INSTITUTO DOCTUM DE EDUCAÇÃO E TECNOLOGIA

FACULDADES INTEGRADAS DE CARATINGA

CIÊNCIA DA COMPUTAÇÃO

ESTUDO SOBRE A IMPORTÂNCIA DA MANUTENÇÃO DE *SOFTWARE*

EDUARDO HENRIQUE TONEL LOPES

Caratinga
2017

Eduardo Henrique Tonel Lopes

ESTUDO SOBRE A IMPORTÂNCIA DA MANUTENÇÃO DE *SOFTWARE*

Monografia apresentada ao Curso de Ciência da Computação das Faculdades Integradas de Caratinga como requisito parcial para obtenção do título de bacharel em Ciência da Computação orientada pelo Prof. Msc. Elias de Souza Gonçalves.

Caratinga

2017

FACULDADES DOCTUM DE CARATINGA

FOLHA DE APROVAÇÃO

O Trabalho de Conclusão de Curso intitulado: ESTUDO SOBRE A IMPORTÂNCIA DA MANUTENÇÃO DE SOFTWARE, elaborado pelo aluno EDUARDO HENRIQUE TONEL LOPES foi aprovado por todos os membros da Banca Examinadora e aceita pelo curso de Ciência da Computação das Faculdades Doctum de Caratinga, como requisito parcial para a obtenção do título de:

BACHAREL EM CIÊNCIA DA COMPUTAÇÃO

Caratinga, 12 de dezembro de 2017

Msc. Elias de Souza Gonçalves

Esp. Maicon Vinicius Ribeiro

Msc. Glauber Luis da Silva Costa

RESUMO

O *software* encontra-se cada vez mais presente em nossas vidas. Se em outro tempo se encontravam computadores somente em grandes empresas, institutos de pesquisa e em bases militares, hoje se têm diversos aparelhos com capacidade de processamento, sejam estes computadores, celulares, TVs entre outros. Muitas das vezes, tais softwares passam por modificações mesmo depois de implantados, para que atendam a novos requisitos a fim de atender os seus usuários. Contudo, é necessário que estas mudanças ocorram em um tempo hábil e de maneira eficaz, obedecendo a critérios que levam a uma manutenção segura e eficaz.

Através de pesquisas realizadas sobre a manutenção de *software* e suas características, observou que as pesquisas de vários autores sobre o tema, enfatizaram a importância de se aplicar conceitos desde o início do projeto de um *software* até sua implantação, para que a etapa de manutenção ocorra sem conturbações.

A manutenção de *software* possui um custo mais elevado do que a etapa de desenvolvimento em aproximadamente dois terços, tendo a necessidade de que sejam adotados métodos que proporcionem a redução deste custo, a fim de melhorar a distribuição do orçamento do projeto. 25% dos projetos nas grandes empresas de desenvolvimento de *software* não são concluídos devidos à falta de uma análise adequada durante o desenvolvimento e que há casos em que 50% ou mais das falhas são identificadas somente após a implantação do software.

Com o intuito de confrontar as opiniões dos autores com as opiniões de atuantes da área de Tecnologia da Informação, foi elaborado um questionário e disponibilizado para que os atuantes pudessem responder as questões que foram embasadas nas opiniões de vários autores sobre a manutenção de *software.*

Através das 80 respostas dos entrevistados que atuam na área de Tecnologia da Informação, foi possível conhecer as maiores dificuldades encontradas na etapa de manutenção do *software* e os fatores que causam o declínio da sua manutenibilidade. Além de conhecer fatores que possam

prejudicar a qualidade da manutenção dos produtos de *software*, também foi possível aferir que algumas propostas como a aplicação de uma disciplina sobre a manutenção de *software* e suas diretrizes e a reusabilidade são bem aceitas para auxiliar na etapa de manutenção do *software*.

Palavras-chave: Manutenção de *software*; Manutenibilidade;

ABSTRACT

Software is increasingly present in our lives. If in another time we found computers only in some big companies, research institutes and im military bases, today we have a lot of devices with processing capacity, like computers, cellphones, TVs among others. So many times, some softwares undergo some modifications even after being implemented, to accord new requisitions in order to attend their users. However, it is necessary that these changes occur in a timely manner and effective manner, obbeying some criteria that takes into a safety and effective maintenance.

Through realized researches about software maintenance and yours characteristics, it was observed that researches from various authors about the subject of work, emphasized the importance of implementing the concepts since the beginning of a software project until your implementation, so that the maintenance stage occur without disturbances.

The software maintenance has a higher cost than the development stage in approximately two-thirds, having the necessity to adopt methods to provide a cost reduction, in order to improve the project budget distribution. 25% of projects in software development big companies are not completed due the lack of adequate analysis during the development and there are cases in 50% or more of the failures are identified only after the software implementation.

With the purpose of compare the authors opinions with the opinions of acting persons from Information Tecnology area, a questionnaire was elaborated and made available so that the acting persons could answer the questions that was based on the opinions of several authors about software maintenance.

Through 80 respondent’s responses that act in Information Tecnology area, it was possible to know the greatest difficulties in the software maintenance stage and the factors that cause the decline of your maintainability. In addition to knowing the factors that may harm the maintenance quality of software products, it was also possible to compare some proposals like the application of a discipline about software maintenance and its guidelines and reusability which are well accepted to assist in the software maintenance stage.

Keywords: Software maintenance; maintainability.

LISTA DE ILUSTRAÇÕES

SUMÁRIO

INTRODUÇÃO

O presente trabalho tem como objeto de pesquisa a manutenção de software e a manutenibilidade. O objetivo geral deste trabalho tem o intuito de demonstrar a importância da manutenção de *software* e suas relações com todo o desenvolvimento do sistema através de um estudo realizado com base nos autores influentes no assunto deste trabalho e no levantamento de opiniões dos atuantes na área da computação, para que se tenha noção das principais dificuldades que se encontram ao atingir a manutenção e meios de se obter uma etapa de manutenção com menos riscos.

Para atingir o objetivo geral deste trabalho, foi necessário realizar pesquisas bibliográficas acerca da manutenção de *software*, suas diretrizes e sua importância; definir o que é *software* e seu processo de produção; desenvolver um questionário, aplicá-lo ao público-alvo e extrair seus dados; apresentar os resultados obtidos na pesquisa realizada, compará-los com os estudos levantados pelos autores citados nesta monografia e apresentar as devidas conclusões.

Segundo Sommerville (2011), a manutenção de *software* detém, aproximadamente, dois terços do orçamento contra um terço do orçamento para desenvolvimento, ocupando então uma proporção maior do orçamento geral de um projeto de TI (Tecnologia da Informação). Também observado por Sommerville (2011), os custos voltados à manutenção de *software*, dependendo do tipo de sistema, pode ser de até quatro vezes maior que o custo do desenvolvimento.

Com o alto crescimento da utilização de produtos de *software*, viu-se a necessidade de implementações de novos sistemas e do aprimoramento constante dos sistemas já existentes. Para que a manutenção destes produtos de *software* ocorra rapidamente para atender as necessidades do cliente, é de suma importância que o sistema desenvolvido se encontre manutenível, para que evite casos mais extremos, como por exemplo, a implementação de praticamente todo o *software* do início novamente, deve-se obedecer a critérios que ajudam a alcançar as características que levam a uma manutenção de maneira correta e sem percalços, dos quais serão descritos neste trabalho mais adiante.

Quando um *software* já se encontra em sua etapa de manutenção, os atuais

responsáveis pela manutenção do sistema, podem encontrar dificuldades em reconhecer o que certa funcionalidade está realizando. Esta dificuldade pode acabar gerando atrasos na entrega da modificação a ser feita, gerando transtornos aos envolvidos com o sistema (clientes e desenvolvedores).

A importância de se ter um sistema manutenível reflete diretamente no desempenho do atual desenvolvedor e daqueles que futuramente poderão estar trabalhando na continuação do projeto em questão. Visando o atual cenário em torno da manutenção dos produtos de *software*, foi desenvolvido um questionário composto por 21 perguntas a fim de analisar se o entendimento sobre manutenção de *software* dos respondentes está de acordo com as opiniões propostas dos autores citados neste trabalho.

Com base nas 80 respostas coletadas, foi possível realizar comparações com as opiniões dos autores e dos entrevistados sobre a manutenção de *software* e manutenibilidade. Também foi possível conhecer quais os tipos de manutenção mais ocorrente no dia a dia dos usuários. Das respostas obtidas, foi possível analisar quais as maiores dificuldades encontradas no ambiente de trabalho em diferentes atuações, como no desenvolvimento e no gerenciamento. Através delas, foi possível fazer uma análise das principais causas de problemas que levam ao declínio da manutenibilidade do *software*, além de saber a opinião dos respondentes sobre algumas propostas que possam auxiliar no aumento da qualidade da manutenção em geral.

Será necessário primeiramente, entender a definição do que é um *software*, o processo de produção do *software*, seu ciclo de vida e exemplos de modelos de ciclo de vida, os principais conceitos sobre manutenção de *software* e manutenibilidade. Para facilitar o entendimento, além desta introdução o trabalho encontra-se estruturado da seguinte forma:

- **Capítulo 1 - Referencial Teórico:** Aponta trabalhos com os conceitos necessários acerca dos assuntos principais que são fundamentais para o entendimento correto deste trabalho;
- **Capítulo 2 – Metodologia:** Descreve o desenvolvimento da metodologia adotada neste trabalho. As definições do público-alvo, o desenvolvimento das questões, a coleta dos dados o tratamento das informações estão detalhadas neste capítulo;

- **Capítulo 3 – Resultados:** A discussão dos dados obtidos e a descrição dos resultados da pesquisa realizada com os entrevistados foram detalhadas e seus resultados exibidos através de gráficos, sendo descritos posteriormente;
- **Capitulo 4 – Conclusão:** Através dos resultados obtidos pela metodologia proposta, tem-se a dissertação dos resultados obtidos através da confrontação realizada entre as opiniões dos respondentes com os estudos realizados dos autores;
- **Capítulo 5 – Trabalhos futuros:** Sugestões de trabalhos com abordagens relacionadas com o tema proposto neste trabalho, que possam ser implementadas futuramente.

1. REFERENCIAL TEÓRICO

Este capítulo aborda os principais conceitos necessários para um entendimento mais didático sobre o trabalho proposto. As definições citadas neste capítulo são tratadas sob a ótica dos renomados autores que contribuíram para o desenvolvimento desta pesquisa. Os principais temas utilizados como alicerce para o desenvolvimento são: A definição de *software*; o processo de produção de um *software*; modelos de ciclo de vida de um *software*; manutenção de *software* e manutenibilidade.

1.1. A DEFINIÇÃO DE *SOFTWARE*

Atualmente, o *software* se encontra presente na maioria das rotinas do dia-a-dia. Com o passar do tempo, os produtos de *softwares* foram, de certa forma, substituindo outras ferramentas e se tornando cada vez mais úteis em amplos sentidos, sendo encontrados na área de entretenimento em geral, produtos eletrônicos, como também em serviços de infraestrutura governamentais. Em um sentido mais amplo, caracteriza-se que um *software* pode ser um conjunto pré-definido de instruções que são capazes de executar funções das quais se esperam um resultado final satisfatório e rápido (SOMMERVILLE, 2011; PRESSMAN, 2011).

De acordo com Sommerville (2011, p.4) “Softwares são programas de computador e documentação associada. Produtos de *software* podem ser desenvolvidos para um cliente específico ou para o mercado em geral”. Reforçando a ideia de *software* com Pressman (2011), que se refere a um *software* podendo ser um produto, e também um meio de distribuição de produtos e que independente de seu formato, os softwares são encarregados de distribuir o mais importante – a informação.

Segundo Pressman (2011), há sete grandes categorias de *software*, sendo elas:

1. ***Software* de sistema** – Conjunto de programas criados com a finalidade de atender a outros programas. Exemplificados pelos compiladores, gerenciadores de arquivos e editores em geral, são preparados para processar estruturas complexas. Caracterizada por interagir de uma maneira pesada com o

hardware da máquina, utilizando os recursos disponíveis intensamente, e também possuem estrutura de dados complexas e múltiplas interfaces externas.

2. ***Software* de aplicação** – Programas que solucionam uma necessidade específica de negócio. Aplicações deste tipo de programa são caracterizadas por apresentarem formas que facilitem operações comerciais, tomadas de decisões técnicas ou administrativas e também controlar funções de negócios em tempo real (por exemplo, transações em pontos de venda ou controles de processos de fabricação em tempo real).
3. ***Software* científico/engenharia** – Caracterizados por algoritmos de processamento numérico pesado. Suas aplicações englobam as áreas de astronomia, vulcanologia, automatização, biologia molecular e também em tensões de mercado em geral.
4. ***Software* embutido** – residente num produto ou sistema, sendo utilizado para controlar funções para o usuário final e para o próprio sistema. Aplicam funções limitadas e específicas para controle, como por exemplo: painel de forno de um micro-ondas; sistemas de freios e controle do combustível de um automóvel.
5. ***Software* para linha de produtos** – Desenvolvido para prover capacidade específica de utilização de muitos clientes diferentes. Pode-se abranger mercados limitados e particularizados, como também em mercados de consumo em massa.
6. **Aplicações para *web*** – Abrangendo uma ampla área de atuação em redes, os *webApps* (como são referenciados), em sua forma mais simples, apresentam informações através de um conjunto de arquivos hipertextos interconectados, apresentando informações através de textos e gráficos limitados. Com o surgimento do Web 2.0, passaram a se caracterizar com ambientes computacionais amplos, fornecendo não só recursos especializados para o usuário final, encontrando-se também conectados a bancos de dados corporativos e aplicações comerciais.
7. ***Software* de inteligência artificial** – Utiliza algoritmos não numéricos para a solução de problemas complexos dos quais não são passíveis de computação ou análise direta. Aplicações nessa área incluem: robótica, redes neurais artificiais, reconhecimentos de padrões de imagens e voz, jogos, provas de

teoremas e sistemas especialistas.

Como se percebe, os produtos de *software* estão aplicados de diversas maneiras e em diversos lugares, para que o acesso à informação e à praticidade na realização de atividades seja ela profissional ou lazer, esteja ao alcance de qualquer pessoa. Em seguida, demonstram-se os processos de produção de um software.

1.2. PROCESSOS DE PRODUÇÃO DE UM *SOFTWARE*

Para que se conheçam os processos de produção do *software*, deve-se primeiramente ter o discernimento de um *software* com um simples programa, e também do significado de um processo de uma maneira geral. Segundo Paula Filho (2001), que se refere aos softwares como produto: programas que são utilizados para diversão de seu desenvolvedor e para resolver um único problema, sem utilização por partes de terceiros, estes são considerados fora do escopo do que se entende por *software*. Entende-se processo por um conjunto de etapas parcialmente ordenadas, constituindo atividades, métodos, práticas, transformações a fim de alcançarem uma meta estabelecida. Um processo é definido quando sua documentação detalha os produtos, os passos, os atuantes no desenvolvimento do projeto, o que será utilizado e o resultado que se espera atingir.

Em sua obra, Pressman (2011) afirma que a definição de produção de *software* não é seguir à risca um único modo de como desenvolver um *software*, mas sim uma abordagem flexível que possibilita aos envolvidos no projeto selecionar um conjunto de métodos e ações apropriadas para aquele projeto em questão, para que seja entregue com qualidade e dentro do prazo determinado, satisfazendo o(s) cliente(s) e patrocinador(es).

Embora existam vários processos de *software* diferente, é essencial que todos tenham quatro características fundamentais. Segundo Sommerville (2011), essas quatro características são:

1. **Especificação do *Software***: Funcionalidade e restrições para seu funcionamento devem ser especificadas.
2. **Projeto e Implementação do *Software***: O software tem a finalidade de atender às especificações.
3. **Validação de *Software***: O software deve ser validado, para que atenda às necessidades do cliente.

4. **Evolução de *Software*:** O software deverá evoluir, a fim de atender as futuras necessidades do cliente.

Reforçando a ideia sobre as características gerais, Sommerville (2011) afirma: - “De alguma forma, essas atividades fazem parte de todos os processos de *software*. Na prática, são atividades complexas em si mesmas, que incluem subatividades como validação de requisitos, projetos de arquitetura e testes unitários”.

Sendo assim, tem-se a noção de que para produzir um *software*, procura-se respeitar certos princípios, para que se obtenha ao longo do tempo de produção, um desenvolvimento uniforme, respeitando todas as suas fases e evitando maiores problemas. A seguir, serão apresentados modelos do ciclo de vida de um *software* e suas características.

1.3. MODELOS DE CICLO DE VIDA DO *SOFTWARE*

Antes de abranger modelos do ciclo de vida do *software*, deve-se entender o que engloba todo o ciclo de vida de um *software*. De acordo com Paula Filho (2001), o *software*, assim como todo produto industrial, ele também possui um ciclo de vida: projetado e criado a partir de uma necessidade percebida; desenvolvido, transformando-se em um conjunto de itens entregue a um cliente; entra em operação, sendo utilizado internamente em um processo de negócio, estando sujeito a atividades de manutenção, caso seja necessário e então, ao final de sua vida útil, sendo retirado de operação. A seguir, alguns dos principais modelos de ciclo de vida do *software*:

1.3.1. Modelo Cascata

De acordo com Paula Filho (2001) e Pressman (2011), o modelo cascata descreve uma sequência de atividades do ciclo de vida de um *software* de forma linear e sequencial. Mesmo sendo um dos modelos mais antigos que se tem conhecimento na engenharia de *software*, o modelo cascata é ainda utilizado por empresas. As principais fases do modelo cascata, segundo Sommerville (2011), são:

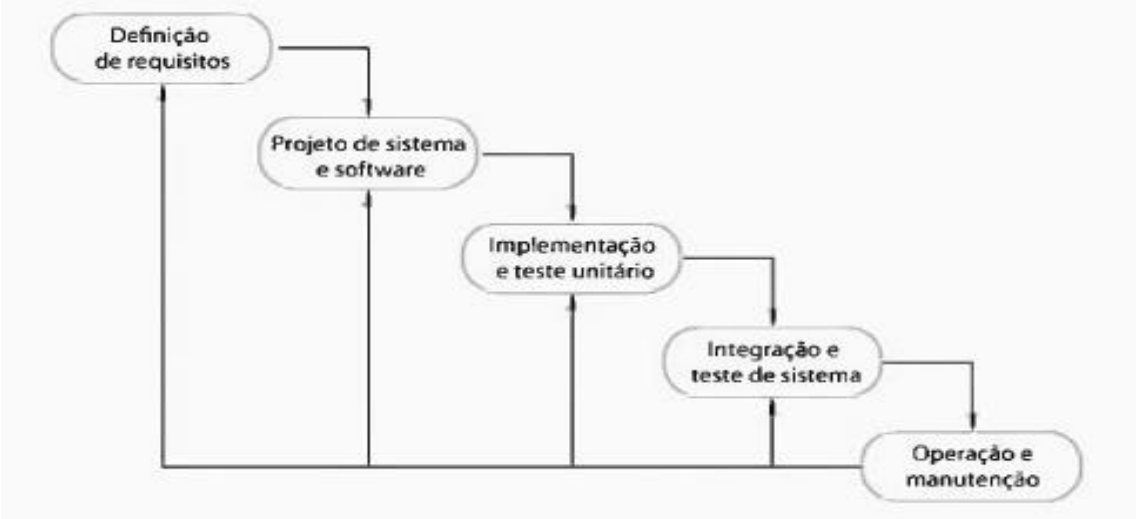
1. **Definição de Requisitos** – Fase onde são definidas as metas, serviços

prestados pelo *software* e suas restrições através de abordagens com os usuários.

2. **Projeto de sistema e *software*** – São alocados tanto os requisitos para sistemas de hardware como de *software* por meio da definição geral da arquitetura do sistema. Envolve a identificação e descrição das abstrações fundamentais do sistema de *software* e seus relacionamentos.
3. **Implementação e teste unitário** – Durante esta fase, o projeto de *software* é desenvolvido como um conjunto de programas ou unidades de programas. O teste unitário tem como finalidade verificar se cada unidade atenda a especificação proposta a ela.
4. **Integração e teste do sistema** – Realiza-se a integração das unidades do sistema e as mesmas são testadas como um sistema completo, assegurando que os requisitos do *software* tenham sido atendidos. Após os testes, o sistema é entregue ao cliente.
5. **Operação e manutenção** – A fase mais longa do ciclo de vida de um *software*. O sistema é colocado em uso. Em um contexto geral, a manutenção envolve a correção de erros não descobertos durante os estágios iniciais do ciclo de vida, com melhora da implementação das unidades do sistema e ampliação de seus serviços em resposta às descobertas de novos requisitos.

O modelo cascata e suas fases estão representados pela Figura 1 a seguir:

Figura 1 - Modelo Cascata



Fonte: Sommerville (2011)

A Figura 1 demonstra o modelo cascata e suas fases. Ao alcançar a fase de operação e manutenção, percebe-se que esta etapa conecta com todas as outras, tendo assim na fase de manutenção que repetir os passos para a projeção das implementações pendentes até que elas estejam integradas ao sistema.

Ao utilizar o modelo cascata como modelo de ciclo de vida na produção do *software*, Peters e Pedrycz (2001) ressaltam que o modelo cascata permite gerenciar cada etapa do desenvolvimento do *software* através das documentações produzidas em cada fase. Porém, Pressman (2011) afirma que, ao utilizar o modelo cascata como referência, com a complexidade que se exige na maioria dos produtos de *software* e também com as mudanças que podem ocorrer (alterações, exclusões, inserções de requisitos), estagnariam as outras fases subsequentes, deixando a equipe de outra etapa estagnada. O modelo cascata não é uma melhor opção para demonstrar para o cliente o desenvolvimento de seu produto, visto que a abordagem é mais direcionada para os projetistas envolvidos do que para o próprio cliente em si, pois ele teria uma noção de seu produto somente na penúltima etapa do ciclo de vida do *software*.

1.3.2. Modelo RUP (Rational Unified Process)

O modelo de ciclo de vida RUP, de acordo com Sommerville (2011); Vasco, Vitofht e Estante (2006), se caracteriza por ser uma metodologia iterativa, podendo ser adaptada conforme as características do produto e projetos de *software*. Em comparação com o modelo cascata, no qual suas fases estão voltadas ao aspecto técnico, as quatro fases do RUP são relacionadas aos negócios, sendo elas:

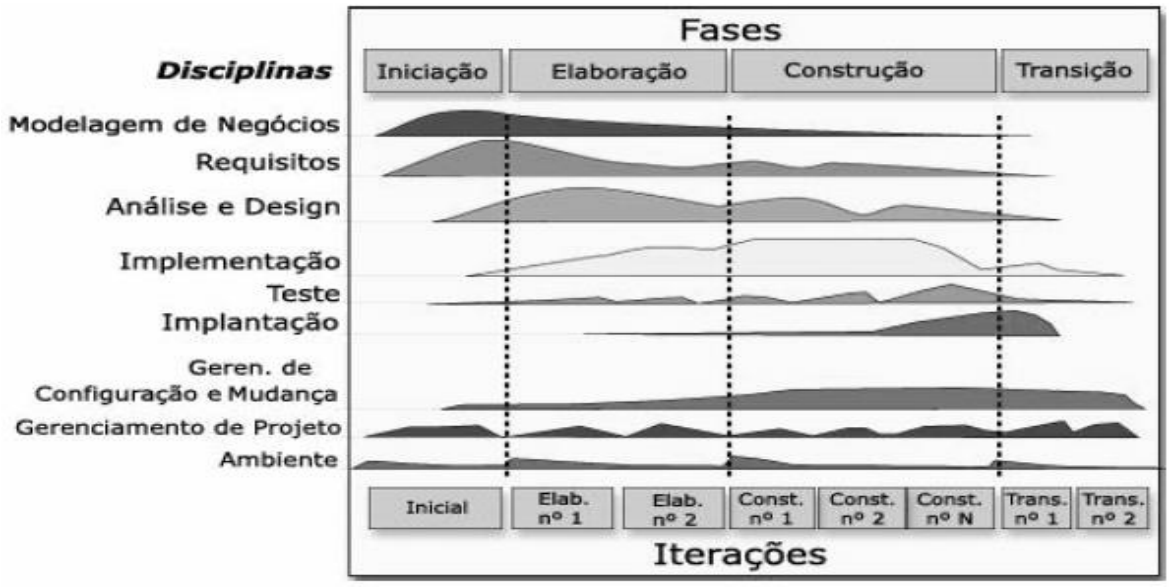
1. **Concepção ou Iniciação** – Fase onde é implementado um plano de negócio para o sistema. Identificam-se todas as entidades externas que irão interagir com o sistema e definir as iterações. Procura-se obter uma visão final do produto através de um levantamento básico feito.
2. **Elaboração** – Fase na qual se discute a dimensão do problema a ser resolvido, definindo o planejamento das atividades, realizando o levantamento dos recursos necessários e os maiores riscos do projeto. Espera-se ter ao final desta fase um plano de desenvolvimento do *software*.
3. **Construção** – Fase que envolve a implementação do *software* e o desenvolvimento do código-fonte do sistema. Durante esta fase, as partes do

sistema podem ser desenvolvidas em várias segmentações, para melhor gerenciamento. Ao final desta fase, deve-se ter a documentação do sistema e o mesmo deve se encontrar funcionando.

4. **Transição** – Fase final e de grande complexidade, pois é onde ocorre a transferência do sistema da comunidade de desenvolvimento para os usuários finais, em seu ambiente de funcionamento real, sendo feita a primeira avaliação do produto. Ocorre também o treinamento dos usuários e possíveis responsáveis pela manutenção do sistema.

A seguir a Figura 2 demonstra as fases do modelo RUP:

Figura 2 - Fases do RUP



Fonte: Vasco, Vitofht, Estante (2006)

A Figura 2 representa o modelo RUP, suas iterações e com as fases citadas no RUP, sendo destacadas também as disciplinas a serem realizadas ao utilizar o modelo RUP. Observando a figura, vê se que as disciplinas propostas se relacionam com cada fase do RUP, tendo uma intensidade diferente em cada iteração.

Após conhecer alguns modelos de ciclo de vida e suas características, na próxima subseção deste trabalho serão apresentados os conceitos da manutenção de *software*.

1.4. MANUTENÇÃO DE *SOFTWARE*

Calazans (2005) e Sommerville (2011) citam que a fase de manutenção inicia

logo quando o *software* é liberado para os usuários finais. Com a finalidade de dar continuidade ao ciclo de vida de um *software*, a etapa de manutenção se caracteriza por mudanças no *software*, a fim de atender as necessidades do usuário, seja aplicando uma correção em algum componente existente que não esteja funcionando corretamente, modificando alguma funcionalidade e também por meio de adição de novas funções.

De acordo com Santos (2007), a manutenção de *software* é equivocadamente comparada com a manutenção de hardware. A manutenção de *software* ocorre através de correção de erros e implementações de novas funções para o sistema, enquanto a manutenção do hardware consiste em substituir peças ou usar técnicas para a prolongação da vida útil das mesmas, visto que os produtos de *software* não possuem um prazo de validade estabelecido.

A manutenção do *software* não é responsabilidade exclusiva de uma pessoa. Paula Filho (2001) destaca que a formação de equipes que ficarão encarregadas das tarefas relacionadas à manutenção de *software* tende a ser problemática. Sendo assim, para que as responsabilidades referentes à manutenção sejam aplicadas de forma correta dentro de um ambiente profissional, o autor destaca seis papéis dentro da organização e seus deveres, sendo eles:

1. **Gerência de Manutenção de *Software***: Responsável pela comunicação direta com a gerência executiva e gerência de configurações de *software*. O gerente de manutenção é o principal responsável por tomar as providências necessárias para que a manutenção ocorra respeitando os procedimentos de manutenção;
2. **Gerentes de Produto**: Gerenciado pela gerência de manutenção de *software*, o gerente de produto é responsável pela manutenção de um produto ou de um grupo correlato de produtos. O gerente de produto tem a responsabilidade de garantir os procedimentos corretos de manutenção, no nível respectivo do produto, providenciar a adição ao ambiente do cliente as modificações realizadas e comunicar aos clientes quaisquer providências que devam tomar em consequências das atividades de manutenção;
3. **Representantes dos usuários**: O representante dos usuários pode ser representado por um contato dentro do ambiente do cliente ou também um membro da equipe fornecedora que seja da área de contato com clientes e usuários (marketing, por exemplo). O representante dos usuários se encarrega

de receber as notificações de problemas, da parte do usuário e realizar suas análises, para que defina se há a necessidade de manutenção ou outra providência, sendo assim, formalizar a solicitação e encaminhar aos responsáveis;

4. **Proprietários dos Itens**: Responsáveis por efetuar as devidas modificações no produto. Cabe ao proprietário dos itens emitir um parecer sobre as solicitações dos itens que são de sua responsabilidade, realizar as análises, implementar, testar e documentar as modificações necessárias e posteriormente submeter ao gerente de produto estas alterações;
5. **Comissão de Controle de Configurações do *Software***: Responsáveis pelas tomadas de decisões importantes relacionadas à gestão de configurações, em nível de produto ou de um grupo de produtos. Tem como dever representar todos os afetados pelas futuras mudanças, emitir parecer sobre solicitações de manutenção dos produtos sob sua responsabilidade, revisar e autorizar alterações nas linhas de base dos produtos e autorizar a construção de produtos a partir das linhas de base;
6. **Grupo de Gestão de Configurações de *Software***: Responsáveis pela checagem da veracidade da linha de produtos, através de auditorias; comunicação com à gerência de manutenção sobre problemas relacionados à gestão de configurações encontrados dentro dos produtos, para que providenciem as devidas correções; comunicação com à gerência executiva caso um problema não seja resolvido no nível de produto e checagem das providências adotadas pelos gerentes de produto para a resolução dos problemas encontrados.

Ainda segundo Sommerville (2011), os orçamentos voltados para a manutenção do *software* podem chegar a dois terços em comparação ao valor voltado para as etapas de desenvolvimento do projeto de *software*. Pressman (2011) afirma em sua obra que não é incomum uma empresa do setor de TI voltar 60% a 70% de seus recursos para a manutenção do *software*. Peters e Pedrycz (2001) e Sommerville (2011) classificam a manutenção do *software* em três categorias, sendo elas:

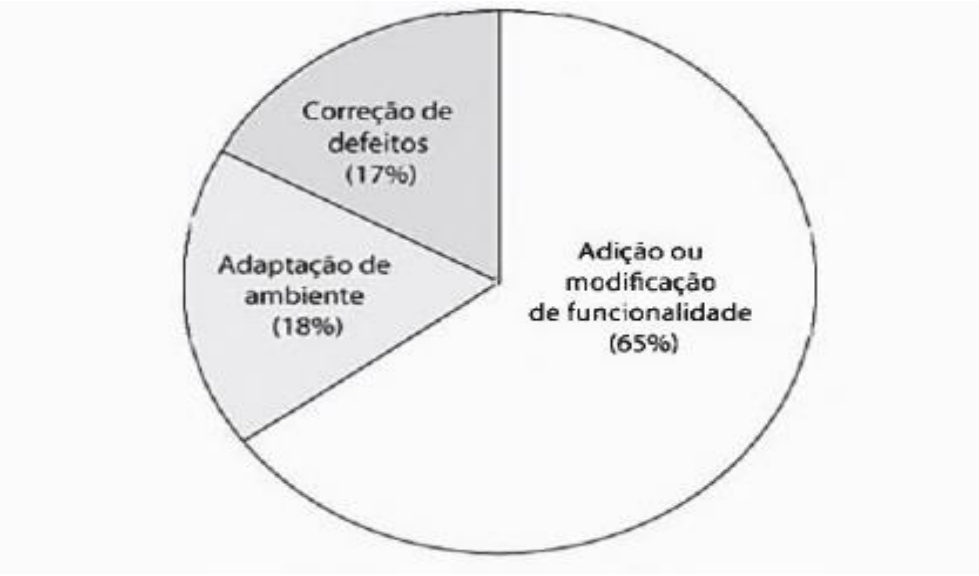
1. **Manutenção corretiva** – Referente às falhas voltadas a erros no sistema. As falhas encontradas são caracterizadas por erros lógicos, erros de design ou na

codificação do sistema.

2. **Manutenção de aperfeiçoamento** – Manutenção voltada para a aprimoração do *software* além dos requisitos originais implementados, partindo da premissa de identificação do usuário para a melhoria do sistema, podendo assim, melhorar as funcionalidades ou a eficiência do sistema.
3. **Manutenção adaptativa** – Modificações com a finalidade de acomodar mudanças do ambiente externo, migrando ou adaptando o sistema para diferentes hardwares e outros produtos de *software* (frameworks, compiladores, sistemas operacionais, bancos de dados).

Sommerville (2011) destaca que, a distribuição dos custos de manutenção pode variar de uma empresa para outra, todavia, globalmente, a manutenção corretiva não é a mais custosa, sendo a manutenção de aperfeiçoamento é a que consome mais recurso destinado à manutenção. A Figura 3 demonstra essa distribuição aproximada de custos:

Figura 3 - Distribuição do esforço de manutenção



Fonte: Sommerville (2011)

Segundo Ghezzi, Jazayeri e Mandrioli (1991), a manutenção de *software* é vista através de duas qualidades distintas (mesmo que a distinção entre as duas não seja sempre esclarecida corretamente), sendo elas: a reparabilidade e evolutividade, A reparabilidade se baseia na correção dos defeitos com uma quantidade limitada de trabalho, podendo ser considerados pequenos ajustes em todo o escopo do sistema. Um *software* com módulos bem definidos e com a utilização de uma

linguagem de alto nível é mais fácil de analisar do que um *software* monolítico com uma linguagem de baixo nível. A confiabilidade no *software* aumenta inversamente à necessidade de reparação. A evolutividade se baseia na viabilidade de aplicação das modificações realizadas em funcionalidades já existentes ou a inserção de uma nova funcionalidade no sistema. A evolutividade do *software* diminui de acordo com a inserção de novas funcionalidades ou de modificações, pois o risco de “quebrar” outras aplicações já existentes cresce.

A etapa de manutenção do *software* pode se apresentar complexa, pelo fato de depender de outros fatores que elevam sua complexidade. Shooman (1983) e Calazans (2005) destacam alguns pontos que podem dificultar a etapa da manutenção e causar frustações tanto nos profissionais envolvidos e também nos usuário finais, sendo eles:

- Falta de tempo para estudar novos materiais e obtê-los a fim de solucionar um problema;
- Mudanças na equipe responsável pela manutenção (rotatividade);
- Tempo de processamento das funções do sistema;
- Pouco interesse do usuário e falta de entendimento das funcionalidades e do sistema em geral;
- Baixa qualidade da documentação do sistema;
- Mudanças no hardware e *software* do sistema para a utilização do mesmo;
- Previsões de reais necessidades dos usuários ao solicitar manutenção;
- Adequação de especificações de *design* do sistema;
- Falta de reuniões programadas para levantamento de requisitos.

Observando a complexidade da etapa de manutenção e a sinergia que a mesma possui com todo o projeto do *software*, para que sejam evitadas complexidades que gerem transtornos, dados de processos podem auxiliar na previsão destas ações a serem tomadas, auxiliando na manutenibilidade do sistema (SOMMERVILLE, 2011). A seguir, serão apresentados a manutenibilidade de *software* e meios de previsão que possam auxiliar a manutenibilidade.

1.5. MANUTENIBILIDADE

A manutenibilidade, em geral, diz respeito à facilidade na qual um *software* pode ser alterado ou corrigido, com a finalidade de atender demandas dos usuários do sistema. Quando um sistema é manutenível, a demanda por desenvolvimento pode ser menor, na medida em que os sistemas atuais podem evoluir para atender as demandas necessárias, com isso, aumentando as solicitações por manutenção (BRUSAMOLIN, 2004).

De acordo com Santos (2007), é essencial que a manutenibilidade seja, desde o início do projeto, tratada como um requisito do sistema. Assumindo um papel cada vez mais relevante no ciclo de vida de um *software*, é de suma importância que a manutenibilidade seja pautada durante todo o desenvolvimento do projeto do *software*.

Brusamolin (2004) afirma em seu trabalho que existem quatro fatores impactantes na manutenibilidade, sendo eles:

1. **Arquitetura**: Ao investir em arquitetura de *software*, tem-se um aprimoramento, além da manutenbilidade, em outras características de qualidade. As divisões de aplicação em componentes e dos componentes em classe facilitaram a manutenção do código-fonte, diminuindo assim o esforço para modifica-lo e o potencial de inclusão de erros foi reduzido;
2. **Tecnologia**: Deve se observar o tipo de tecnologia que será utilizada para o desenvolvimento do projeto de *software*, para que a manutenção futuramente tenha um impacto negativo menor. Analisando a utilização de linguagens orientadas a objetos, observou que o uso destas linguagens impacta positivamente se tratando de alterações localizadas, pois evita uma maior degradação do código-fonte;
3. **Documentação**: Os responsáveis pela manutenção do sistema levam muito tempo para entender o funcionamento do sistema quando a documentação não se encontra disponível ou mal-elaborada;
4. **Compreensibilidade do programa**: Segundo a pesquisa de Brusamolin (2004), os desenvolvedores de *software* ficam entre 47% e 62% do tempo de trabalho tentando compreender a documentação e as lógicas relacionadas aos programas.

Após um sistema ter atingido em seu ciclo de vida a manutenção do *software*, dados do processo podem ser capazes de auxiliar na previsão da manutenibilidade.

Sommerville (2011) apresenta exemplos de métricas que podem ser usadas para avaliar se a manutenibilidade não se encontra em declínio, sendo eles:

1. **Número de solicitações de manutenção corretiva**: Quando há um aumento no relatório de falhas encontradas e *bugs*, pode ser indício que estão sendo introduzidos mais erros do que correções, causando um declínio na manutenibilidade;
2. **Tempo médio necessário para a análise de impacto**: Reflete o número de funcionalidades do programa que é atingido pela solicitação de mudança. Se esse tempo aumenta, isso implica que um número maior de funcionalidades está sendo afetado, ocorrendo uma diminuição na manutenibilidade do sistema.
3. **Tempo médio gasto para implementação de uma solicitação de mudança**: Diferente da análise de impacto, embora tendo uma correlação entre ambos. Este tempo médio refere-se a quantidade necessária de tempo para modificar o sistema e sua documentação, após a avaliação dos componentes afetados. Quanto maior for este tempo, menor a manutenibilidade do sistema.
4. **Número de solicitações pendentes**: Ao longo do tempo, um aumento neste número pode implicar em uma baixa manutenibilidade.

Tendo em vista de que a etapa de manutenção do *software* é considerada somente após a implementação do mesmo, é importante definir que a vida útil do *software* pode ser avaliada de acordo com a sua manutenibilidade, pois quanto mais defasada ela se encontrar, mais complicada será a evolução do *software*.

O próximo capítulo descreve a metodologia utilizada para alcançar os objetivos deste trabalho.

2. METODOLOGIA

O principal objetivo deste trabalho foi avaliar o quão fundamental é a etapa da manutenção do *software*. Para atender a esta finalidade, foram realizadas pesquisas bibliográficas a fim de conhecer as características do desenvolvimento do *software* e suas etapas de produção até atingir a etapa de manutenção e sua aplicação neste processo, dando continuidade ao ciclo de vida do *software*.

Através de pesquisas bibliográficas acerca das etapas de construção do *software* e principalmente da etapa de manutenção, foram analisados fatores que podem auxiliar ou dificultar o andamento do processo em geral da manutenção.

Partindo do princípio de observação da necessidade e importância de se aplicar a etapa de manutenção de *software* o mais correto possível, foi elaborado um questionário (Anexo I) com atuantes na área da computação (profissionais em geral e estudantes), com o intuito de conhecer suas experiências sobre o assunto proposto neste trabalho, para definir a melhor maneira de se obter uma manutenibilidade maior do produto de *software* e conhecer os maiores obstáculos da etapa de manutenção. Detalhes sobre a construção do questionário descritos nas subseções seguintes.

2.1. PÚBLICO ALVO DO QUESTIONÁRIO

Foram selecionados como público respondente do questionário, além de profissionais ligados ao desenvolvimento, gerenciamento e supervisão de produtos de *software* e também estudantes da área da computação, dos quais obteve-se a opinião dos respondentes acerca da manutenção de *software* e suas características. De acordo com a área de atuação do respondente e de sua experiência, o mesmo tem questões voltadas ao seu perfil de atuação e conhecimento sobre o tema retratado nesta monografia.

O questionário foi apresentado aos respondentes por intermédio de um e-mail (contendo o endereço de acesso ao questionário) e também por meio de postagens em *websites*, *blogs* e grupos em redes sociais voltados para o público da área da computação.

2.2. DESENVOLVIMENTO DO QUESTIONÁRIO

Para a disponibilização do questionário, foi utilizada a ferramenta *Google Docs*. Através das pesquisas bibliográficas levantadas neste documento, foram desenvolvidas questões que direcionam sobre o conhecimento e opinião dos respondentes acerca da manutenção de *software* em geral, como também especificando a manutenção em determinados pontos dos quais foram pesquisados para este trabalho.

As vinte e uma questões que compuseram o questionário foram desenvolvidas com base nos estudos realizados dos seguintes autores citados neste trabalho, sendo eles: Anquetil (2005), Brusamolin (2004), Calazans (2005), Paduelli (2006 e 2007), Pressman (2011), Ribeiro (2013), Silva, Matos, Souza e Moura (2010), Sommerville (2011) e Ventorin (s.d.).

Inicialmente no questionário, com o intuito de facilitar a compreensão do mesmo, foi elaborado um resumo do principal objetivo, da metodologia utilizada para criar as questões e também de uma breve apresentação do autor do questionário.

Dispostas de forma sequencial, as questões foram divididas em quatro seções: sobre a qualidade do respondente, sobre a manutenção de *software* – abordagens gerais, sobre a manutenção de *software* aplicada ao ambiente de trabalho e no desenvolvimento e sobre o gerenciamento e manutenção. Para direcionar corretamente o público-alvo de cada seção, as opções apresentadas na questão 3 serviram de base para redirecionar o respondente para a seção correta. As primeiras duas seções foram vinculadas a todo o público respondente, enquanto a terceira e a quarta seção somente foram acessadas de acordo com as respostas assinaladas anteriormente na primeira etapa. Na terceira etapa, foram excluídos os respondentes que afirmaram ser estudantes e na quarta seção, somente os que afirmaram atuar no apoio estratégico ou gerencial e também aqueles respondentes que assinalaram a opção “Outros”.

As seções do questionário tiveram o objetivo de observar um tipo específico de informação. Na primeira seção, intitulada “sobre a qualidade do respondente”, tendo da primeira à quarta questão reservada a ela, buscou-se analisar o perfil do respondente, contendo questões sobre sua experiência profissional e formação acadêmica. Na segunda seção intitulada “sobre a manutenção de *software* – abordagens gerais”, contendo a quinta, sexta e sétima questões, tendo como

conteúdo o conhecimento do respondente sobre o assunto e a abordagem do tema manutenção de *software* no ambiente acadêmico. Na seção “sobre a manutenção de *software* aplicada ao ambiente de trabalho e no desenvolvimento”, voltada somente a uma parte do público respondente - tendo da oitava à décima quarta questão destinada a esta seção é questionada a opinião dos respondentes sobre a manutenção de *software* no ambiente de trabalho, sendo englobadas também práticas do desenvolvimento nesta seção. Na quarta seção, destinada a um grupo seleto de respondentes, tem como objetivo analisar a manutenção de *software* juntamente do gerenciamento e supervisão em geral. Mais detalhes de cada seção serão descritas nas subseções adiante.

2.2.1. Primeira seção: sobre a qualidade do respondente

Nesta primeira seção se encontraram questões referentes ao perfil dos respondentes visando obter informações básicas sobre eles.

Com o intuito de conhecer a experiência profissional e acadêmica do respondente, foram compostas a primeira e a segunda questão, descritas a seguir:

- 1 – Qual seu nível de formação acadêmica?
- 2 – Há quanto tempo você se encontra exercendo qualquer atividade relacionada à área da computação?

As demais questões desta seção que compuseram o questionário (citadas a seguir) buscam conhecer a área de atuação do respondente e a quantidade de pessoas que trabalham em sua área.

- 3 - Qual sua atual função relacionada à instituição em que você se encontra vinculada? As opções para esta questão foram: Estudante; Apoio Técnico (infraestrutura e redes); Apoio operacional; Apoio Gerencial ou estratégico e “outros”.
- 4 - Qual a quantidade de profissionais, aproximadamente, vinculados à instituição em que você trabalha?

As perguntas propostas nesta seção visam conhecer melhor o respondente e suas qualificações, para que seja feita a separação correta do público das próximas seções do questionário corretamente, e também para auxiliar melhor na extração das informações desejadas.

2.2.2. Segunda seção: Sobre a manutenção de *software* – abordagens gerais

Nesta segunda seção foram distribuídas três questões, sendo elas da quinta à sétima. Todos os respondentes participaram desta seção. Inicialmente nesta seção buscou-se perguntar ao respondente o nível de conhecimento que o mesmo possui acerca do assunto, tendo na quinta questão este objetivo, possuindo como resposta uma linha sequencial de 1(pouco) a 5(muito):

- 5 – Do ponto de vista geral, qual o seu conhecimento sobre a manutenção de *software*?

As demais questões buscaram indagar o respondente sobre o tratamento dado às instituições de ensino sobre a manutenção de *software.* Segundo proposta feita por Paduelli (2007) e Anquetil (2005) de propor uma disciplina separadamente, foram elaborados as questões seis e sete, para que se confronte com as ideias do autor e definir o melhor meio de abordagem nas instituições de ensino sobre a manutenção de *software*:

- 6 – De seu ponto de vista, qual a importância dada pelas instituições de ensino ao se discutir sobre o ciclo de vida de um *software* e a etapa de manutenção de *software*? As opções desta questão foram de 1(pouco) a 5(muito).
- 7 – Para que se tenha conhecimento sobre os conceitos da manutenção de *software* e o seu desenvolvimento dentro de um ambiente profissional, em sua opinião, os ambientes de graduação deveriam propor uma disciplina separadamente para a abordagem deste assunto? As opções foram sim e não.

2.2.3. Terceira seção: Sobre a manutenção de *software* aplicada ao ambiente de trabalho e no desenvolvimento

Os respondentes desta seção foram aqueles que, em sua instituição profissional, são responsáveis por funções relacionadas ao apoio operacional, gerencial ou outra função que não tenha sido citada como opção na questão três.

Como base para a criação das sete questões referentes a esta seção, foram utilizados os estudos dos autores sobre manutenção de *software* como Sommerville (2011), Ribeiro (2013), Peters e Cedryz (2001), Calazans (2005), Brusamolin (2004) e Shooman (1983). Analisando as opiniões dos respondentes com as dos autores em suas obras, espera-se definir os meios que mais afetam tanto positivamente como negativamente a etapa de manutenção do *software* e também dos fatores que mais causam declínio na manutenibilidade de um *software*.

Como apresentado na Seção 2.4 por Peters e Pedrycz (2001) e Sommerville (2011), a manutenção de *software* é classificada em três categorias cujas informações são aferidas na seguinte questão:

- 8 – Qual o tipo de manutenção no *software* que você mais costuma realizar em seu ambiente de trabalho?

Segundo Sommerville (2011) e Shooman (1983), a manutenção do *software* pode vir a ser prejudicada caso haja um remanejamento da equipe que desenvolveu o *software* para outra equipe durante a fase de manutenção. Essas afirmações estão sendo aferidas na seguinte questão:

- 9 – Em seu ambiente de trabalho, como é realizada a manutenção de um *software*? As opções para esta questão são: A empresa não realiza a manutenção de *software*, somente o seu desenvolvimento; A manutenção é realizada pela mesma equipe do desenvolvimento; A manutenção é prioridade de uma equipe, porém podem contar com o auxílio necessário da equipe que desenvolveu o *software*;

De acordo com Ribeiro (2013) a reusabilidade de código-fonte pode impactar na etapa de manutenção de *software*. Tal afirmação encontra-se aferida na seguinte questão:

- 10 – A reusabilidade de código-fonte auxilia no processo de manutenção de *software*?

Shooman (1983), Silva e Matos e Souza e Moura (2010) e Calazans (2005) definiram em suas pesquisas, alguns pontos que podem dificultar a manutenção do *software*. Estes pontos foram aferidos na seguinte questão de múltipla escolha:

- 11 – Durante o processo de manutenção, marque as maiores dificuldades encontradas:

Segundo Brusamolin (2004), existem quatro fatores que impactam na manutenibilidade do *software*. Tais fatores são evidenciados na seguinte questão:

- 12 – Qual o fator que causa mais impacto na manutenibilidade do *software*?

Segundo Sommerville (2011), existem métricas que são capazes de auxiliar na previsão de um possível declínio da manutenibilidade do *software*. Estas métricas foram aferidas na seguinte questão:

- 13 – Qual métrica a seguir mais impacta no declínio da manutenibilidade do *software*?

Ao iniciar um ciclo de vida de um *software*, é definido o modelo de seu ciclo de vida. Embora Sommerville (2011) destaque que todos os modelos possuem características similares, a definição de um modelo de ciclo de vida de um *software* específico, pode impactar a etapa de manutenção de certa forma. Sendo assim, questão seguinte evidencia:

- 14 – O modelo de ciclo de vida selecionado ao iniciar um projeto pode impactar positivamente o projeto do *software* quando for atingida a sua etapa de manutenção?

De acordo com os estudos realizados nas pesquisas e trabalhos dos autores citados, as questões desta seção foram desenvolvidas com o intuito de comparar as opiniões dos autores com as opiniões dos respondentes.

2.2.4. Quarta seção: Sobre o gerenciamento e manutenção

Os respondentes desta questão foram todos aqueles que não afirmaram atuar como estudantes na questão 3. Como base para a criação das sete questões referentes a esta seção, foram utilizados as opiniões dos autores acerca do tema desta seção: Sommerville (2011), Paduelli (2006) e Ventorin (s.d.). Analisando as opiniões dos respondentes com os estudos levantados dos autores citados, espera-se confrontar ambas opiniões – respondentes e autores – a fim de apresentar melhores soluções para o gerenciamento de uma etapa de manutenção do *software*.

De acordo com Sommerville (2011), ao iniciar um projeto de *software*, são estipulados os recursos em cada etapa deste processo, incluindo a manutenção.

Tais recursos devem ser distribuídos cuidadosamente para que atendam as necessidades dos envolvidos com o sistema. Tais afirmações são aferidas nas seguintes questões:

- 15 – Os custos levantados e distribuídos inicialmente em um projeto de *software*, levando em conta o valor estimado para a manutenção de *software*, são alcançados?
- 16 – Como é realizada a distribuição dos custos do orçamento estimado para a manutenção em sua empresa, de acordo com as três categorias de manutenção existentes segundo Sommerville (adaptação, correção, adição /modificação)?
- 17 – O treinamento dos usuários do sistema auxilia em uma redução no custo da manutenção do *software*?
- 18 – Qual o fator mais influente – visando o custo da manutenção - na adaptação de um *software* para acomodar-se ao ambiente externo?

Segundo Paduelli (2006), a alta rotatividade de profissionais e equipes influencia na manutenibilidade do *software.* Essa afirmação é aferida na seguinte questão:

- 19 – No ponto de vista de gerenciamento de equipes, qual o peso da rotatividade da equipe responsável pela manutenção de *software* em geral?

De acordo com Ventorin (s.d.), que define o *backlog* é a incapacidade de desenvolver novos sistemas devido a fatores internos na empresa. Tais afirmações estão presentes nas seguintes questões:

- 20 – O *backlog* (demanda de novos produtos de *software* maior que a capacidade de produção) influencia na manutenção dos produtos de *software* de responsabilidade da empresa?
- 21 – Visto que o *backlog* pode ser uma das causas de uma manutenção de *software* conturbada, marque qual a principal causa do *backlog* na sua empresa.

2.3. COLETA E TRATAMENTO DE DADOS

Para a coleta de dados, foi utilizada para a criação do formulário *online* a ferramenta *Google Docs*. Utilizando esta ferramenta, tornou-se possível a criação do

formulário e seu compartilhamento pela *internet*, disponibilizando também as respostas através da *internet*.

O questionário foi divulgado através de um *link* que permitia o acesso às questões. Este *link* foi enviado através de um e-mail para atuantes na área da computação, sendo eles alunos, desenvolvedores, supervisores, gerentes, profissionais que trabalham com banco de dados, redes de computadores, serviços técnicos envolvendo *hardwares*. Além da divulgação via e-mail, teve-se a disponibilização do questionário em redes sociais para profissionais da área, incluindo também alunos e ex-alunos do curso de Ciência da Computação.

O *link* foi disponibilizado no dia 01 de setembro às 02h27min, ficando disponível para resposta até às 23h59min do dia 20 de outubro de 2017. Nesse período foram coletadas 80 (oitenta) respostas.

As respostas automaticamente foram enviadas para uma planilha *online* disponibilizada pelo *Google Docs*. A partir desta planilha, foi possível estudar cada resposta separadamente. Após a exportação, as respostas foram visualizadas através das ferramentas disponibilizadas no *Microsoft Office*, os dados foram organizados de acordo com o que cada questão propôs, sendo também organizado de acordo com o grupo de respondentes citados na questão três do questionário (Anexo I).

Os resultados foram disponibilizados através de gráficos, para facilitar o entendimento das respostas. Estes gráficos podem ser observados na próxima seção.

3. RESULTADOS

Serão apresentados nesta seção os resultados das respostas obtidas no questionário. Foram no total 80 respondentes, dos quais a metade dos entrevistados afirmou trabalhar na área operacional em suas respectivas empresas. Para que a compreensão dos resultados seja demonstrada de forma clara, as respostas serão exibidas através de gráficos. Nas subseções seguintes constarão os gráficos e suas respectivas descrições.

3.1. EXTRAÇÃO DOS RESULTADOS

Durante o processo de análise das respostas obtidas, uma das 80 entrevistas foi considerada inválida ao estudo. O motivo pela invalidação da entrevista ocorreu pelo fato do respondente não corresponder ao público-alvo do questionário por ter respondido a questão sobre sua função relacionada à instituição na qual ele trabalha como “auxiliar de escritório”. Sendo assim, sua entrevista foi desconsiderada e as respostas analisadas à seguir não contêm a resposta do entrevistado em questão e com isso, serão contados somente as 79 respostas válidas.

As respostas serão organizadas de acordo com as quatro seções do questionário, sendo elas: Primeira seção: Sobre a qualidade do respondente, segunda seção: Sobre a manutenção de software – abordagem geral, terceira seção: sobre a manutenção de *software* aplicada ao ambiente de trabalho e no desenvolvimento, quarta seção: sobre o gerenciamento e manutenção. As questões serão apresentadas na mesma ordem que foram disponibilizadas no questionário.

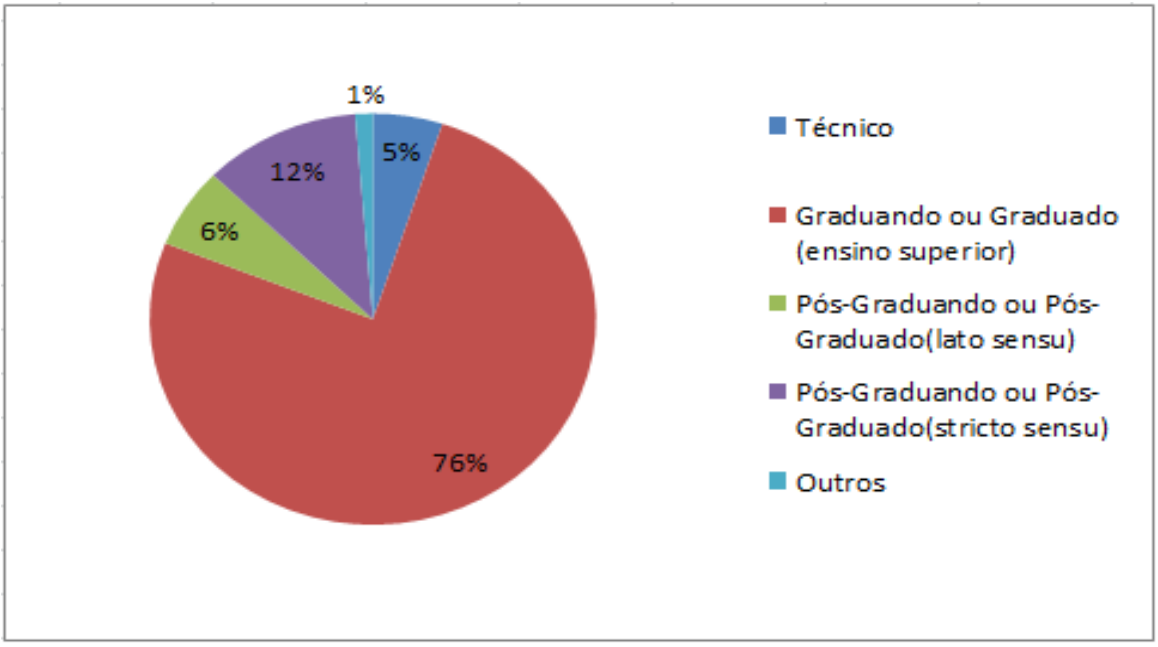
3.1.1. Primeira seção: Sobre a qualidade do respondente

Serão descritos nesta seção os resultados obtidos das respostas da primeira etapa do questionário. Cada questão terá seu gráfico próprio contendo os resultados. Esta seção contém 4 perguntas com o objetivo de conhecer os entrevistados, voltadas à formação profissional do respondente. Todas as questões tiveram 79 respondentes ao todo. Cada subtópico desta seção faz referência a uma questão do questionário.

3.1.1.1. Qual seu nível de formação acadêmica?

Esta primeira questão visou conhecer a formação acadêmica do respondente. Por meio dela, será possível aferir mais adiante as soluções apontadas por cada um destes grupos. Os resultados analisados posteriormente são referentes ao gráfico à seguir.

Gráfico 1 – Formação Acadêmica



Fonte: Próprio autor

A maioria dos entrevistados, sendo representados por aproximadamente 76%, precisamente por 60 pessoas, declararam estarem graduados no ensino superior ou em andamento. 9 dos entrevistados, aproximadamente 12% são pós-graduados (*stricto sensu*) ou estão em processo de conclusão. Aproximadamente 6% dos entrevistados, 5 deles, são pós-graduados (*lato sensu*) ou estão em processo de conclusão. 4 entrevistados (5%) declaram ter formação técnica. Apenas um entrevistado informou outro tipo de formação acadêmica, sendo ela “Ensino Médio”.

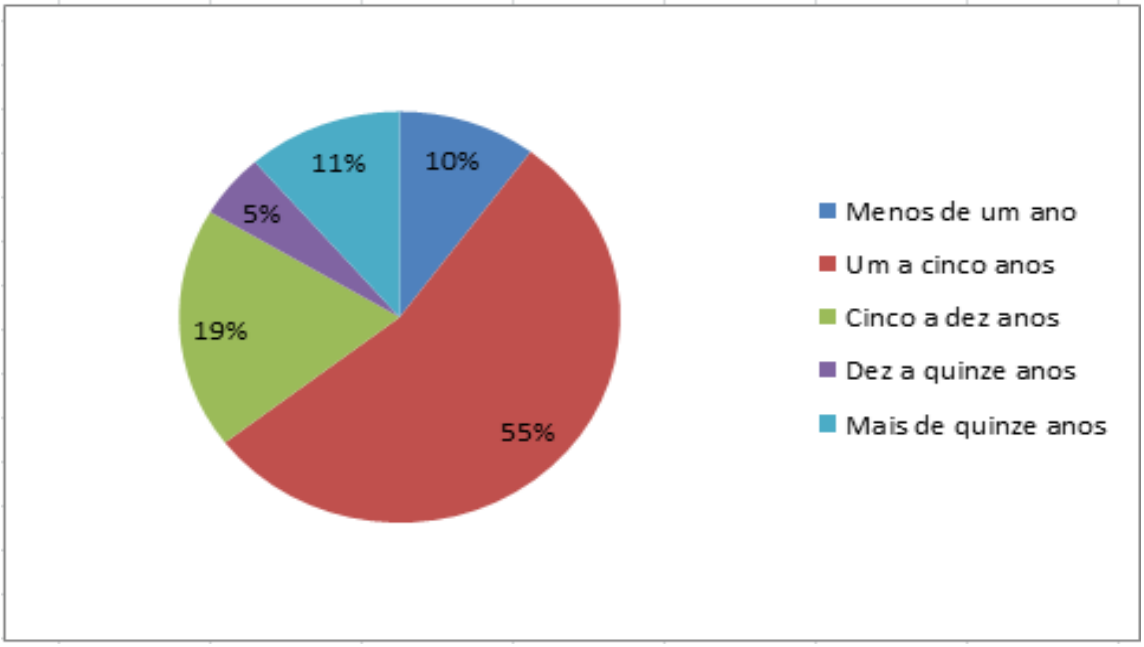
Tendo a ampla maioria dos entrevistados de formação de graduação e pós-graduação, totalizando aproximadamente 94% dos entrevistados, é possível afirmar que este grupo de entrevistados possui maturidade em relação ao que se trata de manutenção de *software* e suas características. É também possível afirmar que o

questionário atingiu o público-alvo desejado do qual se espera uma base de conhecimento sobre a manutenção de *software*.

3.1.1.2. Há quanto tempo você se encontra exercendo qualquer atividade relacionada à área da computação?

Como a experiência é de suma importância na absorção de conhecimento e de boas práticas profissionais, esta questão busca conhecer o tempo de atuação dos respondentes. Os resultados analisados posteriormente são referentes ao gráfico à seguir.

Gráfico 2 – Tempo de atuação



Fonte: Próprio autor

A maioria dos entrevistados afirmou trabalhar com qualquer atividade relacionada à computação em um período de um a cinco anos, sendo aproximadamente 55% dos entrevistados (43 pessoas). 19% dos entrevistados afirmaram trabalhar entre cinco a dez anos com computação (15 pessoas). 9 entrevistados que afirmaram trabalhar a mais de quinze anos com computação, sendo aproximadamente 11% do total de entrevistados. Os entrevistados com menos tempo de serviço compreendem a uma taxa de aproximadamente 10% (8 pessoas). Somente aproximadamente 5% dos entrevistados (4 pessoas) afirmaram trabalhar de dez a quinze anos.

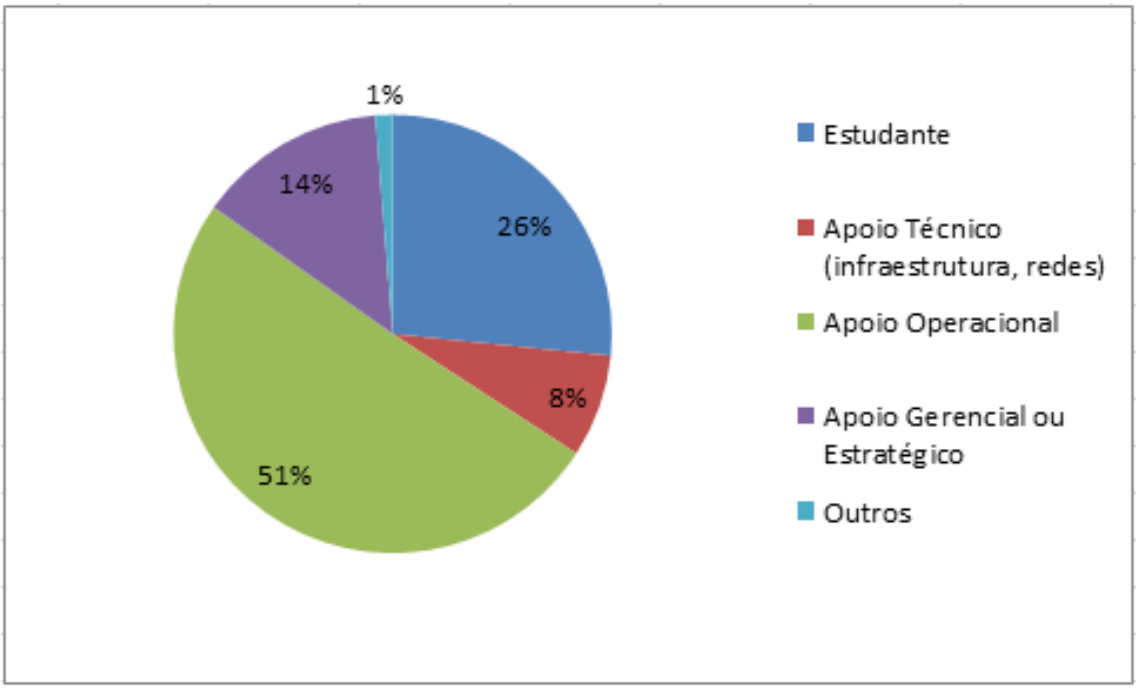
Sendo assim, é possível afirmar que a maioria dos respondentes ainda não possui longa experiência exercendo alguma atividade relacionada à computação embora demonstrem imensa percepção sobre o assunto em questão.

3.1.1.3. Qual sua atual função relacionada à instituição em que você se encontra vinculada?

Esta questão tem como objetivo separar os grupos dos respondentes. Dependendo da resposta a esta questão, foi possível definir se o respondente se encaixaria no perfil da terceira e quarta seção de perguntas.

Esta questão visa analisar qual a real ligação entre o respondente e a manutenção de *software.* Os resultados analisados posteriormente são referentes ao gráfico à seguir.

Gráfico 3 – Função desempenhada



Fonte: Próprio autor

A maioria dos entrevistados (51% aproximadamente, 40 pessoas no total) afirmaram, desempenhar funções relacionadas ao apoio gerencial da instituição. 21 pessoas (26% aproximadamente) declararam que a função desempenhada na instituição é de estudante. Aproximadamente 14% (11 pessoas exatamente) afirmaram desempenhar funções relacionadas ao apoio gerencial ou estratégico. 6 pessoas (8% aproximadamente) trabalham desempenhando funções voltadas ao

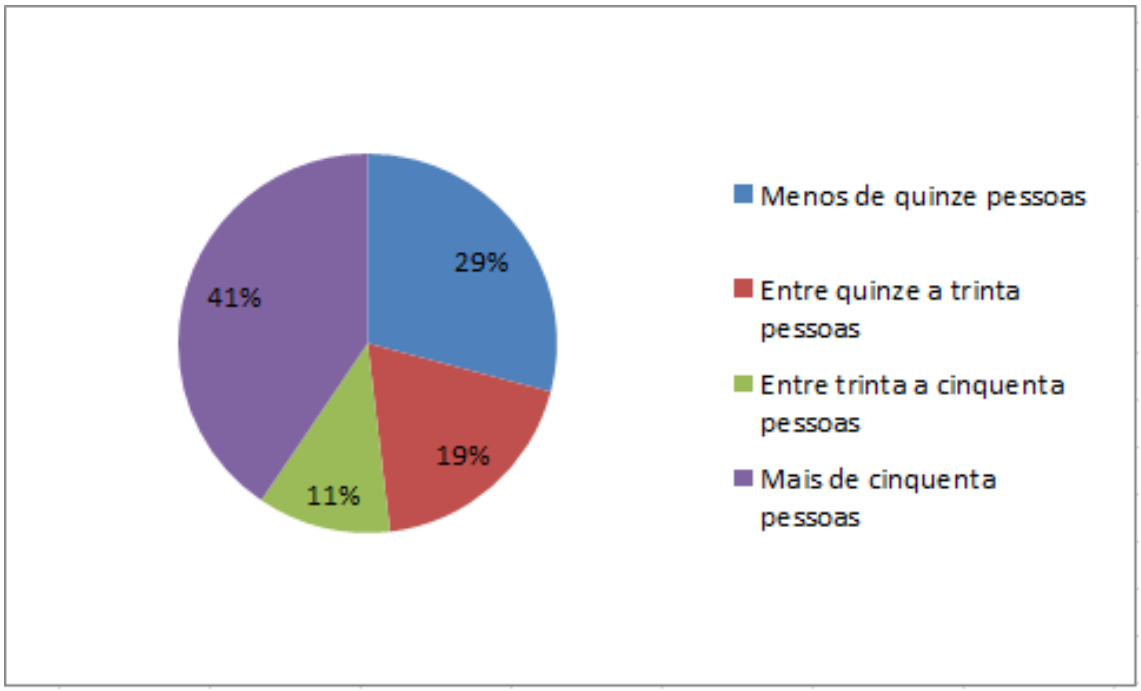
apoio técnico. Somente 1 pessoa declarou desempenhar função diferente das disponibilizadas, afirmando desempenhar a função de “instrutor de ensino profissional”.

Analisando as respostas obtidas para esta questão através do Gráfico 3, é correto afirmar que o questionário alcançou o público desejado para que se obtenha respostas nas questões específicas da terceira e quarta seção, voltadas para respondentes com atividades voltadas para o apoio operacional, gerencial e estratégico.

3.1.1.4. Qual a quantidade de profissionais, aproximadamente, vinculados à instituição em que você trabalha?

Para que se tenha uma noção da importância de haver sistemas em bom funcionamento em uma empresa, a quantidade de funcionários em uma instituição a torna dependente cada vez mais de procedimentos ágeis e confiáveis, substituindo alguns procedimentos rudimentares que levariam mais tempo para serem efetuados. Os resultados analisados posteriormente são referentes ao gráfico à seguir.

Gráfico 4 – Quantidade de profissionais vinculados à instituição



Fonte: Próprio autor

41% aproximadamente dos respondentes (32 pessoas), afirmaram que mais de cinquenta pessoas estão vinculados à instituição na qual eles se encontram. Aproximadamente 29% (23 pessoas), disseram haver menos de quinze pessoas

vinculadas à instituição. 15 entrevistados (19% no total) afirmaram que são entre quinze e trinta profissionais vinculados à instituição onde trabalham. 9 entrevistados (11% aproximadamente) disseram haver entre trinta a cinquenta pessoas vinculadas na instituição em que eles atuam.

Com os resultados levantados nesta questão, apresentados pelo Gráfico 4, percebe-se que a maioria dos entrevistados afirmam estar vinculados em uma empresa com um número considerável de funcionários (mais de 50), porém, é válido observar que os resultados se encontram bem distribuídos, com uma quantidade considerável de entrevistados exercendo algum tipo de função em empresas de pequeno e médio porte.

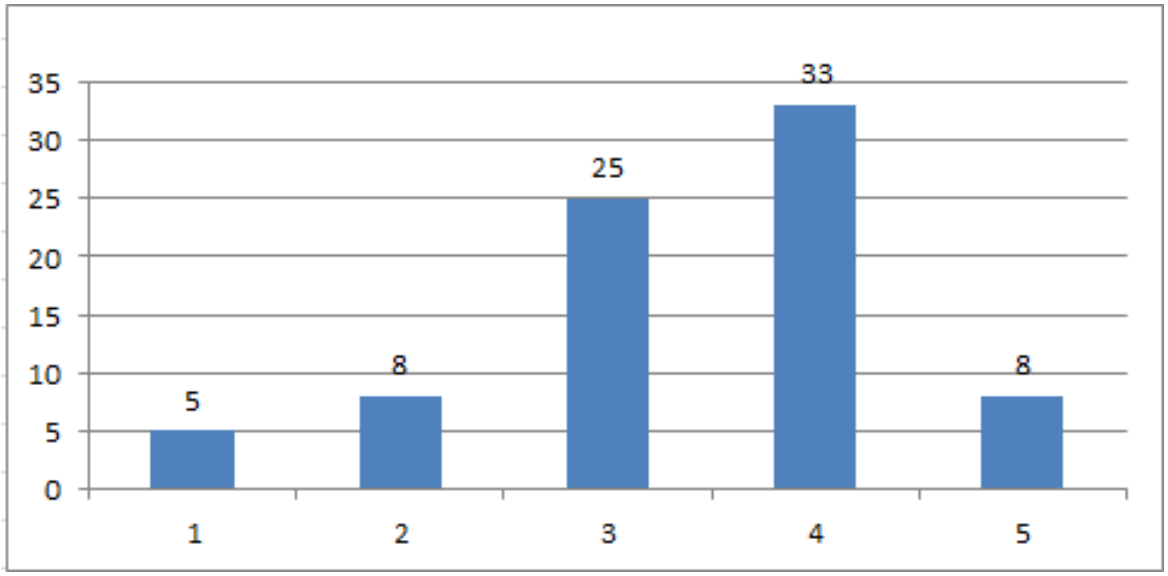
3.1.2. Segunda seção: sobre a manutenção de *software* - abordagens gerais

Os resultados descritos nesta seção são referentes a segunda seção do questionário. Cada questão terá seu gráfico próprio contendo os resultados. Esta seção contém 3 perguntas com os objetivos de abordar o respondente sobre seu nível de conhecimento do assunto proposto neste trabalho e da abordagem do assunto no meio acadêmico. Todas as questões tiveram 79 respondentes ao todo. Cada subtópico desta seção referencia a uma questão do questionário.

3.1.2.1. Do ponto de vista geral, qual o seu conhecimento sobre a manutenção de *software*?

Esta questão tem como objetivo indagar os respondentes qual o grau de conhecimento deles sobre manutenção de *software*. Nesta questão, as opções dadas ao respondente foram de 1 (pouco ou nenhum conhecimento) a 5 (muito conhecimento). Os resultados analisados posteriormente são referentes ao gráfico à seguir.

Gráfico 5 – Manutenção de *software*: Conhecimento geral



Fonte: Próprio autor

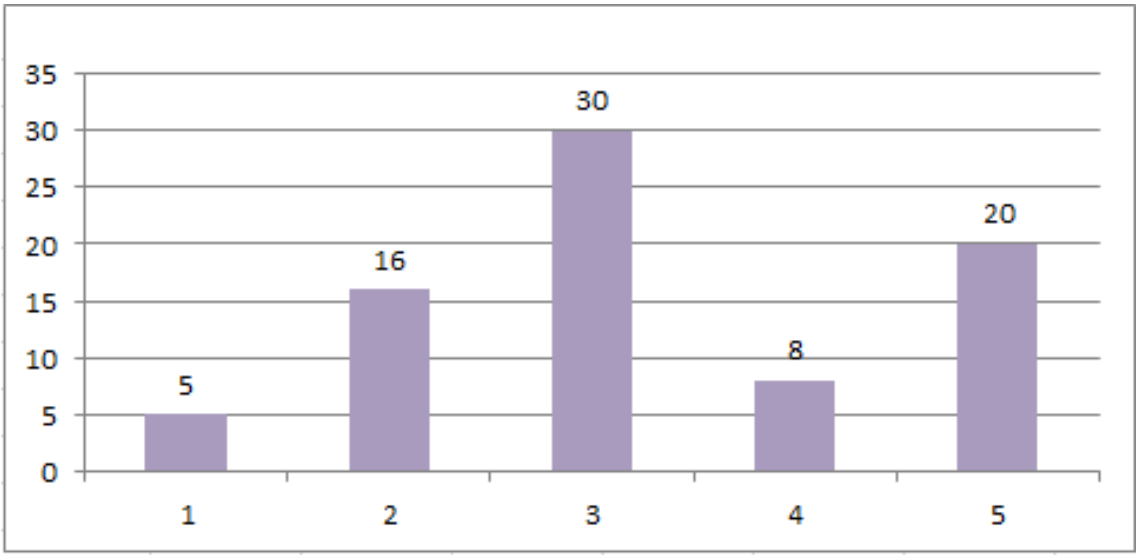
A maioria dos respondentes (33 pessoas) escolheram a opção 4. A opção 3 foi selecionado por 25 pessoas. 8 pessoas marcaram a opção 5, mesma quantidade de pessoas que marcaram a opção 2. Somente 5 pessoas marcaram a opção 1.

Pode-se observar nesta questão que a maioria dos respondentes afirmou possuir um bom grau de conhecimento sobre o assunto proposto neste trabalho. Com isso, pode se afirmar que o assunto abordado no questionário não é desconhecido para a maioria dos respondentes.

3.1.2.2. De seu ponto de vista, qual a importância dada pelas instituições de ensino ao se discutir sobre o ciclo de vida de um *software* e a etapa de manutenção do *software*?

Esta questão buscou conhecer a opinião dos respondentes perante a importância que as instituições de ensino dão ao discutir sobre o ciclo de vida e a etapa de manutenção de um *software*. As opções dadas a esta questão foram de 1 a 5: 1 representa uma baixa importância enquanto 5 representa uma alta importância dada pelas instituições de ensino sobre os assuntos abordados pela questão. Os resultados analisados posteriormente são referentes ao gráfico à seguir.

Gráfico 6 – Instituições de ensino – ciclo de vida e manutenção de *software*



Fonte: Próprio autor

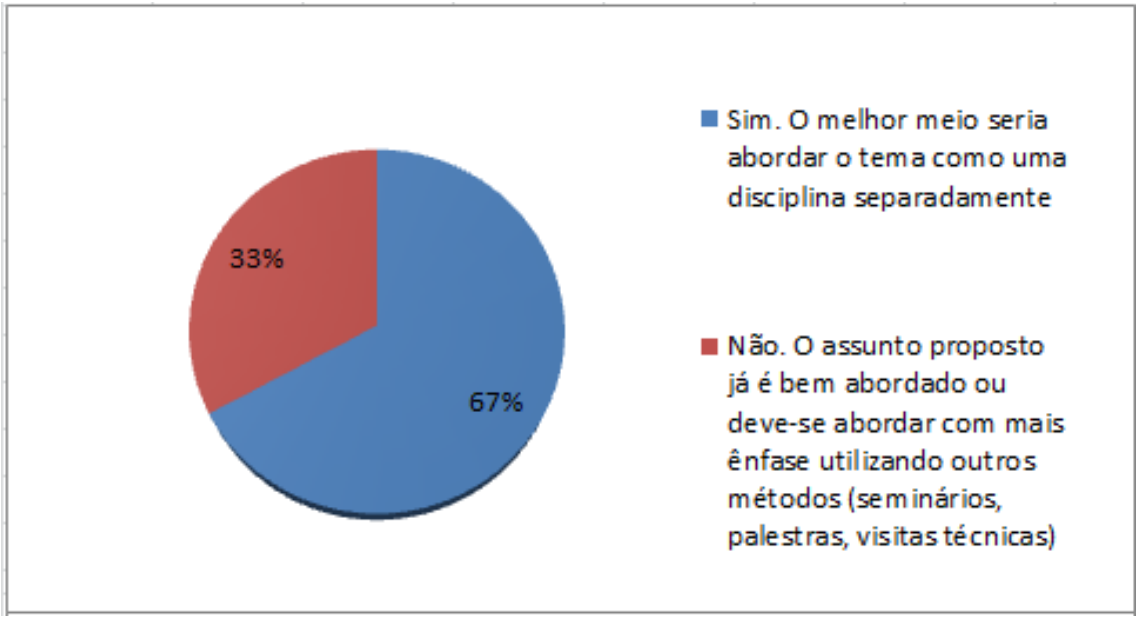
A maior parte dos entrevistados (30 respondentes) assinalaram a opção 3. A segunda maioria, representada por 20 pessoas, escolheram a opção 5. Outros 16 respondentes assinalaram a opção 2. 8 pessoas marcaram a opção 4 e somente 5 pessoas optaram pela opção 1.

Pode-se aferir que a maioria dos respondentes, representados pelos que assinalaram as opções 1, 2 e 3 (51 pessoas), afirmam que as instituições de ensino dão uma média (representado por 30 pessoas que assinalaram a opção 3) a baixa importância (representados por 21 pessoas que assinalaram as opções 1 e 2) dado aos assuntos abordados na questão 6, enquanto outros 28 respondentes (pessoas que assinalaram as opções 4 e 5) afirmam que as instituições dão uma alta relevância a estes assuntos. Portanto, é possível afirmar que a opinião dos respondentes foi de encontro com a proposta feita por Anquetil (2005) e Paduelli (2007), sobre propor uma disciplina separadamente para tratar os assuntos relacionados à manutenção do *software*.

3.1.2.3. 7 – Para que se tenha conhecimento sobre os conceitos da manutenção de *software* e o seu desenvolvimento dentro de um ambiente profissional, em sua opinião, os ambientes de graduação deveriam propor uma disciplina separadamente para a abordagem deste assunto?

Esta questão procurou abordar os entrevistados sobre a proposta dos ambientes de graduação abordar a manutenção de *software* e suas diretrizes como uma disciplina separadamente. Os resultados analisados posteriormente são referentes ao gráfico à seguir.

Gráfico 7 – Proposta de uma disciplina para manutenção de *software*



Fonte: Próprio autor

67% aproximadamente dos entrevistados (53 pessoas) disseram que a manutenção de *software* deveria ser abordada como uma disciplina separadamente. 26 entrevistados (33% aproximadamente), afirmaram que não é necessário abordar a manutenção de *software* em uma disciplina separadamente.

Com base nestas informações, a grande maioria dos entrevistados afere que abordar a manutenção de *software* e seus conceitos como uma disciplina separadamente é uma melhor maneira para obter conhecimento sobre a mesma nos ambientes de graduação.

3.1.3. Terceira seção: Sobre a manutenção de *software* aplicada ao ambiente de trabalho e no desenvolvimento:

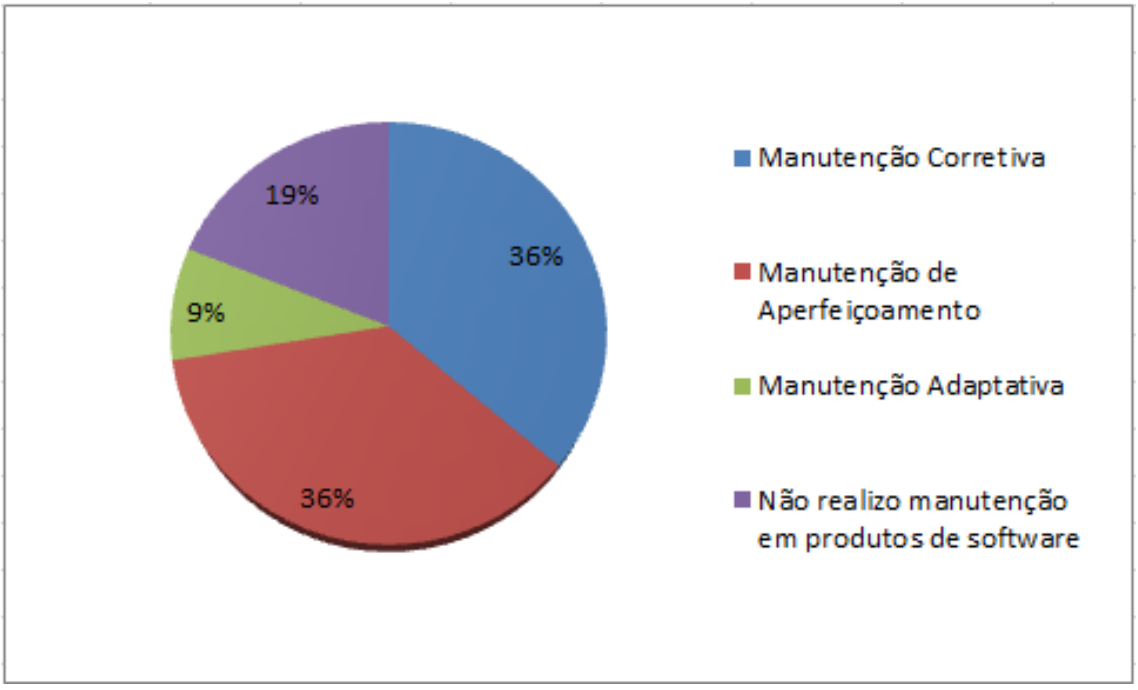
Os resultados descritos nesta seção são referentes a terceira seção do questionário. Cada questão terá seu gráfico próprio contendo os resultados. Esta seção contém 7 perguntas com os objetivos de abordar o respondente sobre a aplicação da manutenção de *software* aplicada no desenvolvimento dos produtos de *software* e no ambiente profissional. Foram selecionados os respondentes para esta seção aqueles que afirmaram atuar no apoio operacional, estratégico, gerencial ou

em outras funções (vide questão 3). Viu-se necessário selecionar somente estes respondentes nesta etapa, pois a mesma possui um conteúdo específico para quem trabalha diretamente com o ambiente de desenvolvimento dos produtos de *software*. Todas as questões tiveram 58 respondentes ao todo. Cada subtópico desta seção faz referência a uma questão do questionário. Os resultados analisados posteriormente são referentes ao gráfico à seguir.

3.1.3.1. Qual o tipo de manutenção no *software* que você mais costuma realizar em seu ambiente de trabalho?

Esta questão tem como objetivo saber qual o tipo de manutenção o respondente realiza com maior frequência em seu ambiente de trabalho.

Gráfico 8 – Tipo de manutenção realizada



Fonte: Próprio autor

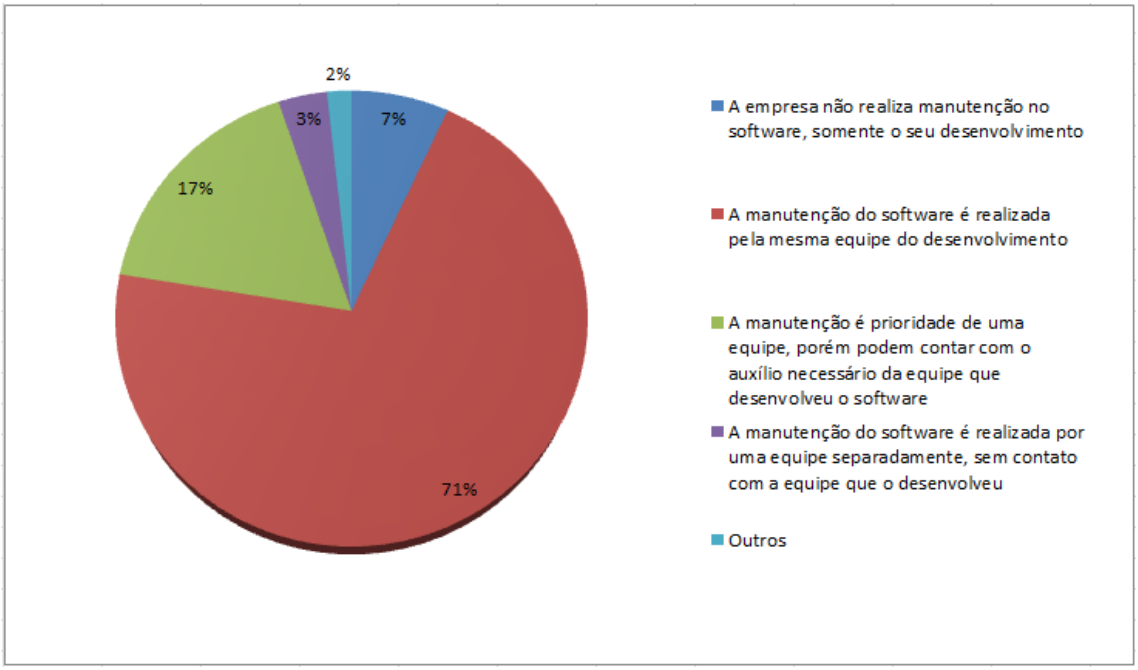
A maioria dos respondentes afirmou que a manutenção corretiva e a manutenção de aperfeiçoamento são as mais realizadas, tendo cada uma destas opções selecionadas por 21 pessoas (36% aproximadamente). 11 pessoas não realizam manutenção em produtos de *software* e somente 5 respondentes disseram que a manutenção adaptativa é realizada com mais frequência em seu ambiente de trabalho.

Com base nestas informações levantadas, pode-se aferir que as manutenções para correções de erros e de melhorias no sistema são as mais recorrentes nos produtos de *software*.

3.1.3.2. Em seu ambiente de trabalho, como é realizada a manutenção de um *software*?

Nesta questão procura-se saber, de maneira geral, qual o processo de realização da manutenção dos produtos de *software* na empresa que o respondente esteja atuando. Os resultados analisados posteriormente são referentes ao gráfico à seguir.

Gráfico 9 – Realização de manutenção de *software* na empresa



Fonte: Próprio autor

A ampla maioria dos respondentes (71% aproximadamente, sendo 41 pessoas no total) disse que a mesma equipe que desenvolveu o sistema realiza a sua manutenção. Outras 10 pessoas (17% aproximadamente) afirmam que a manutenção é prioridade de uma equipe, porém contam com o auxílio da equipe que desenvolveu o *software*. 4 pessoas (aproximadamente 7%) dizem que a empresa realiza somente o desenvolvimento do *software*. 2 pessoas (3% aproximadamente) afirmam que a equipe responsável pela manutenção de *software* não possui contato com a equipe que o desenvolveu. Somente 1 pessoa optou pela opção outros,

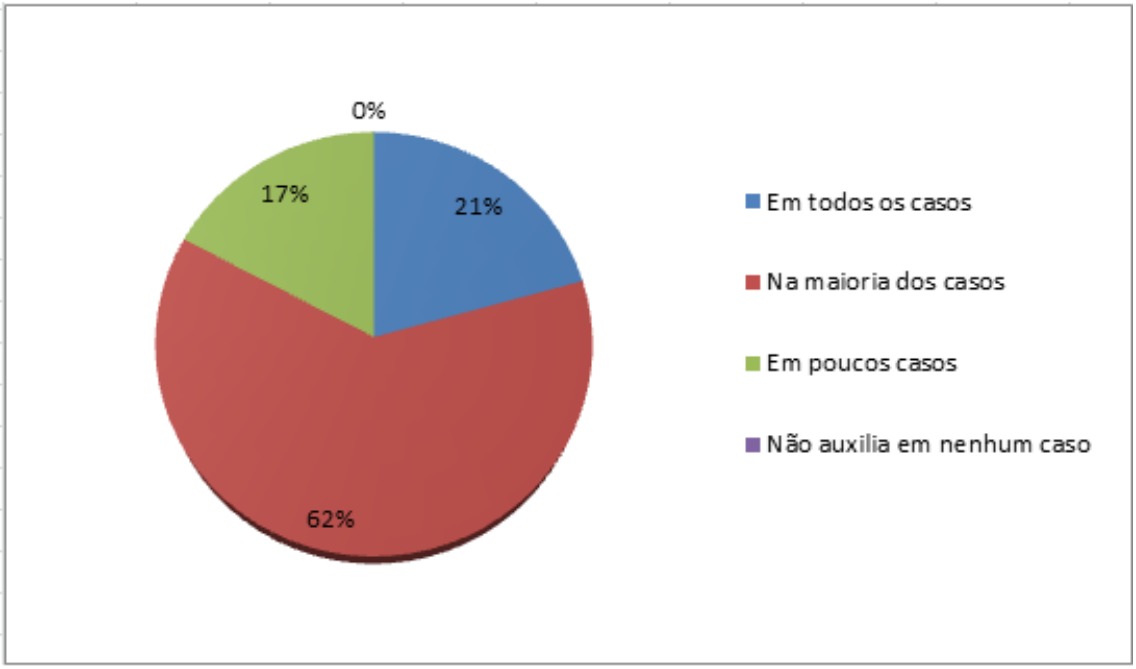
relatando que “todas equipes de desenvolvimento deveriam poder realizar manutenção em qualquer *software*”.

O fato da maioria dos respondentes afirmarem que a manutenção dos produtos de *software* na empresa em que atuam é realizada pela mesma equipe que o desenvolveu, pode-se afirmar que os responsáveis possuem conhecimento da estrutura do sistema e de suas características, porém, dependendo das outras funções a eles vinculadas, é provável que isto gere um atraso nas tarefas relacionadas à manutenção dependendo das prioridades definidas para o funcionário em um determinado período de tempo.

3.1.3.3. A reusabilidade de código-fonte auxilia no processo de manutenção de *software*?

Esta questão busca saber a opinião do respondente sobre a utilização de procedimentos que envolvem a reusabilidade do código-fonte auxiliam na manutenção de *software*. Os resultados analisados posteriormente são referentes ao gráfico à seguir.

Gráfico 10 – Reusabilidade e manutenção de *software*



Fonte: Próprio autor

A maioria dos entrevistados (62% aproximadamente, 36 pessoas no total) relatam que a reusabilidade pode auxiliar na maioria dos casos na manutenção de um produto de *software*. 12 pessoas (21% aproximadamente) aferem que a reusabilidade pode auxiliar em todos os casos de manutenção do *software.* 17%

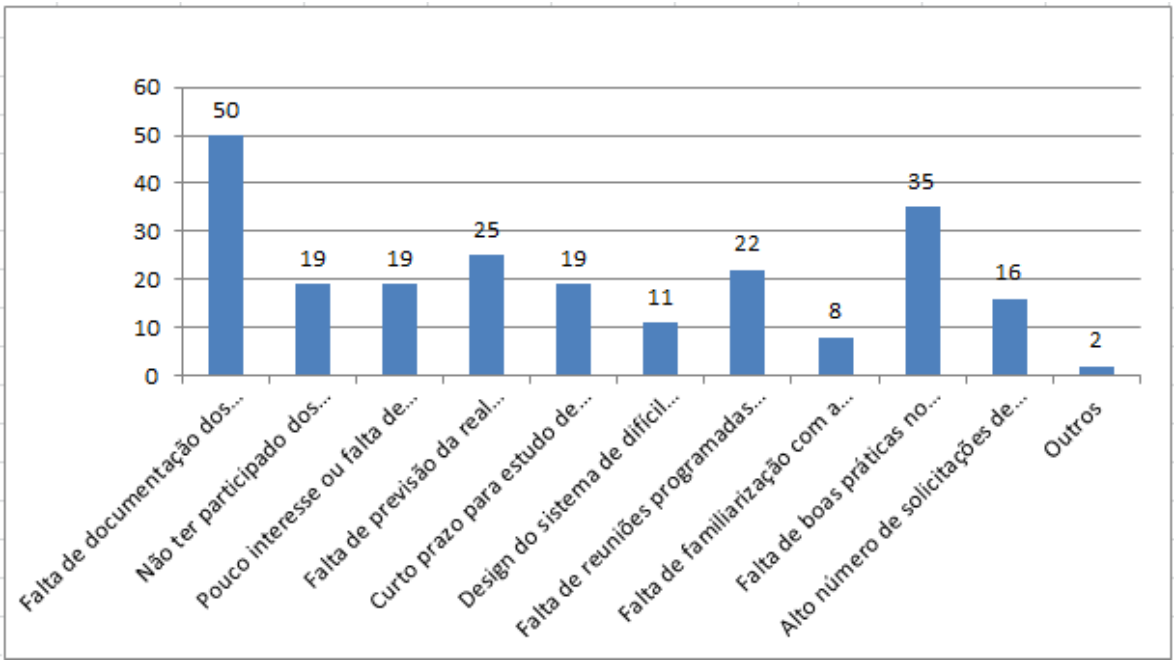
aproximadamente dos respondentes, representados no total por 10 pessoas, afirmam que a reusabilidade pode auxiliar na manutenção em poucos casos. Nenhum respondente assinalou que a reusabilidade não auxilia a manutenção de *software*.

De acordo com os respondentes, a reusabilidade de código-fonte auxilia positivamente na maioria dos casos quando se trata da manutenção de *software*, havendo concordância com os estudos feitos por Ribeiro (2013). Para que seja analisado qual conceito específico da reusabilidade de *software* será necessário um estudo específico com suas aplicações visando a manutenção de *software*.

3.1.3.4. Durante o processo de manutenção, marque as maiores dificuldades encontradas:

Para esta questão, foi possível o respondente selecionar mais de uma opção como resposta. Sendo assim, a soma das respostas ultrapassará a totalidade de respondentes nesta questão (58 respondentes ao total). Esta questão busca conhecer as maiores dificuldades encontradas para que se tenha uma manutenção de *software* consistente. Os resultados analisados posteriormente são referentes ao gráfico à seguir. Para melhor visualização das opções presentes, visualize-as no Anexo I no final deste trabalho.

Gráfico 11 – Principais dificuldades na manutenção



Fonte: Próprio autor

A grande maioria dos entrevistados (50 pessoas) declarou que a falta de documentação dos processos e funcionalidades do *software* é a principal dificuldade encontrada na manutenção de *software*. A segunda opção mais assinalada pelos respondentes (35 pessoas) foi a falta de boas práticas no desenvolvimento do código-fonte, seguido pela falta de previsão da real necessidade do usuário (selecionado por 25 pessoas).

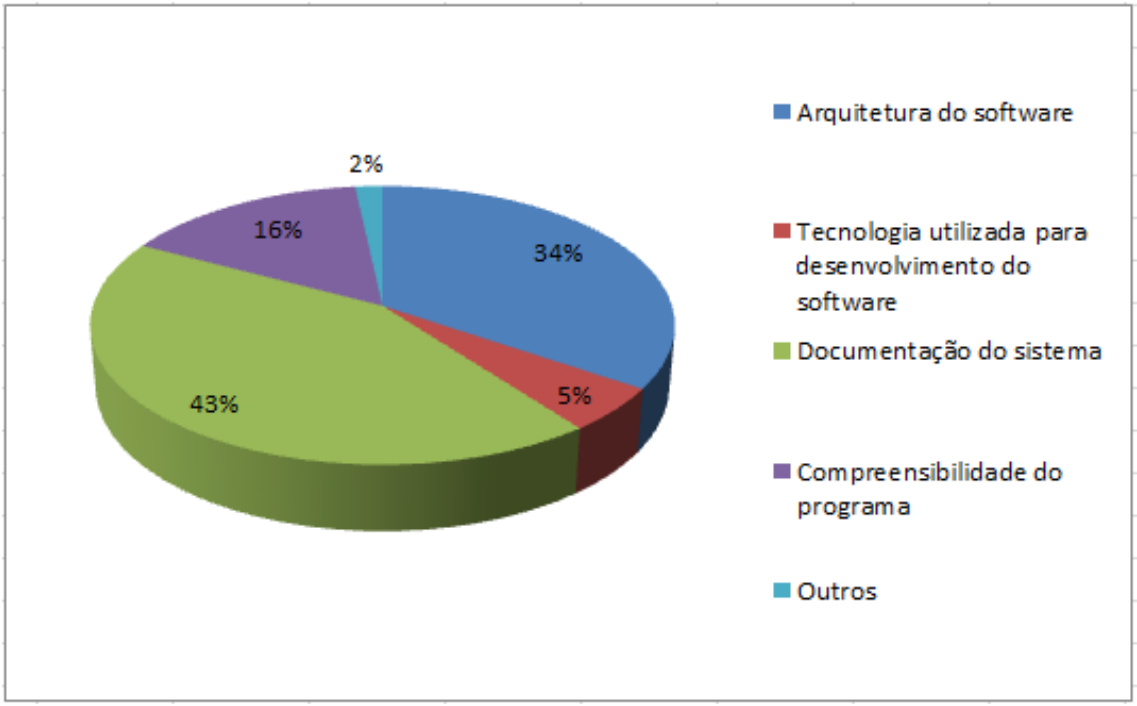
A quarta opção mais selecionada, assinalada por 22 pessoas, foi a falta de reuniões programadas para analisar as necessidades reais do usuário. Três opções foram assinaladas por 19 pessoas: não ter participado dos processos de desenvolvimento do *software*, pouco interesse ou falta de entendimento do usuário no funcionamento do *software* e o curto prazo para estudo de materiais e novos processos a fim de se obter a melhor solução para o problema. 16 respondentes escolheram o alto número de solicitações de manutenção em espera. 11 pessoas afirmaram que o design do sistema de difícil entendimento é uma das maiores dificuldades. 8 pessoas apontaram a falta de familiarização com a linguagem utilizada no desenvolvimento do *software*. Somente 2 pessoas descreveram outras dificuldades além das previamente descritas, sendo elas a falta de clareza no objetivo que se deseja alcançar e a falta de utilização de uma arquitetura concretizada no mercado e de um bom padrão de design.

Com base nestas informações, podemos aferir que, das dificuldades levantadas por Shooman (1983), Silva et al (2010) e Calazans (2005), os principais obstáculos encontrados na manutenção do *software*, de acordo com os respondentes, estão correlacionados diretamente com a ausência da documentação do sistema para melhor entendimento das funcionalidades, dos processos e do acompanhamento das mudanças que foram realizadas ao longo do tempo; a falta de uma padronização e boas práticas de desenvolvimento causam atrasos em relação ao entendimento da codificação; a análise real das necessidades dos usuários pode ser afirmada pela falta de requisitos ou pelo levantamento de requisitos errados. Para que isso não ocorra, é necessário que a captação de requisitos seja mais detalhada e que aconteçam reuniões frequentemente para as definições das necessidades dos usuários.

3.1.3.5. Qual o fator que causa mais impacto na manutenibilidade do *software*?

Esta questão procura aferir quais dos fatores que foram apresentados nas opções desta questão mais impactam na manutenibilidade de um *software*. Nesta questão o respondente teve a opção de determinar outro fator além dos previamente citados.

Gráfico 12 – Fatores impactantes na manutenibilidade



Fonte: Próprio autor

A maioria dos entrevistados, representando 43% no total (25 pessoas), afirmaram que a documentação do sistema é o fator mais impactante na manutenibilidade do sistema. A arquitetura de *software* foi apontada por 36% dos respondentes (20 pessoas) como o fator mais impactante. 9 pessoas (aproximadamente 16%) descrevem a compreensibilidade do programa como o principal fator para impactar a manutenibilidade de um sistema. Somente 5% aproximadamente dos respondentes (3 pessoas no total) assinalaram que a tecnologia utilizada para desenvolvimento do *software* causa um impacto negativo na manutenibilidade. Somente 1 respondente assinalou a opção “Outros”, especificando que a maior causa de impacto na manutenibilidade são as más práticas de desenvolvimento.

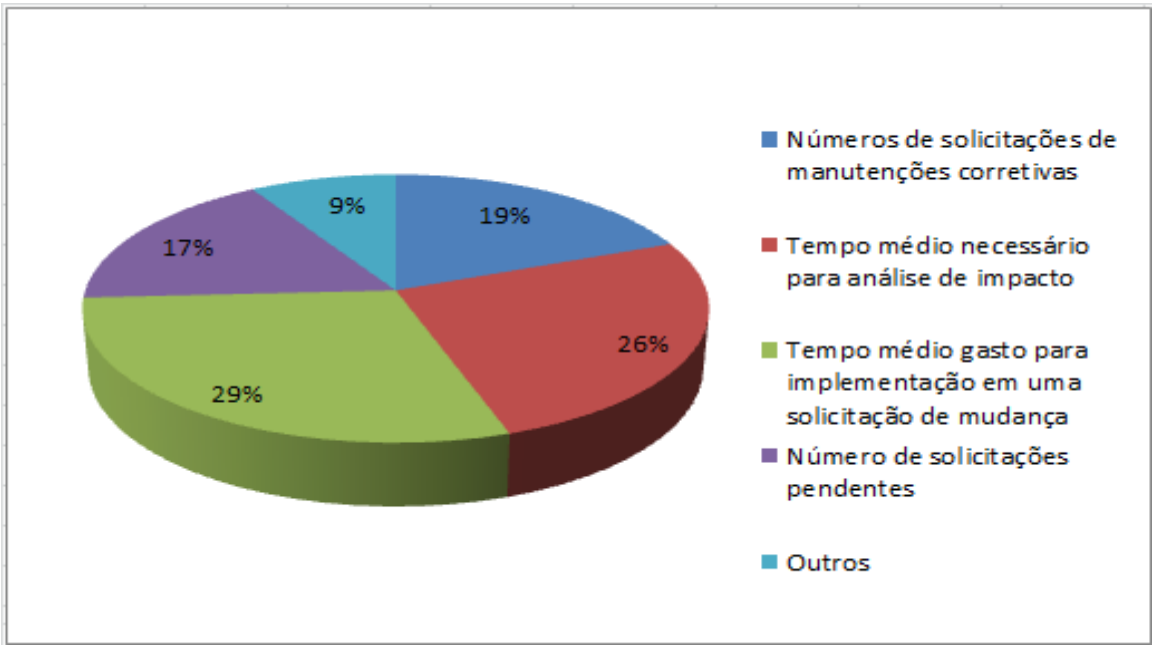
Com base nos estudos realizados, viu-se que a manutenibilidade é a facilidade de alterar ou adicionar funcionalidade em um sistema e quanto mais manutenível o sistema se apresentar, mais fácil será de realizar as devidas

modificações propostas pelos usuários. De acordo com os respondentes, dos fatores apresentados pelo autor (disponíveis como opção na questão), os que mais causam danos à manutenibilidade do *software,* são a documentação do sistema e arquitetura de *software*. A falta de uma documentação adequada para o sistema é apontada como o principal fator, pois Brusamolin (2004) afirma que o tempo para entendimento do *software* em si demanda mais tempo e a escolha de uma arquitetura adequada para o *software* diminui a presença de erros, causando um aumento na manutenibilidade. Sendo assim, é correto afirmar que para se evitar uma queda na manutenibilidade dos sistemas, é aconselhável definir sua arquitetura logo no início e documentar todos os processos detalhadamente, para que as futuras manutenções ocorram de maneira mais amena.

3.1.3.6. Qual métrica a seguir mais impacta no declínio da manutenibilidade do *software*?

Esta questão busca saber do entrevistado quais das métricas que foram assinaladas como opção de resposta mais impacta no declínio da manutenibilidade de um *software*. Caso o respondente não esteja de acordo, foi oferecida a opção "Outros" permitindo o entrevistado exemplificar outra métrica que causa um maior impacto no declínio da manutenibilidade do *software*. Os resultados analisados posteriormente são referentes ao gráfico à seguir.

Gráfico 13 – Métricas impactantes na manutenibilidade



Fonte: Próprio autor

A maioria dos entrevistados (17 pessoas, 29% aproximadamente) afirmam que o tempo médio gasto para implementação em uma solicitação de mudança é a métrica que mais impacta no declínio da manutenibilidade de um sistema. 15 entrevistados (representados por aproximadamente 26% do total) declaram que o tempo médio necessário para análise de impacto é a principal métrica causadora do declínio da manutenibilidade. 11 entrevistados, representando 19% do total, aferem que o número de solicitações de manutenções corretivas impacta mais no declínio da manutenibilidade do que as outras métricas apresentadas. 17% dos entrevistados aproximadamente, totalizando 10 pessoas, afirmam que o número de solicitações pendentes.

Outros 5 respondentes selecionaram a opção “Outros”, descrevendo que a falta de testes automatizados, continuidade de boas práticas ao longo do ciclo de vida do *software*, falta de uma arquitetura ao longo do crescimento da base de códigos e descrição clara dos problemas e um destes 5 entrevistados afirmou que nenhuma das métricas atrapalha no declínio da manutenibilidade do *software*.

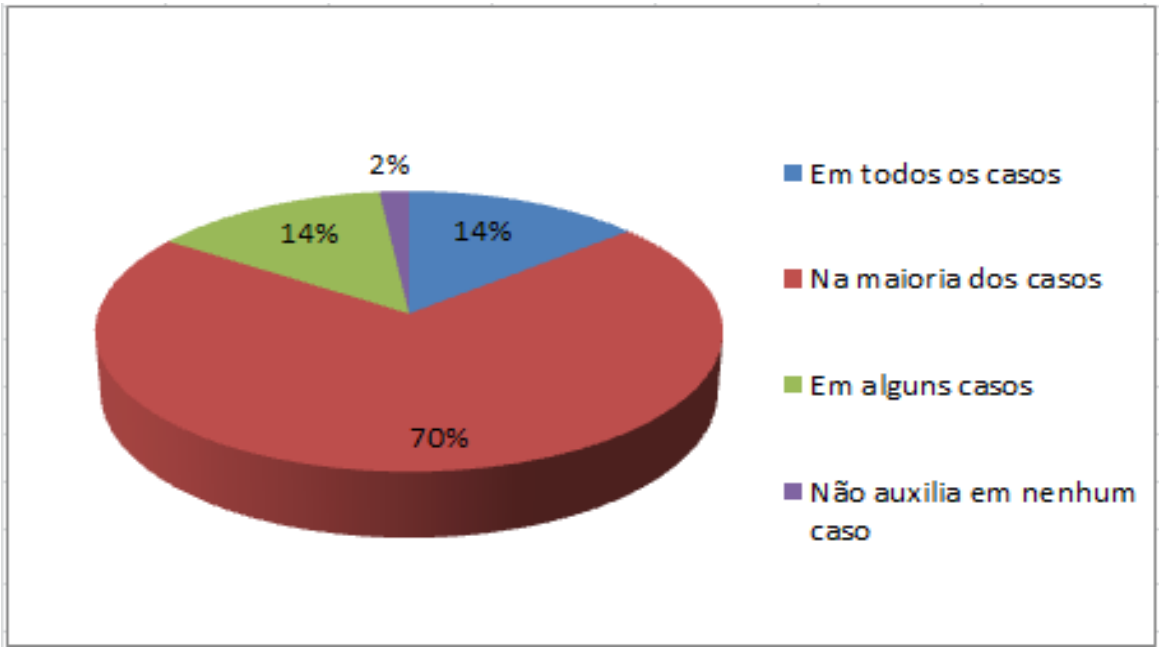
Apesar da maioria dos entrevistados afirmarem que o tempo médio para a análise de impacto e para a implementação de mudança são as principais métricas que levam ao declínio da manutenibilidade (sendo escolhidas por 55% aproximadamente dos respondentes), as demais métricas – juntamente com as outras métricas definidas pelos entrevistados - obtiveram uma quantidade satisfatória de escolha (45% aproximadamente). Sendo assim, foi possível definir quais as métricas (dados obtidos durante a manutenção do *software* capazes de prever um declínio na manutenibilidade) apontadas por Sommerville (2011) em sua obra mais impactam nos sistemas estão relacionadas com o atraso na finalização da demanda. Levando em consideração que a manutenção de aperfeiçoamento é a mais frequente, é correto afirmar que este atraso recorrente neste tipo de manutenção está relacionado com a falta de uma documentação adequada para auxílio no levantamento destes novos requisitos.

3.1.3.7. O modelo de ciclo de vida selecionado ao iniciar um projeto pode impactar positivamente o projeto do *software* quando for atingida a sua etapa de manutenção?

Esta questão busca saber do entrevistado se, ao selecionar no início de um projeto um determinado modelo de ciclo de vida do *software* impacta positivamente

quando for atingida a etapa de manutenção deste *software* em questão. Os resultados apresentados no gráfico à seguir vão de encontro com a opinião de Sommerville (2011), ao relatar que cada modelo possui características similares, porém, a definição certa do tipo de modelo de ciclo de vida do software pode impactar o sistema de várias formas, inclusive positivamente. Os resultados analisados posteriormente são referentes ao gráfico à seguir.

Gráfico 14 – Modelo de ciclo de vida e manutenção de *software*



Fonte: Próprio autor

A ampla maioria dos entrevistados, representados por 70% (41 pessoas), afirmam que a seleção de um determinado modelo de ciclo de vida para àquele *software* inicialmente impacta positivamente ao atingir a etapa de manutenção. 8 pessoas descrevem que a seleção de um modelo de ciclo de vida específico auxilia a manutenção ao chegar na etapa da mesma, mesma quantidade de entrevistados que disseram que somente em alguns casos esta escolha pode auxiliar (14% aproximadamente). Somente um entrevistado disse não auxiliar em nenhum caso.

Com base nestes dados levantados, percebe-se a importância de selecionar com cautela um tipo de ciclo de vida de um *software*, para que, ao alcançar a etapa de manutenção tenha-se resultados positivos aos envolvidos com o sistema.

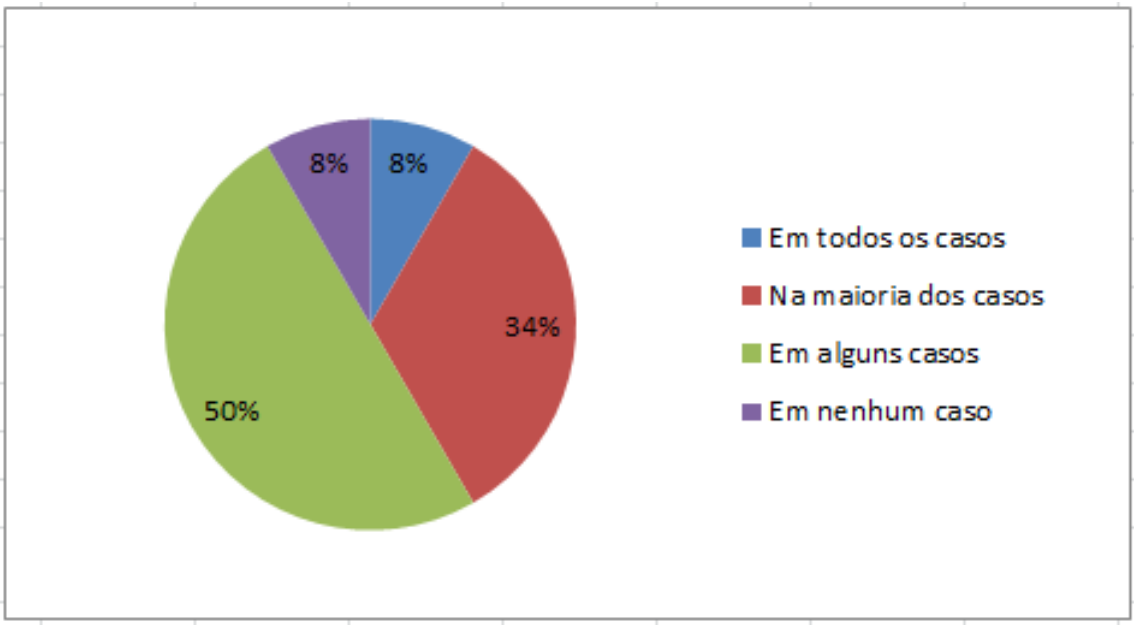
3.1.4. Quarta seção: sobre o gerenciamento e manutenção:

Os resultados descritos nesta seção são referentes à quarta seção do questionário. Cada questão terá seu gráfico próprio contendo os resultados. Esta seção contém 7 perguntas com os objetivos de abordar o respondente sobre as atividades relacionadas aos processos gerenciais e estratégicos de um projeto de *software*. Foram selecionados os respondentes para esta seção aqueles que afirmaram atuar no apoio gerencial, estratégico ou em outras funções (vide questão 3). Viu-se necessário selecionar somente estes respondentes nesta etapa, pois a mesma possui um conteúdo específico para quem trabalha diretamente com o ambiente de gerenciamento dos produtos de *software*. Todas as questões tiveram 12 respondentes ao todo. Cada subtópico desta seção faz referência a uma questão do questionário.

3.1.4.1. Os custos levantados e distribuídos inicialmente em um projeto de *software*, levando em conta o valor estimado para a manutenção de *software*, são alcançados?

Esta questão buscou indagar o respondente sobre os custos que são levantados ao iniciar um projeto, para que se tenha a noção se estes valores são alcançados ou não. Os resultados analisados posteriormente são referentes ao gráfico à seguir.

Gráfico 15 – Custos distribuídos inicialmente e alcançados



Fonte: Próprio autor

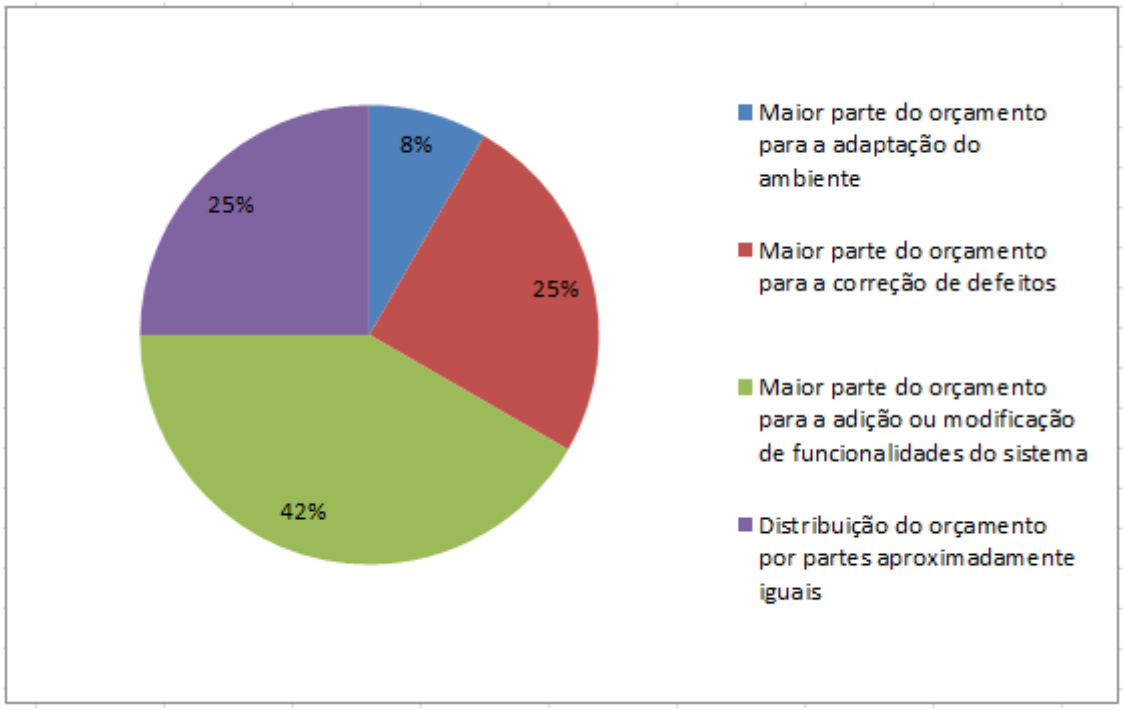
A metade dos respondentes (6 pessoas) afirma que os custos distribuídos inicialmente em um projeto são alcançados somente em alguns casos. 4 pessoas afirmaram (34% aproximadamente) que os custos são distribuídos e alcançados na maioria dos casos. Somente uma pessoa afirmou que estes custos são alcançados em todos os casos e um respondente afirmou que tais custos não são alcançados em nenhum caso.

Com base nas informações aferidas nesta questão, percebe-se que os custos não estão sendo distribuídos corretamente na maioria das vezes, segundo os respondentes deste questionário.

3.1.4.2. Como é realizada a distribuição dos custos do orçamento estimado para a manutenção em sua empresa, de acordo com as três categorias de manutenção existentes segundo Sommerville (adaptação, correção, adição/modificação)?

Esta questão buscou indagar o respondente sobre a distribuição do orçamento que é destinado à manutenção de *software* em sua empresa, de acordo com três categorias que Sommerville (2011) exemplificou em sua pesquisa. Os resultados analisados posteriormente são referentes ao gráfico à seguir.

Gráfico 16 – Distribuição dos custos da etapa de manutenção



Fonte: Próprio autor

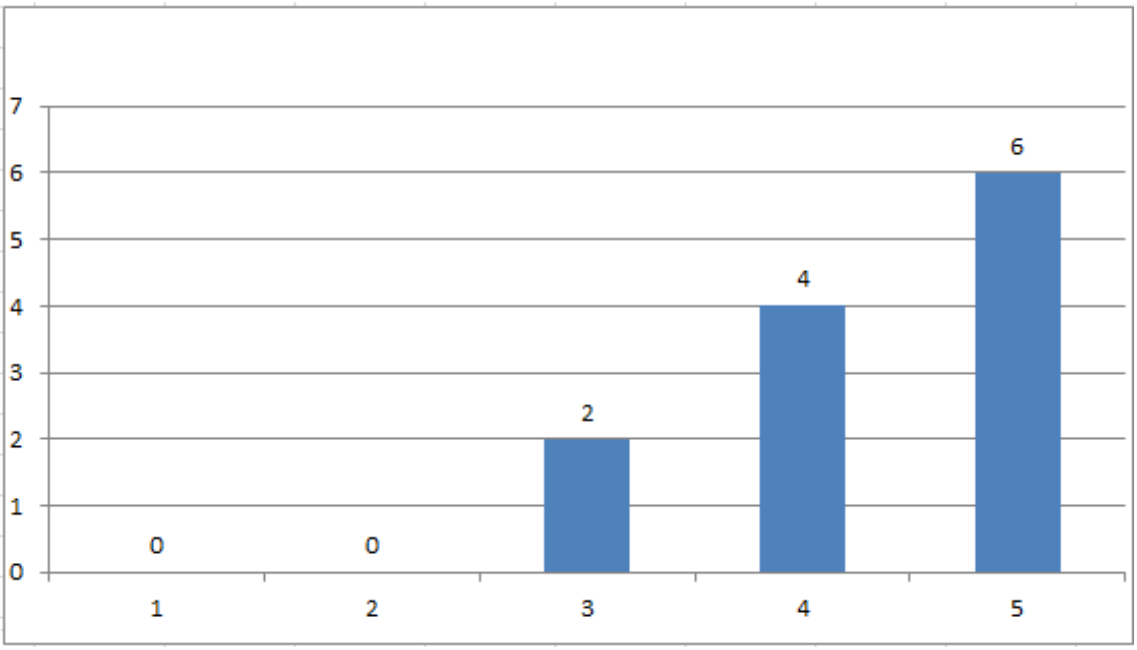
A maioria dos entrevistados, representados por 5 pessoas (42% aproximadamente), disseram que a maior parte do orçamento é destinada à adição ou modificação de funcionalidades do sistema. Três pessoas (25% no total) afirmam que a maior parte do orçamento é destinada à correção de defeitos, mesma quantidade de entrevistados que afirmam que a distribuição do orçamento é realizada igualmente a todas as categorias. Somente um respondente alega que a maior parte do orçamento é destinada para a adaptação do ambiente.

Com base nas respostas da maioria dos entrevistados (representando 67%, 8 pessoas que selecionaram a segunda e terceira opção), pode-se afirmar que a distribuição do orçamento destinado à manutenção é voltada para a adição ou modificação de funcionalidades para a correção de defeitos do sistema. A maioria de

3.1.4.3. O treinamento dos usuários do sistema auxilia em uma redução no custo da manutenção do *software*?

Esta questão buscou saber a opinião dos respondentes sobre a redução de custo na manutenção do software perante treinamentos dos usuários do sistema a ser utilizado por eles. Nesta questão, as opções dadas ao respondente foram de 1 (não reduz os custos) a 5 (reduz significativamente os custos).

Gráfico 17 – Treinamento de usuários para a redução de custos na manutenção de *software*



Fonte: Próprio autor

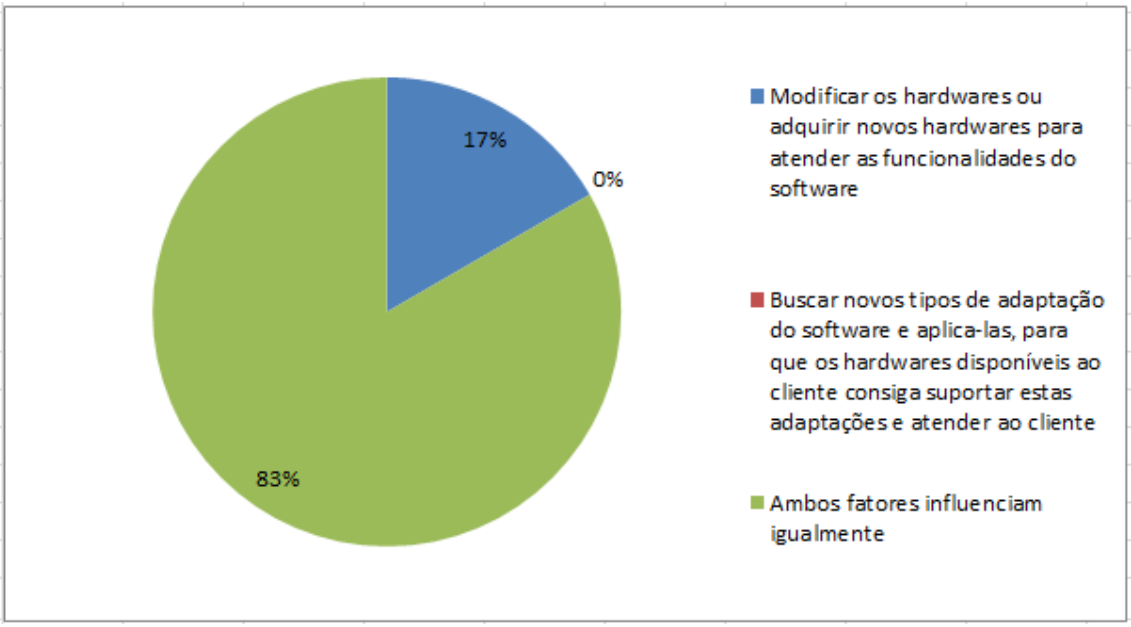
A maioria dos entrevistados (6 pessoas) selecionaram a opção 5, afirmando que a redução de custos é significativa quando se treina seus usuários. Outros 4 entrevistados selecionaram a opção 4. Somente 2 entrevistados assinalaram a opção 3.

Com base nos resultados apresentados pelo Gráfico 17, é possível afirmar que o treinamento de usuários está diretamente relacionado a uma redução de custos da manutenção do *software*.

3.1.4.4. Qual o fator mais influente – visando o custo da manutenção - na adaptação de um *software* para acomodar-se ao ambiente externo?

Buscando conhecer a opinião do entrevistado, nesta questão foram expostos aos respondentes fatores que influem na adaptação de um *software* a um ambiente externo visando o custo da manutenção. Os resultados analisados posteriormente são referentes ao gráfico à seguir.

Gráfico 18 – Fatores influentes ao adaptar um sistema ao ambiente externo visando o custo da manutenção



Fonte: Próprio autor

A maioria dos respondentes desta questão (aproximadamente 83%, 10 pessoas no total) acredita que ambos os fatores previamente citados na questão influenciam igualmente no custo da manutenção de *software*. Somente 2 respondentes afirmam que modificar hardwares ou adquirir novos hardwares para

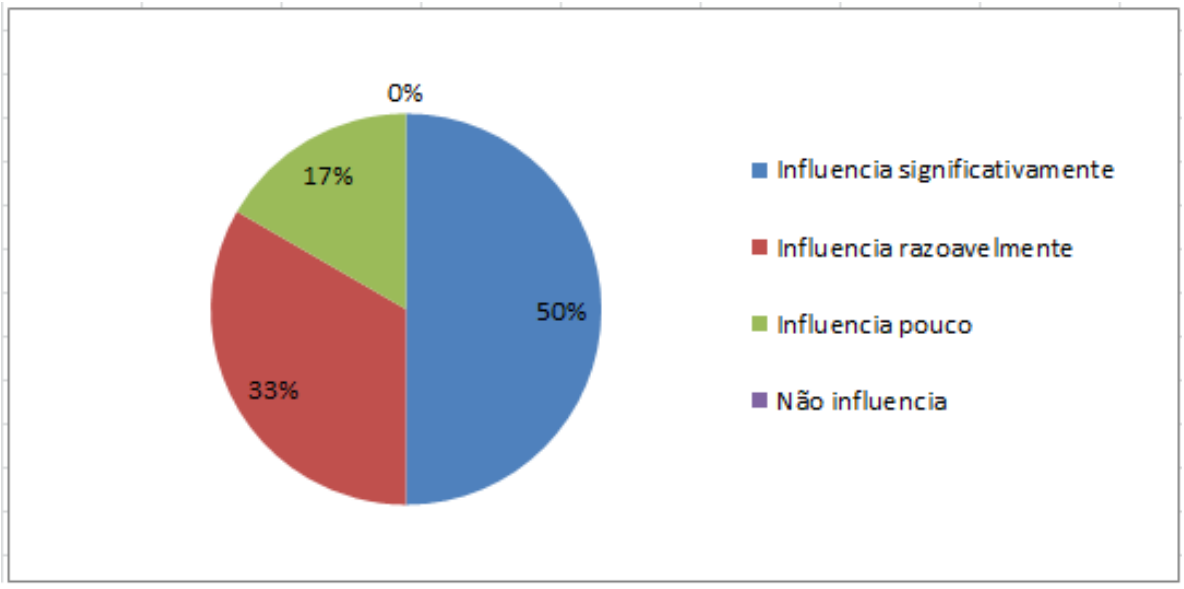
atender as funcionalidades do *software* tem mais influência sobre a busca de novos tipos de adaptação do *software* e aplicá-las, para que os hardwares disponíveis ao cliente consigam suportar as adaptações.

De acordo com os resultados apresentados, ambos os fatores influenciam igualmente no custo da manutenção referente à adaptabilidade do *software* ao ambiente externo.

3.1.4.5. No ponto de vista de gerenciamento de equipes, qual o peso da rotatividade da equipe responsável pela manutenção de *software* em geral?

Os respondentes desta questão puderam opinar qual o peso da rotatividade dos membros das equipes que são responsáveis pela manutenção dos produtos de *software*. Os resultados analisados posteriormente são referentes ao gráfico à seguir.

Gráfico 19 – Rotatividade das equipes de manutenção



Fonte: Próprio autor

A maioria dos entrevistados (50%, 6 pessoas no total), afirmam que a rotatividade das equipes responsáveis pela manutenção dos produtos de *software* influencia significativamente na qualidade da manutenção. Aproximadamente 33% dos respondentes desta questão (4 pessoas), afirmam que a rotatividade das equipes influencia razoavelmente. Somente 2 respondentes disseram que a

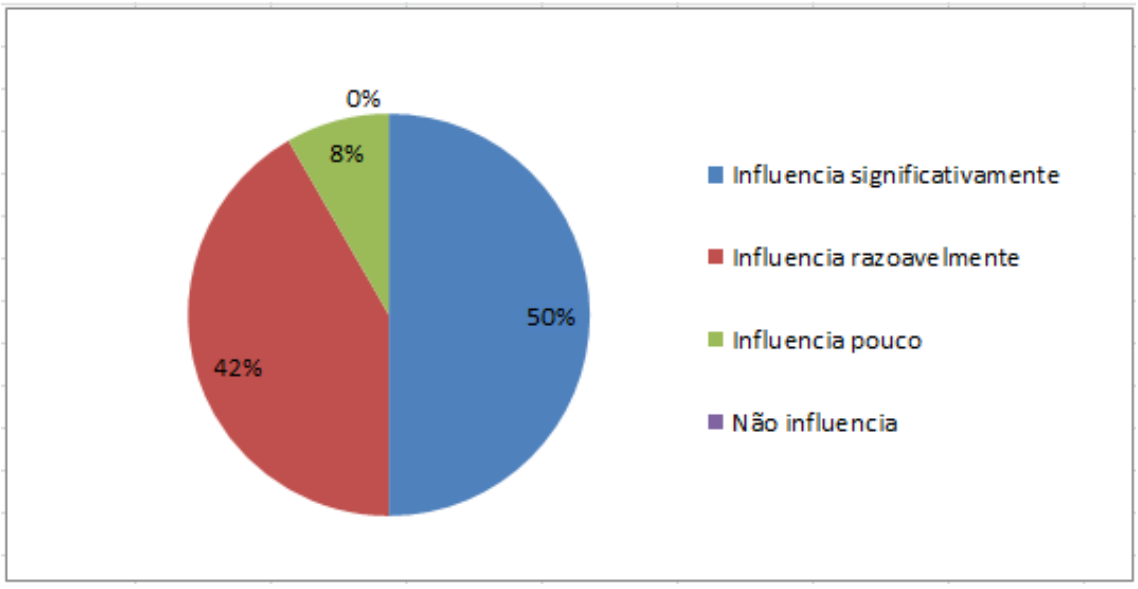
rotatividade tem pouca influência no processo de manutenção. Nenhum respondente afirmou não que a rotatividade venha a influenciar na manutenção.

Com base nos dados levantados, percebe-se que a rotatividade das equipes responsáveis pela manutenção de *software* gera grande influência no desempenho e na qualidade na etapa de manutenção de um determinado *software*.

3.1.4.6. O *backlog* (demanda de novos produtos de *software* maior que a capacidade de produção) influencia na manutenção dos produtos de *software* de responsabilidade da empresa?

Esta questão esperava-se conhecer a opinião dos respondentes sobre a influência do *backlog* na manutenção dos produtos de *software* de responsabilidade da empresa. Os resultados analisados posteriormente são referentes ao gráfico à seguir.

Gráfico 20 – Influência do *backlog* na manutenção dos produtos de *software* de uma empresa



Fonte: Próprio autor

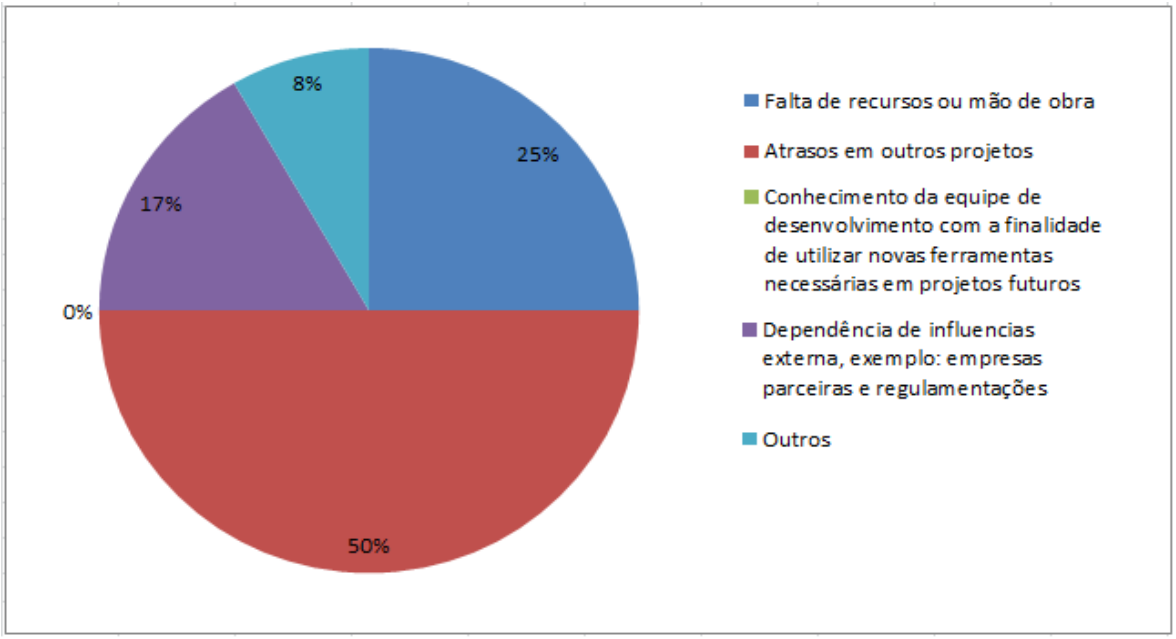
A maioria dos respondentes (50%, representados por 6 pessoas) afirma que o *backlog* possui grande influência na manutenção dos produtos de *software* da empresa. 5 entrevistados responderam que a influência do *backlog* ocorre razoavelmente e somente 1 entrevistado afirma que o *backlog* possui pouca influência na manutenção dos outros produtos de *software*. Nenhum entrevistado afirmou que o *backlog* não influencia na manutenção dos produtos de *software* da empresa.

Com base nos estudos levantados no Gráfico 20, pode-se inferir que o *backlog* tem influência direta na manutenção dos produtos de *software* de uma empresa.

3.1.4.7. Visto que o *backlog* pode ser uma das causas de uma manutenção de *software* conturbada, marque qual a principal causa do *backlog* na sua empresa.

Esta questão busca conhecer a opinião dos entrevistados em relação à qual pode ser a principal causa do *backlog* na empresa que o mesmo esteja atuando. Os resultados analisados posteriormente são referentes ao gráfico à seguir.

Gráfico 21 – Causas do *backlog* na empresa



Fonte: Próprio autor

A principal causa, segundo a maioria dos respondentes desta questão (6 pessoas no total, representando 50% dos respondentes) são os atrasos em outros projetos da empresa. A falta de recursos ou mão de obra foi a opção selecionada de 25% dos entrevistados (3 pessoas no total). 2 respondentes afirmam que a dependência de influências externa são a principal causa do *backlog* na empresa. Somente 1 respondente selecionou a opção “Outros”, afirmando que a principal causa é a falta de mão de obra especializada e a ausência de requisitos nos projetos. Nenhum respondente selecionou a opção “conhecimento da equipe de

desenvolvimento com a finalidade de utilizar novas ferramentas necessárias em projetos futuros”.

Segundo os resultados obtidos e apresentados pelos gráficos 20 e 21, o *backlog* está diretamente ligado aos problemas com a manutenção nos produtos de *software* da empresa, sendo a principal causa apontada para o *backlog* os atrasos nas atuais demandas e a falta de mão de obra. Sendo assim, é correto afirmar que a prorrogação dos cronogramas estabelecidos nas atuais demandas é a principal causa ao impedir a iniciação de novos projetos na empresa.

4. CONCLUSÃO

Com base nas respostas coletadas através do questionário e dos estudos realizados, foi possível aferir que a manutenção de *software*, de maneira geral, é um assunto conhecido entre os respondentes, dos quais a ampla maioria desenvolve alguma atividade em seu dia a dia relacionado à manutenção. Também foi possível conhecer quais os maiores problemas encontrados durante a etapa de manutenção do *software* e de propostas e atividades que auxiliam a manutenção do *software* de certa forma.

Através das respostas obtidas dos entrevistados, foi possível concluir que a maioria dos respondentes não possui uma vasta experiência atuando na computação em geral, sendo profissionais que atuam de 1 a 5 anos na área (Gráfico 2). Ao serem questionados sobre a quantidade de pessoas vinculadas à instituição que eles encontram exercendo alguma atividade, obteve-se uma pequena margem entre as opções propostas, da qual se afere que profissionais de ambientes diversificados de trabalho foram consultados (Gráfico 4). Pode-se afirmar também que os respondentes possuem conhecimento geral sobre o tema proposto, pois a maioria deles possui uma graduação ou se encontra em processo de graduação, sendo assim, o público-alvo a ser alcançado proposto na metodologia deste trabalho foi alcançado com sucesso.

A maioria dos respondentes afirma que as instituições de ensino não dão uma atenção necessária à manutenção, visto a complexidade e importância da mesma em uma instituição (Gráfico 6). Com isso, a maioria afirma que as instituições de ensino devem propor uma disciplina separadamente para a manutenção de *software*, a fim de conhecer seus conceitos e o desenvolvimento da manutenção. Contudo, um estudo destinado à aplicação desta disciplina separadamente para averiguar se haverão resultados positivos definirá se é viável o custo-benefício desta proposta (Gráfico 7).

Os entrevistados que disseram atuar no apoio operacional, estratégico ou gerencial e em outros segmentos, afirmaram que o tipo de manutenção que realizam com mais frequência são dos tipos manutenção corretiva e de aperfeiçoamento (Gráfico 8). Com base nestas afirmações e nas respostas obtidas sobre a distribuição dos recursos para a manutenção, a maioria dos entrevistados

relacionados ao apoio estratégico e gerencial afirmam que grande parte destes recursos é direcionada à manutenção de aperfeiçoamento. A segunda mais escolhida entre estes respondentes foi à destinação dos recursos em maior parte para a manutenção corretiva, juntamente com a distribuição igualitária dos recursos (Gráfico 16). Com base nestas comparações, pode-se afirmar que a maioria dos recursos destinados à manutenção está sendo bem distribuído em alguns casos de acordo com a demanda de manutenção apresentada nas empresas. Para um melhor aproveitamento dos recursos na maioria dos casos, visto a grande diferença no consumo dos recursos, é necessário que a maior parte destes recursos seja direcionada à manutenção de aperfeiçoamento, pois há uma grande demanda da mesma de acordo com os entrevistados.

As quatro principais dificuldades na etapa de manutenção do *software* apontadas pelos respondentes no questionário foram (Gráfico 11): A falta de documentação das funcionalidades e dos processos do *software*; a falta de boas práticas no desenvolvimento do código-fonte do *software;* falta de previsão da real necessidade do usuário e a falta de reuniões programadas para analisar a real necessidade dos usuários. Para que estas dificuldades sejam evitadas, deve-se elaborar e alimentar desde o início uma documentação para o sistema em questão, adotar boas práticas no desenvolvimento do código-fonte e manter um contato contínuo com o cliente, procurando revisar os requisitos levantados para que as alterações solicitadas possam ser efetuadas em tempo hábil.

Através dos dados obtidos e disponibilizados no Gráfico 13, pode-se afirmar as métricas “tempo médio gasto para a implementação em uma solicitação de mudança” e o “tempo médio necessário para uma análise de impacto” são consideradas pelos entrevistados como as mais preocupantes, se tratando do declínio da manutenibilidade do sistema. É possível afirmar, com base em outras respostas obtidas, que as duas métricas mais preocupantes (que estão diretamente relacionadas com o tempo gasto até a finalização da demanda) podem ser nulificadas caso haja uma documentação adequada do sistema, pois permitiria a eles um conhecimento mais ágil do objetivo real de uma funcionalidade em específico.

Mesmo que a manutenção de adaptação do ambiente não tenha sido considerada a mais recorrente segundo os entrevistados que atuam com desenvolvimento ou atividades similares, os entrevistados que atuam no apoio

estratégico ou gerencial de uma empresa afirmaram que ambos os fatores que foram apresentados a eles na questão 18 (o Gráfico 18 contém os resultados desta questão) influenciam igualmente se tratando do custo da manutenção especificamente. A partir destas informações, é possível afirmar que nos casos de adaptação do ambiente, os responsáveis pela manutenção devem propor o melhor meio de acordo com seus recursos para solucionar estas solicitações de manutenção, pois não haverá influência significativa nos custos da manutenção.

A escolha correta do modelo de ciclo de vida do *software* mostrou influenciar positivamente a etapa de manutenção do *software* segundo ampla maioria dos entrevistados (Gráfico 14). Sendo assim, pode-se afirmar que a definição certa do tipo de modelo de ciclo de vida do *software* pode impactar o sistema de várias formas, inclusive positivamente.

A maioria dos respondentes que atuam na gerência ou na área estratégica, afirmam que o treinamento de usuários visando à redução de custos da manutenção possui uma significativa eficiência neste quesito (Gráfico 17). O devido conhecimento das funcionalidades do sistema e suas restrições são importantes para que os usuários finais evitem solicitar modificações redundantes no sistema.

A rotatividade foi considerada pela equipe gerencial como um problema significativo na manutenção de *software* (Gráfico 19). Além dos problemas levantados anteriormente pelos respondentes neste trabalho, a rotatividade de funcionários em uma empresa causa um atraso natural até que o novo funcionário conheça todos os processos e produtos da empresa, deixando assim durante este processo algumas demandas em atraso.

É necessário que se tenha uma atenção maior com o orçamento do projeto, principalmente com os recursos voltados para a manutenção do *software*. Foi constatado através das respostas dos entrevistados que somente às vezes os custos que são definidos inicialmente no projeto são alcançados (Gráfico 15). Para que os valores estipulados sejam alcançados sem que haja um aumento destes custos, desde o início do projeto é preciso se atentar aos requisitos e o desenvolvimento passo a passo do projeto para que não haja problemas maiores com a manutenção do *software*.

Tendo analisado as respostas obtidas dos entrevistados com os levantamentos feitos nos estudos dos autores que serviram de alicerce para este trabalho, foi possível aferir os principais pontos, tanto positivos como negativos que

envolvem a manutenção do *software*, para que se obtenha uma manutenibilidade maior do *software*.

Tendo a ciência de que a etapa de manutenção do *software é* a mais longa de seu tempo de vida, pode-se afirmar que de acordo com os resultados obtidos, os atrasos são as principais causas da queda da manutenibilidade do *software*, que estão diretamente relacionadas à falta de uma documentação adequada do sistema, a ausência de boas práticas no desenvolvimento do código-fonte e a falta da real necessidade do usuário. A rotatividade dos funcionários relacionados à manutenção também foi considerada bem influente na queda da manutenibilidade do *software*. A ausência de uma arquitetura do *software* mostrou-se um fator impactante na negatividade da manutenibilidade dos sistemas. Pode-se afirmar também que a reusabilidade, a escolha de uma modelo de ciclo de vida específico para o *software* e o treinamento de usuários são importantes aliados para uma etapa de manutenção do *software* ausente de maiores problemas.

5. TRABALHOS FUTUROS

Uma possível continuação deste trabalho poderia ser um acompanhamento do desenvolvimento de dois produtos de *software* do mesmo porte, utilizando como base um ciclo de vida específico cada um e o mesmo valor orçamentário, com a finalidade de comparar a manutenibilidade de ambos após um tempo de sua implementação, verificando qual ciclo de vida pode ser mais viável e rentável para a implementação daquele tipo de produto desenvolvido.

A realização de um estudo sobre a manutenção de *software* em diferentes tipos de *software*, tendo como objetivo central levantar o custo da manutenção, levando em consideração o detalhamento da matéria-prima utilizada nos diferentes tipos de *software* e a análise da manutenibilidade em cada um dos produtos de *software.*

Realizar um estudo com dois grupos de estudantes que ainda não iniciaram os estudos em engenharia de *software*. Serão lecionados para uma equipe os conceitos de manutenção de *software* que é utilizado já pela instituição de ensino e para a outra equipe, será lecionado os conceitos de manutenção de *software* como uma disciplina separada da engenharia de *software.* Após lecionar os devidos conteúdos para as equipes, cada equipe desenvolverá um *software* até sua implementação. Uma terceira equipe já de profissionais acompanhará as solicitações de manutenção dos produtos de *software* implementados. Sendo assim, o objetivo central do trabalho seria a análise da manutenibilidade de cada *software*, tendo também o objetivo de observar se os conceitos sobre manutenção estão sendo bem aplicados nas instituições de ensino, preparando assim os graduandos melhor para o mercado de trabalho.

6. REFERÊNCIAS

ANQUETIL, Nicolas; DIAS, Márcio G. *Proposta de uma disciplina sobre manutenção de softwares na graduação.* XXV Congresso da Sociedade Brasileira de Computação, São Leopoldo, RS – 2005.

BRUSAMOLIN, V. *Manutenibilidade de Software.* In: Revista Digital Online, v.2, jan. 2004.

CALAZANS, A. T. S.; OLIVEIRA, M. A. L. *Avaliação de Estimativa de Tamanho para Projetos de Manutenção de Softwares.* Toffano Seidel Calazans et al./Proc. of Argentine Symposium on Software Engineering, 2005.

GHEZZI, Carlo; JAZAYERI, Mehdi; MANDRIOLI, Dino. *Fundamentals of Software Engineering.* New Jersey: Prentice Hall, 1991.

PADUELLI, Mateus M. *Manutenção de Software: problemas típicos e diretrizes para uma disciplina específica***.** 2007. 144 p. Dissertação (Mestrado) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos, SP, 2007.

PADUELLI, Mateus M.; SANCHES, Rosely. *Problemas em manutenção de Software: caracterização e evolução.* – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos, SP, 2006.

PAULA FILHO, Wilson P. *Engenharia de Software: fundamentos, métodos e padrões.* Editora LTC. 2001.

PETERS, James F; PEDRYCZ, Witold. *Engenharia de Software.* / James F. Peters, Witold Pedrycz; tradução de Ana Patricia Garcia – Rio de Janeiro: Campus, 2001.

PRESSMAN, Roger S. *Engenharia de Software [recurso eletrônico]: uma abordagem profissional.* Roger S. Pressman . 7ªed. Dados Eletrônicos. Porto Alegre: AMGH, 2011.

RIBEIRO, Maicon V. *O reúso de código-fonte como metodologia de aprimoramento da produção de softwares.* Trabalho de conclusão de curso. Faculdades Integradas de Caratinga – Caratinga. 2013.

SANTOS, Rodrigo P. *Critérios de Manutenibilidade para construção e avaliação de produtos de software orientados a aspectos.* Monografia de graduação. Universidade Federal de Lavras. – Lavras, MG – 2007.

SHOOMAN, Martin L. *Software Engineering – Design, Reliability and Management.* McGraw-Hill – Singapore – 1983.

SILVA, Ronan A.; MATOS, Simone; SOUZA, João H.;MOURA, Louisi F. *A manutenção de software nas empresas.* Congresso Internacional de Administração – 2010.

SOMMERVILLE, Ian. *Engenharia de Software.* / Ian Sommerville; Tradução Ivan Bosnic e Kalinca G. de O. Gonçalves; revisão técnica Kechi Hirama. 9ªed. São Paulo: Pearson Prentice Hall, 2011.

VASCO, Carlos G.; VITOFHT, Marcelo H.; ESTANTE, Paulo Roberto C. *Comparação entre metodologias RUP e XP.* Pontífica Universidade Católica do Paraná (PUCPR) – Curitiba, PR – 2006.

VENTORIN, Alessandro J. *Principais problemas relacionados ao desenvolvimento de sistemas*. Faculdade Capixaba de Nova Venécia – UNIVEN.

7. ANEXOS

7.1. ANEXO 1: QUESTIONÁRIO

Manutenção de Software

Este questionário tem o intuito de analisar a importância da manutenção de software e suas abordagens.
As perguntas estão fundamentadas na opinião de autores sobre a manutenção de software, conforme pesquisa científica realizada entre fevereiro e agosto de 2017.
Dados pessoais não serão analisados neste questionário.

Autor: Eduardo Henrique Tonel Lopes
8º Período, Ciência da Computação
Faculdades Integradas de Caratinga - FIC
eduardotonel93@gmail.com

*Obrigatório

Sobre a qualidade do respondente:

1 – Qual seu nível de formação acadêmica? *

- ◯ Técnico;
- ◯ Graduando ou graduado (ensino superior);
- ◯ Pós-Graduando ou Pós-Graduado (lato sensu);
- ◯ Pós-Graduando ou Pós-Graduado (stricto sensu);
- ◯ Outro: _____

2 – Há quanto tempo você se encontra exercendo qualquer atividade relacionada à área da computação? *

- ◯ Menos de um ano;
- ◯ Um a cinco anos;
- ◯ Cinco a dez anos;
- ◯ Dez a quinze anos;
- ◯ Mais de quinze anos;

3 – Qual sua atual função relacionada à instituição em que você se encontra vinculada? *

- ◯ Estudante;
- ◯ Apoio técnico (infraestrutura, redes);
- ◯ Apoio Operacional (desenvolvedor de softwares, analista de testes, suporte de sistemas, analista de sistemas, webdesigner);
- ◯ Apoio Gerencial ou Estratégico (Coordenador de equipes, diretor, CEO, CIO, gerente de projetos, gerente de qualidades, gerente de testes);
- ◯ Outro: _____

4 - Qual a quantidade de profissionais, aproximadamente, vinculados à instituição em que você trabalha? *

- ◯ Menos de quinze pessoas;
- ◯ Entre quinze a trinta pessoas;
- ◯ Entre trinta a cinquenta pessoas;
- ◯ Mais de cinquenta pessoas;

Sobre a manutenção de software - abordagens gerais:

5 - Do ponto de vista geral, qual o seu conhecimento sobre a manutenção de software? *

	1	2	3	4	5	
Pouco	◯	◯	◯	◯	◯	Muito

6 – De seu ponto de vista, qual a importância dada pelas instituições de ensino ao se discutir sobre o ciclo de vida de um software e a etapa de manutenção do software? *

	1	2	3	4	5	
Baixa Importância	◯	◯	◯	◯	◯	Alta Importância

7 – Para que se tenha conhecimento sobre os conceitos da manutenção de software e o seu desenvolvimento dentro de um ambiente profissional, em sua opinião, os ambientes de graduação deveriam propor uma disciplina separadamente para a abordagem deste assunto? *

- ◯ Sim. O melhor meio seria abordar o tema como uma disciplina separadamente;
- ◯ Não. O assunto proposto já é bem abordado ou deve-se abordar com mais ênfase utilizando outros métodos (seminários, palestras, visitas técnicas);

Sobre a manutenção de software aplicada ao ambiente de trabalho e no desenvolvimento:

8 – Qual o tipo de manutenção no software que você mais costuma realizar em seu ambiente de trabalho? *

- ◯ Manutenção Corretiva;
- ◯ Manutenção de Aperfeiçoamento;
- ◯ Manutenção Adaptativa;
- ◯ Não realizo manutenção em produtos de software;

9 – Em seu ambiente de trabalho, como é realizada a manutenção de um software? *

- ◯ A empresa não realiza manutenção no software, somente o seu desenvolvimento;
- ◯ A manutenção do software é realizada pela mesma equipe do desenvolvimento;
- ◯ A manutenção é prioridade de uma equipe, porém podem contar com o auxílio necessário da equipe que desenvolveu o software;
- ◯ A manutenção do software é realizada por uma equipe separadamente, sem contato com a equipe que o desenvolveu;
- ◯ Outro: _____

10 – A reusabilidade de código-fonte auxilia no processo de manutenção de software? *

- ○ Em todos os casos;
- ○ Na maioria dos casos;
- ○ Em poucos casos;
- ○ Não auxilia em nenhum caso;

11 – Durante o processo de manutenção, marque as maiores dificuldades encontradas: *

- ☐ Falta de documentação dos processos e funcionalidades do software;
- ☐ Não ter participado dos processos de desenvolvimento do software;
- ☐ Pouco interesse ou falta de entendimento do usuário no funcionamento do sistema;
- ☐ Falta de previsão da real necessidade do usuário;
- ☐ Curto prazo para estudo de materiais novos processos a fim de se obter a melhor solução para o problema;
- ☐ Design do sistema de difícil entendimento;
- ☐ Falta de reuniões programadas para analisar as necessidades reais dos usuários;
- ☐ Falta de familiarização com a linguagem utilizada no desenvolvimento do software;
- ☐ Falta de boas práticas no desenvolvimento código-fonte;
- ☐ Alto número de solicitações de manutenção em espera;
- ☐ Outro: _____

12 - Qual o fator que causa mais impacto na manutenibilidade do software? *

- ◯ Arquitetura do software;
- ◯ Tecnologia utilizada para desenvolvimento do software;
- ◯ Documentação do sistema;
- ◯ Compreensibilidade do programa;
- ◯ Outro: _____

13 – Qual métrica a seguir mais impacta no declínio da manutenibilidade do software? *

- ◯ Números de solicitações de manutenções corretivas;
- ◯ Tempo médio necessário para análise de impacto;
- ◯ Tempo médio gasto para implementação em uma solicitação de mudança;
- ◯ Número de solicitações pendentes;
- ◯ Outro: _____

14 – O modelo de ciclo de vida selecionado ao iniciar um projeto pode impactar positivamente o projeto do software quando for atingida a sua etapa de manutenção? *

- ◯ Em todos os casos;
- ◯ Na maioria dos casos;
- ◯ Em alguns casos;
- ◯ Não auxilia em nenhum caso;

Sobre o gerenciamento e manutenção:

15 – Os custos levantados e distribuídos inicialmente em um projeto de software, levando em conta o valor estimado para a manutenção de software, são alcançados? *

- ◯ Em todos os casos;
- ◯ Na maioria dos casos;
- ◯ Em alguns casos;
- ◯ Em nenhum caso;

16 – Como é realizada a distribuição dos custos do orçamento estimado para a manutenção em sua empresa, de acordo com as três categorias de manutenção existentes segundo Sommerville (adaptação, correção, adição/modificação) ? *

- ◯ Maior parte do orçamento para a adaptação do ambiente;
- ◯ Maior parte do orçamento para a correção de defeitos;
- ◯ Maior parte do orçamento para a adição ou modificação de funcionalidades do sistema;
- ◯ Distribuição do orçamento por partes aproximadamente iguais;

17 – O treinamento dos usuários do sistema auxilia em uma redução no custo da manutenção do software? *

	1	2	3	4	5	
Não reduz os custos	◯	◯	◯	◯	◯	Reduz significativamente os custos

18 – Qual o fator mais influente – visando o custo da manutenção - na adaptação de um software para acomodar-se ao ambiente externo ? *

- ◯ Modificar os hardwares ou adquirir novos hardwares para atender as funcionalidades do software;
- ◯ Buscar novos tipos de adaptação do software e aplica-las, para que os hardwares disponíveis ao cliente consiga suportar estas adaptações e atender ao cliente;
- ◯ Ambos fatores influenciam igualmente;
- ◯ Outro: _____

19 – No ponto de vista de gerenciamento de equipes, qual o peso da rotatividade da equipe responsável pela manutenção de software em geral ? *

- ◯ Influencia significativamente;
- ◯ Influencia razoavelmente;
- ◯ Influencia pouco;
- ◯ Não influencia;

20 – O backlog (demanda de novos produtos de software maior que a capacidade de produção) influencia na manutenção dos produtos de software de responsabilidade da empresa? *

- ◯ Influencia de forma ampla;
- ◯ Influencia razoavelmente;
- ◯ Influencia pouco;
- ◯ Não influencia;

21 – Visto que o backlog pode ser uma das causas de uma manutenção de software conturbada, marque qual a principal causa do backlog na sua empresa. *

- ◯ Falta de recursos ou mão de obra;
- ◯ Atrasos em outros projetos;
- ◯ Conhecimento da equipe de desenvolvimento com a finalidade de utilizar novas ferramentas necessárias em projetos futuros;
- ◯ Dependência de influencias externa, exemplo: empresas parceiras e regulamentações;
- ◯ Outro: _____