



International  
Requirements  
Engineering  
Board



Klaus Pohl · Chris Rupp

# Fundamentos da Engenharia de Requisitos

Um Guia de Estudo para o  
Exame CPRE-FL Certified Professional for  
Requirements Engineering – Foundation level  
em conformidade com o padrão IREB

# Sumário

## PRÓLOGO DO EDITOR

5

COMO EVOLUÍMOS PARA INTEGRAR A ENGENHARIA DE TESTES COM A ENGENHARIA DE REQUISITOS E PORQUE RESOLVEMOS TRADUZIR ESTE LIVRO PARA A LÍNGUA PORTUGUESA BRASILEIRA.

5

## O EXAME CPRE “CERTIFIED PROFESSIONAL FOR REQUIREMENTS ENGINEERING”

7

## PREFÁCIO DO ORIGINAL

9

## CONTRIBUIÇÕES ADICIONAIS

11

## SUMÁRIO

15

### DOWNLOADS

23

*SYLLABUS CPRE-FL (ementa do conhecimento requerido por um engenheiro de Requisitos para a certificação CPRE-FL)*

23



23

*GLOSSÁRIO de termos da engenharia de requisitos*

23



23

*QUICK-GUIDE CPRE-FL (guia pictográfico do Syllabus desenvolvido pela T&M)*

23



23

<b>DICAS DE ESTUDO E PREPARAÇÃO PARA A CERTIFICAÇÃO CPRE-FL</b>	<b>25</b>
<i>I -Estude cada um das nove unidades de ensino na sua sequencia numérica</i>	25
<i>II -Estude cada Unidade de Ensino alternando entre as três visões/guias disponíveis (Quick-Guide CPRE-FL ↔Syllabus CPRE-FL ↔Este Livro)</i>	25
<i>III – Estude de forma Visual, Auditiva e Cinestésica e aplique na prática</i>	26
<i>IV - Quer saber mais? Consulte a bibliografia indicada e ou realize um curso de preparação conosco26</i>	
<b>1 INTRODUÇÃO E FUNDAMENTOS</b>	<b>27</b>
<b>1.1 INTRODUÇÃO</b>	<b>27</b>
<b>1.1.1 Dados e Fatos de Projetos Comuns</b>	<b>27</b>
<b>1.1.2 O Que É a Engenharia de Requisitos?</b>	<b>28</b>
<b>1.1.3 Incorporar a Engenharia de Requisitos em Modelos de Processo</b>	<b>31</b>
<b>1.2 FUNDAMENTOS DA TEORIA DA COMUNICAÇÃO</b>	<b>31</b>
<b>1.3 CARACTERÍSTICAS DE UM ENGENHEIRO DE REQUISITOS</b>	<b>32</b>
<b>1.4 TIPOS DE REQUISITOS</b>	<b>34</b>
<b>1.5 IMPORTÂNCIA E CATEGORIZAÇÃO DE REQUISITOS DE QUALIDADE</b>	<b>35</b>
<b>1.6 RESUMO</b>	<b>36</b>
<b>2 OS LIMITES DO SISTEMA E DO CONTEXTO</b>	<b>39</b>
<b>2.1 CONTEXTO DO SISTEMA</b>	<b>39</b>
<b>2.2 DEFINIR OS LIMITES DO SISTEMA E DO CONTEXTO</b>	<b>40</b>
<b>2.2.1 Definir o Limite do Sistema</b>	<b>41</b>
<b>2.2.2 Definir o Limite do Contexto</b>	<b>44</b>
<b>2.3 DOCUMENTAR O CONTEXTO DO SISTEMA</b>	<b>45</b>
<b>2.4 RESUMO</b>	<b>45</b>
<b>3 ELICITAR REQUISITOS</b>	<b>47</b>
<b>3.1 FONTES DE REQUISITOS</b>	<b>47</b>
<b>3.1.1 Stakeholders e sua Importância</b>	<b>47</b>
<b>3.1.2 Lidar com os Stakeholders no Projeto</b>	<b>48</b>
<b>3.2 CATEGORIZAÇÃO DE REQUISITOS CONFORME O MODELO DE KANO</b>	<b>50</b>
<b>3.3 TÉCNICAS DE ELICITAÇÃO</b>	<b>52</b>
<b>3.3.1 Tipos de Técnicas de Elicitação</b>	<b>52</b>

25	3.3.2 <i>Técnicas de Pesquisa</i>	53
25	3.3.3 <i>Técnicas de Criatividade</i>	54
25	3.3.4 <i>Técnicas Baseadas em Documentos</i>	56
26	3.3.5 <i>Técnicas de Observação</i>	57
26	3.3.6 <i>Técnicas de Apoio</i>	58
	3.4 RESUMO	59
27	<b>4 DOCUMENTAR REQUISITOS</b>	61
27	4.1. DESIGN DO DOCUMENTO	61
27	4.2 TIPOS DE DOCUMENTAÇÃO	62
28	4.2.1 <i>As Três Perspectivas dos Requisitos</i>	62
31	4.2.2 <i>Documentação de Requisitos Usando Linguagem Natural</i>	63
31	4.2.3 <i>Documentação de Requisitos Usando Modelos Conceituais</i>	63
32	4.2.4 <i>Documentos de Requisitos Híbridos</i>	64
34	4.3 ESTRUTURAS DOS DOCUMENTOS	65
35	4.3.1 <i>Estruturas Padronizadas de Documentos</i>	65
36	4.3.2 <i>Conteúdos Padrão Customizados</i>	66
39	4.4 USO DOS DOCUMENTOS DE REQUISITOS	68
39	4.5 CRITÉRIOS DE QUALIDADE PARA DOCUMENTOS DE REQUISITOS	69
40	4.5.1 <i>Não-ambiguidade e Consistência</i>	69
41	4.5.2 <i>Estrutura Clara</i>	70
44	4.5.3 <i>Modificabilidade e Extensibilidade</i>	70
45	4.5.4 <i>Completude</i>	70
45	4.5.5 <i>Rastreabilidade</i>	71
47	4.6 CRITÉRIOS DE QUALIDADE PARA REQUISITOS	71
47	4.7 GLOSSÁRIO	73
47	4.8 RESUMO	75
48	<b>5 DOCUMENTAR REQUISITOS USANDO LINGUAGEM NATURAL</b>	77
50	5.1 EFEITOS DA LINGUAGEM NATURAL	77
52	5.1.1 <i>Nominalização</i>	78

5.1.2	<i>Substantivos sem Indicador de Referência</i>	79
5.1.3	<i>Quantificadores Universais</i>	80
5.1.4	<i>Condições Especificadas de Forma Incompleta</i>	80
5.1.5	<i>Verbos de Processo Especificados de Forma Incompleta</i>	81
5.2	CONSTRUÇÃO DE REQUISITOS USANDO TEMPLATES	82
5.3	RESUMO	86
<b>6</b>	<b>DOCUMENTAR REQUISITOS USANDO MODELOS</b>	<b>87</b>
6.1	O TERMO MODELO	87
6.1.1	<i>Propriedades de Modelos</i>	88
6.1.2	<i>Linguagens de Modelagem</i>	88
6.1.3	<i>Modelos de Requisitos</i>	89
6.1.4	<i>Vantagens dos Modelos de Requisitos</i>	89
6.1.5	<i>Uso Combinado de Modelos e Linguagem Natural</i>	90
6.2	MODELOS DE METAS	90
6.2.1	<i>Documentação de Metas Usando Árvores E/OU</i>	91
6.2.2	<i>Exemplo de Árvores E/OU</i>	91
6.3	CASOS DE USO	92
6.3.1	<i>Diagramas de Casos de Uso UML</i>	92
6.3.2	<i>Especificações de Casos de Uso</i>	95
6.4	TRÊS PERSPECTIVAS SOBRE REQUISITOS	98
6.5	MODELAGEM DE REQUISITOS NA PERSPECTIVA ESTRUTURAL	100
6.5.1	<i>Diagramas Entidade-Relacionamento</i>	100
6.5.2	<i>Diagramas de Classes UML</i>	102
6.6	MODELAGEM DE REQUISITOS NA PERSPECTIVA FUNCIONAL	106
6.6.1	<i>Diagramas de Fluxo de Dados</i>	106
6.6.2	<i>Modelos da Perspectiva Funcional e Fluxos de Controle</i>	108
6.6.3	<i>Diagramas de Atividades UML</i>	109
6.7	MODELAGEM DE REQUISITOS NA PERSPECTIVA COMPORTAMENTAL	113
6.7.1	<i>Statecharts</i>	113
6.7.2	<i>Diagramas de Estados UML</i>	115

79	6.8 RESUMO	117
80	<b>7 VALIDAR E NEGOCIAR REQUISITOS</b>	<b>119</b>
80	7.1 FUNDAMENTOS DA VALIDAÇÃO DE REQUISITOS	119
81	7.2 FUNDAMENTOS DA NEGOCIAÇÃO DE REQUISITOS	120
82	7.3 ASPECTOS DE QUALIDADE DOS REQUISITOS	120
86	7.3.1 <i>Aspecto de Qualidade "Conteúdo"</i>	121
87	7.3.2 <i>Aspecto de Qualidade "Documentação"</i>	122
87	7.3.3 <i>Aspecto de Qualidade "Acordo"</i>	123
88	7.4 PRINCÍPIOS DE VALIDAÇÃO DE REQUISITOS	124
88	7.4.1 <i>Princípio 1: Envolvimento dos Stakeholders Corretos</i>	124
89	7.4.2 <i>Princípio 2: Separação Entre a Identificação de Falhas e a Correção de Erros</i>	125
89	7.4.3 <i>Princípio 3: Validação a Partir de Diferentes Pontos de Vista</i>	125
90	7.4.4 <i>Princípio 4: Mudança Adequada do Tipo de Documentação</i>	125
90	7.4.5 <i>Princípio 5: Construção de Artefatos de Desenvolvimento</i>	126
91	7.4.6 <i>Princípio 6: Revalidação de Requisitos</i>	126
91	7.5 TÉCNICAS DE VALIDAÇÃO DE REQUISITOS	127
92	7.5.1 <i>Parecer de Especialista (Commenting)</i>	127
92	7.5.2 <i>Inspeção</i>	127
95	7.5.3 <i>Walkthrough</i>	129
98	7.5.4 <i>Leitura Baseada em Perspectiva</i>	129
100	7.5.5 <i>Validação por Protótipos</i>	131
100	7.5.6 <i>Utilização de Checklist para a Validação</i>	133
102	7.6 NEGOCIAR REQUISITOS	134
106	7.6.1 <i>Identificação de Conflitos</i>	134
106	7.6.2 <i>Análise de Conflitos</i>	135
108	7.6.3 <i>Resolução de Conflitos</i>	136
109	7.6.4 <i>Documentação da Resolução de Conflitos</i>	138
113	7.7 RESUMO	138
113	<b>8 GERENCIAR REQUISITOS</b>	<b>141</b>
115		

8.1	DESIGNAR ATRIBUTOS PARA REQUISITOS	141
8.1.1	<i>Atributos para Requisitos em Linguagem Natural e Modelos</i>	141
8.1.2	<i>Esquema de Atributos</i>	141
8.1.3	<i>Tipos de Atributos de Requisitos</i>	143
8.2	VISUALIZAÇÕES DE REQUISITOS	145
8.2.1	<i>Visualizações Seletivas da Base de Requisitos</i>	145
8.2.2	<i>Visualizações Consolidadas dos Requisitos</i>	147
8.3	PRIORIZAÇÃO DE REQUISITOS	148
8.3.1	<i>Método de Priorização de Requisitos</i>	148
8.3.2	<i>Técnicas de Priorização de Requisitos</i>	149
8.4	RASTREABILIDADE DE REQUISITOS	152
8.4.1	<i>Vantagens de Requisitos Rastreáveis</i>	152
8.4.2	<i>Rastreabilidade Definida por Finalidade</i>	153
8.4.3	<i>Classificação de Relacionamentos de Rastreabilidade</i>	153
8.4.4	<i>Representação da Rastreabilidade de Requisitos</i>	155
8.5	VERSIONAMENTO DE REQUISITOS	157
8.5.1	<i>Versões de Requisitos</i>	158
8.5.2	<i>Configurações de Requisitos</i>	159
8.5.3	<i>Baselines de Requisitos</i>	160
8.6	GERENCIAMENTO DE MUDANÇAS DE REQUISITOS	161
8.6.1	<i>Mudanças de Requisitos</i>	161
8.6.2	<i>O Comitê de Controle de Mudanças</i>	163
8.6.3	<i>A Solicitação de Mudança</i>	164
8.6.4	<i>Classificação das Solicitações de Mudanças Recebidas</i>	165
8.6.5	<i>Método Básico para Mudanças Corretivas e Adaptativas</i>	166
8.7	RESUMO	169
9	APOIO POR FERRAMENTA	169
9.1	APOIO POR FERRAMENTA: VISÃO GERAL	170
9.2	FERRAMENTAS DE MODELAGEM	171
9.3	FERRAMENTAS PARA GERENCIAMENTO DE REQUISITOS	

141	9.3.1 <i>Ferramentas Especializadas para Gerenciamento de Requisitos</i>	172
141	9.3.2 <i>Aplicativos Padrão de Escritório</i>	172
141	9.4 INTRODUZINDO FERRAMENTAS	173
143	9.5 AVALIAÇÃO DE FERRAMENTAS	174
145	9.5.1 <i>Perspectiva do Projeto</i>	175
145	9.5.2 <i>Perspectiva do Usuário</i>	176
147	9.5.3 <i>Perspectiva do Produto</i>	176
148	9.5.4 <i>Perspectiva do Processo</i>	176
148	9.5.5 <i>Perspectiva do Fornecedor</i>	176
149	9.5.6 <i>Perspectiva Técnica</i>	177
152	9.5.7 <i>Perspectiva Econômica</i>	177
152	9.6 RESUMO	177
153	<b>BIBLIOGRAFIA</b>	<b>179</b>
153	<b>ÍNDICE REMISSIVO</b>	<b>185</b>
155		
157		
158		
159		
160		
161		
161		
161		
163		
164		
165		
166		
169		
169		
170		
171		

## Downloads

Os seguintes materiais complementares e traduzidos para o português estão disponíveis para download em [www.tmtestes.com.br](http://www.tmtestes.com.br) e ou [www.ibqts.com.br](http://www.ibqts.com.br) :

### **SYLLABUS CPRE-FL (ementa do conhecimento requerido por um engenheiro de Requisitos para a certificação CPRE-FL)**



### **GLOSSÁRIO de termos da engenharia de requisitos**



### **QUICK-GUIDE CPRE-FL (guia pictográfico do Syllabus desenvolvido pela T&M)**



# Dicas de Estudo e Preparação para a Certificação CPRE-FL

## I -Estude cada um das nove unidades de ensino na sua sequencia numérica

- Os nove capítulos do livro correspondem exactamente às nove unidades de ensino (com um total de 53 tópicos de ensino). Cada unidade de ensino é construída sobre as anteriores.
- Permita-se um tempo adequado de assimilação do conhecimento, pelo menos uma semana por unidade de ensino, estudando uma hora por dia e você estará preparado.

1	• O SYLLABUS CPRE-FL é composto de 9 Unidades e (53) Tópicos de Ensino
2	• Introdução e Fundamentos (6)
3	• Limites e Contexto dos Sistemas (4)
4	• Elicitação de Requisitos (4)
5	• Documentação de Requisitos (8)
6	• Documentar Requisitos usando Linguagem Natural (3)
7	• Documentar Requisitos usando Modelos (8)
8	• Validar e Negociar Requisitos (7)
9	• Gerenciar Requisitos (7)
	• Apoio por Ferramentas (6)

## II -Estude cada Unidade de Ensino alternando entre as três visões/guias disponíveis (*Quick-Guide CPRE-FL®↔Syllabus CPRE-FL↔Este Livro*)

- O *Quick-guide CPRE-FL* é um guia pictográfico do Syllabus.
- O *Syllabus CPRE-FL* é a descrição do conhecimento necessário e suficiente exigido de um Engenheiro de Requisitos.
- Este *Livro* é um guia de estudo detalhado de cada unidade de ensino.
- Você decide se prefere estudar do específico ao geral (*Quick-Guide→Livro*) ou se prefere estudar do geral ao específico (*Livro↔Quick-Guide*).



### III – Estude de forma Visual, Auditiva e Cinestésica e aplique na prática

- Apenas ler os materiais sugeridos não é suficiente para dominar assunto tão vasto, complexo e rico quanto a Engenharia de Requisitos.
- Estimule seu aprendizado e fixação do conteúdo estudando usando seus três sentidos: Visual, Auditivo e Cinestésico, como segue:
- Peça a um conhecido escolher aleatoriamente um assunto em alguma página no Quick-Guide CPRE-FL®, no Syllabus CPRE-FL® ou neste livro. Discorra sobre este assunto em voz alta (auditivo). Faça um resumo por escrito deste assunto (cinestésico). Represente o assunto através de uma figura no power-point (visual).
- Estudos em grupo onde cada um desafia a conhecimento do outro – e aproveita para esclarecer e ser esclarecido – são outra forma muito eficaz de aprendizado.
- Apresente o assunto, uma unidade de ensino por vez e uma sub-unidade de ensino por vez a um grupo de colegas. Talvez eles não aprendam muita coisa mas você aprenderá com certeza (descobrir nestas apresentações que ainda não domina o assunto já é um aprendizado e um estímulo a saber mais e melhor).
- Pratique a Engenharia de Requisitos segundo o modelo IREB no seu dia-a-dia. Dúvidas surgirão. Procure possíveis respostas neste livro, no Quick-Guide CPRE-FL® e ou no Syllabus CPRE-FL®.

**IV - Quer saber mais? Consulte a bibliografia indicada e ou realize um curso de preparação conosco**



Recognized  
Training Provider  
2012



Licensed  
Examination Institute

# 1 Introdução e Fundamentos

O impacto da Engenharia de Requisitos (ER) no desenvolvimento de sistemas de sucesso e focados no cliente não pode mais ser ignorado. Tornou-se prática comum destinar recursos à engenharia de requisitos. Além disso, é cada vez maior a compreensão de que o papel do engenheiro de requisitos é claramente diferenciado e abrange uma série de atividades complexas.

## 1.1 Introdução

Segundo dados do Chaos Report publicado pelo Standish Group em 2006, muita coisa melhorou na execução de projetos de software nos doze anos entre 1994 e 2006. Enquanto aproximadamente 30% dos projetos de software investigados em 1994 falharam, esse número foi de apenas 20% em 2006. O número de projetos que não cumpriram os prazos, estouraram o orçamento ou não corresponderam às expectativas do cliente caiu de 53% para 46% [Chaos 2006]. Jim Johnson, presidente do Standish Group, cita três motivos para o desenvolvimento positivo desde 1994. Um deles é que a comunicação de requisitos melhorou muito nos últimos dez anos. Esses dados têm importância, pois a maneira como são manipulados os requisitos de um sistema é um fator que contribui significativamente para a falha de projetos e/ou prazos e orçamentos estourados.

### 1.1.1 Dados e Fatos de Projetos Comuns

Segundo estudos realizados no passado, cerca de 60% de todos os erros em projetos de desenvolvimento de software têm origem durante a fase de engenharia de requisitos [Boehm 1981]. Esses erros, no entanto, frequentemente são descobertos apenas em estágios mais avançados do projeto, ou depois que o sistema foi implementado, pois requisitos incorretos ou incompletos podem ser interpretados por desenvolvedores de tal maneira que pareçam (subjetiva e subconscientemente) corretos ou completos. A falta de requisitos muitas vezes não é detectada durante a fase de projeto e execução, porque os desenvolvedores confiam que o trabalho entregue pelos engenheiros de requisitos seja de alta qualidade. Os desenvolvedores implementam aquilo que o documento de requisitos diz, ou aquilo que acham que está dizendo. Requisitos ambíguos, incompletos ou incorretos inevitavelmente levam ao desenvolvimento de um sistema que não possui propriedades críticas, ou

A engenharia de requisitos como origem de erros

possui propriedades que não foram solicitadas.

Durante o desenvolvimento do projeto, quanto mais tarde um defeito nos requisitos for corrigido, mais altos serão os custos associados com sua correção. Por exemplo, o esforço necessário para corrigir um defeito durante a programação é até 20 vezes maior do que realizar a correção durante a engenharia de requisitos. Se o defeito for corrigido durante os testes de aceitação, o esforço exigido pode ser até 100 vezes maior [Boehm 1981].

O custo dos erros durante a engenharia de requisitos

Os sintomas de uma engenharia de requisitos inadequada são tão numerosos quanto suas causas. Com frequência, requisitos estão faltando ou não estão claramente formulados. Por exemplo, se os requisitos não refletem as necessidades do cliente de forma precisa, ou se estão descritos de maneira imprecisa, permitindo dessa forma diversas interpretações, o resultado muitas vezes é um sistema que não atende as expectativas do cliente ou dos usuários.

Sintomas e causas de uma engenharia de requisitos deficiente

O motivo mais comum para requisitos deficientes é a noção equivocada por parte dos *stakeholders* de que muitas coisas são óbvias e não precisam ser explicitadas. Isso resulta em problemas de comunicação entre as partes envolvidas decorrentes de diferenças em experiência e conhecimento. Para piorar as coisas, é comum que o cliente muitas vezes deseja uma rápida integração de resultados mais recentes em um sistema produtivo.

A importância de uma boa engenharia de requisitos

A crescente importância de sistemas baseados intensivamente em software nos projetos industriais, bem como a necessidade de lançar no mercado sistemas cada vez mais inovadores, individualizados e abrangentes – e isso de maneira mais rápida, melhor e com um nível mais alto de qualidade – exige uma engenharia de requisitos eficiente. Requisitos completos sem defeitos formam a base para o desenvolvimento bem sucedido de sistemas. Os riscos em potencial devem ser identificados durante a engenharia de requisitos e obrigatoriamente ser reduzidos o mais cedo possível para permitir que o projeto progride com sucesso. Falhas e lacunas no documento de requisitos devem ser descobertas num estágio inicial para evitar cansativos processos de mudança.

### 1.1.2 O Que É a Engenharia de Requisitos?

Para o desenvolvimento bem sucedido de um projeto, é necessário conhecer os requisitos para o sistema e documentar os mesmos de maneira adequada.

**Definição 1-1: Requisito**

- (1) Uma condição ou capacidade necessária para um usuário resolver um problema ou alcançar um objetivo.
- (2) Uma condição ou capacidade que deve ser alcançada ou estar presente em um sistema ou componente de sistema para satisfazer um contrato, norma, especificação ou outro documento formalmente imposto.
- (3) Uma representação documentada de uma condição ou capacidade como em (1) e (2).

[IEEE Standard 610.12-1990]

O termo *stakeholder* é essencial na engenharia de requisitos. Os *stakeholders* são, entre outras coisas, as mais importantes fontes de requisitos. Não levar um *stakeholder* em consideração muitas vezes resulta em uma elicitação fragmentada de requisitos, isto é, resulta em requisitos incompletos [Macaulay 1993]. *Stakeholders* são aquelas pessoas ou organizações que têm algum impacto sobre os requisitos. Isso inclui as pessoas que irão interagir com o sistema (Por exemplo, usuários ou administradores), pessoas que simplesmente têm interesse no sistema mas provavelmente não irão utilizá-lo (Por exemplo, a alta gerência; um *hacker* do qual o sistema precisa ser protegido, *stakeholders* de sistemas concorrentes), mas também entidades legais, instituições, etc., pois elas são formadas por pessoas de carne e osso que podem muito bem decidir influenciar ou definir os requisitos do sistema.

Stakeholders

**Definição 1-2: Stakeholder**

Um *stakeholder* de um sistema é uma pessoa ou uma organização que tem uma influência (direta ou indireta) nos requisitos de um sistema.

Durante o processo de desenvolvimento, a engenharia de requisitos deve eliciar os requisitos dos *stakeholders*, documentar os requisitos de forma adequada, validar e verificar os requisitos, e gerenciar os requisitos ao longo de todo o ciclo de vida do sistema [Pohl 1996].

Meta da  
engenharia de  
requisitos

#### Definição 1-3: Engenharia de requisitos

- (1) A engenharia de requisitos é uma abordagem sistemática e disciplinada para a especificação e gerenciamento de requisitos, com os seguintes objetivos:
  - (1.1) Conhecer os requisitos relevantes, estabelecer um consenso entre os *stakeholders* a respeito de tais requisitos, documentar os requisitos de acordo com determinados padrões e gerenciar os requisitos de forma sistemática.
  - (1.2) Compreender e documentar as expectativas e necessidades dos *stakeholders*, especificar e gerenciar os requisitos para minimizar o risco de entregar um sistema que não atenda às suas expectativas e necessidades.

As quatro atividades centrais para atingir esses objetivos são:

- *Elicitação*: Durante a elicitação de requisitos, diversas técnicas são utilizadas para obter requisitos dos *stakeholders* e de outras fontes, e para desenvolver os requisitos em maior detalhe (ver capítulo 3).
- *Documentação*: Durante a documentação, os requisitos elicitados são descritos da forma mais adequada. Diversas técnicas são utilizadas para documentar os requisitos, seja por meio da linguagem natural ou de modelos conceituais (ver capítulos 4, 5 e 6).
- *Validação e Negociação*: Para garantir que os critérios de qualidade previamente definidos sejam atingidos, os requisitos documentados devem ser validados e negociados desde o princípio (ver capítulo 7).
- *Gerenciamento*: O gerenciamento de requisitos é ortogonal a todas as outras atividades, abrangendo toda e qualquer medida necessária para estruturar requisitos, preparar os mesmos para que possam ser utilizados por diferentes papéis, manter sua consistência após eventuais mudanças e

Quatro  
atividades  
centrais da  
engenharia de  
requisitos

assegurar sua implementação (ver capítulo 8).

Diferentes restrições de projeto influenciam a engenharia de requisitos. Por exemplo: pessoas, fatores de domínio, restrições organizacionais (como a distribuição física ou a disponibilidade de tempo dos membros do projeto) têm forte impacto na escolha de técnicas adequadas.

Restrições

### 1.1.3 Incorporar a Engenharia de Requisitos em Modelos de Processo

Modelos pesados de processo, tais como o modelo Waterfall [Royce 1987] ou o modelo V [V-Modell 2004], têm por meta eliciar e documentar completamente todos os requisitos num estágio inicial do projeto, antes de serem tomadas quaisquer decisões a respeito de projeto ou execução. O objetivo de tais modelos é de eliciar todos os requisitos antes do desenvolvimento propriamente dito. Como consequência, nesses modelos de processo a engenharia de requisitos é compreendida como uma fase inicial do desenvolvimento de sistemas, finita e restrita no tempo.

A engenharia de requisitos como uma fase distinta

Modelos mais leves de processo, tais como *eXtreme Programming* [Beck 1999], por outro lado, apenas elicitam os requisitos necessários quando eles estão para ser implementados, visto que é difícil "prever" funcionalidades futuras e os requisitos mudam ao longo do projeto. Nesses modelos de processo a engenharia de requisitos é vista como um processo contínuo e abrangente, que engloba e integra todas as fases do desenvolvimento de sistemas.

A engenharia de requisitos como um processo contínuo e paralelo

## 1.2 Fundamentos da Teoria da Comunicação

Requisitos precisam ser comunicados. Na maioria dos casos, utiliza-se um meio de comunicação regrado e acessível ao parceiro de comunicação – a linguagem natural.

A linguagem como meio de comunicação de requisitos

Para que a transmissão de informações funcione de modo satisfatório, é necessário que exista um código em comum entre um indivíduo e outro. O emissor codifica sua mensagem e o receptor precisa decodificá-la. Tal código em comum é intrínseco a duas pessoas que falam o mesmo idioma (por exemplo, português), têm a mesma origem cultural e experiências semelhantes. Quanto maior a semelhança entre suas origens culturais e educacionais, suas áreas de especialização e suas rotinas diárias de trabalho, melhor será a troca de informações entre eles. No entanto, na maioria das vezes tais condições ideais não existem entre *stakeholders*. É sensato, portanto, chegar a um acordo sobre uma linguagem comum e sobre a

maneira de utilizar essa linguagem. Esse objetivo pode, por exemplo, ser alcançado através de glossários (ver capítulo 4), nos quais todos os termos importantes são explicados. Alternativamente, isso pode ser feito adotando-se uma linguagem descriptiva formal, por exemplo, a Linguagem de Modelagem Unificada OMG, ou UML (ver capítulo 6).

Outro fator importante é o tipo de meio de comunicação. Na comunicação verbal, o sucesso da comunicação depende fortemente da redundância (por exemplo: linguagem e gestos, ou linguagem e entonação) e do *feedback*. Na comunicação técnica escrita, pelo contrário, as informações são transmitidas com um mínimo de redundância e *feedback*.

Além dos problemas decorrentes dos diferentes vocabulários de domínio e diferentes meios de comunicação, pode-se muitas vezes observar que as informações não são adequadamente transmitidas, ou nem mesmo são transmitidas. Isto pode ser atribuído a transformações naturais que ocorrem na percepção humana. Esses efeitos transformacionais são, em particular, a *focalização* e a *simplificação*, e podem impactar mais ou menos significativamente na comunicação.

A comunicação – isto é, a expressão de conhecimento através da linguagem – é necessariamente simplificadora por natureza. O autor pressupõe que o leitor tenha algum tipo de conhecimento implícito anterior sobre o tema. São justamente as simplificações que surgem a partir da expressão de conhecimento pela linguagem que se tornam problemáticas em termos de requisitos, pois os requisitos podem tornar-se suscetíveis a diversas interpretações. No capítulo 5, a documentação de requisitos em linguagem natural é discutida em maior detalhe.

O tipo de meio de comunicação

Acomodação linguística

Conhecimento implícito anterior do tema

### 1.3 Características de um Engenheiro de Requisitos

O papel desempenhado pelo engenheiro de requisitos em um projeto muitas vezes o coloca no centro das atenções. Ele geralmente é a única pessoa que tem contato direto com os *stakeholders* e possui tanto a competência quanto a responsabilidade de familiarizar-se ao máximo com o domínio, buscando compreendê-lo da melhor maneira possível. Ele é a pessoa que identifica as necessidades por trás das afirmações dos *stakeholders*, devendo aprimorá-las para que os arquitetos e desenvolvedores – geralmente leigos no que se refere ao domínio em questão – possam compreender e implementar as mesmas. O engenheiro de requisitos é, de certa forma, um tradutor que comprehende o domínio, bem como sua linguagem específica, suficientemente bem, e também possui suficiente *know-how* de TI para estar a par dos problemas com os quais os desenvolvedores se deparam e para conseguir comunicar-se com eles no mesmo nível. Por esses motivos todos os engenheiros de requisitos tem um papel central no projeto.

Papel central

Para conseguir cumprir todas suas tarefas, o engenheiro de requisitos requer muito mais do que conhecimento de processos. Muitas das capacidades exigidas devem ter como base a experiência prática.

Sete capacidades necessárias para um engenheiro de requisitos

- *Raciocínio analítico:* O engenheiro de requisitos deve ser capaz de familiarizar-se com domínios que lhe sejam desconhecidos e deve compreender e analisar problemas e relacionamentos complexos. Visto que os *stakeholders* muitas vezes discutem requisitos problemáticos a partir de exemplos concretos e soluções não ideais, o engenheiro de requisitos deve ser capaz de distanciar-se das afirmações concretas do *stakeholder*.
- *Empatia:* O engenheiro de requisitos tem a difícil tarefa de identificar as verdadeiras necessidades de um *stakeholder*. Uma exigência fundamental para que possa fazer isso é ter uma boa intuição e demonstrar empatia por pessoas. Além disso, ele deve identificar problemas que podem surgir dentro de um grupo de *stakeholders* e agir conforme necessário.
- *Competência comunicativa:* Para eliciar os requisitos dos *stakeholders*, interpretá-los corretamente e comunicar os requisitos de maneira adequada, um engenheiro de requisitos deve possuir boa capacidade de comunicação, além de saber ouvir, fazer as perguntas certas na hora certa, perceber quando uma afirmação não contém as informações desejadas e fazer mais perguntas se necessário.
- *Resolução de conflitos:* Divergências de opinião entre *stakeholders* frequentemente são causa de conflitos durante a engenharia de requisitos. O engenheiro de requisitos deve saber identificar conflitos, atuar como mediador entre as partes envolvidas e aplicar técnicas adequadas para resolver o conflito.
- *Moderação:* O engenheiro de requisitos deve ser capaz de atuar como mediador entre diferentes opiniões e liderar discussões. Isso vale tanto para conversas individuais quanto para discussões de grupo e *workshops*.
- *Auto-confiança:* Como o engenheiro de requisitos muitas vezes está no centro das atenções, ele ocasionalmente está exposto a críticas também. Como consequência, ele deve ter um alto nível de auto-confiança e a capacidade de se defender em caso de fortes objeções em relação a suas opiniões, jamais levando tais comentários como uma crítica pessoal.
- *Persuasão:* O engenheiro de requisitos desempenha, de certa forma, o papel de advogado dos requisitos dos *stakeholders*. Ele deve ser capaz de representar os requisitos em reuniões de equipe e apresentações, além de consolidar opiniões divergentes, facilitar uma decisão em caso de

discordância e criar consenso entre os *stakeholders*.

## 1.4 Tipos de Requisitos

Em geral podemos distinguir três tipos de requisitos:

- *Requisitos funcionais* definem a funcionalidade oferecida pelo sistema a ser desenvolvido. Esses requisitos são geralmente subdivididos em requisitos funcionais, requisitos comportamentais e requisitos estruturais (ver capítulo 4).

### Definição 1-4: Requisito Funcional

Um requisito funcional é um requisito relacionado ao resultado de algum comportamento a ser fornecido por uma função do sistema.

- *Requisitos de qualidade* definem qualidades desejadas do sistema a ser desenvolvido e muitas vezes influenciam a arquitetura do sistema mais do que os requisitos funcionais. Tipicamente, requisitos de qualidade definem o desempenho, a disponibilidade, a confiabilidade, a escalabilidade ou a portabilidade de um sistema. Requisitos desse tipo são frequentemente classificados como requisitos não funcionais.

### Definição 1-5: Requisito de Qualidade

Um requisito de qualidade é um requisito relacionado a uma questão de qualidade não coberta por um requisito funcional.

- *Restrições* não podem ser influenciadas pelos membros da equipe. Requisitos desse tipo podem restringir o próprio sistema (por exemplo: "O sistema deverá ser implementado utilizando serviços da web") ou o processo de desenvolvimento ("O sistema deverá estar disponível no mercado no mais tardar no segundo trimestre de 2012"). Ao contrário de requisitos funcionais e de qualidade, as restrições não são implementadas, mas são cumpridas, pois meramente limitam o espaço da solução disponível durante o processo de desenvolvimento.

**Definição 1-6: Restrição**

Uma restrição é um requisito que limita o espaço da solução além do que seria necessário para cumprir os respectivos requisitos funcionais e de qualidade.

Além da classificação em requisitos funcionais, requisitos de qualidade e restrições, diversas classificações de requisitos são utilizadas na prática. Por exemplo, existem inúmeras classificações sugeridas por várias normas, CMMI [SEI 2006] ou SPICE [ISO/IEC 15504-5]. Outros esquemas de classificação descrevem atributos de requisitos, tais como o nível de detalhamento de um requisito, sua prioridade ou o grau de obrigação legal de um requisito (ver capítulos 4 e 8).

## 1.5 Importância e Categorização de Requisitos de Qualidade

Na prática diária, os requisitos de qualidade de um sistema frequentemente não são documentados, são documentados inadequadamente, ou negociados de forma imprópria. Tais circunstâncias podem ameaçar o sucesso de um projeto, ou a aceitação posterior do sistema sob desenvolvimento. Por esse motivo o engenheiro de requisitos deve dar destaque especial à elicitação, à documentação e à negociação de requisitos de qualidade durante o processo de desenvolvimento.

Tipicamente, vários tipos diferentes de qualidades desejadas do sistema são classificadas na categoria *requisito de qualidade*. Para que se possa lidar com requisitos de qualidade de forma estruturada, vários esquemas diferentes de classificação dos requisitos de qualidade já foram propostos. A norma ISO 9126 [ISO/IEC 9126], por exemplo, sugere um esquema de classificação para os requisitos de qualidade que também pode ser utilizado como uma estrutura padrão para a documentação de requisitos e como uma lista de verificação para a elicitação e validação de requisitos. A lista abaixo inclui algumas categorias típicas:

- Requisitos que definem a qualidade das funções do sistema, especialmente em relação à sua adequabilidade, segurança de uso (*safety*) e segurança de dados e recursos (*security*), acurácia dos cálculos, a interoperabilidade das funções e sua respectiva conformidade com normas.
- Requisitos que definem a confiabilidade das funcionalidades, especialmente em relação à sua robustez, tolerância a falhas e recuperabilidade.
- Requisitos que definem a usabilidade do sistema, especialmente em

relação à sua compreensibilidade, facilidade de aprendizagem, facilidade de uso.

- Requisitos que definem a eficiência do sistema, especialmente em relação a seu comportamento em termos de tempo (tempo de computação) ou de consumo (utilização de recursos).
- Requisitos que definem a modificabilidade de um sistema, especialmente em relação à sua analisabilidade, modificabilidade, estabilidade e testabilidade.
- Requisitos que definem a portabilidade de um sistema, especialmente em relação aos aspectos capacidade de aderir a padrões ou convenções relacionados à portabilidade, bem como a capacidade para substituir um outro sistema especificado (*replaceability*) adaptabilidade, capacidade para ser instalado (*installability*).

Atualmente os requisitos de qualidade costumam ser especificados utilizando a linguagem natural. No entanto, inúmeras abordagens para documentar requisitos de qualidade por meio de modelos já foram sugeridas ao longo dos últimos anos.

O engenheiro de requisitos é responsável por assegurar que os requisitos de qualidade sejam tão objetivos e verificáveis quanto possível. Isso tipicamente requer que os requisitos sejam quantificados. Por exemplo, um requisito de qualidade relacionado à eficiência do sistema poderia especificar que um sistema deverá processar 95 por cento de todas as requisições dentro de 1,5 segundos, e que o processamento de requisições não deverá exceder 4 segundos em qualquer momento. Isso pode fazer com que os requisitos de qualidade sejam refinados por meio de requisitos funcionais adicionais, como no caso de um requisito de qualidade relacionado à segurança do sistema, em que um requisito funcional especifica o algoritmo exato de criptografia para atender à necessidade de criptografia exigida por determinado requisito de qualidade.

Requisitos de qualidade estão muitas vezes associados a diferentes requisitos funcionais. Como resultado, requisitos de qualidade devem sempre ser mantidos separados dos requisitos funcionais. Em outras palavras, requisitos de qualidade não devem ser misturados aos requisitos funcionais e devem ser documentados separadamente, com uma documentação explícita sobre sua relação com os requisitos funcionais.

## 1.6 Resumo

A engenharia de requisitos é indispensável para desenvolver sistemas que satisfaçam o cliente, cumpram os prazos e se mantenham dentro do orçamento. A meta da engenharia de requisitos é documentar os requisitos do cliente da forma mais completa e qualificada possível, bem como identificar e resolver problemas

nos requisitos o mais cedo possível. Uma engenharia de requisitos bem sucedida deve fundamentalmente incluir os *stakeholders* certos e incorporar as quatro atividades centrais da engenharia de requisitos (*elicitação, documentação, validação/negociação e gerenciamento*) no processo de desenvolvimento do sistema. No centro das atenções encontra-se o engenheiro de requisitos, profissional que atua como o principal ponto de contato na engenharia de requisitos e possui profundo conhecimento tanto do domínio quanto do processo, além de uma grande variedade de competências interpessoais (*soft skills*).

## 2 Os limites do Sistema e do Contexto

Os requisitos de um sistema a ser desenvolvido não "existem" simplesmente, eles precisam ser elicitados. Na engenharia de requisitos, o propósito de definir os limites do sistema e do contexto é identificar a parte do ambiente que influencia os requisitos do sistema a ser desenvolvido.

### 2.1 Contexto do Sistema

No processo de desenvolvimento, a engenharia de requisitos cumpre a função de identificar todos aqueles aspectos materiais e imateriais que têm algum relacionamento com o sistema. Para isso, antecipa-se como o sistema ficará quando se tornar realidade. Ao fazer isso, aqueles aspectos do mundo real que irão possivelmente influenciar os requisitos de um sistema podem ser identificados. Para que se possam especificar os requisitos de um sistema de maneira correta e completa, é necessário identificar os relacionamentos entre aspectos materiais e imateriais individuais da forma mais precisa possível. A parte da realidade que é relevante para os requisitos de um sistema é chamada de contexto do sistema.

Antecipar o sistema em operação

#### Definição 2-1: Contexto do Sistema

O contexto do sistema é a parte do ambiente do sistema que é relevante para a definição e a compreensão dos requisitos de um sistema a ser desenvolvido.

Entre outros, os seguintes aspectos da realidade influenciam o contexto de um sistema:

Aspectos de contexto no contexto do sistema

- Pessoas (*stakeholders* ou grupos de *stakeholders*).
- Sistemas em operação (outros sistemas técnicos ou hardware).
- Processos (técnicos ou físicos, processos de negócio).
- Eventos (técnicos ou físicos).

- Documentos (leis, normas, a documentação do sistema).

Não considerar o contexto do sistema de forma correta ou completa durante a engenharia de requisitos pode resultar em requisitos incorretos ou incompletos. Isto faz com que o sistema opere com base em requisitos incorretos ou incompletos, o que frequentemente está por trás de falhas do sistema durante a operação. Tais erros muitas vezes não são detectados durante os procedimentos de validação, que determinam se o sistema atende aos requisitos especificados, mas somente durante a operação, acarretando consequências às vezes catastróficas.

Consequências da consideração incorreta ou incompleta do contexto

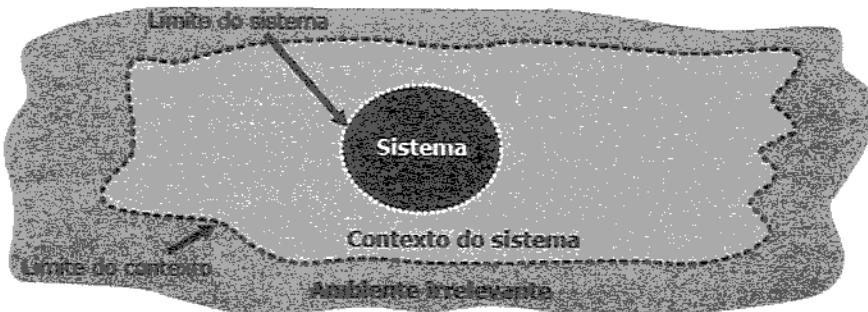
A origem dos requisitos do sistema encontra-se no contexto do sistema a ser desenvolvido. Por exemplo, *stakeholders*, normas pertinentes e diretrizes legais exigem propriedades funcionais específicas que o sistema a ser desenvolvido deve possuir em suas interfaces. Assim, um requisito é definido para um contexto específico e somente pode ser definido corretamente em relação a esse contexto específico. Quanto melhor compreendido for o contexto de um requisito (por exemplo, por que o sistema técnico "X" no contexto do sistema é a origem de determinado requisito), mais baixa será a probabilidade da interpretação incorreta do requisito. Daí a importância de uma documentação do contexto do sistema ou de informações sobre o contexto do sistema que sejam orientadas para sua finalidade de uso.

Contexto do sistema e contexto do requisito

## 2.2 Definir os Limites do Sistema e do Contexto

O engenheiro de requisitos é responsável pela definição adequada do contexto do sistema. Para tanto, é necessário estabelecer uma separação entre o contexto do sistema, o sistema a ser desenvolvido e os aspectos da realidade irrelevantes para o sistema (ver figura 2-1):

- *Definir o limite do sistema:* Ao definir o limite do sistema, uma decisão deve ser tomada: Quais aspectos pertencem ao sistema a ser desenvolvido e quais aspectos fazem parte do contexto do sistema?
- *Definir o limite do contexto:* Ao definir o limite do contexto, a pergunta a ser respondida é: Quais aspectos pertencem ao contexto do sistema (isto é, têm alguma relação com o sistema a ser desenvolvido) e quais aspectos são parte do ambiente irrelevante?



**Figura 2-1** Limites do sistema e do contexto de um sistema

Assim, os limites do sistema e do contexto definem o contexto do sistema. O contexto do sistema abrange todos os aspectos que são relevantes em relação aos requisitos para o sistema a ser desenvolvido. Esses aspectos não podem ser alterados ou modificados pelo processo de desenvolvimento do sistema.

Limites do sistema e do contexto definem o contexto do sistema

### 2.2.1 Definir o Limite do Sistema

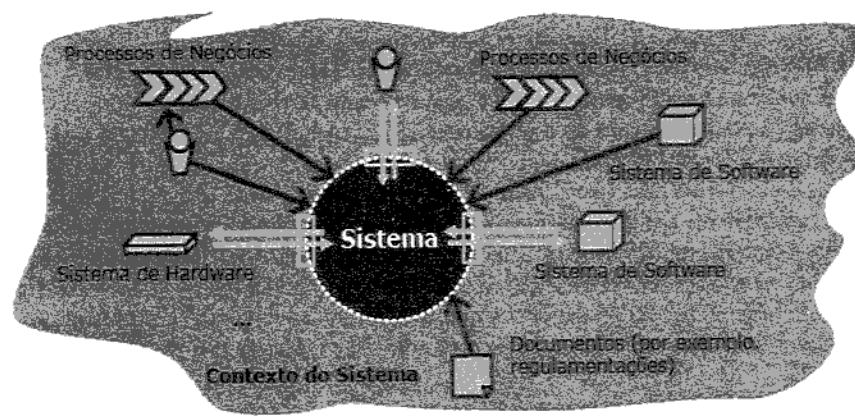
O limite do sistema separa o objeto de interesse (isto é, o sistema) do seu ambiente. Quando o limite do sistema é definido, tanto o escopo do desenvolvimento (isto é, os aspectos que são cobertos pelo sistema a ser desenvolvido) quanto os aspectos que não são parte do sistema são determinados. Por esse motivo definimos o limite do sistema como segue:

#### Definição 2-2: Limite do Sistema

O limite do sistema separa o sistema a ser desenvolvido do seu ambiente (isto é, separa a parte da realidade que pode ser modificada ou alterada pelo processo de desenvolvimento) daqueles aspectos do ambiente que não podem ser mudados ou modificados pelo processo de desenvolvimento.

Assim, todos os aspectos que se encontram dentro do limite do sistema podem ser alterados durante o desenvolvimento do sistema. Por exemplo, um sistema existente consistindo de componentes de hardware e software e que será substituído pelo novo sistema pode estar dentro do limite do sistema. Aspectos dentro do contexto do sistema podem ser processos de negócio, processos técnicos, pessoas e papéis,

estruturas organizacionais e componentes da infraestrutura TI. A figura 2-2 mostra de forma esquemática o contexto do sistema de um sistema. O contexto do sistema é constituído por outros sistemas, por grupos de *stakeholders* que de uma forma ou outra utilizam a interface do sistema a ser desenvolvido, além de fontes adicionais de requisitos e suas interrelações.



**Figura 2-2** Tipos de aspectos dentro do contexto do sistema

Fontes e destinos [DeMarco 1978], entre outros, podem ser utilizados para identificar as interfaces do sistema com seu ambiente. Fontes fornecem dados de entrada (*inputs*) ao sistema. Destinos recebem dados de saída (*outputs*) do sistema. Possíveis fontes e destinos para um sistema são:

- (Grupos de) *stakeholders*.
- Sistemas existentes (tanto técnicos quanto não técnicos).

Fontes e destinos interagem com o sistema a ser desenvolvido por meio das interfaces do sistema. Usando essas interfaces, o sistema fornece sua funcionalidade ao ambiente, monitora o ambiente, influencia parâmetros do ambiente e controla operações do ambiente. Dependendo do tipo da respectiva fonte ou destino, o sistema necessita diferentes tipos de interface (interface homem-máquina, interface de hardware ou interface de software). O tipo de interface por sua vez pode também impor restrições específicas ou fontes adicionais de requisitos para o sistema a ser desenvolvido.

Frequentemente o limite do sistema não é definido de forma precisa até o final do processo de engenharia de requisitos. Antes disso, algumas ou mesmo várias interfaces, bem como funções e qualidades esperadas do sistema a ser

Fontes e destinos como ponto de partida

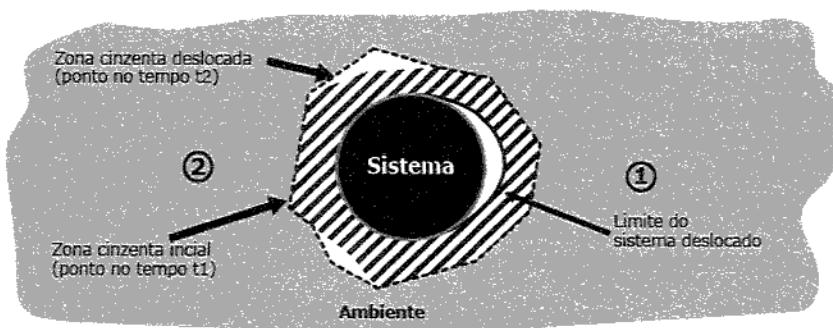
Interfaces:  
Interação entre sistema e ambiente

Zona cinzenta entre sistema e contexto do

desenvolvido são apenas parcialmente conhecidas, ou nem sequer são conhecidas. Nós nos referimos a essa separação inicialmente vaga do sistema e de seu contexto como a zona cinzenta entre o sistema e o contexto (ver figura 2-3). No início do processo de engenharia de requisitos, por exemplo, pode não estar claro se o sistema deveria implementar certa função (como por exemplo, "pagar com cartão de crédito"), ou se existe algum outro sistema no contexto do sistema fornecendo tal função que deveria ser usado (por exemplo, "processamento de pagamentos").

Não somente o limite do sistema pode deslocar-se dentro da zona cinzenta (① na figura 2-3), mas a própria zona cinzenta pode deslocar-se durante o processo de engenharia de requisitos (② na figura 2-3). Este tipo de deslocamento é causado por aspectos que inicialmente pertencem ao contexto do sistema mas são modificados durante o desenvolvimento do sistema. Tal situação ocorre durante a engenharia de requisitos, por exemplo, quando no contexto do sistema não está claro se determinadas atividades de um processo comercial deveriam ou não ser implementadas ou suportadas pelo sistema a ser desenvolvido. Nessa situação, não está claro quais aspectos pertencem ao sistema – podendo, assim, ser mudados ou modificados – e quais aspectos pertencem ao contexto do sistema. Isto causa um correspondente deslocamento da zona cinzenta entre o sistema e o contexto do sistema (ver figura 2-3).

sistema

Ajuste da zona  
cinzenta

**Figura 2-3** Zona cinzenta do limite do sistema

A zona cinzenta se desloca, por exemplo, quando interfaces são atribuídas ao limite do sistema e a zona cinzenta é ampliada para incluir aspectos do ambiente relacionados a essas interfaces.

Fornos e  
destinos como  
ponto de  
partida

Interfaces:  
Interação entre  
sistema e  
ambiente

Zona cinzenta  
entre sistema e  
contexto do

### 2.2.2 Definir o Limite do Contexto

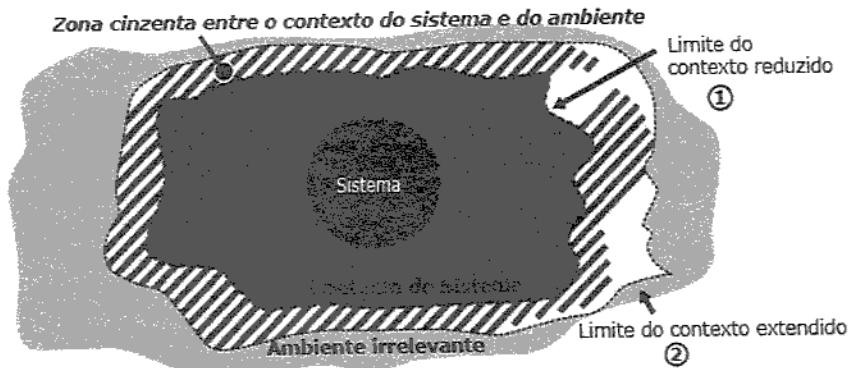
O limite do contexto estabelece a distinção entre os aspectos do contexto, isto é, aqueles aspectos do ambiente que precisam ser considerados durante a engenharia de requisitos (como fontes de requisitos, por exemplo) e aqueles aspectos que são irrelevantes para o sistema. O limite do contexto pode ser definido como segue:

**Definição 2-3:** Limite do Contexto

O limite do contexto separa a parte relevante do ambiente de um sistema a ser desenvolvido da parte irrelevante, isto é, a parte que não influencia o sistema a ser desenvolvido e, sendo assim, não precisa ser considerado durante a engenharia de requisitos.

No início do processo de engenharia de requisitos, com frequência apenas parte do ambiente e alguns relacionamentos específicos entre o ambiente e o sistema a ser desenvolvido são conhecidos. Ao longo da engenharia de requisitos é necessário consolidar o limite entre o contexto do sistema e o ambiente irrelevante, analisando aspectos relevantes dentro do ambiente em termos de seus relacionamentos com o sistema. Além do limite do sistema, o limite do contexto tipicamente também se desloca durante a engenharia de requisitos. Por exemplo, pode ocorrer que determinada norma jurídica, antes considerada relevante para o sistema a ser desenvolvido, passe a não mais impactar o sistema, ou não seja mais considerada relevante. Consequentemente, o contexto do sistema será reduzido (① na figura 2-4). Se, por outro lado, uma nova norma jurídica for identificada que influencia o sistema, o contexto do sistema será devidamente ampliado (② na figura 2-4).

Consolidação  
e  
deslocamento  
do limite do  
contexto



**Figura 2-4** Zona cinzenta entre contexto do sistema e ambiente irrelevante

Tendo em vista que o limite do contexto separa o contexto do sistema daquelas partes do ambiente que são irrelevantes para o sistema, uma definição completa e precisa do limite do contexto para sistemas complexos é virtualmente impossível. Além disso, no caso de alguns aspectos isolados do ambiente, pode ser impossível determinar se eles influenciam o sistema a ser desenvolvido ou se eles são influenciados ou não pelo sistema. Estas duas observações explicam a existência de uma zona cinzenta em relação ao limite do contexto (ver figura 2-4).

Essa zona cinzenta, portanto, engloba certos aspectos do ambiente a respeito dos quais não se pode afirmar com clareza se estão relacionados ao sistema ou não. Ao contrário da zona cinzenta entre o sistema e o limite do sistema, que deve obrigatoriamente ser esclarecida durante a engenharia de requisitos, não é necessário esclarecer inteiramente a zona cinzenta entre o contexto do sistema e o ambiente irrelevante.

Zona cinzenta entre contexto do sistema e ambiente irrelevante

Esclarecer e deslocar a zona cinzenta

## 2.3 Documentar o Contexto do Sistema

Para documentar o contexto do sistema (especialmente os limites do sistema e do contexto), são muitas vezes utilizados diagramas de casos de uso [Jacobson et al. 1992] (ver seções 4.2.3 e 6.3.1) ou diagramas de fluxos de dados [DeMarco 1978] (ver seção 6.6.1). Nos diagramas de fluxo de dados, o contexto é modelado por meio de fontes e destinos no ambiente do sistema representando a origem ou destino de fluxos de dados (ou fluxos de material, energia, dinheiro, etc.). Nos diagramas de casos de uso, são modelados atores (tais como pessoas ou outros sistemas) no ambiente do sistema e suas relações de uso com o sistema. Para modelar o contexto do sistema, podem também ser utilizados diagramas de classes UML [OMG 2007] (ver seção 6.5.2). Para documentar de maneira mais completa possível o contexto de um sistema, várias formas de documentação são tipicamente utilizadas.

## 2.4 Resumo

O contexto do sistema é a parte da realidade que influencia o sistema a ser desenvolvido e dessa forma também influencia os requisitos do sistema. Para que se possam eliciar os requisitos do sistema a ser desenvolvido, é necessário em primeiro lugar definir o limite entre o sistema e o contexto do sistema, bem como o limite entre o contexto do sistema e o ambiente irrelevante. Uma vez definidos os

limites do sistema, o escopo do sistema estará determinado. O escopo engloba aqueles aspectos que podem ser modificados e projetados durante o desenvolvimento do sistema. Ao mesmo tempo, também define-se quais aspectos pertencem ao ambiente e que, dessa forma, não podem ser alterados durante o desenvolvimento, podendo inclusive representar restrições para o sistema a ser desenvolvido.

O limite do contexto separa a parte do ambiente que influencia os requisitos do sistema a ser desenvolvido da parte que não influencia os requisitos. Aspectos típicos dentro do contexto do sistema são os *stakeholders* (por exemplo, usuários do sistema) e documentos (por exemplo, normas a serem consideradas), bem como outros sistemas que, por exemplo, interagem com o sistema a ser desenvolvido. Definir adequadamente os limites do sistema e do contexto constitui o fundamento para uma elucidação sistemática de requisitos para o sistema a ser desenvolvido.

## 3 Elicitar Requisitos

Uma atividade central da engenharia de requisitos é a elicitação de requisitos para o sistema a ser desenvolvido. A base para a elicitação de requisitos é formada pelo conhecimento sobre o contexto do sistema a ser desenvolvido obtido durante a engenharia de requisitos, que inclui as fontes de requisitos a serem analisadas e investigadas.

### 3.1 Fontes de Requisitos

Existem três tipos diferentes de fontes de requisitos:

- *Stakeholders* (ver seção 1.1.2) são pessoas ou organizações que (direta ou indiretamente) influenciam os requisitos de um sistema. Exemplos de *stakeholders* são usuários do sistema, operadores do sistema, desenvolvedores, arquitetos, clientes e testadores. Três tipos de fontes de requisitos
- *Documentos* muitas vezes contêm informações importantes que podem fornecer requisitos. Exemplos são documentos universais, tais como normas e documentos legais, bem como documentos específicos do domínio ou da organização, tais como documentos de requisitos e relatórios de falhas de sistemas legados.
- *Sistemas em operação* podem ser sistemas anteriores ou legados, bem como sistemas concorrentes. Tendo a oportunidade de testar o sistema, os *stakeholders* podem formar uma ideia sobre o sistema atual e solicitar extensões ou modificações com base em suas impressões.

#### 3.1.1 Stakeholders e sua Importância

Identificar os *stakeholders* relevantes é uma tarefa central da engenharia de requisitos [Glinz e Wieringa 2007]. Para o engenheiro de requisitos, os *stakeholders* são importantes fontes de requisitos para o sistema (ver seção 1.1.2). É tarefa do engenheiro de requisitos coletar, documentar e consolidar as metas e requisitos parcialmente conflitantes de diferentes *stakeholders* [Potts et al. 1994] (ver

Importância dos Stakeholders

capítulo 8).

Não identificar ou não considerar os *stakeholders* pode resultar em significativas repercussões negativas para o progresso do projeto, pois certos requisitos podem não ser detectados. Esses requisitos ignorados entrarão em cena mais tarde sob forma de solicitações de mudança durante a operação do sistema. Corrigir essas questões retroativamente implica em altos custos adicionais. Portanto, é essencial identificar todos os *stakeholders* e integrá-los aos procedimentos de elicitação.

Uma técnica auxiliar para a identificação de *stakeholders* é a manutenção de *checklists* atualizadas, o que permite a elicitação sistemática e dirigida de *stakeholders* relevantes. Se a lista de *stakeholders* for atualizada muito tarde ou de forma incompleta, importantes aspectos do sistema poderão não ser detectados, o objetivo do projeto poderá não ser cumprido ou significativos custos adicionais poderão surgir para a correção de eventuais problemas. O ponto de partida para a elicitação dos *stakeholders* são as recomendações de *stakeholders* relevantes feitas por gerentes ou especialistas do domínio, por exemplo. Com base nessas sugestões, *stakeholders* relevantes podem ser identificados.

Consequências  
de não  
considerar  
*Stakeholders*

Identificar  
todos os  
*Stakeholders*

### 3.1.2 Lidar com os *Stakeholders* no Projeto

Pode-se observar com frequência na prática que projetos complexos e "difíceis" envolvem muitos *stakeholders*. Devido aos limitados recursos, no entanto, os *stakeholders* mais indicados para a elicitação de requisitos devem ser selecionados cuidadosamente. Para documentar os *stakeholders* no processo de desenvolvimento, faz sentido usar tabelas e planilhas que contenham (pelo menos) os seguintes dados: nome, função (papel), dados pessoais e de contato, disponibilidade ao longo do projeto (quando e onde), relevância do *stakeholder*, área e nível de expertise, além dos objetivos e interesses do *stakeholder* em relação ao projeto.

Gerenciar  
*Stakeholders*

Lidar com *stakeholders* também significa uma troca contínua de informações: atualizações periódicas de status e envolvimento contínuo dos *stakeholders* auxiliam o engenheiro de requisitos a transformar em colaboradores pessoas que antes eram apenas afetadas pelo projeto, isto é, transformar *stakeholders* principalmente afetados em *stakeholders* plenamente integrados e corresponsáveis.

Transformar  
os afetados em  
colaboradores

*Stakeholders* que não recebem a devida atenção do engenheiro de requisitos podem assumir uma postura excessivamente crítica em relação ao projeto. Além disso, alguns *stakeholders* podem demonstrar falta de motivação por estarem suficientemente satisfeitos com o sistema legado, ou por receio de mudanças, ou por um viés negativo devido a projetos anteriores. É tarefa do engenheiro de requisitos fornecer apoio ao gerente do projeto para convencer todos os *stakeholders* a respeito dos benefícios do projeto. Para evitar mal-entendidos e disputas sobre competências, é recomendável firmar acordos formais sobre as tarefas, responsabilidades e a autoridade gerencial das partes envolvidas, bem como determinar objetivos individuais, canais de comunicação e *feedback* que podem ser

"Acordos  
individuais"  
com os  
*Stakeholders*

Consequências  
de não  
considerar  
*Stakeholders*

Identificar  
os  
*Stakeholders*

Gerenciar  
*Stakeholders*

Transformar  
os afetados em  
colaboradores

"Acordos  
individuais"  
com os  
*Stakeholders*

utilizados pelos *stakeholders*. Dependendo da cultura empresarial, esse acordo e determinação pode se feito verbalmente (isto é, com um "aperto de mãos") ou, mais formalmente, por meio de um documento escrito. Os acordos individuais devem ser assinados pelos gerentes.

Inúmeros direitos e deveres resultam do acordo com os *stakeholders*:

#### Direitos e Deveres do Engenheiro de Requisitos:

- Falar a linguagem dos *stakeholders*.
- Familiarizar-se inteiramente com o domínio da aplicação.
- Criar um documento de requisitos.
- Ser capaz de apresentar resultados de trabalho (por meio de diagramas e gráficos).
- Manter um relacionamento respeitoso com *stakeholders*.
- Apresentar suas ideias e alternativas, bem como seus resultados.
- Permitir que os *stakeholders* demandem propriedades do sistema que venham a simplificar e facilitar sua utilização.
- Assegurar que o sistema atenda às exigências funcionais e de qualidade dos *stakeholders*.

Direitos e  
Deveres dos  
Engenheiros  
de Requisitos

#### Direitos e Deveres dos *Stakeholders*:

- Introduzir o engenheiro de requisitos no domínio da aplicação.
- Suprir o engenheiro de requisitos com requisitos.
- Documentar cuidadosamente os requisitos.
- Tomar decisões em tempo hábil.
- Respeitar as estimativas de custo e viabilidade feitas pelo engenheiro de requisitos.
- Priorizar requisitos.
- Ispecionar os requisitos que o engenheiro de requisitos documenta (como protótipos, etc.).
- Comunicar imediatamente as mudanças nos requisitos.
- Aderir ao processo de mudança previamente determinado.
- Respeitar o processo de engenharia de requisitos implantado.

Direitos e  
Deveres dos  
*Stakeholders*

Além disso, o engenheiro de requisitos deve planejar e organizar os canais de Técnicas de

comunicação, bem como elaborar um cronograma estruturado para as atividades de engenharia de requisitos a serem executadas em conjunto com os *stakeholders*. Essa organização e o tipo de comunicação são influenciados de forma significativa pelas técnicas de elicitação utilizadas durante a engenharia de requisitos.

elicitação determinam a comunicação e o processo de elicitação

### 3.2 Categorização de Requisitos Conforme o Modelo de Kano

Saber a importância de um requisito para a satisfação dos *stakeholders* é de grande utilidade para a elicitação de requisitos. Os fatores que determinam a satisfação do cliente são classificados nas seguintes categorias, sempre de acordo com as respectivas propriedades do produto [Kano et al. 1984].

Influência dos requisitos na satisfação dos stakeholders

- *Fatores básicos de satisfação (dissatisfiers)* são propriedades evidentes e pressupostas (conhecimento subconsciente).
- *Fatores esperados de satisfação (satisfiers)* são propriedades explicitamente exigidas do sistema (conhecimento consciente).
- *Fatores inesperados de satisfação (delighters)* são propriedades do sistema que o *stakeholder* não conhece ou espera, e que ele descobre apenas ao utilizar o sistema – uma surpresa agradável e útil (conhecimento inconsciente).

Com o passar do tempo, os fatores inesperados se transformam em fatores esperados e em fatores básicos, respectivamente, conforme o usuário vai se acostumando às propriedades do sistema. Ao eliciar requisitos, todas as três categorias devem ser consideradas.

elucidação  
determinam a  
comunicação e  
o processo de  
elucidação

Influência dos  
requisitos na  
satisfação dos  
*stakeholders*

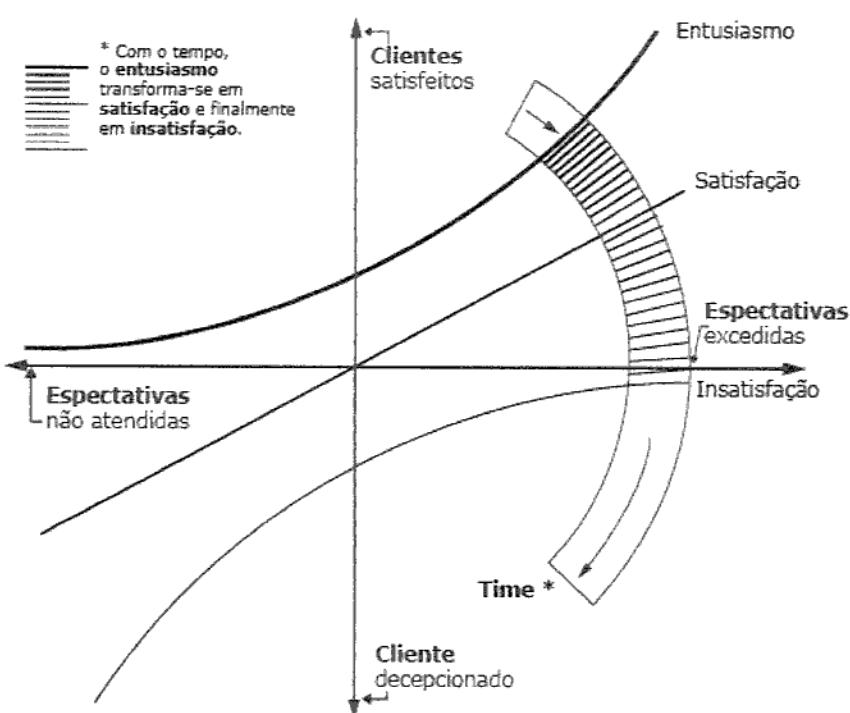


Figura 3-1 Representação gráfica do modelo Kano

Fatores básicos de satisfação, ou requisitos subconscientes, devem ser atendidos pelo sistema de qualquer maneira. Caso contrário, os *stakeholders* ficarão decepcionados e insatisfeitos (daí o termo *dissatisfiers*). Mesmo plenamente atendidos, os fatores básicos de satisfação não geram uma atitude positiva por parte do *stakeholders*, servindo apenas para ajudar a evitar um enorme descontentamento. Fatores básicos de satisfação são predominantemente influenciados pelos sistemas existentes. Sendo assim, técnicas de observação e técnicas centradas em documentos são especialmente adequadas para eliciar esses fatores.

Fatores esperados de satisfação, ou requisitos conscientes, são propriedades conscientemente conhecidas e explicitamente exigidas pelos *stakeholders*. Quando essas propriedades são atendidas, os *stakeholders* ficam contentes e satisfeitos (daí o termo *satisfiers*), o que é desejável. Se algumas propriedades exigidas estiverem faltando, o *stakeholder* provavelmente não aceitará o produto. Sua satisfação decresce proporcionalmente a cada fator esperado de satisfação que estiver faltando. Esses fatores podem ser elicitados muito bem com técnicas de pesquisa.

Fatores básicos de satisfação (*dissatisfiers*)

Fatores esperados de satisfação (*satisfiers*)

Fatores inesperados de satisfação (*delighters*), ou requisitos inconscientes, são propriedades de um sistema cujo valor somente é reconhecido quando o *stakeholder* pode testar o sistema na prática, ou quando o engenheiro de requisitos as propõe. Técnicas de criatividade são as mais indicadas para eliciar esse tipo de fatores.

Fatores inesperados de satisfação (*delighters*)

### 3.3 Técnicas de Elicitação

O principal objetivo de todas as técnicas de elicitação é auxiliar o engenheiro de requisitos a identificar o conhecimento e os requisitos dos *stakeholders*. Como e quando aplicar uma determinada técnica vai depender das condições encontradas. Aplicar a técnica conscientemente e de maneira apropriada para cada situação permite customizar o processo de elicitação de requisitos, levando em consideração as restrições do projeto e permitindo que os requisitos sejam elicitados da forma mais ampla e completa possível.

Não existe método universal de elicitação de requisitos

#### 3.3.1 Tipos de Técnicas de Elicitação

As técnicas de elicitação têm a finalidade de identificar os requisitos conscientes, inconscientes e subconscientes dos *stakeholders*. Entretanto, não existe um método universal para eliciar esses requisitos [Hickey and Davis 2003]. Cada projeto possui restrições específicas e características individuais, sendo de certa forma um caso único, porém sempre há técnicas de elicitação compatíveis com o projeto. Os fatores mais influentes na escolha de técnicas apropriadas de elicitação são os seguintes:

Fatores que influenciam a escolha de técnicas de elicitação

- A distinção entre requisitos conscientes, inconscientes e sub-conscientes a serem elicitados.
- As restrições em termos de tempo e de orçamento, bem como de disponibilidade dos *stakeholders*.
- A experiência do engenheiro de requisitos com determinada técnica de elicitação.
- As oportunidades e riscos do projeto.

O primeiro passo importante ao escolher uma técnica adequada de elicitação consiste em realizar uma análise de restrições críticas para o projeto, isto é, identificar os chamados fatores de risco. Na maioria dos casos, esses fatores são consequência de influências humanas, organizacionais e técnicas, como ilustrado a seguir.

Fatores de risco

Durante a fase de elicitação de requisitos, etapa fortemente influenciada pelos *stakeholders*, uma boa comunicação é essencial. Para assegurar uma comunicação de alta qualidade entre o engenheiro de requisitos e os *stakeholders*, é

Influências humanas

importantes determinar o tipo de requisito, o nível esperado de detalhamento do requisito e a experiência do engenheiro de requisitos e dos entrevistados com diferentes técnicas de elicitação.

Competências sociais e cognitivas dos *stakeholders*, bem como sua experiência com dinâmicas de grupo, também influenciam significativamente a escolha de técnicas apropriadas de elicitação. Outro fator influente é se o conhecimento elicitado é explícito (conscientemente conhecido) para cada indivíduo ou se ele é implícito ou inconsciente (isto é, oculto).

Fatores organizacionais de risco que o projeto pode enfrentar devem ser investigados também. Entre outros aspectos, isso inclui a distinção entre contratos de preço fechado (empreitadas) e contratos de prestação de serviço (alocação), ou se o sistema a ser criado é um novo desenvolvimento ou uma extensão de um sistema legado, bem como a disponibilidade de tempo e deslocamento dos *stakeholders*.

Além disso, é preciso considerar o conteúdo operacional dos requisitos. Se o sistema for muito complexo, é recomendável utilizar uma abordagem estruturante durante a elicitação para desconstruir os conteúdos operacionais em partes compreensíveis.

Outro fator com forte influência na escolha de técnicas de elicitação é o nível esperado de detalhamento dos requisitos. Requisitos abstratos podem ser elicitados bastante bem por meio de técnicas de criatividade. Junto aos *stakeholders* é possível criar ou obter uma visão geral do sistema ou de suas propriedades mais importantes. Técnicas baseadas em pesquisas ou observações podem ajudar a eliciar requisitos com nível médio de detalhamento [Robertson 2002]. Requisitos minuciosamente detalhados podem ser elicitados com técnicas centradas em documentos, isto é, técnicas que utilizam documentos existentes, pois deles podem ser extraídas informações até qualquer grau desejado de detalhamento.

É recomendável combinar diferentes técnicas, pois isso minimiza muitos dos riscos inerentes ao projeto. Pontos fracos e desvantagens de uma técnica podem ser compensados pelo uso de outra técnica que apresenta pontos fortes onde a primeira técnica eventualmente seja deficitária.

Influências organizacionais

Influências do conteúdo operacional

Combinar técnicas de elicitação permite minimizar os riscos inerentes ao projeto

### 3.3.2 Técnicas de Pesquisa

As técnicas de pesquisa têm por objetivo eliciar as mais precisas e imparciais declarações possíveis dos *stakeholders* a respeito de seus requisitos. Todas as técnicas de pesquisa partem do pressuposto que o respondente é capaz de expressar explicitamente seu conhecimento e que ele ou ela está comprometido(a) a investir tempo e esforço para a elicitação. As técnicas de pesquisa são geralmente dirigidas pelo engenheiro de requisitos, pois é ele quem faz as perguntas. Isso, no entanto, pode ter como resultado que preocupações do *stakeholder* venham a ser esquecidas,

Elicitar conhecimento explícito

substituídas ou ignoradas.

- Durante uma *entrevista*, o engenheiro de requisitos faz perguntas previamente determinadas para um ou mais *stakeholders* e documenta as respostas. Questões que surgirem durante a conversa podem ser discutidas imediatamente, e o engenheiro de requisitos pode revelar requisitos subconscientes com perguntas bem pensadas. Entrevistas podem ser utilizadas ao longo de toda a fase de desenvolvimento do sistema. Um entrevistador experiente pode controlar individualmente o desenrolar da conversa, dedicando-se totalmente a cada *stakeholder* e procurando obter informações sobre aspectos específicos, assegurando dessa maneira que as respostas sejam completas. A principal desvantagem dessa técnica de elicitação é o tempo que ela consome. Entrevista
- Outra maneira de eliciar requisitos dos *stakeholders* é fazer uso de questões abertas e/ou fechadas (por exemplo, questões de múltipla escolha). Se um grande número de participantes precisam ser sondados, um questionário *online* é uma opção viável. Questionários podem eliciar grande número de informações em curto espaço de tempo e a baixo custo. Se as respostas estão previamente determinadas, até mesmo *stakeholders* que não têm a capacidade de expressar seu conhecimento de forma explícita podem fornecer uma avaliação. Uma desvantagem de utilizar um questionário é que ele somente pode ser utilizado para coletar os requisitos que o engenheiro de requisitos já conhece ou conjectura. Elaborar um questionário apropriado é muitas vezes uma tarefa complicada que exige muito tempo e requer conhecimento profundo do domínio em questão e das orientações psicológicas para criar questionários. Além disso, ao contrário das entrevistas, os questionários não fornecem uma retroalimentação imediata entre o pesquisador e o pesquisado. Sendo assim, o reconhecimento de que determinada pergunta foi esquecida ou mal formulada somente surge depois que os questionários foram avaliados. Questionário

### 3.3.3 Técnicas de Criatividade

As técnicas de criatividade têm a finalidade de desenvolver requisitos inovadores, esboçar uma visão inicial do sistema e eliciar fatores inesperados de satisfação (*delighters*). As técnicas de criatividade não são geralmente muito adequadas para estabelecer requisitos precisos sobre o comportamento do sistema. As seguintes técnicas de criatividade são frequentemente utilizadas [Maiden e Gizikis 2001]: Estabelecer inovações

- Na técnica de *brainstorming*: Ideias são coletadas durante certo período de tempo, geralmente em grupos de 5 a 10 pessoas. As ideias são documentadas por um moderador sem serem discutidas, julgadas ou comentadas inicialmente. Os participantes usam ideias de outros Brainstorming

participantes para desenvolver novas ideias originais, ou para modificar ideias existentes. Mais tarde, as ideias coletadas são submetidas a uma análise rigorosa. Essa técnica é particularmente eficaz quando envolve um grande número de pessoas de diversos grupos de *stakeholders*. Entre as vantagens dessa técnica estão o fato de que um grande número de ideias podem ser coletadas em um curto espaço de tempo, e diversas pessoas podem expandir essas ideias colaborativamente. A coleta imparcial dessas ideias permite o surgimento inesperado de novas soluções. O *brainstorming* é geralmente menos eficaz quando a dinâmica do grupo é confusa ou quando há membros muito dominantes no grupo. Para tais situações, outras técnicas de criatividade podem ser mais adequadas, tais como o método 6-3-5 (Seis participantes, cada um escreve três ideias numa folha de papel. Esta folha é repassada aos outros membros, que comentam e desenvolvem as ideias recebidas, processo este que se repete cinco vezes), um método também chamado *brainwriting*.

- *Brainstorming paradox* é uma modificação do *brainstorming* tradicional: As ideias coletadas são eventos que não devem acontecer. O grupo mais tarde desenvolve medidas para evitar que os eventos anteriormente coletados venham a acontecer. Por meio desse processo os participantes muitas vezes percebem quais ações podem acarretar resultados negativos. Riscos podem ser identificados antecipadamente com esse método, e medidas preventivas podem ser desenvolvidas. As vantagens e desvantagens dessa técnica são as mesmas do *brainstorming* clássico.
- Mudança de perspectiva: Entre as técnicas que empregam uma mudança de perspectiva (adotar diferentes pontos de vista extremos), a mais comum é a chamada *Six Thinking Hats* [DeBono 2006], conhecida em português como método Seis Chapéus, ou os Seis Chapéus do Pensamento. Cada chapéu representa uma perspectiva específica adotada, uma após a outra, por cada participante. As soluções daí resultantes abordam o problema a partir de perspectivas diferentes. Assim, até mesmo *stakeholders* que têm absoluta convicção a respeito de suas ideias podem ser persuadidos a adotar um ponto de vista diferente. Essa técnica é extraordinariamente benéfica quando os *stakeholders* não conseguem expressar seu conhecimento sem imparcialidade, ou quando estão obstinadamente presos a suas opiniões. Por outro lado, essa técnica não pode ser aplicada se os requisitos exigem um nível de detalhamento muito preciso, pois isso tornaria a técnica muito cansativa.
- *Técnicas de analogia (biônica / bissociações)*: Na técnica biônica, problemas que o projeto enfrenta são comparados (mapeados) a uma situação análoga que ocorre na natureza. As soluções proporcionadas pela natureza são investigadas e então transferidas (mapeadas) de volta ao projeto. Na bissociação, as analogias não precisam necessariamente ter sua origem na natureza. Essas técnicas pressupõem que cada participante é

Brainstorming paradox

Mudança de perspectiva

Técnicas de Analogia

capaz de pensar analogicamente, que existe muito tempo à disposição e que os participantes têm profundo conhecimento do domínio com o qual a analogia será feita. Técnicas de analogia podem ser aplicadas de forma velada ou aberta. Quando a técnica é empregada de forma velada, os participantes somente conhecem a analogia. O engenheiro de requisitos tem a incumbência de transferir (mapear) os resultados para o problema real. Quando a técnica é usada abertamente, os *stakeholders* conhecem tanto o problema real quanto a analogia.

### 3.3.4 Técnicas Baseadas em Documentos

Técnicas baseadas em documentos reutilizam soluções e experiências feitas com sistemas existentes. Quando um sistema legado é substituído, essa técnica assegura que a funcionalidade completa do sistema legado possa ser identificada. Técnicas baseadas em documentos devem ser combinadas com outras técnicas de elicitação para que a validade dos requisitos elicitados possa ser determinada e novos requisitos para o novo sistema possam ser identificados.

- *A arqueologia de sistema* é uma técnica que extrai informações necessárias para construir um novo sistema a partir da documentação ou da implementação (código) de um sistema legado, ou de um sistema concorrente. A técnica é frequentemente aplicada quando o conhecimento explícito sobre a lógica do sistema foi parcialmente ou completamente perdido. Ao analisar o código existente, o engenheiro de requisitos pode assegurar que nenhuma das funcionalidades do sistema legado será esquecida, e a lógica de sistema legado é novamente elicitada. Esse método produz grande quantidade de requisitos muito detalhados e exige muito trabalho. Entretanto, a arqueologia de sistema é a única técnica que pode assegurar que todas as funcionalidades do legado serão implementadas no novo sistema. Nos casos em que está absolutamente claro que o sistema legado e o novo sistema diferem em termos de funcionalidade, técnicas adicionais de elicitação, por exemplo, técnicas de criatividade, deverão ser empregadas inicialmente.  
Arqueologia de Sistema
- *A leitura baseada em perspectiva* (ver seção 7.5.4) é aplicada quando documentos precisam ser lidos a partir de uma perspectiva específica, por exemplo, a perspectiva do implementador ou aquela do testador. Aspectos contidos nos documentos mas que não dizem respeito à perspectiva atual são ignorados. Isso permite uma análise estritamente focada em partes específicas da documentação existente. Dessa forma, aspectos relacionados à tecnologia ou à implementação podem ser separados de aspectos operacionais essenciais que são relevantes para o sistema sucessor.  
Leitura baseada em perspectiva
- *Reutilização*: Requisitos que foram compilados anteriormente e atualizados para um determinado padrão de qualidade podem ser reutilizados. Para isso, os requisitos são armazenados em um banco de  
Reutilização

dados, por exemplo, e disponibilizados no nível exigido de detalhamento para reutilização. Através da reutilização de requisitos, os custos com os procedimentos de elicitção podem ser significativamente reduzidos.

### 3.3.5 Técnicas de Observação

Quando especialistas de domínio não se dispõem a dedicar o tempo necessário para compartilhar sua *expertise* com o engenheiro de requisitos, ou se eles não são capazes de expressar e representar seu conhecimento, as técnicas de observação são muito úteis. Durante a observação, o engenheiro de requisitos observa os *stakeholders* enquanto trabalham. O engenheiro de requisitos documenta todos os passos e dessa forma elictita os processos que o sistema deverá suportar, bem como potenciais erros, riscos e questões em aberto. Todos esses são requisitos em potencial que deverão ser formulados. Os *stakeholders* podem ativamente demonstrar seu conhecimento no uso do aplicativo, ou podem permanecer passivos, com o engenheiro de requisitos apenas observando. O engenheiro de requisitos deve fazer perguntas sobre os processos observados, para que possa determinar a situação como ela deveria ser. Caso contrário, o engenheiro corre o risco de documentar decisões tecnológicas desatualizadas e processos não otimizados (isto é, a situação como ela é, e não como ela deveria ser). Como o engenheiro de requisitos é um observador externo, são boas as suas chances de identificar processos inefficientes e sugerir soluções mais indicadas. Ele está mais distanciado dos processos do que os *stakeholders*, que repetem com frequência os passos operacionais sem questioná-los criticamente. Técnicas de observação são muito apropriadas para elictitar requisitos detalhados e fatores básicos de satisfação (*dissatisfiers*), pois o engenheiro de requisitos pode identificar fatores considerados óbvios, ou aqueles que os *stakeholders* apenas conhecem no subconsciente. Além disso, o engenheiro de requisitos torna-se familiarizado com a linguagem do domínio, o que simplifica o processo de elicitção. Fatores esperados de satisfação (*satisfiers*) somente podem ser observados se estiverem implementados no sistema legado ou se forem utilizados ativamente nos processos atuais. Portanto, essa técnica não é indicada para o desenvolvimento de novos processos. Durante o desenvolvimento do sistema, técnicas de elicitção como observação de campo e *apprenticing* são muito apropriadas.

Questionar  
observações e  
otimizar  
processos

Observação de  
Campo

- *Observação de campo:* Nesta técnica o engenheiro de requisitos, acompanhando o especialista ou os usuários do sistema no próprio local de trabalho, observa e documenta os processos e procedimentos operacionais executados. A partir dessas observações, o engenheiro de requisitos formula os requisitos. Muitas vezes, esse processo pode receber o apoio adicional de gravações de áudio ou vídeo. Essa técnica é apropriada para procedimentos operacionais que são difíceis de expressar verbalmente, mas somente pode ser utilizada se os procedimentos são fisicamente visíveis.

- *Apprenticing*: Nesta técnica o engenheiro de requisitos precisa ativamente aprender ou realizar os procedimentos dos *stakeholders*. Como se fosse um aprendiz em treinamento, o engenheiro de requisitos é estimulado a questionar procedimentos operacionais pouco claros e complexos com o propósito de adquirir experiência do domínio. Dessa forma, ele pode vivenciar requisitos que os *stakeholders* consideram absolutamente óbvios e por isso não conseguem elucidar. Outra vantagem é que o típico equilíbrio de forças entre o engenheiro de requisitos e o respectivo especialista é invertido, pois o *stakeholder* nesse momento adota o papel de "mestre", como detentor de um conhecimento que o aprendiz ainda não possui.

### 3.3.6 Técnicas de Apoio

As técnicas de apoio funcionam como apoio adicional para as técnicas de elicitação e procuram compensar eventuais pontos fracos e desvantagens da técnica de elicitação selecionada.

- Na técnica dos mapas mentais (*mind mapping*), cria-se uma representação gráfica dos relacionamentos e das interdependências refinadas entre termos. Mapas mentais são usados como técnica de apoio para o *brainstorming* ou *brainstorming paradox*. Mapas mentais
- Em reunião conjunta, o engenheiro de requisitos e os *stakeholders* elaboram os objetivos do sistema (ou os detalhes de uma certa funcionalidade). Por exemplo, as interfaces de usuário necessárias do sistema podem ser projetadas em um *workshop* [Gottesdiener 2002]. Workshops
- Na técnica CRC (*Class Responsibility Collaboration*), aspectos do contexto e seus respectivos atributos e propriedades são anotadas em fichas de arquivo. Requisitos são então formulados utilizando essas fichas. Cartões CRC
- Gravações de áudio e vídeo são apropriadas para eliciar requisitos quando os *stakeholders* não estão disponíveis, quando o orçamento é limitado ou quando o sistema é altamente crítico. Especialmente durante observações de campo, elas podem ajudar a capturar processos de alta velocidade. A desvantagem dessa técnica é que os *stakeholders* muitas vezes têm a sensação de estarem sendo supervisionados ao serem filmados e podem, em consequência disso, fornecer declarações imparciais, ou mesmo, em casos extremos, recusar-se a cooperar. Gravações de áudio e vídeo
- A modelagem de casos de uso documenta a visão externa do sistema a ser desenvolvido. Um caso de uso possui um evento desencadeador (*trigger*) que dispara o caso de uso, bem como um resultado esperado (*outcome*). Cada caso de uso é uma funcionalidade que deve ser suportada pelo sistema a ser desenvolvido (ver seção 6.3). Modelar sequências de ações
- Protótipos são apropriados para questionar requisitos estabelecidos e eliciar requisitos em situações onde os *stakeholders* apenas têm uma vagaProtótipos

compreensão do que necessita ser desenvolvido. Consequências potenciais de requisitos novos ou modificados podem mais facilmente ser identificados. Por exemplo, protótipos de interfaces com o usuário são frequentemente utilizados na prática para encontrar requisitos funcionais adicionais.

### 3.4 Resumo

A elicitação de requisitos é uma atividade central da engenharia de requisitos. Além dos documentos e sistemas legados, os *stakeholders* são a principal fonte de requisitos. É importante estabelecer inicialmente um acordo sobre direitos e responsabilidades mútuas entre os *stakeholders* e o engenheiro de requisitos para facilitar a comunicação e a cooperação, bem como para integrar os *stakeholders* no processo de elicitação. A escolha da técnica de elicitação correta para o respectivo projeto é feita pelo engenheiro de requisitos com base nas restrições culturais, organizacionais e de domínio encontradas.

## 4 Documentar Requisitos

Na engenharia de requisitos, informações que tenham sido estabelecidas ou elaboradas durante diferentes atividades devem obrigatoriamente ser documentadas. Entre essas informações encontram-se, por exemplo, protocolos de entrevistas e relatórios de atividades de validação e acordo, bem como solicitações de mudanças. A principal e mais importante tarefa de documentação na engenharia de requisitos, entretanto, consiste em documentar os requisitos para o sistema de forma apropriada.

### 4.1. Design do Documento

Uma técnica de documentação é qualquer tipo de representação mais ou menos formal que facilita a comunicação entre *stakeholders* e aprimora a qualidade dos requisitos documentados. Em princípio, qualquer tipo de técnica de documentação pode ser utilizada para documentar requisitos, seja uma documentação baseada na linguagem natural e redigida em prosa livre, um texto também em línguagem natural, porém mais estruturado, ou técnicas mais formais, como diagramas de estados.

#### Definição 4-1: Documento de Requisitos / Especificação de Requisitos

Uma especificação de requisitos é uma coleção de requisitos representada de forma sistemática, tipicamente para um sistema ou componente, atendendo a determinados critérios.

Durante o ciclo de vida de um documento de requisitos, muitas pessoas são encarregadas da documentação. Durante a comunicação, a documentação desempenha um papel de apoio e de orientação a objetivos. As principais razões para documentar requisitos são as seguintes:

Razões para a documentação

- *Requisitos formam a base para o desenvolvimento do sistema.* Requisitos de qualquer tipo influenciam, direta e indiretamente, a análise, o design, a implementação e as fases de teste. A qualidade de um requisito ou de um

Papel central dos requisitos

documento de requisitos tem forte impacto sobre o desenvolvimento do projeto, e por consequência, sobre seu êxito.

- *Requisitos têm relevância legal.* Requisitos são legalmente vinculantes para o contratante e o contratado, sendo que o contratante pode mover um processo contra o contratado em caso de não cumprimento. Documentar os requisitos pode ajudar a rapidamente resolver conflitos legais entre duas ou mais partes. Relevância legal
- *Documentos de requisitos são complexos.* Sistemas com milhares de requisitos, que por sua vez são caracterizados por complexas interdependências em múltiplos níveis, não são incomuns na prática. Sem uma documentação adequada, manter a situação sob controle pode se tornar bastante difícil para os envolvidos. Complexidade
- *Requisitos devem ser acessíveis para todas as partes envolvidas.* Projetos tendem a passar por determinados "desenvolvimentos" com o decorrer do tempo – tanto em termos de conteúdo quanto de pessoal. Quando os requisitos podem ser acessados de forma permanente, evitam-se incertezas e desconhecimentos, e novos membros da equipe podem rapidamente colocar-se a par do projeto. Acessibilidade

Outro argumento a favor de uma boa documentação de apoio ao projeto é que empregados quase nunca compartilham a mesma compreensão de um determinado assunto. Sendo assim, requisitos devem ser documentados de forma a atender às exigências de qualidade de todos os envolvidos.

## 4.2 Tipos de Documentação

Requisitos para um sistema podem ser documentados a partir de três perspectivas diferentes. Na prática, tanto a linguagem natural quanto a modelagem conceitual são utilizadas para esse fim, ou muitas vezes emprega-se uma combinação apropriada entre as duas.

### 4.2.1 As Três Perspectivas dos Requisitos

Os requisitos para um sistema podem ser documentados a partir de três perspectivas diferentes do sistema a ser desenvolvido:

- *Perspectiva estrutural:* Na perspectiva estrutural, adota-se uma perspectiva estático-estrutural dos requisitos do sistema. Por exemplo, a estrutura dos dados de entrada e saída, os aspectos estático-estruturais de uso, bem como as relações de dependência entre o sistema e o contexto do sistema podem ser documentados (por exemplo, os serviços de um Perspectiva estrutural

sistema externo).

- *Perspectiva funcional:* A perspectiva funcional documenta quais informações (dados) são recebidas do contexto do sistema e processadas pelo sistema ou por uma de suas funções. Esta perspectiva também documenta quais dados retornam para o contexto do sistema. A ordem de execução das funções que processam os dados de entrada também é documentada.
- *Perspectiva comportamental:* Na perspectiva comportamental, informações sobre o sistema e sobre como o sistema está integrado ao contexto do sistema são documentadas a partir dos estados do sistema, documentando-se as reações do sistema frente a eventos no contexto do sistema, as condições que justificam uma transição de estados, bem como os efeitos que o sistema deverá ter sobre seu ambiente (por exemplo, efeitos do sistema sob análise que representam eventos no contexto do sistema de outro sistema).

Perspectiva funcional

Perspectiva comportamental

#### 4.2.2 Documentação de Requisitos Usando Linguagem Natural

A linguagem natural, especialmente a prosa livre, é a forma de documentação de requisitos mais aplicada na prática. Comparado com outras formas de documentação, a prosa possui uma nítida vantagem: nenhum *stakeholder* precisa aprender uma nova notação. Além disso, a linguagem pode ser usada para grande variedade de finalidades – o engenheiro de requisitos pode utilizar a linguagem natural para expressar qualquer requisito.

A documentação baseada na linguagem natural é apropriada para documentar requisitos em qualquer das três perspectivas. Entretanto, a linguagem natural pode resultar em requisitos ambíguos. Além disso, requisitos de diferentes tipos e perspectivas correm o risco de serem misturados, ainda que não intencionalmente, durante a documentação. Nesse caso, é difícil isolar informações relacionadas a uma determinada perspectiva em meio a todos os requisitos em linguagem natural.

Vantagens do uso da linguagem natural

Desvantagens do uso da linguagem natural

#### 4.2.3 Documentação de Requisitos Usando Modelos Conceituais

Ao contrário da linguagem natural, os diferentes tipos de modelos conceituais não podem ser utilizados universalmente. Ao documentar requisitos por meio de modelos, linguagens especiais de modelagem relacionadas com a perspectiva apropriada devem ser utilizadas. Mesmo que a linguagem de modelagem selecionada para uma tarefa de documentação seja aplicada corretamente, seu emprego garante que os modelos criados retratam informações relacionadas apenas com a respectiva perspectiva. O modelo retrata os requisitos documentados de forma muito mais compacta, sendo assim de compreensão mais fácil para um leitor.

Perspectiva legal

Perspectiva de identidade

Perspectiva de habilidade

Perspectiva cultural

treinado do que a linguagem natural. Além disso, modelos conceituais oferecem um menor grau de ambiguidade (isto é, menos possibilidades de interpretação) do que a linguagem natural, devido a seu maior grau de formalidade. Entretanto, usar linguagens de modelagem conceitual para a documentação de requisitos exige conhecimentos específicos de modelagem. A lista abaixo inclui curtas descrições dos mais importantes diagramas discutidos mais detalhadamente no capítulo 6.

- Um *diagrama de caso de uso* permite obter uma rápida visão geral das funcionalidades do sistema especificado. Um caso de uso descreve quais funções são oferecidas ao usuário pelo sistema e como essas funções se relacionam na interação com outras entidades externas. Todavia, casos de uso não descrevem as responsabilidades das funções detalhadamente (ver seção 6.3). Visão geral das funções do sistema
- *Diagramas de classes* são utilizados na engenharia de requisitos, entre outras coisas, para documentar requisitos com respeito à estrutura estática dos dados, para documentar dependências estático-estruturais entre o sistema e o contexto do sistema, ou para documentar termos complexos do domínio de forma estruturada (ver seção 6.5.2). Modelagem estrutural dos dados e estruturamento de termos
- O uso de *diagramas de atividades* torna possível documentar processos de negócio, ou dependências sequenciais do sistema em relação a processos no contexto do sistema. Diagramas de atividades também são apropriados para modelar o caráter sequencial dos casos de uso ou para modelar uma especificação detalhada da interação das funções que processam dados (ver seção 6.6.3). Modelagem de sequências
- *Diagramas de estados* são utilizados na engenharia de requisitos para documentar comportamentos de um sistema desencadeados por determinados eventos (*event-driven behavior*). O foco desse tipo de modelo são os estados individuais em que o sistema pode se encontrar, os eventos e suas respectivas condições que desencadeiam uma transição de estados, bem como os efeitos que o sistema tem no ambiente. Comportamentos desencadeados por determinados eventos

#### 4.2.4 Documentos de Requisitos Híbridos

Acima de tudo, documentos de requisitos contêm requisitos. Além disso, em muitas situações faz sentido documentar decisões, explicações importantes e outras informações relevantes também. Dependendo do público-alvo do documento, da perspectiva sobre o sistema e do conhecimento documentado, selecionam-se tipos apropriados de documentação. Tipicamente os documentos contêm uma combinação de linguagem natural e modelos conceituais. A combinação permite diminuir eventuais desvantagens de um dos dois tipos de documentação através dos pontos fortes do outro tipo. A combinação dos tipos de documentação explora as vantagens de ambos. Por exemplo, modelos podem ser complementados por comentários e requisitos em linguagem natural, ao passo que requisitos em linguagem natural podem ser resumidos e suas dependências representadas

claramente por meio de modelos.

## 4.3 Estruturas dos Documentos

Documentos de requisitos incluem uma grande quantidade de informações diferentes, que devem ser bem estruturadas para o leitor. Para isso, pode-se fazer uso de estruturas padronizadas de documentos ou definir individualmente uma estrutura documental customizada.

Influência dos requisitos na satisfação

### 4.3.1 Estruturas Padronizadas de Documentos

Modelos padronizados de documentos oferecem uma estrutura previamente definida, isto é, estereótipos predefinidos conforme os quais as informações podem ser classificadas. Com o uso de modelos padronizados, uma estrutura aproximada e uma curta descrição do conteúdo das principais seções são previamente determinadas. O uso de modelos padronizados apresenta as seguintes vantagens:

- Modelos padronizados simplificam a inclusão de novos membros na equipe.
- Modelos padronizados permitem a rápida localização de conteúdos desejados.
- Modelos padronizados permitem uma leitura e validação seletiva dos documentos de requisitos.
- Modelos padronizados permitem a verificação automática dos documentos de requisitos (por exemplo, para verificar se o documento está completo).
- Modelos padronizados permitem a reutilização simplificada dos conteúdos dos documentos de requisitos.

Adaptação de estruturas padronizadas existentes

Vale observar que essas estruturas devem ser ajustadas para as propriedades específicas do projeto para atender às suas respectivas restrições. Nos seguintes parágrafos, três das estruturas padronizadas de documentos mais amplamente utilizadas serão apresentadas.

O RUP (*Rational Unified Process*) [Kruchten 2001] é geralmente utilizado para sistemas de software desenvolvidos com o uso de métodos orientados a objetos. O cliente cria um *modelo de negócio* que contém diferentes artefatos do mundo dos negócios (por exemplo, regras de negócio, casos de uso e metas de negócio), os quais servem de base para requisitos do sistema ao longo do ciclo do desenvolvimento. O contratado usa as estruturas da *especificação de*

Rational Unified Process

requisitos de sistema (*software requirements specification*, ou *SRS*) para documentar todos os requisitos de software. Essas estruturas estão estreitamente relacionadas com a norma IEEE 830, como descrito a seguir.

A norma IEEE standard 830-1998 [IEEE Std. 830-1998] (Prática Recomendada para Especificação de Requisitos de Software) contém uma estruturação que foi especialmente projetada para o documento de requisitos de software. Esta estrutura padrão sugere dividir o documento de requisitos em três capítulos principais:

- Introdução (por exemplo, metas do sistema, limites do sistema).
- Descrição Geral (por exemplo, perspectiva do sistema, características dos usuários, restrições de desenvolvimento).
- Requisitos Específicos (por exemplo, requisitos funcionais, de desempenho, requisitos de qualidade, interfaces).

O Modelo-V [V-Modell 2004] do Ministério do Interior da Alemanha (BMI, na sigla em alemão) define diferentes estruturas, dependendo de quem é o autor do documentos de requisitos:

Modelo V

- A *Especificação de Requisitos do Cliente*, conhecido em alemão como *Lastenheft*, ou Caderno de Encargos, é criada pelo contratante (o cliente) e descreve todas as exigências impostas ao contratado em relação ao contrato, isto é, os entregáveis e serviços. Além disso, em muitos casos as exigências dos usuários também são documentadas, incluindo todas as restrições ao sistema e ao processo de desenvolvimento. Portanto, o caderno de encargos geralmente descreve *o que* é feito e *para que* algo é feito.
- A *Especificação de Requisitos do Sistema*, conhecido em alemão como *Pflichtenheft*, ou Caderno de Obrigações, tem como base o caderno de encargos e contém as sugestões de implementação elaboradas pelo contratado. Portanto, o caderno de obrigações descreve como concretizar os requisitos e as restrições do caderno de encargos.

### 4.3.2 Conteúdos Padrão Customizados

Como descrito na seção 4.3.1, estruturas padronizadas de documentos são adaptadas às condições específicas do projeto. As seguintes questões deveriam ser abordadas por qualquer estrutura selecionada.

O conteúdo mínimo de uma especificação de Requisitos

#### Introdução

A introdução contém informações sobre o documento como um todo. Essas informações permitem obter uma visão geral do sistema.

- *Finalidade:* Essa seção discute o propósito para o qual o documento foi criado e identifica o público-alvo do documento de requisitos.
- *Cobertura do sistema:* Essa parte consiste do sistema a ser desenvolvido. Ela indica o nome do sistema e os principais objetivos e vantagens que resultam da implementação do sistema.
- *Stakeholder:* Essa seção contém uma lista de *stakeholders* e suas informações relevantes (ver seção 3.1.1).
- *Definições, acrônimos e abreviações:*<sup>2</sup> Nesta seção, os termos utilizados no documento são definidos para que possam ser usados de forma consistente em todo o documento.
- *Referências:*<sup>3</sup> Todos os documentos referenciados pelo documento de requisitos são listados aqui.
- *Visão Geral:* Ao final do capítulo introdutório, o conteúdo e a estrutura das seções subsequentes do documento de requisitos devem ser brevemente explicados.

### Perspectiva Geral

Nesta seção, documentam-se informações adicionais que aumentam a compreensibilidade dos requisitos. Ao contrário da introdução, essas informações são meramente operacionais e não dizem respeito à administração, ao gerenciamento ou aos aspectos organizacionais do documento de requisitos.

- *Ambiente do sistema:* A integração do sistema dentro do ambiente é uma questão chave nesse parágrafo. Aqui podem ser encontrados os resultados da definição sobre os limites do sistema e do contexto.
- *Descrição da arquitetura:* Nesta seção documentam-se as interfaces operacionais do sistema (por exemplo, interfaces de usuários, interfaces de hardware e software, interfaces de comunicação). Além disso, informações adicionais, relacionadas, por exemplo, com limitações de armazenamento, também são discutidas.
- *Funcionalidade do sistema:* Esta seção descreve globalmente as funcionalidades e tarefas do sistema. Isso pode ser documentado, por exemplo, usando um diagrama de casos de uso.
- *Usuários e público-alvo:* Os diferentes usuários do sistema que compõem o público-alvo são listados nessa seção.

<sup>2</sup> Essa seção também pode ser tratada como um apêndice do documento.

<sup>3</sup> Essa seção também pode ser tratada como um apêndice do documento.

- *Restrições:* Nessa seção devem ser listadas todas as condições que não foram documentadas até este ponto e que podem interferir na engenharia de requisitos.
- *Pressupostos:* Aqui são documentadas decisões tais como não implementar certos aspectos do sistema por razões orçamentárias, ou outros pressupostos globais sobre o contexto do sistema nos quais os requisitos se baseiam.

### Requisitos

Essa parte contém requisitos funcionais e requisitos de qualidade.

### Apêndices

Nos apêndices podem ser documentadas informações adicionais que completam o documento. Por exemplo, os apêndices podem incluir documentos adicionais relacionados com as características dos usuários, normas, convenções ou informações gerais sobre o documento de requisitos.

### Índice

O índice tipicamente contém um sumário (isto é, a estrutura dos capítulos) e um índice remissivo. Em documentos de requisitos altamente dinâmicos, essa pode ser uma seção crítica a ser atualizada constantemente.

## 4.4 Uso dos Documentos de Requisitos

Ao longo do projeto, os documentos de requisitos servem de base para diferentes tarefas:

- *Planejamento:* Com base no documento de requisitos é possível definir pacotes concretos de trabalho e marcos para a implementação do sistema.
- *Projeto arquitetônico:* Os requisitos detalhados documentados (juntamente com as restrições) servem de base para o projeto arquitetônico do sistema.
- *Implementação:* Com base no projeto arquitetônico, o sistema é implementado fazendo uso dos requisitos.
- *Teste:* Baseado nos requisitos que constam no documento de requisitos, é possível desenvolver casos de teste que podem ser utilizados mais tarde na validação do sistema.
- *Gerenciamento de mudanças:* Quando os requisitos mudam, o documento

Documentos de requisitos como base para o desenvolvimento do sistema

de requisitos pode ser utilizado como base para analisar até que ponto outras partes do sistema serão atingidas, o que possibilita estimar o esforço exigido pela mudança.

- *Uso do sistema e manutenção do sistema:* Depois que o sistema foi desenvolvido, o documento de requisitos é usado para fins de manutenção e apoio. Dessa forma o documento de requisitos pode ser utilizado para analisar defeitos e deficiências concretas que surgirem durante o uso do sistema. Por exemplo, é possível deduzir se um defeito resulta do uso incorreto do sistema, de um erro nos requisitos ou de um erro na implementação.
- *Gerenciamento do contrato:* O documento de requisitos em muitos casos é o principal objeto de um contrato entre um contratante e um contratado.

## 4.5 Critérios de Qualidade para Documentos de Requisitos

Para tornar-se uma base para os processos subsequentes de desenvolvimento, o documento de requisitos deve atender certos critérios de qualidade. Além dos critérios de qualidade sugeridos na norma IEEE 830-1998, o documento de requisitos deve possuir uma estrutura clara e ser razoavelmente compreensível. Assim, o documento de requisitos deve atender aos seguintes critérios:

- Não-ambiguidade e consistência [IEEE Std. 830-1998].
- Estrutura clara.
- Modificabilidade e extensibilidade [IEEE Std. 830-1998].
- Completude [IEEE Std. 830-1998].
- Rastreabilidade [IEEE Std. 830-1998].

### 4.5.1 Não-ambiguidade e Consistência

Documentos de requisitos somente podem ser consistentes e não ambíguos quando os requisitos individuais forem consistentes e não ambíguos. Além disso, é preciso assegurar que os requisitos individuais não se contradigam mutuamente. Para tal, recomenda-se o uso de modelos conceituais (ver capítulo 6). Outro aspecto da não ambiguidade envolve a identificação única de um documento de requisitos, ou de um requisito, entre o conjunto de todos os documentos de requisitos, ou de todos os requisitos, em determinado projeto de desenvolvimento (ver seção 8.5).

A qualidade dos requisitos individuais é pré-requisito

#### 4.5.2 Estrutura Clara

Para assegurar que o documento de requisitos possa ser lido por qualquer *stakeholder*, o documento deve ser adequadamente abrangente e claramente estruturado. Infelizmente, não é possível oferecer sugestões simples e claras a respeito da abrangência adequada de um documento de requisitos. Um documento de requisitos muito abrangente e bem estruturado pode ser tão apropriado quanto um documento menos abrangente, pois uma estrutura clara irá permitir que um leitor pule as partes que não são relevantes para ele. Um documento não estruturado ou mal estruturado, por outro lado, não seria apropriado com o mesmo alto nível de abrangência, pois o documento precisaria ser lido integralmente antes que o *stakeholder* pudesse identificar as partes relevantes para ele. Um bom ponto inicial são as estruturas descritas na seção 4.3.1.

Permite leitura seletiva

#### 4.5.3 Modificabilidade e Extensibilidade

Documentos de requisitos devem ser facilmente extensíveis. Sempre há requisitos a serem modificados, alterados, adicionados ou removidos conforme um projeto progride. Sendo assim, a estrutura dos documentos de requisitos deve ser facilmente modificável e extensível. Os documentos de requisitos de um projeto devem ser passíveis de gerenciamento pelo controle de versões do projeto.

Conteúdo e estrutura devem promover a modificabilidade

#### 4.5.4 Completude

Documentos de requisitos devem ser completos, isto é, eles devem conter todos os requisitos relevantes (além das informações adicionais exigidas) e cada requisito deve ser documentado de forma completa.<sup>4</sup> Para cada função do sistema, devem ser descritas todos os dados de entrada (*inputs*), os fatores que o influenciam e as reações exigidas do sistema. Isso inclui a descrição de erros e casos de exceção em particular. Adicionalmente, requisitos de qualidade, tais como requisitos que dizem respeito aos tempos de resposta ou à disponibilidade e usabilidade do sistema, devem ser anotados.

Dois tipos de completude em documentos de requisitos

Fatores formais também contribuem para a completude. Gráficos, diagramas e tabelas devem ser rotulados adequadamente. Outro aspecto importante é que os diretórios de referência e índice devem ser consistentes. Adicionalmente, definições e referências normativas que denotam termos específicos devem ser incluídas em qualquer documento de requisitos. A abrangência de um documento de requisitos constitui um desafio durante a engenharia de requisitos. Muitas vezes é preciso encontrar um meio-termo entre os recursos de tempo disponíveis e a

Fatores formais, Evidências, Referências e Fontes contribuem para a completude

<sup>4</sup> Estritamente falando, essa afirmação é verdadeira apenas para o documento de requisitos da próxima *release* do sistema (ver seção 8.5.3).

completude dos documentos de requisitos.

#### 4.5.5 Rastreabilidade

Um importante critério de qualidade é a rastreabilidade de relacionamentos entre documentos de requisitos e outros documentos (por exemplo, o modelo de um processo comercial, planos de teste ou planos de projeto). Esses documentos podem ter sido criados em fases anteriores ou em fases subsequentes de desenvolvimento, ou ao mesmo tempo que os documentos de requisitos. Entre outras coisas, a rastreabilidade proporciona apoio para o gerenciamento de mudanças (ver seção 8.4).

Relacionamento com outros documentos de desenvolvimento

### 4.6 Critérios de Qualidade para Requisitos

Os critérios de qualidade definidos na norma IEEE 830-1998 podem ser aplicados tanto para requisitos individuais como para documentos completos de requisitos. Além dos critérios formais, critérios adicionais de qualidade para requisitos individuais podem ser estipulados e têm o propósito de aumentar a aceitação da especificação por parte do leitor. Os critérios formais conforme a norma IEEE 830-1998 e os critérios que aumentam a aceitação dos requisitos por parte do leitor estão brevemente explicados na lista abaixo:

Critérios de qualidade para aumentar a aceitação do leitor

- *Acordado:* Um requisito está acordado se ele está correto na opinião dos *stakeholders*, e todos os *stakeholders* o aceitam como válido.
- *Priorizado:* [IEEE Std. 830-1998] Quando a complexidade ou abrangência de um sistema alcança um certo limiar, é importante classificar os requisitos em termos de importância, obrigatoriedade legal ou prioridade. Esse é o caso especialmente quando nem todas as funcionalidades de um sistema podem ser implementadas imediatamente em determinada versão do sistema. Nesse caso, os *stakeholders* devem classificar os requisitos.
- *Não ambíguo:* [IEEE Std. 830-1998] Um requisito que está documentado de forma não ambígua pode ser interpretado somente de uma maneira. Não deve ser possível interpretar o requisito de maneira diferente. Todos os leitores do requisito devem chegar ao mesmo entendimento do requisito.
- *Válido e atualizado:* Um requisito documentado deve representar os fatos e condições do contexto do sistema de forma a ser válido no que se refere às características atuais do contexto do sistema. Essas características podem ser as ideias dos diferentes *stakeholders*, normas relevantes ou

Permite leitura  
seletiva

Conteúdo e  
estrutura devem  
promover a  
modificabilidade

Dois tipos de  
completude em  
documentos de  
requisitos

Fatores formais,  
Evidências,  
Referências e  
Fontes  
contribuem para  
a completude

do sistema (ver

interfaces para sistemas externos.

- *Correto:* [IEEE Std. 830-1998] Um requisito está correto se ele representa de forma adequada a ideia do *stakeholder*. Isso também significa que o requisito pode não expressar mais do que aquilo que o *stakeholder* estava tentando dizer. Para verificar isso, o *stakeholder* deve poder ler e compreender a documentação dos requisitos. Portanto, o grau de exatidão de uma especificação somente pode ser verificado se o critério de compreensibilidade for atendido.
- *Consistente:* [IEEE Std. 830-1998] Requisitos devem ser consistentes em relação a todos os outros requisitos, isto é, os requisitos não devem contradizer-se mutuamente, independente do seu grau de detalhamento ou do tipo de documentação. Além disso, um requisito deve ser formulado de modo a permitir consistência consigo mesmo, isto é, o requisito não pode contradizer-se.
- *Verificável:* [IEEE Std. 830-1998] Um requisito deve ser descrito de forma a permitir sua verificação. Isso significa que testes ou mensurações que comprovem a funcionalidade exigida pelo requisito possam ser realizados.
- *Realizável:* A implementação de cada requisito deve ser possível, observadas as restrições organizacionais, legais, técnicas ou financeiras. Isso significa que um membro da equipe de desenvolvimento deveria envolver-se na avaliação dos objetivos e requisitos para poder indicar os limites técnicos da implementação de determinado requisito. Além disso, os custos da implementação devem ser incorporados à avaliação. Há situações em que *stakeholders* retiram um requisito quando os custos de sua implementação se tornam conhecidos.
- *Rastreável:* [IEEE Std. 830-1998] Um requisito é rastreável se a sua origem e implementação, bem como sua relação com outros documentos, podem ser retracadas, isto é, se o requisito permite o rastreamento das mesmas. Isto pode ser feito por meio de identificadores únicos de requisitos. Usando esses identificadores únicos, requisitos derivados de outros requisitos em algum nível diferente de especificação podem ser conectados. Por exemplo, um objetivo de sistema pode ser rastreado por todos os níveis de abstração, do projeto até a implementação e teste. Detalhes podem ser encontrados na seção 8.4.
- *Completo:* [IEEE Std. 830-1998] Cada requisito individualmente deve descrever completamente a funcionalidade que ele especifica. Requisitos que ainda estão incompletos devem ser especificamente marcados, por exemplo inserindo "ASD" (a ser definido) no respectivo campo de texto, ou determinando um status correspondente. Essas marcações podem então ser localizadas sistematicamente, adicionando-se devidamente as informações que faltam.
- *Compreensível:* Requisitos devem ser compreensíveis para cada

*stakeholder*. Consequentemente, o tipo de documentação de requisitos (ver seção 4.2) pode variar significativamente, dependendo do estágio de desenvolvimento (e portanto, dependendo do pessoal envolvido). Na engenharia de requisitos, é importante definir estritamente os termos utilizados.

Além dos critérios de qualidade para requisitos, existem ainda duas regras fundamentais que aprimoram a legibilidade de requisitos:

- *Frases curtas e parágrafos curtos*: Considerando a limitada memória de curto prazo humana, circunstâncias relacionadas entre si deveriam ser descritas em não mais do que sete frases.
- *Formular apenas um requisito por frase*: Utilize a voz ativa para formular requisitos, empregando apenas um verbo de processo. Frases longas e complexas, entremeadas de orações subordinadas, devem ser evitadas.

Princípios fundamentais de compreensibilidade

## 4.7 Glossário

Uma causa frequente de conflitos na engenharia de requisitos é que as pessoas envolvidas no processo de desenvolvimento têm diferentes interpretações de termos. Para evitar esses conflitos, é necessário que todos os envolvidos no processo de desenvolvimento compartilhem o mesmo entendimento da terminologia utilizada. Portanto, todos os termos relevantes devem ser definidos em um glossário comum. Um glossário é uma coleção de definições para termos, apresentando os seguintes elementos:

- Termos técnicos específicos do contexto.
- Abreviações e acrônimos.
- Conceitos do dia-a-dia que possuem um sentido específico no dado contexto.
- Sinônimos, isto é, termos diferentes com o mesmo sentido.
- Homônimos, isto é, termos idênticos com sentidos diferentes.

Ao definir o significado de termos, é possível aumentar consideravelmente a compreensibilidade dos requisitos. Mal-entendidos e interpretações diferentes de termos, que podem levar a conflitos, podem ser evitados desde o início.

Com frequência em diferentes projetos são utilizados termos similares, ou até mesmo idênticos. Isso pode ocorrer, por exemplo, quando um sistema é desenvolvido para diferentes clientes mas dentro do mesmo domínio. Nesse caso,

Definições consistentes

Reutilização de verbetes de glossários

verbetes de glossários já existentes deveriam ser reutilizados. Pode até ser viável definir tais termos em um glossário universal para vários projetos. O esforço adicional para criar tal glossário será compensado em futuros projetos. Para certos domínios, já existem compilações publicamente acessíveis de definições para termos. Elas podem servir como fundamento para a definição de glossários específicos. Por exemplo, na norma IEEE Std. 610.12-1990, encontram-se definições de termos típicos da engenharia de software.

### Regras para Usar um Glossário

Visto que a criação de um glossário é absolutamente obrigatória, os seguintes aspectos devem ser considerados:

- *O glossário deve ser gerenciado de forma centralizada:* Em qualquer dado momento somente pode haver um glossário válido, que também deve ser acessível de forma centralizada. Não pode haver múltiplos glossários válidos.
- *A responsabilidade deve ser atribuída:* Uma pessoa específica deve receber a atribuição de manter o glossário e assegurar sua consistência e atualização. Os recursos necessários para realizar essa tarefa devem ser incluídos no plano do projeto.
- *O glossário deve ser mantido ao longo do projeto:* Para assegurar a consistência e atualização do glossário, ele deve ser mantido ao longo do ciclo de vida do projeto pela pessoa incumbida dessa responsabilidade.
- *O glossário deve ser de acesso comum:* As definições dos termos devem estar acessíveis para todo o pessoal envolvido no projeto. Essa é única maneira de poder assegurar uma compreensão comum dos termos.
- *A utilização do glossário deve ser obrigatória:* Todo o pessoal envolvido no projeto deve ser obrigado a utilizar exclusivamente os termos e as definições de termos da forma como foram definidos no glossário.
- *O glossário deve incluir as fontes dos termos:* Para que se possam resolver questões e problemas em qualquer momento ao longo do ciclo de vida do projeto, a origem de um dado termo deve poder ser determinada.
- *O glossário deve ser aprovado pelos stakeholders:* Somente os stakeholders podem validar de forma confiável as definições operacionais para seu respectivo contexto de projeto. Cada definição deve ser validada pelo stakeholders ou seus representantes. Além disso, as definições individuais de termos no glossário devem ser explicitamente aprovadas. Essa aprovação sinaliza que o respectivo termo está correto e que seu uso é obrigatório.
- *Os verbetes no glossário devem possuir uma estrutura consistente:* Todos os verbetes no glossário devem ser estruturados da mesma forma. Para servir de apoio para uma documentação consistente, recomenda-se utilizar

Regras básicas  
para usar um  
glossário

um *template* para definições de glossário. Além da definição e do significado de um termo, o *template* deve especificar possíveis sinônimos e homônimos.

Para reduzir o trabalho de ajustar termos constantemente, recomenda-se iniciar a compilação do glossário já no começo do projeto.

## 4.8 Resumo

A documentação de requisitos desempenha um papel central na engenharia de requisitos. Considerando que a quantidade de requisitos é geralmente muito vasta, é muito importante estruturar os requisitos de forma clara, para que o pessoal não envolvido no projeto também compreenda os requisitos. Além disso, dessa maneira a localização e a modificação de requisitos é simplificada e acelerada. Isso facilita muito o atendimento dos critérios de qualidade dos documentos de requisitos. A utilização de estruturas customizadas de documentação demonstrou ser adequada para esse propósito. Essas estruturas são preenchidas com requisitos específicos do projeto, escritos em linguagem natural, em conjunto com modelos conceituais de requisitos.

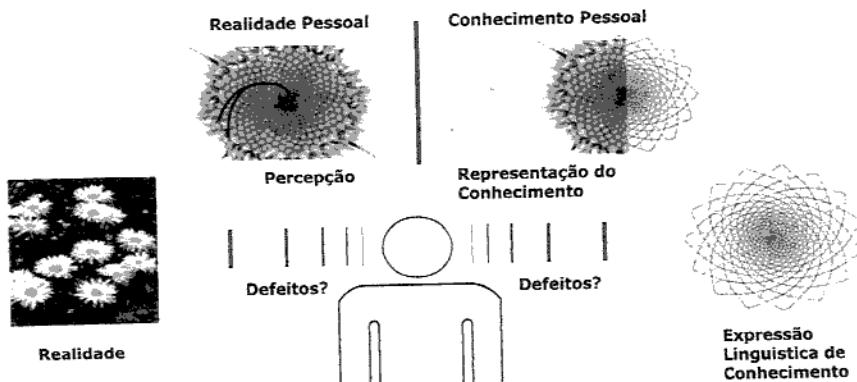
## 5 Documentar Requisitos usando Linguagem Natural

Requisitos elicitados para o sistema a ser desenvolvido são frequentemente documentados usando linguagem natural. A linguagem natural tem a vantagem de (supostamente) não exigir tempo de preparo para ser lida e compreendida pelos *stakeholders* [Robertson e Robertson 2006]. Além disso, a linguagem natural é universal no sentido de que pode ser utilizada para descrever quaisquer circunstâncias. Entretanto, existem algumas dificuldades associadas ao uso da linguagem natural para a documentação de requisitos.

### 5.1 Efeitos da Linguagem Natural

Como a linguagem natural é inherentemente ambígua e sentenças em linguagem natural muitas vezes podem ser interpretadas de múltiplas maneiras, é necessário colocar ênfase especial nas eventuais ambiguidades de tais sentenças para satisfazer o critério de não-ambiguidade. Requisitos são definidos e lidos por pessoas com diferentes níveis de conhecimento, diferentes origens sociais e diferentes experiências. A diversidade entre as pessoas envolvidas nos processos de desenvolvimento pode levar a mal-entendidos, visto que as pessoas interpretam informações de modo diferente (formando uma chamada "estrutura profunda" em suas mentes) e assim também compreendem as informações de modo diferente (por exemplo, como um requisito). Durante tal processo (isto é, a percepção e a representação de informações), ocorrem os chamados "efeitos transformacionais", que se manifestam de forma diferente em cada pessoa, mas podem ocorrer em todos os seres humanos [Bandler e Grinder 1975, Bandler 1994].

Percepção  
subjetiva



**Figura 5-1** Efeitos transformacionais na percepção e representação do conhecimento

3º Simpósio Brasileiro de Documentação de Sistemas

O fato que os efeitos transformacionais seguem certas regras pode ser explorado pelo engenheiro de requisitos para eliciar a estrutura profunda (isto é, o que o autor do requisitos realmente quis dizer) a partir de sua estrutura superficial (isto é, os requisitos). A seguinte lista inclui os cinco efeitos transformacionais mais relevantes para a engenharia de requisitos.

Efeitos transformacionais

- Nominalização.
- Substantivos sem indicador de referência.
- Quantificadores universais.
- Condições especificadas de forma incompleta.
- Verbos de processo especificados de forma incompleta.

Redução de processos

### 5.1.1 Nominalização

Através da nominalização, um processo (às vezes de longa duração) é convertido em um evento (singular). Dessa forma, todas as informações necessárias para descrever o processo acuradamente são perdidas. A palavra de processo *transmitir* transforma-se no substantivo *transmissão*. Outros típicos exemplos de nominalização são os termos entrada (*input*), registro (*booking*) e aceitação (*acceptance*).

**Exemplo 5-1:** Nominalização

"Em caso de *crash* do sistema, será realizado um *restart* do sistema"

Os termos *crash* do sistema e *restart* descrevem, respectivamente, processos que deveriam ser analisados com maior precisão.

Em princípio não há argumentos contra o uso de termos nominalizados para descrever processos complexos. No entanto, o processo deve ser explicitamente definido pelo termo utilizado. A definição de um termo nominalizado não deve deixar qualquer margem de interpretação dos processos e deve representar o processo de forma precisa, incluindo quaisquer exceções que possam ocorrer, bem como todos os parâmetros de entrada e saída. Portanto, as nominalizações não precisam necessariamente ser evitadas, mas elas deveriam ser utilizadas somente se o processo subjacente estiver completamente definido. Durante a análise da linguagem de um texto, todas as nominalizações deveriam ser examinadas para determinar se elas foram definidas de forma suficientemente detalhada em outra parte do documento de requisitos e se elas estão claras para todos os *stakeholders*. Se esse não for o caso, deverá ser criado outro requisito ou um verbete no glossário.

Definir processos completamente

### 5.1.2 Substantivos sem Indicador de Referência

Assim como os verbos de processo, os substantivos são frequentemente especificados de forma incompleta. Linguistas referem-se a isso como a falta de um indicador de referência, ou como um indicador inadequado de referência. Exemplos de substantivos especificados de forma incompleta são termos como: *o usuário, o controlador, o sistema, a mensagem, os dados ou a função*.

Substantivos com referência faltante

**Exemplo 5-2:** Substantivos sem indicadores de referência

Os dados deverão ser exibidos para o usuário no terminal.

O exemplo suscita as seguintes perguntas: Quais dados, exatamente? Qual usuário, exatamente? Qual terminal, exatamente? Se acrescentarmos essas informações, o requisito ficará com a seguinte redação:

**Exemplo 5-3:** Substantivos com indicadores de referência adicionados

O sistema deverá exibir os dados de faturamento para o usuário cadastrado no terminal em que ele estiver logado.

Especificar quantidades e frequências

### 5.1.3 Quantificadores Universais

Quantificadores universais especificam quantidades e frequências. Eles agrupam um conjunto de objetos e fazem uma declaração sobre o comportamento desse grupo. Ao utilizar quantificadores universais existe o risco do comportamento ou da propriedade especificada não se aplicar a todos os objetos do conjunto especificado. *Stakeholders* tendem a agrupar objetos mesmo que esses objetos sejam casos especiais ou exceções, onde o comportamento especificado não se aplica a todos os objetos de um determinado grupo.

Deve ser verificado se o comportamento especificado realmente se aplica a todos os objetos aos quais os quantificadores se referem. Quantificadores universais podem facilmente ser identificados por meio de "palavras-gatilho" (*trigger words*) como *nenhum*, *todos*, *cada*, *alguns* ou *nada*.

Identificar quantificadores universais

**Exemplo 5-4:** Quantificadores universais

O sistema deverá mostrar todos os conjuntos de dados em cada submenu.

Nesse caso, é preciso fazer as seguintes perguntas: Realmente em cada submenu? Realmente todos os conjuntos de dados?

Identificar e esclarecer as estruturas das condições

### 5.1.4 Condições Especificadas de Forma Incompleta

Condições especificadas de forma incompleta são outro indicador de perda potencial de informações. Requisitos contendo condições especificam o comportamento que deve ocorrer quando a condição for atendida. Além disso, os requisitos devem especificar qual comportamento deve ocorrer se a condição não for atendida (a parte que costuma estar faltando). Especialmente em estruturas condicionais complexas, tabelas de decisão podem ser ferramentas valiosas para encontrar variantes não especificadas de condições ou ações. Palavras gatilho são, por exemplo: *se ... então*; *caso...; se... ou*; *dependendo de ...*

**Exemplo 5-5:** Condições especificadas de forma incompleta

O sistema do restaurante deverá oferecer todas as bebidas para um cliente registrado com mais de 20 anos.

Pelo menos um aspecto continua não especificado no exemplo acima: Que bebidas o sistema vai oferecer para um cliente que tenha 20 anos ou menos? Se essa questão for esclarecida, o requisito poderá ser reformulado como segue:

**Exemplo 5-6:** Condições especificadas de forma mais completa

O sistema do restaurante deverá oferecer:

Todas as bebidas não alcoólicas para qualquer usuário registrado com menos de 21 anos.

Todas as bebidas, inclusive as alcoólicas, para qualquer usuário acima de 20 anos.

### 5.1.5 Verbos de Processo Especificados de Forma Incompleta

Alguns verbos de processo requerem mais de um substantivo para serem especificados de forma completa. Por exemplo, o verbo transmitir exige pelo menos três suplementos para ser considerado completo: *o quê* é transmitido, *de onde* é transmitido e *para onde* é transmitido. A intuição do que é lingüisticamente apropriado (em inglês *Feel for language* e em alemão *Sprachgefühl*) é uma ferramenta valiosa para auxiliar na avaliação de que verbos de processo devem ser suplementadas para serem considerados como completos. De forma similar, advérbios e adjetivos também poderão necessitar de suplementação. Apesar de nestes (os advérbios) este efeito linguístico ser muito menos frequente do que naqueles (os verbos), são de mais difícil reconhecimento.

O uso de palavras de processo especificadas de forma incompleta pode ser evitado ou minimizado na maioria das vezes se os requisitos passarem a ser formulados na voz ativa e não na voz passiva.

Completar palavras de processo

Evitar a voz passiva

**Exemplo 5-7:** Requisito formulado na voz passiva

Para logar um usuário, os dados de login são inseridos.

Nesse requisito formulado na voz passiva, não está claro quem vai inserir os dados de *login*. Também não está claro onde e como isso será feito. Se esse requisito for

Usar voz ativa

reformulado na voz ativa, no mínimo o agente ou a pessoa responsável deverá ser incluído.

O mesmo requisito na voz ativa poderia ser formulado como segue:

**Exemplo 5-8: Requisito formulado na voz ativa**

O sistema deve permitir ao usuário inserir seu *username* e senha usando o teclado do terminal.

## 5.2 Construção de Requisitos usando Templates

*Templates* de requisitos fornecem uma abordagem simples e facilmente compreensível para reduzir os efeitos transformacionais de linguagem ao documentar requisitos. *Templates* fornecem o apoio para que o autor possa obter alta qualidade e não-ambiguidade sintática, com baixo custo e tempo otimizado.

Qualidade por meio de templates de requisitos e glossários

**Definição 5-1: Template de Requisitos**

Um *template* de requisitos é um padrão para a estrutura sintática de requisitos individuais.

Para assegurar clareza sintática na documentação também recomenda-se utilizar *templates* de requisitos em conjunto com glossários de projeto (ver seção 4.7).

A seguir apresentamos uma descrição passo-a-passo da aplicação correta dos *templates* de requisitos.

### Passo 1: Determinar a Obrigatoriedade Legal

Inicialmente você deve determinar o grau de obrigatoriedade legal para um requisito. Geralmente faz-se a distinção entre requisitos legalmente obrigatórios, requisitos recomendados com caráter de urgência e requisitos futuros. Para fazer essa distinção em um requisito, você pode utilizar, respectivamente, os verbos modais **deverá** (*shall*), **deveria** (*should*) e **irá** (*will*). Alternativamente, a obrigatoriedade legal de um requisito pode ser documentada por meio de um atributo específico de requisito.

Qual é a obrigatoriedade legal de um requisito?

## Passo 2: Determinar o Núcleo do Requisito

O núcleo de cada requisito é a funcionalidade que ele especifica (por exemplo, imprimir, salvar, colar ou calcular). Essa funcionalidade é denominada de *processo*. Processos são atividades e somente podem ser descritos por meio de verbos. O processo que representa o comportamento do sistema através de um requisito deve ser descrito no passo 2.

Determinar o processo exigido

Como as palavras de processo determinam a semântica, elas devem ser definidas tão claramente quanto possível e ser utilizadas de forma mais consistente possível (ver seção 4.7).

## Passo 3: Caracterizar a Atividade do Sistema

Para requisitos funcionais, a atividade do sistema pode ser classificada como um dos seguintes tipos relevantes:

- *Atividade autônoma de sistema*: O sistema executa o processo de forma autônoma.
- *Interação com usuário*: O sistema fornece o processo como um serviço para o usuário.
- *Requisito de interface*: O sistema executa um processo na dependência de uma terceira parte (por exemplo, outro sistema). O sistema é passivo e espera um evento externo.

No passo 3, qualquer tipo de atividade do sistema especificado por um requisito do sistema é documentado usando exatamente um de três *templates* de requisitos. Esses *templates* de requisitos são descritos mais detalhadamente nas seções seguintes.

Depois de executados os passos 1 a 3, a estrutura do requisito terá sido desenvolvida (ver figura 5-2). As palavras escritas entre os sinais de "maior que" e "menor que" devem ser devidamente substituídas.

idade por  
de  
es de  
sítios e  
ários

é a  
gatoriedade  
de um  
sítio?

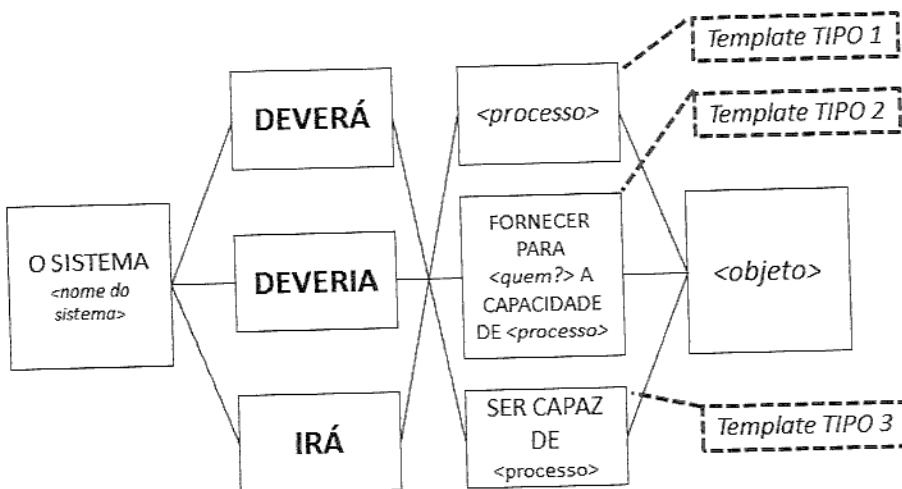


Figura 5-2 O núcleo de um requisito e sua obrigatoriedade legal

O primeiro tipo de *template* é utilizado quando requisitos são construídos para representar atividades que são executadas de forma autônoma. O usuário não interage com a atividade. Definimos o seguinte *template* de requisitos:

Template Tipo 1: Atividade autônoma de sistema

**O SISTEMA DEVERÁ/DEVERIA/IRÁ <verbo de processo>**

**<Verbo de processo>** representa um verbo de processo conforme descrito no passo 2 acima, por exemplo, *imprimir* no caso de uma funcionalidade de impressão, ou *calcular* para algum cálculo executado pelo sistema.

Se o sistema fornece uma funcionalidade para um usuário (por exemplo através de uma interface de entrada), ou se o sistema interage diretamente com o usuário, os requisitos são construídos usando o *template* tipo 2:

Template Tipo 2: Interação com o usuário

**O SISTEMA DEVERÁ/DEVERIA/IRÁ fornecer <a quem?> a capacidade de <verbo de processo>**

O usuário que interage com o sistema está integrado ao requisito por meio de **<quem?>**.

Se o sistema executa uma atividade e depende de sistemas adjacentes, deverá ser utilizado o tipo 3 de *template*. Sempre que mensagens ou dados são recebidos de um sistema adjacente, o sistema deve reagir executando um

Template Tipo 3: Requisito de interface

comportamento específico. O seguinte *template* tem se mostrado apropriado:

O SISTEMA DEVERÁ/DEVERIA/IRÁ ser capaz de  
 <verbo de processo>

#### Passo 4: Inserir Objetos e Complementos

Alguns verbos de processo necessitam um ou mais objetos adicionais para serem considerados completos (ver seção 5.1.5). No passo 4, os objetos que possivelmente estiverem faltando, bem como os complementos (adverbiais) dos objetos, são identificados e adicionados ao requisito. Por exemplo, o *template* de requisitos para o verbo de processo *imprimir* é complementado com as informações sobre *o que* está sendo impresso e *onde* está sendo impresso. A complementação pode ser vista na figura 5-3.

Completar verbos de processos

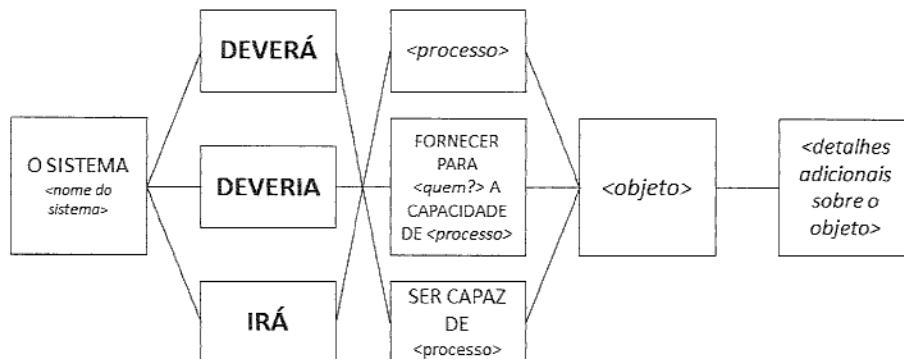


Figura 5-3 O *template* de requisito completo, sem especificação de condições

#### Passo 5: Determinar Condições Lógicas e Temporais

Tipicamente os requisitos não documentam funcionalidades contínuas, mas sim funcionalidades que são executadas ou fornecidas somente sob certas condições lógicas ou temporais. Para diferenciar facilmente entre condições lógicas e temporais, escolhemos a conjunção temporal "assim que" para condições temporais e a conjunção condicional "se" para condições lógicas. A conjunção "quando" não deixa claro se uma condição temporal ou lógica está sendo descrita e deveria portanto ser evitada. No passo 5, requisitos de qualidade que descrevem as condições nas quais um requisito é atendido são adicionados ao início do requisito como uma oração subordinada.

Adicionar condições



**Figura 5-4** O *template* de requisitos completo, com especificação de condições

*Templates* de requisitos devem ser usados quando os membros do projeto têm interesse em um processo formal de desenvolvimento. Estilo e criatividade são fortemente limitados quando *templates* de requisitos são utilizados. A experiência mostra que a melhor prática não é impor o uso dos *templates* de requisitos de forma compulsória, mas oferecer treinamento para utilizar o método e tratá-lo como uma ferramenta complementar.

### 5.3 Resumo

Requisitos de sistema são documentados frequentemente em linguagem natural. Algumas típicas vantagens derivadas da linguagem natural incluem uma melhor legibilidade dos requisitos, a aplicabilidade universal da linguagem natural para qualquer circunstância, além da não-exigência de qualquer experiência prévia em termos de notação. Por outro lado, existem várias desvantagens que resultam do fato que os requisitos em linguagem natural não têm um estrutura formal, por exemplo, a ambiguidade. Como os membros do projeto interpretam requisitos de modo diferente devido a diferenças em seus respectivos conhecimentos, suas origens sociais e suas experiências, o uso de linguagem natural na prática muitas vezes causa mal-entendidos. As desvantagens podem ser minimizadas durante a documentação de requisitos – por exemplo, através do uso de *templates* de requisitos e da verificação de efeitos transformacionais de linguagem nos requisitos.

## 6 Documentar Requisitos Usando Modelos

Durante a documentação de requisitos usando modelo na engenharia de requisitos, três tipos de requisitos são documentados de forma independente e utilizados de forma conjunta:

- *Metas* descrevem as intenções dos *stakeholders* ou de grupos de *stakeholders*. Metas podem potencialmente estar em conflito entre si.
- *Casos de uso* e *cenários* documentam sequências de utilização do sistema. Cenários são agrupados em casos de uso.
- *Requisitos de sistema* (geralmente denominados de requisitos) descrevem detalhadamente funções e qualidades que o sistema a ser desenvolvido deverá implementar ou apresentar. Além disso, requisitos de sistema fornecem informações completas e precisas que servem como subsídios para os passos subsequentes do desenvolvimento.

Na prática, os requisitos são frequentemente documentados usando a linguagem natural. Entretanto, podemos observar que requisitos estão sendo documentados cada vez mais através de modelos. Modelos de requisitos são utilizados adicionalmente à documentação de requisitos em linguagem natural e substituem parcialmente requisitos que seriam documentados em linguagem natural.

### 6.1 O Termo Modelo

Um modelo é uma imagem que abstrai da realidade ou que funciona como uma representação abstrata da realidade a ser criada. A modelagem pode ser aplicada a objetos materiais ou imateriais de uma realidade existente ou de uma realidade a ser desenvolvida. Segundo [Stachowiak 1973], definimos o termo *modelo* como segue:

Modelos como imagens abstratas da realidade

**Definição 6-1: *Modelo***

Um modelo é uma representação abstrata de uma realidade existente ou de uma realidade a ser criada.

### 6.1.1 Propriedades de Modelos

Cada modelo possui três importantes propriedades, que também são as vantagens prevalentes dos modelos:

- *Representação da realidade*: Cada modelo representa certos aspectos da realidade observada sobre seus elementos de modelagem. A criação de modelos pode ser descritiva ou prescritiva por natureza. Na construção de modelos descritivos, o modelo resultante documenta a realidade existente. Na construção de modelos prescritivos, o modelo resultante serve de protótipo para uma realidade fictícia. Dependendo da perspectiva, os próprios modelos podem ser descritivos ou prescritivos ao mesmo tempo. Por exemplo, um modelo é descritivo em relação à concepção do *stakeholder* que está construindo o modelo e prescritivo em relação ao sistema a ser desenvolvido.
- *Redução da realidade*: Os modelos reduzem a realidade mapeada. É comum diferenciar entre seleção e compressão. Durante a seleção, apenas aspectos específicos que fazem parte do discurso do sistema são modelados. Na compressão, ao contrário, resumem-se aspectos do conteúdo do sistema.
- *Pragmatismo*: Um modelo é sempre construído para uma finalidade específica e dentro de um contexto especial. O propósito do modelo pode afetar a construção e a redução proposta da realidade dentro dos modelos. Idealmente, um modelo contém apenas as informações necessárias para sua respectiva finalidade.

Tipicamente, utilizam-se modelos gráficos na engenharia de requisitos. Seus elementos de modelagem são conceitualizações de objetos materiais ou imateriais, ou de pessoas, na realidade.

### 6.1.2 Linguagens de Modelagem

Linguagens específicas de modelagem são utilizadas para construir modelos conceituais. Uma linguagem de modelagem é definida por sua sintaxe e semântica:

Sintaxe e  
semântica

- Sintaxe: A sintaxe de uma linguagem de modelagem define os elementos de modelagem a serem utilizados e especifica as combinações válidas entre eles.
- Semântica: A semântica define o significado dos elementos de modelagem individuais, servindo assim de fundamento para a

interpretação dos modelos da respectiva linguagem de modelagem.

Linguagens de modelagem conceitual podem ser classificadas como formais, informais e semiformais, dependendo de seu grau de formalização. O grau de formalização de uma linguagem de modelagem depende da rigidez das definições formais (por exemplo, cálculo matemático) que definem a sintaxe e a semântica.

Diferentes graus de formalização

### 6.1.3 Modelos de Requisitos

Modelos conceituais que documentam os requisitos de um sistema são chamados modelos de requisitos. A UML (*Unified Modeling Language*) é frequentemente utilizada para construir modelos de requisitos [OMG 2007]. A UML praticamente tornou-se o padrão para a construção de sistemas de software baseada em modelos. A linguagem UML consiste de um conjunto de linguagens de modelagem parcialmente complementares, utilizadas especificamente na engenharia de requisitos para modelar os requisitos de um sistema a partir de diferentes perspectivas. Extensos exemplos de modelagem utilizando a UML podem ser encontrados em [Rupp et al. 2007], por exemplo.

Uma diferença considerável entre o uso convencional de modelos conceituais no desenvolvimento do sistema e o uso de modelos para a documentação de requisitos é que os modelos convencionais documentam soluções escolhidas durante o desenvolvimento do sistema, ao passo que os modelos de requisitos retratam aspectos específicos do problema de fundo.

Modelos de requisitos vs. modelos de projeto

### 6.1.4 Vantagens dos Modelos de Requisitos

A pesquisa sobre a cognição humana demonstrou que informações podem ser percebidas e memorizadas de forma mais rápida e melhor quando retratadas graficamente em vez de usar a linguagem natural [Glass e Holyoak 1986], [Kosslyn 1988] e [Mietzel 1998]. Essas descobertas podem ser aplicadas particularmente para o uso de modelos de requisitos.

Maior facilidade de compreensão

Uma vantagem adicional do uso de modelos de requisitos é que, ao contrário da linguagem natural, as linguagens de modelagem utilizadas têm um enfoque estritamente definido. Um exemplo seriam os diferentes tipos de mapas que podem ser desenhados para representar uma cidade. Dependendo de qual aspecto da cidade está sendo mapeado (modelado), diferentes tipos de abstração podem ser utilizadas para construir o mapa. Por exemplo, um mapa do metrô vai mostrar as estações e as linhas de metrô. No entanto, a distância entre cada ponto do mapa pode não retratar precisamente a distância entre as estações, mas representar, em vez disso, uma estimativa do tempo de deslocamento entre as estações. Um mapa das ruas da cidade, por outro lado, retrata fielmente as ruas,

Foco nas perspectivas da documentação

avenidas e a localização dos pontos turísticos. Os dois modelos retratam a mesma realidade, mas cada um apresenta um enfoque diferente a partir da abstração definida por sua finalidade.

Os modelos de requisitos também apresentam a vantagem de que os diferentes tipos de elementos de modelagem dentro da mesma linguagem de modelagem determinam tanto o método de abstração quanto o que está sendo abstraído e o que não está.

### 6.1.5 Uso Combinado de Modelos e Linguagem Natural

O uso combinado de linguagem natural e modelos de requisitos permite explorar as vantagens das duas técnicas de documentação, enquanto minimiza suas desvantagens. Por exemplo, requisitos formulados em linguagem natural podem ser resumidos e ter suas interrelações representadas através do uso de modelos. Por outro lado, a linguagem natural pode enriquecer os modelos de requisitos e os elementos de modelagem com informações adicionais.

## 6.2 Modelos de Metas

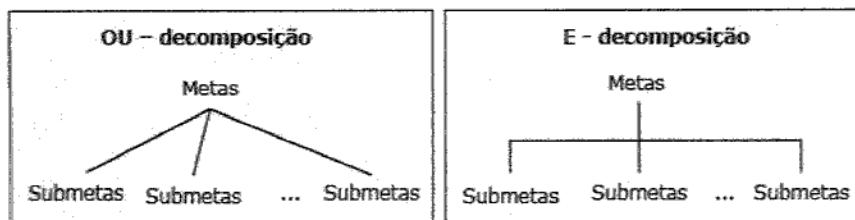
Muitos métodos na engenharia de requisitos são baseados nas considerações explícitas das intenções dos *stakeholders* através de metas [van Lamsweerde et al. 1991] e [Yu 1997]. Normalmente, é mínimo o esforço exigido para explicitamente considerar metas durante a engenharia de requisitos. No entanto, a modelagem de metas tem um impacto muito positivo sobre a engenharia de requisitos e sobre a qualidade e abrangência dos requisitos. Metas são a descrição – feita por um *stakeholder* (isto é, uma pessoa ou uma organização) – de uma propriedade característica do sistema a ser desenvolvido ou do projeto de desenvolvimento.

Metas são muito apropriadas para refinar a visão de um sistema. Refinar uma meta também é conhecido como decomposição de metas. Metas podem ser documentadas usando linguagem natural (por exemplo, através do uso de *templates* previamente preparados) ou modelos de metas. Uma técnica de modelagem de metas largamente conhecida e empregada é o uso de árvores E/OU. Por meio de árvores E/OU, decomposições hierárquicas podem ser documentadas. O tipo de relação de refinamento é descrito pela representação gráfica das ramificações, ou arestas. A direção da decomposição de metas não é representada pelas arestas, mas pela estrutura *top-down* da árvore.

Documentação baseada em linguagem natural e em modelos

### 6.2.1 Documentação de Metas Usando Árvores E/OU

Podemos distinguir dois tipos de relações de decomposição usando árvores E/OU. A figura 6-1 mostra de forma esquemática esses tipos de decomposição, bem como seus elementos de modelagem.



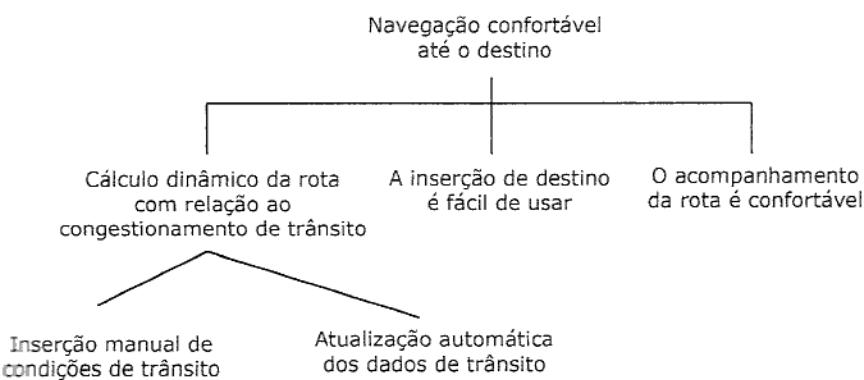
**Figura 6-1** Modelagem de decomposição de metas usando árvores E/OU

Com respeito às relações de decomposição, diferenciamos entre a decomposição-E e a decomposição-OU. No caso da decomposição-E, cada uma das metas subordinadas deve ser atingida para que a meta superordenada (a raiz) seja atingida. Em contrapartida, na decomposição-OU, para que a meta superordenada seja atingida é suficiente que pelo menos uma meta subordinada seja atingida.

Decomposição-E vs.  
decomposição-OU

### 6.2.2 Exemplo de Árvores E/OU

A figura 6-2 mostra uma árvore E/OU que documenta a decomposição hierárquica da meta "Navegação até o destino".



**Figura 6-2** Modelo de meta na forma de uma árvore E/OU

Como mostra o modelo de meta na figura 6-2, a meta "navegação até o destino" é refinada em três metas subordinadas – "cálculo dinâmico da rota em relação a congestionamentos de trânsito", "inserção do destino" e "acompanhamento da rota" – através de uma decomposição E. O modelo expressa que todas as metas subordinadas devem ser atingidas para que a meta superordenada possa ser considerada atingida. A meta subordinada "cálculo dinâmico da rota em relação a congestionamentos de trânsito", por sua vez, é refinada em duas metas subordinadas: "inserção manual de condições de trânsito" e "atualização automática de dados de trânsito". O tipo de relação de decomposição expressa que apenas uma das metas subordinadas precisa ser atingida para que a meta superordenada possa ser considerada atingida.

Modelagem de metas com árvores E/OU

## 6.3 Casos de Uso

Casos de uso foram propostos pela primeira vez por [Jacobson et al. 1992] como um método para documentar as funcionalidades de um sistema em planejamento ou existente por meio de modelos simples. A abordagem dos casos de uso é baseada em dois conceitos utilizados em conjunto:

- Diagramas de Casos de Uso.
- Especificações de Casos de Uso.

### 6.3.1 Diagramas de Casos de Uso UML

Diagramas de casos de uso UML [OMG 2007] (ver seção 4.2.3) são modelos simples para documentar de forma esquemática as funções de um sistema a partir do ponto de vista do usuário, bem como as inter-relações das funções de um sistema e as relações entre essas funções e seu ambiente.

#### Elementos de Modelagem dos Diagramas de Casos de Uso UML

A figura 6-3 apresenta os elementos de modelagem mais essenciais dos diagramas de casos de uso, conforme definidos na UML [OMG 2007].

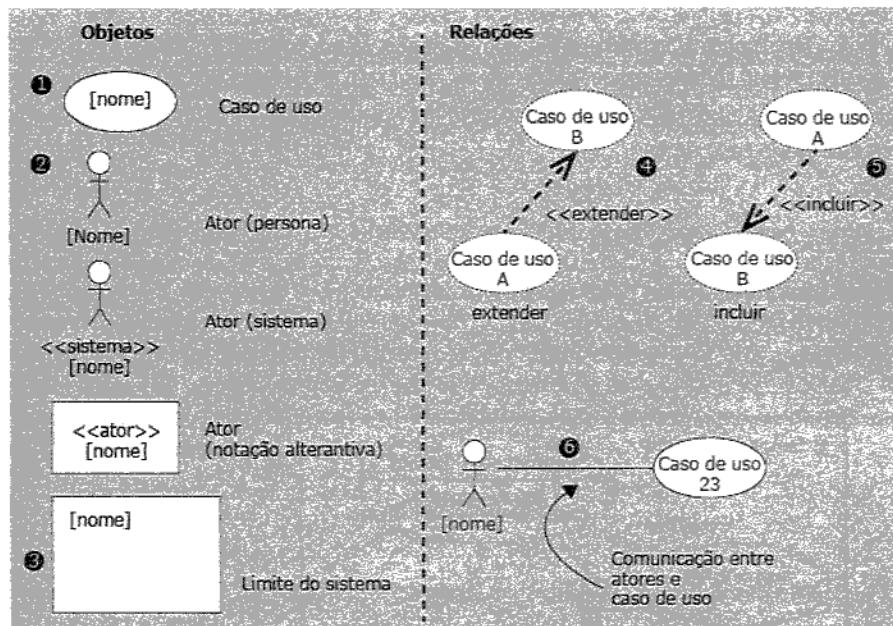


Figura 6-3 Elementos essenciais de modelagem em diagramas de casos de uso

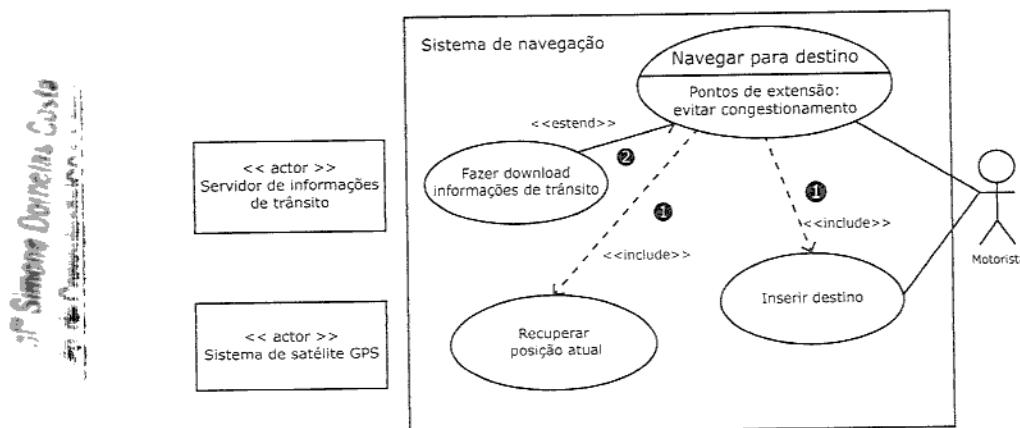
1. *Casos de uso:* Definidos para o sistema são representados por meio de elipses. Essas elipses contêm o nome do caso de uso. Outra alternativa é escrever o nome embaixo da elipse.
2. *Atores:* Encontram-se fora do limite do sistema e representam pessoas ou sistemas que interagem com o sistema modelado. Atores são representados por um retângulo que recebe o nome do ator e o rótulo «ator». Se o ator for uma pessoa, pode ser utilizada uma figura humana estilizada. Se o ator for um sistema, tanto pode ser utilizado um retângulo ou uma figura, em conjunto com o rótulo «sistema».
3. *Limites do sistema:* Dentro do diagrama de casos de uso separam as partes do caso de uso que pertencem ao sistema, das partes (pessoas ou sistemas) que estão fora do limite do sistema. Como opção, o nome do sistema pode ser anotado junto ao limite do sistema no diagrama.
4. *Relação de extensão (extend):* Uma relação de extensão expressa que uma sequência de interações que faz parte do caso de uso A estende alguma sequência de interações para o caso de uso B em um ponto especificado.
5. *Relação de inclusão (include):* Uma relação de inclusão de um caso de uso para outro caso de uso indica que a sequência de interações do primeiro

caso de uso inclui a sequência de interações do outro caso de uso.

- ⑥ **Relação entre atores e casos de uso:** Se há comunicação entre um caso de uso e um ou mais atores durante a execução de um caso de uso, a comunicação deve ser anotada através de uma relação de comunicação entre os respectivos atores e o caso de uso.

### Exemplo de Diagramas de Caso de Uso UML

A figura 6-4 apresenta um exemplo de um diagrama de caso de uso.



**Figura 6-4** Um exemplo usando elementos de modelagem em diagramas de caso de uso

O modelo abrange os elementos de casos de uso "fazer download de informações de trânsito", "recuperar posição atual" e "inserir destino". As relações na figura 6-4 identificadas por números são explicadas em maior detalhe abaixo:

- ① O caso de uso "navegar para destino" está associado aos casos de uso "inserir destino" e "recuperar posição atual" através de uma relação de inclusão. A relação indica que os passos de interação definidos nos casos de uso "inserir destino" e "recuperar posição atual" estão incluídos no caso de uso "navegar para destino".
- ② A relação de extensão entre os casos de uso "fazer download de informações de trânsito" e "navegar para destino" indica que os passos de interação definidos no caso de uso "fazer download de informações de

*Include*

*Extend*

"trânsito" estão incluídos nos passos de interação do caso de uso "navegar para destino" se uma certa condição, tal como "evitar congestionamento", estiver dada. O ponto de extensão "evitar congestionamento" indica o passo no caso de uso "navegar para destino" no qual os passos adicionais de interação são executados.

A UML também fornece uma relação de generalização entre casos de uso e atores. Nesse caso, os casos de uso ou atores "especialistas" herdam as propriedades dos casos de uso ou dos atores "generalistas" [Rumbaugh et al. 2005]. Por exemplo, os atores "mecânico da assistência técnica" e "representante do atendimento ao cliente" podem ser generalizados como o ator "funcionário". O ator generalista reúne todos os aspectos que os atores "mecânico da assistência técnica" e "representante do atendimento ao cliente" têm em comum (por exemplo, a identidade do funcionário).

Generalização

### 6.3.2 Especificações de Casos de Uso

Diagramas de casos de uso apresentam as funções relevantes do sistema a partir da perspectiva de um usuário, bem como as relações específicas entre as funções do sistema ou entre funções do sistema e determinados aspectos no contexto do sistema. Com a exceção do nome do caso de uso e suas relações, os diagramas de casos de uso não documentam qualquer informação sobre os casos de uso individuais, tal como a interação sistemática entre um caso de uso e um ator. Essa informação é documentada textualmente por meio de *templates* adequados utilizados conjuntamente com diagramas de casos de uso.

A literatura pertinente propõe diferentes *templates* para a especificação textual de casos de uso (por exemplo, [Cockburn 2001]). Esses *templates* definem o tipo de informações que devem ser documentadas para um caso de uso e sugerem uma estrutura apropriada para as informações. Portanto, as referências de *template* documentam o conhecimento (baseado na experiência) a respeito da documentação textual estruturada de casos de uso. O *template* na tabela 6-1 é apropriado para especificar textualmente casos de uso.

*Templates* de referência para a documentação de casos de uso

Template para Documentação Textual de Caso de Uso		
Item	Seção	Conteúdo / Explicação
1	Identificação	Identificador único do caso de uso.
2	Nome	Denominação única do caso de uso.
3	Autores	Nomes dos autores envolvidos nessa descrição de caso

		de uso.
4	Prioridade	Importância do caso de uso de acordo com a técnica de priorização aplicada.
5	Criticalidade	Criticalidade do caso de uso, isto é, qual seria o dano causado pela falha do caso de uso.
6	Fonte	Identificação da fonte da qual foi elicitado o caso de uso ( <i>stakeholder/documento/sistema</i> ).
7	Responsável	O <i>stakeholder</i> responsável pelo caso de uso.
8	Descrição	Breve descrição do caso de uso.
9	Trigger	Nome do evento que dispara a execução do caso de uso.
10	Atores	Lista de todos os atores envolvidos neste caso de uso.
11	Pré-condições	Lista de todas as restrições que precisam ser atendidas antes que o caso de uso possa iniciar sua execução.
12	Pós-condições	Lista de todos os estados em que o sistema pode se encontrar imediatamente após a execução do cenário principal.
13	Resultado	Descrição dos resultados produzidos durante a execução do caso de uso.
14	Cenário principal	Descrição do cenário principal do caso de uso.
15	Cenários alternativos	Descrição dos cenários alternativos do caso de uso, ou lista dos eventos desencadeadores de cenários alternativos. Frequentemente, podem ocorrer pós-condições diferentes daquelas descritas no ponto (12).
16	Cenários de exceção	Descrição dos cenários de exceção do caso de uso, ou lista dos eventos desencadeadores de cenários de exceção. Frequentemente, pós-condições diferentes daquelas descritas no ponto (12) podem ocorrer.
17	Qualidades	Referências cruzadas para requisitos de qualidade.

Tabela 6-1 *Template* para documentação textual de caso de uso

O *template* para a especificação de casos de uso contém os seguintes atributos:

- Atributos para a identificação única de casos de uso (linhas 1 e 2).
- Atributos de gerenciamento (linhas 3 a 7).
- Atributos para a descrição do caso de uso (linha 8).
- Atributos específicos de caso de uso, por exemplo, o *trigger* (evento desencadeador) (linha 9), atores (linha 10), pré e pós-condições (linhas 11 e 12), o resultado do caso de uso (linha 13), o cenário principal (linha 14), os cenários alternativos e de exceção (linhas 15 e 16) e as referências cruzadas para os requisitos de qualidade (linha 17).

Ítems de um *template* de caso de uso

A tabela 6-2 apresenta a especificação do caso de uso "navegar para destino" através do *template* de referência sugerido na tabela 6-1.

Seção	Conteúdo / Explicação
Identificação	UC-12-37
Nome	Navegar para destino
Autores	John Smith, Sandra Miller
Prioridade	Importância para o sucesso do sistema: <b>Alta</b> Risco tecnológico: <b>Alto</b>
Criticalidade	Alta
Fonte	C. Warner (especialista do domínio para sistemas de navegação)
Responsável	J. Smith
Descrição	O condutor do veículo digita o nome do destino. O sistema de navegação guia o condutor para o destino desejado.
Trigger	O condutor deseja navegar até seu destino.
Atores	Condutor, servidor de informações de trânsito, sistema de satélite GPS.
Pré-condições	O sistema de navegação está ativado.
Pós-condições	O condutor chegou a seu destino.
Resultado	Orientação de rota até o destino.
Cenário principal	O sistema de navegação pergunta pelo destino desejado.  O condutor insere o destino desejado.  O sistema de navegação localiza o destino em seus mapas.  Baseando-se na posição atual e no destino desejado, o sistema de navegação calcula uma rota adequada.  O sistema de navegação coleta uma lista de pontos de rota ao longo do caminho.  O sistema de navegação mostra um mapa da posição atual e indica a rota para o próximo ponto de rota.  Quando o último ponto de rota for atingido, o sistema de navegação mostra as palavras "Destino alcançado" na tela.
Cenários alternativos	4a O cálculo da rota deve observar as informações de trânsito e evitar congestionamentos.  4a1. O sistema de navegação solicita informações de trânsito atualizadas do servidor.  4a2. O sistema de navegação calcula uma rota que

	não apresenta congestionamentos de trânsito.
Cenários de exceção	<i>Trigger:</i> O sistema de navegação não recebe o sinal de GPS do sistema de satélite. → RQ*.04 (tempo de reação após input do usuário)
Qualidades	→ RQ.15 (facilidade de operação)  * ( <i>RQ = Requisito de Qualidade</i> )

Tabela 6-2 Exemplo de documentação de caso de uso baseado em *template*

## 6.4 Três Perspectivas sobre Requisitos

Ao documentar requisitos com modelos, distingue-se tipicamente três tipos de perspectivas: dados, função e comportamento (ver seção 4.2.1). Cada perspectiva é documentada separadamente, usando linguagens de modelagem conceitual apropriadas [Davis 1993], [Pohl et al. 2005]: Documentação separada por cada perspectiva

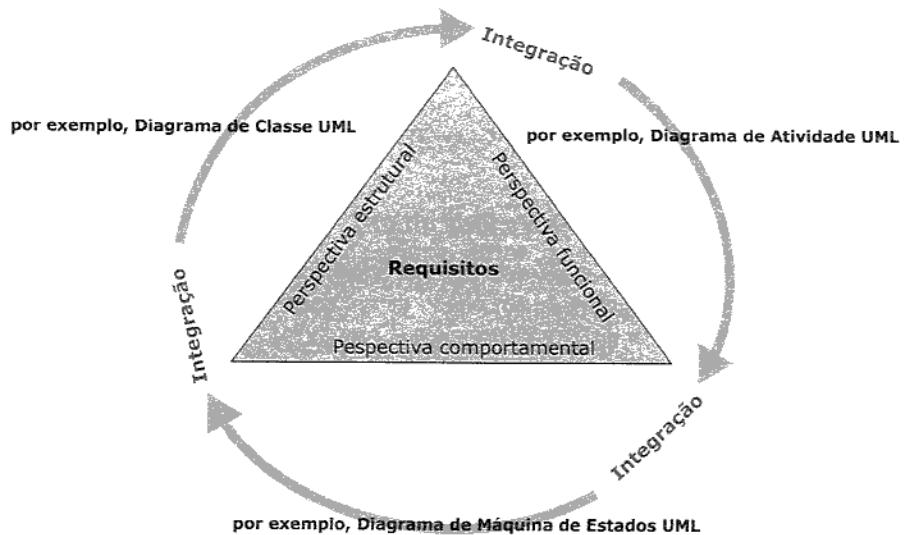
- *Perspectiva estrutural:* Nessa perspectiva, são documentadas as estruturas dos dados de entrada e saída, bem como aspectos estático-estruturais das relações de uso e dependência do sistema no contexto do sistema.
- *Perspectiva funcional:* Essa perspectiva documenta qual informação do contexto do sistema está sendo manipulada pelo sistema a ser desenvolvido e quais dados estão sendo transmitidos para o contexto do sistema pelo sistema.
- *Perspectiva comportamental:* Nessa perspectiva, documenta-se a integração do sistema no contexto do sistema com base em estados. Isto pode ser feito, por exemplo, pela documentação da reação do sistema frente a eventos que ocorrem no contexto do sistema, das condições que desencadeiam uma mudança de estado ou dos efeitos que o sistema tem sobre seu ambiente.

A figura 6-5 ilustra as três perspectivas sobre requisitos funcionais e dá um exemplo de uma linguagem de modelagem adequada para cada perspectiva que pode ser utilizada para documentar os requisitos. Assim, aspectos de requisitos relacionados com a estrutura estática podem ser modelados usando diagramas de classes UML, por exemplo. Requisitos na perspectiva funcional podem ser modelados usando diagramas de atividade UML. Requisitos na perspectiva comportamental podem ser modelados usando diagramas de estados (ver seções 6.6 e 6.7).

Exemplos das três perspectivas

Certos aspectos dos modelos de uma perspectiva específica podem também ser encontrados em outras perspectivas. As três perspectivas, portanto, não são totalmente separadas. Por exemplo, os dados que têm sua estrutura estática definida num diagrama de classes UML podem eventualmente também ser encontrados na perspectiva funcional, pois representam as entradas e saídas de ações em um diagrama de atividade UML. Como as três perspectivas não são separadas, os modelos podem ser verificados reciprocamente em termos de completude e consistência com respeito à informação modelada nas interseções.

Perspectivas não  
são totalmente  
separadas



**Figura 6-5** Três perspectivas sobre requisitos

## 6.5 Modelagem de Requisitos na Perspectiva Estrutural

Várias linguagens de modelagem diferentes são apropriadas para modelar aspectos estruturais de requisitos na perspectiva estrutural. Normalmente, modelos de entidade-relacionamento, extensões do modelo tradicional de entidade-relacionamento conforme Chen [Chen 1976] e, cada vez mais, diagramas de classes UML (por exemplo, [Rumbaugh et al. 2005] estão sendo usados como modelos para requisitos na perspectiva estrutural.

### 6.5.1 Diagramas Entidade-Relacionamento

Diagramas entidade-relacionamento são tradicionalmente utilizados para modelar a perspectiva estrutural pois apresentam a estrutura de um objeto de um universo de discurso através de tipos de entidades e tipos de relacionamentos [Chen 1976].

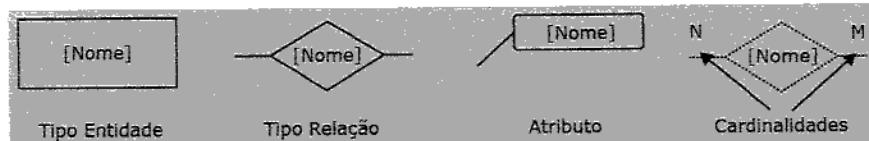
Várias extensões têm sido sugeridas para o modelo de entidade-relacionamento. Essas extensões envolvem principalmente os relacionamentos de generalização/especialização, os mecanismos de herança e os papéis das entidades, estendendo o modelo com uma notação (mín, máx) para as cardinalidades dos relacionamentos.

O modelo tradicional de entidade-relacionamento

Extensões para o modelo de entidade-relacionamento

#### Elementos de modelagem de Diagramas de Entidade-Relacionamento

Conforme [Chen 1976], a linguagem de modelagem utilizada para construir diagramas de entidade-relacionamento inclui os elementos de modelagem apresentados na figura 6-6.



**Figura 6-6** Importantes elementos de modelagem em diagramas de entidade-relacionamento, conforme Chen

*Tipos de entidades* definem um conjunto de entidades dentro do universo de discurso (isto é, objetos com as mesmas propriedades, tais como pessoas ou itens). Um tipo de entidade (muitas vezes erroneamente chamado de entidade) abstrai a

Classificação:  
Abstração a partir de objetos

partir das características concretas dessas entidades e consequentemente classifica um conjunto em entidades uniformes. Por exemplo, o tipo de entidade "piloto" classifica todas as pessoas dentro do universo de discurso que têm a característica de possuirem uma licença de pilotagem.

Um *tipo de relacionamento* abstrai a partir de uma característica concreta de um relacionamento e de entidades relacionadas entre si. Um tipo de relacionamento classifica o conjunto de relacionamentos uniformes entre tipos de entidades dentro do universo de discurso. Por exemplo, o tipo de relacionamento "executa" pode ser definido entre os dois tipos de entidade "piloto" e "voo" para representar relacionamentos "executa" concretos entre pilotos concretos e voos concretos. Se um relacionamento concreto "é passageiro" for definido entre um passageiro concreto "John Locke"<sup>5</sup> e um voo concreto com o número do voo "OA 815"<sup>6</sup>, então esse relacionamento indica que "John Locke" é um passageiro do voo com o número de voo "OA 815".

Um *atributo* pode se definido tanto para tipos de entidades quanto para tipos de relacionamentos. Um atributo define as propriedades de um tipo de entidade ou de um tipo de relacionamento. Atributos possíveis para o tipo de entidade "passageiro" poderiam ser, por exemplo: "nome de família", "nome próprio", "número do passaporte" e "número do assento".

Um modelo de entidade-relacionamento documenta a estrutura de um universo de discurso por meio de tipos de entidade (isto é, classes de entidades uniformes) e relacionamentos (isto é, classes de relacionamentos uniformes). Um modelo entidade-relacionamento é definido no nível de modelagem e define o conjunto de todas as instâncias válidas no nível de instância.

A *cardinalidade* de um relacionamento (binário) define o número de instâncias de relacionamento das quais uma entidade pode fazer parte [ElsMari e Navathe 2006]. Se nenhuma cardinalidade está anotada para um tipo de relacionamento específico, assume-se que um número arbitrário de entidades (em outras palavras, no mínimo zero entidades) podem participar de tal relacionamento. Usar cardinalidades para relacionamentos, portanto, limita o número de instâncias que são em princípio possíveis em um diagrama entidade-relacionamento.

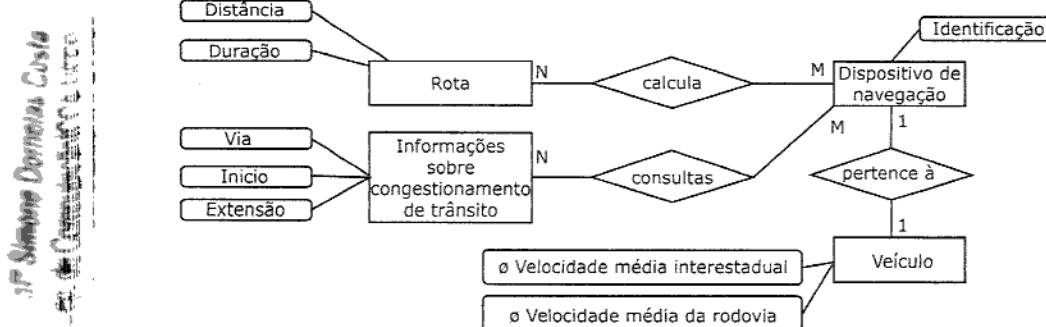
### Exemplo de um Diagrama Entidade-Relacionamento

O modelo entidade-relacionamento apresentado na figura 6-7 mostra quatro tipos de entidades (isto é, classes de entidades) e três tipos de relacionamento (isto é, classes de relacionamentos). Os tipos de entidade individuais possuem atributos

<sup>5</sup> Mais precisamente, se há uma entidade que é uma instância do tipo de entidade "passageiro" e que possui uma identidade única e o valor de atributo "John Locke" para o atributo "nome".

<sup>6</sup> Mais precisamente, se há uma entidade que é uma instância do tipo de entidade "voo" e que possui uma identidade única e o valor de atributo "OA 815" para o atributo "número do voo".

que descrevem propriedades específicas das entidades associadas. Por exemplo, o tipo de entidade "informações sobre congestionamento de trânsito" possui os atributos "via", "início" e "extensão", que indicam, respectivamente, a via na qual no momento há um congestionamento, as coordenadas GPS do ponto inicial do congestionamento e a extensão do congestionamento. O tipo de relacionamento "consulta" entre os tipos de entidade "dispositivo de navegação" e "informações sobre congestionamento de trânsito" significa que no nível de instância existe um relacionamento entre um dispositivo de navegação concreto e a informação sobre zero ou mais congestionamentos concretos. As cardinalidades dos tipos de entidades em relação ao tipo de relacionamento "consulta" significam que um dispositivo de navegação concreto pode consultar informações sobre um número arbitrário ("N") de congestionamentos de trânsito. Na direção inversa, qualquer informação sobre congestionamentos de trânsito pode ser consultada por um número arbitrário ("M") de dispositivos de navegação.



**Figura 6-7** Diagrama entidade-relacionamento(modelo de dados) conforme Chen

### 6.5.2 Diagramas de Classes UML

Diagramas de classes UML também podem ser utilizados para modelar a perspectiva estrutural dos requisitos de um sistema a ser desenvolvido. Um diagrama de classes consiste em um conjunto de classes e associações entre classes. Classes e associações em diagramas de classes UML são semelhantes a tipos de entidades e tipos de relacionamentos nos diagramas entidade-relacionamento. Modelos de classes possuem elementos adicionais de modelagem (por exemplo, elementos que permitem a especificação de operações válidas nas instâncias de uma classe) e têm, portanto, maior poder descritivo.<sup>7</sup>

Perspectiva  
estática: Dados /  
Estrutura

<sup>7</sup> Uma visão geral dos diferentes elementos de modelagem UML pode ser encontrada, por exemplo, em [OMG 2007].

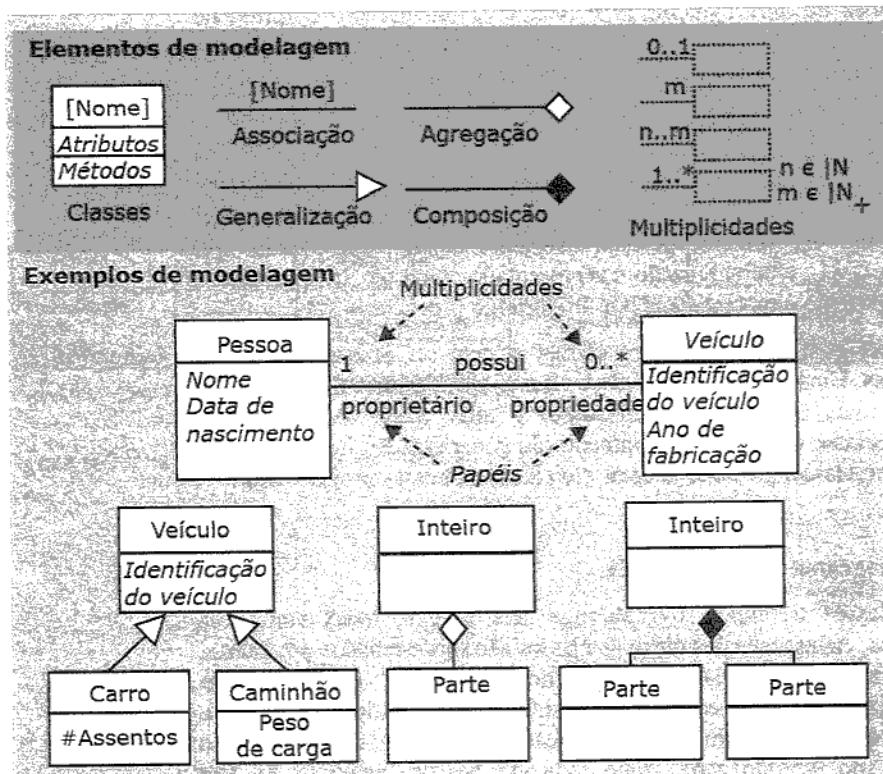


Figura 6-8 Importantes elementos de modelagem em diagramas de classes UML

### Elementos de Modelagem de Diagramas de Classes

A figura 6-8 apresenta importantes elementos de modelagem de diagramas de classe UML, bem como diversos exemplos de modelagem.

Uma *classe* é representada por um retângulo separado em seções (também chamadas de compartimentos). A seção superior apresenta atributos que são descritos em maiores detalhes pela instância da classe. Na seções inferiores, encontram-se listadas todas as operações que podem ser executadas com as instâncias da classe. Dependendo do objetivo da modelagem, isto é, dependendo da finalidade do modelo, os compartimentos para atributos e/ou métodos também podem ser ocultados ou excluídos completamente.

*Associações* entre classes são representadas por linhas. Associações podem opcionalmente receber nomes. Além disso, *multiplicidades* podem ser anotadas em cada ponta de uma associação. Multiplicidades são afirmações sobre o nível de instância de uma classe e representam quantas instâncias de uma classe

Classes

Associações,  
multiplicidades  
e papéis

podem ser associadas de uma maneira específica com um número definido de instâncias de outra classe. Ao anotar *papéis* opcionais em uma das pontas de uma associação, ou em ambas, o significado das instâncias de uma classe em relação à associação pode ser refinado adicionalmente.

**Agregações e composições** são tipos específicos de associações. Ambos descrevem um relacionamento entre um todo e suas partes constituintes. Uma composição documenta uma associação mais forte do que uma agregação, pois uma parte constituinte de uma composição não pode existir sem o todo. Em modelos de classes UML, uma agregação é representada como um losango vazio, enquanto uma composição é representada por um losango preenchido.

Além disso, **generalizações** entre classes podem ser documentadas em diagramas de classes. Uma generalização entre classes UML é um relacionamento entre uma classe mais específica (a subclasse) e uma classe mais geral (a superclasse). A subclasse em um relacionamento de generalização herda todas as propriedades da superclasse, podendo adaptar e/ou aumentar essas propriedades.

Agregação e  
composição

Generalização

### Exemplo de um Diagrama de Classes UML

O diagrama de classes na figura 6-9 abrange seis classes, todas com seus respectivos atributos. As associações entre as classes são representadas por linhas. Por exemplo, existe uma associação com o nome "calcula" entre a classe "dispositivo de navegação" e a classe "rota". Considerando as multiplicidades, essa associação indica que um sistema de navegação pode calcular um número arbitrário de rotas (no mínimo zero, como indicado por um asterisco \*). Em contrapartida, cada rota pode ser calculada por um número arbitrário (\*) de dispositivos de navegação. Uma rota é uma agregação de no mínimo um, mas arbitrariamente muitos (1...\*) segmentos de vias, e cada segmento de via pertence a um número arbitrário (\*) de rotas. Um segmento de via é definido por um nome de via e por pontos iniciais e finais. A figura 6-9 também mostra que "dispositivo de navegação com função evitar congestionamentos" é uma especialização da classe geral "dispositivo de navegação". A subclasse "dispositivo de navegação com função evitar congestionamentos" herda as propriedades (neste caso, o atributo "identificação") de sua superclasse "dispositivo de navegação" e aumenta o conjunto de atributos com um atributo que especifica um limite determinado para a extensão do congestionamento (limiar de congestionamento), o qual, uma vez ultrapassado, desencadeia um recálculo da rota.

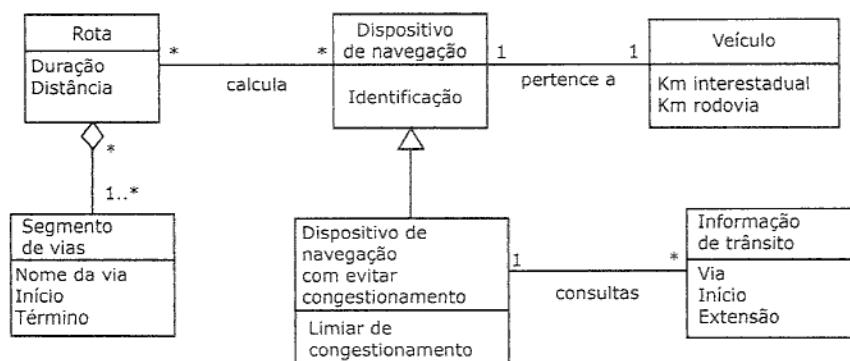


Figura 6-9 Diagrama de classes em notação UML

## 6.6 Modelagem de Requisitos na Perspectiva Funcional

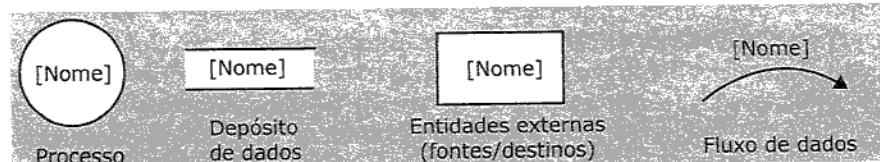
A perspectiva funcional de requisitos está voltada para a transformação de dados de entrada recebidos do ambiente em dados de saída liberados para o ambiente do sistema. Existem várias abordagens diferentes baseadas em modelos que podem ser utilizadas para modelar a perspectiva funcional dos requisitos. A maioria dessas técnicas baseia-se nas abordagens de análise estruturada de sistemas dos anos 70 e 80, tais como a análise estruturada [DeMarco 1978, Weinberg 1978] ou a análise essencial de sistemas [McMenamin e Palmer 1988].

### 6.6.1 Diagramas de Fluxo de Dados

No centro das atenções da modelagem de requisitos na perspectiva funcional encontram-se diagramas que modelam a funcionalidade do respectivo sistema por meio de processos (funções), repositórios de dados, fontes (*sources*) e destinos (*sinks*) no ambiente do sistema, bem como fluxos de dados. Um tipo de modelo funcional de uso comum são os diagramas de fluxos de dados, conforme propostos na análise estruturada [DeMarco 1978]. Modelos de fluxo de dados permitem a modelagem do sistema em diferentes níveis de abstração.

#### Elementos de Modelagem nos Diagramas de Fluxo de Dados

A figura 6-10 apresenta os elementos de modelagem encontrados em diagramas de fluxo de dados, na notação proposta por [DeMarco 1978].



**Figura 6-10** Importantes elementos de modelagem para diagramas de fluxo de dados, conforme DeMarco

**Processos** representam as funções necessárias em um determinado sistema para transformar os dados que fluem para o sistema (fluxo de informações).<sup>8</sup> Um processo absorve os dados de entrada, processa esses dados e gera o resultado sob forma de dados de saída. De que maneira (isto é, *como*) os dados são transformados não está representado nos diagramas de fluxo de dados.

Manipulação de dados

**Repositórios, ou depósitos de dados** são conceitos abstratos projetados para representar dados persistentes. Processos podem acessar dados em um depósito de dados no modo leitura ou escrita. Dessa forma, os processos podem tanto obter acesso a dados necessários de entrada quanto armazenar de forma permanente dados de saída.

Dados persistentes

**Fontes/destinos** descrevem objetos (como pessoas, grupos de pessoas, departamentos, organizações ou sistemas) no ambiente do sistema que trocam dados com o sistema. Fontes/destinos são aspectos do ambiente do sistema e não podem ser alterados durante o desenvolvimento do sistema (ver seção 2.1). Fontes são aspectos do ambiente do sistema que fornecem dados ao sistema, ao passo que destinos recebem dados do sistema.

Objetos no ambiente do sistema

Um **fluxo de dados** descreve dados que são transportados entre processos, depósitos de dados e fontes/destinos [Yourdon 1989]. Nos modelos de requisitos, um fluxo de dados pode modelar o transporte de objetos materiais e imateriais, isto é, fluxo de dados ou fluxo de material. Tipicamente, apenas os fluxos de dados mais importantes são modelados em diagramas de fluxo de dados. Fluxos de dados irrelevantes para os requisitos do sistema podem ser ignorados.

Fluxo de dados

### Exemplo de um Diagrama de Fluxo de Dados

A figura 6-11 apresenta um diagrama simplificado de fluxo de dados para um sistema de navegação, usando a notação proposta por DeMarco. As interfaces do sistema para o contexto são definidas pelos fluxos de dados para as fontes "sistema de satélite GPS" e "servidor de informações de trânsito", bem como para o destino "condutor".

Interfaces do sistema

A funcionalidade do sistema de navegação é dividida em três processos distintos. O processo 1, denominado "calcular rota", recebe informações de trânsito atualizadas por meio de sua interface com a fonte "servidor de informações de trânsito", bem como dados sobre a localização atual do veículo através de sua interface com a fonte "sistema de satélite GPS". Além disso, o processo "calcular rota" recebe do condutor do veículo o destino desejado. A rota calculada é

Processo "calcular rota"

<sup>8</sup> Na análise estruturada, o fluxo de dados, informações, documentos ou materiais são considerados um fluxo de dados.

SI 03 (2012) 2070

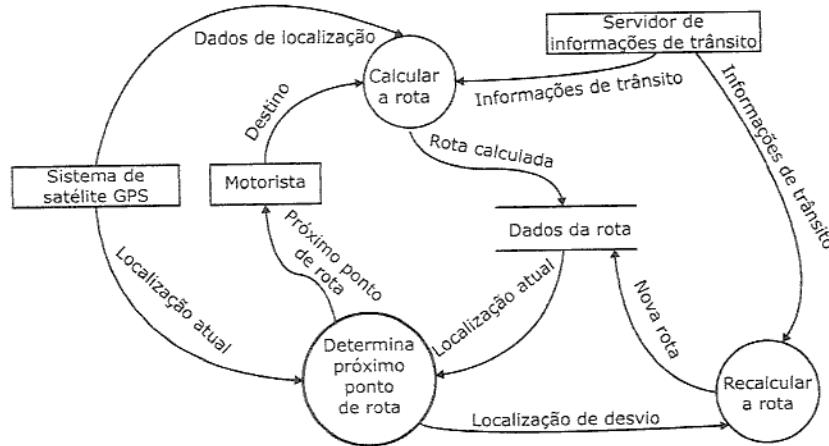
armazenada no depósito de dados "dados da rota".

O processo 2, "determinar o próximo ponto de rota", acessa o depósito de dados para dele extrair dados relacionados com a rota atual. O processo determina o próximo ponto de rota e gera essa informação.

O processo 3, "recalcular rota", traça uma nova rota para o destino. Para tal, o processo coleta informações de trânsito a partir da fonte "servidor de informações de trânsito" e, eventualmente, informações sobre a posição atual. A rota recalculada é armazenada no depósito de dados "dados da rota".

Processo  
"determinar  
novo ponto de  
rota"

Processo  
"recalcular rota"



**Figura 6-11** Diagrama de fluxo de dados, na notação proposta por DeMarco

### 6.6.2 Modelos da Perspectiva Funcional e Fluxos de Controle

Em diagramas de fluxo de dados não é possível visualizar quais condições desencadeiam quais processos. Os diagramas apenas apresentam as dependências de dados dos processos em um sistema e documentam os dados de entrada necessários, bem como os dados de saída gerados. Abordagens utilizadas na análise estruturada de sistemas, entretanto, muitas vezes oferecem descrições adicionais de comportamento e de fluxo de controle. Isso pode ser alcançado seja pelo uso de formas específicas de documentação, como mini-especificações na análise estruturada, seja através de extensões implícitas de linguagem dos modelos de fluxo de dados. Extensões de linguagem possibilitam a modelagem de aspectos adicionais, por exemplo, o fluxo de controle entre funções, como em SA/RT [Ward e Mellor 1985, Hatley e Pirbhai 1988].

### 6.6.3 Diagramas de Atividades UML

Diagramas de atividades UML são apropriados para modelar sequências de ações [OMG 2007]. Além desses diagramas, podem ser utilizadas cadeias de processos determinados por eventos (EPC, na sigla em inglês, ou *event-driven process chains*) para modelar sequências de atividades [Keller et al. 1992], especialmente no desenvolvimento de sistemas de informação. Diagramas de atividades UML representam o fluxo de controle entre atividades ou ações. No caso de uma progressão sequencial de ações, uma ação subsequente é executada depois que cada ação precedente termina. A figura 6-12 apresenta importantes elementos de modelagem dos diagramas de atividades UML [OMG 2007].

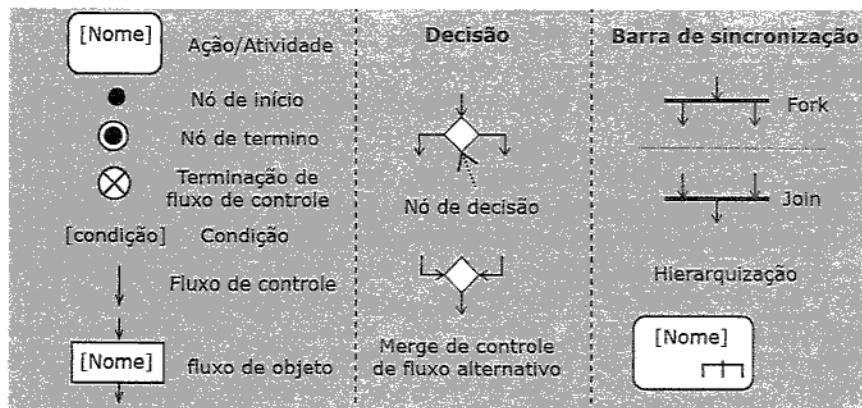


Figura 6-12 Elementos de modelagem em diagramas de atividades UML

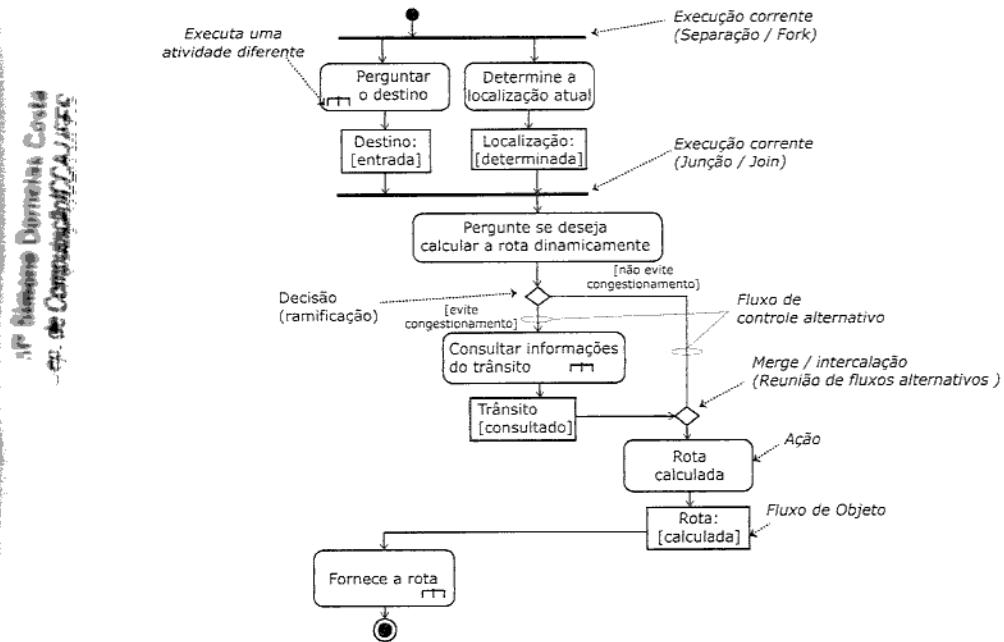
Diagramas de atividades são gráficos de fluxo de controle que consistem de nós de atividade e do fluxo de controle entre esses nós (isto é, as setas no gráfico de fluxo de controle representando transições). Nós de atividade executam uma ação. Os nós de início e de fim nos diagramas de atividades têm uma semântica definida. O nó de início representa um evento que inicia a execução do diagrama de atividades. Os nós de fim são nós especiais que representam o fim do diagrama de atividades.

Nós de atividade

A representação de fluxos de controle em diagramas de atividades pode ser obtida através do uso de nós de decisão. Nesses nós de decisão são anotadas as condições que desencadeiam fluxos alternativos de controle. Além disso, as barras de sincronização permitem a execução concorrente de fluxos de objetos. Um tipo especial de fluxos de controle são os fluxos de objetos. Através do uso de linhas divisórias de atividades (*swimlanes*), diferentes atividades podem ser documentadas como responsabilidades de atores específicos.

### Modelagem de Sequência usando Diagramas de Atividades UML

O diagrama de atividades na figura 6-13 documenta o processo "navegar para destino". Dados de entrada e saída podem ser documentados através da modelagem de fluxos de objetos adicionais ao longo das linhas. Os fluxos de dados e objetos são tipos especiais de fluxos de controle do diagrama de atividades. Cada ação é executada se, e somente se, determinadas ações prévias foram executadas e todos os fluxos de entrada de objetos estiverem disponíveis. O diagrama de ação na figura 6-13 também apresenta fluxos de objetos que são documentados além das ações e dos fluxos de controle.



**Figura 6-13** Diagrama de atividades em notação UML

O diagrama de atividades acima documenta a sequência de ações necessárias para que um dispositivo de navegação calcule uma rota. O modelo documenta que inicialmente o destino desejado é solicitado e a localização atual do veículo é determinada. Essas duas ações ocorrem simultaneamente, independentes uma da outra. O destino informado (fluxo de objeto: objeto → destinação; estado → inserir) e a localização determinada (fluxo de objeto: objeto → localização; estado → determinada) são transmitidos. Caso o condutor tenha selecionado a opção de evitar congestionamentos de trânsito automaticamente, o sistema busca informações atualizadas de trânsito. Uma vez recebidas as informações atualizadas

de trânsito, ou caso o condutor não tenha selecionado a opção de evitar congestionamentos de trânsito automaticamente, o sistema calcula uma rota até o destino. A rota calculada é transmitida para o condutor.

Diagramas de atividades são apropriados para documentar os relacionamentos e as condições de execução de cenários principais, alternativos e de exceção. Nós de decisão representam ramificações no fluxo de controle entre o cenário principal e os cenários alternativos e de exceção.

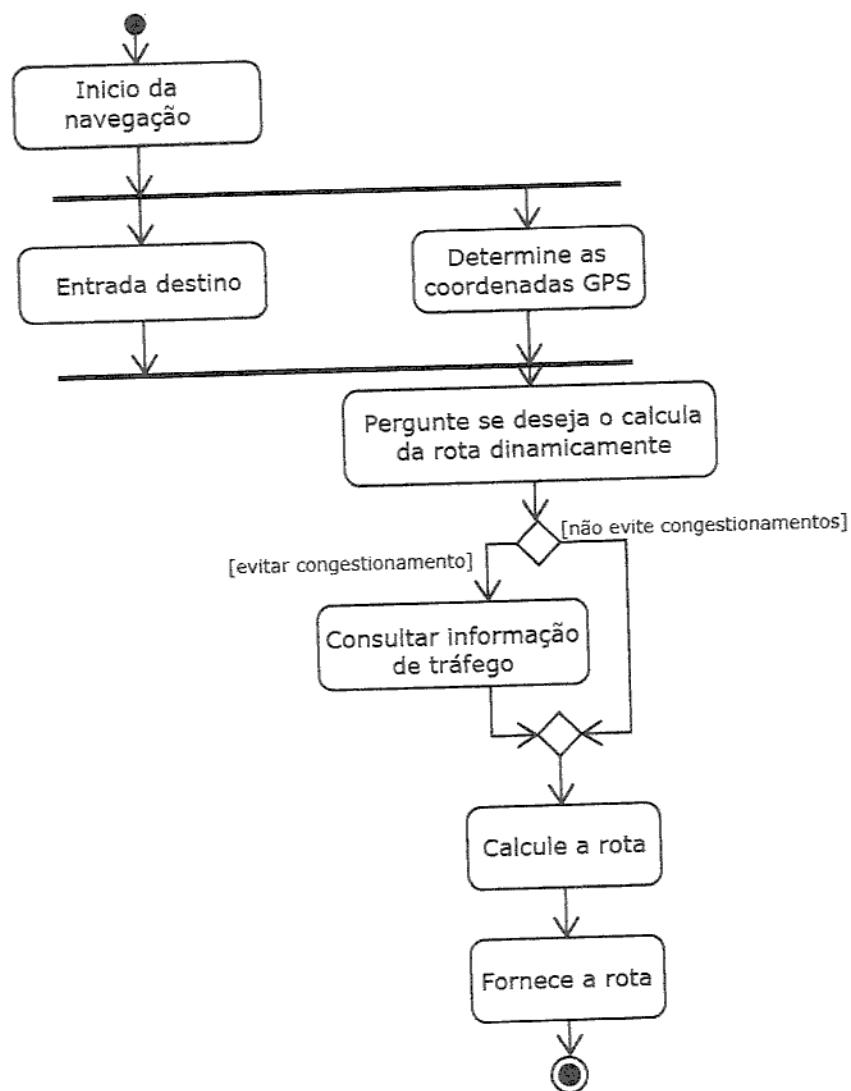
Sequências de modelagem de um caso de uso

### Fluxo de Controle de Cenários Principais e Alternativos

O diagrama de atividades na figura 6-14 apresenta o fluxo de controle do cenário principal e alternativo do caso de uso "navegar para destino" documentado na figura 6-2. Ramificações alternativas de fluxo de controle começam nos nós de decisão que documentam os cenários respectivamente alternativos e de exceção de um cenário principal específico.

O diagrama de atividades indica que inicialmente a ação "começar navegação" é executada. Depois disso, as ações "inserir destino" e "determinar as coordenadas GPS" são executadas simultaneamente e de forma independente entre si. Uma vez executadas as duas ações, o sistema pergunta se o condutor deseja que a rota seja calculada dinamicamente (ação "perguntar se deseja calcular a rota dinamicamente"). Se o condutor não solicitar que a rota seja calculada dinamicamente (seleção "não evitar congestionamentos de trânsito"), nenhuma ação específica será tomada (ver tabela 6-1 → cenário principal). Se o condutor selecionar o cálculo dinâmico de rota (seleção "evitar congestionamentos"), informações atualizadas de trânsito serão determinadas (ação "consultar informações de trânsito", ver tabela 6-1 → cenário alternativos). Depois disso, a rota é calculada (ação "cálculo da rota") e informada ao condutor (ação "informar rota").

Cenário principal e cenário alternativo



**Figura 6-14** Documentação do fluxo de controle de cenários usando diagramas de atividades UML

## 6.7 Modelagem de Requisitos na Perspectiva Comportamental

Para modelar o comportamento dinâmico de um sistema, empregam-se tipicamente abordagens de modelagem baseadas na teoria dos autômatos. A definição de uma máquina de estados finitos engloba um conjunto de estados e um conjunto de transições que, dependendo do estado atual da máquina, são executadas a partir de um determinado evento.

No âmbito da modelagem de sistema, utilizam-se com frequência extensões de máquinas de estados finitos baseadas nos conceitos das chamadas máquinas de Mealy [Mealy 1955] e máquinas de Moore [Moore 1956], respectivamente. Nas máquinas de Mealy, os dados de saída dependem do estado atual da máquina e dos dados de entrada. Em contraste, nas máquinas de Moore, os dados de saída dependem exclusivamente do estado atual.

Máquinas de estados finitos

Máquinas de Mealy e Moore

Statecharts =  
Máquinas de estados +  
hierarquização +  
condições +  
concorrência

### 6.7.1 Statecharts

Devido aos desafios associados ao uso de máquinas de estados finitos na prática (tais como a falta de apoio para abstração), o conceito de autômatos tornou-se uma técnica para modelar o comportamento reativo de um sistema. Uma técnica amplamente utilizada para modelar o comportamento de um sistema é o uso de *statecharts* [Harel 1987]. *Statecharts* são um tipo de autômatos baseados em máquinas de estados finitos, mas que são estendidos para suportar a hierarquização de estados com a finalidade de documentar condições de estados de transição e modelar comportamentos concorrentes. A Figura 6-15 apresenta os elementos de modelagem de *statecharts* na notação proposta por Harel [Harel 1987].

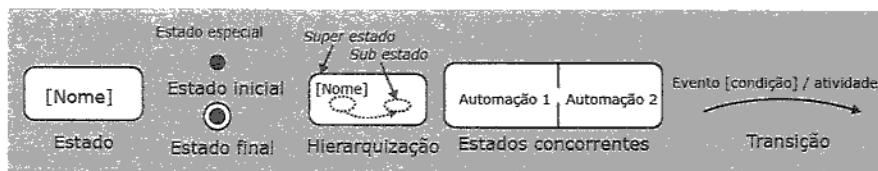


Figura 6-15 Elementos de modelagem em statecharts

Um **estado** define um momento no qual o sistema apresenta um comportamento específico e espera determinado evento ocorrer para então executar uma transição definida.

Estado

Uma **transição** é desencadeada por um evento específico que ocorre em um determinado estado. Uma transição descreve a mudança de um estado para o próximo. A mudança de estados pode adicionalmente depender de alguma condição. O sistema pode executar atividades específicas se o sistema se encontra em determinado estado (comportamento típico de máquinas de Moore) ou se o sistema executa uma transição para outro estado (comportamento típico de máquinas de Mealy). Essas atividades podem ser direcionadas para o próprio sistema ou para o ambiente do sistema.

Transição com condição e atividade

**Statecharts** permitem a decomposição hierárquica (ou refinamento hierárquico) de estados, que por sua vez representam autômatos. O estado inicial recebe a denominação superestado, sendo definido por diversos estados compostos. A hierarquização permite abstrair detalhes irrelevantes de um estado considerando e/ou modelando – dependendo da finalidade do modelo – apenas o superestado, e não o subautômato completo que define o superestado. O comportamento detalhado do sistema pode, caso necessário, ser decomposto (refinado) pela definição dos respectivos autômatos parciais.

Hierarquização e abstração

Além da decomposição hierárquica de um estado em autômatos detalhados, um estado pode ser decomposto em vários autômatos concorrentes. Os autômatos concorrentes podem ser sincronizados através de condições de transição (por exemplo, "estando o autômato A no estado 4"). A figura 6-16 apresenta um modelo comportamental de um dispositivo de navegação veicular através de um statechart. O dispositivo de navegação encontra-se inicialmente no estado "dispositivo de navegação inativo".

Concorrência

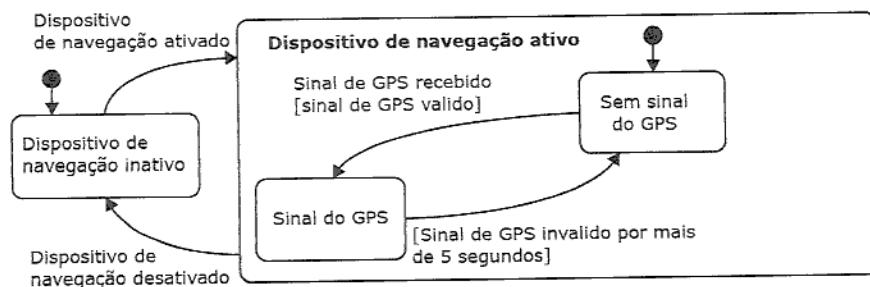


Figura 6-16 Statechart simplificado de um dispositivo de navegação veicular

Ao ligar o dispositivo de navegação (evento "dispositivo de navegação ativado"), o sistema transita para o superestado "dispositivo de navegação ativo" (mais precisamente, o sistema transita para estado inicial "sem sinal do GPS" do superestado "dispositivo de navegação ativo"). O superestado "dispositivo de navegação ativo" é refinado por um autômato parcial que consiste em dois estados. Por exemplo, se um sinal de GPS é recebido no estado "dispositivo de navegação

Transição para superestado

ativo: sem sinal do GPS<sup>9</sup>, o sistema transita para o estado "dispositivo de navegação ativo: sinal do GPS" e emite um aviso. Se o dispositivo é desativado quando se encontra no estado "dispositivo de navegação ativo" (evento: "dispositivo de navegação desativado"), o sistema transita para o estado "dispositivo de navegação inativo".

### 6.7.2 Diagramas de Estados UML

Para modelar o comportamento reativo de sistemas, a Unified Modeling Language (UML) [OMG 2007] oferece máquinas de estados essencialmente baseadas em *statecharts*. A figura 6-17 apresenta os mais importantes elementos de modelagem para diagramas de estados UML. A notação dos elementos de modelagem dos diagramas de estados UML foi em grande parte adaptada a partir dos *statecharts*. A UML 2, no entanto, estendeu os elementos de modelagem dos *statecharts* com a possibilidade, por exemplo, de definir pontos explícitos de entrada e saída de estados hierárquicos [OMG 2007].

Modelagem de comportamento reativo de um sistema usando UML

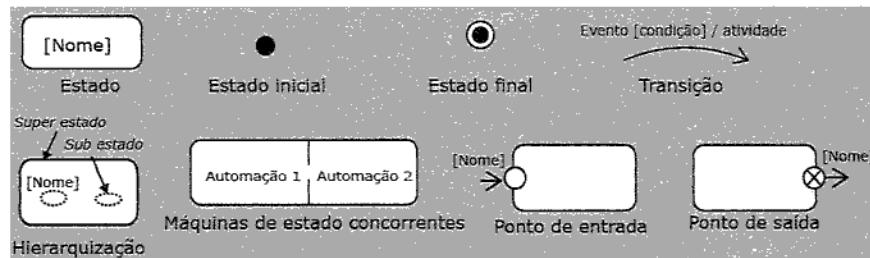


Figura 6-17 Elementos de modelagem UML 2 para máquinas de estados

Da mesma forma como nos *statecharts*, um estado define um período durante o qual um sistema apresenta um comportamento específico e está esperando que um determinado evento ocorra. Uma transição é desencadeada por um evento que ocorre em um estado específico e descreve a mudança de um estado para o próximo. Uma transição pode ser dependente de uma condição. Além disso, o sistema pode executar ações dirigidas ao próprio sistema ou a seu ambiente.

Estados e transições

Dependendo da finalidade do modelo, máquinas de estados permitem a combinação hierárquica de estados em superestados, dessa forma abstraindo os comportamentos potencialmente muito complexos desses estados. Além da

Hierarquização e concorrência

<sup>9</sup> Para fins de identificação única, um estado que faz parte de um superestado é representado da seguinte forma: "superestado: estado". Portanto, o estado "sem sinal do GPS" no superestado "dispositivo de navegação ativo" é representado como "dispositivo de navegação ativo: sem sinal do GPS".

decomposição hierárquica de estados em autômatos parciais, um estado pode ser decomposto em várias máquinas de estados concorrentes. Assim como em *statecharts*, a sincronização entre máquinas de estados concorrentes pode ser obtida através de condições.

A UML 2 define pontos de entrada e de saída (*Entry Point* e *Exit Point*) como uma extensão dos *statecharts* que permitem uma hierarquização adicional de estados. Um ponto de entrada é um pseudoestado externamente visível imediatamente associado a um estado interno. Um ponto de saída é um pseudoestado externamente visível tendo como origem um estado interno. Um superestado dentro de uma máquina de estados pode ter um número arbitrário de pontos de entrada e de saída, que podem ser identificados por um nome [Rumbaugh et al. 2005].

A figura 6-18 apresenta um diagrama de estados UML possuindo dois pontos de entrada explicitamente especificados ("inserir novo destino" e "último destino") e um ponto de saída ("navegação concluída") além dos elementos de modelagem apresentados na seção 6.7.1.

Encapsulamento de estados internos usando pontos de entrada e pontos de saída

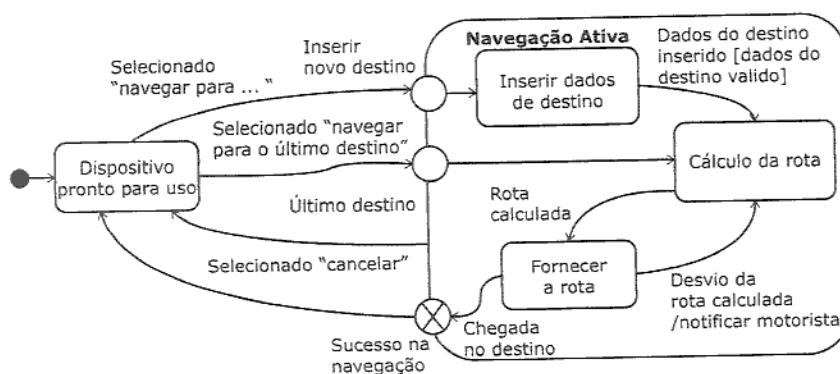


Figura 6-18 Diagrama de estados na notação UML 2

O diagrama de estados na figura 6-18 documenta o comportamento reativo de um dispositivo de navegação. Inicialmente o sistema encontra-se no estado "dispositivo pronto para uso". Ao selecionar "navegar para...", o sistema muda para o superestado "navegação ativa" e, dentro do superestado, para o subestado "inserir dados de destino" através do ponto de entrada "inserir novo destino". Alternativamente, o sistema passa do estado "dispositivo pronto para uso" para o estado interno "cálculo da rota" do superestado "navegação ativa" através do ponto de entrada "último destino" assim que ocorrer o evento "navegar para último destino". Após o estado "navegação ativa: inserir dados de destino", o sistema transita para o estado "navegação ativa: cálculo de rota", desde que os dados de destino sejam válidos.

Depois de calculada a rota no estado "navegação ativa: calcular a rota", o

sistema transita para o estado "navegação ativa: informar rota". Se um desvio da rota for detectado (evento: "desvio da rota calculada") no estado "navegação ativa: cálculo de rota", o condutor é notificado (atividade: "notificar condutor"). O sistema transita do estado "navegação ativa" para o estado "dispositivo pronto para uso" quando ocorrer o evento "cancelar". Se o sistema está no estado "navegação ativa: cálculo da rota" e o destino é alcançado, o sistema sai do superestado "navegação ativa" pelo ponto de saída "navegação concluída", transitando para o estado "dispositivo pronto para uso".

## 6.8 Resumo

Além da linguagem natural, requisitos podem ser documentados por meio de modelos. Requisitos em linguagem natural e modelos de requisitos são tipicamente empregados em conjunto, o que permite explorar as vantagens das duas formas de documentação.

A documentação de requisitos com base em modelos apresenta, entre outras vantagens, o fato que descrições gráficas (imagens) podem ser compreendidas de maneira mais rápida e melhor do que descrições em linguagem natural. Entre os modelos frequentemente usados na engenharia de requisitos encontram-se os modelos de metas (sob forma de árvores E/OU) e diagramas de casos de uso, bem como modelos conceituais para documentar requisitos a partir de três perspectivas: estrutural, funcional e comportamental. Para cada uma dessas três perspectivas existem linguagens de modelagem conceitual apropriadas, dotadas de meios específicos para documentar as informações apresentadas, conforme a finalidade de cada uma, respectivamente.

## 7 Validar e Negociar Requisitos

Validar e acordar requisitos durante a engenharia de requisitos têm o propósito de assegurar que os requisitos documentados atendam os critérios de qualidade previamente determinados, como exatidão e acordo (ver seção 4.6). As técnicas e princípios apresentados podem ser utilizados para validar e acordar requisitos individuais ou documentos completos de requisitos.

### 7.1 Fundamentos da Validação de Requisitos

Durante a engenharia de requisitos é necessário revisar a qualidade dos requisitos desenvolvidos. Entre outros objetivos, os requisitos são apresentados aos *stakeholders* com o propósito de identificar desvios entre os requisitos definidos e os desejos e necessidades reais dos *stakeholders*.

Durante a validação dos requisitos, a decisão se um requisito possui o nível necessário de qualidade (ver capítulo 4) é tomada, e se os requisitos podem ser aprovados para uso nas demais atividades de desenvolvimento (tais como projeto, implementação e teste). Essa decisão deve ser tomada com base em critérios de aceitação previamente definidos.

Assim, o objetivo da validação de requisitos é descobrir erros nos requisitos documentados. Exemplos típicos de erros em requisitos são ambiguidade, incompletude e contradições (ver seção 7.3).

Documentos de requisitos são documentos de referência para todas as demais atividades de desenvolvimento. Consequentemente, erros afetam negativamente todas as atividades posteriores de desenvolvimento. Um erro de requisito descoberto quando o sistema já está implementado e operando exige a revisão de todos os artefatos afetados pelo erro, tais como código fonte, artefatos de teste e descrições de arquitetura. Portanto, a correção de erros nos requisitos quando o sistema já está em operação implica custos significativos.

Um contrato entre cliente e contratado baseia-se frequentemente em documentos de requisitos. Erros críticos em requisitos podem levar ao não cumprimento de acordos contratuais, como por exemplo, o escopo de suprimento e serviços, a qualidade esperada ou os prazos de conclusão.

Requisitos devem ser aprovados

Objetivo da validação

Proliferação de erros

Riscos legais

## 7.2 Fundamentos da Negociação de Requisitos

Se não houver consenso entre os *stakeholders* a respeito dos requisitos, e se consequentemente os requisitos não puderem ser implementados coletivamente no sistema, cria-se um conflito não somente entre os requisitos contraditórios, mas também entre os *stakeholders* que exigem requisitos contraditórios. Por exemplo, um *stakeholder* poderia exigir o desligamento do sistema em caso de falha, enquanto outro *stakeholder* poderia exigir a reinicialização do sistema.

A aceitação do sistema é ameaçada por conflitos não resolvidos, pois tais conflitos fazem com que os requisitos de pelo menos um grupo de *stakeholders* não possam ser implementados. No pior cenário, um conflito pode causar a retirada de apoio por parte do *stakeholder*, levando ao fracasso do projeto de desenvolvimento (ver [Easterbrook 1994]). Além de representar riscos, conflitos podem também servir de oportunidade para a engenharia de requisitos, pois conflitos entre *stakeholders* exigem uma solução que tem o potencial de ajudar a revelar novas ideias e apresentar diferentes opções para o desenvolvimento (ver [Gause e Weinberg 1989]). Portanto, lidar abertamente com o conflito e procurar resolvê-lo durante a engenharia de requisitos pode aumentar a aceitação.

O objetivo da negociação é chegar a uma compreensão comum e acordada dos requisitos do sistema a ser desenvolvido entre todos os *stakeholders*.

A validação e negociação de requisitos é uma atividade a ser realizada (em graus variados de intensidade) ao longo de todo o processo de engenharia de requisitos. Portanto, a validação e negociação de requisitos gera trabalho adicional e, consequentemente, custos adicionais. Entretanto, as vantagens obtidas pela validação e negociação de requisitos conforme descritas acima (redução do custo global, aumento de aceitação, estímulo para soluções criativas e inovações) são em geral significativamente maiores do que os custos gerados pelo esforço adicional.

Requisitos contraditórios causam conflitos

Riscos e oportunidades dos conflitos

Objetivo da negociação de requisitos

Redução de custos e riscos em fases posteriores

## 7.3 Aspectos de Qualidade dos Requisitos

Uma das principais metas da utilização de critérios de qualidade (como por exemplo, completude, inteligibilidade, acordo) na validação de requisitos é a possibilidade da verificação sistemática de requisitos (ver seção 1.1.2). Para assegurar uma validação objetiva e consistente é necessário que cada critério de qualidade seja substanciado e detalhado. Em conformidade com os objetivos globais do processo de engenharia de requisitos, a validação é realizada com os

seguintes objetivos:

- *Conteúdo*: Todos os requisitos foram elicitados e documentados com o nível apropriado de detalhamento?
- *Documentação*: Todos os requisitos foram documentados em conformidade com as diretrizes de documentação e especificação previamente determinadas?
- *Acordo*: Todos os *stakeholders* concordam com os requisitos documentados e todos os conflitos conhecidos foram resolvidos?

Cada um dos três objetivos implica uma abordagem individual que enfoca aspectos específicos de qualidade dos requisitos. Os seguintes aspectos de qualidade foram portanto definidos:

- Aspecto de qualidade "conteúdo".
- Aspecto de qualidade "documentação".
- Aspecto de qualidade "acordo".

Três aspectos de qualidade

Um requisito deve ser aprovado para as atividades posteriores de desenvolvimento apenas se todos os três aspectos de qualidade tiverem sido verificados. Os aspectos de qualidade serão descritos em detalhe nas seções seguintes e substancialmente com diversos critérios de qualidade para aumentar a sutileza de sua diferenciação (sem que a descrição, no entanto, tenha a pretensão de ser completa).

### 7.3.1 Aspecto de Qualidade "Conteúdo"

O aspecto de qualidade "conteúdo" refere-se à verificação de erros de conteúdo nos requisitos. Erros de conteúdo em requisitos influenciam negativamente as atividades subsequentes de desenvolvimento, fazendo com que essas atividades sejam baseadas em informações que apresentam falhas.

Erros de conteúdo em requisitos ocorrem quando critérios específicos de qualidade para requisitos (ver seção 4.6) ou para documentos de requisitos (ver seção 4.5) são violados. A validação de requisitos com respeito ao aspecto de qualidade "conteúdo" será considerada bem-sucedida uma vez aplicada nos seguintes tipos de erros sem que falhas significativas tenham sido detectadas:

- *Completude (conjunto de todos os requisitos)*: Todos os requisitos relevantes para o sistema a ser desenvolvido (para o próximo *release* do sistema) foram documentados?
- *Completude (requisitos individuais)*: Cada requisito contém todas as

Critérios de teste do aspecto de qualidade "conteúdo"

informações necessárias?

- *Rastreabilidade:* Todos os relacionamentos relevantes de rastreabilidade (por exemplo, para fontes relevantes de requisitos) foram definidos?
- *Exatidão/adequação:* Os requisitos refletem acuradamente os desejos e necessidades dos *stakeholders*?
- *Consistência:* É possível implementar todos os requisitos definidos para o sistema conjuntamente? Não há contradições?
- *Nenhuma decisão de design prematura:* Existem decisões antecipadas de design presentes nos requisitos que não sejam decorrentes de restrições (por exemplo, restrições que especificam uma determinada arquitetura cliente-servidor a ser usada)?
- *Verificabilidade:* É possível definir critérios de aceitação e teste com base nos requisitos? Os critérios foram definidos?
- *Necessidade:* Cada requisito contribui para o cumprimento dos objetivos propostos?

### 7.3.2 Aspecto de Qualidade "Documentação"

O aspecto de qualidade "Documentação" abrange a verificação de falhas na documentação dos requisitos ou a verificação de violações das diretrizes de documentação em vigor, tais como a inteligibilidade dos formatos de documentação, o cumprimento de diretrizes organizacionais ou diretrizes específicas do projeto relacionadas com a documentação de requisitos, bem como a estrutura dos documentos de requisitos.

Ignorar as diretrizes de documentação pode levar aos seguintes riscos, entre outros:

- *Comprometimento das atividades de desenvolvimento:* Pode tornar-se inviável realizar atividades de desenvolvimento baseadas em determinado formato de documentação.
- *Não compreensão dos requisitos:* Requisitos podem não ser compreendidos ou sua utilidade pode não ser compreendida pelas pessoas que precisam entendê-los.
- *Incompletude:* Informações relevantes não estão documentadas nos requisitos.
- *Requisitos ignorados:* Se os requisitos não estão documentados no ponto onde deveriam estar nos documentos de requisitos, esses requisitos podem passar despercebidos em atividades posteriores.

Consequências da violação das regras de documentação

A validação de requisitos com respeito ao aspecto de qualidade "documentação" Critérios de

será considerada bem-sucedida uma vez aplicada nos seguintes tipos de erros sem que falhas significativas tenham sido detectadas:

- *Conformidade com o formato da documentação:* Os requisitos estão documentados nos formatos previamente determinados? Por exemplo, um template específico de requisitos ou uma linguagem específica de modelagem foram utilizados para documentar os requisitos?
- *Conformidade com a estrutura da documentação:* A estrutura da documentação foi mantida? Por exemplo, todos os requisitos foram documentados no ponto apropriado no documento?
- *Inteligibilidade:* Todos os requisitos documentados podem ser compreendidos no contexto dado? Por exemplo, todos os termos usados foram definidos em um glossário (ver seção 4.7)?
- *Não-ambiguidade:* A documentação dos requisitos permite uma única interpretação, ou múltiplas interpretações diferentes são possíveis? Por exemplo, um requisito redigido em linguagem natural não apresenta qualquer tipo de ambiguidade?
- *Conformidade com as regras de documentação:* As regras e diretrizes de documentação previamente determinadas foram cumpridas? Por exemplo, a sintaxe da linguagem de modelagem foi utilizada de forma correta?

teste do aspecto de qualidade "documentação"

### 7.3.3 Aspecto de Qualidade "Acordo"

O aspecto de qualidade "acordo" abrange a verificação de eventual falta de acordo entre os *stakeholders* a respeito dos requisitos.

Ao longo do processo de engenharia de requisitos os *stakeholders* adquirem maior conhecimento do sistema a ser desenvolvido. Devido a esse conhecimento adicional, a opinião dos *stakeholders* com respeito a um requisito já acordado pode mudar. Durante a validação de requisitos, os *stakeholders* têm a oportunidade de solicitar mudanças sem prejudicar as atividades subsequentes de desenvolvimento.

Última oportunidade para alterações

A validação de requisitos com respeito ao aspecto de qualidade "acordo" será considerada bem-sucedida uma vez aplicada nos seguintes tipos de erros sem que falhas significativas tenham sido detectadas:

- *Acordado:* Todos os *stakeholders* relevantes estão de acordo com cada requisito?
- *Acordado após alteração:* Todos os *stakeholders* estão de acordo com cada requisito após o mesmo ter sido alterado?
- *Conflitos resolvidos:* Todos os conflitos conhecidos com respeito a

Três critérios de teste do aspecto de qualidade "acordo"

requisitos foram resolvidos?

## 7.4 Princípios de Validação de Requisitos

Levar em consideração os seguintes princípios de validação de requisitos melhora a qualidade dos resultados da validação:

- *Princípio 1:* Envolvimento dos *stakeholders* corretos.
- *Princípio 2:* Separação entre a identificação de falhas e a correção de erros.
- *Princípio 3:* Validação a partir de diferentes pontos de vista.
- *Princípio 4:* Mudança adequada do tipo de documentação.
- *Princípio 5:* Construção de artefatos de desenvolvimento.
- *Princípio 6:* Revalidação de requisitos.

Os princípios individuais serão explicados nas seções abaixo:

### 7.4.1 Princípio 1: Envolvimento dos Stakeholders Corretos

A escolha de *stakeholders* para a validação de requisitos depende dos objetivos da validação bem como dos requisitos a serem auditados.

Ao reunir a equipe de auditoria, no mínimo os seguintes aspectos deveriam ser levados em consideração:

Em termos gerais, recomenda-se evitar que o autor de um requisitos também seja a pessoa a validar o requisito. O autor utilizará seu conhecimento anterior ao ler ou revisar o requisito. Esse conhecimento prévio pode influenciar negativamente a identificação de erros, pois trechos que eventualmente apresentam falhas na documentação de requisitos ou nos próprios requisitos são subconscientemente retificados pelo conhecimento do próprio autor, podendo assim facilmente passar despercebidos.

Independência  
do autor

Auditores adequados podem ser identificados tanto dentro quanto fora da organização de desenvolvimento. Auditorias internas são realizadas por *stakeholders* que são membros da organização de desenvolvimento e podem ser utilizados para validar resultados intermediários ou requisitos preliminares. A coordenação e organização de uma validação interna é facilitada pela disponibilidade dos *stakeholders* dentro da organização. Uma auditoria externa requer um nível mais alto de esforço, pois é preciso identificar auditores e (eventualmente) contratá-los. Além disso, auditores externos precisam familiarizar-se com o contexto do sistema a ser desenvolvido. Devido ao maior esforço, uma

Auditores  
internos vs.  
externos

auditoria externa somente deveria ser realizada em requisitos que apresentam um alto nível de qualidade.

#### 7.4.2 Princípio 2: Separação Entre a Identificação de Falhas e a Correção de Erros

A separação entre a identificação de falhas e a correção de erros propriamente ditos revelou-se proveitosa no domínio da garantia de qualidade de software. O mesmo princípio pode ser aplicado na validação de requisitos. Durante a validação, as falhas identificadas são imediatamente documentadas. Mais tarde, cada falha identificada é novamente verificada para determinar se ela é de fato um erro.

Separar a identificação de erros da correção de erros permite que os auditores se concentrem na identificação. Medidas para corrigir os erros somente serão tomadas depois que as medidas de identificação tiverem sido concluídas. Isso traz as seguintes vantagens: os recursos disponíveis para a correção de erros podem ser utilizados de forma eficaz; a identificação prematura de erros não causa erros adicionais; nenhum erro (ou suposto erro) é corrigido sem que haja necessidade para tal, pois investigações posteriores do erro podem muito bem constatar que um suposto erro não é de fato um erro. Dessa forma, diminui a probabilidade de erros significativos (que sempre podem estar presentes) passarem despercebidos pelo fato de o auditor estar concentrado em corrigir um erro anterior em vez de identificar novos erros.

Princípio básico

Concentração na identificação de erros

#### 7.4.3 Princípio 3: Validação a Partir de Diferentes Pontos de Vista

Validar requisitos a partir de diferentes pontos de vista é outro princípio que mostrou ser proveitoso. Nesse princípio, os requisitos são validados e acordados a partir de diferentes perspectivas (por exemplo, por diferentes pessoas, ver seção 7.5.4). Métodos comparáveis são utilizados em outras disciplinas também. Por exemplo, em julgamentos as circunstâncias são muitas vezes relatadas a partir da perspectiva de diferentes pessoas para obter uma imagem global mais completa.

Validação baseada em perspectiva

#### 7.4.4 Princípio 4: Mudança Adequada do Tipo de Documentação

Trocar o tipo de documentação durante a validação de requisitos utiliza os pontos fortes de um tipo de documentação para compensar os pontos fracos de outros tipos

Pontos fortes e pontos fracos

de documentação. Por exemplo, boa inteligibilidade e expressividade são pontos fortes de textos redigidos em linguagem natural. No entanto, seus pontos fracos são o potencial de ambiguidade e a dificuldade de expressar circunstâncias complexas. Modelos gráficos conseguem representar circunstâncias complexas muito bem, mas os elementos individuais de modelagem são restritos em termos de expressividade.

Transcrever um requisito já documentado em outro formato de documentação simplifica a identificação de erros. Por exemplo, ambiguidades em requisitos em linguagem natural podem ser identificadas muito mais facilmente transcrevendo os requisitos para uma representação baseada em modelos.

de diversos tipos de documentação

Identificação simplificada de erros

Adequação dos requisitos para a criação de artefatos de projeto, de teste e de documentação do usuário

#### 7.4.5 Princípio 5: Construção de Artefatos de Desenvolvimento

O objetivo da construção de artefatos de desenvolvimento é validar a qualidade dos requisitos que têm por finalidade servir de base para a criação de artefatos do projeto, artefatos de teste ou o manual do usuário. Ao longo da validação, as atividades geralmente realizadas em fases subsequentes para construir os respectivos artefatos de desenvolvimento são executadas com pequenas amostras. Por exemplo, o auditor trabalha intensivamente com um requisito criando um caso de teste. Dessa forma, erros (por exemplo, ambiguidades) podem ser identificados no requisito. Esse tipo de validação, todavia, exige muitos recursos, pois atividades subsequentes de desenvolvimento devem ser executadas pelos menos parcialmente.

#### 7.4.6 Princípio 6: Revalidação de Requisitos

A validação ocorre em um momento específico durante o processo de desenvolvimento, apoiando-se no nível de conhecimento dos auditores naquele momento. Durante a engenharia de requisitos os *stakeholders* adquirem conhecimentos adicionais sobre o sistema em planejamento. Consequentemente, uma validação positiva dos requisitos não garante que os requisitos ainda continuem válidos em um momento posterior. A validação de requisitos deveria ser realizada várias vezes nos seguintes casos (entre outros):

- Grande quantidade de ideias e tecnologias inovadoras usadas no sistema.
- Acréscimo significativo de conhecimento durante a engenharia de requisitos.
- Projetos de longa duração.
- Validação de requisitos realizada muito cedo.
- Domínio desconhecido.
- Reutilização de requisitos.

## 7.5 Técnicas de Validação de Requisitos

Nas seções seguintes, serão apresentadas diversas técnicas para validação de requisitos. Técnicas de validação de manuais, também conhecidas pelo termo genérico *revisão*, são frequentemente utilizadas para a validação de requisitos. Três principais tipos de revisão podem ser diferenciados:

- Parecer de especialista (*Commenting*).
- Inspeção.
- *Walkthrough*.

Além das revisões, as seguintes três técnicas demonstraram sua utilidade para a validação de requisitos:

- Leitura baseada em perspectiva.
- Validação por protótipos.
- Utilização de *checklists*.

As seis técnicas serão descritas abaixo. Antes de aplicar qualquer uma dessas técnicas, devem ser tomadas as medidas de preparação necessárias, tais como identificar e convidar os *stakeholders* corretos, ou organizar espaços e suprimentos.

### 7.5.1 Parecer de Especialista (*Commenting*)

Na técnica de validação "parecer de especialista" (*commenting*), o autor entrega seus requisitos para outra pessoa (por exemplo, um colega). O objetivo é obter do colega uma opinião de especialista sobre a qualidade de um requisito. O colega revisa o requisito com o objetivo de identificar problemas que prejudicam a qualidade do requisito (por exemplo, ambiguidade ou erros), conforme critérios de qualidade previamente determinados. As falhas identificadas são marcadas no documento de requisitos e brevemente comentadas.

Validação individual dos requisitos

### 7.5.2 Inspeção

Inspeções de software, ou de qualquer outro produto, são realizadas para verificar sistematicamente a presença de erros em artefatos de desenvolvimento através da

Etapas típicas de uma

aplicação de um processo rigoroso [Laitenberger e DeBaud 2000].

inspeção

Uma inspeção é tipicamente subdividida em diversas etapas [Gilb e Graham 1993]: Planejamento, visão geral, detecção de defeitos, correção de defeitos, *follow-up* e reflexão. As etapas de planejamento, visão geral, detecção de falhas e coleta de falhas são relevantes para a validação de requisitos (ver Princípio 2: Separação entre identificação de falhas e correção de erros, na seção 7.4.2). A preparação individual é parte obrigatória das inspeções. Uma sessão de inspeção geralmente tem por finalidade coletar e avaliar indicativos de erros. Ocasionalmente, sessões exclusivamente dedicadas à inspeção não são realizadas durante o processo de inspeção.

Na etapa de planejamento são determinados, entre outros aspectos, o objetivo da inspeção, os resultados a serem inspecionados, bem como os papéis e os participantes.

Planejamento

Na etapa da visão geral, o autor explica os requisitos a serem inspecionados para todos os membros da equipe, de modo a criar uma compreensão comum sobre os requisitos entre todos os inspetores.

Visão Geral

Na fase de detecção de falhas, os inspetores buscam falhas nos requisitos. A detecção de falhas pode ser realizada individualmente por cada inspetor ou pode ser realizada de forma conjunta pela equipe. A inspeção individual apresenta a vantagem de que cada inspetor pode concentrar-se nos requisitos. Por outro lado, inspeções em equipe apresentam a vantagem de que a comunicação entre os inspetores cria efeitos de sinergia durante a detecção de falhas. Quaisquer falhas encontradas ao longo da etapa de detecção de falhas devem ser documentadas sistematicamente.

Detecção de falhas

Na fase de coleta de falhas, todas as falhas identificadas são coletadas, consolidadas e documentadas. Durante a consolidação, são identificadas falhas que foram detectadas várias vezes, bem como falhas que não são de fato erros. Este último caso pode ocorrer, por exemplo, se um inspetor fizer suposições erradas sobre um requisito, ou não interpretar corretamente alguma restrição. Além disso, na consolidação também documentam-se os erros identificados e as respectivas medidas corretivas em uma lista de erros. Inspeções também são conhecidas como *revisões técnicas*.

Coleta e consolidação de falhas

Para que se possa realizar uma inspeção, é necessário designar pessoal adequado para os seguintes papéis:

Papéis durante a inspeção

- *Organizador*: O organizador planeja e supervisiona o processo de inspeção.
- *Moderador*: O moderador lidera a sessão e assegura o seguimento do processo de inspeção previamente definido. É recomendável que um moderador neutro seja selecionado, pois ele poderá eventualmente ter que mediar entre opiniões divergentes de autores e inspetores.
- *Autor*: O autor explica para os inspetores os requisitos que criou na etapa da visão geral, sendo mais tarde responsável pela correção dos erros

identificados.

- *Leitor:* O leitor apresenta sucessivamente os requisitos a serem inspecionados e orienta os inspetores em meio aos requisitos. O papel do leitor deveria ser dado a um *stakeholder* neutro, para que os inspetores possam concentrar sua atenção nos requisitos e não na interpretação do autor. O moderador muitas vezes também é o leitor.
- *Inspetores:* Os inspetores são responsáveis pela localização de falhas e pela comunicação de suas constatações aos outros membros da equipe do projeto.
- *Secretário:* O secretário prepara a ata da sessão, onde são registrados os resultados da inspeção.

### 7.5.3 Walkthrough

Na validação de requisitos, um *walkthrough* é uma versão "light" da revisão. Um *walkthrough* é menos severo do que uma inspeção e os papéis que o processo envolve são menos diferenciados entre si. Durante um *walkthrough*, os papéis mínimos a serem ocupados são o revisor (comparável ao inspetor), o autor e o secretário, além do papel do moderador, eventualmente.

Um *walkthrough* de requisitos tem por objetivo identificar falhas de qualidade nos requisitos por meio de um processo compartilhado, e proporcionar um conhecimento compartilhado dos requisitos entre todas as pessoas envolvidas. Como preparativo para um *walkthrough*, os requisitos a serem validados são distribuídos para todos os participantes e inspecionados. Durante a sessão de *walkthrough*, os participantes discutem os requisitos a serem validados passo-a-passo, sob orientação do moderador/leitor. O autor de um requisito é quem geralmente apresenta o requisito aos outros participantes. Assim, os autores têm a oportunidade de acrescentar informações adicionais para o grupo junto com o requisito propriamente dito (por exemplo, requisitos alternativos, decisões tomadas e a justificativa para essas decisões). Um secretário documenta as falhas de qualidade identificadas durante a sessão.

Revisão  
"light"

Discussão das  
falhas de  
qualidade  
identificadas  
em uma sessão  
de grupo

### 7.5.4 Leitura Baseada em Perspectiva

A leitura baseada em perspectiva é uma técnica de validação de requisitos na qual os requisitos são verificados a partir de diferentes perspectivas [Basilí et al. 1996]. A leitura baseada em perspectiva é tipicamente aplicada em conjunto com outras técnicas de revisão (por exemplo, durante inspeções ou *walkthroughs*). Está comprovado que enfocar perspectivas específicas ao ler um documento traz melhores resultados na validação de requisitos. Possíveis perspectivas para a validação podem ser obtidas, por exemplo, a partir dos diversos públicos de um

Verificar  
requisitos a  
partir de uma  
perspectiva  
definida

requisito [Shull et al. 2000]:

- *Perspectiva do usuário/cliente:* Os requisitos são verificados a partir da perspectiva do cliente ou do usuário, para determinar se os requisitos descrevem as funções e qualidades desejadas do sistema.
- *Perspectiva do arquiteto de software:* Os requisitos são verificados a partir da perspectiva do arquiteto de software, para confirmar se todas as informações necessárias para o projeto arquitetônico (por exemplo, se todas as propriedades relevantes foram descritas) estão incluídas nos requisitos.
- *Perspectiva do testador:* Os requisitos são verificados a partir da perspectiva do testador, para determinar se contêm as informações necessárias para gerar casos de teste a partir dos requisitos.

Os três aspectos de qualidade (ver seção 7.3) também apresentam três possíveis perspectivas para a validação de requisitos:

- *Perspectiva do conteúdo:* Na perspectiva do conteúdo, o auditor verifica o conteúdo dos requisitos, concentrando-se na qualidade do conteúdo do requisito documentado.
- *Perspectiva da documentação:* Na perspectiva da documentação, o auditor certifica-se de que todas as diretrizes de documentação para requisitos e para documentos de requisitos foram seguidas.
- *Perspectiva do acordo:* Na perspectiva do acordo, o auditor verifica se todos os *stakeholders* estão de acordo com um requisito, isto é, se os requisitos estão acordados e os conflitos foram resolvidos.

Além disso, perspectivas adicionais que surgem a partir do contexto específico do projeto de desenvolvimento podem ser criadas conforme necessário.

Durante a validação baseada em perspectiva, para cada auditor é atribuída uma perspectiva (no momento apropriado), a partir da qual o auditor deve ler e validar o requisito. Instruções detalhadas para realizar a validação devem ser estabelecidas para cada perspectiva, pois o auditor pode não estar familiarizado com todos os detalhes relevantes da perspectiva que lhe foi atribuída. Recomenda-se associar perguntas a cada instrução de validação, a serem respondidas pelo conteúdo dos requisitos, ou pelo auditor depois de ler os requisitos, respectivamente. Além disso, as instruções de validação podem ser acompanhadas por uma lista de verificação apresentando de forma resumida os aspectos de conteúdo mais importantes que deveriam ser abordados por um requisito dentro da respectiva perspectiva.

Durante o acompanhamento (*follow-up*) de uma sessão de leitura baseada em perspectiva, os resultados de cada perspectiva são coletados e consolidados. Por um lado, os resultados da leitura baseada em perspectiva contêm as respostas para as perguntas previamente formuladas; por outro, podem também haver questões

Perspectivas de validação a partir dos aspectos de qualidade dos requisitos

Definir diretrizes de validação para cada perspectiva

Follow-up

ainda em aberto que os auditores perceberam ao ler os requisitos. A consolidação pode tomar a forma de uma sessão em grupo, semelhante à revisão.

A leitura baseada em perspectiva pode ser tanto uma técnica independente para a validação de requisitos quanto uma técnica de apoio para outras técnicas de validação, tais como inspeções ou revisões de documentos de requisitos através da leitura baseada em perspectiva.

Técnica de apoio para outras técnicas

### 7.5.5 Validação por Protótipos

A validação de requisitos por protótipos permite que auditores experimentem os requisitos testando-os na prática. Experimentar requisitos diretamente através de protótipos [Jones 1998] é o método mais eficaz para identificar erros em requisitos. Os *stakeholders* podem testar o protótipo e comparar sua própria ideia sobre como o sistema deveria ser implementado com o protótipo em suas mãos, identificando dessa forma eventuais discrepâncias entre suas ideias e a implementação atual.

Dependendo do uso posterior do protótipo, distingue-se entre protótipos descartáveis e protótipos evolutivos [Sommerville 2007]. Protótipos descartáveis não são submetidos a manutenção após terem sido utilizados. Protótipos evolutivos são criados com o objetivo de serem desenvolvidos adicionalmente e aprimorados em etapas posteriores. Ao contrário dos protótipos descartáveis, a implementação desempenha um papel muito mais significativo neste caso. Consequentemente, o esforço necessário para criar protótipos evolutivos é muito maior.

Protótipos evolutivos vs. protótipos descartáveis

Antes que um protótipo possa ser implementado, os requisitos a serem validados pelo protótipo devem ser selecionados. O conjunto de requisitos a serem validados é limitado pelos recursos de desenvolvimento (por exemplo, tempo, verbas, etc.) que podem ser alocados para a validação. Um critério de seleção, por exemplo, pode ser a criticalidade de um requisito.

Seleção de requisitos relevantes

Os seguintes preparativos devem ser efetuados para validar requisitos por meio de protótipos:

- *Manual/instruções:* Os usuários do protótipo devem receber as informações necessárias para que possam usar ou aplicar o protótipo. Isso pode ser feito através de um manual ou por meio de instruções apropriadas.
- *Cenários de validação:* Cenários de validação que os usuários do protótipo possam executar com o protótipo devem ser preparados. Um cenário de validação define, por exemplo, todos os conjuntos de dados relevantes ou

Preparativos para a validação

interações com usuários.

- **Lista de verificação com critérios de validação:** Uma lista de verificação deve ser criada para a validação de requisitos, contendo critérios de validação com os quais o protótipo (e por extensão os requisitos) possam ser validados.

O auditor deve validar o protótipo sem ser influenciado, isto é, o auditor deve executar os cenários de validação de forma independente e sem acompanhamento, assegurando, assim, a imparcialidade dos resultados da validação.

Executar a validação

Durante a validação, os auditores podem (e devem) executar cenários alternativos e divergentes, bem como utilizar o protótipo de forma exploratória e experimental depois que os cenários de validação obrigatórios já tenham sido executados. Por exemplo, casos de erros que tenham permanecido ocultos até então podem ser identificados. Para a validação experimental do protótipo, o auditor precisa conhecer o escopo do protótipo, isto é, o conjunto de requisitos que foram considerados quando o protótipo foi criado. Sem conhecimento dos requisitos implementados, um auditor não pode decidir se um erro identificado pode ser rastreado para a falta de um requisito ou se o requisito foi conscientemente omitido do protótipo.

A validação de requisitos por protótipos, portanto, permite dois tipos de documentação de resultados:

Documentação dos resultados da validação

- **Protocolo do auditor:** O auditor documenta os resultados e experiências feitas durante a validação do protótipo, através de cenários de validação, por exemplo, bem como de uma lista de verificação que lhe tenha sido fornecida.
- **Protocolo de observação:** O auditor pode ser observado por uma segunda pessoa. A segunda pessoa cria um chamado protocolo de observação. Esse protocolo pode revelar sintomas adicionais importantes de erros nos requisitos. Por exemplo, se o auditor vier a hesitar em determinado passo no cenário de validação ao usar o protótipo e o observador documentar essa indecisão, isso pode ser sinal de falta de clareza, e como tal, uma indicação de menor comprehensibilidade do protótipo. Em certas circunstâncias, pode ser recomendável registrar a validação em vídeo para que a situação de validação possa ser analisada detalhadamente no acompanhamento (*follow-up*). Uma gravação de vídeo pode mostrar, por exemplo, a realização de requisitos relacionados com propriedades antropométricas (tais como aspectos ergonômicos) ou a utilização intuitiva, e pode ser investigada em maiores detalhes.

Os resultados da validação são analisados após a conclusão da validação. Sugestões de mudanças de requisitos são consolidadas. Caso mudanças significativas dos requisitos sejam necessárias, é recomendável revisar o protótipo e refazer a

Análise dos resultados da

validação.

validação

### 7.5.6 Utilização de *Checklist* para a Validação

Uma *checklist* (ou lista de verificação) é um conjunto de perguntas e/ou afirmações sobre determinada circunstância. *Checklists* podem ser aplicadas sempre que muitos aspectos precisam ser considerados em um ambiente complexo e que nenhum aspecto possa ser omitido. Uma lista de verificação para a validação de requisitos contém perguntas que facilitam a identificação de erros [Boehm 1984]. O uso de *checklists* para a validação de requisitos é muito comum na prática. *Checklists* podem ser utilizadas em todas as técnicas para validação de requisitos previamente apresentadas.

Antes que uma lista de verificação possa ser utilizada, cada pergunta ou afirmação deverá ser individualmente definida. As fontes para perguntas ou afirmações na lista abaixo podem ser utilizadas para criar *checklists* como material de apoio para a validação de requisitos:

- Os três aspectos de qualidade dos requisitos (ver seção 7.3).
- Princípios de validação de requisitos (ver seção 7.4).
- Critérios de qualidade para documentos de requisitos (ver seção 4.5).
- Critérios de qualidade para requisitos individuais (ver seção 4.6).
- Experiências dos auditores em projetos anteriores.
- Estatísticas de erros [Chernak 1996].

Criação de  
*checklists*

*Checklists* não são necessariamente completas. Ao usar uma lista de verificação, sempre deve-se procurar oportunidades para aprimorar a lista de verificação para uso futuro. Por exemplo, se uma pergunta foi esquecida, a lista de verificação deve ser aumentada para incluir a pergunta adicional. Perguntas ambíguas, ou perguntas que não são compreensíveis, devem ser marcadas e revisadas. Perguntas desatualizadas e não mais válidas devem ser removidas.

Aprimorar as  
*checklists*

*Checklists* podem oferecer respaldo à validação de requisitos de diversas maneiras. Elas podem servir de orientação para o auditor, que pode seguir as *checklists* a seu próprio critério (por exemplo, durante uma revisão).

*Checklists*  
como  
orientação

A lista de verificação pode especificar uma lista de perguntas a ser estritamente seguida. Essas perguntas devem obrigatoriamente ser respondidas pelo auditor para validar os requisitos. Nesse caso, a lista de verificação serve como um meio para abordar a validação de forma estruturada. Por exemplo, a lista de

*Checklists*  
como meio de  
estruturação

verificação pode detalhar o processo exato que os auditores deverão aplicar, o que garante que cada auditor validará os requisitos da mesma maneira. Isso torna os resultados mais comparáveis entre si.

Formas híbridas de aplicação de *checklists* também são possíveis. Por exemplo, uma lista de verificação pode conter perguntas obrigatórias para a leitura baseada em perspectiva, mas pode também conter sugestões que o auditor decide seguir ou não.

Aplicar *checklists* para a validação de requisitos de maneira bem-sucedida depende da maleabilidade e complexidade da lista de verificação. Um grande número de perguntas pode dificultar o uso da lista de verificação, pois o auditor não possui uma visão aprofundada das perguntas, sendo assim forçado a consultar a lista de verificação frequentemente. Recomenda-se, portanto, elaborar a lista de verificação de tal forma que ela não seja mais longa do que uma página [Gilb e Graham 1993]. Além disso, perguntas formuladas de forma demasiadamente genérica ou abstrata podem dificultar o uso da lista de verificação. Por exemplo, a pergunta "O requisito está formulado adequadamente?" pode resultar em grande número de respostas diferentes, dependendo do que o auditor considera um requisito adequadamente formulado. Sendo assim, as perguntas devem ser formuladas da forma mais precisa possível.

Aplicação  
bem-sucedida  
das *checklists*

## 7.6 Negociar Requisitos

Para alcançar um acordo sobre os requisitos de um sistema a ser desenvolvido, é necessário identificar eventuais conflitos e resolver esses conflitos. Isso é feito por meio do gerenciamento sistemático de conflitos. O gerenciamento de conflitos na engenharia de requisitos abrange as seguintes quatro tarefas:

- Identificação de conflitos.
- Análise de conflitos.
- Resolução de conflitos.
- Documentação da resolução de conflitos.

Quatro tarefas  
do  
gerenciamento  
de conflitos

Essas quatro tarefas serão explicadas nas seções abaixo:

### 7.6.1 Identificação de Conflitos

Conflitos podem surgir ao longo de todas as atividades de engenharia de requisitos. Por exemplo, diferentes *stakeholders* podem gerar requisitos conflitantes durante a elicitação.

Por uma série de motivos, conflitos entre requisitos e conflitos entre Identificação

*stakeholders* nem sempre são evidentes. Ao longo do processo de engenharia de requisitos, o engenheiro de requisitos deve ficar atento para conflitos em potencial, para que possam ser identificados, analisados e resolvidos desde cedo.

de conflitos em todas as atividades de engenharia de requisitos

### 7.6.2 Análise de Conflitos

Durante a análise de conflitos, deve ser determinado o motivo por trás de um conflito que tenha sido identificado. Conforme [Moore 2003], existem diferentes tipos de conflitos.

Determinar o tipo de conflito

Um *conflito de conteúdo* entre dois ou mais *stakeholders* é caracterizado por uma falta de informações, por informações equivocadas ou por interpretações diferentes de alguma informação. Por exemplo, considere o seguinte requisito: "R131: O tempo de resposta do sistema planejado não deverá exceder um segundo". Um conflito de conteúdo poderá surgir entre dois *stakeholders* a respeito desse requisito se um deles considerar 1 segundo um tempo de resposta lento demais e o outro considerar inviável implementar um tempo de resposta de 1 segundo (isto é, o tempo é curto demais).

Conflito de conteúdo

Um *conflito de interesses* entre dois ou mais *stakeholders* é caracterizado por interesses ou objetivos subjetiva e objetivamente diferentes entre *stakeholders*. Um conflito de interesses entre dois ou mais *stakeholders* pode surgir, por exemplo, quando um *stakeholder* tem como foco principal manter os custos do sistema planejado no mínimo, enquanto outro *stakeholder* deseja principalmente um alto nível de qualidade. Um conflito de interesses entre esses dois *stakeholders* surge quando o primeiro rejeita um requisito com base em seu custo estimado, enquanto o segundo insiste em sua implementação por razões de qualidade.

Conflito de interesses

Um *conflito de valores* é caracterizado por divergências relacionadas a valores fundamentais entre os *stakeholders* (por exemplo, diferenças culturais, ideais pessoais, etc.) Por exemplo, pode surgir um conflito de valores quando um *stakeholder* é favorável a tecnologias de código aberto, enquanto outro *stakeholder* prefere tecnologias de código fechado.

Conflito de valores

Um *conflito de relacionamento* é caracterizado por fortes emoções, conceitos estereotipados de relacionamentos, comunicação deficiente ou conduta interpessoal negativa entre *stakeholders* (insultos, falta de respeito, etc). Por exemplo, um conflito de relacionamento surge quando dois *stakeholders* do mesmo nível ou posição (por exemplo, gerentes de departamento) rejeitam os requisitos da outra parte e procuram se valorizar impondo seus próprios requisitos para o projeto.

Conflito de relacionamentos

Um *conflito estrutural* é caracterizado por níveis distintos de autoridade ou poder. Por exemplo, um conflito estrutural pode surgir entre um subalterno e seu superior se este invariavelmente rejeita os requisitos definidos pelo subalterno, por

Conflito estrutural

não reconhecer a competência do subalterno para elaborar requisitos.

Com frequência, é difícil classificar de forma inequívoca os conflitos que eventualmente surgem. Por exemplo, determinado conflito pode ser um conflito de relacionamento com claros componentes estruturais. De forma semelhante, um conflito de interesse pode ser um conflito de valores também. É recomendável, portanto, uma vez identificado um conflito, analisá-lo a partir de todos os tipos de conflitos, possibilitando dessa forma a determinação de todos os possíveis motivos do conflito, bem como a seleção das estratégias de resolução mais apropriadas.

Múltiplos motivos para conflitos

### 7.6.3 Resolução de Conflitos

A resolução de conflitos é muito importante para a negociação de requisitos, pois a estratégia de resolução de conflitos tem forte influência na disposição das pessoas envolvidas (clientes, consultores, desenvolvedores) de continuarem a trabalhar juntos. Por exemplo, uma resolução de conflitos considerada injusta por pelo menos uma das partes do conflito pode levar a uma redução de comprometimento e disposição para colaborar com o projeto. Por outro lado, uma resolução considerada justa por todas as partes pode aumentar a disposição de cooperar, pois isso sinaliza que as ideias de todos a respeito do sistema planejado estão sendo consideradas.

Uma boa estratégia de resolução de conflitos é um fator de sucesso

Independente da estratégia escolhida, é essencial envolver todos os *stakeholders* relevantes. Se todos os *stakeholders* relevantes não forem considerados, algumas opiniões e pontos de vista continuarão não sendo considerados. O conflito, portanto, somente será solucionado em parte, ou de forma incompleta. Nos parágrafos abaixo, serão introduzidos diversas técnicas de resolução de conflitos.

Envolvimento de todos os stakeholders

Na técnica de resolução de conflitos "*Acordo*", todas as partes em conflito negociam uma solução para o conflito. As partes trocam informações, argumentos e opiniões entre si e procuram convencer as outras partes sobre seus próprios pontos de vista para chegar a uma solução de consenso.

Acordo

Na técnica de resolução de conflitos "*Compromisso*", todas as partes envolvidas no conflito procuram chegar a um compromisso entre soluções alternativas. Ao contrário de um acordo, um compromisso consiste em uma composição de diferentes partes das soluções alternativas. Adicionalmente, um compromisso pode significar que todas as soluções alternativas propostas até aquele momento são descartadas, com soluções completamente novas passando a ser criadas.

Compromisso

Na técnica de resolução de conflitos "*Votação*", todas as partes envolvidas no conflito votam sobre soluções alternativas. As alternativas a serem votadas são apresentadas para todos os *stakeholders* relevantes. Cada *stakeholder* dá seu voto para uma das alternativas e aquela com o maior número de votos é aceita como a resolução do conflito.

Votação

Na técnica de resolução de conflitos "*Análise de alternativas*" (*Definition of variants*), o sistema é desenvolvido de forma a permitir a definição de

Análise de

alternativas pela derivação de variantes, pela seleção de parâmetros que definem as variantes do sistema, ou pela seleção de propriedades variáveis de sistema. Dessa forma, o sistema pode atender aos diferentes interesses dos *stakeholders*.

alternativas

Na técnica de resolução de conflitos "*Manda quem pode*" (*overruling*), um conflito é resolvido pela organização hierárquica. Isso significa que a parte do conflito em posição hierárquica superior ganha o conflito ao rejeitar as objeções das partes organizacionais inferiores. Se ambas as partes ocupam o mesmo nível organizacional, o conflito será resolvido por um *stakeholders* superior, ou por alguma terceira parte com poder de decisão. Essa técnica de resolução de conflitos somente é recomendável se outras técnicas fracassaram (se, por exemplo, não foi possível chegar a um compromisso) ou se outras técnicas não se aplicam devido a limites de recursos (por exemplo, tempo).

"Manda quem pode"

Na técnica de resolução de conflitos "*Obter mais informações*" (*Consider all Facts, ou CAF*), todos os fatores influentes são investigados, de modo a coletar o máximo de informações possíveis sobre o conflito. Essa informação é utilizada durante a resolução. Ao priorizar os fatores influentes, determina-se sua respectiva relevância. Com base nos resultados dessa técnica, pode-se utilizar a técnica de resolução de conflitos "*Análise de repercussões*".

"Obter mais informações"

Na técnica de resolução de conflitos "*Análise de repercussões*" (*Plus-Minus-Interesting, ou PMI*), todas as repercussões positivas e negativas de uma determinada alternativa de solução são investigadas, permitindo assim sua avaliação. As repercussões positivas são colocadas na categoria "*Plus*", enquanto as repercussões negativas são colocadas na categoria "*Minus*". Replicações que não são positivas nem negativas são colocadas na categoria "*Interesting*". As repercussões na categoria "*Interesting*" não podem ser avaliadas antes de serem investigadas adicionalmente, para determinar se sua influência é positiva ou negativa.

Análise de repercussões

Na técnica de resolução de conflitos "*Matriz de decisão*" (*Decision Matrix*), cria-se uma tabela com as soluções alternativas nas colunas e todos os respectivos critérios de decisão nas linhas. Os critérios de decisão podem ser identificados com a técnica "*Obter mais informações*". Para cada cruzamento entre um critério e uma solução alternativa, é feita uma avaliação com uma pontuação, por exemplo, entre irrelevante (0 pontos) até altamente relevante (10 pontos). A tabela 7-1 apresenta uma matriz de decisão.

Matriz de decisão

	Solução Alternativa 1	Solução Alternativa 3	Solução Alternativa 3
Critério 1	3	6	2
Critério 2	5	4	10
Critério 3	10	3	5
Soma	18	13	17

Tabela 7-1 Matriz de decisão

Para encontrar uma solução, calculam-se os valores das respectivas colunas, isto é, as avaliações dos critérios de cada alternativa são somadas. A alternativa com a soma mais alta é considerada a decisão. No exemplo da tabela 7-1, a decisão seria a alternativa 1.

#### 7.6.4 Documentação da Resolução de Conflitos

Conflitos não podem ser evitados durante a engenharia de requisitos. Uma resolução de conflito deve sempre ser documentada de forma rastreável. Se uma resolução de conflito não for documentada de maneira apropriada, o projeto poderá correr os seguintes riscos (entre outros):

Riscos associados à documentação de conflitos

- *Enfrentar repetidamente os mesmos conflitos:* Determinado conflito pode surgir novamente durante a engenharia de requisitos. Sem documentação apropriada da resolução do conflito, ele precisará ser novamente analisado e resolvido. Isto causa trabalho adicional, além de potencialmente gerar mais conflitos, ou fazer com que as decisões previamente tomadas sejam revogadas.
- *Resolução inapropriada de conflitos:* Ao longo do processo de engenharia de requisitos, a resolução de conflitos pode revelar-se incorreta ou inadequada. Nesse caso, o conflito deverá ser novamente investigado e resolvido. Sem documentação apropriada, informações relevantes que foram avaliadas nas análises e resoluções iniciais correm o risco de serem esquecidas, fazendo com que o novo processo de resolução de conflitos leve novamente a falsos resultados.

Nos dois casos, a documentação apropriada do conflito e de sua resolução proporciona respaldo para o processo de engenharia de requisitos, assegurando que as informações relevantes já conhecidas possam ser levadas em consideração.

### 7.7 Resumo

A qualidade dos requisitos elicitados e documentados deve ser assegurada durante a engenharia de requisitos de forma a garantir que os requisitos atendam adequadamente aos desejos e ideias dos *stakeholders*. É necessário, portanto, validar os requisitos com respeito a seu conteúdo e documentação, bem como determinar se os diferentes *stakeholders* chegaram a um acordo a respeito dos requisitos. Existem diversas técnicas que podem ser selecionadas e combinadas de forma eficaz para validar os requisitos, dependendo das peculiaridades dos objetivos do projeto. Entre as técnicas mais comuns de validação de requisitos encontram-se diversos tipos de revisões de requisitos (parecer de especialista, inspeção, *walkthrough*), bem como a leitura baseada em perspectiva e a validação

por meio de protótipos e *checklists*.

Para a validação de requisitos, é necessário identificar conflitos entre *stakeholders*, bem como analisar e resolver os mesmos de modo adequado. Um gerenciamento sistemático dos conflitos oferece respaldo para a análise e resolução dos conflitos identificados ao longo do processo de validação de requisitos ou de outras atividades da engenharia de requisitos.

## 8 Gerenciar Requisitos

O gerenciamento de requisitos abrange as seguintes atividades, cada qual com seu propósito: designar atributos para requisitos, definir visualizações de requisitos, priorizar e rastrear requisitos, determinar o versionamento de requisitos e gerenciar as mudanças de requisitos. O gerenciamento de requisitos refere-se tanto a requisitos individuais quanto a documentos completos de requisitos.

### 8.1 Designar Atributos para Requisitos

Informações sobre os requisitos devem ser documentadas ao longo do ciclo de vida de um sistema. Isso inclui, por exemplo, identificadores únicos de um requisito, o nome do requisito, o autor e as fontes do requisito, bem como a pessoa responsável pelo requisito.

#### 8.1.1 Atributos para Requisitos em Linguagem Natural e Modelos

Para documentar informações sobre requisitos, a estruturação dessas informações sob forma de atributos tem se revelado útil. Atributos de um requisito são definidos por um nome único, uma curta descrição dos conteúdos e o conjunto de valores possíveis que podem ser designados para o atributo.

A maneira mais simples de definir atributos de requisito é através de uma estrutura tabelar (um *template*). O *template* define as informações relevantes a serem documentadas. Essas informações, isto é, os atributos (ou tipos de atributos) definidos, podem ser diferentes para cada tipo de requisito. Por exemplo, o *template* para requisitos funcionais pode ser diferente do *template* para um requisito de qualidade com respeito aos tipos de atributos definidos e/ou quanto aos valores de atributos permitidos.

Designação de atributos para requisitos baseada em templates

#### 8.1.2 Esquema de Atributos

O conjunto de todos os atributos definidos para uma classe de requisitos (por exemplo, requisitos funcionais, requisitos de qualidade) é denominado um esquema de atributos. Esquemas de atributos são geralmente customizados para atender às necessidades individuais do projeto.

Ao longo do projeto, os atributos dos requisitos recebem valores apropriados de atributos. A figura 8-1 apresenta um exemplo de designação de atributos para um requisito que inclui os atributos "identificador", "nome" e "descrição do requisito", bem como atributos que permitem documentar a estabilidade do requisito, sua fonte e seu autor.

Designação de atributos de requisitos

Nome do atributo	Preenchimento do atributo (valor do atributo)		
Identificador	Req-10		
Name	Desvio Dinâmico de Congestionamentos de Trânsito Evitar tráfego congestionado dinamicamente		
O sistema deve calcular automaticamente uma rota alternativa se detectado que o congestionamento do tráfego excedeu o limite crítico configurado.			
Estabilidade	Responsabilidade	Fonte	Autor
Estável	J. Locke	Product Management	B. Wagner

Figura 8-1 Exemplo de designação de atributos de requisito

O requisito documentado a partir do *template* simples apresentado na figura 8-1 tem o código "Req-10" como seu identificador único. O requisito tem o nome "Desvio Dinâmico de Congestionamentos de Trânsito" e uma descrição que especifica o conteúdo do requisito. A estabilidade do requisito é classificada como "estável", "J. Locke" é identificado como a pessoa responsável pelo requisito, o requisito tem por origem "Product Management" e "B. Wagner" é o autor do requisito.

O leitor do requisito (por exemplo, o contratado, o gerente de produto, o desenvolvedor, o gerente de projeto) conta com uma vantagem significativa quando se utiliza uma documentação baseada em *templates*: as informações do mesmo tipo sempre podem ser localizadas na mesma posição (por exemplo, a estabilidade do requisito sempre estará na seção "estabilidade" do *template*). Além disso, a designação de atributos baseada em *templates* proporciona ao engenheiro de requisitos a vantagem de que fica mais difícil esquecer informações importantes e que essas informações, com o suporte da estrutura do *template* e dos valores do *template* previamente determinados, podem ser documentadas de forma sistemática e correta.

### 8.1.3 Tipos de Atributos de Requisitos

As diversas normas da engenharia de requisitos e as ferramentas mais pertinentes para a documentação e gerenciamento de requisitos frequentemente fornecem um conjunto de atributos previamente definidos. Na tabela 8-1 encontram-se alguns tipos de atributos muito utilizados na engenharia de requisitos.

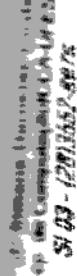
Atributos  
frequentemente  
e utilizados

Tipo de Atributo	Significado
Identificador	Identificador curto e único para um requisito dentro do conjunto de todos os requisitos considerados.
Nome	Nome caracterizador e único.
Descrição	Descreve brevemente o conteúdo do requisito.
Versão	Versão atual do requisito.
Autor	Especifica o autor do requisito.
Fonte	Especifica a fonte ou as fontes do requisito.
Estabilidade	Especifica a estabilidade aproximada do requisito. A estabilidade é o número de mudanças que podem ser esperadas em relação ao requisito. Possíveis graus de estabilidade são: "fixa", "estabilizada" e "volátil".
Criticalidade	Uma estimativa da extensão de perdas e danos, bem como a probabilidade de sua ocorrência.
Prioridade	Especifica a prioridade do requisito com relação às propriedades de priorização selecionadas, por exemplo: "importância para aceitação pelo mercado", "ordem de implementação", "custo da oportunidade perdida em caso de não implementação".

**Tabela 8-1** Tipos de atributos frequentemente utilizados

Além dos atributos de requisitos listados na tabela 8-1, existem muitos outros tipos de atributos adicionais para documentar informações importantes de um requisito. A tabela 8-2 apresenta uma seleção de tipos de atributos adicionais para requisitos.

Tipos de  
atributos  
adicionais para  
requisitos



Tipo de Atributo	Significado
Responsável	Especifica a pessoa, grupo de <i>stakeholders</i> , organização ou unidade organizacional responsável pelo conteúdo de um requisito.
Tipo de requisito	Especifica o tipo de requisito ("requisito funcional", "requisito de qualidade", "restrição") dependendo do esquema de classificação aplicado.
Status do conteúdo	Especifica o status atual do conteúdo do requisito (por exemplo, "ideia", "conceito", "conteúdo detalhado").
Status da validação	Especifica o status atual da validação (por exemplo, "não validado", "contém erro", "em processo de correção").
Status do acordo	Especifica o status atual da negociação de acordo (por exemplo, "não acordado", "acordado", "em conflito").
Esforço	Esforço estimado / efetivo para implementar o requisito.
Release	Designa a <i>release</i> na qual o requisito deverá ser implementado.
Obrigatoriedade legal	Especifica o grau de obrigatoriedade legal do requisito.
Referências cruzadas	Especifica os relacionamentos com outros requisitos (por exemplo, se a implementação do requisito requer a implementação anterior de outro requisito).
Informações gerais	Neste atributo, podem ser documentadas quaisquer informações consideradas relevantes (por exemplo, se a negociação desse requisito está agendada para a próxima reunião com os <i>stakeholders</i> ).

**Tabela 8-2** Tipos adicionais de atributos de requisitos

Os tipos de atributos sugeridos constituem a base para definir um esquema de atributos no projeto de desenvolvimento. Para definir um esquema de atributos, no mínimo os seguintes aspectos específicos devem ser considerados:

- Propriedades específicas do projeto (por exemplo, tamanho do projeto, desenvolvimento local ou distribuído em vários pontos, risco do projeto).
- Restrições organizacionais (por exemplo, normas e regulamentos organizacionais).
- Propriedades e regulamentos do domínio da aplicação (por exemplo, modelos de referência, diretrizes de modelagem, normas).

Customização  
do esquema de  
atributos para  
um projeto  
específico

- Restrições do processo de desenvolvimento (por exemplo, legislação de responsabilidade civil, normas de processo).

Ao empregar ferramentas de gerenciamento de requisitos, a definição da estrutura de atributos dos requisitos geralmente não é realizada a partir de tabelas, mas baseada em modelos, por meio de modelos de informação. Uma definição de esquema de atributos baseada em modelos determina os tipos de atributos e as limitações dos valores de atributos, algo semelhante à definição baseada em *templates*. Além disso, a definição de esquema de atributos baseada em modelos permite determinar relacionamentos entre tipos de atributos de diferentes esquemas de atributos.

Definição de atributos por meio de modelos de informação

Além das vantagens da definição baseada em tabelas, a designação de atributos de requisitos a partir de modelos permite também considerar as dependências entre requisitos através do acesso seletivo aos requisitos. Isto ajuda a manter a consistência nos atributos dos requisitos. Além disso, o modelo de informação de uma designação de atributos de requisitos baseada na modelagem pode servir de base para a definição de uma estrutura de atributos a ser utilizada em uma ferramenta de gerenciamento de requisitos (ver seção 9.3). Adicionalmente, *templates* para a designação de atributos de requisitos podem ser gerados com base no modelo de informação.

Vantagens da designação de atributos baseada em modelos

## 8.2 Visualizações de Requisitos

A estruturação de requisitos por meio de modelos de informação permite a geração de visualizações específicas de requisitos. Na prática, é possível perceber que a quantidade de requisitos e o número de dependências entre os requisitos aumentam constantemente. Para manter a complexidade dos requisitos dentro de limites manejáveis para os membros do projeto, é necessário acessar os requisitos de forma seletiva, dessa forma filtrando os requisitos de acordo com a tarefa atual.

As visualizações de requisitos são muitas vezes definidas para diferentes papéis desempenhados no processo de desenvolvimento. Exemplos incluem o arquiteto, o programador, o gerente de projeto e o testador. É comum definir múltiplas visualizações para um determinado papel, para oferecer respaldo às subatividades de cada papel. Uma visualização específica também pode ser aplicada para múltiplos papéis.

Definição de visualizações a partir de papéis específicos

### 8.2.1 Visualizações Seletivas da Base de Requisitos

Uma visualização apresenta parte de todas as informações disponíveis sobre

requisitos (a base de requisitos). A partir de uma visualização, é possível:

- Selecionar requisitos específicos, isto é, nem todos os requisitos estarão incluídos em uma visualização.
- Ocultar certos atributos de requisitos, isto é, nem todos os atributos de um requisito estarão incluídos em uma visualização.
- Combinar esses princípios de seleção de forma aleatória, isto é, somente um subconjunto de todos os requisitos disponíveis e somente um subconjunto de todos os atributos disponíveis estarão incluídos em uma visualização.

A Figura 8-2 exemplifica a geração de três visualizações, representadas por meio de uma tabela definida com base na estrutura dos atributos. Nos três casos, as visualizações são geradas através da seleção de tipos de atributos, bem como pela determinação dos atributos que deverão estar disponíveis. A definição da primeira visualização (1), por exemplo, determina que sejam selecionados apenas os requisitos pelos quais "J. Locke" é responsável e que possuem uma estabilidade "fixa". De todos os requisitos selecionados, apenas os atributos "identificador", "nome", "descrição" e "autor" estão sendo considerados.

Geração de visualizações seletivas

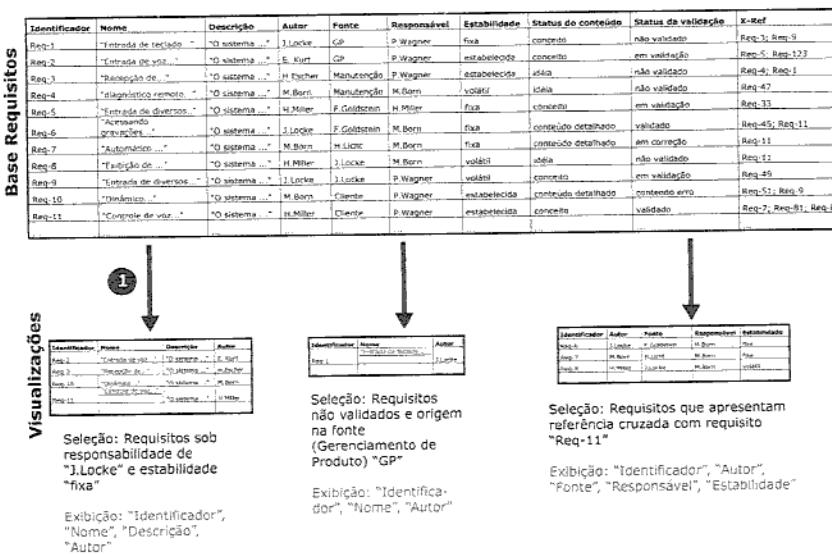


Figura 8-2 Visualizações seletivas dos requisitos

### 8.2.2 Visualizações Consolidadas dos Requisitos

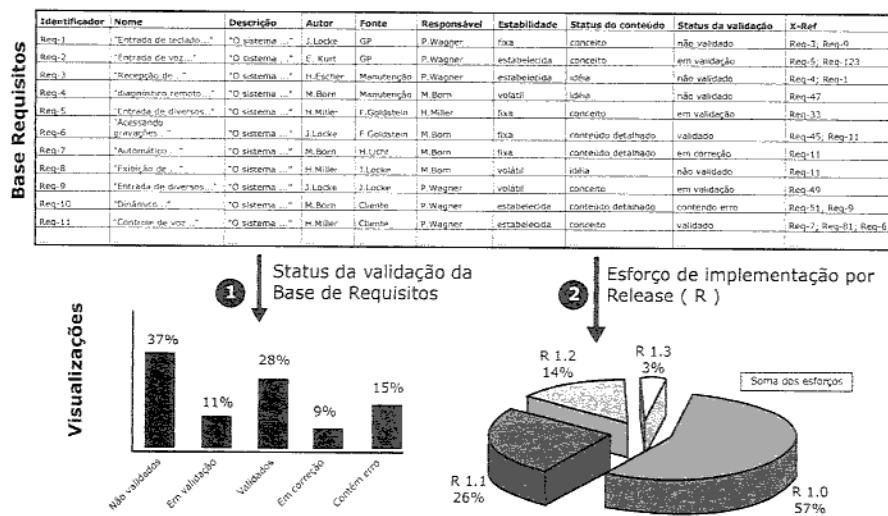
Além de selecionar informações existentes na base de requisitos, visualizações também podem conter dados gerados ou consolidados que não estão imediatamente contidos nos requisitos. Visualizações que somente contêm dados gerados ou consolidados são chamadas de visualizações consolidadas.

Visualizações consolidadas podem ser definidas através da agregação de dados contidos na base de requisitos. Uma visualização consolidada pode, por exemplo, conter informações sobre o percentual de requisitos que se originam de uma fonte específica.

Uma visualização individual pode também consistir em uma combinação de dados gerados, consolidados e selecionados.

Geração de visualizações consolidadas

Combinação entre consolidação e seleção de dados



**Figura 8-3** Visualização consolidada gerada a partir de uma base de requisitos.

A figura 8-3 apresenta duas visualizações consolidadas dos requisitos. A visualização (1), "Status de validação da base de requisitos", agrupa requisitos de acordo com o status atual de validação e calcula o percentual de requisitos "não validados", "em validação", "validados", "em processo de correção" e "contêm erro". O resultado é exibido na forma de um gráfico de barras na figura acima. A visualização (2), "Esforço de implementação por release", apresenta o esforço estimado e efetivo da implementação dos requisitos de determinada release. Para calcular esses dados agregados, os requisitos são agrupados por suas respectivas releases e seu esforço de implementação somado. O resultado é apresentado sob a forma de um gráfico de pizza na figura 8-3.

Nota do editor: Existem dois erros na figura 8.3 acima; Vide nota de rodapé<sup>10</sup>.

## 8.3 Priorização de Requisitos

Requisitos são priorizados ao longo da engenharia de requisitos através do uso de diferentes critérios de priorização em todas as subatividades. Requisitos podem ser priorizados por sua ordem de implementação, por exemplo. Devido às diferentes priorizações na várias subatividades, a prioridade de um requisito pode ser determinada por um ou mais atributos (por exemplo: prioridade do contratado e prioridade devido à urgência de implementação entre outros).

### 8.3.1 Método de Priorização de Requisitos

Para priorizar um conjunto de requisitos, é preciso primeiro definir uma meta (isto é, uma finalidade). Além disso, as restrições de priorização devem ser documentadas, tais como a disponibilidade dos diversos *stakeholders* e grupos de *stakeholders*, ou os recursos disponíveis para priorização.

Dependendo da meta da priorização, escolhe-se o critério para priorizar os requisitos (ou a combinação de um ou mais critérios). A seguir, alguns exemplos típicos de critérios de priorização:

- Custo de implementação.
- Risco.
- Prejuízo devido à implementação malsucedida.
- Volatilidade.
- Importância.
- Duração da implementação (isto é, quanto tempo o requisito levará para ser implementado).

Dependendo da meta da priorização e dos critérios de priorização selecionados, é geralmente necessário incluir diferentes *stakeholders* no processo de priorização. A seleção de *stakeholders* adequados pode garantir a disponibilidade do conhecimento especialista exigido durante o processo de priorização. Os

Determinar a meta e as restrições da priorização

Determinar critérios de priorização

Determinar stakeholders

<sup>10</sup> Erro #1: No gráfico de barras, considerando apenas os 11 requisitos que aparecem na tabela da figura 8-3, a distribuição percentual das colunas deveria ser de 36%, 27%, 18%, 9% e 9%, respectivamente, da esquerda para a direita. Erro #2: O gráfico de pizza faz referência a identificadores de requisitos (R1.2, R1.3,...) que não são os identificadores utilizados na tabela (Req1, Req3,...).

*stakeholders* que deveriam ser incluídos são, dependendo da meta e dos critérios de priorização, os desenvolvedores, gerentes de projeto, clientes ou usuários, por exemplo.

Além disso, os requisitos a serem priorizados devem ser selecionados. Ao selecionar requisitos, devemos nos assegurar que os requisitos selecionados são do mesmo nível de abstração. Priorizar requisitos com níveis de detalhamento consideravelmente diferentes levará a resultados inconsistentes e enganosos, pois os *stakeholders* tendem a atribuir uma prioridade mais alta para requisitos em níveis mais altos de abstração do que para requisitos mais refinados e concretos.

Seleção de requisitos

Com base nas propriedades determinadas da priorização (por exemplo, restrições, critérios de priorização, etc.), uma técnica adequada de priorização ou uma combinação de múltiplas técnicas pode ser selecionada.

Seleção de técnicas de priorização

### 8.3.2 Técnicas de Priorização de Requisitos

Existem inúmeras técnicas de priorização. As técnicas diferem principalmente em termos do tempo e esforço que exigem, mas também com respeito à adequabilidade dos diferentes critérios de priorização e propriedades de projeto.

O espectro de técnicas de priorização cobre desde a classificação simples a partir de um único critério até abordagens analíticas complexas de priorização, tais como a AHP (Analytic Hierarchy Process) [Saaty 1980], a *Cost-Value Approach* [Karlsson e Ryan 1997], ou o QFD (*Quality Function Deployment*) [Akao 1990].

Técnicas ad hoc e técnicas analíticas

Em muitos projetos, técnicas simples de priorização *ad hoc* tais como estabelecer um *ranking* ou classificação de requisitos, podem ser apropriadas. Especialmente quando considerados os recursos disponíveis, o uso de técnicas *ad hoc* muitas vezes é recomendável.

Se o processo de decisão for considerado demasiadamente incompreensível, ou se os resultados contiverem demais erros, abordagens analíticas de priorização devem ser empregadas (adicionalmente). Na prática, diversas técnicas de priorização são utilizadas em combinação para priorizar os requisitos [Lehtola e Kauppinen 2006].

#### Ranking e Top 10

Duas técnicas consagradas para a priorização de requisitos são, por exemplo, as seguintes [Lauesen 2002]:

- *Ranking*: Nessa técnica, diversos *stakeholders* selecionados ordenam os requisitos a serem priorizados de acordo com um critério específico.
- *Top 10*: Nessa técnica, selecionam-se os "n" requisitos mais importantes para um critério definido. Mais tarde, uma classificação

ordenada para esses requisitos é determinada. Essa ordem no "ranking" representa a importância dos requisitos selecionados de acordo com o critério definido.

### Classificação por Critério Único

Outra técnica de priorização frequentemente utilizada na prática tem por base a classificação de requisitos de acordo com a importância da execução do requisito para o sucesso do sistema. Esse tipo de priorização é baseada na atribuição de cada requisito para uma das seguintes classes de prioridade [IEEE Std. 830-1998]:

- *Mandatory*: Um requisito *mandatory* (obrigatório, imprescindível) é um requisito que deverá ser implementado a qualquer custo, sob pena de ameaçar o sucesso do sistema.
- *Optional*: Um requisito *optional* (opcional) é um requisito que não precisa necessariamente ser implementado. Negligenciar alguns requisitos dessa classe de prioridade não ameaça o sucesso do sistema.
- *Nice-to-have*: Requisitos *nice-to-have* (isto é, requisitos que "*seriam interessantes, bonitinhos, cosméticos*") são requisitos que não influenciam o sucesso do sistema se não forem implementados.

Na prática, diferenciar entre requisitos "optional" e "nice-to-have" pode ser muito difícil. Por esse motivo, a classificação de requisitos exige critérios de classificação que sejam o mais objetivamente verificáveis quanto possível.

### Classificação Kano

A abordagem Kano apresentada na seção 3.2 também proporciona respaldo para a priorização de requisitos. Usando a abordagem Kano, podemos classificar e priorizar requisitos em termos de sua aceitação no mercado. Para tal, classificam-se os requisitos nas três seguintes classes de propriedades (veja também a figura 3-1):

- *Dissatisfiers* (propriedades básicas de satisfação): Um requisito especifica uma propriedade básica de satisfação que o sistema deverá obrigatoriamente possuir para ser lançado no mercado com sucesso.
- *Satisfiers* (propriedades esperadas de satisfação): Um requisito especifica uma propriedade esperada de satisfação se os clientes exigem de forma consciente a propriedade associada ao requisito. As propriedades *satisfiers* determinam o grau de satisfação do cliente. Um aumento no número de *satisfiers* geralmente resulta em aumento de satisfação do cliente.
- *Delighters* (propriedades inesperadas de satisfação): Um requisito especifica uma propriedade inesperada de satisfação se os clientes não exigem conscientemente determinada propriedade do sistema, ou se os stakeholders não esperam a implementação da propriedade. A satisfação do cliente aumenta de forma exponencial com a implementação de requisitos classificados como *delighters*.

As três propriedades da abordagem de Kano

Com base nos requisitos classificados de acordo com a abordagem Kano, é possível realizar uma priorização dos requisitos para planejar, por exemplo, as diferentes *releases* do sistema.

### Matriz de Priorização de Wiegers

A matriz de priorização de Wiegers [Wiegers 1999] é uma abordagem analítica para a priorização de requisitos. O núcleo central da abordagem é uma matriz de priorização com a qual as prioridades dos respectivos requisitos podem ser determinadas de forma sistemática. A figura 8-4 apresenta a estrutura de uma matriz de priorização de Wiegers, bem como o método pelo qual as prioridades são calculadas.

Cálculo das prioridades de requisitos

① Peso Ponderado x	Benefício (x2)	Prejuízo (x1)			Custo (x1)		Risco (x0,5)			
② Requisitos a priorizar	Benefício Relativo	Prejuízo relativo	Total	(%)	Custo Relativo	(%)	Risco relativo	(%)	Prioridade	Posição no ranking
R <sub>1</sub>	5	3	13	16,9%	2	13,3%	1	9,1%	0,944	1
R <sub>2</sub>	9	7	25	32,5%	5	33,3%	3	27,3%	0,691	3
R <sub>3</sub>	5	7	17	22,1%	3	20,0%	2	18,2%	0,759	2
R <sub>4</sub>	2	1	5	6,5%	1	6,7%	1	9,1%	0,579	4
R <sub>5</sub>	4	9	17	22,1%	4	26,7%	4	36,4%	0,492	5
Total	25	27	77	100%	15	100%	11	100%	-	
	③	④	⑤		⑥		⑦		⑧	⑨

Figura 8-4 Cálculo de prioridades com uma matriz de priorização de Wiegers

O cálculo de prioridades através de uma matriz de priorização de Wiegers será apenas rapidamente delineado abaixo. Informações mais detalhadas podem ser encontradas em [Wiegers 1999].

O cálculo de prioridades pela matriz de priorização de Wiegers pode ser realizado como segue:

Método sistemático para determinar as prioridades de requisitos

- Determinar o peso ponderado do benefício, prejuízo, custo e risco.

- Determinar os requisitos a serem priorizados.

- Estimar o benefício relativo.

- Estimar o prejuízo relativo.

- Calcular os valores totais e percentuais de cada requisito:

$$\text{Valor\%}(R_i) = \frac{\text{Benefício}(R_i)}{\text{Prejuízo}(R_i)} \times \frac{\text{PesoPonderadoBeneficio}}{\text{PesoPonderadoPrejuízo}} +$$

- Estimar o custo relativo e calcular o percentual de custo de cada requisito.

- Estimar o risco relativo e calcular o percentual de risco de cada requisito.

- Calcular as prioridades dos requisitos individuais:

$$\text{Prioridade(Ri)} = \frac{\text{Valor\%}(Ri)}{[\text{Custo\%}(Ri) \times \text{Peso Ponderado Custo} + \text{Risco\%}(Ri) \times \text{Peso Ponderado Risco}]}.$$

- ⑨ Determinar a posição no *ranking* de prioridade de cada requisito.

Na prática, fica claro que abordagens analíticas de priorização, tais como a matriz de priorização de Wiegers esboçada acima, exigem consideravelmente mais tempo e esforço do que as abordagens *ad hoc*, de modo que essas abordagens *ad hoc* devem ser favoráveis em muitos casos. Porém, as abordagens analíticas possuem a vantagem de que o grau de subjetividade nos resultados do processo de priorização pode ser significativamente reduzido, levando assim a resultados mais objetivos e comprehensíveis em situações complexas e críticas de priorização.

## 8.4 Rastreabilidade de Requisitos

Um aspecto importante do gerenciamento de requisitos é assegurar a rastreabilidade dos requisitos. A rastreabilidade de um requisito é a possibilidade de rastrear os requisitos ao longo de todo o ciclo de vida do sistema (ver seção 4.5.5).

### 8.4.1 Vantagens de Requisitos Rastreáveis

O uso de informações rastreáveis oferece respaldo ao desenvolvimento do sistema em muitos aspectos, sendo muitas vezes pré-condição para estabelecer e utilizar certas técnicas durante o processo de desenvolvimento [Pohl 1996; Ramesh 1998]:

Vantagens da rastreabilidade de requisitos

- *Verificabilidade:* A rastreabilidade de requisitos permite verificar se um requisito foi implementado no sistema, isto é, se o requisito foi implementado através de uma propriedade do sistema.
- *Identificação de propriedades desnecessárias do sistema:* A rastreabilidade de requisitos permite identificar as chamadas soluções "folheadas em ouro" no sistema desenvolvido, dessa forma permitindo a identificação de propriedades desnecessárias. Para tal, cada propriedade do sistema (funcional ou qualitativa) é inspecionada para determinar se a mesma contribui para a implementação de um determinado requisito do sistema.
- *Identificação dos requisitos desnecessários:* Rastrear requisitos de volta à sua origem permite identificar requisitos que não contribuem para qualquer meta do sistema e não estão associados a qualquer fonte. Geralmente não há motivo para esses requisitos existirem, não havendo, portanto, necessidade de implementá-los.
- *Análise de impacto:* A rastreabilidade de requisitos permite analisar os efeitos de eventuais mudanças nos requisitos (gerenciamento de mudanças). Por exemplo, a rastreabilidade de requisitos permite identificar

os artefatos de requisitos que devem ser modificados quando os requisitos que lhe servem de base passam por alguma mudança.

- *Reusabilidade:* A rastreabilidade de requisitos permite a reutilização de artefatos de requisitos em outros projetos. Ao comparar os requisitos de algum projeto anterior com os requisitos de um novo projeto através de relacionamentos de rastreamento, podem ser identificados artefatos de desenvolvimento (por exemplo, componentes, casos de teste) que podem ser adaptados e/ou reutilizados no novo projeto de desenvolvimento.
- *Determinação de responsabilidade (accountability):* A rastreabilidade de requisitos permite a atribuição retroativa de esforços de desenvolvimento para um requisito. Depois que um requisito foi implementado, por exemplo, todos os esforços parciais para o respectivo artefato de desenvolvimento podem ser somados e associados ao requisito.
- *Mantenção:* A rastreabilidade de requisitos simplifica a manutenção do sistema. Por exemplo, a causa e o efeito de falhas podem ser identificadas, os componentes do sistema que são afetados pela falha podem ser determinados e o esforço para corrigir o erro por trás da falha pode ser estimado.

#### 8.4.2 Rastreabilidade Definida por Finalidade

Tendo em vista que recursos são geralmente muito limitados durante projetos de desenvolvimento, é praticamente inviável capturar todas as informações concebíveis que possibilitam a rastreabilidade dos requisitos ao longo do ciclo de vida do sistema.

Para estabelecer a rastreabilidade de requisitos de maneira eficaz e eficiente, as informações a serem registradas devem ser selecionadas em termos de sua finalidade. Em outras palavras, somente as informações com um propósito claro para o desenvolvimento ou evolução do sistema [Dömges e Pohl 1998; Ramesh e Jarke 2001] devem ser registradas. Registrar informações de rastreabilidade sem uma finalidade definida tem como resultado que as informações registradas muitas vezes não podem ser utilizadas proveitadamente no projeto de desenvolvimento. Informações de rastreabilidade registradas dessa forma costumam oferecer apenas traços gerais e incompletos, não estruturados e incorretos quanto ao seu futuro uso.

Finalidade das  
informações  
de  
rastreabilidade

#### 8.4.3 Classificação de Relacionamentos de Rastreabilidade

A literatura a respeito do tema rastreabilidade de requisitos propõe diferentes tipos de rastreabilidade de requisitos. Uma diferenciação comum é a distinção feita entre a rastreabilidade pré-especificação de requisitos e a rastreabilidade pós-especificação de requisitos [Gotel e Finkelstein 1994]. Assim, distinguimos três

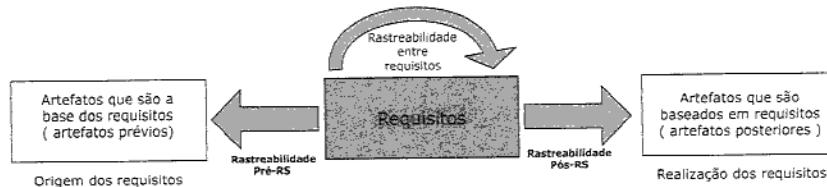
Rastreabilidade  
e pré-  
especificação  
(Pre-RS) c

tipos de rastreabilidade:

- A *rastreabilidade pré-especificação de requisitos (Pre-RS<sup>11</sup>)*: Abrange relacionamentos de rastreabilidade entre requisitos e aqueles artefatos que formam a base dos requisitos, por exemplo, artefatos como a fonte ou origem de um requisito (artefatos prévios).
- A *rastreabilidade pós-especificação de requisitos (Pos-RS)*: Envolve as informações de rastreabilidade entre requisitos e artefatos oriundos de atividades subsequentes de desenvolvimento. Por exemplo, tais artefatos poderiam ser componentes, implementações ou casos de teste que pertencem a um requisito (artefatos posteriores).
- A *rastreabilidade entre requisitos*: Envolve o mapeamento de dependências entre requisitos. Um exemplo desse tipo de rastreabilidade seria a informação de que um requisito refina, generaliza ou substitui outro requisito.

rastreabilidade  
pós-  
especificação  
(Pos-RS)

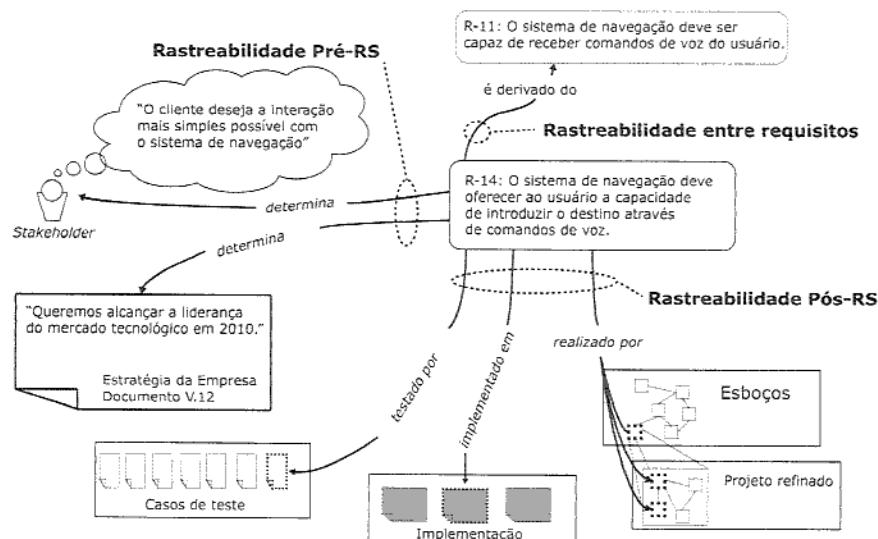
A figura 8-5 apresenta os três tipos de rastreabilidade de requisitos na engenharia de requisitos.



**Figura 8-5** Tipos de rastreabilidade de requisitos

A figura 8-6 apresenta os três tipos de rastreabilidade de requisitos, usando como exemplo o requisito "R-14". A rastreabilidade pré-RS abrange os relacionamentos do requisito "R-14" com sua origem. A origem desse requisito são os artefatos no contexto do sistema que influenciam o requisito. A rastreabilidade pós-RS do requisito "R-14" consiste dos relacionamentos com os componentes no esboço preliminar do projeto, no projeto refinado e em sua respectiva implementação, bem como nos casos de teste usados para testar o sistema e verificar a implementação do requisito no sistema desenvolvido.

<sup>11</sup> RS = Requirements Specification



**Figura 8-6** Exemplo com os três tipos de rastreabilidade

Além disso, a figura 8-6 mostra a rastreabilidade entre requisitos. O relacionamento de rastreabilidade entre os requisitos "R-14" e "R-11" documenta que o requisito "R-14" foi derivado do requisito "R-11".

#### 8.4.4 Representação da Rastreabilidade de Requisitos

As informações de rastreabilidade de requisitos podem ser representadas de diversas formas. As abordagens mais comuns para representar a rastreabilidade são referências textuais simples, *hyperlinks*, matrizes de rastreabilidade e grafos de rastreabilidade.

##### Referências Textuais e Hyperlinks

Esta simples maneira de representar informações de rastreabilidade de um requisito consiste em anotar o artefato-alvo como uma referência textual no requisito (artefato inicial), ou estabelecer um *hyperlink* entre o artefato inicial e o artefato-alvo. Ao conectar artefatos, diferentes tipos de *hyperlinks* com conexões semânticas específicas podem ser utilizados.

##### Matrizes de Rastreabilidade

Outra técnica comum para representar e documentar informações de rastreabilidade entre requisitos, bem como entre requisitos e artefatos anteriores e posteriores no

processo de desenvolvimento, são as matrizes de rastreabilidade. Nas linhas de uma matriz de rastreabilidade constam os artefatos iniciais (requisitos). Nas colunas, os artefatos-alvo (por exemplo, fontes dos requisitos, artefatos de desenvolvimento, requisitos). Se existe um relacionamento de rastreabilidade entre um artefato inicial na linha "n" com um artefato-alvo na coluna "m", a célula (n,m) é marcada na matriz de rastreabilidade.

A figura 8-7 mostra uma simples matriz de rastreabilidade para o relacionamento de rastreabilidade "derivado" existente entre dois requisitos. Um registro na matriz especifica que existe um relacionamento de rastreabilidade do tipo "derivado" de um requisito "*Req-n*" para outro requisito "*Req-m*", no sentido de que "*Req-n*" foi derivado de "*Req-m*".

Interpretação  
de uma matriz  
de  
rastreabilidade

		Artefatos Alvo				
		Req-1	Req-2	Req-3	Req-4	Req-5
Artefatos Iniciais	Req-1		X			
	Req-2			X		
	Req-3					X
	Req-4			X		
	Req-5					

**Figura 8-7** Representação de informações de rastreabilidade em uma matriz de rastreabilidade

Na prática, fica evidente que matrizes de rastreabilidade são de difícil manutenção conforme aumenta o número de requisitos. Uma matriz de rastreabilidade que documenta os relacionamentos de refinamento entre apenas 2000 requisitos, por exemplo, contém mais de 4 milhões de células. Além disso, muitas matrizes de rastreabilidade precisam ser criadas para que se possa representar de forma clara as informações disponíveis (por exemplo, com respeito aos diversos tipos de relacionamentos de rastreabilidade).

Manutenção  
de matrizes de  
rastreabilidade

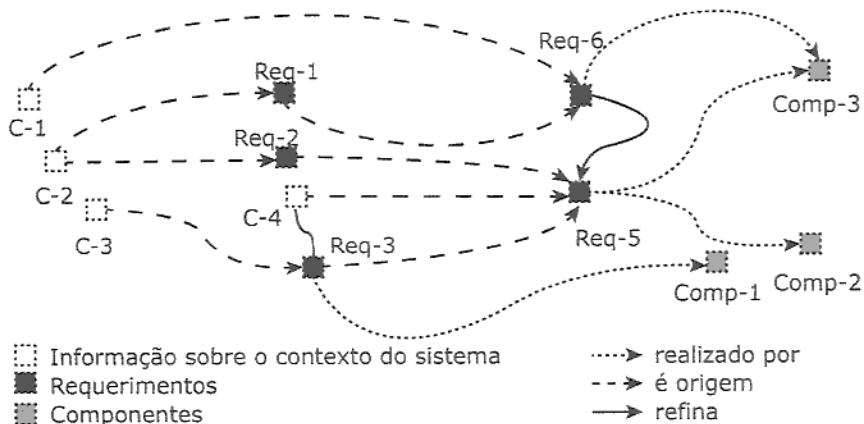
### Grafos de Rastreabilidade

Um grafo de rastreabilidade é um gráfico no qual todos os nós representam artefatos e todas as linhas representam relacionamentos entre artefatos. A distinção entre diferentes artefatos e tipos de rastreabilidade pode ser feita através da

designação de diferentes atributos para os nós e as linhas do grafo.

A figura 8-8 mostra a representação de informações de rastreabilidade através de um exemplo simples. No grafo de rastreabilidade, um tipo de nó é definido para cada tipo de artefato (informações do contexto "C", requisito "Req-n", componente "Comp-n"). Além disso, três tipos de linhas são definidas, representando três tipos de relacionamentos de rastreabilidade ("executado por", "é origem", "refina").

Grafo de rastreabilidade para diferentes artefatos de desenvolvimento



**Figura 8-8** Representação de rastreabilidade em grafo de rastreabilidade (vista parcial)

Caso seja necessário gerenciar as informações de rastreabilidade sobre artefatos anteriores (por exemplo, *stakeholders* e protocolos de entrevistas) e artefatos posteriores (por exemplo, casos de teste e componentes), cadeias de rastreabilidade para o respectivo requisito podem ser criadas em diferentes níveis, incluindo uma rastreabilidade do requisito ao longo de todo o ciclo de vida do sistema. Ferramentas comuns para manutenção de requisitos permitem a definição de níveis de representação ao criar cadeias de rastreabilidade. Assim, dependendo do nível selecionado, podem ser gerados e exibidos apenas os relacionamentos imediatos de um requisito, ou então cadeias completas de rastreabilidade. As cadeias de rastreabilidade são o fundamento para uma análise de impacto abrangente durante o gerenciamento das mudanças de requisitos.

Cadeias de rastreabilidade

## 8.5 Versionamento de Requisitos

Durante o ciclo de vida de um sistema, os requisitos de um sistema passam por

mudanças conforme novos requisitos são acrescentados e requisitos existentes são retirados ou alterados. Existem diversas razões para as mudanças nos requisitos. Uma razão possível, por exemplo, é o fato de que os *stakeholders* aprendem cada vez mais sobre o sistema conforme a engenharia de requisitos progride. Como resultado, novos requisitos e alterações nos requisitos surgem na mente dos *stakeholders*. Devido a essas mudanças, recomenda-se fortemente realizar um versionamento adequado dos requisitos.

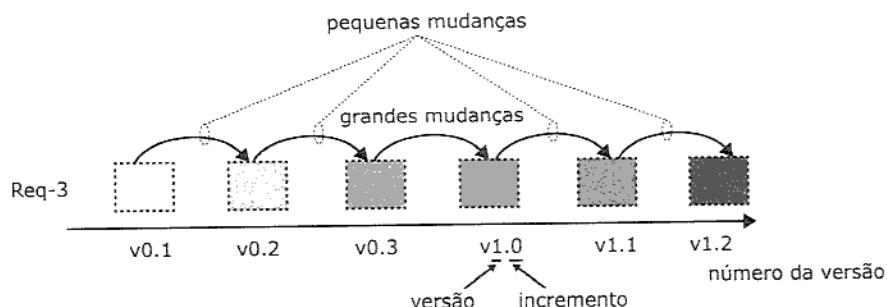
O versionamento dos requisitos tem por objetivo proporcionar acesso aos estados de desenvolvimento específicos de requisitos individuais ao longo do ciclo de vida do sistema. A versão de um requisito é definida por seu conteúdo específico do estado de desenvolvimento, sendo marcada por um número único de versão. As informações sujeitas ao gerenciamento de versões podem ser requisitos textuais isolados, frases, seções de documentos de requisitos ou documentos completos de requisitos, bem como modelos de requisitos e modelos parciais de requisitos.

Informações sujeitas ao controle de versões

### 8.5.1 Versões de Requisitos

Ao versionar requisitos, pode-se distinguir entre a versão e o incremento do número da versão. Por exemplo, a versão de número 1.2 refere-se a um requisito de versão 1 e incremento 2.

A figura 8-9 ilustra o método para atribuir números de versões. Como demonstrado na figura, no caso de mudanças menores ao conteúdo, o número do incremento é aumentado por 1. Se ocorrerem mudanças maiores, o número da versão é aumentado. Se o número da versão é aumentado, o incremento passa para o valor inicial (0). O número da versão pode ser antecedido pela letra "v" para torná-lo mais compreensível e para facilitar sua identificação como tal.



**Figura 8-9** Versões de requisitos

Além dessa estruturação bastante simples por meio de números de versões e do método proposto para o versionamento de requisitos, outros métodos para atribuir números de versões são amplamente utilizados. Por exemplo, é possível distinguir entre o identificador da versão, o identificador do incremento e identificador do

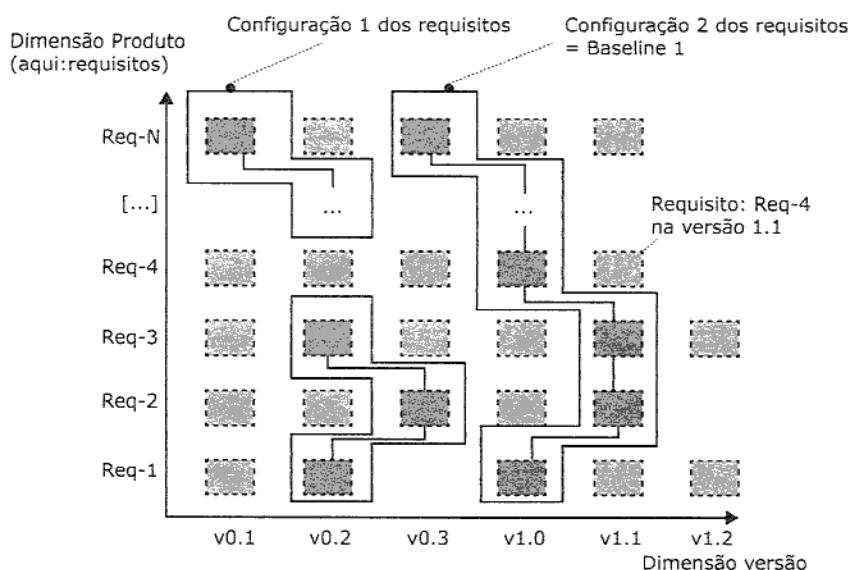
sub-incremento (v1.2.12).

### 8.5.2 Configurações de Requisitos

Uma configuração de requisitos consiste em um conjunto de requisitos com a condição adicional que cada requisito selecionado está presente na configuração de requisitos com exatamente uma versão, identificada pelo número da versão.

O gerenciamento de configurações de requisitos pode ser descrito em duas dimensões [Conradi e Westfechtel 1998]: Na dimensão "Produto", o gerenciamento de configurações lida com requisitos individuais dentro da base de requisitos. Na dimensão "Versão", o gerenciamento de configurações considera os vários estados de desenvolvimento como parte do gerenciamento de versões dentro da dimensão "Produto". A figura 8-10 ilustra as duas dimensões de gerenciamento de configurações de requisitos. Os requisitos estão representados no eixo dos requisitos, enquanto as diversas versões estão representadas no eixo das versões.

Dimensões do gerenciamento de configurações de requisitos



**Figura 8-10** Dimensões do gerenciamento de configurações de requisitos (baseado em [Conradi e Westfechtel 1998])

Um configuração de requisitos agrupa um conjunto definido de requisitos logicamente conectados (mais precisamente, versões de requisitos), sendo que cada requisito da base de requisitos pode ocorrer no máximo uma vez na configuração de requisitos. Uma configuração de requisitos não precisa conter uma versão de cada

Propriedades das configurações de requisitos

requisito sendo considerado na dimensão "Produto" (ver figura 8-10, configuração de requisitos 1).

Uma configuração de requisitos possui as seguintes propriedades:

- *Coesão lógica entre os requisitos*: Os requisitos contidos na configuração são direta e logicamente conectados entre si, isto é, os requisitos foram agrupados em uma configuração em comum, orientada para um objetivo específico.
- *Consistência dos requisitos*: Os requisitos contidos na configuração não se contradizem uns aos outros, isto é, a configuração contém requisitos livres de contradições em suas respectivas versões.
- *Identificação única*: Uma configuração possui um identificador único (ID) que pode ser utilizado para identificar a configuração de forma exclusiva.
- *Inalterabilidade dos requisitos*: Uma configuração define um determinado e inalterável estado da especificação. Se ocorrer alguma mudança nos requisitos de uma configuração, o resultado será uma nova versão dos requisitos e possivelmente da configuração.
- *Possibilidade de retorno a versões anteriores da base de requisitos (rollback)*: Se houver necessidade de desfazer mudanças nos requisitos, as configurações oferecem a possibilidade de retornar os requisitos para uma versão específica dentro de uma configuração. Consequentemente, um estado consistente da especificação pode ser mantido.

### 8.5.3 Baselines de Requisitos

*Baselines* de requisitos são configurações específicas de requisitos que tipicamente consistem de versões estáveis de requisitos, muitas vezes também definindo uma etapa de desenvolvimento (*release*) de um sistema. Devido a essa característica, *baselines* de requisitos são geralmente visíveis externamente (por exemplo, para o cliente). O uso de *baselines* de requisitos oferece respaldo a várias importantes atividades do processo de desenvolvimento:

- *Base para o planejamento de releases*: *Baselines* de requisitos são configurações de requisitos "estáveis", especialmente marcados para o cliente. *Baselines*, portanto, atuam como base de comunicação não somente para o planejamento, mas também para a definição de estados de desenvolvimento (*system releases*).
- *Estimativa do esforço envolvido com a implementação*: Como as *baselines* podem ser utilizadas para a definição de *releases* do sistema, elas também podem ser usadas para estimar o esforço necessário para completar uma *release* do sistema. Isso pode ser feito estimando o esforço parcial para implementar um requisito da *baseline* e somando o esforço total para o restante da *baseline*.

Configuração  
vs. *Baselines*

- *Comparação com produtos concorrentes:* Baselines de requisitos podem ser utilizadas para comparar o sistema planejado com sistemas concorrentes.

## 8.6 Gerenciamento de Mudanças de Requisitos

Requisitos mudam ao longo de todo o desenvolvimento e ciclo de vida do sistema. Isso significa que novos requisitos são acrescentados e requisitos existentes são modificados ou removidos.

### 8.6.1 Mudanças de Requisitos

As razões para mudanças nos requisitos podem ser múltiplas. Além das mudanças que derivam diretamente de erros nos requisitos ou de requisitos incompletos, a evolução do próprio contexto pode exigir a mudança de requisitos. Por exemplo, mudanças na aplicação do sistema desejada pelo *stakeholder*, modificações na legislação, novas tecnologias ou concorrência adicional no mercado são fatores que podem influenciar os requisitos e exigir alterações. Mudanças de requisitos, no entanto, também podem resultar de falhas no sistema depois de sua implementação, caso possa ser demonstrado que a falha foi causada por um erro no requisito.

Razões para mudanças

Mudanças nos requisitos não são necessariamente algo negativo. São meramente uma indicação de que os *stakeholders* estão acompanhando o sistema atentamente e aprendendo cada vez mais sobre suas funções, qualidades e restrições. Se apenas raramente ocorrem solicitações de mudanças durante o desenvolvimento do sistema, isso pode ser sinal de baixo interesse dos *stakeholders* no sistema a ser desenvolvido.

Mudanças não são necessariamente algo negativo

Por outro lado, se as solicitações de mudanças ocorrem com muita frequência, desenvolver um sistema com o qual todos os *stakeholders* estejam de acordo torna-se praticamente impossível. Uma alta frequência de mudanças é um indicativo, entre outros problemas, de uma execução inadequada das atividades de engenharia de requisitos, tais como as técnicas de elicitação e negociação. Além disso, uma alta frequência de mudanças absorve muitos recursos no projeto de desenvolvimento.

A frequência de mudanças como um indicador de qualidade de processo

### 8.6.2 O Comitê de Controle de Mudanças

Ao longo do ciclo de vida do sistema é necessário canalizar solicitações de mudanças de requisitos e definir um processo estruturado que resultará em uma decisão justificada sobre a aprovação (ou não) de determinada solicitação de mudança e sobre como ela será aprovada. Mudanças podem referir-se a requisitos

individuais (por exemplo, redefinir um requisito) ou podem referir-se ao documento completo de requisitos. Tanto a avaliação de mudanças de requisitos quanto a decisão sobre a execução de determinada mudança são geralmente responsabilidade de um comitê de controle de mudanças. O comitê de controle de mudanças tem as seguintes atribuições:

- Estimar o esforço para executar a mudança (eventualmente designando terceiros para realizar a análise do esforço).
- Avaliar solicitações de mudanças, por exemplo, em termos de custo-benefício.
- Definir mudanças de requisitos, ou definir novos requisitos com base nas solicitações de mudanças.
- Aceitar ou rejeitar as solicitações de mudanças.
- Classificar as solicitações de mudanças recebidas.
- Priorizar as solicitações de mudanças aceitas.
- Alocar as solicitações de mudanças aceitas para projetos de modificações.

Atribuições do comitê de controle de mudanças

O comitê de controle de mudanças (CCM) poderá, em certos casos, delegar essas atribuições para terceiros. Decisões sobre mudanças devem ser negociadas e acordadas com o cliente e todos os *stakeholders* envolvidos no projeto de desenvolvimento. O comitê de controle de mudanças, portanto, deve incluir os seguintes *stakeholders*, entre outros, dependendo das propriedades do sistema a ser desenvolvido e do processo de desenvolvimento:

- Gerente de mudanças.
- Cliente.
- Arquiteto.
- Desenvolvedor.
- Gerente de configuração.
- Representante do cliente.
- Gerente de produto.
- Gerente de projeto.
- Representante da garantia de qualidade.
- Engenheiro de requisitos.

Representantes no comitê de controle de mudanças

O coordenador do comitê de controle de mudanças é o gerente de mudanças, que tem, entre outras, a tarefa de atuar como mediador entre as partes em caso de conflito e negociar decisões com as respectivas partes. Além disso, o gerente de

O papel do gerente de

mudanças é responsável também pela comunicação e documentação das decisões. mudanças

### 8.6.3 A Solicitação de Mudança

Para que se possam gerenciar as mudanças de requisitos durante a engenharia de requisitos, elas devem ser documentadas de forma apropriada para essa finalidade. Uma solicitação de mudança documenta a mudança desejada e contém informações adicionais para o gerenciamento da solicitação de mudança.

Template para solicitações de mudanças

Uma solicitação de mudança deve conter as seguintes informações:

Informações de mudanças

- *Identificador*: O identificador possibilita identificar de forma única uma solicitação de mudança em qualquer ponto ao longo do ciclo de vida do sistema.
- *Título*: O título resume o tema principal da solicitação de mudança em uma breve afirmação.
- *Descrição*: A descrição documenta a solicitação de mudança da forma mais precisa possível, podendo também conter informações sobre o efeito das mudanças.
- *Justificativa*: Aqui são listadas as mais importantes razões que tornam a mudança necessária.
- *Data da solicitação*: A data na qual a solicitação foi protocolada.
- *Solicitante*: O nome da pessoa que solicitou a mudança.
- *Prioridade (do ponto de vista do solicitante)*: A importância da solicitação de mudança, do ponto de vista do solicitante.

Além das informações acima, as seguintes informações são úteis para o gerenciamento das mudanças de requisitos:

Informações para o gerenciamento da solicitação de mudança

- *Avaliador da mudança*: A pessoa que verifica se a mudança foi corretamente realizada.
- *Status da análise de impacto*: Sinaliza se uma análise de impacto da solicitação de mudança já foi realizada.
- *Status da decisão do comitê de controle de mudanças*: Sinaliza se o comitê já tomou uma decisão sobre a solicitação de mudança.

- *Prioridade do comitê de controle de mudanças:* Documenta a prioridade atribuída pelo comitê para a solicitação de mudança.
- *Responsável:* Documenta a pessoa encarregada de realizar a mudança de requisito.
- *System release:* Documenta em qual *release* o requisito modificado será implementado.

#### 8.6.4 Classificação das Solicitações de Mudanças Recebidas

Uma vez protocolada, a solicitação de mudança é classificada pelo gerente de mudanças e pelo comitê de controle de mudanças. Tipicamente o gerente de mudanças classifica previamente as solicitações de mudanças recebidas, as quais serão então apresentadas, adaptadas (caso necessário) e, por fim, aprovadas (ou rejeitadas) na próxima reunião do comitê de controle de mudanças. Uma solicitação de mudança pode ser classificada em uma das seguintes três categorias:

- *Mudança corretiva de requisito:* Uma solicitação de mudança é classificada como corretiva se o motivo para a solicitação de mudança for uma falha do sistema durante sua operação, sendo que essa falha pode ser atribuída a um erro nos requisitos.
- *Mudança adaptativa de requisito:* Uma solicitação de mudança é classificada como adaptativa se uma mudança solicitada exige uma complementação ao sistema. Uma eventual razão para uma solicitação de mudança adaptativa pode ser uma mudança no contexto do sistema, por exemplo, a disponibilidade de uma nova tecnologia, ou uma alteração no limite do sistema (ver seção 2.2).
- *Mudança excepcional (hotfix):* Uma solicitação de mudança é classificada como uma mudança excepcional se a mudança precisa ser feita imediatamente a qualquer custo. Mudanças excepcionais podem ser tanto corretivas quanto adaptativas.

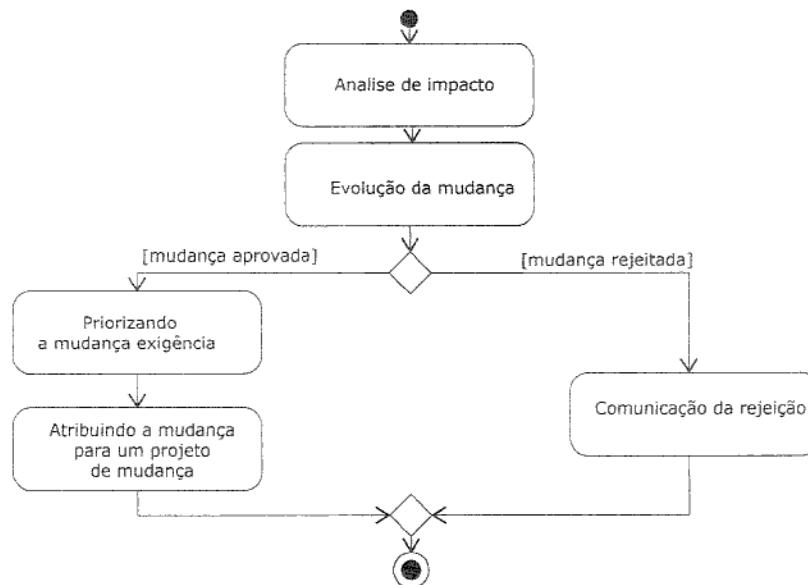
O método para processar solicitações de mudanças de requisitos depende de sua classificação. Por exemplo, mudanças excepcionais devem ser analisadas, avaliadas, decididas e eventualmente implementadas imediatamente. Mudanças adaptativas de requisitos, ao contrário, são geralmente processadas mais tarde, em lotes, tipicamente assim que a próxima (ou subsequente) *system release* estiver iminente. Por outro lado, mudanças corretivas de requisitos costumam ser analisadas, avaliadas e implementadas, caso necessário, de forma relativamente rápida depois que a solicitação de mudança foi protocolada.

Mudanças corretivas, adaptativas e excepcionais

Diferentes métodos de processamento

### 8.6.5 Método Básico para Mudanças Corretivas e Adaptativas

A figura 8-11 ilustra o principal método para processar solicitações de mudanças. Esse método pode ser adaptado para as respectivas particularidades de organizações e projetos específicos.



**Figura 8-11** Método para processar solicitações de mudanças

Durante a análise de impacto, estima-se o esforço necessário para realizar a mudança. Para tal, todos os requisitos afetados pela mudança são selecionados, incluindo quaisquer requisitos recentemente definidos. A seguir, os artefatos posteriores de desenvolvimento que eventualmente terão de ser modificados ou novamente desenvolvidos são identificados (por exemplo, casos de teste ou componentes). Para cada artefato afetado, o esforço para implementar a mudança é determinado, sendo o esforço total para a mudança calculado a partir da soma de todos os esforços parciais.

Análise de impacto

A integração consistente das mudanças na base de requisitos geralmente apenas influencia o esforço total de forma insignificante. A parcela mais significativa do esforço total costuma ser gerada pelas adaptações necessárias aos artefatos posteriores de desenvolvimento.

A identificação daqueles requisitos e artefatos posteriores de desenvolvimento que serão afetados por uma mudança de requisito pode ser automatizada, ou pelo menos apoiada, pelas informações de rastreabilidade. Caso nenhuma das informações de rastreabilidade necessárias (ou nem todas) estejam disponíveis, especialistas do domínio ou especialistas de desenvolvimento podem ser consultados sobre as consequências da solicitação de mudança recebida.

Uma vez completada a análise de impacto, o comitê de controle de mudanças avaliará a solicitação recebida. Para tal, o custo e o benefício são comparados e avaliados em relação aos recursos disponíveis. Por exemplo, o benefício da mudança pode ser a possibilidade de evitar perda de prestígio, de melhorar a posição no mercado, ou de evitar penalidades contratuais.

No passo seguinte, as mudanças aprovadas serão priorizadas pelo comitê de controle de mudanças. Mais tarde, as mudanças de requisitos são alocadas para um projeto de modificação, ou para a próxima *release* (ou outra subsequente) para implementação.

O planejamento, o controle da implementação e a validação das mudanças aplicadas com êxito são tipicamente responsabilidade do gerente de mudanças ou do comitê de controle de mudanças, podendo, naturalmente, ser delegadas a terceiros.

Usar as informações de rastreabilidade

Avaliar uma mudança

Implementar as mudanças aprovadas

Validar as mudanças de requisitos

## 8.7 Resumo

O gerenciamento de requisitos é uma atividade central da engenharia de requisitos. Essa atividade tem por objetivo (1) assegurar a constante disponibilidade dos requisitos documentados, bem como de outras informações relevantes, ao longo de todo o ciclo de vida do sistema ou do produto, (2) estruturar essas informações de forma apropriada (por exemplo, através de atributos de requisitos) e (3) assegurar o acesso seletivo a essas informações. O gerenciamento de requisitos engloba técnicas nas seguintes categorias:

- *Designar atributos aos requisitos:* Para permitir o gerenciamento de requisitos, as propriedades dos requisitos são documentadas por meio de atributos de requisitos.
- *Priorizar requisitos:* Requisitos são priorizados em diferentes momentos, durante diferentes atividades e conforme diferentes critérios. Dependendo do objetivo da priorização e do objeto a ser priorizado, diferentes técnicas de priorização devem ser utilizadas.
- *Rastreabilidade dos requisitos:* Durante o gerenciamento de requisitos, informações de rastreabilidade dos requisitos são registradas, organizadas e mantidas para que possam ser utilizadas informações sobre referências cruzadas e dependências entre requisitos, ou entre requisitos e outros

artefatos de desenvolvimento.

- *Versionamento de requisitos:* O versionamento e a configuração de requisitos possibilitam a disponibilização de informações sobre estados específicos de desenvolvimento de requisitos e documentos de requisitos ao longo do ciclo de vida do sistema ou do produto.
- *Gerenciamento das mudanças de requisitos:* Geralmente o comitê de controle de mudanças é responsável pelo processamento das solicitações de mudanças. O comitê de controle de mudanças decide se uma solicitação de mudança é aprovada ou rejeitada, e determina sua prioridade. O comitê também realiza uma análise de impacto para estimar o impacto da mudança em todos os requisitos e artefatos de desenvolvimento, bem como os recursos necessários para implementar a mudança.

## 9 Apoyo por Ferramenta

As diferentes atividades da engenharia de requisitos devem ser apoiadas por ferramentas apropriadas, idealmente integrando e continuando a processar as informações existentes. Essas informações tanto podem ter sido geradas durante a engenharia de requisitos (por exemplo, requisitos formulados em linguagem natural ou baseados em modelos) quanto podem ter sido usadas como base para os requisitos (por exemplo, atas de reuniões, documentos de metas, listas de *stakeholders*). Na prática, as ferramentas mais conhecidas da engenharia de requisitos são as ferramentas que oferecem apoio para o gerenciamento de requisitos (ver capítulo 8). Este capítulo trata principalmente das ferramentas para gerenciamento de requisitos. Além das ferramentas para gerenciamento de requisitos, também existem ferramentas de apoio para a elicitação, documentação, negociação e validação de requisitos.

### 9.1 Apoyo por Ferramenta: Visão Geral

Inúmeras ferramentas utilizadas durante o desenvolvimento do sistema também podem ser empregadas durante a engenharia de requisitos. Nesse sentido, ferramentas para o gerenciamento de testes, o rastreamento de "bugs" ou o gerenciamento de configurações muitas vezes oferecem a possibilidade de gerenciar requisitos, ou podem ser estendidas para essa função. Uma vantagem de usar tais ferramentas para o gerenciamento de requisitos é que os requisitos podem ser bem integrados aos artefatos originalmente criados pelas próprias ferramentas, tais como casos de teste ou solicitações de mudanças. Por exemplo, se requisitos são gerenciados por uma ferramenta de gerenciamento de teste, e não por uma ferramenta distinta para gerenciamento de requisitos, pode-se omitir uma interface entre duas ferramentas, o que simplifica muito o rastreamento de casos de teste e seus respectivos requisitos.

Ferramentas *wiki* também são empregadas hoje em dia como apoio para a engenharia de requisitos. Por exemplo, glossários podem ser elaborados de forma colaborativa, ou requisitos de sistema podem ser trabalhados de forma participativa. As ferramentas *wiki* mostraram ser muito úteis na prática, especialmente no caso de sistemas com grande número de *stakeholders*.

Ferramentas utilizadas durante o desenvolvimento do sistema podem ser empregados durante a engenharia de requisitos

Apoio por ferramentas tipo *wiki*

Outros tipos de ferramentas podem ajudar a aumentar a efetividade e a eficiência da engenharia de requisitos. Mapas mentais elaborados durante sessões de *brainstorming* podem servir como um material estruturador auxiliar. Ferramentas de apresentação podem ajudar a esboçar um conceito de análise. Se protótipos são utilizados, ferramentas de simulação ou ambientes de teste podem ajudar a simular a operação do sistema. Ferramentas para projetar protótipos de interface gráfica com o usuário (protótipos GUI, *Graphical User Interface*) ou ambientes de desenvolvimento podem ilustrar interfaces com usuários e funções, bem como servir de base para discussões. Ferramentas de fluxogramas e programas de visualizações podem ser utilizados para gerar diferentes diagramas e gráficos.

Além disso, ferramentas comuns em cenários de trabalho cotidianos, tais como o pacote de aplicativos *Office* podem ser bem aproveitados na engenharia de requisitos. Programas de e-mail, *chat*, agendas de endereços, aplicativos de calendário e plataformas de *groupware*, bem como ferramentas para gerenciamento, planejamento e controladoria de projetos, são ferramentas de trabalho utilizadas diariamente que podem auxiliar a engenharia de requisitos. Essas ferramentas oferecem apoio aos *stakeholders* na comunicação, planejamento e coordenação de suas tarefas.

Ferramentas para estruturar, apresentar, visualizar e estimular

Ferramentas para gerenciamento de projetos e de comunicação, aplicativos de escritório

## 9.2 Ferramentas de Modelagem

Além de serem formuladas em linguagem natural, as informações na engenharia de requisitos também são documentadas em modelos, que podem ser gerados com ferramentas de modelagem (ver capítulo 6). Essas ferramentas não apenas oferecem a possibilidade de criar os modelos, como também permitem analisar os modelos em termos de exatidão sintática.

Ao escolher ferramentas de modelagem, é importante seguir critérios semelhantes aos critérios para ferramentas especializadas de engenharia de requisitos (ver seção 9.5). A ferramenta de modelagem deverá fornecer uma identidade única para cada elemento do modelo, possibilitando a rastreabilidade entre os diferentes modelos e permitindo sua manipulação por múltiplos usuários. Além disso, ferramentas de modelagem devem oferecer algum tipo de funcionalidade para controle das versões em relação aos modelos e aos elementos dos modelos.

Um importante aspecto relacionado à aplicação de diferentes ferramentas é a integração e a rastreabilidade entre artefatos das diferentes ferramentas (casos de uso, modelos de comportamento e casos de teste). A escolha pela ferramenta de modelagem ou pela ferramenta de gerenciamento de requisitos deve ser realizada tendo em vista a interface entre as duas ferramentas. Isso significa que uma interface já deveria estar presente ou ser de fácil criação. Tal interface deve permitir mudanças de rastreamento em modelos e/ou requisitos, bem como o gerenciamento

Rastreabilidad e entre múltiplas ferramentas

da rastreabilidade entre modelos e requisitos (ver capítulo 8). Se ocorrer alguma mudança nos requisitos, é indispensável fazer as necessárias modificações nos elementos do modelo associado também. Da mesma forma, se um modelo mudar, as modificações necessárias devem ser integradas aos requisitos de linguagem natural também.

### 9.3 Ferramentas para Gerenciamento de Requisitos

Para fornecer apoio às técnicas de gerenciamento (como descritas no capítulo 8) de maneira otimizada, uma ferramenta para gerenciamento de requisitos deve possuir as seguinte propriedades básicas:

- Gerenciar diversos tipos de informações (requisitos em linguagem natural, modelos conceituais, esboços, planos de teste, solicitações de mudanças).
- Gerenciar relacionamentos lógicos entre as informações (rastreabilidade entre requisitos ou entre requisitos e suas implementações).
- Permitir a identificação única de cada artefato gerenciado.
- Editar as informações gerenciadas (acessibilidade de múltiplos usuários, controle de acesso, gerenciamento de configurações e versões).
- Permitir diferentes visualizações das informações gerenciadas, dependendo da finalidade.
- Organizar as informações gerenciadas (agrupar, estruturar hierarquicamente, designar atributos, anotações de informações adicionais).
- Gerar relatórios ou resumos sobre as informações gerenciadas (por exemplo, relatórios sobre solicitações de mudanças dos requisitos).
- Gerar diferentes tipos de documentos de saída a partir das informações gerenciadas (por exemplo, gerar documentos de requisitos para uma *system release* específica).

Propriedades necessárias das ferramentas

Dependendo do número de funções e da abrangência das funções básicas, as ferramentas de gerenciamento de requisitos podem ser classificadas de duas maneiras:

- Ferramentas especializadas.
- Aplicativos padrão de escritório.

### 9.3.1 Ferramentas Especializadas para Gerenciamento de Requisitos

Ferramentas nessa categoria foram desenvolvidas especificamente como apoio para as técnicas de gerenciamento de requisitos, controlando toda e qualquer atividade associada ao gerenciamento. As propriedades de tais ferramentas são (ver capítulo 8):

- Gerenciamento de requisitos e atributos a partir de modelos de informação.
- Organização de requisitos (por meio de níveis hierárquicos).
- Gerenciamento de configurações e versões a nível de requisito.
- Definição de *baselines* de requisitos.
- Acessibilidade e gerenciamento de múltiplos usuários (controle de acesso).
- Gerenciamento de rastreabilidade.
- Consolidação dos requisitos elicitados (geração de visualizações).
- Apoio para gerenciamento de mudanças (controle de mudanças).

Propriedades das ferramentas para gerenciamento de requisitos

As diversas ferramentas para gerenciamento de requisitos disponíveis no mercado apresentam uma estrutura semelhante. As ferramentas mais comuns possuem uma interface com a qual o usuário pode acessar todas as funções necessárias para realizar as tarefas do gerenciamento de requisitos. Os dados gerenciados são armazenados num banco de dados e podem ser editados por meio de um editor integrado. Diferentes funções de importação e exportação de documentos asseguram que os dados importados de sistemas externos podem ser lidos pela ferramenta de gerenciamento de requisitos e que os dados exportados podem ser lidos por sistemas externos.

Arquitetura de ferramentas para gerenciamento de requisitos

Tendo em vista que essas ferramentas atendem à maior parte das funções básicas do gerenciamento de requisitos, elas são muito indicadas para gerenciar as informações relevantes da engenharia de requisitos. Uma visão geral dos produtos disponíveis no mercado que fornecem apoio para a engenharia de requisitos pode ser encontrada, por exemplo, no website do INCOSE (*International Council of Systems Engineering*) e do método Volere.

Adequação das ferramentas para gerenciamento de requisitos

### 9.3.2 Aplicativos Padrão de Escritório

Em muitos projetos, aplicativos padrão de escritório ainda são usados para gerenciar requisitos (processamento de textos e planilhas, por exemplo). As

principais razões para tal incluem, por um lado, a ampla disseminação desses aplicativos, e por outro lado, o fato de não exigirem um esforço adicional para familiarizar-se com eles. Quando usados em conjunto com *templates* – tais como os *templates* para documentação de requisitos (ver seção 5.2) – esses aplicativos são adequados para documentar e, até certo ponto, gerenciar requisitos (relacionamentos de rastreabilidade, por exemplo, podem ser estabelecidos com *hyperlinks*).

Porém, essas ferramentas apoiam as funções básicas do gerenciamento de requisitos apenas de forma limitada. Elas não fornecem um mecanismo de controle de versões a nível de requisitos, nem possuem funcionalidades de apoio para técnicas específicas de gerenciamento de requisitos (por exemplo, a possibilidade de manter relacionamentos de rastreabilidade entre artefatos individuais de forma automatizada). Algumas das funções básicas podem ser emuladas com outras ferramentas. Por exemplo, um aplicativo de escritório utilizado em conjunto com alguma ferramenta para controle de versões pode atender aos requisitos de controle ativo de versões, ou gerenciamento de acesso para múltiplos usuários. Entretanto, a produtividade e o desempenho em termos de gerenciamento de requisitos que podem ser alcançados com ferramentas especializadas não podem ser obtidos com aplicativos padrão de escritório.

Aplicativos de escritório oferecem apenas apoio limitado

## 9.4 Introduzindo Ferramentas

Antes de poder investir qualquer esforço na procura da ferramenta ideal de apoio para o gerenciamento de requisitos, é recomendável que as responsabilidades em relação à engenharia de requisitos já tenham sido atribuídas dentro da organização ou no projeto. Além das partes responsáveis, também as técnicas e processos necessários para alcançar o objetivo da engenharia de requisitos e do gerenciamento de requisitos (ver capítulo 8) devem ser definidos. Afinal, até a mais sofisticada ferramenta para gerenciamento de requisitos é apenas um instrumento de apoio para o engenheiro de requisitos e a engenharia de requisitos.

Atribuir responsabilidades

As ferramentas disponíveis somente poderão ser avaliadas após a definição de todos os processos e de todas as técnicas, e depois que todos os envolvidos sejam capazes de seguir essas restrições. Os seguintes aspectos devem ser considerados na escolha e introdução de ferramentas para engenharia de requisitos:

A ferramenta segue o método

- A escolha e introdução de ferramentas absorve recursos na organização. Isso vale não apenas para o pessoal encarregado da introdução de uma ferramenta, mas também para seus futuros usuários. Esses esforços devem ser considerados durante a avaliação.

Considerar os recursos necessários

- Na prática, é problemático introduzir uma ferramenta quando um projeto de desenvolvimento já está em andamento. Enquanto que o esforço adicional para o treinamento de funcionários pode ser muito bem estimado, os riscos associados com a introdução de uma nova ferramenta enquanto o projeto está em andamento costumam ser subestimados. A resistência dos funcionários ou as deficiências da ferramenta que surgem em sua implementação podem influenciar negativamente o projeto. Tais riscos podem ser evitados com a introdução de novas ferramentas em projetos piloto, nos quais devem ser incluídos recursos adicionais para a introdução da ferramenta, treinamento de pessoal e customização do processo. Projeto piloto
- Uma ferramenta apropriada deve ser escolhida em um processo de avaliação de ferramenta. Uma vez pesquisados os fabricantes e definidos os critérios críticos "*must-have*", potenciais candidatos para a introdução podem ser selecionados e investigados em maiores detalhes. Para tal, é necessário criar um catálogo de critérios que descrevem quais requisitos uma ferramenta para engenharia de requisitos deve atender. As ferramentas a serem avaliadas podem então ser classificadas conforme esses requisitos. Avaliação
- Os custos para uma ferramenta geralmente vão além do simples custo de licenciamento. Tipicamente, também devem ser considerados os custos para o treinamento de pessoal e para uma eventual customização da ferramenta, bem como os custos de apoio. Custos
- É necessário que os futuros usuários da ferramenta conheçam, formarem ativamente e dominem os processos e atividades que irão enfrentar durante a engenharia de requisitos. Os usuários devem receber treinamento em termos de processos, técnicas e o respectivo apoio por ferramentas. Instrução de pessoal

## 9.5 Avaliação de Ferramentas

Devido aos muitos tipos diferentes de ferramentas disponíveis, avaliar ferramentas em termos de sua adequação como apoio para a engenharia de requisitos é uma tarefa muito cansativa e desafiadora na prática.

Para avaliar as ferramentas da forma mais objetiva possível, é recomendável adotar diferentes perspectivas em relação às ferramentas na engenharia de requisitos. Definir diferentes perspectivas sobre ferramentas possibilita analisar a adequação de uma ferramenta de forma sistemática e priorizar os requisitos da ferramenta individualmente. A figura 9-1 apresenta algumas perspectivas que poderiam ser utilizadas para avaliar a adequação de uma ferramenta na engenharia de requisitos.

Perspectivas sobre ferramentas na engenharia de requisitos

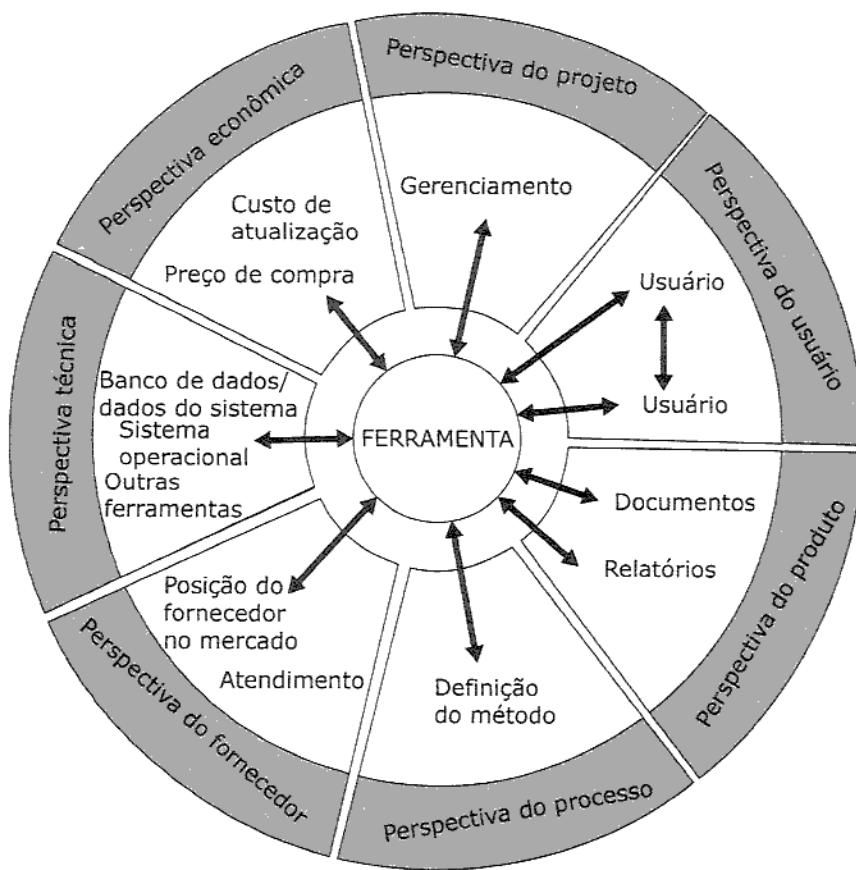


Figura 9-1 Perspectivas sobre uma ferramenta de engenharia de requisitos

Para cada perspectiva devem ser definidos critérios sob medida para os aspectos centrais da respectiva perspectiva.

### 9.5.1 Perspectiva do Projeto

A perspectiva do projeto mostra até que ponto a ferramenta pode fornecer apoio para o projeto. Os critérios relevantes são o apoio durante a preparação, o planejamento e a execução do projeto. Para a preparação do projeto, critérios relacionados com a definição de tipos de informações e documentos específicos para o projeto podem ser considerados. Para o planejamento do projeto, critérios para definir o escopo de etapas e para determinar como as informações e

Apoio para o projeto

documentos criados a partir da ferramenta estão relacionados com as etapas. A execução do projeto inclui critérios relacionados ao controle e gerenciamento do projeto com base nas informações e documentos criados com a ferramenta.

### 9.5.2 Perspectiva do Usuário

A perspectiva do usuário considera os requisitos para a ferramenta que surgem a partir da perspectiva dos usuários (por exemplo, capacidade para múltiplos usuários). A avaliação a partir da perspectiva do usuário tem como foco a utilização da ferramenta, o mapeamento de papéis e o apoio ao trabalho em equipe. Mais precisamente, isso significa que os diferentes *stakeholders* envolvidos em um projeto de desenvolvimento devem estar adequadamente mapeados para o gerenciamento apropriado dos usuários e dos direitos de acesso. Isso permite aos usuários o devido acesso às funções da ferramenta e às informações armazenadas, de acordo com seus respectivos papéis.

Perspectiva dos futuros usuários

### 9.5.3 Perspectiva do Produto

A perspectiva do produto engloba as funcionalidades que a ferramenta possui (por exemplo, diferentes tipos de documentação para requisitos). Entre outros aspectos, esta perspectiva considera os tipos de documentos, visualizações e relatórios que podem ser gerados, bem com a rastreabilidade entre os produtos selecionados.

Funcionalidades da ferramenta

### 9.5.4 Perspectiva do Processo

A perspectiva do processo concentra-se no apoio metodológico oferecido pela ferramenta (por exemplo, eventuais orientações, manutenção de relacionamentos de rastreabilidade). Considerações da perspectiva do processo incluem a capacidade de documentar atividades no interior da ferramenta, bem como determinar em que medida a ferramenta oferece orientações metodológicas. Em termos de orientações metodológicas, pode-se distinguir entre diferentes graus de obrigatoriedade. A orientação pode ser de natureza estrita e restritiva, ou então pode oferecer sugestões e dicas mais flexíveis. Além do apoio metodológico oferecido pela ferramenta, também pode-se avaliar nesta perspectiva em que medida um modelo de processo específico para o projeto pode ser definido.

Apoio metodológico oferecido pela ferramenta

### 9.5.5 Perspectiva do Fornecedor

A perspectiva do fornecedor considera a posição de mercado, bem como os diversos serviços oferecidos por determinado fabricante. Ao escolher uma ferramenta, não somente os aspectos funcionais são pertinentes, mas também as restrições que devem ser cumpridas para que a ferramenta possa ser aplicada. Aspectos como o reconhecimento de marca, por exemplo, e a reputação do

Posição de mercado do fabricante e suporte fornecido

fornecedor são frequentemente utilizados como critérios de decisão. Tendo em vista o custo de aquisição relativamente alto, além dos contratos de longo prazo para serviços de assistência, a decisão por determinada ferramenta implica num relacionamento muito estreito com o fornecedor.

### 9.5.6 Perspectiva Técnica

A perspectiva técnica envolve condições técnicas do contexto que o sistema deve satisfazer. Aspectos importantes na perspectiva técnica são, por exemplo, a interoperabilidade da ferramenta, o desempenho do repositório utilizado, os *hardwares* e *softwares* necessários, a escalabilidade da ferramenta. A interoperabilidade da ferramenta pode ser determinada, por exemplo, investigando até que ponto as funcionalidades da ferramenta são acessíveis via API e em que medida a integração de processos, dados e controles é possível. A escalabilidade da ferramenta pode ser determinada, por exemplo, definindo o número máximo de usuários ou de objetos (por exemplo, pacotes de conteúdo ou documentos) que podem ser suportados. O desempenho do repositório utilizado pode ser medido pelo volume possível de importação e exportação de dados, o desempenho das interfaces de consulta ou os conceitos de segurança disponíveis.

A capacidade da ferramenta em termos de desempenho e interoperabilidade

### 9.5.7 Perspectiva Econômica

A perspectiva econômica analisa os possíveis custos relacionados com a aquisição, introdução e manutenção de uma ferramenta (por exemplo, os custos de licenciamento, treinamento de pessoal, custos de apoio). O valor dos custos relevantes pode consistir dos custos de integração, operação, manutenção e infraestrutura, bem como os custos para customização metodológica e os custos de aquisição.

Custos de introdução e acompanhamento

## 9.6 Resumo

Ao gerenciar requisitos durante a engenharia de requisitos, é necessário armazenar as informações de maneira a satisfazer os critérios de qualidade para o gerenciamento de requisitos. Ferramentas oferecem apoio para o engenheiro de requisitos nessa tarefa. Essas ferramentas podem ser divididas em ferramentas profissionais para gerenciamento de requisitos, ferramentas de modelagem e aplicativos padrão de escritório, diferenciando-se entre si pelas funcionalidades oferecidas ao engenheiro de requisitos. Por esse motivo, é preciso realizar uma avaliação cuidadosa antes de escolher uma ferramenta, para não impactar desnecessariamente o processo de introdução da ferramenta.

# Fundamentos da Engenharia de Requisitos

Um Guia de Estudo para o  
Exame CPRE-FL Certified Professional for Requirements Engineering – Foundation  
level, em conformidade com o padrão IREB

Klaus Pohl · Chris Rupp

## Sobre o IREB

As atividades e práticas da engenharia de requisitos vêm se tornando cada vez mais complexas. Para assegurar um alto nível de conhecimento e competência entre engenheiros de requisitos, a organização *International Requirements Engineering Board* (IREB) desenvolveu uma certificação padronizada denominada *Certified Professional for Requirements Engineering* (CPRE). Essa certificação define, em vários níveis de treinamento, as habilidades requeridas de um engenheiro de requisitos.

Este livro destina-se ao autoaprendizado e cobre todo o currículo exigido pelo exame *Certified Professional for Requirements Engineering – Foundation Level*, conforme definido pelo IREB.

A missão do *International Requirements Engineering Board* é contribuir para a padronização e educação nas áreas de análise de negócios e engenharia de requisitos provendoementas de conhecimento (*syllabi*) e exames, destarte melhorando e aperfeiçoando a aplicação da engenharia de requisitos.

Os membros do *Board* do IREB são especialistas independentes e de renome internacional nas áreas de economia, consultoria, pesquisa e ciência. O IREB é uma entidade sem fins lucrativos.

Para mais informações visite  
[www.certified-re.com](http://www.certified-re.com)

## Sobre a T&M

A missão da T&M é fornecer sólida expertise em Engenharia de Requisitos e Engenharia de Testes , através de serviços, consultoria, capacitação e certificação inovadores e de alto valor agregado, com foco na expertise, disciplina e governança da qualidade.

Para mais informações visite  
[www.tmtestes.com.br](http://www.tmtestes.com.br)

A T&M recomenda o IBQTS como órgão certificador CPRE-FL

