Validação de Requisitos de Sistema para Geração de Use Cases

Aluno: Ricardo Miguens Nizoli
Universidade Federal do Rio Grande do Sul
Instituto de Informática
nizoli@inf.ufrgs.br

Orientadora: Profa. Dra. Ana Maria de Alencar Price
Universidade Federal do Rio Grande do Sul
Instituto de Informática
anaprice@inf.ufrgs.br

Resumo

Dentre as fases do desenvolvimento de um software, a determinação, a expressão e a validação dos requisitos – que formam a Engenharia de Requisitos – podem ser consideradas de vital importância para que o ciclo de desenvolvimento tenha sucesso. Qualquer que seja a metodologia adotada para o desenvolvimento, é a partir dos requisitos que se basearão todos os outros passos necessários para alcançar o objetivo final, que é o software. Este artigo introduz a dissertação de mestrado onde propõe-se a criação de um método para a validação dos requisitos de sistemas. No método proposto, após a aceitação do usuário, os requisitos são definidos como *use cases*, organizados de forma a auxiliar as fases subsequentes do desenvolvimento.

Palavras-chave: Engenharia de Requisitos, UML, Validação, Casos de Uso.

Abstract

Among the software development phases, the determination, the expression and the validation of the requirements – which embody the Requirements Engineering – can be considered of vital importance for the success of the development cycle. Whatever methodology is adopted for the development, it is starting from the requirements that all other necessary steps will be based on to achieve the final objective, the software itself. This paper introduces the dissertation where is proposed the creation of a method for system requirements validation. In the proposed method, after the user's acceptance, the requirements are defined as use cases, organized in a way to assist the subsequent development phases.

Key words: Requirements Engineering, UML, Validation, Use Cases.

1. Introdução

Tendo como foco o aumento da qualidade de produtos de software, nota-se que um número cada vez maior de desenvolvedores está adotando (ou pretende adotar) algum procedimento sistemático para testar seu produto. E dentre aqueles que ainda não o fizeram, a grande maioria concorda com a importância de integrar o teste ao ciclo de desenvolvimento. Na verdade, existe um consenso sobre a necessidade de adotar o teste já nas primeiras fases de desenvolvimento do software.

Entretanto, isto raramente é verificado. Pode-se dizer que é costume entre os desenvolvedores preocuparem-se com o teste, ou com a validação do que foi feito, apenas durante a fase de implementação. Este fator implica um problema grave: quanto mais tarde um erro for descoberto, maiores os custos e o tempo necessários para solucioná-lo.

Diante disso, erros que ocorram nas etapas que compõem a Engenharia de Requisitos (ER), se não descobertos e solucionados a tempo, certamente terão um resultado "catastrófico" nas fases finais do ciclo. O conjunto de requisitos do sistema é a base para todo o ciclo de desenvolvimento: caso este conjunto contenha erros, todo o ciclo fica comprometido, enquanto tais erros não forem corrigidos.

A desculpa mais comum para que não se adote a validação dos requisitos pode ser considerada a falta de um método eficiente para tanto, que não eleve demais os custos e o tempo de desenvolvimento. Durante o estudo, pôde-se notar que nem mesmo em um processo de desenvolvimento genérico fazendo uso da UML (*Unified Modeling Language*) a etapa de validação de requisitos é seriamente tratada [2].

A proposta deste trabalho é criar um método para a validação dos requisitos, agregando a este, quando conveniente, etapas propostas em outros trabalhos relevantes na área. No método a ser desenvolvido, além da validação dos requisitos, será possível, ao final de sua aplicação, gerar *use cases* que auxiliarão a fase de projeto do sistema. Propõe-se ainda a criação de um protótipo de uma ferramenta de apoio ao método desenvolvido.

2. Propostas para Engenharia de Requisitos

Considerou-se para o trabalho o modelo proposto por Jarke e Pohl [1], onde para a ER são identificadas as etapas de Determinação, Expressão e Validação dos Requisitos. Esta proposta é considerada pela comunidade como o modelo iterativo clássico para ER.

Em [4], o autor argumenta que, apesar do crescente interesse pela ER, pode-se observar que a maior parte dos trabalhos e métodos propostos na literatura dirigem sua atenção para os problemas de expressão dos requisitos, criando vários modelos diferentes conforme as informações do domínio do problema e do sistema a ser desenvolvido. Nota-se portanto a falta de trabalhos que abordem centralmente a validação de requisitos, com algumas exceções (ex.: [3]). Entretanto, podem-se citar algumas técnicas para validação de requisitos:

- Avaliação Informal;
- Prototipação;
- Verificação Formal;
- Reutilização de Domínio.

Uma possível razão da pouca utilização destas (e de outras) técnicas de validação se dá ao fato de existir um pequeno número de ferramentas que suportem esta atividade. Um método voltado exclusivamente para validar requisitos, por melhor que seja, dificilmente será utilizado por algum analista, e duas razões para isto podem ser identificadas:

a) O tempo e os custos dispensados a esta atividade podem tornar-se relativamente altos;
b) Não será fornecido nenhum resultado aplicável diretamente no ciclo de desenvolvimento.

Desta forma, um método que, assistido por uma ferramenta que o suporte, torne possível a validação dos requisitos de um sistema e que, além disso, forneça subsídios **reais** para as etapas subsequentes do ciclo de desenvolvimento (*use cases*, por exemplo), pode ser considerado de grande interesse para quem desenvolve sistemas. Na seguinte subseção é mostrado, de forma bastante resumida, o método proposto no trabalho.

2.1 A Proposta do Trabalho

Conforme já salientado, o objetivo principal deste trabalho é propor um método para validação de requisitos de sistemas e posterior geração de *use cases* a partir dos requisitos que o usuário tenha validado. Neste momento, a concepção do método está sendo finalizada, tornando possível descrevê-lo como um conjunto de etapas a serem cumpridas interativamente entre analistas e usuários a partir de uma definição de requisitos já existente. Portanto, a *entrada* para o método é um documento de requisitos determinados e expressados. As etapas que compõem o método são descritas nas próximas subseções.

2.1.1. Divisão do documento de requisitos em *packages*

Considera-se, no escopo do trabalho, uma *package* como uma seção no documento de requisitos que descreve uma característica em particular do sistema. Em um sistema de controle bancário, por exemplo, uma *package* fundamental seria *Controle de Terminais de Auto-Atendimento.* Uma lista de *packages* deve ser apresentada ao usuário para que este verifique ausências ou excessos de definições.

2.1.2. Identificação de use cases de alto nível

Neste momento, o analista já começa a elaboração dos *use cases*. No escopo do trabalho, os *use cases* de alto nível representam os processos mais importantes de forma breve, usualmente duas ou três sentenças [3]. Uma lista de *use cases* de alto nível deve ser apresentada ao usuário, na medida que algo que pareça fundamental ao analista pode não ser assim considerado pelo usuário, e vice-versa. No exemplo, poderiam ser definidos os seguintes *use cases* de alto nível: *Saques, Extratos/Saldos*, *Pagamentos*, etc.

2.1.3. Identificação dos use cases de alto nível que serão expandidos

Um *use case* expandido mostra mais detalhes que um de alto nível [3]. Portanto, devem ser analisados os processos que merecem maior interesse e esforço para validação. Novamente, uma lista é gerada, sendo apresentada ao usuário. Para o exemplo, poderia ser expandido o *use case* de alto nível *Pagamento* em: *Pagamento de Água/Luz/Telefone*, *Pagamento de Títulos* e *Pagamento de INSS*.

2.1.4. Identificação de obstáculos não descritos nos requisitos

Nesta etapa, o analista verifica, para cada *use case* expandido, a existência de possíveis situações de erro (ou exceções) não contidas no documento de requisitos. A apresentação desta lista ao usuário é fundamental, uma vez que ele é a pessoa (a princípio) mais indicada para verificar a veracidade destes obstáculos. Para o exemplo seguido, um possível obstáculo para *Pagamento de Títulos* seria *Código de Barras Danificado*.

2.1.5. Para cada use case expandido, identificar as funções que utiliza

Conforme descrito por Larman, um *use case* possui uma seção denominada *Curso Típico de Eventos*, que descreve as ações do usuário e as respostas do sistema [2]. Existe a possibilidade, portanto, de que cada uma das respostas do sistema possa ser representada por

uma função. Além disso, estuda-se a possibilidade de representar-se uma ação do usuário por meio de uma função.

2.1.6. Validação 1

Os *use cases* são apresentados ao usuário. Com o uso de uma ferramenta de apoio, o usuário poderá navegar através do *Curso Típico de Eventos*, verificando a completitude e possíveis inconsistências/ambigüidades nos *use cases*. Na tela da ferramenta, deverão estar sempre presentes o nome do *use case* expandido, o nome do *use case* de alto nível que o gerou, os atores, o propósito, uma breve descrição e as referências cruzadas com outros *use cases*. Existem duas possibilidades neste momento: caso o usuário não valide o *use case*, deverá ser identificado o local e o tipo de erro, para que seja corrigido pelo analista; caso o *use case* seja validado, o método segue. Um *use case*, após corrigido, é novamente apresentado ao usuário para que seja revalidado, e este processo de revalidação pode e deve ocorrer quantas vezes forem necessárias.

2.1.7. Elaboração de Diagramas

Neste momento, o analista possui todas as informações necessárias para a elaboração dos *use cases* de alto nível e expandidos. Diferentemente do processo de desenvolvimento tradicional da UML, no método proposto as informações já estão validadas, e portanto, os *use cases* já estão validados no momento de sua concepção. Esta etapa é denominada Elaboração de Diagramas na medida em que estuda-se a possibilidade de integração do protótipo proposto com ferramentas *CASE* que suportem a criação de *use cases* (ex.: Rational Rose).

2.1.8. Validação 2

Opcionalmente, o analista poderá apresentar a representação gráfica dos *use cases* gerados ao usuário. No entanto, caso o usuário encontre um erro, o processo é o mesmo que na primeira validação, ou seja, deverá ser indicado o local e o tipo de erro para que o analista corrija-o.

3. Conclusões

3.1 Estágio Atual do Trabalho

De forma bastante simplista, para o cumprimento dos objetivos do trabalho, já foi feito um estudo de trabalhos relevantes na área de ER e foi proposto um método para validação de requisitos e geração de *use cases*. As próximas atividades consistem na validação do método proposto e na criação de um protótipo de ferramenta de suporte a este método.

3.2 Contribuições e Resultados Esperados

A expectativa é que o método proposto propicie uma solução para alguns dos problemas referentes à validação de requisitos, considerada por todos como uma etapa importante, mas ainda pouco explorada em pesquisas e trabalhos científicos.

4. Referências Bibliográficas

[1] Jarke, M et all. Requirements Engineering: An Integrated View of Representation, Process and Domain. In: 5th European Software Engineering Conference, 1993. Proc

[2] Larman, C. *Applying UML and Patterns: An Introduction to Object-Oriented Analisys and Design*. Prentice-Hall: New Jersey.1997.

[3] Leite, J.C.S.P.; Freeman, P.A. Requirements Validation Through Viewpoint Resolution. IEEE Trans. Soft. Engin., V.17, N.12, December 1991, pp 1253-1269.

[3] M.S.Pimenta. TAREFA: Une Approche pour I'Ingénierie des Besoins des Systèmes Interactifs. Toulouse: Doctorat de Lúniversite Toulouse I, 1997, 285p.