Gerenciamento de Requisitos em Scrum baseado em Test Driven Development

Rafael Soares[1], Thiago Cabral[1], Fernanda Alencar[1,2]

[1] Programa de Pós Graduação em Engenharia da Computação, Universidade de Pernambuco, Rua Benfica, 455 – Madalena – Recife/PE, Brasil
{rhas, tclm,fernandaalenc}@ecomp.poli.br
[2] Universidade Federal de Pernambuco, DES-CTG, Av. de Arquitetura, s/n., CDU - Recife/PE, Brasil.
fernanda.ralencar@ufpe.br

Abstract. In order to improve the success rate of software projects various development methodologies have been proposed as alternative to traditional models of development, the agile methodologies. Some steps of the traditional methodology were changed or even canceled, on agile methods, particularly at requirements engineering phase. Traditional methodologies rely on the documentation to manage the project and share knowledge. Meanwhile, the agile methodologies focus on the interaction between those involved in the project to deal with the same goals. There are several challenges with respect to requirements engineering in agile methods. Among these are problems at the process of requirement's elicitation and analyses. This work proposes to analyze tools, techniques, or patterns, like Test Driven Development, in order to better manage the process of requirement's elicitation and validation at Scrum.

Resumo. No intuito de melhorar a taxa de sucesso dos projetos de software várias metodologias foram propostas como alternativa aos modelos tradicionais de desenvolvimento de software, as metodologias ágeis. Nas metodologias ágeis alguns passos foram substituídos ou até mesmos cancelados, sobretudo na fase de engenharia de requisitos. As metodologias tradicionais dependem da documentação para gerenciar o projeto, tanto como para disseminar o conhecimento. Enquanto isso, as metodologias ágeis focam nas interações entre os envolvidos no projeto para lidar com o mesmo propósito. Existem diversos desafios com relação à engenharia de requisitos dentro das metodologias ágeis. Dentre esses, estão problemas no processo de elicitação e análise de novos requisitos. Esse trabalho propõe analisar ferramentas, técnicas ou padrões, como *Test Driven Development*, a fim de melhorar o cenário do processo de elicitação e validação de requisitos no Scrum.

1 Introdução

Tradicionalmente a gerência de requisitos é uma atividade chave no processo da engenharia de requisitos e tem como objetivo principal controlar a evolução dos requisitos, seja por constatação de novas necessidades, seja por constatação de deficiências nos requisitos registrados nos produtos em desenvolvimento. Nesse contexto o rastreamento é fundamental e a documentação primordial.

Nas metodologias tradicionais de desenvolvimento, mantêm-se o foco na geração de documentação sobre o projeto e no cumprimento rígido de processos. Na proposta ágil a ideia é concentrar as atenções no desenvolvimento em si e nas relações entre os participantes [1]. Nos métodos ágeis os requisitos são desenvolvidos de forma incremental, de acordo com as prioridades do cliente.

O Scrum é uma das mais populares metodologias ágeis para o planejamento e gerenciamento de projetos. É especialmente utilizada para projetos de manutenção e desenvolvimento de software. No Scrum os requisitos são tratados de forma incremental, a cada iteração (sprint). Assim, a preocupação com a interação entre os envolvidos no projeto, de forma que os processos de elicitação, validação e manutenção sejam realizados, deve ser maior do que a preocupação em documentar tais processos de fato [1]. Porém, vários são os problemas com relação à engenharia de requisitos no Scrum apontados na literatura e registrados por Jaqueira et al. [11], [5]. Dentre esses, destacam-se os problemas enfrentados durante o processo gerenciamento de requisitos, sobretudo relacionados às elicitação e análise de requisitos. Tais problemas podem levar a requisitos falhos, mal entendidos, ou pouco especificados, podendo impactar negativamente no produto final e na evolução de novos produtos.

Nos dias atuais, a técnica de desenvolvimento orientada a testes (do inglês, Test Driven Development – TDD), proposta por Kent Beck [3], que se baseia em um ciclo curto de repetições, vem ganhando notoriedade entre os projetistas de software. Sobretudo em projetos que usam metodologias ágeis, que como dito, apresentam algumas fragilidades. Em TDD, primeiramente, o desenvolvedor escreve um caso de teste automatizado que define a melhoria desejada ou a nova funcionalidade. A partir daí é produzido um código que pode ser validado pelo teste. Posteriormente, esse código será refatorado em um novo código, mas em padrões aceitáveis.

Com base nisso, buscou-se identificar trabalhos relacionados que tivessem a mesma preocupação com relação às fragilidades dos métodos ágeis, em particular, o Scrum. Da análise inicial, concluiu-se que alguns estudos [12], [13], [14] indicam que a utilização de TDD juntamente com o Scrum traz benefícios ao projeto. Em [12], aponta-se que proporciona melhora no projeto do código e na cobertura de testes das aplicações. Em [13], aponta-se que a utilização da técnica de TDD faz com que os desenvolvedores tenham mais confiança durante a manutenção do código; e, que a produtividade das equipes de desenvolvimento aumenta. Em [14], afirma-se que a utilização de TDD melhora o entendimento dos requisitos.

Considerando os benefícios encontrados com o uso de TDD, e as fragilidades com relação às metodologias ágeis, neste trabalho propõe-se integrar TDD ao Scrum, com vistas a melhorar o processo gerenciamento da elicitação e análise de requisitos. Para

isso, está em curso uma revisão sistemática da literatura, de modo a aprofundar a identificação das abordagens que tratam da mesma temática e quais as questões ainda não trabalhadas.

Esse artigo está estruturado como segue: na seção 2 são apresentados os objetivos do trabalho; as possíveis contribuições da proposta são discutidas na seção 3; algumas conclusões esperadas são apontadas na seção 4; e, por fim, na seção 5 tem-se o andamento da pesquisa e os trabalhos futuros.

2 Objetivos da Pesquisa

O Scrum defende que a interação entre os envolvidos no projeto pode ser uma ferramenta poderosa na elicitação de novos requisitos, tanto como na análise de requisitos. Todavia, as pesquisas [11], [5] mostram que o processo de elicitação e análise de requisitos, no Scrum, encontra alguns desafios, tais como: disponibilidade do cliente; equipe multifuncional, negligência dos requisitos não funcionais, falta de documentação formal, dentre outros. Já os problemas encontrados na fase de análise são: falta de integrantes especializados em análise; falta de documentos formais para análise; dentre outros.

Dessa forma, estamos investigando as vantagens em se propor e utilizar casos de testes, com base no paradigma TDD, bem como um mecanismo de como fazer essa integração. O foco está na melhoraria do processo de gerenciamento da elicitação e análise dos requisitos dentro do Scrum. Tem-se como preocupação, preservar os princípios básicos do Scrum: indivíduos e interações são mais valorizados do que processos e ferramentas; software funcional no lugar de documentação extensiva; colaboração com o cliente sobre a negociação de contratos; e, respostas a mudanças em vez de seguir um plano. Pretende-se chegar a um conjunto de boas práticas para a adoção dos casos de testes que possa ser utilizado pela indústria. Para tanto, deve-se pensar em um processo de execução relativamente simples e que possa ser executado com facilidade e agilidade dentro de uma iteração de desenvolvimento, cobrindo o maior número possível de riscos.

Para a automatização dos testes, identificou-se, na literatura e nos ambientes de desenvolvimento na indústria, a existência de algumas ferramentas que ajudam na implementação de casos de teste com métodos ágeis [15]. Dentre essas, em virtude de da familiaridade no seu uso em projetos reais, elegeu-se considerar, inicialmente, a ferramenta JUNIT [10].

A ferramenta JUNIT, além de ajudar a implementar os casos de testes ágeis, em TDD, dentro do ambiente de desenvolvimento, dão indicação, ao desenvolvedor, onde as mudanças podem interferir e impactar no projeto como um todo.

Argumentamos que o uso de casos de testes em TDD juntamente com o JUnit auxilia a equipe de desenvolvimento tanto na hora da realização de mudanças e evolução no código, como na detecção de requisitos falhos. Lembrando-se que tal detecção de falhas, pode ainda levar à descoberta de novos requisitos antes não pensados pela equipe.

3 Contribuições Científicas

Com já explicitado na seção anterior, pretende-se utilizar casos de testes através da técnica TDD, com auxílio da ferramenta JUNIT, visando melhorar o processo de elicitação e análise de requisitos, em projetos de software desenvolvidos com o framework Scrum. Pressupõe-se que a utilização da técnica TDD, com auxílio do JUNIT, pode oferecer uma alternativa mais simples e rápida para o processo de elicitação e análise de novos requisitos dentro do Scrum.

Algumas impressões iniciais já foram coletadas a partir do desenvolvimento de um projeto real, onde se utilizou o Scrum com a integração de casos de testes desenvolvidos com a ferramenta JUNIT.

O projeto teve como objetivo desenvolver um sistema de apoio a decisões gerenciais na área de fiscalização financeira. A equipe de desenvolvimento era composta por 11 pessoas, programadores, testes, analistas e o gerente de projetos. A duração do projeto foi de 3 anos e 8 meses e, a pedido do cliente, a equipe de desenvolvimento utilizou a técnica de TDD com auxílio da ferramenta JUNIT. Essa proposição se deu com o intuito de amenizar os problemas da engenharia de requisitos, já que a metodologia de desenvolvimento utilizada era o Scrum.

No início a equipe de desenvolvimento teve a sensação de que havia uma perda de tempo, pois antes que a implementação fosse iniciada todos os cenários de testes deveriam ser escritos. Porém, constatou-se que a discussão gerada, proveniente da utilização do TDD, entre os envolvidos do projeto, ajudou também na elicitação de novos requisitos e na detecção de requisitos falhos, diminuindo-se o retrabalho.

Pelo fato do JUNIT oferecer uma visão de quais testes estão sendo afetados, em função de alguma alteração no código, houve não apenas um aumento da confiança dos desenvolvedores em realizar tais mudanças, mas também houve uma melhora na gerencia dessas alterações.

Assim, pretende-se constatar com a pesquisa que um dos principais pontos positivos do TDD é justamente estimular o desenvolvedor a escrever casos de testes antes mesmo de escrever o código funcional e que pode ser facilmente integrado ao Scrum.

Espera-se, melhorar o processo de elicitação e análise de novos requisitos no Scrum. No processo de elicitação de novos requisitos, discutir, escrever e executar os casos de testes pode fazer com que novos requisitos, que não foram detectados previamente pela equipe, possam surgir. Pretende-se verificar que a integração de TDD com o JUnit no Scrum possa também melhorar o processo de análise de requisitos falhos.

4 Conclusões

Mesmo que o Scrum esteja em ascensão, no mundo da indústria de software, e que se mostre eficiente em alguns casos, existem muitos problemas, sobretudo com relação à engenharia de requisitos. Gerenciar os requisitos é uma prática, que dentre muitos

benefícios, serve para comunicar a equipe de desenvolvimento qual o comportamento do sistema e serve de auxilio na manutenção do código já existente.

Tendo em vista a importância do gerenciamento dos requisitos dentro de um projeto de software e que essa fase da engenharia de requisitos não está bem abordada no Scrum é proposta a utilização de casos de testes com base na técnica TDD e com o auxílio da ferramenta JUNIT. A experimentação e uso inicial dessa ferramenta permitiram concluir que a detecção dos impactos resultantes das mudanças ou da evolução dos requisitos de um produto de software fica facilitada com o uso de casos de testes.

Objetiva-se melhorar o cenário dos impactos das mudanças de requisitos em um projeto desenvolvido com o Scrum. O desenvolvimento orientado a testes pode se tornar excessivamente trabalhoso devido a dependências entre classes da aplicação, mas é justamente na dependência que se acredita ter a chave para o gerenciamento das mudanças e do descobrimento de novos requisitos.

Para guiar o processo, pretende-se propor um conjunto de boas práticas de forma a integrar esses casos de testes em paralelo com o uso do Scrum, sem infringir os princípios básicos dos métodos ágeis. Isso já começou a ser testado e parece se adequar razoavelmente bem. Espera-se ampliar o universo da pesquisa através de uma revisão sistemática da literatura mais expressiva.

5 Trabalhos Futuros e em Andamento

Esse é um trabalho que inicia uma nova área de pesquisa no grupo de requisitos, onde se investiga encontrar uma solução para a problemática de gerenciamento em métodos ágeis, o Scrum em especifico, e a criação de casos de testes segundo TDD.

Como mencionado, uma revisão sistemática da literatura está sendo executada com o fim de investigar se já existem indícios sobre os benefícios que o Test Driven Development traga para a elicitação e análise de requisitos em Scrum. Após esse levantamento, serão propostas as boas práticas e implantadas em alguns projetos, a fim de se atestar experimentalmente os resultados da integração proposta.

Pretende-se usar o método de pesquisa experimental a fim de assegurar o teste das hipóteses por meio de um experimento controlado, projetado e aplicado em um projeto real de forma a produzir dados necessários. O processo de experimentação pretendido compreenderá: definição da hipótese; concepção do protocolo experimental; definição de um conjunto de regras no qual se enquadra o experimento; a execução do experimento; e, por fim, a análise e interpretação dos resultados.

Pensa-se também a criação de uma ferramenta que possa interagir com o JUNIT no intuito de levantar dados e gerar artefatos baseados nos casos de testes, escritos no JUNIT. Objetiva-se a geração automática de documentação de forma a se ter uma indicação prática e simples, possivelmente através de forma gráfica, da dependência entre requisitos e o impacto das mudanças que possam advir.

6 Referências

1. Cohn, M.: Aplicando Métodos Ágeis com Sucesso: Desenvolvimento de software com Scrum. Boston, MA. (2011)
2. Schwaber, K.: Agile Software Development with SCRUM. New York, NY. (2010)
3. Beck, K.: TDD Desenvolvimento Guiado Por Testes. New York, NY. (2010)
4. Sommerville, I.: Engenharia de software. 8° Ed, São Paulo, Brasil. (2001)
5. Jaqueira, A.: Uso de Modelos i* para Enriquecer Requisitos em Métodos Ágeis. Dissestação UFRN, Natal, RN. (2013)
6. Cohn, M.: User Stories Applied: For Agile Software Development. Boston, MA. (2004)
7. Glenford, J., Wiley, J.: The Art of Software Testing. Jersey City, NJ. (2004)
8. Cohn, M.: Agile Estimating and Planning. Boston, MA. (2005)
9. Shore, J.,Warden, S.:The art of agile development, San Francisco, CA. (2008)
10. Massol, V. ,Husted, T.: JUnit in action. Los Angeles, CA. (2010)
11. A. Jaqueira, E. Andreotti, M. Lucena, E. Aranha: Desafios de Requisitos em Métodos Ágeis: uma revisão sistemática. 3rd Brazilian Workshop on Agile Methods, São Paulo, (2012)
12. M. Siniaalto, P. Abramhamsson.: A Comparative Case Study on the Impact of Test-Driven Development on Program Design and Test Coverage. Oulu, Finland. (2010)
13. D. Janzen, H. Saiedian.: On the Influence of Test-Driven Development on Software Design. Kansas,USA. (2011)
14. F. Ricca, M. Torchiano, M di Pienta, M. Ceccato, P. Tonella.: Using Acceptance Tests as a Support for Clarifying Requirements: a Series of Experiments. Trento, Italy. (2011)
15. Silva, M.; Moreno, A. Automação em Testes Ágeis. Revista de Sistemas e Computação, v. 1, n. 2, p. 139-164, Salvador (2011).