LEVANTAMENTO DE REQUISITOS SEGUNDO O MÉTODO VOLERE

THE REQUIREMENTS ELICITATION ACCORDING TO THE VOLERE METHOD

RESUMO

Fazer um bom levantamento e especificação de requisitos é algo primordial para quem trabalha com desenvolvimento de sistemas. Esse levantamento pode não garantir que o software contemple todas as reais necessidades dos usuários, mas tende a antecipar o surgimento dos erros de entendimento e inconsistências, aprimorando o processo de desenvolvimento de produtos de software. Este artigo trata as metodologias existentes para descoberta destes requisitos, discorrendo mais detalhadamente sobre a estrutura da metodologia Volere.

Palavras-Chave: Requisitos, Volere, Engenharia de Software

ABSTRACT

Carrying out a good requirement elicitation and specification is crucial to those who work with the development of software systems. This elicitation does not guarantee that the software will meet all the users' real needs, but tries to anticipate the occurrence of potential misunderstandings and inconsistencies, thus improving the software products' development process. This paper addresses the existing methodologies that leverage the requirements elicitation, presenting the structure of the Volere methodology in more detail.

Keywords: Requirements, Volere, Software Engineering

Sonia Maria Antunes Da Silva
Graduando em Curso de Sistemas de Informação
Fundação de Estudos Sociais do Paraná
Instituto de Ciências Sociais do Paraná
Rua General Carneiro, 216
CEP: 80060-150 Centro
Curitiba -Paraná - Brasil
(41) 3028-6500
soniam@onda.com.br

Marcos Rodrigo Bonin
Graduando em Curso de Sistemas de Informação
Fundação de Estudos Sociais do Paraná
Instituto de Ciências Sociais do Paraná
Rua General Carneiro, 216
CEP: 80060-150 Centro
Curitiba - Paraná - Brasil
(41) 3028-6500
mbonin@uol.com.br

Marco Antonio Paludo
Mestre em Informática Aplicada pela PUC – PR
Professor da Disciplina de Engenharia de Software
Fundação de Estudos Sociais do Paraná
Instituto de Ciências Sociais do Paraná
Rua General Carneiro, 216
CEP: 80060-150 Centro
Curitiba - Paraná - Brasil
(41) 3028-6532
paludo@fesppr.br

INTRODUÇÃO

A Análise de Requisitos é a primeira atividade técnica no desenvolvimento do *software*, e pode ser entendida como responsável por definir os serviços que um sistema deve realizar, sua interface com os demais elementos e sob quais restrições o sistema deve operar. Os requisitos dos sistemas devem estabelecer o que o sistema deve fazer ao invés de como isto será feito.

O termo Análise de Requisitos ou Engenharia de Requisitos, segundo Stokes (2003 *apud* FISCHER, 2001, p.74), refere-se a uma coleção dos processos de extração, especificação, verificação e validação, apresentados na seqüência:

- A extração de requisitos é o exercício de agrupar as informações a fim de verificar exatamente o que o cliente ou usuário está requerendo.
- A especificação refere-se tanto ao processo como ao resultado deste processo. Resulta em um documento denominado Especificação dos Requisitos, em que toda a informação obtida sobre o processo está reunida. É quando ocorre o maior esforço de análise, o fluxo dos dados é avaliado, as funções definidas e detalhadas, o comportamento do software entendido no contexto do ambiente e as restrições do projeto são incluídos.
- A verificação é feita para assegurar que este documento não contém inconsistências, uma vez que os requisitos não devem ser conflitantes entre si.
- A validação preocupa-se em assegurar que o documento descreve com precisão o sistema que o cliente deseja, incluindo todas as funcionalidades e restrições impostas por ele.

Segundo Sommerville (2003, p. 82), o termo definição de requisitos serve para a fase inicial do processo, quando os requisitos são estabelecidos ao cliente e correspondem a macro especificações de serviços que o sistema deve realizar. Deve estar descrito em um documento em linguagem natural, acrescido de alguns diagramas. Já o termo especificação de requisitos é usado no outro extremo do processo, quando os requisitos estão detalhadamente definidos e um documento preciso é gerado, também conhecido por Especificação Funcional. No final uma

especificação um Documento de Requisitos de Software é montando. Esse documento une a definição e a especificação dos requisitos.

No desenvolvimento do processo alguns papéis estão envolvidos: os requerentes, os facilitadores e os implementadores. Os requerentes são os clientes e usuários, e representam as pessoas que precisam do sistema. Os facilitadores são analistas, e seu papel é o de desenvolver, ao longo do processo, as técnicas de extração, especificação, verificação e validação, numa descrição precisa do sistema que o requerente quer. Já os implementadores são engenheiros, projetistas e gerentes de projeto, que elaboram o sistema base do processo que efetivamente constroem os sistemas com base no documento de requisitos e no processo do software.

1. DEFINIÇÃO DE REQUISITO

Os requisitos são uma coleção de sentenças que devem descrever de modo claro, sem ambigüidades, conciso e consistente todos os aspectos significativos do sistema proposto. Eles devem conter informações suficientes para permitir que os implementadores construam um sistema que satisfaça os requerentes, e nada mais.

Conforme proposto por Sommerville (2003, p. 83), um requisito é tratado como funcional quando descreve um serviço ou função que o sistema deve realizar. Paralelamente pode haver requisitos não-funcionais, que são restrições impostas tanto ao sistema quanto ao seu desenvolvimento.

2. COMO IDENTIFICAR REQUISITO

Não existem, até o momento, técnicas capazes de lidar satisfatoriamente com todas as faces da análise de requisitos. Entretanto, existem ferramentas e técnicas capazes de resolver, com eficiência, parte do problema.

- Rápida Prototipação: trabalha com dois dos maiores problemas da análise, a validação dos requisitos e sua representação de forma compreensível aos diferentes leitores.
- Animação: é similar à rápida prototipação, porém as especificações são simplesmente executadas como um filme. A animação é um

recurso mais pobre que a prototipação, pois demonstra apenas alguns aspectos dos requisitos.

- Revisões: constituem em geral uma abordagem mais simples: a leitura dos requisitos. A desvantagem está no gasto excessivo de tempo, especialmente em grandes projetos.
- Prova das propriedades do sistema: baseia-se na crença de que o uso de linguagens formais. Mediante provas, chega-se à conclusão de que o sistema opera corretamente. No entanto, existem muitas restrições ao uso de linguagens formais. A desvantagem é que modelar o mundo real consome muito tempo, e nada nos assegura a consistência do sistema em relação ao mundo real.

3. POR QUE UTILIZAR REQUISITOS DE SOFTWARE

Ao tratar um software de porte pequeno, estático e sem muitas modificações parece apenas mais uma burocracia falar sobre especificação de requisitos, pois um simples documento descritivo resolveria esse problema. No entanto, para produtos de software comerciais de médio e grande porte, análise e levantamento de requisitos são de extrema importância. Segundo Khawar (2003, p. 26), com um bom Documento de Requisitos de Software é possível visualizar e interferir em um sistema sem causar problemas à aplicação existente. Além disso, o esforço gasto na localização de qualquer tipo de modificação ou implementação se torna aceitável.

4. TÉCNICAS PARA LEVANTAMENTO DE REQUISITOS

Os métodos que os analistas empregam para conceber um novo sistema são conhecidos como métodos de análise ou modelagem do sistema. Esses métodos ensinam a construir modelos abstratos do sistema. Nesta etapa de análise e modelagem os requisitos ficam bem claros, pois é na modelagem que eles são evidenciados.

Como afirmam Booch *et al.* (2000, p. 06), um modelo é uma simplificação da realidade, e modelos são construídos para compreender melhor o sistema que está sendo desenvolvido.

Os modelos seguem a técnica apresentada por Edsger Dijkstra (FISCHER, 2001, p. 85), em que um problema complexo é quebrado em uma série de problemas menores, o famoso “Dividir para Conquistar”.

Pela modelagem são atingidos quatro objetivos, no que se refere ao sistema: (i) modelos ajudam o desenvolvedor e o cliente a visualizarem um sistema como ele é ou deve ser, (ii) modelos nos permitem especificar a estrutura ou o comportamento de um sistema, (iii) modelos nos dão um molde que nos guia na construção de um sistema e, finalmente, (iv) modelos documentam as decisões tomadas.

Existem muitas técnicas de modelagem e, dentre elas, as consideradas mais expressivas são apresentadas a seguir, baseadas em diferentes tipos de abstração:

- Casos de Uso: uma descrição de um conjunto de seqüências de ações, resultantes da interação do sistema com um ator (um tipo de requerente). As ações produzem um resultado que pode ser observado pelo ator (BOOCH *et al.*, 2000, p. 19).
- *Viewpoints*: são mecanismos que permitem considerar aspectos do sistema percebidos por diferentes requerentes. Ao se analisar o sistema por vários aspectos obtém-se uma especificação mais adequada. Os *viewpoints* permitem melhor entendimento das necessidades dos usuários mediante cada uma das atividades do processo (FINKELSTEIN *et al.*, 1992, p. 5).
- CORE – Expressão Controlada dos Requisitos: o sistema a ser modelado é limitado e particionado em perspectivas ou *viewpoints*. Os *viewpoints* podem ser físicos ou funcionais, representando organizações, homens, entidades de hardware ou software e processos. No final, os *viewpoints* combinados produzem uma representação total das atividades, (FINKELSTEIN *et al.*, 1992, p. 5).
- DFD – Diagrama de Fluxo de Dados: define o fluxo de dados entre os vários processos funcionais, em que um processo é a transformação em algum dado de entrada em dados de saída, de acordo com alguma funcionalidade (FINKELSTEIN *et al.*, 1992, p. 4).
- RLP – Processador de Requisitos de Linguagem: na tentativa de solucionar o problema dos diversos tipos de leitores envolvidos, o RLP permite o desenvolvimento de linguagens formais para os

requisitos de especificação das aplicações, garantindo assim que os requisitos estejam completos, consistentes, sem ambigüidade e sem redundância.

- SREM – Metodologia da Engenharia de Requisitos de Software: é uma ferramenta baseada na engenharia de requisitos e validação de sistemas. Nessa metodologia, os caminhos dos dados consistem nas mensagens de entrada, na seqüência das tarefas processadas que envolvem o fluxo do controle, e nas mensagens da saída. É utilizada (a metodologia) para recomendar melhorias em grandes sistemas, (FINKELSTEIN *et al.*, 1992, p. 6).
- UML – *Unified Modeling Language*: linguagem para visualizar, especificar, construir e documentar os requisitos e informações de um sistema (BOOCH *et al.*, 2000, p. 13).
- Volere: é um método completo de obtenção de requisitos, baseado nos casos de uso (FISCHER, 2001, p. 87).

5. VOLERE

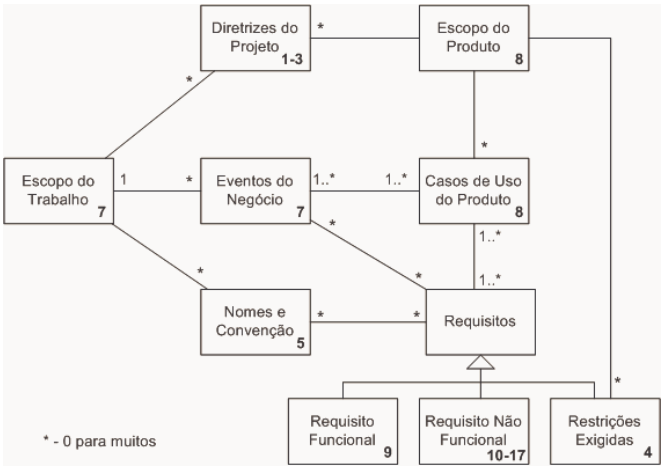
De todas as técnicas existentes no mercado para levantamento de requisitos, optou-se por aprofundar o método Volere. Conforme citado pelos autores James e Suzanne Robertson (2004), este é um método que já foi adotado por milhares de organizações no mundo todo e seu modelo parte do princípio da sumarização de experiências em desenvolvimento de software para montagem de um modelo simples e completo.

5.1 Por onde começar

Uma dica importante para trabalhar com Volere é entender bem seu Modelo de Especificação. Tal modelo está disponível para *download* na internet pelo *site*: http://www.volere.co.uk/ e traz a fundamentação de todo o processo de especificação de requisitos. O material apresentado a seguir é baseado no documento de especificação de requisitos Volere, conforme aparece em Robertson & Robertson (1995).

5.2 Como começar o projeto

Figura 1 - Estrutura dos requisitos do método Volere.



Fonte: Robertson (2000)

Os oito primeiros itens do modelo trazem a visão da viabilidade do projeto, e o analista deve estar seguro das informações armazenadas em cada um deles, para assim conseguir visualizar o projeto e chegar à especificação dos requisitos.

O Modelo Volere está dividido conforme estrutura apresentada na Figura 1 e descrito na seqüência:

a) Direcionamento do Projeto

- A finalidade do produto, ou seja, o propósito do produto, o que motiva ao desenvolvimento do projeto.
- Cliente, patrocinador e outros *stakeholders*: neste item são identificados o cliente, o patrocinador e as pessoas que detêm o domínio do problema em estudo.
- Usuários do produto: aqui são listados os potenciais usuários do produto, elencando fatos relevantes sobre cada um deles, tais como escolaridade, visão quanto ao trabalho, quanto à tecnologia, etc.

b) Restrições do Projeto

- Restrições obrigatórias: aqui são indicadas as restrições quanto a software, tecnologia a ser aplicada no projeto, ambiente de execução, ambiente de trabalho, dentre outras coisas.
- Convenções de nomes e definições: deve-se, neste item, descrever todos os termos significativos usados no projeto, como em um dicionário de termos.

- Fatos e suposições relevantes: tratam os fatores externos que têm efeito no produto, mas que não exigem mudanças nos requisitos. Exemplo: regras de negócio, sistemas externos que interferem no produto.

c) Exigências Funcionais

- O escopo do projeto: trata o projeto em toda a sua extensão. A partir do estudo do modelo atual é montado um modelo proposto e pelo diagrama de contexto é gerada uma lista de eventos. Com estas duas informações em mãos, diagrama de contexto + lista de eventos, fica fácil visualizar o projeto.
- O escopo do produto: neste item, cada evento da lista é tratado e, pelos diagramas de caso de uso, cada um deles é identificado como um processo independente ou parte de algum cenário. Cada um dos eventos da lista trata normalmente um requisito funcional.

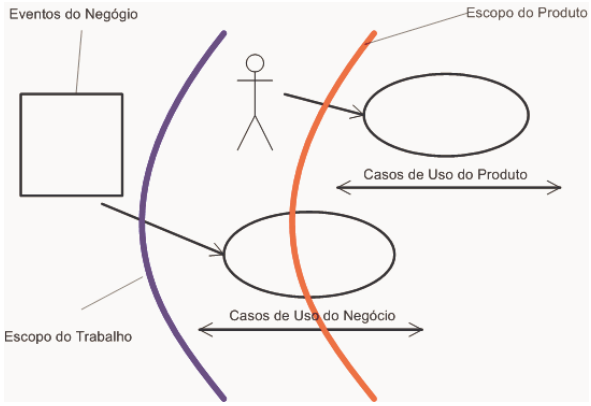
Ressalta-se a observação de que os itens citados anteriormente são imprescindíveis ao modelo Volere no que tange à especificação de requisitos, e a partir do momento em que eles foram identificados, problemas como ambigüidade, escopo do projeto ou produto podem ser facilmente identificáveis.

6. IDENTIFICANDO EVENTOS OU CASOS DE USO

Após a análise do escopo do projeto e identificação de todos os eventos, a próxima tarefa será dedicada à investigação. O contexto do trabalho, visto no Escopo do Projeto, define o que deverá ser endereçado. Cada um dos fluxos da entrada e da saída da lista de eventos representa um evento do negócio.

O analista de sistemas nunca deve perder de vista que o sistema deve ter a cara do cliente, ou seja, ele deve ser como o cliente o vê. Para esta definição, os casos de uso do Produto e do Negócio devem ser considerados e são ilustrados na Figura 2.

Figura 2 – Fronteiras de abstração dos casos de uso.



Fonte: Robertson & Robertson (2004)

7. ESCOLHENDO UM EVENTO CHAVE

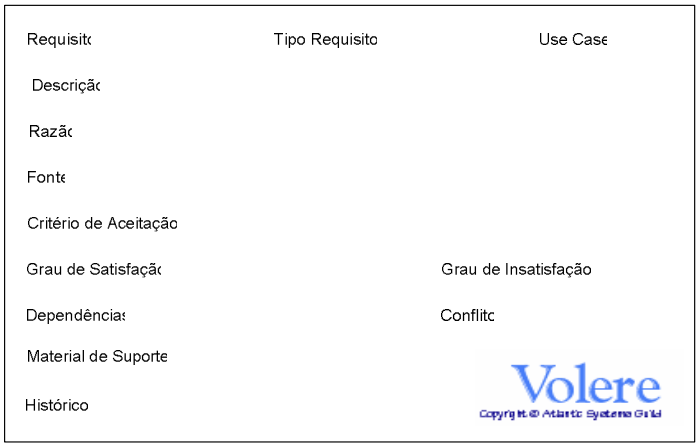
Em seu artigo, Robertson & Robertson (2004), enfatizam que haverá um evento do negócio, ou às vezes diversos, em lista do evento que é relacionado como a razão para desenvolver o projeto.

Nesta etapa, já com os casos de uso prontos, é possível simular o uso do evento e neste momento os *stakeholders* (detentores do domínio do negócio) podem atuar em conjunto para confirmar, ou não, a continuação do evento.

Cada caso do uso de produto representa algo que você quer que o produto faça. Assim, tem-se um número de requisitos funcionais associados a ele.

Segundo Robertson & Robertson (1995, p. 4), cada caso de uso deve ter elencados, por meio dos cartões, os seus requisitos funcionais, os seus requisitos não funcionais e seus requisitos de restrições.

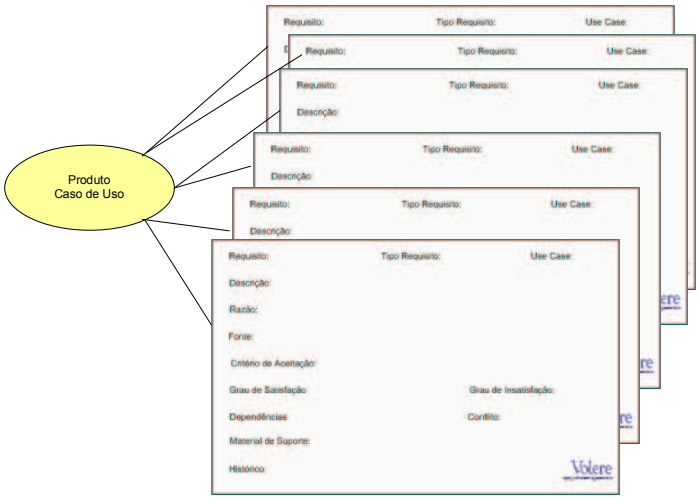
Figura 3 – Cartão Volere



Fonte: Robertson & Robertson (2004)

Cada requisito de software e caso de uso deve estar descrito em um cartão, conforme modelo apresentado na figura acima. Nos cartões Volere são definidos a descrição, a razão, o critério de aceitação, as dependências, os conflitos, entre outros detalhes sobre cada um dos requisitos. Cada caso de uso pode ter diversos cartões associados a ele.

Figura 4 – A relação Volere Caso de Uso



Fonte: Robertson & Robertson (2004)

8. ESTUDO DE CASO E CONCLUSÕES

Durante o desenvolvimento do software *PetSoft*, projeto de conclusão de curso/2005 para a Empresa Bichos e Caprichos, foram aplicadas todas as etapas do método Volere até o capítulo 17 e pode-se concluir que o processo, embora trabalhoso, se tornou muito produtivo, pois permitiu que a amplitude completa do projeto fosse evidenciada, conforme um dos objetivos do método.

Ainda, conforme o processo evoluiu para as fases de projeto lógico e projeto físico, os requisitos especificados nos cartões Volere nortearam o desenvolvimento dos requisitos funcionais e não-funcionais de maneira adequada e completa.

A última aplicação dos requisitos Volere irá ocorrer na entrega e homologação do produto final, quando serão utilizados para validação final do software contra os requisitos esperados e explicitados.

REFERÊNCIAS

BOOCH, G.; RUMBAUGH, J.; JACOBSON, I. **UML:** guia do usuário. Tradução por Fábio Freitas. Rio de Janeiro: Campus, 2000, xx, 472 p.

FISCHER, M. C. B. O. **Estudo de Requisitos para um Software Educativo de Apoio ao Ensino da Introdução à Computação.** Dissertação de Mestrado. Local: USP – Universidade de São Paulo, 2001.

FINKELSTEIN, L: HUANG, J: INKELSTEIN, A; NUSEIBEH, B. **Using Software Specification Methods for Measurement Instrument Systems**. Part 1: Structured Methods. London: City University, 1992.

KHAWAR, A. Z.; UMRYSH, C. **Desenvolvendo Aplicações Comerciais em Java com J2EE e UML**. São Paulo: Moderna, 2003.

ROBERTSON, J.; ROBERTSON, S. **Volere Requirements: How to Get Started.** London: Adison-Wesley, 2004. Disponível em <http://www.volere.co.uk/gettingstarted.htm> Acesso em: 08/05/2005.

_____. **Volere: Requirements Specification Template.** London: Adison-Wesley, 1995.

ROBERTSON, S. **Requirements Fundamentals: the basis for effective testing.** London: Adison-Wesley, 2000. Disponível em <http://www.volere.co.uk/fundamentals.pdf> Acesso em: 05/05/2005.

SOMMERVILLE, I. **Engenharia de software.** Tradução Maurício de Andrade. São Paulo: Addison-Wesley, 2003. xiv, 592 p.