Criação, Aplicação e os Resultados de Uma Metodologia para Testes Funcionais

Patrícia Corrêa Fonseca e Raimar Torres dos Santos[1]

Objetivos e justificativa: Atualmente as empresas buscam alcançar os estágios mais avançados de maturidade em padrões e processos, que possuam aceitação global, em escalas como o SW-CMM, CMMI, GuiaPMBOK, ISO. Enfrentam a falta de vivência e sucesso nesta integração. As técnicas de desenvolvimento de softwares são ainda incipientes, pelo menos, na prática. E esta imaturidade acarreta projetos com problemas de orçamento, prazos ultrapassados e clientes insatisfeitos. Pesquisas comprovam que o custo aumenta quando não há um planejamento adequado de desenvolvimento, através de um processo bem estruturado, com gestão de qualidade.

Entre 1960 e 1970 o desenvolvimento de software era focado na escrita de código e testes de linhas dos mesmos. Muito pouco esforço era gasto na integração de grandes sistemas. Testes eram desnecessariamente utilizados para provar que o produto estava correto. Em seguida, até 1980, os desenvolvedores começaram a aumentar o esforço nas fases de requisitos, analises e projetos preliminares. Após 1990 os investimentos de tempo e recursos para integrar diferentes partes do software e para testes foram incrementados. A quantidade de esforço para levantar os requisitos também foi incrementada juntamente com a fase de projeto, que ganhou maior importância. A tabela a seguir ilustra a porcentagem do esforço de desenvolvimento de software distribuída em cada fase, ao longo do tempo. [TAS 02]

<table>
<tr><th></th><th>Requisitos e Análise</th><th>Projeto Preliminar</th><th>Projeto Detalhado</th><th>Implementação/Testes de unidade</th><th>Integração e Testes</th><th>Testes de Sistema</th></tr>
<tr><td>1960-1970</td><td colspan="2">10%</td><td></td><td>80%</td><td colspan="2">10%</td></tr>
<tr><td>1980</td><td colspan="2">20%</td><td colspan="2">60%</td><td colspan="2">20%</td></tr>
<tr><td>1990</td><td>40%</td><td colspan="2">30%</td><td colspan="3">30%</td></tr>
</table>

Tabela 1 - Porcentagem esforço por fase por período

[1] É importante salientar que este trabalho evoluiu de um projeto piloto para uma aplicação real. Agradecimento especial à toda equipe responsável: Glênio Araújo, Laudecy Alves, Marcelo Motta, Roberto Cavalcante entre outros.

Conceito: *Testes podem ser usados para descobrir a presença de erros, mas não para mostrar a sua ausência. [DIJ]*

Testes de software é o processo de executar o software de uma maneira controlada com o objetivo de descobrir diferenças entre o comportamento previsto e o comportamento observado. O teste indica apenas que existe alguma falha. A atividade que mostra a natureza da falha e a corrige é a depuração. Melhorias em testes e implantação de métricas podem reduzir os custos de desenvolvimento de uma dada qualidade e ainda melhorar o desempenho do desenvolvimento.

Beizer [BEI 90] reportou que tipicamente o tempo gasto pelas atividades de testes corresponde a 50% do tempo gasto no desenvolvimento do programa.

Segundo Myers, as seguintes regras podem indicar os objetivos de testes: [MYE 95]

- Testes é o processo de executar um programa com a intenção de encontrar erros.
- Um bom caso de teste é um que tem alta probabilidade de encontrar um erro ainda não descoberto.
- Um bom teste é o que encontra um erro ainda não descoberto.

Princípios básicos de testes: Davis, em [DAV 95], sugere um conjunto de princípios básicos de testes de softwares, adaptados por Pressman, que são: [PRE 97]

- Todos os testes devem rastrear os requisitos;
- O planejamento das atividades de testes deve ser iniciado tão logo os requisitos estiverem definidos;
- O principio de Pareto implica que 80% dos erros serão rastreados em 20% de todo o código;
- O teste deve começar “in the small” e terminar “in the large”, ou seja, os primeiros planejamentos e execuções de testes focam em módulos individuais. Com o progresso do testes, tais planejamentos e execuções focam em módulos maiores ou em módulos integrados;
- Não é possível fazer testes exaustivamente;
- Para ser mais eficaz, o teste deve ser conduzido por uma terceira pessoa, não sendo a que codificou o sistema.

Impactos devido à infra-estrutura inadequada de testes: Uma das maiores inadequações com a infra-estrutura de testes de software é a dificuldade em determinar se a aplicações e sistemas irão inter operar. Por exemplo, se a aplicação A e aplicação B inter operam e se a aplicação B e a aplicação C inter operam, quais são as perspectivas da aplicação A e C inter operarem? [NIS 97]

Automatizar a geração de códigos de testes pode consumir mais tempo e ser mais caro que desenvolver o padrão ou produto que será testado. [TAS 02]

A falta de métodos rigorosos para determinar quando um produto está bom o bastante para ter uma versão liberada conduz desenvolvedores a usarem combinações de métodos não analíticos para decidirem quando o software está bom para ser liberado. O problema é devido não só ao desacordo em definir o quanto é o suficiente de ser testado, mas também em saber quais testes devem ser executados para determinar esta suficiência. [TAS 02]

Infra-estrutura adequada de testes permite desenvolvedores encontrarem e corrigirem mais erros tão logo quanto possível com menor custo. O melhor beneficio de melhorias na infra-estrutura é a redução do custo no processo de desenvolvimento e a redução de suporte ao consumidor após a venda do produto. Um benefício adicional é o aumento da confiança na qualidade do produto.

Melhorias na infra-estrutura de testes incluem: ferramentas de testes de integração, geração automática de códigos de testes, métodos para determinar nível de qualidade aceitável das versões, métricas e processos de medições.

A implantação de processo de testes de software numa organização não é tarefa trivial. Para ter sucesso, precisa ter o comprometimento da gerencia, papeis bem definidos na equipe, profissionais especializados, devem ser disponibilizados ferramentas e recursos e para alcançar níveis superiores de maturidade, precisa-se de tempo de implantação. Não é na primeira experiência que tudo funcionará bem.

Realizar testes para verificar se os requisitos são atendidos é uma forma de garantir a qualidade e também reduzir custos, à medida que antecipa a identificação de falhas. Reservas de recursos para atividades de testes inseridas no planejamento do desenvolvimento de software vêm se tornando cada dia mais comum. Porém esta prática ainda é carente de técnicas padronizadas que a direcione, sem deixar que seja subjetiva à experiência do líder de equipe, isto, considerando ainda o melhor caso, onde existe uma equipe separada para tais atividades.

Essas atividades são alvos de estudos promissores: desenvolver práticas eficazes e eficientes de planejamento, identificar quando os testes já executados são eficientes, dimensionar as equipes, medir a confiabilidade considerando sistemas cada vez maiores, mais complexos, cada vez com maior necessidade de integração a outros sistemas. E de acordo com as características e necessidades de cada projeto, vários testes podem e devem ser planejados, como, por exemplo, testes de carga, de integração, de unidade, funcionais.

Por ser uma área ainda com muitas carências e que, ao mesmo tempo, tem forte impacto no processo de desenvolvimento de software, surgiu o interesse por realizar este trabalho, em parceria com a Datamec/Unisys, o que permitiu aplicação e avaliação dos resultados.

Descrição do produto: a principal contribuição deste trabalho consiste em uma metodologia, de natureza exploratória e aplicada, dirigida a soluções para o problema específico de testes funcionais. Esta foi desenvolvida a partir de uma base teórica e de entrevistas com profissionais experientes. Foi aprimorada através de aplicação prática, inicialmente em um projeto piloto e em seguida, em projetos reais. Juntamente com a metodologia são divulgados á comunidade: características dos ambientes onde a metodologia foi desenvolvida e aprimorada, resultados obtidos e lições aprendidas com sua aplicação. Este estudo é valioso para empresas que tenham características semelhantes ao ambiente utilizado na experiência e que pretendam investir em qualidade e produtividade a partir de uma metodologia para testes funcionais.

Metodologia de execução: natureza aplicada, de objetivo exploratório que aborda qualitativamente o problema. A coleta de dados foi realizada através de observação no ambiente real e em entrevistas (não estruturada) com profissionais experientes.

Resultados Relevantes: O primeiro produto deste trabalho encontra-se documentado na monografia. Registra uma revisão bibliográfica de qualidade de software, dos conceitos de testes e sua importância no processo de desenvolvimento. Lições aprendidas com a experiência prática permitem que os processos sejam continuamente ajustados, tornando-se progressivamente mais adequados, robustos, estáveis e previsíveis. Considerando a importância da experiência prática, entrevistas para angariar conhecimento de profissionais experientes em testes foram realizadas com o intuito de subsidiar a definição da estratégia de

implantação do processo de testes. É importante salientar que os relatos dos colaboradores são valiosos na medida em que evidenciam as situações cotidianas e o que isso acarreta no futuro. Porém, antes de explorá-los, deve ser realizado um trabalho de categorização, considerando: o porte da organização; a maturidade de processo de desenvolvimento; o tamanho da equipe e a característica dos produtos desenvolvidos, para evitar conclusões incorretas. Em seguida é apresentada uma caracterização da indústria em estudo (maturidade, caracterização do ambiente e da equipe) e a estratégia para implantação de testes (definição de métricas, personalização dos artefatos, ferramentas e planejamento de capacitação da equipe), conclusões e contribuições.

A continuidade do trabalho consistiu na aplicação da metodologia e a avaliação dos efeitos, que estão apresentados sucintamente na conclusão deste artigo.

O trabalho apresenta segmentado em:

- Uma monografia de especialização: “Estudo Sobre Implantação de Práticas Sistemáticas de Testes no Desenvolvimento de Software”.

- Uma metodologia baseada na experiência prática, plausível de ser realizada em empresas que apresentam as características focadas pelo projeto;

Aplicabilidade dos resultados e principais impactos na infra-estrutura física da instituição (aquisição de equipamentos e/ou ferramentas), região, município e país: auxiliar empresas brasileiras a implantar práticas de testes de forma eficaz. O trabalho abrange todas as empresas que não tem um processo bem estabelecido de testes e que se enquadram em determinadas características, dentre elas: priorização de software livre; utilização de testes, porém sem um processo sistemático e sem política de métricas.

Características Inovadoras: A construção é a partir de metodologias de testes já existentes e agrega experiências aplicadas na prática além de tratar as dificuldades encontradas por outros profissionais experientes.

Conclusão e Perspectivas Futuras: A partir da teoria e dos estudos de casos apresentados, podemos observar algumas características marcantes da aplicação da metodologia. No início, três projetos de softwares utilizaram a metodologia para seu desenvolvimento. Atualmente, o processo encontra-se implantado, com uso institucionalizado dos

artefatos. Estas foram alterações estruturais na organização que auxiliaram alcançar grandes benefícios comprovados pela obtenção do nível 3 do CMMI em dezembro de 2005.

O conhecimento do que precisa ser testado estava “na cabeça” do analista que testava, era um risco, pois a qualidade estava intimamente ligada à pessoa e não ao processo. Após a institucionalização do processo esse risco foi drasticamente minimizado.

Os bugs (erros, falhas e melhorias) encontrados passaram a ser registrados e acompanhados através da utilização da ferramenta GENESIS, desenvolvida internamente. A dificuldade de reprodução dos bugs é pouco freqüente. A técnica utilizada para definir quando parar os testes é o tempo alocado para cada atividade.

A ferramenta GENESIS foi desenvolvida para dar suporte ao processo, não apenas de testes, mas da organização como fábrica de software. Tem outras funcionalidades, além do acompanhamento de bugs. Dentre elas, é possível gerenciar atividades diárias entre as áreas funcionais, utilizando o conceito de WBS (Work Breakdown Structure) e registrar horas trabalhadas. O suporte provido por esta ferramenta foi de grande importância. Permitiu obtenção fácil de algumas métricas normalmente difíceis de serem coletadas, com, por exemplo, horas trabalhadas por atividades específicas.

Com as métricas coletadas, foi possível identificar e ordenar as funcionalidades mais instáveis fazendo com que o planejamento de testes para tais funcionalidades estabelecesse maior esforço. Esse comportamento identificado está conforme o princípio de Pareto, onde 80% dos problemas estão em 20% do módulo.

Referências Bibliográficas:

[FON] FONSECA, Patrícia C., *“Estudo Sobre Implantação de Práticas Sistemáticas de Testes no Desenvolvimento de Software”*, 2005.

[BEI 90] BEIZEL, B. *Software System Techniques.* New York: Van Nostrande Reinhold, 1990.

[BOE 81] Boehm, B. *Software Engeneering Economics.* Prentice-Hall, 1981, p37.

[DAV 95] DAVIS, A. 201 *Principles of Software Developmente*, McGray-Hill, 1995

[DIJ] DIJKSTRA, Edsger W.

[IEE 93] Institute of Electrical and Electronics Engineers/American National. Standards Institute (IEEE/ANSI). 1993. “Software Reliability.” Washington, DC: American Institute of Aeronautics and Astronautics.

[ISO/IEC 25000] ISO/IEC FCD 25000 Software Engineering -- Software Product Quality Requirements and Evaluation (SQuaRE) -- Guide to SQuaRE

[ISO/IEC 9126] SOFTWARE ENGINEERING - Product quality – Quality in use metrics

[MCC 77] MCCALL, J., P. Richards, and G. Walters. *Factors in Software Quality*,1977.

[MYE 95] MYESS, Glenford J. *The art of software testing* 1995

[NIS 97] National Institute of Standards and Technology

[PRE 97] Pressman, Roger S. *Software Engineering – A practitioner’s approach* 1997 Fourth Edition

[SAN 04] SANTOS, Luana G. N. O. *Testes de Software melhoria de qualidade* - Monografia de final de Curos 2004

[TAS 02] TASSEY Gregory, Ph.D. *The Economic Impacts of Inadequate Infrastructure for Software Testing*, 2002