UMA METODOLOGIA PARA ENGENHARIA DE REQUISITOS PARA PEQUENAS EQUIPES DE DESENVOLVIMENTO DE SOFTWARE

João Alexandre Bonin de Mello[1]

RESUMO: Este artigo propõe uma metodologia para engenharia de requisitos simplificada, para ser utilizada em pequenas equipes. As fases da metodologia são: 1) levantar os requisitos de negócio; 2) classificar quantitativamente estes requisitos de acordo com a importância de cada um deles para o cliente; 2) a partir dos requisitos de negócio, elaborar a especificação de requisitos de software; 3) desenvolver os casos de uso de acordo com os requisitos de software; 4) elaborar uma matriz de dependência de requisitos; 5) Utilizar a matriz de dependência de requisitos para planejar as liberações, levando-se em conta a importância de cada requisito para o cliente; 6) Especificar os requisitos; 7) Validar os requisitos.
PALAVRAS-CHAVE: requisitos de software; engenharia de software; gestão de requisitos; metodologias de desenvolvimento de software.

A METODOLOGY OF REQUEREMENTS ENGENEERUNG FOR SMALL TEAMS OF SOFTWARE DEVELOPMENT

ABSTRACT: This article has the purpose provide a simplified methodology for software requirements engineering for use in small teams. The phases of the methodology are: 1) to raise the business requirements; 2) to classify quantitatively these requirements in accordance with the importance of each one of them for the customer; 2) to elaborate a specification of software in accordance with the business requirements; 3) to develop the use cases in accordance with the software requirements; 4) to elaborate a matrix of dependence of requirement; 5) to use a matrix of dependence of requirement to plan the releases, taking in account the importance of each requirement to the costumer; 6) to specify the requirements; 7) to validate the requirements.
KEY WORDS: software requirements; software engineering; requirements; software development methodologies.

1. Introdução

Os requisitos estão associados à boa parte dos defeitos injetados em softwares. Em geral, as metodologias de desenvolvimento de software são intensivas no uso de mão-de-obra, o que obriga o uso de ferramentas especializadas

[1]Professor adjunto da Universidade Paranaense UNIPAR - Mestre em Engenharia de Produção pela UFSC - alexandre@unipar.br

e de alto custo, sendo assim, pequenas equipes de desenvolvimento de software acabam por ficar à margem das melhores práticas de levantamento e gestão de requisitos. A metodologia proposta, neste artigo, procura simplificar a atividade de engenharia de requisitos para torná-la viável a pequenas equipes.

2. Requisitos e engenharia de requisitos

Os requisitos estabelecem um conjunto de características necessárias à aceitação de um software por parte do cliente, descrevendo quais atividades o software deve desempenhar, quais suas limitações e restrições, além de outras características não ligadas diretamente às funções desempenhadas pelo software, como metas de qualidade (CRISTEL e KANG, 2003, BLASCHEK, 2002, PAULA, 2002, PETERS e PEDRYCZ, 2001, PRESSMAN, 2001, QUEDAS, 2003).

A engenharia de requisitos fornece um processo adequado para entender as necessidades do cliente, negociar uma solução possível, descrever os requisitos de forma clara e concisa e gerenciar as mudanças que ocorrem ao longo do ciclo de vida do software (CRISTEL e KANG, 2003, BLASCHEK, 2002, PAULA, 2002, PRESSMAN, 2001, QUEDAS, 2003). O processo de engenharia de requisitos pode ser descrito como um processo interativo e continuado de entendimento das necessidades do usuário e tradução destas necessidades em um futuro produto de software. Este processo pode ser dividido em cinco atividades (PRESSMAN, 2001, BLASCHEK, 2002):

a.levantamento (ou elicitação) de requisitos, que envolve a coleta organizada dos requisitos de negócio;
b.análise e negociação de requisitos, que procura catalogar e classificar os requisitos em subconjuntos, negociar prazos de liberações de acordo com a importância dos requisitos para o cliente;
c.especificação de requisitos, que documenta a funcionalidade do software, metas de qualidade e restrições que irão servir de base para o processo de desenvolvimento de software;
d.validação de requisitos, que verifica se todos os requisitos do sistema foram declarados de forma não ambígua e em sintonia com as necessidades do cliente; e,
e.gestão de requisitos, que procura identificar, controlar e rastrear requisitos e modificações de requisitos a qualquer momento no ciclo de vida do software.

3. A relação entre requisitos e defeitos em softwares

A engenharia de requisitos é a etapa mais crítica do processo de desenvolvimento de software e a que injeta os defeitos mais difíceis de serem

corrigidos em fases posteriores do ciclo de vida do software (KRISTEL e KANG, 2003). Se atribuirmos o custo de $1,00 para uma mudança em um software durante a fase de requisitos, este custo cresce 1,5 a 6 vezes durante a fase de desenvolvimento e de 60 a 100 vezes, do custo original, após a entrega do produto ao cliente (PRESSMAN, 2001).

Freqüentemente, os requisitos estão incompletos, inconsistentes, são instáveis, ou foram levantados, mas não foram implementados. Os principais problemas detectados nos requisitos são (BLASCHEK, 2002, MARQUIONI, 2003, PAULA, 2002, PRESSMAN, 2001):

a.problemas de escopo: os limites do sistema são mal definidos;
b.problemas de entendimento: os clientes não estão certos de suas necessidades ou omitem informações, que acreditam ser óbvias; por outro lado, os desenvolvedores têm dificuldade em entender as necessidades do cliente;
c.problemas de instabilidade: os requisitos mudam durante o ciclo de desenvolvimento e os desenvolvedores têm dificuldades de administrar a mudança;
d.problemas de rastreamento: os requisitos foram levantados, mas não implementados devido à falta de um método de rastreamento dos mesmos durante o desenvolvimento.
Boa parte destes problemas poderia ser eliminada se as organizações dispusessem de um processo formal de engenharia de requisitos que permita (BLASCHEK, 2002, PAULA, 2002, QUEDAS, 2003):
a.a delimitação clara do escopo do sistema;
b.a compreensão dos requisitos, tanto por parte dos desenvolvedores quanto por parte dos clientes;
c.o rastreamento desses requisitos em qualquer fase do ciclo de vida do software;
d.o estabelecimento de relações de dependência entre os requisitos e entre estes e os demais artefatos produzidos no processo de desenvolvimento;
e.um método quantitativo de classificação dos requisitos, de acordo com a importância dos mesmos para o cliente, visando à negociação de prazos e das funcionalidades implementadas em cada liberação do produto;
f. a validação dos requisitos por meio de revisões formais, a fim de detectar possíveis problemas nos requisitos antes do início da construção do software.

4. A metodologia

A metodologia proposta procura simplificar o processo de engenharia de requisitos, facilitando seu uso por pequenas equipes. Para isso, devem-se assumir algumas premissas da programação extrema, como a entrega em

pequenas liberações, a alta taxa de disponibilidade do cliente, o planejamento das liberações em conjunto do mesmo, com o objetivo de colocar-se em produção as funcionalidades de maior valor e o alto grau de comunicabilidade entre os membros da equipe. Estas premissas têm por objetivo reduzir o volume de documentação ao mínimo necessário a um bom entendimento dos requisitos do sistema.

O diagrama de atividades da FIGURA 1 apresenta as atividades propostas na metodologia. Estas atividades são discutidas em detalhes a seguir. Para tal, será utilizado como modelo um sistema fictício de controle de consumo de combustível em uma frota de veículos.

O processo começa com a determinação do objetivo central do sistema, em seguida, este objetivo é desdobrado em requisitos de negócio que descrevem as necessidades de informação do cliente em relação ao sistema. Cada requisito de negócio irá gerar um ou mais requisitos de softwares, que descrevem as funções que o sistema irá desempenhar para atender aos requisitos de negócio (requisitos funcionais), metas de qualidade, restrições tecnológicas e outras limitações (requisitos não funcionais). Os requisitos de negócio são então classificados quantitativamente, segundo sua importância para o cliente.

Após a classificação dos requisitos de negócio é estabelecida uma matriz de dependência de requisitos, esta matriz permite rastrear os requisitos para o planejamento das liberações, de acordo com a importância dos mesmos para o cliente, uma vez que a metodologia parte do princípio que o software será entregue ao cliente em uma série de pequenas liberações.

Para cada liberação, os casos de uso devem ser descritos de forma sucinta a fim de captarem-se todas as particularidades contidas nos processos atendidos pelo software e um modelo conceitual de classes deve ser elaborado para que se possa captar o modelo estático das informações controladas pelo sistema.

4.1. Atividades

4.1.1. Determinação do objetivo do sistema

O objetivo do sistema (QUADRO 1) sintetiza o valor que o sistema agregará ao negócio do cliente. Deve ser descrito em um parágrafo curto e delimitar as necessidades do produto (PAULA, 2002).

Quadro 1 - Exemplo de objetivo do sistema

Objetivo do produto:
Controlar o consumo de combustíveis de uma frota de veículos

4.1.2.Elicitação dos requisitos de negócio

Os requisitos de negócio determinam as necessidades de informação do cliente em relação ao produto de software a ser desenvolvido. O desenvolvedor, durante o desenvolvimento desta atividade, deve ater-se às necessidades de informação do cliente em relação ao software, detalhando o objetivo do sistema sem, contudo, detalhar os requisitos funcionais que o software deverá cumprir para atender estas necessidades. O detalha os requisitos de negócio para o sistema exemplo.

4.1.3.Determinação dos requisitos de software

Requisitos de software especificam as funcionalidades e demais características que o software deve possuir para atender aos requisitos de negócio. Estes podem ser classificados em funcionais, que descrevem as funções que o software deverá desempenhar para atender os requisitos de negócio, enquanto que os requisitos não funcionais descrevem atributos de qualidade, desempenho e restrições impostas pela tecnologia e outros fatores que poderão influenciar o produto. O, a seguir, ilustra os requisitos funcionais para o sistema.

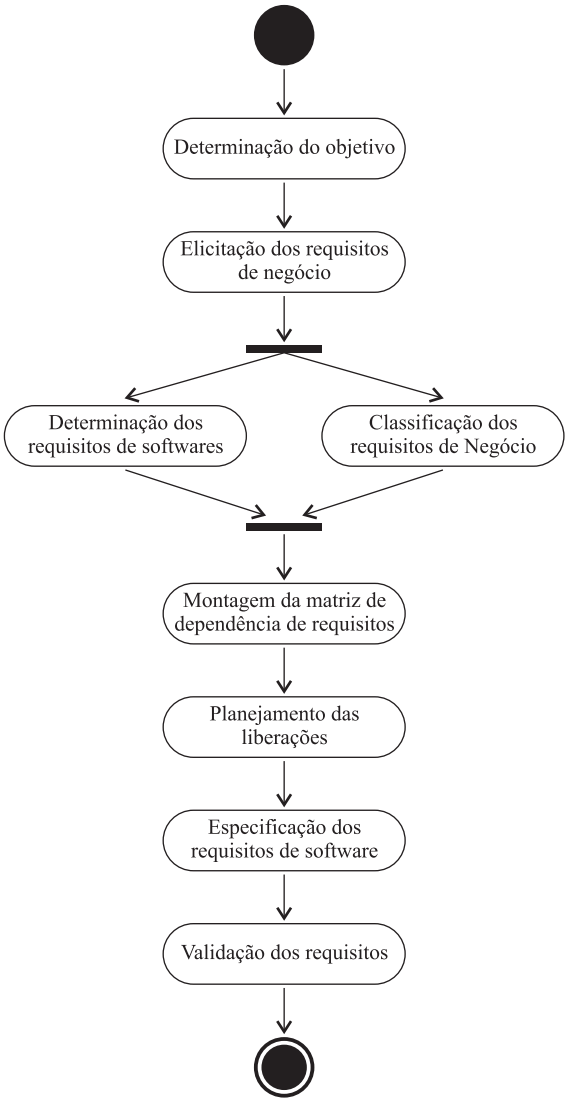

Figura 1 - Atividades da metodologia

Quadro 2 - Exemplo de requisitos de negócio

Número	Requisito	Detalhes
RN01	Conhecer os indicadores de consumo por veículo	Para um mesmo veículo deve-se conhecer o consumo total de combustível em um período de tempo, por quilômetro rodado e em comparação com o consumo médio dos veículos do mesmo modelo
RN03	Conhecer os indicadores de consumo por motorista	Comparar o consumo médio de combustível de cada motorista dirigindo o mesmo modelo de veículo para detectar possíveis vícios de direção e adotar atitudes corretivas
RN03	Medir a qualidade dos combustíveis por posto de abastecimento	Comparar o consumo médio de combustível de um mesmo modelo de veículo em diversos postos de abastecimento para detectar discrepâncias na qualidade dos combustíveis

Quadro 3 - Requisitos funcionais

Número	Requisito	Detalhes
RF01	Cadastramento de Veículos	Inclusão, alteração e baixa lógica de veículos
RF02	Cadastramento de Motoristas	Inclusão, alteração e baixa lógica de motoristas
RF03	Cadastramento de Postos de Combustível	Inclusão, alteração e baixa lógica de postos de combustível
RF04	Registro de Saída de Veículo	Registrar data, hora, placa, motorista e quilometragem do veículo no momento da saída
RF05	Registro de Chegada de Veículo	Registrar data, hora e quilometragem do veículo no momento da chegada. Deve haver uma saída anterior para que a chegada seja registrada
RF06	Registro de Abastecimentos	Registrar a data, a quilometragem, a placa do veículo e os dados da nota fiscal de venda de combustíveis.
RF07	Consulta ao consumo por veículo	Calcula o consumo total e a quantidade de quilômetros por litro de combustível em um período de tempo de um grupo de veículos selecionados. Gera um gráfico comparativo de consumo médio por veículo
RF08	Comparativo de consumo de combustível por motorista	Compara o consumo médio de combustível de diversos motoristas para um mesmo modelo de veículo em um determinado período de tempo
RF09	Comparativo de consumo por posto de combustível	Compara o consumo médio de combustível por posto de combustível para um mesmo modelo de veículo

Para cada requisito de negócio, normalmente, existirão um ou mais requisitos funcionais, por exemplo, para atender ao requisito de negócio RN01 – conhecer os indicadores de consumo por veículo -, são necessários os requisitos funcionais RF01 – cadastramento de veículos, RF06 – registro de Abastecimentos RF07 e consumo de combustível por veículo.

Normalmente, alguns desenvolvedores deixam para determinar as saídas do sistema (relatórios e consultas) para etapas posteriores do desenvolvimento. Na metodologia proposta, as saídas do sistema devem ser determinadas nesta atividade, pois estas consomem horas de trabalho e determinam as entradas do sistema. Esta atitude tem por objetivo melhorar o planejamento das liberações.

4.1.4.Classificação dos requisitos de negócio

Nesta atividade, os requisitos de negócio são classificados, quantitativamente, segundo sua importância para o usuário. Esta atividade se concentra na classificação dos benefícios que o software proporcionará e não nas funções que o mesmo executará e servirá de base para o planejamento das futuras liberações.

A metodologia para classificação dos requisitos de negócio baseia-se nos conceitos de análise de valor, aplicados pela primeira vez nos anos 40, na General Eletric, pela equipe do engenheiro Lawrence D. Miles, a qual quantifica a importância das funções que um determinado produto desempenha em benefício do cliente. Para efetuar a análise de valor será utilizado um método de avaliação numérica funcional, conhecido por Diagrama de Mudge, que determina uma hierarquia entre as funções do produto baseado em uma análise comparativa das funções duas a duas até que todas sejam comparadas entre si pelo cliente.(POSSAMAI, 1999).

Inicialmente, a Matriz de Mudge é preenchida com os requisitos de negócio (FIGURA 2).

		RN02	RN03	Total	%
Conhecer os indicadores de consumo por veículo	RN01				
Conhecer os indicadores de consumo por motorista		RN02			
Medir a qualidade dos combustíveis por posto de abastecimento			RN03		

Figura 2 - Matriz de mudge

Em seguida, o cliente responde às seguintes perguntas:

1. O requisito RN01 é mais importante que o requisito RN02?
2. Se a resposta for sim, coloca-se RN01 na matriz, caso contrário, coloca-se RN02 na matriz.
3. Em seguida, pergunta-se o quanto um requisito é mais importante que o outro e determina-se um peso ao requisito de maior importância de acordo com a seguinte pontuação:
   a. Peso 1 – pouco mais importante
   b. Peso 3 – moderadamente mais importante
   c. Peso 3 – muito mais importante

Para o preenchimento do exemplo fictício, supôs-se que o cliente respondeu às perguntas da seguinte maneira:

1. O requisito RN01 é moderadamente mais importante que o requisito RN02.
2. O requisito RN01 é moderadamente mais importante que o requisito RN03.
3. O requisito RN02 é pouco mais importante que o requisito RN03.

É importante ressaltar que, para se preencher a matriz, deve-se supor que todos os requisitos sejam atendidos pelo produto para evitar suposições do tipo "Se o requisito A não estiver presente no produto, o produto não funcionará, então o requisito B é muito mais importante que o requisito, portanto este requisito é mais importante".

Após o preenchimento da matriz, somam-se os pesos associados a cada requisito e preenche-se a coluna Total. Em seguida, calcula-se o percentual de importância de cada requisito ( ).

		RN02	**RN03**	**Total**	**%**
Conhecer os indicadores de consumo por veículo	**RN01**	RN01-2	RN01-2	4	80%
Conhecer os indicadores de consumo por motorista		**RN02**	RN02-1	1	20%
Medir a qualidade dos combustíveis por posto de abastecimento			**RN03**	0	0%

Figura 3 - Matriz de mudge preenchida e totalizada

Analisando-se a matriz preenchida, conclui-se que o requisito RN01 tem 80% de importância para o cliente, sendo essencial para o sucesso do produto, enquanto que o requisito RN02 tem 20% de importância para o cliente. O requisito RN03 obteve 0% de importância para o cliente, isso não quer dizer que o mesmo seja desnecessário, mas que este é menos importante que todos os outros requisitos do sistema.

4.1.5.Montagem da matriz de dependência dos requisitos

A matriz de dependência de requisitos permite o rastreamento de cada um dos requisitos funcionais em relação aos requisitos de negócio e vice-versa. A matriz é montada, primeiramente, com os requisitos de negócio e, em seguida, com os requisitos funcionais. Em seguida, preenche-se cada coluna com o número dos requisitos. Se um requisito for dependente de outro requisito, coloca-se um (X) indicando a dependência na célula da matriz, localizada na intersecção entre os dois requisitos.

Quadro 4 - Matriz de dependência de requisitos

		RF01	RF02	RF03	RF04	RF05	RF06	RF07	RF08	RF09
Requisitos de Negócio										
RN01	Conhecer os indicadores de consumo por veiculo	X					X	X		
RN03	Conhecer os indicadores de consumo por motorista	X	X		X	X	X		X	
RN01	Medir a qualidade dos combustíveis por posto de abastecimento	X		X			X			X
Requisitos funcionais										
RF01	Cadastramento de veículos									
RF02	Cadastramento de motoristas									
RF03	Cadastramento de postos de combustível									
RF04	Registro de saída de veículo	X	X							
RF05	Registro de chegada de veículo	X			X					
RF06	Registro de abastecimentos	X	X	X						
RF07	Consulta de comsumo por veículo	X					X			
RF08	Comparativo de consumo de combustível por motorista	X	X				X			
RF09	Comparativo de consumo de combustível por posto de combustível	X					X			

4.1.6. Planejamento das liberações

A partir da elaboração da matriz de Mudge e da matriz de dependência de requisitos, pode-se efetuar o planejamento das liberações do produto. Analisando a matriz de Mudge, as estratégias mais viáveis são:

1. implementar os três requisitos de negócio em uma única liberação;
2. implementar os requisitos RN01 e RN02, que são os mais importantes, segundo a matriz de Mudge, na primeira liberação e o requisito RN03, em uma segunda liberação;
3. implementar cada requisito em uma liberação separada, iniciando-se pelo requisito RN03.

Supondo-se que a estratégia adotada seja implementar cada requisito em uma liberação separada, deve-se analisar a matriz de dependência de requisitos, selecionando os requisitos funcionais necessários à implementação de cada requisito de negócio. Assim, para implementar o requisito de negócio RN01, é necessário implementar os requisitos funcionais RF01, RF06 e RF07. Para implementar o requisito funcional RF01, não é necessário implementar nenhum outro requisito. Para implementar o requisito funcional RF06, é necessário implementar os requisitos RF01, RF02 e RF03. Para implementar o requisito RF07 é necessário implementar os requisitos RF01 e RF06.

A primeira liberação será composta, portanto, pelos requisitos funcionais RF01, RF02, RF03, RF06 e RF07.

4.1.7.Especificação dos requisitos

A especificação dos requisitos começa com um diagrama de contexto (Figura 4), contendo os casos de uso do sistema, em seguida, descreve-se sucintamente cada um dos casos de uso. Um caso de uso modela uma interação completa entre um ator e o sistema (PÁDUA, 2002, LARMAN 2000). Normalmente, cada requisito funcional gera pelo menos um caso de uso. Deve-se cuidar para não detalhar, criar muitos casos de uso, pois o objetivo destes é compreender os requisitos de sistema e qualquer detalhamento excessivo implica aumento do tempo gasto na modelagem, sem gerar contribuição para a qualidade do produto final. Os casos de uso devem ser descritos textualmente, a fim de capturar detalhes importantes a respeito dos mesmos. A linguagem utilizada deve ser simples e totalmente desprovida de termos técnicos, com os quais o usuário não está familiarizado. A descrição pode ser complementada com um glossário com os termos do sistema normalmente utilizados pelo cliente. Este glossário serve para os desenvolvedores se familiarizarem com os termos do negócio e não para o cliente se familiarizar com termos técnicos utilizados pelos desenvolvedores.

Casos de uso podem ser descritos de diversas maneiras, desde uma descrição de alto nível, com algumas poucas sentenças, até uma descrição expandida, modelando em detalhes o diálogo entre ator e sistema.

Descrições expandidas são particularmente úteis em casos de uso em grandes projetos, em que, freqüentemente, o trabalho é dividido em diversas equipes e a comunicação é executada por escrito. Em pequenas equipes, geralmente, a mesma equipe se encarrega de todo o trabalho de desenvolvimento, desde os requisitos até a implantação e suporte.

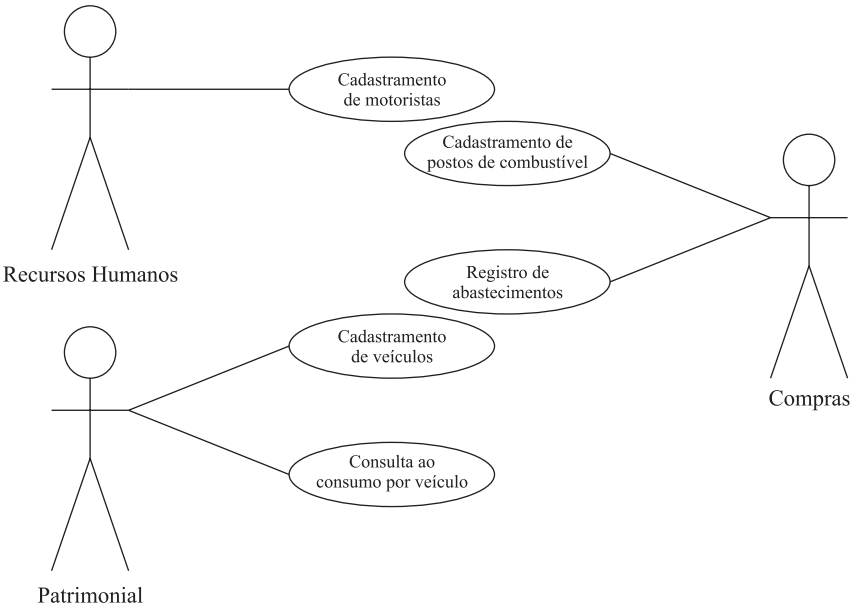

Figura 4 - Diagrama de contexto

A programação extrema utiliza *user stories* que, assim como casos de uso, tomam uma abordagem de caminho. Uma *user storie* é a menor quantidade de informações necessárias para que o cliente defina um caminho através de um sistema (ASTELS, MILLER e NOVAK, 2002).

Na metodologia proposta, casos de uso devem ser descritos em conjunto com o cliente e de maneira simplificada. Em boa parte dos casos de uso, apenas uma descrição sucinta basta para o caso de uso (Quadro 5). Esta descrição pode ser retirada da lista de requisitos. Em outras é necessário descrever o tratamento de algumas exceções (Quadro 6). Neste caso, deve-se cuidar para não descrever

aquelas exceções comuns a qualquer iteração (o usuário informa um código não cadastrado, por exemplo), mas somente aquelas importantes para o caso de uso. Em alguns casos é necessário descrever cálculos ou regras específicas impostas pelo modelo de negócio, neste caso, utiliza-se uma seção para regras de negócio. Em relatórios é importante anexar ao caso de uso um modelo de relatório, para que o *layout* do mesmo seja aprovado previamente pelo cliente.

O modelo de diálogo entre ator e sistema, que comandos o ator pode acionar e que ações estes comandos desencadeiam devem seguir uma padronização e, se possível, para todas as iterações ator-sistema e descritas por meio de uma interface padrão. Isto retira da descrição do caso de uso boa parte da complexidade, acelerando o processo de desenvolvimento e melhorando o entendimento do sistema por parte do usuário.

Quadro 5 – Exemplo de caso de uso

Nome do caso de uso	Cadastramento de motoristas	**Requisito**	RF02
Descrição	Inclusão e alteração e baixa no cadastro de motoristas		
Exceções	Ação do ator	Resposta do sistema	
Regras de negócio			

Quadro 6 - caso de uso com exeção

Nome do caso de uso	Registro de Chegada de Veículo	Requisito	RF05
Descrição	Registrar data, hora e quilometragem do veículo no momento da chegada. Deve haver uma saída anterior para que a chegada seja registrada		
Exceções	Ação do ator	Resposta do sistema	
	O ator lança a placa de um veículo que não tem saída registrada	Bloqueia a ação e emite uma mensagem ao ator, permitindo ao ator corrigir a placa do veículo ou desistir da operação	
Regras de negócio			

4.1.8.Modelo de Classes Conceitual

Um modelo conceitual é uma descrição de coisas do mundo real que fazem parte do domínio do problema, ele não descreve artefatos de software, tais como janelas ou bancos de dados (LARMAN, 2000). Embora este modelo adicione certa complexidade ao modelo, o usuário pode entendê-lo facilmente mediante alguns esclarecimentos. A utilidade do mesmo, durante a especificação de requisitos, é entrar em acordo com o cliente a respeito do domínio de informações do sistema (FIGURA 5).

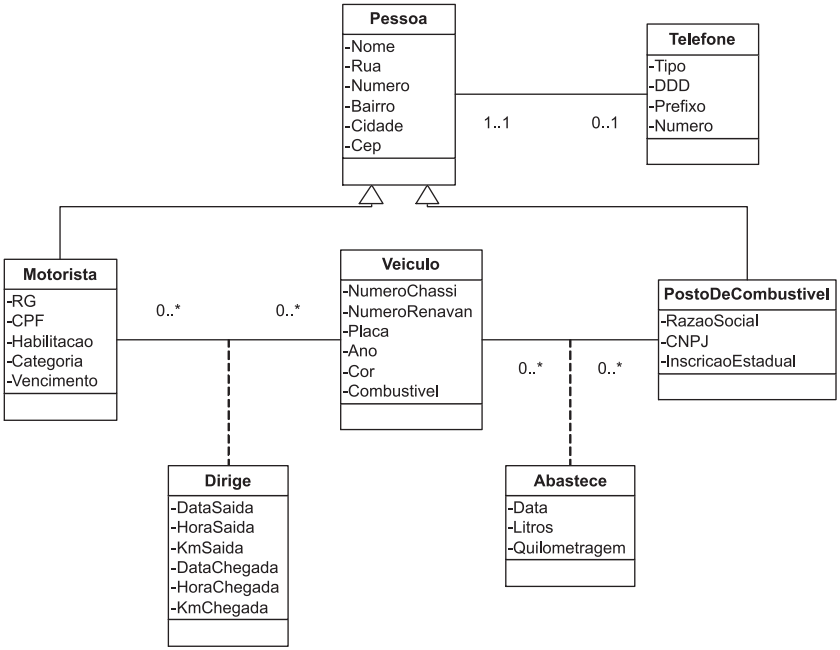

Figura 5 - Modelo conceitual

4.1.9. Validação dos requisitos

Corrigir um erro durante a fase de requisitos tem um custo até 100 vezes menor do que corrigi-lo após o software ter entrado em produção. O processo de validação dos requisitos visa assegurar que os requisitos sejam consistentes e completos em relação ao objetivo do sistema. A matriz de dependência de requisitos permite verificar se os requisitos estão sendo atendidos. Uma técnica simples, porém, útil é utilizar dados reais do cliente e fazer simulações em papel ou planilhas eletrônicas, utilizando os casos de uso e o modelo conceitual. A programação extrema (Astels, Miller e Novak, 2001) recomenda que se desenhem cenários de testes durante a fase de confecção das histórias de usuário. Uma boa prática, portanto, é escrever cenários de teste logo após a descrição do caso de uso.

5. Conclusão e estudos futuros

A metodologia proposta procura simplificar a atividade de engenharia de requisitos, a fim de possibilitar sua aplicação por pequenas equipes. O processo de classificação de requisitos leva em conta apenas o benefício

proporcionado ao cliente em cada requisito de negócio. Enquanto isso, um processo mais completo de classificação de requisitos estará sendo desenvolvido e deverá levar em consideração a relação benefício/esforço/estabilidade de cada requisito. O esforço e a estabilidade podem ser estimados, utilizando-se a Matriz de Mudge para capturar a percepção de esforço dos desenvolvedores para cada requisito funcional. Técnicas mais adequadas para quantificação do esforço podem utilizar-se de pontos de caso de uso ou pontos de função para estimar o tamanho de cada requisito e, a partir do tamanho, o esforço de desenvolvimento. O esforço necessário ao desenvolvimento dos requisitos funcionais deve, então, ser distribuído entre os requisitos de negócio.

Referências

ASTELS, D. MILLER, G. NOVAK, M. **Extreme Programming**: Guia prático. Rio de Janeiro: Campus, 2002.

BLASHEK, J.R. Gerência de Requisitos: O principal problema do software. **Developers CIO Magazine**, Rio de Janeiro, n. 70, p. 46-47, jun. 2002.

CRISTEL, M. G. e KANG, K. Issues in Requirements Elicitation. Disponível em www.sei.cmu.edu. Acesso em 01/05/2003.

LARMAN, C. **Utilizando UML e Padrões:** Uma introdução à análise e ao projeto orientados a objetos. Porto Alegre: Bookman, 2000.

MARQUIONI, C. E. Gerência de Requisitos: Os principais problemas típicos. **Developers CIO Magazine**, Rio de Janeiro, n.81, p. 11-12, maio 2003.

QUEDAS. M. S. A Gerência de Requisitos no Ciclo de Desenvolvimento. **Developers CIO Magazine**, Rio de Janeiro, n. 79, p. 30-31, mar. 2003.

PAULA FILHO. W. P. **Engenharia de Software**: Fundamentos, métodos e padrões. 2 ed. Rio de Janeiro: Livros Técnicos e Científicos, 2002.

PETERS, J, PEDRYCZ, W. **Engenharia de Software**. Rio de Janeiro: Campus, 2001.

POSSAMAI, O. **Qualidade do Projeto ao Produto**. Notas de aula da disciplina Qualidade do Projeto ao Produto. Mestrado em Engenharia de Produção. Universidade Federal de Santa Catarina, 1999.

PRESSMAN, R. S. **Software Engineering**: A practitioners approach. 5 ed. Boston: McGraw-Hill, 2001.

Recebido em 05/04/2005
Aprovado em 23/06/2005