

MEC-SETEC
INSTITUTO FEDERAL DE MINAS GERAIS – Campus Formiga
Curso de Ciência da Computação

**UM LEVANTAMENTO SOBRE A UTILIZAÇÃO DE TESTES DE *SOFTWARE* EM
EMPRESAS DE MICRO E PEQUENO PORTE NO CENTRO-OESTE MINEIRO**

Marcela das Graças Fonseca

Orientador: Profa. Dra. Paloma Maira de Oliveira
Lima

FORMIGA – MG
2018

MARCELA DAS GRAÇAS FONSECA

**UM LEVANTAMENTO SOBRE A UTILIZAÇÃO DE TESTES DE *SOFTWARE* EM
EMPRESAS DE MICRO E PEQUENO PORTE NO CENTRO-OESTE MINEIRO**

Trabalho de Conclusão de Curso apresentado ao Instituto Federal de Minas Gerais – Campus Formiga, como requisito parcial para a obtenção do título de Bacharel em Ciência da Computação.

Orientador: Profa. Dra. Paloma Maira de Oliveira Lima.

004 Fonseca, Marcela das Graças.
Um levantamento sobre a utilização de testes de software em
empresas de micro e pequeno porte no Centro-oeste mineiro / Marcela das
Graças Fonseca. -- Formiga : IFMG, 2018.
49p.. : il.

Orientador: Prof. Dr. Paloma Maira de Oliveira Lima
Trabalho de Conclusão de Curso – Instituto Federal de Educação,
Ciência e Tecnologia de Minas Gerais – *Campus* Formiga.

1. Testes de software. 2. Ferramentas de teste. 3. Testes
automatizados. 4. Testes manuais. 5. Tamanho das empresas. I. Título.

CDD 004

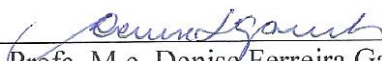
Marcela das Graças Fonseca

“Um levantamento sobre a utilização de testes de software em empresas de micro e pequeno porte no Centro-oeste mineiro”

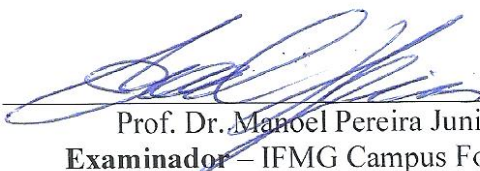
Trabalho de Conclusão de Curso apresentado ao Instituto Federal Minas Gerais – Campus Formiga, como requisito parcial para a obtenção do título de Bacharel em Ciência da Computação.



Prof. Dra. Paloma Maira de Oliveira Lima
Orientadora – IFMG Campus Formiga



Prof. M.e. Denise Ferreira Garcia Rezende
Examinador – IFMG Campus Formiga



Prof. Dr. Manoel Pereira Junior
Examinador – IFMG Campus Formiga

Formiga/MG, 21 de Novembro de 2018

Agradecimentos

Primeiramente, quero agradecer a Deus pela minha vida e a de cada uma das pessoas que foram cruciais para que eu chegasse até aqui. Agradeço por cada oportunidade e pela força que Ele me deu para superar todas as dificuldades. Agradeço, também, à Nossa Senhora pela interseção e cuidado.

Agradeço aos meus pais, Efigenia e Sebastião, que não mediram esforços para me ajudar a alcançar meus objetivos e me apoiaram em cada decisão. À minha irmã, Gabriela, por estar sempre comigo, pela compreensão, amizade e por me ajudar tantas vezes. Aos demais familiares e aos meus amigos pela torcida e incentivo de sempre.

Agradeço ao Túlio, meu namorado e amigo, pela paciência, pelo amor e companheirismo nessa jornada. À minha amiga Renata, que me acompanha desde o Ensino Médio e foi muito importante para a realização deste sonho. À Ana Paula, Flávia e Guilherme que também foram essenciais para fortalecer a minha caminhada. Aos demais colegas de turma que, cada um a seu modo, marcaram esse ciclo.

Por fim, agradeço aos meus professores e a todos os servidores do IFMG – Campus Formiga que se dedicam todos os dias para oferecer um ensino de qualidade. À minha orientadora, Paloma, por ter aceito com carinho o convite para me auxiliar na concretização dessa etapa e por toda sua solicitude. Aos membros da banca, Profa. Denise e Prof. Manoel, que se dispuseram com consideração a participar deste momento e contribuir com a evolução deste trabalho.

“Por vezes sentimos que aquilo que fazemos não é senão uma gota de água no mar. Mas o mar seria menor se lhe faltasse uma gota”.

Madre Teresa de Calcutá

RESUMO

Este documento apresenta um estudo quantitativo relativo à aplicação de testes de *software* em empresas de desenvolvimento de sistemas de micro e pequeno porte da região Centro-oeste de Minas Gerais. O objetivo geral do trabalho é conhecer um pouco mais sobre como as empresas da região trabalham com testes de *software*. Para a coleta dos dados, foi elaborado um questionário *online*, do qual 20 empresas participaram efetivamente. Os resultados mostraram que o objetivo principal das empresas pesquisadas com a aplicação de testes é obter um produto final com maior qualidade. Destaca-se também que o processo manual de testes é predominante nas empresas da região se comparado com testes automatizados. Foi observado que os erros mais comuns encontrados no processo de testes são erros relacionados à interpretação dos requisitos, funcionalidade e codificação. Finalmente, foram elencadas as vantagens da aplicação de testes de *software* sob a perspectiva do empresário.

Palavras-chave: Testes de *software*. Ferramentas de teste. Testes automatizados.

LISTA DE ILUSTRAÇÕES

Figura 1 – Forma de execução dos testes nos <i>softwares</i> desenvolvidos pelas empresas	29
Figura 2 – Forma de execução dos testes relacionada ao tempo de atuação das empresas no mercado	30
Figura 3 – Porcentagem de tempo gasto com os testes de <i>software</i> pelas empresas que consideraram a limitação de tempo para a atividade de teste como dificuldade	32
Figura 4 – Percentual de utilização de ferramentas automatizadas entre empresas de micro e pequeno porte	33
Figura 5 – Percentual de microempresas que utilizam ferramentas automatizadas e porcentagem do tempo total do projeto gasto com os testes de <i>software</i> pelas mesmas	33
Figura 6 – Erros mais comuns de serem encontrados nos <i>softwares</i> desenvolvidos.....	34
Figura 7 – Porcentagem de empresas nas quais a atividade de teste é realizada exclusivamente pelos desenvolvedores do sistema ou pela equipe de testes	35

LISTA DE TABELAS

Tabela 1 – Variáveis analisadas pelo estudo e descrição das mesmas.	26
Tabela 2 – Motivos para as empresas não adotarem a prática automatizada de testes.....	31
Tabela 3 – Responsáveis por realizar os testes nos <i>softwares</i> desenvolvidos pelas empresas versus porcentagem de empresas nas quais eles atuam.	36
Tabela 4 – Motivos para as empresas aplicarem testes automatizados.	36
Tabela 5 – Vantagens dos testes automatizados de <i>software</i> em comparação aos testes manuais versus porcentagem de empresas que consideraram a vantagem.	37

LISTA DE ABREVIATURAS

IEEE	<i>Institute of Electrical and Electronics Engineers</i> (Instituto de Engenheiros Eletricistas e Eletrônicos)
IFMG	Instituto Federal de Minas Gerais
SEBRAE	Serviço Brasileiro de Apoio às Micro e Pequenas Empresas
TCC	Trabalho de Conclusão de Curso

SUMÁRIO

1 INTRODUÇÃO.....	14
1.1 Justificativa	15
1.2 Objetivos	15
2 FUNDAMENTAÇÃO TEÓRICA	16
2.1 Tipos de testes de <i>software</i>	18
2.1.1 Teste de unidade.....	18
2.2.2 Teste de integração.....	21
2.2.3 Teste de sistema	23
3 MATERIAIS E MÉTODOS.....	25
3.1 Seleção das empresas	25
3.2 Elaboração do questionário.....	25
3.3 Variáveis analisadas.....	26
4 ANÁLISE DOS RESULTADOS	29
4.1 Forma de aplicação dos testes.....	29
4.2 Algumas limitações da automação.....	30
4.3 Dificuldades dos testes de <i>software</i>	32
4.4 Tempo gasto com a atividade de testes.....	32
4.5 Utilização de ferramentas para os testes	32
4.6 Principais erros encontrados pelos testes.....	33
4.7 Responsáveis pelos testes de <i>software</i>	34
4.8 Vantagens dos testes automatizados em relação aos manuais	36
4.9 Ameaças à validade.....	38
5 CONSIDERAÇÕES FINAIS	39
6 REFERÊNCIAS BIBLIOGRÁFICAS	40
APÊNDICE A – QUESTIONÁRIO SOBRE O USO DE TESTES DE <i>SOFTWARE</i>	43

1 INTRODUÇÃO

No mundo atual, é possível perceber a presença de *softwares* nas mais variadas áreas e atividades do dia a dia do ser humano. A opção por automatizar as mais diversas tarefas, tem trazido mais agilidade à vida das pessoas. Entretanto, para que essa agilidade seja de fato concretizada, os *softwares* precisam ser eficientes, ágeis e, principalmente, não conter erros que prejudiquem sua performance, ou seja, precisam ter qualidade [PFLEEGER, 2004].

Neste contexto de qualidade de *software*, os testes são de suma importância, já que, com o auxílio dos mesmos, é possível identificar potenciais erros/falhas/*bugs* que passaram despercebidos durante a fase de desenvolvimento e que poderão ter consequências impactantes no desempenho do *software* [LANZA e MARINESCU, 2006]. Atualmente, várias empresas de desenvolvimento de sistemas já reconhecem a fase de testes como uma etapa importante na construção de um *software*. Os testes colaboram fortemente para verificar se a aplicação está funcionando corretamente e se atende aos requisitos especificados. Vale ressaltar que a atividade de testar um programa é repetitiva, demanda tempo e trabalho intensivo e está propensa a erros humanos [MALDONADO et. al, 2007]. Ademais, “empresas gastam muito tempo e dinheiro na manutenção de *software*, investimento que seria economizado, caso um processo de teste de *software* fosse bem implantado e conduzido em paralelo com o processo de desenvolvimento.” [VIDAL, 2011].

Os testes de *software* podem ser manuais ou automatizados. Existem técnicas automatizadas diversificadas que podem ser aplicadas em diferentes momentos e de variadas formas para validar os aspectos principais do *software*. Segundo SAP (2011), conforme citado por Malanovicz et al. (2014), “Testes Automáticos são aqueles executados por ferramentas automatizadas, sendo cada resultado armazenado automaticamente pela ferramenta de teste, e podendo reutilizar os Casos de Teste em outros planos de testes”. A grande vantagem da automação está em minimizar o tempo gasto com os testes e na possibilidade de refazê-los diversas vezes, tendo em vista que, realizada a correção de um erro no sistema, é preciso refazer os testes anteriores e realizar novos, o que não gera um desgaste para a ferramenta que está realizando os mesmos, ao contrário do que aconteceria se estivessem sendo feitos de forma manual [PRESSMAN, 1995].

O SEBRAE (2016) classifica o porte dos estabelecimentos considerando a quantidade de pessoas ocupadas nos mesmos. Para este trabalho, as empresas selecionadas são do setor de comércio e serviços, sendo assim, considera-se microempresa aquela que possui até 9 pessoas

ocupadas; pequena empresa de 10 a 49 pessoas ocupadas; média empresa de 50 a 99 pessoas ocupadas e grande empresa com 100 ou mais pessoas ocupadas [SEBRAE, 2016]. Não foram incluídas empresas de médio e grande porte na pesquisa, pois não foram levantadas empresas com tais classificações na região avaliada.

1.1 Justificativa

Este trabalho de conclusão de curso (TCC) se justifica em dois âmbitos distintos: para a comunidade de analistas e testadores de *software* e para o crescimento pessoal do acadêmico. Como explanado anteriormente, os testes de *software* são cruciais para garantir que o produto final possua qualidade. Conforme levantado, muitas empresas ainda trabalham apenas com a aplicação de testes manuais e existe uma carência de conhecimento e utilização das ferramentas de testes automatizados. Assim, este trabalho se justifica, pois, visa estudar as vantagens da aplicação de testes automatizados de *software*.

Espera-se que o estudo auxilie a comunidade de analistas e testadores de *software* e até mesmo as empresas a perceber como se encontra o cenário da aplicação de testes de *software* na região, quem são os responsáveis por aplicá-los e os benefícios de utilizar-se testes automatizados ou manuais de acordo com a demanda do contexto em que estão inseridos.

Como crescimento pessoal, este trabalho se justifica visto que o tema foi escolhido para, aliando-se aos conhecimentos obtidos ao decorrer do curso, ser um objeto de estudo para o autor, que poderá aprofundar-se no tema, visando um diferencial para adentrar ao mercado de trabalho.

1.2 Objetivos

O objetivo geral deste trabalho é verificar a aplicação de testes de *software* em empresas de micro e pequeno porte do Centro-oeste mineiro. Além disso, descobrir as principais causas que levam essas empresas a aplicá-los ou não.

Para alcançar o objetivo geral, foram propostos os seguintes objetivos específicos:

- Elencar empresas de micro e pequeno porte do Centro-oeste mineiro;
- Estudar sobre técnicas e ferramentas de testes de *software*;
- Elaborar um questionário para ser enviado às empresas;
- Analisar os resultados obtidos através das respostas ao questionário.

2 FUNDAMENTAÇÃO TEÓRICA

De acordo com Myers et al. [2004 apud DELAMARO, JÚNIOR e ANDRADE, 2017], a atividade de teste de *software* é um processo, ou uma série de processos, cuja função é executar um sistema com a intenção de encontrar possíveis erros. Segundo o padrão IEEE 610.12-1990, teste de *software* consiste de uma atividade que submete um componente – ou um sistema – a condições específicas, sendo os resultados observados, ou até mesmo gravados, e, posteriormente, avaliados seguindo determinados aspectos do elemento sob teste. A norma IEEE 829-2008 define o termo como:

1. Conjunto de um ou vários casos de teste, i.e., diversas entradas de teste, condições de execução e resultados almejados, determinados para alcançar um requisito específico;
2. Conjunto de um ou vários procedimentos de teste, i.e., instruções da sequência de ações, como configuração, realização e avaliação de resultados, para a execução de determinado caso de teste;
3. Conjunto de um ou vários procedimentos e casos de teste.

De acordo com Pressman [1995 apud Myers, 1990], existem algumas regras que podem ser definidas como objetivos dos testes, são elas: a atividade de teste é o procedimento de execução de um programa com a finalidade de encontrar erros; um caso de teste é adequado se possui alta probabilidade de apontar um erro ainda não revelado; um teste obtém êxito se demonstra erros ainda não evidenciados.

Pfleeger (2004) afirma que o objetivo dos testes é desvendar defeitos e existem diversas formas de torná-los mais efetivos e eficientes em relação aos esforços dedicados. Pressman (1995) destaca que “a atividade de teste de *software* é um elemento crítico da garantia de qualidade de *software* [...] e o destaque crescente do *software* como elemento de sistema e os 'custos' envolvidos associados às falhas de *software* são forças propulsoras para uma atividade de teste cuidadosa e bem planejada.”.

Sommerville (2011) afirma que, nos testes manuais, “um testador executa o programa com alguns dados de teste e compara os resultados com suas expectativas; ele anota e reporta as discrepâncias aos desenvolvedores do programa.”. Já nos testes automatizados, Sommerville (2011) destaca que “os testes são codificados em um programa que é executado cada vez que o

sistema em desenvolvimento é testado.”, sendo que essa técnica de testes é, geralmente, executada mais rapidamente que a primeira, em especial quando envolve testes de regressão¹.

Segundo Fewster e Graham (1999), conforme citado por Bernardo (2011), “Testes automatizados são programas ou *scripts* simples que exercitam funcionalidades do sistema em teste e fazem verificações automáticas nos efeitos colaterais obtidos.”. Bernardo (2011) afirma que, estando os testes de *softwares* independentes de intervenção humana, os benefícios do computador podem ser aproveitados. Entre esses, o autor destaca a fiel reproduzibilidade de um conjunto de ações, possibilidade de executar testes em paralelo e a velocidade dessa execução, o volume e o momento das execuções são flexíveis e há facilidade de criação de casos de testes mais complexos.

Bernardo (2011) afirma que a tarefa de realizar testes manuais de um caso de teste é efetiva e rápida, mas a repetição de um grande conjunto de testes desse tipo se torna cansativa. Não raramente, os testadores de *software* podem optar por priorizar casos de testes considerados mais críticos e por não verificarem novamente todos os casos a cada alteração de significância no código. A partir disso, vários erros podem surgir, trazendo prejuízos para os desenvolvedores que acabarão perdendo tempo para encontrar e corrigir erros. Além de que, os infortúnios atingem também o cliente, devido ao atraso na entrega, que pode ser de um *software* de qualidade comprometida [SOMMERVILLE, 2006].

Mas o aspecto mais crítico deste cenário é o efeito "bola de neve". Como é necessário muito esforço para executar todo o conjunto de testes manuais, dificilmente a bateria inteira de testes é executada novamente a cada correção de um erro, como seria desejável. Muitas vezes, a correção de uma falha pode adicionar erros de regressão que são defeitos adicionados em módulos do sistema que estavam funcionando corretamente mas que foram danificados por alguma manutenção desastrada. A tendência é esse ciclo se repetir até que a manutenção do sistema se torne uma tarefa tão custosa que passa a valer a pena reconstruí-lo completamente [BERNARDO, 2011].

De acordo com Pressman (1995), a verificação faz referência a um conjunto de tarefas que garantem a implementação correta de determinada função do *software*. Já a validação, trata-

¹ Sommerville (2011) define o termo como “reexecução de testes anteriores para verificar se as alterações no programa não introduziram novos *bugs*.”.

se de um grupo de tarefas que atestam que o *software* foi criado e pode ser encontrado através dos requisitos especificados pelo cliente.

Delamaro, Júnior e Andrade (2017) definem que “a atividade de teste é comumente dividida em etapas, cada fase do teste de *software* possui objetivos distintos. Nesse contexto, cada etapa da fase de teste apresenta características únicas e serve para guiar o processo de teste de *software*.”. As fases compreendidas nos testes são apresentadas nas subseções seguintes, com base em Delamaro et al. [2007 apud Delamaro, Júnior e Andrade, 2017].

2.1 Tipos de testes de *software*

2.1.1 Teste de unidade

Sommerville (2011) define que “o teste unitário é o processo de testar os componentes do programa, como métodos ou classes de objetos. As funções individuais ou métodos são o tipo mais simples de componente.” Nessa técnica, ao invés de se testar todo o programa, concentra-se na realização de testes nos menores blocos que o constituem. A finalidade do teste de módulo – outra denominação para o termo – é comparar se, de fato, o funcionamento do módulo codificado corresponde ao especificado [MYERS, SANDLER e BADGETT, 2011]. Os testes de unidade buscam ocasionar falhas relacionadas a uma implementação incorreta de algoritmos ou estruturas de dados errôneas [DELAMARO, JINO e MALDONADO, 2007]. Pressman (1995) define testes de unidade como segue:

O teste de unidades concentra-se no esforço de verificação da menor unidade de projeto de *software* – o módulo. Usando a descrição do projeto detalhado como guia, caminhos de controle importantes são testados para descobrirem erros dentro das fronteiras do módulo. A complexidade relativa dos testes e os erros detectados como resultado deles são limitados pelo campo de ação restrito estabelecido para o teste de unidade. O teste de unidade baseia-se sempre na caixa branca, e esse passo pode ser realizado em paralelo para múltiplos módulos [PRESSMAN, 1995].

Testes de unidades consistem em testes de caixa-branca [MYERS, SANDLER e BADGETT, 2011], i.e., “[...] um minucioso exame dos detalhes procedimentais. Os caminhos lógicos através do *software* são testados, fornecendo-se casos de teste que põem à prova conjuntos específicos de condições e/ou laços” [PRESSMAN, 1995]. Myers, Sandler e Badgett (2011) destacam que esse tipo de teste é mais viável quando se testa partes menores do sistema,

como é feito nos testes unitários, já que estes se concentram em encontrar erros associados à lógica de programação.

Normalmente, esses testes são realizados em conjunto com o processo de codificação do *software*. Quando o desenvolvedor termina de escrever o código, revisando-o para eliminar possíveis erros sintáticos, os casos de teste são elaborados [SOMMERVILLE, 2011]. Os testes podem ser feitos pelos próprios programadores [SOMMERVILLE, 2011], mas, em alguns processos de desenvolvimento, são definidos pares de programadores/testadores [CUSAMANO e SELBY, apud SOMMERVILLE, 2011].

Myers, Sandler e Badgett (2011), indicam três motivações para a realização de testes de unidade:

1. Tendo o foco voltado para os menores componentes do programa, essa técnica se torna uma forma de administrar os casos de teste;
2. O processo de depuração torna-se mais fácil, já que se tem conhecimento do módulo no qual o erro se encontra;
3. Teste de unidade institui o paralelismo no processo de teste, i.e., o teste de diversos módulos simultaneamente.

Pressman (1995) destaca os testes que intercorrem como parte da execução do teste de unidade, os quais são listados a seguir com seus respectivos objetivos:

- O teste da interface com o módulo tem como objetivo garantir que as informações passam adequadamente para dentro e para fora da unidade testada;
- O teste da estrutura de dados local visa a garantia de que houve digitação consistente, que os nomes das variáveis estão bem redigidos, não ocorre *underflow* e/ou *overflow*, são tratadas exceções de endereçamento e que a iniciação de variáveis e os tipos de dados estão corretos, sendo que dados armazenados de forma temporária terão sua integridade preservada;
- As condições de limite são submetidas a teste para assegurar que a unidade testada opera nos limites estabelecidos de forma pertinente;
- O teste dos caminhos independentes através da estrutura de controle é feito para certificar que todas as instruções pertencentes àquele módulo são executadas ao menos uma vez;
- Os caminhos de tratamento de erros são testados para garantir que a descrição do erro é compreensível e diz respeito ao erro de fato encontrado e que oferece

informações para a localização do mesmo e ainda que não acontece intervenção no sistema antes que o erro seja verificado e devidamente tratado.

Segundo Sommerville (2011), os casos de teste são efetivos se mostram que um elemento usado de maneira correta realiza o que é esperado dele e se revelam defeitos, caso os mesmos existam, nos componentes. Define-se, assim, dois tipos de casos de teste a serem elaborados. O primeiro consiste em exprimir o funcionamento adequado dos componentes do programa. Já o segundo deve ser inspirado nas experiências do testador em ações que acarretam erros naquele tipo de elemento. Entradas incomuns devem ser aderidas com a finalidade de mostrar que essas não ocasionam defeitos nos componentes [SOMMERVILLE, 2011]. Sommerville (2011) indica duas técnicas para a escolha de casos de teste unitário: teste de partição e testes baseados em diretrizes.

Pressman (1995) afirma que a análise da especificação do *software* pode fornecer um bom direcionamento para a geração de casos de teste que sejam capazes de identificar erros em determinadas categorias, as quais seguem listadas:

1. Comparação de dados de diferentes tipos ou comparações incorretas;
2. Resultado de igualdade quando a mesma é improvável devido a erros de precisão;
3. Precedência de operadores, operadores lógicos ou variável incorretos;
4. Fim de laço de repetição inexistente ou inapropriado ou variáveis controladoras dos mesmos modificadas impropriamente;
5. Falha para sair quando encontrada iteração divergente.

Myers, Sandler e Badgett (2011) definem que, para criar casos de teste unitário, é preciso analisar a lógica da unidade a ser testada utilizando um ou mais métodos de teste de caixa branca e complementá-los aplicando métodos de teste de caixa preta à especificação da unidade. Uma alternativa para executar os casos de teste é a elaboração de métodos ou funções que atuam como ponto de partida para a execução do programa, geralmente denominados *main*, nos quais possui uma chamada ao método/função a ser testado e comandos de impressão de valores na tela, permitindo que o testador verifique as saídas geradas e compare-as com as esperadas [BERNARDO, 2011]. A tarefa de teste tende a ser demorada e trabalhosa, o que demanda uma seleção de casos de teste efetivos [SOMMERVILLE, 2011].

2.2.2 Teste de integração

Pressman (1995) apresenta o teste de integração como:

[...] uma técnica sistemática para a construção da estrutura de programa, realizando-se, ao mesmo tempo, testes para descobrir erros associados a interfaces. O objetivo é, a partir dos módulos testados no nível de unidade, construir a estrutura de programa que foi determinada pelo projeto [PRESSMAN, 1995].

Bernardo (2011) explica o conceito como “[...] uma denominação ampla que representa a busca de erros de relacionamento entre quaisquer módulos de um *software*, incluindo desde a integração de pequenas unidades até a integração de bibliotecas das quais um sistema depende, servidores e gerenciadores de banco de dados.” O teste de integração consiste em verificar se a especificação do programa é cumprida quando os componentes do sistema interagem [PFLEEGER, 2004 apud LIMA, 2004]. Mesmo que uma unidade do programa esteja totalmente correta, não se pode afirmar que os módulos/funções pelos quais ela será chamada também estejam [BERNARDO, 2011].

Pressman (1995) apresenta duas formas de integração do sistema: a incremental e a não-incremental e destaca as vantagens da primeira em relação à segunda. Para ele, a integração incremental permite que os erros sejam separados e corrigidos mais facilmente, visto que consiste na construção e no teste de pequenas partes do programa, enquanto na abordagem não-incremental o programa é testado como um todo depois de estar completo, o que resulta em um conjunto de erros difíceis de serem corrigidos. A integração incremental possui duas abordagens: a *top-down* e a *bottom-up* [PRESSMAN, 1995]. A integração *top-down* consiste em integrar os módulos em um fluxo de cima para baixo considerando-se a hierarquia de controle. Tendo a unidade de controle principal como ponto de partida, as demais unidades dependentes dela e sucessivamente dependentes são inseridas na integração de forma que primeiro seja considerada a largura da estrutura em construção ou a profundidade da mesma [PRESSMAN, 1995].

Pressman (1995) define cinco passos para o processo de integração, os quais seguem:

1. A unidade de controle principal é utilizada como um *driver* de teste e os *stubs* são substituídos pelos módulos diretamente dependentes ao de controle principal;

2. Os *stubs* dependentes são substituídos pelas respectivas unidades, dependendo da maneira de incorporação à integração selecionada, um por vez;
3. A cada vez que uma unidade é integrada, realiza-se testes na integração;
4. A cada vez que um conjunto de testes é concluído, substitui-se um *stub* pela unidade a qual ele representa;
5. Pode-se realizar teste de regressão para garantir que não foram incorporados novos erros.

Nessa abordagem de integração, tem-se a possibilidade de identificar pontos de decisão ou controle antecipadamente. Caso haja problemas significativos na estrutura de controle, é essencial que esses sejam revelados o quanto antes seja possível [PRESSMAN, 1995]. De acordo com Pressman (1995), podem surgir problemas de ordem logística durante essa abordagem como, por exemplo, a solicitação de processamento em níveis baixos da hierarquia, para a realização de um teste satisfatório das camadas acima, que ainda não tenham tido seus módulos reais incorporados. Uma das vantagens dessa abordagem está em testar antecipadamente as unidades de controle do sistema. Entretanto, encontra-se uma grande desvantagem na necessidade de *stubs*, os quais demandam tempo para sua construção, e nas dificuldades de teste relacionadas a eles [PRESSMAN, 1995].

Já a integração *bottom-up* equivale à inserção de módulos em um fluxo de baixo para cima, dispensando-se os *stubs*, já que as unidades de nível mais baixo do sistema já serão incorporadas, não sendo preciso simulá-las. Nessa abordagem, o conceito de *driver* é utilizado mais vezes, especialmente nas camadas mais inferiores. À medida que se desloca a integração para cima, a presença de *drives* vai diminuindo sua necessidade [PRESSMAN, 1995]. Nessa abordagem, tem-se a vantagem de o projeto de casos de teste ser mais fácil que na *top-down*, aliada à ausência de *stubs* [PRESSMAN, 1995]. Entretanto, “o programa não existe como entidade até que o último módulo seja adicionado.” [MYERS, 1979 apud PRESSMAN, 1995].

Pressman (1995) define os passos para esse tipo de integração como:

1. As unidades de baixo nível formam *clusters*, executando uma sub função específica do sistema;
2. Um *driver* é construído para controlar os casos de teste, sendo responsável pela entrada e saída dos mesmos;
3. O *cluster* formado é testado;

4. O *driver* é removido e os *clusters* são unidos ao passa-se para uma camada superior da estrutura.

O analista de *software* precisa identificar as unidades críticas ao realizar o teste de integração para testá-las o mais breve possível. Essas unidades têm pelo menos uma das seguintes características: engloba diversos requisitos de *software*, possui alto nível de controle, é propensa a erros ou possui requisitos de desempenho [PRESSMAN, 1995].

2.2.3 Teste de sistema

Um *software* não possui processamento exclusivo em um computador, ele é um membro de um sistema completo, composto por *hardware* e outros *softwares*. Quando um *software* é desenvolvido e incorporado a esse conjunto, uma sucessão de testes de integração e validação precisa ser feita [PRESSMAN, 1995]. Alguns comportamentos do sistema só podem ser visualizados quando as funcionalidades são integradas, podendo ocorrer de não terem sido planejados [SOMMERVILLE, 2011]. Para constatar essas reações à integração do *software*, Sommerville (2011) afirma que “é preciso desenvolver testes que verifiquem se o sistema está fazendo apenas o que ele supostamente deve fazer.”

De acordo com Sommerville (2011), “o teste de sistema verifica se os componentes são compatíveis, se interagem corretamente e transferem os dados certos no momento certo, por suas interfaces. [...] Deve centrar-se em testar as interações entre os componentes e objetos que compõem um sistema.”. O padrão IEEE 610.12-1990 apresenta o conceito como testes feitos em um sistema que se encontra completo e integrado, tendo por objetivo verificar se o mesmo age de acordo com os requisitos documentados. Por fim, a finalidade do teste de sistema é garantir que o desempenho do *software*, em conjunto com os demais e com o *hardware*, seja bem-sucedido [DELAMARO, JINO e MALDONADO, 2007]. Pressman (1995) afirma que as fases de teste realizadas durante o desenvolvimento do *software* aumentam significativamente as chances de que o sistema tenha um bom funcionamento com suas partes integradas.

A principal técnica empregada na fase de teste de sistema é a funcional, na qual o sistema é considerado uma caixa-preta, o que significa que nenhum detalhe de implementação é levado em conta e a análise do *software* é realizada como se o mesmo estivesse sendo manipulado pelo usuário [DELAMARO, JINO e MALDONADO, 2007]. Essa fase de teste é realizada coletivamente, sendo que, em algumas empresas, considera-se a possibilidade de delegar uma equipe que não inclua projetistas e programadores [SOMMERVILLE, 2011] como

feito nas etapas abordadas anteriormente. Pressman (1995) aborda os tipos de testes de sistema, definidos por Beizer (1984), os quais seguem listados a seguir:

- Teste de recuperação: tem como objetivo provocar falhas no *software* para verificar se o mesmo realiza a recuperação de forma satisfatória;
- Teste de segurança: visa constatar que as técnicas de proteção anexadas ao *software* serão capazes de preservá-lo de acessos indevidos;
- Teste de estresse: consiste na execução do *software* requisitando recursos em proporções anormais se comparadas às que o mesmo solicitará em seu funcionamento;
- Teste de desempenho: o objetivo é testar a performance de *run-time* do *software* integrado ao sistema. Esse teste pode ser combinado com o anterior e também exigir a medição da utilização dos recursos de *hardware*, levando à descoberta de circunstâncias que ocasionam degradação e falhas no sistema.

3 MATERIAIS E MÉTODOS

Esta seção apresenta a descrição das etapas compreendidas no estudo realizado com o objetivo de informar como o trabalho foi conduzido, bem como os critérios para elaboração do questionário, a ferramenta utilizada para criá-lo e coletar as respostas recebidas e as variáveis submetidas à análise.

3.1 Seleção das empresas

A fim de analisar a utilização de testes de *softwares*, definiu-se a região Centro-oeste do estado de Minas Gerais como área de estudo. Para selecionar as empresas participantes, foram realizadas pesquisas por empresas de desenvolvimento de sistemas nos municípios da região, coletando-se dados relevantes, como a quantidade de funcionários, o nome fantasia e a razão social da empresa, telefone de contato, endereço de *e-mail*, nome do responsável, dentre outros. Após a seleção de 40 empresas, foi observado que essas se tratavam de empresas de micro e pequeno porte. Esse fator motivou delimitar o contexto do trabalho para empresas desse tamanho (até quarenta e nove funcionários).

3.2 Elaboração do questionário

Foi elaborado um questionário *online* com dezesseis perguntas relacionadas, principalmente, a testes de *software*. O questionário é apresentado no Apêndice A. Tais perguntas foram formuladas a partir da leitura de artigos acadêmicos sobre o assunto e com base nas experiências dos autores deste projeto. O foco principal durante a elaboração das questões foi conseguir coletar informações que exibissem o cenário da aplicação de testes de *software* sob a ótica das empresas selecionadas, além de relatarem a visão das mesmas sobre as vantagens advindas da execução de testes.

O questionário foi criado usando a ferramenta Formulários Google, a qual é uma ferramenta *online* e gratuita que permite a coleta e organização de informações, seja em pequena ou em grande quantidade, além de dar suporte ao compartilhamento dos formulários criados, possibilitando que outra pessoa autorizada possa editar. As respostas recebidas por meio do formulário são coletadas automaticamente e organizadas pela ferramenta, que transforma as informações em gráficos, permitindo melhor visualização das mesmas [GOOGLE DOCUMENTATION, 2018].

Um aspecto que recebeu bastante atenção na elaboração do formulário foi o tempo de resposta do questionário. Procurou-se ser o mais objetivo possível, sem deixar de perguntar o necessário, para que o tempo gasto pelos entrevistados pudesse ser otimizado e, assim, fosse possível conseguir uma quantidade considerável de respostas.

Com as perguntas finalizadas, foi criado um texto padrão, o qual continha o *link* para resposta ao questionário, e enviado para o endereço de *e-mail* de cada empresa, encontrado nas pesquisas anteriormente feitas. Nesse texto, informou-se o tempo estimado para a resposta do questionário, o qual foi presumido em dois minutos, além do fim para o qual o estudo se destinava.

3.3 Variáveis analisadas

Com a obtenção das respostas das empresas ao questionário, realizou-se a análise das principais variáveis do estudo, a qual foi feita usando estatística descritiva. A Tabela 1 traz as variáveis submetidas à análise e uma breve descrição das mesmas.

Tabela 1 – Variáveis analisadas pelo estudo e descrição das mesmas.

Variável	Descrição
Tamanho da empresa	É medida pela quantidade de funcionários da companhia, considerando-se os critérios estabelecidos pelo SEBRAE (2016). No caso deste estudo, as empresas foram separadas em microempresas ou empresas de pequeno porte.
Aplicação de testes de <i>softwares</i>	Permitiu visualizar a prática da fase de testes pelas empresas participantes da pesquisa.
Técnica empregada na execução dos testes	Permitiu observar o cenário de aplicação de testes automatizados nas empresas avaliadas e quantidade das que ainda não aplicam tal técnica.
Forma de execução dos testes e tempo de atuação das empresas no mercado	Possibilitou a percepção da aplicação de testes automatizados em relação ao tempo de atuação das empresas.

Motivos para não aplicar testes automatizados	Pode-se perceber as principais razões para que as empresas não tenham interesse em implantar a técnica automatizada de testes de <i>software</i> .
Dificuldades para a realização dos testes	Permitiu visualizar os principais obstáculos que as empresas encontram para realizar os testes de <i>software</i> .
Tempo gasto com a atividade de testes	Possibilitou verificar a porcentagem de tempo que as empresas analisadas gastam com a atividade de testes dentro do prazo total do projeto.
Uso de ferramentas de testes e porte da empresa	Através das informações obtidas quanto à utilização de ferramentas para testes de <i>software</i> e ao tamanho do estabelecimento em relação à quantidade de funcionários, foi possível analisar em qual tipo de empresa o uso de ferramentas de testes é mais comum.
Principais erros encontrados nos sistemas desenvolvidos	Permitiu observar em quais classes estão concentrados os erros nos sistemas criados e verificar como os testes automatizados poderiam auxiliar na diminuição de tais erros.
Responsáveis pela realização dos testes	Foi possível verificar quais são os envolvidos na etapa de testes nas empresas.
Motivos para as empresas aplicarem testes automatizados	Pode-se perceber as principais razões que motivam as empresas para a aplicação de testes automatizados, destacando seus benefícios.
Principais vantagens dos testes automatizados em comparação aos manuais	Possibilitou verificar as principais vantagens percebidas pelas empresas na aplicação de testes automatizados de <i>software</i> comparada a de testes manuais.

Para a geração dos gráficos, foi utilizada a ferramenta Planilhas Google, a qual é *online* e gratuita e permite a criação, edição e colaboração de planilhas. A ferramenta possui a

funcionalidade de criação de gráficos, fórmulas integradas, tabelas dinâmicas e filtragem dos dados, entre outras [GOOGLE DOCUMENTATION, 2018].

4 ANÁLISE DOS RESULTADOS

Neste capítulo são discutidos os resultados obtidos através da pesquisa realizada. Ressalta-se que, após o *e-mail* com o *link* do questionário ser enviado para as 40 empresas selecionadas, foram obtidas 20 respostas. Logo, as discussões a seguir são sobre as empresas que participaram efetivamente do questionário.

4.1 Forma de aplicação dos testes

A análise quantitativa dos dados das respostas de 20 empresas, permitiu perceber que, destas, 56,25% são microempresas, enquanto 43,75% são empresas de pequeno porte, seguindo os critérios de classificação estabelecidos pelo SEBRAE (2016). Além disso, foi observado que as empresas geralmente aplicam algum tipo de teste de *software* visando qualidade nos produtos desenvolvidos, sendo que apenas uma empresa consultada afirmou não utilizar nenhuma técnica de testes. Quanto ao tipo de teste aplicado, Bernardo e Kon (2008) afirmam que, diversas vezes na indústria brasileira de *software*, são realizados testes manuais após a conclusão de módulos ou até mesmo do sistema como um todo para garantir que os mesmos contêm as características que lhes asseguram qualidade. Reiterando essa afirmação, a Figura 1 mostra que, em 53,3% das empresas pesquisadas, os testes são executados somente de forma manual, i.e., feitos pelos próprios desenvolvedores ou por uma equipe escalada exclusivamente para a realização dos mesmos [MALDONADO et. al, 2007].

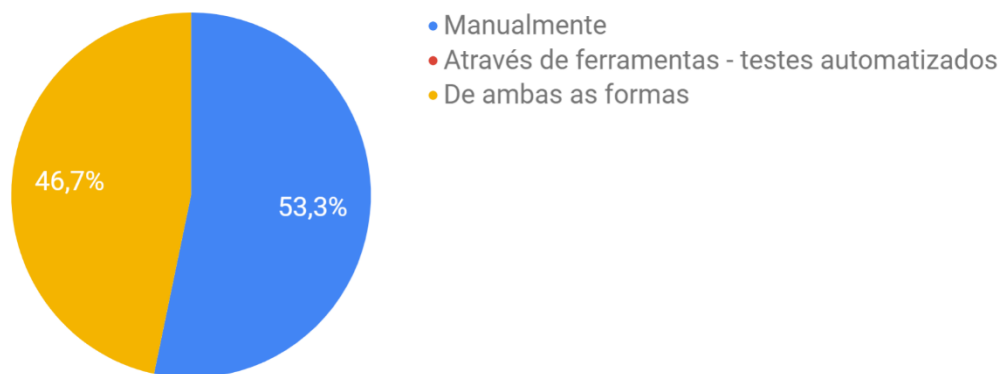


Figura 1 – Forma de execução dos testes nos *softwares* desenvolvidos pelas empresas.

A Figura 1 ainda indica que 46,7% das empresas combinam as duas técnicas de testes: manual e automatizada. Assim, foi possível constatar na prática a afirmação de Maldonado e outros (2007) que a aplicação de testes automatizados de *software* ainda é menor que a de testes manuais. Percebe-se que não foram encontradas empresas que usam apenas ferramentas

automatizadas para o teste de *software*. Frequentemente, na prática, os testes combinam as abordagens manual e automatizada [SOMMERVILLE, 2011]. Embora os testes automatizados consigam cobrir uma vasta gama de testes, os manuais ainda são indispensáveis [FEWSTER e GRAHAM, 1999 apud ROSA, 2016]. Assim sendo, o emprego de testes automatizados em conjunto aos manuais possibilita que as características específicas de cada categoria contribuam para aumentar a eficiência e diminuir o tempo dos testes [PRETTZ, 2014]. Sommerville (2011) afirma que “o uso de testes automatizados tem aumentado consideravelmente nos últimos anos. Entretanto, os testes nunca poderão ser totalmente automatizados, já que testes automáticos só podem verificar se um programa faz aquilo a que é proposto.”.

Nesse contexto de forma de aplicação dos testes, Bernardo (2011) afirma que “o tema Automação de Testes ainda é recente no Brasil.”. A Figura 2 reproduz a afirmação anterior, sendo possível verificar que, nas empresas com menos de 15 anos de atuação no mercado, a aplicação de testes automatizados é mais comum do que nas empresas com mais de 15 anos.

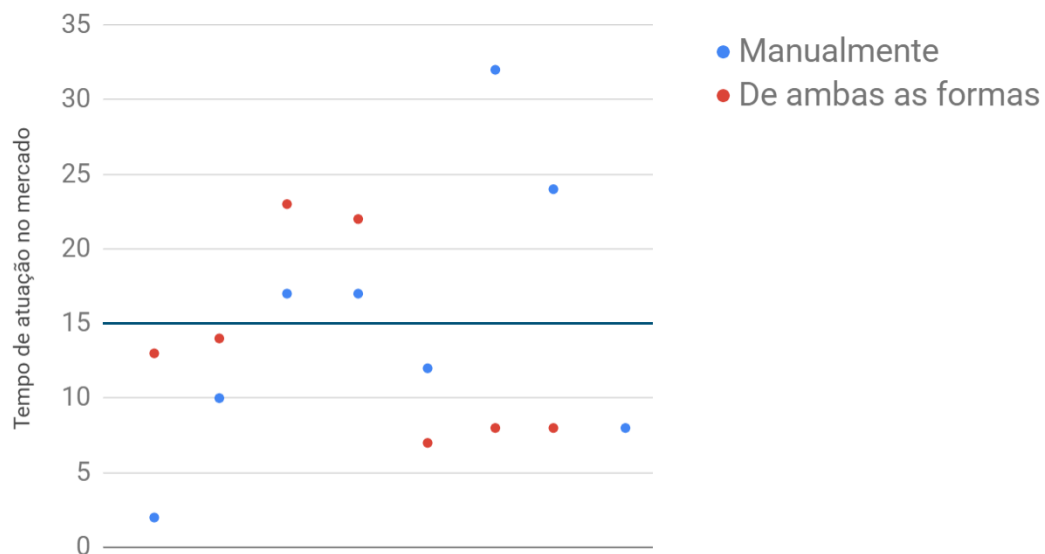


Figura 2 – Forma de execução dos testes relacionada ao tempo de atuação das empresas no mercado.

4.2 Algumas limitações da automação

Segundo Bernardo (2011) “o tema Automação de Testes [...] é de muito interesse por parte das empresas nacionais.”. Apesar de perceber-se grande aceitação da prática automatizada de testes pelas empresas pesquisadas, como já exibido na Figura 1, a maioria das empresas avaliadas que não aplicam testes automatizados não acompanha a premissa do autor, já que afirma não ter interesse em empregar a técnica automatizada em seus testes.

De acordo com Sommerville (2011), os testes automatizados não podem cobrir todos os tipos de testes existentes, como é o caso do teste de interface gráfica, por exemplo. “O *script* de teste não pode verificar tudo que um humano pode verificar” [FEWSTER e GRAHAM, 1999 apud ROSA, 2016], assim, os testes manuais são indicados em casos que exigem a interpretação do usuário [PRETTZ, 2014]. Dentre as causas apontadas pelas empresas para não adotar a técnica automatizada, a confecção de *softwares* que visam a usabilidade² reproduz as asserções anteriormente feitas. Essa questão precisa ser analisada por um ser humano e se apresenta como uma limitação dos testes automatizados. Outro motivo destacado foi o prazo e, conforme Paula Filho (2005), “testes manuais planejados podem ser necessários, por exemplo, quando não há prazo ou recursos suficientes para implementar testes automatizados em tempo para os testes de sistema.”.

A partir da análise das respostas obtidas, foi possível perceber que existe uma carência de conhecimento por parte de algumas empresas consultadas das aplicações que os testes automatizados possuem e como essas podem ajudá-las a otimizar seus processos, além da falta de profissionais especializados na área que possam auxiliá-las, tanto agregando *know-how* sobre o assunto, quanto cooperando para a implantação desse tipo de técnica de teste nos sistemas desenvolvidos por elas. A Tabela 2 exhibe as demais razões que levam as empresas consultadas a não utilizarem recursos automatizados.

Tabela 2 – Motivos para as empresas não adotarem a prática automatizada de testes.

Motivos para as empresas não implantarem testes automatizados de <i>software</i>
Ainda não foi discutido essa questão internamente e não foi planejado.
O perfil dos clientes exige agilidade para a entrega das soluções e é necessário que todo tempo disponível seja para atender as solicitações.
O modelo de negócio.
Os sistemas desenvolvidos visam a usabilidade, por isso, os testes têm que ser manuais.
Prazo.

² Nielsen (1993) afirma que a usabilidade está associada à facilidade do aprendizado e do uso de determinada interface e, ainda, à satisfação do usuário ao realizá-lo.

4.3 Dificuldades dos testes de *software*

Dentre as dificuldades elencadas, a apontada como maior foi a limitação do tempo para a atividade de teste devido a restrições no cronograma de entrega, com 68,75% de concordância, o que condiz com a afirmação de Paula Filho (2005) que apresenta o cronograma restrito como um dos fatores que mais dificultam os testes. A falta de profissionais especializados na área de testes e o fato de não haver um procedimento de teste de *software* adequado conhecido também se destacaram nas respostas das empresas, com 25 e 12,5% de consideração, respectivamente.

4.4 Tempo gasto com a atividade de testes

Quanto ao tempo gasto com os testes de *software*, a maioria das empresas respondeu gastar de 5 a 15% com a atividade. Pressman (1995) recomenda que o esforço de teste gaste entre 30 e 40% do tempo de um projeto, o que foi afirmado ser feito apenas por 14,29% das empresas pesquisadas. Foi possível perceber que esse baixo intervalo de tempo dedicado à fase de testes está estreitamente relacionado à principal dificuldade apontada, sendo que, das empresas que consideraram a restrição de tempo como limitação, nenhuma afirmou dedicar mais de 30% do tempo aos testes nos *softwares* desenvolvidos, como mostra a Figura 3.

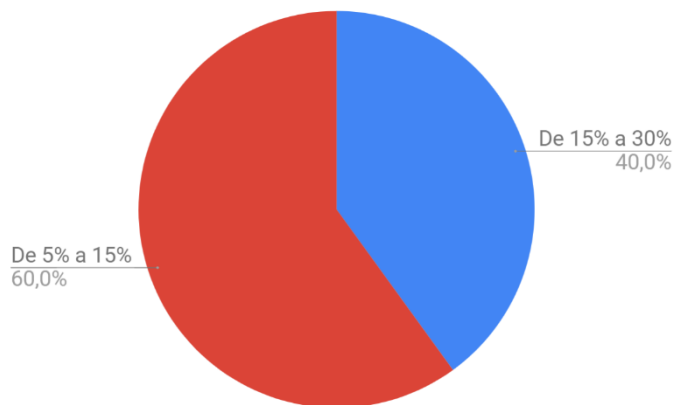


Figura 3 – Porcentagem de tempo gasto com os testes de *software* pelas empresas que consideraram a limitação de tempo para a atividade de teste como dificuldade.

4.5 Utilização de ferramentas para os testes

Observou-se também que 37,5% das empresas utilizam ferramentas automatizadas de testes de *software* e, dentre elas, a grande maioria, 66,7%, é classificada como empresa de micro porte – o que significa que possui menor número de funcionários que as demais –, como mostra a Figura 4. O uso de ferramentas na etapa de testes por essa categoria de empresas pode ser associado ao número reduzido de funcionários, os quais precisam cumprir outras tarefas e não

possuem grande disponibilidade de tempo para realizar os testes dos módulos e sistemas. A Figura 5 indica que, das microempresas que contam com ferramentas automatizadas na fase de testes, três quartos gastam menos de 30% do tempo total do projeto com os testes.

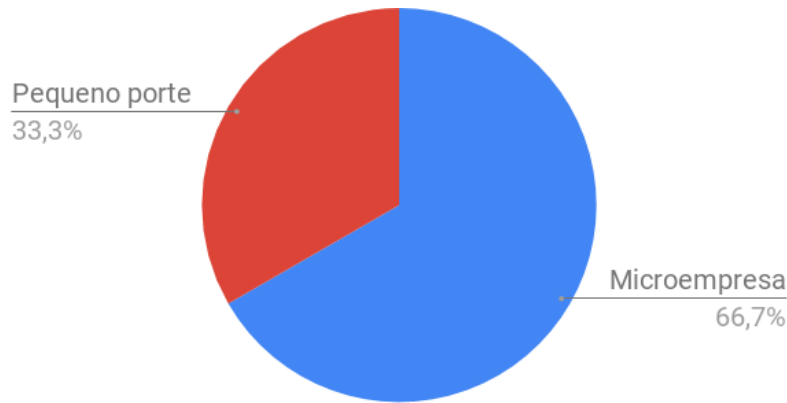


Figura 4 – Percentual de utilização de ferramentas automatizadas entre empresas de micro e pequeno porte.

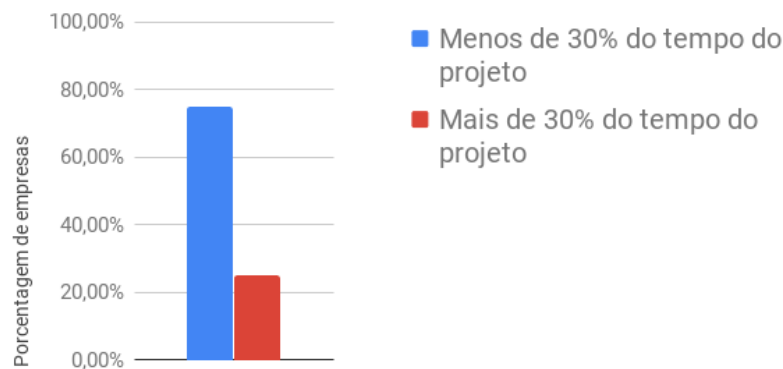


Figura 5 – Percentual de microempresas que utilizam ferramentas automatizadas e porcentagem do tempo total do projeto gasto com os testes de *software* pelas mesmas.

As empresas foram questionadas, ainda, sobre quais ferramentas são utilizadas na fase de testes, estando entre as citadas DUnit³, Selenium⁴, Jenkins⁵ e Sonar⁶. Algumas empresas mencionaram contar com ferramentas próprias, as quais não serão descritas neste trabalho devido ao caráter sigiloso das respostas do questionário.

4.6 Principais erros encontrados pelos testes

As empresas também foram questionadas sobre os erros mais frequentes encontrados pela aplicação de testes de *software*. Para isso, foram elencados os seis erros mais comuns de acordo com a literatura da área [CHILLAREGE et al., 1992 apud PFLEEGER, 2012]. Vale

³ <http://dunit.sourceforge.net/>

⁴ <https://www.seleniumhq.org/>

⁵ <https://jenkins.io>

⁶ <https://www.sonarqube.org/>

ressaltar que a empresa poderia marcar quantas respostas desejasse, incluindo todas ou nenhuma opção, o que pode fazer com que a soma das porcentagens ultrapasse 100%.

Pfleeger (2004) citou em sua obra algumas pesquisas sobre os erros de *software*. Um estudo empírico feito por Basili e Perricone (1984), o qual avaliou os erros encontrados em *softwares* de tamanho médio, revelou que 48% desses erros são oriundos de especificações ou requisitos que estão incorretos ou foram mal interpretados. O Computer Weekly Report (1994) exibiu uma pesquisa onde 44,1% dos erros de *software* aconteciam na fase de especificação e Perry e Stieg (1993) concluíram que 79,6% dos erros de interface e 20,4% dos erros de implementação são resultados dessa etapa [PFLEEGER, 2004].

A observação dos dados obtidos na pesquisa realizada para este trabalho revelou que, nas empresas avaliadas para esta região, os erros de interpretação de análise dos requisitos são predominantes e, geralmente, acompanham outros erros. Das empresas que consideraram os erros de interpretação dos requisitos, mais de 70% afirmaram encontrar demais erros, reforçando os achados dos estudos citados anteriormente. Em seguida, os erros apontados como os mais encontrados pelos testes nas empresas foram os de codificação e funcionalidade. A Figura 6 apresenta os resultados obtidos e confirma os achados da literatura da área.

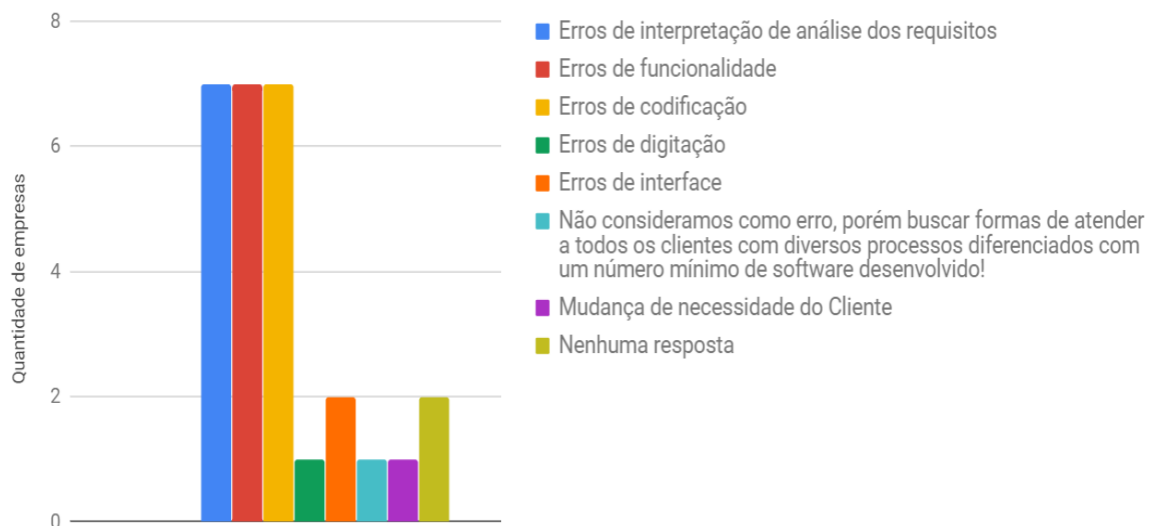


Figura 6 – Erros mais comuns de serem encontrados nos softwares desenvolvidos.

4.7 Responsáveis pelos testes de *software*

Em relação aos responsáveis pela realização dos testes, Pfleeger (2004) defende que os testes devem ser feitos por uma equipe especializada que não esteja ligada ao processo de

desenvolvimento do *software*. O cenário observado na pesquisa não reproduziu a proposição do autor já que, em mais da metade das empresas consultadas, os desenvolvedores participam da fase de testes e em 18,75% essa atividade é realizada exclusivamente por eles, como exibe a Figura 7. Pfleeger (2004) destaca alguns motivos para justificar sua tese, entre os quais está a possibilidade de que o desenvolvedor não consiga detectar algumas falhas por estar muito envolvido com o código que escreveu. O autor também cita que, caso exista uma equipe independente – como acontece em 56,25% das empresas participantes do estudo deste trabalho –, os testes podem ser realizados em paralelo ao desenvolvimento do sistema, sendo que esse grupo “pode participar das revisões dos requisitos e do projeto, testar os componentes de código individualmente bem como o sistema integrado apresentado para o cliente.” [PFLEEGER, 2004].

Kanij et. al (2011) afirma que “os serviços de teste estão sendo terceirizados para empresas especializadas e a maioria dos grandes projetos de desenvolvimento de *software* já envolve o uso de testadores especializados.” [apud SILVA, 2012]. Foi possível notar que a adesão à terceirização dos testes é pequena nas empresas do Centro-oeste mineiro pesquisadas, sendo que apenas 6,25% disseram adotá-la. Constatou-se, ainda, que menos de 10% dessas empresas inserem o cliente na fase de testes. A Tabela 3 traz a relação dos responsáveis pelos testes e a porcentagem de empresas em que eles atuam.

Acredita-se que essa não conformidade com a literatura é devido ao tamanho das empresas analisadas, sendo esse um dos achados interessantes deste TCC.

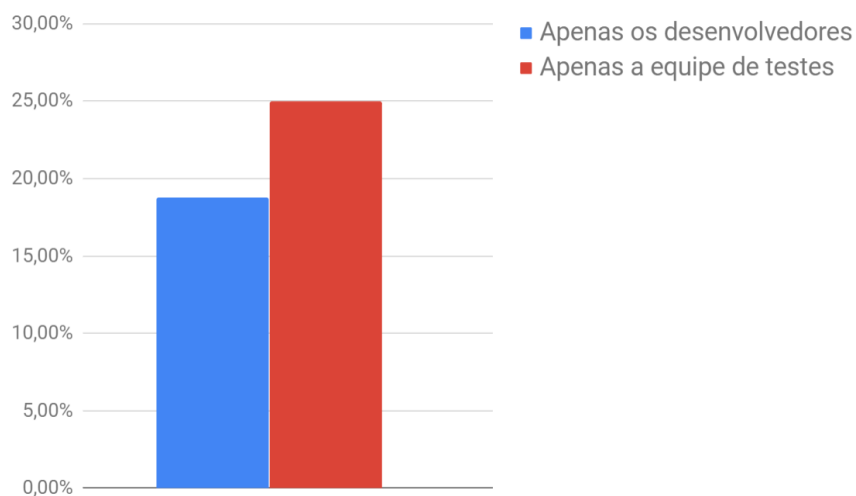


Figura 7 – Porcentagem de empresas nas quais a atividade de teste é realizada exclusivamente pelos desenvolvedores do sistema ou pela equipe de testes.

Tabela 3 – Responsáveis por realizar os testes nos *softwares* desenvolvidos pelas empresas *versus* percentagem de empresas nas quais eles atuam.

Responsáveis pelos testes nos <i>softwares</i> desenvolvidos	Porcentagem de empresas
O(s) desenvolvedor(es) do sistema.	68,75%
Equipe de testes da empresa.	56,25%
Os colaboradores responsáveis pelo suporte técnico.	6,25%
Os clientes que solicitaram o sistema.	6,25%
Os testes são terceirizados, feitos por especialistas em testes.	6,25%
Nenhuma resposta.	6,25%

4.8 Vantagens dos testes automatizados em relação aos manuais

Mesmo que existam motivos para algumas empresas não aderirem à automação, muitas vantagens são reconhecidas pelas que a incorporam ao processo de teste. Dessas empresas, a maioria cita a garantia de qualidade do produto entregue ao cliente como uma das principais motivações para o uso de testes automatizados, assim como citam Crispin e Gregory [2009, apud BERNARDO, 2011] em sua obra. A Tabela 4 apresenta as principais razões que impulsionam as empresas participantes da pesquisa a incorporar a técnica automatizada à fase de testes.

Tabela 4 – Motivos para as empresas aplicarem testes automatizados.

Principais motivações para aplicar testes de <i>software</i> nos produtos desenvolvidos pelas empresas
Qualidade.
Q&A.
Rapidez dos testes e qualidade do produto final.
Qualidade do produto.
Obtenção de um determinismo padrão nos processos de testes, com o objetivo de blindar o máximo os impactos aos operadores de sistema.
Qualidade de <i>software</i> entregue e possibilidade de rodar o teste automático na manutenção corretiva e evolutiva do <i>software</i> sem quebrar a regra de negócio do mesmo.

Finalmente, foi realizado um estudo sobre os benefícios observados pelas empresas na utilização de testes automatizados de *software* em comparação a de testes manuais. Foi possível constatar que o maior destaque foi do ganho relacionado à diminuição do tempo gasto na fase de teste, o que reitera que a grande vantagem da automação está em minimizar o tempo gasto com os testes e na possibilidade de refazê-los diversas vezes, como afirma Pressman (1995) em sua obra. Sommerville (2011) também cita esse benefício, alegando que “essa forma é geralmente mais rápida que o teste manual, especialmente quando envolve testes de regressão.”. A Tabela 5 exhibe a relação entre as vantagens dos testes automatizados sobre os testes manuais, as quais foram incluídas no questionário com base na literatura [BERNARDO, 2011; FANTINATO et al., 2004; GANDARA, 2012; MYERS et al., 2004; RAFI et al., 2012], e a porcentagem de empresas que afirmaram percebê-las.

Tabela 5 – Vantagens dos testes automatizados de *software* em comparação aos testes manuais *versus* porcentagem de empresas que consideraram a vantagem.

Vantagem	Porcentagem
Redução do tempo gasto na execução dos testes.	85,7%
Melhoria da qualidade do produto final e da construção do <i>software</i> .	71,4%
Aumento na confiança de liberação do produto.	71,4%
Maior produtividade e redução de custos destinados à atividade de testes.	71,4%
Alto índice de cobertura, com diversos casos de testes.	71,4%
Diminuição de esforços humanos, com a eliminação do trabalho repetitivo de inserção e observação de dados.	57,1%
Possibilidade de arquivar os <i>scripts</i> e utilizá-los na reexecução dos testes, podendo alterá-los.	42,9%

A Tabela 5 exhibe a ampla concordância sobre as vantagens da técnica automatizada, visto que grande parte dessas tiveram a confirmação da maioria das empresas participantes. Assim como afirma a literatura – e pode ser constatado nas Tabelas 4 e 5 –, a automação dos

testes está sendo considerada a principal forma de aumentar a eficiência da atividade de testes [FANTINATO et. al, 2004]. Caso seja realizada de maneira apropriada, a técnica se torna uma das melhores formas de “reduzir o tempo de teste no ciclo de vida do *software*, diminuindo o custo e aumentando a produtividade do desenvolvimento de *software* como um todo, além de, conseqüentemente, aumentar a qualidade do produto final.” [KANER, 1997 apud FANTINATO et. al, 2004]. Sendo assim, mesmo que a automação não consiga cobrir todos os tipos de testes existentes [BERNARDO e KON, 2008; FEWSTER e GRAHAM, 1999 apud ROSA, 2016; Oliveira et al., 2014 apud JÚNIOR, ANDRADE e DELAMARO, 2017], ela pode otimizar o tempo gasto com testes repetitivos que dispensam o foco absoluto dos testadores [PRETTZ, 2014] e o tempo economizado pode ser dedicado aos testes que necessitam da atenção humana, como é o caso dos testes de usabilidade dos sistemas. Ao final, as empresas terão a capacidade de entregar aos seus clientes *softwares* de maior qualidade, considerando as funcionalidades propostas e a construção de sistemas mais intuitivos para a utilização pelo usuário final.

4.9 Ameaças à validade

Algumas questões apresentam-se como ameaças à validade deste trabalho, como a dificuldade de realizar-se um estudo empírico, em especial por envolver pessoas externas ao instituto no qual foi realizado e o fato de não ser possível assegurar que as empresas responderam de maneira fiel ao cenário que se encontra dentro delas, sendo que a variação das respostas interfere diretamente nos resultados. Para amenizar esse risco, o questionário foi enviado ao proprietário da empresa e salientou-se que os dados da mesma não seriam divulgados.

5 CONSIDERAÇÕES FINAIS

O desenvolvimento do presente trabalho possibilitou ao autor melhor percepção da importância que a fase de testes possui no processo de construção de um *software* e como a identificação tardia de erros pode trazer prejuízos às empresas, tanto financeiros, quanto em relação ao tempo que será gasto com as correções. Foi observado que as empresas de desenvolvimento de *software* da região do Centro-oeste mineiro vêm utilizando testes de *software* como alternativas para garantir que um requisito seja cumprido com menor esforço e, principalmente, economizando tempo e recursos.

Constatou-se que essas empresas utilizam alguma técnica de teste, destacando-se a prática manual, e preocupam-se com a qualidade dos produtos desenvolvidos. Verificou-se, ainda, que as microempresas utilizam mais ferramentas de testes automatizados quando comparadas às empresas de pequeno porte, o que pode estar associado ao menor número de funcionários das primeiras em relação às últimas. Dentre os erros apontados como mais encontrados nas empresas avaliadas, destacam-se os erros de interpretação dos requisitos, funcionalidade e codificação. Tais erros também são citados como destaque na literatura vigente. Por fim, observou-se que as empresas conseguem perceber os benefícios trazidos pelo uso de técnicas automatizadas de testes de *software*, principalmente em relação à diminuição do tempo de execução e dos custos com os mesmos, ao crescimento da produtividade na construção do *software* e ao consequente aumento da qualidade do produto final.

O objetivo deste trabalho foi alcançado a partir da possibilidade de perceber-se como a aplicação de testes vigora em empresas de micro e pequeno porte do Centro-oeste mineiro, sendo que foi possível identificar que existe uma lacuna no mercado por profissionais especializados em testes de *software*.

Como trabalhos futuros, espera-se agregar conhecimento sobre o tema, o qual é bastante amplo, conhecendo-se melhor as técnicas de testes e como eles são de fato executados. Pretende-se, ainda, estudar as principais ferramentas de testes automatizados disponíveis, compreendendo as etapas do teste e aprendendo a utilizá-las nos casos mais apropriados.

6 REFERÊNCIAS BIBLIOGRÁFICAS

BERNARDO, P. C. **Padrões de testes automatizados**. 2011. Dissertação (Mestrado em Ciência da Computação)-Instituto de Matemática e Estatística, Universidade de São Paulo, São Paulo, 2011. Disponível em: <<http://www.teses.usp.br/teses/disponiveis/45/45134/tde-02042012-120707/en.php>>. Acesso em: 06 set. 2018.

BERNARDO, P. C.; KON, F. A importância dos Testes Automatizados. **Engenharia de Software Magazine**, v. 3, p. 54-57, 2008.

CRESPO, A. N. et al. Uma Metodologia para Teste de Software no Contexto da Melhoria de Processo. **Simpósio Brasileiro de Qualidade de Software**, S.I, p.271-285, 2004. Disponível em:

<https://www.researchgate.net/profile/Adalberto_Crespo/publication/237497188_Uma_Metodologia_para_Testes_de_Software_no_Contexto_da_Melhoria_de_Processo/links/54e5d1040cf2cd2e028b338b.pdf>. Acesso em: 08 set. 2018.

DELAMARO, M. E.; JÚNIOR, M.; ANDRADE, S. **Capítulo 6: Automatização de teste de software com ênfase em teste de unidade**, 2017.

FANTINATO, Marcelo et al. AutoTest – Um framework reutilizável para a automação de teste funcional de software. **Simpósio Brasileiro de Qualidade de Software**, 2004.

Google Forms - create and analyze surveys, for free. Disponível em: <<https://docs.google.com/forms/u/0/>>. Acesso em: 07 ago. 2018.

Google Sheets - create and edit spreadsheets online, for free. Disponível em: <<https://docs.google.com/spreadsheets/u/0/>>. Acesso em: 07 ago. 2018.

IEEE. **Ieee standard glossary of software engineering terminology**. Standard 610.12, IEEE Press, 1990.

IEEE. **Ieee standard for software and system test documentation**. Standard 829, IEEE Press, 2008.

LANZA, M.; MARINESCU, R. **Object-Oriented Metrics in Practice: Using Software Metrics to Characterize, Evaluate, and Improve the Design of Object-Oriented Systems**. S.I: Springer, 2006. 206 p.

LIMA, G. M. P. S.; TRAVASSOS, G. H. Testes de integração aplicados a software orientado a objetos: Heurísticas para ordenação de classes. In: (SBQS 2004) **III Simpósio Brasileiro de Qualidade de Software**, Sociedade Brasileira de Computação, Brasília, DF, 2004. Disponível em: <<https://www.cos.ufrj.br/uploadfile/es63204.pdf>>. Acesso em: 20 ago. 2018.

LUFT, C. C. **Teste de software: uma necessidade das empresas**. 2012. 91 f. TCC (Graduação) - Curso de Ciência da Computação, Departamento de Ciências Exatas e Engenharias, Universidade Regional do Noroeste do Estado do Rio Grande do Sul, Santa Rosa, 2012. Disponível em: <<http://bibliodigital.unijui.edu.br:8080/xmlui/bitstream/handle/123456789/1307/TCC%20-%20Cristiane%20Carline%20Luft.pdf?sequence=1>>. Acesso em: 25 jun. 2018.

MALANOVICZ, A. V.; GUILLEN, C. M. B. Análise do processo de teste de software na implantação de ERP SAP “com” e “sem” o uso da ferramenta de automação de testes ECATT. **XXXIV Encontro Nacional de Engenharia de Produção**, 2014.

MALDONADO, J. C.; DELAMARO, M. E.; JINO, M. **Introdução ao Teste de Software**. 1 ed. Editora Campus/Elsevier. 2007.

MYERS, G. J.; SANDLER, C.; BADGETT, T. **The art of software testing**. John Wiley & Sons, 2011.

NIELSEN, J. **Usability Engineering**. New York, NY: Academic Press, 1993.

PAULA FILHO, W. de P. **Engenharia de Software**. São Paulo SP, LTC, 2003.

PFLEEGER, S. L. **Engenharia de Software**. 2ª ed., Editora Pearson Prentice-Hall, 2004, ISBN 978-85- 8791-831-4.

PRESSMAN, R. S. **Engenharia de software**. São Paulo: Makron books, 1995.

PRETTZ, J. B. **Uma proposta de um processo de teste em uma empresa de pequeno porte**. 2014. 11 f. TCC (Bacharelado em Sistemas de Informação)- Centro de Educação Superior do Norte - CESNORS, Universidade Federal de Santa Maria – UFSM, [S.l.], 2014. Disponível em: <https://repositorio.ufsm.br/bitstream/handle/1/12812/TCCG_SIFW_2014_PRETTZ_JULIA_NO.pdf?sequence=1&isAllowed=y>. Acesso em: 05 out. 2018.

RAFI, D. M. et al. Benefits and limitations of automated software testing: Systematic literature review and practitioner survey. In: **Proceedings of the 7th International Workshop on Automation of Software Test**. IEEE Press, 2012. p. 36-42.

ROSA, D. da. **Testes de aceitação automatizados: construindo e integrando com a ferramenta Jenkins**. 2016. 119 f. TCC (Bacharelado em Sistemas de Informação)- Universidade do Sul de Santa Catarina, Palhoça, 2016. Disponível em: <https://riuni.unisul.br/bitstream/handle/12345/3560/112573_Daniel.pdf?sequence=1&isAllo wed=y>. Acesso em: 24 set. 2018.

SERVIÇO BRASILEIRO DE APOIO ÀS MICRO E PEQUENAS EMPRESAS - SEBRAE. **Anuário do trabalho nos pequenos negócios**: 2016. 9. ed. São Paulo, SP: DIEESE, 2018. Disponível em: <<https://www.dieese.org.br/anuario/2018/anuarioDosTrabalhadoresPequenosNegocios.html>>. Acesso em: 04 set. 2018.

SILVA, T. G. da. **Jogos sérios em mundos virtuais: uma abordagem para o ensino-aprendizagem de teste de software**. 2012. 89 f. Dissertação (Mestrado em Ciência da Computação)- Programa de Pós-graduação em Informática, Universidade Federal de Santa Maria, Santa Maria, 2012. Disponível em: <<https://repositorio.ufsm.br/bitstream/handle/1/5398/SILVA,%20TARCILA%20GESTEIRA%20DA.pdf>>. Acesso em: 24 set. 2018.

SOMMERVILLE, I. **Engenharia de Software**: 8 Ed. São Paulo: Pearson Education, 2007.

VIDAL, Adriana Rocha. **Teste Funcional Sistemático Estendido: Uma Contribuição na Aplicação de Critérios de Teste Caixa-Preta**. 2011. 138 f. Dissertação (Mestrado em Ciência da Computação)- Programa de Pós-Graduação do Instituto de Informática, Universidade Federal de Goiás, Goiânia, 2011. Disponível em: <https://repositorio.bc.ufg.br/tede/bitstream/tede/2887/5/Teste_funcional_sistematico_estendido_uma_contribuicao_na_aplicacao_de_criterios_de_teste_caixa_preta.pdf>. Acesso em: 24 set. 2018.

APÊNDICE A – Questionário sobre o uso de testes de *software*

Com finalidade de entender e analisar como está o uso de testes de *softwares* no estado de MG.

Você gastará no máximo 2 minutos para nos contar um pouco da realidade da sua empresa.

As informações obtidas são confidenciais e de caráter exclusivamente acadêmico.

Desde já, agradecemos sua participação!

Marcela Fonseca - Graduanda em Ciência da Computação - IFMG - Campus Formiga

Paloma Oliveira – Orientadora

*(As questões que contêm um * são obrigatórias).*

1. Nome da empresa: * _____
2. Cidade: * _____
3. Quantos funcionários a empresa possui? * *(Apenas uma opção deve ser selecionada nesta questão)*
 - Menos de 10
 - De 10 a 30
 - De 30 a 50
 - Mais de 50
4. A empresa aplica testes de *software* em seus produtos? * *(Apenas uma opção deve ser selecionada nesta questão)*
 - Sim
 - Não
5. A empresa tem interesse em implantar testes automatizados de *software*? * *(Apenas uma opção deve ser selecionada nesta questão)*
 - Sim

- Não
6. Quais os motivos para a empresa não implantar testes automatizados de *software*?

7. Quem são os responsáveis por realizar os testes nos *softwares* desenvolvidos pela empresa? * (Mais de uma opção pode ser selecionada nesta questão)
- O(s) desenvolvedor(es) do sistema
- Equipe de testes da empresa
- Os testes são terceirizados, feitos por especialistas em testes
- Os clientes que solicitaram o sistema
- Outro: _____
8. Quais os erros mais comuns de serem encontrados nos sistemas desenvolvidos pela empresa? (Mais de uma opção pode ser selecionada nesta questão)
- Erros de interpretação de análise dos requisitos
- Erros de codificação
- Erros de funcionalidade
- Erros de digitação
- Erros de interface
- Outro: _____
9. Quais as principais dificuldades encontradas na realização dos testes de *software*? * (Mais de uma opção pode ser selecionada nesta questão)
- É um processo caro
- O tempo para a atividade de teste é limitado devido a restrições no cronograma de entrega do *software*
- Faltam profissionais especializados na área de teste de *software*
- A implantação de um processo de teste de *software* apresenta dificuldades
- Não há um procedimento de teste de *software* adequado conhecido

- Não há técnicas de teste de *software* adequadas conhecidas
- Não há conhecimento sobre como planejar a atividade de teste de *software*
- Outro: _____
10. Como são realizados os testes de *software* na empresa? * (*Apenas uma opção deve ser selecionada nesta questão*)
- Manualmente
- Através de ferramentas - testes automatizados
- De ambas as formas
11. A empresa utiliza alguma ferramenta para testes automatizados? * (*Apenas uma opção deve ser selecionada nesta questão*)
- Sim
- Não
12. Quais ferramentas são utilizadas para a realização dos testes de *software*?

13. Qual porcentagem de tempo do projeto, em média, é gasta com os testes automatizados de *software*? * (*Apenas uma opção deve ser selecionada nesta questão*)
- Até 5%
- De 5% a 15%
- De 15% a 30%
- De 30% a 50%
- Acima de 50%
14. De acordo com sua experiência ou de sua equipe, quais as vantagens dos testes automatizados em relação aos testes manuais? * (*Mais de uma opção pode ser selecionada nesta questão*)
- Redução do tempo gasto na execução dos testes.
- Melhoria da qualidade do produto final e da construção do *software*.

- () Aumento na confiança de liberação do produto.
- () Diminuição de esforços humanos, com a eliminação do trabalho repetitivo de inserção e observação de dados.
- () Maior produtividade e redução de custos destinados à atividade de testes.
- () Possibilidade de arquivar os *scripts* e utilizá-los na reexecução dos testes, podendo alterá-los.
- () Alto índice de cobertura, com diversos casos de testes.
- () Outro: _____

15. Qual a principal motivação para aplicar testes de *software* nos produtos da empresa?
