Redundante com RSA03	RSA03	Substituir PR07 por RSA03
RA04	Redundante com RSA08	RSA08	Substituir RA04 por RSA08

Os critérios para a determinação do estado de satisfação dos requisitos são os seguintes:

- **Satisfatório (SA):** é o estado daquele requisito que foi avaliado e, no estágio atual de desenvolvimento do sistema, não foram encontrados riscos de que o sistema não vá satisfazê-lo;
- **Insatisfatório (IS):** é o estado daquele requisito que foi avaliado e foram encontrados conflitos com outros requisitos ou não conformidades que representam uma tendência ou um risco significativo de que sistema não vá satisfazer a esse requisito;
- **Não avaliado (NA):** é o estado daquele requisito que ainda não foi avaliado, devido ao nível de detalhamento do sistema ou à prioridade adotada nas análises.

No caso de estudo atual priorizam-se as análises dos requisitos de usabilidade e segurança e, dessa forma, a maioria dos requisitos funcionais não foi avaliada.

Tabela 7.9: Requisitos do SSC-STE do ponto de vista da Empresa

Requisitos da Empresa do SSC-STE (Descrição Mnemônica)	NS	SA	IS	NA
OB01A. Integração dos sistemas da subestação conforme modelo de tarefas	4	X		
OB03. Compatibilização com sistemas legados, utilizando tecnologia digital	2			X
OB04. Automonitoramento e autodiagnóstico, com bloqueio por defeito	1	X		
OB05. Arquitetura aberta e utilização de protocolos normalizados	1			X
OB06. Intervenção de manutenção sem desligamento de equipamentos	1			X
OB11. Recursos para aquisição, registro, interconexão com ONS e GPS	2	X		
OB17. Supervisão do Controle Automático de Tensão – CAT	1			X
OB18. Coordenação, supervisão e controle das instalações acessórias	2			X
OB20. Supervisão SOE sem filtragem na mesma interligação com ONS	1			X
PE01. Recursos de supervisão e controle para interconexão com ONS	1			X
PE02. Recursos de supervisão e controle para sala de comando remota	1			X
DE09. Habilitação dos Operadores para as atividades de tempo real	2			X
DE19. Filtragem por banda morta definida de acordo com ONS	2			X
RA01: Supervisão do histórico de operação dos equipamentos principais	1			X
RA02: Alteração remota dos parâmetros de configuração da BD	1		X	
RA03: Parametrização através de terminais remotos em redes de comunicação	1		X	
RSA06A. Controle de acesso a informações e parametrização de privilégios	1		X	
RSA07. Mecanismos de detecção de violação de barreira de acesso remoto	1		X	

RSA08. Limitação de acesso remoto para parametrização do SSC ou relês	1	X		
RSA09. Treinamento dos operadores para evitar geração de eventos inseguros	5	X		
RSA10. Treinamento dos operadores para detecção de falhas e intrusos	5	X		
RUS01. Facilidade de aprendizado da IHC	1	X		
RUS03. Simplicidade de operação da IHC	1	X		
RUS07. Coerência e padronização da IHC	2	X		
RUS08. Taxa de erros de operação da IHC menor que 1%	1	X		
RUS11. Priorização de alarmes na IHC	3			X
RUS12. Otimização da IHC para algoritmos de controle	1	X		
RUS15. Treinamento de usuários de acesso pouco freqüente	3			X
RUC03. Tempo de resposta de ações críticas do operador maior que 30 min.	2		X	
RUC05. Sinalização da fronteira com outras subestações	2			X
RUC06. Ordenação da seqüência de eventos	1			X
RUC10. Sinalização independente de estado e bloqueio de equipamento	1		X	
Legenda: E: Empresa, I: Informação, C: Computação, G: Engenharia, T: Tecnologia e respectivo nível de sensibilidade (NS)				

Tabela 7.10: Requisitos do SSC-STE do ponto de vista da Informação

Requisitos da Informação do SSC-STE (Descrição Mnemônica)	NS	SA	IS	NA
OB01A. Integração dos sistemas da subestação conforme modelo de tarefas	4	X		
OB03. Compatibilização com sistemas legados, utilizando tecnologia digital	3			X
OB04. Automonitoramento e autodiagnóstico, com bloqueio por defeito	2	X		
OB05. Arquitetura aberta e utilização de protocolos normalizados	1			X
OB06. Intervenção de manutenção sem desligamento de equipamentos	1			X
OB07. Proteções com supervisão de tensão para bloqueio de operação e alarme	2	X		
OB08. Seleção de modo de operação através de chave seletora e/ou do SSC	2			X
OB09. Registrador de Perturbações com contato para sinalização de eventos	2			X
OB10. Registro de sinais da teleproteção para análise de perturbações	2			X
OB11. Recursos para aquisição, registro, interconexão com ONS e GPS	2	X		
OB12. Comunicação de dados analógicos parametrizável	2			X
OB13. Comunicação de dados digitais por exceção	2			X
OB14. Registro SOE monitora principais pontos da instalação	2			X
OB17. Supervisão do Controle Automático de Tensão – CAT	2			X
OB18. Coordenação, supervisão e controle das instalações acessórias	2			X
OB20. Supervisão SOE sem filtragem na mesma interligação com ONS	1			X
OB21. Informações para ONS em valor de engenharia sem filtragem	1			X

OB24. Protocolo com identificação do estado operacional dos sistemas	2			X
OB26. Supervisão de grandezas analógicas e sinalizações de estado com tempo	3			X
OB27. Supervisão da potência ativa trifásica em pontos de interligação	3			X
OB28. Supervisão e armazenamento de estados e grandezas do processo	3			X
PE01. Recursos de supervisão e controle para interconexão com ONS	2			X
PE02. Recursos de supervisão e controle para sala de comando remota	2			X
PE04. Período de varredura de comunicação configurável por tipo de grandeza	2			X
PR03. Automatismos de controle sem condições inseguras	2	X		
PR04. Automatismos de controle não causam indisponibilidade operacional	2	X		
PR05. Informações de IHC não podem induzir Operador a erros	2	X		
DE15. Medições obtidas a partir de fonte única de informação	1		X	
DE18. Períodos de aquisição parametrizáveis	1			X
RA01: Supervisão do histórico de operação dos equipamentos principais	2			X
RA02: Alteração remota dos parâmetros de configuração da BD	2		X	
RA03: Parametrização através de terminais remotos em redes de comunicação	2		X	
RSA02A. Supervisão e comunicação de informações com validade dos dados	5		X	
RSA03. Redes de comunicação com protocolo de comunicação seguro	1	X		
RSA06A. Controle de acesso a informações e parametrização de privilégios	3		X	
RSA07. Mecanismos de detecção de violação de barreira de acesso remoto	2		X	
RSA08. Limitação de acesso remoto para parametrização do SSC ou relês	1	X		
RUS01. Facilidade de aprendizado da IHC	2	X		
RUS02. Facilidade de uso da IHC	1			X
RUS03. Simplicidade de operação da IHC	4	X		
RUS06. Uso limitado de cores na IHC	2		X	
RUS07. Coerência e padronização da IHC	5	X		
RUS08. Taxa de erros de operação da IHC menor que 1%	5	X		
RUS09. Evitar cansaço visual, desatenção e sobrecarga de sinalização na IHC	3			X
RUS10. Sinalização contínua da origem de comando e comunicação com ONS	2		X	
RUS11. Priorização de alarmes na IHC	2			X
RUS12. Otimização da IHC para algoritmos de controle	2	X		
RUS13. Otimização da IHC para manutenção e parametrização de equipamento	2		X	
RUS14. Padronização e facilidade de formatação de relatórios de operação	1	X		
RUC02. Registro e sinalização passo a passo das ações automáticas	3	X		
RUC03. Tempo de resposta de ações críticas do operador maior que 30 min.	3		X	
RUC04. Sinalização da causa de bloqueio por intertravamento	2		X	
RUC05. Sinalização da fronteira com outras subestações	1			X
RUC06. Ordenação da seqüência de eventos	2			X
RUC08. Sinalização contínua de falha do SSC que cause bloqueio operacional	1			X

RUC09. Utilização de senhas pessoais para acesso a supervisão e controle	1	X		
RUC10. Sinalização independente de estado e bloqueio de equipamento	2		X	
Legenda: E: Empresa, I: Informação, C: Computação, G: Engenharia, T: Tecnologia e respectivo nível de sensibilidade (NS)				

Tabela 7.11: Requisitos do SSC-STE do ponto de vista da Computação

Requisitos da Computação do SSC-STE (Descrição Mnemônica)	NS	SA	IS	NA
OB01A. Integração dos sistemas da subestação conforme modelo de tarefas	3	X		
OB02. Utilização de tecnologia digital numérica	1			X
OB03. Compatibilização com sistemas legados, utilizando tecnologia digital	2			X
OB04. Automonitoramento e autodiagnóstico, com bloqueio por defeito	3		X	
OB05. Arquitetura aberta e utilização de protocolos normalizados	3			X
OB06. Intervenção de manutenção sem desligamento de equipamentos	2			X
OB07. Proteções com supervisão de tensão para bloqueio de operação e alarme	1	X		
OB08. Seleção de modo de operação através de chave seletora e/ou do SSC	1			X
OB09. Registrador de Perturbações com contato para sinalização de eventos	1			X
OB10. Registro de sinais da teleproteção para análise de perturbações	1			X
OB11. Recursos para aquisição, registro, interconexão com ONS e GPS	3	X		
OB12. Comunicação de dados analógicos parametrizável	3			X
OB13. Comunicação de dados digitais por exceção	3			X
OB16. Testes de conectividade entre subsistemas	2			X
OB17. Supervisão do Controle Automático de Tensão – CAT	1			X
OB18. Coordenação, supervisão e controle das instalações acessórias	1			X
OB20. Supervisão SOE sem filtragem na mesma interligação com ONS	2			X
OB21. Informações para ONS em valor de engenharia sem filtragem	1			X
OB23. Compatibilidade de protocolos de comunicação com ONS	2			X
OB24. Protocolo com identificação do estado operacional dos sistemas	2			X
OB26. Supervisão de grandezas analógicas e sinalizações de estado com tempo	2			X
OB27. Supervisão da potência ativa trifásica em pontos de interligação	1			X
OB28. Supervisão e armazenamento de estados e grandezas do processo	1			X
PE01. Recursos de supervisão e controle para interconexão com ONS	1			X
PE02. Recursos de supervisão e controle para sala de comando remota	1			X
PE04. Período de varredura de comunicação configurável por tipo de grandeza	2			X
PR01. Não geração de comandos espúrios em caso de falha ou manutenção	3		X	
PR03. Automatismos de controle sem condições inseguras	3		X	

PR04. Automatismos de controle não causam indisponibilidade operacional	3		X	
PR05. Informações de IHC não podem induzir Operador a erros	3	X		
DE01. Uniformidade de desempenho entre sistemas integrados e equipamentos	2			X
DE02. Projeto, fabricação e ensaio de equipamentos conforme normas ABNT	1			X
DE05. Taxas de aquisição de dados (supervisão de estado a cada 4s)	1			X
DE06. Aquisição de dados com selo de tempo (exatidão e resolução <1ms)	1			X
DE07. Atualização de mudança de estado no centro de operação remoto <4s	1			X
DE08. Relógio do sistema e selo de tempo com a mesma exatidão e resolução	1			X
DE10. Atualização das medições analógicas (parametrizável) <4s	1			X
DE11. Sinalizações de estado reportadas por exceção	2			X
DE12. Varredura de integridade parametrizável, sob demanda ou por evento	2			X
DE14. Período de aquisição de grandezas de estado pelo ONS típico de 2s	1			X
DE15. Medições obtidas a partir de fonte única de informação	1		X	
DE17. Idade máxima do dado <tempo varredura +4 ou +10s (por exceção <8s)	2			X
DE18. Períodos de aquisição parametrizáveis	2			X
DE22. Disp. crítica >98,5%; Disp. geral >97,5%; Disp. concentradores >99%	2		X	
RE02. Consoles de IHC redundantes	1			X
RE03. Capacidade registro em tempo real >300 alarmes e >300 eventos	1			X
RE04. Disp. funções essenciais >99,99%; Disp. sistema completo >99,95%	2		X	
RE05. MTBF > 20.000horas	2		X	
RE06. MTTR < 2horas	1			X
RE09. Atualização de estados na IHC da SE <1s	1			X
RE11. Atualização de telas na SE <1s	1			X
RE12. Execução de comando remoto <2,5s	1			X
RE13. Expansibilidade física e lógica >20%	2			X
RE14. Ocupação máxima dos processadores <70%	1			X
RA01: Supervisão do histórico de operação dos equipamentos principais	2			X
RA02: Alteração remota dos parâmetros de configuração da BD	2		X	
RA03: Parametrização através de terminais remotos em redes de comunicação	1		X	
RSA01. Característica de falha-segura nos controladores do SSC	1		X	
RSA02A. Supervisão e comunicação de informações com validade dos dados	4		X	
RSA03. Redes de comunicação com protocolo de comunicação seguro	2	X		
RSA04. Replicação de lógicas de intertravamento em vários nós do SSC	1	X		
RSA05. Lógica de intertravamento local operacional mesmo na falha do SSC	1	X		
RSA06A. Controle de acesso a informações e parametrização de privilégios	4		X	
RSA07. Mecanismos de detecção de violação de barreira de acesso remoto	4		X	
RSA08. Limitação de acesso remoto para parametrização do SSC ou relês	3	X		
RUS02. Facilidade de uso da IHC	2			X

RUS03. Simplicidade de operação da IHC	3	X		
RUS04. Utilização de representações gráficas na IHC	3	X		
RUS05. Conforto da representação visual da IHC	3			X
RUS06. Uso limitado de cores na IHC	1		X	
RUS07. Coerência e padronização da IHC	4	X		
RUS08. Taxa de erros de operação da IHC menor que 1%	3	X		
RUS09. Evitar cansaço visual, desatenção e sobrecarga de sinalização na IHC	2			X
RUS10. Sinalização contínua da origem de comando e comunicação com ONS	1		X	
RUS11. Priorização de alarmes na IHC	4			X
RUS12. Otimização da IHC para algoritmos de controle	3	X		
RUS13. Otimização da IHC para manutenção e parametrização de equipamento	3		X	
RUS14. Padronização e facilidade de formatação de relatórios de operação	3	X		
RUC02. Registro e sinalização passo a passo das ações automáticas	2	X		
RUC03. Tempo de resposta de ações críticas do operador maior que 30 min.	4		X	
RUC04. Sinalização da causa de bloqueio por intertravamento	3		X	
RUC06. Ordenação da seqüência de eventos	2			X
RUC07. Integração de parametrização remota sem degradar desempenho	1		X	
RUC08. Sinalização contínua de falha do SSC que cause bloqueio operacional	2			X
RUC09. Utilização de senhas pessoais para acesso a supervisão e controle	2	X		
RUC10. Sinalização independente de estado e bloqueio de equipamento	3		X	
RUC11. Possibilidade de exportação de informações adquiridas em tempo real	2			X
Legenda: E: Empresa, I: Informação, C: Computação, G: Engenharia, T: Tecnologia e respectivo nível de sensibilidade (NS)				

Tabela 7.12: Requisitos do SSC-STE do ponto de vista da Engenharia

Requisitos da Engenharia do SSC-STE (Descrição Mnemônica)	NS	SA	IS	NA
OB01A. Integração dos sistemas da subestação conforme modelo de tarefas	3	X		
OB02. Utilização de tecnologia digital numérica	1			X
OB03. Compatibilização com sistemas legados, utilizando tecnologia digital	1			X
OB04. Automonitoramento e autodiagnóstico, com bloqueio por defeito	3		X	
OB05. Arquitetura aberta e utilização de protocolos normalizados	3			X
OB06. Intervenção de manutenção sem desligamento de equipamentos	2			X
OB07. Proteções com supervisão de tensão para bloqueio de operação e alarme	1	X		
OB08. Seleção de modo de operação através de chave seletora e/ou do SSC	1			X
OB09. Registrador de Perturbações com contato para sinalização de eventos	1			X
OB10. Registro de sinais da teleproteção para análise de perturbações	1			X

OB11. Recursos para aquisição, registro, interconexão com ONS e GPS	3	X		
OB12. Comunicação de dados analógicos parametrizável	3			X
OB13. Comunicação de dados digitais por exceção	3			X
OB16. Testes de conectividade entre subsistemas	2			X
OB17. Supervisão do Controle Automático de Tensão – CAT	1			X
OB18. Coordenação, supervisão e controle das instalações acessórias	1			X
OB20. Supervisão SOE sem filtragem na mesma interligação com ONS	2			X
OB21. Informações para ONS em valor de engenharia sem filtragem	1			X
OB23. Compatibilidade de protocolos de comunicação com ONS	2			X
OB24. Protocolo com identificação do estado operacional dos sistemas	2			X
OB26. Supervisão de grandezas analógicas e sinalizações de estado com tempo	2			X
OB27. Supervisão da potência ativa trifásica em pontos de interligação	1			X
OB28. Supervisão e armazenamento de estados e grandezas do processo	1			X
PE01. Recursos de supervisão e controle para interconexão com ONS	3			X
PE02. Recursos de supervisão e controle para sala de comando remota	3			X
PE04. Período de varredura de comunicação configurável por tipo de grandeza	2			X
PR01. Não geração de comandos espúrios em caso de falha ou manutenção	3		X	
PR03. Automatismos de controle sem condições inseguras	3		X	
PR04. Automatismos de controle não causam indisponibilidade operacional	3		X	
PR05. Informações de IHC não podem induzir Operador a erros	3		X	
DE01. Uniformidade de desempenho entre sistemas integrados e equipamentos	3			X
DE02. Projeto, fabricação e ensaio de equipamentos conforme normas ABNT	1			X
DE05. Taxas de aquisição de dados (supervisão de estado a cada 4s)	2			X
DE06. Aquisição de dados com selo de tempo (exatidão e resolução <1ms)	2			X
DE07. Atualização de mudança de estado no centro de operação remoto <4s	2			X
DE08. Relógio do sistema e selo de tempo com a mesma exatidão e resolução	2			X
DE10. Atualização das medições analógicas (parametrizável) <4s	2			X
DE11. Sinalizações de estado reportadas por exceção	2			X
DE12. Varredura de integridade parametrizável, sob demanda ou por evento	2			X
DE14. Período de aquisição de grandezas de estado pelo ONS típico de 2s	2			X
DE15. Medições obtidas a partir de fonte única de informação	1		X	
DE17. Idade máxima do dado <tempo varredura +4 ou +10s (por exceção <8s)	2			X
DE18. Períodos de aquisição parametrizáveis	2			X
DE22. Disp. crítica >98,5%; Disp. geral >97,5%; Disp. concentradores >99%	3		X	
RE02. Consoles de IHC redundantes	3			X
RE03. Capacidade registro em tempo real >300 alarmes e >300 eventos	3			X
RE04. Disp. funções essenciais >99,99%; Disp. sistema completo >99,95%	3		X	
RE05. MTBF > 20.000horas	3		X	

RE06. MTTR < 2horas	1			X
RE09. Atualização de estados na IHC da SE <1s	2			X
RE11. Atualização de telas na SE <1s	2			X
RE12. Execução de comando remoto <2,5s	2			X
RE13. Expansibilidade física e lógica >20%	2			X
RE14. Ocupação máxima dos processadores <70%	1			X
RA01: Supervisão do histórico de operação dos equipamentos principais	2			X
RA02: Alteração remota dos parâmetros de configuração da BD	2		X	
RA03: Parametrização através de terminais remotos em redes de comunicação	3		X	
RSA01. Característica de falha-segura nos controladores do SSC	4		X	
RSA02A. Supervisão e comunicação de informações com validade dos dados	3		X	
RSA03. Redes de comunicação com protocolo de comunicação seguro	3	X		
RSA04. Replicação de lógicas de intertravamento em vários nós do SSC	5		X	
RSA05. Lógica de intertravamento local operacional mesmo na falha do SSC	3		X	
RSA06A. Controle de acesso a informações e parametrização de privilégios	4		X	
RSA07. Mecanismos de detecção de violação de barreira de acesso remoto	5		X	
RSA08. Limitação de acesso remoto para parametrização do SSC ou relês	2	X		
RUS04. Utilização de representações gráficas na IHC	2	X		
RUS05. Conforto da representação visual da IHC	1			X
RUS10. Sinalização contínua da origem de comando e comunicação com ONS	2		X	
RUS11. Priorização de alarmes na IHC	1			X
RUS13. Otimização da IHC para manutenção e parametrização de equipamento	1		X	
RUS14. Padronização e facilidade de formatação de relatórios de operação	2	X		
RUC02. Registro e sinalização passo a passo das ações automáticas	1	X		
RUC03. Tempo de resposta de ações críticas do operador maior que 30 min.	4		X	
RUC04. Sinalização da causa de bloqueio por intertravamento	3		X	
RUC06. Ordenação da sequência de eventos	2			X
RUC07. Integração de parametrização remota sem degradar desempenho	3		X	
RUC08. Sinalização contínua de falha do SSC que cause bloqueio operacional	3			X
RUC09. Utilização de senhas pessoais para acesso a supervisão e controle	1	X		
RUC10. Sinalização independente de estado e bloqueio de equipamento	4		X	
RUC11. Possibilidade de exportação de informações adquiridas em tempo real	4			X
Legenda: E: Empresa, I: Informação, C: Computação, G: Engenharia, T: Tecnologia e respectivo nível de sensibilidade (NS)				

Tabela 7.13: Requisitos do SSC-STE do ponto de vista da Tecnologia

Requisitos da Tecnologia do SSC-STE (Descrição Mnemônica)	NS	SA	IS	NA
OB01A. Integração dos sistemas da subestação conforme modelo de tarefas	2	X		
OB02. Utilização de tecnologia digital numérica	3			X
OB03. Compatibilização com sistemas legados, utilizando tecnologia digital	3			X
OB04. Automonitoramento e autodiagnóstico, com bloqueio por defeito	2		X	
OB05. Arquitetura aberta e utilização de protocolos normalizados	3			X
OB06. Intervenção de manutenção sem desligamento de equipamentos	2			X
OB09. Registrador de Perturbações com contato para sinalização de eventos	1			X
OB11. Recursos para aquisição, registro, interconexão com ONS e GPS	3	X		
OB12. Comunicação de dados analógicos parametrizável	3			X
OB13. Comunicação de dados digitais por exceção	3			X
OB16. Testes de conectividade entre subsistemas	2			X
OB17. Supervisão do Controle Automático de Tensão – CAT	1			X
OB18. Coordenação, supervisão e controle das instalações acessórias	1			X
OB20. Supervisão SOE sem filtragem na mesma interligação com ONS	1			X
OB23. Compatibilidade de protocolos de comunicação com ONS	3			X
OB24. Protocolo com identificação do estado operacional dos sistemas	3			X
PE01. Recursos de supervisão e controle para interconexão com ONS	3			X
PE02. Recursos de supervisão e controle para sala de comando remota	3			X
PE03. Evolução tecnológica com outros protocolos padrões	3			X
PE04. Período de varredura de comunicação configurável por tipo de grandeza	3			X
PR01. Não geração de comandos espúrios em caso de falha ou manutenção	3		X	
PR03. Automatismos de controle sem condições inseguras	3		X	
PR04. Automatismos de controle não causam indisponibilidade operacional	3		X	
PR05. Informações de IHC não podem induzir Operador a erros	3	X		
DE01. Uniformidade de desempenho entre sistemas integrados e equipamentos	3			X
DE02. Projeto, fabricação e ensaio de equipamentos conforme normas ABNT	3			X
DE03. Atendimento aos requisitos de compatibilidade eletromagnética	3			X
DE05. Taxas de aquisição de dados (supervisão de estado a cada 4s)	3			X
DE06. Aquisição de dados com selo de tempo (exatidão e resolução <1ms)	3			X
DE07. Atualização de mudança de estado no centro de operação remoto <4s	3			X
DE08. Relógio do sistema e selo de tempo com a mesma exatidão e resolução	3			X
DE10. Atualização das medições analógicas (parametrizável) <4s	3			X
DE11. Sinalizações de estado reportadas por exceção	2			X
DE12. Varredura de integridade parametrizável, sob demanda ou por evento	3			X

DE14. Período de aquisição de grandezas de estado pelo ONS típico de 2s	3			X
DE16. Medições de tensão com precisão <1% (demais analógicas <2%)	2			X
DE17. Idade máxima do dado <tempo varredura +4 ou +10s (por exceção <8s)	2			X
DE18. Períodos de aquisição parametrizáveis	2			X
DE22. Disp. crítica >98,5%; Disp. geral >97,5%; Disp. concentradores >99%	3		X	
RE01. Condições ambientais (TA <40C, UR 50 a 90%)	3			X
RE02. Consoles de IHC redundantes	2			X
RE03. Capacidade registro em tempo real >300 alarmes e >300 eventos	2			X
RE04. Disp. funções essenciais >99,99%; Disp. sistema completo >99,95%	3		X	
RE05. MTBF > 20.000horas	3		X	
RE06. MTTR < 2horas	3			X
RE09. Atualização de estados na IHC da SE <1s	3			X
RE11. Atualização de telas na SE <1s	3			X
RE12. Execução de comando remoto <2,5s	3			X
RE13. Expansibilidade física e lógica >20%	3			X
RE14. Ocupação máxima dos processadores <70%	3			X
RE15. Reserva mínima de memória >30%	2			X
RA02: Alteração remota dos parâmetros de configuração da BD	3		X	
RA03: Parametrização através de terminais remotos em redes de comunicação	3		X	
RSA01. Característica de falha-segura nos controladores do SSC	5		X	
RSA02A. Supervisão e comunicação de informações com validade dos dados	3		X	
RSA03. Redes de comunicação com protocolo de comunicação seguro	5		X	
RSA04. Replicação de lógicas de intertravamento em vários nós do SSC	2	X		
RSA05. Lógica de intertravamento local operacional mesmo na falha do SSC	4	X		
RSA06A. Controle de acesso a informações e parametrização de privilégios	5		X	
RSA07. Mecanismos de detecção de violação de barreira de acesso remoto	1		X	
RSA08. Limitação de acesso remoto para parametrização do SSC ou relês	2		X	
RUS02. Facilidade de uso da IHC	3			X
RUS03. Simplicidade de operação da IHC	1	X		
RUS04. Utilização de representações gráficas na IHC	5	X		
RUS05. Conforto da representação visual da IHC	2			X
RUS07. Coerência e padronização da IHC	3	X		
RUS08. Taxa de erros de operação da IHC menor que 1%	2	X		
RUS13. Otimização da IHC para manutenção e parametrização de equipamento	3		X	
RUS14. Padronização e facilidade de formatação de relatórios de operação	4	X		
RUC05. Sinalização da fronteira com outras subestações	2			X
RUC07. Integração de parametrização remota sem degradar desempenho	4		X	
RUC09. Utilização de senhas pessoais para acesso a supervisão e controle	3	X		

RUC10. Sinalização independente de estado e bloqueio de equipamento	5		X	
RUC11. Possibilidade de exportação de informações adquiridas em tempo real	3			X
Legenda: E: Empresa, I: Informação, C: Computação, G: Engenharia, T: Tecnologia e respectivo nível de sensibilidade (NS)				

7.3 Resumo dos Resultados

Os resultados observados nesta primeira iteração do ciclo de desenvolvimento da arquitetura do SSC-STE, utilizando-se a MERUSA, são os seguintes:

- A partir do modelo de negócios do SSC-STE, elicitaram-se 79 requisitos: 28 caracterizados como obrigações (**OB01 a OB28**), 4 como possibilidades de expansões (**PE01 a PE04**), 7 como proibições (**PR01 a PR07**), 23 como requisitos de desempenho (**DE01 a DE23**) e 17 como restrições específicas (**RE01 a RE17**);
- Na análise dos requisitos funcionais (utilizando-se a comparação entre modelos de cenários – UCM e modelo dos objetivos - GRL), foram identificados mais 4 requisitos adicionais (**RA01 a RA04**);
- A avaliação da sensibilidade desses requisitos em relação a cada ponto de vista da arquitetura revelou que 13 desses requisitos (cerca de 16% do total) eram redundantes (**OB15, OB19, OB25, DE04, DE13, DE20, DE21, DE23, RE07, RE08, RE10, RE16 e RE17**), sendo eliminados do processo de análise (e do ciclo de desenvolvimento da arquitetura);
- O processo de análise de segurança identificou 11 requisitos que já haviam sido elicitados anteriormente como relacionados com a segurança (**OB04, OB07, OB22, PR01, PR02, PR03, PR04, PR05, PR06, PR07 e RA04**). Utilizando-se ferramentas próprias de análise de segurança (FTA e HAZOP), bem como realizando-se a comparação entre modelos de cenários de segurança – UCM e modelo dos objetivos de segurança – GRL, foram elicitados 10 novos requisitos de segurança (**RSA01 a RSA10**). A análise de segurança integrada mostrou que 4 desse total de 21 requisitos associados à

segurança (cerca de 19% do total), provenientes da análise dos requisitos funcionais, eram parcialmente redundantes com os novos requisitos e foram eliminados, resultando no conjunto final de 17 requisitos utilizados para análise de segurança;

- O processo de avaliação de usabilidade identificou que 19 requisitos já elicitados anteriormente eram relacionados com a usabilidade (**OB01, OB06, OB14, OB22, OB24, OB26, OB27, OB28, PE02, PR02, PR05, PR06, DE09, DE15, RE02, RA01, RA02, RA03 e RA04**). A partir da análise do perfil dos usuários, da análise de tarefas (centrada no foco do Operador de Subestação) e dos princípios de projeto da IHC, foram elicitados 15 novos requisitos de usabilidade (**RUS01 a RUS15**) e mais 11 requisitos complementares de usabilidade (**RUC01 a RUC11**). Na análise de usabilidade integrada foi identificado que 7 desse total de 45 requisitos associados à usabilidade (cerca de 16% do total), provenientes da análise dos requisitos funcionais, eram parcialmente redundantes com os novos requisitos e foram eliminados, resultando no conjunto de 38 requisitos utilizados para avaliação de usabilidade;
- No processo de análise dos requisitos integrados, foi estabelecida uma priorização destes em função do nível de sensibilidade de cada um para as cinco visões de arquitetura (definido como *energia média* do requisito). A análise dos requisitos integrados envolveu apenas os dez requisitos de maior *energia média* e identificou 4 requisitos redundantes (**OB22, PR02, PR06 e RUC01**) e a análise de compromissos gerou a revisão de 3 requisitos (**OB01A, RSA02A e RSA06A**).

Assim, o processo de desenvolvimento, considerando apenas os requisitos funcionais, identificou um total de 70 requisitos válidos (já excluídos os 13 redundantes e incluídos os 4 adicionais resultantes da análise comparada entre cenários e objetivos). Do total de requisitos válidos 11 são relacionados com a segurança e 19 com a usabilidade. Esse conjunto de requisitos pode ser considerado o

máximo que pode ser obtido nas aplicações atuais de sistemas de supervisão e controle para subestações de transmissão do SIN, uma vez que o mesmo foi gerado a partir das especificações do ONS e da análise de especificações de requisitos de diversas concessionárias nacionais de transmissão de energia, portanto elicitados de modo *ad hoc* por especialistas do domínio dessas empresas (EPTE, 1998); (ANA, 2001); (CEMIG, 2002); (CPFL, 2002); (FURNAS, 2002); (ONS, 2002); (IBAMA, 2003); (MMA, 2003); (MME, 2003).

Utilizando-se a MERUSA, nos ciclos específicos de desenvolvimento para cada RNF, foram elicitados adicionalmente 6 requisitos de segurança e 19 requisitos de usabilidade.

Portanto, pode-se concluir que, com apenas a primeira iteração do ciclo de desenvolvimento completada (e trabalhando-se num elevado nível de abstração das arquiteturas), a MERUSA ampliou em 54% o número de requisitos de segurança e em 100% os requisitos de usabilidade na especificação do sistema.

Finalmente, deve ser observado que, dos 11 requisitos relacionados com a segurança elicitados na análise funcional, 7 foram incluídos no contrato ambiental do SSC em função da experiência do autor deste trabalho na área de análise de segurança (**PR01, PR02, PR03, PR04, PR05, PR06 e PR07**). Isto mostra que a quantidade de requisitos relacionados com a segurança, usualmente encontrada nas especificações de requisitos utilizadas pelas concessionárias nacionais de transmissão de energia, é bem menor que a considerada na análise anterior, indicando que a utilização da MERUSA pode potencialmente contribuir com bem mais que 54% na ampliação da cobertura dos requisitos de segurança.

8 CONCLUSÃO

Este trabalho apresenta uma metodologia de especificação de requisitos de usabilidade e segurança orientada para arquitetura (MERUSA) aderente ao processo de desenvolvimento de sistemas baseados em objetos distribuídos.

Na MERUSA, o modelo de referência ODP é utilizado como infra-estrutura padrão para descrição do sistema distribuído, sendo verificado que uma boa parte das características já previstas nesse modelo são aderentes às necessidades típicas (requisitos) dos sistemas distribuídos, em especial aos sistemas relacionados com a segurança. Entretanto, algumas características fundamentais de sistemas relacionados com a segurança (tais como simplicidade, falha segura, intertravamentos, verificações de razoabilidade) não estão previstas no modelo ODP e precisam ser analisadas desde o início do processo de desenvolvimento (no nível da arquitetura), para que não gerem compromissos com os demais requisitos do modelo.

De modo similar, os atributos de usabilidade do sistema também podem ser adequadamente representados nos pontos de vista do modelo ODP, desde que sejam realizadas avaliações das características de usabilidade em todos os pontos de vista do modelo, utilizando-se técnicas específicas para elicitar e avaliar esse tipo de requisito.

O ATAM também é utilizado como referência básica na MERUSA, fornecendo o processo básico de análise de arquiteturas e identificação de compromissos entre requisitos e características de arquitetura.

A MERUSA considera que os requisitos de segurança e usabilidade devem possuir ciclos de elicitação e análise parcialmente independentes das atividades equivalentes dos requisitos funcionais. Essa independência parcial caracteriza-se por atividades de elicitação e análise que utilizam ferramentas próprias para esse tipo de requisitos, sendo conduzidas por grupos de analistas não associados às demais atividades de projeto; ao mesmo tempo, os resultados dessas análises individuais, bem como os grupos de analistas que as geraram, devem periodicamente (ao final de uma iteração completa de um ciclo da metodologia) analisar em conjunto todos os

requisitos do sistema, interagindo, nesse momento, de modo intensivo com todos os envolvidos com o projeto. Esses ciclos de elicitação e análise de requisitos individualizados, seguidos de análise conjunta, podem ser repetidos várias vezes, em função da evolução (detalhamento) dos requisitos e/ou da arquitetura do sistema.

Para a avaliação da MERUSA é desenvolvida uma meta-arquitetura para os Sistemas de Transmissão de Energia do Sistema Elétrico Brasileiro, que permite identificar as características de um Sistema de Supervisão e Controle aplicado nesse domínio (SSC-STE). A especificação dos requisitos de usabilidade e segurança do SSC-STE é o caso de estudo utilizado para a verificação da eficácia da MERUSA.

A verificação da eficácia da MERUSA é realizada analisando-se os resultados obtidos na primeira iteração do ciclo de desenvolvimento do SSC-STE. Apesar do nível de conhecimento a respeito do sistema ter sido limitado ao primeiro ciclo de iteração, a especificação de requisitos resultante pode ser comparada com especificações de requisitos elaborados por empresas concessionárias de transmissão de energia nacionais, para subcontratação de sistemas de supervisão e controle de sistemas de transmissão de energia. Comparando-se ambos os conjuntos de requisitos podem ser analisados aspectos de completeza, precisão e rastreabilidade dos requisitos obtidos com a utilização da MERUSA em relação aos requisitos exigidos nos editais de subcontratação desse tipo de sistemas (elicitados de modo *ad hoc* por especialistas do domínio das empresas concessionárias de transmissão de energia nacionais).

8.1 Contribuições da Tese

A MERUSA estabelece um conjunto organizado de atividades e técnicas, que contribuem para a obtenção da completeza da especificação de requisitos não funcionais, assim como orientam as definições de arquitetura do sistema (considerando-se que essa arquitetura descreve o sistema sob vários pontos de vista), provendo a identificação dos compromissos entre os requisitos e provendo critérios para auxiliar a resolução dos mesmos.

Os resultados da aplicação da MERUSA à especificação de um sistema de supervisão e controle distribuído de transmissão de energia (SSC-STE) mostram que a cobertura dos requisitos de segurança e usabilidade pode ser ampliada entre 54 e 100% já em uma iteração do ciclo de desenvolvimento da arquitetura (onde o número de ferramentas de análise é limitado e os resultados das avaliações são eminentemente qualitativos). Esse aumento significativo da cobertura dos requisitos, associado à participação dos diversos grupos de *stakeholders* nas análises integradas de requisitos, assegura maior completeza da especificação de requisitos com a utilização da metodologia proposta.

As atividades de avaliação de sensibilidade dos requisitos, associadas às análises integradas dos requisitos não funcionais, realizadas no caso de estudo do SSC-STE, identificaram vários requisitos com conteúdo redundante ou parcialmente redundante (cerca de 16% dos RF, 19% dos requisitos de segurança e 16% dos requisitos de usabilidade), mostrando que a MERUSA contribui significativamente para a precisão da especificação de requisitos, reduzindo a ambigüidade entre os requisitos e melhorando a compreensão das necessidades do negócio (especialmente através das atividades de análises individuais seguidas pelas análises integradas dos requisitos).

Outro aspecto relevante da MERUSA é a característica evolutiva dos ciclos de elicitação e análise de requisitos. A metodologia considera que o processo de elicitação se inicia com a identificação das necessidades do domínio (definidas no modelo de negócio) e com a análise dos requisitos a partir dos primeiros níveis de abstração da arquitetura do sistema. Esse processo acompanha o detalhamento, a implantação, a operação e a manutenção do sistema de maneira sistemática, registrando as considerações (*rationale*) que levam à elicitação, à decomposição e à substituição dos requisitos, bem como registrando a associação entre a satisfação dos requisitos e as características da arquitetura e/ou as características do sistema, através de análises específicas e análises integradas dos requisitos (tipo *Walkthrough*). Essa sistematização do processo, acompanhada dos registros das análises, fornecem informações para uma perfeita rastreabilidade dos requisitos, cujo estado evolutivo

(estado de satisfação de cada requisito em relação a cada ponto de vista de arquitetura) pode ser aferido ao final de cada ciclo iterativo da metodologia.

A meta-arquitetura para os Sistemas de Transmissão de Energia do Sistema Elétrico Brasileiro, elaborada para a avaliação da MERUSA, foi desenvolvida utilizando-se uma grande generalidade de conceitos. Essa generalização, associada ao elevado nível de abstração utilizado, permitem considerá-la aplicável para o desenvolvimento de qualquer sistema digital de supervisão e controle que atue neste domínio. Com pequenas modificações, esse modelo pode ser aplicado também para a especificação de outros sistemas digitais integrantes do SIN, tais como, sistemas de supervisão e controle de geração, sistemas de telecomunicações, sistemas de monitoramento, entre outros, uma vez que os requisitos do domínio de aplicação (neste caso originários da especificação do SIN) são similares.

Além da proposição de uma metodologia para especificação de requisitos de usabilidade e segurança orientada para arquitetura de sistemas distribuídos e abertos, este trabalho de tese apresenta as seguintes contribuições:

➤Agrega numa mesma metodologia um método de desenvolvimento de arquitetura baseado em avaliações de sensibilidade de requisitos (ATAM) e um método de descrição de arquitetura para sistemas de processamento distribuído (RM-ODP); propondo a integração de ambos para a especificação e análise dos RNF de sistemas distribuídos e abertos;

➤Propõe uma metodologia de especificação e análise dos RNF baseada na utilização das ferramentas tradicionais de análise desses requisitos, na identificação prematura dos atributos de qualidade do sistema, na utilização de diversas visões complementares da arquitetura do sistema (cada uma voltada para um tipo de *stakeholder* específico), na análise de risco dos compromissos entre os requisitos (obtida através da análise integrada dos diversos requisitos e das diversas visões de arquitetura do sistema) e na resolução dos conflitos entre esses requisitos (definido como um ponto de sincronização do processo de desenvolvimento da arquitetura, onde é realizada a avaliação da conformidade de todos os modelos de arquitetura);

➤A análise da adequação do modelo de referência ODP às técnicas utilizadas no desenvolvimento de projetos com características de segurança mostra que as funções ODP já definidas no modelo de referência não atendem a todas as necessidades desse tipo de projeto. Assim, evidencia-se a necessidade de que funções ODP adicionais venham a ser definidas para esse modelo, de modo a atender aspectos como: mecanismos de falha segura, segregação de fluxo de dados críticos, mecanismos de intertravamento para bloqueio, mecanismos de verificação de erros por reversão, mecanismos de verificação de erros por razoabilidade;

➤A interpretação das cinco visões do modelo ODP como arquiteturas individuais, caracterizadas por tipos de RNF, e orientadas para cada tipo de *stakeholder* específico, proposta neste trabalho também é uma contribuição. Esse tipo de enfoque mostra-se adequado para facilitar a identificação dos RNF e permitir a fácil rastreabilidade dos requisitos. Outra contribuição para o modelo ODP é a realização da análise individual e integrada das visões do modelo ODP no ponto de sincronismo da metodologia, assegurando não só a verificação da conformidade entre as visões do modelo, mas principalmente a análise dos compromissos entre os requisitos e a resolução de eventuais conflitos entre as visões consideradas;

➤O processo de obtenção da meta-arquitetura para o Sistema de Transmissão de Energia do Sistema Elétrico Brasileiro é outra contribuição a ser destacada neste trabalho. Esse processo utiliza como ponto de partida o modelo de negócios do sistema, considera os contratos ambientais e os conceitos de abstração de arquitetura fornecidos pelo modelo ODP, para a obtenção de uma representação dos principais elementos que compõem o sistema (e de suas inter-relações principais), denominada de meta-arquitetura do sistema. Esse processo pode ser utilizado em outros domínios de aplicação para obtenção de uma meta-arquitetura de referência, a qual permite instanciar outras arquiteturas mais detalhadas para o sistema;

➤A aplicação da MERUSA no caso de estudo do Sistema de Transmissão de Energia do Sistema Elétrico Brasileiro mostra que deve haver uma ampliação na qualidade dos requisitos relacionados com a segurança, bem como nos requisitos de

usabilidade, usualmente considerada nas especificações para contratação de Sistemas de Supervisão e Controle utilizadas pela concessionárias nacionais de transmissão de energia. Apesar do aumento do número de requisitos implicar em custos maiores no desenvolvimento e implantação desses sistemas, os benefícios obtidos durante a sua operação (com a redução da sua indisponibilidade e maior facilidade de utilização) devem recompensar de modo significativo esse acréscimo de custo inicial.

8.2 Trabalhos Semelhantes

Um processo semelhante a este foi proposto por Grünbalher, Egyed e Medvidivic (2001), onde uma ferramenta automática foi elaborada para mapear requisitos de software na arquitetura do sistema; entretanto esse processo não considerou o tratamento dos requisitos não funcionais, limitando-se a classificá-los (GRÜNBACHER, 2001).

Outro trabalho similar foi apresentado por Alleman (2000), envolvendo a análise de requisitos (funcionais e não funcionais) centrada no desenvolvimento da arquitetura de sistemas de manufatura. O autor também propôs a utilização do modelo ODP como estratégia de descrição da arquitetura, porém não discute a questão da análise e resolução dos compromissos entre requisitos e não apresenta tratamento específico para requisitos não funcionais (ALLEMAN, 2000).

Uma interessante metodologia centrada em fatores humanos (análise de tarefas) e envolvendo requisitos de segurança, usabilidade e desempenho foi proposta por Daouk e Leveson (2001). Nesta proposta os processos de elicitação e análise de requisitos de segurança e usabilidade seguem processos paralelos (similar à MERUSA), entretanto, não existe uma atividade explícita de análise de compromissos, nem é evidente a necessidade de sincronização com a arquitetura do sistema, o que pode inviabilizar o desenvolvimento de sistemas complexos (o exemplo da aplicação dessa metodologia foi no desenvolvimento de uma das partes de um controlador de tráfego aéreo) (DAOUK, 2001).

8.3 Trabalhos Futuros

Como trabalhos futuros considera-se a necessidade da aplicação da MERUSA em outros domínios, especialmente em casos onde os requisitos de segurança (no sentido de *safety*) tenham um peso maior, para uma avaliação mais precisa da sua eficácia com relação a aspectos de segurança.

Deve ser considerada a possibilidade de que a MERUSA possa, eventualmente, ser expandida para outros tipos de RNF em sistemas distribuídos e abertos (especialmente confiabilidade, desempenho, interoperabilidade e expansibilidade).

Outra área de pesquisa futura é o desenvolvimento de ferramentas (ou investigar a adaptação de ferramentas já disponíveis) para facilitar o processo de verificação cruzada de requisitos. Ferramentas automáticas para suporte aos registros de rastreabilidade também devem ser investigadas, especialmente aquelas que possuem integração com diagramas de representação em UML.

A meta-arquitetura desenvolvida para Sistemas de Transmissão de Energia no Sistema Interligado Nacional pode ser detalhada (e instanciada numa aplicação real) para consolidar essa representação como um padrão mais completo de especificação para esse tipo de sistema. Adicionalmente, seria interessante desenvolver variações dessa meta-arquitetura para outros sistemas no domínio do SIN (tais como: sistemas de geração, sistemas de monitoramento), de modo a garantir um padrão mínimo de qualidade na especificação de todos os sistemas digitais envolvidos nesse domínio.

Na metodologia apresentada não foi discutida em detalhe a dependência entre os diversos grupos de análise relacionados com os RNF (que realizam suas atividades de modo independente) e o grupo de desenvolvimento do sistema principal (dedicado ao desenvolvimento dos requisitos funcionais). A interface entre esses grupos independentes necessita de estudos mais detalhados para sua definição, incluindo os limites de responsabilidade específicos. Deve ser observado que o detalhamento dessa interface e das respectivas responsabilidades geralmente será especializado por tipo e porte da aplicação.

9 BIBLIOGRAFIA DE REFERÊNCIA

- (ABNT, 1999) **ABNT - Guia para Utilização das Normas Sobre Avaliação de Qualidade de Produto de Software – ISO/IEC 9126 e ISO/IEC 14598.** – Associação Brasileira de Normas Técnicas – SC10 – Subcomitê de Software, PR, 1999
- (ALBERTS, 2003) **ALBERTS, C. J.; DOROFEE, A. J. Managing Information Security Risks – The OCTAVE Approach.** Addison-Wesley – Pearson Education, Inc., MA, 2003
- (ALLEMAN, 2000) **ALLEMAN, G. B. Architecture-Centered Information Systems In The Manufacturing Domain.** Technical Report - Niwot Ridge Consulting, <http://www.niwotridge.com>, 2000
- (AMYOT, 2002) **AMYOT, D.; MUSSBACHER, G. URN – vers une nouvelle norme pour la description visuelle d'exigences.** Colloque Francophone sur l'Ingénierie des Protocoles - CFIP'02 – Montréal - May 2002. Disponível em: <<<http://www.UseCaseMaps.org/pub/cfip2002.pdf>>>. Acesso em: 01 de mar. 2004
- (AMYOT, 2003) **AMYOT, D. Introduction to the User Requirements Notation: Learning by Example.** (CSRS'03), May 2003. Disponível em: <<<http://www.UseCaseMaps.org/urn>>>. Acesso em: 01 de mar. 2004
- (ANA, 2001) **ANA - Regime Institucional,** Agência Nacional de Águas, abr. 2001. Disponível em: <<<http://www.ana.gov.br/Institucional/default.asp>>>. Acesso em: 26 de jan. 2004
- (AVELINO, 2004) **AVELINO, V. F. MERUSA: Caso de Estudo – Sistema de Supervisão e Controle de Subestação de Transmissão de Energia do Sistema Interligado Nacional.** Departamento de Engenharia de Computação e Sistemas Digitais da Escola Politécnica da Universidade de São Paulo, São Paulo, SP, 2004
- (BANNON, 2000) **BANNON, L. Use-centred Design and Evaluation of Systems.** Web site from European Commission Framework 4 – Cooperative Technologies for Complex Work – COTCOS,

2000. Disponível em: <<<http://www-sv.cict.fr/cotcos/pjs/Authors/Bannon.htm>>>. Acesso em: 11 de abr. 2004
- (BASS, 1999) BASS, L.; KAZMAN, R. **Architecture-Based Development**. Technical Report: CMU/SEI-99-TR-007, Software Engineering Institute, Carnegie Mellon University, PA, 1999
- (BASS, 2001) BASS, L.; KLEIN, M., MORENO, G. **Applicability of General Scenarios to the Architecture Tradeoff Analysis Method**. Technical Report: CMU/SEI-2001-TR-014, Software Engineering Institute, Carnegie Mellon University, PA, 2001
- (BECERRA, 1998) BECERRA, J. L. R. **Aplicabilidade do Padrão de Processamento Distribuído e Aberto nos Projetos de Sistema Abertos de Automação**. Tese de Doutorado – Departamento de Engenharia de Computação e Sistemas Digitais da Escola Politécnica da Universidade de São Paulo, São Paulo, SP, 1998
- (BEVAN, 1999) BEVAN, N. **Quality in Use: Meeting User Needs for Quality**. Journal of Systems and Software, Vol. 49, N° 1 (89-96), 1999. Disponível em: <<<http://www.usability.serco.com/papers/qiuse.pdf>>>. Acesso em: 01 de jun. 2004
- (BROWN, 1998) BROWN, M.; LEVESON, N. G. **Modeling Controller Tasks for Safety Analysis**. HESSD'98 - 2nd Workshop on Human Error, Safety, and System Development, Washington, April 1998. Disponível em: <<http://www.dcs.gla.ac.uk/~johnson/papers/seattle_hessd.html>>. Acesso em: 11 de abr. 2004
- (CARLSON, 1975) CARLSON, A. B. **Communication Systems: An Introduction to Signals and Noise In Electrical Communication**. McGraw-Hill, NY, 1975
- (CEMIG, 2002) CEMIG - **Sistema de Supervisão, Controle, Proteção e Registro de Perturbações – SE Bom Despacho 3 – Especificação Básica – ER/SE-4362**. Companhia Energética de Minas Gerais, Minas Gerais, 2002

- (CHRISTEL, 1992) CHRISTEL, M. G.; KANG, K. C. **Issues in Requirements Elicitation**. Technical Report: CMU/SEI-92-TR-12, Software Engineering Institute, Carnegie Mellon University, PA, 1992
- (CPFL, 2002) **CPFL - Automação SE's Baixada Santista – Especificação Técnica – DES/2002**. Companhia Piratininga de Força e Luz, São Paulo, SP, 2002
- (CYSNEIROS, 2001) CYSNEIROS, L. M. **Requisitos Não Funcionais: Da Elicitação ao Modelo Conceitual**. PhD Thesis – Departamento de Informática – Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro, RJ, 2001
- (DAOUK, 2001) DAOUK, M.; LEVESON, N. G. **An Approach to Human-Centered Design**. Massachusetts Institute of Technology – Department of Aeronautics and Astronautics – Research by NASA – NAG-1-1894, 2001. Disponível em: <<<http://web.mit.edu/afs/athena.mit.edu/activity/h/hfes/www/daouk1.pdf>>>. Acesso em: 11 de abr. 2004
- (DE BRUIN, 2001) DE BRUIN, H.; VAN VLIET, H. **Scenario-Based Generation and Evaluation of Software Architectures**. Generative and Component-Based Software Engineering (GCSE'01), LNCS, Springer-Verlag, Germany, 2001
- (DEVANBU, 2000) DEVANBU, P. T.; STUBBLEBINE, S. **Software Engineering for Security: A Roadmap**. ICSE'2000 - Proc. 22nd International Conference on Software Engineering, In: "The Future of Software Engineering", Anthony Finkelstein (Ed.), ACM Press, Limerick, 2000
- (DIX, 1998) DIX, A. J.; FINLAY, J. E.; ABWOD, G. D.; BEALE, R. **Human-Computer Interaction**. Prentice-Hall, NJ, 1998
- (DORFMAN, 1997) DORFMAN, M. **Requirements Engineering**. In: Software Requirements Engineering, Dorfman, M.; Thayer, R. H., IEEE Computer Society Press, Los Alamitos, CA, 1997
- (DRORI, 2003) DRORI, O. **Integration of HCI Needs with SE Methods Using OODPM Methodology**. IFIP - WG2.7/13.4 - Interact 2003 Workshop, In: "Closing the Gaps: Software

- Engineering and Human-Computer Interaction”, M. B. Harning & J. Vanderdonckt (Eds.), Université catholique de Louvain - Institut d’Administration et de Gestion, Belgium, (20-26), 2003. Disponível em: <<<http://www.se-hci.org/bridging/interact/proceedings.html>>>. Acesso em: 01 de jun. 2004
- (EAMES, 1999) EAMES, D. P.; MOFFETT, J. **The Integration of Safety and Security Requirements Engineering**. In: "Computer Safety, Reliability and Security: 18th International Conference, SAFECOMP'99, (27-29), 1999
- (EEC, 2004) **EEC Note No. 01/04 Review of Techniques to Support the EATMP Safety Assessment Methodology**, EUROCONTROL Experimental Centre – European Organization for the Safety Air Navigation, Project SRD-3-E1, Volume II - Annex, Brussels, 2004
- (EPTE, 1998) **EPTE - ETT Miguel Reale – Sistema de Proteção, Medição, Controle e Supervisão Digital – Especificação Técnica – ET-TCN/E-001/98**. Empresa Paulista de Transmissão de Energia Elétrica, São Paulo, 1998
- (FAA, 2000) **FAA System Safety Handbook: Practices and Guidelines For Conducting System Safety Engineering and Management – Chapter 10: System Software Safety**, US Federal Aviation Administration Technical Center, Washington, DC, 2000
- (FAROOQUI, 1995) FAROOQUI, K.; LOGRIPO, L. **Architecture for open Distributed Software Systems**. Handbook of Parallel and Distributed Computing. McGraw Hill Publishers, NY, 1995
- (FILGUEIRAS, 2000) FILGUEIRAS, L. V. **Introdução ao Projeto de Interfaces Homem-Computador – Notas de Aula**. Departamento de Engenharia de Computação e Sistemas Digitais da Escola Politécnica da Universidade de São Paulo, São Paulo, SP, 2001

- (FIORINI, 1998) FIORINI, S. T.; LEITE, J. C. S. P.; LUCENA, C. J. P. **Organizando Processos de Requisitos**. Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro, 1998
- (FURNAS, 2002) **FURNAS - Especificações Técnicas – Painéis de Proteção e Controle para SE Samambaia – 6ª Ampliação – CO.DAQ.G.0013.2202**. Furnas Centrais Elétricas, Rio de Janeiro, 2002
- (GARDINER, 1998) GARDINER, S. **Testing Safety Related Software**. Springer-Verlag, London, 1998
- (GRANOLLERS, 2003) GRANOLLERS, T.; LORÉS, J.; PERDRIX, F. **Usability Engineering Process Model. Integration with Software Engineering**. HCI International 2003, Creta, Greece, June 2003. Disponível em: <<<http://griho.udl.es/publications/2003>>>. Acesso em: 01 de jun. 2004
- (GROSS, 2001) GROSS, D.; YU, E. **From Non-Functional Requirements to Design through Patterns**. In: Requirements Engineering, (6:18-36), Springer-Verlag, London, 2001
- (GROZDANOVIC, 2001) GROZDANOVIĆ, M. **Human Factor and Preventive Engineering** Facta Universitatis – Working and Living Environmental Protection - Vol. 2 - Nº 1 - 2001. Disponível em: <<<http://facta.junis.ni.ac.yu/facta/walep/walep2001>>>. Acesso em: 01 de mar. 2004
- (GRÜNBACHER, 2001) GRÜNBACHER, P.; EGYED, A.; MEDVIDOVIC, N. **Reconciling Software Requirements and Architectures: The CBSP Approach**. IEEE Transactions on Software Engineering (202-211), 2001
- (HARTSON, 2004) HARTSON, R.; HIX, D. **CS5714 Usability Engineering – An Iterative, Evaluation-Centered Life Cycle For Interaction Development**. Web site from Virginia Tech – Department of Computer Science – course notes, 2004. Disponível em: <<<http://courses.cs.vt.edu/~cs5714/spring2004>>>. Acesso em: 01 de jun. 2004

- (HFDG, 2001) **HFDG - Human Factors Design Guide Update (Report Number DOT/FAA/CT-96/01): A Revision to Chapter 8 - Computer Human Interface Guidelines.** US Federal Aviation Administration Technical Center – DOT/FAA/ACT-01/08, Washington, DC, 2001
- (HRS, 2002) **HRS/HSP-002-REP-01 Technical Review of Human Performance Models and Taxonomies of Human Error in ATM (HERA),** EUROCONTROL - European Air Traffic Management Programme, Edition 1.0, Brussels, 2002
- (HUNTER, 1998) HUNTER, A.; NUSEIBECH, B. **Managing Inconsistent Specifications: Reasoning, Analysis and Action.** ACM Transactions on Software Engineering and Methodology, Vol. 7, N° 4 (335-367), 1998
- (HUSSEY, 2000) HUSSEY, A.; ATCHINSON, B. **Safe Architectural Design Principles.** Technical Report: 00-19, Software Verification Research Center - School of Information Technology, The University of Queensland, Queensland, Australia, 2000
- (IBAMA, 2003) **IBAMA - Regime Institucional,** Instituto Brasileiro do Meio Ambiente, jun. 2003. Disponível em: <<<http://www.ibama.gov.br/>>>. Acesso em: 26 de jan. 2004
- (IEC-1508-1, 1997) **IEC-1508-1 Functional Safety of Electrical/Electronic/Programmable Electronic Safety-Related Systems – Part 1: General requirements.** International Electrotechnical Commission, 1997
- (IEC-1508-4, 1997) **IEC-1508-4 Functional Safety of Electrical/Electronic/Programmable Electronic Safety-Related Systems – Part 4: Definitions and abbreviations.** International Electrotechnical Commission, 1997
- (IEC-1508-5, 1997) **IEC-1508-5 Functional Safety of Electrical/Electronic/Programmable Electronic Safety-Related Systems – Part 5: Examples of methods for the determination of safety integrity levels.** International Electrotechnical Commission, 1997

- (IEEE-STD-1471, 2000) **IEEE Std 1471 IEEE Recommended Practice for Architectural Description of Software-Intensive Systems.** IEEE Computer Society – Software Engineering Standards Committee, New York, NY, 2000
- (IEEE-STD-610.12, 1990) **IEEE Std 610.12 IEEE Standard Glossary of Software Engineering Terminology.** IEEE Computer Society – Software Engineering Standards Committee, New York, NY, 1990
- (IEEE-STD-830, 1993) **IEEE Std 830 IEEE Recommended Practice for Software Requirements Specifications.** IEEE Computer Society – Software Engineering Standards Committee, New York, NY, 1993
- (ISO-10746-1, 1995) **ISO-10746-1 Recommendation X.901/ISO/IEC 10746-1: Basic Reference Model of Open Distributed Processing - Part 1: Overview and Guide to Use,** International Organization for Standardization, 1995
- (ISO-10746-2, 1995) **ISO-10746-2 Recommendation X.902/ISO/IEC 10746-2: Basic Reference Model of Open Distributed Processing - Part 2: Foundations,** International Organization for Standardization, 1995
- (ISO-10746-3, 1995) **ISO-10746-3 Recommendation X.903/ISO/IEC 10746-3: Basic Reference Model of Open Distributed Processing - Part 3: Architecture,** International Organization for Standardization, 1995
- (ISO-10746-4, 1995) **ISO-10746-4 Recommendation X.904/ISO/IEC 10746-4: Basic Reference Model of Open Distributed Processing - Part 4: Architecture Semantics,** International Organization for Standardization, 1995
- (ISO-13407, 1999) **ISO 13407:1999 Human-centred design processes for interactive systems,** International Organization for Standardization, 1999
- (ISO-15414, 2002) **ISO/IEC 15414:2002(E) Information Technology – Open Distributed Processing - Reference Model -Enterprise**

- Language**, International Organization for Standardization, 2002
- (ISO-9126-1, 2000) **ISO/IEC FDIS 9126-1:2000 Software Engineering - Product quality – Part 1: Quality Model**, International Organization for Standardization, 2000
- (ISO-9126-4, 2000) **ISO/IEC DTR 9126-4:2001 Software Engineering – Product quality – Part 4: Quality in use metrics**, International Organization for Standardization, 2000
- (ISBELL, 1999) ISBELL, D.; HARDIN, M.; UNDERWOOD, J. **MARS Climate Orbiter Team Finds Likely Cause of Loss** – Press Release 99-113, NASA, 1999
- (ITU-T-Z.150, 2003) **ITU-T Recommendation Z.150 (02/2003) User Requirements Notation (URN) – Language requirements and framework**. International Telecommunication Union – Series Z: Languages and General Software Aspects for Telecommunications Systems – Formal Description Techniques (FDT) – User Requirements Notation (URN), Feb. 2003. Disponível em: <<<http://www.UseCaseMaps.org/urnl>>>. Acesso em: 15 de mar. 2004
- (JACOBSON, 1992) JACOBSON, I., CHRISTERSON, M.; JONSSON, P.; OVERGAARD, G. **Object-Oriented Software Engineering: A Use Case Driven Approach**. ACM Press/Addison-Wesley, Mass., 1992
- (KAZMAN, 1998) KAZMAN, R. et al; **The Architecture Tradeoff Analysis Method**. Technical Report: CMU/SEI-98-TR-008, Software Engineering Institute, Carnegie Mellon University, PA, 1998
- (KAZMAN, 2000) KAZMAN, R.; KLEIN, M.; CLEMENTS, P. **ATAM: Method for Architecture Evaluation**. Technical Report: CMU/SEI-2000-TR-004, Software Engineering Institute, Carnegie Mellon University, PA, 2000
- (KIRWAN, 1994) KIRWAN, B. **A Guide to Practical Human Reliability Assessment**. Taylor and Francis, London, 1994

- (KLEIN, 1999) KLEIN, M.; KAZMAN, R. **Attribute-Based Architectural Styles**. Technical Report: CMU/SEI-99-TR-022, Software Engineering Institute, Carnegie Mellon University, PA, 1999
- (KNIGHT, 2002) KNIGHT, J. C. **Safety Critical Systems: Challenges and Directions**. ICSE'2002 – Proceedings of the 24th International Conference on Software Engineering, (547-550), 2002
- (KOTONYA, 1995) KOTONYA, G.; SOMMERVILLE, I. **Integrating Safety Analysis and Requirements Engineering** Lancaster University – Computing Department, Cooperative Systems Engineering Group – Technical Report: CSEG/9/1995. Disponível em: <<http://www.comp.lancs.ac.uk/computing/research/cseg/95_rep.html>>. Acesso em: 05 de jan. 2004
- (KOTONYA, 1998) KOTONYA, G.; SOMMERVILLE, I. **Requirements Engineering – Processes and Techniques**. John Wiley & Sons, NY, 1998
- (LARSEN, 2000) LARSEN, L. B. **Introduction to Usability Engineering**. Web site from Center for PersonKommunikation, 2000. Disponível em: <<http://it.civil.auc.dk/it/education/reports/usability_intro.pdf>>. Acesso em: 11 de abr. 2004
- (LEITE, 1992) LEITE, J. C. S. P. **Enhancing the Semantics of Requirements Statements**. Proceedings of the XII International Conference of the Sociedad Chilena de Ciencia de la Computacion, (281-297), Santiago, 1992
- (LEITE, 1994) LEITE, J. C. S. P. **Engenharia de Requisitos – Notas de Aula**. Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro, 1994
- (LEVESON, 1996) LEVESON, N.G. **Safeware: System Safety and Computers**. Addison-Wesley, MA, 1995
- (LIU, 2001) LIU, L.; YU, E. **From Requirements to Architectural Design – Using Goals and Scenarios**. In: “From Software Requirements to Architectures Workshop” (STRAW 2001), Toronto, Canada, 2001

- (LUTZ, 1999) LUTZ, R. R.; SHAW, H. **Applying Adaptive Safety Analysis Techniques**. Proc. 10th International Symposium on Software Reliability Engineering, Boca Raton, FL 1999
- (LUTZ, 2000) LUTZ, R. R. **Software Engineering for Safety: A Roadmap**. ICSE'2000 - Proc. 22nd International Conference on Software Engineering, In: "The Future of Software Engineering", Anthony Finkelstein (Ed.), ACM Press, Limerick, 2000
- (MAYHEW, 1999) MAYHEW, D. J. **The Usability Engineering Lifecycle: a practitioner's handbook for user interface design**. Morgan Kaufmann Publishers, CA, 1999
- (MMA, 2003) **MMA - Estrutura Regimental do Ministério do Meio Ambiente**, Ministério do Meio Ambiente, jun. 2003. Disponível em: <<http://www.mma.gov.br/cgmi/institu/estrutut/estr_org.cfm>>. Acesso em: 26 de jan. 2004
- (MME, 2003) **MME - Modelo Institucional do Setor Elétrico**, Ministério de Minas e Energia, dez. 2003. Disponível em: <<<http://www.mme.gov.br>>>. Acesso em: 26 de jan. 2004
- (NATO, 2001) **NATO – The Human Factor in System Reliability - Is Human Performance Predictable?**. North Atlantic Treaty Organization, RTO Meeting Proceedings 32 – RTO-MP-032 - AC/323(HFM)TP/12 – RTO/NATO, 2001. Disponível em: <<[http://www.pmfhk.cz/Katedry/RTO/MP-032/MP-032-\\$\\$ALL.pdf](http://www.pmfhk.cz/Katedry/RTO/MP-032/MP-032-$$ALL.pdf)>>. Acesso em: 01 de mar. 2004
- (NBR-9241-11, 2002) **NBR 9241-11:2002 Requisitos Ergonômicos para Trabalho de Escritórios com Computadores – Parte 11 – Orientações sobre Usabilidade**. Associação Brasileira de Normas Técnicas, 2002
- (NEA, 1998) **NEA/CSNI/R(98)1 Critical Operator Actions: Human Reliability Modeling and Data Issues**, Principal Working Group No.5 – Task 94-1, Nuclear Energy Agency – Committee on the Safety of Nuclear Installations, OECD – Organisation for Economic Co-operation and Development, Paris, 1998

- (NIELSEN, 1992) NIELSEN, J. **The Usability Engineering Life Cycle**. Computer, Mar (12-22), 1992
- (NIELSEN, 1993) NIELSEN, J. **Usability Engineering**. Academic Press, NY, 1993
- (NIELSEN, 1994) NIELSEN, J.; Mack, R. **Usability Inspection Methods**. John Wiley & Sons, NY, 1994
- (NITRD, 2001) **NITRD – Networking and Information Technology Research and Development – Supplement to the President’s Budget for FY 2002**. Interagency Working Group on Information Technology Research and Development - Executive Office of the President of the United States, July 2001. Disponível em: <<<http://www.ccic.gov/pubs/blue02/index.html>>>. Acesso em: 23 de jan. 2004
- (NUSEIBEH, 2000) NUSEIBEH, B. A.; EASTERBROOK, S. M. **Requirements Engineering: A Roadmap**. In: “The Future of Software Engineering” Anthony Finkelstein (Ed.), ACM Press, Limerick, 2000
- (OMAN, 2001) OMAN, P.; SCHWEITZER, E. O.; ROBERTS, J. **Safeguarding IEDS, Substations and SCADA Systems Against Electronic Intrusions**. Technical Report, Schweitzer Engineering Laboratories, Inc., Pullman, WA, 2001
- (OMG, 1999) **OMG - Unified Modeling Language Specification**. Object Management Group, v. 1.3, 1999
- (OMG, 2001) **OMG - Common Object Request Broker Architecture (CORBA)**. Object Management Group, v. 2.6, 2001
- (ONS, 2002) **ONS - Procedimentos de Rede**, Operador Nacional do Sistema, mar. 2002. Disponível em: <<<http://www.ons.org.br/ons/procedimentos/index.htm>>>. Acesso em: 10 de jan. 2004
- (PARRY, 1995) PARRY, G. W. et all. **Process Description for ATHEANA: A Technique for Human Error Analysis**. Brookhaven

- National Laboratory Technical Report, L-2415195-2, Upton, NY, 1995
- (PÁSCOA, 2002) PÁSCOA, J. E. P.; CAMARGO JR, J. B. **Fatores e Subfatores para Avaliação da Segurança em Software de Sistemas Críticos**. Boletim Técnico da Escola Politécnica da USP – BT/PCS/0215, EPUSP, SP, 2002
- (PAULK, 1993) PAULK, C. M.; CURTIS, B.; CHRISSIS, M. B.; WEBER, V. C. **Capability Maturity Model for Software**. Technical Report: CMU/SEI-93-TR-024, Software Engineering Institute, Carnegie Mellon University, PA, 1993
- (PERLMAN, 1999) PERLMAN, G. **Web-Based User Interface Evaluation with Questionnaires**. Web site from acm.org, 1999. Disponível em: <<<http://www.acm.org/~perlman/question.html>>>. Acesso em: 11 de abr. 2004
- (PITAC, 1999) **PITAC - Information Technology Research: Investigating in Our Future**. President's Information Technology Advisory Committee – Report to the President Executive Office of the President of the United States, Feb. 1999. Disponível em: <<<http://www.ccic.gov/pitac/report/index.html>>>. Acesso em: 23 de jan. 2004
- (POPOV, 1999) POPOV P.; STRIGINI, L.; ROMANOVSKY, A. **Choosing Effective Methods for Design Diversity - How to Progress from Intuition to Science**. In: "Computer Safety, Reliability and Security: 18th International Conference, SAFECOMP'99, (272-285), 1999
- (PROMAIN, 2001) **PROMAIN - State of Art in Human Factors**. Progress in Maintenance and Management of Railway Infrastructure – Contribution to Report to Council of Decision Makers, 2001. Disponível em: <<<http://promain.server.de/servlet/is/270>>>. Acesso em: 01 de jun. 2004
- (PUTMAN, 2001) PUTMAN, J.. **Architecting with RM-ODP**. Prentice-Hall, NJ, 2001

- (PYLA, 2003) PYLA, P. S.; PÉREZ-QUIÑONES, M. A.; ARTHUR, J. D.; HARTSON, H. R. **Towards a Model-Based Framework for Integrating Usability and Software Engineering Life Cycles**. IFIP - WG2.7/13.4 - Interact 2003 Workshop, In: “Closing the Gaps: Software Engineering and Human-Computer Interaction”, M. B. Harning & J. Vanderdonckt (Eds.), Université catholique de Louvain - Institut d’Administration et de Gestion, Louvain-la-Neuve, Belgium, (67-74), 2003. Disponível em: <<<http://www.se-hci.org/bridging/interact/proceedings.html>>>. Acesso em: 01 de jun. 2004
- (RIDAO, 2001) RIDAO, M.; DOORN, J.; LEITE, J. C. S. P. **Domain Independent Regularities in Scenarios**. IEEE Transactions on Software Engineering (120-127), 2001
- (ROLLAND, 1999) ROLLAND, C.; PRAKASH, N. **From Conceptual Modeling to Requirement Engineering**. Annals of Software Engineering, Special Volume on Comparative Studies of Engineering Approaches for Software Engineering, 1999
- (RUBIN, 1994) RUBIN, J. **Handbook of Usability Testing – how to plan, design and conduct effective tests**. Wiley Technical Communication Library, NY, 1994
- (RUSHBY, 1999) RUSHBY, J. **Using Model Checking to Help Discover Mode Confusions and Other Automation Surprises**. Proc. 3rd Workshop on Human Error, Safety and System Development, 1999
- (SCHACH, 1999) SCHACH S. R. **Classical and Object-Oriented Software Engineering**. McGraw Hill, NY, 1999
- (SIEWIOREK, 1995) SIEWIOREK, D. P.; SWARZ, R. S. **The Theory and Practice of Reliable System Design**. Digital Press, MA, 1995
- (SLABODKIN, 1998) SLABODKIN, G. **Software Glitches Leave Navy Smart Ship Dead in the Water**, 1998. Disponível em: <<<http://www.gcn.com/archives/gcn/1998/July13/cov2.html>>>. Acesso em: 05 de jan. 2004

- (SOMMERVILLE, 1993) SOMMERVILLE, I.; et al. **Integrating Ethnography into the Requirements Engineering Process**. First International Symposium on Requirements Engineering, IEEE Computer Society Press, Los Alamitos, 1993
- (STANDISH, 1995) STANDISH GROUP. **Software Chaos**, 1995. Disponível em: <<<http://www.standishgroup.com/chaos.html>>>. Acesso em: 09 de jan. 2004
- (STAVRIDOU, 1998) STAVRIDOU, V.; DUTERTRE, B. **From Security to Safety and Back**. Computer Security, Fault Tolerance, and Software Assurance: From Needs to Solutions - Workshop II: Williamsburg, VA, 1998
- (STEEN, 1999) STEEN, M.; DERRICK, J. **Formalizing ODP Enterprise Policies** In 3rd International Enterprise Distributed Object Computing Conference - EDOC'99, University of Mannheim, IEEE Publishing, GB, 1999
- (STOREY, 1997) STOREY, N. **Safety-Critical Computer Systems**. Addison-Wesley, MA, 1997
- (TAYLOR, 2000) TAYLOR, C. **Comparison of ODP Viewpoints in Object-Z: A Case Study**, University of Kent – Computing Laboratory, Technical Report: 7-00, UK, 2000. Disponível em: <<<http://www.cs.ukc.ac.uk/pubs/2000>>>. Acesso em: 9 de jan. 2004
- (TORRES, 2002) TORRES, R. J. **Practitioner's Handbook for User Interface Design and Development**. Prentice-Hall, NJ, 2002
- (UML, 1997) **UML Summary**, 1997. Disponível em: <<<http://www.rational.com>>>. Acesso em: 15 de jan. 2004
- (USABILITY, 2003) **Usability Net**. Web site from European Union, 2003. Disponível em: <<<http://www.usabilitynet.org/home.htm>>>. Acesso em: 07 de jun. 2004
- (USOR, 2002) **USOR – A Collection of User Oriented Methods**. The Royal Institute of Technology – June 2002. Disponível em: <<<http://www.nada.kth.se/cid/usor/>>>. Acesso em: 01 de mar. 2004

- (VAN LAMSWEERDE, 2000) VAN LAMSWEERDE, A. **Requirements Engineering in the Year 00: A Research Perspective.** ICSE'2000 - Proc. 22nd International Conference on Software Engineering, In: "The Future of Software Engineering", Anthony Finkelstein (Ed.), ACM Press, Limerick, 2000
- (VILLER, 1998) VILLER, S.; SOMMERVILLE, I. **Coherence: Social Analysis for Software Engineers** Lancaster University – Computing Department, Cooperative Systems Engineering Group – Technical Report: CSEG/6/98, UK, 1998. Disponível em: <<[http:// www.comp.lancs.ac.uk/computing/research/cseg/98_rep.html](http://www.comp.lancs.ac.uk/computing/research/cseg/98_rep.html)>>. Acesso em: 10 de jan. 2004
- (VORA, 1995) VORA, P.; HELANDER, M. **A Teaching Method as an Alternative to the Concurrent Think-Aloud Method for Usability Testing.** In: "Proceedings of Sixth International Conference on Human-Computer Interaction: Symbiosis of Human and Artifact: Future Computing and Design for Human-Computer Interaction", Y. Anzai, K. Ogawa & H. Mori (Eds.), Elsevier Science & Technology Books, Amsterdam, The Netherlands, (375-380), 1995
- (WANG, 2000) WANG, C. **A Security Architecture for Survivability Mechanisms** Dissertation – Faculty of School of Engineering and Applied Science - University of Virginia, 2000
- (WICKENS, 1992) WICKENS, C. D. **Engineering Psychology and Human Performance.** Second Edition, HarperCollins, NY, 1992
- (ZHANG, 1999) ZHANG, Z. **Usability Evaluation Methods.** Web site from Drexel University, 1999. Disponível em: <<<http://www.pages.drexel.edu/~zwz22/UsabilityHome.html>>>. Acesso em: 11 de abr. 2004