INSTITUTO DOCTUM DE EDUCAÇÃO E TECNOLOGIA

FACULDADES INTEGRADAS DE CARATINGA

CIÊNCIA DA COMPUTAÇÃO

ENGENHARIA DE REQUISITOS: ESTUDO SOBRE DESENVOLVIMENTO ÁGIL E TRADICIONAL

CLERYANNE CRISTHYNE ALVES CEVIDANES

Caratinga
2015

CLERYANNE CRISTHYNE ALVES CEVIDANES

ENGENHARIA DE REQUISITOS: ESTUDO SOBRE DESENVOLVIMENTO ÁGIL E TRADICIONAL

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 pela professora Msc. Fabrícia Pires Souza Tiola.

Caratinga
2015

FACULDADES INTEGRADAS DE CARATINGA
TRABALHO DE CONCLUSÃO DE CURSO
TERMO DE APROVAÇÃO

TÍTULO DO TRABALHO
ENGENHARIA DE REQUISITOS: UM ESTUDO SOBRE DESENVOLVIMENTO ÁGIL E TRADICIONAL

por

Cleryanne Cristhyne Alves Cevidanes

Este Trabalho de Conclusão de Curso foi apresentado perante a Banca de Avaliação composta pelos professores Fabrícia Pires Souza Tiola, Bruno Vieira Becattini e Maicon Vinícius Ribeiro, às 21 horas do dia 09 de dezembro de 2015 como requisito parcial para a obtenção do título de bacharel. Após a avaliação de cada professor e discussão, a Banca Avaliadora considerou o trabalho aprovado, com a qualificação: EXCELENTE.

Trabalho indicado para publicação: (X)SIM ( )NÃO

Caratinga, 14 de DEZEMBRO de 2015

Professor Orientador e Presidente da Banca

Bruno Vieira Becattini
Professor Avaliador 1

Maicon Ribeiro
Professor Avaliador 2

Aluno(a)

Coordenador (a) do Curso

DEDICATÓRIA

Aos meus pais que nunca mediram esforços para que meus sonhos se tornassem reais.

Aos meus avós, Mariem e Geraldo que são para mim exemplo de força, determinação, caráter e amor.

Obrigada por tudo, amo vocês!

AGRADECIMENTOS

Agradeço primeiramente a Deus, que nunca me desamparou mesmo nos momentos em que eu não merecia.

Agradeço a minha mãe pela paciência, ensinamentos e motivação durante todos esses anos. Ao meu pai que mesmo distante e não entendendo nada sobre computação me motivava e se orgulhava de mim.

Aos meus avós maternos pelas histórias, pelos conselhos, principalmente a minha vó Mariem pelo colo e pelas orações constantes.

A minha irmã, que dormia com a luz acesa enquanto eu virava noites e mais noites escrevendo a monografia, valeu Mah!

As minhas amigas Camila, Jéssica, Júnia, Licianny, Marina e Patrícia e meu amigo Lucas por estarem presentes me apoiando mesmo que de longe.

Ao meu namorado, pelo companheirismo, paciência diária e pelas ideias trocadas, obrigada, eu te amo!

Aos amigos e colegas de turma por esses quatro anos inesquecíveis, principalmente a Jéssica pelas conversas, trabalhos e tudo mais que passamos juntas durante esses anos.

A minha orientadora Fabrícia, pelos conselhos, ideias e pela paciência comigo durante esse ano, muito obrigada!

Aos meus professores por todo ensinamento, dedicação e apoio durante esses quatro anos de curso.

E aos demais familiares e colegas que acompanharam de perto e sempre me ajudaram.

RESUMO

A criação de um software pode ser considerada complexa. Na Engenharia de Requisitos temos que levantar os dados, compreendê-los e adequá-los ao gosto do cliente, fazendo com que os prazos sejam cumpridos e comprometendo a qualidade do software. Utilizando métodos ágeis poderíamos ter um planejamento, compreensão e um produto final de maior qualidade e com o mínimo de problemas, levando em consideração as consequências de levantar requisitos sem um método ideal e sem o preparo adequado, trazendo a insatisfação do cliente e futuramente trazendo mais dificuldade para ser atualizado. O objetivo desse trabalho é analisar e comparar os desenvolvimentos ágil e tradicional. Foi aplicado um questionário com objetivo de comparar as metodologias e técnicas utilizadas pelos profissionais que trabalham com desenvolvimento de software e pode-se concluir dele que a maioria dos desenvolvedores possuem pouca experiência e desenvolvem utilizando metodologias ágeis. Com esse trabalho conclui-se que os métodos tradicionais ainda são utilizados pelos desenvolvedores porém estão entrando em desuso e os métodos ágeis estão sendo mais utilizados pelas empresas.

Palavras-chave: Requisitos, Software, Métodos ágeis, Métodos tradicionais, desenvolvimento

ABSTRACT

The creation of software may be considered complex. In Requirements Engineering we have to get the data, understand them and adjust them to the customer's taste, so that deadlines are met and compromising the quality of the software. Using agile methods could have a plan, understanding and an end product of higher quality and with minimal problems, taking into account the consequences of raising requirements without an ideal method and without adequate preparation, bringing the customer dissatisfaction and eventually bringing more difficulty to be updated. The aim of this study is to analyze and compare the agile and traditional development. A questionnaire in order to compare the methodologies and techniques used by professionals working with software development and it can be concluded that most developers have little experience and develop using agile methodologies has been applied. With this work we conclude that traditional methods are still used by developers but are entering into disuse and agile methods are being used by most companies.

Keywords: Requirements, Software, Agile methods, traditional methods, development

LISTA DE SIGLAS

AR – Analista de Requisitos

ER – Engenharia de Requisitos

FDD – *Feature Driven Development*

IEEE - *Institute of Electrical and* Electronics Engineers

MA’s – Metodologias ágeis

RUP – *Rational Unified Process*

UML – *Unified Modeling Language*

XP – *Extreme programming*

LISTA DE ILUSTRAÇÕES

SUMÁRIO

INTRODUÇÃO

Empresas de tecnologia da informação que prestam serviço de desenvolvimento de software sofrem com prazos a serem cumpridos, comunicação falha com os clientes, entre outros problemas. Com o intuito de melhorar os problemas que aparecem á medida que um projeto é desenvolvido, passaram a utilizar as metodologias ágeis.

Entender requisitos é considerado uma tarefa complicada em qualquer método de desenvolvimento. Várias metodologias são usadas com o intuito de desenvolver esses requisitos de maneira correta, satisfazendo as necessidades do cliente. O objetivo dos métodos ágeis é formar um diálogo constante com o cliente à medida que os requisitos evoluem, priorizando e os classificando.

Esse trabalho tem como objetivo fornecer uma comparação entre o desenvolvimento ágil e o desenvolvimento de software tradicional, como é feita elicitação de requisitos nas empresas que trabalham com o desenvolvimento de software, como elas lidam com os projetos, com a comunicação entre cliente e empresa e até mesmo entre os componentes da equipe de desenvolvimento, as documentações, metodologias e técnicas utilizadas para desenvolver softwares.

Para alcançar o objetivo dessa pesquisa foi necessário conhecer as principais técnicas de especificação de requisitos de softwares e apresentar os métodos que são utilizados no desenvolvimento tradicional, conhecer mais sobre o Manifesto ágil que vem sendo reconhecido nesses últimos anos e os métodos de desenvolvimento ágeis mais utilizados, suas vantagens e desvantagens frente à especificação de requisitos no método tradicional.

Para o estudo proposto foi necessário redigir e aplicar um questionário voltado para profissionais que trabalham com desenvolvimento de software, a fim de coletar dados necessários para analisar quais métodos são utilizados nas empresas, os desafios encontrados durante a fase de elicitação de requisitos, descobrir se as empresas seguem e concordam com os princípios e valores defendidos pelo Manifesto ágil e como é a comunicação entre empresa e cliente.

Esse questionário foi dividido em cinco seções sendo elas: Identificação do participante onde o respondente respondia qual cargo ele ocupava, sua idade, entre

outros; Engenharia de requisitos, onde eram apresentadas perguntas sobre os desafios encontrados pelos participantes, as técnicas de elicitação de requisitos que eram mais conhecidas; Metodologias tradicionais, onde o participante podia marcar quantos anos de experiência ele tinha sobre os métodos tradicionais, quais eram na opinião dele as vantagens e desvantagens desses métodos e quais técnicas eles utilizam durante o desenvolvimento tradicional; Metodologias ágeis, onde o respondente assinalava quantos anos ele tinha de experiência trabalhando com desenvolvimento ágil, quais metodologias mais utilizadas por eles, quais técnicas de elicitação eram utilizadas, as vantagens e desvantagens dos métodos ágeis e podiam expressar a opinião deles sobre fatos que ocorrem ou deveriam ocorrer em suas empresas de acordos com princípios e valores defendidos pelo Manifesto ágil; Relacionamento com o cliente, onde os profissionais davam sua opinião de acordo com a experiência deles com os clientes, se esses clientes aprovam os softwares, se achavam importante acompanhar o desenvolvimento do software, entre outros assuntos.

A pesquisa foi dividida em quatro capítulos sendo eles: Referencial teórico, onde foi falado sobre a Engenharia de Requisitos, modelos de processos de software, técnicas de elicitação de requisitos, surgimento e valores do Manifesto ágil, os métodos ágeis XP e Scrum; Metodologia, que foi dividido entre a subseção público alvo do questionário em questão, o detalhamento de cada seção do questionário aplicado e as ferramentas utilizadas para analisar e exibir os resultados do trabalho; Análise e Resultados onde foram exibidos os gráficos com os resultados do questionário aplicado, dividido por subseções nomeadas como as seções do questionário e dentro dessas subseções havia outros tópicos com as perguntas analisadas.

1. REFERENCIAL TEÓRICO

De acordo com Pressman (2011), na década de 70, o desenvolvimento de sistemas era feito de forma desorganizada, desestruturada e sem planejamento, o que resultava em um software de péssima qualidade e que não fazia o que os clientes exigiam. Quando foi notada essa realidade, algumas empresas de desenvolvimento viram a necessidade de transformar aqueles processos e fases de desenvolvimento do software em fases estruturadas, planejadas e padronizadas.

A partir dessa época, foram surgindo vários métodos de desenvolvimento, além de novas linguagens de programação que facilitavam o entendimento do software pelos desenvolvedores e os clientes.

As mudanças frequentes do software e de sua documentação demandavam tempo e para diminuir os problemas encontrados surgiram os métodos ágeis, que focam no código e otimização para as alterações de requisitos.

Com o surgimento das metodologias de desenvolvimento ágil fez com que acontecesse a divisão das metodologias em duas categorias, as metodologias tradicionais que são baseadas no projeto, criando documentos para servirem de guias para as fases de desenvolvimento do software e as metodologias ágeis que se baseiam no código, utilizando menos documentação e adotando fases menores.

Apesar das metodologias ágeis serem apontadas como alternativa ás abordagens tradicionais para o desenvolvimento de software, os métodos tradicionais, são ainda as mais utilizadas em circunstâncias onde os requisitos permaneçam estáveis e os requisitos futuros são previsíveis.

Nessa seção será falado sobre a Engenharia de Requisitos, Técnicas de Engenharia de Requisitos, Metodologias de desenvolvimento Tradicionais, Manifesto Ágil, Metodologias de desenvolvimento Ágil, XP e SCRUM.

1.1 ENGENHARIA DE REQUISITOS

Os requisitos são todas as descrições do sistema, todas as suas funcionalidades e restrições, todos descritos pelo próprio cliente que é parte importante nesse processo de criação do produto.

> Os requisitos de um sistema são as descrições do que o sistema deve fazer os serviços que oferece e as restrições a seu funcionamento. Esses requisitos refletem as necessidades dos clientes para um sistema que serve a uma finalidade determinada, como controlar um dispositivo, colocar um pedido ou encontrar informações. O processo de descobrir, analisar, documentar e verificar esses serviços e restrições é chamado de engenharia de requisitos. (Sommerville, 2011)

A figura a seguir tem relação com a obtenção, documentação e verificação dos requisitos, quando colocados em prática os requisitos sofrem mudanças e com isso a visão dos envolvidos no projeto acaba mudando, em vista disto, são feitas várias alterações no hardware, software e ambiente organizacional onde o sistema está envolvido.

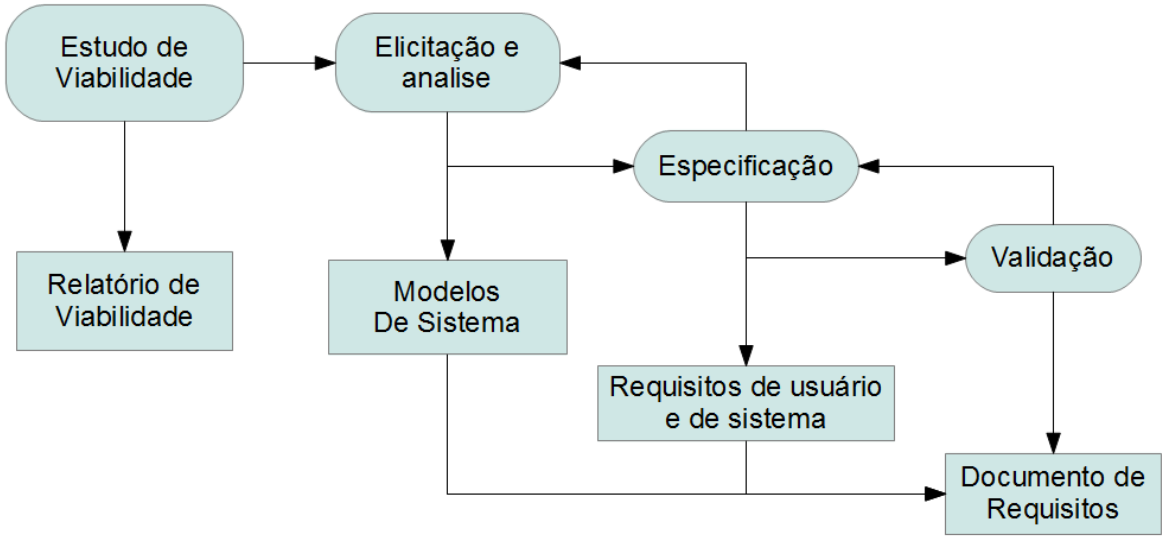

Figura 1 - Processo de Engenharia de Requisitos

Fonte: Sommerville, 2007

A Figura 1 mostra os processos da Engenharia de Requisitos, começando pelo estudo de viabilidade onde é feito o relatório de viabilidade do software,

passando para a elicitação e análise de requisitos onde tudo que todos os requisitos são idealizados, são feitos modelos do software, os requisitos são especificados e classificados em requisitos de usuário e de sistema, esses requisitos depois de classificados são mostrados ao cliente para que eles sejam validados e depois é feito o documento desses requisitos.

Os requisitos do software podem ser funcionais e não-funcionais, nas próximas seções serão descritos os dois tipos.

1.1.1 Requisitos funcionais

Requisitos funcionais descrevem as funcionalidades que o sistema terá. São definidos de acordo com o tipo de software que será desenvolvido, o tipo de usuários que vão utilizar e a finalidade do software.

Segundo Sommerville (2011) “A imprecisão na especificação de requisitos é a causa de muitos problemas da engenharia de software. É compreensível que um desenvolvedor de sistemas interprete um requisito ambíguo de uma maneira que simplifique sua implementação”. O cliente às vezes opta por outro caminho, fazendo com que outros requisitos sejam definidos, com isso novas alterações serão feitas no software, gerando atrasos na entrega e aumento dos custos.

> A especificação dos requisitos funcionais de um sistema deve ser completa e consistente. Completude significa que todos os serviços requeridos pelo usuário devem ser definidos. Consistência significa que os requisitos não devem ter definições contraditórias. Na prática, para sistemas grandes e complexos, é praticamente impossível alcançar completude e consistência dos requisitos. Uma razão para isso é que ao elaborar especificações para sistemas complexos é fácil cometer erros e omissões. Outra razão é que em um sistema de grande porte existem muitos *stakeholders.* (Sommerville, 2011).

As exigências do usuário devem ser definidas e essas definições não devem ser contraditórias. Geralmente projetos grandes são um desafio já que não é tarefa fácil alcançar a completude e consistência dos requisitos, pois como são softwares complexos eles tendem a ter erros e omissões de requisitos. Outro motivo que leva a gerar erros durante a especificação dos requisitos do usuário é o fato de existir

vários clientes de uma mesma empresa que geralmente tem opiniões diferentes sobre os requisitos do sistema.

1.1.2 Requisitos não funcionais

Requisitos não funcionais estão conectados a qualidade do software e não estão relacionados às funcionalidades específicas que o sistema oferece ao usuário. Esses requisitos estão relacionados ao tempo de resposta, confiabilidade, integridade e segurança do software.

Segundo Sommerville (2011) “Requisitos não funcionais são frequentemente mais críticos que requisitos funcionais individuais”. Um cliente e/ou usuário pode encontrar formas de contornar funções do software que não atenda suas necessidades. Porém, não atender a um requisito não funcional pode fazer com que o sistema falhe.

Os requisitos não funcionais são divididos em:

- Requisitos de produto que especificam o comportamento do sistema;
- Requisitos organizacionais que são requisitos oriundos das políticas e dos procedimentos da organização do cliente e dos desenvolvedores.
- Requisitos externos que derivam de fatores externos ao software.

Os requisitos de produto especificam o comportamento do software, como desempenho, memória alocada, entre outros. Os requisitos organizacionais são provenientes de políticas e procedimentos da empresa do cliente e da equipe de desenvolvimento, como padrões que devem ser usados e linguagem de programação. Já os requisitos externos incluem todos os requisitos que são resultados de fatores externos ao software e seu processo de desenvolvimento, são classificados externos os requisitos que devem ser seguidos para assegurar que o software funcionado dentro da lei e requisitos éticos.

Na próxima seção será falado sobre as técnicas da engenharia de requisitos tradicionais.

1.2 TÉCNICAS DE ENGENHARIA DE REQUISITOS TRADICIONAL

A engenharia de requisitos tradicional tem como objetivo esclarecer todos os passos do projeto antes mesmo de dar início ao desenvolvimento do sistema.

Todos os passos previstos antes de iniciar o desenvolvimento do software são baseados em duas suposições. Segundo D'Amorim (2008), "quanto mais tarde os erros forem descobertos maiores são os custos para corrigi-los. É possível determinar um conjunto de requisitos estáveis, antes de se começar a projetar e implementar o sistema." Portanto, as técnicas tradicionais para a elicitação de requisitos tem como objetivo prever todo o projeto através dos requisitos estáveis do software para evitar erros, mau funcionamento, entre outros.

Os métodos tradicionais são chamados também de metodologias pesadas ou orientadas a documentação. Eles surgiram na época onde os projetos de software eram diferentes da atualidade, onde fazer alterações e correções tinha um custo muito alto, pois não eram todos que tinham acesso a computadores e as ferramentas de desenvolvimento de software não existiam, fazendo com que todo o projeto fosse planejado e documentado antes mesmo de iniciar o desenvolvimento (Soares, 2004).

Com o passar dos anos os processo tradicionais foram sendo padronizados para servirem como guia para a equipe envolvida no projeto de desenvolvimento do software, por esse motivo foram criados padrões como os padrões IEEE (*Institute of Electrical and Electronics Engineers*) 830 – "Práticas Recomendadas para Especificação de Requisitos de Software" e IEEE 1233 – "Guia para Desenvolvimento de Especificação de Requisitos de Sistemas".

1.2.1 Elicitação de Requisitos

A elicitação de requisitos faz referência à identificação e delimitações do software por meio de reuniões entre clientes e desenvolvedores. Essas delimitações do software irão influenciar em todas as técnicas de elicitação e vão definir o

contexto do software que será entregue ao cliente. Entender sobre diversas áreas é imprescindível para antecipar o software que será desenvolvido (COSTA, 2011).

As técnicas mais importantes para fazer o levantamento dos requisitos são: Entrevista, *brainstorming*, cenários, etnografia e mapa mental e serão retratadas nesta seção.

1.2.1.1 Entrevista

A técnica de entrevista é considerada a mais utilizada durante a elicitação de requisitos. São praticamente obrigatórias em qualquer desenvolvimento de software, por serem levadas de uma maneira mais natural entre as pessoas envolvidas.

Geralmente, o analista de requisitos se reúne com os clientes para discutir os requisitos e a partir desses requisitos ele constrói um entendimento dos requisitos do software. Entrevistas são essenciais para ter uma compreensão com detalhes do problema e para captar diversos requisitos gerais do software.

> Na fase inicial da elicitação, o desenvolvedor pode entrevistar os usuários para obter informações sobre o trabalho que eles executam. Além disso, nos primeiros encontros as questões tendem a ser mais genéricas e tornam-se mais específicas nos encontros posteriores, conforme o desenvolvedor vai gradualmente compreendendo o domínio e fique claro que tipo de informações ele realmente necessita obter dos usuários (Silva 2009).

As entrevistas podem ser usadas durante todo o decurso da Engenharia de Requisitos em projetos pequenos ou grandes, onde a fonte da captação de requisitos sejam os próprios clientes e/ou usuários. O tipo de entrevista, o nível de detalhes e a validade dos requisitos captados, vão depender do tipo de entrevista e com a experiência do responsável por entrevistar os usuários.

1.2.1.2 Brainstorming (Tempestade cerebral)

O brainstorming é uma técnica que possui a possibilidade de desenvolver maneiras inovadoras pertinentes com o tema estabelecido. Geralmente, essa técnica é feita em grupo, mas também pode ser feita individualmente.

Normalmente dividido em duas fases, o brainstorming tem fase de geração de ideias, onde são reunidas, e a fase onde é avaliada e discutida cada ideia. Na fase onde as ideias são estabelecidas, não é aconselhado criticar e nem avaliá-las, deixar para fazer a avaliação das ideias na segunda fase. Cada ideia pode levar a novas ideias e assim sucessivamente.

Segundo Costa (2011) “A técnica de brainstorming leva a uma melhor compreensão do problema para todos e um sentimento de que todos cooperaram para atingir o objetivo”.

O brainstorming é uma técnica que pode aproximar a equipe de desenvolvimento através das ideias que forem surgindo e todos podem entender melhor o que o cliente está pedindo para ser desenvolvido.

1.2.1.3 Etnografia

A etnografia é uma técnica de elicitação de requisitos organizacionais voltada para observação e os requisitos de sistema que possam ser esquecidos. Ela se baseia no analista de requisitos ou a pessoa responsável pela elicitação de requisitos fica observando o ambiente de trabalho para onde o software será desenvolvido (COSTA, 2008).

O AR (Analista de Requisitos) analisa a rotina de trabalho da equipe envolvida, tomando nota e planejamento imagens de como funciona o trabalho dessa equipe. A etnografia auxilia o analista a descobrir e compreender os requisitos reais e implícitos que o software deverá ter.

Oliveira (2011) comenta que dentre as orientações para a etnografia, deve-se preocupar sempre em procurar formas não padronizadas de trabalho, tomar nota de forma detalhada de todas as práticas de trabalho, analisando-as e chegando a uma

conclusão e, sempre que possível, combinar a etnografia com entrevistas abertas e outras técnicas de elicitação.

A técnica de etnografia podem mostrar detalhes significativos do processo que são frequentemente ignorados pelas outras técnicas de elicitação. Entretanto, por se tratar de uma técnica com foco no usuário final, essa abordagem não é adequada para obter requisitos organizacionais ou de domínio, não identificam novas características para serem acrescentadas a um software, portanto não é uma abordagem completa para elicitação e deve ser usada apenas para completar outras técnicas.

1.2.1.4 Mapa Mental

A técnica mapa mental consiste em um método que auxilia no armazenamento, organização e priorização de informações. Utilizando palavras e imagens para facilitar a lembrança de ideias específicas.

Os mapas mentais tem o objetivo de auxiliar a composição de pensamentos, rapidamente. É considerado por Matuda (2010) um método que aguça a criatividade, e estimula maior o uso do potencial do cérebro, visto que, usamos apenas uma porção de nossa capacidade cerebral.

Mapas mentais é um método eficiente que visa auxiliar no armazenamento, organização e priorização de informações. São usadas palavras e imagens – chave, que auxiliam na lembrança de ideias específicas.

Os mapas tem uma maneira diferente da forma ocidental de ler e escrever e possui características peculiares de organização. É composto por um tema central cujas informações se proliferam do centro para as extremidades, tornando-se assim mais atrativo para o entendimento humano. A técnica usa várias cores, desenhos, símbolos e imagens para ser mais estimulante (Matuda, 2010).

Na técnica mapas mentais o analista poderá utilizar qualquer uma das técnicas para levantar requisitos e as particularidades de cada um e o mapa mental que é montado direciona o analista para a prática de elicitação de requisitos que seja mais adequada para aquele determinado projeto.

1.2.2 Documentação

Assim que os clientes chegam num consenso com a empresa sobre os requisitos do software, o que foi combinado entre as duas partes deve ser ajustado e retratado em um documento de requisitos.

Esta prática foi nomeada como documentação de requisitos. O documento gerado tem como particularidade um grau de formalidade e detalhamento que irá depender de dois fatores: riscos identificados para o projeto do sistema e do nível de experiência e habilidades dos seus prováveis leitores (Alves, 2009).

O nível de detalhamento do documento de requisitos dependerá do tipo de software que está sendo desenvolvido e do processo d desenvolvimento utilizado. Quando o software for desenvolvido por uma empresa externa, as especificações de sistema crítico devem ser precisas e detalhadas. Quando ocorrer uma flexibilidade maior dos requisitos e quando um processo de desenvolvimento interno e iterativo for usado, o documento de requisitos poderá conter menos detalhes e qualquer ambiguidade será resolvida durante o desenvolvimento do sistema (Sommerville, 2011).

A documentação de requisitos é indispensável quando uma empresa externa está desenvolvendo o software. Porém, as metodologias ágeis de desenvolvimento argumentam que os requisitos mudam tão rapidamente que um documento de fica desatualizado desde o momento que o mesmo é redigido, por isso o esforço é desperdiçado (Sommerville, 2007).

A documentação de requisitos servirá como uma espécie de guia onde toda a equipe poderá consultar ao longo de todo o desenvolvimento do software, pois é muito fácil os envolvidos no projeto esquecerem os requisitos que são aplicados ao software como um todo quando focam nos requisitos funcionais para a próxima versão daquele mesmo sistema.

O documento de requisitos é a declaração aprovada dos requisitos de sistema e deve ser organizado de maneira que tanto clientes quando equipe de desenvolvimento consigam entende-lo e utilizá-lo.

1.2.3 Verificação/Validação de requisitos

Sommerville (2011) afirma que, “a validação de requisitos dedica-se a mostrar que os requisitos realmente definem o sistema que o usuário deseja. A validação de requisitos se sobrepõe à análise: se está relacionada à descoberta de problemas com os requisitos”.

Segundo Sommerville (2007), durante a fase de validação dos requisitos, é necessário fazer algumas verificações na documentação de requisitos, são elas:

- Verificação de validade: Um usuário pode pensar que um software é indispensável para realizar determinadas funções. Porém, estudando e analisando mais a fundo é possível identificar quais são as funções adicionais e diferentes que serão necessárias.
- Verificação de consistência: Os requisitos não podem entrar em conflito na documentação, ou seja, não podem existir restrições ou descrições contraditórias para uma mesma função do software.
- Verificação de completeza: No documento de requisitos deve estar incluso os requisitos que descrevam todas as funções e as restrições exigidas pelo cliente.
- Verificações de realismo: Para saber se aqueles requisitos podem ser desenvolvidos, é necessário verifica-los. Essas verificações devem levar em consideração o orçamento e o prazo para desenvolver o software.
- Facilidade de verificação: Para reduzir o potencial desacordo entre cliente e desenvolvedores, os requisitos devem ser escritos de modo que sejam fáceis de verificar.

A validação dos requisitos é uma fase importante, pois os erros em uma documentação de requisitos podem acarretar custos excessivos de retrabalho quando são descobertos durante a fase de desenvolvimento do software ou até depois que o mesmo está operando. Fazer uma correção de um problema de requisitos, mudando o software, custa muito mais caro do que a correção de erros do projeto, pois uma mudança nos requisitos acarreta alteração no software, que deverá ser testado novamente.

1.2.4 Gerenciamento de requisitos

Os requisitos dos softwares de grande porte mudam constantemente por serem desenvolvidos para lidar com diversos tipos de problemas. Como esses problemas não são totalmente definidos, os requisitos acabam sendo incompletos. Durante o desenvolvimento do sistema os clientes acabam entendendo mais sobre o problema que acaba mudando constantemente. Esses requisitos devem evoluir para acompanhar as novas visões do problema. Após o sistema instalado, novos requisitos surgirão, pois é difícil para usuários e clientes do software anteciparem os efeitos de um novo software na empresa. Com isso novas necessidades e prioridades serão descobertas (Sommerville, 2007).

Os requisitos para softwares baseados em computadores sofrem alterações, e o desejo de modificar os requisitos insiste ao longo da vida de um software. O gerenciamento de requisitos consiste em uma série de atividades que auxilia a equipe de projetos a reconhecer, controlar e acompanhar as necessidades e as atualizações que o software necessite em qualquer momento mesmo que o projeto avance (Pressman, 2011).

A gerência de requisitos é uma etapa onde é necessário compreender e controlar as mudanças de requisitos nos softwares. É necessário acompanhar os requisitos individuais e manter a ligação com os requisitos dependentes, de maneira que seja viável avaliar o impacto que a mudança nos requisitos causa. O processo de gerenciamento de requisitos deve se iniciar logo depois que tenha disponível uma versão do documento de requisitos, porém o planejamento das mudanças deve ocorrer durante o levantamento dos requisitos.

A próxima seção irá falar sobre os métodos de desenvolvimento tradicionais existentes.

1.3 METODOLOGIAS DE DESENVOLVIMENTO TRADICIONAIS

Metodologias de desenvolvimento tradicionais são conjuntos de práticas sugeridas para o desenvolvimento de software. Essas metodologias podem ser divididas em fases, para ordenar e gerenciar todas as fases.

Os métodos tradicionais possuem como característica principal, a divisão do projeto em fases ou etapas. Essas fases são estabelecidas e incorporam atividades como análise, modelagem, desenvolvimento e testes.

No final de cada uma dessas fases é gerado um marco em forma de documento, como diagramas de casos de uso ou uma versão do software (Utida, 2012).

Metodologias pesadas geralmente são desenvolvidas no “modelo em cascata”, e a cada alteração do projeto, será necessário à volta ao inicio do projeto para alteração da documentação ou de outro marco (Pressman, 2011).

A seguir nessa seção serão expostos com mais detalhes os modelos de processo utilizados no desenvolvimento tradicional de software. São eles: Modelo Cascata, Modelo Incremental, Modelo de Prototipagem, Modelo Espiral, Modelo de desenvolvimento concorrente, RUP e suas fases.

1.3.1 Modelo em Cascata

O modelo cascata, conhecido também como ciclo de vida clássico, sugere uma abordagem tradicional para o desenvolvimento de software, partindo do levantamento dos requisitos por parte do cliente, avançando pelas fases de planejamento, modelagem, construção, emprego e resultando no suporte contínuo do software final. A Figura abaixo mostra os passos do modelo cascata.

Segundo Pressman (2011) “O modelo cascata é o paradigma mais antigo da engenharia de software. Entretanto, ao longo das ultimas décadas, as críticas a este modelo de processo fez com que até mesmo seus mais ardentes defensores questionassem sua eficácia”.

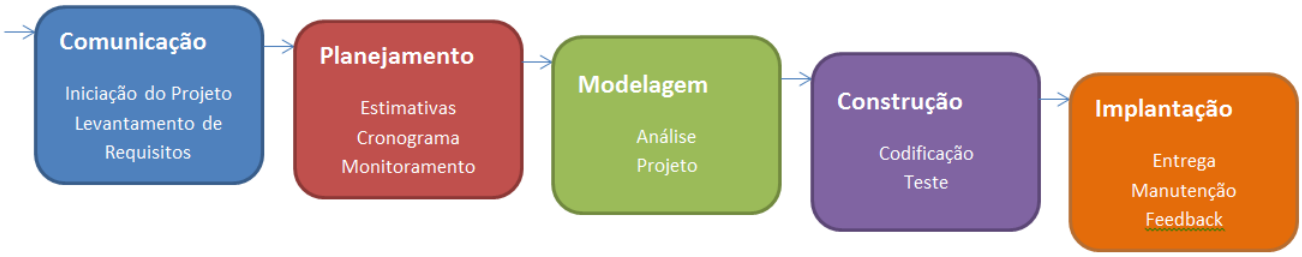

Figura 2 - Modelo Cascata

Fonte: Sommerville, 2011

A Figura 2 mostra as fases do modelo de desenvolvimento cascata, começando com a comunicação entre equipe e cliente onde é dado início ao projeto de software e é feito o levantamento dos requisitos, logo após é feito o planejamento de custos, o cronograma do projeto e o monitoramento. Durante a modelagem do software é feito a análise dos requisitos e a documentação do projeto, depois é feita a construção do software onde ele será desenvolvido e testado pela equipe de desenvolvimento. Por fim, a implantação daquele sistema, nessa fase o software é entregue ao cliente, é feita manutenções sempre que necessário e a empresa recebe um *feedback* do sistema desenvolvido.

1.3.2 Modelo Incremental

Esse modelo é uma adaptação do “Modelo Sequencial Linear”. De acordo com Pressman (2011) “os requisitos iniciais do software são razoavelmente bem definidos, entretanto, devido ao escopo geral do trabalho de desenvolvimento, o uso de um processo puramente linear não é utilizado”.

O modelo incremental combina parte dos fluxos de processos lineares e paralelos. Ele aplica sequências lineares, de forma escalonada, à medida que o tempo vai se estendendo. Cada uma dessas sequências lineares realizam progressos do software de modo semelhante aos incrementais gerados por um fluxo de processos evolucionários (Pressman, 2011).

O modelo incremental é mais utilizado em projetos longos, nos quais novas funcionalidades são acrescentadas ao longo do tempo e o prazo de entrega é curto.

Como há uma versão estável do produto ao final de cada incremento, o cliente vê a evolução do produto (Utida, 2012).

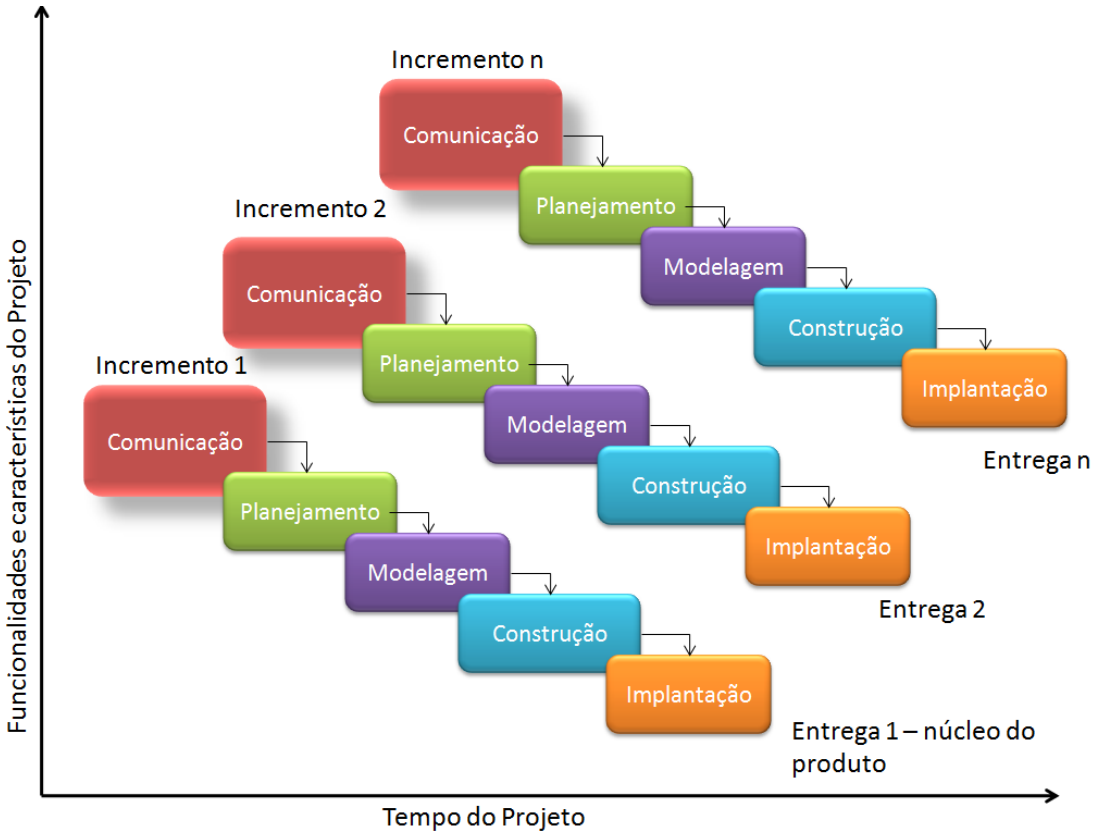

Figura 3 - Modelo Incremental

Fonte: Pressman, 2011

A Figura 3 mostra todo o processo do Modelo Incremental. Em cada um dos incrementos é realizado todo o ciclo do desenvolvimento de software, do planejamento aos testes do sistema já em funcionamento. Cada etapa produz um sistema totalmente funcional, mesmo não cobrindo todos os requisitos estabelecidos no início do projeto.

1.3.4 Modelo de Prototipagem

De acordo com Utida (2012), esse modelo é utilizado quando alguns requisitos de sistema não são definidos de maneira clara pelo cliente, apenas seus

objetivos, e não como os dados serão processados e como a saída será demonstrada.

Apesar de a prototipação ser mais usada como um modelo de processos isolado é mais comum ser usada como um método passível de ser implementado no contexto de qualquer um dos modelos tradicionais. Independente da maneira como a prototipagem é aplicada, quando os requisitos não estão claros, o modelo da prototipação ajuda os interessados a ficar a par do que está sendo desenvolvido (Pressman, 2011).

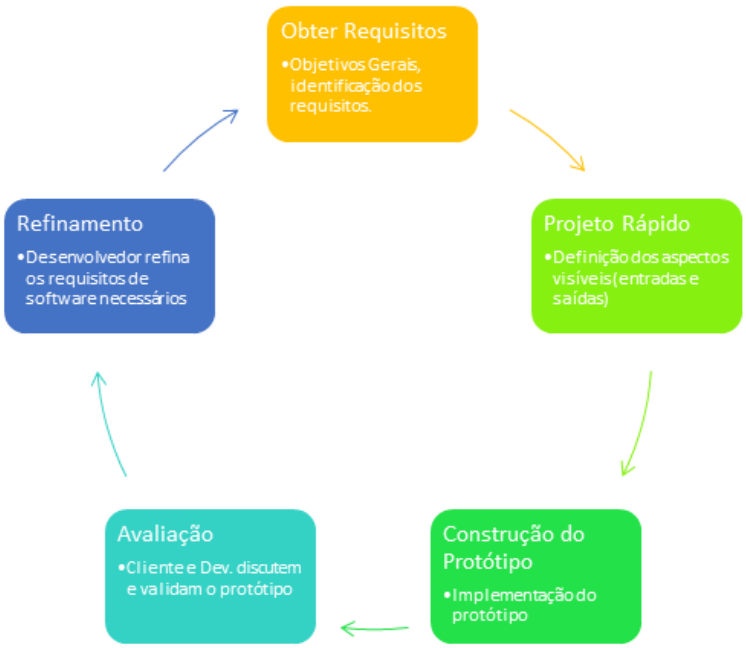

Figura 4 - Modelo de prototipagem

Fonte: Pressman, 2011

O modelo de prototipagem representado na Figura 4 mostra todas as fases do processo de prototipação. O ciclo começa na comunicação com os clientes, onde são obtidos os requisitos e objetivos gerais do software proposto, seguindo o ciclo é definido os aspectos visíveis do software, as entradas e saídas. Logo após é iniciada a implementação do protótipo do sistema para futuramente o cliente e equipe de desenvolvimento discutir sobre essa prévia do software e fazer a avaliação do mesmo. Por fim é feito o refinamento dos requisitos necessários no software e excluem aqueles que não são necessários no software.

1.3.5 Modelo Espiral

Considerado como uma forma melhor elaborada do método cascata, o modelo incremental é baseado em uma sequência de fases que culminam em versões incrementais do software (Bona, 2002).

O modelo espiral de desenvolvimento é um gerador de modelos de processos, dirigidos a riscos e é usado para conduzir a engenharia de sistemas intensivos de software, que ocorrem de forma concorrente e tem múltiplos envolvidos (Pressman, 2011).

Na Figura 5 é apresentado o modelo espiral. Nele os processos são divididos em três estágios e organizados de maneira iterativa em espiral. O esforço e tempo determinado para cada estágio dependerá do tipo de software.

> O modelo em espiral acomoda abordagens de desenvolvimento em que os requisitos são desenvolvidos para diferentes níveis de detalhes. O número de iterações na espiral pode variar, de modo que pode-se sair da espiral depois que alguns ou todos os requisitos de usuário tiverem sido elicitados (Sommerville, 2007).

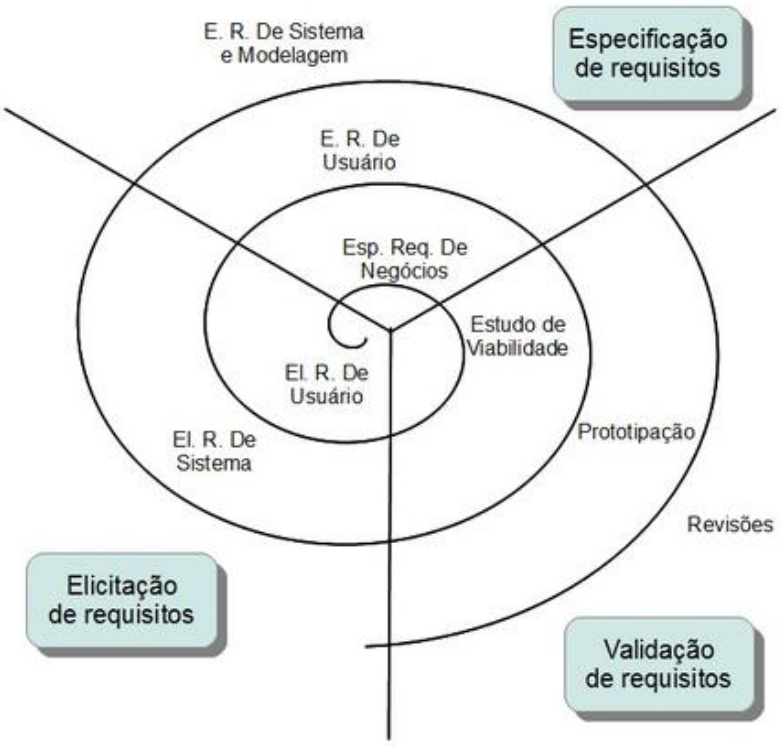

Figura 5 - Modelo em espiral dos processos de Engenharia de Requisitos

Fonte: Sommerville, 2011

A Figura 5 representa o modelo espiral. O processo começa pelo centro do espiral na elicitação de requisitos de usuário e em sentido horário são feitas as outras tarefas como a especificação dos requisitos de negócios, estudo de viabilidade, após esse estudo é feita a elicitação de requisitos de sistema e novamente é feita a elicitação de requisitos de usuário, sempre evoluindo e melhorando software, a medida que o software vai evoluindo são feitos ajustes no planejamento, custo e cronograma de acordo com o *feedback* dos clientes.

1.3.6 RUP – RATIONAL UNIFIED PROCESS

O RUP foi desenvolvido pela *Rational Software* e comprado pela IBM que hoje o mantêm. Ele foi criado para estender a produtividade da equipe de desenvolvimento de software, sem perder a qualidade do produto. Esta metodologia é um framework flexível, podendo ser usado tanto por equipes grandes ou pequenas.

Segundo Utida (2012), o RUP é uma metodologia que iterativa, que trabalha com ciclos, onde cada ciclo se dedica a uma nova versão do software. Cada ciclo é dividido em quatro fases, sendo elas: Concepção, elaboração, construção e transição.

O principal objetivo do RUP é atender as necessidades dos usuários garantindo um desenvolvimento de software de qualidade, que consiga cumprir um cronograma e um orçamento previsível.

É considerado um processo pesado e/ou tradicional por vários especialistas e é mais utilizado por grandes equipes de desenvolvimento e a grandes projetos de software (Alves, 2012).

O RUP possibilita decidir quem será o responsável por cada tarefa que tenha q ser feita, como essa tarefa será feita e quando ela deverá ser feita, apresentando todas as metas peculiares de desenvolvimento para que as mesmas sejam completadas.

1.3.6.1 FASES DO RUP

De acordo com Alves (2012), o RUP ordena o desenvolvimento de software em quatro frases. Cada uma dessas fases possui um papel essencial para que todas as metas traçadas e o software sejam feitas com qualidade.

- **Concepção ou iniciação:** A iniciação engloba a comunicação com o cliente e o planejamento do software. Nessa fase é onde acontece a identificação e a especificação dos casos de uso do projeto. É também feita nessa primeira fase um plano de projeto, onde são analisados os riscos, delimitados os custos e os prazos finais e onde são estabelecidas as prioridades do software.
- **Elaboração:** Na segunda fase é realizada uma análise de toda a extensão do software. É definida também qual arquitetura será utilizada no sistema. Geralmente, essa arquitetura deverá ser estável e robusta. Logo após definir a arquitetura, será feita outras análises, construção dos projetos dos casos de uso que posteriormente serão documentados.
- **Construção:** Na fase de construção é onde é feito o desenvolvimento de todos os recursos do software. É nessa fase que acontece a produção de códigos e os testes com prévias do software, para atestar que os casos de uso estão sendo desenvolvidos perfeitamente. É desejável executar processos de testes estáveis.
- **Transição:** O objetivo dessa fase é certificar que o software estará disponível para os usuários finais. As tarefas nessa fase abrangem o treinamento dos usuários finais e os testes finais do software para que o mesmo tenha os níveis de qualidade adequados.

A Figura 6 irá ilustrar as fases do RUP citadas nos tópicos: Concepção ou iniciação, Elaboração, Construção e Transição.

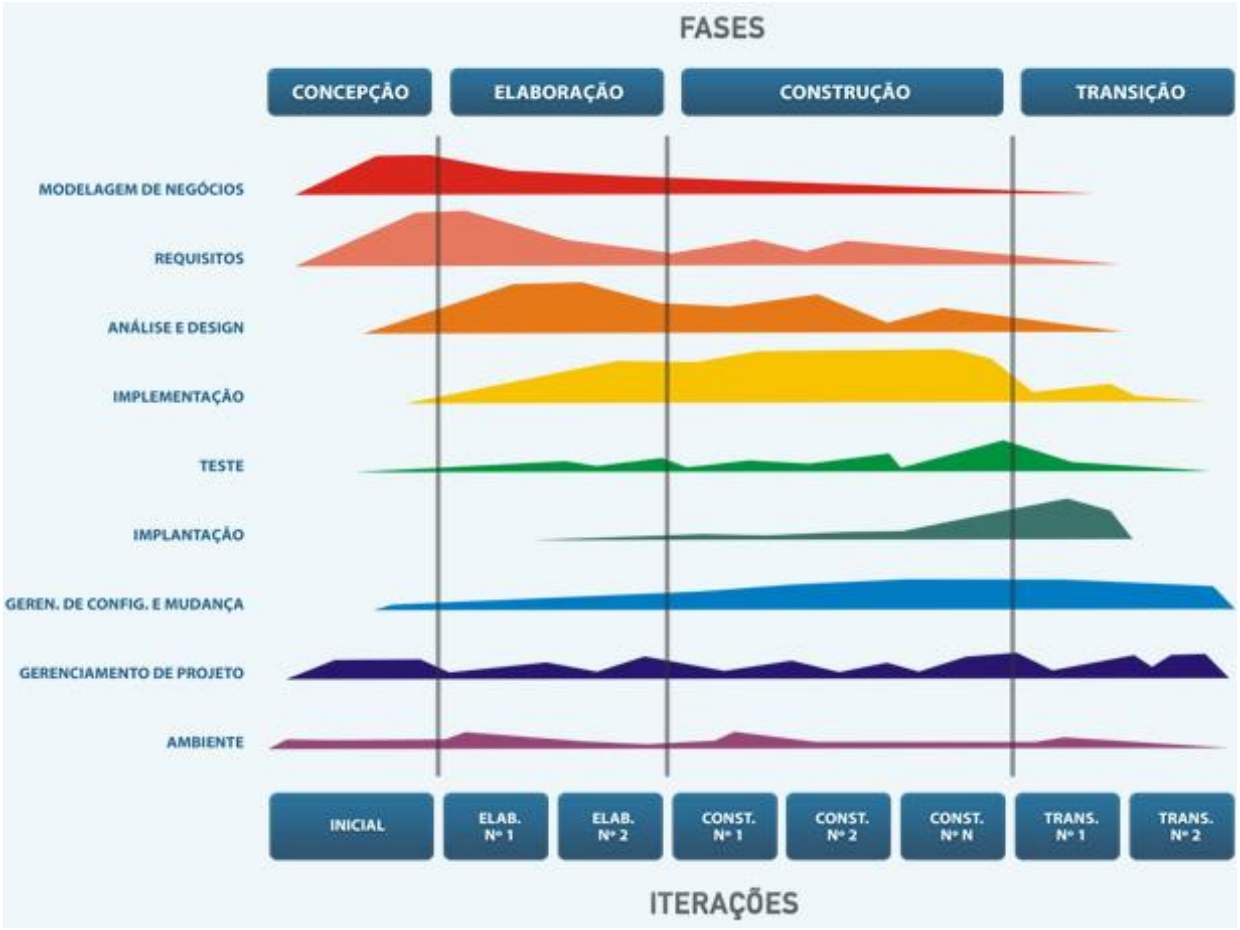

Figura 6 - As fases do RUP

Fonte: Utida, 2012

A Figura 6 mostra as fases do RUP que possui duas dimensões, a horizontal representa o tempo e ciclo de vida do processo, os aspectos dinâmicos do processo caracterizado por ciclos, fases, iterações e pontos de controle e a vertical, representando o fluxo do processo, chamadas de disciplinas onde aparece os aspectos estáticos do processo e é descrito através de atividades, disciplinas, artefatos e regras. O RUP propõe que um software seja construído através de uma sucessão de iterações incrementais.

A próxima seção irá mostrar a evolução das metodologias de desenvolvimento a partir do Manifesto Ágil criado em 2001.

1.4 MANIFESTO ÁGIL

O termo “Metodologias Ágeis” se tornou conhecido e vem se expandindo entre empresas de desenvolvimento devido ao Manifesto Ágil que foi criado em 2001. O foco nas pessoas e interações cria novas situações para se abordar requisitos e certas práticas são aconselhadas para obter sucesso desta atividade (Alves et. al. 2009).

Utida (2012) afirma que “É importante entender que o manifesto ágil não nos diz para esquecer os processos e ferramentas, a documentação, a negociação ou o planejamento, mas devemos tratá-los de maneira diferente, priorizando o foco em outros conceitos”.

Os princípios comuns compartilhados por todos esses métodos são: “Indivíduos e interações são mais importantes que processos e ferramentas”, “Software funcionando é mais importante do que documentação completa e detalhada”, “Colaboração com o cliente é mais importante do que negociação de contratos” e “Adaptação a mudanças é mais importante do que seguir o plano inicial”. Eles foram estabelecidos através do Manifesto Ágil e serão citados ao decorrer dessa seção.

1.4.1 Indivíduos e interações são mais importantes que processos e ferramentas

Segundo Utida (2012), “a capacitação e qualidade dos profissionais envolvidos no desenvolvimento do projeto implica diretamente nos resultados do produto tanto na qualidade como no desempenho no decorrer do projeto”. Porém, ter o melhor time de profissionais não é garantia de sucesso, este depende do processo e em um processo ruim, até os melhores profissionais não terão capacidade de obter esse sucesso.

Outro fator que é importante lembrar, é que mesmo uma empresa com ótimos profissionais e um bom processo, ainda é necessário a boa comunicação entre todos da equipe.

> Trabalhando em equipe, é mais produtivo um desenvolvedor mediano que saiba se comunicar com o restante do time do que um talento na programação que não consiga isto e trabalhe sozinho. Vale lembrar que as ferramentas utilizadas são importantes e influenciam para o sucesso final do projeto, mas ainda não devem ser mais importantes que a qualidade e o nível de interação da equipe (Utida, 2012).

O princípio citado defendido pelo Manifesto Ágil afirma que um indivíduo que saiba se comunicar bem acaba sendo produtivo para todo o projeto e vale mais a pena do que ter um ótimo profissional que não saiba se comunicar. Outro fator que leva a ter sucesso durante o projeto são as ferramentas que aumentam a qualidade do sistema e o nível de comunicação entre a equipe.

1.4.2 Software funcionando é mais importante do que documentação completa e detalhada

Em um projeto tradicional, a documentação é extremamente importante para obter um software de sucesso, principalmente porque uma documentação bem feita do software ainda é de suma importância para auxiliar nas tomadas de decisão no decorrer de seu desenvolvimento e, além disso, é mais simples para entender as funcionalidades do software no documento do que ter que ler diretamente no código.

É necessário ficar atento com o excesso de documentos, documentação a mais pode ser pior do que a falta dela, pois se a documentação não estiver em sincronia com o progresso do projeto, elas acabam distorcendo a realidade, acarretando em tomada de decisões erradas (Utida, 2012).

O manifesto recomenda que somente a documentação necessária seja feita, e a mesma deve estar sempre em sincronia com o sistema. Com o mínimo de documentação possível é necessário um bom relacionamento da equipe, pois o conhecimento sobre o projeto é transmitido durante o seu desenvolvimento, trabalhando em equipe.

1.4.3 Colaboração com o cliente é mais importante do que negociação de contratos

Para o desenvolvimento de um software de boa qualidade e de sucesso é necessário passar pela aceitação do cliente, já que ele que vai dar um *feedback* contínuo das prévias do software, garantindo que o software que esteja sendo desenvolvido atenda as necessidades atuais.

Ao contrário das metodologias tradicionais, nas metodologias ágeis os contratos devem determinar a forma que ocorrerá a comunicação entre cliente e equipe de desenvolvimento, o custo do projeto, os prazos e requisitos não são especificados, pois durante o projeto alguns requisitos podem ser desnecessários e também pode surgir a necessidade de adicionar outros que não foram descritos no contrato (Utida, 2012).

O Manifesto ágil defende que os contratos sejam mais simbólicos, não especificados os requisitos, pois nas metodologias ágeis os requisitos são modificados frequentemente e com isso o prazo e custo podem ser menores ou maiores dependendo do projeto.

1.4.4 Adaptação a mudanças é mais importante do que seguir o plano inicial

Os requisitos são instáveis e várias mudanças vão ocorrer, o melhor a se fazer é desenvolver um projeto que possa se adaptar as mudanças. O planejamento pode ocorrer, mas valendo para um período menor. Uma vez que, é inevitável ter alterações no projeto e planos extensos serão mais difíceis de ser consolidados (Utida, 2012).

O manifesto defende que uma equipe que trabalha com desenvolvimento ágil não deverá pensar em sempre seguir roteiros e planos, o correto a se fazer é sempre ter opções e criatividade para mudanças repentinas nos requisitos e se adaptarem àquelas mudanças fazendo com que o projeto não pare e que não perca a qualidade do software final.

1.5 METODOLOGIAS DE DESENVOLVIMENTO ÁGEIS

As metodologias ágeis surgiram em meados da década de 90, quando vários desenvolvedores viram que a abordagem da Engenharia de Software era bem pesada. Esses métodos fizeram com que a equipe responsável pelo software focasse no seu desenvolvimento, e não em sua compreensão e documentação. Segundo Sommerville (2011) “Métodos ágeis, universalmente, baseiam-se em uma abordagem incremental para a especificação, o desenvolvimento e a entrega do software”.

Os “MA’s” (Métodos Ágeis) chegaram para melhorar o processo de desenvolvimento dos softwares, já que eles são mais adequados em projetos onde os requisitos do sistema mudam rapidamente durante o desenvolvimento do sistema e também por propor a entrega do software em um prazo menor, em funcionamento e podendo sofrer alterações posteriores sem prejudicar o sistema por completo.

Segundo Sommerville (2011) Os métodos ágeis têm como princípios:

- Envolvimento do cliente: Os clientes devem estar mais envolvidos diretamente no decorrer do projeto, fornecendo e priorizando novos requisitos e avaliando as alterações no produto, com isso o produto final será melhor e funcional.
- Entrega incremental: O sistema é desenvolvido em fases juntamente com o cliente, que especifica os requisitos que serão incluídos em cada uma das fases.
- Pessoas, não processos: Cada membro da equipe deve ter suas habilidades reconhecidas e exploradas. Devem desenvolver cada um a sua maneira, sem instruções prévias.
- Aceitar as mudanças: O software deve ser projetado para aceitar mudanças, pois os requisitos sempre sofrem alterações.
- Manter a simplicidade: Projetar e desenvolver um software focado na simplicidade. Buscando sempre eliminar a complexidade do sistema.

Os métodos ágeis existentes hoje carregam os valores e princípios do Manifesto Ágil, métodos como XP (*Extreme Programming*), Scrum, FDD e Crystal os trazem por isso são considerados ágeis (Bernardo, 2014).

A finalidade do desenvolvimento ágil é expandir a capacidade de reação e responder a mudanças de requisitos, de ambiente, de clientes e necessidades tecnológicas em todos os níveis organizacionais (Coelho, 2011).

Nas seções a seguir será falado sobre quatro metodologias ágeis, são elas: XP, Scrum, Crystal e FDD.

1.6 XP - EXTREME PROGRAMMING

O XP (*Extreme Programming*) é uma metodologia ágil que se define em cinco valores, são eles comunicação, simplicidade, *feedback*, coragem e respeito. Esses valores são usados como um guia das atividades e tarefas específicas do XP.

Essa metodologia faz sucesso por ajudar a criar softwares de melhor qualidade, desenvolvidos em menos tempo e com um orçamento menor. Seus objetivos são alcançados através de um pequeno conjunto de valores, princípios e práticas, que diferem substancialmente da forma tradicional do desenvolvimento de um software. No XP o que realmente importa não é como uma pessoa se comporta e sim como os indivíduos se comportam como uma equipe e como organização.

Segundo Pressman (2011) “A Extreme Programming emprega uma abordagem orientada a objetos como seu paradigma de desenvolvimento preferido e envolve um conjunto de regras e práticas constantes no contexto de quatro atividades metodológicas: planejamento, projeto, codificação e testes”.

1.6.1 Valores da XP

Segundo Pressman (2011) o XP possui cinco valores, são eles:

- **Comunicação:** Esse valor tem como objetivo de sustentar uma relação direta entre cliente e equipe de desenvolvimento, assim todos os envolvidos no projeto fazem parte da equipe, trabalhando junto para solucionar todos os problemas, resultado em um software de ótima qualidade.

- **Simplicidade:** O XP incentiva as práticas que possam reduzir a complexidade do software. O projeto deve ser traçado utilizando a solução mais simples possível e ao mesmo tempo alcance os resultados esperados. Simplificar as fases do projeto sempre que possível e descartar coisas que sejam relacionadas com funcionalidades futuras.
- ***Feedback*:** A equipe de desenvolvimento consegue o *feedback* através de testes do software, que são realizados durante todo o desenvolvimento do software. Entregas de prévias permitem a verificação do entendimento e mudanças de requisitos se forem necessárias.
- **Coragem:** Seguindo as práticas propostas pelo XP, o desenvolvedor se sentirá mais confiante, o que vai lhe ajudar em alguns momentos, por exemplo, modificar o código que já esteja funcionando para torna-lo mais simples, descartar código desnecessário, investir em teste de desenvolvimento, pedir ajuda a pessoas da equipe que sejam mais experientes, falar a um cliente que um determinado requisito não será desenvolvido no prazo estipulado e abandonar processos formais e fazer documentação na forma de código.
- **Respeito:** Toda a equipe envolvida no projeto se respeita e contribui de forma individual ou em grupo. Desenvolvedores e clientes também se respeitam e aceitam a experiência de cada um.

Os princípios chaves e as práticas que consistem o XP, não tem nada de novo quando são analisadas separadamente. As práticas comuns dos métodos ágeis foram reunidas e alinhas de maneira que as mesmas se tornaram independentes. E assim surgiu uma nova metodologia de desenvolvimento de software, o XP.

1.6.2 Os processos do XP

> “Você codifica porque se você não codificar você não terá nada. Você testa porque se você não testar você não saberá quando você terminou de

> codificar. Você ouve porque se você não ouvir você não saberá o que codificar ou o que testar. E você projeta para que você possa codificar, testar e ouvir indefinitivamente" (BECK, 2004)

De acordo com Pressman (2011), a XP emprega um tratamento orientado a objetos como seu paradigma de desenvolvimento preferido e envolve um conjunto de regras e práticas constantes no contexto de quatro atividades metodológicas: planejamento, projeto, codificação e testes.

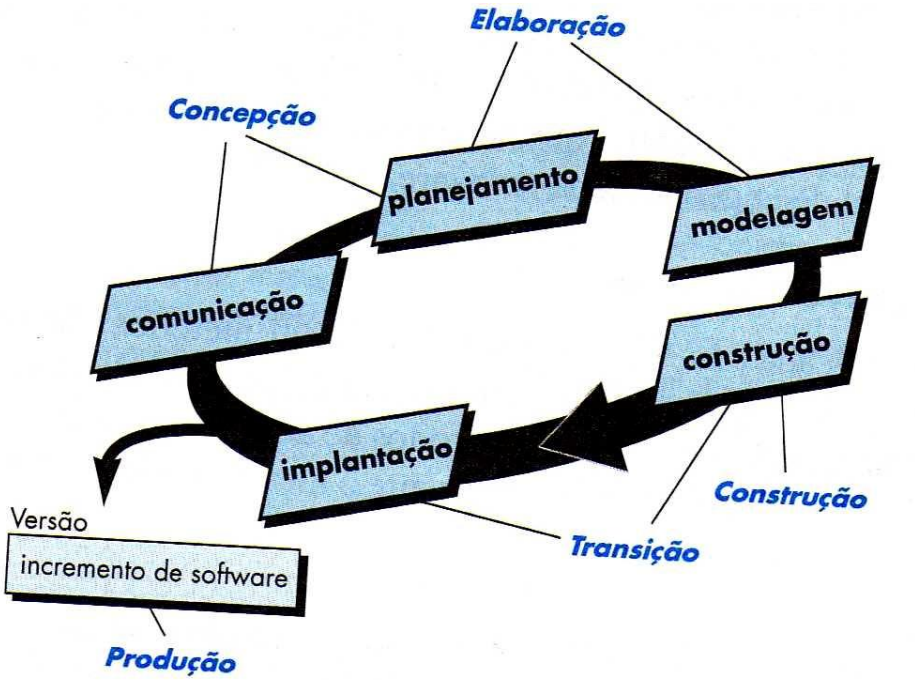

Figura 7 - Processos do XP

Fonte: Pressman, 2011

A Figura 7 ilustra o processo XP e destaca alguns conceitos e tarefas-chave associados a cada uma das atividades metodológicas. Logo após as atividades serão sintetizadas nos parágrafos a seguir.

1.6.2.1 Planejamento

O planejamento inicia com a atividade de ouvir, onde a equipe passa por uma capacitação onde eles irão entender mais sobre o ambiente de negócios do software

e possibilita que se consiga ter uma percepção ampla dos resultados que deverão ser alcançados, fatores principais e funcionalidades do software. Após a primeira atividade é hora de criar conjunto de histórias de usuários, que descreverão os resultados, características e as funcionalidades necessárias para o desenvolvimento do software. Cada história é escrita pelos próprios clientes e é colocada em uma ficha, onde posteriormente serão classificadas por prioridade. De acordo com essa classificação por prioridade, os membros da equipe pegam cada uma dessas histórias e atribuem a elas um custo, que é medido em semanas que aquela função levará pra ser desenvolvida (Pressman, 2011).

> Clientes e desenvolvedores trabalham juntos para decidir como agrupar histórias para a versão seguinte a ser desenvolvida pela equipe XP. Conseguindo chegar a um compromisso básico para uma versão, a equipe XP ordena as histórias a ser desenvolvidas em uma das três formas: todas serão implementadas imediatamente, as histórias de maior valor serão deslocadas para cima no cronograma e implementadas primeiro ou as histórias de maior risco serão deslocadas para cima no cronograma e implementadas primeiro.
> Depois de a primeira versão do projeto ter sido entregue, a equipe XP calcula a velocidade do projeto. De formas simples, a velocidade do projeto é o número de histórias de clientes implementadas durante a primeira versão. Assim, a velocidade do projeto pode ser utilizada para ajudar a estimar as datas de entrega e o cronograma para versões subsequentes e determinar se foi assumido um compromisso exagerado para todas as histórias ao longo de todo o projeto de desenvolvimento. Se ocorrer um exagero, o conteúdo das versões é modificado ou as datas finais de entrega são alteradas. (Pressman, 2011).

Conforme o desenvolvimento do software avança, o cliente poderá acrescentar histórias de usuário, alterar uma existente, dividir ou eliminar alguma outra. Logo após, a equipe de desenvolvimento reconsidera as versões remanescentes e modifica o projeto de acordo com as alterações feitas.

1.6.2.2 Projeto

O XP tem o foco na simplicidade do código para otimizar o tempo de desenvolvimento e para uma possível alteração futura. Sempre que a equipe de desenvolvedores encontrarem algum código complicado ele será substituído por um simples (Utida, 2012).

Escolher uma metáfora do sistema com o objetivo de explicar o projeto do software, descartando a documentação extensa. Com isso, novas pessoas poderão contribuir com o projeto rapidamente.

Pressman (2011) afirma que, um aspecto central na XP é o de que a elaboração do projeto ocorre tanto antes como depois de se ter iniciado a codificação.

1.6.2.3 Codificação

Na fase de codificação a presença do cliente é indispensável, a comunicação é muito importante e deve ser constante entre cliente e equipe de desenvolvimento. Como já foi falado anteriormente o cliente, com ajuda dos desenvolvedores, define as histórias e seus prazos.

Pequenas versões incrementadas do software são feitas para que o cliente possa ir testando as funcionalidades que ele propôs e dar um *feedback* mais rápido para a equipe. Todo código deve ser desenvolvido por duas pessoas em um único computador, com isso o software será de mais qualidade.
(Utida, 2012).

1.6.2.4 Testes

Udita (2012) afirma que todo código deve ter sua própria unidade de teste. Sempre que for adicionada uma nova funcionalidade no software, é necessário iniciar um novo ciclo de testes onde os códigos deverão ser testados em todas as unidades antes de ser lançado, assegurando que todas as funcionalidades sempre irão funcionar. Quando um *bug* é encontrado, a equipe de desenvolvimento cria um teste para assegurar que o mesmo não ocorra novamente.

Durante as reuniões, onde são planejadas novas funcionalidades, são criados testes de aceitação baseados nos cartões de histórias de usuário. Uma história de usuário não é considerada completa se não tiver sido passada pelo teste de

aceitação. Os próprios clientes são responsáveis por verificar a exatidão do teste de aceitação e rever os resultados para que decidam quais falharam no teste.

1.7 SCRUM

O Scrum é considerado uma das mais importantes metodologias ágeis. O nome inspirado na formação Scrum nos times de “Rugby” que se formavam pequenas equipes multidisciplinares. Tem como principal função o gerenciamento de projetos de organização, utilizado onde o desenvolvedor ainda não tem uma noção dos resultados que determinado projeto terá (Sommerville, 2011).

Segundo Pressman (2011), os processos são empíricos onde todos os ciclos de *feedbacks* giram em torno de um núcleo de técnicas de gerenciamento apresentadas anteriormente. Utilizado principalmente por empresas que tem certeza que o resultado será o esperado.

No Scrum os papéis são bem definidos e tem diversas etapas a serem cumpridas em prazos, visando entregar o produto de forma ágil e atendendo ao mesmo tempo as expectativas do cliente.

Segundo Pressman (2011) “O Scrum enfatiza o uso de um conjunto de padrões de processos de software que provaram ser eficazes para projetos com prazos de entrega apertados, requisitos mutáveis e críticos de negócio. Cada um desses padrões de processos define um conjunto de ações de desenvolvimento”.

1.7.1 Time Scrum

A equipe do Scrum é formada pelo cliente, equipe de desenvolvimento e pelo Scrum Master. Essa equipe possui membros independentes e autossuficientes para escolher a melhor forma de realizar seu trabalho, ao invés de serem dirigidas ou gerenciadas por outros de fora da equipe (Braz, 2013).

Membros de um time ser considerados multifuncionais significa que não existem funções específicas para cada um. Todos os membros da equipe trabalham

de forma colaborativa na execução de suas atividades, criando assim um ambiente de trabalho de propriedade coletiva.

Braz (2013) afirma que, com esse modelo de equipe, consegue-se alcançar melhor a flexibilidade, a criatividade e a produtividade.

1.7.2 Fluxo do Scrum

De acordo com Brito (2010), um projeto ágil baseado em Scrum é iniciado com uma visão macro do sistema que será desenvolvido. À medida que o projeto vai sendo desenvolvido, essa visão vai se tornar mais clara e compreensível. A Figura 8 e os próximos parágrafos irão mostrar o fluxo geral do processo Scrum.

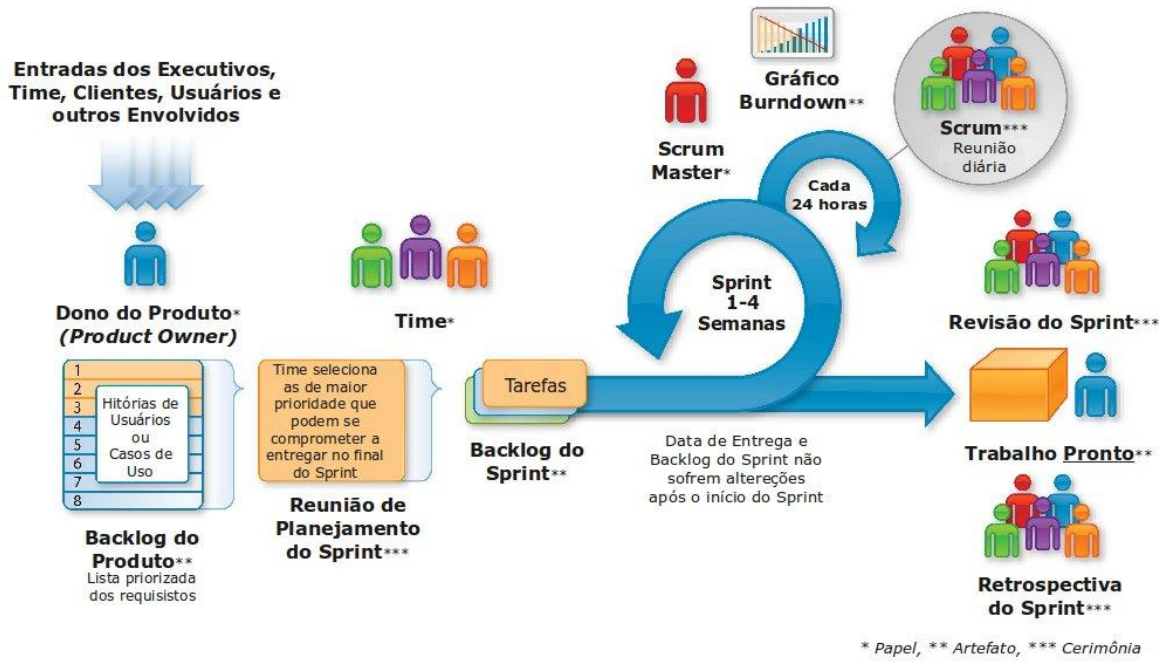

Figura 8 - Fluxo de processos do Scrum

Fonte: Pressman, 2011

Segundo Brito (2011), antes de iniciar a *Sprint*, uma reunião de planejamento (*Sprint Planning Meeting*) do projeto. A reunião é dividida em duas partes: a primeira, é o *Sprint Planning*, onde os requisitos de maior importância que devem ser desenvolvidos primeiro são apresentados pelo *Product Owner* a equipe de

desenvolvimento, que por sua vez participa, argumenta e auxilia na priorização dos requisitos. As funcionalidades ficarão para o próximo *Sprint* e o objetivo da Sprint que se inicia são definidos. Após essa primeira parte, a presença do *Product Owner* não é mais necessária, pois nesse momento o time vai definir as tarefas necessárias para a realização de cada item do *backlog* do *Sprint*.

As tarefas durante o desenvolvimento do software são realizadas em Sprints. Cada um desses Sprints dura entre 2 e 3 semanas. Na prática, as empresas as vezes são mais flexíves com a duração desses Sprints, dependendo das caracteristicas do projeto, do time e a maneira como vão se dedicar para desenvolver o software.

> A imprevisibilidade do ambiente de desenvolvimento de software pode dificultar o trabalho da equipe de desenvolvimento. No Scrum os requisitos selecionados para um Sprint não podem ser alterados no decorrer do mesmo. Os requisitos especificados ao início do período continuam imutáveis até que seja entregue esta parte do produto. Para uma próxima iteração alterações podem ser feitas (Leite, 2013).

Nas empresas de desenvolvimento que trabalham com Scrum, é realizada todos os dias uma reunião de aproximadamente 15 minutos, onde a equipe relata o que já foi feito, o que está fazendo, e quais as dificuldades encontradas durante o projeto, assim todos os envolvidos ficam cientes do andamento do desenvolvimento e também incentivando o compartilhamento do conhecimento, assim todos os membros da equipe trocam informações e aprendem entre eles durante o projeto (Leite, 2013).

Quando termina um *Sprint*, a equipe de desenvolvimento apresenta o que já foi feito numa reunião de revisão do *Sprint* ao *Product Owner* através de uma demonstração. O *product owner* faz diversos testes para verificar se os requisitos atenderam as expectativas e determina se a equipe atingiu a meta. A equipe deve sempre se preocupar com a qualidade do software que será entregue no final do *Sprint* para não atrapalhar o software inteiro e acabando por atrasar o projeto (Carvalho, 2013).

Os métodos ágeis estão sendo apontados como uma alternativa no lugar das abordagens tradicionais de software. As metodologias tradicionais deveriam ser usadas em situações onde os requisitos do software são estáveis e seus requisitos futuros sejam previsíveis.

Através dessa pesquisa foi observado que as empresas estão procurando meios para se destacarem num mercado competitivo e buscam no desenvolvimento ágil um modo de se tornarem um diferencial no mercado, porém essas empresas não utilizam os métodos ágeis seguindo os valores e princípios do Manifesto ágil e ainda utilizam algumas técnicas dos métodos tradicionais.

Na próxima seção será mostrada a metodologia escolhida para entender mais sobre as empresas e desenvolvedores que estão adotando o desenvolvimento ágil e as que continuam trabalhando com o desenvolvimento tradicional.

2. METODOLOGIA

O presente trabalho teve o objetivo de analisar e comparar as metodologias tradicionais e ágeis que são utilizadas na engenharia de requisitos. Com esse intuito foram realizadas pesquisas a fim de entender melhor as fases da engenharia de requisitos e como as empresas lidam com os processos das metodologias tradicionais e ágeis.

Após a realização do estudo sobre a engenharia de requisitos, através de pesquisa bibliográfica foi observada a maneira que as empresas tratam das fases de seus projetos de software e como seria viável utilizar metodologias ágeis durante esses processos visando melhorias na comunicação entre desenvolvedores e clientes, resultando em softwares de qualidade e harmonia dentro da empresa.

Foi realizada por meio de um questionário uma pesquisa com diversos profissionais relacionados à engenharia de requisitos e desenvolvimento de software, a fim de compreender sobre suas experiências sobre o assunto e definir a viabilidade da engenharia de requisitos ágeis como metodologia eficaz capaz de melhorar de maneira geral todas as fases de um projeto de softwares.

O questionário teve como finalidade confirmar o que foi observado durante a pesquisa sobre o desenvolvimento ágil e tradicional, porque as empresas estão migrando para o desenvolvimento ágil ou porque continuam utilizando os métodos tradicionais, mostrar se o *feedback* dos clientes é positivo nas empresas que utilizam os métodos ágeis e se o Manifesto ágil foi adotado por essas empresas totalmente ou parcialmente. Detalhes sobre a elaboração do questionário serão descritos na subseção seguinte.

2.1 PÚBLICO ALVO DO QUESTIONÁRIO

Foi selecionado como público respondente do questionário, profissionais relacionados ao desenvolvimento de softwares, dispostos segundo o cargo deles

dentro da instituição onde trabalham: Gerente de projetos, analista de testes, engenheiro de software, analista de sistemas, desenvolvedor de software.

O questionário foi apresentado por meio de um e-mail, contendo um endereço de acesso ao questionário ou por intermédio de postagens em blogs e grupos nas redes sociais especializados em engenharia de requisitos e métodos ágeis. Desse modo, profissionais associados à análise de requisitos puderam conhecer o presente estudo e responder ao questionário.

2.2 ELABORAÇÃO DO QUESTIONÁRIO

As questões que compuseram o questionário foram construídas fundamentadas na opinião de autores sobre o assunto, tais como Carvalho (2013), Coelho (2011), Ghai (2012), Leite (2013), Pressman (2011), Soares (2004) e Sommerville (2011). O questionário foi composto por um total de trinta e oito questões organizadas estrategicamente em cinco seções, tendo como objetivo simplificar o processo de participação por parte dos entrevistados.

Com o intuito de favorecer a compreensão do questionário por parte do público respondente, foi feita uma pequena introdução sobre o objetivo do questionário. Logo após no questionário estão organizadas as questões divididas em cinco seções: Identificação do participante, Engenharia de Requisitos, Metodologias Tradicionais, Metodologias Ágeis e Relacionamento com o Cliente. O questionário completo está em anexo como Apêndice A desse trabalho.

Cada seção do questionário tem o intuito de compreender um perfil específico de conhecimento. A primeira seção, intitulada “Identificação do participante” engloba questões sobre o entrevistado, como região em que mora, idade, nível de formação acadêmica e experiência profissional.

A segunda seção, “Engenharia de Requisitos” tem o objetivo de conhecer os desafios encontrados pelos profissionais, fatores importantes e técnicas que os mesmos utilizam durante a engenharia de requisitos. A terceira seção, “Metodologias Tradicionais” pergunta a opinião dos participantes em relação aos métodos tradicionais suas vantagens e desvantagens e as técnicas utilizadas

durante a utilização dessa abordagem. A quarta seção, “Metodologias ágeis” tem o intuito de explorar a opinião dos entrevistados sobre as práticas ágeis utilizadas na instituição onde o mesmo trabalha. A quinta seção, “Relacionamento com o Cliente” pergunta sobre o comportamento do cliente durante as fases do projeto de software. As subseções a seguir referem-se às seções do questionário aplicado.

2.2.1 Primeira seção: Identificação do participante

Nesta primeira seção estão colocadas questões que visam saber informações essenciais sobre o entrevistado.

As três questões descritas a seguir, tem o objetivo de identificar e classificar os participantes:

- Cidade onde reside:
- Estado onde reside:
- Qual sua idade?

Com base nessas questões, foi possível identificar e classificar os entrevistados de acordo com a idade e a região do país em que vivem e atuam.

A quinta e sexta questão como descritas a seguir visam conhecer a experiência dos participantes:

- Qual seu nível de formação?
- Há quantos anos você trabalha com desenvolvimento de software?

A partir dessas questões, foi possível conhecer a experiência de trabalho do entrevistado com o desenvolvimento de software. O restante das questões desta primeira seção tem como objetivo conhecer outros fundamentos importantes sobre a experiência dos participantes quanto ao desenvolvimento de softwares e o perfil das empresas onde os mesmos trabalham:

- Qual o número de funcionários de sua empresa?
- Qual o seu papel na equipe de desenvolvimento?
- Diariamente, você está envolvido com a produção de softwares de que tipo?

Estas perguntas visam definir o perfil do participante com base em informações práticas do seu cotidiano e o porte da empresa o qual ele trabalha.

2.2.2 Segunda seção: Engenharia de Requisitos

As questões dessa seção visam conhecer a opinião prática dos participantes sobre a engenharia de requisitos, os desafios encontrados desde a coleta dos requisitos até o final do projeto, os fatores mais importantes durante o desenvolvimento dos softwares, a importância da participação dos clientes, as técnicas de elicitação de requisitos que são utilizadas pela empresa na qual o participante trabalha e como é apresentação dos requisitos coletados para a validação dos clientes.

As três primeiras questões dessa seção foram feitas com o intuito do participante responder de acordo com a experiência dele sobre o assunto:

- O que você considera ser um desafio durante a Engenharia de Requisitos?
- Qual a importância dos seguintes fatores durante o desenvolvimento de um software.
- Classifique o quão importante é para você a participação do cliente durante o desenvolvimento do software.

A segunda questão da seção pergunta sobre a importância de alguns fatores e são eles: Qualidade do produto, satisfação do cliente e resultados de vendas. O participante respondia essa questão marcando para cada fator se os mesmos eram importantes, pouco importantes, importantes, muito importantes ou se o participante não tivesse conhecimento sobre os fatores abordados poderia marcar a opção “Não sei”.

Já a terceira questão a qual o participante deveria classificar a importância da participação do cliente durante o desenvolvimento do software, ele marcava uma escala de 0 a 5, sendo zero “Pouco importante” e cinco “Muito importante”.

As outras duas questões dessa seção o participante respondeu com base nos trabalhos realizados na empresa que ele trabalha:

- Quais técnicas de requisitos são utilizadas nos projetos de desenvolvimento de software na sua empresa?
- Como você costuma apresentar os requisitos coletados para que os mesmos sejam validados pelo cliente?

Nessas duas ultimas questões dessa seção as opções apresentadas foram selecionadas a partir das informações coletadas durante as pesquisas e foram fundamentadas por Pressman (2011) e Sommerville (2011).

Na questão que pergunta sobre as técnicas utilizadas, o participante teve a opção de marcar em cada técnica apresentada se ele a conhecia e a utiliza na empresa, conhecia e não utilizava na empresa e se não conhecia a técnica mencionada. Já na questão que pergunta sobre como é apresentado ao cliente os requisitos para a validação, o participante pode selecionar mais de uma opção.

2.2.3 Terceira seção: Metodologias Tradicionais

As questões presentes nessa seção tem o objetivo de conhecer mais sobre quais metodologias o participante tem conhecimento e quais são usadas em sua organização, quais as vantagens e as desvantagens de se utilizar as metodologias tradicionais. A primeira questão dessa seção teve como objetivo apenas conhecer um pouco mais sobre a experiência do entrevistado:

- Há quanto tempo trabalha ou trabalhou utilizando metodologias tradicionais?

A partir dessa questão, foi possível conhecer a experiência profissional do entrevistado com as metodologias tradicionais.

As demais questões dessa seção foram idealizadas com base nas informações coletadas durante a pesquisa relacionada às metodologias tradicionais:

- Quais metodologias tradicionais são utilizadas nos projetos de desenvolvimento de software na sua empresa?
- Mencione outras metodologias tradicionais que são utilizadas na sua empresa e não foram citadas na questão anterior.

- Na sua opinião, quais são as vantagens das metodologias tradicionais?
- Na sua opinião quais são as desvantagens das metodologias tradicionais?

Na questão que pergunta sobre as metodologias tradicionais que são utilizadas, o participante teve a opção de marcar em cada técnica apresenta se ele a conhecia e utilizava na empresa, conhecia e não utilizava na empresa e se não conhecia a técnica mencionada. E após responder poderia mencionar na próxima questão metodologias que ele utiliza ou já utilizou na empresa em q trabalha e não foram citadas na questão anterior.

As opções apresentadas nas questões sobre as vantagens e desvantagens das metodologias tradicionais foram selecionadas a partir da opinião de vários autores pesquisados, entre eles Filho (2008), Coelho (2011), Ghai (2012), Junior et. al. (2011), Leite (2013), Soares (2004), além de outros autores que contribuíram indiretamente na concepção do assunto de maneira geral. Nessas questões o participante poderia marcar várias opções e mencionar alguma outra que ele considerasse vantagem e/ou desvantagem.

2.2.4 Quarta seção: Metodologias Ágeis

As questões presentes nessa seção foram embasadas na opinião de autores sobre o assunto, tais como Bassi (2008), Carvalho (2013), Cintra (2006), Coelho (2011), Ghai (2012), Girardi et al. (2012), Júnior et al. (2011), Leite (2013), Manifesto Ágil (2001), Pressman (2011), Silva (2006), Soares, M. (2004) Soares, L. (2007) e Sommerville (2011). A primeira questão dessa seção teve como objetivo apenas conhecer um pouco mais sobre a experiência do entrevistado:

- Há quanto tempo trabalha ou trabalhou utilizando metodologias ágeis?

A partir dessa questão, foi possível conhecer a experiência profissional do entrevistado com as metodologias ágeis.

A segunda questão dessa seção pergunta sobre as metodologias ágeis que são utilizadas na empresa em que o participante trabalha:

- Quais metodologias ágeis são utilizadas nos projetos de desenvolvimento de software na sua empresa?
- Mencione outras metodologias ágeis que são utilizadas na sua empresa e não foram citadas na questão anterior.

Nessa segunda questão o participante teve a opção de marcar em cada metodologia apresentada se ele a conhecia e utilizava na empresa, conhecia e não utilizava na empresa e se não conhecia a técnica mencionada. As opções de resposta dessa segunda questão foram selecionadas com base nas informações recolhidas durante a pesquisa sobre as metodologias ágeis. Após responder a questão anterior o participante poderia mencionar na próxima questão, metodologias ágeis que ele utiliza ou já utilizou na empresa em q trabalha e não foram citadas na questão anterior.

A quarta questão dessa seção foi feita com o intuito de saber a opinião do participante sobre documentação de requisitos:

- Classifique a importância de uma documentação mais formal durante a utilização de alguma metodologia ágil.

Nessa questão o participante deveria classificar a importância de uma documentação mais formal durante um projeto ágil, ele podia marcar em uma escala de 0 (zero) a 5 (cinco), sendo zero “Pouco importante” e cinco “Muito importante”.

Pressman (2011) e Sommerville (2011) descreveram em seus livros algumas práticas ágeis utilizadas durante as fases da engenharia de requisitos. Algumas dessas práticas foram citadas na questão a seguir:

- Quais práticas ágeis sobre requisitos são utilizadas na sua empresa?

Essa questão teve como objetivo conhecer mais sobre as práticas ágeis que o participante conhece e utiliza em sua empresa, e nessa questão ele poderia marcar se ele conhecia e utilizava a prática ágil na empresa, conhecia e não utilizava na empresa e se não conhecia a prática ágil mencionada na questão.

As demais questões dessa seção foram escritas baseadas nas informações coletadas durante o estudo sobre as metodologias ágeis, principalmente sobre o Manifesto Ágil:

- Na sua opinião, quais são as vantagens das metodologias ágeis?

- Na sua opinião, quais são as desvantagens das metodologias ágeis?
- Classifique a importância dos valores do Manifesto Ágil citados abaixo

As opções apresentadas nas questões sobre as vantagens e desvantagens das metodologias ágeis foram selecionadas a partir da opinião de vários autores pesquisados, entre eles Bassi (2008), Carvalho (2013), Coelho (2011), Leite (2013), Manifesto Ágil (2001), Pressman (2011), Silva (2006), Soares, M. (2004) Soares, L. (2007) e Sommerville (2011) além de outros autores que contribuíram indiretamente no entendimento do assunto de maneira geral. Nessas questões o participante poderia marcar várias opções e mencionar alguma outra que ele considerasse vantagem e/ou desvantagem das metodologias ágeis.

Já na questão que pede para classificar a importância os valores do Manifesto ágil, o participante podia classificar os quatro princípios marcando se ele considerava os mesmos “Muito importante”, “Importante”, “Pouco importante”, “Não é importante”, ou caso não soubesse classificar poderia ser marcado a opção “Não sei”.

As demais questões dessa seção tem o objetivo de relacionar princípios do Manifesto ágil com o ambiente em que o participante trabalha com as metodologias ágeis:

- Projetos desenvolvidos utilizando métodos ágeis requerem um ritmo constante de trabalho durante todas as fases. Seu ambiente de trabalho oferece motivação para que seu rendimento seja bom?
- De acordo com um dos princípios do Manifesto ágil a melhor maneira de compartilhar informações para e entre uma equipe de desenvolvimento é através de conversa aberta, de forma presencial. Em sua empresa, as informações são transmitidas dessa forma?
- Sua equipe se reúne regularmente para pensar em maneiras de se tornarem mais eficazes e ajustarem o comportamento de acordo com aquilo que foi pensado?
- Os prazos estabelecidos no início dos projetos são cumpridos?

As questões citadas acima tiveram o intuito de prover uma comparação entre o Manifesto ágil e seus princípios e o que realmente acontece na organização onde o participante trabalha e identificar se nas empresas eles tem contato direto com as práticas e princípios do manifesto.

2.2.5 Quinta seção: Relacionamento com o Cliente

As questões dessa seção tem o objetivo de conhecer mais sobre a comunicação entre o cliente e a organização. Todas as questões são focadas no comportamento do cliente em algumas etapas da engenharia de requisitos ágeis:

- Como você classifica o grau de satisfação do cliente em relação ao número de reuniões feitas ao longo do projeto desenvolvido com metodologias ágeis?
- Com que frequência ocorre mudanças de requisitos por parte do cliente?
- As exigências feitas pelo cliente são sempre acatadas pela equipe?
- O cliente demonstra mais confiança no projeto acompanhando todos os passos de desenvolvimento do software?
- O cliente aprova as prévias do sistema funcionando que são apresentadas em curtos intervalos de tempo?
- Marque qual/quais motivo(s) levam os clientes a reprovar as prévias do sistema.
- Como você classifica o grau de satisfação do cliente em relação à entrega do produto final?

As questões que pedem ao participante para classificar o grau de satisfação do cliente, podem ser marcadas de 0 (zero) a 5 (cinco), sendo zero “Insatisfeito” e cinco “Muito satisfeito”. Já a questão que pede para classificar a frequência com que os clientes mudam os requisitos, o participante pode marcar de 0 (zero) a 5 (cinco) também, porém zero indica uma baixa frequência e cinco indica uma alta frequência de mudanças.

As demais questões dessa seção são de múltipla escolha, onde o participante pode marcar “Sim”, “Não” e “Às vezes” e são focadas no cliente e a equipe responsável pelo projeto, onde as perguntas tem o objetivo de conhecer as decisões que a equipe toma quando o cliente altera algum requisito do projeto, se ele demonstra confiança e se aprova as prévias do produto que a equipe passa para ele.

2.3 COLETA DE DADOS

Foi usado o *Google Docs* para a elaboração de um questionário *online.* Por meio dessa ferramenta, foi viável a elaboração e distribuição do questionário, além de permitir aos participantes responder e acessar as questões com calma e atenção através de seus próprios computadores.

O questionário foi divulgado através de um *link* que permitia o acesso às questões. Esse *link* foi enviado por *e-mail* a profissionais da área de TI, alunos e ex-alunos do curso de Ciência da Computação das Faculdades Integradas de Caratinga. Também houve divulgação do questionário em fóruns de contribuição sobre engenharia de software, engenharia de requisitos e métodos ágeis, através das redes sociais *Facebook* e *Linkedin* a profissionais relacionados à metodologias ágeis, desenvolvimento de software, engenharia de requisitos e TI.

O link foi disponibilizado no dia 06 de outubro de 2015 às 11h30min, ficando disponível para resposta até às 11h30min do dia 26 de outubro de 2015. O questionário foi respondido por 82 (oitenta e duas) pessoas no total.

2.4 TRATAMENTO DE DADOS

As respostas coletadas foram encaminhadas automaticamente para uma planilha *online* criada pelo *Google Docs.* Através desta planilha foi permitido visualizar cada resposta individualmente. Após ser exportada como uma planilha a

ser visualizada através do Microsoft Excel 2010, os dados foram dispostos de acordo com o perfil de cada participante da pesquisa. Desse modo, foi possível a analisar as respostas de acordo com o papel que os participantes exercem na equipe de desenvolvimento.

Os resultados foram exibidos através de gráficos com a intenção de favorecer a compreensão. Eles podem ser visualizados na seção “Análises e Resultados” que está exposta a seguir.

3. ANÁLISES E RESULTADOS

Nesta seção serão apresentados os resultados obtidos por meio da aplicação do questionário elaborado conforme metodologia descrita no capítulo 2. Um total de 82 pessoas responderam ao questionário sendo uma boa parte (80 pessoas de todas as respostas) alegaram que trabalham com o desenvolvimento de softwares e tinham uma experiência sobre o assunto abordado.

Para maior compreensão dos resultados, as respostas serão apresentadas por meio de gráficos. A organização das respostas bem como os gráficos serão descritos nas subseções seguintes.

3.1 DISCUSSÃO DOS RESULTADOS

Durante o processo de apuração dos dados obtidos, dois participantes foram impróprios para a pesquisa. Os dois entrevistados participaram do questionário alegando ter experiência na área de TI, porém afirmaram que não possuíam experiência em desenvolvimento de software. Desse modo, as respostas exibidas nessa seção são referentes aos 80 participantes aptos, ou seja, desenvolvedores, analistas, engenheiros de softwares e gerentes.

As respostas apresentadas a seguir serão categorizadas em cinco seções do questionário, sendo: Primeira seção: Identificação do participante, Segunda seção: Engenharia de Requisitos, Terceira seção: Metodologias Tradicionais, Quarta seção: Metodologias Ágeis, Quinta Seção: Relacionamento com o Cliente.

3.1.1 Primeira seção: Identificação do participante

Essa seção contou com as respostas de 80 (oitenta) pessoas, de 14 estados brasileiros onde 52% residem em Minas Gerais, 22% moram em São Paulo, 28%

residem em outros Estados como Rio de Janeiro, Goiás, Santa Catarina, Pernambuco, Rio Grande do Sul, entre outros. Entre os entrevistados 54% têm idade entre 15 e 25 anos, 34% com idade entre 26 e 35 anos, 9% com idade entre 36 e 45 anos e a minoria representada por 4% dos entrevistados com idade entre 46 e 55 anos.

Sobre a formação acadêmica dos entrevistados a maioria, 70% são graduados ou estão se graduando, 18% são pós-graduados ou pós-graduando, 9% dos participantes são mestres e mestrando, 3% são doutores ou doutorandos e a minoria são os técnicos que correspondem a 2% dos entrevistados.

Cerca de 34% dos entrevistados trabalham como Desenvolvedores de software, 16% ocupam o cargo de Analista de Testes, 13% são Gerentes de projetos, 8% são Engenheiros de Software, 5% trabalham como Analista de Testes e os outros 25% são os profissionais que não se encaixam nos cargos citados anteriormente, são eles: Analista de Requisitos, Arquitetos de Negócios, Analista de Métricas, Analista de planejamento, CEO (*Chief Executive Officer*) e facilitador, CTO (Diretor técnico), Analista de *Business Intelligence*, Diretor de TI, Analista de Suporte e desenvolvimento e estagiário de desenvolvimento.

A maioria dos respondentes, 73% são colaboradores de empresas com 0 a 50 funcionários, 10% trabalham em empresas que possuem de 50 a 500 funcionários, outros 10% trabalham em empresas com mais de 1000 (mil) funcionários e a minoria, cerca de 8% trabalham em empresas que contém entre 500 e 1000 funcionários.

A maior parte dos entrevistados, 70% trabalham com o Desenvolvimento de software no período de 0 e 5 anos, 20% trabalham no período de 5 a 10 anos, 11% trabalha com desenvolvimento no período de 10 e 15 anos e a minoria trabalha com software a mais de 15 anos.

Foi constatado que 60% dos respondentes desenvolvem softwares para web e 45% estão envolvidos com o desenvolvimento de softwares de Gerenciamento Empresarial, 15% desenvolvem softwares de propósito geral, 10% trabalham com software para *smartphones* e *tablets* e 14% desenvolvem outros tipos de softwares, como por exemplo, sistemas financeiros e bancários e 4% dos entrevistados estão envolvidos com desenvolvimento de Sistemas Operacionais.

O principal objetivo desta seção do questionário era conhecer o perfil dos participantes. De acordo com as respostas, conclui-se que a maioria dos

respondentes são profissionais graduados ou com a graduação em andamento, desenvolvem softwares com finalidades diversas e trabalham em empresas de pequeno e médio porte. Visivelmente, o questionário alcançou o público desejado, composto por profissionais que lidam com o desenvolvimento de sistemas diariamente.

Serão discutidas nas próximas seções as técnicas mais utilizadas pelos respondentes desde a elicitação de requisitos no início do projeto até o software já na fase final, as vantagens e desvantagens das Metodologias Ágeis e tradicionais, os desafios encontrados durante os projetos utilizando essas metodologias e o relacionamento com o cliente durante todo o projeto.

3.1.2 Segunda seção: Engenharia de Requisitos

Nesta seção serão descritas as respostas das questões presentes na segunda seção do questionário que visam conhecer a experiência dos participantes sobre a engenharia de requisitos, os desafios encontrados desde a coleta dos requisitos até o final do projeto, os fatores mais importantes durante o desenvolvimento dos softwares, a importância da participação dos clientes. Cada uma das subseções seguintes trata de uma questão específica do questionário.

3.1.2.1 O que você considera um desafio durante a Engenharia de Requisitos?

De acordo com a pesquisa elaborada, conclui-se que a comunicação com o cliente e as mudanças que são feitas após o início do desenvolvimento do software e isso pode ser um grande desafio para a Engenharia de requisitos. No Gráfico 1 a seguir, serão visualizadas as respostas dos participantes.

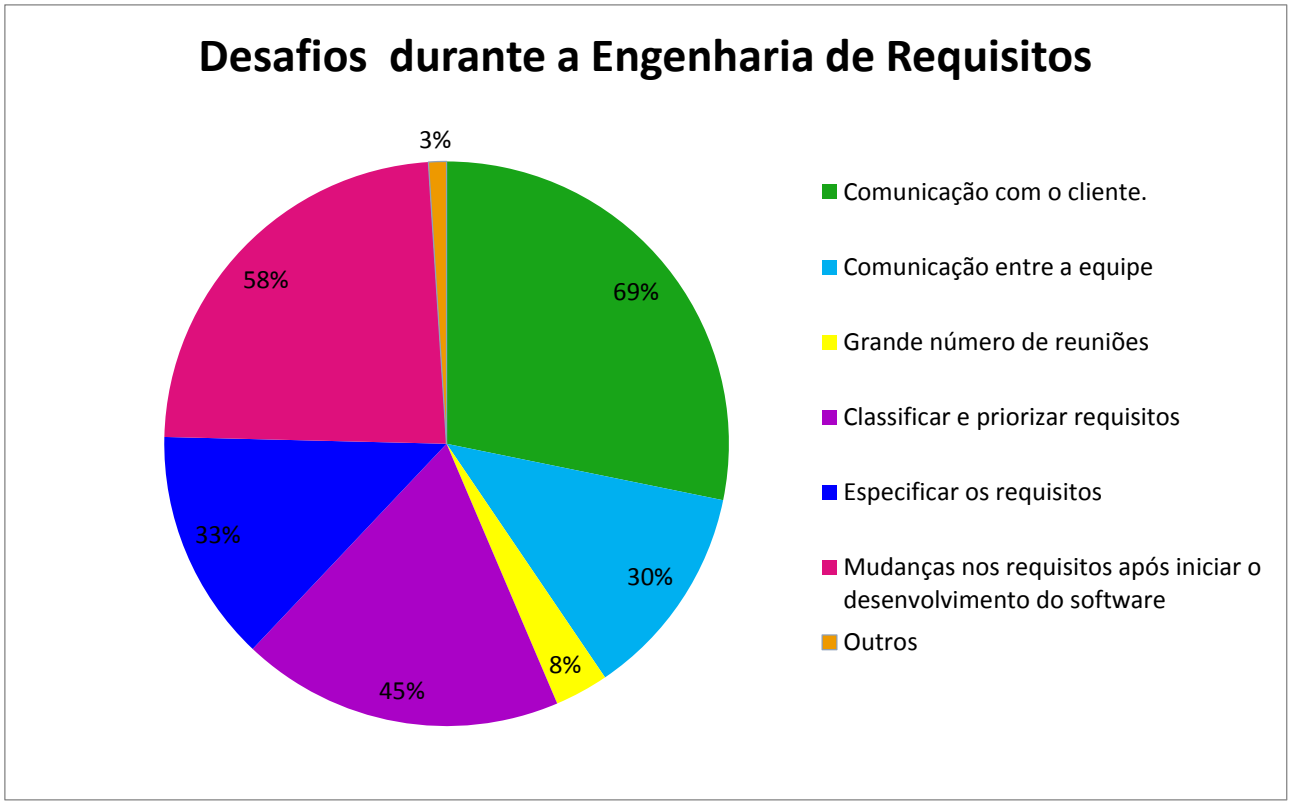

Gráfico 1 - Desafios durante a Engenharia de Requisitos

Fonte: Própria autora

Diariamente, uma empresa de desenvolvimento de software se vê diante de mais de um obstáculo. Portanto, permitiu-se que os entrevistados, selecionassem mais de uma opção de resposta para esta questão.

A maioria dos entrevistados, aproximadamente 69% alega que a comunicação com o cliente durante a Engenharia de Requisitos é o maior desafio enfrentado. Cerca de 58% alegam também que a mudança nos requisitos após o início do desenvolvimento do software também é um desafio muito grande para a equipe de desenvolvimento

3.1.2.2 Qual a importância dos seguintes fatores durante o desenvolvimento de um software.

Para essa pergunta, os respondentes deveriam classificar os fatores “Qualidade do Produto”, “Satisfação do cliente” e “Resultado de vendas”, como

“Muito importante”, “Importante”, “Pouco Importante”, “Não é importante” e “Não sei”. Como a maioria selecionou a opção “Muito Importante” foram selecionadas apenas essas respostas, observe os percentuais no Gráfico 2 a seguir.

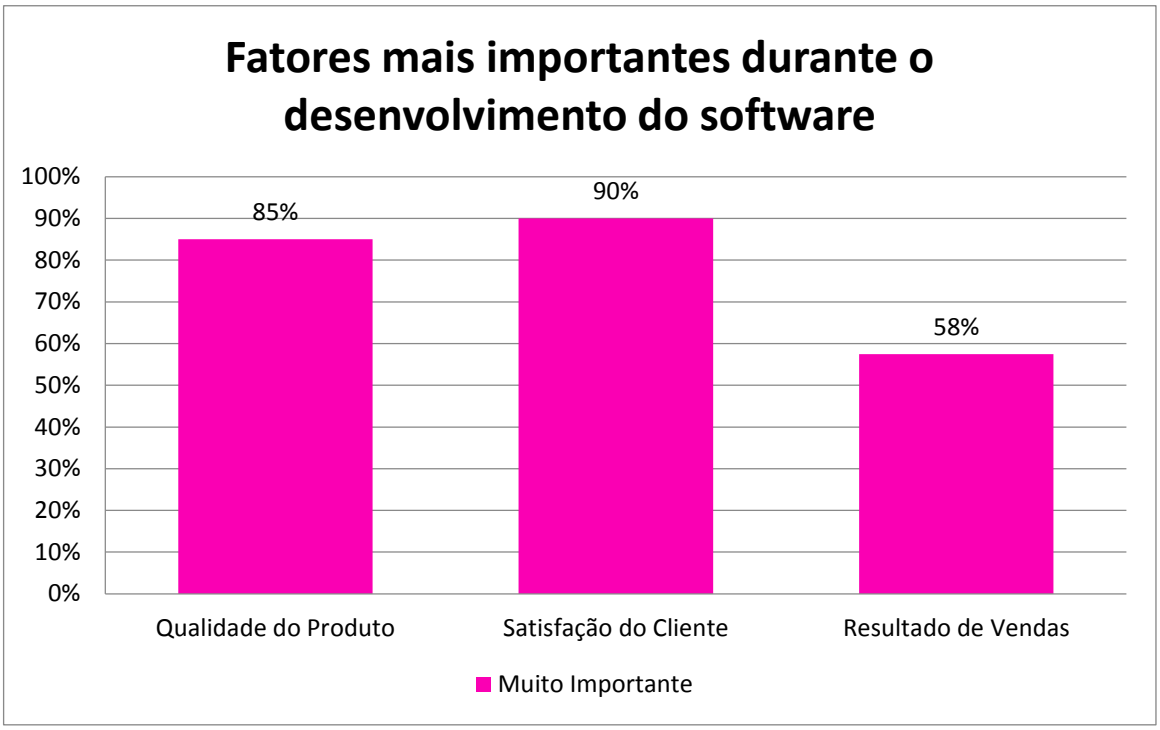

Gráfico 2 - Fatores mais importantes durante o desenvolvimento do software

Fonte: Própria autora

A maioria dos respondentes, 90% classificaram a “Satisfação do Cliente” o fator mais importante durante o desenvolvimento, seguido dos 85% que são referentes a “Qualidade do Produto” e a minoria, 58% classificou o ”Resultado de Vendas” um fator importante durante o desenvolvimento do software.

Nessa mesma seção, consta uma questão referente a participação do cliente durante o desenvolvimento do software e a maioria, 65% dos respondentes alegaram que a participação do cliente é muito importante para o projeto. Com base nesses resultados pode-se afirmar que as empresas focam mais em satisfazer o cliente, manter uma qualidade do software durante o projeto do software e dão importância a uma aproximação maior do cliente durante o projeto, concordando com os valores do Manifesto Ágil (2001).

3.1.2.3 Quais técnicas de requisitos são utilizadas nos projetos de desenvolvimento de software na sua empresa?

Para essa pergunta, os respondentes deveriam marcar se as técnicas eram conhecidas e usadas na empresa em que ele trabalha, se ele as conhecia e não eram utilizadas na empresa ou se não conhecia aquela técnica apresentada. Para a análise foi considerada apenas as respostas onde os participantes alegaram que conheciam e usavam aquelas técnicas nas suas empresas, como pode ser observado no gráfico a seguir.

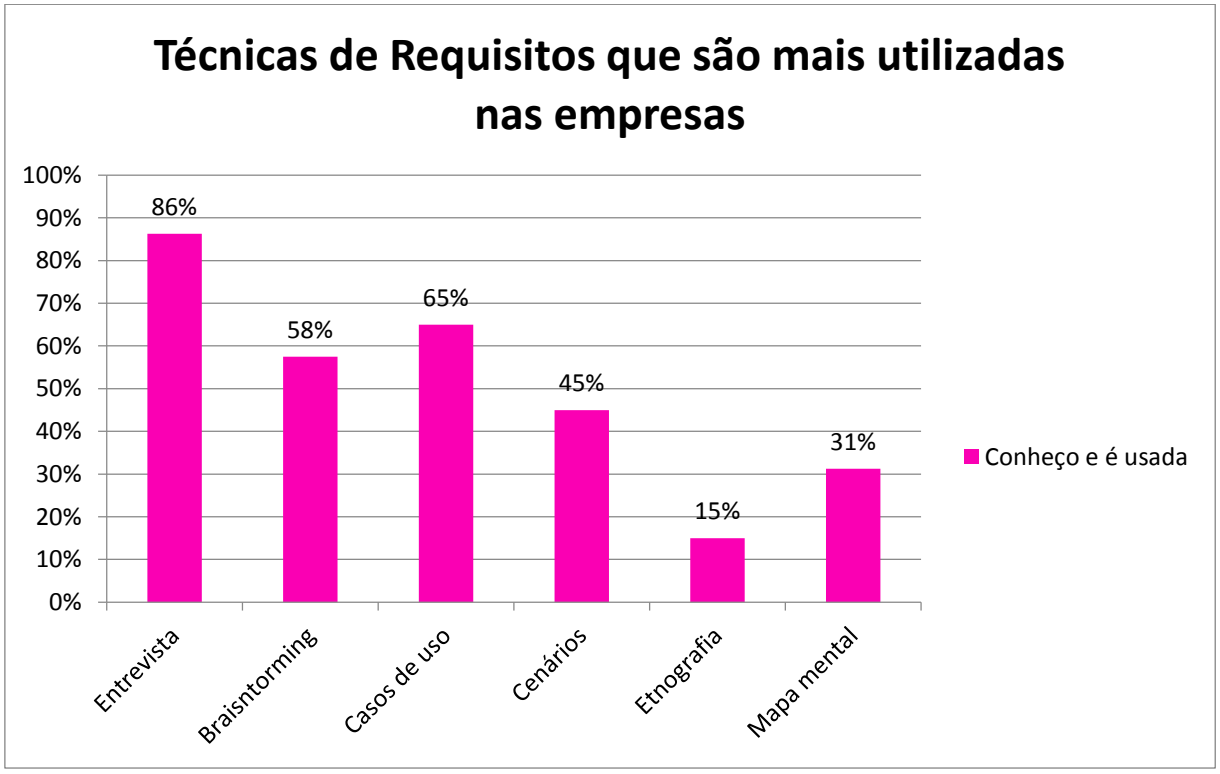

Gráfico 3 **-** Técnicas de Requisitos que são mais utilizadas nas empresas

Fonte: Própria autora

Grande parte dos respondentes, 86% utilizam a técnica de Entrevista para a elicitação dos requisitos, seguido de 65% que utilizam os Casos de uso como técnica mais usada, o terceiro mais usado de acordo com a pesquisa é a técnica Brainstorming com 58% dos participantes apontando ela como umas das mais usadas nas empresas, a quarta técnica mais usada são os Cenários, 45% das pessoas apontaram ela como a técnica mais usada em suas empresas. E as

técnicas menos utilizadas nas empresas foram Mapa Mental e Etnografia, com 31% e 15% de respostas respectivamente.

A última questão dessa seção que indaga como os requisitos especificados são apresentados aos clientes para que sejam validadas, quais técnicas que são utilizadas e 64% dos respondentes afirmaram que a apresentação dos requisitos é feita com Diagramas de casos de uso que foi apontada como uma das técnicas mais usadas nas empresas para fazer o levantamento de requisitos. Alguns outros respondentes apontaram os Fluxogramas (31%) e as *User Stories* – Histórias de Usuários (29%) como as formas mais usadas para apresentar os requisitos para os clientes.

Na seção a seguir serão exibidas as respostas referentes às Metodologias Tradicionais utilizadas durante os projetos de software, suas vantagens e desvantagens.

3.1.3 Seção três: Metodologias Tradicionais

Nessa seção serão descritas as respostas das questões presentes na terceira parte do questionário que visam conhecer a experiência dos participantes sobre a as metodologias tradicionais, há quanto tempo os respondentes trabalham com essas metodologias, quais dessas metodologias são as mais utilizadas nas empresas, as vantagens e desvantagens das metodologias tradicionais segundo os participantes da pesquisa.

3.1.3.1 Há quanto tempo trabalha ou trabalhou utilizando metodologias tradicionais?

A primeira pergunta dessa seção tem o objetivo de identificar a experiência que os respondentes têm sobre as metodologias tradicionais, a apuração dos dados são apresentados no Gráfico 4 a seguir.

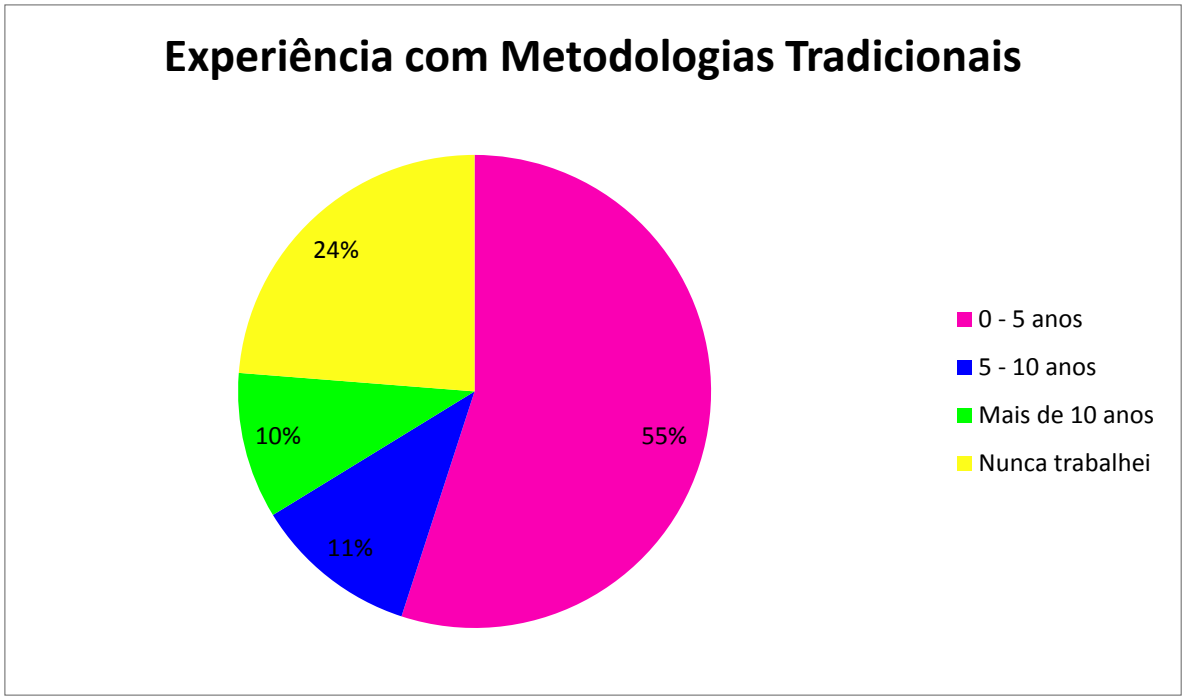

Gráfico 4 - Experiência com Metodologias Tradicionais

Fonte: Própria autora

Como foi apresentado no Gráfico 4, a maioria dos respondentes, 55% trabalham ou trabalharam com metodologias tradicionais entre 0 e 5 anos, 24% dos respondentes nunca trabalharam com as metodologias tradicionais, uma pequena parcela, 10% dos participantes possuem experiência de mais de 10 anos com metodologias tradicionais e cerca de 11% possuem experiência entre 5 e 10 anos. Com isso pode-se concluir que a maioria desses profissionais possui pouca experiência com métodos tradicionais ou nunca utilizaram métodos tradicionais.

3.1.3.2 Quais metodologias tradicionais são utilizadas nos projetos de desenvolvimento de software na sua empresa?

Para essa pergunta, os respondentes deveriam marcar se as metodologias apresentadas são conhecidas e usadas na empresa em que ele trabalha, se ele as conhecia e não eram utilizadas na empresa ou se não conhecia aquela metodologia apresentada. Para a análise foi considerada apenas as respostas nas quais os

participantes alegaram que conheciam e usavam aquelas metodologias nas suas empresas, como pode ser observado no Gráfico 5.

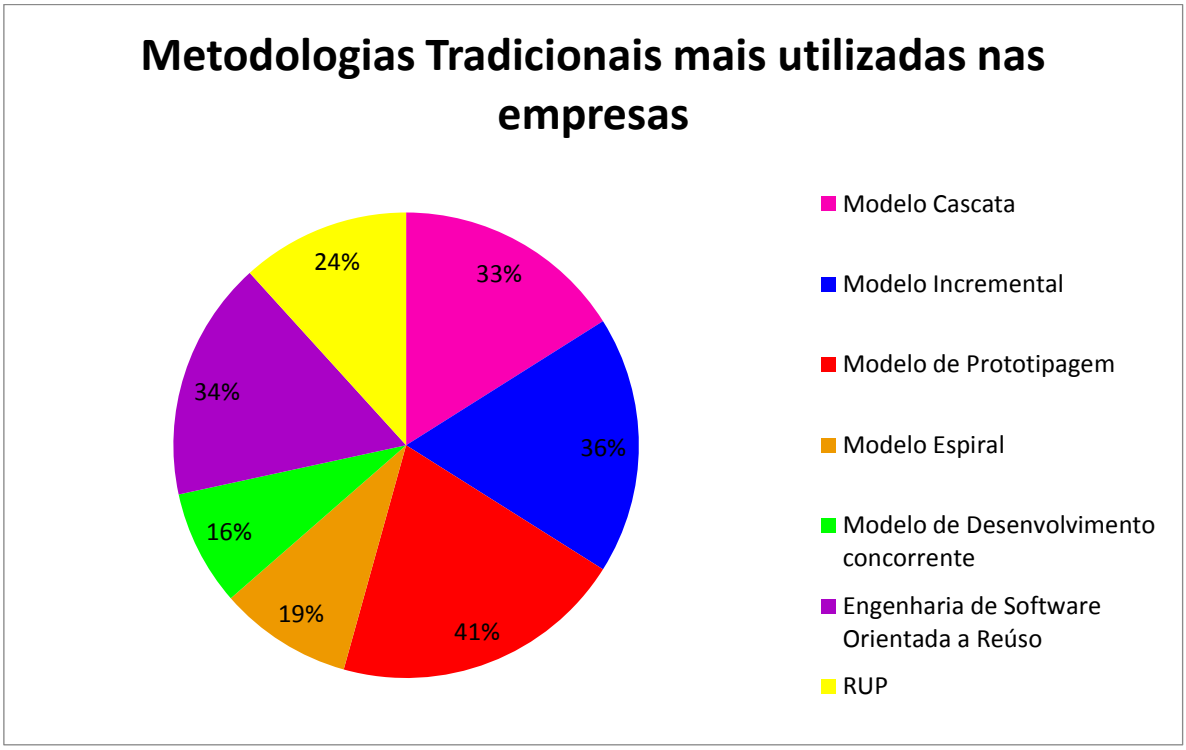

Gráfico 5 - Metodologias Tradicionais mais utilizadas nas empresas

Fonte: Própria autora

A maioria dos entrevistados, 41% respondeu conhecer e utilizar na empresa o Modelo de Prototipagem, seguido dos 36% que responderam utilizar o Modelo Incremental, 34% optam por usar a Engenharia de software orientada a reuso e 33% afirmou utilizar e conhecer mais o Modelo Cascata. As outras metodologias citadas na questão não foram muito apontadas como conhecidas e utilizadas nas empresas de desenvolvimento, entre elas podemos citar o RUP (24%), Modelo Espiral (19%) e Modelo de Desenvolvimento Concorrente (16%).

Após essa questão havia um espaço onde o entrevistado poderia mencionar outras metodologias tradicionais que são utilizadas em suas empresas e foi observado que os entrevistados mencionaram as metodologias ágeis Scrum, XP e Lean como metodologias tradicionais e com essas menções podemos concluir que os profissionais não conhecem os métodos tradicionais de desenvolvimento, possuem pouca experiência em desenvolvimento e por conta disso utilizam apenas

metodologias ágeis, confirmando que os métodos tradicionais estão entrando em desuso nas empresas de desenvolvimento de software.

3.1.3.3 Na sua opinião, quais são as vantagens das metodologias tradicionais?

Para essa questão, os participantes deveriam marcar quais as vantagens das metodologias tradicionais que são utilizadas na empresa em que trabalham. Nessa questão foi permitido aos entrevistados que selecionassem mais de uma opção de resposta para essa questão e por esse motivo, a soma de vantagens das metodologias tradicionais assinaladas pelos respondentes ultrapassam o número de respondentes.

As respostas dessa questão são visualizadas no Gráfico 6:

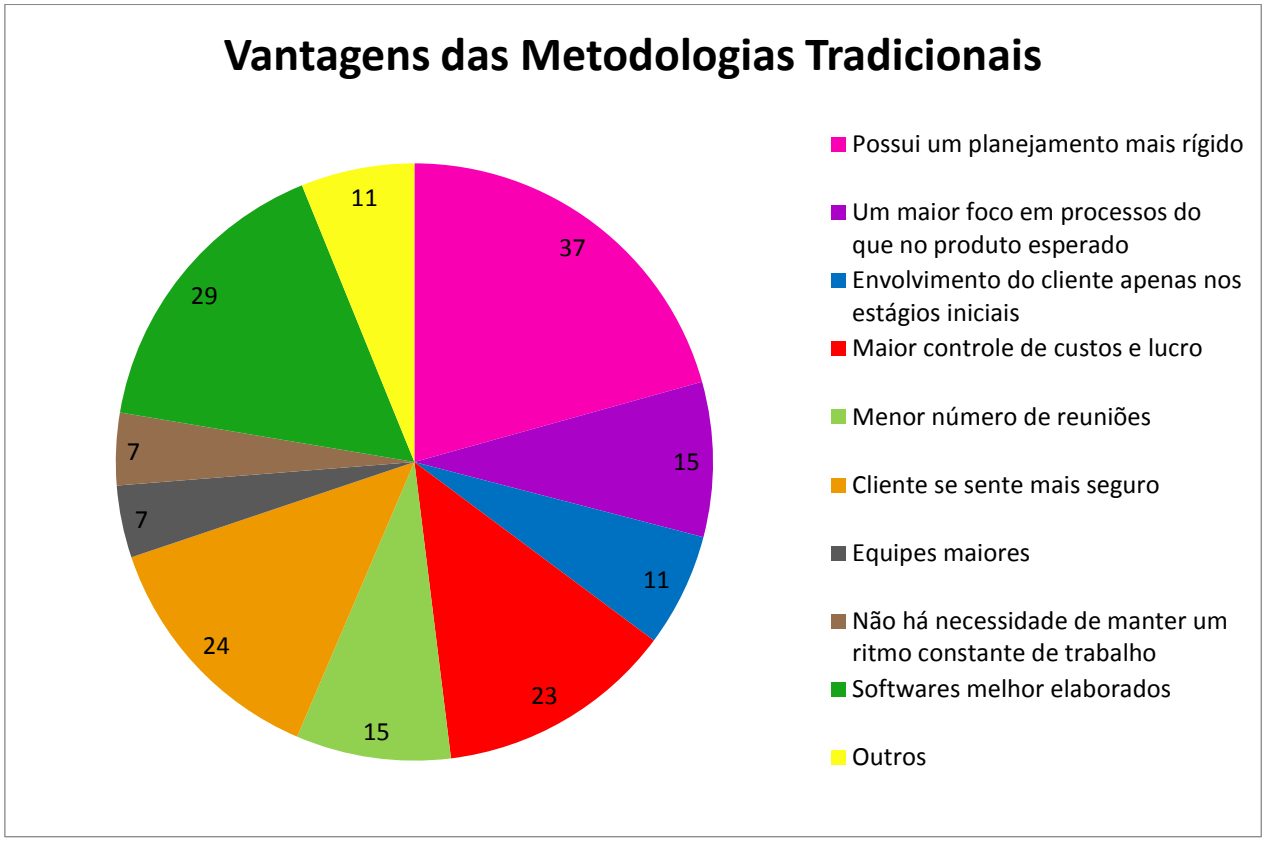

Gráfico 6 - Vantagens das Metodologias Tradicionais

Fonte: Própria autora

A maior parte dos participantes assinalaram como vantagem “Possui um planejamento mais rígido” (46% que correspondem a 37 pessoas), “Softwares melhor elaborados” (36% que correspondem a 29 pessoas), “Cliente se sente mais seguro” (30% que correspondem a 24 pessoas) e “Maior controle dos custos e lucros”(29% que correspondem a 23 pessoas).

Dos onze participantes (14%) que apontaram outras vantagens, dois deles mencionaram como vantagens a “Previsibilidade de escopo de software” que as metodologias tradicionais dão e “Um maior foco em processos do que no produto esperado”.

Os outros nove participantes que também marcaram a opção “Outros” alegaram que não enxergam as metodologias tradicionais como vantajosas no cenário atual de desenvolvimento de software.

Na próxima seção serão apresentadas as desvantagens das Metodologias Tradicionais e a partir delas, juntamente com a análise do Gráfico 6 será possível confirmar se os respondentes tem uma certa resistência aos métodos tradicionais de desenvolvimento de software.

3.1.3.4 Na sua opinião quais são as desvantagens das metodologias tradicionais?

Para essa questão, os participantes deveriam marcar quais as desvantagens das metodologias tradicionais que são utilizadas na empresa em que trabalham. Nessa questão também foi permitido aos entrevistados que selecionassem mais de uma opção de resposta e por esse motivo, a soma de desvantagens das metodologias tradicionais assinaladas pelos respondentes ultrapassam 80 respostas.

As respostas dessa questão são visualizadas no Gráfico 7:

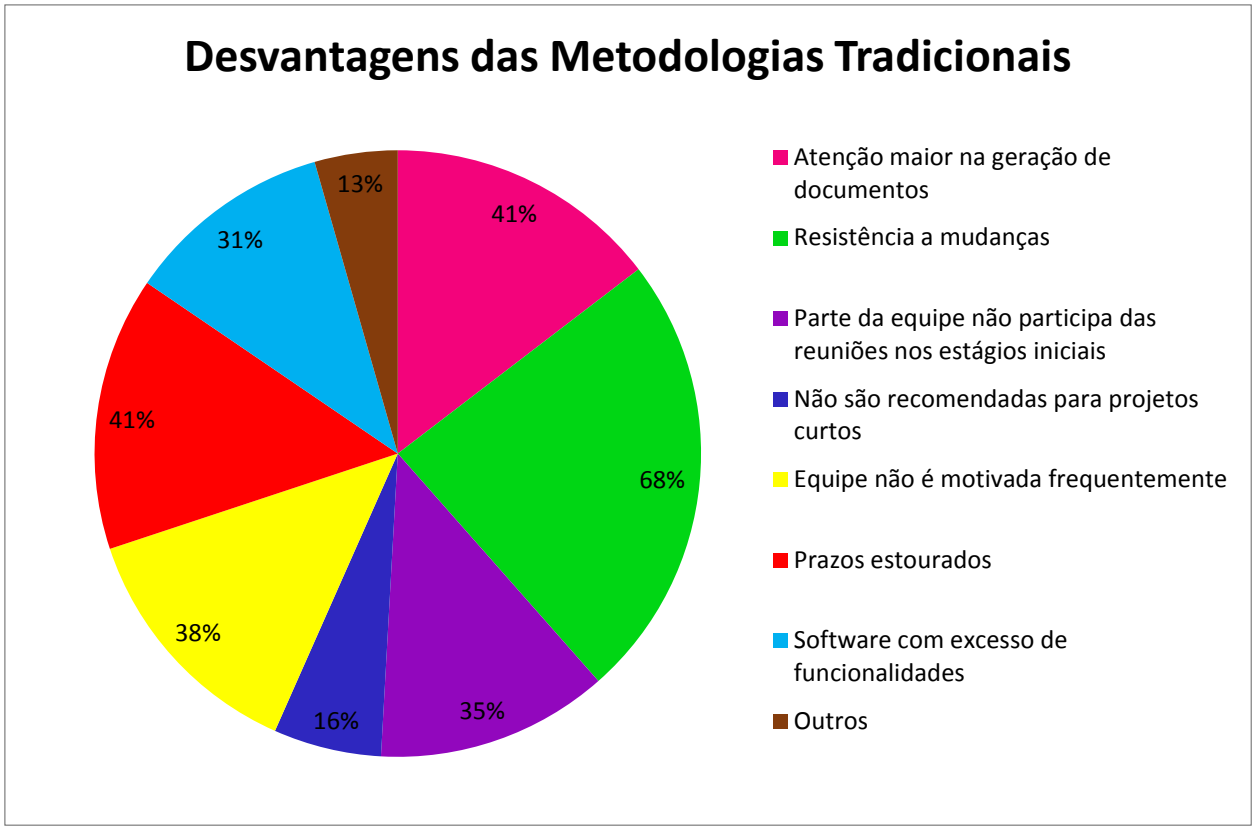

Gráfico 7 - Desvantagens das Metodologias Tradicionais

Fonte: Própria autora

A maior parte dos participantes assinalaram como desvantagem “Resistência a mudanças” (68% que correspondem a 54 pessoas), “Prazos estourados” (41% que correspondem a 33 pessoas), “Atenção maior na geração de documentos” (41% que correspondem a 33 pessoas), “Equipe não é motivada frequentemente” (38% que correspondem a 30 pessoas) e “Parte da equipe não participa das reuniões nos estágios iniciais” (35% que correspondem a 28 pessoas).

Dez participantes (13%) mencionaram outras desvantagens como: “Desperdício de tempo, prazo e custo”, “Cliente distante”, “Processo engessado”, “Pouca comunicação face a face”, “Cliente vê apenas uma versão do software do final” e “Falta de agilidade no mercado atualmente tão competitivo”.

Pode-se concluir com as informações exibidas nos Gráficos 6 e 7 que as desvantagens das metodologias tradicionais que foram apontadas e mencionadas pelos respondentes podem ser consideradas afirmações de uma influência maior do do que as vantagens assinaladas por eles, pois mesmo na questão onde se pede para mencionar outras vantagens os respondentes aproveitaram para alegar que na opinião deles não existem vantagens no uso de metodologias tradicionais. Outra

questão importante para essa afirmação é a observação feita na questão onde se menciona as desvantagens dos métodos tradicionais, onde temos até 68% dos respondentes marcando pelo menos uma opção de desvantagem e fazendo questão de mencionar outras desvantagens nas metodologias.

3.1.4 Seção quatro: Metodologias Ágeis

Nessa seção serão descritas as respostas das questões presentes na quarta parte do questionário que visam conhecer a opinião dos participantes sobre a as metodologias ágeis, há quanto tempo os respondentes trabalham com essas metodologias, quais dessas metodologias são as mais utilizadas nas empresas, a importância de uma documentação mais formal durante a utilização de alguma metodologia ágil, as práticas ágeis são mais utilizadas nas empresas, as vantagens e desvantagens das metodologias ágeis segundo os participantes da pesquisa e questões relacionadas aos valores do Manifesto Ágil já apresentados neste trabalho no Capítulo 1.

3.1.4.1 Há quanto tempo trabalha ou trabalhou utilizando metodologias ágeis?

A primeira pergunta dessa seção tem o objetivo de identificar a experiência que os respondentes têm sobre as metodologias ágeis, como pode ser visualizado no Gráfico 8.

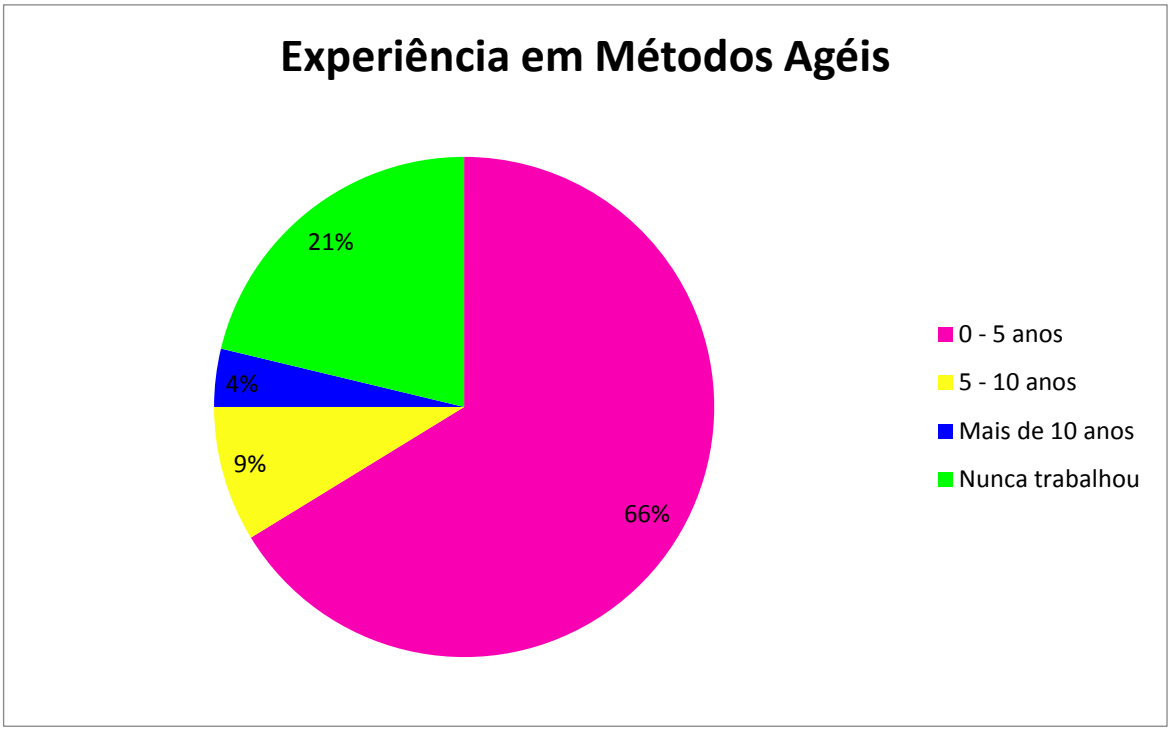

Gráfico 8 - Experiência em Métodos Ágeis

Fonte: Própria autora

Como foi apresentado no Gráfico 8, a maioria dos respondentes, 66% trabalham ou trabalharam com metodologias ágeis entre 0 e 5 anos, 9% possuem experiência entre 5 e 10 anos, uma pequena parcela, 4% dos participantes possuem experiência de mais de 10 anos com metodologias ágeis e cerca de 21% dos respondentes nunca trabalharam com as metodologias ágeis o que nos leva a crer que são profissionais que ainda utilizam métodos tradicionais para desenvolver software.

3.1.4.2 Quais metodologias ágeis são utilizadas nos projetos de desenvolvimento de software na sua empresa?

Para essa pergunta, os respondentes deveriam marcar se as metodologias ágeis apresentadas eram conhecidas e usadas na empresa em que ele trabalha, se ele as conhecia e não eram utilizadas na empresa ou se não conhecia aquela metodologia apresentada. Para a análise foi considerada apenas as respostas nas

quais os participantes alegaram que conheciam e usavam aquelas metodologias ágeis nas suas empresas, como pode ser observado no Gráfico 9.

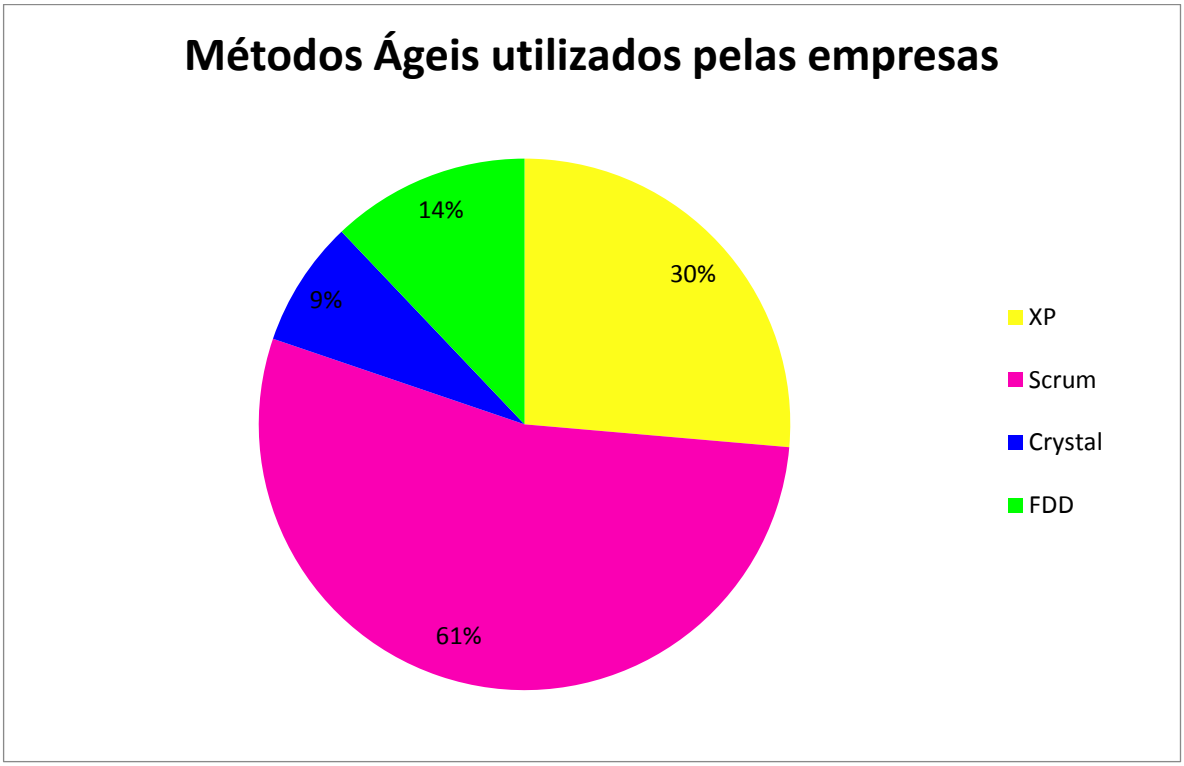

Gráfico 9 - Métodos Ágeis utilizados pelas empresas

Fonte: Própria autora

A maioria dos entrevistados, 61% responderam que conhecem e utilizam o Scrum na empresa, concordando com Brito (2010) quando ele diz que o Scrum tem sido largamente utilizado em projetos de desenvolvimento de software. O segundo mais utilizado nas empresas é o XP com 30% das respostas, 14% utilizam o FDD nos projetos da empresa e a minoria, cerca de 9% afirmaram que utilizam e conhecem o Crystal.

Após essa questão o respondente tinha a oportunidade de citar outras metodologias ágeis existentes e que não foram mencionadas na questão anterior. Alguns participantes mencionaram as metodologias Kanban, *Dynamic Systems Development Method* - Metodologia de desenvolvimento de sistemas dinâmicos (DSDM), *Lean, Adaptative Software Development* – Desenvolvimento Adaptativo de Software (ASD) e o *Test Driven Development* - Desenvolvimento Orientado a Testes (TDD), com isso nota-se o surgimento de novas metodologias ágeis nas empresas de desenvolvimento de software.

3.1.4.3 Classifique a importância de uma documentação mais formal durante a utilização de alguma metodologia ágil.

Essa questão teve como objetivo exibir a importância de uma documentação de requisitos mais formal mesmo utilizando um método ágil. Nessa questão os respondentes deveriam utilizar as classificações de 0 a 5, sendo classificação = 0 “Não é importante, classificação = 1 “Indiferente”, classificação = 2 “Pouco importante”, classificação = 3 “Importante”, classificação = 4 “Muito importante” e classificação = 5 “Extremamente importante”. No Gráfico 10 é possível observar as respostas referentes a questão.

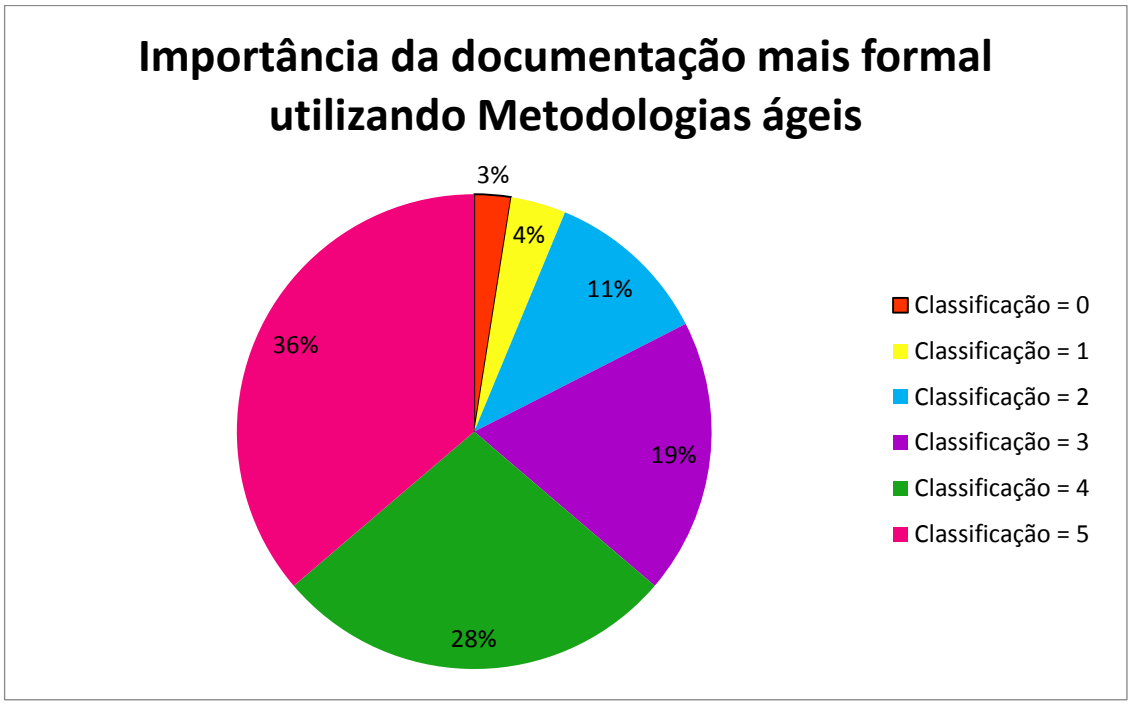

Gráfico 10 - Importância de uma documentação mais formal utilizando Metodologias Ágeis

Fonte: Própria autora

Pode-se concluir com os resultados do Gráfico 10 que 36% dos respondentes consideram utilizar uma documentação formal mesmo utilizando metodologias ágeis. Apenas 14 pessoas referentes a 3%, 4% e 11% dos respondentes apresentados no Gráfico 10, classificaram a documentação formal como insignificante ou pouco importante durante o desenvolvimento de um projeto ágil.

Com isso pode-se afirmar que a maioria dos entrevistados, cerca de 66 pessoas referentes a 36%, 28% e 19% dos respondentes apresentados no Gráfico 10, não estão totalmente de acordo com o primeiro princípio que o Manifesto Ágil (2001) defende, onde alegam que em um projeto ágil o “Software funcionando é mais importante do que documentação completa e detalhada”.

3.1.4.4 Quais práticas ágeis sobre requisitos são utilizadas na sua empresa?

Para essa pergunta, os respondentes deveriam marcar se as técnicas utilizadas nos projetos ágeis eram conhecidas e usadas na empresa em que ele trabalha, se ele as conhecia e não eram utilizadas na empresa ou se não conhecia aquela técnica apresentada. Para a análise foi considerada apenas as respostas em que os participantes alegaram que conheciam e usavam aquelas técnicas nas suas empresas, como pode ser observado no Gráfico 11.

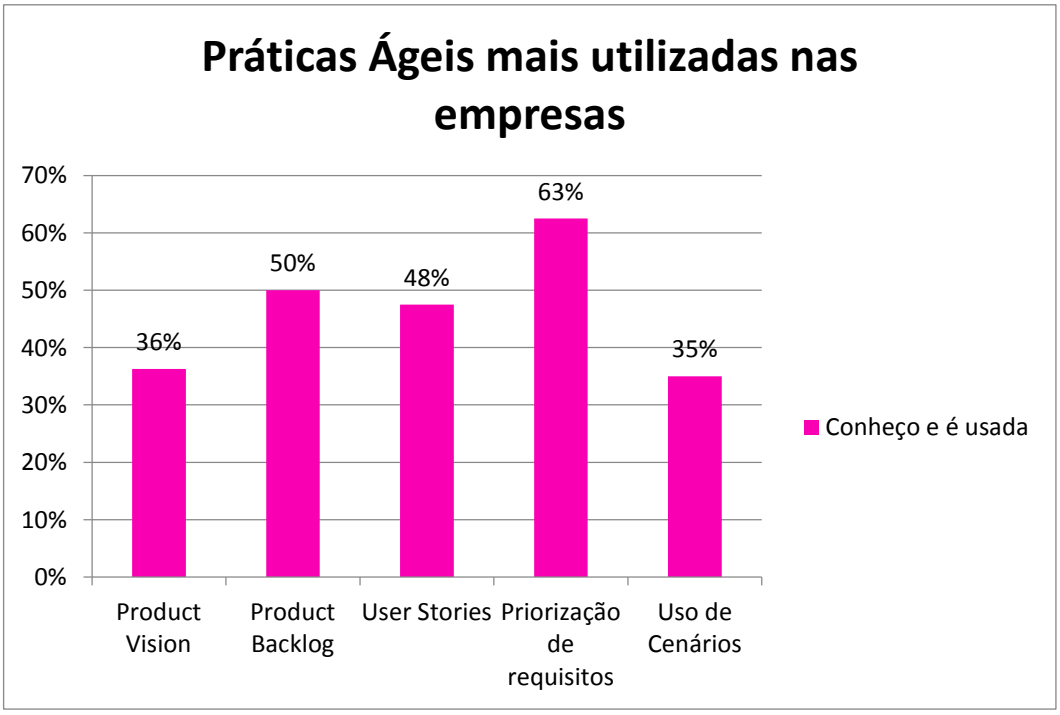

Gráfico 11 - Práticas Ágeis mais utilizadas nas empresas

Fonte: Própria autora

Pode-se concluir que parte dos respondentes, 63% utilizam a prática ágil “Priorização de requisitos” para a elicitação dos requisitos, seguido de 50% que utilizam mais o “*Product Backlog*”, a técnica de “*User Stories”* ou Histórias de usuário foi apontada como mais usada por 48% dos entrevistados. As práticas ágeis menos utilizadas nas empresas foram “*Product Vision*” e “Cenários”, com 36% e 35% de respostas respectivamente.

3.1.4.5 Na sua opinião, quais são as vantagens das metodologias ágeis?

Para essa questão, os participantes deveriam assinalar quais as vantagens das metodologias ágeis que são utilizadas na empresa em que atuam. Nessa questão foi permitido aos entrevistados que selecionassem mais de uma opção de resposta e por esse motivo, a soma de vantagens das metodologias ágeis assinaladas pelos respondentes ultrapassam a quantidade de pessoas que responderam a este questionário.

As respostas dessa questão são visualizadas no Gráfico 12:

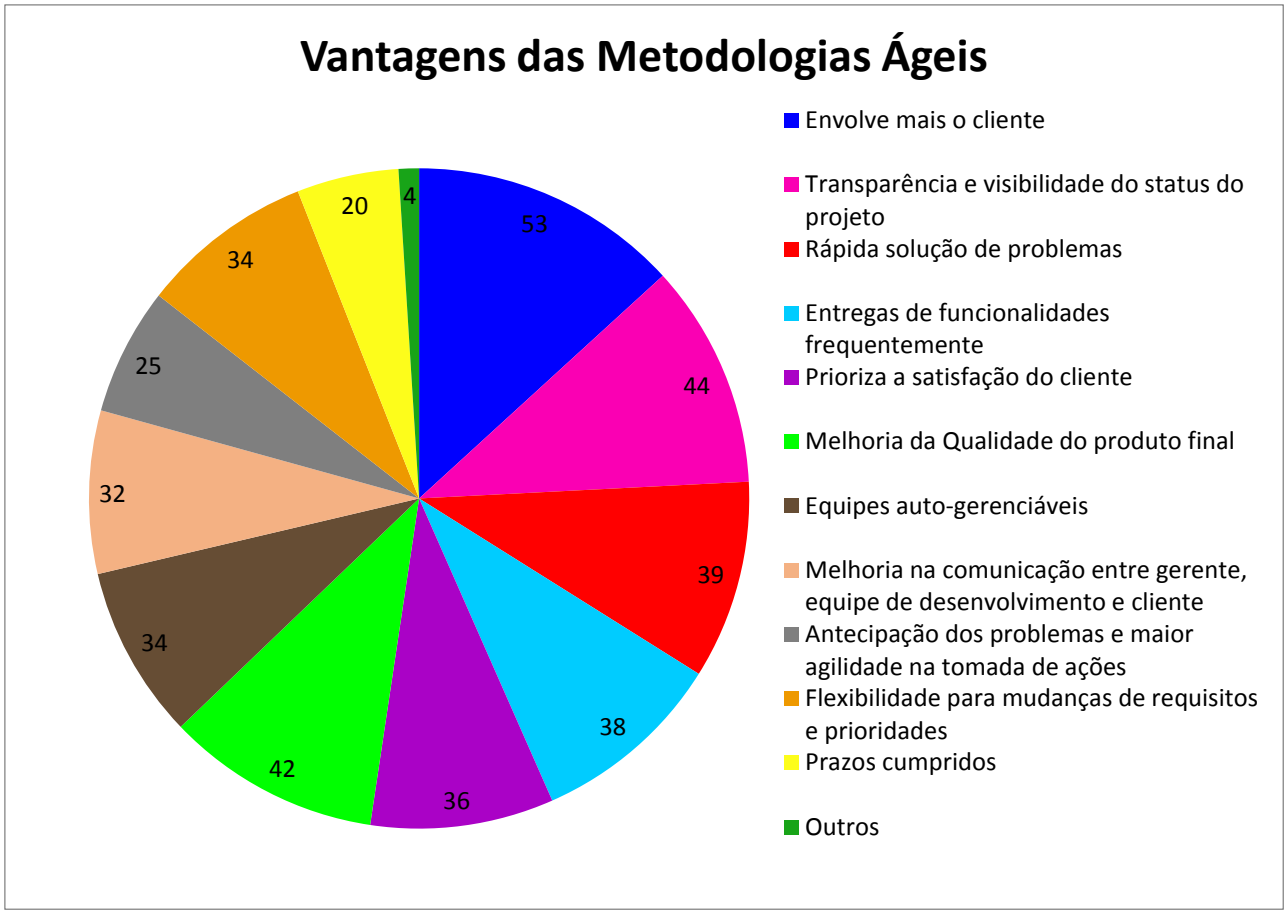

Gráfico 12 - Vantagens das Metodologias Ágeis

Fonte: Própria autora

A maior parte dos participantes assinalaram como vantagem “Envolve mais o cliente” (66% que correspondem a 53 pessoas), “Transparência e visibilidade do status do projeto” (55% que correspondem a 44 pessoas), “Melhoria da Qualidade do produto final” (53% que correspondem a 42 pessoas), “Rápida solução de problemas” (49% que correspondem a 39 pessoas), “Entregas de funcionalidades frequentemente” (48% que correspondem a 38 pessoas) e “Prioriza a satisfação do cliente” (45% que correspondem a 36 pessoas).

Um dos participantes marcou também a opção “Outros” mencionou que os Métodos Ágeis são recomendados também para projetos longos e adicionou essa recomendação como vantagem das metodologias ágeis.

Com isso pode-se afirmar que a maioria dos entrevistados concordou com alguns dos 12 valores do Manifesto Ágil (2001), que serviram como base para a apresentação das vantagens mencionadas nessa questão.

3.1.4.6 Na sua opinião, quais são as desvantagens das metodologias ágeis?

Para essa questão, os participantes deveriam marcar quais as desvantagens das metodologias ágeis que eles utilizam no dia a dia. Nessa questão foi permitido aos entrevistados que selecionassem mais de uma opção de resposta para essa questão e por esse motivo, a soma de desvantagens das metodologias ágeis assinaladas pelos respondentes ultrapassam a quantidade de pessoas que responderam a este questionário.

As respostas dessa questão são visualizadas no Gráfico 13:

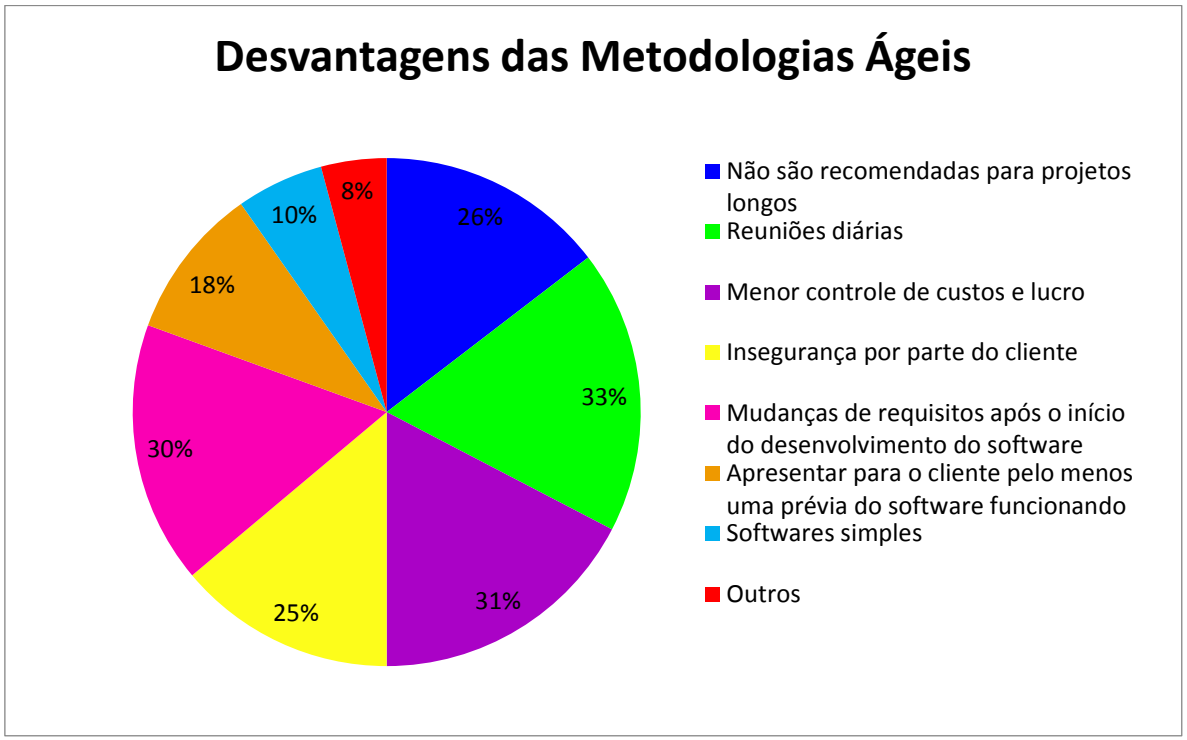

Gráfico 13 - Desvantagens das Metodologias Ágeis

Fonte: Própria autora

A maior parte dos participantes assinalaram como desvantagem das metodologias ágeis “Reuniões Diárias” (33% que correspondem a 26 pessoas) que às vezes podem ser inconvenientes para os envolvidos no projeto, “Menor controle de custos e lucro” (31% que correspondem a 25 pessoas), “Mudanças nos requisitos após o início do desenvolvimento do software” (30% que correspondem a 24

pessoas), 25% dos entrevistados marcaram como desvantagem “Não são recomendadas para projetos longos” e 25% afirmam que a insegurança por parte do cliente é uma desvantagem num projeto ágil.

Alguns participantes (8% dos respondentes) mencionaram outras desvantagens como: “Insegurança na implantação por parte de equipe pouco preparada ou pela direção da empresa ligada a métodos clássicos”, “Difícil de gerenciar, depende da cultura da empresa” e “Não considero as opções apresentadas aqui como desvantagens”.

Pode-se concluir que os respondentes tiveram muito mais resistência em assinalar desvantagens dos métodos ágeis, alguns até alegaram que não existem desvantagens em colocar em prática um projeto ágil.

As outras questões dessa seção foram baseadas nos valores dos Manifesto ágil (2001) e serão exibidas a seguir.

3.1.4.7 Classifique a importância dos valores do Manifesto Ágil citados abaixo:

Para essa pergunta, os respondentes deveriam classificar os quatro valores principais do Manifesto Ágil (2001) como “Muito importante”, “Importante”, “Pouco Importante”, “Não é importante” e “Não sei”. Foram selecionadas apenas as respostas em que os participantes classificaram como o manifesto exibido como “Muito importante” e “Importante”, como pode ser observado no Gráfico 14.

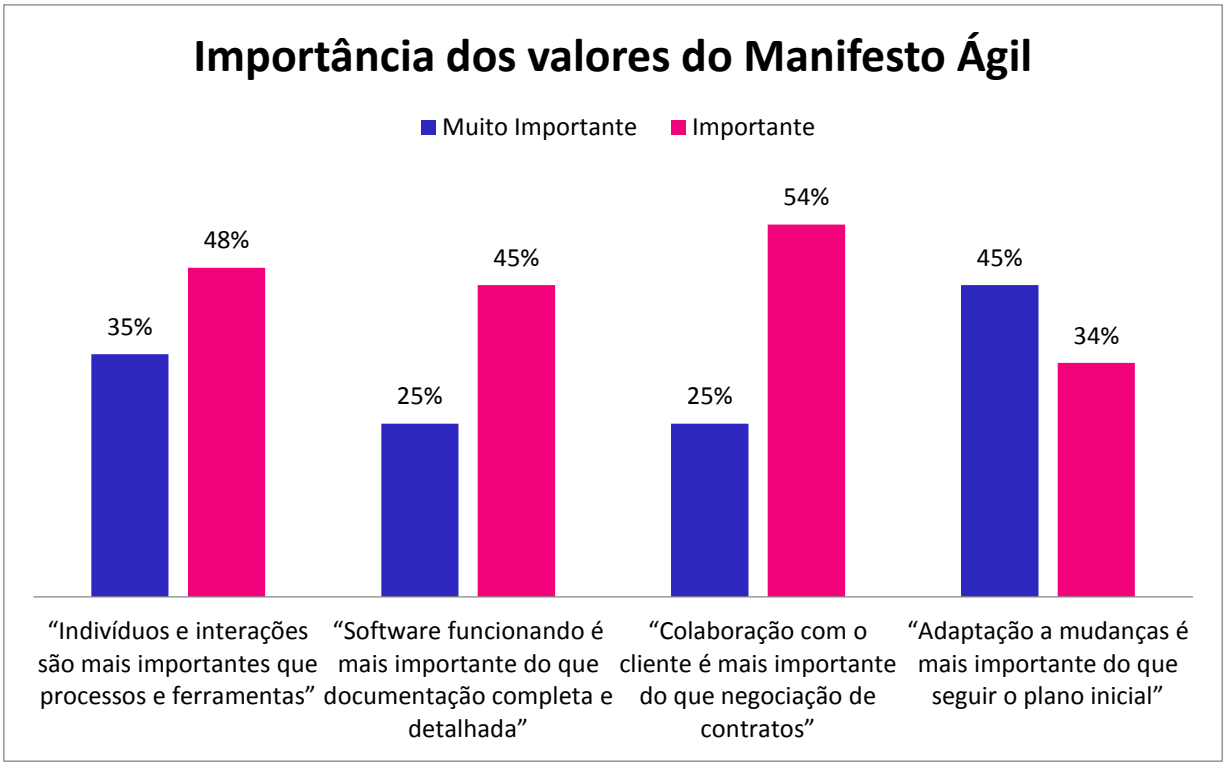

Gráfico 14 - Importância dos valores do Manifesto Ágil

Fonte: Própria autora

A maioria dos respondentes, 83% classificaram o valor “Indivíduos e interações são mais importantes que processos e ferramentas”, 79% afirmam que “Adaptação a mudanças é mais importante do que seguir o plano inicial” e “Colaboração com o cliente é mais importante do que negociação de contratos”. como o mais importante. O valor, “Software funcionando é mais importante do que documentação completa e detalhada”, foi apontado como importante por 70% dos participantes.

Conclui-se após a análise dos resultados dessa questão que os respondentes não concordam totalmente nos valores do Manifesto Ágil, mas podem estar procurando maneiras melhores de desenvolver os softwares de acordo com eles, concordando com Bernardo (2014).

As próximas quatro questões dessa seção foram baseadas nos princípios do Manifesto Ágil de 2001 e tem como objetivo descobrir se alguns desses princípios são colocados em prática nas equipes de desenvolvimento seja de forma total, parcial ou não são colocadas em prática.

3.1.4.8 Projetos desenvolvidos utilizando métodos ágeis requerem um ritmo constante de trabalho durante todas as fases. Seu ambiente de trabalho oferece motivação para que seu rendimento seja bom?

“Construir projetos ao redor de indivíduos motivados. Dando a eles o ambiente e suporte necessário, e confiar que farão seu trabalho” (Manifesto Ágil, 2001).

Essa questão teve como objetivo descobrir se o ambiente de trabalho do participante oferece a motivação necessária para que seu desempenho durante o trabalho seja bom. O gráfico a seguir exibe os resultados dessa questão.

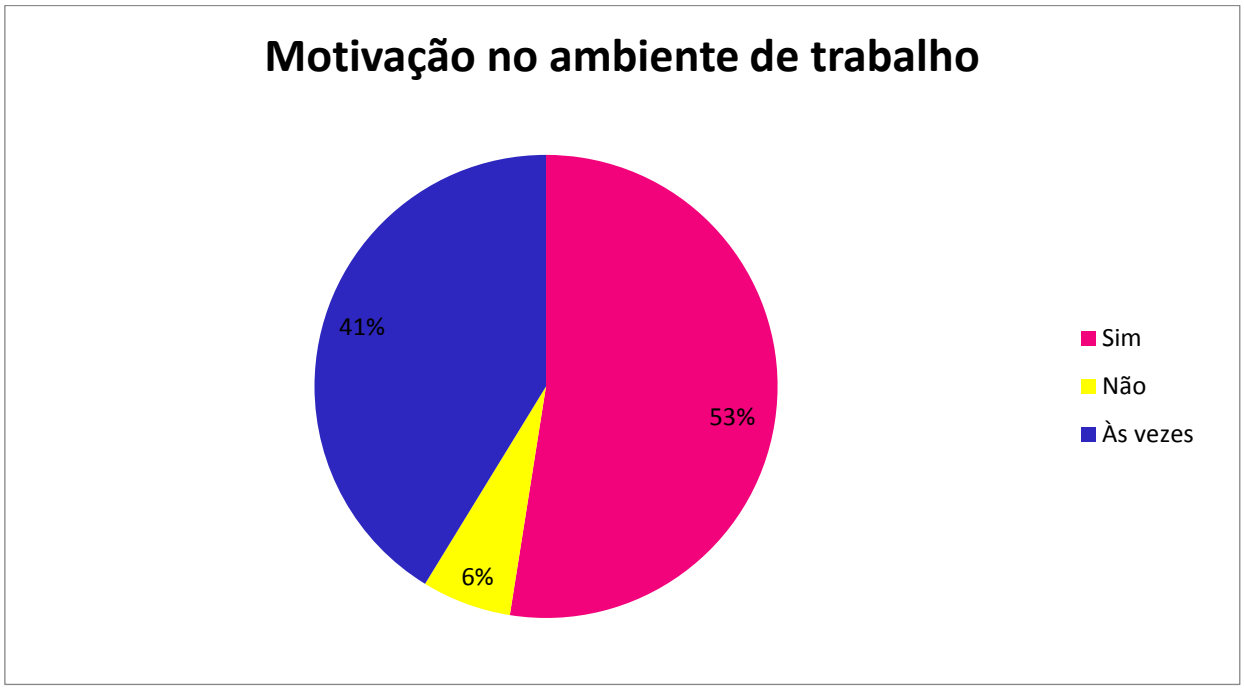

Gráfico 15 - Motivação no ambiente de trabalho

Fonte: Própria autora

Como se pode observar no Gráfico 15, cerca de 53% afirmaram que são motivados em seu ambiente de trabalho, contra 41% que afirmam ter um ambiente motivacional algumas vezes e a minoria, cerca de 6% dos entrevistados, alegam não trabalharem em um ambiente motivacional. Podem-se concluir que a maioria dos entrevistados trabalham em um ambiente motivador, o que entra em concordância com o princípio defendido pelo Manifesto Ágil.

3.1.4.9 De acordo com um dos princípios do Manifesto ágil a melhor maneira de compartilhar informações para e entre uma equipe de desenvolvimento é através de conversa aberta, de forma presencial. Em sua empresa, as informações são transmitidas dessa forma?

"O Método mais eficiente e eficaz de transmitir informações para, e por dentro de um time de desenvolvimento, é através de uma conversa cara a cara" (Manifesto Ágil, 2001).

Essa questão teve como objetivo descobrir se a equipe de desenvolvimento do entrevistado compartilha as informações de maneira eficiente e compartilham informações através de conversas informais cara a cara. O gráfico a seguir exibe os resultados dessa questão.

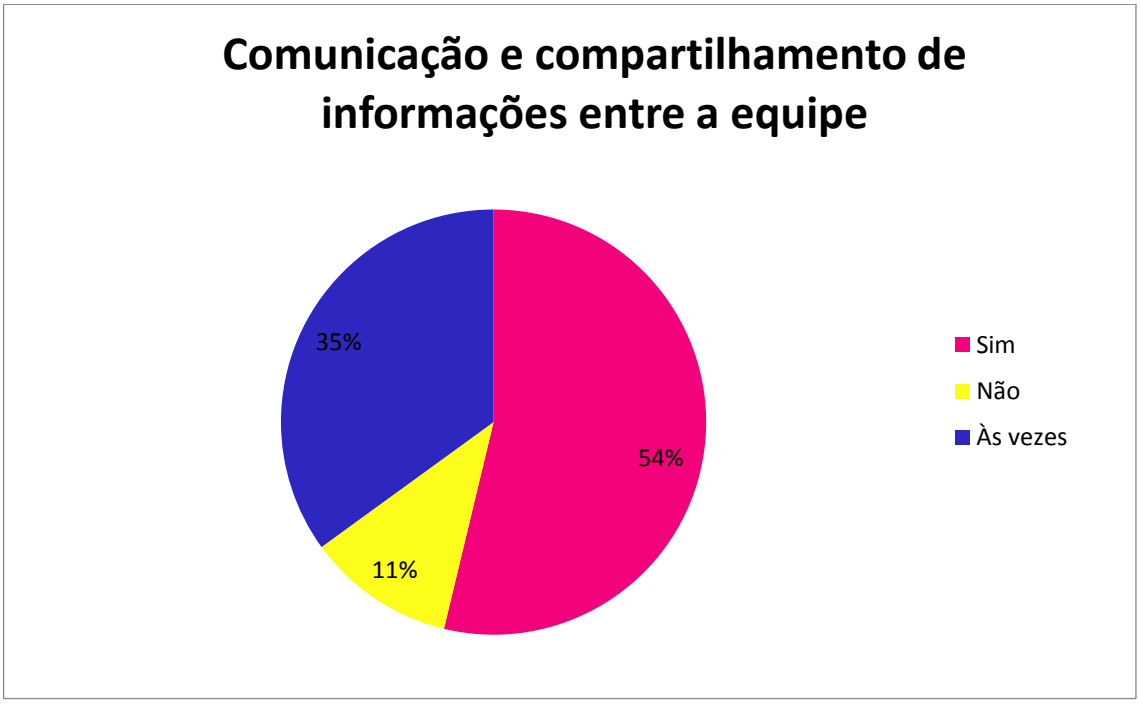

Gráfico 16 - Comunicação e compartilhamento de informações entre a equipe

Fonte: Própria autora

Como se pode observar no Gráfico 16, cerca de 54% afirmaram que há comunicação e compartilhamento de informações entre a sua equipe de desenvolvimento, contra 35% que afirmam que essa comunicação ocorre apenas de vez em quando e a minoria, 11% dos respondentes afirma que não ocorre comunicação e nem compartilhamento de informações em nenhum momento em sua equipe de trabalho. Pode-se concluir que a maioria dos entrevistados trabalham com uma equipe de desenvolvimento que se comunica frequentemente e compartilha informações de forma eficiente.

3.1.4.10 Sua equipe se reúne regularmente para pensar em maneiras de se tornarem mais eficazes e ajustarem o comportamento de acordo com aquilo que foi pensado?

"Em intervalos regulares, o time reflete em como ficar mais eficiente, então, se ajustam e otimizam seu comportamento de acordo" (Manifesto Ágil, 2001).

Essa questão teve como objetivo descobrir se a equipe de desenvolvimento do participante se reúne para propor melhorias nos softwares e no modo como conduzem os projetos de desenvolvimento. O gráfico a seguir exibe os resultados dessa questão.

Gráfico 17 - Equipe se reúne para pensar em maneiras de se tornarem mais eficazes

Fonte: Própria autora

Como se pode observar no Gráfico 17, cerca de 46% afirmaram que a equipe se reúne regularmente para pensar em maneiras de serem mais eficientes, contra 38% que afirmam que essas reuniões entre a equipe para serem mais eficazes não acontecem em nenhum momento nas organizações a qual eles trabalham e a minoria, 16% dos respondentes afirmam que às vezes a equipe de desenvolvimento se reúne para pensar em maneiras de se tornarem uma equipe melhor. Pode-se concluir que nessa questão quase metade dos entrevistados afirmaram que se reúnem com a sua respectiva equipe para pensar em maneiras de se tornarem mais eficientes durante um projeto de software, concordando com o princípio defendido pelo Manifesto Ágil.

3.1.4.11 Os prazos estabelecidos no início dos projetos são cumpridos?

"Nossa maior prioridade é satisfazer o cliente, através da entrega adiantada e contínua de software de valor" (Manifesto Ágil, 2001).

Essa questão teve como objetivo descobrir se a equipe de desenvolvimento do participante entrega o produto no prazo estabelecido no início do projeto do software. O gráfico a seguir exibe os resultados dessa questão.

Gráfico 18 - Os prazos são cumpridos pela empresa?

Fonte: Própria autora

Como se pode observar no Gráfico 18, cerca de 49% afirmaram que os prazos não são cumpridos pela equipe de desenvolvimento, contra 38% que afirmam que entregam os softwares no prazo estabelecido no início do projeto e a minoria, 14% afirmam que às vezes os prazos previamente determinados são atendidos pela equipe. Podemos concluir que nessa questão quase metade dos entrevistados afirmaram que não cumprem os prazos de entrega do software para o cliente, discordando do princípio do Manifesto Ágil que colocam como prioridade satisfazer o cliente, entregando o software no prazo estabelecido no início do projeto ou até mesmo antes desse prazo final.

A próxima seção são questões referentes ao relacionamento do cliente com a empresa e equipe de desenvolvimento de software.

3.1.5 Quinta seção: Relacionamento com o Cliente

Essa seção tem como objetivo saber mais sobre a comunicação entre cliente e a organização. Todas as questões são focadas no comportamento do cliente em algumas fases da engenharia de requisitos ágeis.

3.1.5.1 Como você classifica o grau de satisfação do cliente em relação ao número de reuniões feitas ao longo do projeto desenvolvido com metodologias ágeis?

Essa questão tem o objetivo de saber mais sobre a satisfação do cliente em relação às inúmeras reuniões que são realizadas durante o desenvolvimento de um projeto ágil. No Gráfico 19 é possível ver a classificação de satisfação.

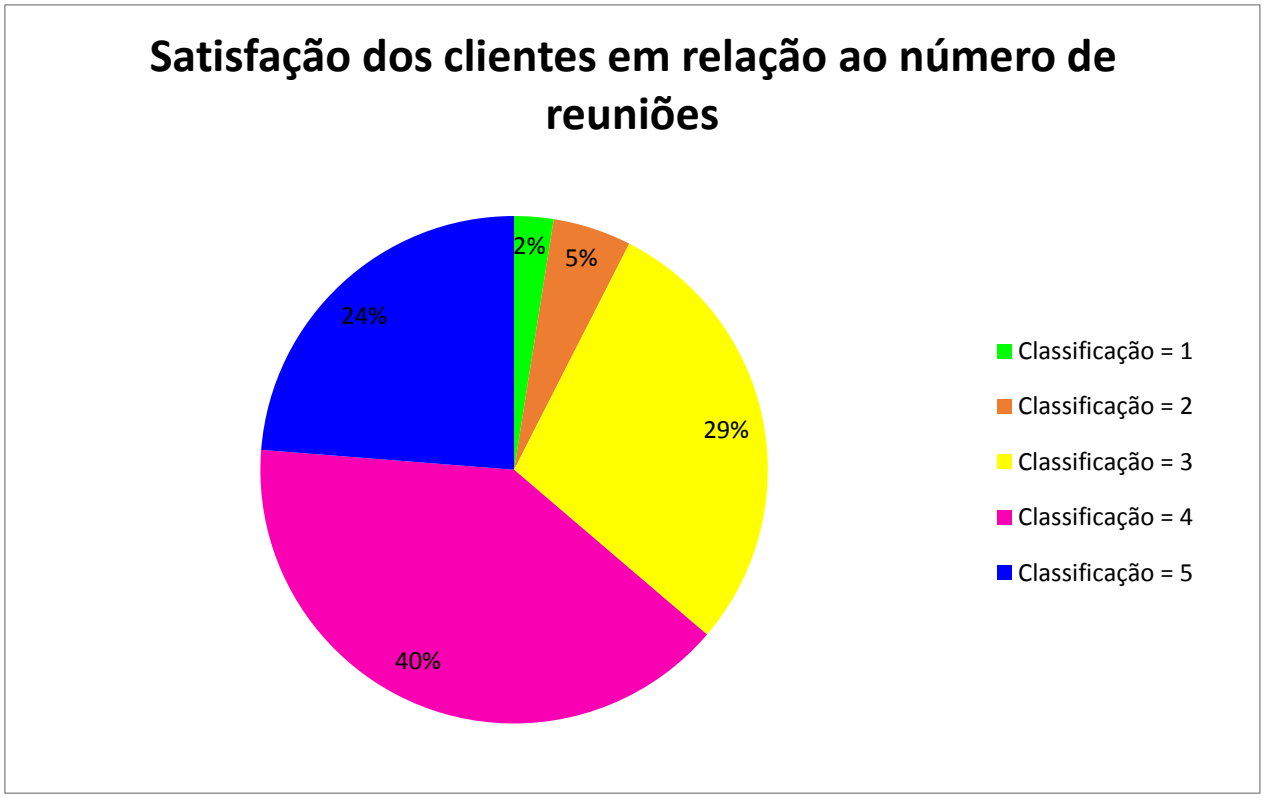

Gráfico 19: Satisfação dos clientes em relação ao número de reuniões

Fonte: Própria autora

Pode-se observar que 40% dos respondentes marcaram a classificação = 4 que quer dizer que o cliente fica satisfeito com o número de reuniões, contra 29% que deram classificação = 3 que significa que o cliente está pouco satisfeito com o número de reuniões, 24% dos respondentes consideram o cliente muito satisfeito com as reuniões, 5% classificam que os clientes se sentem indiferentes com o número de reuniões e 2% dos participantes afirmam que os clientes se sentem insatisfeitos com o número de reunião.

Conclui-se que o número de reuniões influencia muito pouco na satisfação do cliente pois segundo a classificação dos participantes a maioria dos clientes se sentem satisfeitos ou pouco satisfeitos com as reuniões, apenas uma pequena parcela se sente insatisfeita, tudo dependerá do andamento do projeto e como são feitas essas reuniões e depende muito do cliente, cada um tem sua maneira de pensar, tem suas atividades diárias e acrescentar reuniões talvez não seja tão satisfatório. Há também empresas em que o cliente sentirá prazer em acompanhar e dar o seu *feedback* e há outras que o cliente não ficará tão satisfeito em se reunir sempre com a equipe responsável pelo projeto.

3.1.5.2 Com que frequência ocorre mudanças de requisitos por parte do cliente? As exigências feitas pelo cliente são sempre acatadas pela equipe?

Essas duas perguntas tem como objetivo descobrir a frequência na qual o cliente resolve mudar os requisitos e se as exigências que ele faz são sempre feitas pela equipe de desenvolvimento, uma vez que o Manifesto ágil (2001) defende o princípio de que as equipes devem sempre aceitar as mudanças de requisitos e sempre ter como prioridade a satisfação do cliente.

O gráfico a seguir exibirá se as exigências feitas são sempre acatadas pela equipe:

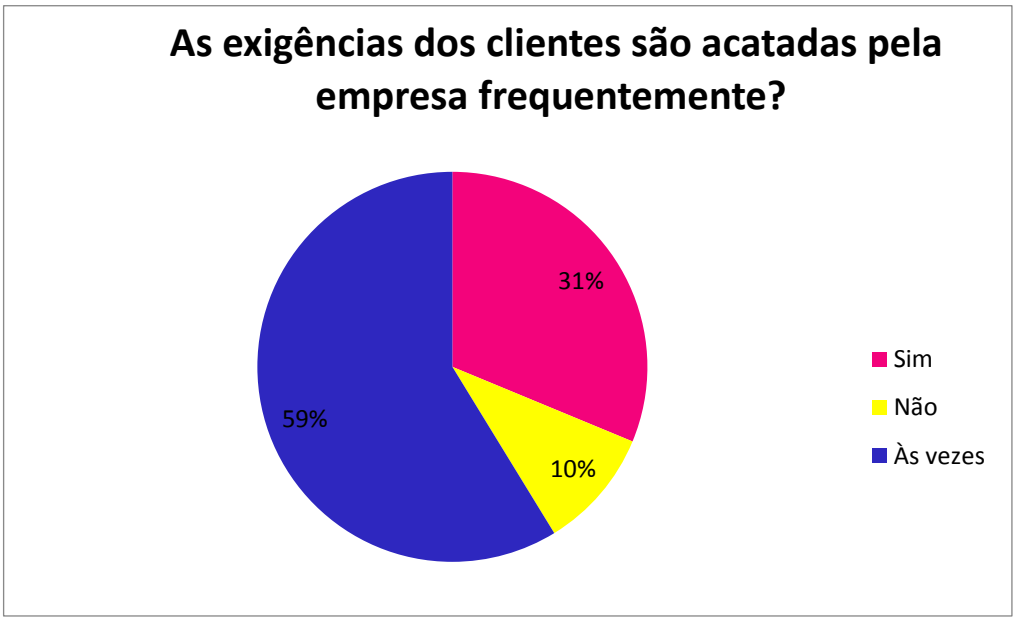

Gráfico 20 - As exigências dos clientes são acatadas pela empresa frequentemente?

Fonte: Própria autora

No Gráfico 20 é possível observar que 59% dos respondentes afirmam que as exigências do software que o cliente faz não são sempre acatadas pela equipe de desenvolvimento, entrando em conflito com um dos princípios do Manifesto Ágil que defende que as empresas devem sempre priorizar a satisfação do cliente. Aproximadamente 31% dos participantes afirmaram que acatam as exigências do cliente frequentemente e apenas 10% afirmaram que não atendem as exigências dos clientes de maneira nenhuma.

A partir do princípio do Manifesto ágil (2001), “aceitar mudanças de requisitos, mesmo no fim do desenvolvimento” foi perguntado aos respondentes com que

frequência o cliente modifica os requisitos e foi constatado que 31% dos participantes afirmaram que as modificações dos requisitos são muito frequentes, são frequentes de acordo com 28% e pelo menos pouco frequente de acordo com 29% dos entrevistados. Logo após foi exibida a questão que está representada no Gráfico 20 sobre as exigências dos clientes são acatadas pela equipe de desenvolvimento. Com as informações recolhidas das duas respostas pode-se afirmar que a maioria dos clientes pedem modificações no projeto durante o desenvolvimento do mesmo e as equipes de desenvolvimento nem sempre aceitam e acatam a essas exigências, indo a desacordo com alguns princípios do Manifesto ágil.

3.1.5.3 O cliente demonstra mais confiança no projeto acompanhando todos os passos de desenvolvimento do software?

Essa questão teve o objetivo de identificar se o cliente demonstra mais confiança na organização e na equipe que está desenvolvendo o software se ele acompanhar todas as fases de perto. O Gráfico 21 exibe as respostas dos participantes sobre a questão.

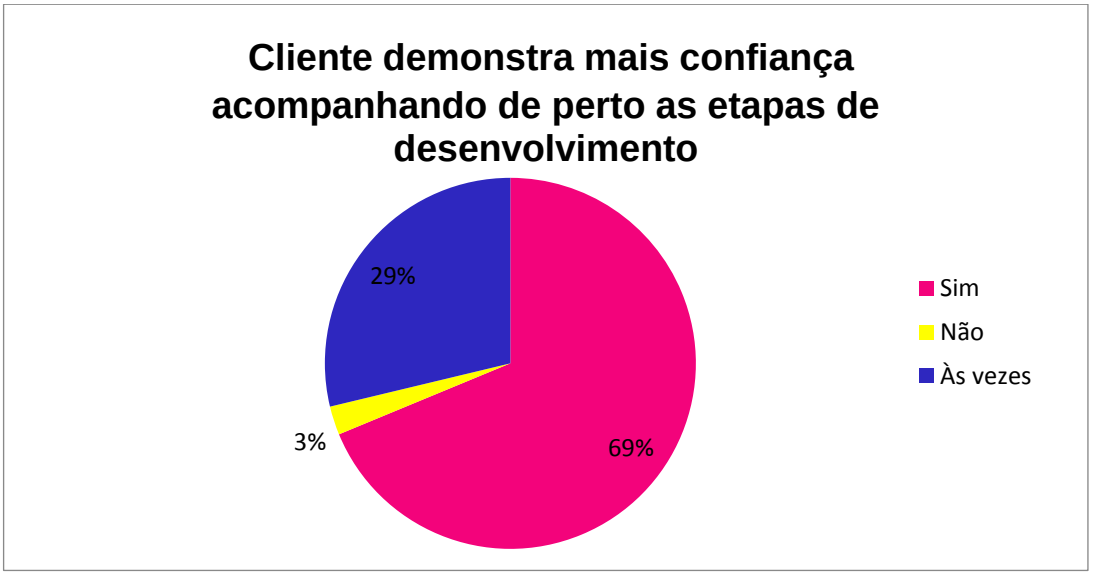

Gráfico 21 - Cliente demonstra mais confiança acompanhando de perto as etapas de desenvolvimento

Fonte: Própria autora

Pode-se observar no Gráfico 21 que 69% dos respondentes afirmam que o cliente fica mais confiante se acompanhar todas as fases do projeto de perto, contra apenas 3% dos entrevistados que alegaram que o cliente não demonstra essa confiança mesmo acompanhando todas as etapas do projeto e 29% dos respondentes afirmam que essa confiança por parte do cliente só acontece de vez em quando.

Com isso conclui-se que segundo a opinião dos participantes, quando o cliente acompanha de perto todas as fases do projeto de desenvolvimento do sistema ele fica mais confiante e essa confiança acaba dando um *feedback* muito bom para a empresa e para equipe que por fim se sente mais confiante para se tornarem melhores.

3.1.5.4 O cliente aprova as prévias do sistema funcionando que são apresentadas em curtos intervalos de tempo? Qual/quais motivo(s) levam os clientes a reprovar as prévias do sistema.

Essa questão teve como objetivo descobrir se o cliente aprova as prévias do software que são entregues a ele em intervalos pequenos e logo após assinalar quais os motivos que influenciam o cliente a reprovar essas prévias. O Gráfico 22 exibe esses motivos.

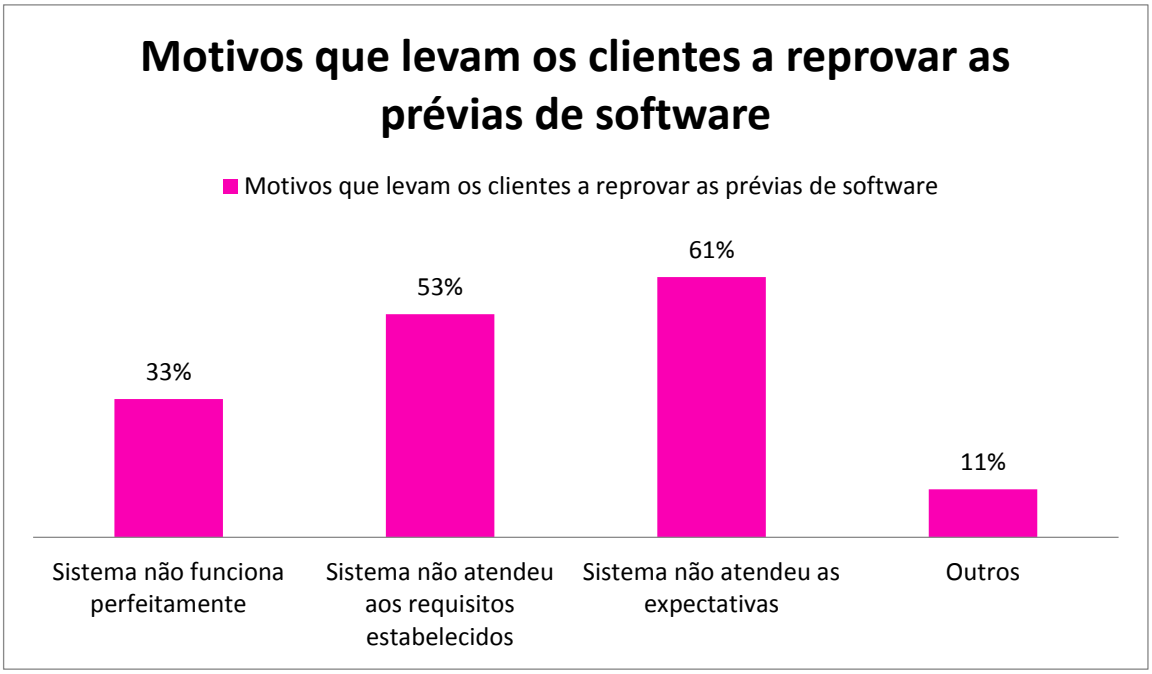

Gráfico 22 - Motivos que levam os clientes a reprovar as prévias de software

Fonte: Própria autora

Pode-se observar nessas duas questões colocadas em pauta nessa seção que 49% dos respondentes afirma quem os clientes aprovam as prévias de software sempre, contra 43% que afirma que essa aprovação acontece de vez em quando, dependendo muito do software que está sendo desenvolvido e da equipe que está desenvolvendo.

No Gráfico 22 aponta que 61% dos clientes reprovam as prévias do software pois não atendeu as expectativas dele, 53% afirmam que essa reprovação ocorre devido aos requisitos estabelecidos não foram atendidos, 33% dos respondentes alegam que os clientes reprovam o sistema pois ele não funciona corretamente

Apenas 11% dos entrevistados assinalaram a opção “Outros” e afirmaram que na maioria das vezes o cliente não aprovam as prévias do software devido ao prazo que foi estourado ou como o cliente gosta de ver o software pronto e funcionando, apresentar esses protótipos o deixará insatisfeito principalmente se o sistema gerar algo que não foi esperado durante essas prévias e com isso, ele não dará o *feedback* adequado e reprovará a prévia do sistema.

3.1.5.5 Como você classifica o grau de satisfação do cliente em relação à entrega do produto final?

Foi possível aos respondentes dessa questão classificar o grau de satisfação do cliente assinalando de 0 (zero) a 5 (cinco), sendo zero “Insatisfeito” e cinco “Muito satisfeito”. O resultado dessa questão é exibido no gráfico a seguir.

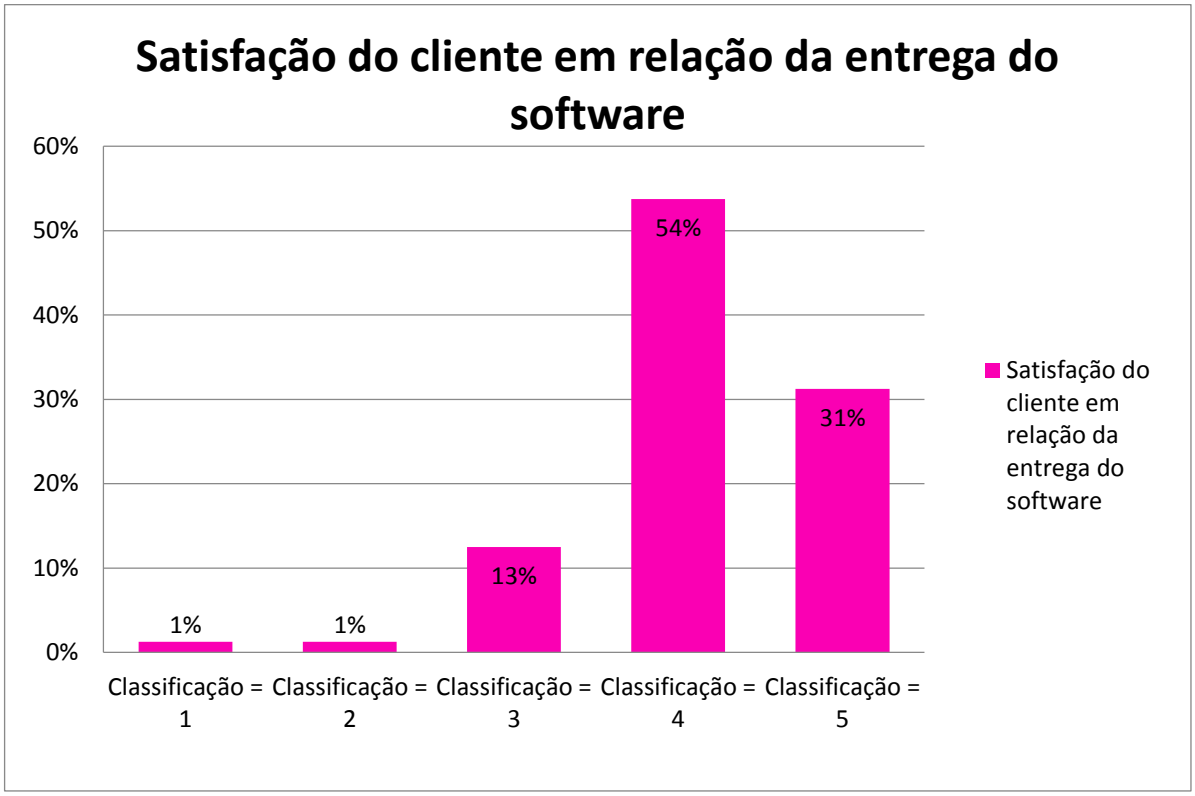

Gráfico 23 - Satisfação do cliente em relação a entrega do software final

Fonte: Própria autora

A maior parte dos entrevistados (54% dos participantes) atribuíram a o valor 4 que significa que os clientes ficam satisfeitos com o software final. Outros 31% dos participantes atribuíram o valor 5 que significa que os clientes ficam muito satisfeitos com o software final. Com isso pode-se concluir que 85% dos participantes afirma que o software desenvolvido com métodos ágeis são bem aceitos pelos clientes. A alternativa que tinha o valor 3 foi assinalada apenas por 10 respondentes e o valor 2 e 1 foram assinalados por apenas 1 pessoa cada e nenhum respondente classificou com o valor 0 e por esse motivo ele não está sendo representado no gráfico. É

possível inferir que segundo a opinião geral dos entrevistados os clientes ficam bem satisfeitos com o software final que é entregue no final do projeto.

Foi evidenciado por meio desse questionário as vantagens das metodologias ágeis foram expostos por meio das respostas dos participantes, os desafios diários encontrados pelas empresas que trabalham com o desenvolvimento de software, desde a comunicação entre eles e a comunicação com o cliente e as técnicas usadas nas empresas para obter um software de qualidade sem perder o foco nos princípios do Manifesto Ágil.

Após uma análise detalhada das respostas obtidas através do questionário como citado anteriormente, foi possível visualizar melhor elementos práticos e teóricos e confrontá-los com a opinião de autores.

Com os resultados obtidos foi possível concluir que a maioria dos participantes trabalha como desenvolvedor e possuem pouca experiência em desenvolvimento ágil e também tradicional. É possível afirmar que as metodologias tradicionais estão entrando em desuso haja vista que os respondentes afirmaram ser inexperientes ou possuírem pouca experiência sobre o assunto. Conclui-se também que as empresas estão se adaptando aos princípios e valores defendidos pelo Manifesto ágil e os seus clientes estão tendo uma ótima aceitação dos softwares desenvolvidos com metodologias ágeis.

4. CONCLUSÃO

Os métodos ágeis surgiram como uma resposta às falhas que ocorrem durante projetos que utilizam métodos tradicionais e ganharam seu espaço nas empresas. Foi observado que as metodologias retratadas no “Manifesto Ágil” e as que surgiram após o manifesto podem ser modificadas e até utilizadas juntamente com outras metodologias, para que a empresa tenha o melhor desempenho possível e se destaque no mercado.

Para alcançar o objetivo deste trabalho foi desenvolvido um questionário onde havia questões que induziam o participante a dar a sua opinião sobre a Engenharia de requisitos, os métodos tradicionais, métodos ágeis, o Manifesto ágil e o relacionamento de sua equipe de desenvolvimento e o cliente.

A partir dos estudos realizados e nas respostas levantadas através do questionário, foi possível concluir que de uma maneira geral a Engenharia de Requisitos ágeis é vantajosa, sendo capaz de melhorar o desempenho das equipes durante o desenvolvimento de software, melhorando a qualidade, confiabilidade, reduzindo os prazos e os custos do sistema.

Pode-se concluir que os métodos ágeis são aceitos por grande parte dos profissionais de TI relacionados com o desenvolvimento de software e os mesmos se empenham para se adaptar aos valores e princípios do Manifesto ágil, mesmo que ainda não tenham adotado totalmente o desenvolvimento de projetos ágeis em suas empresas.

Ainda por meio dos resultados foi visto que as metodologias tradicionais estão sendo pouco utilizadas pelos respondentes e eles apontam mais desvantagens do que vantagens do uso do desenvolvimento tradicional.

Segundo os participantes, os clientes aprovam os projetos desenvolvidos com métodos ágeis e isso faz com que as empresas procurem sempre estar se atualizando e capacitando os profissionais para que os mesmos aprendam a se comunicar com os clientes.

Tendo sido realizadas algumas comparações entre os valores e princípios do Manifesto ágil sobre o assunto e as respostas obtidas através do questionário, concluiu-se que de modo geral a opinião dos entrevistados divergiu pouquíssimas vezes com o Manifesto.

Foi constatado que os profissionais entrevistados são mais jovens, possuem pouca experiência e a maioria trabalha mais com desenvolvimento ágil e não possuem muita experiência com o desenvolvimento de software tradicional, concluindo que os métodos ágeis são mais vantajosos do que as metodologias tradicionais e estão conquistando o espaço nas empresas de desenvolvimento e as metodologias tradicionais estão entrando em desuso.

5. TRABALHOS FUTUROS

Uma possível continuação desse estudo poderia ser a realização de um *framework* que comparasse e analisasse os métodos ágeis e tradicionais para cada tipo de projeto que uma empresa fosse desenvolver, assim selecionando quais metodologias seriam indicadas. Um estudo sobre a Engenharia de requisitos utilizando metodologias híbridas.

Uma proposta de melhoria do processo de Engenharia de Requisitos em uma empresa de desenvolvimento que utilize metodologias tradicionais através da Engenharia de Requisitos ágeis a fim de aperfeiçoar os serviços prestados, com software simples, seguindo os princípios e valores do Manifesto ágil.

A criação de um modelo de documentação simplificada para ser utilizado como documentação formal de softwares desenvolvidos utilizando metodologias ágeis para que os desenvolvedores tenham uma visão clara do software e tenham um “passo a passo” para seguirem.

REFERÊNCIAS

ALVES, Bruno Junqueira. **Estudo comparativo entre métodos ágeis e tradicionais de fabricação de software.** UPE, 2012.

ALVES, Daniela De Castro Pereria. **Técnicas de engenharia de requisitos no desenvolvimento ágil.** UFPE, 2014.

ALVES, Sérgio De Rezende; ALVES, André Luiz. **Engenharia de requisitos em metodologias ágeis.** PUC - Goiás, 2009.

BALLE, Andrea Raymundo. **Análise de Metodologias Ágeis: Conceitos, Aplicações e Relatos sobre XP e Scrum.** UFRGS, 2011.

BECK, K. **Programação Extrema (XP) Explicada: Acolha as Mudanças**, Bookman, 2004.

BERNARDO, Kleber. **Métodos ágeis: Convencendo um cético.** 1 ed. [S.L.: s.n.], 2014. 39 p.

BONA, Cristina. **Avaliação de processos de software: UM ESTUDO DE CASO EM XP E ICONIX.** UFSC, 2002.

BRAZ, Alan. **Método ágil aplicado ao desenvolvimento de software confiável baseado em componentes.** UNICAMP, 2013.

BRITO, Rebeka Sales De. **Uma proposta para modelagem de requisitos não-funcionais em projetos ágeis.** UFPE, 2010.

CARVALHO, Anderson Abner De. **Scrummps 2.0 - evolução de uma ferramenta interativa para suporte ao SCRUM:** Evolução de uma Ferramenta interativa para suporte ao Scrum e MPS.BR. UFLA, 2013.

CINTRA, Caroline Carbonell**. A implantação de um processo de engenharia de requisitos baseado no RUP alcançando nível 3 de CMMI incluindo a utilização de práticas de métodos ágeis**: Porto Alegre: UFRGS, 2006. 160 p.

D'AMORIM, Fernanda Rodrigues dos Santos. **Engenharia de Requisitos para Métodos Ágeis**. UFP - Centro de Informática, 2008.

COELHO, Cristiane Dos Santos**. Relato de experiência na implantação de um método ágil em uma equipe de desenvolvimento de software**. UFLA, 2011.

COSTA, Gustavo Henrique De Carvalho. **Engenharia de requisitos no desenvolvimento de software ágil.** UFPE, 2011.

Desafios e benefícios trazidos pela implementação do método ágil SCRUM. 2011. Bridge Consulting.

DORIGAN, José André**. Um modelo de processo de Engenharia de Requisitos para padronização e aumento de qualidade.** UEL, 2013.

FERNANDES, Matheus Ramos. **SCRUM E XP: Um comparativo no processo de desenvolvimento de software.** FUMEC, 2011.

FILHO, Dairton Luiz Bassi. **Experiências com desenvolvimento ágil.** USP, 2008.

GHAI, Sunaina; KAUR, Jagpuneet. **Analysis of user requirements gathering practices in agile and non-agile software development team.** 2012.

JUNIOR, Cleo Hickmann; YANZER, Anderson. **Aplicação de métodos ágeis em um processo de desenvolvimento de software.** ULBRA, 2011.

LEITE, Sarah Figueiredo. **Inspeção de usabilidade aplicada a métodos ágeis: Um estudo de caso.** UFLA,2013.

MAINART, Domingos De A.; SANTOS, Ciro M.. **Desenvolvimento de software: processos ágeis ou tradicionais? Uma visão crítica.** UFVJM, 2009.

MANIFESTO, Ágil. **Manifesto para o desenvolvimento ágil de software.** Disponível em: < http://www.manifestoagil.com.br/>. Acesso em: 5 de agosto de 2015.

MATUDA, Danielle M.; BEGOSSO, Luiz C.. **Mapas mentais na engenharia de requisitos:** FEMA, 2010.

NETO, Vicente Bezerra De Souza. **Aplicação de um processo ágil com foco em gestão de riscos:** UPE, 2008.

OLIVEIRA, Victor. **Modelagem da atividade de elicitação do processo de Engenharia de Requisitos: Uma abordagem utilizando dinâmica de sistemas.** Universidade Federal de Viçosa - UFV, 2008.

PEREIRA, Silvia Cássia. **Um estudo empírico sobre a engenharia de requisitos em empresas de produtos de software**. UFPE, 2007.

PRESSMAN, R.S. **Engenharia de Software**. 6ª Ed, McGraw-Hill, 2006.

PRESSMAN, Roger S. **Engenharia de software.** 7ª ed. Porto Alegre: Bookman, 2011.

SILVA, Carlos Eduardo Azevedo Costinhas Da. **Um estudo de caso sobre adoção de práticas ágeis em um ambiente tradicional**. UFRJ, 2013.

SILVA, Samuel Fabiano Barbosa. **Engenharia de requisitos: Uma análise das técnicas de levantamento de requisitos.** FUMEC, 2012.

SILVA, Renato Afonso Cota. **PSP e Métodos Ágeis na melhoria da qualidade em produção de software: Um estudo de caso.** Universidade Federal de Viçosa - UFV, 2006.

SILVA, Thaís Mara. **Uma proposta de novos estudos para melhoria do processo de elicitação de requisitos para uma cooperativa de software livre**. UFLA, 2009.

SILVEIRA, Maria Clara Dos Santos Pinto. **A reutilização de requisitos no desenvolvimento e adaptação de produtos de software.** FEUP, 2006.

SOARES, Liziane Santos. **Obtenção de requisitos para customização de processo de desenvolvimento de software: Uma abordagem utilizando dinâmica de sistemas.** UFV, 2007.

SOARES, Michel dos Santos. **Comparação entre Metodologias Ágeis e Tradicionais para o Desenvolvimento de Software.** Unipac - Universidade Presidente Antônio Carlos Faculdade de Tecnologia e Ciências de Conselheiro Lafaiete, 2004.

SOARES, Michel dos Santos. **Metodologias Ágeis Extreme Programming e Scrum para o Desenvolvimento de Software.** Unipac - Universidade Presidente Antônio Carlos Faculdade de Tecnologia e Ciências de Conselheiro Lafaiete, 2004.

SOMMERVILLE, Ian**. Engenharia de software.** 8ª ed. São Paulo: Pearson Addison-Wesley, 2007.

SOMMERVILLE, Ian**. Engenharia de software.** 9ª ed. São Paulo: Pearson Addison-Wesley, 2011.

SOUZA, Givanaldo Rocha de Souza. **Metodologias Ágeis de Desenvolvimento de Software.** IFRN, 2011.

TOLEDO, Daniel Eduardo Funabashi De. **Um processo ágil de engenharia de requisitos com apoio de padrões de software.** UFSCar, 2008.

UTIDA, Kleber Hiroki. **Metodologias tradicionais e metodologias ágeis: Analise comparativa entre Rarional Unified Process e Extreme Programming.** FATECSP. 2012.

APÊNDICE A – Questionário

Engenharia de Requisitos Tradicional e Métodos Ágeis

Este questionário se destina a uma pesquisa acadêmica para trabalho de conclusão de curso de graduação em Ciência da Computação.
O objetivo é colher informações e opiniões de profissionais de TI sobre as metodologias de gerenciamento e desenvolvimento de softwares e engenharia de requisitos para embasamento do meu trabalho acadêmico.
Os dados obtidos não serão divulgados para nenhum outro fim e seu e-mail é unicamente solicitado para identificação, não entraremos em contato nem forneceremos para terceiros.
Obrigada pela colaboração!

Identificação do participante

E-mail:

Cidade onde reside:

Estado onde reside:

Qual a sua idade?

- ( ) 15 a 25 anos
- ( ) 26 a 35 anos
- ( ) 36 a 45 anos
- ( ) 46 a 55 anos
- ( ) 56 a 60 anos
- ( ) Mais de 60 anos

Qual seu nível de formação?

- ◯ Técnico
- ◯ Graduado / Graduando
- ◯ Pós-graduado / Pós-graduando
- ◯ Mestre / Mestrando
- ◯ Doutor / Doutorando
- ◯ Outro: _____

Há quantos anos você trabalha com desenvolvimento de software?

- ◯ 0 - 5 anos
- ◯ 5 – 10 anos
- ◯ 10 - 15 anos
- ◯ Mais de 15 anos

Qual o seu papel na equipe de desenvolvimento?

- ◯ Desenvolvedor de Software
- ◯ Analista de Sistemas
- ◯ Engenheiro de Software
- ◯ Analista de Teste
- ◯ Gerente de Projetos
- ◯ Outro: _____

Diariamente, você está envolvido com a produção de softwares de que tipo?

- ☐ Software de Gerenciamento Empresarial
- ☐ Software para terminais móveis
- ☐ Software para Web
- ☐ Desenvolvimento de Sistemas Operacionais
- ☐ Softwares aplicativos de propósito geral

Engenharia de Requisitos

O que você considera ser um desafio durante a Engenharia de Requisitos?

- [ ] O que vocês considera ser um desafio durante a Engenharia de Requisitos
- [ ] Comunicação com o cliente.
- [ ] Comunicação entre a equipe.
- [ ] Grande número de reuniões.
- [ ] Classificar e priorizar os requisitos.
- [ ] Especificar os requisitos.
- [ ] Mudanças nos requisitos após iniciar o desenvolvimento do software.
- [ ] Outro: _____

Qual a importância dos seguintes fatores durante o desenvolvimento de um software.

	Não é importante	Pouco importante	Importante	Muito Importante	Não sei
Qualidade do produto	◯	◯	◯	◯	◯
Satisfação do cliente	◯	◯	◯	◯	◯
Resultados de vendas	◯	◯	◯	◯	◯

Classifique o quão importante é para você a participação do cliente durante o desenvolvimento do software.

	0	1	2	3	4	5	
Pouco importante	◯	◯	◯	◯	◯	◯	Muito importante

Quais técnicas de requisitos são utilizadas nos projetos de desenvolvimento de software na sua empresa?

	Conheço e é usada	Conheço mas não é usada	Não conheço
Entrevista	◯	◯	◯
Brainstorming (Tempestade cerebral)	◯	◯	◯
Casos de Uso	◯	◯	◯
Cenários	◯	◯	◯
Etnografia	◯	◯	◯
Mapa Mental	◯	◯	◯

Como você costuma apresentar os requisitos coletados para que os mesmos sejam validados pelo cliente?

- ☐ Diagramas de Casos de Uso
- ☐ User Stories (Histórias de usuário)
- ☐ Diagrama de Fluxo de Dados (DFD)
- ☐ Fluxogramas
- ☐ DER (Diagrama de Entidade e Relacionamento)
- ☐ Outros:

Metodologias Tradicionais

Há quanto tempo trabalha ou trabalhou utilizando metodologias tradicionais?

- ◯ 0 - 5 anos
- ◯ 5 – 10 anos
- ◯ 10 - 15 anos
- ◯ Nunca trabalhei com metodologias tradicionais

Quais metodologias tradicionais são utilizadas nos projetos de desenvolvimento de software na sua empresa?

	Conheço e é usada	Conheço mas não é usada	Não conheço
Modelo Cascata	◯	◯	◯
Modelo Incremental	◯	◯	◯
Modelo de Prototipagem	◯	◯	◯
Modelo Espiral	◯	◯	◯
Modelo de desenvolvimento concorrente	◯	◯	◯
Engenharia de Software orientada a reúso	◯	◯	◯
RUP (Rational Unified Process)	◯	◯	◯

Mencione outras metodologias tradicionais que são utilizadas na sua empresa e não foram citadas na questão anterior.

_____

Na sua opinião, quais são as vantagens das metodologias tradicionais?

- ☐ Possui um planejamento mais rígido
- ☐ Um maior foco em processos do que no produto esperado
- ☐ Envolvimento do cliente apenas nos estágios iniciais
- ☐ Maior controle de custos e lucro
- ☐ Menor número de reuniões
- ☐ Cliente se sente mais seguro
- ☐ Equipes maiores
- ☐ Não há necessidade de manter um ritmo constante de trabalho
- ☐ Softwares melhor elaborados
- ☐ Outro: _____

Na sua opinião quais são as desvantagens das metodologias tradicionais?

- [ ] Atenção maior na geração de documentos
- [ ] Resistência a mudanças
- [ ] Parte da equipe não participa das reuniões nos estágios iniciais
- [ ] Não são recomendadas para projetos curtos
- [ ] Equipe não é motivada frequentemente
- [ ] Prazos estourados
- [ ] Software com excesso de funcionalidades
- [ ] Outro:_____

Metodologias Ágeis

Há quanto tempo trabalha ou trabalhou utilizando metodologias ágeis?

- ◯ 0 - 5 anos
- ◯ 5 – 10 anos
- ◯ 10 - 15 anos
- ◯ Nunca trabalhei com metodologias tradicionais

Quais metodologias ágeis são utilizadas nos projetos de desenvolvimento de software na sua empresa?

	Conheço e é usada	Conheço mas não é usada	Não conheço
XP (Extreme Programming)	◯	◯	◯
Scrum	◯	◯	◯
Crystal	◯	◯	◯
Feature Driven Development (FDD)	◯	◯	◯

Mencione outras metodologias ágeis que são utilizadas na sua empresa e não foram citadas na questão anterior.

_____

Classifique a importância de uma documentação mais formal durante a utilização de alguma metodologia ágil.

	0	1	2	3	4	5	
Pouco importante	◯	◯	◯	◯	◯	◯	Muito importante

Quais práticas ágeis sobre requisitos são utilizadas na sua empresa?

	Conheço e é usada	Conheço mas não é usada	Não conheço
Product Vision (Visão do produto)	◯	◯	◯
Product Backlog (Registros de melhorias do produto)	◯	◯	◯
User Stories (Histórias de Usuário)	◯	◯	◯
Casos de Uso	◯	◯	◯
Priorização de requisitos	◯	◯	◯
Uso de Cenários	◯	◯	◯

Na sua opinião, quais são as vantagens das metodologias ágeis?

- ☐ Envolve mais o cliente
- ☐ Transparência e visibialidade do status do projeto
- ☐ Rápida solução de problemas
- ☐ Entregas de funcionalidades frequentemente
- ☐ Prioriza a satisfação do cliente
- ☐ Melhoria da Qualidade do produto final
- ☐ Equipes auto-gerenciáveis
- ☐ Melhoria na comunicação entre gerente, equipe de desenvolvimento e cliente
- ☐ Antecipação dos problemas e maior agilidade na tomada de ações
- ☐ Flexibilidade para mudanças de requisitos e prioridades
- ☐ Prazos cumpridos

Na sua opinião, quais são as desvantagens das metodologias ágeis?

- ☐ Não são recomendadas para projetos longos
- ☐ Reuniões diárias
- ☐ Menor controle de custos e lucro
- ☐ Insegurança por parte do cliente
- ☐ Mudanças de requisitos após o início do desenvolvimento do software
- ☐ Apresentar para o cliente pelo menos uma prévia do software funcionando
- ☐ Software simples
- ☐ Outros: _____

Classifique a importância dos valores do Manifesto Ágil citados abaixo:

	Muito importante	Importante	Pouco importante	Não é importante	Não sei
"Indivíduos e interações são mais importantes que processos e ferramentas"	◯	◯	◯	◯	◯
"Software funcionando é mais importante do que documentação completa e detalhada"	◯	◯	◯	◯	◯
"Colaboração com o cliente é mais importante do que negociação de contratos"	◯	◯	◯	◯	◯
"Adaptação a mudanças é mais importante do que seguir o plano inicial"	◯	◯	◯	◯	◯

Projetos desenvolvidos utilizando métodos ágeis requerem um ritmo constante de trabalho durante todas as fases. Seu ambiente de trabalho oferece motivação para que seu rendimento seja bom?

- ◯ Sim
- ◯ Não
- ◯ Às vezes

De acordo com um dos princípios do Manifesto Ágil a melhor maneira de compartilhar informações para e entre uma equipe de desenvolvimento é através de conversa aberta, de forma presencial. Em sua empresa, as informações são transmitidas dessa forma?

- ◯ Sim
- ◯ Não
- ◯ Às vezes

Sua equipe se reúne regularmente para pensar em maneira de se tornarem mais eficazes e ajustarem o comportamento de acordo com aquilo que foi pensado?

- ◯ Sim
- ◯ Não
- ◯ Às vezes

Os prazos estabelecidos no início dos projetos são cumpridos?

- ◯ Sim
- ◯ Não
- ◯ Às vezes

Relacionamento com o Cliente

Como você classifica o grau de satisfação do cliente em relação ao número de reuniões feitas ao longo do projeto desenvolvido com metodologias ágeis?

	0	1	2	3	4	5	
Insatisfeito	◯	◯	◯	◯	◯	◯	Muito satisfeito

Com que frequência ocorre mudanças de requisitos por parte do cliente?

	0	1	2	3	4	5	
Baixa	◯	◯	◯	◯	◯	◯	Alta

As exigências feitas pelo cliente são sempre acatadas pela equipe?

◯ Sim

◯ Não

◯ Às vezes

O cliente demonstra mais confiança no projeto acompanhando todos os passos de desenvolvimento do software?

◯ Sim

◯ Não

◯ Às vezes

O cliente aprova as prévias do sistema funcionando que são apresentadas em curtos intervalos de tempo?

◯ Sim

◯ Não

◯ Às vezes

Marque qual/quais motivo(s) levam os clientes a reprovar as prévias do sistema.

☐ Sistema não funciona perfeitamente

☐ Sistema não atendeu aos requisitos estabelecidos

☐ Sistema não atendeu as expectativas

☐ Outro: _____

Como você classifica o grau de satisfação do cliente em relação à entrega do produto final?