METODOLOGIAS DE TESTE DE SOFTWARE

Davi do Amaral, Renato Gomes, Rodrigo Passos de Jesus, Tiago Marques de Araujo, Elias E. Goulart

Centro Universitário Fundação Santo André
Av. Príncipe de Gales, 821 – Santo André – SP.

{profelias_fsa, tiagomarx}@yahoo.com.br

Resumo:
As tecnologias para testes de programas são umas das principais preocupações atuais dos desenvolvedores. As grandes complexidades dos programas, aliado com a reutilização de código impõe novos desafios para a manutenção da qualidade dos produtos. Este trabalho elabora um levantamento da bibliografia atual da área, combinada com um estudo de caso em uma das principais companhias responsáveis pela certificação de programas para computadores. O objetivo é a investigação do estado-do-mercado no Brasil, nesta área.

Palavras-Chave: testes de programas, qualidade de programas, ferramentas de teste.

Absctract:
The technologies for software testing are one the most important attention for development people. The computer programs are big, have high complexity and aside with the reutilization aspects, impose news challenges to maintain its quality. This work starts with a bibliographical review and is combined with a brief case study about an important certification company for softwares. The main objective is investigating the state-of-market at Brazil.

Key-words: software testing, software quality, testing tools.

1. INTRODUÇÃO

Este artigo irá abordar um assunto que está diretamente proporcional à qualidade de software, trata-se de Teste de Software. Este processo consiste em executar o software de maneira controlada com a finalidade de avaliar se o mesmo se comporta conforme o especificado. Todas as metodologias de desenvolvimento de software têm uma disciplina dedicada aos testes, porém muitas vezes é efetuada de maneira inconsistente, podendo gerar problemas futuros.

Com um mundo cada vez mais influenciado pela tecnologia e a eficiência da informática reconhecida, a demanda por softwares só tende a crescer. E o quesito qualidade é de extrema importância para uma maior confiabilidade, principalmente em sistemas críticos, onde o erro pode comprometer a estrutura de uma empresa. Isso torna o teste uma das etapas fundamentais no processo de criação de software.

Partindo-se da necessidade de um maior domínio do assunto, foram feitas análises com base em pesquisas procurando focar as Metodologias de Teste de Software, com a intenção de verificar quais as semelhanças, as vantagens e as desvantagens de cada uma. O objetivo é apresentar os resultados de um estudo de caso feito em uma empresa especializada em testes e agregar ao material pesquisado.

Para realizar este estudo foi utilizada uma metodologia englobando fichamentos, que consiste em analisar artigos científicos publicados na internet e estudo de caso, no qual foi feita uma pesquisa na “Certified Technologies”, empresa especializada em testes de software, com o intuito de colher informações para complementar e melhor entender o tema.

2. METODOLOGIAS DE TESTE DE SOFTWARE

2.1 Conceituação

A tecnologia da informação é utilizada por empresas que querem ter competitividade no mercado, alcançar dados e informações que contribuam nas tomadas de decisões e consequentemente, ter uma vantagem sobre os concorrentes. Com isso, a tecnologia tem que dar o seu retorno, ou seja, passar a confiança necessária para que a empresa possa investir sem medo de prejuízos. No mercado existem inúmeros sistemas de risco que não permitem falhas e trabalham com informações que precisam ser íntegras e satisfatórias à sua solicitação.

Devido aos métodos de desenvolvimento e complexidade dos softwares, é muito difícil dizer que estejam isentos da presença de erros. O erros podem ocorrer por causa de problemas na especificação dos requisitos, problemas na modelagem de negócio, no modo que a funcionalidade deve ser desempenhada, na complexidade do sistema e na mudança de requisitos. Todos os desenvolvedores estão suscetíveis a erros de programação, ainda mais em sistemas de nível alto de complexidade. Para isso, existe um atividade em que se pode avaliar, testar e corrigir tais problemas. Este processo é chamado de Teste de Software, no qual é feito de diversas maneiras e usando diversas metodologias e que pode ser visto como uma parcela do processo que garante a qualidade de software.

O teste deve ser criado com intenção de descobrir o maior número de erros possíveis. Testar o software é a última chance de avaliar a qualidade antes de entregar para o cliente, pois entregar o software com erro é algo muito desagradável e gera insatisfação e desconfiança.

Em um desenvolvimento de software de qualidade, deve-se seguir uma sólida metodologia de trabalho desde o início para não acarretar problemas futuros. Mas com a

pressão dos clientes para a entrega, essa metodologia acaba por ficar inconsistente e gerar problemas que resultam em prejuízos que poderiam ser evitados.

Em meados da década de sessenta, o conceito de testes de software começou a ser utilizado. Sendo tratado como um processo separado e não mais como uma simples revisão de código dos desenvolvedores. Como já era esperado, esta nova metodologia provocou melhoras na qualidade dos sistemas, mas não significou grande redução de custos para os projetos até então. Não significou porque essa prática isolada não era suficiente, ela depende de se testar o software desde o inicio. Hoje em dia, podemos observar a eficiência dos testes na qualidade do sistema que, se bem estruturados e distribuídos, resultam em custos mais baixos. Apesar disso, existem fabricas de software que trabalham como antigamente, e não entendem a causa dos prejuízos financeiros na realização de seus projetos.

2.2 Etapas de Testes

Foram desenvolvidas inúmeras formas de se testar um software, sendo que todos querem alcançar o mesmo objetivo, o de encontrar e remover falhas. A seguir as etapas deste processo:

- Planejamento: é feito um cronograma dos testes que serão realizados contendo as equipes que participarão, como será dividido etc.

- Projeto dos testes: nesta etapa são elaborados e definidos os casos de teste, que são informações fornecidas ao software para saber como ele se comporta.

- Implementação dos testes: Os casos de testes definidos para o teste são implementados geralmente por meio de uma ferramenta de apoio.

- Execução dos testes: depois dos passos anteriores seguidos, é feita a execução do teste propriamente dito.

- Avaliação dos testes: é analisado os resultados para saber se atendeu o especificado, quais os problemas identificados etc.

2.3 Técnicas de Testes

A seguir, algumas das técnicas utilizadas no mercado:

- Caixa branca: identifica os erros de programação, processos estruturais, questões de decisões lógicas, laços, repetições. Seu objetivo é testar unidades de software e seu processo é realizado durante todo o desenvolvimento do software, ou seja, os testes devem estar presentes durante todas as etapas.

- Caixa preta: nesta técnica, não interessa os códigos que estão ali, se entra-se com os dados e verifica-se a saída. Não se tem acesso ao meio do processo. Possibilita identificar erros de interface, acesso aos dados de um banco, de desempenho e velocidade da aplicação. Realizado no fim do desenvolvimento de software, mas também pode ser utilizado durante o processo.

- Caso de testes: a confiabilidade depende da seleção dos casos de teste. O que interessa é o que foi obtido no final, assim é gerado um documento com a especificação das entradas e os resultados que são esperados. São dois os tipos de casos: gerados com base na análise das Entradas x Saídas e gerados com base na análise de trechos do código.

- Ferramentas para testes: visam automatizar todo o processo de teste e, com isso, reduzir tempo e recursos. Neles se encontram propriedades de analisadores estáticos, geradores de dados de teste e auditores de código.

2.4 Tipos de Testes

Cada um desses testes tem as suas particularidades no sistema como um todo. A seguir, os tipos existentes:

- Unidade: são testes de exclusividade do código e da interface disponibilizada de uma unidade. São testadas as classes, os métodos ou até pequenos trechos de código.

- Integração: testa a composição de artefatos já testados como se fosse uma unidade.

- Validação: testa a conformidade com os requisitos do software e verifica se é o que o cliente deseja.

- Sistema: testa a relação entre os elementos do sistema como hardware, software e banco de dados. É testado o sistema como um todo, aqui não se vê apenas o software e sim o sistema.

2.5 Técnicas de injeção de defeitos em softwares

No mercado existem muitos sistemas críticos que não podem ser submetidos a erros, os chamados sistemas críticos. Um exemplo seria o sistema de controle de tráfego aéreo que, um erro poderia se fatal e comprometer a vida de muitas pessoas. Para esses tipos de sistemas, muitas técnicas e metodologias vem sendo desenvolvidas, mas existe um processo muito utilizado que é a injeção controlada de defeitos e a análise e o estudo do comportamento dos defeitos na segurança do sistema. Esta técnica, denominada “Análise de Mutantes”, é um tipo de teste baseado em defeitos que geralmente é introduzido na fase de desenvolvimento do software. O objetivo principal deste critério é encontrar um conjunto de dados de teste A que consiga matar o maior número possível de mutantes não equivalentes, tornando assim o A adequado para o teste do programa B.

Uma ferramenta denominada ITool foi desenvolvida seguindo este conceito, com a intenção de ajudar na resolução de problemas relacionados a sistemas críticos. A ITool é utilizada em plataformas como Solaris e Linux e está dividida em três partes: a interface gráfica, os módulos funcionais executáveis e a base de dados. Sua funcionalidade consiste em sessões de teste que serão confrontados com o programa original que contém os defeitos injetados e ainda oferece inúmeras funcões de teste.

A ferramenta encontra-se em fase de teste e melhorias e pode ser utilizado para o ensino acadêmico. Para a sua realização, foram considerados a características mínimas presentes em uma ferramenta de testes. Passou por um experimento piloto em um sistema complexo para sua validação, visando cada vez mais a sua melhoria.

Com o teste de Software sendo atualmente de suma importância para uma maior confiabilidade de softwares críticos, a criação de ferramentas como a ITool, com alta tecnologia e inúmeras funcionalidades são cada vez mais importante. Os experimentos com sistemas complexos ajudam muito no amadurecimento das ferramentas, portanto a evolução dessas tecnologias só vem a contribuir para a confiabilidade dos sistemas de informação.

2.6 O UML como ferramenta para casos de testes

A UML (Unified Modeling Language) é uma linguagem para especificação, documentação, visualização e desenvolvimento de sistemas orientados a objetos. Sintetiza os principais métodos existentes, sendo considerada uma das linguagens mais expressivas para modelagem de sistemas orientados a objetos. Esta linguagem pode ser utilizada como ferramenta para a criação de casos de testes de software e apoiando o planejamento de testes funcionais.

O objetivo do uso da UML para testes de software, está sobre o apoio que essa ferramenta, junto com o padrão IEEE829, provê a validação dos requisitos implementados. Além disso, a UML oferece uma boa documentação gerada das especificações dos casos de uso. A proposta padroniza a utilização de casos de uso no teste funcional na geração de casos de testes a partir dos casos de uso e no teste de casos de uso estendido.

O primeiro padrão é baseado no fluxo principal, e nos cenários possíveis que podem ocorrer, avaliado por n tipos de entradas/saídas. O segundo, tem por objetivo desenvolver testes de sistemas pela modelagem dos requisitos essenciais em forma de casos de uso estendidos.

Uma ferramenta de auxílio dos testes, foi desenvolvida, baseada nas metodologias descritas, possibilitando a documentação do projeto de teste, seus dados de teste e o procedimento de cada caso de teste.

O desenvolvimento dos casos de uso propostos seguem os seguintes padrões:

1) Ler a descrição do caso de uso, o fluxo principal e os fluxos alternativos do caso de uso;
2) Identificação das pré-condições e pós-condições;
3) Desenvolvimento das especificações de teste de cada um dos cenários;
4) Identificação das variáveis operacionais para o cenário principal;
5) E por ultimo, desenvolver os procedimentos para execução dos testes.

O método e a ferramenta facilitam e organizam todo o processo de teste. Contribui para a re-execução dos testes e além disso, a utilização dos casos de uso é facilitada pela a elicitação de requisitos, e a análise está intimamente ligada ao teste e a validação de requisitos.

2.7 Testes Automatizados

Controlar a qualidade de software é um grande desafio. Esse desafio pode ser superado, através da utilização de Métodos Ágeis de Desenvolvimento de Software, realizadas pela Programação extrema, focada nos Testes Automatizados. A Programação extrema ou XP, é

uma metodologia ágil para equipes pequenas e médias e que irão desenvolver software com requisitos vagos e em constante mudança. Para isso, adota a estratégia de constante acompanhamento e realização de vários pequenos ajustes durante o desenvolvimento de software.

O cenário comum de desenvolvimento se da por, estudar o problema, pensar em uma solução e implementá-la. Após isso, os testes são realizados pelo desenvolvedor, usuários ou equipes especializadas, de forma manual. Esse método é dispendioso e cansativo, trazendo prejuízo as empresas e não garantem qualidade do software final.

A Programação extrema recomenda testes automatizados, que são programas ou scripts simples que exercitam funcionalidades de forma automática, possibilitando a repetição dos testes quantas vezes necessárias e mais elaborados.

Os testes automatizados devem começar pelos testes de unidade, que são pequenos trechos como funções, métodos ou classes, podendo ser realizados pelas ferramentas chamadas de arcabouços, como o JSUnit, CSUnit e etc.

Outro tipo de teste importante é o de aceitação, que é bem abrangente, pois envolve todas as camadas do sistema desde a modelagem correta da base de dados até a interação do usuário com a interface.

Os testeS de software automatizados dão suporte ao complexo mundo de desenvolvimento de software, dando mais segurança à equipe para realizar qualquer tipo de mudança no software garantindo a qualidade do produto final.

2.8 Testes Funcionais

A técnica de teste funcional tem o objetivo de avaliar o comportamento externo do componente de software, sem considerar seu comportamento interno. Os dados de entrada são fornecidos, o teste é executado e o resultado obtido é comparado a um resultado esperado previamente conhecido. Como detalhes de implementação não são considerados, os casos de teste são todos derivados da especificação.

Um tipo de abordagem de teste funcional em programas Java, utilizando uma ferramenta chamada AFTT (*Aspect-based Funcitional Testing Tool*), oferece suporte a alguns critérios funcionais e é discutida pela Programação Orientada a Aspectos (POA). Ela apóia a implementação de forma modular, motiva as pesquisas nas áreas da Engenharia de Software e propõe sua aplicação para construção, fases de engenharia de requisitos, modelagem e teste de programas.

Os testes podem ser encapsulados sob a forma de aspectos, separando o código funcional e o código de teste, levando a criação de cenários de teste de forma mais rápida. O teste funcional tem como objetivo encontrar discrepâncias entre o comportamento atual do sistema em relação a sua especificação. O critério que se destaca é o particionamento em classes de Equivalência. A instrumentação consiste em inserir código auxiliar no programa testado, serve para obter informações ou medidas referentes a sua execução. Algumas ferramentas conhecidas são: o Java Instrumentation Engine, o Guaraná, o Javassist e BCEL e o POA com o auxílio do AspectJ.

A AFTT é uma ferramenta que utiliza aspectos para auxiliar teste funcional de programas Java. Ela disponibiliza uma interface gráfica que permite o testador ter acesso as principais funcionalidades do software e por meio desta os casos de teste podem ser executados, habilitados e desabilitados.

A Programação Orientada a Aspectos traz benefícios ao teste funcional e pode ser utilizada para instrumentação e teste de programas, junto, apoiado pela a AFTT, com ferramenta para análise funcional de cobertura.

3. ESTUDO DE CASO

Para realização do estudo de caso, foi feito um levantamento de informações com um funcionário da “Certified Technologies”, uma empresa do segmento de TI voltada para Teste e Certificação de Software. Com expertise em certificações integradas, a empresa executa testes de sistemas em alta e baixa plataforma, meios de pagamento eletrônico e aplicativos mobile para as tecnologias CDMA, GSM e 3G.

O seu processo de teste e certificação de software é automatizado e gerenciado a partir da utilização de ferramentas próprias e de terceiros. A empresa utiliza uma metodologia com base na norma ISO9126, pois acredita que garante funcionalidade, confiabilidade, usabilidade, eficiência, manutenibilidade e portabilidade dos produtos testados. Existem diversas metodologias no mercado, mas para eles que estão direcionados ao “nicho” de Captura Financeira esta é a mais indicada.

A norma ISO9126 é um certificado de qualidade do processo de software, ou seja, é um conjunto de atributos que têm impacto na capacidade do software de manter o seu nível de desempenho dentro de condições estabelecidas por um dado período de tempo. Seus objetivos são:

1) Definir os requisitos de qualidade de um produto de software;

2) Avaliar as especificações do software durante o desenvolvimento para verificar se os requisitos de qualidade estão sendo atendidos;
3) Descrever as características e atributos do software implementado;
4) Avaliar o software desenvolvido antes da entrega ao cliente;
5) Avaliar o software desenvolvido antes da aceitação pelo cliente.

Quanto à utilização das metodologias, é seguida basicamente a mesma para grande parte de seus clientes, já que seus softwares estão voltados para captura eletrônica de meios de pagamento. Os critérios que devem ser levados em conta na escolha da metodologia a ser utilizada são:

1) “Cobertura” todas as funcionalidades físicas e sistêmicas do produto testado;
2) A ferramenta a ser utilizada nos testes deve garantir a integridade e confiabilidade do Software;
3) O Software testado deve ser eficaz e utilizável por qualquer tipo de pessoa;
4) O Software deve permitir manutenção remota e local;
5) Todo Software deve suportar mudanças e estar compatível com outras tecnologias presentes no Mercado.

Uma das maiores vantagens que a empresa conseguiu visualizar com a sua metodologia direcionada para testes de software em meios de Pagamento foi o Know How adquirido, tendo em vista que são os pioneiros a atender essa parcela do mercado. E a respeito da utilização do Device (ferramenta de testes inédita no Brasil) que foi adquirido à pouco tempo, a maior vantagem foi a exclusividade no mercado. Cada vez mais tem aparecido novas ferramentas e metodologias de testes. O mercado nessa área encontra-se em expansão e evolução, hoje se tem normas e procedimento escritos por orgãos certificados e qualificados mundialmente, os exemplo são o SW-CMM, SE-CMM, ISO 15504 e o mais conhecido CMMI. Aqui no Brasil existem também orgãos certificadores de profissionais, por exemplo o IBQTS e CBTS.

A empresa tem um cliente, por sinal uma operadora de telefonia muito grande, que utilizam seus serviços para testar jogos e aplicativos em seus modelos de celulares. Esse cliente exige uma metodologia chamada Self Tests, onde esses testes são feitos randomicamente. Existem grandes divergências de custos na aplicação das diferentes metodologias, como no caso de testes automatizados, os custos, comparados aos outros tipos de testes, são mais elevados. Ex.: Na empresa, são testadas aplicações de celulares do mundo

inteiro remotamente, e isso só é possível através de uma Máquina/Software chamado “Device Anyware”. A Certified Technologies é a única possuidora de tal tecnologia no país, consequentemente a empresa dispendeu de um custo muito alto para aquisição do mesmo.

Apesar de saberem que os testes estão relacionados com a qualidade dos softwares, muitas empresas não aderem às metodologias de teste de software, mesmo sabendo que isso aumentaria exponencialmente a qualidade, integridade e confiabilidade dos Sistemas desenvolvidos. E isso se deve grande parte pelo fator financeiro, pois nem todas as empresas possuem infra-estrutura e verba para realizar testes em seus produtos.

4. CONSIDERAÇÕES FINAIS

O teste de software é uma prática extremamente promissora. Tudo gira em torno de informação, quer para a boa organização de uma empresa, elaboração e melhoria de processos e até, senão a mais importante, para tomada de decisões. Toda essa estrutura é formada por sistemas de informações, criados por empresas dedicadas apenas ao seu desenvolvimento. Estando a par da situação que a sociedade se encontra, vemos dois principais indicadores para que sejam abertas grandes portas ao teste de software.

O primeiro é a necessidade de extrema confiabilidade nas informações que as empresas necessitam, logo, seus sistemas têm de ser igualmente confiáveis, isso significa não ter inconsistência nos dados, perigos de perda de informações, entrada de dados incompletos, entre outros. Isso exige um grande tempo de análise e testes práticos pelas empresas desenvolvedoras, ai então que entra o segundo indicador. As empresas desenvolvedoras muitas vezes não têm recursos a serem gastos com tais praticas, por isso recorrem às empresas especializadas em teste de software. Elas oferecem inúmeras metodologias de testes num nível de especialização que as empresas desenvolvedoras dificilmente alcançariam, caso não colocassem isto como foco em suas realidades.

Com grande satisfação concluímos este trabalho, pois clareamos nossa mente, assim como clarearemos as mentes de prospectivos leitores, a respeito de um ramo da tecnologia que está relativamente impopular no mercado, mas prossegue mostrando grandes índices reais de crescimento.

5. REFERÊNCIAS

Introdução ao Teste de Software. Disponível em:
< www.univasf.edu.br/.../Aula017_V&VTesteSoftware(resumida).pdf >
Acesso em: 2 set. 2009.

BERNARDO, P. C., KON F. **A importância dos Testes Automatizados**. Disponível em: <http www.ime.usp.br/~kon/.../EngSoftMagazine-IntroducaoTestes.pdf >
Acesso em: 2 set. 2009.

ROCHA A. D. R.; SIMÃO, A. S.; MALDONADO, J. C.; MASIERO, P. C. **Teste Funcional: Uma abordagem Auxiliada por Aspectos**. Disponível em: <wiki.dcc.ufba.br/pub/WASP04/.../WASP04-rocha-final.pdf >
Acesso em: 20 ago. 2009.

NAKAGAWA, E. Y.; MALDONADO, J. C. **ITOOL: Uma Ferramenta para Injeção de Defeitos de Software**. Disponível em: <www.inf.ufsc.br/~sbes99/anais/Sessao-Ferramenta.../18-itool.pdf>
Acesso em: 2 set. 2009.

FONTANA. C. **Avaliação da relação entre eficácia e custo na atividade de teste de software**. Disponível em: <http://www.unimep.br/phpg/mostraacademica/anais/5mostra/1/43.pdf>
Acesso em: 20 ago. 2009.

CRESPO. A. N.; et. al.. **Uma metodologia para teste de Software no Contexto da Melhoria de Processo**. Disponível em: <http://www.dsgconsult.com.br/newsCONFLITO.asp>
Acesso em: 15 set. 2009.

Certificado ISO9126. Disponível em: <http://inf.unisul.br/~vera/egs/ISO9126.htm>
Acesso em: 20 ago. 2009.

Teste de Software. Disponível em: < Fonte: http://pt.wikipedia.org/wiki/Teste_de_Software >
Acesso em: 1 out. 2009.

BIANCHINI, J. ; GRAHL, E. **TestCen: Ferramenta de Suporte ao Planejamento de Teste Funcional de Software a partir de Diagramas de Caso de Uso**. Disponível em: <http://www.dsgconsult.com.br/newsCONFLITO.asp>
Acesso em: 15 set. 2009.

Requisitos de Metodologias de Teste de software para processos ágeis. Disponível em: <homepages.dcc.ufmg.br/~rodolfo/dcc823-1-09/.../igor2.pdf >
Acesso em: 1 out. 2009.

TOZELLI, P. **Métodos de Teste de Software**. Disponível em: < imasters.uol.com.br/...software/teste_de_software/>
Acesso em: 15 set. 2009.

Teste de Software Orientado a Objetos a partir de Especificações UML. Disponível em: < ccic.claretianas.br/ccic2003/resumos/resumofer.doc >
Acesso em: 1 out. 2009.