ESCOLA POLITÉCNICA
DE PERNAMBUCO

d.s.c. UPE
Departamento de Sistemas e Computação

PROCESSO DE AUTOMAÇÃO DE TESTES DE SOFTWARE

Trabalho de Conclusão de Curso

Engenharia da Computação

Paulo César de Oliveira
Orientador: Prof. Dra. Cristine Martins Gomes de Gusmão

Recife, junho de 2008.

UNIVERSIDADE
DE PERNAMBUCO

ESCOLA POLITÉCNICA
DE PERNAMBUCO

d.s.c. UPE
Departamento de Sistemas e Computação

PROCESSO DE AUTOMAÇÃO DE TESTES DE SOFTWARE

Trabalho de Conclusão de Curso

Engenharia da Computação

Este Projeto é apresentado como requisito parcial para obtenção do diploma de Bacharel em Engenharia da Computação pela Escola Politécnica de Pernambuco – Universidade de Pernambuco.

Paulo César de Oliveira
Orientador: Prof. Dra. Cristine Martins Gomes de Gusmão

Recife, junho de 2008.

UNIVERSIDADE
DE PERNAMBUCO

Paulo César de Oliveira

PROCESSO DE AUTOMAÇÃO DE TESTES DE SOFTWARE

Resumo

A tecnologia está fortemente presente no dia-a-dia das pessoas, seja através de hardware ou de software. O software hoje está presente no pequeno comércio, nas pequenas cidades, bem como nas máquinas das grandes indústrias, automóveis e aviões. Com o mercado cada vez mais competitivo e necessitando de aplicações que detenham alta segurança, existe a necessidade de se desenvolver software com alto nível de qualidade. Uma forma de garantir a qualidade de um sistema é através do Teste de Software. O teste de software é uma atividade indispensável nos ciclos de vida de desenvolvimento de software. O objetivo de se testar é encontrar falhas no software, que poderão ser corrigidas e conseqüentemente aumentar a qualidade do software em desenvolvimento. À medida que o software cresce, mais testes são necessários para eliminar e prevenir a existência de defeitos, de forma a garantir uma maior qualidade, porém, quanto mais testes são criados, mais serão executados, mais recursos humanos e de infra-estrutura são necessários. Em contrapartida, o tempo para se testar geralmente não é o suficiente, fazendo com que seja necessário diminuir o tempo de execução dos testes através da automação. A automação de testes resolve alguns problemas, porém, pode trazer outros. Neste contexto, este trabalho propõe desenvolver um Processo de Automação de Testes de Software, com seus papéis, artefatos e atividades, todos pensados de forma a alcançar o objetivo principal que é tornar a execução dos testes mais ágil. Com este processo, busca-se contribuir com os times de teste que desejam automatizar seus testes, prezando pela qualidade e sem criar gargalos para a organização.

Abstract

The technology is ubiquos, through hardware or software. Nowadays, software is present in the small business at the smallest cities as well as inside machines of big factories, cars and aircrafts. As the market is increasingly competitive and requiring complex applications, there is a growing pressure to develop software with high level of quality. One way to ensure the quality of a system is through Software Testing. Software Testing is an important activity in the development life cycle of software. The goal of testing is to find faults in software that may be fixed and consequently to improve its quality. As long as the software size increases, more tests are necessary to eliminate and avoid defects, aiming at guaranteeing a major quality. However, the more tests are created, the more are executed, and more human resources and infrastructure are necessary. The time available to the test activity is not usually enough, so it is mandatory the reduction of the test execution time through automation. The test automation solves some problems, however, it may introduces other ones. In this context, this work proposes the development of a Software Testing Automation Process, establishing roles, artifacts and activities, all in the way to achieve the main objective that is to have a more agile test execution. This process aims at contributing with test teams that want to automate their tests, assuring the quality and without creating bottleneck to the organization.

Sumário

Índice de Figuras

Índice de Tabelas

Agradecimentos

Primeiramente, agradeço a Deus por me dar forças para superar as dificuldades, confiança para sempre ir em busca dos meus objetivos, capacidade de aprender e me adaptar às situações que a vida me apresentou e inteligência pra viver.

Agradeço aos meus pais, Maria de Lourdes e Manoel Oliveira e a toda minha família por estarem sempre me estimulando a crescer na vida e ir em busca do melhor para minha vida, dos meus objetivos e das minhas realizações, sempre através do estudo.

Agradeço a minha namorada, Gleyci Kelli, por ter sempre a compreensão nas horas de dificuldade, e por todo seu amor em todos os momentos da minha vida.

Agradeço todo o apoio da professora Cristine Gusmão, nos encontros que tivemos ou nos e-mails que trocamos, sempre me ajudando para que o meu trabalho fosse feito da melhor maneira possível, da forma correta e sempre me orientando a ir em busca dos meus objetivos.

Agradeço a atenção do meu co-orientador Professor Alexandre Marcos Lins de Vasconcelos, que mesmo não podendo ser formalmente meu co-orientador, aceitou contribuir, e opinar sobre o meu trabalho.

Agradeço aos meus amigos de turma, Thaysa Paiva, Pollyanna Barros e Murilo Pontes, que sempre deram força e sempre estiveram firmes, nos momentos de dificuldades, nas provas, nos trabalhos e projetos que tínhamos que desenvolver.

Agradeço ao meu chefe Elifrancis Soares, por ter me estimulado a desenvolver este trabalho, me ajudando nas indicações de referências, orientações e revisões.

Agradeço a Ellen Polliana e a Márcia Seabra, que também puderam ler e opinar sobre o meu trabalho, me ajudando com suas considerações.

Agradeço aos meus colegas de trabalho, Filipe Motta, Frederico Nascimento, Gleibson Oliveira, Márcia Seabra, Milena Mendes, Renata Souza, Roberto Damiani e Taíse Dias, por sempre me ajudarem a aprender o que era preciso para o bom andamento das atividades, pela compreensão e pela paciência, pela boa vontade e pelo espírito coletivo.

Capítulo 1

Introdução

O desenvolvimento de software é uma atividade complexa, pois lida com a habilidade do desenvolvedor, em pensar e criar soluções para a resolução dos problemas, que os softwares em construção pretendem solucionar. Por ser uma atividade complexa, os desenvolvedores estão sujeitos a cometerem erros, que podem causar *bugs*, também conhecidos como defeitos, no software. Para avaliar se este software está dentro do nível de aceitação, ou seja, se a quantidade de *bugs* é aceitável, temos o Teste de Software.

Teste é uma atividade na qual um sistema é testado e avaliado em uma determinada condição. Seu resultado é registrado de forma a poder ser avaliado [1]. Testar significa comparar o estado atual de um sistema com o estado que o mesmo necessita estar [2].

Desenvolver produtos de qualidade é o grande desafio da indústria de hoje. A complexidade dos sistemas cada vez mais aumenta fazendo com que o investimento em testes de software seja uma boa alternativa em busca da qualidade em um sistema.

Testar, porém, custa tempo e dinheiro [1]. O tempo é um inimigo diário nas grandes empresas, o desafio de desenvolver software com qualidade esbarra sempre no fator tempo. Objetivando fazer mais com menos, as empresas buscam testar seus sistemas, de forma a garantir a sua qualidade, mas utilizando um tempo mínimo. Uma solução para isto é automatizar seus testes [3].

Processo é um conjunto de atividades, ações, produtos de trabalhos e guias que se relacionam e são necessários ao desenvolvimento de uma atividade específica [4]. Para automatizar testes, faz-se necessário seguir um processo de forma a sistematizar os procedimentos necessários para a atividade de automação.

Um processo de automação de casos de teste proporciona a formalização de atividades, papéis e artefatos, de forma a garantir a qualidade e a praticidade dessas atividades.

Este capítulo introdutório tem como objetivo apresentar na Seção 1.1 as motivações para o desenvolvimento deste trabalho; na Seção 1.2, os objetivos que deverão ser alcançados; na Seção 1.3 a metodologia utilizada durante todo o desenvolvimento do presente trabalho; e na Seção 1.4 a estrutura do documento.

1.1 Motivação

Testar consiste em avaliar dinamicamente o comportamento de um software e comparar com o resultado esperado, utilizando para isto um conjunto de Casos de Teste [5]. A execução de testes é a forma mais antiga de se tentar garantir a qualidade de um sistema em desenvolvimento [6].

Porém, a execução dos testes é uma atividade que geralmente é feita geralmente, num curto espaço de tempo. Testar manualmente os casos de teste da organização pode ser uma atividade fácil e prática, caso o sistema a ser testado não seja grande e nem complexo, mas a realidade usual é que inúmeros casos de teste são necessários para testar a maioria dos sistemas existentes.

Com mais casos de teste para serem executados, mas com a mesma quantidade disponível de tempo para que a atividade de testes seja efetuada, uma possível e excelente solução para este problema é automatizar os casos de teste, de forma a ganhar tempo durante a execução. Porém, mais tempo para executar, não significa resultados idênticos da execução manual, devido aos possíveis problemas que podem ser causados no procedimento de automação.

Sabe-se também que, as atividades de teste se tornam repetitivas ao passar do tempo, fazendo com que o trabalho se torne desmotivante. Com a automação, busca-se evitar também que o testador esteja sempre executando, analisando e reportando resultados sempre dos mesmos testes, bem como, faz com que a execução de testes, como a do teste de regressão, possa ser feita de forma mais ágil.

Uma forma de evitar que problemas venham a ocorrer durante a automação, bem como evitar que gerem um elevado e insustentável custo, é evidente a necessidade de criação de um Processo de Automação de Testes de Software, que possa guiar a execução das atividades necessárias, garantindo que as mesmas sejam corretamente executadas.

1.2 Objetivos

Este trabalho tem a finalidade de contribuir com as atividades de teste de software de um time de testes, fazendo com que a execução de seus testes possam ser feitas de forma mais veloz e sem perdas qualitativas.

O objetivo principal deste trabalho é a criação de um Processo de Automação de Teste de Software que garanta que a automação seja feita com qualidade e organização, padronizando a forma base de se trabalhar a automação, através da criação deste processo, sem para isto, executar um processo complexo e com gargalos.

1.3 Metodologia

Para concretização do objetivo exposto na seção 1.2, ações foram definidas e agrupadas através de 3 grandes fases:

Fase 1 – Definição do Escopo da Pesquisa

Foi realizada a definição de escopo que este trabalho iria abranger que consiste em um Processo de Automação de Teste de Software, porém, limitando-se as etapas necessárias para que, já se tendo construído ou definido a ferramenta que será utilizada para automatizar os testes, os casos de testes possam ser automatizados com qualidade e eficiência.

Fase 2 – Revisão da Bibliografia e Fundamentação Teórica

Após definido o escopo deste trabalho, foi realizada uma análise na bibliografia referente aos testes de software e de automação de testes de software. A partir disto, foi feita uma análise e seleção das bibliografias encontradas e sugeridas.

Partiu-se então para a fundamentação teórica deste trabalho que consistiu da revisão bibliográfica que foi efetuada e que resultou numa síntese das áreas importantes para o bom entendimento do presente trabalho.

Fase 3 – Criação do Processo de Automação de Teste de Software

Definido o escopo e concluída a fundamentação teórica, o processo foi construído baseado nos conhecimentos adquiridos na área de automação de Teste de Software. Foram definidos os papéis, as atividades e os artefatos que se faziam necessários para que o Processo de Automação

de Testes de Software fosse construído com um bom nível de qualidade, assim como isto não tornasse o processo pesado para os times de teste.

1.4 Estrutura do Documento

Além deste capítulo introdutório, o documento está organizado da seguinte maneira:

- Capítulo 2 – **Testes de Software** – este capítulo apresenta a fundamentação teórica na área de testes de software necessária para criar uma base para o bom entendimento deste trabalho. Além de demonstrar a importância do teste de software e a influência do tempo na execução destas atividades. É apresentado também conceitos sobre processo e um processo geral de testes de software.
- Capítulo 3 – **Automação de Testes de Software** – este capítulo tem por finalidade apresentar uma base teórica em automação de testes de software. Deseja-se também mostrar o estado da arte na referida área.
- Capítulo 4 – **Processo de Automação de Testes de Software** – este capítulo apresenta o Processo de Automação criado e explica cada componente deste processo, como suas atividades e seus artefatos.
- Capítulo 5 – **Conclusão e Trabalhos Futuros** – este capítulo apresenta os resultados obtidos com a criação do Processo de Automação de Testes de Software, considerações sobre o processo proposto, dificuldades encontradas, trabalhos relacionados e possíveis trabalhos futuros que podem vir a dar continuidade a este trabalho.

Capítulo 2

Testes de Software

Quando um software é desenvolvido, erros podem ser introduzidos, podendo produzir defeitos. Caso esses defeitos sejam executados por um software, falhas podem vir a ocorrer, fazendo com que o sistema se comporte de forma diferente da esperada. Esses erros cometidos podem ser causados por má definição da solução, códigos complexos, infra-estrutura complexa, e principalmente por pressão dos prazos.

Testes de software são necessários de forma a executar o software em busca de erros. Deve-se tentar encontrar estes erros, o mais cedo possível, principalmente, antes do software ser entregue ao cliente, para que possa ser corrigido, e, para isto devem-se criar casos de teste com boa probabilidade de encontrar erros no software.

Mediante esta importância que o teste tem para o desenvolvimento de software, é apresentado neste capítulo, introdução na área de testes de software, bem como sua importância, níveis e técnicas de teste.

Para contextualizar este tema, a seção 2.1 irá mostrar os conceitos básicos na área de teste de software; a seção 2.2 apresentará as fases existentes na atividade de teste; a seção 2.3 tem a finalidade de mostrar os níveis de teste dentro do ciclo de desenvolvimento de software e a seção 2.4 irá expor as técnicas de teste existentes. Na seção 2.5, serão apresentados alguns conceitos sobre Processo e na seção 2.6, será dada uma visão geral sobre um Processo de Teste de Software, de forma a contextualizar, onde o Processo de Automação de Testes proposto e explicado no capítulo 4, está contido

2.1 Conceitos Básicos

Testar é um processo de planejar, preparar e medir objetivando estabelecer características de um software e de demonstrar a diferença entre o resultado esperado do resultado encontrado [1]. Testar reduz a incerteza sobre a qualidade de um software.

Algumas definições são importantes para o entendimento da área de testes de software e serão apresentadas a seguir [2]:

- **Erro**: Um erro é um engano, uma idéia incorreta, ou um equívoco no desenvolvimento de software.
- **Falta** (defeito, *bug*): Uma falta é introduzida num software a partir de um erro. É um problema que pode ocasionar um comportamento incorreto, diferente do esperado.
- **Falha**: Uma falha é a impossibilidade de um sistema ou componente de se comportar da forma esperada e especificada nos seus requisitos.
- **Caso de Teste**: É um item de teste composto por um conjunto de entradas, que são os dados e informações que serão passadas como entrada para o software; condições de execução, ou seja, em que condições iniciais o software deve estar, de forma a permitir a execução de um caso de teste; passos a serem executados; e resultados esperados após a execução daquele caso de teste.
- **Qualidade**: Está relacionado com o grau em que um sistema, componente, ou processo atinge os requisitos especificados, ou atinge as expectativas e necessidades do usuário.

O teste exige que o desenvolvedor descarte noções preconcebidas de “corretividade” do software recém desenvolvido e não se incomode com o trabalho duro de um engenheiro de testes para projetar casos de teste que “quebrem” o software.

Teste é um processo de execução de um software com a finalidade de encontrar um erro. Um bom caso de teste é aquele que tem alta probabilidade de encontrar um erro ainda não descoberto. Um teste bem sucedido é aquele que descobre um erro ainda não descoberto.

Alguns princípios foram sugeridos ao longo dos últimos 40 anos e oferece um guia geral para o processo de teste como um todo [7]:

PRINCÍPIO 1 – Teste demonstra a presença de defeitos

Teste pode demonstrar a presença de defeitos, mas não pode provar que eles não existem. O teste reduz a probabilidade que os defeitos permaneçam em um software, mas mesmo se nenhum defeito for encontrado, não prova que ele esteja perfeito.

PRINCÍPIO 2 – Teste exaustivo é impossível

Testar todas as combinações de entradas e pré-condições não é viável, pois o tempo que se levaria para terminar a execução deste conjunto de casos de teste pode tender ao infinito, exceto para casos triviais. Em vez do teste exaustivo, utilizamos os riscos e prioridades para dar foco aos esforços de teste.

PRINCÍPIO 3 – Teste antecipado

A atividade de teste deve começar o mais breve possível no ciclo de desenvolvimento do software ou sistema e deve ser focado em objetivos definidos. Quanto antes o teste for planejado e executado, antes os *bugs* serão encontrados e menos custosa será a correção destes *bugs*.

PRINCÍPIO 4 – Agrupamento de defeitos

Um número pequeno de módulos contém a maioria dos defeitos descobertos durante o teste, antes de sua entrega, ou exibe a maioria das falhas operacionais. Isto mostra que é necessário dar uma atenção especial ao que pode ser encontrado em volta de onde um problema foi encontrado, de forma a, se possível, focar mais os próximos testes ao redor de algum problema encontrado previamente.

PRINCÍPIO 5 – Paradoxo do Pesticida

Se um mesmo teste, eventualmente o mesmo conjunto de casos de teste, for repetido, várias e várias vezes, estes não vão encontrar nenhum defeito novo após um determinado momento. Para superar este “paradoxo do pesticida”, o caso de teste precisa ser freqüentemente revisado e atualizado; e um conjunto novo e diferente de testes precisa ser escrito para exercitar diferentes partes do software ou sistema com objetivo de aumentar a possibilidade de encontrar mais erros.

PRINCÍPIO 6 – Teste depende do contexto

Teste é feito de forma diferente conforme o contexto. Por exemplo, software de segurança crítica (software do computador de bordo de aviões) é testado de forma diferente de um software de comércio eletrônico.

PRINCÍPIO 7 – A ilusão da ausência de erros

Encontrar e consertar defeitos não ajuda se o sistema construído não pode ser usado e não atende as expectativas e necessidades dos usuários.

Os princípios citados devem ser conhecidos e analisados pelas organizações que trabalham com teste de software, pois esclarecem alguns pontos interessantes que são considerados marcos cruciais para o fracasso das atividades do teste, como por exemplo: acreditar que porque um software é testado, estará 100% livre de defeitos; não perceber que muito freqüentemente erros encontrados em determinadas partes do software escondem erros similares nas redondezas da área afetada; acreditar que um procedimento de teste aplicado a um determinado projeto cairá perfeitamente bem para outro projeto; acreditar que se durante um longo tempo, os casos de teste não encontram erros, significa que o software está perfeito, o que não é o correto, pois na maioria das vezes, aquele teste não é mais capaz de encontrar nenhum erro no software.

Como se pode perceber, estes são princípios interessantes e bastante importantes para toda e qualquer organização que deseja ter sucesso em suas atividades, bem como garantir uma boa qualidade em seus projetos e produtos.

2.2 Fases do Teste de Software

Apesar da execução de testes ser a parte mais visível, outras etapas muito importantes existem na atividade de teste, são elas: Planejamento e Controle do Teste, Análise e Projeto de Teste, Implementação e Execução de Teste, Avaliação do Critério de Saída e Reportando os Resultados e Atividades de Fechamento do Teste [7].

Esta seção visa explicar brevemente cada uma destas atividades, bem como a sua importância para todo o conjunto do teste de software.

Planejamento e Controle do Teste

Planejamento de Teste é a atividade de identificar os objetivos e os produtos liberados em toda a atividade de teste, identificar uma boa estratégia para utilização de recursos, definir escopo da atividade de teste, descrever a abordagem utilizada e definir como será feito o controle. A atividade de Controle do Teste é uma atividade constante que é feita de forma a acompanhar o atual progresso, e comparar com o planejado, envolve também a tomada de decisões de forma que o objetivo seja atingido.

Análise e Projeto de Teste

É a atividade em que, os objetivos de teste são transformados em casos de teste tangíveis. Esta fase tem como objetivo, revisar e avaliar a testabilidade dos requisitos que servirão como base para a criação de casos de teste, identificação e priorização de parte do sistema que poderão ser verificados por casos de teste, e, principalmente, pela construção e priorização dos casos de teste.

Implementação e Execução de Teste

É a atividade em que os procedimentos de teste são especificados através da combinação de casos de teste, que são colocados numa ordem executável, o ambiente inicial é configurado, e então, são executados. Após isto, é feito o registro das saídas encontradas, que serão comparadas com os resultados esperados e então, as diferenças encontradas são reportadas.

Avaliação do Critério de Saída e Reportando Resultados

Esta fase visa avaliar, mediante os resultados parciais do teste, se o objetivo do teste já foi atingido e se, portanto, o teste pode ser finalizado. Finalizado, um relatório que reporta os resultados para todos os envolvidos é elaborado.

Atividades de Fechamento do Teste

Esta última fase tem como finalidade, coletar dados do teste finalizado, checar se as entregas de artefatos planejadas foram realmente feitas, armazenando os artefatos, ambiente e infra-estrutura utilizada no teste, para posterior reuso e analise de lições aprendidas para projetos futuros.

2.3 Níveis de Teste

Um nível de Teste é um grupo de atividades organizadas e gerenciadas conjuntamente e está ligado às responsabilidades do projeto [8].

Nesta seção, será apresentado o *V-model*, um modelo de ciclo de testes que permite que as atividades de teste comecem antes no ciclo de desenvolvimento e não fiquem para o final, como anteriormente foi proposto no modelo *Waterfall* [9].

A partir da breve explicação sobre o *V-model*, serão também explicados os níveis de teste existentes, seus objetivos e metas associadas.

2.3.1 V-model

O *V-model* foi desenvolvido de forma a corrigir os problemas encontrados no modelo *Waterfall*, que é um modelo de desenvolvimento em que uma atividade só pode iniciar quando a anterior tiver acabado, e que algumas atividades importantes são deixadas para o final. O principal problema do modelo *Waterfall* resolvido no *V-model* é que no primeiro, o teste é apenas trabalhado após o software ser desenvolvido, enquanto que no segundo, ele acompanha e está integrado a cada etapa do ciclo de desenvolvimento, como pode ser visto na Figura 2.1.

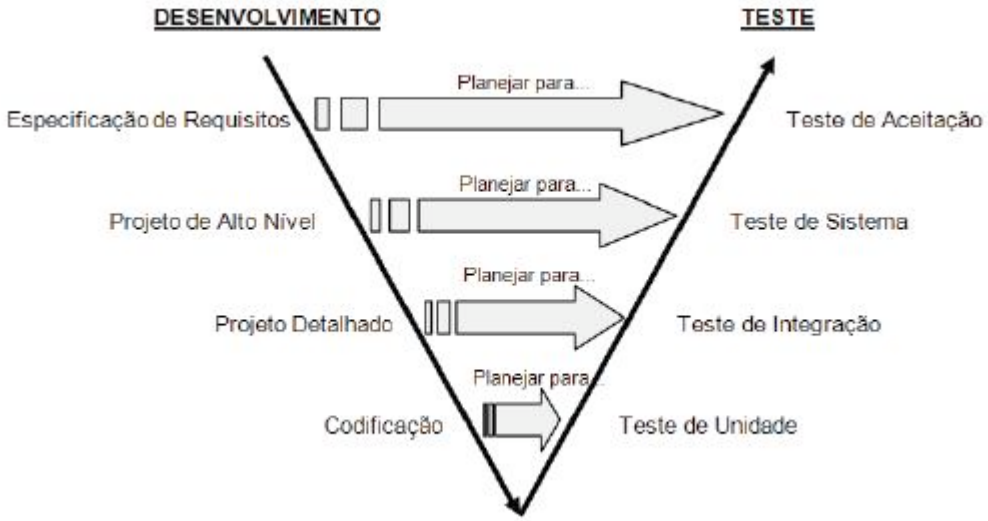

Figura 2.1 *V-Model* [10]

Esta é uma forma interessante de se testar, pois assim, não se deixa a atividade de teste apenas para o final do ciclo de desenvolvimento, ao contrário, os testes vão sendo efetuados ao

longo do desenvolvimento. O *V-Model* mostrado na Figura 2.1 faz com que para cada atividade do desenvolvimento existe um nível de teste correspondente.

Na próxima seção será explicado cada nível de teste presente no *V-Model*: Teste de Unidade, Teste de Integração, Teste de Sistema e Teste de Aceitação.

2.3.2 Níveis de Teste do V-Model

De acordo com a Figura 2.1, o V-Model é composto por quatro níveis de teste, que são planejados baseando-se numa respectiva atividade do desenvolvimento do software [10], e serão descritos a seguir:

Teste de Unidade

Podemos considerar unidades, também conhecidas como componentes, como sendo classes de um software. À medida que o a codificação vai sendo feita, os testes destes componentes vão sendo projetados e executados, de forma a validar os componentes desenvolvidos.

Teste de Integração

A partir da criação dos componentes, o sistema vai se formando com a integração destes componentes, na medida em que novas funcionalidades são necessárias. A interface entre os componentes integrados é uma fonte de problemas em potencial, por isto é importante testar estas interfaces.

Teste de Sistema

Baseado nos requisitos do sistema é possível e necessário testar para verificar se o sistema se comporta de acordo com o que se desejava ao planejar o software. Esta é uma etapa importante do teste, pois é quando realmente se verifica se as funcionalidades do sistema estão como esperado e se os requisitos não-funcionais estão sendo atendidos.

Teste de Aceitação

O teste de aceite é feito geralmente pelo cliente, de forma a verificar se o sistema já está pronto para ser colocado em produção, de acordo com o que havia sido planejado. O objetivo do teste de aceite é estabelecer confiança no sistema e não encontrar defeitos no software.

2.4 Técnicas de Teste

Para testar um software é interessante utilizar alguma abordagem, principalmente, por não ser viável testar 100% do software. Técnicas de teste são utilizadas para que se tenha uma estratégia de criação de testes para um determinado sistema, baseando-se num método proposto. Existem as técnicas caixa-preta, que testam as funcionalidades sem que o time de testes sequer conheça o código. Já as técnicas de teste caixa-branca são estreitamente ligadas ao código-fonte do sistema.

Várias são as técnicas de teste existentes [11], porém como o foco deste trabalho não é influenciado pelas técnicas de teste, apenas uma breve explicação de técnicas caixa-preta e caixa-branca serão dadas nas próximas subseções, bem como serão citadas as principais técnicas de teste.

2.4.1 Técnicas Caixa-Preta

Teste caixa-preta refere-se a testes que são conduzidos baseados nos requisitos do software. Um teste caixa-preta examina algum aspecto fundamental do sistema, pouco se preocupando com a estrutura lógica interna do software.

Não interessa para o testador, como foi feita a implementação, qual linguagem utilizada, os caminhos internos nem a estrutura, como pode ser visto na Figura 2.2.

Figura 2.2 Teste Caixa-Preta

No teste caixa preta o software é realmente uma caixa-preta em que o testador utiliza uma determinada entrada para uma funcionalidade e ele sabe que uma determinada saída será encontrada, não importa como esta saída foi concebida, mas sim que está conforme esperado.

As técnicas existentes de teste caixa-preta são [11]:

- **classe de equivalência**, divide os possíveis valores de entrada em classes, de forma que, se um dos valores de cada classe for testado, tem-se a garantia de que os outros valores daquela classe irão gerar os mesmos resultados;

- **valor limite**, testa as fronteiras, os limites, que são os pontos de potenciais problemas em um software;
- **tabela de decisão**, é utilizada geralmente quando existem regras de negócio complexas, em que várias combinações diferentes podem gerar resultados diferentes;
- ***pairwise***, mediante a especificação de vários itens que precisam ser testados de forma combinada, o *pairwise* faz a junção de pares combinando todas as entradas informadas, e então o Engenheiro de Testes deve remover as combinações que não fazem sentido para a sua regra de negócio e só testar as demais;
- **transição de estados**, esta técnica documenta que a partir de um determinado estado atual e acontecendo um certo evento, o sistema responde de uma forma específica, indo para um determinado estado;
- **análise de domínio**, identifica testes eficientes quando se necessita testar múltiplas variáveis que devam ser testadas juntas; e casos de uso, que testa baseado nos requisitos do sistema e na visão dos usuários.

2.4.2 Técnicas Caixa-Branca

Teste caixa-branca de software é baseado em um exame rigoroso do detalhe procedimental. Caminhos lógicos internos ao software e colaborações entre componentes são testados, definindo-se casos de testes que exercitam conjuntos específicos de condições e/ou ciclos.

O testador deve conhecer a estrutura interna do software, a forma como foi implementada e seus fluxos, como pode ser visto na Figura 2.3.

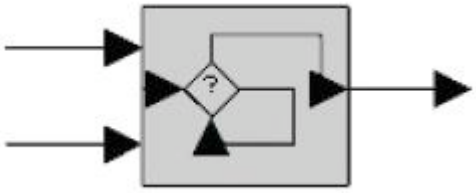

Figura 2.3 Teste Caixa-Branca

Introduzindo uma determinada entrada, o testador sabe exatamente qual o fluxo que esta irá percorrer por dentro do software, quais partes serão exercitadas e qual o resultado esperado. A desvantagem é que o testador vai sempre querer exercitar todos os caminhos e fluxos, porém isto

pode ser um grande problema, pois não haverá tempo hábil para esta atividade, testar exaustivamente é praticamente impossível.

Algumas das técnicas existentes de teste caixa-branca são: fluxo de controle, que identifica os caminhos existentes em um módulo de um código e então cria e executa testes para eles; e fluxo de dados, que testa os valores produzidos pelas computações internas do software, pois não se poderia sentir-se confiante num sistema sem ter executado cada instrução do mesmo.

2.5 Conceitos sobre Processo

Um Processo de Software pode ser definido como um *framework* para as atividades e abordagens necessárias para construir um software com qualidade [12]. Um Processo de Software é um conjunto de atividades e resultados associados que resultarão em um produto [13].

Para que as atividades de uma organização possam ser feitas de forma a atingir os objetivos almejados e que estes objetivos sejam alcançados com qualidade, é preciso que esta organização crie um processo que se adéqüe as necessidades do produto que vai ser gerado.

Nas empresas de desenvolvimento de software, algumas atividades são consideradas básicas e essenciais em seus processos de desenvolvimento [13]. Como este trabalho não foca num processo de desenvolvimento, iremos apenas citar estas atividades, que são: a Especificação de Software, o Desenvolvimento de Software, a Validação do Software e Evolução de Software.

Para organizações que trabalham com teste de software, é necessário um processo que controle as atividades de teste, de forma a organizar e indicar que atividades devem ser realizadas de forma a atingir os objetivos do time, que, para a maioria dos casos, é encontrar *bugs* no software testado.

As atividades constantes em um processo devem ser seguidas, de forma a gerar os artefatos planejados, garantindo a execução do processo com qualidade, em busca do objetivo principal da organização.

2.6 Processo de Testes de Software

Para o melhor entendimento do Processo de Automação de Testes proposto por este trabalho, faz-se necessário um entendimento de um Processo de Testes de Software, de forma a compreender melhor, onde se encaixará o Processo de Automação de Testes proposto, dentro de um processo geral de testes de uma organização.

Podemos perceber as atividades de planejamento de testes, projeto de testes, automação de testes, execução de testes e monitoramento de testes, em um processo de testes básico que é apresentado na Figura 2.4.

Como atividade essencial e inicial, o planejamento de testes consiste em se planejar o escopo do teste, o time que irá trabalhar, as atividades necessárias durante o projeto, um cronograma, os recursos de *hardware* e software que serão utilizados.

Mediante um planejamento de testes feito, a atividade de projeto de testes vem logo em seguida, como sendo o momento em que os requisitos do sistema que será testado transformam-se em casos de teste, que podem já ser testados de forma manual.

A partir da escrita dos casos de testes manual, é possível automatizar os testes da organização, e isto é feito na atividade de automação de testes. É nesta atividade de automação, que o processo proposto neste trabalho, que será apresentado na próxima seção, se encaixa.

A atividade de execução de testes pode ser realizada de duas formas diferentes, manualmente, caso a atividade de automação ainda não seja escopo da organização, ou automaticamente, caso já se tenha os testes automáticos estáveis.

Por fim, a atividade de monitoramento de testes, que é uma atividade muito importante e que se desenvolve durante todo o ciclo de testes ininterruptamente. Consistem de procedimentos para se acompanhar todas as atividades do ciclo e providenciar que possíveis problemas encontrados durante o mesmo possam ser sanados rapidamente, de forma a minimizar todo e qualquer impedimento que possa atrapalhar todo o ciclo de testes.

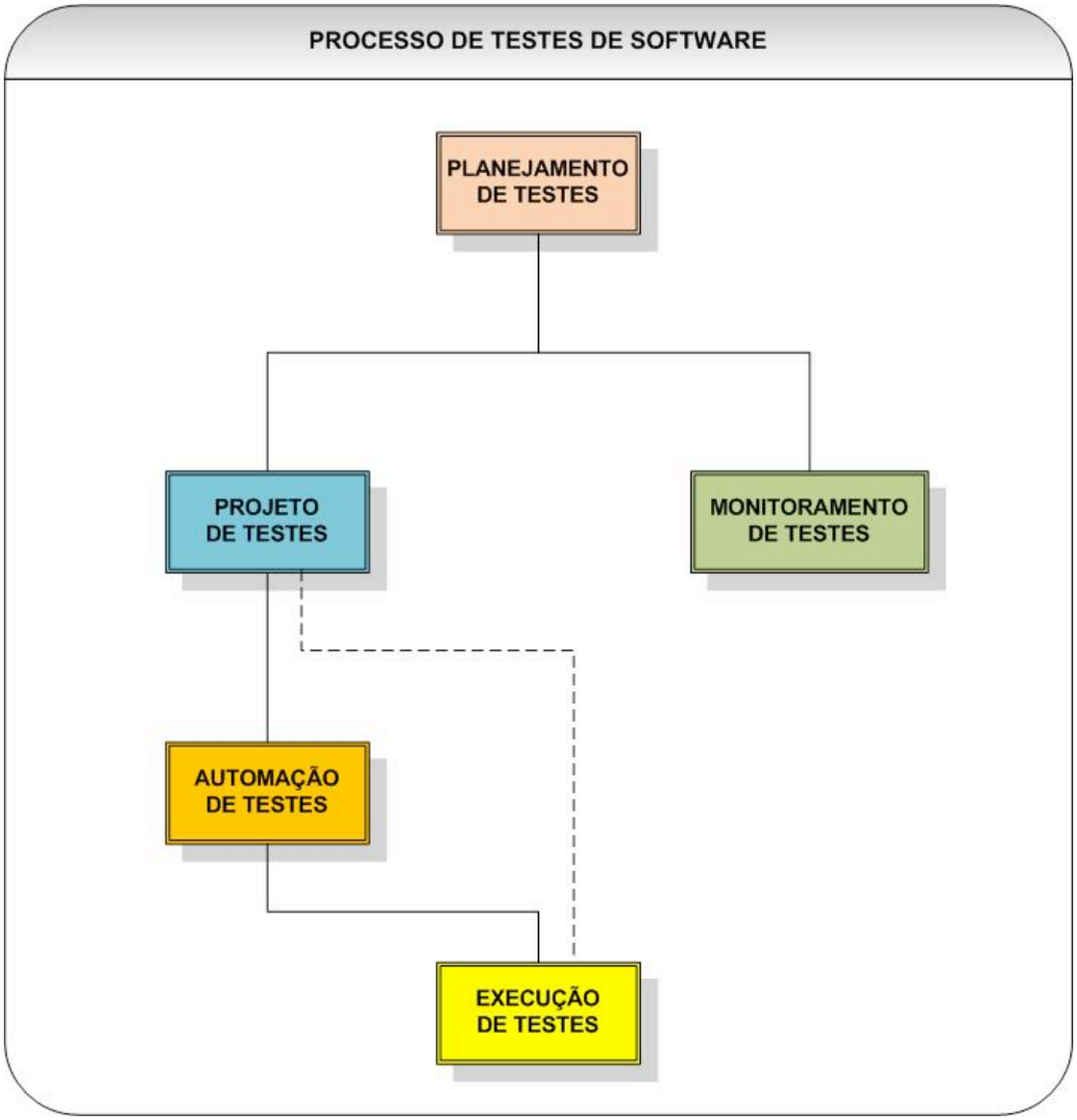

Figura 2.4 Processo de Testes de Software

Capítulo 3

Automação de Testes de Software

Automatizar testes é diferente de testar software. Muitas empresas percebem que automatizar casos de teste é muito mais caro do que executá-los manualmente por uma vez. Se um caso de teste vier a ser automatizado, isto não quer dizer que o mesmo será mais eficiente do que o mesmo executado manualmente. Automatizar apenas afeta quão econômico ele será, e como o mesmo será executado [14]. Vários são os benefícios que a automação pode trazer, bem como vários são os mitos existentes sobre automação, tudo isto será trabalhado neste capítulo.

De forma a fundamentar este tema, na seção 3.1 irá ser mostrado um pouco sobre a base inicial de automação de testes; a seção 3.2 tem a finalidade de mostrar quais são as promessas existentes para quem deseja automatizar testes e na seção 3.3 serão apresentados os problemas mais comumente encontrados na automação de testes.

3.1 Introdução à Automação de Testes

Hoje em dia, as empresas precisam desenvolver software o mais rápido possível, com o mínimo de recursos possível. Estudos realizados [15, 16] mostram que mais de 90% dos desenvolvedores já atrasaram as datas de lançamentos de seus produtos, atrasar *deadlines* é uma rotina de 67% dos desenvolvedores. Além disso, 91% deles foram obrigados a retirar funcionalidades do sistema que estava sendo desenvolvido, de forma a manter os prazos. Levar um produto ao mercado o mais cedo possível, leva ou ao sucesso da empresa ou ao seu fracasso.

Atrasos nos *deadlines* dos desenvolvedores geram menos tempo para que os testadores possam exercitar o software, fazendo com que mais *bugs* possam ser mantidos no software e levados ao ambiente de produção.

Com a necessidade de se mostrar o quanto antes, o andamento do desenvolvimento de um software, muitas *releases* são geradas, sendo isto uma grande causa de possíveis problemas que permanecerão no sistema, devido ao pouco tempo disponível para os testes.

Existe sempre a necessidade de estar testando uma grande quantidade de casos de teste, porém, deseja-se que isto seja feito gastando-se o mínimo de tempo, e para isto, as empresas têm partido para a automação de seus testes.

A automação de testes é bastante interessante para os níveis de teste de integração e de sistema [3]. O teste de integração pode ser necessário, na medida em que uma nova funcionalidade é agregada ao software. Então, é necessário testar além dos testes já existentes, novos testes criados para a esta nova funcionalidade agregada, o que faz com que a quantidade de testes aumente e a automação pode trazer benefícios na economia do tempo de execução destes testes.

No nível de teste de sistema, temos os testes de regressão, em que após uma mudança efetuada no software é interessante se fazer uma regressão, ou seja, testar um conjunto de casos de teste, além dos casos de teste relacionados à mudança, de forma a verificar se uma alteração em uma parte do software não gerou problemas em outras partes do sistema. Isto faz com que vários casos de teste precisem ser executados inúmeras vezes, mediante a quantidade de testes de regressão que forem necessários.

Com a automação, mais casos de teste poderão ser executados gastando um tempo menor, isto ajuda a organização a poder testar mais e o quanto antes. Como o foco da atividade de testes é encontrar *bugs* em um software de forma a elevar a sua qualidade, o quanto antes estes *bugs* forem encontrados, melhor será para a empresa e menos custoso será, como pode ser visto na Tabela 3.1.

Tabela 3.1. Custo de remoção de erro em relação ao nível de teste [17]

Nível	Custo
Teste de Unidade	$15
Teste de Integração	$22
Teste de Sistema	$50
Após a entrega	$100+

Bugs encontrados no nível de teste de unidade gera aproximadamente seis vezes menos custo do que um *bug* encontrado após a entrega ao cliente. Isto reforça os benefícios que o teste

pode trazer para as empresas bem como reforça a importância que a automação tem relacionada ao princípio de que testar o quanto antes, traz ganhos para a qualidade do software.

3.2 A Decisão de Automatizar Testes

A automação de testes de software tem seu lado positivo, que será mostrado na seção 3.3, e o lado negativo em que possíveis problemas podem vir a ocorrer e serão apresentados na seção 3.4.

Várias são os tipos de ferramentas que podem contribuir com a automação de testes de software dentro de uma organização. Elas podem estar presentes em várias atividades do ciclo de teste e conjuntamente podem ajudar positivamente a melhoria e otimização dos testes.

A partir da avaliação de ferramentas [3] para diferentes atividades de teste, foi possível verificar, através da Tabela 3.2, que a automação pode contribuir substancialmente para a melhoria dos testes, principalmente se tratando de economia de tempo. Na Tabela 3.2, é apresentada uma comparação de uma estimativa de tempo gasto (em horas), nas várias atividades dentro do ciclo de vida de teste de software.

Tabela 3.2. Testes Manuais *versus* Testes Automáticos [3]

Atividades de Teste	**Teste Manual (horas)**	**Teste Automático (horas)**	**Percentual de Melhoria com Ferramentas**
Desenvolvimento do Plano de Teste	32	40	-25%
Desenvolvimento dos Casos de Teste	262	117	55%
Execução de Testes	466	23	95%
Análise dos Resultados dos Testes	117	58	50%
Monitoramento do status e da correção de um erro	117	23	80%
Geração de Relatórios	96	16	83%
Duração Total	***1090***	***277***	***75%***

Analisando a Tabela 3.2, pode-se perceber que, com a automação dos testes, o tempo desprendido para a realização das atividades mencionadas nesta tabela se reduz a 25% do tempo

gasto se as mesmas atividades forem desenvolvidas manualmente. Isto demonstra o grande poderio que a automação pode trazer para a organização.

Porém, para alcançar esses marcos com qualidade, é necessário que a equipe seja bastante capacitada em testes de software, bem como em desenvolvimento, pois a atividade de automação nada mais é do que o desenvolvimento para a área de testes, o que justifica o conhecimento na área. Outro fator importantíssimo para o sucesso na automação é que a organização tenha um processo de testes bem definido e que este processo contemple, a partir da decisão de automatizar os testes, o padrão indicado para esta importante atividade de automatizar.

A decisão para automatizar testes de software deve ser bem pensada pela organização de forma a se preparar melhor para o momento da automação. É interessante que a organização perceba se realmente é necessário e se trará reais benefícios, pois o custo da automação não é pequeno, bem como os resultados não são perceptíveis num curto espaço de tempo, pelo contrário, os ganhos da automação de testes começam a ser percebidos apenas após alguns ciclos de execução dos testes automatizados [18]. Podem-se perceber retornos da automação em curto prazo através da automação de testes de sanidade, testes em que se verificam se as condições mínimas necessárias para que a atividade de testes por completo possa ser iniciada; automação de testes de configuração, que são testes que verificam se a configuração de hardware está correta; e automação de testes de stress, testes que utilizam grandes volumes em curto espaço de tempo [18].

Alguns questionamentos [18] poderiam ser feitos, para verificar se a automação de testes pode realmente trazer benefícios para a organização, fazendo com que haja, portanto, uma atenuação dos custos com este processo de automação. Estes questionamentos podem se adequar a algumas organizações, e para outras pode ficar sem sentido, porém, o conjunto de questões pode ser tratado como um questionário genérico para atender a maioria das empresas. Os questionamentos são:

- A interface da aplicação a ser testada está estável?
- Espera-se vender este produto em várias versões?
- Espera-se que este produto esteja estável na versão atual, ou haverá outras versões para a correção de erros?
- Esta organização desenvolve outros produtos de maneira similar? Existirão outras oportunidades de amortizar o custo gasto durante a automação?
- Será necessário rastrear os casos de teste até o nível de requisitos, de forma a mostrar que para determinados requisitos existem casos de teste associados?

- Esta empresa é sujeita a auditoria ou inspeções por organizações que defendem fortemente, longos testes de regressão?
- Quais os tipos de teste que são mais pesados no ciclo de testes desta organização? De que forma a automação pode torná-los menos pesados?

Vários questionamentos de âmbito gerais que poderão facilmente ser aplicados, em sua maioria, em qualquer organização que trabalha com testes de software. Este questionário ajuda a empresa a refletir, planejar e quantificar as vantagens e desvantagens que a automação poderá trazer para suas atividades de teste.

A automação pode falhar, e isto pode acontecer pelo fato de as empresas não terem o mesmo cuidado e o mesmo profissionalismo com a automação como fazem para com os demais projetos de desenvolvimento [19, 20]. Por este fato, é preciso pensar bem se o custo desta automação valerá a pena, pois custa de 3 a 10 vezes mais criar, verificar e documentar testes automáticos, do que manuais [21].

3.3 Benefícios da Automação de Testes

Vários são os benefícios que podem ser alcançados através da automação de teste, dentre eles podemos citar que a eficiência da execução aumenta devido ao fato dos testes serem executados mais rapidamente, ou seja, num mesmo período de tempo T, a quantidade de casos de teste manuais que podem ser executados é menor do que a quantidade de casos de teste automáticos. Outros benefícios [14] podem ser alcançados como:

- Executar um conjunto de casos de teste de regressão sem muito esforço, selecionando os testes que se deseja executar e em pouco tempo, é possível preparar e configurar o ambiente necessário para esta execução.
- Além de executar os testes mais rapidamente, é possível aumentar a freqüência de execução destes casos de teste. À medida que mais casos de teste são executados mais freqüentemente, mais confiança pode-se ter da qualidade do software em desenvolvimento, desde que para isto, os casos de testes que serão executados, já garantam uma boa cobertura do software, ou seja, que garantam que várias funcionalidades do software sejam executadas.
- É possível executar testes que não são possíveis de serem executados manualmente, como os testes de *stress*, através da utilização dos testes automáticos, bem como é possível que

pessoas que não tenham um conhecimento da regra do negócio, possam executar testes automáticos, diferentemente dos testes manuais. Isto traz benefícios para a organização que pode destinar recursos humanos treinados e capacitados na regra de negócio, para outras atividades do ciclo de testes, ao invés de ocupá-los com a execução propriamente dita.

- A melhor utilização de recursos também pode ser alcançada através da automação. Quando falamos de recursos, podemos tratar de dois diferentes tipos: o recurso de *hardware*, e o recurso humano. Sobre o recurso de *hardware*, teremos um recurso disponível por um tempo executando mais testes automaticamente do que teríamos executando manualmente. Sobre os recursos humanos, temos o fator motivacional. O testador executando sempre os mesmos testes durante alguns ciclos pode fazer com que, no futuro, falhas possam ser passadas despercebidamente, por conta da carga de trabalho repetitivo que fica para o testador. Não é motivante para o testador, executar várias vezes o mesmo teste, sabendo que, geralmente, ele deverá ter o mesmo resultado. Além disto, a automação pode ajudar a organização por diminuir a quantidade de pessoas que precisam estar alocadas para executar os casos de teste, podendo utilizar estes recursos em outras atividades do time.
- Automação pode trazer consistência e repetitividade na medida em que os testes podem ser executados exatamente da mesma forma e poderão, portanto, ser reproduzidos em diferentes ambientes quando for necessário, tendo a certeza que o que foi executado em uma plataforma, é o mesmo que estará sendo executado em outra.
- O reuso dos casos de teste automáticos é mais perceptível do que o dos manuais, pois, possivelmente, os testes automáticos serão executados mais vezes do que os testes manuais.
- A automação faz com que os produtos possam ir para o mercado antes do que se tiver que executar um conjunto grande de casos de teste de forma manual, desde que os casos de teste já estejam prontos.
- A confiabilidade de um software aumenta, à medida que uma grande quantidade de testes é executada e seus resultados são, exatamente, os esperados.

Como pudemos ver, os benefícios que podem ser alcançados com a automação dos casos de teste de uma empresa são vários, fazendo com que a qualidade e confiabilidade do software cresçam, bem como fazendo com que menos esforço seja desprendido nesta atividade. Porém,

faz-se necessário um procedimento padronizado para automatizar testes, que vise à qualidade. Este é o grande objetivo deste trabalho, o desenvolvimento de um procedimento padrão para a automação de casos de teste de uma empresa.

3.4 Problemas Comuns da Automação de Testes

Além de vários benefícios que a automação de testes pode trazer para as organizações, alguns problemas podem ser encontrados. Desta forma, é sempre bom estar preparado e, portanto, conhecer os possíveis problemas que podem vir a ocorrer no âmbito da automação de testes. Alguns problemas podem trazer grandes impactos para a organização, dentre os quais podemos citar [14]:

- As expectativas irreais da gerência com relação à utilização de uma determinada ferramenta de automação, ou da própria automação, podem causar grandes prejuízos à organização. É comum que as pessoas pensem que uma determinada ferramenta, vai ser a salvação da empresa para solucionar um determinado problema. É preciso saber que as ferramentas de automação, bem como, a automação em si, têm seus problemas e, possivelmente, não atingirá completamente, o que se espera dela.
- Caso a organização tenha um processo de testes mal planejado, com documentação fraca e com casos de teste que não encontram defeitos no software, não adianta automatizar os testes. A automação de algo que foi mal feito, só irá exibir a fraqueza da organização mais rapidamente.
- A expectativa de que a automação de testes irá descobrir mais *bugs* no software é um grande engano. Os *bugs* geralmente são encontrados na primeira execução de um caso de teste. Partindo do princípio de que o indicado é que o teste seja criado manualmente, para depois ser automatizado, assim que criado de forma manual, o mesmo deve ser executado, para garantir que o mesmo funciona e não demandar tempo na etapa de automação para corrigir o teste manual. Isto fará com que, possivelmente, se existir um *bug* que pode ser encontrado por este caso de teste, este *bug* será encontrado na execução de validação manual do caso de teste.
- Existe a ilusão de que, se um conjunto de casos de teste que foram executados, não encontrar *bugs*, significa dizer que o software está livre de defeitos, o que não é verdade, pois possivelmente, os erros podem não ter sido encontrados, pois os testes não foram feitos de forma a encontrá-los, ou os testes ou seus resultados estão incorretos. A

automação de forma alguma irá resolver este problema, pois a mesma apenas preserva o comportamento descrito nos casos de teste manuais.

- Um dos principais motivos que fazem com que a automação de testes seja um fracasso é quando a manutenção destes testes eleva demasiadamente o custo do software, e às vezes, é mais barato executar os testes manualmente do que manter os testes automáticos gerados. Isto é, na maioria das vezes, percebido apenas no momento em que o software muda, e se faz necessário alterações nos testes existentes. Por isto, é interessante automatizar apenas os testes que estão mais estáveis.
- Quando se utiliza uma ferramenta comercial de execução, que geralmente são bastante complexas, podemos encontrar defeitos que irão gerar problemas, como problemas de ambiente, em que as ferramentas não se adaptam tão rapidamente a dinâmica constante nos ambientes.
- Se a equipe e a gerencia não estiver comprometida e com o foco nos objetivos planejados para serem atingidos com a automação, isto vai ser um grande problema. A automação sofre grande perda quando não existe um procedimento de utilização da ferramenta, ou quando o esforço desprendido para a automação se resume a apenas um dos projetos de uma organização.

São diversos os problemas que podem ocasionar o fracasso da automação de testes, vários fatores que podem elevar os custos e gerar desperdício de tempo do projeto. É preciso que a organização esteja preparada para automatizar seus testes, com um procedimento padronizado para esta automação, e com os casos de teste bem definidos, bem como é preciso que a equipe esteja motivada para a atividade e as ferramentas sejam escolhidas com muita atenção.

A criação de um procedimento para automação, que neste trabalho é chamado de Processo de Automação de Testes, é o objetivo principal deste trabalho e será descrito no próximo capítulo.

Capítulo 4

Processo de Automação de Testes de Software

Vimos que a automação de testes pode fazer com que um time de testes passe a ter ganhos interessantes no que diz respeito a tempo e eficiência. Porém, para atingir os benefícios, se faz necessário que o procedimento de automação de testes seja padronizado, visando sempre, a garantia da qualidade.

Este capítulo irá explicar o Processo de Automação de Testes[1] proposto por este trabalho, que está contido num processo de testes completo, logo após a atividade de projeto de testes, conforme explicado na seção 2.6.

Na seção 4.1 serão explicados os papéis que desempenham atividades, bem como, na seção 4.2, os artefatos que compõem o Processo. Na seção 4.3, um conceito, uma ferramenta e dois *templates* serão apresentados de forma a facilitar o entendimento quando os mesmos forem citados em seguida, durante a explicação do fluxo principal deste processo, que está contido na seção 4.4, onde são explicadas as atividades a serem executadas no processo.

4.1 Papéis do Processo

Alguns papéis são necessários para o desenvolvimento das atividades de um processo. Neste processo, quatro são os papéis que estão envolvidos, o Gerente de Testes, o Engenheiro de Testes e o Engenheiro de Automação. O Desenvolvedor também é mencionado, mesmo não fazendo parte do Processo de Testes de um modo geral.

[1] Processo disponível na *web*: dsc.upe.br/~pco/automationprocess

Cada papel tem algum artefato sob sua responsabilidade, e executa alguma atividade do Processo de Testes. Será apresentada uma descrição sobre o papel, e os artefatos e atividades relacionadas ao mesmo. Nas próximas seções, será falado mais especificamente sobre os artefatos e as atividades.

Gerente de Testes

O Gerente de Testes é responsável para planejar o processo de automação de testes, selecionando a funcionalidade que será automatizada, que pode ser baseado no *Release Notes*, bem como no Plano de Testes da organização.

O Gerente de Testes também deve supervisionar o Engenheiro de Testes na atividade de validação da automação para confirmar se o que foi planejado foi feito como esperado. Isto é extremamente importante, principalmente em grandes organizações, em que as atividades do Gerente de Testes podem sobrecarregá-lo, nestas ocasiões, é interessante que o Engenheiro de Testes seja o executor da validação da automação.

O Gerente de Testes também controla se os *scripts* de teste estão funcionando corretamente após o processo de automação.

Na Figura 4.1, podemos ver que o Gerente de Testes executa as seguintes atividades: Controlar a Automação, Avaliar Automação e Planejar Automação, bem como, é responsável pelo Plano de Automação e pelo Plano de Testes.

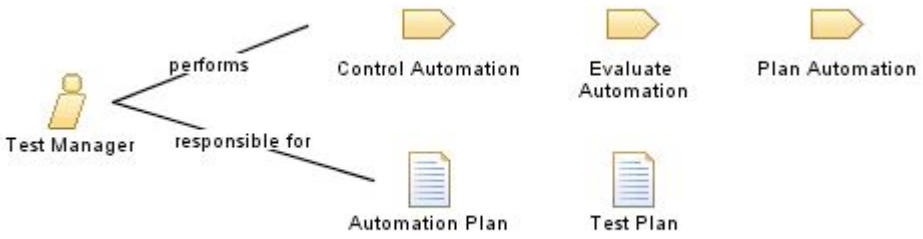

Figura 4.1 Gerente de Testes

Engenheiro de Testes

O Engenheiro de Testes é a pessoa responsável para avaliar quais casos de teste podem ser automatizados, e quais técnicas de seleção de casos de teste podem ser utilizadas. Como o

escopo deste trabalho foca na automação, apenas o conceito de técnicas de seleção será mostrado, sem um detalhamento maior do assunto.

O Engenheiro de Testes com a supervisão do Gerente de Testes pode também validar se a automação planejada foi feita como esperado.

A Figura 4.2 mostra que o Engenheiro de Testes executa a atividade Selecionar Casos de Teste, e é responsável pelos artefatos: Relatório de *Status* da Automação, Casos de Teste e Suíte de Testes.

Figura 4.2 Engenheiro de Testes

Engenheiro de Automação

O Engenheiro de Automação é a pessoa responsável por implementar os *Scripts* de Teste, baseados nos Casos de Teste selecionados na atividade Selecionar Casos de Teste.

Por ter um perfil maior de desenvolvedor do que da área de testes, porém, deve ter os conhecimentos, no mínimo básicos, foi necessário criar o papel do Engenheiro de Automação, que é um profissional com perfil de desenvolvimento com conhecimentos em testes.

A Figura 4.3 apresenta a responsabilidade do Engenheiro de Automação de Testes pelo *Script* de Teste e mostra que ele executa a atividade de Automação de Casos de Teste.

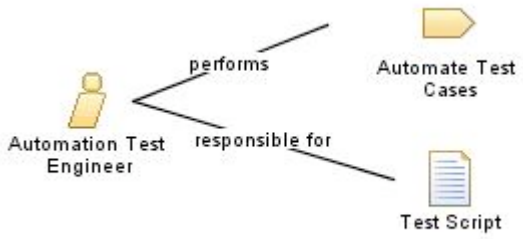

Figura 4.3 Engenheiro de Automação de Testes

Desenvolvedor

O Desenvolvedor não é um papel do Processo de Automação de Testes, mas é a pessoa responsável por codificar o software que está sendo testado e por fazer o artefato *Release Notes* que é uma entrada importante da atividade Planejar Automação.

A Figura 4.4 mostra que o Desenvolvedor é responsável pelo artefato *Release Notes*.

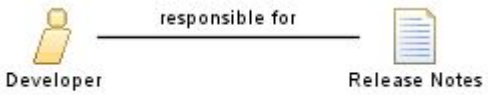

Figura 4.4 Desenvolvedor

4.2 Artefatos do Processo

Os artefatos contidos no Processo proposto, Plano de Testes, *Release Notes*, Plano de Automação, Caso de Teste, *Script* de Teste, Suíte de Testes e Relatório de *Status* da Automação, serão explicados nesta seção. No final, serão apresentados os relacionamentos dentro do processo, dos artefatos mencionados.

Plano de Testes

O Plano de Testes contém todos os objetivos dos testes, o cronograma, o escopo, a abordagem e técnicas de teste usadas pelo time de testes.

Este artefato não é desenvolvido durante o Processo de Automação de Testes, mas sim, na atividade de Planejamento de Testes que precede a automação dos testes criados durante o Projeto de Testes. Porém, o mesmo tem uma importância grande, pois é o Plano de Testes de guia todo o ciclo de testes que contém o Processo de Automação proposto.

Por conter estas informações gerais do Processo de Testes, se torna um artefato de entrada obrigatório para a atividade Planejar Automação, por conter toda e qualquer informação necessária para o desenvolvimento de todas as atividades do ciclo de testes.

Release Notes

O *Release Notes* é o artefato no qual os desenvolvedores descrevem as funcionalidades a serem testadas, a prioridade e as mudanças no software.

Este artefato é importante como entrada opcional da atividade de Planejar Automação, pois mediante uma prioridade definida pelo Desenvolvedor, o Gerente de Testes pode solicitar que sejam automatizados os casos de teste de uma determinada funcionalidade, que naquele momento é a prioridade para o próximo ciclo de execução.

Plano de Automação

O Plano de Automação é o artefato no qual são descritos:

- as funcionalidades que estarão sendo automatizadas,
- o cronograma e
- a equipe envolvida neste ciclo de automação.

O ciclo do Processo de Automação é repetido várias vezes, tantas quantas forem necessárias. Diferentemente do Plano de Testes, que é um documento que tem vida durante todo o projeto, o Plano de Automação é feito cada vez que nova(s) funcionalidade(s) necessita de automação, para isto, o Gerente de Testes deve preencher um novo *template* com as informações pertinentes aquele novo ciclo de automação. Durante toda a vida do projeto, vários serão os Planos de Automação, visto que ele não é estático, e sua vida termina, no final da atividade Controlar Autenticação.

Caso de Teste

O Caso de Teste é a especificação das pré-condições, procedimentos do teste e os resultados esperados, que serão executados pelo time de teste. É uma saída da atividade Projeto de Testes, do Processo de Testes, não sendo, portanto, um resultado do Processo de Automação de Testes, mas serve como subsídio essencial e obrigatório para a automação. A partir dos Casos de Teste criados pelo time de testes é que a organização poderá automatizar estes testes.

***Script* de Teste**

O *Script* de Teste é o artefato gerado pela automação dos casos de teste, através de uma Ferramenta de Automação de Testes. Este artefato é o objeto resultante da automação de um caso de teste. É necessário que a organização tenha um *template* para os *Scripts* de

Teste, de forma a padronizar a sua criação, garantindo que seja independentemente compreensível, com comentários e que seja fácil de ler e entender. Para cada caso de teste, um *Script* de Teste é criado, tendo, portanto, a relação 1:1. Estes *Scripts* de Teste são gerados a partir de uma Ferramenta de Automação que pode ser uma ferramenta de mercado, podendo ser aberta ou proprietária, ou uma ferramenta criada de acordo com as necessidades do time de testes.

Suíte de Teste

A Suíte de Testes é um repositório de algumas informações importantes, sobre os casos de teste da organização. Contém um resumo dos casos de teste, mencionando o *ID*, o objetivo e a forma de execução do caso teste, que pode ser manual ou automático.

A Suíte de Teste vai sendo atualizada à medida que novos Casos de Teste vão sendo criados. É uma forma de facilitar o trabalho de visualização superficial dos Casos de Teste que a organização possui. Quando um Caso de Teste se torna um *Script* de Teste, após o Processo de Automação, a Suíte de Teste é atualizada, informando que o mesmo não é mais manual, mas sim automático, ajudando o Gerente de Testes no momento em que ele prepara os testes para a execução.

Relatório de *Status* da Automação

O Relatório de *Status* da Automação contém todas as informações importantes de cada atividade do Processo de Automação de Testes, um *log* do que foi feito em cada atividade e os problemas encontrados em cada fase. Este artefato é um *log* fase a fase do que aconteceu e do que foi feito durante o Processo de Automação de Testes.

Podemos considerar o Relatório de *Status* da Automação, como sendo um resumo de tudo que aconteceu de importante, durante o Processo de Automação como um todo, é onde estará descrito, caso problemas venham a acontecer, é onde estarão as informações necessárias para as atividades seguintes, é onde o Gerente de Testes ou qualquer membro do time poderá ter uma visão geral da automação.

O artefato vive durante todo o Processo de Automação, mas tem seu fim, após a atividade Controlar Automação terminar. Para cada ciclo de automação, um novo *template* com as informações daquele novo ciclo de testes, é criado.

Os artefatos criados durante a execução do Processo de Automação, o Plano de Automação e o Relatório de *Status* da Automação são extremamente importantes. No fim do Processo de Automação, eles devem ser armazenados de forma a servirem como base para a realização de novos ciclos do Processo de Automação. Isto pode ser feito, levando-se em consideração os ganhos e perdas obtidas em forma de lições aprendidas para que os problemas não ocorram novamente e que os benefícios de repitam. Isto servirá como itens para melhorias do Processo de Automação da organização, pois é um *feedback* de execuções anteriores do processo.

Os artefatos que fazem parte do Processo de Automação proposto se relacionam entre si, na medida em que um artefato pode ser essencial para que outro possa existir, e quando um artefato pode servir de apoio para a criação de outros.

Na Figura 4.5, o relacionamento entre os artefatos é apresentado. O Plano de Automação necessita do Plano de Testes e do *Release Notes*, lembrando que o mesmo é opcional. Para gerar os *Scripts* de Teste necessitam que os Casos de Teste estejam criados, e na medida em que são gerados os *Scripts* de Teste, a Suíte de Teste é atualizada informando que o mesmo é um teste automático a partir de então. Com os trabalhos de criação do *Script* de Teste, e após a sua criação, o Relatório de *Status* da Automação vai sendo gerado.

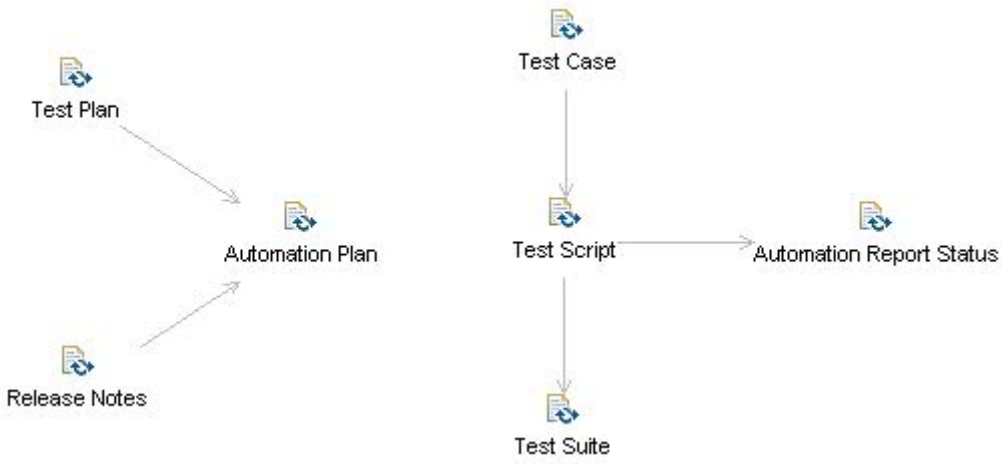

Figura 4.5 Artefatos Envolvidos no Processo de Automação

4.3 Conceito, Ferramenta e Templates do Processo

Fez-se necessário, para um melhor entendimento, a criação de um conceito sobre Técnicas de Seleção de Casos de Teste, de uma explicação simples sobre Ferramenta de Automação e de dois *templates*, um para o Plano de Automação e um para o Relatório de *Status* da Automação.

Conceito: Técnicas de Seleção de Casos de Teste

Técnicas de Seleção de Casos de Teste é uma forma de reduzir a quantidade de Casos de Teste para um objetivo científico. Podem ser selecionados Casos de Teste para fazer um teste de regressão ou para automatizar casos de teste.

Selecionando alguns dos Casos de Teste existentes o custo da automação pode ser reduzido. Existem várias Técnicas de Seleção de Casos de Teste, e pode ser selecionado de acordo com o objetivo do time de testes.

Ferramenta: Ferramenta de Automação

Esta é a ferramenta utilizada pela organização. Pode ser uma ferramenta comercial ou pode ser desenvolvida pelo time. Caso seja uma ferramenta de mercado, antes de comprá-la, caso seja uma ferramenta proprietária, ou adquiri-la gratuitamente, em se tratando de uma ferramenta aberta, é necessário que a organização possa fazer uma avaliação da melhor ferramenta a ser escolhida, de forma que se compre uma ferramenta que atenda as necessidades da organização.

Caso a organização desenvolva uma ferramenta própria, é necessário que a ferramenta seja construída seguindo um processo de desenvolvimento, elucidando requisitos, fazendo uma programação bem documentada e testando a própria ferramenta.

Independente da forma escolhida, a idéia principal é que a organização tenha em mente os seus objetivos, para poder ter em mãos, o que realmente precisa.

***Templates*: Plano de Automação e Relatório de *Status* da Automação**

Os *templates* foram criados para dar suporte às atividades do Processo de Automação proposto por este trabalho.

O *template* do Plano de Automação foi criado para dar suporte ao Gerente de Testes na atividade de Planejar Testes. O *template* do Plano de Automação pode ser encontrado no Apêndice A.

O *template* do Relatório de *Status* da Automação é extremamente necessário para o bom andamento de todo o Processo de Automação, é onde estarão descritos todas as decisões tomadas e todos os problemas encontrados. O *template* do Relatório de *Status* da Automação pode ser encontrado no Apêndice B.

4.4 Atividades do Processo

Um fluxo principal do Processo de Automação foi criado, e está apresentado na Figura 4.6, que mostra todas as atividades que estão contidas no mesmo, Planejar Automação, Selecionar Casos de Teste, Automatizar Casos de Teste, Avaliar Automação e Controlar Automação.

Nesta figura, pode ser visto o relacionamento entre as atividades, mostrando qual a ordem de execução das atividades, no fluxo a esquerda, e o fluxo a direita, que pode ser executado paralelamente ao fluxo da esquerda, desde que, em ciclos de automação diferentes.

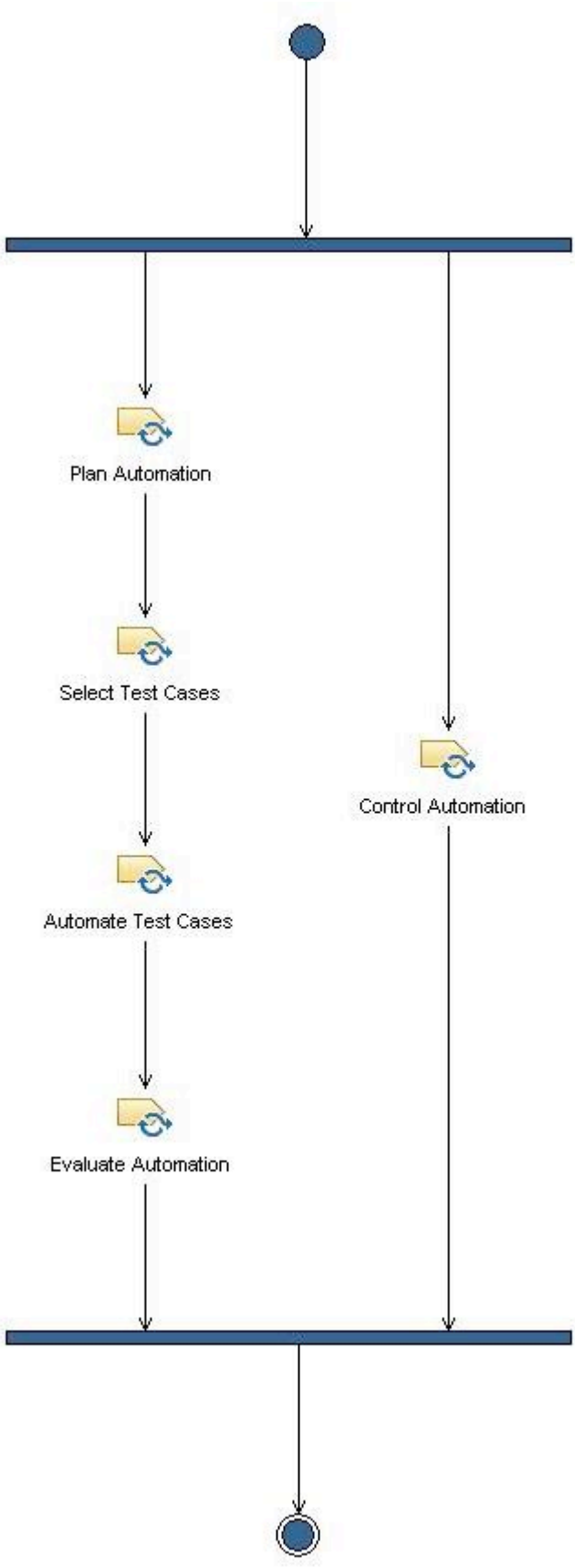

Figura 4.6 Fluxo do Processo de Automação de Testes

Nesta subseção iremos explicar como cada atividade é desenvolvida durante a execução do Processo de Automação proposto.

Planejar Automação

Planejar Automação para indicar qual funcionalidade será automatizada, quem e quando isto será feito.

Nesta atividade o Gerente de Testes solicita e planeja a automação de uma determinada funcionalidade, e para isto define como tudo será feito, através do artefato Plano de Automação, seguindo alguns passos:

- O Gerente de Testes indica qual funcionalidade irá ser automatizada de acordo com o Plano de Testes onde o Gerente de Testes pode selecionar uma funcionalidade baseado na freqüência de uso de alguns Casos de Teste, ou de acordo com o *Release Notes* de uma *build* importante que estará sendo testada e necessita que alguns Casos de Teste sejam automatizados.
- É importante indicar onde os Casos de Teste estão disponíveis para facilitar para quando o Engenheiro de Testes for executar a atividade Selecionar Casos de Teste.
- O Gerente de Testes define o cronograma da automação da funcionalidade selecionada. É algo muito importante que deve ser feito prestando atenção no cronograma do projeto como um todo de forma a não causar prejuízos nas demais atividades da organização.
- De acordo com a alocação do time do projeto como um todo, o Gerente de Testes deve selecionar os membros do time que estão disponíveis para esta atividade mas também prestando atenção se os membros do time disponíveis se adéquam aos papéis necessários para executar o Processo de Automação.
- É necessário indicar qual o ambiente da Automação de Testes que será utilizado durante o Processo de Automação.
- É necessário indicar quantos ciclos de teste, o *Script* de Testes estará sob controle na atividade Controlar Automação.

Selecionar Casos de Teste

Analisar e selecionar quais Casos de Testes poderá ser automatizado. É nesta atividade que o Engenheiro de Testes irá selecionar quais Casos de Testes podem ser automatizados e quais são interessantes de serem automatizados.

Esta é uma atividade bastante detalhada e que necessita que o Engenheiro de Testes tenha um bom conhecimento sobre Técnicas de Seleção de Casos de Teste, bem como da regra de negócio. Isto é feito a partir do seguimento de algumas etapas:

- Apesar da Suíte de Testes conter todos os Casos de Teste, existe alguns que não podem ser automatizados. Existem alguns Casos de Teste que obrigatoriamente necessitam de interferência humana, então existem alguns Casos de Teste que nunca poderão se tornar automáticos, isto depende do tipo de software que está sendo testado. Então, é importante selecionar os Casos de Teste que podem realmente ser automatizados, para evitar perder tempo nas atividades seguinte.
- Apesar de um Caso de Teste poder ser automatizado, algumas vezes não é viável automatizar porque isto pode tomar mais tempo para automatizar do que executar várias vezes.
- De acordo com os Casos de Teste selecionados para ser automatizados, o Engenheiro de Testes deve escolher que Técnica de Seleção de Casos de Teste é adequada a eles. Dependendo da organização, e do tipo de software sendo testado, alguma Técnica de Seleção de Casos de Teste pode ser criada ou pode ser utilizada uma técnica conhecida que já está estabelecida na literatura [22]. A Técnica de Seleção de Casos de Teste escolhida deve ser anotada no Relatório de *Status* da Automação.
- Após definida a melhor Técnica de Seleção de Casos de Teste para ser utilizada, deve ser aplicada nos Casos de Teste selecionados para selecionar os Casos de Testes que serão realmente automatizados, de acordo com a seleção resultante da técnica. É importante anotar no Relatório de *Status* da Automação os Casos de Teste selecionados pela técnica.

Automatizar Casos de Teste

Esta é a atividade principal do Processo de Automação onde a automação realmente é feita. Para isto, alguma ferramenta deve ser utilizada, podendo ser uma criada pela equipe ou alguma ferramenta do mercado, aberta ou proprietária.

Faz-se necessário que esta atividade seja feita seguindo padrões pré-estabelecidos, de acordo com a ferramenta utilizada, podendo ser algo já existente, ou um padrão proposto pela própria empresa. Uma boa automação pode ser alcançada seguindo os seguintes passos:

- Antes de realmente automatizar o teste, é necessário configurar o ambiente de automação de testes, indicado na atividade Planejar Automação. Este é o estágio que a Ferramenta de Automação de Testes é configurada e preparada para ser usada na automação. As outras ferramentas para dar suporte à atividade de automação são também configuradas neste momento. É necessário checar se todo o ambiente necessário está configurado de forma a evitar perda de tempo durante a automação. Algum problema encontrado neste estágio deve ser anotado no Relatório de *Status* da Automação.
- É esta é a atividade principal do processo inteiro e o objetivo real pode ser atingido neste estágio. Com o ambiente configurado, o Engenheiro de Automação de Testes irá automatizar os Casos de Testes. Isto pode ser feito executando uma Ferramenta de Automação de Testes ou implementando os Casos de Teste para ser utilizado por esta ferramenta, ou talvez, pode ser feito de acordo com o caminho de automação definido pela organização. Algum problema encontrado neste estágio deve se anotado no Relatório de *Status* da Automação. É interessante que a organização defina políticas e boas práticas para a utilização da ferramenta, que deverão ser seguidas.

Avaliar Automação

Validar a automação feita para ter certeza que está correta. Isto se faz necessário para verificar se tudo que foi selecionado realmente foi feito. Serve também, para validar se o resultado da atividade Automatizar Casos de Testes é testável e está coerente com o original. Alguns passos são necessários para atingir este objetivo:

- É importante checar se os Casos de Teste selecionados na atividade Selecionar Casos de Teste foram realmente automatizados. Pode ser checado com a ajuda do Relatório de *Status* da Automação que contém a informação dos Casos de Teste selecionados que deveriam ter sido automatizados. Os resultados desta avaliação devem ser anotados no Relatório de *Status* da Automação.

- Uma avaliação importante que deve ser feita é executar os Casos de Teste automatizados (os *Scripts* de Teste) uma vez para garantir que o teste pode ser executado e nenhum problema existe nele. Os resultados do teste devem ser anotados no Relatório de *Status* da Automação. Se algum *Script* de Teste falhou no processo de validação, ele retorna para a atividade Automatizar Casos de Teste para corrigir os problemas encontrados.

Controlar Automação

Monitorar a execução dos Casos de Teste automatizados durante alguns ciclos de teste. Esta é uma forma de se avaliar durante alguns ciclos de teste, se a automação foi feita corretamente e se os *Scripts* não estão com nenhum problema. Deve-se seguir os seguintes passos para executar esta atividade:

- O objetivo desta atividade é acompanhar o resultado dos Casos de Teste automatizados durante alguns ciclos de teste. É importante estabelecer quantos ciclos de teste os *Scripts* de Teste devem estar sob controle. Isto deve ser estabelecido no Plano de Automação. Se algum *Script* de Teste não funcionou corretamente em algum dos ciclos de teste estabelecidos, deve voltar para a atividade Automatizar Casos de Teste, e o problema encontrado deve ser anotado no Relatório de *Status* da Automação. Após a quantidade estabelecida de ciclos de teste, se um *Script* de Teste sempre funcionou corretamente, este caso de teste é liberado e mais nenhum controle deve ser feito com ele.
- É interessante medir o tempo de execução dos Casos de Teste automatizados, de forma a poder fazer uma comparação com o tempo de execução manual do mesmo Caso de Teste para avaliar o ganho que a automação trouxe para a organização.

Capítulo 5

Conclusões e Trabalhos Futuros

Várias são os fatores que podem ocasionar problemas em sistemas desenvolvidos por empresas de software [23]. E a maioria destes fatores é proveniente de erros humanos, testar é uma solução extremamente interessante e pode trazer os benefícios e a segurança ao desenvolvimento de um projeto.

Automatizar a execução dos testes de uma organização pode trazer vários benefícios, como também pode causar prejuízos enormes, para isto, é necessário que a automação seja bem planejada e que a organização defina quais os reais objetivos a serem alcançados. Um Processo de Automação se faz necessário de forma a padronizar como esta automação será feita.

5.1 Contribuições

Este trabalho tem como contribuição principal a proposta de um Processo de Automação de Testes de Software que garanta a uniformização dos trabalhos que são necessários para se automatizar os testes de uma organização.

Com este processo, busca-se garantir uma maior eficácia na utilização do tempo disponível para a execução dos testes, pois a execução de testes automatizados tende a ser bem mais rápida do que a execução de testes manuais.

Como parte integrante do Processo de Automação proposto, não deixando de serem considerados como contribuições importantes, foram criados dois *templates*, um do Plano de Automação e um do Relatório de *Status* da Automação.

Não podemos deixar de mencionar que o Processo de Automação proposto está disponível na internet[2], possibilitando que as pessoas interessadas na área de Automação de Testes de Software possam acessar e visualizar o processo proposto. Isto fará com que a área de Testes de Software, assim como a de Automação de Testes de Software tenha uma visibilidade ainda maior no Brasil, visto que poucos são as referências existentes na área de Automação de Testes de Software.

5.2 Dificuldades Encontradas

Ao realizar este trabalho, podemos destacar algumas dificuldades encontradas que limitaram o seu desenvolvimento. A primeira diz respeito a pouca quantidade de bibliografia disponível na área de Automação de Testes de Software, principalmente tratando de processos de automação. Apenas o trabalho de DUSTIN, RASHKA e JOHN [3] pode contribuir mais diretamente como um processo de automação.

As limitações de prazo e tamanho da documentação também podem ser consideradas como restrições para a realização deste trabalho..

5.3 Trabalhos Relacionados

Para desenvolver este trabalho, dois trabalhos puderam contribuir com mais destaque. O primeiro deles foi o de DUSTIN, RASHKA e JOHN [3], que propõe um processo de automação, mas se preocupando com a automação das atividades do processo de testes, não diretamente com a abordagem e escopo deste trabalho.

Outro trabalho importante é o de FEWSTER e GRAHAM [14], que é uma excelente fonte bibliográfica sobre automação de testes de software, mostrando algumas abordagens de automação existentes, mesmo não sendo o escopo deste trabalho, pode contribuir bem, com todos os conteúdos na área de automação.

[2] Disponível na web em: dsc.upe.br/~pco/automationprocess

5.4 Trabalhos Futuros

Dentro da área de Automação de Testes vários são os trabalhos que podem dar continuidade ao presente trabalho, ampliando o conhecimento e abrangendo mais áreas relacionadas à automação. Uma proposta inicial é criar métricas e um planejamento para a execução de um estudo de caso com o processo proposto, que servirá como avaliação deste Processo de Automação, podendo verificar os benefícios e as perdas que podem ocorrer com a utilização do mesmo, gerando índices de melhoria para o processo.

Outra proposta seria desenvolver as atividades de execução de um Processo de Testes, de forma a garantir que a etapa de execução esteja preparada para funcionar com *Scripts* de Teste criados pelo Processo de Automação proposto.

Outra proposta seria ampliar o Processo de Automação, de forma a padronizar um possível desenvolvimento de uma ferramenta de automação para a organização. Este processo iria abranger desde a etapa de definição dos objetivos para a ferramenta, bem como os requisitos, e a forma de desenvolvimento.

Um estudo na área de Técnicas de Seleção de Casos de Testes é interessante, de forma a avaliar quais são as melhores técnicas existentes, de forma a criar um *guideline* sobre Técnicas de Seleção de Casos de Teste para incluir no processo proposto.

Por fim, um estudo sobre metodologias ágeis poderia ser feito de forma a avaliar se o processo proposto poderia ser melhorado com a utilização de alguma metodologia ágil, e em caso positivo, propor uma melhoria para este processo.

Bibliografia

[1] KOOMEN, T.; POL, M. **Test Process Improvement. A practical step-by-step guide to structured testing**, Addison-Wesley, 218 p., 1999.

[2] Standard Glossary of Software Engineering Terminlogy, IEEE, 1991.

[3] DUSTIN, E.; RASHKA, J.; JOHN, P., **Automated Software Testing. Introduction, Management and Performance**, Addison-Wesley, 575 p., 1999.

[4] AMBLER, S. **Process Patterns: Building Large-Scale Systems Using Object Technology**, Cambridge, University Press/SIGS Books, 1998.

[5] Guide to the Software Engineering Body of Knowledge, IEEE Computer Society, 2004.

[6] HARROLD, M. J., **Testing: A Roadmap**. In: Future of Software Engineering, 22nd International Conference on Software Engineering, June 2000.

[7] Base de Conhecimento para Certificação de Teste, International Software Testing Qualifications Board, 2007.

[8] Glossário Padrão de Termos Utilizados em Teste de Software, International Software Testing Qualifications Board, Versão 1.3br, 2007.

[9] GRAHAM, D.; VEENENDAAL, E.V.; EVANS, I.; BLACK, R., **Foundations of Software Testing – ISTQB Certification,** Thomson, 258 p., 2007.

[10] CRAIG, R.D.; JASKIEL, S.P., Systematic Software Testing. Artech House, 536p., 2002.

[11] COPELAND, L., **A practitioner´s Guide to Test Design**. Artech House, 294 p., 2004.

[12] PRESSMAN, R. S., **Engenharia de Software.** 6.ed., McGraw-Hill Interamericana do Brasil Ltda, 720 p., 2006.

[13] SOMMERVILLE, I., **Software Engineering**. 8.ed. Addison-Wesley, 864 p., 2006.

[14] FEWSTER, M.; GRAHAM, D., **Software Test Automation. Effective use of test execution tools.** Addison-Wesley, 574 p., 1999.

[15] **CenterLine Software, Inc. Survey.** CenterLine é uma empresa de criação de ferramentas e automação de testes de software de Cambridge, Massachusets, 1996.

[16] **The Standish Group Report - CHAOS.** Disponível em: <http://www.educause.edu/ir/library/pdf/NCP08083B.pdf> Acesso em: 11 de maio de 2008

[17] LITTLEWOOD, B., **How Good are Software Reliability Predictions? Software Reliability Achievement and Assessment.** Oxford: Blackwell Scientific Publications, 1987.

[18] KANER, C., **Avoiding Shelfware: A managers' View of Automated GUI Testing.** STAR East, Orlando, FL, 1998.

[19] BACH, J., **Test Automation Snake Oil.** Windows Technical Journal, p. 40-44, 1996.

[20] PETTICHORD, B., **Seven Steps to Test Automation Success.** STAR WEST, San Jose, 1999, Versão 2001. Disponível em: < http://www.io.com/~wazmo/papers/seven_steps.html>

[21] KANER, C., **Improving the Maintainability of Automated Test Suites.** Tenth international Quality Week, San Francisco, CA, 1997.

[22] VIANA, V. M. A., **Uma Metodologia para Seleção de Testes de Regressão para Automação.** Tese de Conclusão de Curso de Mestrado em Ciência da Computação. Centro de Informática, UFPE, Recife.

[23] DELAMARO, M. E.; MALDONADO, J. C.; JINO, M., **Introdução ao Teste de Software**. Editora Campus, 394 p., 2007.

Apêndice A

Template: Plano de Automação

Feature to be Automated

<Indicate here the desired features to be automated>

- *<Feature 1>*
- *<Feature 2>*
- *...*

Test Cases Location

<Indicate here where the test cases of the indicated feature are>

Schedule

<Plan here the activities of the Automation Process, the staff involved in it, and the initial date of the activities>

Activity	Staff	Date
Select Test Cases	*<Staff Name>*	*<aaaa/mm/dd>*
Automate Test Cases	*<Staff Name>*	*<aaaa/mm/dd>*
Evaluate Automation	*<Staff Name>*	*<aaaa/mm/dd>*

Test Automation Environment

<Indicate here all necessary environment to automate the test cases, in other words, the hardware and software involved>

- *<1>*
- *<2>*
- *...*

Control Test Automation

<Indicate here how many test cycles the test scripts generated by the Automate Test Cases activity will be under control in the Control Automation>

Apêndice B

Template: Relatório de *Status* da Automação

Test Cases Not to be Automated

<Indicate here which test cases could not be automated, and its motivation>

Test Case Selection Technique

<Indicate here the test case selection technique that were used to select the test cases to be automated>

Test Cases Selected

<Indicate here the test cases that were selected to be automated, from the specific feature indicated in the Automation Plan>

- *<Test Case 1>*
- *<Test Case 2>*
- *...*

Test Case Automation

<Indicate here if any problem occured during the environment configuration and during the automation itself>

Automation Evaluation

<Indicate if there was any test case planned to be automated that there was not. Also indicate here, if any test case that was automated showed any porblem or inconsistency>

Control Test Automation

<Indicate here if any test case did not work well during the quantity of test cycles planned in the Automation Plan. It is important to indicate the test script and the cause of the problem>