UNIVERSIDADE DO VALE DO RIO DOS SINOS – UNISINOS

UNIDADE ACADÊMICA DE PESQUISA E PÓS-GRADUAÇÃO

ESPECIALIZAÇÃO EM QUALIDADE DE SOFTWARE

Lauren Girardi Cristofoli

PROPOSTA DE UMA ABORDAGEM PARA A GESTÃO DE SOLICITAÇÃO DE ALTERAÇÃO PARA TESTES AUTOMATIZADOS PARA O SETOR DE TI DE UMA CORPORAÇÃO

Porto Alegre

2017

UNIVERSIDADE DO VALE DO RIO DOS SINOS – UNISINOS

UNIDADE ACADÊMICA DE PESQUISA E PÓS-GRADUAÇÃO

ESPECIALIZAÇÃO EM QUALIDADE DE SOFTWARE

Lauren Girardi Cristofoli

PROPOSTA DE UMA ABORDAGEM PARA A GESTÃO DE SOLICITAÇÃO DE ALTERAÇÃO PARA TESTES AUTOMATIZADOS PARA O SETOR DE TI DE UMA CORPORAÇÃO

Trabalho de Conclusão de Curso apresentado como requisito parcial para a obtenção do título de Especialista em Qualidade de Software, pelo curso de Pós-Graduação Lato Sensu em Qualidade de Software da Universidade do Vale do Rio dos Sinos – UNISINOS.
Orientador: Prof. Ms André Luiz de Castro Villas-Boas

Porto Alegre

2017

Proposta de uma abordagem para a gestão de solicitação de alteração para testes automatizados para o setor de TI de uma corporação

Lauren Girardi Cristofoli [1]

[1]Universidade do Vale do Rio dos Sinos (UNISINOS) – São Leopoldo – RS

laurencristofoli@gmail.com

Abstract. *This paper introduces the results of an action research applied in the information technology sector of an organization, in which a formal automated test change management process was applied. The purpose of this work was to design and evaluate how software configuration management practices can improve the processes of creating and changing automated tests by implementing a formal management cycle for requesting changes in test cases.*

Resumo. *Este artigo apresenta o resultado de uma pesquisa-ação realizada do setor de tecnologia de informação de uma organização, na qual se aplicou um processo de gestão formal de alteração para testes automatizados. O objetivo deste trabalho foi conceber e avaliar como as práticas de gerência de configuração de software podem melhorar os processos de criação e alteração para testes automatizados através da implementação de um ciclo de gestão formal para a solicitação de alteração em casos de testes.*

1. Introdução

O software tornou-se uma ferramenta essencial e um diferencial competitivo para as organizações, sendo aplicado em diversas áreas e promovendo a automatização na execução de processos. Entretanto, desenvolver software é uma tarefa complexa e que exige cuidados, em virtude da abstração e intangibilidade dos requisitos, o que possibilita múltiplas soluções para um mesmo cenário [Sommerville, 2011]. Assim, uma das principais preocupações das software houses e fábricas de software é garantir a qualidade do produto entregue. Contudo, para que seja possível um resultado final de qualidade, é importante que haja um cuidado para com o processo de criação desse produto.

Uma das principais práticas que auxiliam nesse processo de garantia de qualidade é o desenvolvimento e a execução de testes bem estruturados, que visam validar se o software atende aos requisitos, se é seguro e opera suas funções corretamente. Ao ser aprovado na etapa de testes, temos redução do retrabalho em produção, melhoria da qualidade do produto e o aumento do grau de satisfação do usuário.

Sobre a definição de teste, Hetzel (1973) afirma que "Testar é o processo de certificar que o programa faz o que era suposto fazer"; para Glen Myers (1979) "Testar é o processo de executar um programa ou sistema com o objetivo de encontrar erros". Ainda sobre a prática de testes, Villas-Boas (2003, p. 1) afirma que “Teste e Validação de Software é uma das áreas da Engenharia de Software e constitui um dos elementos

para aprimorar a produtividade e fornecer evidências da confiabilidade do software.”. Para Pressman (2001),"O teste de software é um elemento crítico da garantia de qualidade de software e representa a última revisão da especificação, do projeto e da codificação”.

A etapa de teste no software também produz artefatos tais como documentação de casos de teste, relatórios de execução e *scripts* de execução de suítes de testes automatizados, por exemplo. Diante disso, observa-se a importância de, assim como nas demais atividades do desenvolvimento de software, manter um controle sobre os produtos oriundos das atividades de testes.

Com o intuito de auxiliar na organização da produção do software e fornecer um fundamento sólido para todas as outras atividades de engenharia de software [CAPRETZ, 1992] a Gerência de Configuração Software (GCS) foca as seguintes funções [BERSOFF et al., 1980; BUCKLE, 1982]:

- Identificação de configuração: itens que constituem uma configuração;
- Controle de configuração: identificar quais passos no processo de alteração afetam uma configuração;
- Auditoria de configuração: identificar quais são as diferenças entre as versões.
  Contabilização da situação de configuração: quais modificações foram feitas por qual programador.

Convém, entretanto, salientar que a GCS busca identificar a configuração de um sistema com a finalidade de controlar as mudanças desta configuração, objetivando manter a sua integridade e possibilitar o seu rastreamento através do ciclo de vida do sistema [BERSOFF et al., 1980]. Assim, a GCS não fornece um método de projeto, nem um modelo de ciclo de vida, nem define como a qualidade dos itens deve ser julgada [VILLAS-BOAS, 2003].

Neste cenário, a fim de garantir a qualidade do desenvolvimento do seu ERP, uma grande empresa do ramo calçadista, situada na Serra Gaúcha, criou, dentro da sua Divisão de Tecnologia de Informação (TI), o setor de Qualidade de Software (SQA), cujo o principal objetivo é, por meio do desenvolvimento de testes automatizados, reduzir o índice de retrabalhos no desenvolvimento de software e aumentar o grau de confiabilidade e satisfação dos usuários com o seu sistema.

A equipe SQA vem atuando desde meados de 2013 e seu principal foco é a produção de testes automatizados. Considerada uma equipe de suporte ao desenvolvimento, intensifica e direciona esforços para atingir o maior percentual possível de cobertura de testes para o sistema ERP. Entretanto, por tratar-se de uma equipe recente, ainda não há um processo bem definido para a solicitação de criação e alteração para testes. Percebeu-se, também, que pelo fato de a equipe SQA não estar corretamente inserida no fluxo de trabalho da equipe ERP, ocorre a falha de comunicação sobre as alterações realizadas nos programas. Somando-se isso à falta de vínculo entre programas e *baselines* de teste, percebeu-se com o passar do tempo que os testes vão ficando desatualizados e sem manutenção, o que, por sua vez, provoca divergências quando esses são executados, resultando no reporte de falsos positivos, ou seja, a equipe de desenvolvimento recebe um alerta de erro quando, na realidade, o teste estava desatualizado. Além do impacto de retrabalho para a equipe de teste, existe o impacto sobre a relação de confiança entre desenvolvedores e testadores.

Diante disso, criou-se a seguinte questão de pesquisa: “Como práticas de gerência de configuração podem melhorar os processos de criação e alteração para testes automatizados no setor de TI de uma corporação?”

A fim de atender à questão de pesquisa, a principal motivação desse trabalho de pesquisa é propor a melhoria da qualidade do produto de software através do uso de um ciclo de gestão formal para solicitação de manutenção de testes automatizados no processo de desenvolvimento de software do setor de TI de uma organização específica.

Diante disso, o objetivo geral dessa pesquisa é conceber e avaliar como as práticas de gerência de configuração podem melhorar os processos de criação e alteração para testes automatizados através da implementação de um ciclo de gestão formal para a solicitação de alteração em casos de testes. Visando atingir o objetivo geral, foram definidos os seguintes objetivos específicos: identificar os pontos críticos do atual processo de gerenciamento de mudança nos testes automatizados; criar e implantar o ciclo formal de solicitação de mudança; analisar os resultados obtidos através da inclusão do processo de gestão para alteração de casos de teste na atividade de execução de testes automatizados que acontece no processo de desenvolvimento de software da organização.

Este artigo está organizado em 6 seções, incluindo Introdução e Conclusão. A seção 2 apresenta o referencial teórico utilizado no desenvolvimento do trabalho, descrevendo os conceitos de teste de software e gerência de configuração e suas práticas. A seção 3 apresenta a metodologia utilizada na construção do estudo. A seção 4 apresenta o estudo elaborado na pesquisa. Na seção 5 é descrita a análise e os resultados na pesquisa.

2. Referencial Teórico

Nessa seção são abordados os conceitos e técnicas de teste de software e gerência de configuração de software.

2.1. O que é teste?

O processo de desenvolvimento de software é composto por uma série de atividades que, quando executadas em conjunto, resultam no produto de software. Algumas dessas atividades são denominadas técnicas de Garantia de Qualidade de Software, pois buscam introduzir boas práticas na construção de software a fim de evitar que o produto final tenha erros ou não atenda ao seu propósito. Entretanto, como detectar essas inconsistências?

A atividade de teste tem por objetivo minimizar a ocorrência de erros e/ou inconsistência no software. Para Sommerville (2011, p.144), “o teste é destinado a mostrar que um programa faz o que é proposto fazer e para descobrir os defeitos do programa antes do uso.”

A etapa de teste é uma das atividades de Verificação e Validação mais utilizada, pois se constitui em um dos elementos que fornece evidências da confiabilidade do software [MALDONADO, 1991]. De acordo com [PRESSMAN, 2005], a atividade de teste possui um papel fundamental durante o desenvolvimento de software porque corresponde a última oportunidade de correção de eventuais problemas no produto antes da sua entrega final ao cliente.

Para Maldonado et al. (2004) “a atividade de teste consiste de uma análise dinâmica do produto e é uma atividade relevante para a identificação e eliminação de erros que persistem”. O teste é, portanto, uma atividade fundamental para a avaliação do software desenvolvido, pois o conjunto de informações oriundas dessa atividade subsidiam as atividades de depuração, manutenção e estimativa de confiabilidade do software [PRESSMAN, 2001]. Entretanto, a atividade de teste demanda atenção e exige conhecimentos, habilidades e infraestrutura específicos [CRESPO et al.., 2004].

De acordo com Sommerville (2011), um teste de software executa um programa com dados fictícios com duas principais finalidades:

1. Demonstrar que o software atende a seus requisitos;
2. Encontrar situações onde o software comporta-se de maneira incorreta/inesperada em relação às especificações. Ou seja, buscar defeitos de software.

Para atingir esses objetivos, existem quatro principais etapas de teste: planejamento de testes, projeto de casos de teste, execução e avaliação dos resultados dos testes [BEIZER, 1990; MYERS, 1979; PRESSMAN, 2001; MALDONADO, 1991].

A fim de garantir assertividade na atividade de teste, é importante que essa seja planejada através de uma estratégia que compreenda o nível de teste (definição da fase de desenvolvimento do software que o teste será aplicado), a técnica de teste a ser utilizada, o critério de teste a ser adotado e o tipo de teste a ser aplicado. [CRESPO et al.., 2004]

O nível de teste deve ser escolhido observando-se alguns aspectos da fase de desenvolvimento que o software se encontra. De acordo com a evolução do desenvolvimento do software, diferentes estágios de teste podem ser contemplados, conforme suas características. Para escolha do nível de teste, estão disponíveis [SOMERVILLE, 2011]:

1. Testes em desenvolvimento, onde o sistema é testado durante o desenvolvimento para descobrir bugs e defeitos. Nesse estágio, estão contemplados testes unitários, de integração e de sistema.
2. Testes de release, onde o sistema é completamente testado antes da sua entrega para o cliente.
3. Testes de usuário, onde os usuários testam o sistema em seu próprio ambiente.

Quando a fase do ciclo do software demanda testes em desenvolvimento, o teste de unidade, usualmente aplicado em unidades menores, visa identificar erros de lógica e de implementação separadamente em cada módulo do sistema. Após os testes unitários, o teste de integração busca validar se a estrutura do sistema funciona adequadamente e verifica se não há erros associados a interfaces entre os módulos. Para completar, o teste de sistema visa identificar erros de funções e características de desempenho que estejam em desacordo com a especificação [PRESSMAN, 2001].

Ao avançar no ciclo de desenvolvimento, são avaliados testes de aceitação, cujo os quais visam mensurar a aceitação do sistema pelo usuário e, quando o software entra em fase de manutenção, aplicam-se testes de regressão [CRESPO et al.., 2004].

Existem duas técnicas de teste que nortearão a escolha de critérios para a geração de casos de teste: teste estrutural, cujo o objetivo é identificar falhas internas (na estrutura) do software e teste funcional, com a finalidade de garantir que os requisitos do software sejam plenamente atendidos [CRESPO et al.., 2004; MALDONADO et al.., 2004].

Nesse viés, Howden (1987) sugere duas classificações de teste: baseado em especificação e baseado em programa. O teste baseado em especificação corresponde ao teste de caixa-preta, onde o objetivo é determinar se o programa satisfaz requisitos funcionais e não-funcionais. Já o teste baseado em programa, também conhecido como teste de caixa-branca, busca inspecionar o correto funcionamento do código-fonte e não da sua especificação [PERRY;KAISER, 1990].

De acordo com a técnica de teste abordada, é necessário definir quais serão os critérios de teste. Os critérios de teste são compostos por um conjunto de elementos e características do software escolhidos para serem testados e têm como objetivo nortear o testador no momento da construção do teste. Em geral, os critérios de teste abrangem elementos de software como linhas de comando, funções implementadas, variáveis definidas no software, laços de repetição, blocos condicionais e requisitos do software. [MALDONADO et al.., 2004].

Por fim, os tipos de teste são definidos de acordo com as características do software que podem ser testadas. Assim, são elencados os testes de funcionalidade, interface, desempenho, carga (stress), usabilidade, volume e segurança [MALDONADO et al.., 2004].

A Figura 1 ilustra a ligação existente entre os níveis de teste, técnicas de teste, tipos de teste e critérios de teste.

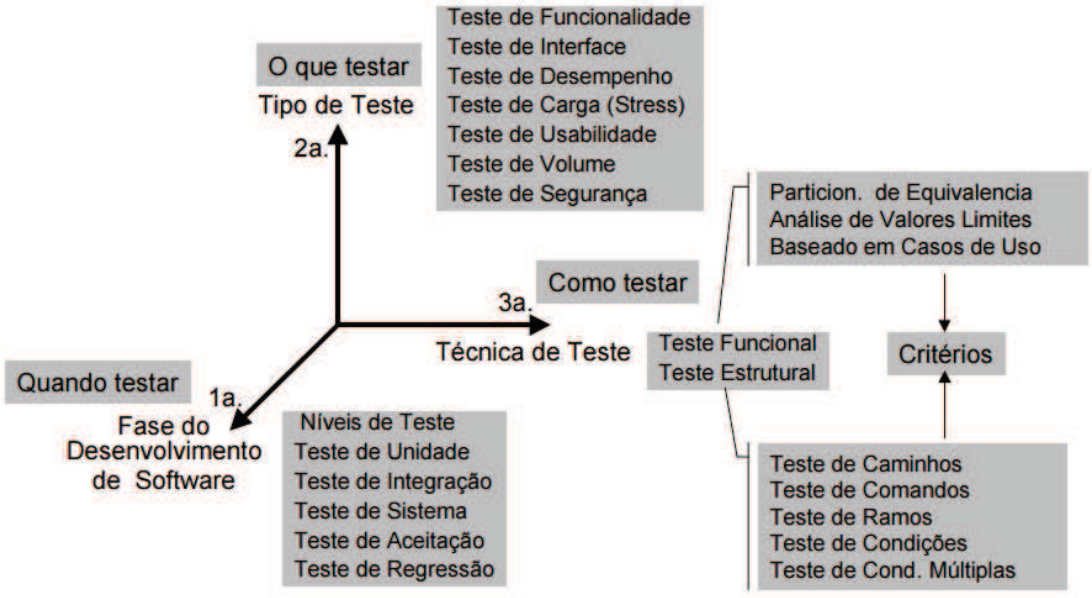

Figura 1. Relação entre níveis, técnicas, critérios e tipos de teste [CRESPO et al.., 2004]

2.2. Teste automatizado

Os testes de software são atividades de alto impacto no processo de desenvolvimento de sistemas de software, cujo o principal objetivo é identificar a presença de falhas e erros no software o mais cedo possível, a fim de minimizar os impactos e custo de correção. Entretanto, embora da sua importância, a atividade de teste pode não ser executada adequadamente, em virtude de fatores como limitações de tempo, recursos e qualificação técnica dos envolvidos [FANTINATO et al.., 2005].

Os testes automatizados são definidos como programas ou *scripts* simples que exercitam funcionalidades do sistema em teste e fazem verificações automáticas nos efeitos colaterais obtidos [FEWSTER; GRAHAM, 1999]. Para Sommerville (2011), nos testes automatizados, os testes são codificados em um programa que é executado cada vez que o sistema em desenvolvimento é testado. Assim, os testes automatizados possuem, como finalidade, verificar se um programa faz aquilo que é proposto [SOMMERVILLE, 2011].

A automação de testes permite que a atividade de testar o sistema seja executada sem a intervenção humana, possibilitando, portanto, que ferramentas computacionais realizem essa tarefa rapidamente [SOMMERVILLE, 2011; BERNARDO, 2011]. Assim, automatizar testes traz uma série de benefícios, como: substituir o trabalho manual de teste, ganho de produtividade no desenvolvimento do software, redução de custo de retrabalho e aumento da repetibilidade e precisão dos testes, possibilidade de execução paralela de testes e a facilidade da criação de casos complexos de testes. [PETROSKI, 2009; BERNARDO, 2011; FANTINATO et al.., 2005].

Entretanto, é importante salientar que uma das principais vantagens em adotar a abordagem de testes automatizados é que os casos de teste podem ser facilmente e repetidamente executados, paralelamente, a qualquer momento e com pouco esforço. Assim, é possível garantir que passos importantes não serão ignorados por falha humana e facilita a identificação de possíveis comportamentos indesejados [BERNARDO, 2011].

2.3. Gerência de Configuração

O processo de desenvolvimento de um software abrange uma série de conteúdos produzidos, tais como documentação, código-fonte, dados, manuais, casos de teste, entre outros. Ao longo do ciclo de vida do software muitos desses itens sofrem alterações, seja por correção de defeitos, seja por mudanças de requisitos [FIGUEIREDO; SANTOS; ROCHA, 2004].

Diante disso, a Gerência de Configuração de Software (GCS) busca evitar a perda do controle do projeto de software através do controle e gerenciamento da evolução do sistema [ESTUBLIER, 2000]. Assim, o processo de GCS foca exclusivamente em controlar as mudanças que acontecem no projeto, evitando a perda do controle de alteração sobre os itens de informação produzidos [WHITGIFH, 1991].

A GCS pode ser definida como uma abordagem disciplinada para gerenciar a evolução do desenvolvimento de software [ESTUBLIER et al., 2002], cujo a qual vem sendo amplamente adotada [BURROWS;DART;GEORGE, 1996] através do seu gerenciamento de artefatos do processo de desenvolvimento, controle de mudanças do software e auxílio no desenvolvimento do software [MEI;ZHANG;YANG, 2001].

Caso as mudanças dos artefatos não sejam devidamente documentadas e comunicadas, a equipe envolvida pode ser impactada por uma série de problemas, tais como: dois ou mais desenvolvedores estarem alterando um mesmo artefato ao mesmo tempo, não saber qual a versão mais atual de um artefato, não refletir alteração nos artefatos impactados por um artefato em alteração, dentre outros [NUNES, 2005]. Assim, ao incluir um processo de gerência de configuração ao longo do ciclo de vida do software é possível evitar que esses transtornos ocorram, através do acompanhamento de artefatos e ferramentas [SANCHES et al, 2001].

2.3.1. O processo de Gerência de Configuração

Diversos autores como [BERSOFF et al., 1980; 1984], [BLISS, 1993], [BUCKLE, 1982] e inclusive a norma NBR ISO 10007 [ABNT, 1996] reconhecem o processo da GCS como sendo constituído por 4 principais atividades (Figura 2):

1. Identificação de configuração;
2. Controle de configuração;
3. Contabilização da Situação de Configuração (status/balanço da configuração);
4. Auditoria de Configuração.

Figura 2. Processo de Gerência de Configuração

A atividade de identificação dos itens de configuração (IC) tem como finalidade nomear os componentes básicos de maneira única, a fim de facilitar o controle sobre os itens. São considerados itens de configuração trechos de código-fonte, documentos e outros artefatos associados. O padrão da ABNT/ISO [ABNT, 1996] prevê as seguintes sub-atividades:

a. seleção dos itens de configuração;
b. determinação de estrutura da configuração;
c. documentação dos itens de configuração;
d. sistema de numeração;
e. estabelecimento de *baselines*.

O controle de configuração tem como produto uma configuração-base (*baseline*) [BERSOFF, 1984]. A *baseline* tem como finalidade formalizar que um ou mais ICs estão em conformidade com os requisitos do projeto e, deste ponto em diante, devem ser formalmente controlados e protegidos contra alterações não autorizadas [VILLAS-BOAS, 2003]. É importante que nessa etapa de processo de controle algumas informações sejam documentadas detalhadamente, como, por exemplo [VILLAS-BOAS, 2003]:

- documentação e justificação da alteração;
- avaliação das consequências da alteração;
- aprovação/recusa dos pedidos de alteração;
- implementação e verificação das alterações e desvios e concessões.

A fim de garantir a integridade da configuração, é importante que os ICs sejam mantidos em um ambiente que possa protegê-los de alterações não autorizadas, perda acidental dos IC, que forneça mecanismos para recuperação controlada de todos os dados armazenados e proporcione consistência entre a configuração construída e a especificada. Assim, é possível conhecer a situação da configuração (*status*) e obter informações importantes como quem efetuou uma alteração, quando ela foi executada e o que foi feito. Portanto, a atividade de contabilização da situação de configuração tem por finalidade permitir o rastreamento de todas as alterações efetuadas sobre os ICs. Esse monitoramento de *status* deve iniciar assim que os ICs são gerados.

Por fim, a auditoria de configuração deve ser executada antes da definição da *baseline*, a fim de certificar que a mesma será gerada com as informações precisas do IC, além de garantir que o produto está de acordo com os requisitos. A auditoria é utilizada para garantir que a passagem de uma configuração a outra não descaracteriza o produto [VILLAS-BOAS, 2003]. Para que esse processo de auditoria aconteça, são executadas duas atividades: verificação, que assegura a coerência entre as configurações; e validação, para determinar a coerência entre as *baselines* e especificações dos requisitos.

O controle efetivo de um IC inicia tão logo esse seja auditado e suas características sejam aceitas como corretas nesse momento. Gera-se então a *baseline*, que representará um marco momentâneo sobre o que vale como base. A partir de então, esse item passa a sofrer um controle formal de alterações. Contudo, ao longo do projeto outros ICs podem constituir novas *baselines*, uma vez que esta representa um marco momentâneo, sendo passível de sofrer alterações.

2.4. Trabalhos Relacionados

Na busca pela garantia da qualidade do produto final de software encontram-se diversas pesquisas que visam aprimorar o processo de desenvolvimento através de propostas de melhorias.

Costa (2006) sugere a implementação de uma estratégia de automação de teste para aprimoramento da atual atividade de testes de uma organização, buscando integrar atividades como planejamento de testes, análises de falhas e gerência de configuração.

Villas-Boas (2003) propõe o uso de gerência de configuração para as atividades de teste unitário de software, buscando caracterizar um item de configuração para o teste e desenvolvendo um ambiente constituído por uma ferramenta de gestão de configuração que automatize a coleta, recuperação e controle do item de configuração de teste definido.

Crespo et. at. (2004) apresenta uma metodologia para teste de software desenvolvida pelo CenPRA (Centro de Pesquisas Renato Archer) que visa integrar as atividades de teste no contexto da melhoria dos processos de desenvolvimento. Essa metodologia proposta prevê um conjunto de atividades de testes que vai desde o

levantamento das necessidades da empresa, até treinamentos e acompanhamento de trabalhos realizados, constituindo um ciclo completo da implantação da atividade de teste.

3. Metodologia

A metodologia que norteará esse trabalho de pesquisa será de pesquisa-ação, de caráter quantitativo e exploratório. Os dados a serem analisados serão obtidos através da aplicação de questionário fechado e através da contagem de testes desatualizados (esta contagem será obtida através da documentação existente na empresa).

3.1. Delineamento da Pesquisa

A pesquisa será composta por duas etapas: revisão bibliográfica e uma pesquisa-ação, de caráter qualitativo, exploratório e de corte transversal. Conforme Gil (2010) orienta, a pesquisa-ação é indicada em casos onde o pesquisador vislumbra diagnosticar um problema específico em uma situação específica, onde o mesmo está envolto no contexto de forma cooperativa e participativa, objetivando alcançar resultados práticos, porém não generalizáveis.

O estudo analisará quantitativamente a melhora no processo de testes no desenvolvimento de software do setor de TI de uma corporação, após a implantação de um ciclo formal de gestão de solicitação de criação e alteração para testes automatizados, utilizando as práticas de gerência de configuração. A análise quantitativa será através de um indicador que mede a quantidade de testes detectados durante o ciclo de execução dos mesmos, que ocorre quinzenalmente.

3.2. Unidade de Análise

A unidade de análise escolhida para o estudo abrange duas equipes do setor de TI de uma corporação: a equipe de desenvolvimento ERP (fábrica de software) e a equipe de Qualidade de Software (SQA).

A SQA, onde a pesquisadora atua como Analista de Qualidade de Software, trabalha como equipe de apoio ao setor de desenvolvimento, desenvolvendo testes automatizados para garantir a qualidade e correto funcionamento do sistema de software. A equipe SQA é formada por quatro profissionais, sendo um Analista de Qualidade de Software e três programadores de teste. Já a equipe ERP é composta por 10 analistas de negócio, 3 analistas de sistema líderes, 4 analistas de sistemas e 12 desenvolvedores de software.

O fator motivador para escolha dessas equipes foi a técnica de amostragem por conveniência, uma vez que a pesquisadora possui facilidade de acesso aos documentos e pessoas envolvidas no estudo [VERGARA, 2007].

Entretanto, convém salientar, que os resultados interpretados referem-se à unidade de análise adotada, não sendo generalizáveis.

3.3. Coleta de Dados

A coleta de dados se dará em duas etapas: a primeira abrangerá a quantidade de testes automatizados que estão em produção porém estão desatualizados em relação ao programa ao qual estão vinculados e/ou desatualizados em relação ao processo. Essa

coleta foi obtida entre os meses de março e abril de 2017 e as informações foram resgatadas através da documentação da corporação.

A segunda parte da coleta será realizada pelo uso do instrumento de coleta questionário, composto por perguntas fechadas, conforme disposto no Apêndice A deste trabalho.

3.4. Análise de Dados

A análise de dados será realizada com base nos resultados dos questionários e comparando a quantidade de testes desatualizados em relação ao processo atual. As conclusões da análise terão foco voltado para a eficácia do novo processo de solicitação de alteração para testes automatizados proposto em relação ao processo atual, já que este contexto corresponde à questão de pesquisa do trabalho.

3.5. Limitações do Método de Estudo

O estudo contempla apenas a equipe de Qualidade de Software, contando com a participação e apoio dos colaboradores de equipe de desenvolvimento ERP da corporação, devido à limitação de tempo para a realização do mapeamento de processo, levantamento de dados, análise dos resultados e conclusão deste trabalho. Assim, os resultados não são generalizáveis, inclusive, no contexto a corporação objeto desse estudo.

4. Estudo

Nesta seção, o estudo de proposta de melhoria é contextualizado, iniciando pela apresentação da unidade de análise, explicando os atuais processos de criação e execução de testes automatizados da organização, passando para a etapa de Coleta Preliminar de Dados, responsável por identificar o cenário atual dos processos de integração da execução de testes automatizados na organização, utilizada como base para a modelagem da proposta da criação do processo de gestão formal de alteração dos testes automatizados. A subseção 4.4 apresenta o detalhamento do processo proposto, na subseção 4.5 explica-se a proposta do estudo e na subseção 4.6 apresenta-se a aplicação da proposta de processo.

4.1. Contextualização da Unidade de Análise

O trabalho foi realizado no setor de Tecnologia de Informação (TI) de uma grande empresa do ramo calçadista. Localizada em Farroupilha, serra gaúcha, a empresa iniciou sua trajetória no mercado em 1971. Atualmente, comercializa seus produtos para mercado interno e externo.

A empresa conta com um quadro de, aproximadamente, 20.000 colaboradores alocados em 5 unidades, sendo uma localizada em Farroupilha – Rio Grande do Sul, 3 unidades no Ceará, nos municípios de Fortaleza, Crato e Sobral e 1 unidade em Teixeira de Freitas – Bahia.

A divisão da Tecnologia de Informação é composta por 160 colaboradores, abrangendo profissionais técnicos em serviços e manutenção de infraestrutura, e técnicos que trabalham com suporte e desenvolvimento de sistemas.

A equipe diretiva da TI é composta por um gerente da divisão, duas assistentes administrativas e dois coordenadores, onde um é responsável pela coordenação da equipe de Sistemas e Fábrica de Software e o outro é responsável pela coordenação das equipes que prestam serviços de suporte e manutenção de infraestrutura. A equipe de sistemas é subdividida em três Fábricas de Software e uma célula de Planejamento e Qualidade de Sistemas, cada qual com seu supervisor; e a equipe de infraestrutura e serviços possui uma célula responsável pelo apoio nos serviços de redes, banco de dados e comunicações, além de possuir uma subdivisão especializada em serviços de TI, tais como suporte técnico ao usuário final (central de serviços, manutenção de equipamentos, entre outros) e apoio administrativo (licenciamento de software, administração de bens, entre outros). A Figura 3 apresenta o organograma da estrutura organizacional.

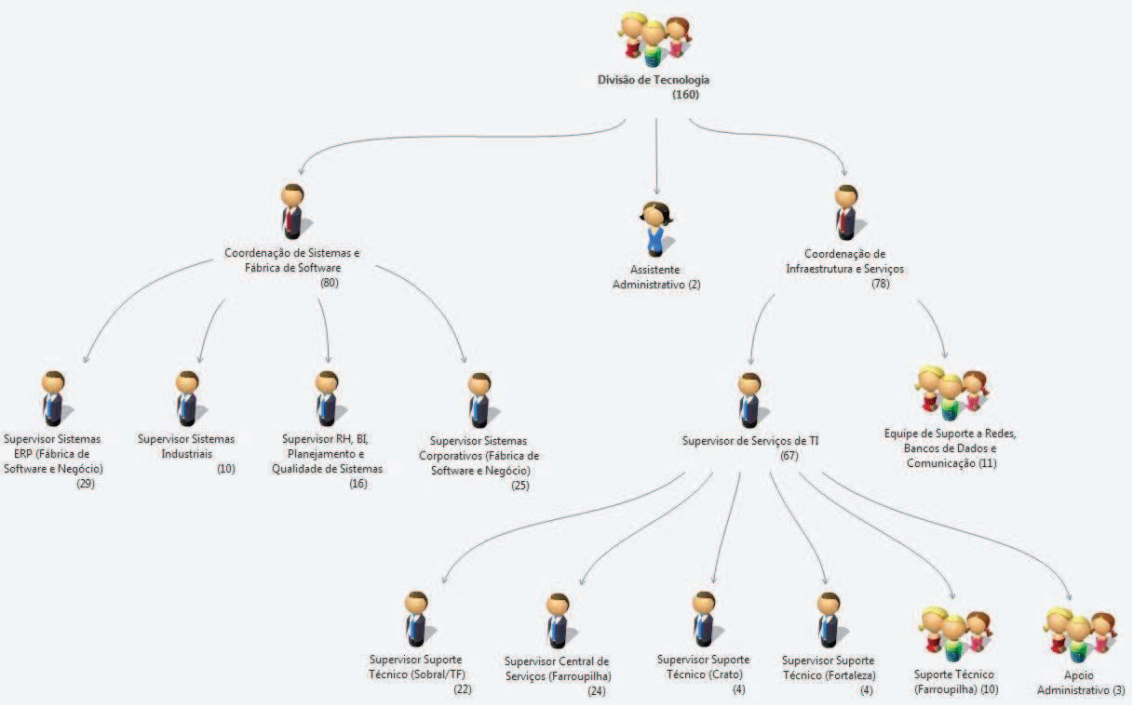

Figura 3. Estrutura organizacional

Na área de infraestrutura estão alocados 78 colaboradores, 40 deles atuam como suporte técnico especializado, 8 colaboradores são atuantes como analistas de redes, ambiente e comunicação e 3 colaboradores são DBA (*database analyst*). Além disso, a infraestrutura conta, ainda, com 24 colaboradores compondo a central de serviços (CS). A finalidade da CS é atuar como suporte primário aos usuários (considerado atendimentos nível 0 e 1) e atua no formato 24/7, disponibilizando suporte 24 horas por dia, 7 dias por semana.

Na área de sistemas, 64 colaboradores estão alocados em Fábricas de Software (FS). As FS são compostas por profissionais atuantes como desenvolvedores, analistas de sistema e analistas de negócio. Atualmente as FS provêm a manutenção e desenvolvimento de sistemas web, sistemas corporativos, ERP e sistemas industriais. Além disso, a equipe de sistemas conta 16 colaboradores que constituem células especializadas em *Business Intelligence* (BI), escritório de projetos (EP) e Qualidade de Software (SQA).

A equipe de Qualidade de Software atualmente é composta por 3 programadores de teste e 1 analista de qualidade de software. A equipe é responsável pela garantia da qualidade sobre a grande maioria dos sistemas da empresa, entretanto, atualmente, seu principal foco é atuar sobre o sistema ERP. A SQA é responsável pelo desenvolvimento e execução dos testes automatizados para o sistema ERP da empresa.

A equipe ERP é composta por 29 colaboradores, onde 10 são analistas de negócio, 3 são analistas de sistema líderes, 4 são analistas de sistemas e os demais são programadores de sistema.

O estudo do presente trabalho foi aplicado, principalmente, na equipe SQA. Entretanto, por possuírem integração entre os seus processos, mudanças também são propostas e replicadas para a equipe de desenvolvimento ERP, que também compõe a unidade de análise.

4.2. O processo de solicitação de criação e execução de testes automatizados

Os testes automatizados desenvolvidos pela SQA são, em sua maioria, projetados para executar testes unitários, de regressão e funcionais sobre o sistema ERP. Portanto, a equipe de qualidade trabalha sob demanda gerada pela FS-ERP.

Existe um grande macroprocesso mapeado (Figura 4) que é subdividido em outros dois subprocessos: criação de teste e execução de teste.

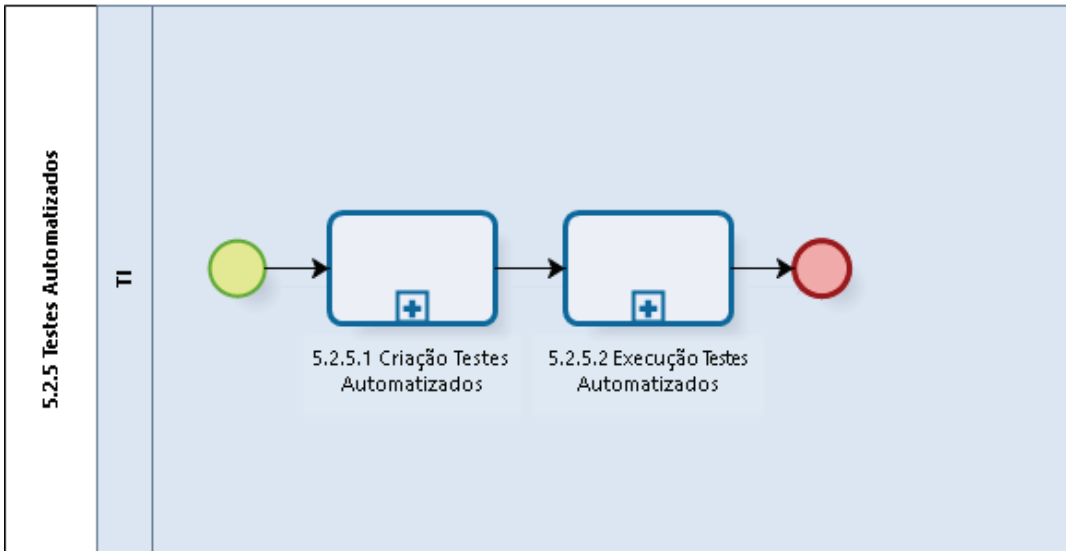

Figura 4. Macroprocesso Equipe SQA

4.2.1. Criação de Testes Automatizados

A solicitação de criação de um teste automatizado (Figura 5) surge a partir da necessidade encontrada pelo analista de negócio da equipe FS-ERP. Por ter contato direto com o usuário final, o analista de negócio é quem conhece os processos e pontos críticos do sistema, o que o qualifica para a tarefa de levantar a necessidade da criação do cenário de teste.

Ao identificar sua necessidade, o analista de negócio encaminha a solicitação de novo teste para a equipe SQA. Essa solicitação não possui uma formalização, podendo ser realizada via e-mail e/ou pessoalmente. Basta apenas que seja encaminhado o

processo e passo-a-passo que o teste automatizado deve executar. Esse material geralmente é produzido em formato vídeo, através da gravação de tela enquanto o analista de negócio simula o uso do sistema.

Ao receber o material, o analista de qualidade de software analisa o(s) cenário(s) de teste solicitado(s) e proposto(s), verificando se possui a estrutura técnica adequada para a criação do teste como, por exemplo, a existência de um ambiente que comporta o caso de teste ou se é necessária a parametrização de um novo ambiente. Ao receber essa demanda, ela é registrada e inserida no *backlog*, as atividades são repriorizadas e encaminhadas para o desenvolvimento.

A equipe SQA utiliza o Kanban como ferramenta de apoio para o planejamento e acompanhamento das atividades e a priorização dos testes a serem desenvolvidos ocorre semanalmente, onde é realizada uma reunião de planejamento semanal. Quando uma solicitação de demanda é aprovada, o teste é encaminhado para o *backlog* para que o programador de testes possa iniciar o seu desenvolvimento.

Após desenvolvido, o teste passa por uma etapa de homologação, onde ele será executado e produzirá logs de auditoria, que contém informações sobre o resultado da execução do programa. Esse log é submetido para avaliação junto ao analista de negócio que solicitou a criação do teste. Se o analista considera que o log está correto, gera-se a *baseline* deste log, o teste é inserido na ferramenta de integração contínua, e disponibilizado em produção, ou seja, fica disponível para execuções. Caso o analista de negócio considere que o log não esta correto e o teste não produz o resultado esperado, o teste retorna para a etapa de desenvolvimento para ajustes, até que ele esteja de acordo com o esperado para entrar em produção.

Por fim, os códigos-fonte são versionados e o caso de teste é documentado na ferramenta Enterprise Architect, onde é informada uma breve descrição sobre o que o teste faz em cada cenário e quais são os programas envolvidos.

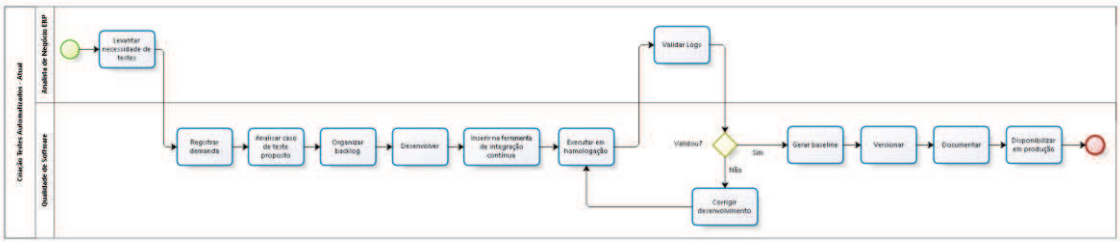

Figura 5. Processo de Solicitação de Criação de Testes Automatizados

4.2.2 Execução de Testes Automatizados

A execução dos testes automatizados acontece de forma dinâmica, através do conceito de integração contínua. Para compreender o processo de execução dos testes, é importante conhecer o processo de entrega de software da equipe ERP.

As atualizações de software são entregues para o usuário final em períodos quinzenais, denominados “Cópia de Quarentena”. A “quarentena” é um repositório destinado a receber todos os códigos-fonte que serão enviados para produção. Mantendo-os centralizados, é mais fácil de aplicar uma bateria de testes a fim de evitar, ou pelo menos reduzir, a quantidade de erros em produção.

A equipe SQA monitora esse diretório, identificando quando os arquivos são movimentados. Ao identificar quais programas devem ser testados, a ferramenta de integração contínua inicia a execução dos testes automatizados relacionados.

A integração entre o processo de desenvolvimento da equipe ERP e o processo de execução de testes da equipe SQA é ilustrado graficamente na Figura 6, a fim de facilitar a compreensão do seu funcionamento.

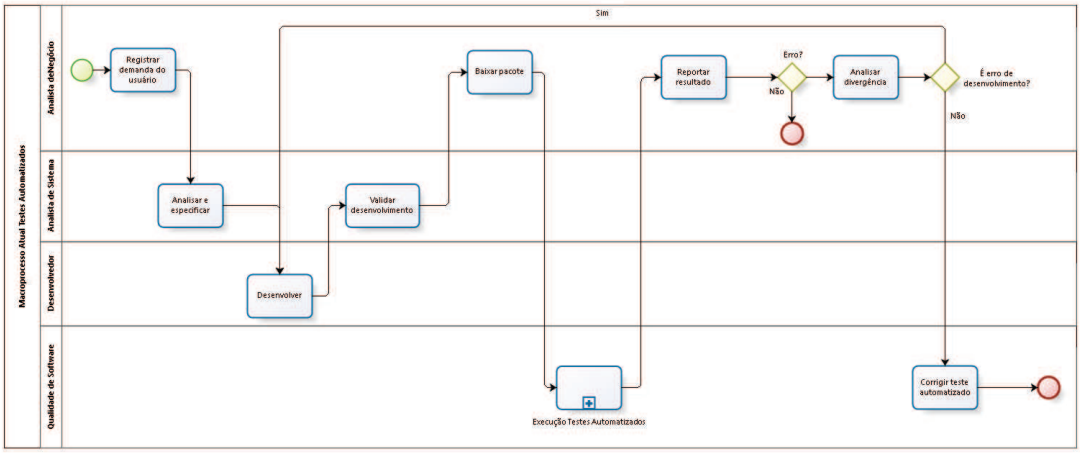

Figura 6. Mapeamento do processo atual de desenvolvimento integrado à execução de testes automatizados

O processo de desenvolvimento do software inicia através do registro de demanda do analista de negócio. Essa demanda é analisada pelo analista de sistemas que a direciona para seu *backlog* e a encaminha para desenvolvimento. Após desenvolvida, a alteração passa pela sua primeira validação, que são os testes manuais executados pelo analista de sistema. Se ele considerar que o software atende aos requisitos, ele autoriza que essa versão seja inserida no ambiente de "quarentena".

O processo de execução dos testes (Figura 7) inicia, portanto, na identificação dos códigos-fontes alterados e seus testes relacionados. Esses testes são automaticamente executados pela ferramenta de integração contínua. Após a sua execução, todo teste produz um log, documento que contém todas as movimentações efetuadas em banco de dados enquanto o teste rodou. Nesse ponto do processo, o log produzido pelo teste é comparado com o log de *baseline*. Caso seja encontrada alguma divergência, essa é reportada para o analista de negócio responsável pelo programa (quem solicitou a criação do teste em questão). Esse reporte é feito via e-mail e toda divergência encontrada é documentada. O teste recebe, então, o status de reprovado. Caso a comparação entre os logs não apresente divergências, o teste é considerado aprovado e seu resultado é igualmente documentado.

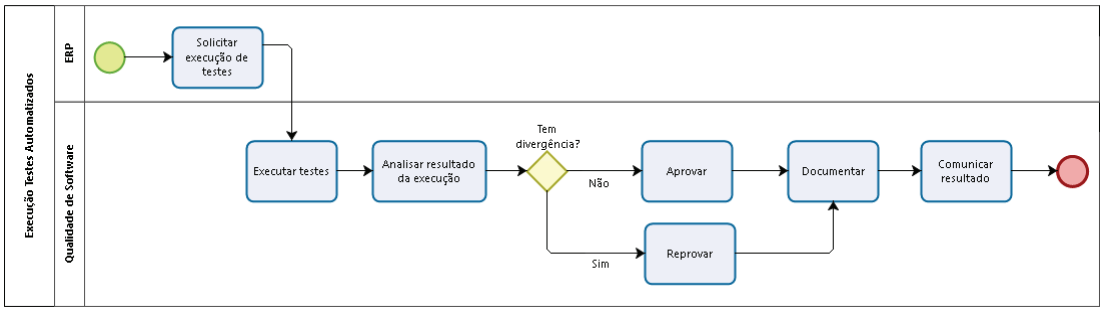

Figura 7. Processo de Execução de Testes Automatizados

Nem todo reporte de divergência é um erro ou inconsistência propriamente dito. Eventualmente, quando algum programa da suíte de testes é alterado e o teste automatizado não é ajustado para essa nova realidade, ele apresentará inconsistências no log de execução. Cabe ao analista de negócio, após receber o reporte, identificar se a divergência é um erro ou não. Caso seja um erro, a equipe de desenvolvimento procede com as correções necessárias sobre o software. Caso não seja um erro, a equipe SQA corrige o teste automatizado (quando necessário) e gera uma nova *baseline*.

Da forma como o macroprocesso está modelado hoje, não existe um fluxo que garanta que ao ser alterado um programa (e/ou processo, quando o teste for de regressão) o teste relacionado será analisado a fim de se validar se ele ainda atende ao programa ou não. Quando o caso de teste abrange apenas um programa, é rápido identificar a alteração e proceder com o ajuste no teste. Contudo, quando se trata de testes de regressão, a tarefa de mapear todos os casos de testes que contém o programa a fim de ajustá-los pode não ser feita, o que provocará o reporte de uma divergência, quando, na verdade, o resultado já devia estar atualizado em *baseline*.

4.3. Coleta preliminar dos dados

A coleta preliminar dos dados buscou identificar no processo atual de desenvolvimento de software e execução de testes quais etapas desse processo oferecem oportunidade de melhoria para garantir que os testes se mantenham sempre atualizados em relação aos processos e programas do sistema, a fim de aumentar a sua cobertura e garantir a qualidade do produto de software. Para atingir esse objetivo, a coleta aconteceu em duas etapas.

A primeira etapa da coleta de dados buscou contabilizar a quantidade de testes automatizados desatualizados em relação ao volume total de testes que se encontram em produção. No período de novembro de 2016 a abril de 2017, a equipe ERP passou pelo processo de migração de versão de sistema. Em meio a esse projeto, a equipe SQA atuou entre os dias 01 de março à 15 de abril de 2017 através da execução de todos os 338 testes automatizados produzidos sobre o sistema ERP. A finalidade dessa atividade era executar o teste e verificar se o log produzido era igual à *baseline*. Em caso de divergência, era realizada uma análise junto ao analista de negócio a fim de identificar se a diferença representava um erro de sistema ou se era necessário gerar uma nova *baseline*. Além disso, foram encontradas situações onde havia alteração de layout de tela e/ou alteração de programas que compunham um processo e que implicavam em ajustes na programação do teste. Para ambos os casos, foi considerado que o teste estava

desatualizado. Ao fim, contabilizou-se 77 testes automatizados que estavam desatualizados e necessitaram de ajustes e geração de *baseline*.

A análise sobre o primeiro resultado obtido demonstrou que o fato de o processo atual da equipe SQA não possuir formalizações para alterações nos testes faz com que, ao longo do tempo, alguns testes sejam esquecidos e não acompanhem a evolução dos seus programas. Além disso, identificou-se uma falha na integração entre os processos de desenvolvimento de software e execução de testes, pois, atualmente, o fluxo não garante que todos os testes que estão relacionados com um programa serão executados e atualizados antes que esse programa seja entregue em produção. Uma das principais consequências de executar um teste desatualizado é o reporte de falso-positivo para os analistas de negócio.

Posteriormente, na segunda etapa elaborou-se, como instrumento de pesquisa, um questionário on-line através de uma ferramenta de pesquisa corporativa oficial, denominada SPS. O questionário foi composto por 12 questões fechadas, objetivas, com preenchimento obrigatório e foi organizado em duas etapas: a primeira contempla questões referentes ao perfil do respondente e a segunda contém questões para o levantamento dos pontos críticos sobre o processo de execução de testes inserido no processo de desenvolvimento de software.

Após concluir o desenvolvimento do questionário, o mesmo passou pela validação do gerente do setor de desenvolvimento da organização, a fim de aprovação, identificação de ajustes e melhorias. Ao término das correções e aprovação do gerente, o questionário foi disponibilizado através da ferramenta oficial de pesquisa da corporação.

O público alvo escolhido para a aplicação do questionário contemplou os analistas de negócio da equipe ERP. A escolha se deu porque, atualmente, eles são os responsáveis por identificar a necessidade de um teste para seus programas. Assim, o analista de negócio é o responsável por homologar o resultado de execução do teste para geração de *baseline*. Logo, é quem recebe os reportes de divergência e possui a tarefa de analisar se o resultado é esperado ou se é um erro.

Portanto, a escolha por esse público para a aplicação do questionário se justifica pelo fato de que eles são diretamente impactados sobre os resultados oriundos das execuções dos testes.

A pesquisa foi então aplicada a um grupo de 10 participantes e ficou disponível entre os dias 27 de abril e 04 de maio de 2017, retornando 9 respostas. As primeiras questões tratavam da caracterização da amostra. Na sequência, as demais questões buscaram indagar os analistas de negócio sobre suas percepções quanto ao atual processo de execução de testes automatizados e o reporte dos resultados.

A análise dos dados permitiu identificar pontos fracos na atual integração entre os processos de desenvolvimento de software e execução dos testes automatizados, que resultam na perda do vínculo entre teste automatizado e o programa relacionado, ocasionando a vulnerabilidade na manutenção e atualização dos testes, o que resulta na possibilidade de erros passarem despercebidos e chegarem em produção, comprometendo, assim, o grau de confiabilidade sobre o processo de execução de testes. Diante disso, constatou-se a necessidade de propor alterações no processo a fim de garantir a qualidade da cobertura dos testes criados e também comprovou-se a

percepção da pesquisadora quanto à necessidade da implantação de um novo processo que possa atrelar as *baselines* de teste aos seus respectivos programas.

4.4. Proposta de gestão de solicitação de alteração para testes automatizados

A partir das conclusões obtidas sobre a análise dos dados, iniciou-se a modelagem para as atividades que compõem o processo de atualização de testes automatizados. O processo proposto busca aplicar práticas de gerência de configuração, tais como integridade da configuração (através do versionamento) e controle e auditoria da configuração, que abrangem a documentação e a própria formalização do processo de solicitação de alteração, a fim de garantir o controle sobre os itens alterados, assegurando a correta vinculação entre teste e programa na atividade de geração da *baseline*.

Nessa proposta, é sugerida a execução da atividade de controle de configuração através da implementação e utilização de um formulário para a solicitação de alteração do teste automatizado (Apêndice B), a fim de documentar os motivos que geraram essa demanda e as alterações que serão realizadas. A atividade de balanço da configuração ocorre quando novas *baselines* são geradas. Nesse ponto é proposto que essa atividade também seja documentada, onde deve ser informado a qual teste a *baseline* ela se refere, quem solicitou a sua geração/atualização, qual programador de teste a atualizou e qual o motivo da atualização (Apêndice C). Com a finalidade de garantir a integridade da configuração, é sugerido, que o processo mantenha o uso de ferramenta de integração contínua e o versionamento dos códigos-fonte através de ferramenta de controle de versão e armazenamento em repositórios. Assim, é possível armazenar um histórico de alterações e possibilita o rastreamento das alterações dos itens de configuração. Por fim, a atividade de auditoria da configuração, ocorre antes da geração da *baseline*, onde em conjunto com o analista de negócio os dados que comporão a nova *baseline* são validados.

Em virtude do fato de a equipe SQA trabalhar em parceria com a equipe de desenvolvimento ERP, ao propor um processo formal de gestão de solicitação de alteração para teste, o processo de desenvolvimento também é impactado. Portanto, também foi elaborada uma proposta de fluxo de trabalho para a equipe ERP que busca aproximar os testes automatizados do processo de desenvolvimento de software propriamente dito.

Através de uma visão macro, a Figura 8 ilustra a nova proposta de processo para integração da execução de testes automatizados ao processo de desenvolvimento.

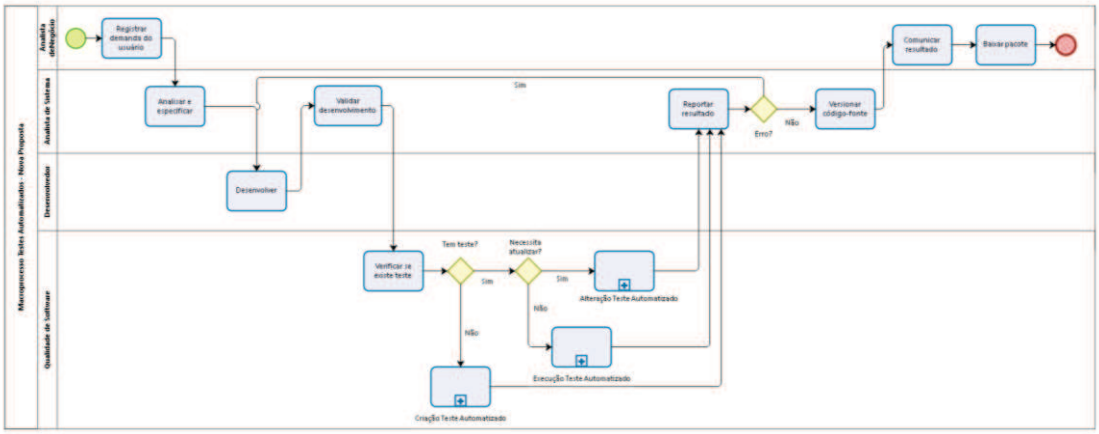

Figura 8. Nova proposta de fluxo de trabalho integrado

Nesse fluxo de trabalho, a execução dos testes automatizados acontecerá durante a etapa de desenvolvimento, buscando garantir que as *baselines* de testes acompanharão a evolução dos programas aos quais elas estão vinculadas. Ou seja, ao alterar o programa, o teste automatizado vinculado será avaliado a fim de identificar a necessidade de atualizá-lo. Caso essa necessidade seja presente, a equipe SQA seguirá o fluxo proposto na Figura 9, onde o teste automatizado será readequado e uma nova *baseline* é gerada.

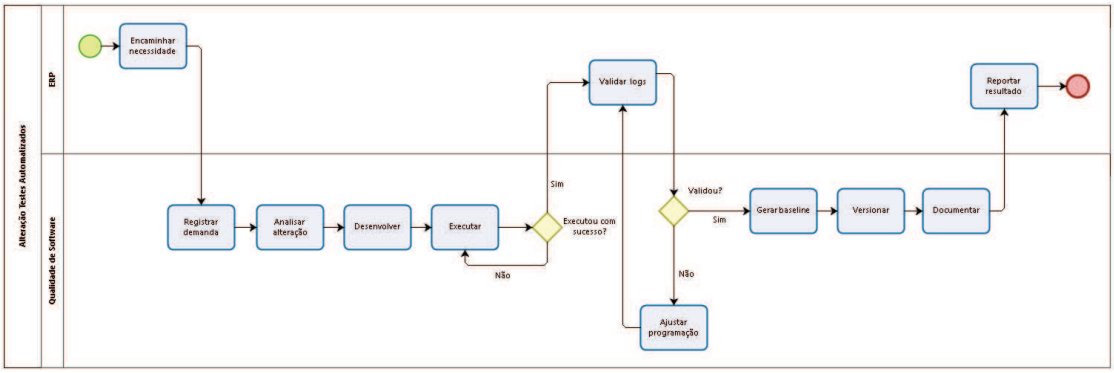

Figura 9. Processo formal de alteração para testes automatizados

Após essa etapa, o resultado da execução do teste será reportado diretamente ao analista de sistemas, que dará continuidade ao seu habitual fluxo, que consiste em versionar os códigos-fonte e encaminhar para o ambiente "quarentena", onde o analista de negócios efetuará testes pontuais. Ao operar dessa forma, é possível minimizar a quantidade de testes automatizados desatualizados em produção.

O macroprocesso de gestão de alteração do teste automatizado que envolve ambas equipes (ERP e SQA) foi mapeado com o auxílio dos analistas de sistema líderes e validado e aprovado pelo gestor imediato da equipe ERP. Já o processo de alteração para teste automatizado, mais específico e que compete apenas à equipe SQA, foi mapeado pela pesquisadora, que atua como Analista de Qualidade de Software e foi validado e aprovado pelos programadores de teste e gestor imediato.

4.5 Aplicação do processo de gestão formal de alteração para testes automatizados

O processo proposto foi aplicado em três períodos de preparação para atualização de versão no cliente. Esses períodos são quinzenais e são denominados "cópia de quarentena". Portanto, o processo foi utilizado entre os dias 29 de maio e 7 de julho de 2017. Ao longo desse período, reuniões semanais foram realizadas junto aos analistas de sistema e analistas líderes a fim de validar o processo, coletar percepções e identificar oportunidades de melhoria no fluxo proposto.

Na primeira cópia de quarentena, compreendida no período entre 29 de maio e 10 de junho de 2017, foram executados 113 testes automatizados, onde foi identificada a necessidade de alterar a estrutura de 2 testes, em virtude de novas implementações nos programas relacionados; e 6 *baselines* foram atualizadas.

Na segunda cópia de quarentena, compreendida entre os dias 12 de junho e 24 de junho de 2017, 111 testes foram executados. A partir dessas execuções, 3 testes precisaram ser alterados para atender à alterações de programas e processos, e 3 *baselines* foram atualizadas.

Na terceira, e última, cópia de quarentena observada, durante o período de 26 de junho à 08 de julho foram executados 124 testes automatizados, onde 5 novas *baselines* foram geradas e nenhum teste apresentou a necessidade atualização em sua codificação.

5. Análise dos dados

Nesta seção são apresentados os resultados obtidos pela organização após a implantação do processo de gestão formal de alteração para testes automatizados. Além disso, são também apresentados os resultados pré implantação do fluxo de trabalho proposto.

5.1. Análise e Resultados pré-proposta de processo

A análise dos resultados foi iniciada observando-se a percepção dos analistas de negócio sobre o atual processo de execução de testes automatizados e o respectivo reporte dos resultados. Buscou-se, então, conhecer o perfil deste público, que atualmente são as pessoas que lidam diretamente com o reporte dos testes automatizados (recebem o reporte de teste, efetuam análise sobre a divergência apontada e retornam um posicionamento).

Os respondentes são, em sua maioria, formados na área de informática (representando 80% do volume total). Apenas 11,11% do público respondente está na empresa a menos de 3 anos. Portanto, 88,89% já trabalham em parceria com a equipe SQA e seus processos a 3 anos ou mais. No que diz respeito ao processo atual de execução, reporte e alteração dos testes automatizados, 77,78% deles consideram que o processo é simples e de fácil entendimento. Entretanto, embora a maioria considere o processo simples, 55,56% dos analistas de negócio reconhecem que não informam à equipe SQA a necessidade de alteração/atualização do teste automatizado quando algum programa da suíte de testes é alterado. Em função disso, 88,89% deles já foram acionados para analisar divergências quando na verdade, o teste estava desatualizado e necessitava ser refeito. Como consequência da falta de atualização dos testes, 22,22% deles encontraram problemas em produção, embora o teste tenha executado com sucesso.

Um teste é dito executado com sucesso quando ele consegue, automaticamente, cumprir todas as etapas que foram codificadas, sem erro de execução. Entretanto,

embora o teste tenha chego até o final, isso não implica que a execução tenha produzido o resultado esperado. Diante disso, se a execução produziu o resultado esperado, diz-se que o teste está aprovado. Do contrário, caso haja alguma divergência, essa é submetida a análise pelo analista de negócio. Se a divergência corresponde a um erro, considera-se que o teste está reprovado. Senão, caso o analista sinalize que é uma divergência esperada, o teste é considerado aprovado.

Neste cenário, em 22,22% dos casos os testes executaram com sucesso e produziram os resultados esperados em relação à *baseline*. Contudo, ao inserir o programa em produção, esse apresentou problemas, resultando em retrabalho. Essa discrepância pode ser atribuída ao fato de o teste estar desatualizado em relação à realidade do usuário, uma vez que a função principal do teste automatizado é simular o uso do programa como um usuário final.

Em complemento ao questionário, realizaram-se análises e execuções de todos os testes que encontram-se ativos em produção, buscando identificar quantos deles estavam desatualizados, seja em suas *baselines*, ou mesmo na forma como os cenários de testes foram construídos. Dessa atividade, concluiu-se que, do total de 338 testes, 77 encontravam-se desatualizados, o que representa uma fatia de aproximadamente 23% de testes comprometidos, conforme ilustrado no Gráfico 1.

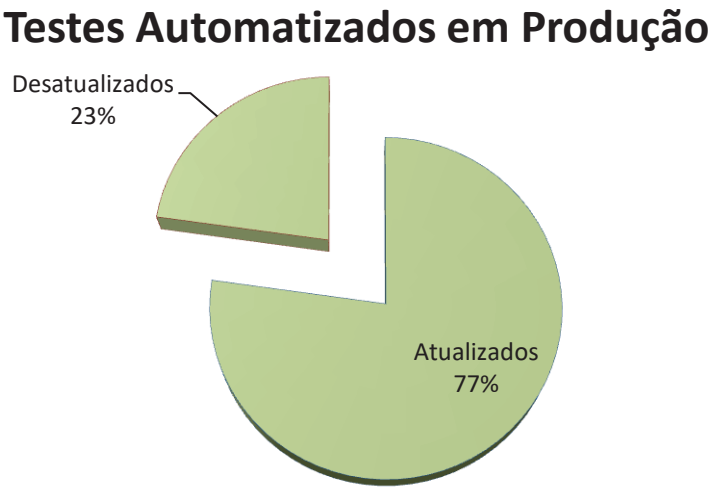

Gráfico 1. Percentual de testes automatizados desatualizados

5.2. Apresentação e análise dos resultados pós-proposta de processo

A análise de resultados obtidos pelo estudo contemplou o período de 29 de maio a 08 de julho de 2017. Durante esse período, foram realizadas contagens quinzenais da quantidade de testes executados e da quantidade de testes que tiveram suas *baselines* atualizadas e/ou necessitaram de alteração em sua programação. Esses dados são apresentados na Tabela 1.

Tabela 1. Contagem de testes que foram atualizados em relação à quantidade de testes executados

Período	Quantidade de testes executados	Quantidade de *baselines* atualizadas	Quantidade de testes alterados
Quarentena 1	113	6	2
Quarentena 2	111	3	3
Quarentena 3	124	5	0
Total		14	5

Ao observar a Tabela 1, somando os totais de *baselines* geradas e testes alterados (atualizados), é possível verificar que 19 testes foram atualizados ainda em fase de desenvolvimento, representando 5,6% do volume total de testes. Através do processo de trabalho proposto, foi possível vincular as *baselines* de testes automatizados com os seus respectivos programas porque os processos de execução dos testes automatizados foram introduzidos durante o processo de desenvolvimento. Assim, o programa alterado não chegará em produção sem antes ter sido testado. Caso um teste automatizado não consiga ser executado por estar desatualizado, há a oportunidade de alterar esse teste a fim de que ele possa reproduzir e simular o funcionamento real do sistema. Caso o teste consiga ser executado até o fim, porém o resultado de sua execução possui comportamento diferente da *baseline*, cabe análise desse resultado a fim de averiguar se isso é um erro de programação, ou se é uma divergência esperada e há necessidade de gerar nova *baseline*. Dessa forma, pode-se garantir que o teste atende ao processo atual e evita-se que erros saiam da programação e cheguem até o analista de negócio e ambiente de produção.

A adoção das práticas de gerenciamento de configuração, através da execução de suas atividades, assegura que o processo aconteça de forma natural e correta. As atividades de gerenciamento de configuração aplicadas foram de controle da configuração, através do uso do formulário de solicitação formal para os testes automatizados, cujo o objetivo é documentar a solicitação da alteração, informando o que deve ser alterado e justificando o porquê da alteração. A atividade de auditoria da configuração ocorre através de acesso e consulta ao formulário, onde é possível auditar se a alteração realizada atende ao que foi solicitado e nada além disso foi alterado, garantindo consistência da atividade de balanço de configuração. Em complemento, a documentação do balanço de configuração visa documentar as alterações realizadas nas *baselines*, informando a data e o motivo pelo qual essas foram geradas, auxiliando no controle da configuração. Por fim, a atividade de integridade da configuração é executada cada vez que um código-fonte de teste é submetido ao versionamento no repositório oficial e sua inclusão é efetuada na ferramenta de integração contínua. Essa atividade acontece com o intuito de preservar um histórico de alterações, possibilitando o armazenamento seguro e rastreamento de todas as alterações realizadas sobre os itens de configuração (códigos-fonte, nesse caso).

Introduzir o processo de execução de testes automatizados no processo de desenvolvimento e agregar a ele as atividades de gerenciamento de configuração proporciona maior controle sobre os resultados, o que permite que um programa e seus testes relacionados não entrem em produção sem que suas alterações tenham sido

desenvolvidas e validadas, reforçando, assim, a garantia da qualidade do produto de software.

6. Considerações Finais

Esse trabalho teve como objetivo principal proporcionar a melhoria da qualidade dos produtos de software do setor de TI de uma corporação, respondendo a questão de pesquisa: Como práticas de gerência de configuração podem melhorar os processos de criação e alteração para testes automatizados no setor de TI de uma corporação?

Para que isso fosse possível, desenvolveu-se um ciclo de gestão formal para solicitação e alteração para testes automatizados utilizando atividades de gerência de configuração, como controle da configuração, balanço da configuração e auditoria da configuração. Através de estudos aprofundados sobre a gerência de configuração e suas atividades e como aplicá-las às atividades de testes, obteve-se embasamento teórico que auxiliou como aporte na implantação do processo de gestão formal sobre as alterações e solicitações de testes automatizados na organização. A implantação desse processo foi avaliada entre os dias 29 de maio e 08 de julho de 2017, o que corresponde a 3 cópias de "Quarentena" (períodos quinzenais de atualização de versão do sistema para o usuário final).

Pode-se concluir que a aplicação das práticas de gerência de configuração agregadas ao processo proposto sobre os testes automatizados permitiu que esses fossem atualizados em paralelo à atividade de desenvolvimento do software, o que garantiu que, ao liberar a versão do software, o teste automatizado correspondente estava atendendo ao processo e resultados esperados. Portanto, com o passar do tempo, essa prática tende a reduzir a quantidade de testes automatizados desatualizados em produção.

Convém ressaltar, entretanto, perante os resultados obtidos após a implantação do processo de gestão formal de testes automatizados uma limitação quanto à influência da pesquisadora, uma vez que a mesma está incluída no ambiente de estudo, podendo não identificar alguma situação existente. Além disso, cabe salientar que a pesquisa contempla apenas duas equipes como amostra (SQA e ERP) e, além disso, dado seu curto tempo de aplicação, os dados não podem ser generalizados. A partir dos dados levantados nesse trabalho, a organização pode institucionalizar o processo formal de solicitação e alteração para testes automatizados para demais equipes do setor de desenvolvimento da TI. Além disso, a empresa pode adotar o processo proposto institucionalizando-o como processo padrão para as equipes de desenvolvimento. Vale ressaltar que o processo ainda está em fase inicial, necessitando de ajustes para contemplar todo o setor.

Como contribuição para a área de conhecimento, esse trabalho apresenta o processo de gestão formal de solicitação e alteração para testes automatizados, servindo como orientação para outros estudos ou organizações que buscam a melhoria contínua dos seus processos de desenvolvimento de produtos de software. Como trabalhos futuros, sugere-se a criação de indicadores que meçam o retrabalho no desenvolvimento de software pré e pós implantação do modelo de gestão formal dos testes automatizados, a fim de contabilizar a redução de custo nos projetos de desenvolvimento da organização, visando comprovar financeiramente o beneficio da atividade de formalizar o processo de alteração para testes para que esses sempre estejam atualizados. Além

disso, sugere-se como continuidade ao trabalho a contabilização do retrabalho que recai sobre a equipe de Qualidade de Software da organização em questão quando são encontradas situações de testes automatizados desatualizados em produção, a fim de verificar qual o impacto sobre a garantia da qualidade do produto de software que está sendo entregue pelo desenvolvimento.

Por fim, conclui-se que as práticas de gerência de configuração aplicadas sobre o processo de solicitação e alteração para testes automatizados propicia a melhora na qualidade dos processos de testes automatizados aliados ao desenvolvimento do software, respondendo à questão de pesquisa deste trabalho através dos resultados obtidos neste estudo.

Referências

ABNT (1996), Gestão da Qualidade - Diretrizes para a Gestão da Configuração. NBR ISO 10007. ABNT.

Beizer, B. (1990), Software Testing Techniques. Van Nostrand Reinhold Company, New York, 2nd edition.

Bernardo, P. C. (2011), Padrões de testes automatizados. Tese de Doutorado. Universidade de São Paulo.

Bersoff, E.H. et al. (1980), Software configuration management, an investment in product integrity. Prentice-Hall.

Bersoff, E.H. (1984), Elements Of Software Configuration Management. IEEE Transactions on Software Engineering, v. 10, n. 1, p. 79-87, janeiro de 1984.

Bliss, M. (1993), Software configuration management - delivering quality software products. Information Systems Management, v. 10, n. 3, p. 35-46, summer.

Buckle, J. K. (1982), Software configuration management. The MacMillan Press Ltd.

Burrows, C., Dart. S., George G.W. (1996), Ovum Evaluates Configuration Management Tools, London, U.K, Ovum Press.

Capretz, M. A .M. (1992), A software maintenance method based on the software configuration management discipline. Tese de Doutorado. University of Durham, Inglaterra.

Costa, M. G. (2006), Estratégia de automação em testes: requisitos, arquitetura e acompanhamento de sua implantação. 116p. Dissertação (mestrado profissional) - Universidade Estadual de Campinas, Instituto de Computação, Campinas, SP.

Crespo, A. N. et al. (2004), Uma metodologia para teste de Software no Contexto da Melhoria de Processo. Simpósio Brasileiro de Qualidade de Software, p. 271-285.

Estublier, J. (2000), “Software Configuration Management: A Roadmap”, Proceedings of the conference on The future of Software engineering, p. 279-289, June 4-11, Limerick, Ireland

Estublier, J. et al. (2002), Impact of the research community on the field of software configuration management: summary of an impact project report, ACM SIGSOFT Software Engineering Notes, v.27 n.5, September

Fantinato, M. et al. (2005), AutoTest–Um framework reutilizável para a automação de teste funcional de software. Cad. CPqD Tecnologia, v. 1, n. 1, p. 119-131.

Fewster, M.; Graham, D. (1999), Software Test Automation. Addison-Wesley Professional.

Figueiredo, S.; Santos, G.; Rocha, A. R. (2004), Gerência de Configuração em Ambientes de Desenvolvimento de Software Orientados a Organização. Simpósio Brasileiro de Qualidade de Software, Brasília.

Gil, A. C. (2010), Como Elaborar Projetos de Pesquisa. 5ª Edição.

Hetzel, W. (1973), Program test methods. Prentice Hall.

Howden, W. E. (1987), Software Engineering and Technology: Functional Program Testing and Analysis. McGrall-Hill Book Co, New York.

Maldonado, J. C. (1991), Critérios Potenciais Usos: Uma Contribuição ao Teste Estrutural de Software. PhD thesis, DCA/FEE/UNICAMP, Campinas, SP, July.

Maldonado, J. C. et al. (2004), Introdução ao Teste de Software-versão 2004-01. São Carlos: ICMC/USP.

Mei, H., Zhang, L., Yang, F. (2001), A Software Configuration Management Model for Supporting Component-Based Software Development. Software Engineering Notes, p. 53-58, vol. 26, no. 2, ACM Press., New York, NY

Myers, G. (1979), The art of software testing. John Wiley and Sons.

Nunes, V. B. (2005), Integrando Gerência de Configuração de Software, Documentação e Gerência de Conhecimento em um Ambiente de Desenvolvimento de Software, Dissertação de mestrado. Universidade Federal do Espírito Santo. Vitória.

Perry, D. E.; Kaiser, G. E. (1990), Adequate testing and object-oriented programming. Journal on Object-Oriented Programming, 2(5):13–19, January/February.

Petroski, B. M. (2009), Geração automática de casos de teste automatizados no contexto de uma suíte de testes em telefones celulares.

Pressman, R. S. (2001), Software engineering - a practitioner's approach. 5th. ed. McGrawHill.

Pressman, R. S. (2005). Software engineering: a practitioner's approach. Palgrave Macmillan.

Sanches R. et al. (2001), Gerência de Configuração. In: Qualidade de Software: Teoria e Prática. Prentice Hall.

Sommerville, I. (2011), Engenharia de Software, 9ª Edição.

Vergara, S. C. (2007), Projetos e relatórios de pesquisa em Administração. 9ª Edição.

Villas-Boas, A. L de C. (2003), Gestão de configuração para teste de software. Dissertação (mestrado) Universidade Estadual de Campinas, Faculdade de Engenharia Elétrica e de Computação. Campinas.

Whitgifh, D. (1991), "Methods and tools for software configuration management", Nova York, NY, John Wiley & Sons.

APÊNDICE A – Coleta de Dados Preliminar: Questionário Aplicado

Pesquisa: QUESTIONÁRIO SOBRE EXECUÇÃO E REPORTE DE TESTES AUTOMATIZADOS

Validade: 27/04/2017 a 04/05/2017

Descrição: Esta pesquisa está relacionada ao curso de especialização em Qualidade de Software, da Universidade do Vale do Rio dos Sinos (Unisinos), sob orientação do professor André Luiz de Castro Villas-Boas (UNICAMP).

Objetivo Geral da pesquisa: avaliar a percepção dos pesquisados sobre o atual processo de construção, execução e reporte de resultados dos testes automatizados referentes ao sistema ERP e, a partir disso, implantar um processo de solicitação e reporte de teste automatizado que atenda às necessidades. A pesquisa visa identificar as falhas no processo atual de construção e atualização dos testes automatizados, que vem a impactar no reporte final, e por isso foi desenvolvido um questionário onde são observados alguns pontos tais como: frequência de reporte de erro, frequência de reporte de falsos positivos e frequência de atualização dos testes já existentes.

Público-alvo: Analistas de negócio.

Este questionário está divido em duas etapas: Etapa 1: Perfil do respondente. Etapa 2: Levantamento de pontos críticos referente à execução dos testes automatizados.

As informações coletadas serão utilizadas somente para esta pesquisa.

É garantido o anonimato ao respondente.

O tempo estimado para responder o questionário é de 10 minutos.

Etapa 1: Perfil do Respondente

1. Nível de formação

a) Ensino Médio Completo

b) Ensino Superior Incompleto

c) Ensino Superior Completo

d) Pós-graduação ou Especialização

2. Tempo de Empresa

a) Menos de 3 anos

b) Entre 3 e 5 anos

c) Mais de 5 anos

3. Área de Formação

a) Informática

b) Outra área

Etapa 2: Levantamento de pontos críticos referente à execução dos testes automatizados

4. O processo de solicitação de alteração/criação de um teste automatizado é simples e claro?

a) Sim

b) Não

5. Você costuma solicitar a atualização do seu teste quando algum programa da suíte é alterado?

a) Sim

b) Não

6. Com que frequência você recebe reporte de erros através da execução de testes automatizados?

a) Diariamente

b) Semanalmente

c) Mensalmente

d) Anualmente

7. Em uma escala mensal, quantas vezes você é acionado para analisar divergências?

a) Menos de 3 vezes

b) De 3 a 5 vezes

c) Mais de 5 vezes

8. Você já recebeu reporte de erro, quando na verdade o teste estava desatualizado?

a) Sim

b) Não

9. O reporte de falsos-positivos tem sido frequente?

a) Sim

b) Não

10. Você considera confiável o reporte de erros do teste automatizado que você recebe?

a) Sim

b) Não

11. Você já encontrou erros em produção, embora o teste tenha sido aprovado?

a) Sim

b) Não

12. O erro em produção não foi identificado pelo teste porque ele estava desatualizado?

a) Sim

b) Não

APÊNDICE B – Formulário de Solicitação de Criação e/ou Alteração para Testes Automatizados

Formulário de Solicitação de Criação/Alteração para Teste Automatizado		
Documento destinado a auxiliar os analistas na elaboração de casos de teste.		
Versão do documento: 1.0	**Data solicitação:**	**Centro de Custo:**
Caso de Teste:		
Responsável:		
Cenário de teste:		
Programas:		
Pré-condições:		
Pós-condições:		
Instruções	**Resultado esperado**	

APÊNDICE C – Documentação de geração de *baseline*

Ciclo de teste / Semana Cópia Quarentena	Caso de Teste	Data	Programador Responsável	Motivo