Anteprojeto de Tese de Doutorado

Sobre a Adoção de Boas Práticas de Desenvolvimento de Software em Projetos de Software Livre

Rodrigo Rocha Gomes e Souza
rodrigo@dcc.ufba.br

Dezembro de 2009

1 Objetivo

No contexto de desenvolvimento de software, boas práticas são estratégias ou atividades que, segundo mostra a experiência, contribuem para o sucesso de um projeto de software. Apesar das vantagens, nem sempre é fácil aplicar boas práticas em um projeto de software.

O principal objetivo desta pesquisa é entender como a adoção de boas práticas de desenvolvimento de software é afetada por variações na forma de organização da equipe de desenvolvedores e por características do projeto. Pretende-se assim identificar em quais contextos determinadas práticas são efetivamente aplicadas e entender os mecanismos que reforçam a adoção de práticas de desenvolvimento de software.

A análise será focada nas atividades de implementação, documentação e controle de qualidade. Serão investigadas, entre outras, as seguintes práticas:

- planejamento e implementação de testes antes da implementação das funcionalidades correspondentes;
- implementação de testes por pessoas diferentes daquelas que implementam funcionalidades;
- integração contínua com o repositório de código-fonte;
- correção de *bugs* antes da implementação de novas funcionalidades;
- revisão de código.

Crowston e Howison [1] recentemente mostraram que os padrões de interação entre desenvolvedores variam bastante de projeto para projeto. Alguns projetos são bastante centralizados em um conjunto pequeno de desenvolvedores que

interagem com boa parte dos demais; outros possuem uma organização mais descentralizada, nas quais os desenvolvedores se organizam em grupos bem definidos. Não é claro, no entanto, como práticas de desenvolvimento se distribuem nessa rede social. O quanto variam as práticas entre diferentes grupos de um projeto?

Sabe-se há muito tempo que é alta a rotatividade de desenvolvedores em empresas de software, e pesquisas recentes mostram que o cenário também ocorre em projetos de software livre [5]. Enquanto os processos praticados por uma empresa são mantidos graças à organização hierárquica de seus funcionários, vale a pena investigar como se dá essa manutenção em projetos de software livre, especialmente os mais descentralizados, face à saída e à entrada de desenvolvedores. Será que, na eventualidade da saída dos desenvolvedores mais produtivos de um projeto, os demais desenvolvedores perpetuam as práticas aplicadas por seus antecessores? Em que casos a saída de desenvolvedores pode causar o abandono de práticas em um projeto?

Pretende-se também estudar diferenças entre as práticas adotadas por projetos com características distintas. Características de projetos a serem estudadas incluem grau de centralização, número de desenvolvedores e tamanho do software. Esse estudo poderá levar a conclusões do tipo "a prática X não é escalável: ela raramente é usada em projetos de grande porte".

2 Motivação

A aplicação de boas práticas de desenvolvimento está associada à melhoria na qualidade do software produzido [3]. A difusão de boas práticas dentro de um projeto é, ainda, uma forma de diminuir a diferença de produtividade entre desenvolvedores menos experientes e desenvolvedores mais experientes, o que contribui para tornar o processo de desenvolvimento de software mais previsível.

Existe, no entanto, uma lacuna entre a teoria e a prática do desenvolvimento de software [2]. Em particular, a difusão de uma boa prática depende não apenas dos benefícios preconizados por seus entusiastas, mas também de questões ligadas ao seu uso efetivo, como o custo de implantação e a escassez de boas ferramentas de apoio.

Por melhor que seja a argumentação, uma prática só é útil quando é efetivamente adotada no desenvolvimento de um software. Por isso, a compreensão dos fatores que influenciam essa adoção é uma parte importante do estudo de uma prática de desenvolvimento. Infelizmente, a maior parte do conhecimento disponível sobre tais fatores é proveniente de observações isoladas e estudos de caso [1]. Essas observações, embora de grande valia, são essencialmente tendenciosas. Uma compreensão mais abrangente dos fatores contribuiria para a proposta de práticas de desenvolvimento mais realistas.

A escolha feita nesta pesquisa de estudar projetos de software livre certamente limita a generalidade dos resultados, uma vez que a organização desses projetos em geral é diferente da organização de projetos comerciais [4]. Essa opção se justifica, no entanto, pela importância que tais softwares adquiriram

em diversos segmentos. Um estudo mostrou que, em 2004, mais de 1 milhão de desenvolvedores no Estados Unidos estavam trabalhando em projetos de software livre. Além disso, projetos de software livre dominam o mercado mundial de servidores web, servidores de e-mail e servidores de DNS [7].

3 Metodologia

Muitos projetos de software livre tornam disponíveis publicamente alguns artefatos produzidos durante o processo de desenvolvimento de software. Esses artefatos incluem código-fonte e documentação mantidos em sistemas de controle de versão, bem como relatórios de *bugs* e mensagens de e-mail trocadas entre desenvolvedores.

A grande disponibilidade de dados relacionados ao desenvolvimento favorece a realização de estudos em larga escala. Nesta pesquisa serão utilizados predominantemente métodos quantitativos de análise de dados para viabilizar o estudo de uma amostra significativa de projetos de software livre.

A seleção da amostra de projetos será feita com base em informações como número de desenvolvedores e tamanho do software. Serão extraídos dados de repositórios de controle de versão, sistemas de acompanhamento de *bugs* e listas de discussão. Ferramentas de análise estática de código serão usadas para extrair relações entre os diversos componentes de um sistema de software.

A identificação das práticas adotadas por um projeto será feita a partir do uso de técnicas de mineração de processos [6] sobre o histórico dos dados extraídos do projeto. Tais técnicas têm como finalidade extrair modelos de processos a partir de dados temporais.

Para responder às questões de pesquisa, serão usados métodos de análise exploratória de dados e inferência estatística, relacionando características de projetos e de desenvolvedores, o histórico de desenvolvimento e a aplicação de boas práticas. A partir dos resultados da análise poderão ser realizadas investigações mais aprofundadas sobre projetos selecionados e comparações com resultados obtidos em estudos de caso.

4 Proposta de Cronograma

As seguintes atividades serão realizados durante este projeto, de acordo com os prazos indicados na Tabela 1:

1. **Revisão bibliográfica**. Serão lidos textos sobre engenharia de software experimental, processos e práticas de desenvolvimento de software, desenvolvimento de software livre, mineração de processos e estatística, entre outros. Ao final será produzido um resumo dos tópicos mais relevantes.
2. **Realização do exame de qualificação**.
3. **Defesa da proposta de tese**.

4. **Planejamento e execução de experimentos**. Espera-se produzir pelo menos dois artigos: um com resultados preliminares e outro com os resultados finais dos experimentos.

5. **Redação da tese**.

6. **Defesa da tese**.

Atividade	2010.1	2010.2	2011.1	2011.2	2012.1	2012.2	2013.1	2013.2
1	•	•	•					
2			•					
3				•				
4				•	•	•		
5						•	•	•
6								•

Tabela 1: Proposta de Cronograma de Atividades

Referências

[1] Kevin Crowston and James Howison. The social structure of free and open source software development. *First Monday*, 10(2), 2005.

[2] Robert L. Glass. The relationship between theory and practice in software engineering. *Commun. ACM*, 39(11):11–13, 1996.

[3] Robert B. Grady. Practical results from measuring software quality. *Commun. ACM*, 36(11):62–68, 1993.

[4] Eric S. Raymond. *The Cathedral and the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary*. O'Reilly and Associates, Inc., Sebastopol, CA, USA, 2001. Foreword By-Young, Bob.

[5] Gregorio Robles and Jesus Gonzalez-Barahona. Contributor Turnover in Libre Software Projects. *Open Source Systems*, pages 273–286, 2006.

[6] Vladimir Rubin, Christian Günther, Wil van der Aalst, Ekkart Kindler, Boudewijn van Dongen, and Wilhelm Schäfer. Process Mining Framework for Software Processes. pages 169–181. 2007.

[7] David A. Wheeler. Why Open Source Software / Free Software (OSS/FS, FOSS, or FLOSS)?: Look at the Numbers! 2007.